Revista Brasileira de Informática na Educação, Volume 18, Número 1, 2010

Bipide – Ambiente de Desenvolvimento Integrado para a Arquitetura dos Processadores BIP

Paulo Vinícius Vieira
Bacharelado em Ciência da Computação
Universidade do Vale do Itajaí
paulovinicius@univali.br

André Luis Alice Raabe
Mestrado em Computação Aplicada
Universidade do Vale do Itajaí
raabe@univali.br

Cesar Albenes Zeferino
Mestrado em Computação Aplicada
Universidade do Vale do Itajaí
zeferino@univali.br

Resumo *O ensino de conceitos introdutórios de programação costuma apresentar um nível de abstração que prejudica o aprendizado de alunos que apresentam dificuldades em lidar com o raciocínio lógico necessário ao entendimento da lógica de programação. Pensando nesta dificuldade, uma família de processadores, denominada BIP – Basic Instruction-set Processor, foi desenvolvida visando proporcionar uma redução da abstração envolvida em conceitos fundamentais da lógica de programação e também apoiar o ensino em disciplinas de fases iniciais do curso de Ciência da Computação. Neste contexto, o artigo apresenta um ambiente de desenvolvimento integrado que possibilita o desenvolvimento, execução e simulação de programas em linguagem Portugol, relacionando-os à arquitetura dos processadores da família BIP. A ferramenta foi utilizada na disciplina de Algoritmos e Programação do primeiro semestre de um curso de Ciência da Computação com uma amostra de 66 alunos (grupo experimental). Um mesmo teste foi realizado por um grupo de controle (n=49) e pelo grupo experimental. O resultado aponta para uma melhoria de desempenho do grupo experimental da ordem de 1,928 pontos na média com 99% de confiança.*

Palavras-Chave: *Aprendizagem de Programação. Simuladores de Arquitetura de Computadores. Compiladores.*

Abstract *Teaching introductory programming generally involves an abstraction level which is sometimes difficult for students with learning disabilities especially in the initial programming. Considering this, a family of processors, called BIP – Basic Instruction-set Processor, was developed in order to reduce the abstraction involved in fundamental concepts of programming and also to promote an interdisciplinary approach for teaching initial concepts on the first year of Computer Science undergraduate courses. This paper presents an integrated development environment (IDE) which enables the development, implementation and simulation of programs written in Portugol (a Portuguese-based programming language), linking them to the architecture of the BIP family. The tool was used in an introductory programming course of the first year of computer science by a sample of 66 students (experimental group). The same test was applied to a control group of students (n=49) and to the experimental group. The result point to an increase of 1.928 points in average in the experimental group with a confidence of 99%.*

Keywords: *Programming Learning. Computer Architecture Simulators. Compilers.*

1 Introdução

As disciplinas da área de Arquitetura e Organização de Computadores são fundamentais na formação de alunos dos cursos de Ciência da Computação e de Engenharia da Computação. Essas disciplinas, além de fornecer os conhecimentos básicos para a compreensão dos conceitos apresentados durante o curso, fornecem subsídios fundamentais para a aprendizagem e compreensão da lógica de programação.

Considerando esse último aspecto, é fato amplamente conhecido que alunos apresentam dificuldades na aprendizagem de conceitos de algoritmos e programação, em especial nas fases iniciais do curso [3, 4, 5, 12]. Essa dificuldade está relacionada, entre outros aspectos, à ausência de afinidade com o raciocínio lógico formal que é o fundamento para a capacidade de abstração dos alunos [22]. Nesse sentido, o estudo da arquitetura do computador cria a possibilidade de estabelecer relações dos conceitos de programação com aspectos concretos do hardware, reduzindo assim a necessidade de abstração.

Nesse contexto, identifica-se uma falta de sincronia nos currículos tradicionais em que o estudo da arquitetura e da organização do computador ocorre depois do ensino da programação. Para contornar esse problema, em muitos cursos, costuma-se apresentar algumas noções básicas de arquitetura e organização de computadores aos alunos em disciplinas que fornecem uma introdução geral à Computação, tipicamente no primeiro ano do curso.

No entanto, essa abordagem normalmente apresenta dois problemas: (*i*) a falta de uma articulação adequada entre os professores das disciplinas introdutórias a fim de estabelecer uma sincronia na apresentação dos conteúdos que possa beneficiar a aprendizagem de programação; e (*ii*) a limitação dos modelos utilizados para a apresentação dos conceitos básicos de arquitetura [8].

Com vistas a resolver essa dificuldade, uma série de processadores com um conjunto de instruções mínimo tem sido utilizada durante as aulas iniciais do curso de Ciência da Computação da Universidade do Vale do Itajaí para auxiliar o aprendizado de conceitos de arquitetura e organização de computadores de modo incremental. Esses processadores servem de referência para a apresentação dos conceitos básicos necessários ao melhor entendimento das abstrações utilizadas nas disciplinas da área de Algoritmos e Programação.

No entanto, esses processadores não possuem ferramentas de auxílio que facilitem sua aplicação em sala de aula, o que poderia melhorar a experiência e o aprendizado dos alunos com sua utilização. Neste contexto, este artigo apresenta um ambiente de desenvolvimento integrado denominado Bipide, que possibilita o desenvolvimento de programas em linguagem Portugol, a sua conversão na linguagem de montagem dos processadores BIP e a simulação desses programas, relacionando-os ao modelo de arquitetura dos processadores BIP.

O ambiente foi utilizado em atividades de sala de aula e possibilitou avaliar a aceitação de sua interface pelos aluno e também o apoio a aprendizagem de conceitos relacionados a programação introdutória como variáveis constantes, atribuição e outros.

O restante deste artigo é organizado em cinco seções, as quais descrevem a arquitetura e a organização do BIP, o desenvolvimento do Bipide, uma comparação com ambientes similares, os resultados de testes e experimentos realizados em sala de aula e as conclusões do trabalho.

2 BIP-Basic Instruction-set Processor

A escolha de modelos de processadores para o ensino de conceitos de arquitetura e organização de computadores é alvo de estudos freqüentes pelos educadores da área, que discutem aspectos que devem ser considerados na escolha de modelos de processadores a serem aplicados no ensino. Enquanto alguns autores e professores optam por utilizar modelos hipotéticos de processadores, outros adotam processadores reais e comerciais como referência para estudos de caso.

Nas fases iniciais de um curso de graduação, os processadores utilizados para o ensino concorrente da lógica de programação e de conceitos de arquitetura de computadores deve facilitar o estabelecimento de relações entre as abstrações lógicas necessárias à programação e sua representação em hardware. Porém, os modelos de processadores tipicamente utilizados são muito abstratos e não permitem estabelecer essas relações. Uma alternativa seria a utilização de processadores mais detalhados, porém esses processadores são demasiadamente complexos para serem aplicados em disciplinas do primeiro ano, e poucos são os livros texto da área que os descrevem propiciando uma integração entre a arquitetura do processador e a programação em alto nível.

Nesse contexto, deve-se buscar uma arquitetura simplificada que possibilite estabelecer uma relação entre os conceitos iniciais de programação apresentados no primeiro ano do curso e as representações em hardware correspondentes. Por exemplo, podem ser citadas algumas relações importantes entre a programação de alto nível e a sua implementação no hardware, sob a forma de conceitos de arquitetura e organização de computadores. Entre essas relações, destacam-se:

- Declaração de variável e alocação de memória;
- Constantes e operandos imediatos;

- Atribuição de variáveis e sua correspondência com as operações de acesso à memória; e
- Operações aritméticas e sua execução no hardware.

Logo, a escolha do processador para ser utilizado nas fases iniciais deve priorizar aspectos didáticos que favoreçam a compreensão das relações entre software e hardware numa abordagem multidisciplinar.

Considerando os aspectos citados anteriormente, foi desenvolvida por pesquisadores do Laboratório de Sistemas Embarcados e Distribuídos (LSED) da Universidade do Vale do Itajaí (UNIVALI) uma especificação arquitetural de uma família de processadores simplificados, denominada BIP (Basic Instruction-set Processor), que auxiliasse no aprendizado de Arquitetura e Organização de Computadores e permitisse relacionar os conceitos de programação em alto nível com sua representação em hardware [11].

Foram inicialmente desenvolvidos três modelos de processador, cada um com suporte incremental ao entendimento de conceitos de Algoritmos e Programação e oferecendo recursos adicionais para seu uso em outras disciplinas, em uma abordagem interdisciplinar [9]:

- BIP I: inclui instruções aritméticas e de acesso à memória de dados, tendo como foco o suporte ao entendimento de conceitos como níveis de linguagem, constantes, variáveis, representação de dados, conjuntos de instruções, programação em linguagem de montagem e geração de código em linguagem de máquina;
- BIP II: acrescenta instruções de desvio, com foco na inclusão de suporte aos conceitos de estruturas de controle para desvios condicionais e incondicionais e laços de repetição; e
- BIP III: acrescenta instruções de lógica focando na inclusão de suporte a operações de lógica bit a bit.

A iniciativa de desenvolver uma família de processadores permitiu que fosse dado um enfoque interdisciplinar aos conceitos de arquitetura e organização de computadores. Essa característica tornou-se um diferencial desta abordagem, uma vez que teve desdobramentos em diferentes disciplinas correlatas, a saber: Computação Básica, Circuitos Digitais, Arquitetura e Organização de Computadores e Compiladores. Dessa forma, a abordagem adotada fornece uma contribuição significativa, uma vez que, muito frequentemente, os alunos relatam que não conseguem compreender plenamente as relações entre as diferentes disciplinas.

Dos três modelos de processadores definidos, foram escolhidos para utilização no projeto Bipide os dois primeiros, por possibilitarem apresentar a implementação em hardware de diversas abstrações estudadas nas aulas iniciais das disciplinas da área de Algoritmos e Programação [14].

2.1 Processador BIP I

O processador BIP I possui uma máquina orientada a acumulador e não possui banco de registradores. Todas as operações de transferência e aritmética envolvem o acumulador e em algumas operações aritméticas é necessário buscar os operandos da memória de dados [11].

Em sua arquitetura, foi definido um único formato de instrução para a representação de todas as instruções que compõem o conjunto de instruções do processador. Esse formato, ilustrado na Figura 1, é composto por dois campos, incluindo um código de operação de 5 bits e um operando explícito de 11 bits.

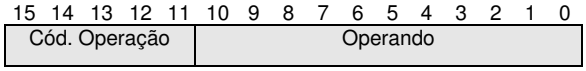

Figura 1: Formato de instrução

A arquitetura do BIP I inclui três registradores: (*i*) PC (Program Counter): aponta para o endereço da próxima instrução; (*ii*) IR (Instruction Register): armazena a instrução que está em execução; e (*iii*) ACC (Accumulator): operando implícito utilizado para armazenamento temporário de dados durante uma operação.

Pelo fato de ter um único registrador de propósito geral (o ACC), foi considerado que as operações aritméticas e lógicas poderiam processar operandos oriundos da memória de dados. Dessa forma, são suportados os modelos de execução Registrador-Registrador e Registrador-Memória.

O conjunto de instruções do BIP I é formado por oito instruções, incluindo instruções de controle (HLT), armazenamento em memória (STO), carga no acumulador (LD e LDI) e de aritmética (ADD, ADDI, SUB e SUBI), as quais são descritas no Quadro 1, onde Mem significa Memória.

Cód. da Operação	Instrução	Operação
00000	HLT	Paralisa a execução
00001	STO operando	Mem[operando]← ACC
00010	LD operando	ACC ← MD[operando]
00011	LDI operando	ACC ← operando
00100	ADD operando	ACC ← ACC + Mem[operando]
00101	ADDI operando	ACC ← ACC + operando
00110	SUB operando	ACC ← ACC – Mem[operando]
00111	SUBI operando	ACC ← ACC – operando

Quadro 1: Conjunto de instruções do BIP I

A instrução HLT tem a função de desabilitar a atualização do PC, paralisando a execução do programa. Nas demais instruções, o PC é incrementado em uma unidade. A instrução STO (*Store*) realiza a transferência do conteúdo do registrador ACC para uma posição do espaço de endereçamento de memória. Com relação às instruções de carga, a instrução LD (*Load*) realiza uma operação de transferência de uma posição do espaço de endereçamento de memória para o acumulador, enquanto que a instrução LDI (*Load Immediate*) carrega uma constante (o operando) no acumulador. Também são disponibilizadas instruções de soma e de subtração entre o acumulador e uma posição do espaço de endereçamento de memória (ADD e SUB) e entre o acumulador e uma constante (ADDI e SUBI).

Com relação aos modos de endereçamento, as instruções LDI, ADDI e SUBI utilizam o Modo Imediato, enquanto que as instruções STO, LD, ADD e SUB utilizam o Modo Direto. Em todas essas instruções, o acumulador é utilizando como um operando implícito, atuando como operando fonte e/ou destino das operações realizadas.

A organização dos processadores BIP apresenta memórias separadas para dados e instruções, como visto na Figura 2. A Unidade Central de Processamento (UCP) é dividida em dois blocos: (*i*) Controle: inclui o registrador PC, um somador para atualizar o valor do PC e o decodificador que gera os sinais de controle necessários para a execução de cada instrução; e (*ii*) Caminho de Dados: inclui o registrador ACC, uma unidade aritmética de soma e de subtração e um circuito para estender o sinal do operando (de 11 para 16 bits).

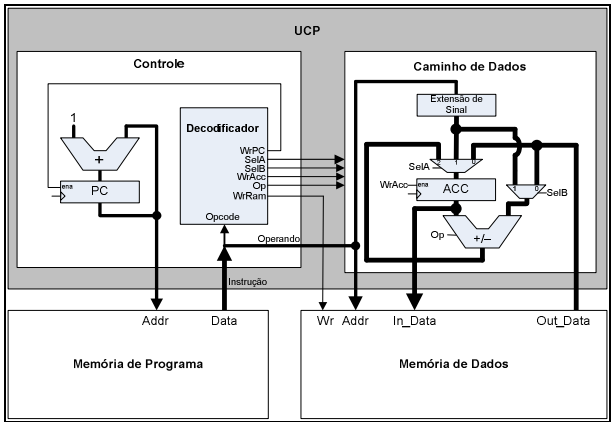

Figura 2: Organização do Processador BIP I

2.2 Processador BIP II

Trata-se de uma extensão do BIP I e possui as mesmas características arquiteturais. Foi especificado para incluir o suporte a estruturas de controle para a implementação de desvios e de laços de repetição. Para tal, o conjunto de instruções do BIP I foi estendido incluindo seis instruções de desvio condicional (BEQ, BNE, BGT, BGE, BLT e BLE) e uma instrução de desvio incondicional (JMP), as quais são descritas no Quadro 2 [9].

Cód. da Operação	Instrução	Operação	Classe
01000	BEQ operando	Se (STATUS.Z=1) então PC ← operando Se não PC ← PC + 1	Desvio condicional
01001	BNE operando	Se (STATUS.Z=0) então PC ← operando Se não PC ← PC + 1	Desvio condicional
01010	BGT operando	Se (STATUS.Z=0) e (STATUS.N=0) então PC ← operando Se não PC ← PC + 1	Desvio condicional
01011	BGE operando	Se (STATUS.N=0) então PC ← operando Se não PC ← PC + 1	Desvio condicional
01100	BLT operando	Se (STATUS.N=1) então PC ← operando Se não PC ← PC + 1	Desvio condicional
01101	BLE operando	Se (STATUS.Z=1) ou (STATUS.N=1) então PC ← operando Se não PC ← PC + 1	Desvio condicional
01110	JMP operando	PC ← operando	Desvio incondicional

Quadro 2: Conjunto de instruções do BIP II

Para suportar as instruções de comparação e desvio condicional, foi acrescentado um registrador de estado à CPU, o qual é denominado STATUS e inclui dois os *flags*: Z (Zero) e N (Negative). Toda instrução de comparação e desvio condicional deve ser precedida por uma instrução de subtração (SUB ou SUBI). Dependendo do resultado dessa operação, os *flags* Z (= 0) e N (< 0) são definidos em 0 ou em 1. As instruções de desvio condicional então verificam o valor desses *flags* para determinar se o desvio deve ser tomado ou não, conforme o tipo de comparação associado [9]. As instruções de desvio utilizam o modo de endereçamento Imediato quando carregam o endereço de desvio no PC, o qual é considerado um operando implícito.

A Figura 3 apresenta a organização Harvard monociclo do processador BIP II. Em relação à organização do processador BIP I, ilustrada previamente na Figura 2, essa organização inclui os circuitos necessários à implementação das instruções de desvio: um multiplexador na entrada do PC para permitir a carga do valor do operando e o registrador STATUS, com os *flags* Z e N (abaixo e à direita da Unidade Aritmética no Caminho de Dados).

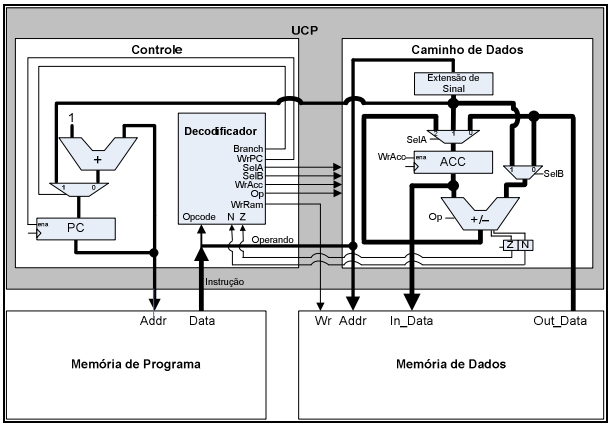

Figura 3: Organização do Processador BIP II

2.3 Enfoque interdisciplinar

A criação de processadores com cunho didático normalmente busca apoiar as disciplinas diretamente envolvidas com o ensino de arquitetura e/ou de organização de computadores. O desenvolvimento dos processadores BIP buscou incluir possibilidades de exploração dos conceitos envolvidos em diferentes disciplinas de cursos superiores na área de Ciência da Computação.

A crença inicial que motivou o enfoque interdisciplinar na concepção da família de processadores BIP foi de que o uso de um modelo simplificado de computador auxilia a melhorar a compreensão de conceitos relacionados à aprendizagem inicial de programação.

Entende-se que, ao reduzir o grau de abstração envolvido com a aprendizagem de conceito ligados ao uso de memória, operações aritméticas, desvios e laços, pode-se facilitar a aprendizagem dos alunos que apresentam problemas em lidar com abstrações. O Quadro 3 destaca algumas relações que podem ser estabelecidas entre as abstrações estudadas nas disciplinas da área de programação e sua representação no nível arquitetural.

Conceitos de Programação	Conceitos de Arquitetura de Computadores
Variável	Posição na memória
Constante	Operando imediato na instrução
Atribuição	Acesso à memória para leitura e/ou escrita de/em uma posição
Operação aritmética	Utilização de unidade de soma/subtração
Comandos com múltiplas operações	Uso de uma instrução para cada operação realizada
Desempenho dos programas	Número de instruções na linguagem de montagem
Papel do compilador	Tradução da linguagem de alto nível para a linguagem de montagem

Quadro 3: Relações entre conceitos de Programação e de Arquitetura suportadas pelo BIP

Entende-se também que, para facilitar a associação entre os conceitos dessas disciplinas, é importante a representação visual dos componentes da organização do processador e a ilustração de como a execução de um algoritmo modifica os valores na memória e nos registradores, mostrando-se o fluxo da informação entre os diferentes componentes do hardware. Essa ilustração é um aspecto importante para efetivamente reduzir o grau de abstração envolvido na aprendizagem desses conceitos.

Conforme Garcia, Rezende e Calheiro [23], a ilustração de algoritmos tem se mostrado uma estratégia interessante para auxiliar a aprendizagem. Mariani [24] complementa que é perceptível a facilidade com que os estudantes desenvolveram novos programas em função da realimentação oferecida pelo ambiente gráfico. Já Gomes [25] enfatiza que a eficácia da interface de um software educativo serve como elemento mediador à construção de situações significativas para a aprendizagem de conceitos específicos.

3 O Ambiente Bipide

O ambiente Bipide insere-se no contexto apresentado disponibilizando um ambiente de desenvolvimento integrado (IDE – Integrated Development Environment) que

possibilita o desenvolvimento, execução e simulação de programas em linguagem Portugol, relacionando esses programas à arquitetura dos processadores BIP.

Esse ambiente é composto por: (*i*) um editor de textos destinado ao desenvolvimento de programas em linguagem Portugol; (*ii*) um compilador que traduz a linguagem Portugol para a linguagem de montagem dos processadores BIP; (*iii*) um simulador dos processadores BIP I e II capaz de simular as instruções desses e exibir graficamente sua execução por meio de animações; e (*iv*) um módulo de ajuda composto por informações teóricas relacionadas à arquitetura e à organização dos processadores BIP e funcionalidades do sistema.

3.1 Projeto

A análise de ferramentas de simulação de arquitetura (Seção 4) levou ao desenvolvimento do projeto aqui apresentado. Foram definidos os requisitos do sistema, diagramas de caso de uso, protótipos de interface e a descrição da gramática a ser utilizada para a construção do compilador. Os requisitos definiram as seguintes características básicas da IDE:

- Escrita e compilação de programas em linguagem Portugol;
- Indicação dos erros encontrados no programa durante a compilação;
- Execução do programa passo a passo;
- Geração de código *assembly* para os processadores BIP I e BIP II; e
- Simulação e animação do funcionamento do programa sobre os processadores BIP I e BIP II.

3.2 Implementação

Para a implementação do Bipide, utilizaram-se duas ferramentas de desenvolvimento de software: Visual Studio 2008 [7] e Expression Blend 2 [6]. O Visual Studio 2008 foi utilizado para a implementação da interface e das classes que compõem o sistema em linguagem C#. Os elementos gráficos e animações que compõem o simulador do processador BIP foram desenvolvidos em WPF (Windows Presentation Foundation) através da ferramenta Expression Blend 2. Essa ferramenta apresenta uma interface orientada por design para aplicações baseadas na linguagem XAML (eXtensible Application Markup Language) e apresenta compatibilidade com a ferramenta Visual Studio 2008.

A ferramenta para geração de compiladores ANTLRWorks [10] foi utilizada para a implementação dos analisadores léxico e sintático do compilador integrado ao ambiente. Também foram definidas através do ANTLRWorks as ações semânticas necessárias para a etapa de geração de código do compilador. Utilizou-se o editor FireEdit [13] para a implementação do editor de código. Esse editor apresenta funcionalidades que auxiliam a escrita de programas como, por exemplo, a utilização de cores diferenciadas para identificar palavras reservadas e símbolos da linguagem (*syntax highlighting*) e identificação de aberturas e fechamentos de parênteses.

3.3 Visão geral das funcionalidades e dos aspectos da interface gráfica

O Bipide apresenta três módulos principais: (*i*) Programação: permite ao usuário escrever e compilar programas; (*ii*) Simulação: permite simular os programas desenvolvidos; e (*iii*) Ajuda: apresenta informações a respeito das funcionalidades do sistema e sobre a arquitetura e a organização dos processadores BIP.

Para a estruturação dos módulos dentro do ambiente utilizou-se o recurso de abas. Na parte superior da interface, são apresentados os painéis de opções através dos quais o usuário pode interagir com o sistema e ter acesso as suas funcionalidades. A Figura 4 apresenta a interface do módulo de programação.

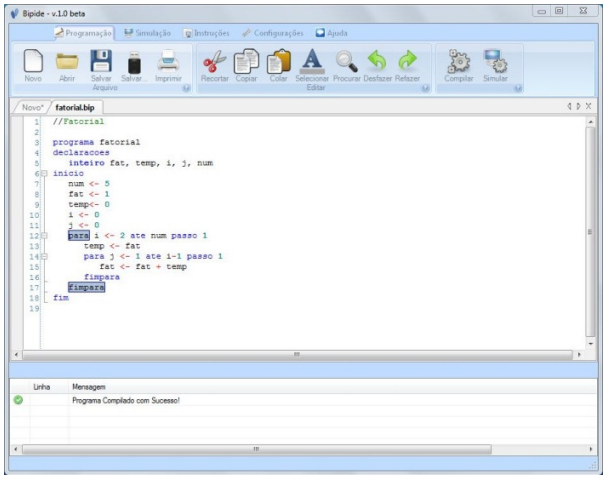

Figura 4: Interface do módulo de programação

O módulo de programação oferece ao usuário recursos e funcionalidades típicas de editores de código fonte como, por exemplo, identificação de palavras reservadas e símbolos da linguagem. Oferece, ainda, opções de gerenciamento e edição de arquivos. Na parte inferior do módulo, são exibidas mensagens de erro ocorridas durante a compilação do programa e outras mensagens que sejam relevantes ao seu desenvolvimento.

No ambiente de simulação, ilustrado na Figura 5, o usuário pode executar o programa passo a passo e simular sua execução nos processadores BIP. Na parte superior, são exibidos o código do programa em linguagem alto

nível e o código *assembly* correspondente que foi gerado durante a compilação do programa. Durante a simulação, a linha do programa Portugol que está sendo executada é destacada, assim como o conjunto de instruções *assembly* que correspondem à instrução em Portugol. Dessa forma, permite-se ao usuário verificar quais instruções *assembly* geradas correspondem àquela linha do programa.

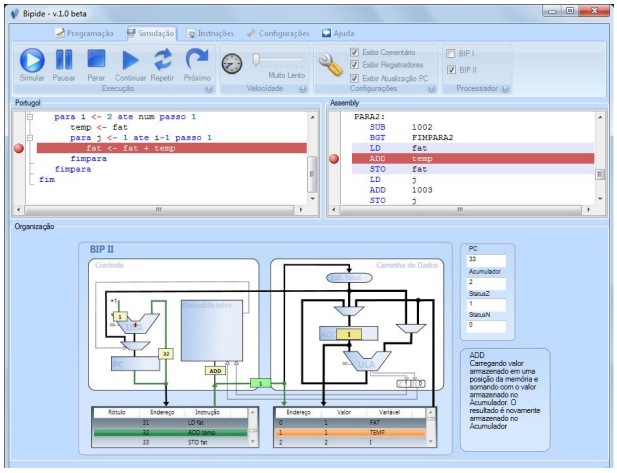

Figura 5: Interface do módulo de simulação

A simulação do programa é realizada através de animações que representam a execução do programa nos processadores BIP I e BIP II. São representados os registradores especiais do processador, as memórias de dados e instrução e seus respectivos valores. A fim de auxiliar o usuário a compreender o que está ocorrendo no processador, são apresentadas mensagens descrevendo as ações executadas e é possível visualizar informações a respeito dos principais elementos do processador. Também são utilizadas diferentes cores para distinguir os valores que representam endereços de memória e dados.

A interface do módulo de simulação foi concebida para que o usuário visualizasse, ao mesmo tempo, os três elementos utilizados: linguagem alto nível, linguagem *assembly* e organização do processador. Isso permite que sejam realizadas comparações e associações entre estes três elementos, colaborando para a redução da abstração apresentada nos conceitos de programação. A Figura 6 ilustra uma das relações estabelecidas, na qual o mesmo valor pode ser ver visto nos três elementos da interface.

O módulo de ajuda é composto por informações teóricas a respeito de Arquitetura e Organização de computadores e apresenta a descrição dos processadores BIP, auxiliando o usuário a compreender a arquitetura e o conjunto de instruções desses processadores. Esse módulo descreve ainda aspectos da interface e as opções do ambiente. O Bipide apresenta ajuda sensível ao contexto, acessível através de ícones de ajuda em cada um dos principais elementos da interface, os quais exibem ao usuário informações a respeito desses elementos.

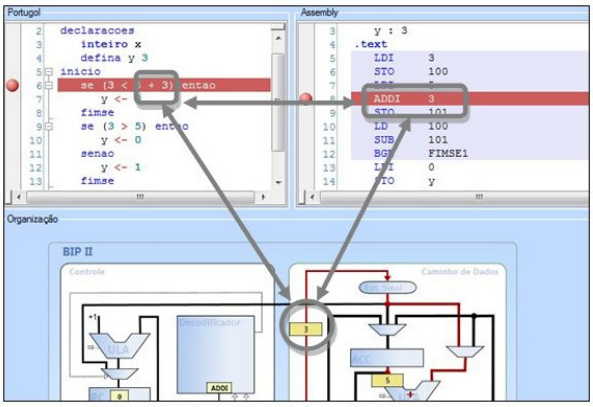

Figura 6: Relação de elementos na interface do módulo de simulação

4 Trabalhos Relacionados

O uso de simuladores é essencial para o ensino adequado da arquitetura de computadores, pois facilita a compreensão do funcionamento dos processadores utilizados. Esses simuladores costumam apresentar uma interface de usuário pouco elaborada e com poucos recursos operacionais, o que pode ocasionar dificuldade em utilizá-los em sala de aula [1, 2].

Com o objetivo de identificar funcionalidades relevantes em ambientes de simulação, foi realizada uma análise das características de alguns simuladores de arquitetura com caráter educacional. Um resumo das funcionalidades identificadas nesses simuladores pode ser visto no Quadro 4, que inclui as funcionalidades do ambiente Bipide. O quadro permite identificar que as características implementadas no Bipide englobam as principais funcionalidade existentes nos trabalhos similares.

Nota-se a partir do Quadro 4 que poucas ferramentas simulam a organização no aspecto considerado fundamental para os objetivos desta pesquisa. Dentre essas ferramentas, apenas o MipsIt possui a possibilidade de programação em linguagem de alto nível, permitindo relacionar conceitos de programação com arquitetura e organização de computadores. No entanto, a arquitetura do processador MIPS, ilustrada no MipsIt, possui complexidade acima do desejável para ilustração de conceitos aos alunos que estão ingressando em cursos de Computação. Já arquitetura do BIP, tendo sido criada para propósito didático, possui a simplicidade desejada e é ilustrada pelo Bipide utilizando o conceito de interface rica (RIA - *Rich Interface Applications*). O Bipide fornece suporte para uso em atividades de ensino em diferentes disciplinas sobre programação introdutória, arquitetura de computadores, circuitos digitais, compiladores e outras.

Funcionalidade	4AC [21]	MipsIt [2]	Neander Win [1]	R10k [20]	Simularq [19]	VLIW-DLX [18]	Bipide
Programação em alto nível	Não	Sim	Não	Não	Não	Não	Sim
Programação em *assembly*	Não	Sim	Sim	Sim	Sim	Sim	Sim
Visualização da Memória	Sim	Sim	Sim	Sim	Sim	Sim	Sim
Visualização dos valores dos registradores	Sim	Sim	Sim	Sim	Sim	Sim	Sim
Execução Passo a Passo	Sim	Sim	Sim	Sim	Sim	Sim	Sim
Simulação da Organização	Não	Sim	Não	Não	Sim	Não	Sim
Identificação de erros de compilação	N.I.	N.I.	Sim	Sim	N.I.	Sim	Sim
Alteração de valores em tempo de execução	N.I.	Não	Não	Sim	N.I.	Não	Não
Arquivos de Ajuda	Não	Não	Não	Sim	Sim	Não	Sim

Quadro 4: Funcionalidades dos simuladores analisados (onde: N.I. = Não Identificado)

5. Testes e Experimentos

Os testes do Bipide foram realizados visando a eliminação de erros existentes no projeto e no desenvolvimento. Durante a implementação das ações semânticas do compilador, foi utilizado um conjunto de aplicações Portugol para sua validação. Esses programas foram compilados e os códigos *assembly* gerados foram analisados e comparados com os resultados esperados.

Para a validação do simulador, foram realizados testes para verificar a simulação de cada instrução do processador. Um conjunto de instruções *assembly* foi utilizado para verificar o estado esperado para os registradores e memórias. Esse mesmo conjunto de instruções já havia sido utilizado para validação da arquitetura do microcontrolador μBIP[1] [11] e, após serem simulados no ambiente Bipide, tiveram os valores resultantes analisados e comparados com o obtido previamente.

Foram conduzidos dois experimentos em sala de aula. O primeiro deles mais focalizado na aceitação da interface e o segundo buscando avaliar os aspectos educacionais ligados ao uso do Bipide.

5.1 Primeiro Experimento

A fim de avaliar a satisfação dos usuários e verificar possíveis melhorias a serem realizadas na interface do Bipide, foi realizado um experimento de utilização com alunos da disciplina de Algoritmos e Programação da Universidade do Vale do Itajaí. Para este experimento foram definidas algumas variáveis de interesse que nortearam a elaboração de um questionário sobre a interface e a usabilidade do ambiente apresentadas no Quadro 5.

O Bipide foi instalado nos computadores do laboratório de informática da disciplina, onde foi utilizado por 21 alunos. Esses alunos puderam usar a ferramenta livremente e foram solicitados a desenvolver um conjunto de algoritmos que pudessem ser compilados no Bipide, podendo, dessa forma, explorar o ambiente e descobrir as funcionalidades oferecidas pelo mesmo. Após sua utilização, foi aplicado um questionário composto por seis perguntas a respeito da qualidade da interface do ambiente.

1. Você acha que o seu nível de conhecimento foi compatível com as telas e vocabulário empregado no sistema?
2. As mensagens de erro lhe auxiliaram de forma adequada?
3. O sistema lhe proporcionou feedback, ou seja, você conseguiu visualizar todas suas ações?
4. A estrutura dos painéis do menu lhe parece disposta de forma lógica por agrupamento de tipos de opções?
5. Qual é o grau de satisfação que você atribui à utilização do sistema?
6. Aponte sugestões e críticas gerais para a melhoria do ambiente.

Quadro 5: questões da avaliação de aceitação pelos usuários

Dos alunos que responderam o questionário, 78,1% atribuíram notas de 8 a 10 às questões do instrumento, como pode ser observado na Tabela 1, o que demonstra indícios da aceitação dos alunos com a utilização da ferramenta.

Questão	Nota Média	Desvio Padrão
Questão 1	7,95	2,11
Questão 2	7,90	2,51
Questão 3	8,33	2,44
Questão 4	9,19	1,12
Questão 5	8,62	1,86

Tabela 1: Avaliação da interface do Bipide

5.2 Segundo Experimento

O segundo experimento foi concebido para coletar evidências empíricas de que a redução da abstração proporcionada pela introdução dos conceitos da arquitetura BIP e pelo uso do Bipide poderiam trazer benefícios positivos para os aprendizes de programação introdutória.

A população de interesse dessa pesquisa são alunos ingressantes em cursos de Ciência da Computação. Dessa forma, organizou-se o experimento com duas amostras de alunos ingressantes em curso de Ciência da Computação. As amostras foram selecionadas de forma não aleatória (quase-experimental), compreendendo as turmas de alunos ingressantes nos semestres 2008/2 (n=49) e

[1] O μBIP é uma quarta versão dos processadores BIP. Ele estende a arquitetura das versões anteriores e acrescenta suporte a instruções de deslocamento lógico, manipulação de vetores, suporte a procedimentos e funcionalidades típicas de microcontroladores [8].

2009/2 (n=66). Os seguintes tratamentos distintos foram dados a estas turmas:

Amostra 1: Grupo de Controle

- *Integrantes*: 49 Alunos ingressantes na disciplina de algoritmos e programação na UNIVALI Campus Itajaí no semestre 2008/2.
- *Tratamento*: Estes alunos não receberam nenhuma explicação sobre arquitetura do processador BIP, foram apenas informados da existência da arquitetura Von Neumann e da sua influência no funcionamento dos programas.

Amostra 2: Grupo Experimental

- *Integrantes*: 66 Alunos ingressantes na disciplina de algoritmos e programação na UNIVALI campus Itajaí no semestre 2009/2.
- *Tratamento*: Estes alunos tiveram uma aula sobre a arquitetura do processador BIP, na qual foram apresentados exemplos de códigos em Portugol e sua representação equivalente no *assembly* do BIP. Foram realizados exercícios sobre a construção de programas em linguagem Portugol e em linguagem de montagem usando a ferramenta Bipide em laboratório. Ao todo o tratamento compreendeu um período de 4 horas aula.

O mesmo instrumento de avaliação foi aplicado aos alunos destas duas amostras. O instrumento é composto por 13 questões e aborda os temas pertinentes a programação introdutória. Cada questão foi classificada conforme os conceitos envolvidos e conforme o objetivo educacional, usando para isso a taxonomia de Bloom [15]. O Quadro 6 ilustra esta classificação.

Questão	Objetivo educacional	Conceitos
1	Conhecimento	Conceito de algoritmo
2	Compreensão	Conceito de algoritmo
3	Compreensão	Papel do Compilador
4	Conhecimento	Linguagens de Programação
5	Aplicação	Tipos, Variáveis e Constantes
6	Aplicação	Variáveis, Constantes e Operações Aritméticas
7, 8, 9 e 10	Análise	Variáveis, Atribuições e Operações Aritméticas
11, 12 e 13	Aplicação	Variáveis, Constantes, Atribuições e Operações Aritméticas

Quadro 6: questões relacionadas aos conceitos avaliados

A aplicação do instrumento nas duas amostras forneceu as seguintes variáveis que foram utilizadas como variáveis dependentes a *nota média* e a *proporção de acertos* (por questão). Como variável independente foi considerada a *utilização do sistema Bipide* (e a decorrente explanação sobre a arquitetura do BIP)

Para responder a pergunta de pesquisa foi definida a seguinte hipótese: *A média no Grupo Experimental é maior do que no Grupo de Controle.*

Uma vez existindo uma melhora significativa na média, seriam buscadas as questões que contribuíram significativamente para elevar a média, analisando para isso as proporções de acertos.

Por tratarem-se de amostras grandes (n > 30) assume-se a característica da normalidade das amostras e a conseqüente utilização de testes paramétricos [16].

Utilizou-se inicialmente o teste F para identificar se ambas as amostras possuem variâncias iguais. Após, utilizou-se o teste T para amostras independentes com variâncias iguais, a fim de testar a hipótese com grau de confiança de 99%.

Para identificação das questões que colaboraram mais fortemente para melhoria da média utilizamos o teste Z para comparação de proporções.

O Quadro 7 sintetiza o design experimental adotado.

Síntese do Design do Experimento	
Tipo de *design*	Quase-experimental, três grupos, apenas pós-teste, seleção não aleatória.
População	Ingressantes em cursos de Ciência da Computação
Amostras	Grupo de Controle (GC): turma 2008/2 (n=49) Grupo Experimental (GE): turma 2009/2 (n=66)
Instrumento de coleta	Prova categorizadas aplicada as duas amostras
Variáveis Dependentes	Nota Média, Proporção de acertos
Variáveis Independentes	Utilização do Bipide
Hipóteses	$H_0 : \mu_{GC} \geq \mu_{GE}$ $H_1 : \mu_{GC} < \mu_{GE}$
Procedimentos Estatísticos	Teste F (comparação de variâncias a 95%) Teste T (comparação de médias a 99%) Teste Z (comparação de proporções a 95%)

Quadro 7: Síntese do design do experimento

5.2.2 Controle das ameaças à validade do experimento

Ameaças à validade interna

As principais ameaças internas à validade do experimento (conforme [17]) e as ações tomadas para reduzir ou anular seus efeitos são listadas a seguir:

- *Ameaça de instrumentação*: Relacionada a uma diferença entre o grau de dificuldade dos instrumentos de coleta, foi eliminada pela utilização do mesmo instrumento para as duas amostras.

- *Ameaça de testagem*: Em que o instrumento promove uma aprendizagem nos participantes que influenciam um resultados futuro. Não se aplica pois o desenho possui apenas pós teste.
- *Ameaça de maturação*: Em que os participantes evoluem naturalmente no intervalo entre pré e pós teste. Também não se aplica por haver apenas pós teste no *design* adotado.
- *Ameaça de história*: Em que um evento externo ao experimento possa ter influenciado o resultado. Esta ameaça é muito difícil de eliminar completamente e os esforços para minimizar sua ação foram: (*i*) aplicar o instrumento no mesmo período de cada semestre; (*ii*) manter contínuo o plano de aulas exceto no que for relacionado ao tratamento; (*iii*) utilização do mesmo professor em todas as amostras; e (*iv*) não aplicação de aulas de recuperação ou de revisão de conteúdos.
- *Ameaça de mortalidade seletiva*: Em que uma das amostras possui maior propensão ao abandonar o experimento. Foram comparadas turmas que ingressam no vestibular de inverno e que geralmente apresentam perfil semelhante. Nenhuma iniciativa explicita foi tomada para minimizar esta ameaça.
- *Ameaça de contaminação*: Em que integrantes de uma amostra repassam conhecimentos a elementos de outra amostra. Esta ameaça foi minimizada pela diferença temporal entre as turmas (semestres distintos). No entanto a presença de alguns alunos repetentes pode ter causado algum grau de contaminação não mensurado pelos pesquisadores.
- *Ameaça de comportamento competitivo*: Em que uma amostra busca ter melhores resultados que o grupo de controle. Não deverá ter efeito significativo uma vez que os alunos não foram informados de que estavam participando de uma experiência comparativa.
- *Ameaça de comportamento compensatório*: Em que medidas compensatórias são dadas a uma das amostras. Não ocorreu por parte dos experimentadores nenhum tipo de comportamento compensatório.
- *Ameaça de expectativa do sujeito*: Em que o fato de estar sendo avaliado altera a performance do sujeito. Não se aplica pois os alunos não sabiam que estavam participando de um experimento.
- *Ameaça de expectativa do experimentador*: Em que o interesse do pesquisador cria algum viés no experimento. Esta ameaça pode ter alguma influência uma vez que um dos pesquisadores atou como professor das turmas utilizadas como amostra e inconscientemente pode ter influenciado.

Ameaça à validade externa

Por tratar-se de um desenho quase-experimental onde a amostragem não é aleatória, a generalização dos resultados para a população não pode ser feita.

5.2.3 Descobertas

As duas amostras utilizadas no experimento são descrias pela Tabela 2.

Amostra	Média (X)	*n*	Variância (σ^2)
Grupo Experimental	7,052	66	5,106
Grupo de Controle	5,716	49	7,034

Tabela 2: Descrição das amostras

A utilização do teste F permitiu identificar a igualdade das variâncias entre as amostras (Fo = 1,377) e, desta forma, indicou a modalidade de teste T seria adequada para a comparação entre as médias (Teste T para duas amostras independentes com variância iguais).

O teste T foi utilizado para verificação da hipótese experimental. A hipótese Nula (H_0) é de que a média do grupo de controle é maior ou igual a média do grupo experimental e a hipótese alternativa (H_1) é de que a média do grupo de experimental é maior que do grupo de controle.

$$H_0 : \mu_{GC} \geq \mu_{GE}$$

$$H_1 : \mu_{GC} < \mu_{GE}$$

A equação do teste T (Equação 1) fornece um valor de T observado (To) que é comparado com um valor crítico tabelado que depende do grau de confiança utilizado.

$$T_o = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{\frac{(n_1-1)\cdot\sigma_1^{\,2} + (n_2-1)\cdot\sigma_2^{\,2}}{(n_1+n_2-2)}} \cdot \sqrt{\frac{1}{n_1}+\frac{1}{n_2}}}$$

Equação 1: Teste T de Student

Ao aplicar a Equação 1, obteve-se o valor de To = 2,910 e percebeu-se que este é maior que o valor crítico tabelado para o grau de confiança de 99%. Ou seja é possível rejeitar a hipótese nula (e consequentemente aceitar a hipótese alternativa) com 99% de confiança. A média do grupo experimental é estatisticamente maior do que do grupo de controle.

A seguir aplicou-se o teste Z em cada uma das 13 questões do instrumento utilizado para verificar quais apresentaram melhoras na proporção de acertos.

Quatro questões apresentaram melhorias com grau de confiança de 95%, são elas as questões 6, 9, 11 e 12. Essas questões referem-se aos conceitos de variáveis, constantes, atribuições e operações aritméticas, que são

conceitos onde a abstração foi reduzida com a explicação sobre o BIP e utilização do Bipide. Adicionalmente, conforme a classificação do Quadro 6 as questões estão ligadas a objetivos de aprendizagem de análise e aplicação que estão nos níveis superiores da taxonomia de Bloom. Isso indica que os alunos apreenderam os conceitos e tiveram aptidão para utilizá-los para resolução de novos problemas. Considera-se esta mais uma evidência da efetividade da proposta didática aplicada.

5.2.4 Observações do Professor

Além dos resultados quantitativos a utilização do Bipide em sala de aula permitiu ao professor realizar observações sobre a aprendizagem dos alunos em sala de aula. Os principais aspectos relatados são descritos a seguir:

- Os alunos demonstraram alguma dificuldade inicial em compreender as instruções de linguagem de montagem, mas rapidamente passaram a adquirir fluência na resolução dos exercícios;
- A compreensão da existência do acumulador tornou a operação de atribuição mais clara. O que acontecia com o resultado intermediário da operação A + 1 em A ← A + 1 antes ficava obscuro;
- Um problema recorrente em vários semestres em que alguns alunos declaravam constantes inteiras como se fossem variáveis (Ex: inteiro 5) ou ainda realizavam operações de atribuição para constantes (Ex: 5 ← 3 + 2) não ocorreram na amostra experimental; e
- Por apresentar aspectos utilizados em outras disciplinas, ficou claro aos alunos do primeiro período que os conceitos seriam importantes também para a seqüência do curso.

6. Conclusões

Os resultados dos experimentos permitem confirmar a efetividade da abordagem proposta, uma vez que os alunos puderam consolidar os conceitos estudados na disciplina, e demonstram indícios de melhoria da utilização desses conceitos para análise e aplicação em novas situações. Desta forma, considera-se positiva a utilização do ambiente Bipide como ferramenta de apoio na adoção de um modelo simplificado de processador para auxiliar o entendimento das abstrações presentes nos conceitos introdutórios de programação.

A abordagem utilizada anteriormente à existência do Bipide resumia-se a uma breve explicação sobre o funcionamento de um computador, ilustrando os componentes básicos da arquitetura de Von Neumann, e, em seguida, realizava-se a introdução de conceitos como variáveis, atribuições e operações de entrada e saída.

Com o Bipide, pode-se detalhar a explicação dos conceitos básicos de arquitetura de computadores e os conceitos apresentados não restringiram-se apenas à disciplina de Algoritmos e Programação.

A análise dos simuladores de arquitetura de caráter educacional, apresentada na Seção 4, permite perceber que o Bipide privilegiou contemplar as principais características verificadas nesses trabalhos. Considera-se que assim torna-se possível ao professor e aos alunos explorarem o uso da ferramenta em uma diversidade maior de possibilidades de aprendizagem.

As perspectivas futuras desta pesquisa incluem: (*i*) implementar o suporte à arquitetura do microcontrolador µBIP provendo instruções para entrada e saída de dados; (*ii*) desenvolver compiladores para diferentes linguagens de alto nível, como C e Java; (*iii*) implementar suporte a diferentes formatos numéricos de representação, como hexadecimal e binário; (*iv*) implementar operações de multiplicação através da expansão de macros; e (*v*) implementar um módulo de otimização do código *assembly* gerado.

Além desses aprimoramentos técnicos, pretende-se também dar continuidade à realização de experimentos controlados com alunos ingressantes a fim de ampliar a força da evidência de melhoria de aprendizagem e também investigar a influência da abordagem em outras variáveis como taxa de reprovação e taxa de abandono.

Agradecimento

Os autores agradecem ao CNPq pelo apoio concedido para realização da pesquisa via Edital Universal 2008, Processo No 477820/2008-5, e à UNIVALI pelo contínuo suporte as atividades dessa pesquisa.

Referências

[1] J. A. S. Borges, G. P. Silva. NeanderWin – Um simulador didático para uma arquitetura do tipo Acumulador. In *Workshop Sobre Educação em Arquitetura de Computadores*, Ouro Preto, 2006.

[2] P. Borunda, C. Brewer, Erten, C. GSPIM: graphical visualization tool for MIPS assembly programming and simulation. In *Technical Symposium on Computer Science Education*, New York, NY, p. 244-248, 2008.

[3] A. Carbone, J. Kaasboll. A survey of methods used to evaluate computer science teaching. In Proceedings of the 3rd Conference on Teaching of Computing, Dublin, p. 41-45, 1998.

[4] J. Good, P. Brna. Program comprehension and authentic measurement: a scheme for analyzing descriptions of programs, Journal of Human-Computer Studies, p. 169-185, 2004.

[5] C. S. Menezes, A. M. Nobre. Um ambiente cooperativo para apoio a cursos de introdução a programação. In *Workshop de Educação em Computação*, Anais do XXII Congresso da Sociedade Brasileira de Computação, Porto Alegre, 2002.

[6] Microsoft Corporation. Microsoft Expression. http://www.microsoft.com/Expression, Jun, 2009.

[7] Microsoft Corporation. Microsoft Visual Studio. http://www.microsoft.com/visualstudio/en-us/default.mspx, Jun 2009.

[8] D. Morandi, et al. Um processador básico para o ensino de conceitos de arquitetura e organização de computadores. In *Hífen, Uruguaiana*, v. 30, páginas 73-80, 2006.

[9] D. Morandi, A. L. A. Raabe, C. A. Zeferino. Processadores para Ensino de Conceitos Básicos de Arquitetura de Computadores. In *Proceedings of the 18th International Symposium on Computer Architecture and High Performance Computing – Workshops*, Porto Alegre, páginas 17-24, 2006.

[10] T. Parr. ANTLRWorks: The ANTLR GUI Development Environment. http://www.antlr.org/works/index.html, Jun 2009.

[11] M. C. Pereira. µBIP: Microcontrolador Básico para o Ensino de Sistemas Embarcados. Trabalho de Conclusão de Curso, Ciência da Computação, Universidade do Vale do Itajaí, Jun 2008.

[12] E. P. Pimentel, et al. Avaliação contínua da aprendizagem, das competências e habilidades em programação de computadores. In *Workshop de Informática na Escola*, Anais do XXIII Congresso da Sociedade Brasileira de Computação, Porto Alegre, 2003.

[13] The Code Project. Fireball.CodeEditor. http://www.codeproject.com/KB/miscctrl/fireballcodeeditor.aspx, Jun, 2009.

[14] P. V. Vieira. Bipide: Ambiente de Desenvolvimento Integrado para a Arquitetura dos Processadores BIP. Trabalho de Conclusão de Curso, Ciência da Computação, Universidade do Vale do Itajaí, Jul 2009.

[15] B. Bloom, Taxonomy of Educational Objectives (1956). Published by Allyn and Bacon, Boston, MA. Copyright (c) 1984 by Pearson Education.

[16] M. Triola. Essentials of Statistics. Addison-Wesley, 2006.

[17] J. Wainer, Métodos de pesquisa quantitativa e qualitativa para Ciência da Computação. Jornadas de Atualizações em Informática, Sociedade Brasileira de Computação, 2007.

[18] M. Bečvář,; S. Kahánek,. VLIW-DLX Simulator for educational purposes. In: Workshop On Computer Architecture Education, 2007, San Diego, California. Proceedings… p. 8-13, New York, NY: ACM, 2007.

[19] R. Cancian,. Simularq: simulador de arquitetura de computador CISC da família Motorola 68000. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Universidade Federal de Santa Catarina, Florianópolis, 1997.

[20] N. Gonçalves Junior et al. R10k: Um simulador de arquitetura superescalar. In: Workshop De Educação Em Arquitetura De Computadores, 2007, Gramado. Anais... SBC, 2007.

[21] R. Moreira; G. Hajdu, Ambiente de Apoio ao Aprendizado de Arquitetura de Computadores. In: Workcomp Sul, Florianópolis. Anais..., 2004.

[22] A. Raabe; J. Silva; L. Giraffa. Um ambiente EAD para Promover Experiências de Aprendizagem Mediadas em uma Disciplina Presencial. Revista Informática na Educação, Porto Alegre - RS, v. 8, n. 1, p. 89-101, 2005.

[23] I. Garcia; P. Rezende; F. Calheiro. Astral: Um Ambiente para Ensino de Estruturas de Dados através de Animações de Algoritmos. Revista Brasileira de Informática na Educação, v.1, n.1, 1997.

[24] A. Mariani. O Mundo dos Atores: uma perspectiva de introdução à programação orientada a objetos. Revista Brasileira de Informática na Educação, v.5, n.5, 1999.

[25] A. Gomes. Referencial Teórico Construtivista para Avaliação de Software Educativo, Revista Brasileira de Informática na Educação, v.16, n.2, 2008.