Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Master's Thesis Nr. 510

Systems Group, Department of Computer Science, ETH Zurich

in collaboration with

CERN

A Very Cool Master Thesis

by

Paolo Rondot

Supervised by

Prof. Alonso Gustavo

April 2024–September 2024

D INFK

Acknowledgements

Abstract

Contents

Chapter 1

Introduction

Chapter 2

Background

2.1 CERN

2.1.1 ATLAS Experiment

2.1.2 Liquid Argon Calorimeter

2.2 High Level Synthesis

Chapter 3

Latome Hardware Design

3.1 Specifications

3.1.1 The Liquid Argon Calorimeter Data Path

The LAr Calorimeter is made of 180,000 cells which are then summed up to form 34,048 supercells. The 12-bit readings of 8 supercells are serialized into one packet and sent through one optic fiber. The 116 LATOME boards are each connected to 48 optic fibers, thus receiving data from $48 \times 8 = 384$ supercells. These boards are responsible for transforming the ADC levels into energy levels, reorganizing the data and summing some energies for LATOMEs covering specific regions of the detector and distributing the data to the Feature Extraction (FEX) system. The FEX system is responsible for processing the data and sending it to the Level 1 Trigger system.

The FEX system is made of 3 groups: the Global Feature Extractor (gFEX), the Jet Feature Extractor (jFEX) and the Electron Feature Extractor (eFEX). Within these groups, gFEX only contains one board which gets summed data from all LATOMEs, thus showing the lowest granularity but with one board covering the whole detector. Then the jFEX is made of 6 boards, each receiving data from 19 LATOMEs, and finally the eFEX, with the highest granularity, is made of 24 boards, each receiving data from 4 LATOMEs.

3.1.2 The LATOME Specifications

The part of the LATOME firmware responsible for the conversion of ADC levels into energy levels is referred to as *User Code* in the LAr group. This code is not owned by the LATOME HLS team. However, the team is responsible for the different data organization steps, also called mapping or remapping, and the summing of energies for LATOMEs in specific regions of the detector. Additionally, since the output energies are only encoded on 10 bits, it is necessary to compress the data from 12 to 10 bits via a Multi Linear Encoder (MLE).

Initially, the whole LATOME firmware was written in VHDL. The implementation of all the functionalities took over 8 years and still had timing violations.

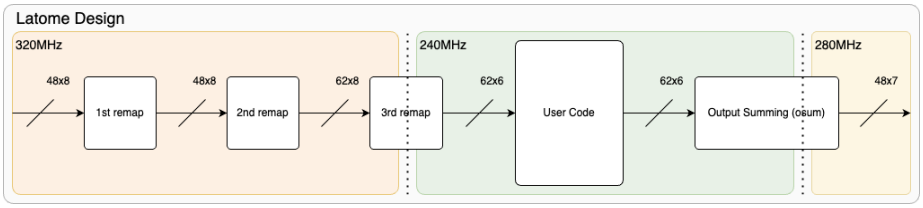

Figure 3.1: Original VHDL Design

Because in the next runs of the LHC, the amount of energy and collision is expected to increase, the LATOME design should be optimized to leave space for other functionalities, and sustain the increase in data.

3.2 Existing Designs

3.2.1 Original VHDL Design

The original design is characterized by more clock domain crossings than the current one.

Because bunch crossings (BC) happen at a frequency of 40MHz, it is necessary to run the different blocks of the design at multiples of this frequency. The figure5.1 shows three different frequencies being used. At the first remap stage, since the frames are made of 8 words, it is necessary to run the blocks at 320MHz. Then the frames become 6 words long, corresponding to a frequency of 240MHz. Finally, the FEX systems expect 7 32 bits words, thus the frequency is 280MHz.

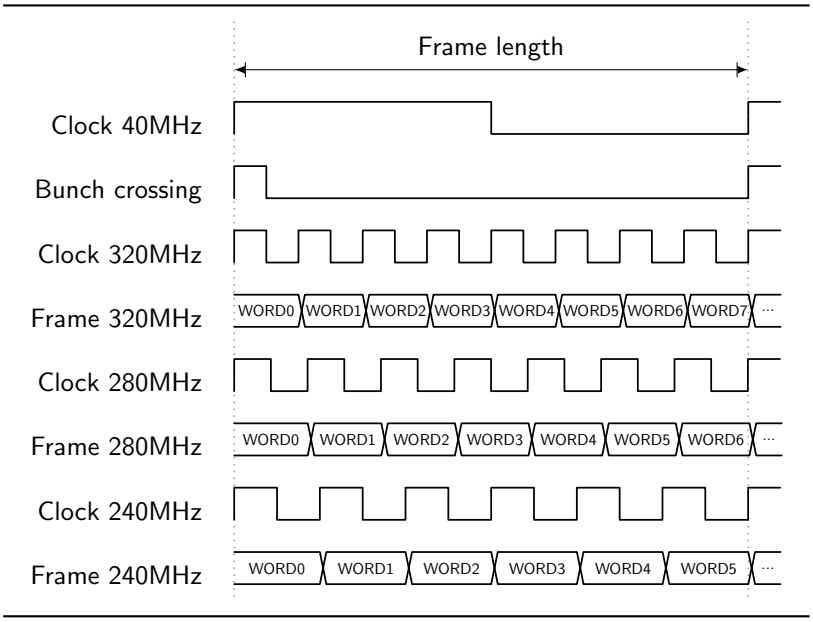

Figure 3.2: Different clocks used in the design

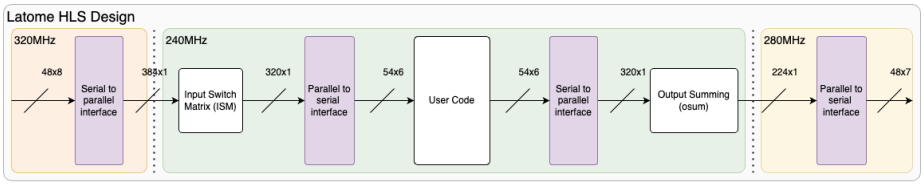

Figure 3.3: Current HLS Design

These clock domain crossings made the original implementation very complex...

3.2.2 Current HLS Design

When developing the new firmware from the specifications in HLS, very interesting insights from Dr. Marcos Oliveira emphasizing benefits of space division multiplexing were taken into account. By using parallel full frames, the frequency of the design blocks is not locked anymore depending on the frame size. This allows to reduce the logic dedicated only to synchronization and time-division multiplexing which can become very complex when there are dependencies between the words of the frame.

The figure 3.3 shows that all the remapping and processing blocks except for the User Code are being processed using parallel data. This is possible thanks to the serial-to-parallel and parallel-to-serial data transformation blocks. This approach drastically simplifies the design and showed very promising results in terms of area reduction, reduced latency.

Chapter 4

Leveraging Time vs Space Division Multiplexing

When a chunk of data is available at a given time for processing, it can all be processed at the same time, or pieces after pieces. Processing simultaneously all the data offers the advantage of a reduced latency, but requires creating copies of the processing blocks for each piece. A sequential processing gives the possibility of having just one processing block treating the data step by step at a price of latency.

A simple parallel with human body is our hands. When writing a text, because of our small number of hands we are condemned to write texts one letter after another. But if we imagined having 1000 hands, and enough space to hold as many pen without interfering with one another, we could imagine writing whole paragraphs at once.

4.1 Time Division Multiplexing

The time division multiplexing approach consists in sharing the resources, or processing blocks, by routing each piece of data depending on time t. In the figure 4.1 this refers to the lower block diagram. Choosing this implementation can be intuitive as computers usually process data using frames of 32 to 64 bits. It has the advantage of reusing design blocks hence possibly reducing area usage, but performs poorly in case of data dependencies. If the word from time $t = 0$ of the frame needs to be available at the same time as the one of $t = 4$, the logic becomes very complex thus canceling the positive area impact unlocked by resource sharing.

This sequential processing approach requires counters used by multiplexers to latch the input data at the right moments.

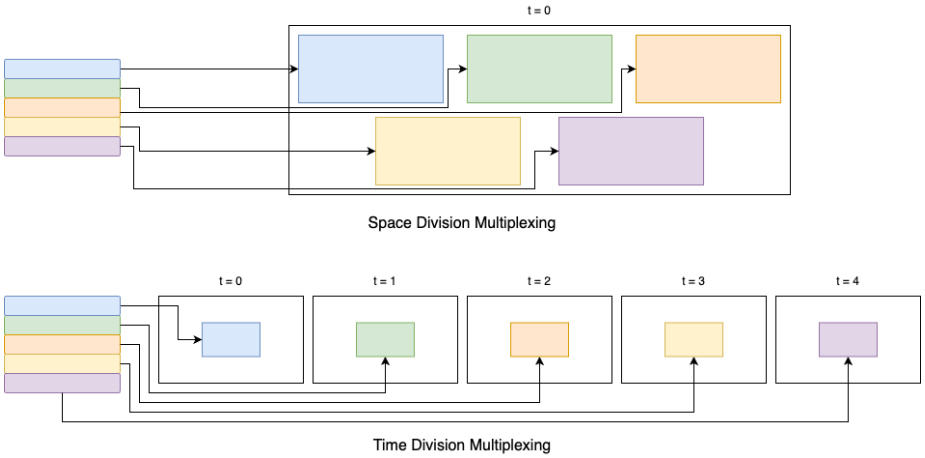

Figure 4.1: Simple Representation of Time and Space Division Multiplexing

4.2 Space Division Multiplexing

Another way of treating the data is to make it all available at the same time, as parallel inputs of the processing block. This approach is more natural as it is closer to what humans experience; we see, hear, feel, taste at the same time, not one after the other. In figure 4.1 it is represented on top. Space Division Multiplexing generally offers small latencies, as every block runs in parallel and reduces the use of sequential logic which would be required in TDM. In fact, using a parallel logic, some processing blocks can become purely combinational ones.

4.3 Tradeoffs

Whether one approach is better than the other is impossible to determine prior to having a deep understanding of the data processing blocks. The most common tradeoff is the one of latency at the price of area. But the Maximal Operating Frequency (F_{max}) can also be traded to improve other metrics.

Having access to all variables at the same time makes the code development easier.

4.4 Leveraging Both Approaches for LATOME

In the version 6 firmware the choice was made to use parallel data as much as possible as it showed promising preliminary results. The input and output interfaces of both ISM and OSUM are indeed fully parallel.

The Output Summing represented in figure 4.2 is made of the following blocks:

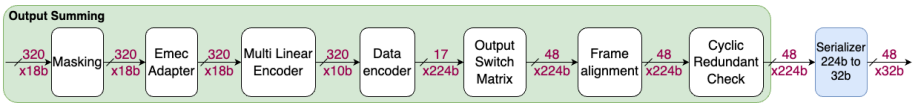

Figure 4.2: Output Summing Block Diagram

- Masking: responsible for disabling some inputs depending on the detector region that a specific LATOME treats
- Emec Adapter: this block performs additional summing for the LATOME responsible for the EMEC region of the detector
- Multi Linear Encoder (MLE): Since the ADC readings are 12 bits long but the FEX systems requires 10 bits long energy levels, this block encodes the ADC readings on 10 bits
- Data Encoder: Once the encoded energy levels are ready, this block creates 17 frames of 224 bits representing the whole data
- Output Switch Matrix (OSM): This switch matrix routes the 17 224 bits frames to the different FEX systems that the LATOME is connected too
- Frame Alignment: Each couple of BC, the LATOME does not send any data but instead it sends an alignment frame built by this block
- CRC9: This final block appends to the last 9 bits of each frame a CRC9 value which it computes

The OSM, which is further described in Section 5.1, takes large inputs of 224 bits. When working with such data widths, different point of optimization arise. The data synchronization becomes much more complex as 224 signals simultaneous signals must be processed. Consequently, such implementations can block the F_{max} to low values. Then, the OSM uses multiplexers to route the data to the FEX systems. It does so by using multiplexers which can become very big if they use multiple inputs of 224 bits. This case shows an example where TDM could be used to reduce the data width by using a serial approach where smaller pieces of the data are processed sequentially.

Chapter 5

Implementation

5.1 Output Switch Matrix

5.1.1 Description

The different boards must route the data differently to the FEX systems depending on the detector region they are connected to. To tackle this problem, either each LATOME has a different version of the firmware, or some code must be added to allow an online configuration procedure defining the routing. The second possibility was chosen in the LAr group, and the solution proposed by Marcos for the V6 firmware was to add an Output Switch Matrix.

The OSM uses multiplexers to route the 17 inputs to the 48 outputs and takes its selection bits from registers configured online via an ipbus. The HLS implementation is very simple, mainly involving a 2D array with 48 rows made of a subgroup of the 17 possible inputs.

Without looking at the different LATOME routings, it appears that, there should be 48 multiplexers of input size $224 \times 17 = 3808$. However when considering the data paths and optimizing for them, the biggest multiplexers have at most 4 input frames, meaning that each row from the HLS 2D array have 4 entries. Figure 4.2 shows the parallel to serial block creating 32 bits words for the FEX systems. Moving this block to the left would have no impact on the latency. However, by moving this block before the Output Switch Matrix, the biggest multiplexers would have an input size of 4×32 instead of 4×224.

An important aspect to monitor is the data dependencies, since these could have a critical impact if using TDM. The OSM block is a routing block which does not change any data in the frames. The Frame Alignment only overwrites the data that it gets as input if the current BC index is an alignment one, hence it does not show data dependencies. Finally the Cyclic Redundant Check can be both computed in a parallel or serial fashion equivalently and only depends on the CRC temporary result from time $t = -1$.

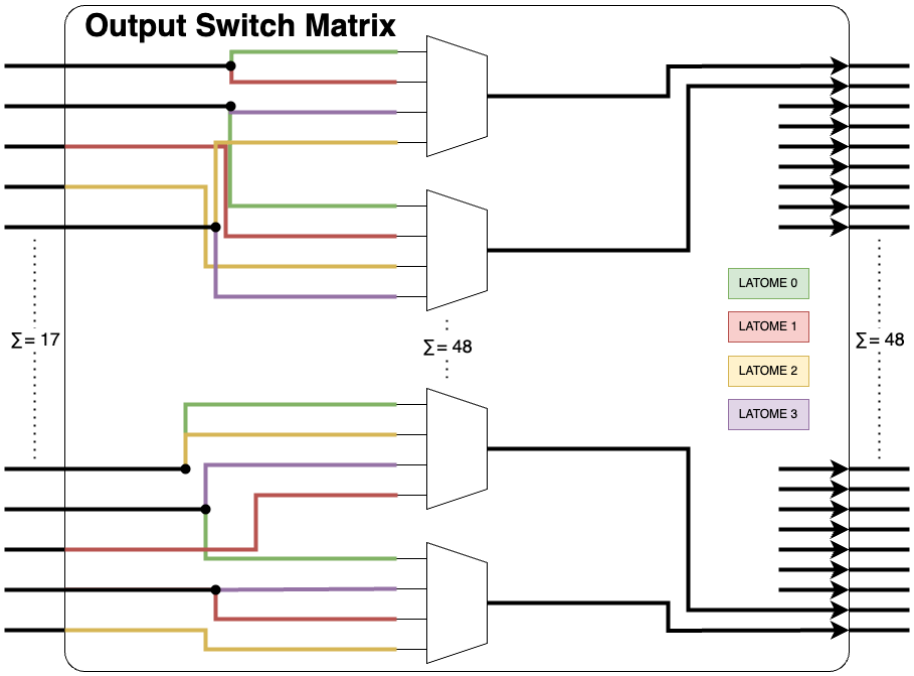

Figure 5.1: Output Switch Matrix Simplified Diagram

5.1.2 Serializing Before the OSM

Serializing the 224-bit full frames to 32-bit words requires 7 clock cycles, as $224/7 = 32$. Hence the operating frequency of the OSUM block must be $7 \times 40 = 280MHz$.

5.2 Optimizing the HLS Directives

Chapter 6

Evaluation

Chapter 7

Conclusion