Simulador de Pulsos do TileCal em Tempo Real Utilizando o Processador SAPHO e FPGA

Leandro da G. Ribeiro*. Luciano M. de Andrade Filho**

**Departamento de Engenharia Elétrica, Universidade Federal de Juiz de Fora, Brasil, (e-mail: ribeiro.leandro@estudante.ufjf.br).*
*** Departamento de Engenharia Elétrica, Universidade Federal de Juiz de Fora, Brasil, (e-mail: luciano.andrade@ufjf.br)*

Abstract: The need for accurate real-time simulation of physical events in high-energy experiments drives the development of new processing architectures. This paper presents an approach based on the SAPHO processor and Field Programmable Gate Arrays (FPGAs) for free-running simulation of Tile Calorimeter pulses in the ATLAS experiment at CERN. SAPHO compiles C code and generates Verilog (.v) code, which is implemented in FPGA to simulate the calorimeter response in real-time. The use of FPGAs enables high parallelization and low latency, essential features for applications requiring fast and deterministic processing. This paper discusses the system architecture, implementation challenges, and benefits of using reconfigurable hardware. Results demonstrate the feasibility of the solution and its applicability for validating new energy reconstruction algorithms, contributing to the improvement of the quality of data acquired by the detector.

Resumo: A necessidade de simulação precisa e em tempo real de eventos físicos em experimentos de alta energia impulsiona o desenvolvimento de novas arquiteturas de processamento. Este artigo apresenta uma abordagem baseada no processador SAPHO e em Field Programmable Gate Arrays (FPGAs) para a simulação free-running de pulsos do calorímetro hadrônico de telhas do experimento ATLAS no CERN. O SAPHO compila código C e gera código Verilog (.v), que é implementado em FPGA para simular a resposta do calorímetro em tempo real. A utilização de FPGAs permite alta paralelização e baixa latência, características fundamentais para aplicações que exigem processamento rápido e determinístico. O artigo discute a arquitetura do sistema, os desafios na implementação e os benefícios do uso de hardware reconfigurável. Resultados demonstram a viabilidade da solução e sua aplicabilidade para validação de novos algoritmos de reconstrução de energia, contribuindo para a melhoria da qualidade dos dados adquiridos pelo detector.

Keywords: FPGA, SAPHO, *pulse simulator, Tile Calorimeter, high event rate*, CERN.

Palavras-chaves: FPGA, SAPHO, simulador de pulsos, Calorímetro Hadrônico de Telhas, alta taxa de eventos, CERN.

1. INTRODUÇÃO

A busca pela compreensão das leis fundamentais da natureza impulsionou o desenvolvimento de experimentos científicos cada vez mais sofisticados. A física de partículas, um dos campos mais avançados da ciência moderna, investiga os constituintes básicos da matéria e as forças fundamentais que regem o universo.

Para isso, são utilizados aceleradores de partículas, que colidem feixes de prótons ou íons a altíssimas energias, permitindo a observação de fenômenos que ocorrem em escalas subatômicas. Essas colisões geram um grande volume de dados que precisa ser registrado, filtrado e analisado em tempo real para que se possa extrair informações relevantes sobre a estrutura da matéria e as leis que governam o cosmos (Aad et al., 2008).

Um dos maiores centros mundiais dedicados à física de partículas é o *European Organization for Nuclear Research* (CERN), que opera o maior e mais poderoso acelerador de partículas do mundo, o Grande Colisor de Hádrons (LHC).

Dentre os detectores instalados ao redor do LHC, o experimento ATLAS se destaca por sua capacidade de registrar uma ampla gama de processos físicos, utilizando uma combinação de sistemas de detecção altamente especializados para medir as partículas resultantes das colisões com alta precisão (Evans & Bryant, 2008).

No ATLAS, o *Tile Calorimeter* (TileCal) é um dos subsistemas responsáveis pela medição da energia das partículas hadrônicas, desempenhando um papel fundamental na identificação de jatos de partículas e na calibração do detector. Para que o TileCal possa operar eficientemente em condições de alta taxa de eventos, é necessário o

desenvolvimento de métodos avançados de processamento de sinais que permitam a extração precisa das informações relevantes dos pulsos gerados pelo detector (The ATLAS Collaboration, 2014).

Nesse contexto, o uso de dispositivos reconfiguráveis, como as *Field Programmable Gate Arrays* (FPGAs), surge como uma solução promissora para lidar com as exigências computacionais do experimento. No entanto, para que essas soluções sejam viáveis, é essencial garantir que a FPGA possa operar com uma taxa de *clock* compatível com a frequência de eventos do LHC. Quanto menor o número de ciclos de *clock* necessários para processar cada pulso do TileCal, maior a capacidade da FPGA de acompanhar a taxa de eventos do experimento sem comprometer a precisão das medições (Lisauskas et al., 2020).

O objetivo deste trabalho é desenvolver um simulador baseado em FPGA, capaz de modelar os pulsos do TileCal em tempo real. Esse simulador permitirá testar e validar novos algoritmos de reconstrução de energia em condições semelhantes às do experimento ATLAS, contribuindo para o aprimoramento das técnicas de análise de dados empregadas no detector. Para viabilizar essa implementação, será utilizada a plataforma SAPHO, que possibilita a conversão de código C diretamente em Verilog, facilitando sua integração ao hardware reconfigurável. Além disso, será avaliada a eficiência do simulador em termos de uso de recursos da FPGA e sua capacidade de operar dentro das restrições de tempo impostas pela alta taxa de eventos do LHC.

2. CONSIDERAÇÕES INICIAIS

2.1 TileCal

O *Tile Calorimeter* (TileCal) é um calorímetro hadrônico do experimento ATLAS, no *Large Hadron Collider* (LHC), localizado no CERN. Ele desempenha um papel essencial na medição da energia de hádrons, como píons e prótons, garantindo a reconstrução precisa dos eventos gerados nas colisões próton-próton.

Esse é um calorímetro de amostragem composto por azulejos (*tiles*) de cintilador plástico, intercalados com placas de aço que atuam como material absorvedor. A interação das partículas incidentes com o aço gera chuveiros hadrônicos, que excitam os cintiladores e produzem luz proporcional à energia depositada. Essa luz é captada por fotodetectores, como *photomultiplier tubes* (PMTs), que a convertem em sinais elétricos processáveis (Amorim et al., 2019).

Cada célula do calorímetro está conectada a duas PMTs redundantes, garantindo a robustez do sistema e minimizando a perda de informação em caso de falha. Os sinais elétricos são amplificados e digitalizados por eletrônica de leitura dedicada, enviando os dados para os sistemas de reconstrução de eventos do ATLAS (ATLAS Collaboration, 2008).

2.2. Pulsos de Saída e o Desafio do Pile-Up

O efeito de *pile-up* ocorre devido à alta taxa de colisões no LHC, onde múltiplos eventos podem se sobrepor temporalmente, dificultando a reconstrução precisa da energia dos sinais. Como o LHC opera com uma frequência de 40 MHz (ou seja, uma colisão a cada 25 ns), sinais de eventos consecutivos podem se acumular e modificar a forma do pulso detectado.

Para ilustrar esse fenômeno, a Figura 1 apresenta duas componentes normais de eventos distintos (linhas azul e verde tracejadas) e a resultante de sua soma (linha vermelha contínua). Esse efeito impacta diretamente a performance da reconstrução de energia, exigindo algoritmos sofisticados para mitigar sua influência e garantir medições precisas.

.

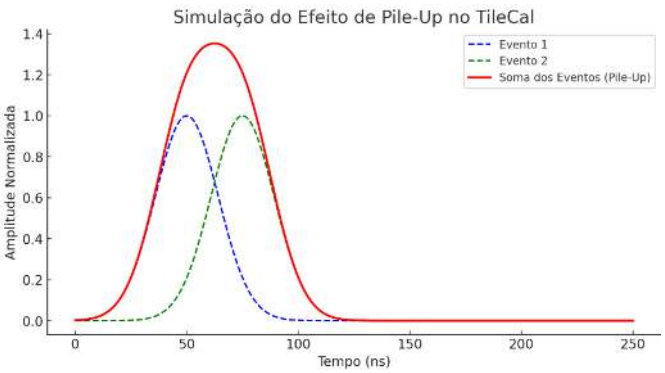

Figura 1 - Ilustração do efeito de *pile-up* (Autoria própria).

O desenvolvimento de simuladores precisos para esses pulsos de saída é essencial para estudar e aprimorar os métodos de reconstrução de energia, minimizando os impactos do pile-up e garantindo a confiabilidade das medições no TileCal.

2.3 Visão Geral do SAPHO

O SAPHO **(***Scalable Architecture Processor for Hardware Optimization*) é um processador *soft-core* de código aberto projetado para alocação automática de recursos de hardware durante a compilação de programas embarcados. Ele é amplamente utilizado em arquiteturas de multiprocessamento e tem sido implementado em diversos sistemas consolidados.

O funcionamento do SAPHO baseia-se em dois compiladores acessíveis por meio de sua Interface de Desenvolvimento Integrada (IDE). O primeiro compilador interpreta programas escritos em um subconjunto da linguagem C e os converte para *Assembly*. O segundo compilador traduz o Assembly gerado para código de máquina, alocando automaticamente os recursos de hardware necessários e gerando um arquivo de configuração em Verilog.

O montador (*assembler*) identifica as instruções produzidas pelo programa escrito pelo usuário e cria apenas os recursos de hardware necessários para sua execução. A maioria das configurações, como o desenvolvimento de circuitos internos da Unidade Lógica Aritmética (ALU), é automatizada. Os programadores podem modificar diversas características adicionais, incluindo o tamanho das palavras, o tipo de ALU (ponto fixo ou ponto flutuante) e o número de portas de entrada e saída, por meio de diretivas de compilação.

Além do próprio processador, que inclui o *core* e duas unidades de memória, diversas ferramentas foram desenvolvidas para facilitar a programação. Entre elas, destacam-se a interface de desenvolvimento, o compilador que interpreta a linguagem C e um *assembler*.

Algumas vantagens do SAPHO incluem:

- Execução de Instrução por Ciclo de *Clock*: Capacidade de executar uma instrução por ciclo de *clock*, mesmo em rotinas com desvios condicionais, sem comprometer o pipeline, graças à sua arquitetura de pipeline de três estágios.
- ALU Configurável: Possibilidade de ajustar a ALU para diferentes tamanhos de palavra e modos de operação (ponto fixo e ponto flutuante).
- Alocação Parametrizada de Recursos: Recursos alocados dinamicamente em tempo de projeto, adaptando-se à aplicação desenvolvida.
- Flexibilidade na Programação: Os usuários podem programar o processador utilizando um subconjunto da linguagem C ou diretamente em Assembly.

3. METODOLOGIA PROPOSTA

Nesta secção serão apresentados os fundamentos necessários e a metodologia matemática utilizada para a concepção do simulador. Na Figura 2, pode-se identificar um modelo base do fluxo de como o simulador funciona:

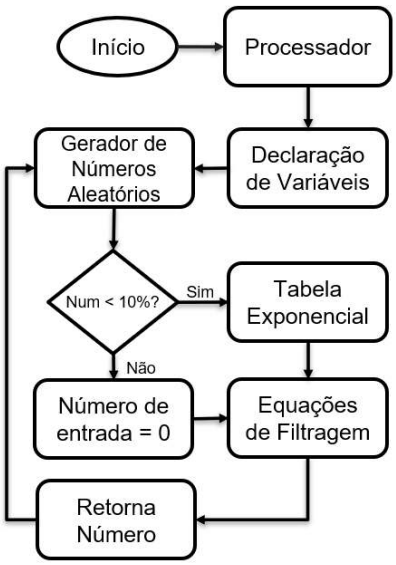

Figura 2: Fluxograma Base do Simulador.

Como foi mencionado anteriormente, o processador foi construído utilizando o SAPHO, e as demais etapas serão explicadas a seguir.

3.1. Gerador de Números Aleatórios

A geração de números pseudoaleatórios desempenha um papel fundamental na simulação de pulsos de saída do TileCal, sendo nesse caso utilizado para representar a amplitude do sinal de saída.

No contexto de um simulador implementado em hardware, a escolha de um método adequado para gerar sequências de números pseudoaleatórios apresenta desafios específicos, uma vez que algoritmos tradicionais, como geradores congruências lineares, podem ser computacionalmente ineficientes ou consumir recursos excessivos.

O gerador utilizado neste trabalho baseia-se em um método congruencial multiplicativo, com parâmetros cuidadosamente escolhidos para garantir um período longo e uma distribuição estatisticamente aceitável dos números gerados. O algoritmo é definido por:

$$rnd = (rnd \cdot a) + c \quad (1)$$

Onde:

- a = 2891336453
- c = 12345
- *seed* = 4123 (semente inicial)
- *rnd* é o valor pseudoaleatório gerado

Após essa operação, a saída é processada com deslocamento e máscara para garantir valores dentro do intervalo desejado.

$$rnd_{out} = (rnd \gg bits) \& \, 127 \quad (2)$$

No contexto do simulador do TileCal, a aleatoriedade dos números gerados influencia diretamente a precisão das simulações, impactando, por exemplo, a modelagem de ruído eletrônico e a distribuição de energia dos pulsos reconstruídos. Assim, a escolha e a implementação de um gerador eficiente são essenciais para garantir que os resultados da simulação reflitam adequadamente as condições reais do detector.

3.2 Taxa de Ocupação e Tabela Exponencial

É importante ressaltar que na maioria dos eventos gerados no calorímetro, a amplitude da energia é igual a zero. Ou seja, nem sempre é gerado um valor não nulo e quando é, possui característica exponencial.

Com o objetivo de simular de forma mais fidedigna, foi considerado uma taxa de ocupação de 10%, ou seja, apenas 10% dos valores aleatórios passarão para a próxima etapa. Isso significa que em 90% dos casos, a entrada da etapa de filtragem será igual a zero.

O valor *rnd_out,* caso diferente de zero, será o endereço de uma tabela de formato exponencial, gerado anteriormente. Isso permitirá que os dados que serão a entrada do filtro possuam

característica exponencial e não uniforme, visto que o gerador de número aleatório funciona de forma uniforme.

3.3 Modelagem Matemática para Simulação de Pulsos com Efeito de Pile-up

A modelagem dos pulsos é realizada através de filtros digitais, que são implementados no domínio discreto. Esses filtros são projetados para reproduzir a resposta do calorímetro hadrônico a um pulso de entrada. Os filtros são definidos por suas funções de transferência no domínio da frequência, que são posteriormente transformadas para o domínio do tempo utilizando a transformada bilinear. Essa transformada é uma técnica comum para converter filtros analógicos em filtros digitais, preservando a estabilidade e a resposta em frequência.

No código MATLAB, cinco filtros diferentes são utilizados, cada um com seus próprios coeficientes de numerador (*b0, b1* e *b2*) e denominador (*a1*, e *a2*). Esses coeficientes são obtidos a partir da transformada bilinear e são utilizados para filtrar o sinal de entrada *x*, que é gerado por um gerador de números pseudoaleatórios. A saída de cada filtro é representada por *y1* a *y5*, que são combinadas para formar o sinal final y6.

As equações de filtragem no domínio do tempo são implementadas diretamente no código MATLAB e podem ser traduzidas para o processador dedicado SAPHO em linguagem C. Cada filtro é representado por uma equação de diferenças, que relaciona a saída atual com as entradas e saídas anteriores. Abaixo, descrevemos as equações para cada um dos filtros:

1. Filtro 1 (y1):

$$y_1 = b_0 \cdot x + b_1 \cdot r_x - a_1 \cdot r_y \quad (3)$$

2. Filtro 2 (y2):

$$y_2 = b_0 \cdot x + b_1 \cdot r_{x1} + b_2 \cdot r_{x2} - a_1 \cdot r_{y1} - a_2 \cdot r_{y2} \quad (4)$$

3. Filtro 3 (y3):

$$y_3 = b_0 \cdot x + b_1 \cdot r_x - a_1 \cdot r_y \quad (5)$$

4. Filtro 4 (y4):

$$y_4 = b_0 \cdot x + b_1 \cdot r_{x1} + b_2 \cdot r_{x2} - a_1 \cdot r_{y1} - a_2 \cdot r_{y2} \quad (6)$$

5. Filtro 5 (y5):

$$y_5 = b_0 \cdot x + b_1 \cdot r_x - a_1 \cdot r_y \quad (7)$$

A saída final do sistema, que simula o efeito de *pile-up*, é obtida pela soma das saídas de todos os filtros:

$$y_{total} = y_1 + y_2 + y_3 + y_4 + y_5 \quad (8)$$

Nas equações acima, os coeficientes *b*0, *b*1, *b*2, *a*1, e *a*2 são os coeficientes dos filtros digitais, obtidos a partir da transformada bilinear aplicada às funções de transferência dos filtros no domínio da frequência. Esses coeficientes são específicos para cada filtro e são calculados no MATLAB. Abaixo, descrevemos o significado de cada termo:

- *b*0, *b*1, *b*2: Coeficientes do numerador do filtro, que multiplicam as entradas atuais e anteriores.
- *a*1 e *a*2: Coeficientes do denominador do filtro, que multiplicam as saídas anteriores.
- *x*: Valor atual do sinal de entrada.
- *rx*, *rx*1, *rx*2, *rx*3, *rx*5, *rx*14, *rx*24: Valores anteriores do sinal de entrada.
- *ry*, ry1, *ry2*, *ry*3, *ry*5, *ry*14, *ry*24: Valores anteriores do sinal de saída.

A implementação dessas equações no processador dedicado SAPHO é realizada diretamente em linguagem C, onde os coeficientes dos filtros são substituídos pelos valores obtidos no MATLAB.

4. RESULTADOS

Neste trabalho, foram desenvolvidos quatro simuladores, nos quais o principal resultado a ser analisado é a quantidade de clocks necessários para que gere uma amostra. Ou seja, após a construção do primeiro, os subsequentes têm a função de reduzir o número de clocks.

Abaixo é possível ver os simuladores criados, assim como suas características:

Tabela 1: Simuladores e Características

Simulador	**Número de processadores**	**Tipo**	**Equações de Filtragem**
1	1	Inteiro	Separadas
2	1	Inteiro	Combinadas
3	1	*Float*	Combinadas
4	5	*Float*	Separadas (uma por processador)

- Processadores: Quantidade de processadores utilizados.
- Tipo do Processador:
  - Inteiro: Processador que opera em ponto fixo. Quando há operações com variáveis sendo do tipo flutuantes, o processador realiza quantização.
  - *Float*: Processador que opera com números de ponto flutuante.
- Equações:
  - Separadas: Cada equação de filtragem é implementada de forma independente.

- Combinadas: As equações de filtragem são implementadas de forma conjunta.

Além disso, é importante pontuar que capacidade que a FPGA precisa ter para simular o sistema real é determinada pela multiplicação entre a quantidade de *clocks* para gerar uma amostra em cada simulador, e a taxa de eventos do LHC, ou seja, 40 MHz:

$$F_{sim} = qte\_clocks_{sim} \cdot 40\ MHz \quad (9)$$

- F_{sim} é a frequência que a FPGA deve ser capaz de operar para implementar o simulador em questão em tempo real, gerando 40 milhões de amostras por segundo.

4.1 Simulador 1

Nesse simulador, o processador é feito em ponto fixo e, além disso, os números aleatórios são gerados dentro do processador. Na figura abaixo é possível visualizar o *RTL viewer* (esquemático gerado) do simulador 1:

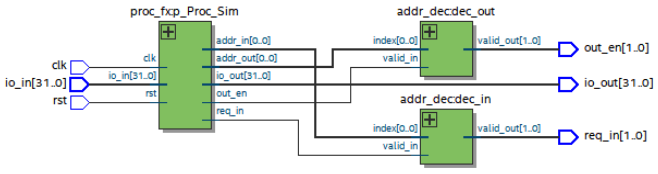

Figura 3. *RTL Viewer* do Simulador 1.

Com um recorte de aproximadamente 800 amostras, pode-se analisar a saída do simulador 1 na Figura 4:

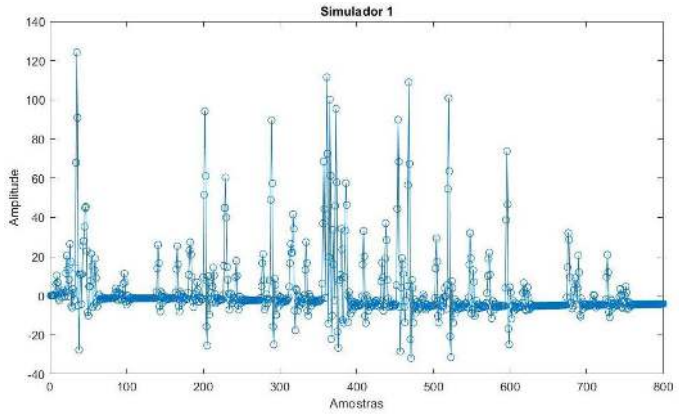

Figura 4. Saída do Simulador 1.

Quantidade de *clocks* por amostra: 5104.

Para esse simulador, de acordo com (9), seria necessário uma FPGA que fosse capaz de funcionar a uma taxa de:

$$F_{sim1} = 5104_{sim1} \cdot 40\ MHz = 204{,}16\ GHz$$

Logo, esse simulador é difícil de ser implementado com a taxa real de colisões, mas sendo válido para estudos.

4.2 Simulador 2

O próximo passo foi unificar as funções de transferência, para que houvesse um número menor de *clocks* final. O *RTL Viewer* é o mesmo, pois a única mudança foi na equação de filtragem, que está dentro do processador.

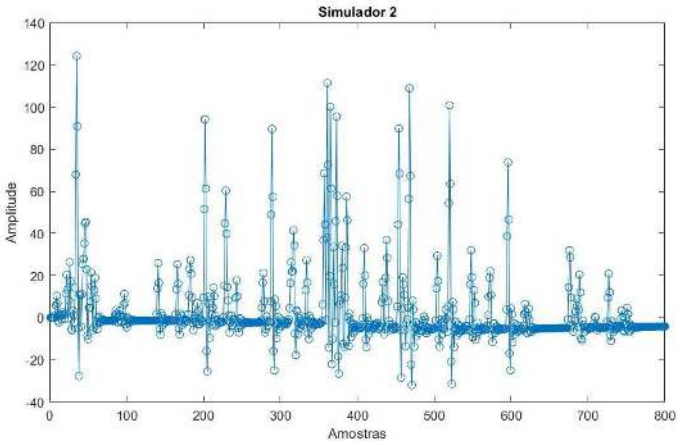

Figura 5. Saída do Simulador 2.

Nota-se que os valores de saída são iguais aos do primeiro simulador apresentado, porém, o número de *clocks* para gerar cada amostra foi 3390 *clocks.* Embora um valor menor, ainda seria necessário um FPGA com capacidade de funcionamento a 135,6 GHz para que o simulador aplicável em tempo real com a mesma taxa de eventos do CERN.

4.3 Simulador 3.

O próximo simulador muda a característica do processador, que passa a ser do tipo ponto flutuante, além de possuir o gerador de números aleatórios em um bloco independente, sendo então, gerado um número a cada *clock*. Pode-se visualizar o *RTL Viewer* na Figura 6:

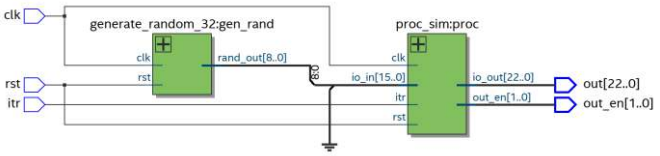

Figura 6. *RTL Viewer* do Simulador 3.

Pode-se visualizar na Figura 7 as amostras de saída do simulador 3:

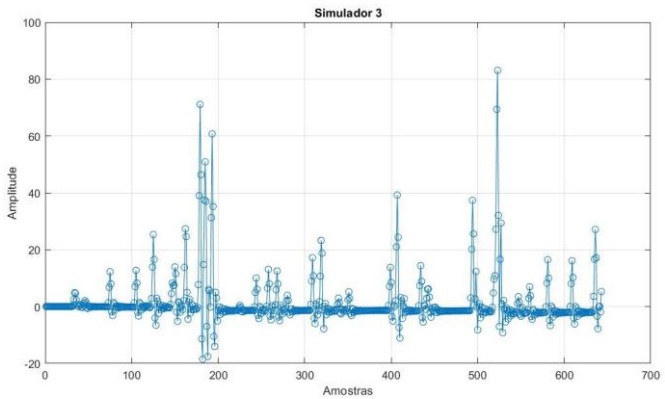

Figura 7. Saída do Simulador 3.

- Quantidade de *clocks* por amostra: 78.

Levando em consideração esse resultado, seria necessário que uma FPGA conseguisse performar a 1,92 GHz para o simulador 3, o que representa uma grande evolução em relação aos simuladores anteriores, sendo mais factível.

4.4 Simulador 4

O último simulador visa reduzir ao máximo o número de *clocks* realizando o paralelismo de processadores.

Assim sendo, foi criado um sistema no qual cinco processadores paralelos recebem o mesmo valor do bloco do gerador de número aleatório. Esse valor é zero em 90% das vezes devido a taxa de ocupação de 10%.

Após todos os processadores realizarem as suas operações e gerarem uma saída de forma síncrona, é realizada uma soma e, por fim, gerado uma amostra.

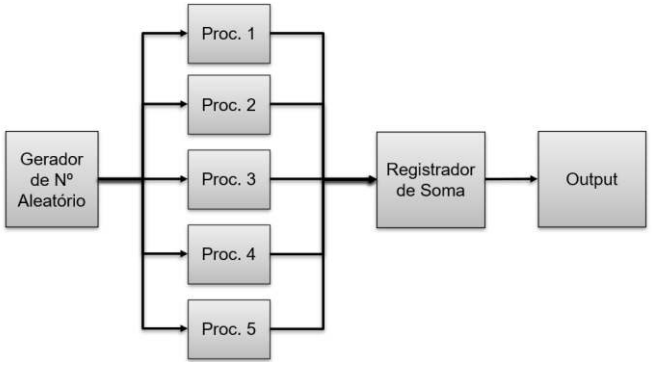

Figura 8. Estrutura do Simulador 4.

Pode-se ver na Figura 9 as amostras de saída do sistema:

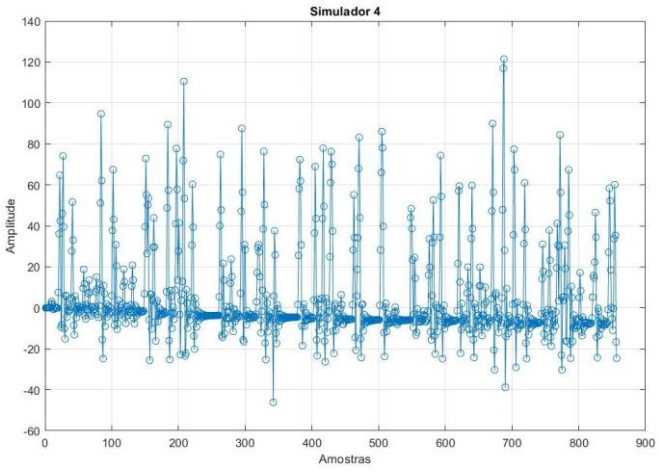

Figura 9. Amostras de Saída do Simulador 4.

- Quantidade de *clocks* por amostra: 32.

Nota-se que, há mais uma redução em relação ao simulador anterior. Em comparação com o primeiro e segundo simuladores, o simulador 4 é mais de 100 vezes mais eficiente.

Para esse simulador, é preciso uma FPGA que possa operar a 1,28 GHz para simular em tempo real, com a mesma taxa de eventos do ATLAS. Apesar de alto, representa uma redução considerável em relação aos anteriores.

6. CONCLUSÕES

É possível notar que houve uma evolução nos simuladores de forma a tornar mais factível sua aplicação para estudos de reconstrução de energia em tempo real. O simulador 4 obteve um resultado em que seria possível aplicar em ambientes com alta taxa de eventos, próximos a taxa real do experimento.

Avaliando os resultados das amostras, percebe-se que a posição do bloco de geração de número aleatório influencia no resultado, pois por se tratar de uma sequência pseudoaleatória, os simuladores 1 e 2 tiveram amostras de amplitudes iguais, enquanto os 3 e 4 de amplitudes distintas, pelo fato de o gerador estar localizado fora do processador.

Além disso, esse método apresenta, em relação a construção dos circuitos convencionais, resultados mais próximos aos reais, visto que não possui erro de quantização, geralmente presente com mais força em simuladores convencionais.

Para trabalhos futuros, será interessante realizar estudos de análise de erro no valor da saída ao retirar certas parcelas de equações, visando diminuir o número de *clocks*, visto que existem coeficientes de valores baixos que exercem pouca diferença, mas impactam no número de *clocks.*

REFERÊNCIAS

Aad, G., Abajyan, T., Abbott, B., Abdallah, J., Khalek, S. A., Abdelalim, A. A., ... & ATLAS Collaboration. (2008). The ATLAS Experiment at the CERN Large Hadron Collider. *Journal of Instrumentation*, *3*(08), S08003.

Amorim, A., Di Simone, A., Dotti, A., et al. (2019). *ATLAS Tile Calorimeter: Calibration, Operation and Performance*. *Journal of Instrumentation*, 14(09), P09012.

Evans, L., & Bryant, P. (2008). LHC Machine. *Journal of Instrumentation*, *3*(08), S08001.

Lager, S., Oreglia, M., Piedra, J., et al. (2016). *Pulse Processing in the ATLAS Tile Calorimeter*. *Nuclear Instruments and Methods in Physics Research A*, 831, 78-92.

Lisauskas, M. A., Ribeiro, A. P., & Souza, R. M. (2020). Real-time Data Processing with FPGA-based Architectures for High-Energy Physics Experiments. *IEEE Transactions on Nuclear Science*, *67*(5), 872-879.

The ATLAS Collaboration. (2014). Pile-Up Effects in the ATLAS Detector and their Impact on Physics Performance. *ATLAS-PUB-2014-004*.