Como citar este artículo: Alves, Gabriel Sá B. and Bittencourt, João Carlos N. and Dias, Anfranserai, "An FPGA Integrated 2D Graphic Processor for Enhanced Digital Design and Computer Architecture Education" in IEEE Revista Iberoamericana de Tecnologias del Aprendizaje. Volume 19. 2024. doi: 10.1109/RITA.2024.3458850

Um Processador Gráfico 2D Integrado a FPGA para Aprimorar o Ensino de Projeto Digital e Arquitetura de Computadores

Gabriel Sá B. Alves, João Carlos N. Bittencourt, *Member, IEEE,* and Anfranserai Dias

***Abstract*— Diante da rápida evolução da educação tecnológica, a integração de sistemas digitais e arquitetura de computadores em um currículo acadêmico unificado tornou-se cada vez mais importante. Essa integração reflete a natureza dinâmica do cenário tecnológico e está alinhada com as crescentes demandas e expectativas dos alunos que buscam competências em ambas as disciplinas. Nesse contexto, há uma necessidade urgente de ferramentas pedagógicas inovadoras e revolucionárias que estejam de acordo com as tendências educacionais atuais e atendam de forma eficaz aos diferentes estilos de aprendizagem dos alunos. Consequentemente, a adoção de abordagens de co-design de FPGA (Field Programmable Gate Array), que integram aspectos de hardware e software de sistemas, representa um avanço significativo para atender às demandas contemporâneas do ensino de arquitetura de computadores, oferecendo uma solução prática, envolvente e dimensionável. Este artigo apresenta o CoLenda, uma nova ferramenta de aprendizado, por meio da implementação de uma plataforma FPGA educacional baseada em um processador gráfico 2D. O objetivo do CoLenda é fornecer uma compreensão abrangente de sua arquitetura, permitindo que os alunos vivenciem um aprendizado prático e prático por meio do desenvolvimento de jogos baseados em gráficos. A introdução desse processador aprimora o processo de aprendizado, permitindo que os alunos se envolvam em projetos digitais e em currículos de arquitetura de computadores, ao mesmo tempo em que estimula a criatividade e as habilidades práticas de solução de problemas. Além disso, tornar o projeto de código aberto para contribuições da comunidade poderia expandir significativamente o potencial e o escopo dessa ferramenta educacional.**

***Index Terms*— game-based, educational tool, hands-on learning, teaching technology**

I. Introdução

A integração de hardware e software tornou-se um requisito essencial no cenário atual de rápida evolução da tecnologia e do setor. De fato, essa integração é impulsionada pela busca de soluções eficientes, flexíveis e inovadoras para desafios tecnológicos complexos. Esse paradigma disruptivo levou ao desenvolvimento de dispositivos inteligentes e sistemas de IA, em que a interação perfeita entre hardware e software é essencial para a funcionalidade e a evolução [1]. Essa sinergia não apenas transformou as práticas do setor, mas também exigiu uma mudança paralela nas abordagens educacionais, principalmente nos currículos de design digital e arquitetura de computadores [2]-[4]. À medida que os engenheiros enfrentam essas demandas dinâmicas do setor por dispositivos inovadores, sua base educacional deve fornecer percepções teóricas sólidas e habilidades práticas em tecnologias de integração e co-projeto de hardware-software para prepará-los para os desafios do ambiente tecnológico atual.

Juntamente com os conceitos tecnológicos fundamentais, um entendimento completo da abstração de interfaces de hardware e comunicação é essencial para o desenvolvimento de dispositivos digitais [5]. A abstração de hardware é uma técnica que permite a integração perfeita do sistema, geralmente facilitada por meio de APIs (Interfaces de Programação de Aplicativos) bem definidas [6]. Uma API permite que o software interaja com o dispositivo de forma padronizada, mascarando a complexidade das interfaces de comunicação subjacentes. Normalmente implementadas por meio de barramentos de interconexão, as interfaces de comunicação eficientes são essenciais para garantir a troca robusta de informações entre sistemas diferentes [7]. Embora sejam cruciais para o setor, elas também constituem a base do currículo de design digital e arquitetura de computadores. Para isso, ferramentas educacionais inovadoras estão sendo desenvolvidas para fornecer aos alunos esse conhecimento especializado [8]-[10]. Ao combinar a compreensão teórica com a aplicação prática, essas ferramentas tornam as abstrações complexas tangíveis para os alunos, preenchendo a lacuna entre os conceitos avançados e sua implementação prática.

Reconhecendo a necessidade de estratégias de aprendizado aprimoradas, as universidades incorporaram dispositivos FPGA (*Field Programmable Gate Array*) em seus currículos para desenvolver habilidades de engenharia de hardware. Esses dispositivos configuráveis possibilitam experiências práticas com hardware, software e técnicas de integração, permitindo a simulação e a validação de projetos de circuitos digitais [11]. Os FPGAs são versáteis por natureza e fornecem os meios necessários para criar ferramentas de aprendizado que permitem que os alunos experimentem soluções em uma variedade de domínios [12]-[14]. Essas atividades incluem o desenvolvimento de tarefas práticas que garantem a assimilação de técnicas de design e a organização prática da arquitetura de sistemas digitais. No entanto, a interação com componentes complexos é um desafio que exige muitas etapas e conhecimento prévio, o que gera frustração por parte do aluno.

Uma alternativa promissora para aproveitar essas experiências de aprendizado baseadas em FPGA no ensino de engenharia é a incorporação de abordagens baseadas em jogos [11]. A utilização de jogos como recurso de aprendizagem se enquadra em uma categoria de atividades conhecidas por sua flexibilidade no fornecimento e na compreensão do conteúdo [15]. A natureza interativa e envolvente dos jogos pode aumentar significativamente a motivação dos alunos, oferecer uma experiência mais holística e facilitar o aprendizado colaborativo baseado em problemas [15]. Especificamente, os dispositivos FPGA podem ser integrados a projetos educacionais baseados em jogos, nos quais os alunos podem projetar e implementar jogos e, ao mesmo tempo, desenvolver uma compreensão dos elementos de processamento subjacentes de maneira estimulante e prática.

Para enfrentar esses desafios educacionais, este trabalho apresenta o processador CoLenda como uma ferramenta inovadora de aprendizado baseada em FPGA, desenvolvida por meio de um processo de co-design de hardware/software para o ensino de design digital e arquitetura de computadores. O CoLenda foi projetado para alunos de graduação e pós-graduação, permitindo que eles criem e explorem jogos bidimensionais e animações gráficas. Um dos principais recursos desse processador é a introdução de uma API projetada especificamente para abstrair as complexidades do hardware, reduzindo assim a curva de aprendizado. Essa API permite que o usuário se concentre nos aspectos criativos do desenvolvimento de jogos e, ao mesmo tempo, obtenha uma compreensão prática dos sistemas digitais e da arquitetura de computadores. Ao integrar a programação de alto nível com uma abordagem acessível à configuração do hardware, o processador pode ser usado para reforçar conceitos teóricos. Além disso, ele cultiva habilidades práticas em áreas como endereçamento de memória, barramentos de interface, controle e sincronização. Essa fusão de tecnologia avançada com uma interface de aprendizado intuitiva visa aprimorar a experiência educacional geral nesse campo.

O restante deste artigo é descrito a seguir. A Seção II discute soluções recentes voltadas para ferramentas e metodologias para sistemas digitais e ensino de arquitetura de computadores. A Seção III descreve a organização da arquitetura de processador proposta. A Seção IV investiga a implementação prática por meio da API CoLenda. Os resultados experimentais são apresentados na Seção V. Por fim, uma discussão sobre possíveis aplicações e perspectivas pedagógicas é apresentada na Seção VI, seguida de conclusões e referências.

II. Trabalhos Relacionados

Atualmente, universidades de todo o mundo estão buscando novas metodologias para o ensino de sistemas digitais e arquitetura de computadores com o apoio de dispositivos FPGA. No entanto, os estudantes de design digital geralmente passam mais tempo aprendendo a sintaxe e o paradigma de programação das linguagens de descrição de hardware (HDL), como Verilog e VHDL, do que aprendendo os conceitos subjacentes [5]. Além disso, a complexidade das ferramentas de automação de projeto eletrônico (EDA) atuais apresenta vários desafios para os alunos. Para enfrentar esse desafio, a literatura recente explorou abordagens inovadoras destinadas a simplificar o processo de design e, assim, reduzir a curva de aprendizado necessária, influenciando diretamente o desenvolvimento da abordagem proposta.

Para atenuar o impacto das barreiras percebidas no aprendizado do aluno, os autores em [16] propuseram uma ferramenta baseada na Web que permite que os circuitos digitais sejam projetados e simulados usando uma HDL simplificada, facilitando o aprendizado do aluno sobre o projeto de circuitos digitais de maneira mais rápida e menos densa. O projeto de processadores RISC usando a HDL Verilog por meio de um simulador de lógica digital baseado na Web é descrito em [10]. Para os experimentos, os autores propuseram uma série de exercícios de laboratório nos quais os alunos modificaram o processador incorporando novos componentes, como caminhos de dados e sinais de controle, para integrar instruções e recursos.Please check with your editor on whether to submit your manuscript by hard copy or electronically for review. If hard copy, submit photocopies such that only one column appears per page. This will give your referees plenty of room to write comments. Send the number of copies specified by your editor (typically four). If submitted electronically, find out if your editor prefers submissions on disk or as e-mail attachments.

O uso de FPGA para melhorar a experiência de ensino em design digital e arquitetura de computadores, propondo o desenvolvimento de processadores padrão, como MIPS e RISC-V, provou ser uma abordagem eficiente para abstrair as complexidades do design e concentrar-se no processo de desenvolvimento. Para isso, os autores em [17] apresentam um experimento destinado a desenvolver um processador MIPS e uma série de avaliações de programação. Ao fazer isso, os alunos adquirem o pensamento sistemático necessário para projetar um sistema de computador complexo. Da mesma forma, os autores de [18] descrevem uma experiência na qual os alunos criaram processadores RISC-V personalizados. Os experimentos relataram um aprimoramento das habilidades práticas dos alunos ao usar um núcleo de processador RISC-V e configurar o fluxo da ferramenta com componentes de hardware adicionais para implementar vários periféricos de microcontroladores.

A aprendizagem baseada em jogos é amplamente usada na educação devido à sua eficácia comprovada no aumento do envolvimento, da motivação e da retenção do aluno [15], [19]. Portanto, projetos lúdicos e interativos, como a criação de jogos em FPGA, também podem contribuir para o desenvolvimento de habilidades de projeto e teste em design digital e, ao mesmo tempo, apoiar conceitos de organização de computadores. Os autores descrevem uma estrutura de curso baseada em laboratório em [20], que tem como objetivo o desenvolvimento de jogos clássicos como Maze e Tetris. De fato, a introdução de conceitos de design digital com aplicativos de jogos foi fundamental para melhorar a percepção dos alunos sobre seu aprendizado. O trabalho de [21] relata o impacto do aprendizado do design de sistemas digitais por meio do desenvolvimento de jogos baseados em FPGA. No processo, os

alunos desenvolvem um projeto complexo durante todo o processo de projeto, incluindo codificação HDL e implementação na placa.

Independentemente dos benefícios da aprendizagem baseada em jogos, ainda há uma lacuna na implementação dessas estratégias educacionais no contexto do treinamento em design digital e organização computacional. Portanto, é fundamental introduzir estruturas de aprendizagem inovadoras que aproveitem os pontos fortes da gamificação e, ao mesmo tempo, abordem os desafios educacionais nesse domínio. Para atingir esse objetivo, as seções a seguir descrevem a abordagem proposta que integra percepções teóricas, habilidades práticas e desenvolvimento de jogos envolventes para capacitar os alunos para o cenário tecnológico atual e futuro.

III. Organização da Arquitetura do CoLenda

O processador CoLenda é fundamental para a solução proposta. A arquitetura que propomos aproveita o desenvolvimento de jogos como uma ferramenta de aprendizagem para explicar o uso de elementos digitais, como memórias, barramentos de comunicação de dados, registros e controle FIFO, de forma prática e divertida. Ele também usa conceitos avançados, como sincronização de dados e comunicação entre hardware e software, para melhorar a compreensão dessa integração em dispositivos modernos. O gerenciamento de recursos e o desenvolvimento de jogos são realizados usando uma linguagem de programação de alto nível, preservando a capacidade de usar a linguagem de montagem do processador. Ao contrário de alguns trabalhos anteriores, nosso foco não é expor os alunos à tarefa de projetar o processador, mas usar os recursos disponíveis para criar jogos e animações enquanto estudamos o que está por trás deles.

As seções a seguir descrevem a organização do sistema CoLenda, concentrando-se nos resultados pedagógicos relacionados aos cursos introdutórios e avançados.

A. Visão Geral da Arquitetura

Antes de se aprofundar nos detalhes da arquitetura do processador, é importante reconhecer a necessidade de uma seleção sistemática de componentes e estratégias de organização de computadores que correspondam aos resultados do aprendizado. Está além do escopo deste documento definir esses resultados, portanto, eles são descritos em termos gerais, deixando a cargo do leitor determinar quais objetivos de aprendizado são adequados. Por exemplo, os educadores podem se concentrar nos resultados relacionados à arquitetura do processador, como o impacto dos diferentes tipos ou recursos do processador no desempenho e na funcionalidade do sistema. Outro resultado em potencial poderia ser a exploração da integração de hardware/software.

A arquitetura geral do processador CoLenda é mostrada na Fig. 1. Em seu núcleo está um processador de uso geral (GPP) RISC, que pode ser implementado como um núcleo MIPS ou RISC-V ou como um núcleo RISC completo fornecido pelo fornecedor do FPGA. Essa especificação permite que cada jogo seja programado em C usando um fluxo de kit de ferramentas dedicado, em vez de ser necessário codificar todo o jogo usando componentes de hardware. Portanto, a abordagem proposta está centrada no desenvolvimento de uma unidade de processador gráfico (GPU) personalizada, projetada especificamente para fornecer recursos básicos de renderização e, ao mesmo tempo, servir como instrumento pedagógico principal.

A arquitetura da GPU consiste em um buffer de instruções, o núcleo da GPU e duas memórias que armazenam o conjunto de instruções que controlam os elementos do jogo (por exemplo, sprites), alteram o plano de fundo e renderizam formas poligonais, como quadrados e triângulos. O objetivo da escolha desses recursos é proporcionar um equilíbrio entre a funcionalidade e a simplicidade do design, de modo que os alunos possam ter uma compreensão completa de como os jogos são processados enquanto interagem com o fluxo da ferramenta.

O buffer de instruções é implementado usando dois módulos First-In-First-Out (FIFO) e opera como uma memória de instrução dedicada. Além disso, ele serve como meio de comunicação interna e transferência de dados entre o GPP e a GPU, pois os componentes de processamento podem ter requisitos de tempo diferentes. Cada FIFO pode armazenar palavras de 16x32 bits e inclui um circuito interno para evitar o transbordamento. Ao fornecer um ponto de entrada para o módulo da GPU, os FIFOs apresentam aos alunos os principais conceitos dos sistemas digitais, especialmente o buffer de dados e o controle de fluxo. Por exemplo, os experimentos que observam como os dados são gerenciados durante os cenários de operação podem permitir o monitoramento de como os FIFOs lidam com cargas de dados variáveis. Ao fazer isso, é possível demonstrar a importância do controle de fluxo para manter a eficiência do sistema.

Os elementos de processamento se comunicam por meio de sinais de interface de barramento padrão e são controlados usando a API CoLenda (descrita na Seção IV). Essa interface padrão permite o desenvolvimento de interfaces entre usuários e jogos, fornecendo funções para controlar os elementos do jogo e conectar dispositivos externos, como um joystick ou um teclado. Como resultado, uma ampla gama de componentes adicionais pode ser desenvolvida em um cenário de aprendizagem. Por exemplo, os alunos podem criar um controlador de áudio e incorporá-lo por meio do barramento, incluindo a interface do software na API. Portanto, esse recurso de aplicativo oferece uma maneira prática para que os alunos explorem como um barramento de dados opera nos níveis de hardware e software e, ao mesmo tempo, compreendam sua função essencial na coordenação dos dispositivos de entrada e saída.

Por fim, a saída *Video Graphic Array* (VGA) foi selecionada devido à sua ampla aceitação no meio acadêmico e ao suporte em placas de desenvolvimento FPGA. Ao adotar o padrão VGA, garantimos que o processador possa ser facilmente conectado ao equipamento de exibição existente, facilitando sua integração em várias salas de aula ou laboratórios. Além disso, para os alunos que estão iniciando sua jornada, uma interface analógica é relativamente simples e mais fácil de entender do que suas contrapartes digitais, por exemplo, DVI ou HDMI. Essa saída tangível é essencial para uma experiência de aprendizado envolvente, pois os alunos podem avaliar imediatamente os efeitos das alterações em seu código.

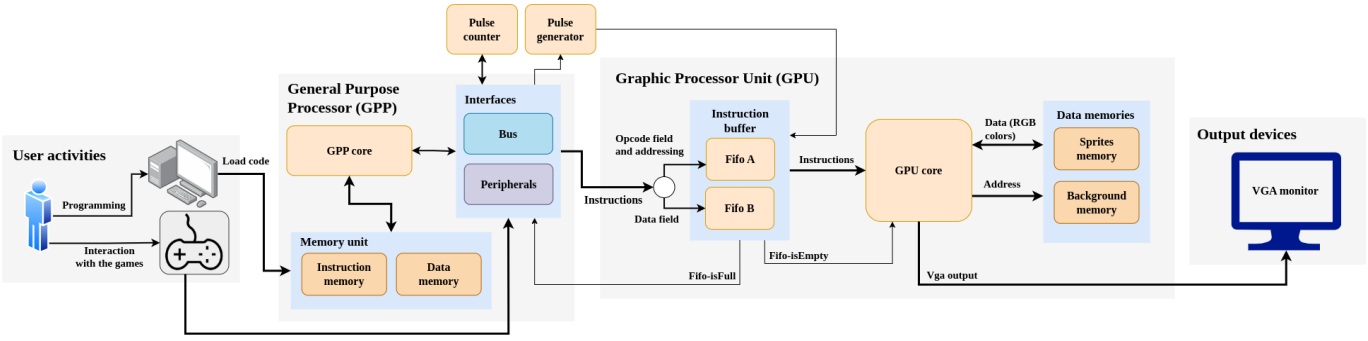


Fig. 1: CoLenda overall systematic representation.

Embora a arquitetura permita um alto grau de personalização a partir de uma perspectiva de nível avançado, o processo de renomeação pode permanecer independente da manipulação do usuário. Portanto, os esforços de programação podem se concentrar principalmente no desenvolvimento da lógica do jogo e na execução de comandos para atualizar os elementos na tela. Essa restrição não apenas simplifica o esforço de codificação, mas também se alinha com a transmissão de uma compreensão profunda do desenvolvimento de sistemas digitais integrados. Dessa forma, os alunos podem se concentrar no desenvolvimento da lógica de controle e, ao mesmo tempo, obter uma visão prática de como o software controla e interage com os componentes de hardware.

No centro dessa proposta, o núcleo da GPU foi redesenhado para oferecer aos alunos uma compreensão intuitiva, porém abrangente, de sua operação e funcionalidade. Essa abordagem promove um ambiente de aprendizado prático em que as metas de aprendizado são reforçadas por meio de aplicações práticas. À medida que os alunos se envolvem no desenvolvimento de soluções, eles adquirem uma compreensão holística do hardware e de suas implicações, aprimorando suas habilidades de design e implementação. A seção a seguir explora a arquitetura da GPU e discute como cada componente se alinha aos conceitos teóricos, demonstrando sua aplicabilidade em ambientes educacionais.

B. Arquitetura da unidade de processamento gráfico

Seguindo os currículos comuns para o ensino de design digital e arquitetura de computadores, a arquitetura de GPU proposta, detalhada na Fig. 2, é adaptada para unir conceitos teóricos a aplicações práticas. Esse alinhamento com os principais objetivos educacionais é evidente em todos os aspectos.

A GPU consiste em 32 registros dispostos em um arquivo de registro dedicado a tarefas como gerenciamento de cor de fundo e manipulação de sprite. Especificamente, a cor de fundo é controlada pela modificação do conteúdo do primeiro registro. Os registros restantes contêm informações sobre sprites. Esses elementos de armazenamento temporário são uma excelente ferramenta para os educadores demonstrarem a manipulação de dados sequenciais, o que é fundamental para o aprendizado das operações do sistema de computador. Por exemplo, a manipulação das posições dos sprites por meio desses registros pode ajudar os alunos a entender como as transferências internas de dados dão suporte ao processamento de funções internas.

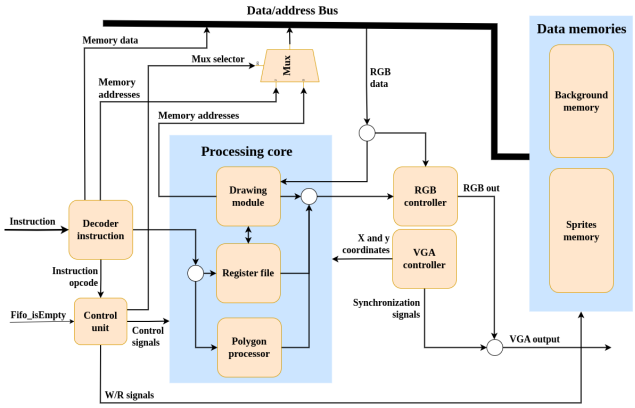


Fig. 2. Overview of the internal structure of the GPU.

A unidade de controle é fundamental para a operação da GPU, pois contém duas máquinas de estado responsáveis por inicializar os componentes internos e gerenciar a execução das instruções. Esse componente oferece uma visão prática dos ciclos de execução, desde a leitura de dados até a computação. Portanto, os educadores podem usar esse recurso para esclarecer as operações, fazendo a ponte entre os conceitos abstratos e suas manifestações tangíveis no hardware. Por exemplo, ao simular uma operação básica, como uma transferência de dados ou um cálculo aritmético, os alunos podem observar como as instruções são metodicamente lidas, decodificadas e executadas. Esse exercício prático desmistifica o conceito abstrato das operações do microprocessador, tornando-o mais concreto.

A integração do módulo de renderização e do controlador VGA fornece um exemplo de sincronização de sinal digital e operação de sinal analógico. Esse componente pode ser usado para demonstrar as complexidades das tecnologias de exibição de vídeo de uma forma acessível aos alunos. Assim, as memórias de sprite e de fundo podem ser aproveitadas para ensinar o uso eficiente da memória em sistemas de computador. Esses componentes fornecem exemplos reais de armazenamento e gerenciamento de dados, uma habilidade essencial no projeto de sistemas digitais. A memória de sprite acomoda 32 mapas de bits para elementos móveis, cada um composto de pixels de dados de 400x9 bits. Portanto, é necessário dimensioná-los de

acordo com as restrições de memória, por exemplo, 20x20 pixels. Considerando o número de elementos gráficos e seu tamanho, os alunos devem gerenciar o espaço de endereço de memória para usar a memória do sprite de forma eficiente.

Por fim, ao empregar aplicações práticas de princípios de processamento digital, o processador de polígonos é essencial para o ensino de alguns tópicos avançados, exemplificando sua capacidade de criar formas básicas, como quadrados e triângulos. Para isso, o processador realiza uma análise de colinearidade para determinar quais pixels na tela devem fazer parte de um quadrado ou triângulo. Esse processo envolve o uso de determinantes com base nas posições dos vértices do polígono e do pixel em questão. Essa operação pode ser explorada em sala de aula por meio do cálculo desses determinantes para entender como o processador decide quais pixels colorir, diminuindo o conceito abstrato de processamento geométrico.

Além disso, o processador de polígonos adota um design de pipeline para realizar a análise de colinearidade. A estrutura do pipeline divide operações complexas em estágios menores, permitindo o processamento simultâneo de várias instruções. Essa arquitetura pode ser usada em uma sala de aula para ensinar os alunos sobre a eficiência e as melhorias de rendimento oferecidas pelas arquiteturas de pipeline. Por exemplo, é possível demonstrar como o pipeline permite que várias operações sejam processadas simultaneamente. Portanto, os alunos podem aprender como o processamento paralelo acelera os cálculos complexos analisando o pipeline. Eles também podem avaliar a contribuição de cada estágio do pipeline para a tarefa geral, compreendendo assim a importância da sincronização e do tempo nos sistemas digitais.

C. Conjunto de instruções da GPU

Para que os alunos possam manipular os recursos da GPU, como elementos móveis, plano de fundo e polígonos, é necessário entender a estrutura e o funcionamento do conjunto de instruções que a GPU pode interpretar e executar. A maioria dos cursos de organização de computadores inclui um tópico específico sobre o conjunto de instruções que um processador pode executar. Ao compreender completamente os detalhes por trás do uso e da manipulação de um conjunto de instruções, os alunos podem se sentir muito mais à vontade para trabalhar com arquiteturas complexas de processadores.

Cada instrução segue um formato padrão com três campos distintos distribuídos em uma palavra de 64 bits. Em geral, um opcode identifica a operação da instrução a ser executada. A sequência de códigos de endereçamento pode selecionar um dos 32 registros do arquivo de registros ou representar um endereço de memória de 12 bits. Por fim, o campo de dados especifica as informações a serem armazenadas (por exemplo, cores ou coordenadas).

Para criar uma arquitetura de conjunto de instruções abrangente e funcional, foi projetado um conjunto de quatro tipos de instruções. Esse conjunto foi cuidadosamente elaborado para se integrar de forma coerente com outras instruções dos processadores RISC, enriquecendo a compreensão dos alunos sobre como elas são projetadas e seu impacto na arquitetura geral. Uma dessas instruções é a WTR (Writing in the Register File), que permite aos usuários configurar o conteúdo por meio de duas operações. A primeira operação modifica a cor de fundo da tela. Para isso, a instrução WRT segue o formato mostrado na Fig. 3. Em particular, o valor no campo de registro é definido como 0b0000, enquanto uma palavra de 9 bits armazena a cor RGB a ser desenhada na tela.

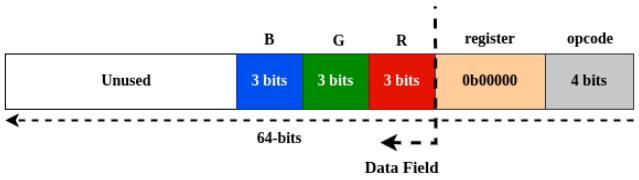


Fig. 3. WTR instruction format to control background colour.

A WTR é usada ainda para configurar um elemento móvel na tela usando o formato mostrado na Fig. 4. Como os bitmaps são armazenados na memória do sprite, um campo de deslocamento é usado para definir seu endereço de memória inicial. As coordenadas X e Y definem as coordenadas de pixel iniciais do elemento, enquanto o campo sp alterna a renderização do elemento na tela.

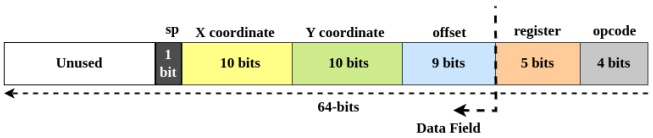


Fig. 4. Using WTR Instructions to set up a movable element.

O domínio do controle de acesso à memória é essencial nos sistemas digitais e nas arquiteturas de computadores modernos. Portanto, a instrução Write in the Sprite Memory (WSM) permite o armazenamento ou a modificação do mapa de bits do elemento em movimento. Sua representação é mostrada na Fig. 5. O opcode identifica a operação da instrução como 0b0001. O campo de endereço de memória especifica o local na memória onde o valor será atualizado. Por fim, os valores RGB especificam os componentes de cor a serem armazenados no local especificado.

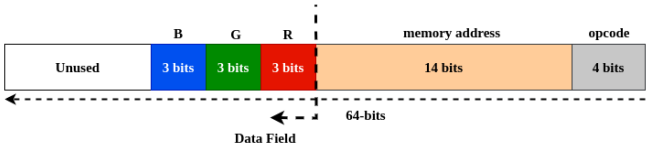


Fig. 5. Representation of the WSM instruction.

A instrução Write in the Background Memory (WBM) é responsável por armazenar ou atualizar o conteúdo na memória de fundo. Sua principal função é estabelecer valores de cor RGB para preencher áreas específicas do plano de fundo. O formato da instrução WBM é mostrado na Fig. 6, que é semelhante ao da instrução WSM, exceto pelo campo de endereço de memória de 12 bits e pelo código de operação, que é definido como 0b0010.

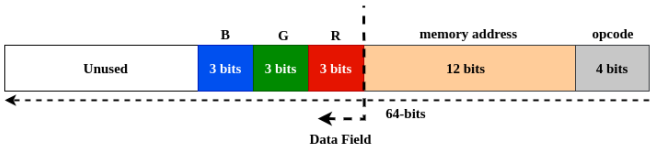


Fig. 6. Representation of the WBM instruction.

Os elementos de plano de fundo são diferentes dos sprites, pois consistem em elementos contíguos que podem abranger toda a tela. Podem ocorrer configurações incorretas se eles não forem tratados adequadamente. Por exemplo, se o tamanho da tela não for levado em conta adequadamente, o plano de fundo poderá ser exibido de forma imprecisa, resultando em uma representação incorreta e na consequente falta de consistência. Para simplificar a distribuição desses elementos, a memória de fundo é organizada em pequenos blocos de 8x8 pixels, com cada endereço de memória atribuído a apenas um desses blocos. Esse valor, escolhido como uma potência de dois, é compatível com as resoluções verticais e horizontais do padrão VGA. Por exemplo, uma resolução de 640x480 pixels resulta em uma distribuição de blocos de 80x60.

Conforme demonstrado na Fig. 7, esse método de organização do plano de fundo da tela permite configurações versáteis baseadas apenas no conteúdo da memória. Nessa configuração, cada posição no espaço de endereço do bloco armazena uma cor RGB. Entretanto, se um espaço de endereço for preenchido com o valor 0b111111110, que não representa nenhuma cor no espaço RGB, o módulo de desenho interpreta isso como uma indicação de que o bloco correspondente está inativo. Consequentemente, os pixels nessa área são preenchidos com a cor base do elemento.

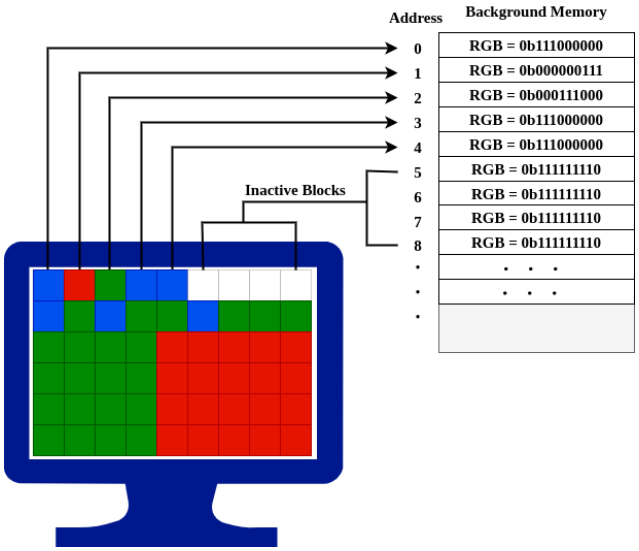


Fig. 7. Background screen organisation.

A instrução Definition of a Polygon (DP) é usada para controlar a operação do processador de polígonos e contém as informações necessárias para renderizar um polígono na tela. A Fig. 8 ilustra a instrução DP, na qual o valor do opcode é definido como 0b0011. Como o módulo tem sua própria memória para armazenar os dados transmitidos por cada comando DP, o aluno deve usar o campo de endereço para definir o local da memória do polígono onde as coordenadas cartesianas são armazenadas. Esse processo também permite o controle de sobreposição do polígono, especificando a prioridade de exibição com base na posição da memória (ou seja, os endereços mais baixos têm prioridade mais alta do que os mais altos).

O processador de polígonos proposto limita o número de formas regulares a duas (ou seja, quadrado e triângulo). Essa restrição permite que os alunos se concentrem na solução e não nas complexidades do processamento de geometria. Portanto, o campo de forma na Fig. 8 seleciona o tipo de polígono a ser exibido. Por fim, para determinar a posição de um polígono, os campos ref_point_x e ref_point_y especificam os coordenados do centro. Além disso, o campo size de 4 bits indica um valor de um conjunto de tamanhos fixos para o polígono. Essa definição permite o controle no nível do usuário e facilita o processamento. A Tabela I representa o conjunto de 15 possíveis dimensões de polígono. O código 0b0000 foi atribuído para controlar o estado de visibilidade do polígono.

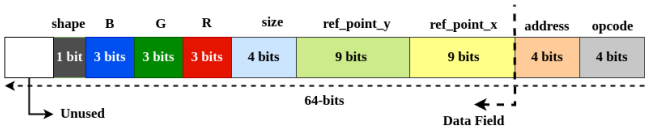


Fig. 8. Representation of DP instruction.

As atividades de laboratório que se concentram no uso dessas instruções ajudarão os alunos a entender melhor os conceitos relacionados à função dos registros, memórias e seus mecanismos de endereçamento, bem como a utilização do barramentos em uma arquitetura integrada. Por exemplo, podem ser propostas tarefas simples para que os alunos possam explorar a criação e a execução de instruções para posicionar e manipular elementos do jogo. Esses exercícios promovem a participação prática, reforçando assim os conceitos teóricos.

TABELA I
TAMANHO DOS CÓDIGOS PARA AS DIMENSÕES DOS (BASE X ALTURA)

Tamanho do Código	Dimensão (pixels)
0b0000	polígono desativado
0b0001	20x20
0b0010	30x30
0b0011	40x40
0b0100	50x50
0b0101	60x60
0b0110	70x70
0b0111	80x80
0b1000	90x90
0b1001	100x100
0b1010	110x110
0b1011	120x120
0b1100	130x130
0b1101	140x140
0b1110	150x150
0b1111	160x160

D. Barramento padrão e protocol de comunicação

Para possibilitar a comunicação entre a GPU e o GPP, foi desenvolvido um protocolo de comunicação simplificado usando FIFOs e um módulo gerador de pulsos, conforme mostrado na Fig. 1. O processador CoLenda adota a técnica de mapeamento de memória para garantir o acesso eficiente à interface de barramento e aos periféricos, proporcionando uma integração perfeita com recursos externos conectados ao processador. Com esse mecanismo, as instruções são enviadas por meio de dois canais de dados de 32 bits.

Especificamente, um canal se conecta ao FIFO~A, que serve para transmitir códigos de operação e endereçar registros e memórias. Ao mesmo tempo, o segundo canal se conecta ao

FIFO~B, que atua como o caminho para o campo de dados das instruções. É importante observar que ter dois canais diferentes de 32 bits facilita a implementação de novas instruções com uma capacidade potencial de até 64 bits.

Depois que as instruções são carregadas, o gerador de pulsos permite que os FIFOs sejam gravados por um curto período. Esse processo garante que as instruções sejam gravadas apenas uma vez, evitando a repetição e a consequente perda de consistência. Seguindo essa abordagem, os professores podem criar atividades práticas para explicar conceitos como o uso de endereços virtuais e mapeamento de memória para acesso eficiente aos recursos de hardware, bem como a integração de módulos e a interação entre hardware e software por meio de um barramento padrão.

II. API CoLenda

O projeto de uma API abrangente é fundamental para proporcionar um aprendizado ubíquo eficaz. A abstração de conceitos complexos promove maior controle sobre o processo educacional, fornecendo lições direcionadas e permitindo a integração de vários materiais de aprendizagem. Essa abstração também pode proporcionar aos alunos uma melhor compreensão do material e uma percepção mais positiva de sua experiência de aprendizagem. A API do CoLenda inclui funções, constantes e tipos compostos, ou seja, structs. Essa estrutura de software abstrai aspectos de baixo nível da arquitetura, tornando a programação de jogos na linguagem C mais simples. Essa abordagem é vantajosa quando as atividades de laboratório não exigem instruções codificadas.

Dois tipos compostos simplificam o gerenciamento de elementos gráficos como estruturas de armazenamento em nível de software. Essas estruturas configuram os sprites usando a instrução WTR, conforme definido na Lista. 1. Os atributos coord_x, coord_y, offset, data_register e active correspondem aos campos de instrução, conforme mostrado na Fig. 4. Para elementos animados, os valores step_x e step_y especificam o número de pixels que o elemento move a cada atualização de coordenadas. Por fim, o atributo collision indica se um elemento em movimento colidiu.

As funções que dão suporte direto ao desenvolvimento de jogos são definidas nas listas. 2, 3 e 4. Cada função envia instruções para a GPU com base em valores de parâmetros especificados. Além disso, uma função privada é fornecida para executar as etapas necessárias para enviar instruções à GPU. Essa função auxiliar segue o protocolo especificado na Seção III-D e transfere dados das variáveis dataA e dataB para os FIFOs A e B, respectivamente. Notavelmente, essa função é restrita à API e não pode ser acessada diretamente pelos usuários de uma perspectiva de desenvolvimento.

Em uma sala de aula, a API pode ser empregada para implementar tarefas de programação ativa envolvendo conceitos básicos, como o uso de registros para configurar dispositivos externos, trabalhar com padrões de acesso à memória e integração de hardware/software. Esses recursos podem servir de base para que os alunos se envolvam em atividades que lhes permitam implementar funções personalizadas na API. Esse processo envolve a compreensão das técnicas de mapeamento de memória para manipular dados de periféricos e barramentos de dados, permitindo que os alunos interajam com os jogos que estão desenvolvendo.

```
typedef struct{
int coord_x, coord_y;
int direction, offset, data_register;
int step_x, step_y;
int active,  collision;
} Sprite;
typedef struct{
int coord_x, coord_y, offset;
int data_register, active;
} Sprite_Fixed;
```

List. 1: Definition of constants and structures.

```
int set_background_color(int R, int G, int B){
unsigned long dataA = dataA_builder(0,0,0);
unsigned long color = B;
color = color << 3;
color = color | G;
color = color << 3;
color = color | R;
return sendInstruction(dataA, color);
}
```

List. 2: Function to set the background's base colour.

```
int set_background_block(
    int column, int line,
    int R, int G, int B)
{
  unsigned long dataA;
  unsigned long color;
  int address = (line * 80) + column;
  dataA = dataA_builder(2, 0, address);
  color = B;
   color = color << 3;
   color = color | G;
   color = color << 3;
   color = color | R;
   return sendInstruction(dataA, color);
}
```

List. 3: Function to set a specific background block.

```
int set_sprite(int registrador, int x, int y,
          int offset, int activation_bit)
{
  unsigned long dataA;
  unsigned long dataB;
  dataA = dataA_builder(0,registrador,0);
  dataB = dataB_builder(x, y, offset, activation_bit);
  return sendInstruction(dataA, dataB);
}
```

List. 4: Function to set a sprite.

Por fim, para demonstrar alguns dos recursos que podem ser explorados com o uso da API CoLenda, a função apresentada na Lista 5 fornece um exemplo prático de uso das estruturas apresentadas na Lista 1. O exemplo usa as coordenadas atuais

de cada sprite para verificar se há colisões entre elementos móveis por meio da sobreposição de retângulos. Para garantir a precisão, os alunos devem verificar se as informações no software correspondem ao que está sendo enviado e armazenado no hardware, pois atualmente é impossível ler os registros dentro do arquivo de registro da GPU.

```
1  int collision(Sprite *sp1, Sprite *sp2)
2  {
3  int h         = 15;
4  int sp1_x     = (*sp1).coord_x;
5  int sp1_y     = (*sp1).coord_y;
6  int sp2_x     = (*sp2).coord_x;
7  int sp2_y     = (*sp2).coord_y;
8  int y_face_1 = sp1_y + h;
9  int y_face_2 = sp2_y + h;
10  int x_face_1 = sp1_x + h;
11  int x_face_2 = sp2_x + h;
12  // Performs collision analysis by overlapping rectangles
13  return sp_collision(sp1_x, sp1_y, sp2_x, sp2_y,
14                 y_face_1,y_face_2,
15                 x_face_1,x_face_1);
16 }
```

List. 5: Function to check collision between elements.

A API do CoLenda inclui uma série de recursos além da funcionalidade detalhada. Em especial, ela apresenta controle de movimento de sprite, o que permite que os alunos manipulem elementos dinâmicos com mais facilidade. Além disso, a API facilita o gerenciamento eficiente de dispositivos de entrada, permitindo a integração de dispositivos de controle externos, como joysticks ou teclados. Esses recursos ampliam o escopo do design de jogos interativos, oferecendo aos alunos uma visão valiosa das complexidades da interação entre hardware e software.

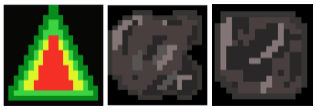
III. Avaliação dos Experimentos e Resultados

Para validar a arquitetura proposta neste documento como uma ferramenta prática de aprendizado, desenvolvemos dois jogos baseados nos clássicos Asteroids e Space Invaders. Além disso, uma animação gráfica personalizada que simula um carro de corrida dirigindo em uma pista usando o processador de polígonos completa a validação. Os aplicativos foram desenvolvidos usando a API CoLenda.

O GPP selecionado para essa avaliação experimental é o processador NIOS II, um softcore RISC de 32 bits baseado na arquitetura Harvard desenvolvida pela Intel e compatível com a maioria dos dispositivos FPGA da Altera/Intel. O processador permite a implementação de aplicativos baseados em C para programar os jogos e, ao mesmo tempo, fornece uma interface padrão entre o GPP, a GPU e os periféricos. Os experimentos foram realizados em um kit de desenvolvimento DE0-Nano usando o chip FPGA Cyclone IV EP4CE22F17C6N. O kit de ferramentas de desenvolvimento NIOS e a síntese de FPGA são baseados no software Quartus Prime Lite Edition versão 20.1. Por fim, um joystick é usado para a interação do usuário com os jogos. Com base na configuração acima, a fase de implementação e teste permite a construção do protótipo mostrado na Fig. 9. É importante observar que a placa DE0-Nano não tem um conector VGA integrado, portanto, uma interface analógica personalizada é conectada ao GPIO do dispositivo.

Fig. 9. CoLenda prototype on a DE0-Nano FPGA platform.

Um conjunto de sprites personalizados foi explicitamente projetado para reproduzir os gráficos originais do Asteroids e do Space Invaders. A Fig. 10a mostra os três tipos de sprites usados na construção do jogo Asteroids. Cada sprite é construído em uma grade de 20x20 pixels, com um código de cor de 9 bits representando cada pixel. Da mesma forma, a Fig. 10b mostra os quatro sprites usados na implementação do jogo Space Invaders. É importante observar que a cor de fundo preta é substituída pelo código RGB 0b111111110 ao gerar seus respectivos bitmaps. Essa codificação permite que o mecanismo de desenho evite renderizar qualquer pixel do sprite que corresponda a essa cor, de forma semelhante à manipulação do plano de fundo.



(a) Asteroid

(b) Space Invaders

Fig. 10: Sprites adapted from the original games.

O desenvolvimento dos jogos Asteroids e Space Invaders seguiu um fluxo específico, conforme mostrado na Fig. 11. Os estágios 1 e 2 envolvem a inicialização dos sprites e dos elementos gráficos do plano de fundo. Esse processo inicial apresenta aos alunos o uso da representação de dados structs e fornece uma demonstração prática de como as instruções são manipuladas e enviadas para a GPU. Essa manipulação inicial aborda aspectos como o uso de registros, memória (por exemplo, para preenchimento de plano de fundo) e comunicação entre o jogo que está sendo desenvolvido e o hardware.

O estágio 3 desempenha um papel fundamental no fluxograma apresentado na Fig. 11, pois lê os sinais do joystick usando a técnica de mapeamento de memória. Cada um desses

sinais é conectado ao GPP por meio do barramento de dados padrão do NIOS e é acessível por meio de um endereço de memória. Os endereços de dispositivo podem ser definidos previamente ou os alunos podem fazer experiências alterando o espaço de endereço de memória durante o processo de design de síntese de hardware. Dessa forma, esse estágio ajuda os alunos a entender como estabelecer a interface entre o GPP e seus periféricos. O mesmo processo de leitura mede o tempo de renderização do quadro durante o estágio 4 usando o módulo contador conectado ao GPP, permitindo que os alunos contem o número de quadros necessários para atualizar os elementos gráficos do jogo.

Os estágios 1 a 4 da Fig. 11 são semelhantes para a maioria dos jogos desenvolvidos usando a API CoLenda, com variações apenas na presença e na renderização de sprites e padrões de fundo. Portanto, o estágio 5 incorpora a lógica real do jogo, que é diferente para cada jogo. Neste experimento, os dois jogos têm a mesma lógica, mas com gráficos diferentes. A lógica envolve a análise de colisões de objetos na tela, a verificação das vidas restantes do jogador e a contagem do número de inimigos atingidos. Um procedimento de tempo verifica o estado dos botões e atualiza a tela de acordo. Caso contrário, ele aguarda até um horário específico para ativar o processo de atualização do sprite. Durante esse processo, os alunos podem modificar o conteúdo dos registros e das memórias e sincronizá-lo com as informações armazenadas no nível do software usando as funções da API ou criando suas próprias funções.

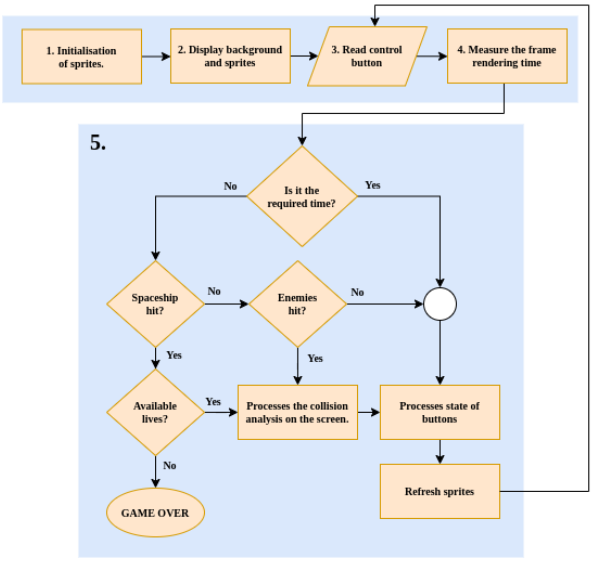
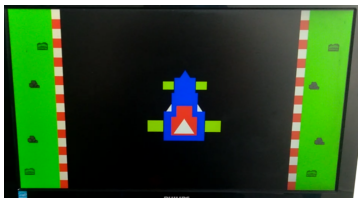


Fig. 11. Overall representation of game execution flowchart.

A implementação do jogo e da animação com base no protótipo é mostrada na Fig. 12. Tanto o Asteroids quanto o Space Invaders foram implementados para suportar 32 objetos em movimento exibidos simultaneamente na tela devido à limitação dos 32 registros na GPU. Entretanto, essa limitação não impede a criação e o uso de jogos mais complexos como forma de aprendizado. Para representar a contagem de vidas do jogador no canto inferior da tela do Space Invaders, usamos blocos de fundo devido à limitação fornecida.

Por fim, para validar a integração do processador de polígonos, a Fig. 12c apresenta o experimento no qual incluímos todos os recursos gráficos mencionados neste trabalho, ou seja, sprites, plano de fundo e polígonos. Essa animação de corrida de carros demonstra o potencial do processador CoLenda no desenvolvimento de uma representação gráfica simples que não exige seguir uma lógica de jogo específica. A animação do carro foi construída no centro da tela usando a técnica de sobreposição de polígonos. Os polígonos sobrepostos foram priorizados para renderização com base em seu espaço de endereço de memória. Os elementos restantes na tela usam os blocos de fundo e os sprites.

(a) Asteroids (b) Space Invaders



(c) Polygon processor validation

Fig. 12: Capture of the gameplay on the prototype.

A realização de experimentos no ambiente de programação CoLenda pode beneficiar significativamente os alunos em diferentes níveis. Esses experimentos podem proporcionar uma experiência de aprendizado estruturada e presencial, guiando os alunos pelos conceitos de arquitetura de computadores. Eles incentivam a criatividade e a solução de problemas dentro das limitações do hardware, ajudando os alunos a entender os princípios fundamentais dos sistemas de computador. Ao implementar atividades semelhantes em sala de aula, os alunos podem ganhar experiência prática e desenvolver uma compreensão mais profunda do assunto. Se empregados sistematicamente, experimentos como os apresentados neste documento podem aprimorar o processo de aprendizagem e preparar os alunos para aplicações no mundo real.

IV. Discussão

Para acompanhar a evolução da tecnologia e as transformações sociais, o ensino de engenharia deve se concentrar em fornecer aos alunos as habilidades e os conhecimentos certos para que tenham sucesso no mundo acelerado de hoje [22]. Isso inclui uma forte ênfase no aprendizado interdisciplinar, na experiência prática e em técnicas inovadoras de solução de problemas. Ao se adaptar a essas necessidades em constante mudança, o ensino de engenharia pode ajudar a garantir que os alunos estejam bem preparados para os desafios e as oportunidades do futuro [23]. Portanto, a incorporação da estrutura do CoLenda nesse cenário representa uma melhoria substancial

nas abordagens pedagógicas [11]. O sistema oferece uma plataforma prática para que os alunos explorem e compreendam a interação entre os componentes de hardware e a funcionalidade do sistema digital, o que é fundamental no design digital moderno por meio do aprendizado baseado em jogos. Em termos de currículo de design digital, ele serve como uma ferramenta prática para que os alunos entendam os princípios de design digital e organização de computadores. Além disso, nos cursos de arquitetura de computadores, o CoLenda facilita uma exploração mais profunda dos ciclos de processamento e das complexidades do desempenho do hardware, solidificando o entendimento dos alunos sobre conceitos teóricos complexos por meio de aplicações tangíveis.

A. Impacto geral nas práticas educacionais atuais

Em direção à instrumentação do amplo campo da engenharia de hardware, o CoLenda se torna uma ferramenta poderosa ao usar uma abordagem baseada em jogos para ajudar a entender o projeto e a integração do sistema. No cenário de aprendizado proposto, os alunos podem examinar como os componentes de hardware, como registros, memória e unidades de processamento, são orquestrados para executar as tarefas que eles projetaram. Essa percepção é fundamental para os engenheiros de hardware que precisam considerar fatores como eficiência, consumo de energia e otimização de desempenho em seus projetos [24]. Além disso, a arquitetura pode ser usada em cursos de eletrônica e sistemas incorporados, nos quais os alunos aprendem sobre integração de hardware e software e obtêm informações sobre o projeto e a implementação de aplicativos eficientes [25].

A adaptabilidade do CoLenda a diferentes níveis educacionais e disciplinas é uma confirmação de seu valor pedagógico. Por exemplo, em cursos introdutórios de programação, a arquitetura pode ser usada para desmistificar conceitos básicos de computação, fornecendo uma representação tangível de como os dados são manipulados em um sistema de computador. Em ambientes mais avançados, ela pode facilitar a análise e a pesquisa aprofundadas sobre tópicos de ponta, como técnicas avançadas de gerenciamento de dados e algoritmos eficientes de processamento de dados. Além disso, a adaptabilidade do processador o torna adequado para projetos interdisciplinares que combinam elementos de física, matemática e ciência da computação, promovendo uma experiência educacional integrada.

B. Preenchendo a lacuna entre teoria e prática

Permitir que os alunos interajam diretamente com os componentes de hardware e os manipulem proporciona uma oportunidade única de aplicar conceitos teóricos em um contexto do mundo real [22], [25]. De fato, as oportunidades de aprendizado oferecidas pelo processador demonstram a capacidade de unir o conhecimento teórico às habilidades práticas. Essa abordagem aprimora os resultados do aprendizado e prepara os alunos para os desafios que encontrarão em suas carreiras profissionais. Além disso, no cenário em evolução da educação, em que o aprendizado remoto e on-line desempenha funções cada vez mais importantes, a solução proposta oferece oportunidades para a criação de laboratórios virtuais. Esses laboratórios podem replicar ou transmitir o ambiente de bancada em tempo real, permitindo que os alunos adquiram experiência prática sem recursos físicos [3].

C. Principais dasafios

A implementação do CoLenda em ambientes educacionais práticos apresenta vários desafios que precisam ser enfrentados para maximizar sua eficácia. Um dos principais desafios é a curva de aprendizado inicial para os alunos que não têm conhecimento prévio de sistemas digitais. O CoLenda integra componentes de hardware e software, exigindo que os alunos compreendam conceitos complexos, como configuração de FPGA, projeto de lógica digital e linguagens de programação como C e Verilog. Isso pode ser muito difícil para os iniciantes e pode prejudicar sua capacidade de se envolver efetivamente com a ferramenta.

Para atenuar essa complexidade, várias estratégias podem ser empregadas. Por exemplo, a introdução gradual de conceitos básicos antes de se aprofundar nos aspectos mais complexos pode ajudar os alunos a construir uma base sólida. Isso pode ser feito começando com um projeto simples de lógica digital e avançando progressivamente para a programação de FPGA. Além disso, o fornecimento de tutoriais e materiais preparatórios que abrangem os princípios fundamentais dos sistemas digitais e da tecnologia FPGA pode preparar os alunos para o trabalho prático. Além disso, a dependência exclusiva de placas FPGA específicas pode restringir a acessibilidade em instituições com recursos limitados. Para resolver esse problema, é fundamental desenvolver alternativas ou versões compatíveis com uma gama mais ampla de dispositivos FPGA. Uma solução promissora é utilizar placas FPGA de código aberto que sejam mais econômicas.

Embora exemplos práticos e experimentos práticos demonstrem o potencial do CoLenda, reconhece-se que é necessária uma avaliação quantitativa confiável de seu impacto educacional. De fato, a realização de estudos de longo prazo para acompanhar o progresso dos alunos que usaram o CoLenda e comparar seu desempenho e resultados de carreira com aqueles que não tiveram acesso à ferramenta pode oferecer uma compreensão abrangente de seus benefícios. No entanto, a coleta e a análise do feedback dos alunos que usaram o CoLenda em seus cursos exigem a implementação de experimentos abrangentes e esforços coletivos que serão o foco de iterações futuras. Essas avaliações ajudariam a medir as melhorias no desempenho dos alunos por meio de avaliações pré e pós-curso. Métricas como notas de testes, taxas de conclusão de projetos e a capacidade de aplicar os conceitos aprendidos em cenários práticos podem, de fato, fornecer evidências do impacto da ferramenta.

D. Trabalhos futuros

Ao considerar métodos de ensino inovadores em cursos de engenharia, um dos principais desafios é a perspectiva do aluno sobre o processo de desenvolvimento de hardware em oposição ao desenvolvimento de software e vice-versa. A mudança inicial entre os dois domínios pode resultar em uma curva

de aprendizado superficial, especialmente para os alunos do primeiro ano ou para aqueles que não têm formação básica em sistemas de computador. Além disso, a dependência de hardware específico, como placas FPGA, pode limitar a acessibilidade em instituições com recursos limitados. Por fim, embora a arquitetura seja versátil, integrá-la perfeitamente a diferentes currículos pode ser um desafio. Portanto, adaptar a aplicação da arquitetura a diferentes níveis e objetivos educacionais exige esforços contínuos de desenvolvimento de currículo e treinamento de educadores.

Em resposta a esses desafios, as futuras iterações deste trabalho devem se concentrar na modularidade e na adaptabilidade. Essa abordagem permitiria atualizações fáceis e a integração de tecnologias emergentes, como IA e aprendizado de máquina. Acompanhar o ritmo dos avanços no design de hardware digital garantirá que o processador continue sendo uma ferramenta educacional de última geração. Além disso, para enfrentar os desafios de acessibilidade, futuros desenvolvimentos poderiam explorar o desenvolvimento de diferentes versões da arquitetura compatíveis com uma ampla lista de dispositivos FPGA. Por fim, para facilitar a integração do Co-Lenda em diferentes currículos, é essencial o desenvolvimento contínuo de materiais e recursos educacionais. Um suporte abrangente para educadores será fundamental para maximizar o potencial pedagógico da arquitetura, abrindo oportunidades para o desenvolvimento de projetos e colaborações interdisciplinares.

V. Conclusão

Este artigo apresentou a arquitetura do coprocessador Co-Lenda, adaptada para fins educacionais em design digital e arquitetura de computadores. O ponto central desse trabalho é a integração de conceitos teóricos com aplicações práticas, fornecendo uma ferramenta de aprendizado abrangente que preenche a lacuna entre a instrução acadêmica abstrata e a compreensão tecnológica tangível. A arquitetura foi projetada para proporcionar uma experiência prática na compreensão e manipulação de elementos de processamento e armazenamento. Para atingir esse objetivo, adotamos uma abordagem baseada na integração de um processador RISC e uma GPU em um FPGA, concentrando-nos em sua aplicação em um cenário acadêmico. Três experimentos totalmente funcionais foram desenvolvidos para validar a eficácia e a aplicabilidade em cenários de aprendizagem. Essas implementações demonstram recursos de processamento consistentes e adequação para envolver e educar os alunos em conceitos complexos de design digital e arquitetura de computadores por meio de um formato interativo e agradável.

Referências

[1] V. K. Nadimpalli, F. Hauser, D. Bittner, L. Grabinger, S. Staufer, and J. Mottok, "Systematic Literature Review for the Use of AI Based Techniques in Adaptive Learning Management Systems," in *Proceedings of the 5th European Conference on Software Engineering Education*, ECSEE '23, (New York, NY, USA), pp. 83–92, Association for Computing Machinery, 2023.

[2] P. Fuentes, C. Camarero, D. Herreros, V. Mateev, F. Vallejo, and C. Martínez, "Addressing Student Fatigue in Computer Architecture Courses," *IEEE Transactions on Learning Technologies*, vol. 15, no. 2, pp. 238–251, 2022.

[3] S. Martin, A. Fernandez-Pacheco, J. A. Ruipérez-Valiente, G. Carro, and M. Castro, "Remote Experimentation Through Arduino-Based Remote Laboratories," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 16, pp. 180–186, May 2021.

[4] N. Calazans and F. Moraes, "Integrating the teaching of computer organization and architecture with digital hardware design early in undergraduate courses," *IEEE Transactions on Education*, vol. 44, no. 2, pp. 109–119, 2001.

[5] E. D. Sozzo, D. Conficconi, A. Zeni, M. Salaris, D. Sciuto, and M. D. Santambrogio, "Pushing the Level of Abstraction of Digital System Design: A Survey on How to Program FPGAs," *ACM Computing Surveys*, vol. 55, no. 5, pp. 106:1–106:48, 2022.

[6] D. Giri, K.-L. Chiu, G. Eichler, P. Mantovani, and L. P. Carloni, "Accelerator Integration for Open-Source SoC Design," *IEEE Micro*, vol. 41, no. 4, pp. 8–14, 2021.

[7] V. Sureshkumar, R. Amin, V. R. Vijaykumar, and S. R. Sekar, "Robust secure communication protocol for smart healthcare system with fpga implementation," *Future Generation Computer Systems*, vol. 100, pp. 938–951, 2019.

[8] R. Agrawal, S. Bandara, A. Ehret, M. Isakov, M. Mark, and M. A. Kinsy, "The BRISC-V Platform: A Practical Teaching Approach for Computer Architecture," in *Proceedings of the Workshop on Computer Architecture Education*, WCAE'19, (New York, NY, USA), pp. 1–8, Association for Computing Machinery, 2019.

[9] I. Mezei, "Evolution of an educational microprocessor," *Computer Applications in Engineering Education*, vol. 28, no. 5, p. 1265 – 1277, 2020. Cited by: 1.

[10] F. Passe, M. Canesche, O. P. V. Neto, J. A. Nacif, and R. Ferreira, "Mind the gap: Bridging verilog and computer architecture," in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 2020-October, 2020. Cited by: 5.

[11] A. Rivadeneyra, F. J. Romero, M. Haider, V. D. Bhatt, J. F. Salmeron, N. Rodriguez, D. P. Morales, and M. Becherer, "Reconfigurable Electronic Platforms: A Top-Down Approach to Learn about Design and Integration of Electronic Systems," *Micromachines*, vol. 13, p. 442, Mar. 2022.

[12] L. M. Guimarães and R. d. S. Lima, "Active learning application in engineering education: Effect on student performance using repeated measures experimental design," *European Journal of Engineering Education*, vol. 46, no. 5, pp. 813–833, 2021.

[13] C. J. Jiménez-Fernández, C. B. Oliva, P. P. Fernández, F. E. Potestad-Ordóñez, and M. Valencia-Barrero, "An Academic Approach to FPGA Design Based on a Distance Meter Circuit," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 15, pp. 123–128, Aug. 2020.

[14] O. De Castro and C. Murrugarra, "GASIM: The Gate Array Graphical Simulator for FPGA Architecture Learning," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 14, pp. 95–105, Aug. 2019.

[15] Z. Yu, M. Gao, and L. Wang, "The effect of educational games on learning outcomes, student motivation, engagement and satisfaction," *Journal of Educational Computing Research*, vol. 59, no. 3, pp. 522-546, 2021.

[16] A. Trost and A. Zemva, "A web-based tool for learning digital circuit high-level modeling," *International Journal of Engineering Education*, vol. 35, no. 4, p. 1224 – 1237, 2019. Cited by: 4.

[17] X. Liu, C. De Vaulx, and J. Yuan, "Experiment design for teaching digital logic and computer organization principle," *SIGAPP Appl. Comput. Rev.*, vol. 20, p. 24–35, apr 2020.

[18] P. Jamieson, H. Le, N. Martin, T. McGrew, Y. Qian, E. Schonauer, A. Ehret, and M. A. Kinsy, "Computer engineering education experiences with risc-v architectures—from computer architecture to microcontrollers," *Journal of Low Power Electronics and Applications*, vol. 12, no. 3, 2022. Cited by: 1; All Open Access, Gold Open Access.

[19] noadmin, "Digital Game-Based Learning for Information Technology: An Exploratory Analysis," Dec. 2022.

[20] C. C. Liu, "Use of fpgas in a digital system design course with computer gaming applications," in *ASEE Annual Conference and Exposition, Conference Proceedings*, vol. 2018-June, 2018. Cited by: 0.

[21] C. J. Jiménez-Fernández, C. B. Oliva, P. P. Fernández, A. G. Soto, F. E. P. Ordóñez, and M. V. Barrero, "Learning VHDL through teamwork FPGA game design," in *2020 XIV Technologies Applied to Electronics Teaching Conference (TAEE)*, pp. 1–5, July 2020.

[22] H. Ye and C. Li, “Engineering Education Understanding Expert Decision System Research and Application,” *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–10, May 2022.
[23] M. Nino and M. A. Evans, “Fostering 21st-Century Skills in Constructivist Engineering Classrooms With Digital Game-Based Learning,” *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 10, pp. 143–149, Aug. 2015.
[24] S. Wei, L. Liu, J. Zhu, and C. Deng, “Hardware Architectures and Circuits,” in *Software Defined Chips: Volume I* (S. Wei, L. Liu, J. Zhu, and C. Deng, eds.), pp. 77–196, Singapore: Springer Nature, 2022.
[25] S. Pasricha, “Embedded Systems Education: Experiences With Application-Driven Pedagogy,” *IEEE Embedded Systems Letters*, vol. 14, pp. 167–170, Dec. 2022.