Uso de Realidade Aumentada para Ensino de Arquitetura de Computadores com MIPS

Geofrangite Câmara da Silva[1], Leiva Casemiro Oliveira[1], Sílvio Roberto Fernandes[1]

[1]Programa de Pós-Graduação em Ciência da Computação (PPgCC), Departamento de Computação, Universidade Federal Rural do Semi-Árido (UFERSA) – Mossoró, RN - Brasil

gcscamara@gmail.com, {leiva.casemiro, silvio}@ufersa.edu.br

Abstract. *This paper presents a simulator under development to assist the MIPS processor architecture learning process. The aim of the tool is provider it higher integration with the standard teaching material, enhancing learning activity. For this, augmented reality is used to recognize figures from the book "Computer Organization and Design, 4ed". For the development of the tool was used Unity and Vuforia. The present results are experimental, obtained in the laboratory, which allowed the recognition of 3 figures of the book. These figures present, in sequence, more and more details of the architecture and different learning purposes.*

Resumo. *Este artigo apresenta um simulador em desenvolvimento para auxiliar no aprendizado da arquitetura do processador MIPS. A ferramenta tem como objetivo permitir maior integração entre o material de estudo e as ferramentas de apoio ao aprendizado. Para isso faz uso da realidade aumentada para reconhecer as figuras do livro "Organização e Projeto de Computadores de Patterson e Hennessy, 4ed.". Para o desenvolvimento da ferramenta foi utilizado Unity e Vuforia. Os resultados atuais são experimentais, obtidos em laboratório, os quais permitiram reconhecimento de 3 figuras do livro. Essas figuras apresentam, em sequência, cada vez mais detalhes da arquitetura e propósitos de aprendizado diferentes.*

1. Introdução

Os cursos de graduação em Ciência da Computação, em geral, possuem grades curriculares compostas de disciplinas que englobam diversas subáreas da computação, desde o *hardware* até o *softwar*e. Dessas disciplinas, as que abordam o *hardware*, como as focadas no ensino de arquitetura de computadores, são tidas como mais complexas em termos de aprendizado. Devido a essa complexidade, tais disciplinas são responsáveis por grande parte da retenção e evasão dos alunos.

Nesse sentido, é comum a adoção de metodologias que recorrem a tecnologias para facilitar o aprendizado. Os simuladores, ferramentas que diminuem o nível de abstração dos conteúdos, são uma tecnologia a qual se recorre muito. Outro exemplo de tecnologia pode ser a Realidade Aumentada (RA) que, pela forma de utilização interativa e rica em elementos visuais, tem sido bastante aplicada no campo da educação.

No contexto do ensino de arquitetura de computadores, vários simuladores foram desenvolvidos. Como, em geral, os cursos baseiam-se na aprendizagem de arquiteturas padronizadas tal como o MIPS (*Microprocessor without interlocked*

pipeline stages) para facilitar o aprendizado dos relacionamentos entre os componentes de um computador, alguns dos simuladores mais populares são o MARS [Vollmar e Sanderson, 2005], WEBMIPS [Branovic, Giorgi e Matinelli, 2004], DIMIPSS [Felix, Pousa e Carvalho, 2006] e VISIMIPS [Kabir, Bari e Haque, 2011]. Embora esses simuladores proporcionem ao aluno uma noção desses relacionamentos, não há uma integração entre eles e o material de estudo (livro) de forma que o aluno não precise se "desligar" de um para utilizar o outro.

A tecnologia RA consiste na sobreposição em tempo real de elementos virtuais no mundo real através do uso de um dispositivo tecnológico que permite a manipulação e visualização desses elementos [Kirner e Siscoutto, 2007]. Uma revisão sistemática conduzida por [Akçcayir e Akçcayir, 2017] mostrou que a quantidade de trabalhos que aplicam RA na educação tem aumentado significativamente nos últimos anos e que ela é aplicável aos diversos níveis de escolaridade, desde a alfabetização até o nível superior. Nessa revisão os autores também identificaram várias vantagens da aplicação da RA na educação, dentre elas: melhora do aprendizado; facilitação do entendimento; aumento da motivação para aprender, do nível de engajamento, do interesse e da satisfação dos alunos; promoção do auto-aprendizado, do aprendizado multissensorial e do aprender fazendo; visualização de conceitos invisíveis, eventos e conceitos abstratos; redução do custo do material de laboratório. No entanto, há poucos trabalhos que aplicam RA para o ensino da computação em especial voltados para o ensino de arquitetura de computadores. Além disso, desconhecemos, até a presente data, a existência de aplicações que utilizam RA no ensino do MIPS e que sejam integradas com materiais didáticos já consolidados.

Esse trabalho apresenta um simulador para dispositivos móveis que está sendo desenvolvido para auxiliar no ensino de arquitetura de computadores. O simulador faz uso da RA para reconhecer algumas das figuras do livro "Organização e Projeto de Computadores" [Patterson e Hennessy, 2004], de Patterson e Hennessy – os ganhadores do Prêmio Turing 2017[1]. O livro é uma das referências mais utilizadas no ensino de arquitetura de computadores, apresenta de forma didática o caminho de dados do MIPS, o que explica a escolha de suas figuras como marcadores para o simulador. A medida que estuda pelo livro, o aluno pode utilizar o simulador para fazer o reconhecimento das figuras e interagir com os modelos 3D presentes na RA para facilitar o aprendizado por meio da obtenção de informações e visualização do funcionamento interno do MIPS.

O restante do trabalho está organizado como segue: a seção 2 apresenta os trabalhos relacionados; a seção 3 detalha a ferramenta proposta descrevendo seu projeto e implementação; a seção 4 mostra os resultados preliminares; e por fim, a seção 5 apresenta as conclusões e trabalhos futuros.

2. Trabalhos Relacionados

Nessa seção são apresentados alguns trabalhos relacionados que demonstram o funcionamento da linguagem *assembly* no MIPS.

[1] https://amturing.acm.org/byyear.cfm

2.1. MARS

O MARS (*Mips Assembly and Runtime Simulator*) é um simulador desenvolvido em JAVA que simula parte do conjunto de instruções do MIPS32. O MARS disponibiliza um editor de texto e um montador MIPS. Dessa forma, para verificar a execução de uma sequência de instruções, o usuário deve fornecer um código *assembly* que terá sua sintaxe verificada pelo montador e, caso esteja correto, será permitida a execução da etapa de simulação. Após a montagem, é possível modificar os valores da memória de dados e dos registradores e, em seguida, realizar a simulação onde o usuário pode escolher executar todas as instruções de uma vez ou a quantidade de instruções executas por segundo ou ainda o modo passo-a-passo, em que a transição acontece apenas ao clique do usuário. Além disso, existem as opções de pausar, retroceder e parar a execução. Uma extensão [Araújo, Pádua e Corrêa Junior, 2014] disponível para as versões mais atuais, a partir da 4.5, possibilita ainda, a visualização do caminho de dados mostrando, por meio da alteração das cores dos barramentos, a sequência em que os dados são transferidos de um componente para outro. Contudo, essa extensão não exibe os valores desses dados.

2.2. WEBMIPS

O WEBMIPS é um ambiente Web de simulação do MIPS que tem como vantagens a execução multiusuário e o fato do usuário não precisar instalar qualquer programa ou extensão para acessá-lo. Seu principal objetivo é a criação da simulação dos cinco estágios passo a passo do funcionamento do *pipeline*. Para tanto, o WEBMIPS disponibiliza um editor e um montador que suporta parte do conjunto de instruções do MIPS e permitem que o usuário insira um código e opte por executá-lo etapa por etapa ou todas as etapas de uma vez. Ao finalizar a execução do código a quantidade de ciclos de *clock* necessários é exibida. Embora as modificações de valores que ocorrem nos caminhos de dados e controle durante a execução das instruções não sejam apresentadas diretamente, é possível visualizar os valores de entrada e de saída dos componentes e os estados dos sinais, no caso da unidade de controle, clicando sobre eles.

2.3. DIMIPSS

O DIMIPSS (*Didact Interactive MIPS Simulator*) é um software multiplataforma de simulação da execução das instruções do MIPS Monociclo. Ele recebe um programa em linguagem de montagem (*assembly*) e o converte para a linguagem de máquina. Após a conversão é possível simular as instruções do código uma por uma visualizando as alterações (destacadas em cinza) feitas na memória de dados, na memória de instruções, no contador de programa e no banco de registradores. Para facilitar o entendimento, os barramentos que possuem sinais usados durante a execução da instrução também são destacados para cada instrução através das cores azul (para o caminho de dados) e vermelho (para o caminho de controle). Semelhante ao WEBMIPS, é possível visualizar os valores de entrada e de saída dos componentes e os estados dos sinais da unidade de controle bastando, para tanto, passar o mouse sobre eles.

2.4. VISIMIPS

O VISIMIPS é um simulador multiplataforma voltado para a simulação dos cinco estágios do funcionamento do pipeline do MIPS32. Ele contém um montador que traduz

as instruções do MIPS32 em código de máquina. Na simulação é possível avançar/retroceder a execução de uma instrução por vez. Também é possível visualizar os valores contidos nos barramentos dos caminhos de dados e de controle passando o *mouse* sobre eles.

2.5. DRMIPS

O DRMIPS [Nova, Araújo e Ferreira, 2013] é um simulador que permite tanto a simulação do MIPS monociclo quanto *pipeline*. Ele contém um montador que traduz as instruções do MIPS em código de máquina. Após a montagem o usuário pode modificar os valores da memória de dados e dos registradores e então, realizar a simulação executando todas as instruções de uma vez ou uma por uma. Na simulação os registradores e posições da memória ativos na execução de cada instrução são destacados de forma colorida. O simulador também mostra o caminho de dados graficamente onde é possível visualizar os valores contidos nos barramentos dos caminhos de dados e de controle. Além da versão multiplataforma para PCs ele apresenta uma versão para dispositivos móveis com Sistema Operacional (SO) Android. Contudo, os recursos de visualização e usabilidade tornam o uso desse simulador desmotivante. A falta de integração com um material de apoio e de recursos para avaliação dos alunos, deixam-no limitado como ferramenta didática.

3. Trabalho Proposto

3.1. Projeto

O simulador denominado de ARMS (Augmented Reality MIPS Simulator) foi pensado para ser usado em conjunto com o livro para auxiliar no aprendizado dos conteúdos nele abordado. Como o livro aborda um subconjunto de instruções do processador MIPS e apresenta seu caminho de dados de forma incremental, por meio de figuras, a ideia é que o aluno possa usar o simulador para visualizar a execução das instruções suportadas pelo caminho de dados representado em cada figura do livro, em cada fase da construção do caminho de dados. Assim, o ARMS está sendo desenvolvido como um aplicativo voltado para dispositivos móveis que rodam o SO Android. Tal aplicativo faz uso da câmera do dispositivo para reconhecer as figuras do livro e criar uma RA que usa modelos 3D dos caminhos de dados com os quais é possível interagir.

3.2. Ferramentas usada para o desenvolvimento

Para a implementação, estão sendo usados as ferramentas Unity 2017.2.0[2], Vuforia SDK (*software development kit*) para Unity[3], Tinkercad[4] e Android SDK[5]. Unity é um *game engine* para o desenvolvimento de jogos e aplicativos de visualização 3D que possibilita fácil exportação para PC (Windows, Mac, Linux), Android, IOS, UWP, consoles e outros. Vuforia é um SDK para desenvolvimento de aplicações RA voltadas para as

[2] https://unity3d.com/pt/

[3] https://www.vuforia.com/

[4] https://www.tinkercad.com/

[5] https://developer.android.com/index.html

plataformas Android, iOS, UWP e óculos digitais. Ela pode ser integrada ao Android Studio, Xcode, Visual Studio e Unity. Tinkercad é ambiente Web que proporciona a criação fácil e intuitiva de objetos 3D bem como sua exportação nos formatos .obj e .stl. O Android SDK é o kit de desenvolvimento de aplicações Android e é requerido pelo Unity para a criação de aplicações para Android.

3.3. Implementação

O primeiro passo da implementação foi transformar as figuras do livro em marcadores. A transformação foi feita convertendo-as em *Image Target,* um dos tipos de *Target* (alvo) que pode ser criado no Vuforia e que possibilita transformar uma imagem qualquer em um marcador. Quando uma imagem é transformada ela passa por uma análise que indica o quão reconhecível ela é. Isso é indicado através da quantidade de estrelas que ela recebe. Essa quantidade varia entre 0 e 5, sendo que quanto mais estrelas uma imagem recebe mais fácil seu reconhecimento. Após adicionar todas as figuras, foi feito o download do banco de dados e sua importação para o Unity.

O próximo passo foi criar os modelos 3D dos componentes do processador. Embora no Unity seja possível criar modelos 3D de algumas formas geométricas como cubo, esferas, cilindros e capsulas, não é possível remodelá-los. Assim, apenas os componentes que podem ser representados por estas formas foram criados no Unity. Os demais foram modelados no Tinkercad e exportados no formado .obj para o Unity.

Com os componentes modelados e as figuras transformadas em marcadores, o modelo 3D de cada figura foi gerado associando os modelos dos componentes aos marcadores. Em seguida, foram inseridos os elementos de GUI (*Graphical User Interface*) e, por fim, foram criados scripts na linguagem C# para definir o comportamento dos componentes 3D e dos elementos de GUI.

4. Resultados Preliminares

Até o momento, foram criados modelos 3D para algumas figuras do livro, na versão em português, e algumas funções foram implementadas para essas figuras. Uma dessas funções permite obter informações dos componentes do processador clicando sobre eles. A outra permite selecionar um dos tipos de instruções já implementados, opcionalmente definir suas configurações de execução (registradores envolvidos, seus valores e valores da memória de dados) e executá-lo, navegando entre suas etapas.

Quando executa o aplicativo é exibida uma tela na qual o usuário deve selecionar qual figura deseja reconhecer. Feito isso, a câmera e o *flash* (para evitar problemas de reconhecimento devido iluminação) do dispositivo são ativados e devem ser apontados para a figura do livro para que a mesma seja reconhecida. Quando a figura é reconhecida o modelo da arquitetura é gerado sobre ela juntamente com os elementos GUI que permitem configurar e controlar a execução de instruções, bem como a visualização dos valores contidos nos barramentos a cada etapa da execução. Caso a figura não seja reconhecida, devido problemas na qualidade da impressão do livro, o simulador permite o carregamento automático dos modelos 3D após um tempo (10 segundos) tentando fazer o reconhecimento.

Um exemplo do funcionamento da RA é mostrado na Figura 1, nela é mostrado o reconhecimento da figura 5.1 do livro que mostra o primeiro modelo incremental do

caminho de dados do MIPS. Quando o ARMS identifica a figura do livro, possibilita ao usuário obter informações sobre os componentes básicos (PC, memória de instruções, banco de registradores, somadores, ALU e memória de dados) do processador MIPS. Para tanto, o usuário pode clicar sobre o modelo 3D gerado de cada um deles, o qual será destacado e informações textuais serão exibidas (em um *pop-up*) sobre a função do componente clicado. Para que o *pop-up* desapareça basta clicar novamente no componente ou em algum outro componente.

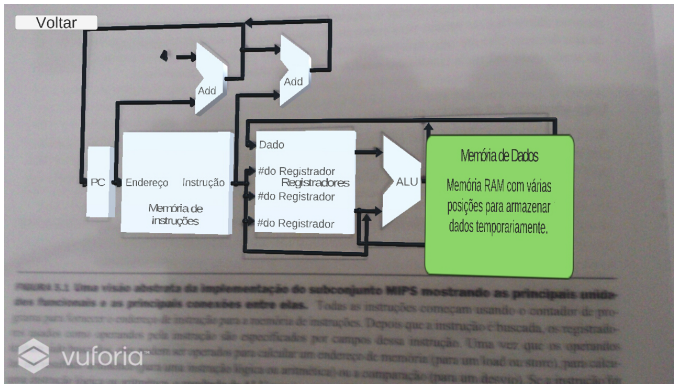

Figura 1. Reconhecimento da figura 5.1 do livro pelo simulador. Destaque para o modelo 3D gerado pela RA

Um outro marcador para a RA é a figura 5.2 do livro. Com ela reconhecida, o usuário poderá fazer a simulação da execução de instruções. Para mostrar como a simulação funciona executamos o programa contido na Figura 2, em que dois valores são lidos da memória de dados, somados e o resultado da soma é escrito na memória de dados.

```
lw t0, 4(t1)
lw t1, 8(t1)
add t2, t0, t1
sw t2, 0(t3)
```

Figura 2. Código *assembly* para testar a RA.

O primeiro passo para executar o código foi salvar os valores na memória de dados. Para isso, com o marcador reconhecido, a instrução *load* foi selecionada fazendo aparecer sobre os componentes a tela exibida na Figura 3(a). Essa tela permite selecionar os registradores envolvidos na execução da instrução, definir seus respectivos valores e os valores da memória de dados e outros valores (*offset* e *immediat*e) quando necessários (A inserção de todos esses valores é opcional, e no caso do usuário não informar os valores padrões são zero). Assim, com a posição 8 da memória de dados selecionada foi inserido o valor 14 e clicado no botão "Salvar" para que o dado fosse salvo. De forma semelhante, o valor 5 foi salvo na posição 12. Com os valores salvos, o registrador t1 foi escolhido como rs e recebeu o valor 4, o registrador t0 como rt e o *offset* recebeu 4. Para salvar os valores dos registradores selecionados e demais valores foi clicado no botão "Definir Valores" (com isso a tela para definir valores também é ocultada).

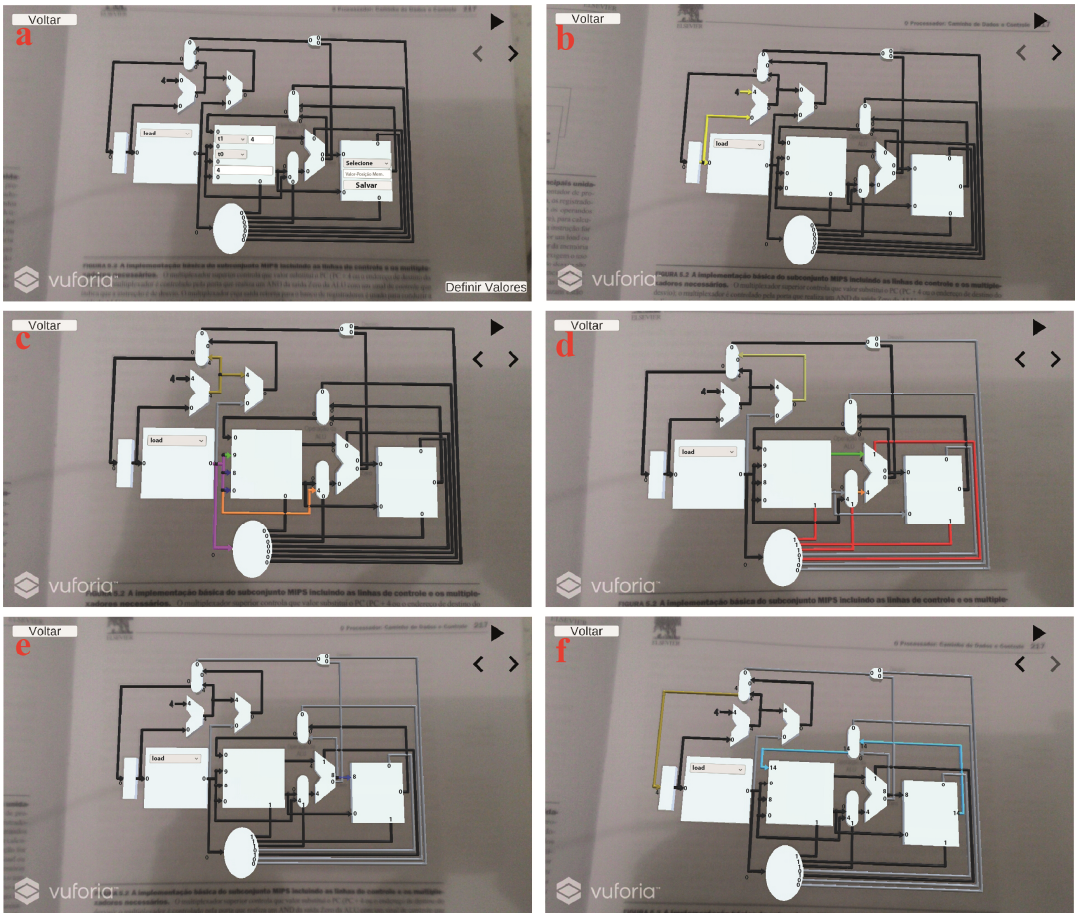

Figura 3. Sequência de etapas realizadas na execução da instrução *load*. (a) Registrador t1 escolhido como rs, registrador t0 escolhido como rt e valor 4 atribuído ao *offset*; (b) O PC (Program Counter) repassa o endereço da instrução a ser executada para a Memória de Instruções e para o Somador; (c) O Somador soma 4 ao endereço vindo do PC e a instrução é decodificada; (d) O endereço do dado a ser lido é calculado pela ALU (Arithmetic Logic Unit) somando o *offset* ao valor de t1; (e) O resultado da ALU é repassado para a Memória de Dados; (f) O dado contido no endereço repassado pela ALU é escrito em t0 e o PC é incrementado em 4.

Com os valores definidos, é possível visualizar as etapas da execução da instrução no modo automático ou no passo a passo. No modo automático, ativado pelo botão "Play", a cada 2 segundos uma etapa da execução da instrução é exibida até que a execução termine. Quando o botão "Play" é clicado também é exibido um controle que possibilita alterar a velocidade de execução, selecionando um valor de 1 (mais lento) a 5 (mais rápido). No modo passo a passo os botões "Anterior" e "Próximo" permitem navegar entre as etapas da execução. Vale salientar que os passos ou etapas da execução não representam ciclos, uma vez que essa organização é monociclo. Tais etapas representam os atrasos dos circuitos combinacionais e tem intuito didático, para facilitar o entendimento do funcionamento do caminho de dados pelos estudantes.

Em cada etapa da execução é possível visualizar a atualização dos valores que saem dos componentes ativos nessa etapa. Além disso, os barramentos ativos nos caminhos de dados e de controle são destacados de forma colorida. Os barramentos do caminho de dados que não são necessários para a instrução em execução são coloridos

com cinza. Os demais são coloridos com cores diversificadas sendo que barramentos que tem os mesmos dados são coloridos com a mesma cor e os que apresentam valores diferentes são coloridos com cores diferentes. No caminho de controle, os barramentos ativos (valor lógico 1) são coloridos em vermelho e os inativos (valor lógico 0) em cinza. Como forma de destacar os barramentos ativos em cada etapa, todos os barramentos que foram coloridos na etapa anterior, exceto os coloridos em cinza, voltam a ser preto na etapa atual.

Continuando a execução do código, a segunda instrução *load* foi executada de forma semelhante a primeira para trazer o valor 5 para o registrador t1. Com os valores nos registradores, foi executada a instrução *add* como seu resultado sendo escrito em t2, como mostra a Figura 4.

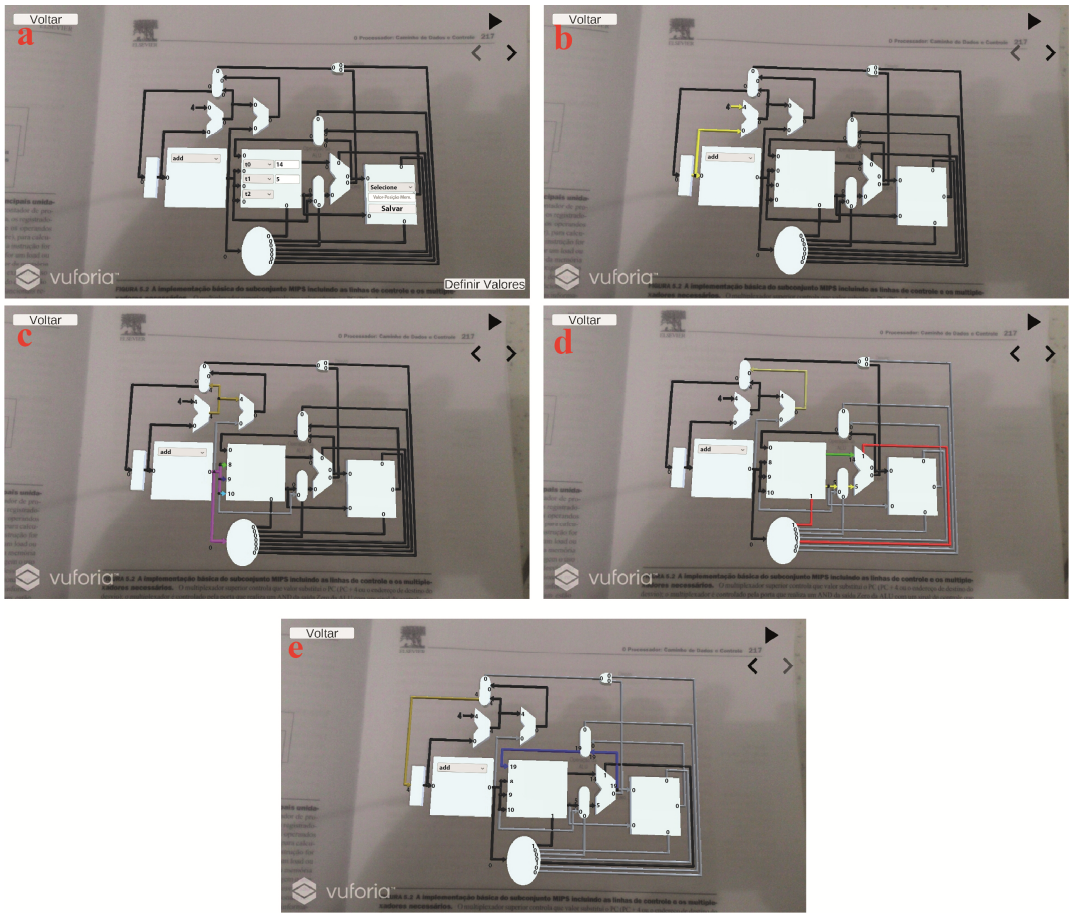

Figura 4. Sequência de etapas realizadas na execução da instrução *add*. (a) Registrador t0 escolhido como rs, registrador t1 escolhido como rt e registrador t2 escolhido como rd; (b) O PC repassa o endereço da instrução a ser executada para a Memória de Instruções e para o Somador; (c) O Somador soma 4 ao endereço vindo do PC e a instrução é decodificada; (d) Os valores contidos em t0 e t1 são somados pela ALU; (e) O resultado da ALU é escrito em t2 e o PC é incrementado em 4.

Por fim, através das etapas mostradas na Figura 5, a instrução *store* armazena o valor contido em t2 na posição 16 da memória de dados.

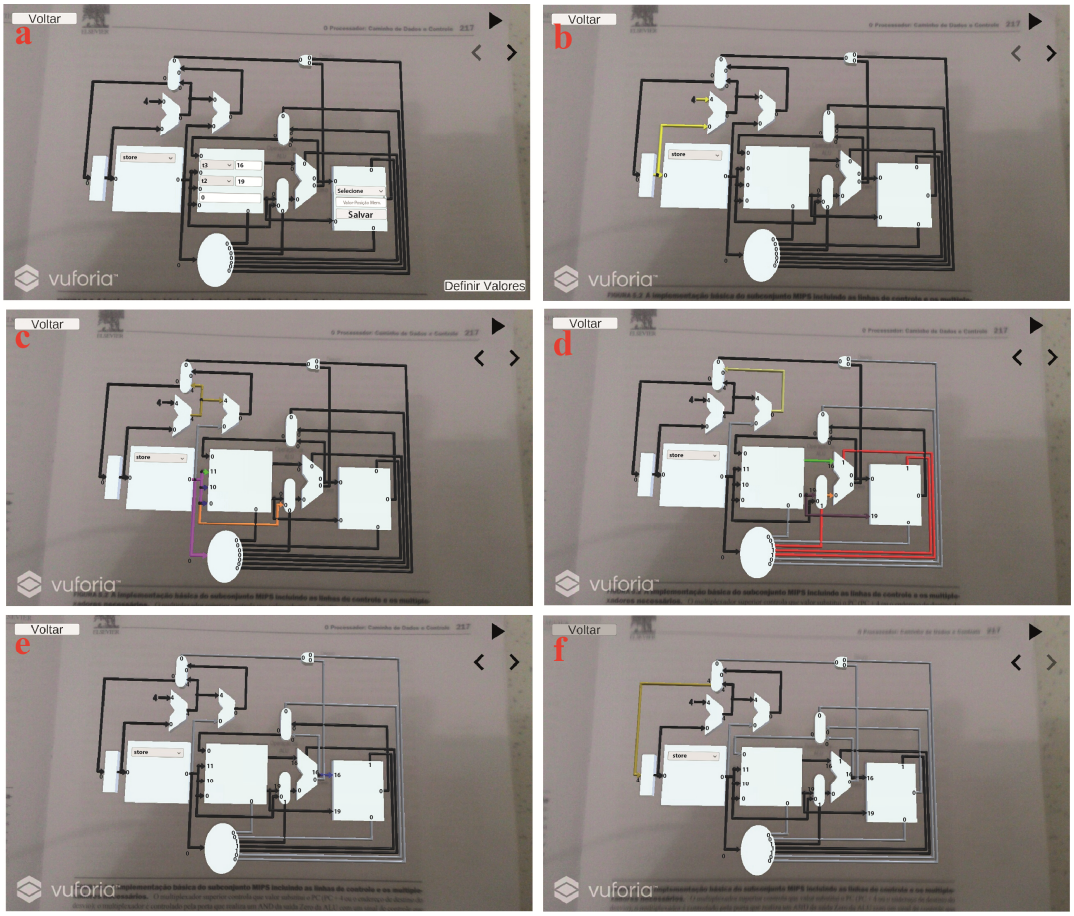

Figura 5. Sequência de etapas realizadas na execução da instrução *store*. (a) Registrador t3 escolhido como rs, registrador t2 escolhido como rt e valor 0 atribuído ao *offset*; (b) O PC repassa o endereço da instrução a ser executada para a Memória de Instruções e para o Somador; (c) O Somador soma 4 ao endereço vindo do PC e a instrução é decodificada; (d) O endereço do dado a ser escrito é calculado pela ALU somando o *offset* ao valor de t3; (e) O endereço calculado pela ALU é repassado para a Memória de Dados e o dado contido em t2 é escrito nele; (f) O PC é incrementado em 4.

Assim como o livro [Patterson e Hennessy, 2004], que apresenta as figuras da arquitetura de forma incremental, cada vez com mais detalhes e compatibilidade com mais instruções, o objetivo final do ARMS é que o reconhecimento das figuras seja feita em sequência. Isso porque as figuras têm objetivos específicos. A primeira reconhecida (Figura 1) objetiva apresentar a função de cada componente e a segunda (Figuras 3, 4 e 5) para fazer uso de registradores, memória e instruções simples. Dessa forma, a ferramenta cria um fluxo para sua utilização como recurso didático, de modo que os recursos mais avançados serão disponibilizados a medida que as etapas anteriores são cumpridas pelo usuário.

5. Conclusões e Trabalhos Futuros

Este artigo apresentou o simulador ARMS que faz reconhecimento das figuras 5.1 e 5.2 do livro "Organização e Projeto de Computadores de Patterson e Hennessy, 4ed.", provendo ao aluno uma ferramenta de simulação em realidade aumentada, desenvolvida para dispositivos com o SO Android com câmera, totalmente integrada ao material de

estudo. Tal livro é o mais utilizado no ensino de Organização e Arquitetura de Computadores, e por isso já existem diversas soluções de simulação, entretanto elas não apresentam integração com o livro didático. Com a ferramenta proposta para dispositivos móveis, ao estudar pelo livro, o aluno pode recorrer ao simulador para facilitar a assimilação dos conceitos apresentados tendo como possibilidade a execução de instruções e obtenção de informações sobre o funcionamento dos componentes do processador.

O ARMS continua em desenvolvido, assim, em breve ele deve fazer o reconhecimento de mais figuras do livro na versão em português e das mesmas equivalentes na versão em inglês e incluir ainda mais funções. Como recurso didático, o livro apresenta apenas um subconjunto das instruções do MIPS, de modo que os caminhos de dados e controles apresentados são limitados a tais instruções. Assim, pretende-se também estender o caminho de dados, adicionando componentes as figuras que são necessários para a execução de instruções não abordadas no livro, a exemplo das instruções jr (*jump register*) e lui (*load upper immediate*), tornando a ferramenta um recurso adicional ao livro. Outra funcionalidade que será adicionada são exercícios que poderão ser aplicados pelo professor para verificar o aprendizado, os quais poderão retornar um *feedback* para o professor. Além disso, pretende-se fazer a validação da ferramenta. Para isso, será feita uma comparação entre o desempenho obtido pelo grupo de teste (que usará o ARMS) e o obtido pelo grupo de controle (que será submetido ao método tradicional de ensino) em testes que serão aplicados antes e depois da utilização da ferramenta. Ao grupo de teste também será aplicado um questionário para avaliação da ferramenta.

Referências

Araújo, M. R. D., Pádua, F. L. C., Corrêa Junior, F. L. (2014). MIPS X-Ray: A MARS Simulator Plug-in for Teaching Computer Architecture. Em *Recent Contributions from Enginnering, Science & IT (iJES).*

Akçcayir, M. e Akçcayir, G (2017). Advantages and challenges associated with augmented reality for education: A systematic review of the literature. Em *Educational Research Review.*

Branovic, I, Giorgi, R e Matinelli, E. (2004). WebMIPS: A New Web-Based MIPS Simulation Environment for Computer Architecture Education. Em *31st Annual International Symposium on Computer Architecture*, Munique, Itália.

Felix, A. F., Pousa, C. V. e Carvalho, M. B. (2006). DIMIPSS: Um simulador didático e interativo do MIPS. Em *Workshop sobre Educação em Arquitetura de Computadores (WEAC)*, Ouro Preto - MG.

Kabir, M. T., Bari, M. T. e Haque, A. L. (2011). Visimips: Visual simulator of MIPS32 pipelined processor. Em *Computer Science & Education (ICCSE).*

Kirner, C. e Siscoutto, R. A (2007). Fundamentos de Realidade Virtual e Aumentada. Em *Realidade Virtual e Aumentada: Conceitos, Projeto e Aplicações.*

Nova, B., Ferreira, J. C. e Araújo, A. (2013). Tool to Support Computer Architecture Teaching and Learning. Em *International Conference of the Portuguese Society for Engineering Education (CISPEE).*

Patterson, D. A. e Hennessy, J. L. (2004) Computer organization and design: the hardware/software interface. Morgan Kaufmann. 4ª edição.

Vollmar, K. e Sanderson, P. (2005). A MIPS Assembly Language Simulator Designed For Education. Em *The Journal of Computing Sciences in Colleges*, Vol. 21, No. 1.