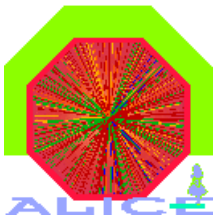


Central Trigger Processor

PRELIMINARY DESIGN REVIEW

Revision 0.2 29 June 2004

DOCUMENT HISTORY

Version	Date	Editor	Comments
Draft 0.1	24.06.04	P. Jovanovic	First draft.
Draft 0.2	29.06.04	P.Jovanovic	The *SYNC* board replaced with the *INT* board and moved to the slot 11; the layout and the backplane changed accordingly (**Figures 3.22.1**, **3.22.2** and **3.23.1**). Version prepared for the *CTP Preliminary Design Review.*

TABLE OF CONTENTS

1.0 INTRODUCTION

1.1 Preface

This document is prepared for the *ALICE CTP Preliminary Design Review* at the time when the *final* LTU production is in progress and the project's TDR [1] had already been officially approved. *Preliminary* is, therefore, misleading (and anathema to the Technical Coordinator). But it connotes our compliance with the ISO9001 structured quality management scheme, a compliance neither complete nor formal, but in spirit, terminology and procedures, a compliance that produced results in our previous projects and, so far, has worked well for the ALICE trigger.

A meagre attempt has been made to set the project background by including some rather general sections, copied from other documents. Only *some* - because the "full story" would have exceeded a manageable size. The main body of the document is an edited version of the *CTP working notes* that have been scrupulously updated throughout the progress of the project, in order to document the development effort, to explain the ideas, the decisions, the reasons for and against, to serve as a reminder.

No background, many details - it's going to be heavy reading, especially for "newcomers" and also, and regrettably so, for the referees. Help might come from numerous references, most of them available on the Web. If everything fails - our humble apologies...

For us, working on the project, the document shall serve as a comprehensive *high-level design*, the first step of the hierarchical top-down procedure we are going to follow. The same approach worked rather well for the LTU. Our hopes are equally high this time.

A very brief résumé, entitled ***At a glance***, has been added for impatient readers...

1.2 Purpose and scope of the document

This is a *working document*, currently in a draft form. Comments and suggestions from the ALICE community are invited and encouraged. The document shall be regularly updated, with changes explained in section ***Document History***.

The document gives a preliminary technical specification of the Central Trigger Processor (CTP), based on the requirements listed in the CTP URD [2] and in a number of proposals [3 to 11], all approved by the ALICE Technical Board. The document does not contain the design details since the design itself is not yet done, but it studies the feasibility of the project and provides the guidelines for the hardware implementation.

Since the project follows recommendations and guidelines of the ISO9001 structured quality management procedure [13], the preliminary specification shall be assessed during the *Preliminary Design Review* in order to check that the expected performance and functionality are met. Later on, a detailed design of each board shall be followed by the *Final Design Review*, before the board is produced and tested.

1.3 Document overview

The present section (***Introduction***) gives a brief account of the purpose and scope of the document, explains its structure, defines terms and acronyms and lists cited documents. The following section (***Technical Specification***) describes the CTP's connections to the external systems, partitions the CTP into functional units, and describes in detail their operation.

1.4 Definitions and acronyms

ADC Analogue to digital converter.

BC Bunch Crossing (clock) - the 40.08 MHz clock, locked to the LHC machine cycle, used to synchronise the pipeline processing system.

BGA Ball Grid Array (IC package).

BUSY Signal generated by a sub-detector to indicate that it cannot accept another **L0** trigger.

CTP Central Trigger Processor [2] - electronic system that receives inputs from ALICE trigger sub-detectors and generates, in each bunch crossing, **L0**, **L1** and **L2** yes/no trigger decisions for all sub-detectors.

DAQ (ALICE) Data Acquisition system.

DCS	(ALICE) Detector Control System.
DPM	Dual port memory.
ECS	(ALICE) Experiment Control System.
HLT	(ALICE) High Level Trigger system.
L0	Level-0 trigger (signal).
L1	Level-1 trigger (signal).
LED	Light Emitting Diode (indicator).
LTC	Local Trigger Crate - a VME crate that contains the local system processor, the LTU, the TTC boards *etc*. .
LTU	Local Trigger Unit (board).
LVDS	*Low Voltage Differential Signalling* - a standard differential signal format.
NRZ	Non Return to Zero (signal format).
PLL	Phase locked loop.
RoI	Region of Interest (option, logic, data).
RoII	RoI Interface (board).
RoIP	RoI Processor (system).
TB	ALICE Technical Board.
TTC	Timing, Trigger and Control (system) [14][15].
TTCcf	TTC Clock Fanout (board) [16].
TTCex	TTC Encoder/Transmitter (board) [18].
TTCit	TTC Interface Test (board).
TTCmi	TTC Machine Interface (system, crate) [16].
TTCrx	Timing, Trigger and Control Receiver ASIC [17].
TTCtx	TTC Transmitter (board) [18].
TTCvi	TTC -VMEbus Interface (board) [19].
U	Unit of height in the standard 19" rack (1U = 1¾").
URD	(CTP) User Requirement Document.

1.5 Signal name abbreviations

BCID[12..1]	***BC** identifier* word, part of the event identifier.
ClT	*Calibration Trigger* flag.
ESR	*Enable Segmented Readout* flag, part of the RoI option.
L1Class[50..1]	*Class [50..1] L1 trigger* status flag.
L1SwC	*Software Class L1 trigger* status.
L2arF	*L2 accept/reject* flag.
L2Class[50..1]	*Class [50..1] L2 trigger* status flag.
L2Cluster[6..1]	*Cluster [6..1] L2 trigger* status flag.
L2Detector[24..1]	*Detector[24..1] L2* readout status flag.
L2SwC	*Software Class L2 trigger* status.
OrbitID[24..1]	***Orbit** identifier* word, part of the event identifier.
RoC[4..1]	*Readout Control [4..1]* bits, (*Software Trigger* only).

1.6 References

[1] *ALICE Technical Design Report on Trigger, Data Asquisition, High Level Trigger and Control System*, CERN-LHCC-2003-062, 7 January 2004.

[2] *ALICE Central Trigger Processor: User Requirement Document*, current version available on the ALICE CTP web site [10].

[3] *Layout and Connections of the ALICE Trigger System*, proposal approved by the ALICE TB on 15 July 2002; available on the ALICE CTP web site [12].

[4] *The CTP Readout and the Interaction Record Data Format*, proposal approved by the ALICE TB in March 2001; available on the ALICE CTP web site [12].

[5] *Region Of Interest Interface*, proposal approved by the ALICE TB on 14 May 2002; available on the ALICE CTP web site [12].

[6] *Content and Format of the L2a Message*, proposal approved by the ALICE TB on 14 May 2002; available on the ALICE CTP web site [12].

[7] *Proposal to Replace the L1 Trigger-type Word with the L1 Message*, proposal approved by the ALICE TB on 20 March 2002; available on the ALICE CTP web site [12].

[8] *Proposal for the Use of the "Explicit NO" in Trigger Conditions*, proposal approved by the ALICE TB on 26 November 2002; available on the ALICE CTP web site [12].

[9] *Proposal for Past-future Protection Handling in ALICE*, proposal approved by the ALICE TB on 26 November 2002; available on the ALICE CTP web site [12].

[10] *Implementation of the Rare-event Handling Logic*, proposal approved by the ALICE TB on 16 December 2002; available on the ALICE CTP web site [12].

[11] *Cross-section Measurement in Heavy Ion Collisions at ALICE*, proposal approved by the ALICE TB on 16 December 2002; available on the ALICE CTP web site [12].

[12] ALICE CTP web site: (ALICE→Projects→Trigger), or, directly, http://www.ep.ph.bham.ac.uk/user/pedja/alice/.

[13] *ISO 9001, Quality Systems*, published by BSI and other national standards bodies.

[14] TTC web site: http://www.cern.ch/TTC/intro.html.

[15] B.G. Taylor, *TTC Distribution for LHC Detectors*, IEEE Trans. Nuclear Science, Vol. 45, June 1998, pp. 821-828.

B.G. Taylor, *LHC Machine Timing Distribution for the Experiments*, Proceedings of the Sixth Workshop on Electronics for LHC Experiments, Crakow, Poland, 11-15 September 2000, pp.312-317.

[16] B.G. Taylor, *TTC Machine Interface (TTCmi) User Manual*, current version available on the TTC web site [14].

[17] J. Christiansen et al., *TTCrx Reference Manual*, current version available on the TTC web site [14].

[18] B.G. Taylor, *TTC Laser Transmitter (TTCex, TTCtx, TTCmx) User Manual*, current version available on the TTC web site [14].

[19] Ph. Farthouat et al., *TTC-VMEbus Interface (TTCvi-MkII)*, current version available on the TTC web site [4].

[20] *LTU Preliminary Design Review* document, approved by the ALICE TB on 22 October 2002; available on the ALICE CTP web site [12].

[21] *A proposal for the System Timing*, proposal approved by the ALICE TB on 22 October 2002; available on the ALICE CTP web site [12].

[22] R. Spiwoks, *Deadtime BEFORE/AFTER Prescaling in the Level-1 CTP*, ATLAS Internal Note ATL-DA-EP-0003, 7 March 2001.

[23] Orlando Villalobos-Baillie, private communication.

2.0 *AT A GLANCE...*

- The ALICE trigger system, situated in the experimental cavern, has a *centralised* layout (**Figure 3.2**): the Central Trigger Processor (CTP), the sub-detector interface (Local Trigger Unit) and the TTC partitions are all installed in adjacent racks.

- The ALICE CTP generates *three levels of hardware triggers* – ***L0***, ***L1*** and ***L2***, before an event is declared suitable for further software assessments (HLT) and transmitted to the ALICE DAQ.

- The 24 sub-detectors of the ALICE experiment are *dynamically partitioned* into up to *6 independent clusters*, with an additional, software-triggered *test cluster*, configured on demand "on the fly"; the cluster configuration is fully arbitrary – clusters could be exclusive, but are more likely to overlap.

- The level of event pile-up is controlled by the *Past-future Protection* – a procedure that selects only the events with either no pile-up at all in a programmable time interval before and after the interaction, or with a number of pile-up interactions up to a programmable limit. The *Past-future Protection* operates independently for each cluster and is performed at all three trigger levels.

- The ALICE CTP generates heavy data traffic over the TTC system. The *Channel A* is used for transmission of the time-critical **L1** signal, in a way similar to most other LHC experiments, but, for each trigger sequence that is confirmed at the ***L1*** level, the following data are also transmitted *via* the TTC's *Channel B*: the *L1 Message* - 5 16-bit words; the *RoI Message* - 4 words; the *L2a Message* - 8 words. Simultaneously, similarly to the other systems, the *Channel B* is also used for transmission of the LHC **Orbit** and of the calibration **Pre-pulse** signal.

- The ALICE CTP receives and generates the following signals:

    - 2 LHC timing signals (**BC** and **Orbit**),
    - 50 trigger inputs (24 **L0**; 20 **L1**; 6 **L2**),
    - 24 sub-detector **BUSY** inputs,
    - 24 independent sets of 7 sub-detector outputs (168 signals in total),
    - a number of signal connections to the RoIP, the DCS, *etc*. .

- The ALICE CTP shall be partitioned in functional blocks located on separate boards:
    - *L0 processor,*
    - *L1 processor,*
    - *L2 processor,*
    - *BUSY processor,*
    - *Fan-out unit,*
    - *Interface board.*

- The boards shall have a *6U form-factor*, 6 to 8 PCB layers, with a moderate track and component density.

- The preliminary specification of the CTP shall be assessed during the *Preliminary Design Review* in order to check that the expected performance and functionality are fully met.

- *"Physicists are romantic"* and *"system designers are grumpy and grim"* - read more in section **3.1.2**...

3.0 TECHNICAL SPECIFICATION

3.1 Introduction

3.1.1 *Background*

The technical requirements for the ALICE Central Trigger Processor [2] have been thoroughly discussed, reviewed, assessed and reassessed over many years. Some issues have been clarified to minute details in the proposals officially approved by the ALICE Technical Board [3 to 11]. The emerging "vision" of the ALICE Central Trigger Processor (CTP) has been recently presented in the Technical Design Report [1]. This document attempts to turn that vision into reality and deals with the details of the CTP design and implementation.

Development and testing of the Local Trigger Unit (*LTU*) [20], a versatile interface between the ALICE CTP and the sub-detector readout electronics, has now been completed and the production of the final 53 boards is in progress. The essential feature of the *LTU* is that the board, when in *Stand-alone* mode, fully emulates the operation of the CTP. The main challenge of the *LTU* design was to establish and implement an *accurate model* of the CTP operation: for a couple of years, the sub-detectors will be developing their front-end electronics with the *LTU* and, when the final system eventually "arrives", they will expect the CTP to *"fully emulate the LTU"*. The existence of such a model is a firm basis and a significant help in the current CTP design.

3.1.2 *General design approach*

The following factors strongly affect the CTP layout, the CTP partitioning into functional units and the general design approach:

Complexity of the logic requirements

Arguably, the ALICE Central Trigger Processor is the most complex among the CTPs of the LHC experiments:

- The ALICE CTP generates *three levels of hardware triggers – **L0**, **L1*** and ***L2***, before an event is declared suitable for further software assessments (HLT) and transmitted to the ALICE DAQ.

- The 24 sub-detectors of the ALICE experiment are *dynamically partitioned* into up to *6 independent clusters*, with an additional, software triggered *test cluster*, configured on demand "on the fly"; the cluster configuration is fully arbitrary – clusters could be exclusive, but are more likely to overlap.

- Due to the complexity of ALICE events, a success or otherwise of the pattern recognition task is strongly dependant upon the level of event pile-up. The level is controlled by the *Past-future Protection* – a procedure that selects only the events with either no pile-up at all in a programmable time interval before and after the interaction, or with a number of pile-up interactions up to a programmable limit. The *Past-future Protection* operates independently for each cluster and is performed at all three trigger levels.

- Among the LHC experiments, the ALICE CTP generates the *highest data traffic over the TTC system*. The *Channel A* is used for transmission of the time-critical **L1** signal, in a way similar to most other experiments, but, for each trigger sequence that is confirmed at the ***L1*** level, the following data are also transmitted *via* the TTC's *Channel B*: the *L1 Message* - 5 16-bit words; the *RoI Message* - 4 words; the *L2a Message* - 8 words. Simultaneously, similarly to the other systems, the *Channel B* is also used for transmission of the LHC **Orbit** and of the calibration **Pre-pulse** signal.

- *Etc.* .

The case for those complex requirement is made elsewhere [1].

Complexity of the system connections

The ALICE CTP receives and generates the following signals:

- 2 LHC timing signals (**BC** and **Orbit**),
- 50 trigger inputs (24 **L0**; 20 **L1**; 6 **L2**),
- 24 sub-detector **BUSY** inputs,
- 24 *independent sets* of 7 sub-detector outputs (168 signals in total),
- a number of signal connections to the RoIP, the DCS, *etc.* .

The total number of connections and their nature strongly affect the design approach.

Flexibility for modifications and upgrades

Like all the other LHC experiments, ALICE should expect a long "working life", during which modifications and upgrades are almost a certainty and the CTP is probably the most likely sub-system to face such a request. The use of FPGAs, with generous amount of logic and memory left for future applications, should provide for the majority of modifications, but, for more extensive changes, a need to re-develop and re-build the electronics hardware could not be excluded. The ease, or otherwise, of such a task strongly depends upon the adopted design approach, and the *modularity of the system* becomes the key issue.

System modularity

The clear advantages of a modular system are, on the other hand, strongly offset by the need to accommodate and handle a mushrooming number of inter-module connections. The difficulties are particularly severe in the LHC environment, with a relatively high clock frequency and very tight timing requirements. As a result, and out of necessity, all the LHC CTP systems have opted for a single module (or nearly so) approach, with the majority of their functions performed on a single, dense, highly multi-layered, 9U form-factor board.

For quite some time, the ALICE CTP design was also heading in the same direction: with the requirement for 50 trigger classes and 24 independent sub-detectors, with a need to propagate the status of the classes and the sub-detectors through all three trigger decision levels, the resulting width of the processing pipeline was well above a manageable size that would permit splitting the system into a succession of functional blocks located on different boards.

Eventually, a sensible and elegant compromise between the unbridled romanticism of physicists (physicists *are* romantic - *they want everything*...) and a grumpy, grim realism of system designers, offered the possibility to "modularise" the ALICE CTP. The key decisions were:

- the "permission" to have up to 2μs of *dead time* between the consecutive **L0** triggers (out of which less than 1.5μs will be used in the final design); the dead time allows for the serialization of data transferred between the processors operating at different trigger levels, which, in turn, significantly reduces the number of physical connections;

- the programmable "grouping" of sub-detectors in *clusters* and limiting their number at any time to 6; dealing with 6 clusters instead of 24 independent sub-detectors brought in another significant decrease of the number of physical connections.

The performance of the ALICE trigger has not been affected by those decisions [1], but the benefits to the overall system design became significant. As a result:

- The ALICE CTP shall be partitioned in functional blocks located on separate boards:
    - *L0 processor,*
    - *L1 processor,*
    - *L2 processor,*
    - *BUSY processor,*
    - *Fan-out unit*

- The boards shall have a *6U form-factor*, 6 to 8 PCB layers, with a moderate track and component density.

3.2 Layout of the ALICE trigger system

This section has been copied from [20]; it is included for the reason of completeness.

The *centralised* layout of the ALICE trigger system [3], shown in **Figure 3.2**, has been approved by the ALICE Technical Board. The hardware shall be situated in the experimental cavern. All the sub-detector TTC partitions shall be *centrally* located, installed in a number of VME crates, in racks adjacent to the rack with the CTP and the ALICE TTC Machine Interface (TTCmi). (The RoI fan-out electronics, or the entire RoI system, could also be mounted in the rack.)

Each VME crate (21 slots) shall accommodate at least 4 TTC partitions (3 boards per partition; 4, if the RoI interface is also required); each crate shall have its own processor. For the maximum number of 24 sub-detectors, the total of 6 VME crates shall be required. The system is scalable - the crates will be added as the number of ALICE sub-detectors increases.

The electrical connections between the CTP and the sub-detector TTC partitions are represented with arrows in **Figure 3.2**; more details are given in section **3.6.1**. The length of the connections is reduced to a practical minimum. It improves the system reliability, diminishes the transmission error rate and eliminates a potential source of system noise.

The **BC** clock distribution network is also reduced to a minimum, which enhances clock stability and eliminates another potential source of system noise.

3.3 Context diagram of the CTP

This section has been copied from [2]; it is included for the reason of completeness.

The interfaces between the CTP and the external systems are shown in the context diagram (**Figure 3.3**).

The *trigger inputs* related to the same bunch crossing, but generated by different trigger sub-detectors, will, generally, arrive at the CTP at different times. The CTP, therefore, has to include programmable elements that delay the signals that arrive early to be aligned in time with those that arrive last.

Calibration requests enable the sub-detectors to “order” calibration triggers during the physics run (for stand-alone tests, the sub-detectors can introduce their own triggers at the level of their local TTC system). The CTP also provides a *pre-pulse* facility used to generate calibration inputs prior to calibration triggers.

A link to the Experiment Control System (ECS) is used to configure and run the CTP. It also carries error and warning massages which emerge from real time analysis of data coming from scalars and diagnostic memories.

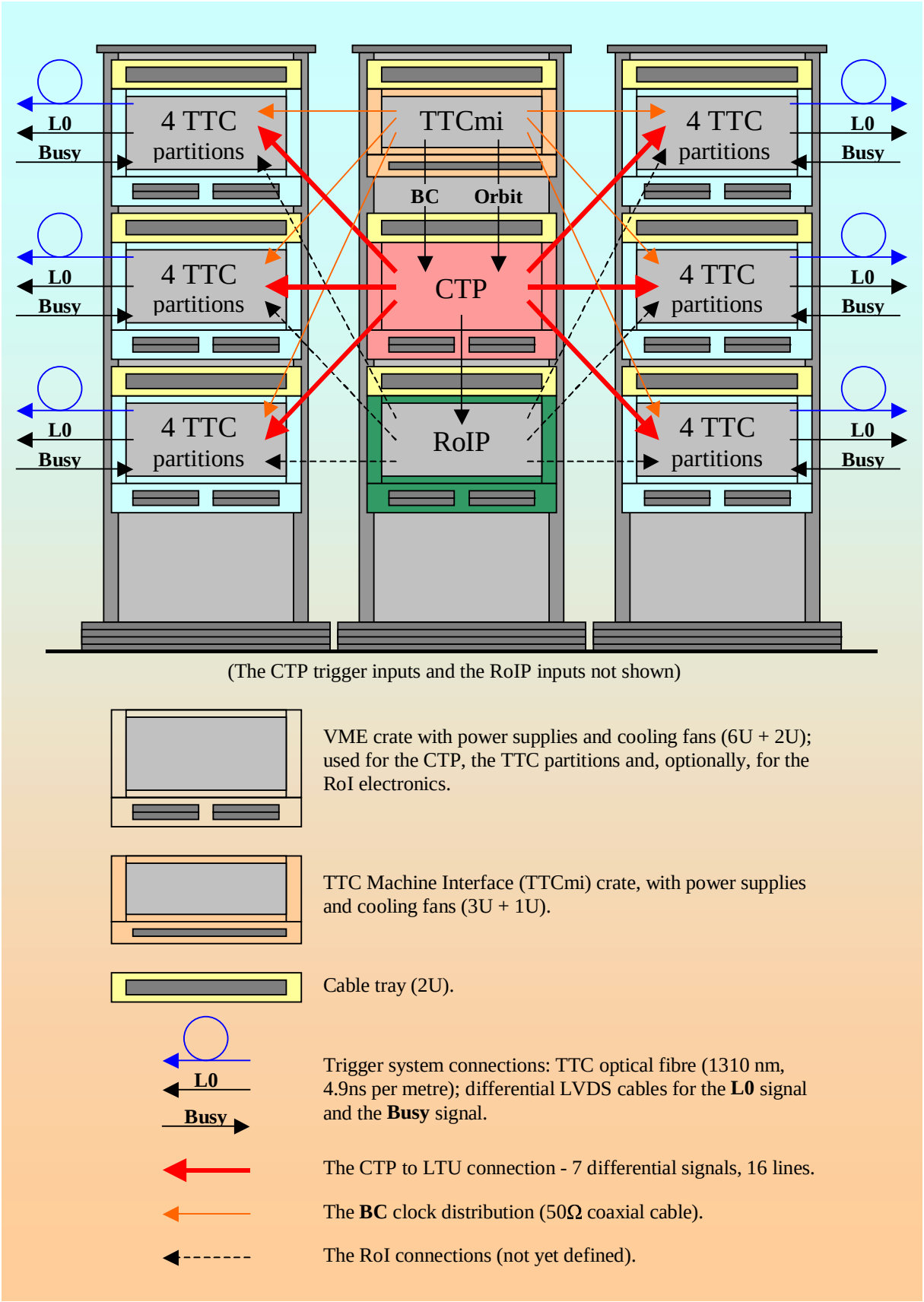


Figure 3.2 Layout and connections of the ALICE trigger system

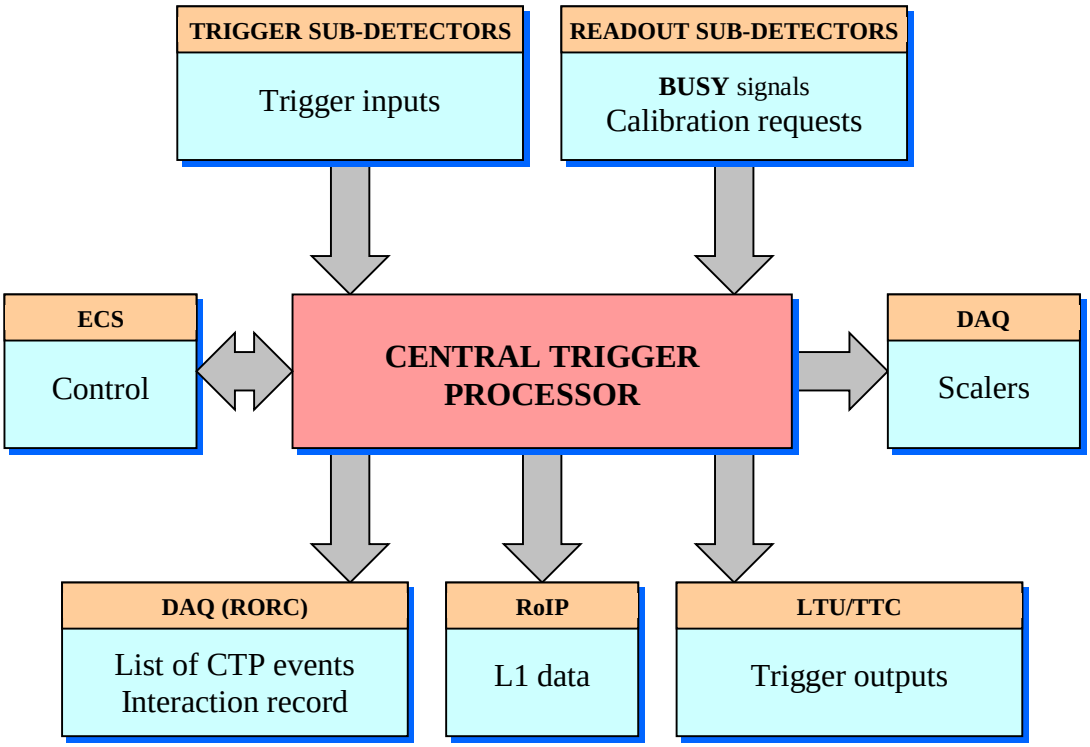


Figure 3.3 Context diagram of the CTP

On top of the information sent to the DAQ for all the **L2**-selected events, the CTP also generates the *Interaction Record* - a list of all the bunch-crossings in which the **Interaction** signal has been detected.

Apart from the **L0** trigger, all the *trigger outputs* are distributed to the front-end electronics of sub-detectors *via* the optical link of the TTC system; the **L0** trigger uses a dedicated, low propagation-delay cable.

For each **L1** trigger, the RoI Processor receives from the CTP the *L1 trigger data*, required to initiate and control the *segmented readout* of the experiment.

3.4 Local Trigger Unit - the CTP interface to sub-detectors

This section has been copied from [20]; it is included for the reason of completeness.

The Local Trigger Unit (*LTU*) serves as an interface between the CTP and the sub-detector readout electronics. The existence of a uniform interface throughout the experiment greatly simplifies configuration and run-control tasks and makes system modifications easier to develop and implement.

In the *stand-alone mode* of operation, the LTU *fully emulates the CTP protocol* and enables sub-detectors to carry out development, test and calibration tasks independently of the CTP, at remote sites, or at times when

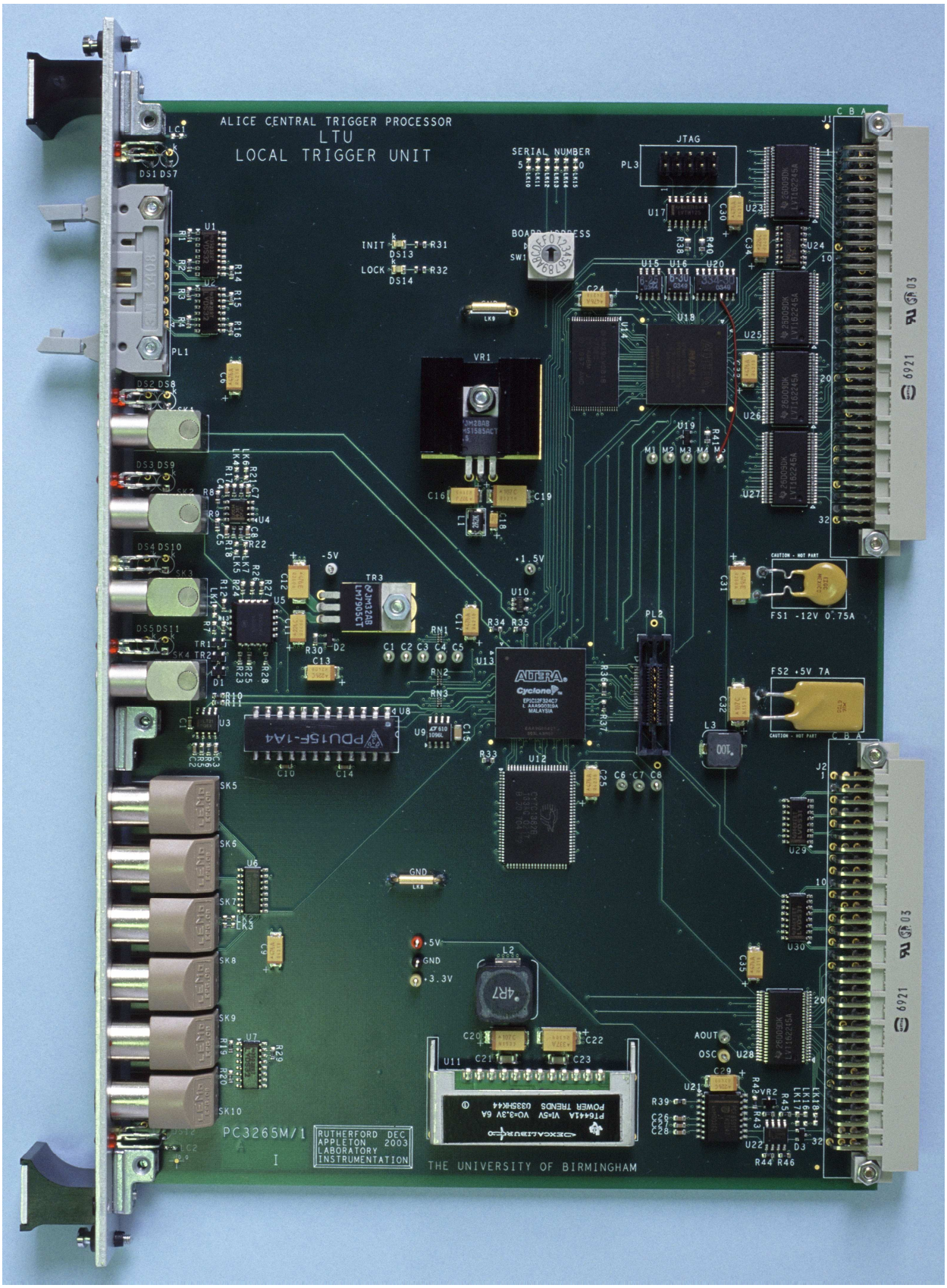


Figure 3.4.1 Local Trigger Unit (*LTU*) board

the CTP is either not available or not required. The timing of the emulated trigger sequences is identical to the timing during the *global run*.

The design and development of the *LTU* board have been completed and the production is in progress. A detailed description of the *LTU* is presented in [20]; the board photograph is shown in **Figure 3.4.1**. Pride apart, the photograph illustrates the expected complexity (or the lack of it) of a typical CTP board.

The diagram in **Figure 3.4.2** shows the connections between the CTP and the *LTU*, and the *LTU* as a part of the sub-detector *TTC partition*. The two other boards, the TTCvi and the TTCex, have been developed by the RD12 collaboration; their description and documentation are available on the TTC Web site [14]. Sub-detectors that participate in the Region of Interest option require an additional board - the *RoI Interface* [5].

The CTP connections to the *LTU* are explained in detail in section **3.6.1**.

The sub-detector readout electronics is located in the experimental cavern, 20 to 50 metres away from the CTP rack. The only electrical connections are the **L0** signal (cables with a low propagation delay required in critical cases) and the **BUSY** input. All the other CTP signals and data are transmitted over the TTC optical fibre (50/125μm graded index multimode fibre, 1310 nm, propagation delay 4.9 ns/m).

Each of the 10 identical optical outputs of the TTCex board can be locally fanned-out by a passive *1:32 optical tree coupler* [18] to a total of 320 possible destinations. The number could be further increased by the addition of a TTCtx board to the sub-detector TTC partition.

At the sub-detector front-end electronics, the optical outputs are “de-multiplexed” by the TTCrx ASIC [17]. The circuit also “recovers” the **BC** clock, with the jitter estimated at around 80ps rms. The sub-detectors that need a lower jitter could be provided with either a dedicated electrical output (jitter of only 7ps rms, but a long cable could increase it significantly); or a dedicated optical signal with a similar, low jitter, in which case a local opto-electrical converter would be required.

The TTCit board is foreseen as an optional debugging and monitoring tool. It could be added, temporarily or permanently, to the sub-detector TTC partition. It could also be installed separately, in *Personnel Accessible Areas* for example, for monitoring of the TTC operation during the physics run. The board requires only a single TTC optical channel; the dedicated **L0** input is optional.

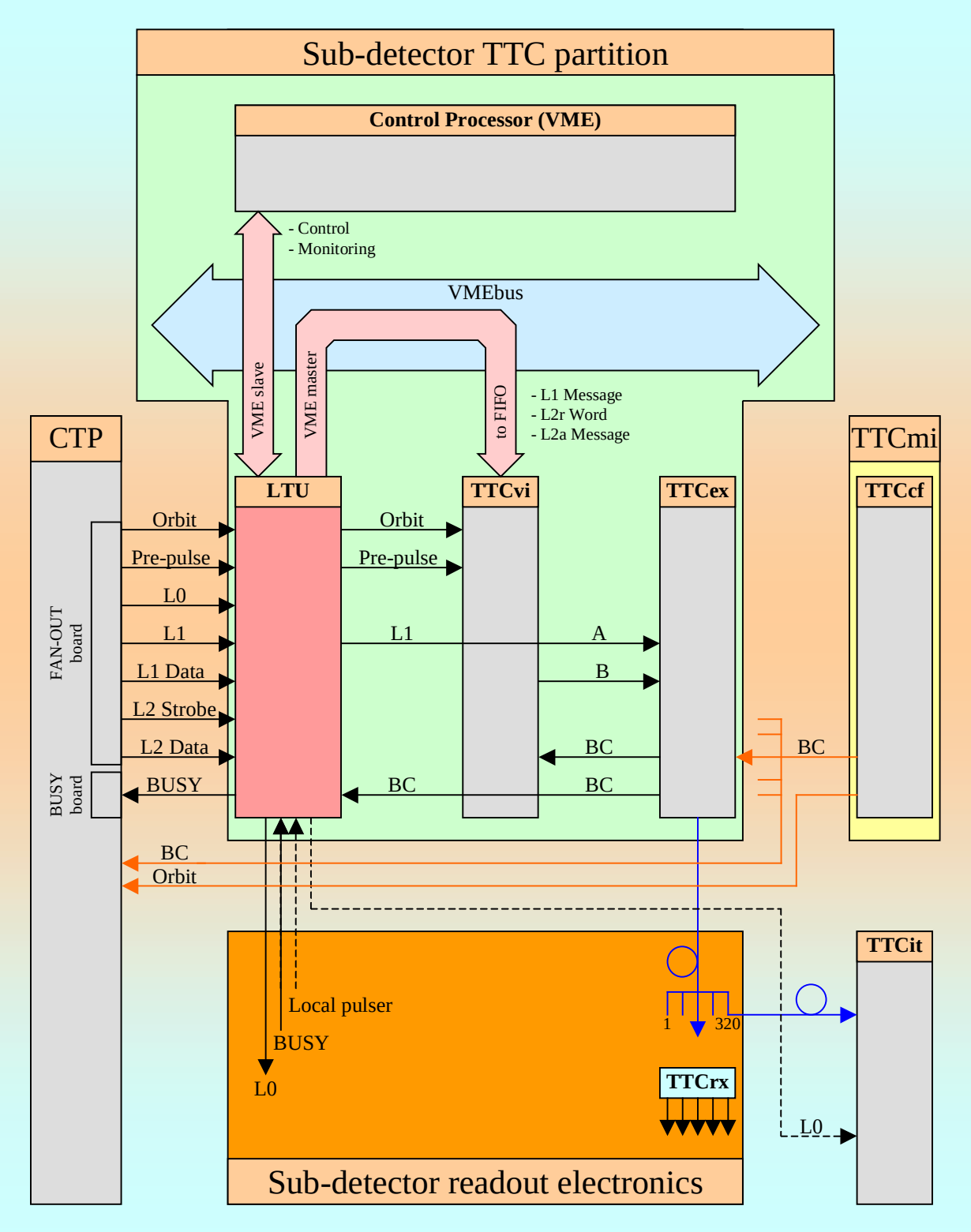


Figure 3.4.2 Context diagram of the *LTU* board

3.5 Block diagram of the CTP

The block diagram of the CTP is shown in **Figure 3.5**. Later in the text, it will be explained that the logic blocks shall be implemented on separate boards:

- *BUSY Processor* (*BUSY*),
- *L0 Processor* (*L0*),
- *L1 Processor* (*L1*),
- *L2 Processor* (*L2*),
- *Fan-out board* (*FO*).
- *Interface board* (*INT*),

For practical reasons, explained later, the interface to the ALICE DAQ (the formatting of the *CPT Readout* and the *Interaction Record*; the connection to the DDL) and the interface to the RoIP will be located on the *INT* board.

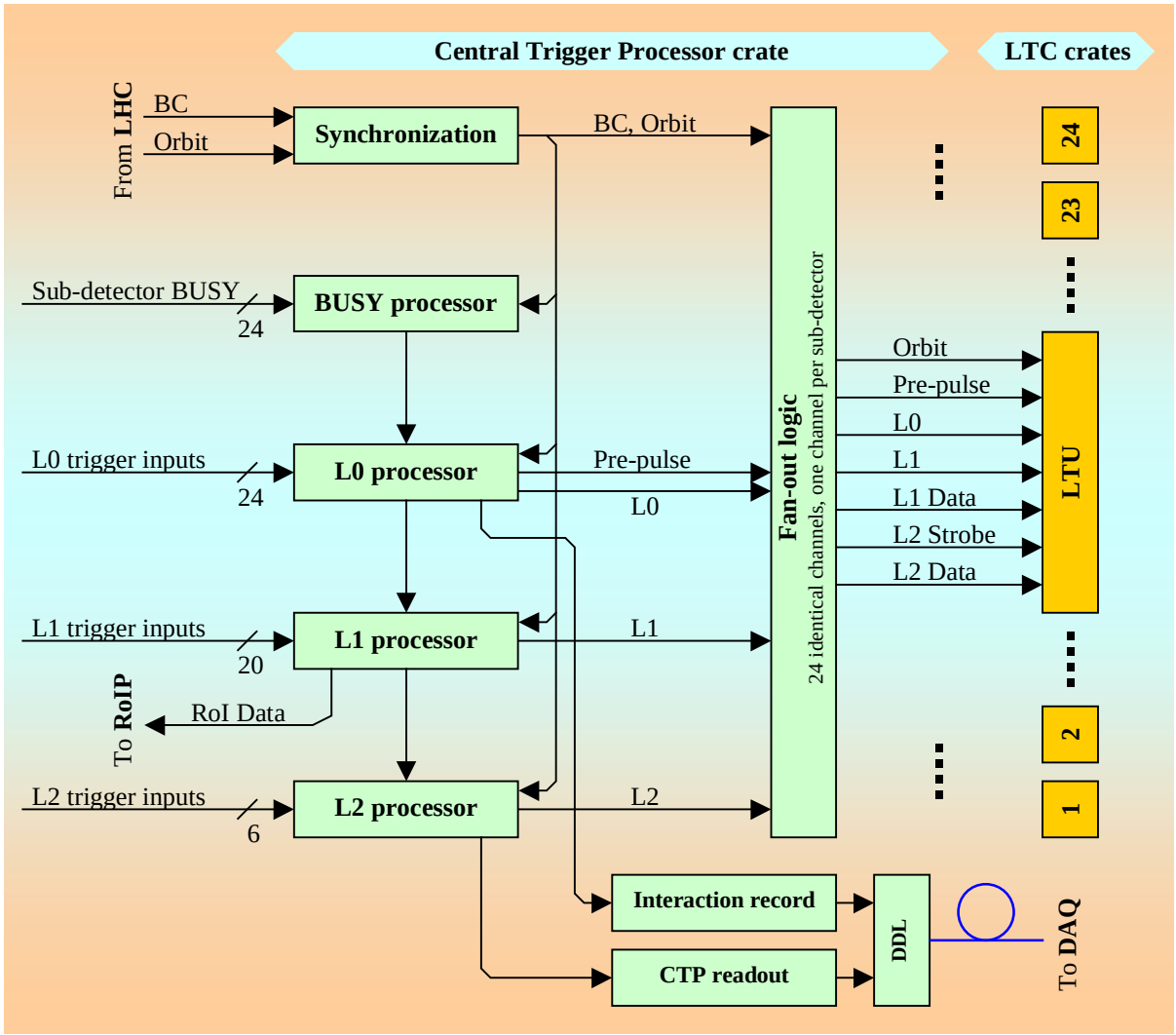


Figure 3.5 Block diagram of the CTP

3.6 CTP connections

Unless stated otherwise, here and in the rest of the document, ”input” and “output” notations are referred to the CTP.

3.6.1 *Connections to the LTU*

CTP outputs

The electrical connections between the *FO* and the *LTU* boards are shown in the context diagram (**Figure 3.4.2)**. All the signals shall have the LVDS format (differential, low noise, low power); the signal polarity shall be *inverted* since the LVDS receiver with unconnected inputs holds its output high, indicating correctly the absence of the signal. The format, the signal polarity, the choice of the connector and the connector pin allocations are fully compatible with the already available *LTU* board.

The list of connections and the connector pin allocation:

Signal	*Connector pin*	
	+	-
/Orbit	15	16
/Pre-pulse	13	14
/L0	11	12
/L1	7	8
/L1 Data	5	6
/L2 Strobe	3	4
/L2 Data	1	2
Ground	9	10

The connection shall be made with a 16-pin IDC connector (SCEM 09.55.05.716.4) and an 8-twisted pair flat-cable. On both the *FO* and the *LTU* boards, the connector is mounted on the front panel. A single *FO* board provides the connections to four *LTU* boards/to four sub-detector TTC partitions.

The **Orbit**, **Pre-pulse**, **L0**, **L1** and **L2 Strobe** signals are 25ns pulses (1 **BC** interval).

The formats of the serial data strings **L1 Data** and **L2 Data** are shown in **Table 3.6.1**. The transmission of the serial bit 1 coincides with the **L1/L2 Strobe** signal. The figure also shows the location of equivalent bits in the *Sequence List* memory of the LTU emulator, and their placement (word/bit) in the *L1/L2a Messages*, delivered to the sub-detector front-ends. The abbreviations are explained in section **1.5**. “Spare” bits have no functional use, but they are set by the CTP (to logic 0) and transmitted to the front-ends; they are reserved for possible future applications.

The serial transmission time of the **L1 Data** message is 1.45μs (58 **BC**s); the transmission of the **L2 Data** takes 2.425μs (97 **BC**s); both times are significantly less than the minimum time between consecutive **L1/L2** trigger signals (~ 5μs) and consecutive transmissions can never overlap. As a result, no queuing or buffering is required.

L1 Data serial format					
Serial bit	*Data*	*Sequence List*		*L1 Message*	
		Word	*Bit*	*Word*	*Bit*
1	Spare	0	15	1	11
2	**ClT**	0	14	1	10
3..6	**RoC[4..1]**	0	13..10	1	9..6
7	**ESR**	0	9	1	5
8	**L1SwC**	0	8	1	4
9..10	**L1Class[50..1]**	0	7..6	1	3..2
11..12		1	15..14	1	1..0
13..24		1	13..2	2	11..0
25..26		1	1..0	3	11..10
27..36		2	15..6	3	9..0
37..42		2	5..0	4	11..6
43..48		3	15..10	4	5..0
49..58		3	9..0	5	11..2

L2 Data serial format					
Serial bit	*Data*	*Sequence List*		*L2a Message*	
		Word	*Bit*	*Word*	*Bit*
1	**L2arF**				
2..13	**BCID[12..1]**			1	11..0
14..25	**OrbitID[24..13]**			2	11..0
26..37	**OrbitID[12..1]**			3	11..0
38..39	Spare	4	14..13	4	11..10
40	**ClT**	4	12	4	9
41	**L2SwC**	4	11	4	8
42..47	**L2Cluster[6..1]**	4	10..5	4	7..2
48..49	**L2Class[50..1]** or **Detector[24..1]**	4	4..3	4	1..0
50..52		4	2..0	5	11..9
53..61		5	15..7	5	8..0
62..68		5	6..0	6	11..5
69..73		6	15..11	6	4..0
74..84		6	10..0	7	11..1
85		7	15	7	0
86..97		7	14..3	8	11..0

Table 3.6.1 **L1 Data** and **L2 Data** serial format

The content of the messages complies fully with the following documents:

- *Proposal to replace the L1 Trigger-type Word with the L1 Message* [7], approved by the TB on 20 March 2002;
- *Content and format of the L2a Message* [6], approved by the TB on 14 May 2002.

In order to enable an automatic synchronization of the CTP signals with the *LTU* **BC** clock ([20], section **3.14.12**), the CTP logic (*FO* board) is required to transmit over the **L1 Data** line a pattern of alternating ones and zeros - the CTP **BC** clock divided by two, a requirement identical to the one made for the trigger inputs, and depicted in **Figure 3.6.3**.

CTP inputs

The **BUSY** outputs from all 24 *LTU* boards are connected to the corresponding inputs of the CTP (*BUSY Processor* board, **Figure 3.5**). Signal format is the differential LVDS; the cable is a single shielded twisted-pair; the connector, on both the LTU and the CTP side is SCEM 09.31.28.070.5. Since the front panel of a 6U form-factor board cannot accommodate 24 individual connectors, the signals are connected indirectly, *via* a pair of passive *FI* boards (12 inputs each, section **3.22**, ***Fan-in board***).

When the *LTU* is in the *stand-alone mode*, its **BUSY** output is automatically *asserted*. The handling of the **BUSY** signal on the *LTU* board is explained in detail in [20], section **3.9**. For the reason of completeness, **Figure 3.6.1**, taken from the same source, is reproduced in this document.

3.6.2 *Trigger inputs*

The CTP shall accept and process:

- **24 L0** trigger inputs,
- **20 L1** trigger inputs,
- **6 L2** trigger inputs.

All trigger inputs, the **L2** including, shall be generated with the *time precision of a single bunch crossing*. In exceptional cases, trigger inputs with somewhat lower time precision might also be acceptable. Those signals shall remain asserted during the full length of their uncertainty interval.

The trigger inputs shall have the NRZ format, be synchronous with the sub-detector **BC** clock and have the maximum edge jitter of ±2 ns.

The CTP shall *synchronize* all the trigger inputs with the local **BC** clock, and *align* them in time so that the data presented to the processor logic correspond to a common bunch crossing.

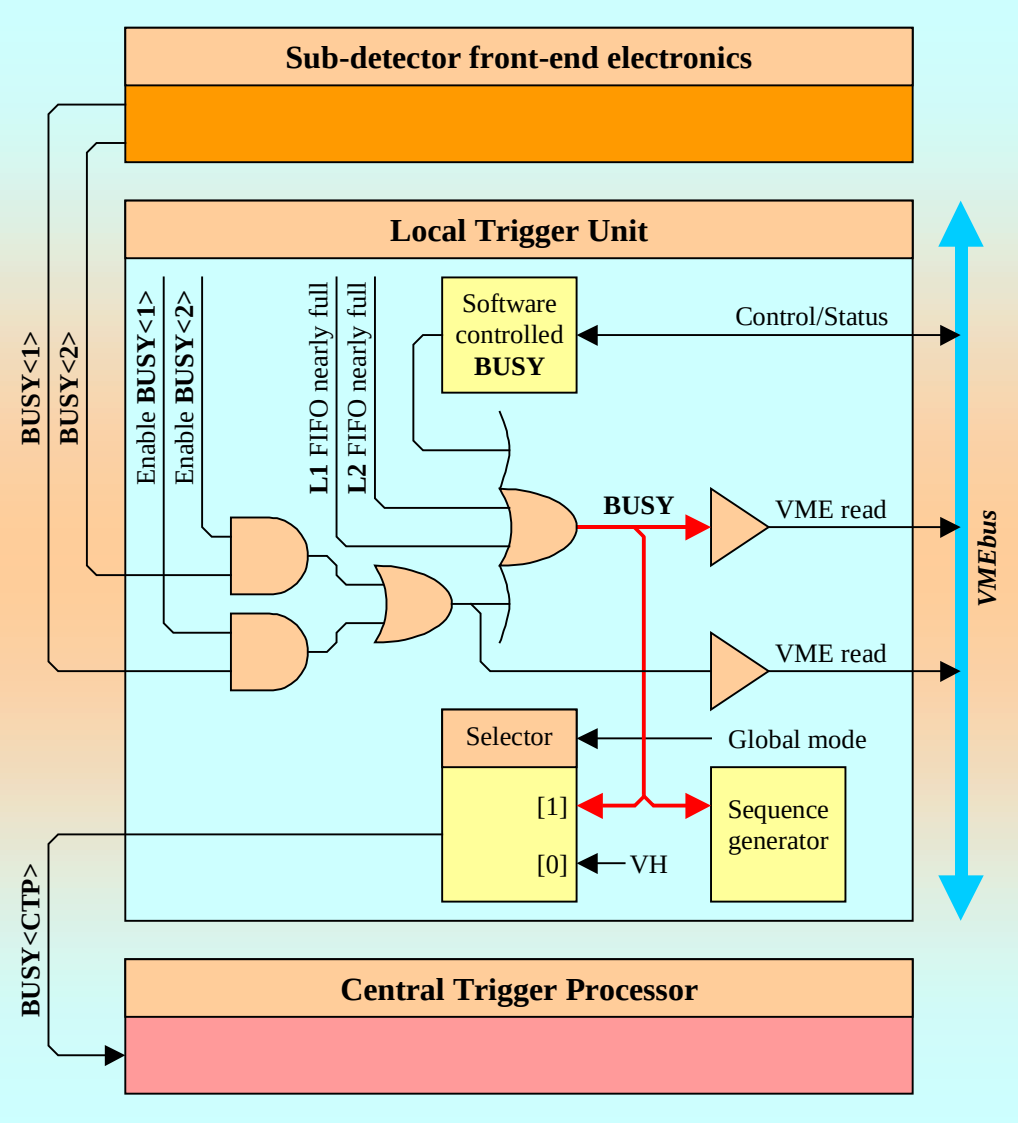


Figure 3.6.1 Handling of the **BUSY** signal on the *LTU* board

Note: *Alignment* and *synchronization* - a reminder:

The *alignment* assures that the trigger signals originating from the same bunch crossing reach the processor logic in the same clock cycle; it is achieved by delaying signals by an appropriate number of *full clock periods*.

The *synchronization* adjusts the phase of the trigger inputs in respect to the **BC** clock (and, indirectly, to the LHC bunch crossing time); the synchronization delays are *within one clock period* - 0 to 25ns.

The *synchronization* will be automatic; during the process, the trigger sub-detectors shall be required to continuously transmit a pattern of “triggers” in alternating bunch-crossing intervals (local **BC** clock divided by two - see **Figure 3.6.3**).

In order to be *aligned*, the trigger signals must reach the CTP inputs within a *window of 16 **BC** intervals (400ns) preceding the corresponding trigger*

decision time; the alignment within the window shall be done by the CTP logic.

3.6.3 *Synchronization of trigger inputs*

There are 50 trigger inputs to the CTP (24 **L0**, 20 **L1**, 6 **L2**), generated at nearly as many locations and all connected *via* individual cables. It is the system requirement that their synchronization with the CTP's **BC** clock shall be automated and their phase monitored.

Also, in order to minimize the overall **L0** trigger latency, the CTP's **BC** clock shall "track" the last-arriving among the **L0** trigger inputs.

In order to meet the above requirements:

- there shall be a programmable delay line on the *BUSY* board (*Data Delay Devices* model PDU15F-1A4, 32 delay steps, 1ns *per* step);
- there shall be an ADC on the *L0*, the *L1* and the *L2* boards (*Linear Technology* model LTC1096LCS8, 8-bit serial ADC), with full access to all the corresponding trigger inputs.

The same hardware has been successfully used on the *LTU* board for a nearly identical application. The measurement of the signal phase in respect to the **BC** clock is explained in [20], section **3.14.12**. The same method shall be used on the CTP boards; during the phase measurement, the trigger sub-detectors are required to generate a pattern of alternating *ones* and *zeros* (**Figure 3.6.3**).

Following the phase measurement of the last-arriving ***L0 Trigger*** input, the **BC** clock from the TTCmi shall be appropriately delayed, before it is distributed to all the CPT boards. The "appropriate delay" shall be derived from a calibrated formula that adjusts the *set-up time* of the last ***L0 Trigger*** to 5-6ns; that minimum value is considered safe even with the trigger input jitter tolerance of ±2ns.

After the **BC** delay is adjusted, a programmable bit shall be set for each trigger input that defines whether the input should be just sampled with the "working", positive edge of the **BC** clock, or sampled first with the negative edge and than again with the positive edge of the clock. The status of the programmable bit is a function of the input phase and it shall be derived from a similar calibrated formula. This widely practised method guarantees, for the 40MHz LHC clock, a *set-up time* and *hold time* of at least 6.25ns, which is considered safe for the quoted jitter tolerance.

A rather general description of the ALICE timing system - design approach and main features, is given elsewhere [21].

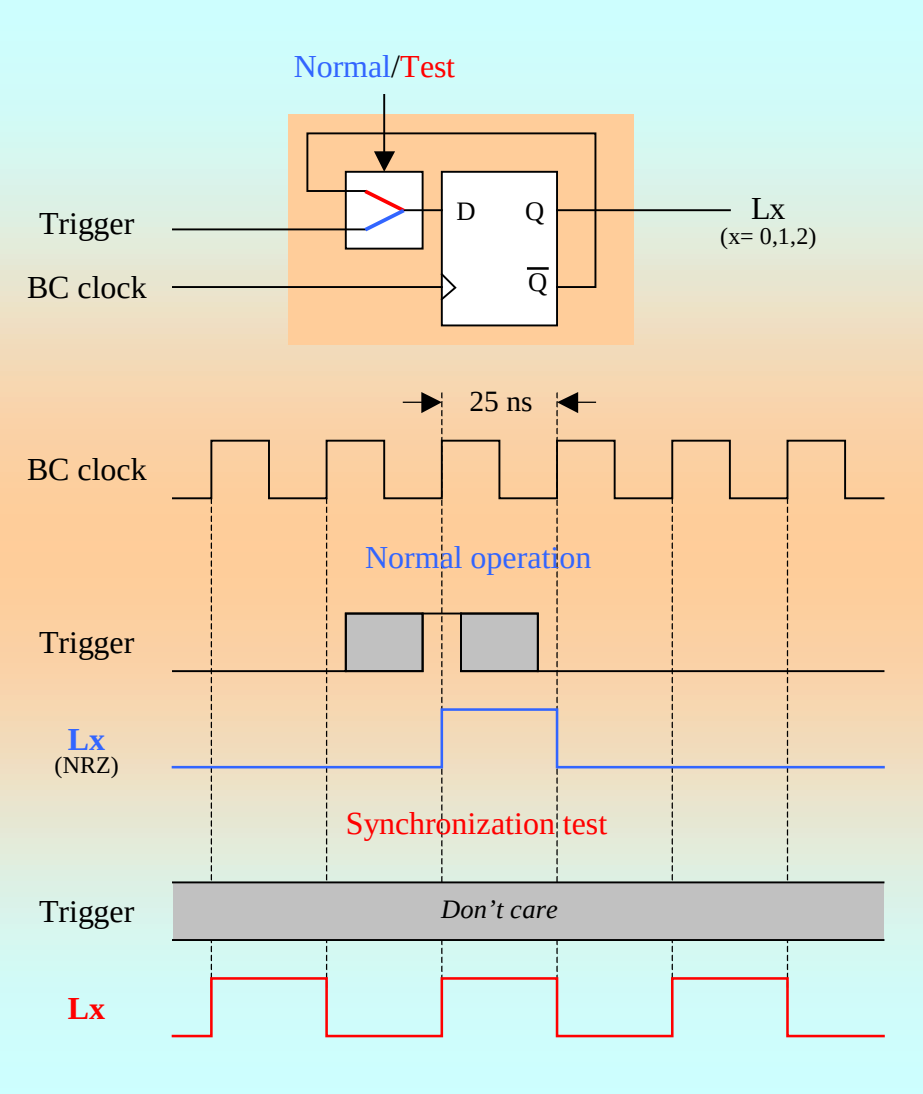


Figure 3.6.3 Synchronization of trigger inputs

3.6.4 *Connection to the Region of Interest Processor (RoIP)*

The *Region of Interest* (RoI) option is still being evaluated since the system will become operational only some years into the LHC run. As a result, the project is not yet fully defined, but, in order to proceed with the development of the CTP, the details of the CTP-RoI *interface* had to be decided and documented [5]. The definition of the interface assures that the "space" is created for the RoI's dedicated hardware and the requirements are set that make the systems compatible and their future integration feasible.

CTP to RoIP output

The serial transmission shall be over two lines - **L1 Strobe** and **L1 Data.** The **L1 Strobe** is an *OR* of all the **Cluster L1** signals (six *physics clusters* and a single *software cluster)* - it coincides with the generation by the CTP of *any* **L1** signal; the serial data stream **L1 Data** contains the following:

– *Enable Segmented Readout* flag **(ESR)**		1
– *Software Class **L1*** trigger status (**L1SwC**)		1
– *Class [50..1] **L1*** trigger status (**L1Class[50..1]**)		50
	Total	**52 bits**

The first bit of the serial data stream **L1 Data** coincides with the **L1 Strobe.** No overlapping of the serial data is possible since the transmission time (1.3μs) is always shorter than the *CTP Dead Time* (minimum time between any consecutive **L1** signals, regardless to which cluster they correspond). The **L1 Strobe** is a 25ns wide pulse; both signals have NRZ format; the signal transitions are synchronized with the CTP's **BC** clock.

The signals are straight copies of the corresponding signals on the CPT backplane - no additional logic is required on the *L1* board. The CTP connection is on the front panel of the *INT* board (**Figure 3.22.1**); the signal format shall be differential LVDS. The connector will be decided during the final design. A possible option is a 10-pin IDC connector; spare pins and spare circuits of the quad LVDS driver could be used to provide spare connections for future use.

A more detailed description of the RoI interface is presented in [5].

RoI BUSY

It might be convenient to add the **RoI BUSY** flag to the interface between the CTP and the RoI system. Since it in not time-critical, the signal could be transmitted *via* the processor network, in a similar way as the **DAQ BUSY**.

The purpose of the signal would be to inform the CTP that, for whatever reason, the RoIP is not able/willing to control/organize the segmented readout. In response, the CTP would halt further generation of the **ESR** flag. The **RoI BUSY** should, therefore, be added as another veto to the **ESR** generation circuit (section **3.15**).

Instead of relying on the **RoI BUSY**, the RoIP could switch to automatically enabling all the sectors in the *RoI Data* message whenever the **ESR** flag is asserted; although the end result is the same, some throughput of the TTC link is wasted and unnecessary waiting added.

Another option would be to set *RoI veto* flag and/or clear *RoI compatible* flag in all the trigger class definitions, but it is a rather "round about" way.

The issue needs to be discussed and an amendment added to the approved version of the RoI Interface proposal [5].

3.6.5 *Interface to the DAQ (RORC)*

The *CTP Readout* and the *Interaction Record* data shall be generated by the CTP and transmitted to the DAQ *via* the ALICE optical Detector Data Link (DDL). The hardware and the communication procedure shall be standard -

identical to the channels that transmit the sub-detector readout. The description of the data format and many aspects of the hardware implementation are presented in [4]; the proposal has been approved by the ALICE TB.

The *CTP Readout* data are generated on the *L2* board and it appears a natural place for the DDL connection. Instead, the interface shall be located on the *INT* board: the necessary data are also accessible *via* the backplane; the *INT* board has more real estate to accommodate a relatively bulky DDL module; the *INT* board contains no part of the CTP logic and will be easier to re-design should a different DDL module emerge - a likely case due to technological advancements during the long "working life" of the experiment.

3.6.6 *Synchronization with the LHC*

The ALICE experiment is synchronized to the LHC with two signals: *Bunch Crossing Clock* (**BC**) and the **Orbit** clock. The inputs are connected to the front panel connectors (50Ω, LEMO, SCEM 09.46.11.180.6) of the *BUSY* board (**Figure 3.22.1**) and distributed, as differential LVDS signals, *via* the backplane, to the other CTP board.

BC input

Signal level: ECL.

The **BC** input is fed from one of the outputs of the *TTC Clock Fanout* module (TTCcf) of the *TTC Machine Interface* (TTCmi)[16]. When the LHC beam is available, the clock is locked to the bunch crossing (40.079 MHz); without the beam, the clock is free-running, at approximately the same frequency.

Orbit input

Signal level: ECL.

The signal is *active low*; about 1μs wide. Its period of 88.924μs/frequency of 11.426kHz corresponds to 3564 bunch crossing intervals.

The input is fed from any of the two identical ORBIT ECL outputs of the LHCrx module in the TTCmi. This is the only direct ALICE connection to the LHC **Orbit** clock. Inside the experiment, the signal is distributed by the CTP, *via* the *FO* boards, *LTU* boards and the sub-detector TTC partitions.

3.6.7 *Interfaces handled by the system software*

Communication with the ECS (**Figure 3.3**), regular readout of a large number of scalers and timers, *etc.* , shall be handled by the system software, using, as a medium, the local network facilities. The management of calibration trigger requests - queuing of the request, arbitration, allocation of priorities, monitoring the success or failure of trigger attempts and reporting back to the

sub-detector that issued the request, is another software task that require careful planning. The organization of the system software is beyond the scope of this document.

3.7 General signal-flow diagram

The general signal-flow diagram is shown in **Figure 3.7**. The main and encouraging conclusion is that the inter-board backplane traffic can be reduced to about 28 signals, clocked at frequencies that never exceed 40 MHz. In those circumstances, the choice of the 6U board form-factor is both feasible and appropriate. The figure also shows that the longest serialized data block contains 52 bits (*Class **L1***) and the corresponding transmission time of 1.3μs is well within the "permitted" *CTP Dead Time* of 2μs.

Note 1: In order to implement the rule that any **L1** signal is always followed by either the **L2a** or the **L2r** decision, the **L2** trigger logic *must* send the serialized **L2** data to the *Fan-out* boards whenever the delayed *Class/Cluster **L1*** is *not equal 0*, and regardless of the status of the *Class/Cluster **L2*** signals. The transmission of the **L2** data with *Cluster **L2*** = 0 leads to the generation of ***L2r** Word* for the corresponding sub-detector(s).

Note 2: Both the *Cluster **L1*** and the *Cluster **L2*** signals are required on the *Fan-out* boards in order to make the **L2a** or the **L2r** decision. The *Cluster **L1*** signals are generated at the time of the **L1** decision and, in order to be used for the **L2** logic, they have to be appropriately delayed; the delay circuit would be required on all the *Fan-out* boards. An alternative, simpler and more economical option is to delay the *Cluster **L1*** signals in an additional channel, "parallel" to the *Class **L1*** delay (serialize/delay/de-serialize sequence), and then transmit them as a part of the **L2** data. That option has been presented in the figure (although the block that delays the *Cluster **L1*** signals has not been shown).

In the following sections, the logic blocks shown in **Figure 3.7** will be described in more details.

3.8 Class-trigger generation

3.8.1 *Trigger classes*

The *trigger class* is a basic processing structure throughout the CTP logic. The CTP shall form **50** independently programmable "physics" trigger classes.

On top of the 50 normal, "physics" trigger classes, there shall be an additional, special trigger class - the *test class*. Apart from the calibration, the only other application foreseen so far for the *test class* is the synchronisation of sub-detector event identifications (bunch number, orbit number). Generation of the *test class* is not time critical and, during the physics run, its rate is going to be very low. It is therefore sufficient to have only one dedicated class, shared by all the applications.

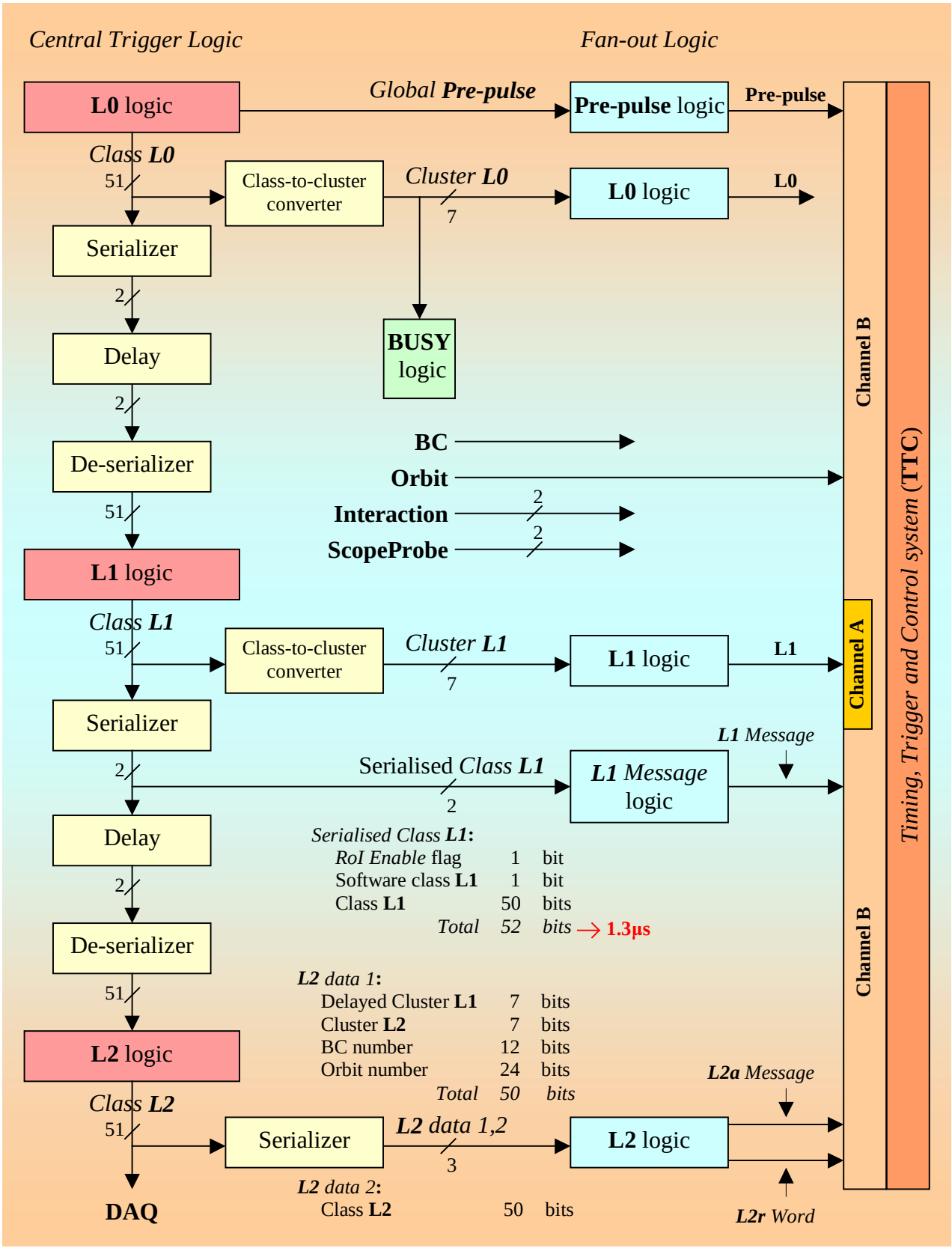


Figure 3.7 General signal-flow diagram

3.8.2 *Sub-detector clusters*

A *cluster* is an internal CTP construct that significantly simplifies the processor logic. It is a programmable and unrestricted grouping of the ALICE sub-detectors; the *clusters* could be exclusive, but it is more likely that they will overlap.

The CTP shall form up to 6 sub-detector *clusters* for "physics" triggers and an additional cluster dedicated to the *Test Class*.

At any time, a *cluster* could be associated with an arbitrary number of trigger classes; a *trigger class*, on the other hand, *affects only a single cluster*. Both associations are programmable; the latter has *significant implications on the design of the CTP electronics*.

3.8.3 *Class L0 trigger logic*

This and the next section deal only with the "physics" classes; the *test class* logic is different and will be described separately (section **3.8.5**).

The *Class **L0** Trigger* logic is shown in **Figure 3.8.3**.

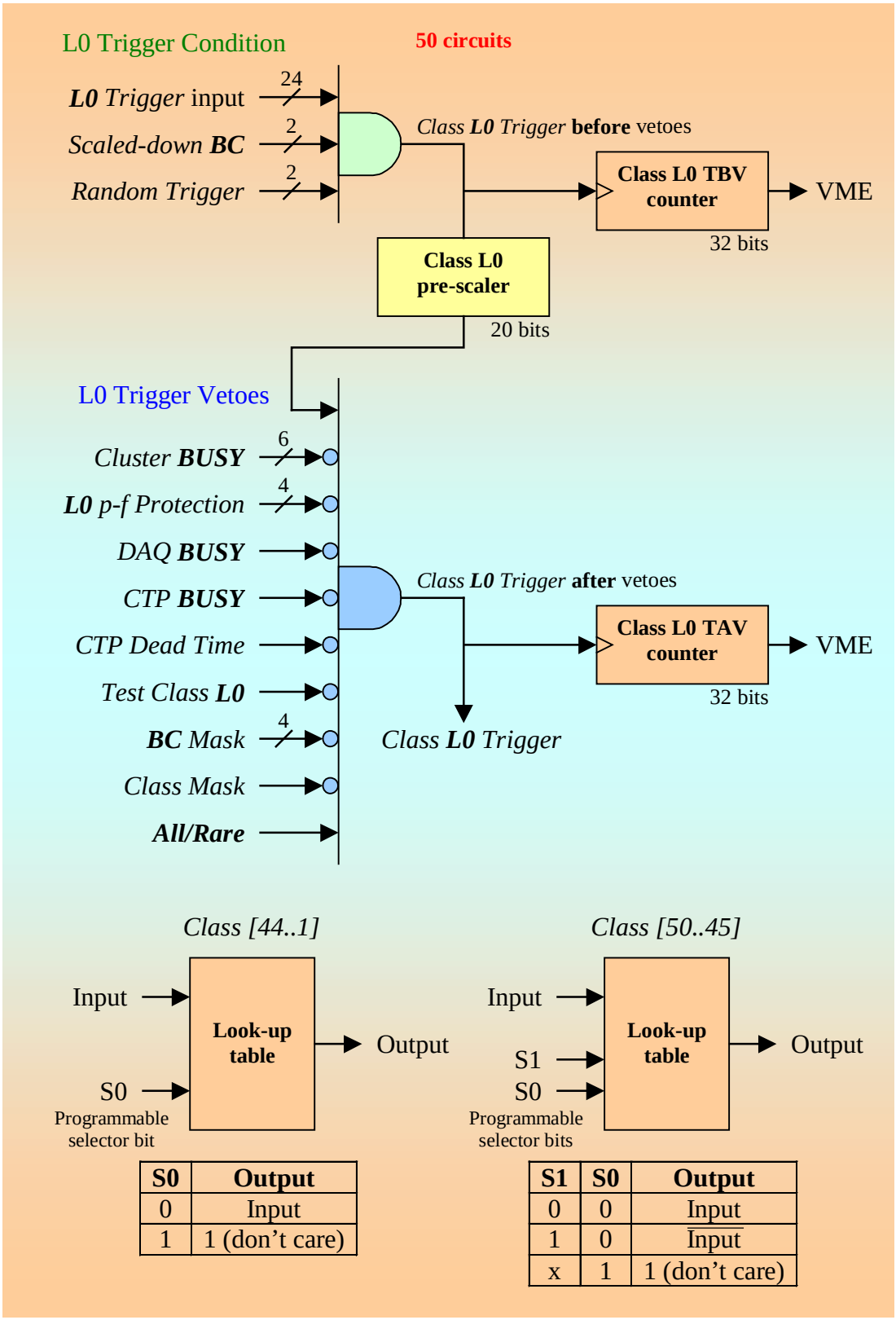
Class L0 Trigger Conditions

The *Class **L0** Trigger Condition* is realized as a logic *AND* of all the 24 ***L0** Trigger* inputs, 2 *Scaled-down **BC*** clocks and 2 *Random Triggers*; all the signals are common to all the classes, but their application is individually programmable. The ***L0** Trigger* inputs for classes 1 to 44, the *Scaled-down **BC*** clocks and the *Random Triggers* can either be selected, or set to a *don't care* state (logic 1). For the classes 45 to 50, the ***L0** Trigger* inputs can be selected; or their complements; or they can be set to a *don't care* state. The need for such a requirement is explained elsewhere [8].

Both *Scaled-down **BC*** inputs are a 25ns-pulse, synchronous with the **BC** clock, with the programmable interval between the pulses in the range from 0 (continuous asserted input) to ~25s (a 30-bit counter circuit).

Both *Random Triggers* inputs are a random pattern of 25ns-pulses synchronous with the ***BC*** clock, generated by a 31-bit linear feedback shift register. The pulse distribution is pseudo-random - the pattern repeats itself every ~53s ($[2^{31} - 1]$ BC intervals). The number of pulses distributed within each pseudo-random pattern-repetition period is programmable (range: 1 to $[2^{31} - 1]$).

Note: The *Scaled-down **BC*** and the *Random Trigger* inputs are indeed *Class **L0** Trigger conditions* when used *independently from the **L0** Trigger inputs*; in combination with the ***L0** Trigger* inputs, they effectively act as *Class **L0** trigger vetoes*. This should be taken into account in the cross-section calculations [11].



S0	Output
0	Input
1	1 (don't care)

S1	S0	Output
0	0	Input
1	0	$\overline{\text{Input}}$
x	1	1 (don't care)

Figure 3.8.3 Class **L0** trigger logic

Class L0 pre-scalers

In order to balance the contribution from different trigger classes, the *Class **L0** triggers* are *pre-scaled*; the pre-scaler is a 20-bit counter with a programmable scaling-down factor (range: 1 to ~10^6).

Note: There are two possible locations for the *Class **L0** pre-scalers* - *before* or *after* the vetoes; in general, the choice might introduce a trigger bias [22]. In the case of ALICE, it has been shown that the effect is negligible [23].

Class L0 Trigger Vetoes

Some signal definitions are given elsewhere: *DAQ **BUSY*** - [2], section **3.6.2**; *CTP **BUSY*** - [2], section **3.1.18**; *CTP Dead Time* - [2], section **3.1.3**; ***BC** Mask* - [2], section **3.1.9**; *Class Mask* - [2], section **3.1.11**.

The *DAQ **BUSY***, the *CTP **BUSY*** and the *CTP Dead Time* are *mandatory* vetoes (not selectable) for all classes/all clusters.

Note 1: In the proposed CTP realization, the *DAQ **BUSY***, the *CTP **BUSY*** and the *Cluster **BUSY*** shall be combined, on the *BUSY* board, into a single signal (**Figure 3.14**).

The *CTP Dead Time* shall be programmable, but the hardware *must* prevent the setting below the necessary minimum - sufficient time to transmit serialized data (52 bits/1.3μs at the moment).

Note 2: In principle, the dead time should always be kept at the minimum value, but the programmable option could be useful for testing purposes (evaluation of the effect of longer dead time, for example).

Each trigger class is associated with a single *cluster*; the corresponding *Cluster **BUSY*** signal reflects the BUSY status of all the sub-detectors included in the cluster. The appropriate signal is selected from the group of 6, and used as a *Class L0 Trigger Veto*; the selection is *mandatory*.

Note 3: The "mandatory" requirement is, again, the consequence of the implementation (see Note 1): if no signal were selected, the obligatory vetoes of the *DAQ **BUSY*** and the *CTP **BUSY*** would have been masked out. It will be, nevertheless, still possible to overwrite the actual sub-detector BUSY/cluster BUSY status - a necessary tool for system testing and debugging, by a temporary action of the control software (**Figure 3.6.1**).

The "mandatory requirement" shall be secured with the selection code that, by design, excludes the "*none*" option.

There are 4 programmable ***L0** Past-future Protection* circuits/signals; any subset of the signals (including *all* and *none*) could be selected as a class veto.

Note 4: The selection of the past-future circuit depends upon the set of sub-detectors that constitute the cluster and is likely to be the same for all the classes that are associated with the same cluster. The proposed selection (*per* class) reflects the implementation; a uniform cluster-protection setting can be preserved, be it at a slight inconvenience, but the circuit offers an additional flexibility to fine-tune the protection for each class.

The *Test Class **L0** Trigger* signal is used as a veto to ensure that no "physics" trigger can occur simultaneously with a software trigger; this is a *mandatory* (not selectable) option.

The *Class Mask* is a flag controlled by the software that inhibits (when asserted) or enables (when cleared) the generation of the *Class **L0** Trigger* signal in the corresponding class.

The **All/Rare** signal is part of the mechanism that boosts the acquisition of "rare" events. The signal is controlled by the system software and it is common to all the classes. The signal should be *selected* for the classes that are *not associated* with rate events. If selected, it acts as a veto when it is cleared - during the acquisition of rare events only.

Note 5: The ALICE handling of rare events is described in more detail in [10]; preliminary discussions and some alternative proposals are also available on the Web [12].

3.8.4 *Class L1 and Class L2 trigger logic*

The *Class **L1*** and *Class **L2** Trigger* generation is depicted in **Figures 3.8.4.1** and **3.8.4.2**. The two circuits are practically identical; although significantly simpler, they are similar to the *Class **L0*** trigger logic (**Figure 3.8.3**).

There are 20 ***L1** Trigger* inputs and 6 ***L2** Trigger* inputs. The signal selection is different for the classes 1-44 and the classes 45-50; the options are identical to those available for *Class **L0** Triggers*.

The *Class **L1** Trigger*/*Class **L2** Trigger* is a coincidence of selected ***L1/L2** Trigger* inputs and the appropriately delayed, corresponding *Class **L0/L1** Trigger.*

The 4 *Past-future Protection* circuits are the only trigger vetoes; their selection is the same as in the case of *Class **L0** Triggers.*

3.8.5 *Test Class trigger logic*

Referenses: CTP URD [2], sections **3.1.7** (***Software trigger***), **3.1.8 (*Calibration trigger*)**, **3.1.15 (*Test class*)**.

Test Class definition

The software trigger shall be *requested* by a VME write (dummy data)**;** beforehand, the *Test Class* must be fully defined. The *Test Class* definition includes:

- *Synchronous/Asynchronous Software Trigger* flag;
- *Calibration Trigger* flag;
- ***BC** Number* (in case of a synchronous, non-calibration trigger);

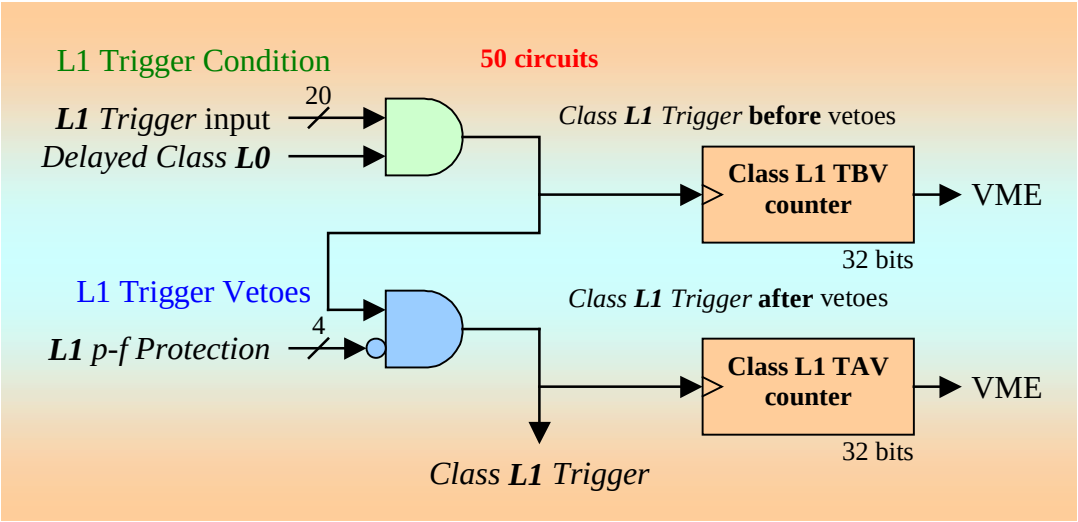


Figure 3.8.4.1 Class **L1** trigger logic

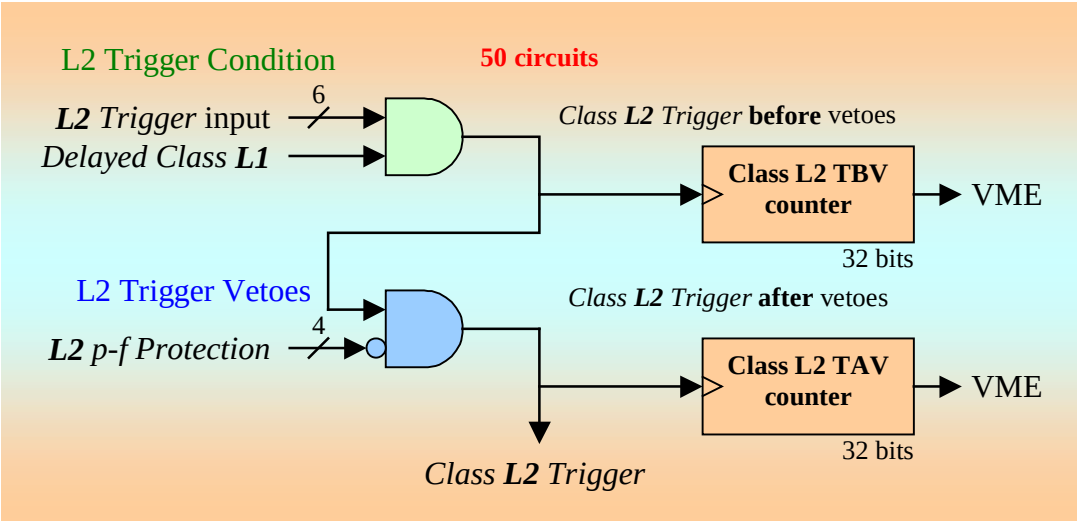


Figure 3.8.4.2 Class **L2** trigger logic

- a list of participating sub-detectors (only a single sub-detector, in case of a calibration trigger);
- *Test Class **L0/L1/L2** Past-future Protection* specification;
- *Readout Control [4..1]* bits; the bits are sub-detector dependent and must be defined for all participating sub-detectors.

In the case of a calibration trigger, the following *system parameters* are also used:

- *Calibration **BC*** is a bunch-crossing in the *Long LHC Gap*, selected for generation of *all* calibration triggers;
- *Pre-pulse **BC*** is a bunch-crossing selected for generation of *all* calibration **Pre-pulse** signals.

The *system parameters* are programmable; they are set during the CTP configuration and cannot be altered "on the fly". A possible change of their value requires coordinated modifications in the configuration files for the CTP, the DAQ and the sub-detector front-ends.

Generation of the Software Trigger

The *Software Trigger Request* is set by a VME write and synchronized with the **BC** clock; the flag is cleared automatically when the *trigger attempt*, successful or otherwise, is eventually made.

Following the request, the generation of the software trigger depends on the selected option.

- *Asynchronous trigger*: following the request, the trigger is generated in the first bunch-crossing that is not masked by the *No Trigger* bunch-crossing mask bit. *No Trigger* mask shall always "cover" the whole *Long LHC Gap*, in order to "reserve" the gap for the calibration triggers only; the mask could also include, for whatever reason, other bunch-crossings. The mask is set during the system configuration and cannot be changed "on the fly".

  Note: The described "wait for a not-masked bunch-crossing " algorithm shall strongly favor the bunch-crossings immediately after any *No Trigger* zone. *Is such a bias significant?* The alternative is to "fail" immediately any attempt that happens to coincide with the declared *No Trigger* area; the option increases the likelihood of asynchronous software trigger failure. *Decision required*. Or is it all rather pedantic?

- *Synchronous non-calibration trigger:* following the request, the trigger is generated in the *selected bunch-crossing*. Any bunch-crossing is "eligible", but the *Long LHC Gap* and, in particular, the C*alibration* ***BC*** ("reserved" for the calibration triggers) should be used with care in order not to confuse the sub-detector logic.

- *Calibration trigger:* after the request, the **Pre-pulse** is generated in the *Pre-pulse* ***BC***, followed by the software trigger in the *Calibration* ***BC*** (always inside the *Long LHC Gap*).

The generation of the *Software Trigger* is depicted in Figure **3.8.5**.

Generation of the Test Class L0 trigger

The circuit is similar to the *Class* ***L0*** logic (**Figure 3.8.5**): the *Test Class* ***L0*** signal is set by the *Software Trigger* if the *Test Class* ***L0*** *Past-future Protection* bit, the *Test Class* ***BUSY***, the *DAQ* ***BUSY,*** the *CTP* ***BUSY*** and the *CTP Dead Time* are all off. A programmable ***BC*** *Mask* is not required at this level since it is used in the generation of the *Software Trigger*. The ***L0*** *Trigger* inputs are ignored.

The *Test Class* ***L0*** signal disables any possible simultaneous "physics" *Class* **L0**. The two types of classes "do not mix": their ***L2a*** *Message* formats are incompatible; also, the sub-detector readout modes are, in general, different.

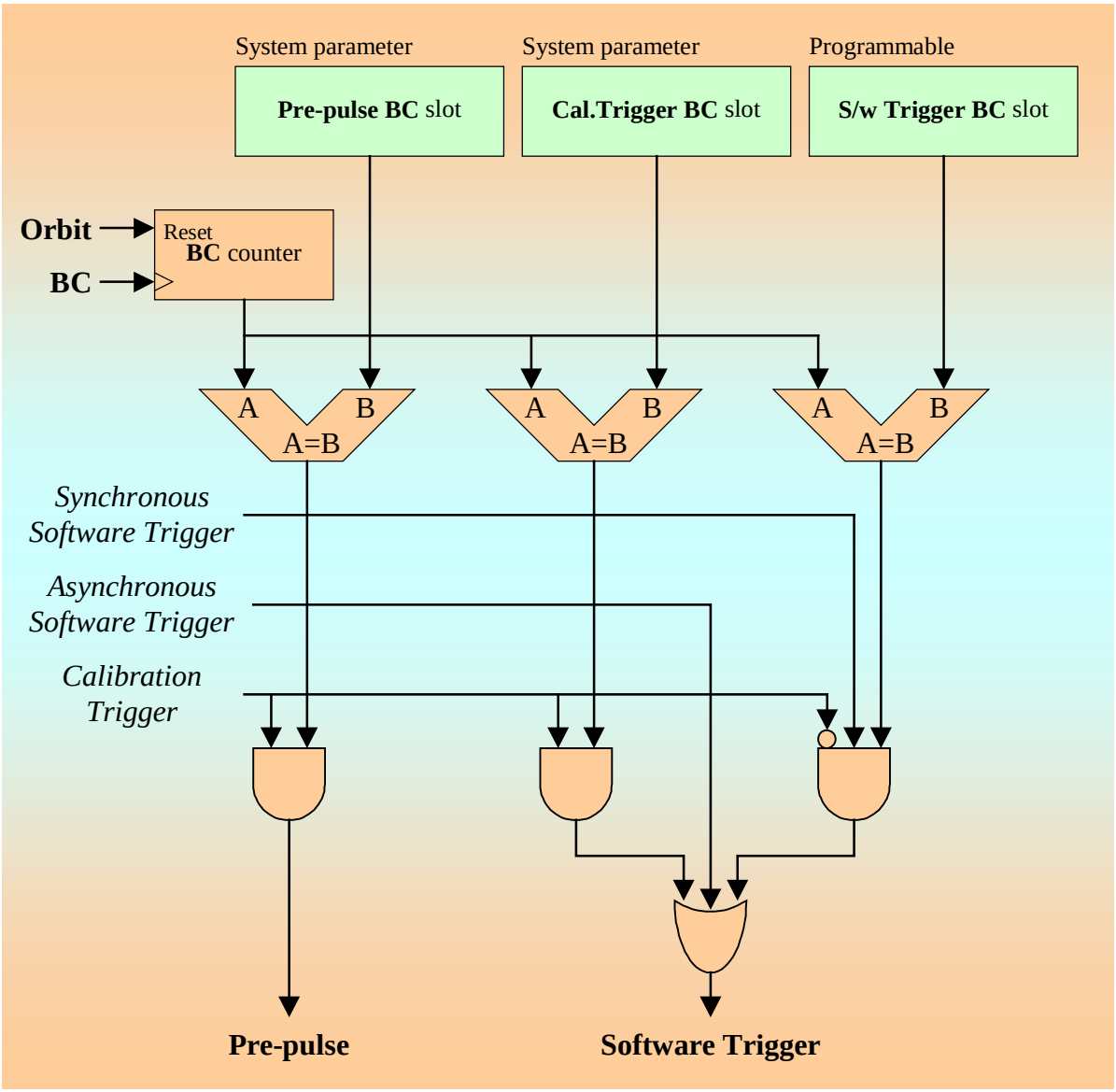


Figure 3.8.5 Generation of the *Software Trigger*

The *Test Class* is given the priority since its rate is low and the corresponding "loss of physics" is negligible.

Note: There is a danger of "signal racing": the generation of the *Software Trigger*, its propagation through the *Test Class* ***L0*** *Trigger* logic and, if successful, the prevention (*veto*) of any simultaneous "physics" *Class* ***L0*** must all be done in the same **BC** interval and in good time. This is just a reminder; the detailed design shall resolve a possible conflict.

The *Test Cluster* ***BUSY*** is generated on the *BUSY* board as an *OR* of **BUSY** signals of all participating sub-detectors - the sub-detectors that constitute the *Test Cluster.*

The *Test Class **L0** Past-future Protection* logic is dedicated to the test class only. It consists of a single general past-future protection circuit, in addition to the four used by the "physics" classes.

The *Test Class **L0*** trigger asserts the *Test Class **L0** Acknowledge* flag. The flag is read and cleared by the software at appropriate time; it is used to monitor the progress of a software trigger.

Generation of the Test Class L1 trigger

The signal is set by the delayed *Test Class **L0*** signal if the *Test Class **L1** Past-future Protection* bit is off. The ***L1** Trigger* inputs are ignored. The past-future protection logic is the same as for the *Test Class **L0*** signal.

The *Test Class **L1*** trigger asserts the *Test Class **L1** Acknowledge* flag. The flag is read and cleared by the software at appropriate time; it is used to monitor the progress of a software trigger.

Generation of the Test Class L2a/L2r signals

The *Test Class **L2a*** signal is set by the delayed *Test Class **L1*** signal if the *Test Class **L2** Past-future Protection* bit is off; otherwise, the **Test Class *L2r*** signal is generated. The ***L2** Trigger* inputs are ignored. The past-future protection logic is the same as for the *Test Class **L0**/**L1*** signals.

The *Test Class **L2a/L2r*** signals assert the corresponding *Test Class **L2a/L2r** Acknowledge* flags. The flags are read and cleared by the software at appropriate time; they are used to monitor the progress of a software trigger.

Following the generation of the *Test Class **L2a/L2r***, the corresponding ***L2a** Message/**L2r** Word* is sent to all participating sub-detectors in the same way it is done for the "physics" classes.

Synchronization of software trigger requests

The CTP hardware provides the *Test Class **BUSY*** flag that is automatically set by the processor when *the Software Trigger Request* is made (VME write, dummy data); the flag is cleared by the processor (another VME write) when it detects the completion of the requested operation.

Note: Strictly speaking, the flag is not necessary since the processor should keep track of its operations; the processor might decide not to use it and rely instead on its own, software generated/software controlled equivalent. Even in that case, the flag could usefully serve as a double check.

The processor *may not* either request another software trigger or load/change the values of registers containing data that specify the current test class configuration while the *Test Class **BUSY*** flag is asserted - previous request hasn't yet been completed.

Monitoring of the Test Class stages

The generation by the CTP hardware of the following status flags has been described in the previous sections; "automatically" means "by the CTP hardware, at the appropriate time":

- *Software Trigger Requ*est (VME write, cleared automatically),
- *Test Class **BUSY*** (VME read and clear, set automatically),
- *Test Class **L0** Acknowledge* (VME read and clear, set automatically),
- *Test Class **L1** Acknowledge* (VME read and clear, set automatically),
- *Test Class **L2a** Acknowledge* (VME read and clear, set automatically),
- *Test Class **L2r** Ackn*owledge (VME read and clear, set automatically).

The *Test Class **L0/L1/L2a/L2r** Acknowledge* signals are generated by the hardware, and read and cleared by the control processor; the approximate duration of their assertion is predictable. The *Test Class **L0** Acknowledge* hardware is located on the *L0* board; the hardware for the *Test Class **L1** Acknowledge* is on the *L1* board; the *Test Class **L2a/l2r** Acknowledge* flags are generated on the *L2* board.

By reading the status of the *Acknowledge* flags at *appropriate time*, the control processor can monitor the progress of the requested test class operation, find out whether it has "succeeded", or identify the stage at which it has failed.

The simplest scheme would require the processor to read, at *appropriate time*, only the *Test Class **L2a** Acknowledge* flag: if the flag is set, the operation has succeeded; otherwise - it failed. If all the flags are read, the software can identify the stage in which the operation has been halted. The *Test Class **L2r** Acknowledge* flag is a redundant double-check; it is read (and cleared) simultaneously with the *Test Class **L2a** Acknowledge*.

The "appropriate time" for *asynchronous triggers*, counted from the moment when the software trigger has been requested, is roughly equal to the **L2** decision time; for *synchronous triggers* (including the *calibration trigger*), the time of one LHC orbit should be added - the worst case of waiting for the selected bunch-crossing. A small safety margin should also be added in both cases.

Following the reading of the *Test Class **L2a** Acknowledge* flag, all the *Acknowledge* flags and, if used, the *Test Class **BUSY*** flag must be cleared by the control processor (VME write, dummy data).

3.9 Generation of the Interaction signals

The CTP shall generate two **Interaction** signals; the signals are simultaneously used by all *Past-future Protection* circuits and as the trigger and the content of the *Interaction Record*.

An additional signal - **Interaction Test**, is used only for testing and on-line evaluation of alternative interaction definitions; the signal is counted, but has no effect on the CTP operation.

The generation of the **Interaction** signals is depicted in **Figure 3.9**. A 16x1 programmable look-up table outputs *any* logic combination of the ***L0*** *Trigger [4..1]* inputs. The output is a normal selection during a physics run. Alternatively, for system testing and development, any of the two *Scaled-down* ***BC*** clocks, or two *Random Triggers* can be selected (the same signals are used for the *Class* ***L0*** *Trigger* generation, **Figure 3.8.3**). In case of the **Interaction Test**, those alternative options are not required.

Note: The *OR* function of the two **Interaction** signals is used as a "trigger" to enter the corresponding bunch-crossing into the *Interaction Record. At any time*, the *OR* must entail the *minimum bias interaction*; example: a simultaneous definition of *central* and *semi-central* interaction is illegal. The note is just a reminder to colleagues who will be developing the expert software editor for the interaction definitions.

Another hint: if the definitions are exclusive, the two interaction signals could be used as a code for up to 3 different types of interaction (example: 0 - no interaction; 1 - *minimum bias*; 2 - *semi-central*; 3 - *central*).

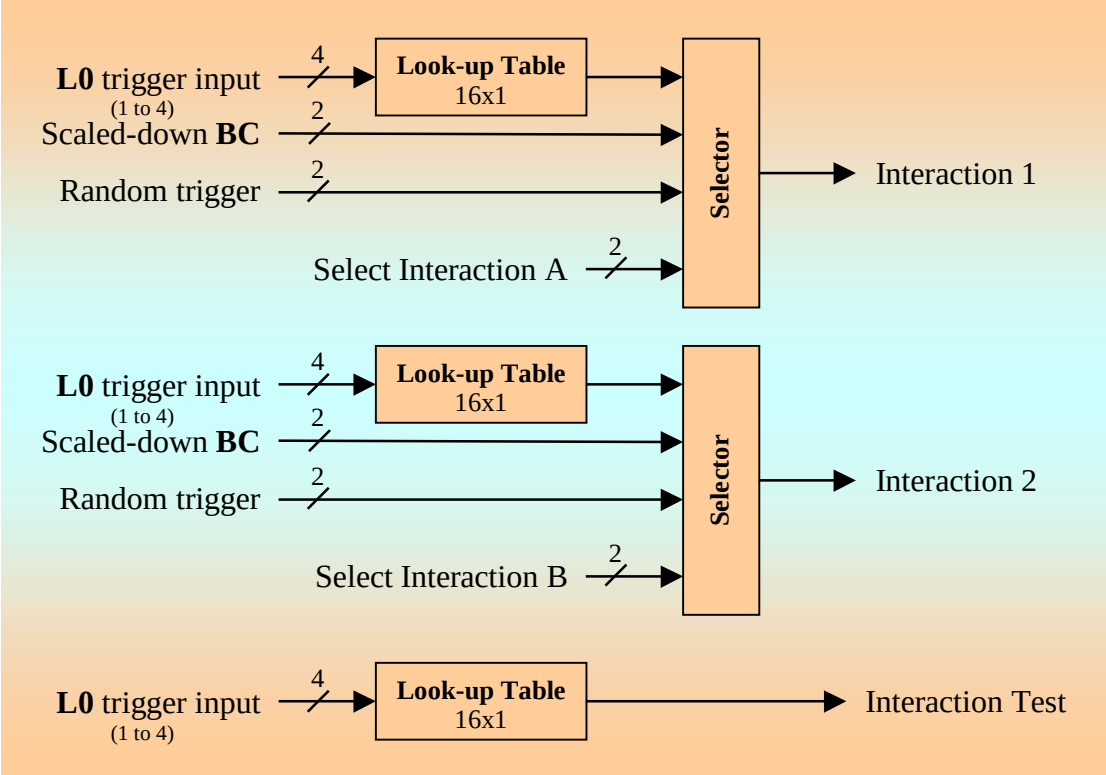


Figure 3.9 Generation of **Interaction** signals

3.10 Past-future Protection circuit

"The past-future protection circuit should be able to differentiate between a low multiplicity and high multiplicity interaction so that, for example, several low multiplicity interactions could be allowed in the past-future protection period, but no other types of interactions". (Quoted from the minutes of the trigger meeting on 27 and 28 September 2001 at CERN.)

There shall be *four* fully programmable past-future protection circuits; each *trigger class* shall be associated with an arbitrary subset of them (including *all* and *none*); the "association" shall be programmable.

There shall be also an additional identical circuit dedicated solely to the *Test Class*.

The circuit is shown in **Figure 3.10.1**. The interaction inputs **INTa** and **INTb** are each a programmable function of the two interaction signals (**Interaction 1** and **Interaction 2**, **Figure 3.9**); their generation is depicted in **Figure 3.10.2**. Each block has two programmable thresholds (**THx1** and **THx2**) and two corresponding outputs (**Px1** and **Px2)**. The protection intervals (**ΔTa**, **ΔTb**) are independently programmable. The two *delay* blocks (**Delay a**, **Delay b**) serve to align the protection results with the time when the protection is checked (the ***Time alignment*** diagram in **Figure 3.10.1**).

The output selection depicted in **Figure 3.10.1** is just a convenient "physics" example. The real circuit is shown in **Figure 3.10.2**: the protection output **P** is an *arbitrary function* of the **P1**, the **P2** and the **Delayed INT** signals.

The **Delayed INT** carries the information about the interaction that triggered the event and the delay serves to align the signal with the time when the protection is checked. The **Delayed INT** signal is common to all the *Past-future Protection* circuits at the same trigger level.

Note: The delay of the **Delayed INT** must have the full **BC** resolution (25ns). Since the maximum delay is ~100μs (**L2** decision time), this can be realized with a circuit based on a *Dual-port Memory* (DPM) with a capacity of 4kx1.

Programmable parameters

A more detailed diagram of the basic *Past-future Protection* block is shown in **Figure 3.10.3**.

The programmable parameters include:

- Two *protection threshold*, **THx1** and **THx2,** defined as a permitted number of interactions during the protection interval.
- Time resolution of the protection interval definition, expressed as a *pre-scaling factor* **N** (the same for both blocks) of the scaled-down **BC** clock that generates the write and read strobe and triggers the address counter of the dual-port memory.

  Example: for **N** = 32, the time resolution is 32·25ns = 800ns.

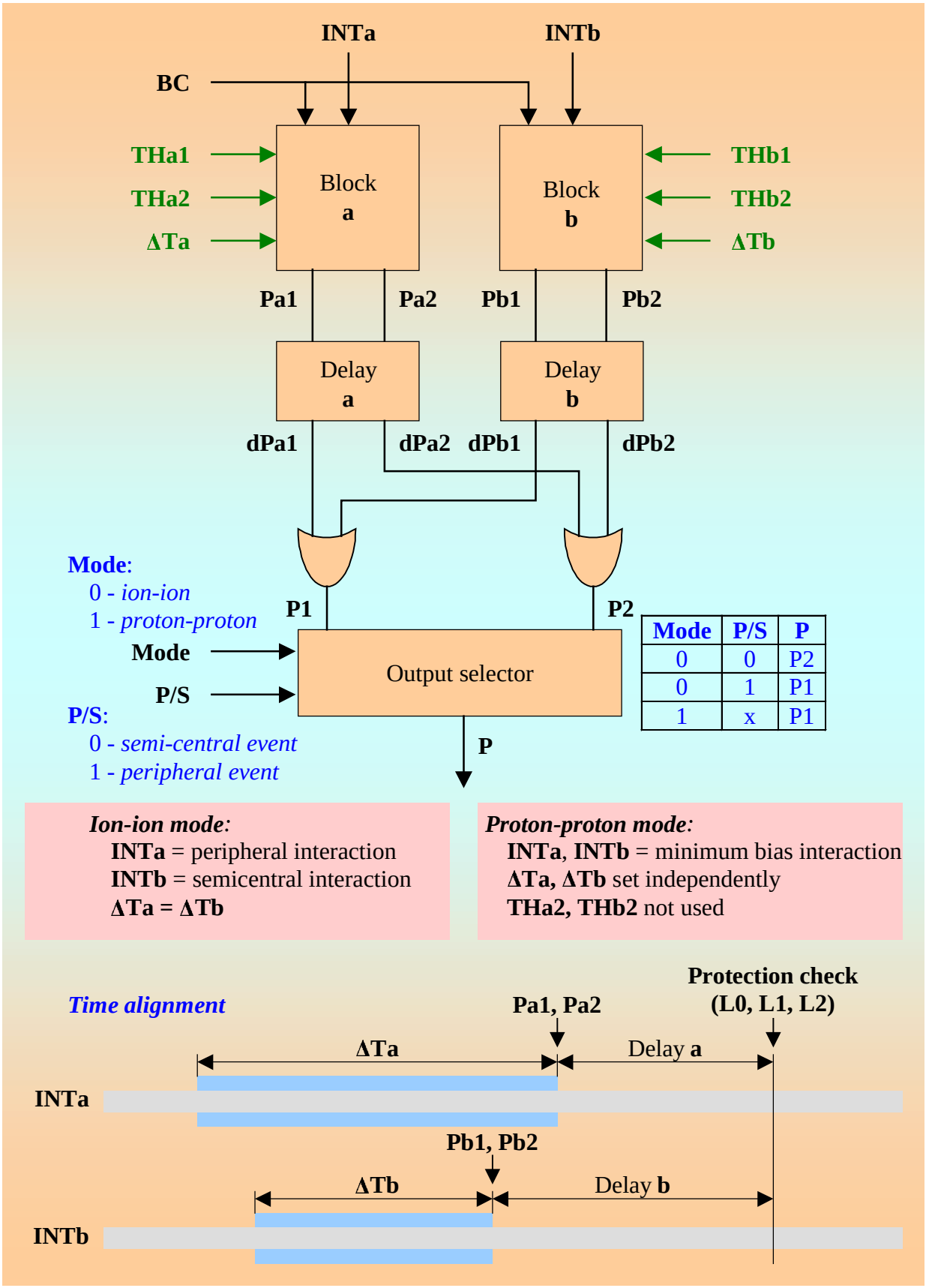


Figure 3.10.1 Block diagram of the *Past-future Protection* circuit

- Duration of the *protection interval*, expressed as a number **ΔTx** of resolution intervals.

  Example: for **ΔTx** = 255 and **N** = 32, the protection interval is

  ΔTx·**N**·25ns = 255·32·25ns = 204.8μs.

The maximum value of the *protection threshold* is 64.

In order to satisfy the requirement that the *resolution* must be better than 1% of the *protection interval*, the capacity of the dual-port memory is chosen to be 256 words. The memory capacity and the maximum duration of the protection interval (~2·100μs) determine the required maximum value of **N**:

$$\mathbf{N_{max}}\cdot 25\text{ns} \geq 2\cdot 100\mu\text{s}:256 \;\rightarrow\; \mathbf{N_{max}} \geq 31.25 \;\rightarrow\; \mathbf{N_{max}} = 32\ .$$

Example 1:

±88μs could be protected with a resolution of 2·88μs:256:25ns = 27.5 → **N**=28 → 28·25ns=*700ns*, which is ~0.8% of the interval; **ΔTx** = 2·88μs:700ns=251.43 → **ΔTx** =252; protection interval is **ΔTx** ·**N**·25ns = 252·700ns = *176.4μs*.

Example 2:

±3.2μs could be protected with the maximum resolution of 25ns (**BC** interval): 256·25ns = 6.4μs.

Example 3:

±10μs could be protected with the resolution of 75ns.

Note: Regardless of the time resolution setting, or, for that matter, the setting of any other programmable parameter, *all the interactions are accurately counted*. The loss of time resolution only makes the boundaries of the protection interval more "fuzzy", but they, nevertheless, remain well within the 1% requirement.

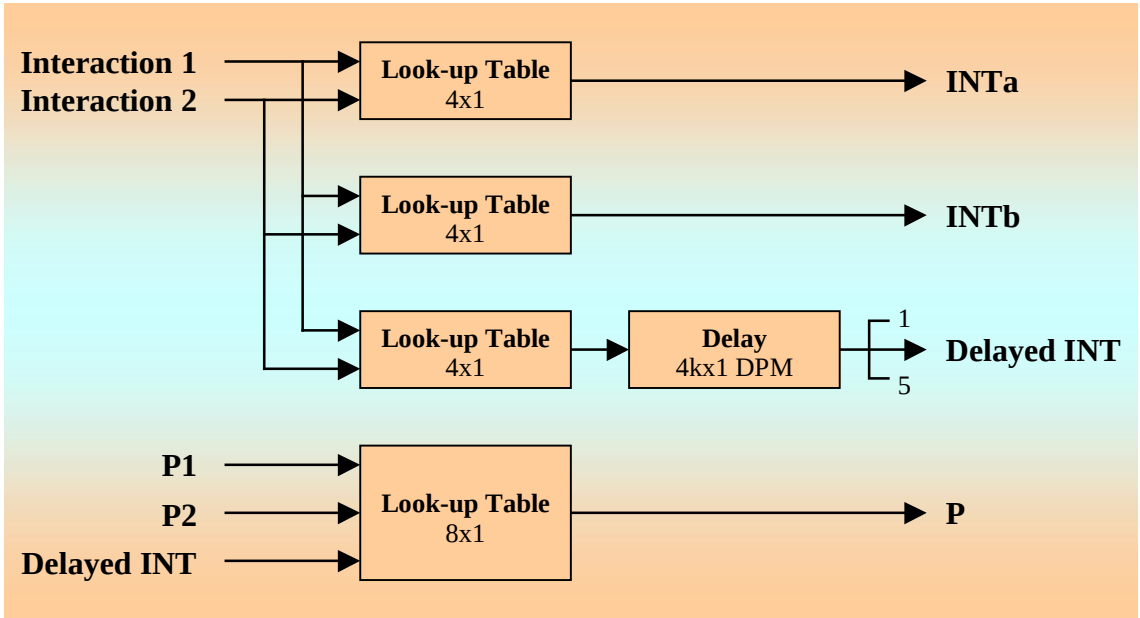


Figure 3.10.2 Generation of the *Past-future Protection* signals

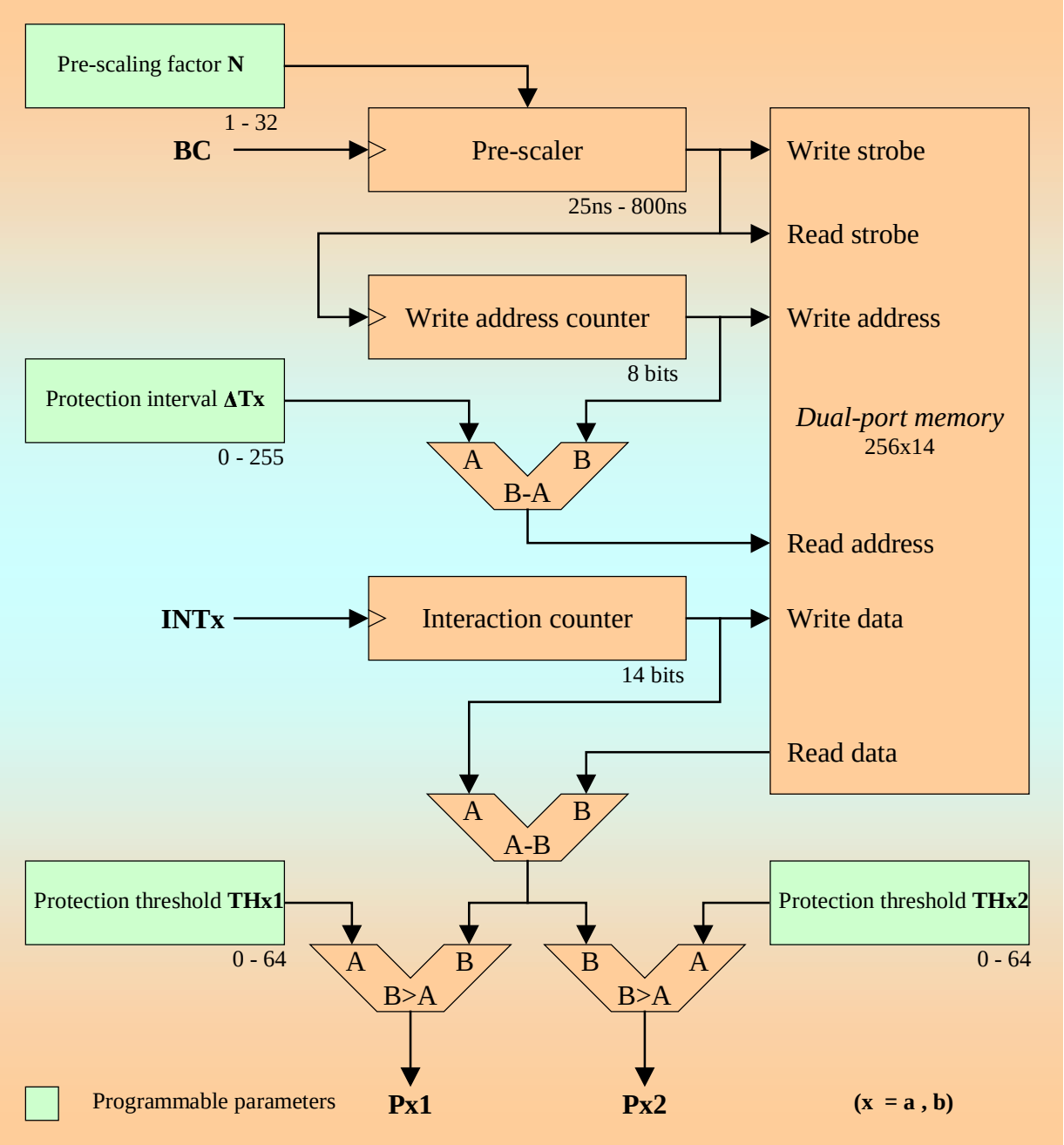


Figure 3.10.3 *Past-future Protection* block A/B circuit

Since the maximum protection interval is ~200μs, the corresponding maximum interaction count is 200μs:25ns = 8k - equivalent to 13 bits. The blocks shall be realized as a *Dual-port Memory* with the capacity of 256 words, 14-bit wide.

Delay blocks

The two *delay* blocks (**Delay a**, **Delay b** - **Figure 3.10.1**) serve to align the protection results with the time when the protection is checked (see the ***Time***

alignment diagram in the figure). Each block shall be realized as a *Dual-port Memory* with the capacity of 2k words, 2-bit wide.

The delay memories shall operate with the same time resolution as the memory in the corresponding protection block - there is no advantage in having a higher resolution in the delay section only. The worst case (the longest delay) is a protection interval that just exceeds the **L1** time. Such an interval can be realized with the resolution of 50ns: 256 x 50ns = 12.8μs → ± 6.4μs. The **L1** to **L2** delay of ~100μs requires a memory with the capacity of 100μs/50ns = 2k.

Note: Just a reminder: if the protection interval is shorter than the **L1** time, there is no need to repeat the protection check at the **L2** time; the "redundant" protection circuit could be used to enhance the protection options at the **L2** level.

3.11 Generation of the Cluster L0/Cluster L1 signals

The generation of the six *Cluster **L0*** signals from the fifty *Class **L0** Triggers* is depicted in **Figure 3.11**. The *Cluster **L0*** signals are sent to all the *Fan-out* boards and used to generate the corresponding sub-detector **L0** triggers.

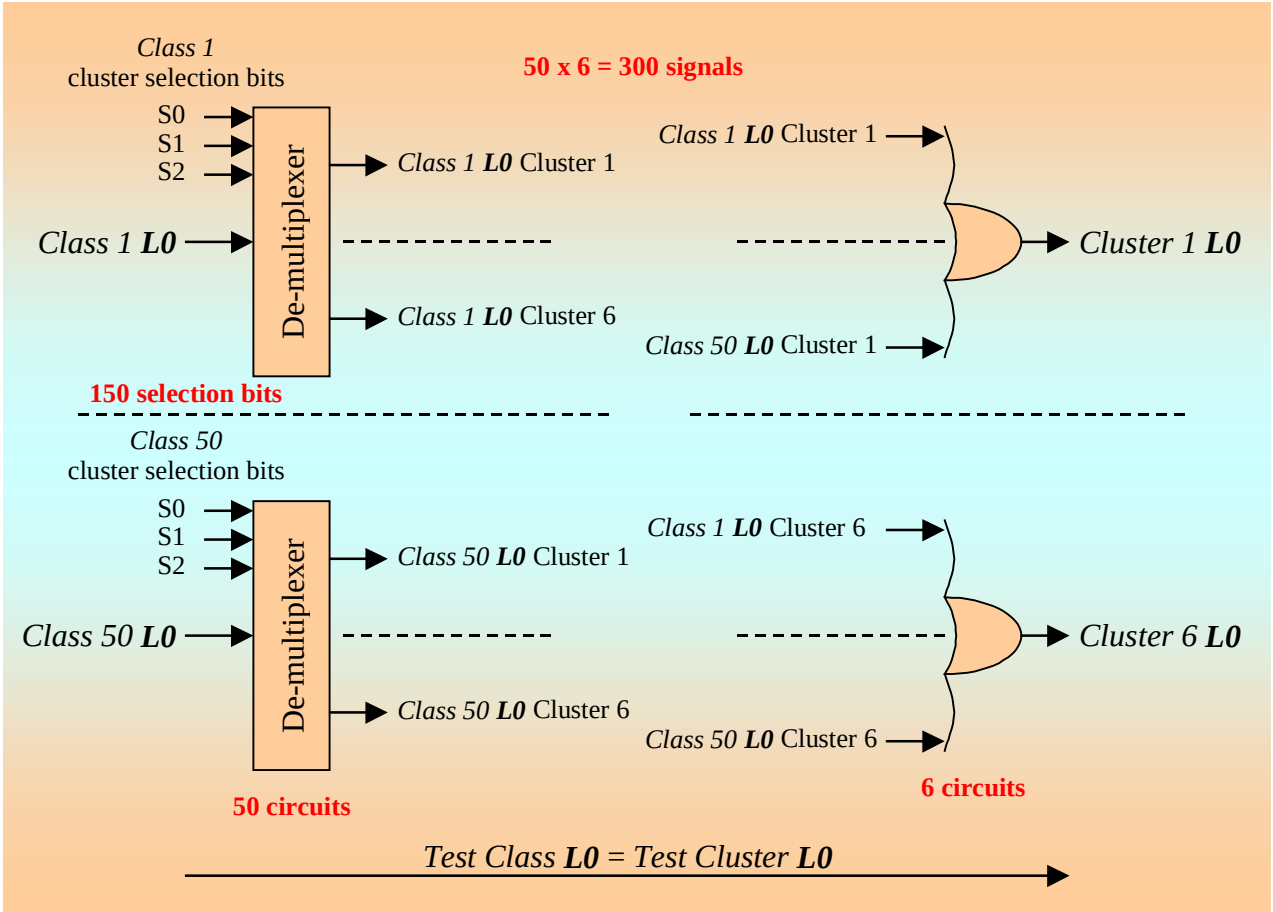


Figure 3.11 Generation of the *Cluster **L0*** signals

The crucial advantage of the "cluster" approach is the reduction from 50 to 6 of the number of signals transmitted to the *Fan-out* boards.

Note: The hardware overhead might not be an "overhead" at all: it is much simpler, on the *Fan-out* boards, to generate the sub-detector **L0**s from the six cluster signals rather than from fifty class signals, and particularly so since the logic operation has to be repeated for each sub-detector (24 times).

The sole *Test Cluster **L0*** is logically identical to the only *Test Class **L0***.

An identical circuit, on the *L1* board, is used to generate the *Cluster **L1*** signals.

3.12 Serialization, delay and de-serialization of the Class Triggers

Class L0 Triggers

The *Class **L0** Trigger* serializer/delay/de-serializer logic blocks are part of the logic shown in **Figure 3.7**.

The details of the circuit implementation are depicted in **Figure 3.12**. The serializer is located on the *L0* board; the delay and the de-serializer are on the *L1* board; the only signals transmitted over the backplane are the 51-bit serial data and the strobe that coincides with the first serial data-bit.

The serialization is triggered by the *Any Class **L0*** signal; the de-serialization starts at the arrival of the strobe.

Note: A reminder: *Any Class **L0*** signal could also be derived from the *Cluster **L0*** signals - 7 instead of 51 inputs to the OR circuit. The same signal could be used to generate *CTP Dead Time* (section **3.14**).

The *Dual-port Memory* with the capacity of 512x2 provides for the maximum programmable delay of 12.8μs.

Class L1 Triggers

A very similar logic structure shall be used to transmit the *Class **L1** Trigger* signals from the **L1** logic block to the **L2** logic (**Figure 3.7**). A *Dual-port Memory* with the capacity of 4kx2 shall provide for a longer programmable delay of up to 102.4μs.

Note: This is an attempt to clarify a somewhat vague description of the transmission of the *Cluster **L1*** signals made in section **3.7** (***General signal-flow diagram***). The seven *Cluster **L1*** signals are serialized and attached behind a header - a binary "1101" is a rather robust 4-bit code, already successfully used on the *LTU* board. This serial stream is transmitted and delayed instead of the single-pulse strobe signal. On the **L2** logic side, the header is recognized as a strobe, and the *Cluster **L1*** signals de-serialized. No overhead - additional delay channel or a backplane connection - is required; no time-penalty is incurred - a delay of 0.1μs in delivering the ***L1/L2a/L2r** Messages* is academic.

All this is just an implementation detail, far too complicated to be included in either **Figure 3.7** or **Figure 3.12**, only a reminder, to be taken into account during the detailed design.

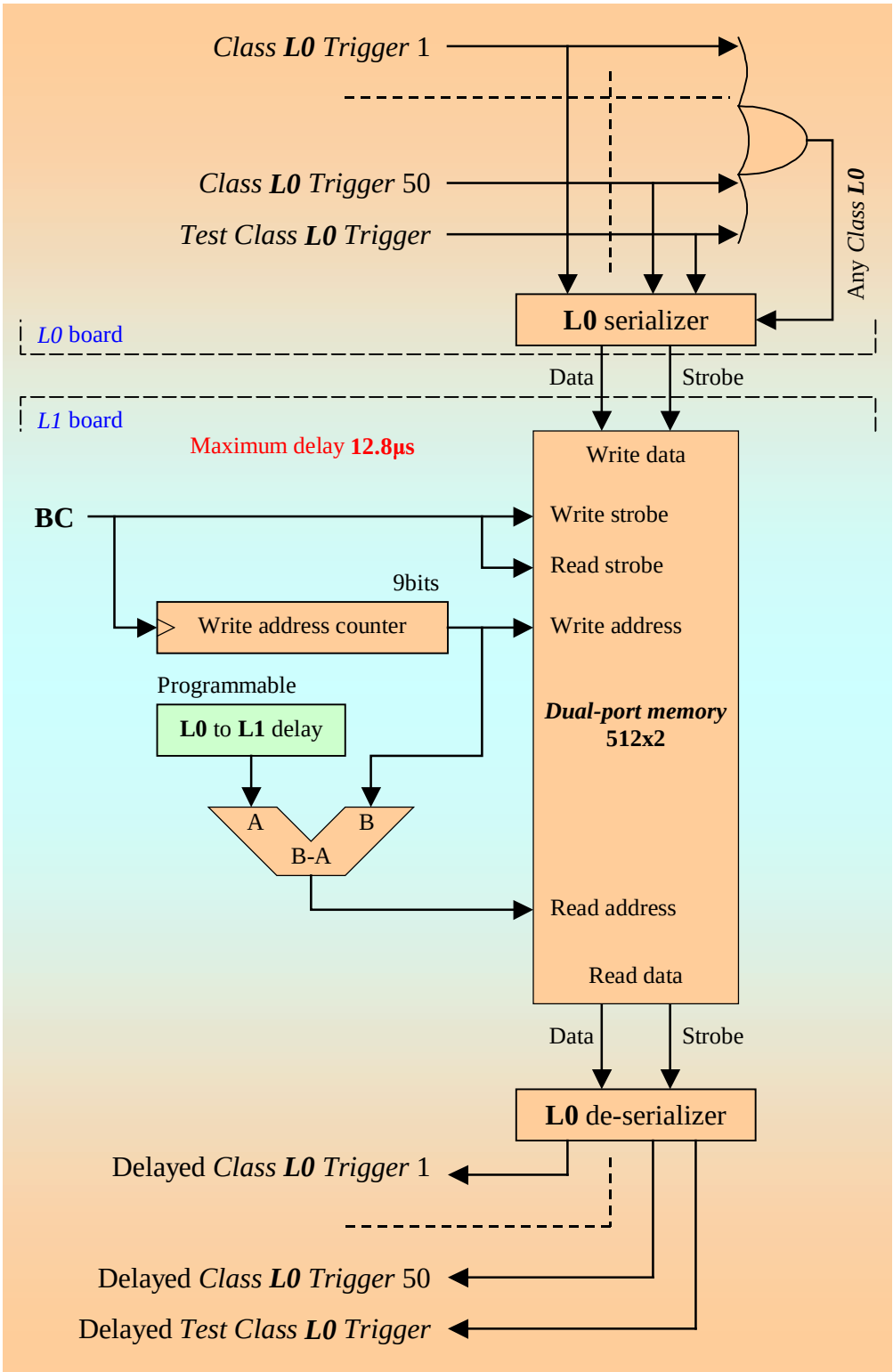


Figure 3.12 Serialization, delay and de-serialization of trigger signals

3.13 Generation of the sub-detector signals (*Fan-out* board)

The generation of the sub-detector signals on the *Fan-out* board (*FO*) is depicted in **Figure 3.13**. All the inputs are global signals, common to all the boards; the logic shown in the figure corresponds to a single sub-detector. A single *FO* board shall provide for four sub-detectors.

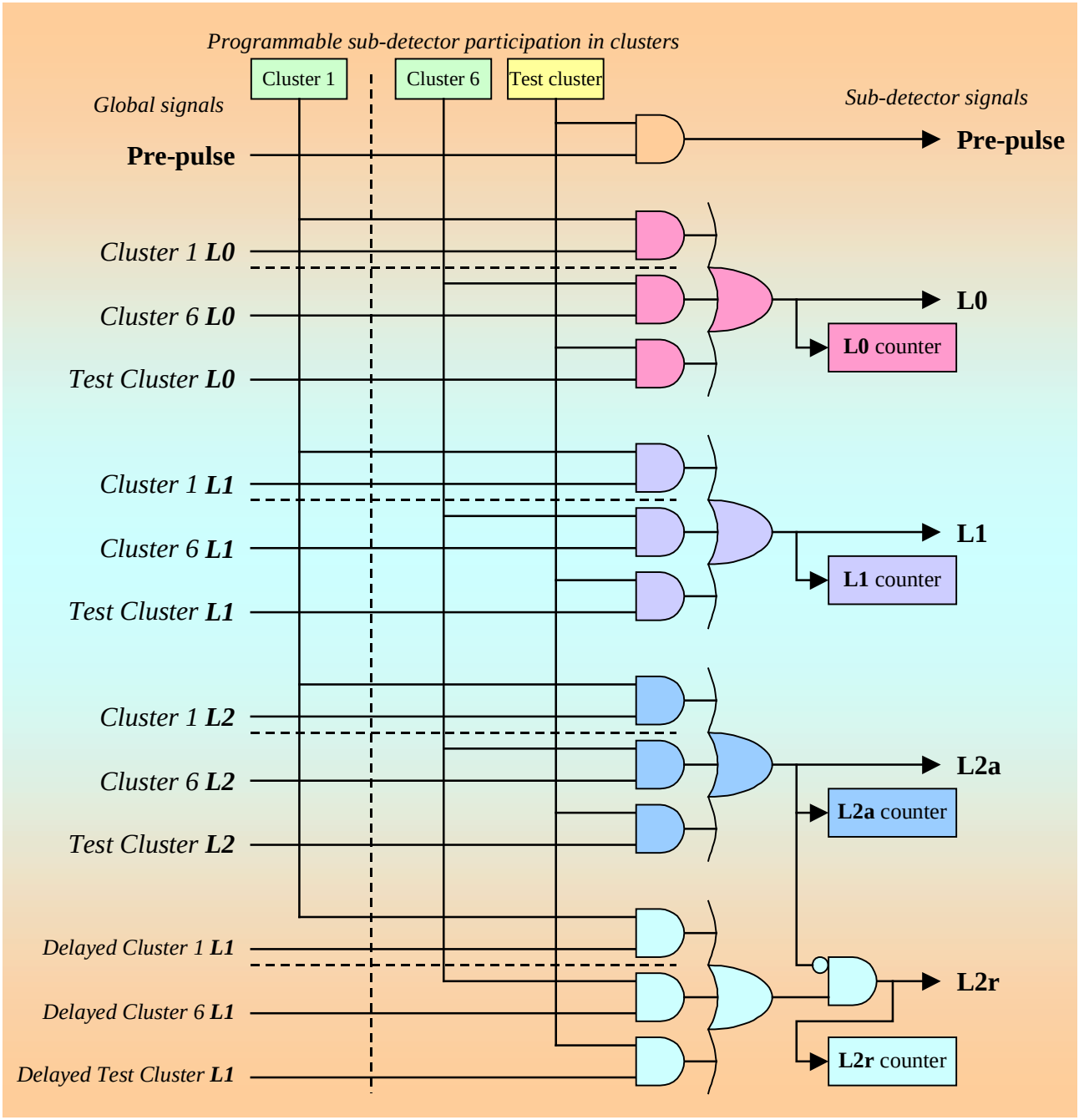


Figure 3.13 *Fan-out* board - generation of the sub-detector signals

The *Delayed Cluster **L1*** signals could be derived from the *Cluster **L1*** signals, distributed at the time of the **L1** decision, but it is more economical to transmit them as a part of the serialized **L2** data (section **3.7**).

Each **L1** signal serves as the strobe and "triggers" the transmission, from the *FO* board to the corresponding *LTU*, of the serialized **L1 Data**. The **L2 Strobe** (section **3.6.1** - ***Connections to the LTU***) is the logic *OR* of the **L2a** and the **L2r** signals; it starts the transmission of the serialized **L2 Data**. The format of the serialized **L1/L2 Data** is shown in **Table 3.6.1**.

3.14 Generation of the Cluster BUSY signals

The generation of the *Cluster **BUSY*** signal is shown in **Figure 3.14**. Six circuits provide for the six "physics" clusters and an additional circuit is used for the test cluster. The logic is located on the *BUSY* board.

BUSY signals from sub-detectors that participate in a given cluster (*Cluster **n*** in the figure) are all *OR*ed together. The cluster configuration is programmable; for the 6 "physics" clusters, it is set at the beginning for the whole duration of the run; for the *test cluster,* the configuration is re-defined on request "on the fly".

In response to a *Cluster **L0*** trigger, and automatically - regardless of the sub-detector **BUSY** signals, the cluster becomes busy until the corresponding **L1** decision time. A circuit that generates the *Cluster Dead Time* (from **L0** to **L1**) is also shown in **Figure 3.14**.

The *Cluster **n** Dead Time* is unconditionally *OR*ed with the other "ingredients" of the *Cluster **n BUSY*** signal.

In general, the cluster configurations overlap. In order to prevent the illegal generation, during the **L0 - L1** interval, of surplus **L0** triggers to sub-detectors participating in more than just a single cluster, the *Dead Time* signals of all the overlapping clusters *must* be also *OR*ed into the overall *Cluster **BUSY*** (**Figure 3.14**).

The "map" of cluster overlapping is implicit in the cluster configuration, but the logic to extract and format the information requires 501 2-input *AND* gates and 21 24-iput *OR*s (or the equivalent in FPGA logic elements) - a massive and unnecessary overhead. Instead, the "map" shall be generated and loaded by the software, together with the cluster configuration.

Note 1: A convenient format would be to "merge" the cluster configuration (24 bits, a bit *per* sub-detector) and the corresponding overlapping "map" (6 bits, a bit *per* "physics" cluster) into a singe control word. A half of the overlapping bits is redundant - the overlapping "status" bits for clusters 1 and 2/2and 1 are always identical, but the redundancy provides for cleaner and clearer both the logic and the software.

A separate control word of a similar format would define the configuration and the "mapping" of the *Test Cluster*.

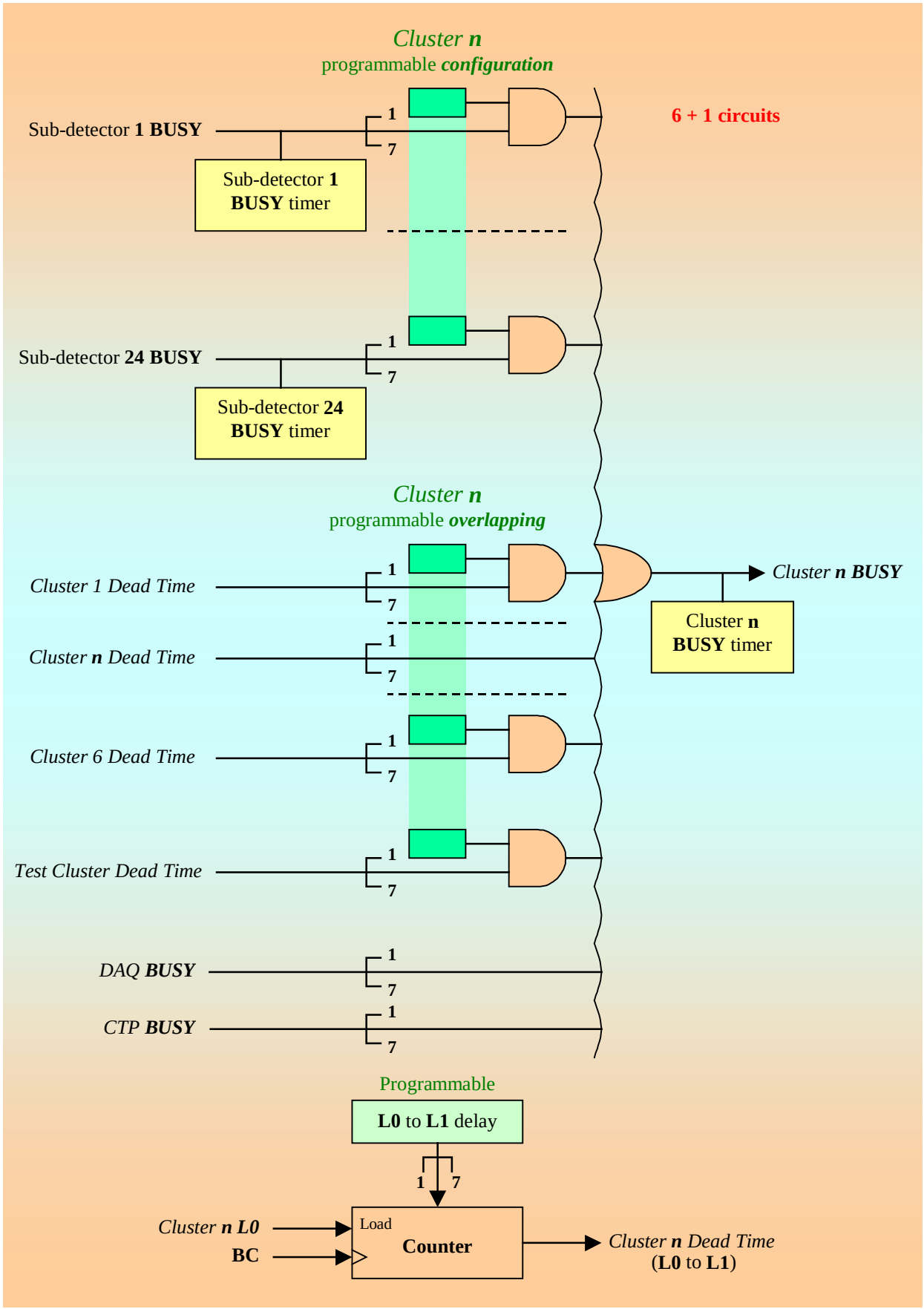


Figure 3.14 Generation of the *Cluster **BUSY*** signal

In order to unload the logic from the crowded *L0 Processor* board (*L0*), the **L0** vetoes that are mandatory and common to all the classes - *DAQ* ***BUSY***, and *CTP* ***BUSY***, shall be *OR*ed with the *Cluster* ***BUSY*** signals on the *BUSY* board; no additional backplane connections are required.

Note 2: The *CTP Dead Time* (~2μs) is also a mandatory veto, common to all the classes/cluster, but its timing is critical since it must prevent a possible trigger in the very next bunch crossing. Such a short latency is hard to achieve if the veto, a response to an **L0** trigger, were to come from another board. For that reason, the *CTP Dead Time* shall be "administered" on the *L0* board.

The latency of the *Cluster* ***BUSY*** signals is not critical since the global *CTP Dead Time* always "protects" the first couple of microseconds.

3.15 Generation of the Enable Segmented Readout (ESR) flag

The **ESR** is generated by the CTP at the time of the **L1** decision. The flag is included in the ***L1*** *Message* and broadcast to all the sub-detectors participating in the trigger; it is also part of the ***L1*** *Data* serially transmitted to the RoIP. When the flag is asserted, the readout format of the sub-detectors participating in the RoI option is controlled by the RoIP - it could include the data from *all*, *none*, or only from a *selected group* of sectors.

Full description of the RoI interface is presented in [5]; this section deals only with the generation of the **ESR** flag. The logic is located on the *L1* board.

The logic diagram of the CTP block that generates the **ESR** is shown in **Figure 3.15**. A programmable description of a trigger class that always requires a complete readout (*all* sectors included) shall have the *RoI Veto* bit asserted; only classes associated with clusters that include a feasible subset of RoI sub-detectors shall have the *RoI Compatible* bit on. The **ESR** flag shall be generated if no active trigger class has the *RoI Veto* set and at least one active cluster is *RoI Compatible*. The RoI option is never enabled in case of the *Test Class*.

The **RoIP BUSY** signal (section **3.6.4**) unconditionally inhibits the generation of the **ESR** flag.

3.16 Monitoring - scalers and timers

The final list of signals to be scaled shall emerge during the detailed design of the CTP boards. The list will certainly include the counters required for the cross-section calculation [11], the counters of the interaction signals, and all the counters already shown in **Figures 3.8.3, 3.8.4.1, 3.8.4.2, 3.13**, *etc.*; many more will be added to enhance the monitoring of the system operation and, also, to help the development and debugging - the experience with the *LTU* board confirmed how valuable they can be.

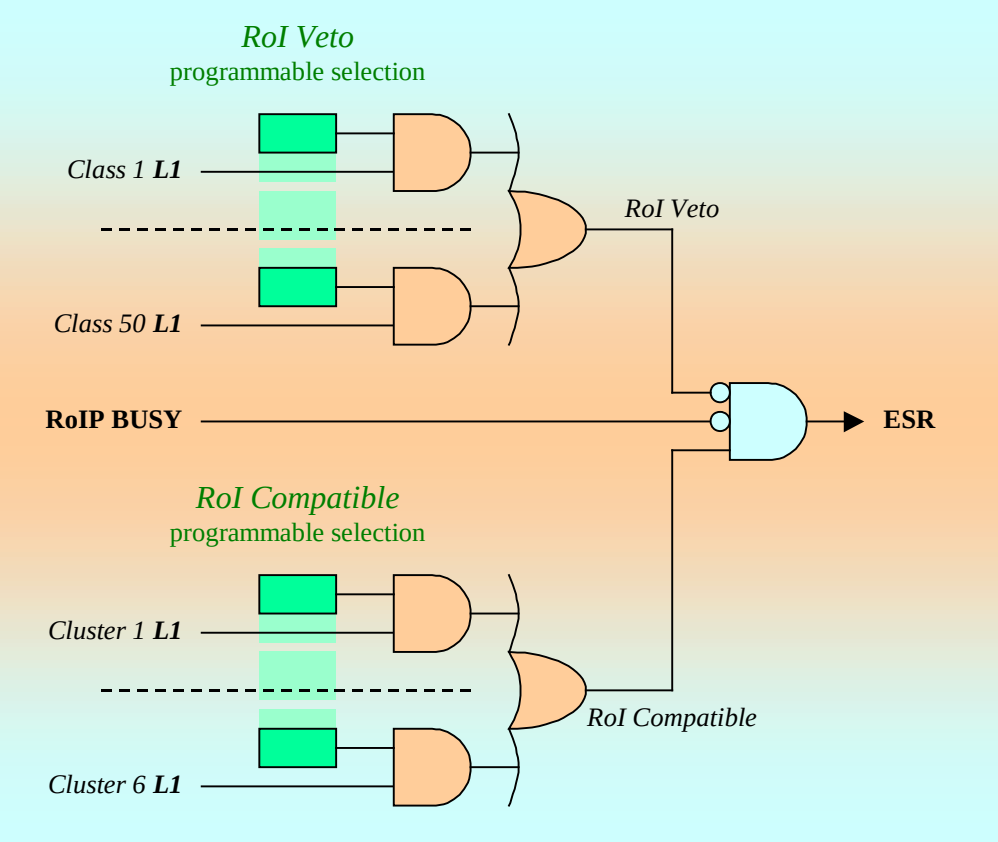


Figure 3.15 Generation of the *Enable Segmented Readout* flag

All the counters shall have the 32-bit capacity. They shall use the *Gray* code in order to enable the readout "on the fly"; during the VME read, the content of the selected counter shall be automatically converted into the binary format.

All the 24 sub-detector **BUSY** inputs and all the *Cluster **BUSY*** signals shall be timed (**Figure 3.14**). The timers shall also be 32-bit *Gray-code* counters, clocked with a ~ 0.4μs clock (**BC** signal divided by 16) when the input signal is asserted. The counter capacity provides for 28.6 minutes of integral signal-active time.

3.17 *SnapShot* memory

The *SnapShot* memory has been successfully used on the *LTU* board - accompanied with a user-friendly content-analysis software, it has proved to be a valuable tool for system development, debugging and monitoring. The option also enables access to samples of data from time intervals selected at random, or at will, rather than determined in a biased way by the trigger decisions (the distribution of trigger inputs, *etc*.).

On some of the boards, the *SnapShot* memory shall be used to "play-back" the recorded/generated signal patterns, which provides a stable "signature" stimulus for the analysis of the hardware operation.

To enable the software "alignment" of data from different *SnapShot* memories, the **Orbit** signal shall be recorded as a time reference by *all* the units (for more details - see [20], section **3.12**).

The *SnapShot* memory shall have the capacity of one million words, which enables the memory to record the data in 294 consecutive LHC orbits (26.2ms).

The signal-sampling with the *SnapShot* memory shall have two modes of operation:

- in the *Sample After* mode, the sampling is "triggered" by a software command generated by the control processor and halted automatically when the memory capacity is exhausted;

- in the *Sample Before* mode, the signals are continuously sampled until the software command halts the operation.

In the *Sample After* mode, the memory contains data from the period following the software "trigger"; in the *Sample Before* mode, the data correspond to the interval preceding the software command.

The *SnapShot* memory shall be implemented on all the CTP boards. The list of signals to be recorded shall emerge during the detailed board design.

3.18 *ScopeProbe* option

The ScopeProbe option enables an easy oscilloscope access to a number of internal signals; the corresponding LEDs provide a quick visual check of the signal status. The option had been successfully used in the NA57 trigger electronics; it has been an invaluable tool for the development and debugging of the *LTU*. The circuit is described in [20], section **3.13**.

A configuration, identical to the one used on the *LTU*, shall be implemented on all the CTP boards. The two outputs from each board, *ScopeProbe* ***A*** and **B**, shall be connected via a multi-point LVDS backplane bus to the *INT* board. The oscilloscope connections and the LED indicators shall be mounted on the front panel of the *INT* board (**Figure 3.22.1**).

Note: Since the CTP electronics will not be accessible when the LHC beam is on, it might be considered, at a later date, to "extend" the oscilloscope connections with an "add on" optical link, in order to reach the control room.

A list of internal signals connected to the *ScopeProbe* outputs shall emerge during the detailed design of the CTP boards.

3.19 CTP control

The word “control”, in this context, denotes periodical measurement of the CTP boards’ temperature and the power supply voltages, and the transmission of the data to the ALICE DCS.

The CTP crate and the six LTC crates have a built-in interface (CAN bus) that is connected to the DCS in a standard way. The option monitors voltages and currents of the crate power supplies, the correct operation of the cooling fans, *etc*. .

On each CTP board, there shall be a temperature sensor, strategically located at a point where the highest temperature is expected, and a multi-channel ADC that measures the *crate supply voltages after the fuse* (+5V, +3.3V, *etc*.) and any *local supply* generated on the board (+1.5V FPGA core supply, *etc*.).

The electrical circuit shall be identical to the one in operation on the *LTU* board: the TC74A5-3.3VCT temperature sensor; the PCF8591 4-channel I^2C ADC and the P82B96 I^2C buffer. The I^2C part is powered by the +5V crate supply *before the fuse*.

Periodical reading of the board temperature shall be done by the CTP CPU and the data shall be transmitted to the DCS as a message of a standard format.

The I^2C connections from all the boards shall be “grouped” in 8 *branches*: six shall come from the six LTCs (4 *LTU*s each); one shall be common to all the 6 *FO* boards and the *INT* board; another brunch shall connect the remaining 4 CTP boards (**Figure 3.23.1**). The branches shall be multiplexed (I^2C multiplexer) to a single I^2C connection (2 twisted-pairs cable for SCL and SDT signals), and connected, via a simple adapter, to the parallel port of the control PC.

The control electronics of the *LTU* board and the interface to the PC have been successfully tested.

3.20 CTP FPGAs

Most of the logic on the CTP boards shall be based on FPGAs. This section deals with the FPGA selection and explains the way they shall be programmed/configured.

3.20.1 *FPGA selection*

On all the CTP boards, there shall be two different types of programmable logic devices, used in two different applications:

VME controller - ALTERA *EPM3512AFC256-7*

Main features:

- EEPROM based - once programmed, *ready to use* upon power-on;
- 512 logic elements;
- 7ns pin-to-pin propagation delay;
- 208 user I/O pins;
- 256-pin BGA package;
- JTAG compatible.

Board logic - ALTERA *Cyclone EP1C12F324C7*

Main features:

- SRAM based - needs to be configured upon power-on;
- 12000 logic elements;
- 52 4kbit RAM blocks
- 2 PLLs
- 249 user I/O pins;
- 324-pin BGA package;
- JTAG compatible.

The FPGA selection is a compromise that takes into account the application requirements, the number of logic elements/memory blocks/IO pins, the cost and availability, *etc.* . The two selected FPGAs have already been used on the *LTU* board; they have performed well, offered the required flexibility and provided sufficient spare capacity for possible future expansion. The Cyclone family is a relatively recent development- an important factor since the FPGAs tend to become obsolete rather quickly ("an FPGA year corresponds to 15 in human life...", P. Alfke, Xilinx, *LHC Electronics,* Stockholm, September 2001).

3.20.2 *FPGA programming and configuration*

The EEPROM based *VME controller* FPGA shall be programmed *in situe, via* the 10-pin *ByteBlaster* connector (JTAG), driven from a PC that runs the FPGA development software (ALTERA's *Quartus II*).

The SRAM based, *board logic* Cyclone FPGA needs to be configured upon power-on. The configuration data, the *Raw Binary File* (.rbf) generated by the FPGA development software, shall be stored, *via* the VME bus, in the on-board flash memory, and loaded into the FPGA either *automatically* - upon power-on, or *on request* from the CTP CPU.

Note: The capacity of the flash memory (Am29LV081B-70EC, 1Mx8) provides for a simultaneous storage of at least a couple of different configurations (283kB each); no application of this feature has been envisaged so far, but it might become a desirable option in the future.

3.21 The use of JTAG

The two types of FPGAs (section **3.20.1**) are the only JTAG compatible devices on the CTP boards. They shall be *both* connected to the same JTAG bus and the 10-pin JTAG (*ByteBlaster)* connector.

The JTAG connection shall be used to program the *VME controller* FPGA (section **3.20.2**). The Cyclone FPGA could also be programmed that way, but the option is of no practical value.

The JTAG connection, driven by the *JTAG Technologies* software package, shall be used to perform the very first test of the CTP boards, immediately after their production; the test shall verify the BGA joints, connections between the FPGAs, connections and the operation of a number of devices interfaced to the FPGAs. The technique has been successfully used in production of the *LTU* boards.

3.22 CTP partitioning and layout

Partitioning and the layout of the CTP system are shown in **Figure 3.22.1**.

Highlights:

- 6 types of boards: *L0***,** *L1***,** *L2***,** *FO***,** *BUSY, INT*;
- 11 slots;
- 30 backplane connections (bus-pairs).

BUSY Processor board (*BUSY*)

(See section **3.14**, ***Generation of the Cluster BUSY signals***.)

Main functions/logic blocks:

- Distribution, *via* the backplane, of the synchronization signals - the **BC** clock and the **Orbit,** to the other CTP boards.
- Programmable delay of the **BC** clock.
- Receiver of the individual sub-detector **BUSY** inputs (*via* the corresponding *LTU*).
- Generation of the *Cluster* ***BUSY*** signals.
- Generation of the *Cluster Dead Time* (**L0** to **L1)**.
- **BUSY** input timers.
- *Cluster* ***BUSY*** timers.
- **BUSY** input and *Cluster* ***BUSY*** monitoring (*ScopeProbe*).
- *SnapShot* memory.
- Monitoring of the board temperature and the supply voltages (I^2C bus).
- Terminating resistors (100Ω) for some LVDS differential signals.

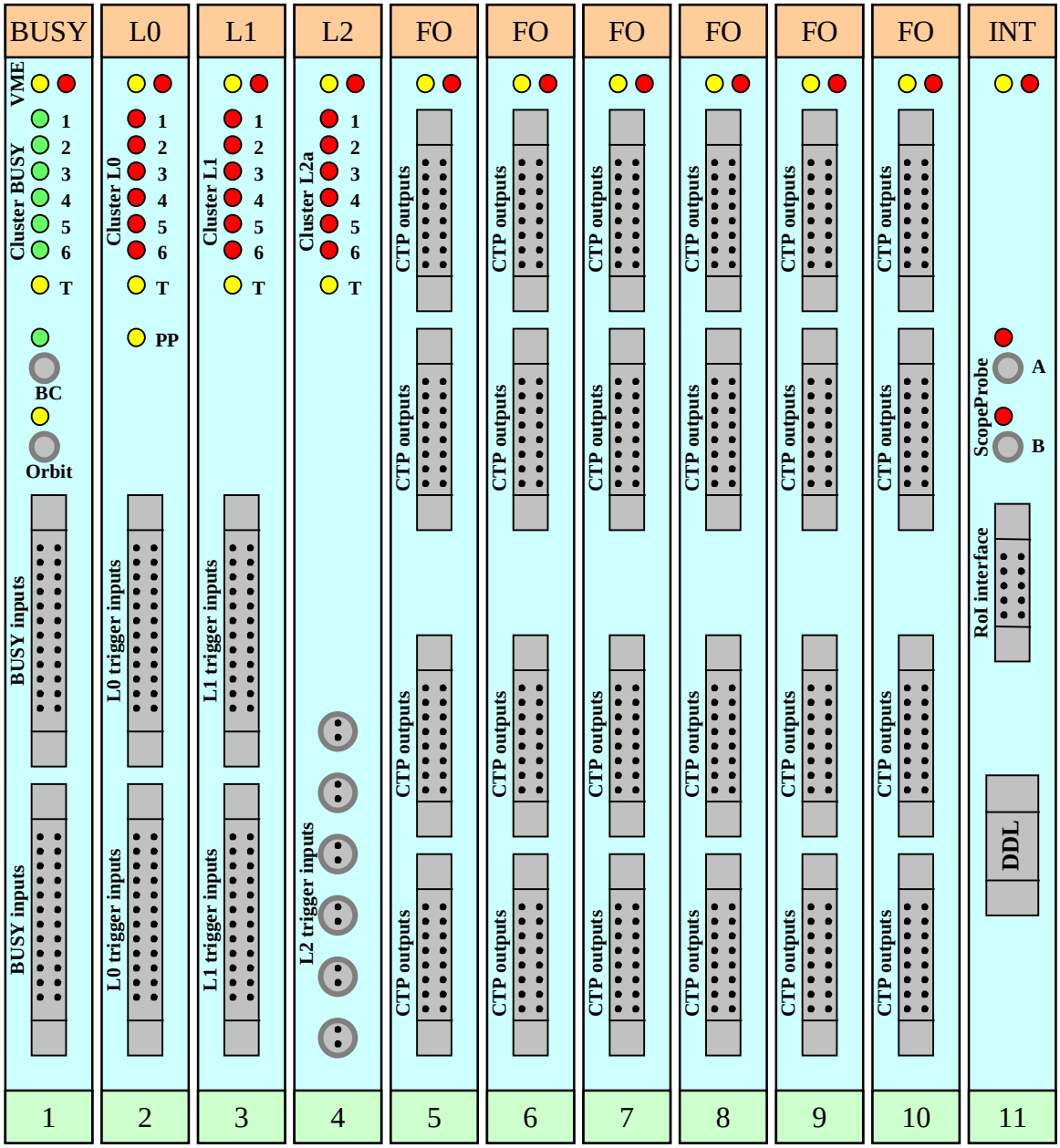


Figure 3.22.1 CTP partitioning and layout

Front panel connections/components:

- 2 LEMO 50Ω connectors (size 00, SCEM 09.46.11.180.6) for the **BC** and the **Orbit** inputs;
- Two 26-way IDC connectors (SCEM 09.55.05.726.2), 12 **BUSY** inputs (LVDS) and 2 ground returns each;
- LEDs.

Note : The advantage of having the **BC** receiver on the *BUSY* board (slot 1) is that the **BC** clock distribution coincides with the dominant signal-flow.

L0 Processor board (*L0*)

(See section **3.8.3**, ***Class L0 trigger logic***.)

Main functions/logic blocks:

- Receiver of the 24 ***L0*** *Trigger* inputs.
- Synchronization and alignment of the ***L0*** *Trigger* signals.
- *L0 Past-future Protection*.
- Generation of the *Class* ***L0*** *Triggers*, including pre-scaling.
- Generation of *periodic* (scaled-down **BC**) and *random* triggers.
- Generation of ***BC*** *Mask* signals.
- Generation of the *CTP Dead Time* (~2μs).
- Scaling of the *Class* ***L0*** *Triggers* before and after vetoes.
- Generation of the *Test Class* ***L0*** *Triggers*.
- *Class* ***L0*** to *Cluster* ***L0*** conversion.
- *Class* ***L0*** serialization.
- *Class/Cluster* ***L0*** *Trigger* monitoring (*ScopeProbe*).
- *SnapShot* memory.
- Generation, scaling and monitoring of the **Interaction** signals.
- Monitoring of the board temperature and the supply voltages (I^2C bus).
- Terminating resistors (100Ω) for some LVDS differential signals.

Front panel connections/components:

- Two 26-way IDC connectors (SCEM 09.55.05.726.2), 12 ***L0*** *Trigger* inputs (LVDS) and 2 ground returns each;
- LEDs.

L1 Processor board (*L1*)

(See section **3.8.4**, ***Class L1 and Class L2 logic***.)

Main functions/logic blocks:

- Receiver of the 20 ***L1*** *Trigger* inputs.
- Synchronization and alignment of the ***L1*** *Trigger* signals.
- Delay and de-serialization of the *Class* ***L0*** *Triggers*.
- *L1 Past-future Protection*.
- Generation of the *Class* ***L1*** *Triggers*.
- Scaling of the *Class* ***L1*** triggers before and after vetoes.
- Generation of the *Test Class* ***L1*** *Triggers*.
- *Class* ***L1*** to *Cluster* ***L1*** conversion.
- *Class* ***L1*** serialization.
- *Class/Cluster* ***L1*** *Trigger* monitoring (*ScopeProbe*).
- *SnapShot* memory.
- Generation of the **ESR** flag (RoI).
- Monitoring of the board temperature and the supply voltages (I^2C bus).
- Terminating resistors (100Ω) for some LVDS differential signals.

Front panel connections/components:

- Two 26-way IDC connectors (SCEM 09.55.05.726.2), 12 ***L1** Trigger* inputs (LVDS) and 2 ground returns each;
- LEDs.

L2 Processor board (*L2*)

(See section **3.8.4**, ***Class L1 and Class L2 logic***.)

Main functions/logic blocks:

- Receiver of the 6 ***L2** Trigger* inputs.
- Synchronization and alignment of the ***L2** Trigger* signals.
- Delay and de-serialization of the *Class **L1** Triggers*.
- *L2 Past-future Protection*.
- Generation of the *Class **L2** Triggers*.
- Scaling of the *Class **L2*** triggers before and after vetoes.
- Generation of the *Test Class **L2** Triggers*.
- Event identification logic (**BC** counter, **Orbit** counter).
- *Class **L2*** to *Cluster **L2*** conversion.
- *Class **L2**/Claster **L1**/Claster **L2**/Event Identifier* serialization.
- *Class/Cluster **L2** Trigger* monitoring (*ScopeProbe*).
- *SnapShot* memory.
- Monitoring of the board temperature and the supply voltages (I^2C bus).
- Terminating resistors (100Ω) for some LVDS differential signals.

Front panel connections/components:

- 6 LEMO 2-pin connectors (SCEM 09.31.28.070.5) for the 6 ***L2** Trigger* inputs (LVDS);
- LEDs.

Fan-out board (*FO*)

(See section **3.13**, ***Generation of the sub-detector signals***.)

Each board provides for *four* sub-detector connections; *six* identical boards are required for the 24 ALICE sub-detectors (including possible experiment expansions). Individual VME addresses are set with the on-board dial.

All the boards receive, *via* the backplane, identical set of signals; they are configured to recognize the data relevant to the sub-detectors to which they are allocated and to generate appropriate outputs.

Main functions/logic blocks:

- Transmission to sub-detectors of the **Orbit** signal.
- Generation and transmission of the sub-detector **Pre-pulse** signal, derived from the global, backplane **Pre-pulse**.

- Generation and transmission of the **L0** trigger, derived from *Cluster **L0*** signals.
- Generation and transmission of the **L1** trigger, derived from *Cluster **L1*** signals.
- Generation and transmission of the serialized **L1 Data,** derived from the serial *Class L1* data.
- Generation and transmission of the **L2 Strobe** and the serialized **L2** Data; the serial data includes the **L2arF** flag.
- Scaling of the **L0**, **L1**, **L2a** and **L2r** signals.
- Monitoring of the **L0**, **L1**, **L2a**/**L2r** triggers (ScopeProbe).
- *SnapShot* memory.
- Monitoring of the board temperature and the supply voltages (I^2C bus).

Front panel connections:

- four 16-way IDC connectors (SCEM 09.55.05.716.4) for the sub-detector/*LTU* outputs (7 LVDS links and 2 ground returns).

Interface board (*INT*)

Main functions/logic blocks:

- The *CTP Readout* logic.
- The *Interaction Record* logic.
- The *DDL Interface* - connection to the ALICE DAQ.
- Interface to the *RoI Processor*.
- A receiver of the **ScopeProbe A/B** inputs (backplane) and a driver of the monitoring outputs (front panel).
- Monitoring of the board temperature and the supply voltages (I^2C bus).
- Termination (100Ω) of some backplane LVDS signals.

Front panel connections/components:

- Standard ALICE DDL opto-connector;
- A 10-way IDC connector, the RoI interface, 4 LVDS outputs (2 signals, 2 spare connections) and 2 ground returns.
- 2 LEMO 50Ω connectors (size 00, SCEM 09.46.11.180.6) for the monitoring outputs (ScopeProbe option);
- LEDs.

Fan-in board (*FI*)

On the *L2* board, the six 2-pin connectors for the individual ***L2** Trigger* inputs fit easily on the front panel (**Figure 3.22.1**). There is not enough room, on the other hand, for all the 24 ***L0** Trigger* inputs (*L0* board); or the 20 ***L1** Trigger* inputs (*L1* board); or the 24 **BUSY** inputs (*BUSY* board). Instead, the individual connections shall be converted into a “group” connection that can be accommodated on the board’s front panel.

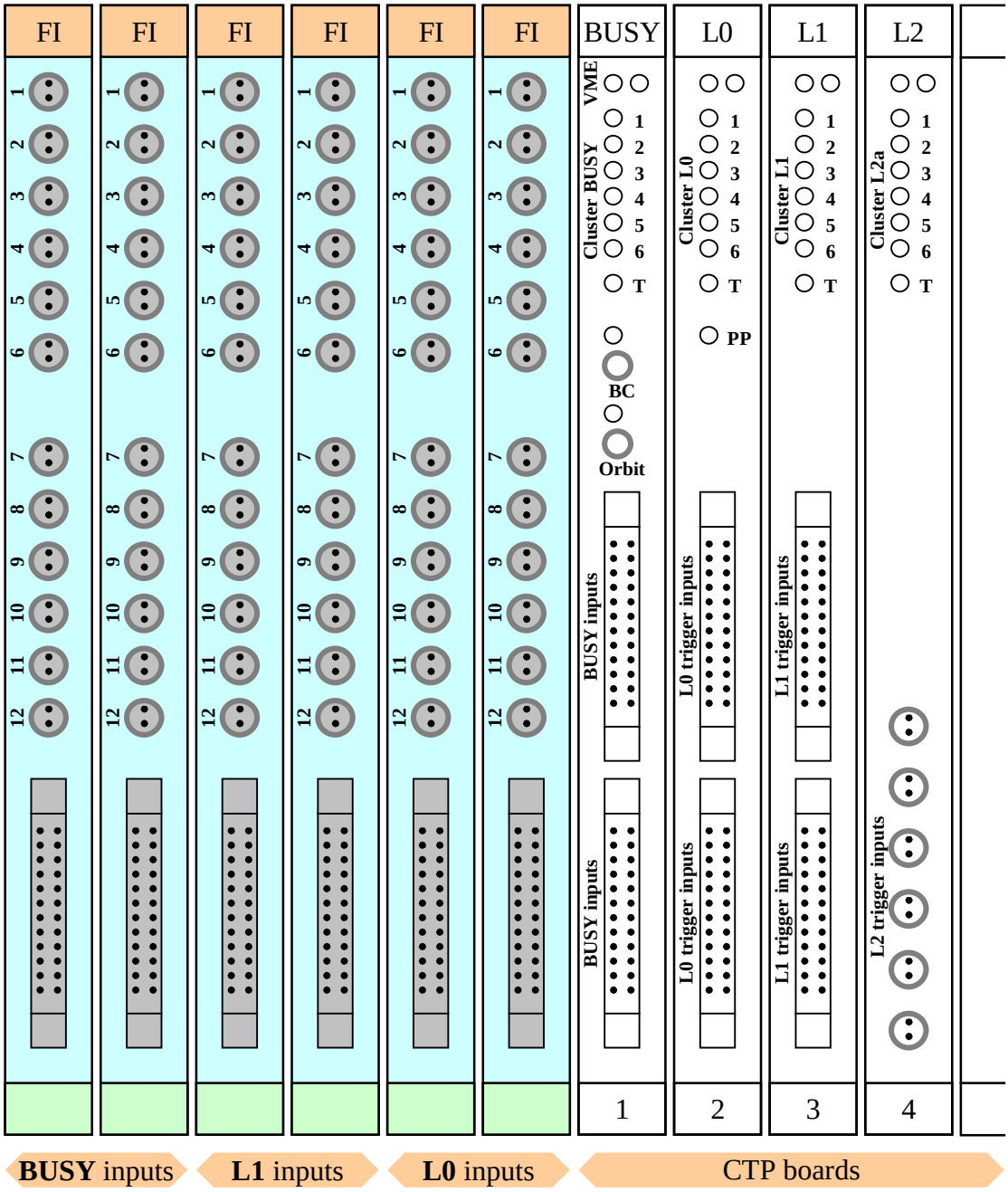


Figure 3.22.2 *Fan-in* boards

The *Fan-in* (*FI*) board is a *completely passive* "patch" from 12 individual 2-pin LVDS connectors (SCEM 09.31.28.070.5) to a single 26-pin IDC connector (SCEM 09.55.05.726.2); 24 pins are used for 12 LVDS links, 2 remaining pins serve as a common ground return. The board has a 6U form-factor, the same front panel as all the other CTP boards and shall be located in the same VME crate. No connection is made to the VME bus, the power supplies or the ground. The double-sided PCB shall have ground-return plane on one side and 12 differential pairs (Z_0=100Ω) on the other.

Two boards shall be used for the 24 ***L0*** *Trigger* inputs; another two for the 20 ***L1*** *Trigger* inputs; and two more for the 24 **BUSY** inputs (**Figure 3.22.2**). The *FI* boards shall be located in the slots preceding the *INT* board and connected to their corresponding CTP boards with short (20-30cm) twisted-pair flat cables.

3.23 The CTP backplane

The CTP backplane connections are shown in **Figure 3.23.1**.

All the 64 VME J2 *user-defined pins* are used for the inter-board connections. Apart for the I^2C control bus lines, all the other connections are LVDS pairs. All the LVDS termination resistors (100Ω) are mounted on the CTP boards.

The 2 pairs of lines allocated to the I^2C control bus are the clock (SCL) and the serial data (SDT) signals, both with the associated ground connection. The bus is split into two branches: one for the 6 *FO* boards; the other for the rest.

The numbers 1 to 32 in the figure only "list" the available *connection pairs*. Although the pin allocations shall only emerge during the detailed design, the layout is close to the final version. The LVDS links (connection pairs) are organized as bus lines, associated with the same connector pins on *all the boards*. Some of the bus connections are *point-to-point*; most are *multi-drop*; only the ScopeProbe lines are *multi-point* links. On the boards, the pins are either *used* or *reserved* (no connection); the J2 *user-defined pins* that are neither *used* nor *reserved* are connected to ground.

Those "layout rules" enable two implementations of the backplane:

- The "*wired*" backplane would use DIN-41612 IDC connectors (64 pin, a+c) and appropriately cut flat cable. The same technique has been successfully applied in the NA57 project (**Figure 3.23.2**), be it at a smaller scale.

- The "*printed*" version would also use the 64-pin (a+c) DIN-41612 connectors, but the connections would be differential pairs with the characteristic impedance Z_0 = 100Ω, *printed* on one side of a double-sided kapton tape, while the other side would serve as a continuous ground plane, connected to all the grounded pins (neither *used* nor *reserved*- see above) of all the CTP boards. The appropriate connectors for the two I^2C control buses would be soldered to the backplane.

  Note 1: The kapton backplane needs to be "rigidized" in the area of the connector soldering.

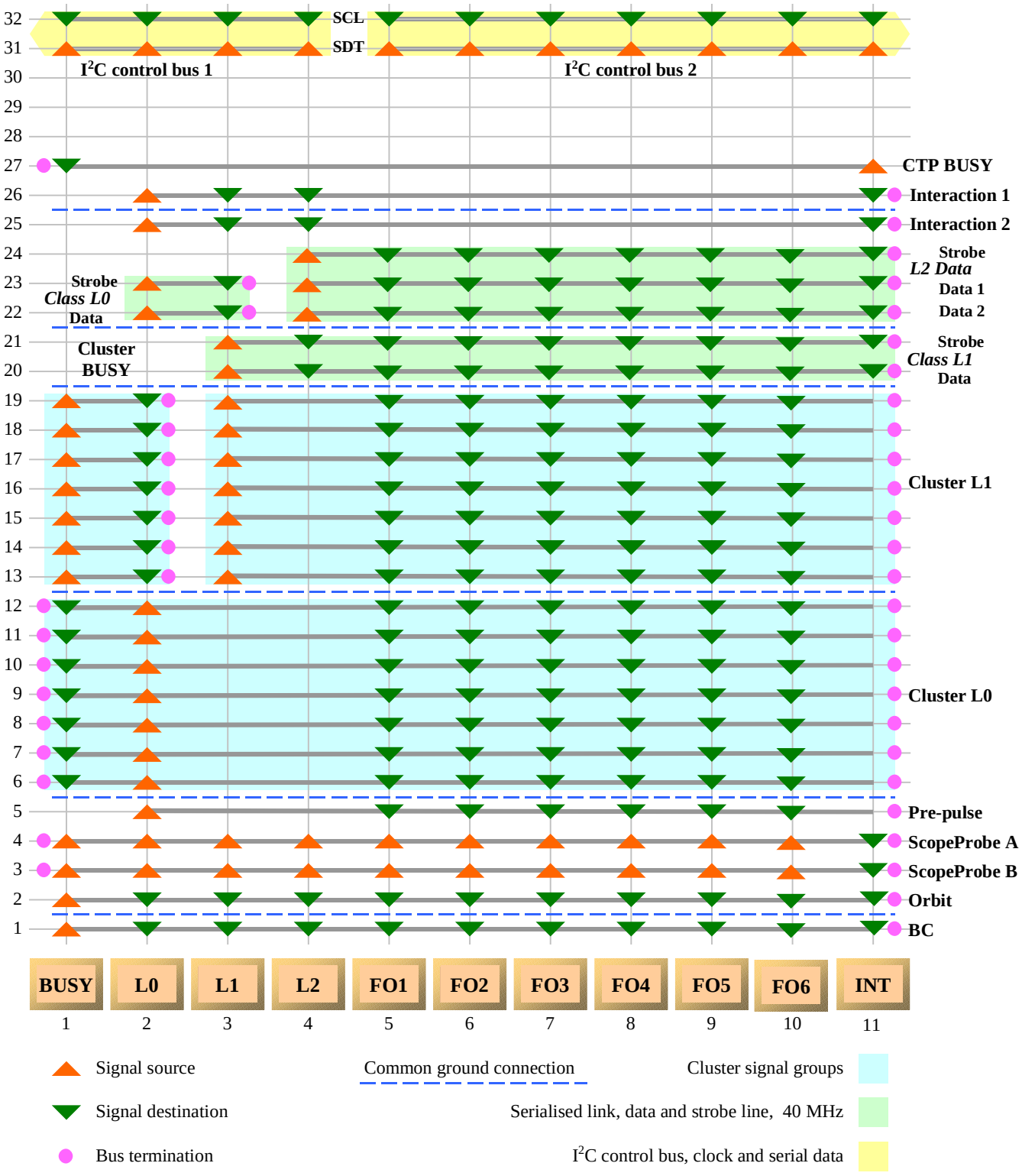


Figure 3.23.1 CTP backplane connections

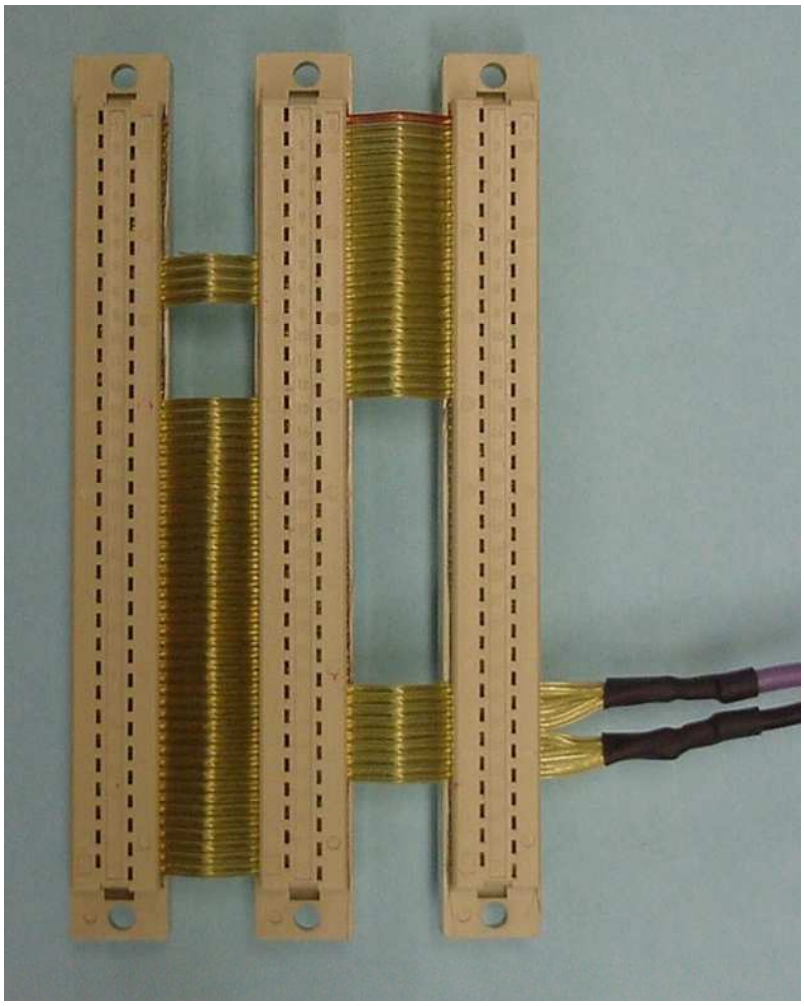


Figure 3.23.2 "Wired" backplane for the NA57 experiment - an example

In order to avoid the cross-talk, the differential LVDS lines are, in both versions, located sufficiently away (>12mm) from the TTL signals on the VME backplane; in case of the *printed* backplane, the ground layer provides the additional shielding.

During the development, the *wired* version will be used. If necessary, the *printed* backplane will be produced for the final system.

Note 2: The use of the **Orbit** signal:

- on the *L0* board, it is used to synchronize the counter that operates the ***BC** Mask*;
- on the *L2* board, the **Orbit** counter is part of the event identification logic;
- on the *INT* board, the **Orbit** is used for the generation of the *Interaction Record*;
- on *all the boards*, the signal serves as a time reference for the alignment of the *SnapShot* memory data.

3.24 Board identification

All the CTP boards shall have the following identification data set by the board hardware/firmware.

Board type code

Internal address on all the boards: 1

The 8-bit code, defines the *board type*:

Board	Code [Hex]
L0	50
L1	51
L2	52
F0	53
BUSY	54
INT	55
LTU	56

Board serial number

Internal address: 2

Among the boards of the same type, each board shall have a unique *serial number*.

CPLD firmware version

Internal address: 3

On each board, a CPLD serves as a VME interface controller. The *version of the firmware* used to program the CPLD is identified with an *8-bit code*.

FPGA firmware version

Internal address (hex): 20, 21, 22, ...

The FPGAs provide most of the logic on the CTP boards. The *version of the firmware* used to configure the FPGA is identified with an *8-bit code*.