DOI: 10.48011/asba.v2i1.1674

Arquitetura Multi-Core de Processadores Reconfiguráveis para Reconstrução Online de Energia no Calorímetro Hadrônico do ATLAS

Melissa Santos Aguiar [*] **Mariana de Oliveira Resende** [*]
Lucca Oliveira Facio Viccini [*] **Kinn Dias** [*] **Tiago Teixeira** [*]
Luciano Manhães de Andrade Filho [*]
José Manoel de Seixas [**]

[*] *Faculdade de Engenharia Elétrica, Universidade Federal de Juiz de Fora, MG, (e-mails: melissa.aguiar@engenharia.ufjf.br, mariana.resende@engenharia.ufjf.br, lucca.viccini@engenharia.ufjf.br, kinn.dias@engenharia.ufjf.br, tiago.teixeira@engenharia.ufjf.br, luciano.andrade@engenharia.ufjf.br).*
[**] *Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia, COPPE/UFRJ, RJ (e-mail: seixasjmd@gmail.com)*

Abstract: Calorimeters are detectors aiming at measuring the energy of fundamental particles that go through their material. For such purpose, electrical pulses are generated by sensors when they interact with said particles. Digital signal processing techniques are applied for pulse detection and estimation in order to infer the energy of the primary particle. Such techniques, when implemented online, need to be of low complexity, making the implementation in dedicated hardware possible. Nevertheless, techniques based on the Sparse Representation (SR) theory have emerged addresing reconstruction efficiency improvement. However, due to their high computational cost, they are not utilized as an option for online processing. In this paper, the customization of a processor and its use in a multi-core architecture in FPGA is presented, enabling the online implementation of techniques based on SR. In order to demonstrate how it operates, the hadronic calorimeter from the ATLAS Experiment was used as the application environment.

Resumo: Calorímetros são sistemas usados para medir a energia de partículas fundamentais que atravessam o seu material. Para tal, pulsos elétricos são gerados por sensores posicionados ao longo do material absorvedor do calorímetro. Técnicas de processamento digital de sinais são empregadas para detectar e estimar parâmetros destes pulsos de forma a inferir a energia das partículas. Tais técnicas, quando implementadas online, necessitam ser de baixa complexidade para que seja possível a sua implementação em *hardware* dedicado. No entanto, técnicas baseadas em teoria de Representação Esparsa (RE) de dados vêm se destacando quanto à eficiência de reconstrução, mas, devido ao seu alto custo computacional, ainda não são utilizadas como uma opção para processamento online. Neste trabalho, é apresentada a customização de um processador e sua utilização em uma arquitetura *multi-core* em FPGA, possibilitando a implementação online de técnicas baseadas em RE. Para demostrar seu funcionamento, foi utilizado o calorímetro hadrônico do Experimento ATLAS como ambiente de aplicação.

Keywords: processors; FPGA; multi-core; reconfigurable hardware; energy reconstruction.
Palavras-chaves: processadores; FPGA; multi-core; hardware reconfigurável; reconstrução de energia.

1. INTRODUÇÃO

A Física de Altas Energias é essencial na exploração de questões fundamentais como as partículas básicas que constituem a matéria e suas dinâmicas de interação (Perkins, 2000). Para tal, são realizados experimentos envolvendo um refinado sistema de instrumentação: os colisionadores de partículas. Estes experimentos consistem em colidir feixes de partículas sub-atômicas que são acelerados a velocidades próximas a da luz, promovendo ambientes que se assemelham ao estado do universo nos instantes iniciais após o *Big Bang* (Wille, 2000).

Nesse contexto, a Organização Europeia para a Pesquisa Nuclear (*Conseil Européen pour la Recherche Nucléaire*), também conhecida como CERN, fundada no ano de 1954, em Genebra, Suíça, vem se destacando mundialmente nos experimentos envolvendo a Física de Altas Energias (Anthony, 2014).

O LHC (*Large Hadron Collider*) é atualmente o principal acelerador de partículas do CERN, sendo considerado o maior e mais energético acelerador de partículas já construído (CERN, 2014). Ele é composto por um anel de 27 km de circunferência, que está a 100 m de profundidade na fronteira entre a França e a Suíça. O seu processo consiste na colisão entre dois feixes de prótons a uma taxa constante de 40 MHz no ponto de interesse dos detectores, que são instrumentos responsáveis pela observação e análise das partículas fundamentais oriundas destas colisões (Aad, 2012).

O ATLAS, ilustrado na Figura 1, é um dos principais experimentos responsáveis pela aquisição dos dados resultantes das colisões no LHC. Ele pesa cerca de 7000 toneladas e suas dimensões são de cerca de 25 metros de altura por 44 metros de largura, sendo composto por sub-dectores dispostos em camadas, onde cada sub-detector é responsável por medir propriedades específicas das partículas geradas pelas colisões.

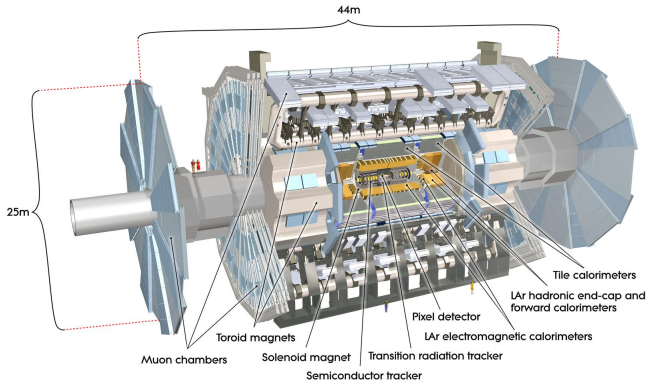

Figura 1. Detector ATLAS e seus sub-sistemas. Extraído de Pequenao (2008).

A energia das partículas geradas é uma importante característica a ser analisada, sendo obtida através dos calorímetros (Wigmans, 2018), que normalmente são segmentados espacialmente em milhares de células, as quais correspondem a canais de leitura. Esses canais contam com uma eletrônica dedicada para processar os dados provenientes das colisões, que acontecem a cada 25 nano segundos (ns), o que demanda um complexo sistema de instrumentação em *hardware*.

O sistema de calorimetria do ATLAS é composto pelo Calorímetro de Argônio Líquido (*LAr - Liquid Argon*) e pelo Calorímetro de Telhas (*TileCal - Tile Calorimeter*). O Calorímetro de Argônio Líquido, também conhecido como Calorímetro Eletromagnético, foi projetado para absorver as partículas que interagem de forma eletromagnética com seu material (Aleksa et al., 2013). Já o Calorímetro de Telhas, que é também chamado de Calorímetro Hadrônico, está localizado em uma camada externa à do LAr e foi projetado para absorver as partículas mais propícias a interagirem de forma hadrônica (Carrio et al., 2013).

Para garantir a correta reconstrução dos eventos de colisão, é necessário um sofisticado sistema de processamento. Tal sistema pode ser subdividido entre dois tipos de processamento: o *online* e o *offline* (Nakahama, 2015). O processamento *online* é responsável por realizar a seleção dos eventos de interesse na medida em que as colisões ocorrem, respeitando a latência do sistema. Já o processamento *offline* faz a análise dos dados armazenados após a colisão (Filho et al., 2015).

Atualizações graduais estão previstas para ocorrer no LHC nos próximos anos, de forma a elevar a densidade de feixes de prótons do acelerador (Cepeda et al., 2018). Com isso, a taxa de interações entre as partículas e a quantidade de partículas elementares detectadas cresce, aumentando também a probabilidade da ocorrência de fenômenos raros, de interesse nos estudos atuais em física fundamental.

Nos calorímetros do Experimento ATLAS, devido ao fato de sua resposta (largura dos pulsos) ser mais lenta do que a taxa de colisão, o aumento na taxa de eventos aumenta a probabilidade de ocorrência de sobreposição entre sinais provenientes de eventos subsequentes, dificultando o processo de reconstrução da energia a partir da medida de parâmetros dos pulsos. Tal efeito é conhecido como empilhamento de sinais ou *pileup* (Cleland and Stern, 1994), o qual é ilustrado na Figura 2, para pulsos do calorímetro hadrônico (largura de 150 ns). Neste exemplo, os parâmetros do pulso central de interesse são mascarados pela presença de um pulso lateral, resultando em um sinal completamente diferente do esperado pela eletrônica de reconstrução.

Devido ao problema da crescente incidência de *pileup* no LHC, técnicas baseadas em deconvolução de sinais vêm sendo propostas e têm apresentado desempenhos superiores às técnicas de estimação de parâmetros adotadas anteriormente, como visto nos trabalhos Peralva et al. (2013), Barbosa et al. (2017) e Duarte et al. (2019). Dentre estas novas propostas, a deconvolução *online* vêm sendo proposta através da utilização de filtros digitais de equalização de canais na forma de estruturas FIR (*Finite Impulse Response*), similares às empregadas em comunicação digital. No entanto, trabalhos recentes têm demonstrado a superioridade na utilização de técnicas de Representação Esparsa (RE) de Dados, devido à característica impulsiva dos dados a serem reconstruídos, como em Teixeira et al. (2018), Teixeira et al. (2019) e Aguiar et al. (2019b).

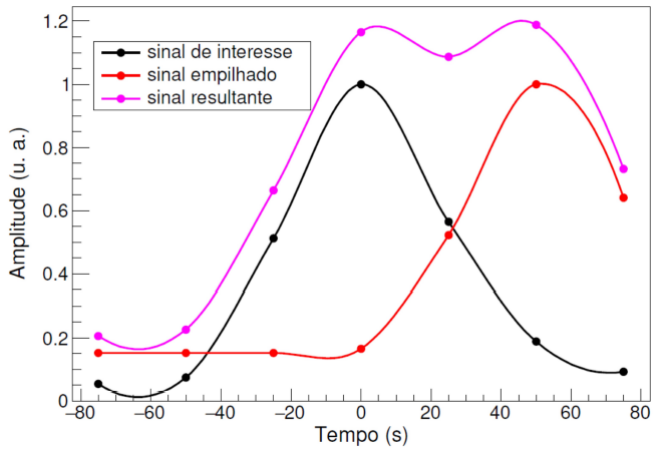

Figura 2. Efeito *pileup* no TileCal (Filho et al., 2015).

Apesar do melhor desempenho das técnicas de RE, sua implementação *online* ainda é um desafio, sendo esta, até então, proposta apenas para ambiente *offline*. O principal objetivo deste trabalho é realizar a implementação, em *hardware*, de um método iterativo de RE visando a reconstrução *online* de energia em Calorímetros de Altas Energias, tendo como foco o Calorímetro Hadrônico do Experimento ATLAS, para os cenários em que ocorre o empilhamento de sinais.

Os trabalhos Aguiar et al. (2019a) e Aguiar et al. (2019b) evidenciam o potencial de um processador dedicado na implementação de métodos iterativos de RE. Porém, tais trabalhos concluem que um único núcleo deste processador ainda não seria capaz de operar em frequências factíveis, sem que ocorra perda de dados. Nesse contexto, o trabalho aqui apresentado propõe uma estrutura com vários núcleos (*multi-core*) de processamento que permite a manipulação do fluxo contínuo e sequencial de dados sem perda de informação, garantindo a implementação *online* de técnicas de RE. Além disso, foram propostas modificações e aprimoramentos no processador dedicado, visando otimizar este tipo de implementação.

O trabalho é organizado da seguinte forma: na Seção 2 é descrito o embasamento matemático do método de reconstrução de energia estudado. Na Seção 3 o ambiente de desenvolvimento do processador embarcado é apresentado. Na Seção 4 é descrita a implementação da estrutura *multi-core* proposta, além das otimizações realizadas no processador. Na Seção 5 os resultados obtidos são discutidos e, por fim, na Seção 6, a conclusão do trabalho é apresentada.

2. DECONVOLUÇÃO POR REPRESENTAÇÃO ESPARSA DE DADOS

Em Peralva (2012) podemos encontrar o modelo convolucional para o fenômeno de *pileup* cuja proposta é baseada em um dado pulso de referência normalizado do calorímetro (amplitude unitária), representado pelo vetor $\mathbf{h}$, uma sequência vetorial $\mathbf{x}$, que representa os valores de energia a serem reconstruídos, por *bunch*, e, por fim, o vetor $\mathbf{r}$, que é a convolução entre estes dois sinais, representando o sinal de leitura da eletrônica de *front-end* do TileCal. Essa amostragem tem seu *clock* sincronizado com a taxa de colisões do LHC.

Como nesse trabalho é feito o uso de uma das características do sinal gerado pelo LHC, ou seja, o janelamento apontado em Herr (2005), utilizou-se a deconvolução iterativa proposta em Duarte (2015) e, para tal, uma formulação do processo de convolução em um formato matricial, conforme apontada na Equação 1, é mais conveniente.

$$\mathbf{r} = \mathbf{H}\mathbf{x} \tag{1}$$

sendo a matriz $\mathbf{H}$, chamada de Matriz de Convolução, formada pelas amostras de referência do sinal $\mathbf{h}$ normalizado e deslocado a cada coluna. Tais valores se repetem em uma diagonal, ficando os demais valores da matriz iguais a zero.

Quando $\mathbf{r}$ e $\mathbf{H}$ são conhecidos (caso em questão), o problema recai em um sistema de equações lineares para a reconstrução da sequência $\mathbf{x}$. No modelo convolucional, o vetor $\mathbf{r}$ é maior que o vetor $\mathbf{x}$. Portanto, há infinitas soluções para este sistema de equações. Conforme demonstrado em Barbosa (2012), na deconvolução de sinais impulsivos, a melhor escolha tende para uma representação mais esparsa de $\mathbf{x}$.

Podemos encontrar em Elad (2010) vários métodos de resolução de sistemas esparsos, porém, para este trabalho, foi considerada uma versão adaptada do método SSF (*Separable Surrogate Functionals*), tal versão foi extraída dos trabalhos de Teixeira et al. (2018) e Teixeira et al. (2019). A Equação 2 representa essa adaptação em relação a equação originalmente encontrada em Elad (2010).

$$\mathbf{x}_{k+1} = S_\lambda \left(\mathbf{x}_k + \mu \left[\underbrace{\mathbf{H}^T\mathbf{r}}_{\text{constante}} - \underbrace{\mathbf{A}\mathbf{x}_k}_{\substack{\text{varia em} \\ \text{cada iteração}}} \right] \right) \tag{2}$$

onde a parte constante da Equação 2 ($\mathbf{H}^T\mathbf{r}$) pode ser calculada uma única vez na inicialização do método, enquanto a parte variável da Equação 2 ($\mathbf{A}\mathbf{x}_k$) é recalculada a cada iteração. Nessa equação, a matriz $\mathbf{A}$ é o resultado da multiplicação da matriz $\mathbf{H}$ por sua transposta ($\mathbf{A} = \mathbf{H}^T\mathbf{H}$).

A Figura 3 representa a diagramação da Equação 2, conforme implementada nesse trabalho.

Ainda em relação a equação original do SSF disponível em Elad (2010), a Equação 2 de Teixeira et al. (2018) possui os parâmetros μ e λ calibrados a partir de simulações e pré-fixados. Com isso, apenas operações de somas, subtrações e multiplicação precisam ser implementadas em *hardware*, facilitando assim implementações como a ocorrida em Aguiar et al. (2019b).

3. O PROCESSADOR SAPHO

Para a implementação do método iterativo de deconvolução foi utilizado o Processador *Soft-Core* (PSC) SAPHO (*Scalable Architecture Processor for Hardware Optimization*), que é capaz de realizar operações aritméticas em ponto fixo e em ponto flutuante. Esse processador foi desenvolvido no Núcleo de Instrumentação e Processamento de Sinais (NIPS) da Universidade Federal de Juiz de Fora,

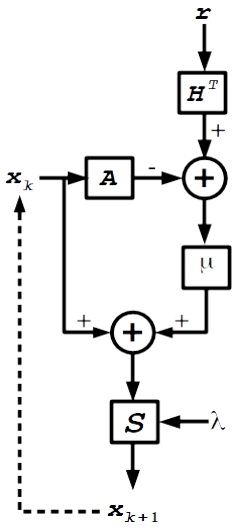

Figura 3. Diagrama da estrutura do SSF modificado.

sendo utilizado como parte integrante em diversos sistemas já consolidados, como em Kapisch et al. (2014), Kapisch et al. (2016), Martins et al. (2016) e Oliveira et al. (2018). Detalhes de projeto deste processador podem ser encontradas nas referências citadas acima.

Diferentemente dos diversos PSCs disponíveis, o SAPHO não possui uma arquitetura fixa. Nele, os recursos são automaticamente alocados em tempo de compilação do programa embarcado, de forma que apenas os elementos lógicos necessários são utilizados, reduzindo o custo computacional do projeto como um todo.

O SAPHO possui uma IDE *(Integrated Development Environment)* desenvolvida em C#, que tem o intuito de executar os compiladores C e *Assembler* de forma transparente para o usuário. O compilador C foi desenvolvido utilizando as ferramentas *flex* e *bison* da GNU (Levine, 2009), e tem a função de interpretar o código do usuário, escrito em um subconjunto da linguagem C (desenvolvido no NIPS), para então gerar o respectivo arquivo em linguagem *Assembly*.

O compilador *Assembler* foi desenvolvido com o auxílio da ferramenta *flex*, com objetivo de fazer o reconhecimento do *opcode* e dos operandos de cada uma das 52 instruções, sendo o responsável por gerar os arquivos em Verilog com a descrição do *hardware* do processador. O *Assembler* reconhece a quantidade final de instruções do programa e o número de variáveis necessárias, criando então as memórias de programa e de dados com o tamanho correto, além de alocar, em *hardware*, somente os recursos de processamento necessários para executar as operações encontradas no código em *Assembly*.

4. IMPLEMENTAÇÃO PROPOSTA

4.1 Estrutura Multi-Core

Para a criação da estrutura *multi-core*, visando a implementação do método iterativo apresentado, o SAPHO foi utilizado para realizar as operações matriciais do algoritmo em ponto fixo. Em cada processador, uma janela do sinal digitalizado é processada integralmente, de acordo com o fluxograma da Figura 3, onde as operações matriciais são calculadas e as iterações do algoritmo são realizadas em um *loop*, por *software*. Após finalizado esse processo, o sinal reconstruído é representado por uma janela de dados gerada na saída.

Na Figura 4 está ilustrado um diagrama da estrutura *multi-core* com 140 desses processadores, em paralelo, cada um operando a uma frequência de 320 MHz com funcionamento em *pipeline*.

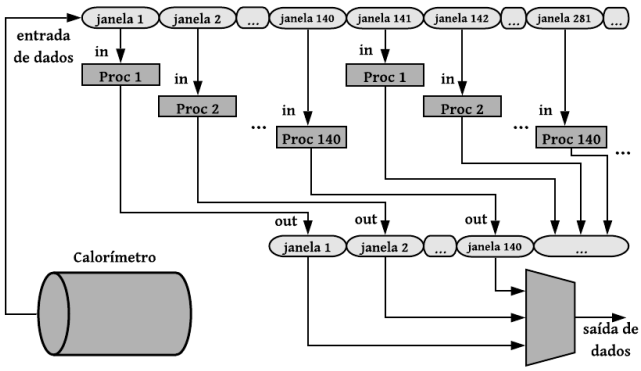

Figura 4. Diagrama da estrutura *multi-core* implementada.

Nesta implementação, cada janela de dados é executada por um processador, mantendo, portanto, um fluxo contínuo e sequencial de processamento. Desta forma, após cada processador receber sua janela de dados de entrada, os mesmos realizam suas operações de forma independente entre si e, na saída, um multiplexador recebe a janela contendo o sinal reconstruído por cada processador, respectivamente, no instante correto. A janela de dados de saída do primeiro processador e a janela de dados de entrada do último processador ocorrem de forma simultânea. Dessa forma, o primeiro processador pode então carregar a sua janela de dados de entrada novamente e o processo se repete sem que ocorra perda de dados.

4.2 Análise e Escolha dos Parâmetros

O grande número de operações matriciais contidas no método da Representação Esparsa emprega um alto custo computacional para o processador.

Com o objetivo de reduzir o número de operações realizadas pelo processador, todas as equações propostas pelo método foram simplificadas ao máximo para otimizar sua implementação, conforme indicado na Equação 2.

Além disso, como o processador opera em ponto fixo, realizou-se uma quantização dos elementos das matrizes (que são constantes). Como o resultado das operações muitas vezes é menor que 1, foi necessário aplicar um ganho a esses valores de forma que eles não fossem zerados quando truncados e também para que uma precisão mínima fosse mantida. Então, os elementos foram multiplicados por um valor fixo de ganho e posteriormente truncados, para se trabalhar apenas com valores inteiros. As análises de quantização serão mostradas na seção de resultados.

Diferentemente da maioria dos processadores que possuem um tamanho fixo de 32 ou 64*bits*, o SAPHO pode operar com um número customizável de *bits*. Dessa forma, após o valor do ganho ser definido e fixado, uma nova

análise foi realizada para determinar o menor número de *bits* necessários para a operação do processador que não comprometesse seu funcionamento e sua precisão.

4.3 Otimizações na Estrutura do Processador

Após realizada a quantização, o código foi escrito utilizando o subconjunto da linguagem C e, após cuidadosa análise, foram criadas duas funções exclusivas dessa linguagem com o intuito de otimizar o processador, visando diminuir o número de elementos lógicos e também o número de instruções do código gerado em *Assembly*.

A primeira função implementada tem o objetivo de simplificar a comparação indicada ao lado esquerdo da Figura 5, que gera as instruções em *Assembly* vistas ao lado direito da mesma figura.

```
if (aux_0  < 0) x_0  = 0; else x_0  = aux_0;  | LOAD 0
if (aux_1  < 0) x_1  = 0; else x_1  = aux_1;  | LES aux_0
if (aux_2  < 0) x_2  = 0; else x_2  = aux_2;  | JZ L3else
if (aux_3  < 0) x_3  = 0; else x_3  = aux_3;  | LOAD 0
if (aux_4  < 0) x_4  = 0; else x_4  = aux_4;  | SET x_0
if (aux_5  < 0) x_5  = 0; else x_5  = aux_5;  | JMP L3end
if (aux_6  < 0) x_6  = 0; else x_6  = aux_6;  | @L3else LOAD aux_0
if (aux_7  < 0) x_7  = 0; else x_7  = aux_7;  | SET x_0
if (aux_8  < 0) x_8  = 0; else x_8  = aux_8;  | @L3end
if (aux_9  < 0) x_9  = 0; else x_9  = aux_9;
if (aux_10 < 0) x_10 = 0; else x_10 = aux_10;
```

Figura 5. Código a ser otimizado em C na esquerda e respectivo em *Assembly* na direita. A primeira linha em C gera as instruções destacadas em *Assembly*.

Para simplificar a sintaxe já conhecida das funções *if* e *else* implementada na figura 5, que tem o intuito de eliminar os valores negativos, foi criado o operador @, visto ao lado esquerdo da Figura 6, que gera o código em *Assembly* (ao lado direito da mesma figura), onde cada linha de código em C corresponde a duas instruções, LOAD e PSET, em *Assembly*. A instrução PSET analisa o valor ao lado direito do @ e, se ele for positivo, a variável da esquerda recebe esse valor, caso contrário, ela recebe 0.

```
x_0@aux_0 ;  | LOAD aux_0
x_1@aux_1 ;  | PSET x_0
x_2@aux_2 ;  | LOAD aux_1
x_3@aux_3 ;  | PSET x_1
x_4@aux_4 ;  | LOAD aux_2
x_5@aux_5 ;  | PSET x_2
x_6@aux_6 ;  | LOAD aux_3
x_7@aux_7 ;  | PSET x_3
x_8@aux_8 ;
x_9@aux_9 ;
```

Figura 6. Código otimizado na esquerda e o respectivo em *Assembly* na direita. Cada linha de código em C corresponde a apenas 2 instruções em *Assembly*.

Já a segunda função tem como propósito simplificar o circuito de divisão, uma vez que, devido ao ganho constante, este valor é repetidamente utilizado nas contas de divisão. Para tal, foi criado o novo operador />, visto no lado esquerdo da Figura 7, que gera em *Assembly* a instrução NORM, exemplificada no lado direito da mesma figura. Esta instrução, definida na ULA (Unidade Lógico-Aritmética), normaliza o número a direita do operador. Assim, os circuitos de divisão implementados são drasticamente simplificados pois a ULA não tem mais a necessidade de verificar o valor do dividendo.

```
x_0  = /> x_0;  | LOAD x_0
x_1  = /> x_1;  | NORM
x_2  = /> x_2;  | SET x_0
```

Figura 7. Código para normalização em C na esquerda e o respectivo em *Assembly* na direita.

5. RESULTADOS

5.1 Estrutura Multi-Core

Após realizadas as devidas simulações de operação da estrutura *multi-core*, utilizando o *software* Modelsim-Altera, foi possível analisar como o número de processadores, o tempo de atraso Δt (tempo necessário para que o sinal reconstruído comece a ser descarregado na saída) e a quantidade de elementos lógicos da estrutura se comportam de acordo com a variação da frequência de operação.

Os resultados dessa análise estão apresentados nas Figuras 8 e 9. Observando esses gráficos, é possível notar que para a comparação entre a frequência de operação de 2GHz com a frequência da primeira implementação (320MHz), o número necessário de processadores diminuiu de 140 para 26 (reduziu em quase 81%), o tempo de atraso também teve uma redução de cerca de 81% (alcançando 35μs) e o número de elementos lógicos teve uma redução de 65%, ou seja, diminuiu em cerca de 61 mil.

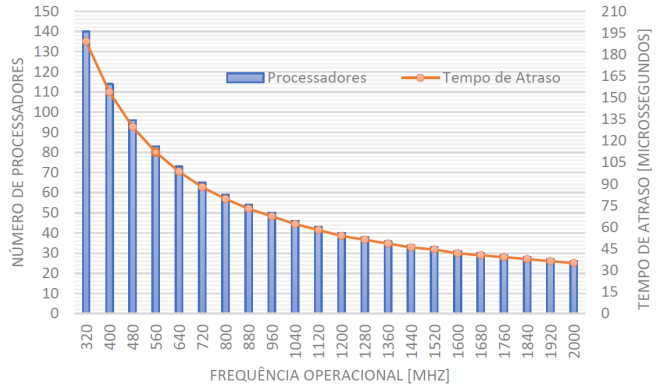

Figura 8. Quantidade de processadores e tempo de atraso (Δt) variando com a frequência de operação.

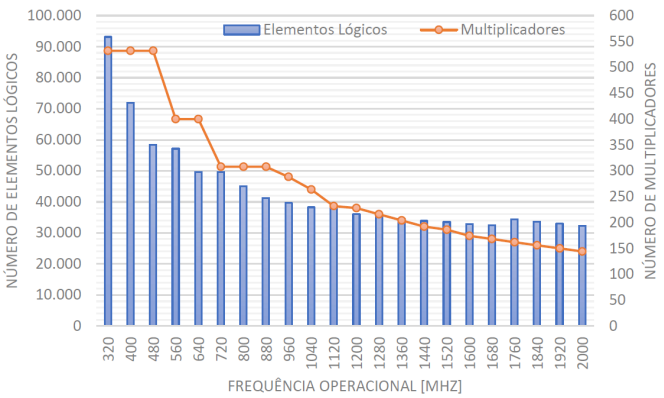

Figura 9. Quantidade de elementos lógicos e multiplicadores variando com a frequência de operação.

5.2 Análise e Escolha dos Parâmetros

Para a definição do valor do ganho a ser utilizado na implementação em ponto fixo do método de RE apresentado foi realizada uma análise, comparando o valor do ganho, em potência de 2, com o valor do RMS do erro entre o dado de saída e o dado real simulado. Através desses resultados, apresentados na Figura 10, é possível constatar que um ganho de valor 2^5 já seria suficiente para se obter o erro mínimo pelo método SSF proposto (Teixeira et al., 2018), sem a necessidade de ocupar muitos *bits* no processamento. Dessa forma, foi escolhido o valor de 2^7 para o ganho com o intuito de garantir uma margem de segurança.

Para reduzir ainda mais o custo computacional do processador, que a princípio estava configurado com 32 *bits*, realizou-se uma análise onde foi constatado que o SAPHO poderia operar com 29 *bits* sem comprometer seu desempenho, assim, o número de *bits* da ULA foi alterado e obteve-se uma redução de 6,02% dos recursos lógicos em cada processador.

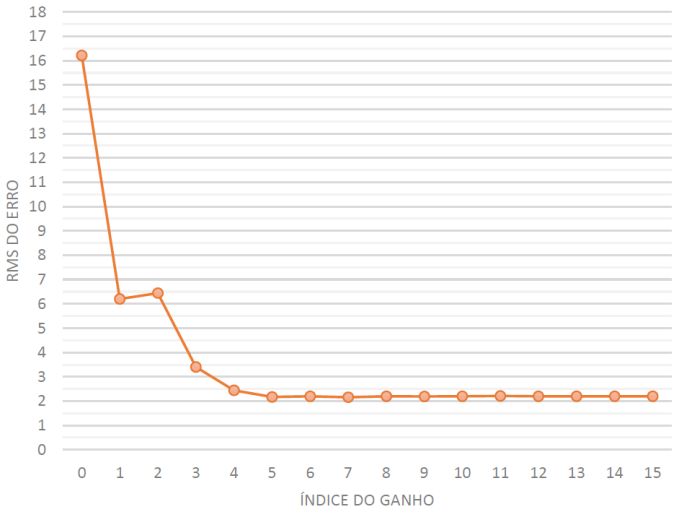

Figura 10. Gráfico do valor RMS do erro pelo ganho.

5.3 Otimizações na Estrutura do Processador

Em razão da implementação da função PSET, na estrutura do SAPHO, com objetivo de diminuir o número de instruções e também o número de *clocks* do processamento, houve uma redução de 11% do número de linhas no código em *Assembly* gerado, como indicado na Tabela 1. Como efeito da efetivação da função NORM, ocorreu uma redução significativa no custo lógico da arquitetura *multi-core*, como pode ser visto na Tabela 2, onde foi analisada a estrutura operando com frequências de 320MHz e 2GHz.

Tabela 1. Número de Instruções em Assembly

Operador	Sem otimização	Com otimização
@ | PSET	5199	4623

Tabela 2. Número de Elementos Lógicos

Frequência	Sem NORM	Com NORM
320 MHz	261844	93159
2 GHz	157103	32216

6. CONCLUSÃO

Com as modificações e aprimoramentos no processador embarcado, em conjunto com os estudos e simulações para definir um valor adequado para o ganho e o número de bits necessários, foi possível realizar a implementação de uma estrutura *multi-core* capaz de operar o método iterativo de deconvolução baseado em Representação Esparsa de dados para realizar a reconstrução do sinal de energia, respeitando a frequência de operação do sistema de aquisição de dados do Calorímetro Hadrônico do ATLAS.

Como próxima etapa desse trabalho, o objetivo é realizar novas modificações na Unidade Lógico-Aritmética e na estrutura do processador para otimizar o tempo de latência e o custo lógico da estrutura *multi-core* apresentada.

AGRADECIMENTOS

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (CAPES) - Código de Financiamento 001. Gostaríamos de agradecer também ao CNPq, RENAFAE, FAPEMIG, FAPERJ pelo apoio financeiro no projeto. E ainda ao Experimento ATLAS, em especial ao *TileCal*, pelo suporte ao desenvolvimento do trabalho.

REFERÊNCIAS

Aad, G. (2012). Observation of a new particle in the search for the standard model higgs boson with the atlas. *Phys. Lett.*, 716, 1–29.

Aguiar, M.S., Andrade Filho, L.M., and Seixas, J.M. (2019a). Processamento embarcado para implementação de um método iterativo de deconvolução visando a reconstrução de energia em calorímetros de altas energias. In *XXXVII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais*. Petrópolis.

Aguiar, M.S., de Oliveira Resende, M., Teixeira, T., de Andrade Filho, L.M., and de Seixas, J.M. (2019b). Implementação em fpga de um método iterativo de deconvolução para operar no sistema de trigger do experimento atlas. In *XXII Encontro Nacional de Modelagem Computacional e X Encontro de Ciência e Tecnologia de Materiais*. Essentia Editora, Juiz de Fora.

Aleksa, M., Cleland, W., Enari, Y., Fincke-Keeler, M., Hervas, L., Lanni, F., Majewski, S., Marino, C., and Wingerter-Seez, I. (2013). Atlas liquid argon calorimeter phase-i upgrade technical design report. *CERN Document Server*.

Anthony, K. (2014). Celebrating the first of a kind. *CERN Document Server*.

Barbosa, D.P., d. A. Filho, L.M., Peralva, B.S., Cerqueira, A.S., and de Seixas, J. (2017). Sparse representation for signal reconstruction in calorimeters operating in high luminosity. *IEEE Transactions on Nuclear Science*, 64(7), 1942–1949. doi:10.1109/TNS.2017.2712420.

Barbosa, D.P. (2012). *Estudo de Técnicas de Otimização para reconstrução de Energia de jatos no Primeiro Nível de Seleção de Eventos do Experimento ATLAS*. Master's thesis, UFJF.

Carrio, F., Kim, H.Y., Moreno, P., Reed, R., Sandrock, C., Shalyugin, A., Schettino, V., Souza, J., Solans, C., Usai, G., and Valero., A. (2013). Design of an fpga-based

embedded system for the atlas tile calorimeter front-end electronics test-bench. *CERN Document Server*.

Cepeda, M., Gori, S., Ilten, P., and Collaboration (2018). Report from Working Group 2: Higgs Physics at the HL-LHC and HE-LHC. *CERN Yellow Rep. Monogr.*, 7(arXiv:1902.00134), 221–584. 364 p. doi:10.23731/CYRM-2019-007.221.

CERN (2014). The large hadron collider. *CERN Document Server*.

Cleland, W. and Stern, E. (1994). Signal processing considerations for liquid ionization calorimeters in a high rate environment. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 338(2-3), 467–497. doi:10.1016/0168-9002(94)91332-3.

Duarte, J.a.P.B.d.S. (2015). *Estudo de Técnicas de Deconvolução para Reconstrução de Energia online no Calorímetro Hadrônico do ATLAS*. Master's thesis, UFJF.

Duarte, J., de Andrade Filho, L., de Simas Filho, E., Farias, P., and de Seixas, J. (2019). Online energy reconstruction for calorimeters under high pile-up conditions using deconvolutional techniques. *Journal of Instrumentation*, 14(12), P12017–P12017. doi:10.1088/1748-0221/14/12/p12017.

Elad, M. (2010). *Sparse and Redundant Representations*. Springer New York, New York. doi:10.1007/978-1-4419-7011-4.

Filho, L.M.A., Peralva, B.S., Seixas, J.M., and Cerqueira, A.S. (2015). Calorimeter response deconvolution for energy estimation in high-luminosity conditions. *IEEE Transactions on Nuclear Science*, 62(6), 3265–3273.

Herr, W. (2005). Dynamic behaviour of nominal and pacman bunches for different lhc crossing schemes. *CERN Document Server*.

Kapisch, E.B., Silva, L.R.M., Cerqueira, A.S., Andrade Filho, L.M., Duque, C.A., and Ribeiro, P.F. (2016). A Gapless Waveform Recorder for Monitoring Smart Grids. *17th International Conference on Harmonics and Quality of Power (ICHQP)*, 130–136.

Kapisch, E.B., Silva, L.R.M., Martins, C.H.N., Barbosa, A.S., Duque, C.A., Andrade Filho, L.M., and Cerqueira, A.S. (2014). An Electrical Signal Disturbance Detector and Compressor Based on FPGA Platform. *16th International Conference on Harmonics and Quality of Power (ICHQP)*, 278–282.

Levine, J.. (2009). *Flex & Bison: Text Processing Tools*. O'Reilly Media, Inc.

Martins, C.H., Monteiro, H.L.M., M., O.M., Silva, L.R.M., Duque, C.A., and Ribeiro, P.F. (2016). A Virtual Instrument for Time Varying Harmonic Analysis. *IEEE Power and Energy Society General Meeting (PESGM)*, 1–5.

Nakahama, Y. (2015). The atlas trigger system: Ready for run-2. *CERN Document Server*.

Oliveira, M.M., Silva, L.R.M., Duque, C.A., ANDRADE FILHO, L.M., and Ribeiro, P.F. (2018). Implementation of an Electrical Signal Compression System Using Sparce Representation. *18th International Conference on Harmonics and Quality of Power (ICHQP)*, 1–5.

Pequenao, J. (2008). Computer generated image of the whole atlas detector. *CERN Document Server*.

Peralva, B.S.M., Filho, L.M.A., Cerqueira, A.S., and Seixas, J.M. (2013). The TileCal Energy Reconstruction for Collision Data Using the Matched Filter. Technical Report ATL-TILECAL-PROC-2013-023, CERN, Geneva.

Peralva, B.S.M. (2012). *Detecção de sinais e estimação de energia para calorimetria de altas energia*. Master's thesis, UFJF.

Perkins, D. (2000). Introduction to high energy physics. *Cambridge University Press*.

Teixeira, T.A., Andrade Filho, L.M., and Seixas, J.M. (2019). Implementação de métodos iterativos de deconvolução para processamento on-line no calorímetro hadrônico do atlas. In *XLII RTFNB - XL ENFPC 2019*. SBF, Campos do Jordão.

Teixeira, T.A., Duarte, J.P.B.S., Andrade Filho, L.M., and Seixas, J.M. (2018). Implementação em fpga de um método recursivo de deconvolução aplicado em calorímetros operando a alta taxa de eventos. In *XXXVI Simpósio Brasileiro de Telecomunicações e Processamento de Sinais*. SBrT, Campina Grande - PB.

Wigmans, R. (2018). Calorimetry. *Oxford University Press*.

Wille, W. (2000). The physics of particle accelerators: An introduction. *Clarendon Press*.