O ensino e aprendizado de Arquitetura e Organização de Computadores num currículo de Engenharia de Computação estruturado em um abordagem prática e sistêmica

Tiago de Oliveira, Luiz Eduardo Galvão Martins, Denise Stringhini, Álvaro Luiz Fazenda, Fábio Augusto Menocci Cappabianco
Instituto de Ciência e Tecnologia
Universidade Federal de São Paulo - Unifesp
São José dos Campos, Brasil
tiago.oliveira@unifesp.br, legmartins@unifesp.br, dstringhini@unifesp.br, alvaro.fazenda@unifesp.br, cappabianco@unifesp.br

***Resumo*—O ensino sobre os conceitos de arquitetura e organização de computadores é crucial para o bom andamento do aluno num currículo de Engenharia de Computação. Além disso, a integração desses conceitos com outras áreas correlacionadas da computação é de extrema importância para que o aluno não adquira uma visão fragmentada de um sistema computacional complexo. Neste artigo apresenta-se a metodologia de ensino realizada para o aprendizado de Arquitetura e Organização de Computadores no currículo do curso de Engenharia de Computação da Universidade Federal de São Paulo. Esse currículo pauta-se transversalmente por uma abordagem prática e sistêmica, proporcionando ao aluno o desenvolvimento de um sistema computacional completo, que envolve desde o desenvolvimento de uma plataforma de hardware, a definição de uma linguagem de programação, o projeto de um compilador, a especificação e desenvolvimento de um sistema operacional até o desenvolvimento de um processo de comunicação em rede entre dois ou mais sistemas computacionais. Por meio de questionários aplicados aos alunos, a metodologia apresentada para o ensino e aprendizado de Arquitetura e Organização de Computadores tem se mostrada adequada, permitindo o desenvolvimento de habilidades relacionadas ao pensamento criativo e auto-motivado, ao mesmo tempo em que capacita o aluno no desenvolvimento da plataforma de hardware necessária para o sistema computacional completo.**

Palavras-chave—Ensino de Arquitetura e Organização de Computadores; Integração entre Hardware e Software; Abordagem Prática e Sistêmica.

I. INTRODUÇÃO

Uma das principais funções de um engenheiro é o projeto, desenvolvimento e a implementação de sistemas. Em consonância com essa prerrogativa, os currículos de referência da Sociedade Brasileira de Computação (SBC) [1], de sociedades e institutos internacionais relacionados à computação (ACM/IEEE) [2] [3] e o CONFEA/CREA abordam a necessidade de um engenheiro de computação ser capaz de projetar, desenvolver e implementar sistemas computacionais completos. O aluno de engenharia de computação deve ser capaz de construir hardware, software, sistemas de comunicações e suas interações, seguindo teorias, princípios, métodos, técnicas e procedimentos da engenharia e da computação. Além disso, o aluno deve adquirir competências e habilidades que lhe permitam realizar estudos, planejar, especificar, projetar, desenvolver e implementar sistemas computacionais de propósito geral ou específico.

Dentro deste contexto, muitos currículos de Engenharia de Computação apresentam em sua matriz curricular algumas unidades curriculares (disciplinas, matérias ou cadeiras) relacionadas ao desenvolvimento de hardware e de software. Apesar de não haver um padrão para os nomes das unidades curriculares nos diversos cursos de Engenharia de Computação, os conteúdos abordados envolvem: Circuitos Digitais, Arquitetura e Organização de Computadores, Linguagens Formais e Autômatos, Compiladores, Sistemas Operacionais, Redes de Computadores, entre outros.

No entanto, apesar desses currículos abordarem o tema relacionado ao projeto e implementação de sistemas computacionais, as unidades curriculares previstas em seus respectivos projetos pedagógicos não costumam ser interligadas, e assim os alunos acabam adquirindo uma visão fragmentada de um sistema computacional realmente complexo.

Para evitar que os alunos tenham essa visão fragmentada no desenvolvimento de um sistema computacional que envolva tanto hardware quanto software, no currículo do curso de Engenharia de Computação da Universidade Federal de São Paulo (Unifesp) [4] as unidades curriculares são subdivididas em dois grandes grupos: as unidades curriculares gerais e as unidades curriculares integradas. Tanto as unidades curriculares gerais quanto as unidades curriculares integradas possuem conteúdos (ementas) que estão relacionados à formação técnica do engenheiro de computação, permitindo o

desenvolvimento de competências e habilidades definidas no perfil do aluno egresso.

No entanto, as unidades curriculares integradas possuem uma função pedagógica e didática fundamental e inovadora para a formação de um profissional diferenciado e bem qualificado. As unidades curriculares integradas são utilizadas para que o aluno possa, de fato, desenvolver um sistema computacional completo durante o seu processo de aprendizagem no decorrer do curso. A cada semestre o aluno deverá desenvolver uma parte do sistema computacional, finalizando-o após seis semestres ou três anos de curso.

Neste artigo, apresenta-se a metodologia adotada nos laboratórios de Circuitos Digitais e de Arquitetura e Organização de Computadores da Unifesp que busca permitir e possibilitar ao aluno a realização de parte desse sistema computacional, focando, basicamente, no desenvolvimento da plataforma de hardware. Nessa metodologia o conteúdo teórico é aplicado de forma prática, evolucionista e totalmente centrada no aluno com o uso de ferramentas e pedagogias de ensino ativas.

Na sequência, na seção II, são apresentados alguns trabalhos relacionados ao ensino de arquitetura e organização de computadores publicados na literatura científica nacional e internacional. Para um melhor entendimento sobre o contexto em que se inserem as unidades curriculares de laboratório de Circuitos Digitais e de Arquitetura e Organização de Computadores, na seção III, abordam-se as unidades curriculares integradas do currículo do curso de Engenharia de Computação da Unifesp e a abordagem prática e sistêmica desse currículo. Por sua vez, na seção IV, especifica-se a metodologia adotada nos laboratórios de Circuitos Digitais e de Arquitetura e Organização de Computadores que possibilita o desenvolvimento de uma plataforma de hardware necessária para a composição do sistema computacional completo. Na seção V, os resultados que vem sendo obtidos nos laboratórios de Circuitos Digitais e de Arquitetura e Organização de Computadores são apontados. Por fim, na seção VI, encontram-se as considerações finais do trabalho.

II. TRABALHOS RELACIONADOS

Têm sido utilizados kits educacionais contendo componentes eletrônicos como microcontroladores e dispositivos lógicos programáveis FPGAs para o ensino e aprendizado de Circuitos Digitais e de Arquitetura e Organização de Computadores com o intuito de permitir que os alunos realizem um salto cognitivo conectando seus conhecimentos teóricos com a experiência prática. Os autores em [5] abordam práticas de ensino para a disciplina de Circuitos Digitais por meio do uso de matrizes de contatos (*protoboards*), componentes eletrônicos e circuitos integrados. Os autores em [6] fazem uma análise do uso de microcontroladores nas universidades do sul do Brasil como ferramenta de apoio ao ensino de Arquitetura de Computadores. Os autores em [7] [8] propõem a resolução pelos alunos de problemas ou projetos envolvendo o desenvolvimento de programas ou soluções em dispositivos lógicos programáveis aplicando uma metodologia baseada em problemas. Os autores em [9] [10] propõem um ensino prático de projeto de processadores em FPGAs seguindo uma metodologia baseada em projetos.

Além disso, com este mesmo intuito de conectar os conhecimentos teóricos com a experiência prática, muitos pesquisadores estão desenvolvendo e produzindo uma variedade de ferramentas educacionais capazes de simularem arquiteturas de computadores [11].

Diversos simuladores [12] têm sido propostos para o uso em disciplinas relacionas com arquitetura e organização de computadores. Como exemplos de simuladores didáticos desenvolvidos no exterior pode-se citar: GNUSim8085 [13], MARS [14] e SIMICS [15]. O simulador GNUSim8085 baseia-se na arquitetura do microprocessador 8085 da Intel, podendo apresentar os valores dos seus registradores e conteúdo da memória durante a execução das instruções. Por sua vez, o simulador MARS foi proposto para a arquitetura MIPS, sendo um dos mais populares e bem documentados, permitindo a visualização do conteúdo de seus registradores, da memória e das instruções em cada estágio do pipeline. Por fim, SIMICS é um simulador capaz de modelar arquiteturas single-core, multi-core e many-core; simular os conjuntos de instruções Alpha, x86, x86-64, Power PC, IPF, MIPS, ARM e Ultra Sparc; e emular sistemas operacionais como Linux, Solaris e Windows.

Especificamente no Brasil, diversos simuladores vêm sendo propostos na literatura científica. Como exemplos recentes pode-se citar: SimuS [16], BIPIDE [17] e MPSoCBench [18].

O simulador SimuS apresenta um ambiente integrado onde o aluno pode editar, compilar, depurar e executar código de programas escrito em linguagem de montagem do processador hipotético Sapiens, baseado no processador Neander [19]. O BIPIDE é um ambiente de desenvolvimento baseado na arquitetura BIP (*Basic Instruction-set Processor*), tendo um editor de código e um compilador desenvolvido para reconhecer algoritmos simples na pseudo-linguagem Portugol. O ambiente pode exibir passo-a-passo uma animação interna da arquitetura em relação ao programa executado. Por sua vez, a ferramenta MPSoCBench é capaz de simular sistemas multiprocessados baseados em ARM, MIPS, PowerPC e SPARC, memórias caches e componentes de interconexão, além de fornecer estimativas de desempenho, temporização e consumo de potência.

Dentro deste contexto, no currículo de Engenharia de Computação da Unifesp busca-se uma nova abordagem para a realização da integração entre teoria e prática possibilitando ao aluno o desenvolvimento de um sistema computacional completo durante sua trajetória acadêmica. A ideia principal é a definição de uma metodologia de desenvolvimento efetivamente centrada no aluno, que o permita ir além da reprodução ou constatação de experimentos; uma nova abordagem que seja capaz de trabalhar e desenvolver habilidades relacionadas ao pensamento criativo e inovador, auto-motivado e que mantenha o aluno entusiasmado na aplicação do conhecimento que está sendo adquirido durante sua vida acadêmica.

III. AS UNIDADES CURRICULARES INTEGRADAS DO CURSO DE ENGENHARIA DE COMPUTAÇÃO DA UNIFESP

As unidades curriculares integradas do curso de Engenharia de Computação da Unifesp possuem um papel fundamental na formação acadêmica do aluno, viabilizando uma experiência única e enriquecedora no processo de desenvolvimento de projetos realmente complexos, promovendo a integração entre hardware e software.

No currículo do curso doze unidades curriculares são integradas em um único grupo, quais sejam: “Circuitos Digitais”, “Arquitetura e Organização de Computadores”, “Linguagens Formais e Autômatos”, “Compiladores”, “Sistemas Operacionais”, “Redes de Computadores” e seis laboratórios denominados “Laboratórios de Sistemas Computacionais”.

As unidades curriculares denominadas “Circuitos Digitais”, “Arquitetura e Organização de Computadores”, “Linguagens Formais e Autômatos”, “Compiladores”, “Sistemas Operacionais” e “Redes de Computadores” são utilizadas para que aluno adquira a base teórica necessária para o desenvolvimento de um sistema computacional completo. Por sua vez, nas unidades curriculares denominadas “Laboratórios de Sistemas Computacionais”, o aluno ao longo de três anos irá desenvolver um sistema computacional completamente integrado.

Além de permitir o desenvolvimento de um sistema computacional, os laboratórios de sistemas computacionais propiciam o treinamento do aluno no que se refere à apresentação oral de ideias e a redação de textos técnicos e científicos de forma clara, concisa e objetiva.

O sistema computacional que deve ser desenvolvido pelo aluno ao longo de três anos se assemelha ao esquema apresentado na Fig. 1. De acordo com este esquema, o aluno irá inicialmente desenvolver o projeto digital de um processador e dos seus sistemas de memória e de entrada/saída. Uma vez descrito esse sistema de hardware, o aluno utilizará o conjunto de instruções de baixo nível (código de máquina) desenvolvido para realizar o projeto de uma linguagem de programação que possua uma sintaxe de nível mais alto do que o código de máquina do processador.

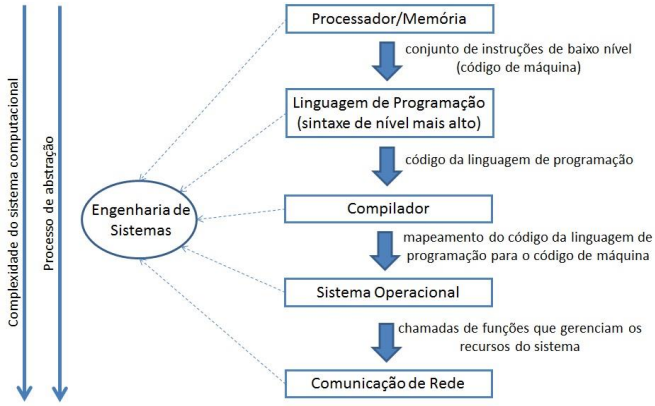

Fig. 1. Diagrama de um sistema computacional integrando o projeto tanto de hardware quanto de software

Tendo desenvolvido a linguagem de programação do sistema de hardware, o aluno deverá projetar e implementar o sistema de compilação, permitindo que a linguagem de programação possa ser traduzida para seu respectivo código de máquina.

A próxima etapa no desenvolvimento desse sistema computacional é a implementação de um sistema operacional que permita o gerenciamento dos recursos do processador e de seu sistema de memória desenvolvidos nas etapas anteriores, fornecendo, com isso, uma interface entre o sistema de hardware e o usuário.

Por fim, o aluno deverá utilizar as funções disponibilizadas pelo sistema operacional desenvolvido na etapa anterior para a realização de um projeto que envolva a comunicação em rede de dois ou mais sistemas.

Note-se que, de acordo com a Fig. 1, a complexidade e o processo de abstração do sistema computacional crescem ao longo do desenvolvimento do projeto. O aluno inicialmente trabalha no nível de portas lógicas e de circuitos digitais, depois passa a trabalhar num nível mais elevado, quando realiza a etapa de projeto da linguagem de programação e de seu correspondente processo de compilação e, na sequência, começa a trabalhar em um nível ainda mais elevado, quando realiza o projeto de um sistema operacional e de um protocolo para a comunicação em rede.

O desenvolvimento de um sistema computacional deve, sempre, ser pensado como um todo. Os problemas que o sistema deve resolver precisam ser analisados e uma solução envolvendo todos os componentes deve ser proposta. O projeto de cada componente do sistema pode ser conduzido utilizando processos específicos e a engenharia de sistemas deve permear todos esses processos, tornando-se possível a concretização de um projeto de elevada complexidade. Por isso, como mostrado na Fig. 1, a engenharia de sistemas é utilizada durante todas as etapas de projeto. O seu objetivo principal foca na definição das necessidades e funcionalidades do sistema, na realização da documentação sistemática de requisitos e na definição de todo o processo de desenvolvimento, desde a síntese até a validação do sistema, introduzindo-se, para isso, métodos e ferramentas que deverão facilitar a execução do projeto.

Diante do exposto nesta seção sobre o desenvolvimento do sistema computacional, a abordagem prática e sistêmica de ensino e aprendizado é consolidada efetivamente no currículo do curso de Engenharia de Computação da seguinte forma: no terceiro semestre do curso os alunos devem cursar a unidade curricular "Circuitos Digitais"; no quarto semestre, os alunos deverão cursar o "Laboratório de Sistemas Computacionais: Circuitos Digitais" e a unidade curricular "Arquitetura e Organização de Computadores"; no quinto semestre, os alunos deverão cursar o "Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores" e a unidade curricular "Linguagens Formais e Autômatos"; no sexto semestre, os alunos cursarão o "Laboratório de Sistemas Computacionais: Engenharia de Sistemas" e a unidade curricular "Compiladores"; no sétimo semestre, os alunos deverão cursar o "Laboratório de Sistemas Computacionais: Compiladores" e a unidade curricular "Sistemas Operacionais"; no oitavo semestre, os alunos cursarão o "Laboratório de

Sistemas Computacionais: Sistemas Operacionais" e a unidade curricular "Redes de Computadores" e; por fim, no nono semestre, os alunos realizarão o "Laboratório de Sistemas Computacionais: Redes de Computadores".

IV. Os Laboratórios de Sistemas Computacionais de Circuitos Digitais e de Arquitetura e Organização de Computadores

A unidade curricular "Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores" ocorre no quinto semestre do curso de Engenharia de Computação após a aprovação dos alunos nas unidades curriculares de "Circuitos Digitais", terceiro semestre, de "Laboratório de Sistemas Computacionais: Circuitos Digitais" e de "Arquitetura e Organização de Computadores", ambos do quarto semestre. A seguir apresenta-se uma descrição dos conteúdos teóricos e práticos abordados nessas unidades curriculares.

A. Circuitos Digitais

A unidade curricular de "Circuitos Digitais" ocorre em quatro horas semanais durante dezoito semanas totalizando uma carga horária de 72 horas e abordando conteúdos mais teóricos que envolvem diversos tópicos [20] [21], sendo eles: sistemas de numeração; funções lógicas, álgebra booleana e portas lógicas; simplificação de funções booleanas; circuitos combinacionais: conversores, decodificadores, multiplexadores, demultiplexadores e geradores de paridade; circuitos combinacionais aritméticos: somadores, subtratores, multiplicadores e comparadores de magnitude; circuitos sequenciais: latches, flip flops e registradores; máquinas de estados finitos: Moore e Mealy e projeto de circuitos combinacionais e sequenciais.

B. Laboratório de Sistemas Computacionais: Circuitos Digitais

A unidade curricular "Laboratório de Sistemas Computacionais: Circuitos Digitais" ocorre em duas horas semanais durante dezoito semanas, totalizando uma carga horária de 36 horas. Essa unidade curricular aborda conteúdos mais práticos [22] [23] tendo como principais objetivos:

- a descrição de sistemas digitais utilizando níveis de abstração diferentes (porta lógica, RTL, comportamental);
- a implementação de circuitos digitais combinacionais utilizando uma linguagem de descrição de hardware;
- a implementação de circuitos digitais sequenciais utilizando uma linguagem de descrição de hardware;
- a realização de simulações e verificação da funcionalidade dos circuitos projetados; e
- a realização de testes e comparação das funcionalidades dos circuitos implementados com os resultados obtidos na simulação.

Para a realização desses objetivos cada aluno possui em sua bancada um kit educacional DE2-115[1] [24] para a implementação em lógica programável dos circuitos digitais que são propostos durante o semestre.

Os alunos iniciam a unidade curricular com implementações em FPGAs de circuitos combinacionais e sequenciais utilizando desenhos ou esquemáticos produzidos com o software Quartus Prime. Após a familiarização dos alunos com os kits DE2-115 e com o software Quartus Prime, são introduzidos conceitos básicos sobre a linguagem de descrição de hardware Verilog e seus níveis de modelagem: modelagem no nível de portas lógicas, modelagem no nível de transferência entre registradores (RTL) e modelagem no nível comportamental. Por fim, os alunos finalizam a unidade curricular implementando circuitos combinacionais e sequenciais descritos em Verilog por meio da realização de projetos práticos envolvendo circuitos aritméticos e máquinas de estados finitos.

C. Arquitetura e Organização de Computadores

A unidade curricular de "Arquitetura e Organização de Computadores" ocorre em quatro horas semanais durante dezoito semanas totalizando uma carga horária de 72 horas e abordando conteúdos mais teóricos que envolvem diversos tópicos [25] [26], sendo eles: organização de computadores: processador, memória, entrada/saída; sistema de memória; componentes da unidade central de processamento: a unidade lógica e aritmética e a unidade de controle; conjunto de instruções; modos de endereçamento; arquitetura RISC e CISC; noções de linguagem de máquina; memória cache; pipeline; arquiteturas superescalares; sistema multiprocessado; memória virtual; e mecanismos de entrada/saída.

Esta unidade curricular aborda os conceitos teóricos necessários para o projeto e desenvolvimento de um sistema computacional composto por processadores, memórias e sistemas de entrada/saída. Com isso, após a conclusão dessa unidade curricular, os alunos podem prosseguir para a realização do "Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores".

D. Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores

Nessa unidade curricular, que ocorre em 4 horas semanais por dezoito semanas totalizando 72 horas, cada aluno deve desenvolver em Verilog sua própria plataforma de hardware, composta por processador, memória e sistema de entrada/saída utilizando o kit DE2-115 e o software Quartus Prime [27]. Sendo assim, os objetivos principais dessa unidade curricular são:

- a descrição da arquitetura de um processador utilizando uma ferramenta de descrição de hardware;
- a utilização de lógica programável para implementar um processador;
- a realização de simulações e testes para verificar a funcionalidade do sistema projetado;
- o desenvolvimento em lógica programável de um sistema de memória;

[1] O kit DE2-115 vem equipado com o FPGA Cyclone IV E, 18 chaves liga/desliga (*switches*), 27 LEDs, 8 displays de 7-segmentos, 1 display LCD 16x2 entre outros componentes. O software Quartus Prime é utilizado para a programação do kit.

- o desenvolvimento em lógica programável de um sistema de comunicação; e
- a elaboração de apresentações orais e redação de textos.

Para o cumprimento desses objetivos, a unidade curricular de "Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores" foi subdividida em quatro estágios denominados Pontos de Checagem (PCs), sendo que no final de cada PC é realizada uma avaliação do progresso do aluno no desenvolvimento de sua plataforma de hardware.

1) Ponto de Checagem 1 (PC1): Durante esse estágio, os alunos são estimulados a pesquisar o funcionamento e arquitetura interna de vários processadores da literatura, tais como o MIPS [28], o ARM [29], Pentium [30], o Neander [19] entre muitos outros encontrados em livros didáticos de arquitetura e organização de computadores e em artigos e datasheets disseminados na internet. Além disso, os alunos buscam também estudar alguns projetos realizados nos semestres anteriores por alunos que já foram aprovados nessa unidade curricular. Após a realização desse levantamento bibliográfico, os alunos começam a definir o conjunto de instruções (ISA) que será suportado pela sua plataforma de hardware e baseando-se nesse conjunto inicial de instruções eles elaboram um esboço de sua arquitetura interna.

2) Ponto de Checagem 2 (PC2): Nesse estágio, o aluno deve corrigir os apontamentos levantados no PC1, realizar um detalhamento maior do esboço da arquitetura interna e projetar em Verilog a unidade de processamento (*datapath*) da plataforma de hardware que foi proposta pelo aluno no PC1. Simulações no software Quartus Prime e testes no kit DE2-115 são realizados pelo aluno para demonstrar o correto funcionamento da unidade de processamento desenvolvida.

3) Ponto de Checagem 3 (PC3): Durante esse estágio, os alunos devem corrigir falhas encontradas ou apontamentos levantados na avaliação do PC2 e realizar a especificação em Verilog da unidade de controle da plataforma de hardware que o aluno propôs desenvolver. Simulações no software Quartus Prime e testes no kit DE2-115 são realizados pelos alunos para demonstrar o correto funcionamento da unidade de controle desenvolvida.

4) Ponto de Checagem 4 (PC4): Nesse último estágio, o aluno deve trabalhar na integração de todos os módulos produzidos, interligando a unidade de processamento e de controle da plataforma de hardware. Simulações no software Quartus Prime e testes no kit DE2-115 são realizados pelos alunos para demonstrar o correto funcionamento de todo o sistema.

A avaliação para aprovação dos alunos na unidade curricular de "Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores" leva em consideração os três eixos formativos dessa unidade curricular, quais sejam: apresentações orais, redação de relatórios técnicos e o desenvolvimento da plataforma de hardware. Em conjunto, esses três eixos visam proporcionar ao aluno competências e habilidades de um engenheiro de computação relacionadas ao projeto, desenvolvimento e a implementação de sistemas digitais, bem como o treinamento no que se refere à apresentação oral de ideias e a redação de textos técnicos e científicos de forma clara, concisa e objetiva. A avaliação desses três eixos formativos ocorre em cada um dos quatro PCs, como explicitado a seguir.

1) No PC1: São avaliados a apresentação oral do aluno e o relatório técnico entregue.

a) Na apresentação oral o aluno deve utilizar recursos de multimídia como datashow por aproximadamente 15 minutos com posterior arguição de uma banca composta pelo docente responsável pela unidade curricular e um tutor[2]. Nessa avaliação, a banca é responsável por dar um retorno ao aluno quanto ao domínio do conteúdo, objetividade e clareza, adequação da proposta, capacidade de comunicação, uso de recursos didáticos e aspectos técnicos do projeto que está sendo proposto pelo aluno. Dessa forma, durante o PC2 será possível trabalhar os apontamentos levantados para que se possa corrigi-los antes da avaliação subsequente.

b) No relatório técnico são apontados ao aluno a organização do relatório, a escrita, coerência, uso correto de citações, de referências, de tabelas e figuras, além das questões de projeto que referem-se ao conjunto de instruções definido, modos de endereçamento suportados, formato das instruções e a compatiblidade entre a arquitetura interna inicial definida e o o conjunto de instruções. O tutor da unidade curricular conversa individualmente com cada aluno sobre o relatório técnico, demonstrando-lhe a forma correta ou mais adequada para corrigir os apontamentos levantados.

2) No PC2: São avaliados a apresentação oral do aluno, o relatório técnico e o projeto da unidade de processamento desenvolvido em Verilog, com observância aos apontamentos realizados na apresentação oral e relatório técnico do PC1. No projeto da unidade de processamento avaliam-se a compatibilidade do projeto em relação ao proposto no PC1, o uso correto das estruturas da linguagem de descrição de hardware Verilog, as simulações de todos os módulos implementados e os testes realizados no kit DE2-115. Todos os apontamentos levantados são novamente repassados ao aluno, direcionando-o para a realização das correções necessárias antes da avaliação do PC3.

3) No PC3: De forma similar ao PC2, são avaliados a apresentação oral do aluno, o relatório técnico e o projeto da unidade de controle desenvolvido em Verilog, com observancia aos apontamentos realizados no PC2. Na avaliação do projeto da unidade de controle observam-se a compatibilidade dos sinais de controle em relação ao projeto realizado no PC2, o uso correto das estruturas da linguagem de descrição de hardware Verilog, as simulações de todos os módulos implementados e os testes realizados no kit DE2-115.

4) No PC4: Nesse último estágio, são avaliados a apresentação oral, o relatório técnico e o produto final (plataforma de hardware). Para que a aprovação do aluno na unidade curricular seja possível, é necesário que o produto final esteja funcionando corretamente no kit DE2-115. Para tanto, o aluno deve demonstrar tanto no relatório técnico como

[2] Aluno de pós-graduação com formação na área de engenharia de computação ou em área afim que auxilia o docente na realização da unidade curricular.

na apresentação de seu produto final que todas as funcionalidades da plataforma de hardware proposta pelo aluno podem ser executadas sem erros. Na apresentação do produto final o aluno é orientado a apresentar testes no kit DE2-115 utilizando algoritmos que já foram vistos na unidade curricular de "Lógica de Programação" e que permitam a utilização de instruções que:

- façam uso do sistema de entrada/saída projetado, mostrando resultados processados pela plataforma de hardware desenvolvida e recebendo dados externos enviados pelo usuário;
- utilizem o sistema de memória desenvolvido para receber dados a serem processados pela plataforma de hardware e armazenem os resultados obtidos; e
- realizem loops, saltos condicionais e incondicionais, acessem a unidade lógica e aritmética entre outros módulos da arquitetura projetada.

Como exemplo de produto final esperado no PC4, mostra-se na Fig. 2 um dos algoritmos executado na plataforma de hardware desenvolvida por um dos alunos da unidade curricular de "Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores".

```
quociente  <= 0;
dividendo <=  valor proveniente do usuário;
exibir valor do dividendo;
divisor <= valor proveniente do usuário;
exibir valor do divisor;
temp = dividendo - divisor;
enquanto (temp ≥ 0 )
{
        quociente = quociente + 1;
        temp = temp - divisor;
}
exibir quociente
```

Fig. 2. Algoritmo que retorna o quociente entre o dividendo e o divisor.

Neste teste realizado no kit DE2-115, a plataforma de hardware recebe, de acordo com as instruções que acessam o sistema de entrada/saída projetado pelo aluno, o valor do dividendo e do divisor provenientes das chaves liga/desliga. O resultado é calculado por meio da subtração do dividendo pelo divisor até que se chegue a um número menor ou igual a zero, indicando o final da divisão. Para cada subtração realizada um contador é incrementado, o qual ao final do processo será o quociente da divisão. Vale observar que no algoritmo esquematizado não há o tratamento de exceção no caso de uma divisão por zero.

Esse algoritmo foi traduzido para a linguagem de máquina utilizando o conjunto de instruções especificado pelo aluno. Na Fig. 3 podem-se observar os mnemônicos utilizados e os seus respectivos códigos binários que, em conjunto, representam a sequência de instruções que a plataforma de hardware do aluno deve executar para processar o algoritmo da Fig. 2. Para que isso ocorra, a codificação realizada na Fig. 3 foi mapeada para a memória de instruções da plataforma de hardware.

Na Fig. 4 pode-se observar a plataforma de hardware executando o algoritmo com a inserção de um valor pelo usuário por meio das chaves liga/desliga do kit DE2-115.

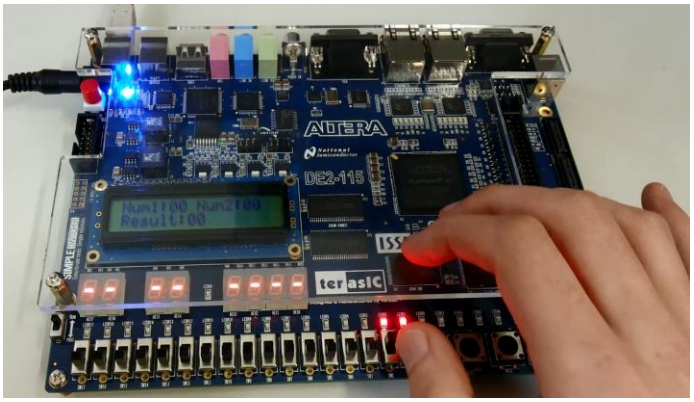

Fig. 4. Execução da plataforma de hardware desenvolvida pelo aluno com a inserção de um valor númerico no módulo de entrada/saída projetado.

Por fim, na Fig. 5, pode-se observar a plataforma de hardware mostrando ao usuário o valor 03 para o quociente, após receber os valores 06 e 02 como dividendo e divisor, respectivamente. Note-se que o resultado 03 obtido é mostrado tanto no display de 7-segmentos quanto no display LCD. No display LCD é possível observar também os valores inseridos para o dividendo (Num1: 06) e para o divisor (Num2: 02).

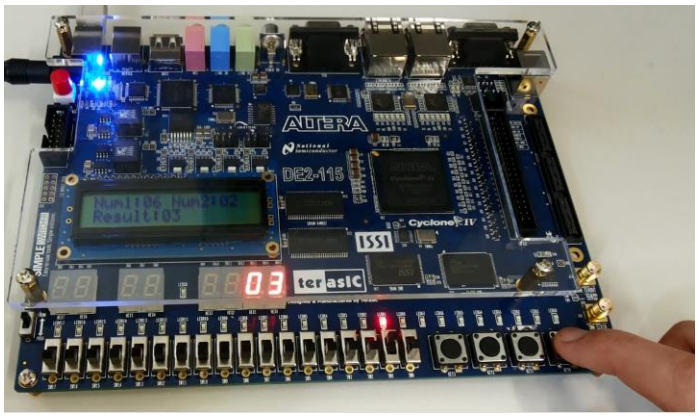

Fig. 5. Resultado obtido na plataforma de hardware desenvolvida pelo aluno após o final de processamento do algoritmo mapeado na memóra de instruções.

Vale comentar que, além do algoritmo esquematizado na Fig. 2, outros algoritmos foram testados na plataforma de hardware desenvolvida pelo aluno para garantir que todas as instruções especificadas estão funcionando corretamente. Sequencias de Fibonacci, multiplicação de números utilizando somas e subtrações, ordenação de números e verificação de primalidade são alguns exemplos de algoritmos que costumam ser utilizados pelos alunos para a realização de testes da arquitetura interna projetada por eles.

```
/*nop*/  ram_instr[0][0] = 32'b10101000000000000000000000000000;
/*loadi*/ram_instr[0][1] = 32'b01100000110000000000000000000000;//RD[00011]=0
/*out*/  ram_instr[0][2] = 32'b11000111111000000000000000000000;//RAMES[0][0]=RD[11111]=127 ->representa o traço (-)
/*out*/  ram_instr[0][3] = 32'b11000111111000000000000000000001;//RAMES[0][1]=RD[11111]=127 ->representa o traço (-)
/*out*/  ram_instr[0][4] = 32'b11000111111000000000000000000010;//RAMES[0][2]=RD[11111]=127 ->representa o traço (-)
/*out*/  ram_instr[0][5] = 32'b11000111111000000000000000000011;//RAMES[0][3]=RD[11111]=127 ->representa o traço (-)
/*nop*/  ram_instr[0][6] = 32'b10101000000000000000000000000000;

/*in*/   ram_instr[0][7] = 32'b10111000010000000000000000000000;//RD[00001] = Switches
/*out*/  ram_instr[0][8] = 32'b11000000010000000000000000000000;//RAMES[0][0]=RD[00001] //primeiro valor é exibido nos 7-segmentos
/*out*/  ram_instr[0][9] = 32'b11000000010000000000000000000100;//RAMES[0][4]=RD[00001] //primeiro valor é exibido no lcd

/*out*/  ram_instr[0][10] = 32'b11000111100000000000000000000000;//RAMES[0][0]=RD[11110]=126 ->representa o display apagado
/*out*/  ram_instr[0][11] = 32'b11000111100000000000000000000001;//RAMES[0][1]=RD[11110]=126 ->representa o display apagado
/*out*/  ram_instr[0][12] = 32'b11000111100000000000000000000010;//RAMES[0][2]=RD[11110]=126 ->representa o display apagado
/*out*/  ram_instr[0][13] = 32'b11000111100000000000000000000011;//RAMES[0][3]=RD[11110]=126 ->representa o display apagado

/*out*/  ram_instr[0][14] = 32'b11000111110000000000000000000000;//RAMES[0][0]=RD[11111]=127 ->representa o traço (-)
/*out*/  ram_instr[0][15] = 32'b11000111110000000000000000000001;//RAMES[0][1]=RD[11111]=127 ->representa o traço (-)
/*out*/  ram_instr[0][16] = 32'b11000111110000000000000000000010;//RAMES[0][2]=RD[11111]=127 ->representa o traço (-)
/*out*/  ram_instr[0][17] = 32'b11000111110000000000000000000011;//RAMES[0][3]=RD[11111]=127 ->representa o traço (-)
/*nop*/  ram_instr[0][18] = 32'b10101000000000000000000000000000;

/*in*/   ram_instr[0][19] = 32'b10111000010000000000000000000000;//RD[00010] = Switches
/*out*/  ram_instr[0][20] = 32'b11000000010000000000000000000000;//RAMES[0][0]=RD[00010] //segundo valor é exibido nos 7-segmentos
/*out*/  ram_instr[0][21] = 32'b11000000010000000000000000000101;//RAMES[0][5]=RD[00010] //segundo valor é exibido no lcd

/*out*/  ram_instr[0][22] = 32'b11000111100000000000000000000000;//RAMES[0][0]=RD[11110]=126 ->representa o display apagado
/*out*/  ram_instr[0][23] = 32'b11000111100000000000000000000001;//RAMES[0][1]=RD[11110]=126 ->representa o display apagado
/*out*/  ram_instr[0][24] = 32'b11000111100000000000000000000010;//RAMES[0][2]=RD[11110]=126 ->representa o display apagado
/*out*/  ram_instr[0][25] = 32'b11000111100000000000000000000011;//RAMES[0][3]=RD[11110]=126 ->representa o display apagado

/*sub*/  ram_instr[0][26] = 32'b00010000010000100001000000000000;//RD[00001]=RS[00001]-RT[00010]
/*jni**/ ram_instr[0][27] = 32'b10011000000000000000000000011110;//PC=ram_instr[0][30]
/*addi*/ ram_instr[0][28] = 32'b00001000110001100000000000000001;//RD[00011]=RS[00011]+1
/*jumpi*/ram_instr[0][29] = 32'b01111000000000000000000000011010;//PC=ram_instr[0][26]
/*out*/  ram_instr[0][30] = 32'b11000000110000000000000000000011;//RAMES[0][3]=RD[00011]=? -> coloca o resultado no ultimo display
/*out*/  ram_instr[0][31] = 32'b11000000110000000000000000000110;//RAMES[0][6]=RD[00011]
```

Fig. 3. Mnemônicos e seus respectivos códigos binários de acordo com as instruções projetadas na plataforma de hardware do aluno para a execução do algoritmo que retorna o quociente entre o o dividendo e o divisor.

A aprovação na unidade curricular de "Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores" está condicionada ao correto funcionamento do produto final, com a plataforma de hardware desenvolvida pelo aluno sendo capaz de executar todas as instruções definidas para o sistema computacional. Para a média final, leva-se em consideração o histórico do aluno durante todos os estágios (PC1, PC2, PC3 e PC4) e a sua evolução durante o semestre.

V. Resultados obtidos e discussão

Para avaliar o uso da abordagem aqui proposta, dois questionários foram elaborados e repassados a dois grupos de alunos de graduação do curso de Engenharia de Computação da Universidade Federal de São Paulo (Unifesp). O primeiro grupo é composto por alunos que cursaram o Laboratório de Sistemas Computacionais: Circuitos Digitais e matricularam-se no Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores. O segundo grupo refere-se a alunos que estão matriculados em Laboratórios de Sistemas Computacionais posteriores ao Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores.

Os questionários foram elaborados para permitir investigar (I) o índice de aceitação dos alunos em relação a essa nova abordagem, (II) o grau de dificuldade enfrentado pelos alunos durante o desenvolvimento de sua arquitetura, (III) o nível de aproveitamento do aprendizado proporcionado e, (IV) o interesse em continuar o desenvolvimento do sistema computacional nos próximos semestres.

O questionário para o primeiro grupo de alunos, apresentado na Tabela I, é constituído de 6 perguntas de múltipla escolha estratificado em 4 escalas, sendo a escala 0 considerada como resposta negativa, a de menor satisfação ou menor nota e a escala 3 sendo considerada como resposta positiva, a de maior satisfação ou maior nota.

TABELA I. Questionário Repassado aos Alunos do Primeiro Grupo

Pergunta 1	Você preferiria uma abordagem mais convencional, sem a integração entre as unidades curriculares?
Pergunta 2	Você conseguiu colocar em prática suas ideias iniciais de projeto da sua arquitetura?
Pergunta 3	Você se sente interessado em continuar o desenvolvimento do seu sistema computacional nos próximos laboratórios?
Pergunta 4	Você se sente motivado em estudar a teoria necessária para o desenvolvimento do seu sistema computacional nos próximos laboratórios?
Pergunta 5	Você considera que atingiu os objetivos de aprendizado que esperava obter?
Pergunta 6	Você enfrentou muita dificuldade no projeto da sua arquitetura?

O questionário para o segundo grupo de alunos, apresentado na Tabela II, é constituído de 5 perguntas de múltipla escolha estratificado em 4 escalas, sendo a escala 0 considerada como resposta negativa, a de menor satisfação ou menor nota e a escala 3 sendo considerada como resposta positiva, a de maior satisfação ou maior nota.

TABELA II. QUESTIONÁRIO REPASSADO AOS ALUNOS DO SEGUNDO GRUPO

Pergunta 1	Você considera que a abordagem adotada proporciona um aprendizado diferenciado em relação a projetos desenvolvidos separadamente em unidades curriculares isoladas?
Pergunta 2	O desenvolvimento do seu sistema computacional tem te motivado no aprendizado das unidades curriculares teóricas correlacionadas?
Pergunta 3	Você se sente interessado em continuar o desenvolvimento do seu sistema computacional nos próximos laboratórios?
Pergunta 4	Você acredita que os laboratórios de sistemas computacionais tem reduzido sua visão fragmentada sobre o projeto de um sistema computacional completo?
Pergunta 5	Você tem realizado melhorias na sua arquitetura inicial?

O resultado da avaliação realizada é sumarizado na Fig. 6, onde um total de 23 alunos do primeiro grupo participaram da pesquisa respondendo as perguntas da Tabela I. Observa-se que a abordagem adotada nos laboratórios tem tido um alto índice de aprovação pelos alunos, como pode ser observado pela baixa média das escalas obtido para a pergunta 1 (média = 0,4).

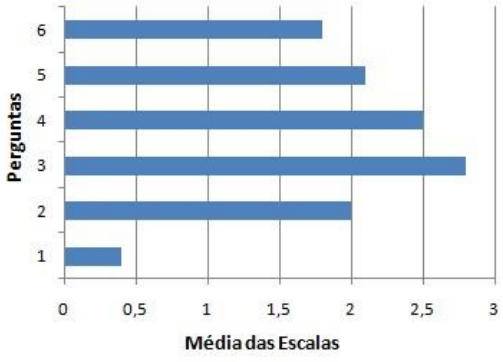

Fig. 6. Resultado da avaliação realizada pelo primeiro grupo de alunos.

A média apresentada para a pergunta 2 (média = 2) demonstra que os alunos estão aplicando suas ideias iniciais no desenvolvimento do projeto, o que é um indicativo de que a abordagem adotada tem explorado e estimulado a criatividade dos alunos, além de estar motivando-os tanto na realização de projetos práticos (média de 2,8 para a pergunta 3) quanto no estudo de assuntos teóricos relacionados (média de 2,5 para a pergunta 4).

Contudo, o alto índice obtido na pergunta 6 (média de 1,8) desperta preocupação, pois indica que uma quantidade razoável de alunos está enfrentando dificuldades na realização de seus projetos. Embora a maior parte dos alunos esteja conseguindo superar as dificuldades de projeto durante os laboratórios, em torno de 28% dos alunos acabam reprovando a unidade curricular de Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores.

Para um melhor detalhamento, diversos alunos foram convidados individualmente para conversar e expor suas opiniões sobre os motivos que os levaram a ter dificuldades ou que causaram as reprovações na unidade curricular. Basicamente, dois principais pontos chamaram a atenção:

- A maioria dos alunos que apresentaram dificuldade ou reprovaram não conseguiram fazer a associação da teoria aprendida na unidade curricular teórica de Arquitetura e Organização de Computadores com a abordagem prática do Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores. Muitos alunos reclamaram que as bibliografias disponíveis ou encontradas por eles não foram suficientes para que essa associação ou entendimento ocorresse.
- Alguns alunos reportaram problemas relacionados ao gerenciamento e a organização de arquivos eletrônicos de seus projetos gerados pelo software Quartus Prime. Uma prática que, infelizmente, não vem sendo adotada por alguns alunos é a replicação dos arquivos de projeto em mais de uma mídia de armazenamento como cópias de segurança. Furtos de notebooks, falhas técnicas em HDs e arquivos apagados erroneamente ocorreram e foram relatados, causando problemas para alguns alunos.

Diante do exposto, nos próximos oferecimentos da unidade curricular de Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores, duas medidas, descritas na sequência, serão tomadas na tentativa de eliminar ou amenizar esses problemas.

Como primeira medida, serão selecionados e disponibilizados aos alunos os melhores projetos desenvolvidos pelas turmas anteriores. Os projetos que forem bem executados e que possuírem relatórios com uma boa documentação da arquitetura desenvolvida serão disponibilizados via Moodle [31], uma plataforma de aprendizagem a distância utilizada oficialmente pela Unifesp como apoio aos cursos presenciais de graduação da universidade. Essa medida visa oferecer material de apoio mais específico e aderente com a proposta de desenvolvimento de um sistema computacional.

Como segunda medida, os alunos deverão submeter periodicamente pelo Moodle seus arquivos de projeto do software Quartus Prime, relatórios produzidos e apresentações de slides. Essa medida visa manter uma cópia institucional de segurança de cada etapa no desenvolvimento do sistema computacional do aluno.

Por fim, apresenta-se na Fig. 7 o resultado da avaliação realizada pelo segundo grupo, onde um total de 19 alunos participaram da pesquisa respondendo as perguntas da Tabela II.

As médias das escalas acima de 2,0 para as perguntas 1 e 4 indicam que os alunos consideram que a abordagem adotada possibilitou um aprendizado diferenciado, reduzindo a visão fragmentada sobre o projeto de um sistema computacional. De modo semelhante, os valores de média obtidos para as perguntas 2 e 3 demonstram que mesmo aqueles que já foram aprovados no Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores mantêm-se

continuamente motivados no desenvolvimento do sistema computacional, mantendo-se, inclusive, o interesse no estudo dos assuntos teóricos correlacionados.

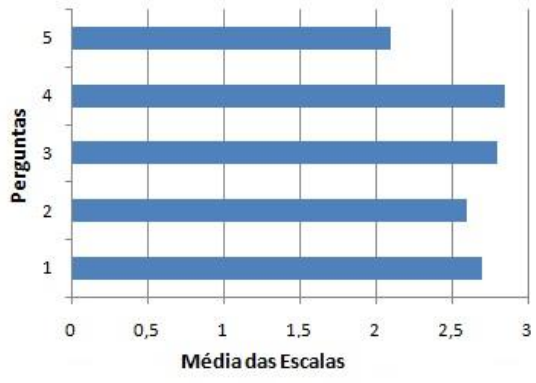

Fig. 7. Resultado da avaliação realizada pelo segundo grupo de alunos.

Vale a pena destacar que a análise dos dados obtidos para a pergunta 5 com uma média de escalas de 2,1 revela que os alunos continuaram a realizar melhorias em seu sistema computacional mesmo após terem sido aprovados no Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores. Isso é um indicativo de que a abordagem adotada está tendo êxito em manter o aluno auto-motivado e entusiasmado na aplicação do conhecimento que está sendo adquirido e aprimorado durante sua trajetória acadêmica em melhorias na arquitetura inicialmente projetada no Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores, implicando um aprendizado contínuo sobre conceitos relacionados a arquitetura e organização de computadores.

VI. Considerações finais

Este artigo apresentou a metodologia de ensino e aprendizado realizada nas unidades curriculares de laboratório de Sistemas Computacionais: Circuitos Digitais e de laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores direcionada a um currículo de Engenharia de Computação diferenciado, o qual estrutura-se em uma abordagem prática e sistêmica com o objetivo de integrar transversalmente diversos conceitos de computação de forma evolucionista. A metodologia adotada nesses dois laboratórios permite a integração de conceitos de hardware e software, proporcionando um ambiente de aprendizagem efetivamente centrado no aluno.

Sob a ótica ou o ponto de vista dos alunos que cursaram as unidades curriculares de Laboratório de Sistemas Computacionais: Circuitos Digitais e de Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores, a metodologia tem produzido resultados interessantes, buscando imprimir habilidades e atitudes que vão além da reprodução de experimentos, enfatizando-se e incentivando-se o pensamento criativo, inovador e auto-motivado na aplicação de conhecimentos teóricos em projetos práticos, como demonstrado pelos resultados obtidos nos questionários aplicados. Em particular, destaca-se ainda, que a metodologia adotada tem tido êxito em reduzir a visão fragmentada no desenvolvimento de um sistema computacional complexo.

Por fim, vale ressaltar que, após a aprovação na unidade curricular de "Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores", o aluno continuará a desenvolver o sistema computacional completo durante o seu processo de aprendizagem, devendo dar sequência aos seus estudos matriculando-se na unidade curricular de "Laboratório de Sistemas Computacionais: Engenharia de Sistemas". O sistema completo compreende o desenvolvimento da arquitetura da plataforma de hardware, a definição de uma linguagem de programação, o projeto de um compilador, a definição de um sistema operacional e um processo de comunicação em rede entre dois ou mais sistemas computacionais, o qual acontecerá no último laboratório de sistemas computacionais denominado "Laboratório de Sistemas Computacionais: Redes de Computadores", previsto para ocorrer no nono semestre da matriz curricular do curso.

Referências

[1] SBC – Sociedade Brasileira de Computação. Currículo de referência da SBC para cursos de Graduação em Computação, 2005.

[2] The Joint Task Force on Computing Curricula. IEEE Computer Society and Association for Computing Machinery. Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering, 2004.

[3] The Joint Task Force on Computing Curricula. The Association for Computing Machinery, The Association for Information Systems and The Computer Society. Computing Curricula, 2005.

[4] Unifesp. Projeto Pedagógico do Curso de Graduação do Bacharelado em Engenharia de Computação, fevereiro de 2015.

[5] Leandro S. G. de Carvalho e Fabíola G. Nakamura, “Práticas de Ensino na Disciplina de Circuitos Lógicos”, International Journal of Computer Architecture Education (IJCAE), Vol. 2, No. 1, p. 09-12, Dezembro 2013.

[6] Vinícius B. da Silva e Jean F. P. Cheiran, “Análise do uso de microcontroladores como ferramenta de apoio ao ensino-aprendizagem de Arquitetura de Computadores”, International Journal of Computer Architecture Education (IJCAE), Vol. 4, No. 1, p. 01-04, Dezembro 2015.

[7] Wagner L. A. de Oliveira, Anfranserai M. Dias, Antonio L. Apolinário Jr., Angelo A. Duarte e Tiago de Oliveira. Aplicando PBL no Ensino de Arquitetura de Computadores. In: PBL2010 International Conference, São Paulo. PBL 2010 - Congresso Internacional, 2010.

[8] Wagner L. A. de Oliveira, Anfranserai M. Dias, Antonio L. Apolinário Jr., Angelo A. Duarte e Tiago de Oliveira. Ensino de Arquitetura de Computadores: Uma Abordagem Utilizando a Metodologia de Aprendizagem Baseada em Problemas. In: Carlos Augusto Paiva da Silva Martins; Philippe Olivier Alexandre Navaux; Rodolfo Jardim de Azevedo; Sérgio Takeo Kofuji. (Org.). Arquitetura de Computadores: educação, ensino e aprendizado. 1ed.: Sociedade Brasileira de Computação - SBC, 2012, p.34-73.

[9] Ricardo O. Duarte e Pedro F. D. Garcia, Metodologia de Ensino Orientada para Projetos e Criação de Material Didático: Um relato de caso da disciplina sistemas, processadores e periféricos Laboratório, da Escola de Engenharia da UFMG. Revista Docência do Ensino Superior, v. 1, p. 01-18, 2011.

[10] Ricardo O. Duarte e Pedro F. D. Garcia, “Ensino Prático de Projeto de Processadores Segundo uma Metodologia de Ensino-Aprendizagem baseada em Projetos na Escola de Engenharia da UFMG”, International Journal of Computer Architecture Education (IJCAE), Vol. 1, No. 1, p. 11-20, Dezembro 2012.

[11] J. Djordjevic, A. Milenkovic, N. Grbanovic. An Integrated Environment for Teaching Computer Architecture. IEEE Micro, 20(3), p. 66-74, 2000.

[12] B. Nikoli, V. Milutinovic, "A survey and evaluation of simulators suitable for teaching courses in computer architecture and organization", IEEE Transactions on Education, Vol. 52, No. 4, November 2009.

[13] Z. Sridhar, "GNUSim8085, versão 1.3.7", https://launchpad.net/gnusim8085. Acesso em 26 de julho de 2017.

[14] K. Vollmar and P. Sanderson. Mars: an education-oriented MIPS assembly language simulator. In *ACM SIGCSE Bul- letin*, volume 38, pages 239–243. ACM, 2006.

[15] P. S. Magnusson et Al, "Simics: A Full System Simulation Platform" In: *Computer IEEE*, v. 35-2, pg. 50-58, fev 2002.

[16] Gabriel P. Silva e José Antônio dos S. Borges, "SimuS: Um simulador para o Ensino de Arquitetura de Computadores", International Journal of Computer Architecture Education (IJCAE), Vol. 5, No. 1, p. 07-12, Dezembro 2016.

[17] Paulo V. Vieira, André L. A. Raabe e Cesar A. Zeferino, "Projeto BIP: Impactos de 10 anos de Uso de uma Proposta Interdisciplinar de Ensino de Computação", International Journal of Computer Architecture Education (IJCAE), Vol. 5, No. 1, p. 32-37, Dezembro 2016.

[18] Liana Duenha e Rodolfo Acevedo, "Utilização dos simuladores do MPSoCBench para o ensino e aprendizagem de Arquitetura de Computadores", International Journal of Computer Architecture Education (IJCAE), Vol. 5, No. 1, p. 26-31, Dezembro 2016.

[19] Weber, R. F. Fundamentos de Arquitetura de Computadores. 2. ed. Porto Alegre. Sagra-Luzzatto, 2004.

[20] Thomas L. Floyd. Sistemas Digitais - Fundamentos e Aplicações, 9. ed. Bookman, 2007.

[21] Ronald J. Tocci, Neal S. Widmer e Gregory L. Moss. Sistemas Digitais: Princípios e Aplicações, 11 ed., Pearson 2011.

[22] César da Costa. Projetos de Circuitos Digitais com FPGA. 3.ed, Érica, 2014.

[23] M. Morris Mano and Michael D. Ciletti. Digital Design. 4rd ed. Prentice Hall, 2006.

[24] Altera. Kit Educional DE 2-115. Disponível em: <https://www.altera.com/support/training/university/boards.html#de2-115>. Acesso em 30 de julho de 2017.

[25] J. L. Hennessy and D. A. Patterson, Computer Architecture: A Quantitative Approach, 5rd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011.

[26] M. Morris Mano and Charles L. Kime. Logic and Computer Design Fundamentals, 4rd ed. Prentice-Hall, 2007.

[27] Altera. Quartus II Handbook, 2015. Disponível em: <https://www.altera.com/en_US/pdfs/literature/hb/qts/qts-qps-handbook.pdf>. Acesso em 30 de julho de 2017.

[28] José Hiroki Saito. Introdução à arquitetura e à organização de computadores : Síntese do processador MIPS. Edufscar, 2010.

[29] D. A. Patterson and J. L. Hennessy, Computer Organization and Design: The Hardware/Software Interface - ARM Edition, 1rd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2016.

[30] Barry B. Brey, Intel Microprocessors, 8rd ed. Pearson, 2008.

[31] Moodle. Modular Object-Oriented Dynamic Learning Environment (Moodle), 2017. Disponível em: <https://moodle.org>. Acesso em 30 de julho de 2017.