RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

FLEXIBILIDADE EM SISTEMAS ELETRÔNICOS: IMPLEMENTAÇÃO DINÂMICA DE FUNCIONALIDADES EM DISPOSITIVOS DE ELETRÔNICA PROGRAMÁVEL

FLEXIBILITY IN ELECTRONICS SYSTEMS: DYNAMIC IMPLEMENTATION OF FUNCTIONALITIES IN PROGRAMMABLE ELECTRONICS DEVICES

FLEXIBILIDAD EN SISTEMAS ELECTRÓNICOS: IMPLEMENTACIÓN DINÁMICA DE FUNCIONALIDADES EN DISPOSITIVOS ELECTRÓNICOS PROGRAMABLES

Newton Silva Duarte[1]

e585643
https://doi.org/10.47820/recima21.v5i8.5643

PUBLICADO: 08/2024

RESUMO

Este artigo apresenta o potencial evolucionário dos dispositivos lógicos programáveis FPGA (*Field Programmable Gate Array*) na transformação do desenvolvimento e atualização de equipamentos eletrônicos. Ele se concentra na implementação de um *Soft Core*, que é um núcleo de processamento adaptável e configurável por *software*, que potencializa o processamento e a flexibilidade dos sistemas. O objetivo é explorar a aplicabilidade prática desta tecnologia, demonstrando o desenvolvimento de um sistema baseado na FPGA Spartan 7 da XILINX. A metodologia considerou o uso da suíte VIVADO para configurar os Blocos Lógicos programáveis, a implementação de um *Soft Core* e o IDE VITIS da XILINX, onde foi desenvolvido um *firmware* em C. Os principais resultados obtidos foram a integração inicial de um sensor de temperatura. Posteriormente suas funcionalidades foram ampliadas com a ativação de um sensor do tipo giroscópio. Os resultados foram validados com o uso do analisador lógico *Analog Discovery* 2 da DIGILENT. O artigo conclui que a adoção dessa tecnologia pode eliminar a necessidade de revisões físicas das placas eletrônicas diante da necessidade de adição de novas funcionalidades, resultando em economia significativa de tempo, custo e recursos. Foi proposta uma abordagem que não apenas aumenta a flexibilidade e adaptabilidade dos equipamentos eletrônicos, mas também estende sua vida útil e alinha o *design* do produto às necessidades específicas dos clientes. Esta abordagem representa um avanço substancial na criação de soluções eletrônicas sustentáveis, de menor consumo energético, mais eficientes e flexíveis.

PALAVRAS-CHAVE: Eletrônica programável. FPGA. *Soft Core*.

ABSTRACT

This article presents the evolutionary potential of FPGA (Field Programmable Gate Array) programmable logic devices in transforming the development and upgrading of electronic equipment. It focuses on implementing a Soft Core, a software-adaptable and configurable processing core that enhances system processing and flexibility. The aim is to explore the practical applicability of this technology by demonstrating the development of a system based on XILINX's Spartan 7 FPGA. The methodology involved using the VIVADO suite to configure programmable Logic Blocks, implementing a Soft Core, and the XILINX VITIS IDE, where firmware was developed in C. The main results obtained included the initial integration of a temperature sensor. Subsequently, its functionalities were expanded with the activation of a gyroscope sensor. The results were validated using DIGILENT's Analog Discovery 2 logic analyzer. The article concludes that adopting this technology can eliminate the need for physical revisions of electronic boards when new functionalities are required, resulting in significant time, cost, and resource savings. A proposed approach increases the flexibility and adaptability of electronic equipment, extends its lifespan, and aligns product design with specific customer needs. This approach represents a substantial advancement in creating sustainable electronic solutions that are more energy-efficient, flexible, and efficient.

KEYWORDS: *Programmable Electronics. FPGA. Soft Core.*

[1] COPPE/UFRJ.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

RESUMEN
Este artículo presenta el potencial evolutivo de los dispositivos lógicos programables FPGA (Field Programmable Gate Array) en la transformación del desarrollo y actualización de equipos electrónicos. Se centra en la implementación de un Soft Core, que es un núcleo de procesamiento adaptable y configurable por software, que potencia el procesamiento y la flexibilidad de los sistemas. El objetivo es explorar la aplicabilidad práctica de esta tecnología, demostrando el desarrollo de un sistema basado en la FPGA Spartan 7 de XILINX. La metodología consideró el uso de la suite VIVADO para configurar los Bloques Lógicos programables, la implementación de un Soft Core y el IDE VITIS de XILINX, donde se desarrolló un firmware en C. Los principales resultados obtenidos fueron la integración inicial de un sensor de temperatura. Posteriormente, sus funcionalidades fueron ampliadas con la activación de un sensor tipo giroscopio. Los resultados fueron validados con el uso del analizador lógico Analog Discovery 2 de DIGILENT. El artículo concluye que la adopción de esta tecnología puede eliminar la necesidad de revisiones físicas de las placas electrónicas cuando se requiere la adición de nuevas funcionalidades, resultando en un ahorro significativo de tiempo, costo y recursos. Se propuso un enfoque que no solo aumenta la flexibilidad y adaptabilidad de los equipos electrónicos, sino que también extiende su vida útil y alinea el diseño del producto con las necesidades específicas de los clientes. Este enfoque representa un avance sustancial en la creación de soluciones electrónicas sostenibles, de menor consumo energético, más eficientes y flexibles.

PALABRAS CLAVE*: Electrónica programable. FPGA. Soft Core.*

1 INTRODUÇÃO

Com o aumento da densidade de transistores nas pastilhas de silício, uma litografia com três nanômetros pode ter em média 50 bilhões de transistores em uma única pastilha de uma polegada. Os principais benefícios desta conquista tecnológica são a capacidade de processamento cada vez maior em oposição a um espaço, consumo energético e custo cada vez menor (Rieger, 2019). Com este aumento de densidade de transistores nas pastilhas de silício e a crescente adoção de componentes eletrônicos baseados em lógica programável, tem-se por consequência o incremento da capacidade computacional e a possibilidade de reconfiguração dos circuitos eletrônicos de forma bastante flexível, e até viabilizando recursos de computação acelerada sob demanda. A principal motivação é proporcionar um circuito compacto e otimizado que possa ser continuamente ajustado por *software*, em vez de ter que modificar o projeto de placas de circuito e demais componentes de equipamentos eletrônicos. Estas funcionalidades podem tornar as aplicações mais versáteis e escaláveis, permitindo que sensores de Internet das Coisas possam ser reprogramados e reconfigurados remotamente com mais recursos e funcionalidades sem a necessidade de substituição ou modificação da eletrônica (Aziz *et al.*, 2022).

A pergunta de pesquisa a ser respondida é se um microcontrolador com um ou mais núcleos, circuitos de processamento de sinais, entre outros componentes podem ser configurados em uma FPGA - *Field Programable Gate Array* através de um programa em linguagem específica sob demanda, conseguindo um resultado similar ou superior em relação a um circuito dedicado, construído especificamente para o mesmo fim? O principal conceito apresentado é o de *soft core*, que entre outras aplicações permite implementar por *software* um ou mais núcleos em uma FPGA. Outras facilidades podem ser configuradas, como adicionar novos componentes virtuais, embarcar uma rede neural ou outros recursos de Inteligência Artificial otimizada em um dispositivo IoT - *Internet of*

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Things instalado na borda de uma rede, com desempenho bem maior do que se estivesse implementada em um servidor de rede ou na nuvem, caso em que se demanda uma razoável capacidade de conectividade e banda disponível em ambos os casos para atender tal necessidade de computação centralizada (Beggs, 2021).

Alguns destes componentes podem ser uma combinação de lógica programável e um sistema completo em um único circuito integrado, o que é conhecido como SOC - *System On a Chip*, que pode funcionar inclusive como protótipo no desenvolvimento de circuitos integrados do tipo ASIC - *Application Specific Integrated Circuits*, que são componentes de alta densidade fabricados para aplicações específicas, tais como as GPUs, CPUs, MCUs entre outros componentes dedicados que necessitam de grande escala de produção para ter custos competitivos (Gundersen, 2019).

O objetivo geral deste projeto é apresentar todo o processo de configuração e implementação de recursos em uma FPGA com pelo menos um núcleo tipo *Soft Core* e demais componentes necessários na readequação de blocos lógicos programáveis. Os objetivos específicos consideram: (i) Montar o circuito apresentado em bancada; (ii) Implementar um microprocessador baseado em *software* - *Soft Core*; (iii) Conectá-lo a dois tipos de sensores: Um sensor de temperatura e um giroscópio (Faisal; Purboyo; Ansori, 2019); (vi) O último objetivo específico considera apresentar todo o sistema integrado em funcionamento, tendo alguns de seus sinais lógicos apresentados em um analisador lógico e os seus resultados de medições apresentados em tela.

Um dos maiores problemas com o uso de dispositivos eletrônicos é a rápida obsolescência dos equipamentos, o que culmina com um substancial acréscimo de lixo eletrônico e incremento dos seus consequentes danos ao meio ambiente. Apesar do lixo eletrônico ter o seu reaproveitamento cada vez mais frequente, o impacto causado ao meio ambiente na busca de novos elementos naturais para construir novos equipamentos também é alto. Utilizando recursos atuais de engenharia, pretende-se mitigar esta degradação ambiental com a criação de dispositivos com um maior tempo de vida e que possam ser reprogramados e reaproveitados em outras funções de forma mais inteligente (Lavez; De Souza; Leite, 2011). A adoção de dispositivos de eletrônica programável, a reciclagem de componentes, a reeducação ambiental e práticas sustentáveis de design em engenharia podem impactar significativamente na redução do lixo eletrônico e preservar o meio ambiente.

Devido ao aumento da densidade de transistores na litografia dos circuitos integrados e a consequente redução dos custos dos componentes, cada vez mais os engenheiros vêm desenvolvendo produtos com uma maior capacidade de expansão e reprogramação futura, o que chega ser antagônico com o modelo de negócio até então vigente, onde um produto é desenvolvido para durar por pouco tempo e tem o seu custo de produção menor possível para se obter ganhos em escala (Rossini; Naspolini, 2017). Um outro modelo de negócio vem sendo adotado, que pretende resolver este dilema, é a utilização de equipamentos sob demanda, tendo seu pagamento sendo feito de forma recorrente de acordo com o uso e funcionalidades, tornando a eletrônica programável ferramenta imprescindível neste novo modelo de negócio.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

2 FUNDAMENTAÇÃO TEÓRICA

O museu da história do computador em Palo Alto na Califórnia apresenta em seu acervo toda a evolução do que hoje conhecemos como computador moderno. Segundo Tanenbaum (Tanenbaum; Feamster; Wetherall, 2021), a arquitetura de computadores está organizada em seis gerações, de zero até a quinta: A geração zero, composta pelos computadores mecânicos, que se tornaram conhecidos na segunda guerra mundial como importante ferramenta para quebrar a criptografia utilizada pelo inimigo. (1642~1945); A primeira geração, onde estão classificados os computadores a válvulas (1945~1955); A segunda geração, marcada pela introdução dos materiais semicondutores de silício que compõem os transistores (1955~1965); A terceira geração marcada pelos circuitos integrados e pela tendência exponencial de aumentar a densidade de transistores integrados em um só componente (1965~1980); A quarta geração, que caracteriza a integração de transistores em escala muito grande - VLSI - *Very Large Scale Integration*, quando foram introduzidos os primeiros dispositivos baseados em lógica programável, as PLAs - *Programable Logic Arrays*. Foram introduzidas as unidades de processamento integradas em um único chip, que hoje conhecemos como microcontroladores e microprocessadores. Assim surgiram as primeiras arquiteturas CISC - *Complex Instruction Set Computer* e sua evolução, a RISC - *Reduced Instruction Set Computer.* A quarta geração se iniciou em 1980 e continua até os dias de hoje em ritmo evolutivo, e que continua a trazer avanços como a GUI - *Graphical User Interface*, processamento gráfico, processamento neural, entre outros feitos.

Ainda na década de 80, uma empresa recém-nascida, ainda hoje reconhecida como XILINX, desenvolveu uma técnica inteligente para montar circuitos integrados, que não exigia um grande investimento para se fabricar um componente com muita integração. Nascia então um novo componente, inspirado no conceito da antiga PLA, mas com uma quantidade muito maior de portas lógicas genéricas, incluindo outros tipos de circuitos programáveis, tais como *flip flops*, conversores digitais, processadores de sinais, entre outros componentes que caracterizaram o que se tornou conhecido como FPGA - *Field Programable Gate Array* (Trimberger, 1993). Desde então o *hardware* passou a ser tão maleável quanto o *software*. Esta década ainda foi marcada com o surgimento dos computadores RISC de 64 bits, que marcou a tendência de uma maior capacidade de processamento, em um menor espaço e maior eficiência energética.

A contemporaneidade nos trouxe a quinta geração, vigente em paralelo com a quarta, mas partindo de uma concepção de computação ubíqua, que compactou os computadores, melhorou a eficiência térmica e energética, que somado a evolução no desenvolvimento das baterias, permitiu que a computação pudesse estar presente em todo lugar. Desde um computador de mesa, portátil ou dispositivos até então conhecidos somente nos livros, filmes e desenhos de ficção científica (Duarte, 2021). Surgiram então dispositivos como *smartphones, smart watches*, sensores industriais e vestíveis (Patel, 2012), as GPUs - *Graphic Processor Unit* (Lindholm, 2008), que caracterizam a computação acelerada, entre outras tecnologias que podem contar com recursos nativos e embarcados de Inteligência Artificial, inaugurando uma tendência denominada computação em borda

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

- *Edge Computing* (Shi, 2016). Alguns autores já reconhecem uma sexta geração, iniciada com a chegada dos primeiros computadores quânticos (Fu, p. 325, com adaptações), que tem se mostrado altamente eficientes na solução de equações diferenciais complexas como as FNOs - *Fourier Neural Operator*, que em CFD - *Computer Fluid Dynamics*, tem mostrado melhores resultados em menor tempo que as tradicionais equações de *Navier-Stokes*, mesmo quando comparados com os cenários de computação tradicional centralizada ou distribuída, computação acelerada baseada em GPUs e computação quântica, este mesmo simulada, com desempenho bem superior.

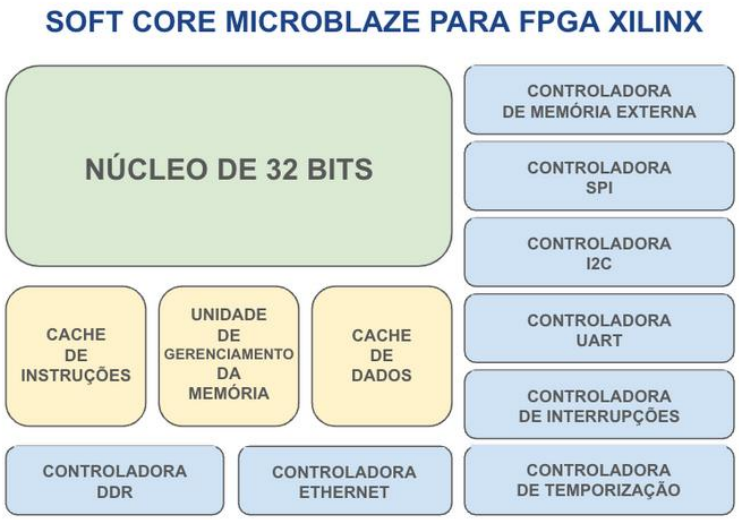
Os dispositivos de lógica programável, tais como FPGA e FPAA - *Field Programable Analog Array* permitem a utilização de blocos em formato de programas que podem ser adquiridos do próprio fabricante do componente ou ainda de várias empresas especializadas no mercado. Estes blocos são conhecidos como IP - *Intelectual Property* e podem ser adicionados em novos designs de produto a fim de ganhar tempo e eficiência no desenvolvimento de novos equipamentos. Neste artigo, foi utilizado a IP de um microprocessador, fornecido sem custos pelo fabricante da FPGA, que vem sendo aperfeiçoado nos últimos dez anos e hoje representa umas das melhores alternativas de *Soft Cores* no mercado. Este recurso permite eliminar do fluxo de trabalho, a necessidade de se programar do zero uma unidade de processamento. Com os recursos de IPs disponíveis para aquisição no mercado, mesmo dispositivos eletrônicos avançados, como radares, Lidars - *Light Detection and Ranging*, arranjos complexos de modulação em radiofrequência como *Beamforming*, tratamento de dados de sensores hiper espectrais, entre outras aplicações, podem ser otimizadas e ter sua prototipação em menor tempo. Estes componentes programáveis ainda possuem recursos de proteção da propriedade intelectual desenvolvida, dificultando a cópia não autorizada do que foi desenvolvido e utilizado, além de contar com classes de componentes dedicados a missões críticas, como o uso em aplicações aeroespaciais tolerantes a radiação (Rockett *et al*., 2007, p. 2).

2.1. Microprocessadores baseados em *software*

Neste artigo foram considerados os *Soft Cores* disponíveis para a plataforma XILINX Spartan 7: ARM Cortex M1, ARM Cortex M3, RISC V e Microblaze. Inicialmente optou-se pelo uso do *Soft Core* ARM M0, no entanto, devido ao tempo que este *Soft Core* está disponível e sem atualizações significativas, implica na utilização de uma plataforma de desenvolvimento obsoleta, o que trouxe inúmeras incompatibilidades com o sistema operacional Windows 11. Optou-se então pelo *Soft Core* XILINX Microblaze, que é um microprocessador com instruções RISC (Figura 1), atualizado e compatível com a versão atual da plataforma de desenvolvimento, o que acelerou o trabalho.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 1 — Diagrama em blocos da arquitetura do Soft Core Microblaze da XILINX



Fonte: O autor (2024)

2.2 Plataforma de desenvolvimento FPGA E IDE

Visando a melhor integração e experiência na elaboração deste artigo, foi adotada a plataforma de desenvolvimento VIVADO 2021.1 e a IDE - *Integrated Development Environment* VITIS 2021.1 da XILINX para o sistema operacional Windows 11.

2.3 Instrumentos e ferramentas

A placa de desenvolvimento utilizada é a Digilent Arty S7-50 com suas características apresentadas no Quadro 1 e sua imagem apresentada na Figura 2.

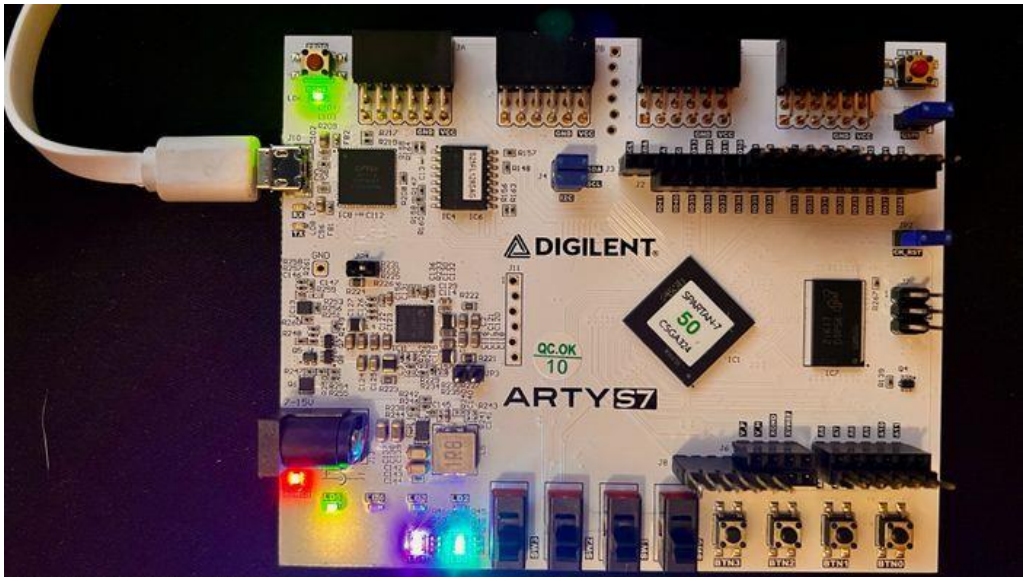
Quadro 1 — Tabela de recursos da placa de desenvolvimento DIGILENT ARTY S7-50

RECURSOS	QUANTIDADE
CLBs - Blocos Lógicos Configuráveis	8.150
LUTs (*Look Up Table*) por CLB	4
Flip Flops	65.200
DSPs - Processadores Digitais de Sinais em blocos	120
Memória tipo RAM da FPGA	337 Kb
Memória tipo RAM DDR3L da placa ARTY S7	256 Mb
Memória tipo FLASH QUAD SPI da placa ARTY S7	16 Mb
PMODs - Módulos de Expansão para Periféricos	4

Fonte: Adaptado de https://www.digilent.com

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

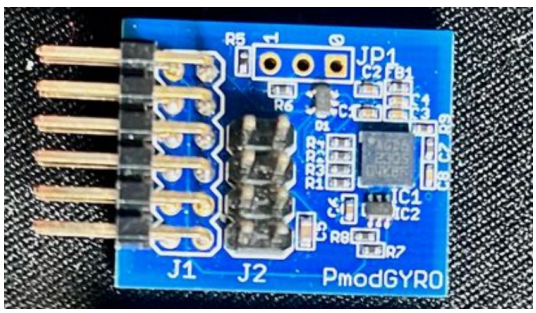
Figura 2 — Placa de desenvolvimento Digilent S7-50



Fonte: O autor (2024)

O PMOD com os sensores tipo Giroscópio e Temperatura, conforme ilustrado na Figura 3 foi adicionado no conector PMOD JA da placa ARTY S7.

Figura 3 — Módulo PMOD com sensores de temperatura e giroscópio



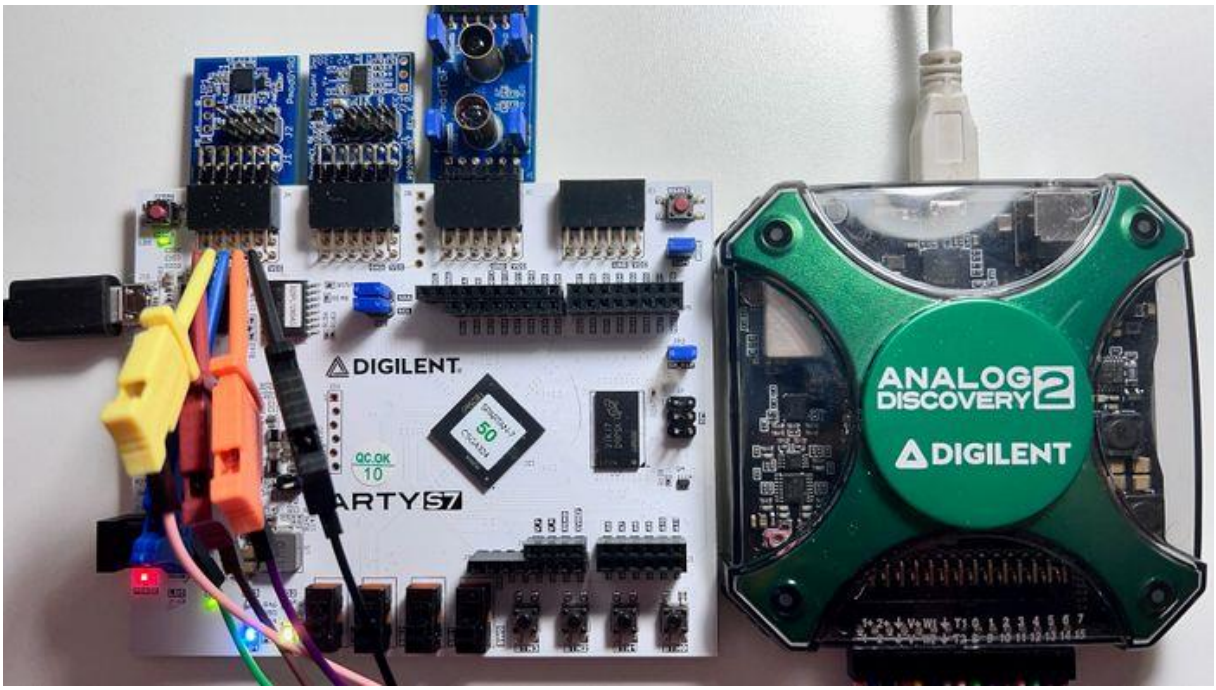
Fonte: O autor (2024)

.2.3.1 Analisador lógico Digilent Analog Discovery 2

O analisador lógico utilizado foi o Digilent Analog Discovery 2, apresentado na Figura 4, que além da funcionalidade de analisador lógico de 16 canais com decodificação de SPI, I2C, UART, CAN, I2S, 1Wire, HDMI CEC, entre outros protocolos, disponibiliza funcionalidades de Osciloscópio duplo traço de 30Mhz, gerador de formas de onda, analisador de espectro, analisador de impedâncias, voltímetro, *data logger*, gerador de padrões, entre outras. O equipamento é utilizado com o *software* Waveforms da Digilent.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 4 — Analisador Digilent Analog Discovery 2 em uso



Fonte: O autor (2024)

3 MÉTODO

O método utilizado contempla a pesquisa de natureza aplicada, tendo seu problema abordado de forma quantitativa, e no que couber qualitativa, incluindo a investigação da bibliografia, culminando com a montagem de um circuito eletrônico em bancada, desenvolvimento de código específico e respectivos resultados.

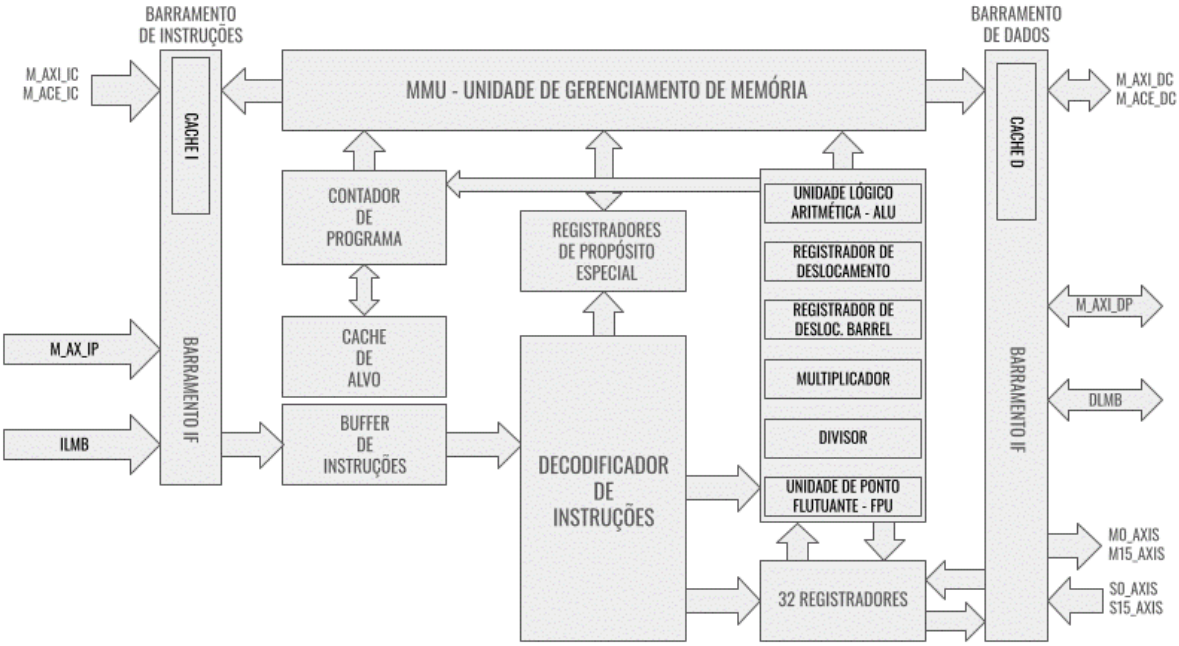
Inicia-se a partir da utilização de uma FPGA para se chegar a um SOC - *System on a Chip* personalizado, definido por *software*. Utilizaremos alguns blocos já pré-definidos (IPs), tais como *Soft Cores*, temporizadores, controladores de I/O, entre outros encontrados em um SOC de mercado. Entre os procedimentos técnicos pretende-se apresentar uma pesquisa prática, mas sem desconsiderar os aspectos empíricos, de forma experimental em laboratório. As atividades desenvolvidas, sua sequência e interdependência estão relatadas a seguir.

3.1 Implementação do Processador Soft Core Microblaze

É um microprocessador em *software* de arquitetura RISC Harvard de 32/64 bits para as FPGAs da XILINX. Suporta opções avançadas de arquitetura, como unidades de gerenciamento de memória, cache de instruções e dados, Unidade de ponto flutuante, entre outras. Na Figura 5 podemos visualizar um diagrama funcional do núcleo Microblaze.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 5 — Diagrama funcional do Núcleo MicroBlaze



Fonte: O autor (2024)

O microprocessador MicroBlaze geralmente é utilizado em uma das seguintes configurações pré-definidas: (I) Microcontrolador simples executando aplicações desenvolvidas em C nativamente (*Baremetal*); (II) Microprocessador em tempo real com cache e com uma unidade de proteção de memória e memória integrada acoplada executando FreeRTOS; (III) Microprocessador de aplicativos com unidade de gerenciamento de memória executando LINUX. Neste artigo foi implementado um núcleo MicroBlaze tipo *Baremetal*, um módulo de RAM e três PMODs.

Após baixar e instalar a suíte VIVADO e a IDE VITIS da XILINX, com as respectivas bibliotecas, foi criado um projeto VIVADO para configurar a FPGA com o microprocessador com algumas entradas e saídas específicas. Então um *design* em blocos foi criado com o IP *Integrator*, que é um módulo do VIVADO utilizado para gerenciar as propriedades intelectuais - IP. Atribui-se um nome ao *design* de blocos e adiciona a IP do processador MicroBlaze ao diagrama de blocos.

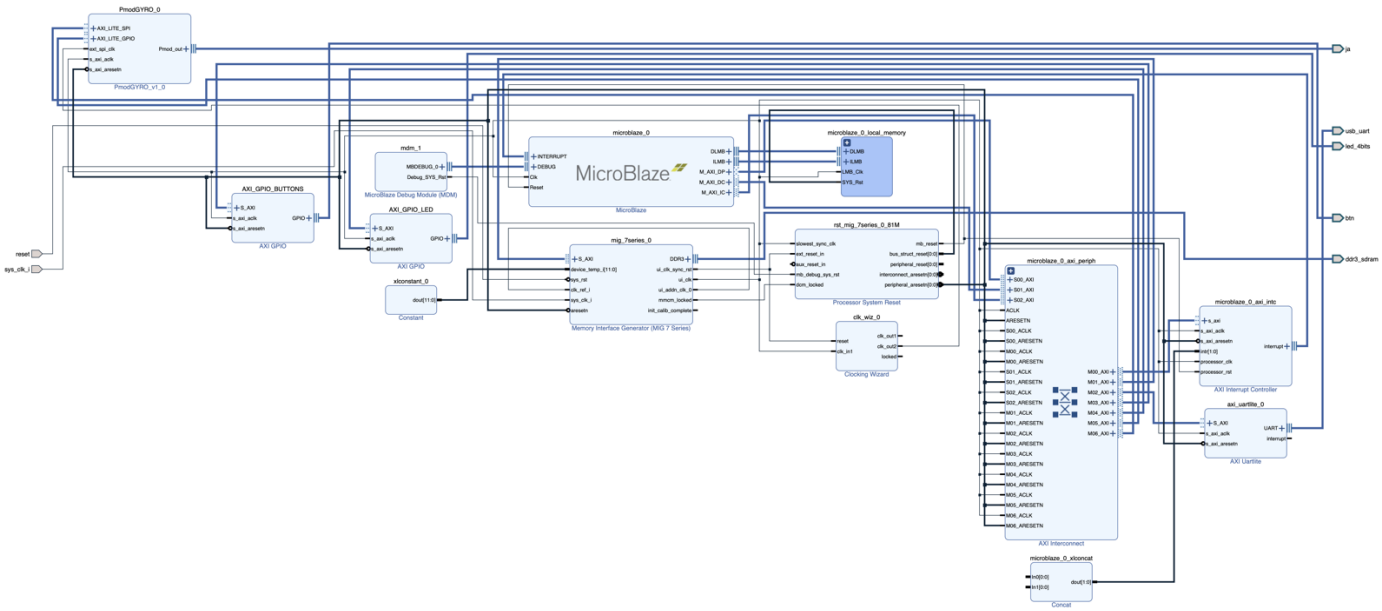
Ao criar o projeto com o uso de memória RAM DDR - *Random Access Memory Double Data Rate*, deve-se adicionar primeiro a *interface* DDR com a opção de conexão automática ao diagrama. Com este processo é adicionada uma MIG - *Memory Interface Generator* e uma *interface* DDR externa ao *design* e dois pinos de *Clock*, que foram modificados posteriormente.

O Arquivo mestre XDC - *Xilink Design Constraints* para placas Digilent deve ser baixado do site da Digilent. Ele contém as restrições que a placa utilizada impõe aos projetos que a utilizam, tais como conexão a pinos específicos, frequências de *Clock etc.* É importante conferir as conexões e fazer alguns ajustes para que o circuito funcione adequadamente, principalmente a frequência de *Clock* das *interfaces* PMOD.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

A IP do microprocessador MicroBlaze foi adicionada ao *design*. A execução da automação de bloco tornou-se imperiosa para que a infraestrutura adicional necessária seja adicionada ao projeto. Neste momento foi definido como o IP do MicroBlaze se conecta aos demais componentes do projeto.

Figura 6 — Diagrama esquemático do projeto conectado ao PMOD com Giroscópio e sensor de temperatura no canto superior esquerdo



Fonte: O autor (2024) - Disponibilizado em:
https://github.com/nsduarte/FPGA_TCC24/blob/main/baremetal_diagrama.pdf

3.2 Conectando Periféricos ao Diagrama de Blocos

Para que o *firmware* possa imprimir os dados em um console serial é necessário que um periférico UART esteja conectado no circuito e previsto em seu diagrama. Para fazer as conexões que ainda não foram conectadas ao microprocessador, foi executada a opção do VIVADO *Run Connection Automation* na barra verde do assistente de *design* do projeto para conectá-los automaticamente. O diagrama esquemático do projeto é apresentado na Figura 6.

3.3 Validação Adicionando Periféricos Gpio ao Design De Blocos

Para adicionar periféricos existem as seguintes opções: (I) Usar arquivos da placa para gerar as restrições automaticamente; (II) IP Conectado aos pinos escolhidos manualmente. Seguimos com a segunda opção por ser mais flexível. Como teste, os leds da placa foram conectados para fazer um teste preliminar de funcionamento, que se demonstrou adequado. Como os arquivos da placa não foram utilizados, é necessário adicionar um arquivo XDC - *XILINX Design Constraints* para informar ao VIVADO quais os pinos da FPGA devem ser conectados a *interface*.

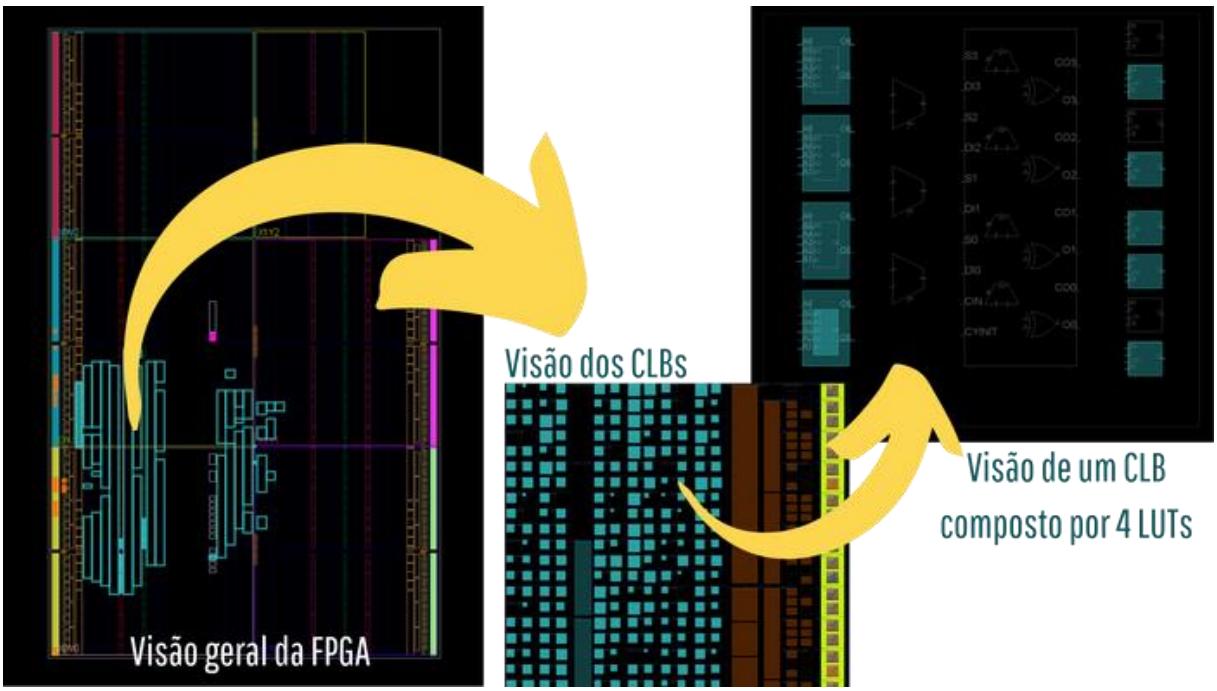
RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

O próximo passo é revisar o mapa de endereços, na opção do editor de endereços do VIVADO, passando também pela opção de mapa de endereçamento para visualizar a alocação das áreas. Caso o VIVADO não atribua os endereços adequadamente, este processo deverá ser corrigido manualmente com o cuidado de não sobrepor os intervalos.

A seguir, o diagrama de blocos deve ser validado. Nesta etapa é executada uma verificação automática do diagrama para conhecer algum problema potencial que comprometa o projeto. Se problemas forem encontrados, eles serão retratados em uma caixa de diálogo, especificamente no painel inferior. Com o diagrama validado, gera-se o *Wrapper* HDL para o *design* de blocos. Este processo traduz o *design* de blocos em um arquivo fonte padrão .bd para ser lido pelo VIVADO e para construir o *design* real.

Constrói-se o projeto com a geração do arquivo tipo *bitstream* selecionando as opções de como a síntese e a implementação devem ser executadas. Esta fase é de extrema importância para conhecer a necessidade de recursos necessários para o funcionamento do sistema. O processo de síntese gera um arquivo formato xsa, que contém todos os parâmetros de *hardware* que pode ser usado para criar bibliotecas, simular o funcionamento do circuito, visualizar o arranjo das LUTs e CLBs implementados no *hardware*, conforme ilustrado na Figura 7, entre outras funções.

Figura 7 — Visão sobre o arranjo dos blocos da FPGA, ampliados para visualizar os Blocos, os CLBs e as LUTs



Fonte: O autor (2024)

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

3.4 Exportando a Plataforma Fixa de *Hardware* Pós Síntese

Uma vez o projeto construído e validado, o *design* deve ser exportado do Vivado para que possa ser utilizado pelo VITIS. Com isso a IDE do VITIS terá acesso as informações do *hardware* para o qual um *firmware* será desenvolvido. Exportando o *hardware* após a geração do *bitstream* permite que a placa seja programada diretamente no IDE do VITIS. Para iniciar este processo, basta selecionar a opção de exportar *hardware* no menu do VIVADO. Neste caso a exportação deve considerar a opção de inclusão do *Bitstream* para que a placa possa ser programada. O arquivo resultante deste processo é o .xsa - *XILINX Shell Architecture*, gravado no diretório mais conveniente.

3.5 Desenvolvimento do *Firmware*

Com o VITIS iniciado, cria-se um projeto de aplicação, importa-se o arquivo .xsa e define-se algumas opções do projeto. Vale ressaltar que um projeto de sistema pode conter vários projetos de aplicativos, que podem ser executados ao mesmo tempo. Foi escolhida a opção de escrever programa em C e o da criação do projeto concluída. Então começa o desenvolvimento do programa em C na IDE.

Um aplicativo VITIS foi criado com todas as fontes e dependências necessárias (*Build*) e todas as definições de *hardware* e *firmware* foram transferidas para a placa ARTY S7-50.

4 RESULTADOS E DISCUSSÃO

Após o processo de montagem do circuito em bancada, que seguiu com a implementação de recursos e configuração de um núcleo de processamento tipo *Soft Core*, a visualização dos blocos lógicos utilizados, a conexão a um sensor giroscópio e o desenvolvimento do *firmwar*e, são apresentados os resultados de todo o sistema em funcionamento.

A configuração da FPGA foi efetuada sem a necessidade de desenvolver um código específico em VHDL ou VERILOG, o que simplificou bastante o desenvolvimento deste projeto. Embora a suíte de *software* permita o desenvolvimento de códigos para a definição e configuração da FPGA, a automação do processo deixa o trabalho mais rápido e eficiente. O uso de PIs também facilitam muito o processo de configuração, embora algumas conexões tenham que ser efetuadas manualmente. Mesmo assim, as tarefas ficam mais rápidas do que descrever o *hardware* em uma linguagem específica. Outro benefício é a possibilidade de minimizar os erros, que demandariam muito tempo para a análise dos problemas apresentados.

Entre as lições aprendidas no projeto, a mais importante foi a correta configuração do *clock* dos periféricos. Demandou-se um tempo significativo para pesquisar e descobrir por que o sensor PMOD conectado através de SPI não funcionava. Com o analisador lógico, foi descoberto que o *clock* padrão da placa ARTY S7 para os periféricos era de 100 Mhz. Tornou-se necessário adicionar a PI do assistente do *clock* para dividi-lo por dois e disponibilizar na SPI do periférico uma frequência de 50Mhz. Com a frequência correta o PMOD passou a funcionar adequadamente.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Com as configurações de *hardware* implementadas e testadas com os leds existentes na placa ARTY S7, iniciou-se a segunda fase do processo que foi a implementação do periférico PMOD, o que demorou um pouco mais até achar a solução para o problema de *clock*. Uma vez resolvido, partiu-se para o desenvolvimento de um código em C para fazer a leitura do PMOD e apresentar na tela do sistema. Este código é apresentado na figura 8 e depois de compilado, é implementado na placa.

Figura 8 — Código em C para ler os dados dos sensores e apresentá-los na tela

```
/************ Includes ************/

#include "PmodGYRO.h"
#include "sleep.h"
#include "xil_cache.h"
#include "xil_printf.h"


/************ Funções ************/

void INICIAR();

void RUN();

void LIMPAR();

void ATIVAR_CACHE();

void DESATIVAR_CACHE();

/************ Variáveis Globais ************/

PmodGYRO sensor;

/************ Aplicação ************/

int main() {
   INICIAR();
   RUN();
   LIMPAR();
   return 0;
}

void INICIAR() {
   ATIVAR_CACHES();
   GYRO_begin(&sensor, XPAR_PMODGYRO_0_AXI_LITE_SPI_BASEADDR,
         XPAR_PMODGYRO_0_AXI_LITE_GPIO_BASEADDR);

   // Configurar Threshold Registers
   GYRO_setThsXH(&sensor, 0x0F);
   GYRO_setThsYH(&sensor, 0x0F);
   GYRO_setThsZH(&sensor, 0x0F);

   GYRO_enableInt1(&sensor, GYRO_INT1_XHIE);    // Threshold interrupt
   GYRO_enableInt2(&sensor,GYRO_REG3_I2_DRDY); // Data Rdy/FIFO interrupt
}

void RUN() {
   int16_t xAxis = 0;
   int16_t yAxis = 0;
   int16_t zAxis = 0;
   int8_t temp = 0;
   int trig = 0;

//   print("Iniciando...\n\r");
   while (1) {
      usleep(500000);

      if (GYRO_Int1Status(&sensor) != 0) {
         xil_printf("\x1B[2J");
         xil_printf("\x1B[H");
//         xil_printf("Threshold alcançado\n\r");
```

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218


```
            trig = 1;
         }
         if (GYRO_Int2Status(&sensor) != 0) {
            if (trig == 1) {
               trig = 0;
            } else {
               xil_printf("\x1B[2J"); // Clear screen
               xil_printf("\x1B[H");  // Reset cursor to 0,0
            }
//            xil_printf("Lendo dados..\n\r\n\r");

            xAxis = GYRO_getX(&sensor);
            yAxis = GYRO_getY(&sensor);
            zAxis = GYRO_getZ(&sensor);
            temp = GYRO_getTemp(&sensor);

//            temp = 0;
            temp = (temp - 32) * 5/9;

//            xil_printf("X Axis: 0x%04x\n\r", xAxis & 0xFFFF);
//            xil_printf("Y Axis: 0x%04x\n\r", yAxis & 0xFFFF);
//            xil_printf("Z Axis: 0x%04x\n\r", zAxis & 0xFFFF);

            xil_printf("X Axis: %d\n\r", xAxis);
            xil_printf("Y Axis: %d\n\r", yAxis);
            xil_printf("Z Axis: %d\n\r", zAxis);

            xil_printf("\n\r");

            xil_printf("Temperatura: %d deg C\n\r", temp);
         }
      }
   }

   void LIMPAR() {
      GYRO_end(&myDevice);
      DESATIVAR_CACHES();
   }

   void ATIVAR_CACHES() {
   #ifdef __MICROBLAZE__
   #ifdef XPAR_MICROBLAZE_USE_DCACHE
      Xil_DCacheEnable();
   #endif
   #ifdef XPAR_MICROBLAZE_USE_ICACHE
      Xil_ICacheEnable();
   #endif
   #endif
   }

   void DESATIVAR_CACHES() {
   #ifdef __MICROBLAZE__
   #ifdef XPAR_MICROBLAZE_USE_DCACHE
      Xil_DCacheDisable();
   #endif
   #ifdef XPAR_MICROBLAZE_USE_ICACHE
      Xil_ICacheDisable();
   #endif
   #endif
   }
```

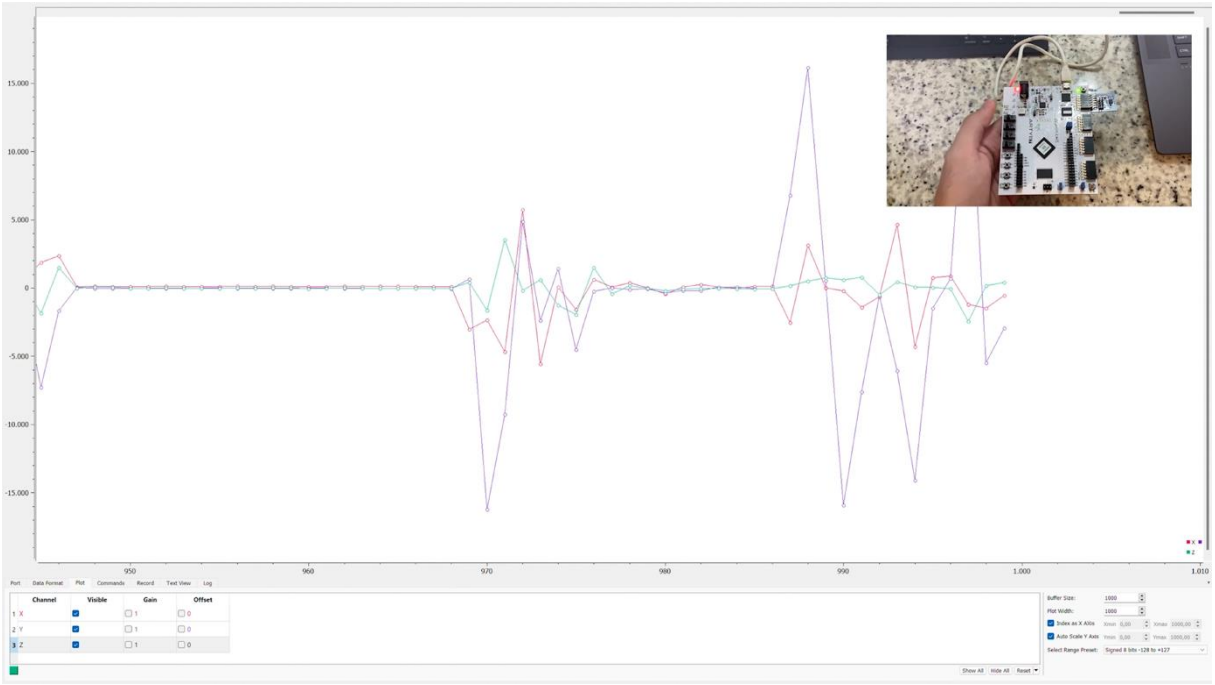
Fonte: O autor (2024) - Disponibilizado em: https://github.com/nsduarte/FPGA_TCC24/blob/main/tcc/src/main.c

Para iniciar o sistema e começar a medir os resultados, torna-se mandatório a utilização de um cabo serial para conectar a placa de desenvolvimento de um lado e um computador do outro, utilizando o *software* Tera Term para emular um terminal serial. Depois de tudo configurado e com o *software* carregado, visualiza-se na tela do computador conectado os resultados das medidas em tempo real de temperatura e do funcionamento do giroscópio, que varia suas grandezas expressas em mg (mili gravidades) de acordo com o movimento executado.

Para melhor ilustrar os resultados, utilizamos um PC com o *software Serial Plot* para fazer a aquisição dos dados em tempo real e apresentá-los graficamente, conforme apresentado na figura 9 e disponibilizado no vídeo em: https://youtu.be/LY_OUKA1Y_c

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 9 — Representação gráfica dos dados adquiridos em tempo real



Fonte: O autor (2024) – Disponibilizado em: https://youtu.be/LY_OUKA1Y_c

Pretendia-se explorar neste trabalho outros PMODs como o ToF - *Time of Flight* e Acelerômetro, mas os módulos chegaram danificados e o tempo para substituição dado não era compatível com o cronograma deste projeto. Foram avaliadas as duas maiores plataformas para desenvolvimento de FPGA do mercado: INTEL ALTERA e AMD XILINX.

Com os resultados apresentados e disponibilizados em <https://github.com/nsduarte/FPGA_TCC24>, os objetivos gerais e específicos previstos foram atingidos, ilustrando todo o processo de configuração e implementação de recursos em uma FPGA com pelo menos um núcleo tipo *Soft Core* e demais componentes necessários na readequação de blocos lógicos programáveis.

5 CONSIDERAÇÕES

Conclui-se que um microcontrolador com um ou mais núcleos, circuitos de processamento de sinais, entre outros componentes, podem ser configurados em uma FPGA - *Field Programable Gate Array* através de um programa em linguagem específica e sob demanda. O funcionamento do sistema é muito similar a um processador físico - *Hard Core*, construído para o mesmo fim, mas com flexibilidade e escalabilidade muito superior, permitindo a inclusão de novas funcionalidades, novos núcleos e até a reconfiguração de todo *hardware* embarcado através de *software* específico ou de forma interativa na plataforma de implementação, conforme apresentado neste artigo.

A adoção da tecnologia apresentada pode eliminar a necessidade de revisões físicas das placas eletrônicas diante da necessidade de adição de novas funcionalidades, resultando em

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

economia significativa de tempo, custo e recursos. Foi proposta uma abordagem que não apenas aumenta a flexibilidade e adaptabilidade dos equipamentos eletrônicos, mas também amplia os benefícios para a sociedade, estendendo a vida útil e alinhando o *design* do produto às necessidades específicas dos clientes. Esta abordagem representa um avanço substancial na criação de soluções eletrônicas sustentáveis, de menor consumo energético, mais eficientes e flexíveis.

Os conceitos apresentados neste artigo podem ser futuramente aperfeiçoados e explorados em tópicos complementares e enriquecidos com recentes tecnologias e aplicações, tais como *Open Hardware* (IEEE - Computer Society, 2023, p. 17-19), RISC-V, processamento *Multi Core* (Gray, 2016, p. 17-20), Inteligência Artificial nativa embarcada, podendo resultar em produtos cada vez mais sustentáveis e duráveis, otimizando os investimentos e principalmente os recursos naturais.

REFERÊNCIAS

AZIZ, Syed Mahfuzul *et al*. Remote reconfiguration of FPGA-based wireless sensor nodes for flexible Internet of Things. **Computers and Electrical Engineering**, v. 100, 31 mar. 2022. doi.org/10.1016/j.compeleceng.2022.107935.

BEGGS, Arthur de Natos. **Projeto e Desenvolvimento de processadores RISC-V**: em FPGA. Brasilia, 2021. Trabalho de Conclusão de Curso (Engenharia Mecatrônica) - Universidade de Brasília, Brasilia, 2021.

DUARTE, Newton Silva. A Dinâmica Ética das Máquinas em Inteligência Artificial: O Novo Papel do Professor de Filosofia. **Revista Ibero-Americana de Humanidades, Ciências e Educação**, v. 7, n. 6, p. 968-995, 2021. doi.org/10.51891/rease.v7i6.1446, 2021.

FAISAL, I. Arun; PURBOYO, T. Waluyo; ANSORI, A. A Review of accelerometer sensor and gyroscope sensor in IMU sensors. **J. Eng. Appl. Sci**, v. 15, n. 3, p. 826-829, 2019.

FU, Xiang. A heterogeneous quantum computer architecture. *In:* **PROCEEDINGS OF THE ACM INTERNATIONAL CONFERENCE ON COMPUTING FRONTIERS**. 2016, p. 323-330.

GRAY, I. GRVI Phalanx: A Massively Parallel RISC-V FPGA Accelerator. *In:* **IEEE 24TH ANNUAL INTERNATIONAL SYMPOSIUM ON FIELD-PROGRAMMABLE CUSTOM COMPUTING MACHINES (FCCM)**. 2016, doi: 10.1109/FCCM.2016.12.: IEEE, p. 17-20.

GUNDERSEN, Martin. **Analyzing an FPGA Neural Network Accelerator Design for Implementation in an ASIC**. 2019 Tese (Doutoado) - Faculty of Information Technology and Electrical Engineering, Ntnu - Norwegian University Of Sciences And Technology, 2019.

HOLLER, R *et al*. Open-Source RISC-V Processor IP Cores for FPGAs: Overview and Evaluation. *In:* **8TH MEDITERRANEAN CONFERENCE ON EMBEDDED COMPUTING (MECO), BUDVA, MONTENEGRO**, 2019. doi: 10.1109/MECO.2019.8760205, p. 1-6.

IEEE. Open Hardware: From open systems (OCP) to ISAs (RISC-V) and interconnects (CXL, UCIe) the open-source movement will expand into hardware. **Technology Predictions**, p. 24, 2023.

LAVEZ, Natalie; DE SOUZA, Vivian Mansano; LEITE, Paulo Roberto. O papel da logística reversa no reaproveitamento do “lixo eletrônico”: um estudo no setor de computadores. **Revista de Gestão Social e Ambiental**, v. 5, n. 1, 2011.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

LINDHOLM, Erik, et al. "NVIDIA Tesla: A unified graphics and computing architecture." **IEEE Micro**, v. 28, n. 2, p. 39-55, 2008. doi:10.1109/MM.2008.31.

PATEL, Shyamal, et al. "A review of wearable sensors and systems with application in rehabilitation." **Journal of Neuro Engineering and Rehabilitation**, v. 9, n. 1, p. 1-17, 2012. doi:10.1186/1743-0003-9-21.

RIEGER, Michael L. Retrospective on VLSI value scaling and lithography. **Journal of Micro/Nanolitography, MEMS, and MOEMS**, v. 18, n. 4, 26 nov. 2019. doi: 10.1117/1.JMM.18.4.040902.

ROCKETT, Leonard et al. Radiation-hardened FPGA technology for space applications. *In:* **IEEE AEROSPACE CONFERENCE**, p. 1-7, 2007.

ROSSINI, Valéria; NASPOLINI, S. H. D. F. Obsolescência programada e meio ambiente: a geração de resíduos de equipamentos eletroeletrônicos. **Revista de direito e sustentabilidade**, v. 3, n. 1, p. 51-71, 2017.

SHI, Weisong, et al. "Edge computing: Vision and challenges." **IEEE Internet of Things Journal**, v. 3, n. 5, p. 637-646, 2016. doi:10.1109/JIOT.2016.2579198.

TANENBAUM, Andrew; FEAMSTER, Nick; WETHERALL, David. **Redes de Computadores**. Porto Alegre: Bookman Editora, 2021. 796 p.

TRIMBERGER, Stephen M. Field-Programmable Gate Array Technology. **Proceedings of the IEEE**, v. 81, n. 7, p. 1031-1040, 1993. doi:10.1109/5.231342.