Estendendo a Arquitetura dos Processadores BIP para Ampliar o Seu Potencial de Uso em Disciplinas de Introdução a Programação

Paulo V. Vieira, Paulo R. M. Rech, Roney C. Mensch, Cesar A. Zeferino, André L. A. Raabe

Mestrado em Computação Aplicada
Universidade do Vale do Itajaí - Univali
Rua Uruguai, 458, C.P. 360, 88302-202, Itajai, SC, BRASIL
{pauloviniccius, paulorech, menschroney, zeferino, raabe}@univali.br

Resumo

Para reduzir a abstração de conceitos de lógica de programação nos primeiros semestres de cursos da área de computação, a família de processadores BIP e o ambiente de desenvolvimento integrado Bipide foram desenvolvidos. As versões iniciais dos processadores da família BIP, suportados pela ferramenta Bipide, apresentavam algumas limitações, não suportando chamadas de procedimentos e a interação com o usuário por meio de operações de entrada e saída, o que impedia sua adoção na solução de problemas mais complexos ou que necessitassem de alguma interação com o aluno. Neste contexto, este trabalho apresenta o desenvolvimento do processador BIP IV, que estende as características dos processadores BIP, agregando novas funcionalidades e aumentando a abrangência de sua utilização. Também foram implementadas modificações na IDE Bipide a fim de suportar o uso do processador BIP IV. Com isso, aumenta-se a interação do aluno com a ferramenta e seu potencial de utilização na resolução de problemas mais complexos.

1. Introdução

Com o objetivo de auxiliar o processo de ensino-aprendizagem nos semestres iniciais de cursos de Ciência da Computação, foi proposta uma abordagem interdisciplinar abordando conceitos de arquitetura e organização de computadores nas disciplinas iniciais ligadas à aprendizagem de programação [1]. Esse enfoque parte de um problema frequente relatado por pesquisadores dessa área [2, 3, 4, 5, 6] que apontam o alto nível de abstração envolvido nos conceitos de programação como um dos aspectos que dificultam a aprendizagem inicial de programação.

Alguns autores acreditam que a redução da abstração envolvida nesses conceitos pode diminuir os problemas de aprendizagem apresentados pelos alunos iniciantes em programação [7, 8]. Nesse contexto, identifica-se uma distância entre os momentos em que os alunos aprendem a lógica de programação (tipicamente, no primeiro semestre do curso) e os conceitos fundamentais de arquitetura de computadores (em geral, no segundo ano do curso).

Buscando reduzir essa distância, foi desenvolvida uma família de processadores de baixa complexidade denominada BIP (Basic Instruction-set Processor) e uma ferramenta educacional chamada Bipide [9]. A família BIP foi concebida com o objetivo de auxiliar o aprendizado de conceitos de arquitetura e organização de computadores por alunos iniciantes na área de Computação. Essa série de processadores foi desenvolvida tendo com base duas diretrizes fundamentais: (*i*) facilitar o aprendizado pelos alunos; e (*ii*) promover o reuso do processador em diferentes disciplinas do curso (interdisciplinaridade).

A família BIP foi especificada em níveis incrementais de complexidade de arquitetura e organização, sendo que o processador BIP I oferece apenas instruções de aritmética e de acesso à memória [10]. O BIP II inclui instruções de desvio condicional e incondicional [11], enquanto que o BIP III acrescenta suporte para operações de lógica bit-a-bit. O µBIP, por sua vez, foi desenvolvido com o intuito de ser utilizado no ensino de Sistemas Embarcados, agregando periféricos e funcionalidades típicas de microcontroladores aos processadores BIP [12].

A ferramenta Bipide consiste em um ambiente de desenvolvimento integrado (*IDE - Integrated Development Environment)* baseado na arquitetura dos processadores BIP I e BIP II, que oferece

funcionalidades que auxiliam na compreensão e utilização desses processadores no ensino [9]. Este ambiente possibilita a codificação de programas em Portugol (pseudolinguagem utilizada para facilitar o ensino de algoritmos), a tradução para a linguagem de montagem do processador e o acompanhamento da execução desses programas em um simulador da arquitetura e da organização dos processadores da família BIP.

Essa prática didática foi utilizada ao longo de quatro semestres letivos e apresentou resultados positivos conforme relatado em [9]. No entanto, os processadores BIP I, II e III não suportam chamadas de procedimentos ou a interação com o usuário através de operações de entrada-e-saída, o que restringia sua utilização a algoritmos muito simples. Esse aspecto gerava limitações quanto ao uso da família BIP para o ensino de conceitos mais abrangentes da aprendizagem de computação. Analisando a bibliografia de apoio e os algoritmos normalmente utilizados nas disciplinas da área de algoritmos, percebe-se que muitos autores [13, 14] adotam problemas que necessitam interações como entrada-e-saída de dados. Baseado nisso, conclui-se que a interação, via entrada-e-saída de dados, é um componente importante para auxiliar a despertar mais interesse por parte do aluno.

O trabalho apresentado em [15] descreve a concepção de uma nova versão de processador, denominado BIP IV, o qual adiciona à família BIP instruções de entrada-e-saída (E/S), manipulação de vetores e chamadas de procedimentos. Os autores expõem ainda as alterações implementadas no ambiente Bipide para suportar esse novo processador.

Nesse contexto, este artigo estende a pesquisa apresentada em [15] e relata uma pesquisa que busca evidenciar as limitações encontradas com a inserção do Bipide e da família BIP em uma disciplina inicial de programação. Além disso, descreve-se com maiores detalhes a arquitetura e a organização do processador BIP IV e as novas funcionalidades acrescidas ao ambiente Bipide. Entre os benefícios da implementação em questão estão: (*i*) maior interação dos alunos com a ferramenta; (*ii*) aprendizagem de novas funcionalidades e do seu funcionamento no processador; (*iii*) extensão do uso da ferramenta para disciplinas de semestres posteriores aos já aplicados, uma vez que permite algoritmos mais complexos; e (*iv*) a continuidade da pesquisa relacionada à uma prática didática interdisciplinar na área de computação.

2. Família de processadores BIP

Uma prática essencial para o ensino de computação é a experimentação realizada em laboratórios, onde os estudantes podem aplicar o conhecimento teórico obtido em sala de aula na solução de problemas práticos relacionados às disciplinas estudadas. Nesse aspecto, a escolha de modelos de processadores para o ensino de conceitos de arquitetura e organização de computadores é alvo de estudos frequentes pelos educadores da área como, por exemplo, no trabalho apresentado por Clements [16] que discute aspectos que devem ser considerados na escolha de modelos de processadores a serem aplicados no ensino. Dessa forma, alguns autores e professores optam por utilizar modelos hipotéticos de processadores, outros adotam processadores reais e comerciais como referência para estudos de caso [16, 17].

Para as fases iniciais de um curso de graduação, a seleção de processadores para o ensino concorrente da lógica de programação e de conceitos de arquitetura de computadores deve facilitar as relações entre as abstrações lógicas vivenciadas pelos alunos que estão começando a programar e as representações em hardware correspondentes. Podem ser citadas, como exemplo, algumas relações 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 operadores imediatos;
- Atribuição de variáveis e sua correspondência com as operações de acesso à memória; e
- Operações aritméticas e sua execução no hardware.

Percebe-se, no entanto, que os modelos de processadores tipicamente utilizados por professores de disciplinas introdutórias costumam ser muito abstratos e não permitem estabelecer essas relações. Uma alternativa seria utilizar modelos de processadores mais detalhados, como aqueles adotados nas disciplinas específicas da área de Arquitetura de Computadores (e.g. MIPS, x86,...). 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. Uma exceção é o livro “Organização e Projeto de Computadores”, de Patterson e Hennessy [18], no qual os autores descrevem o processador MIPS com ênfase na exploração da interface entre o hardware e software, permitindo a interligação dos conceitos apresentados

com aqueles estudados nas disciplinas de programação. Essa abordagem favorece a interdisciplinaridade e a relação entre os conceitos mais abstratos da programação de alto nível com a realização física do processador. No entanto, o pouco embasamento dos alunos nas fases iniciais torna inadequado o uso de processadores com o grau de complexidade do MIPS, e alternativas mais simples podem e devem ser buscadas.

Nesse sentido, discussões continuadas entre professores das áreas de Programação e de Arquitetura de Computadores na Universidade do Vale do Itajaí (UNIVALI) levaram à concepção de uma arquitetura simplificada de processador tendo duas diretrizes principais de projeto:

1. Facilitar o aprendizado e o entendimento dos conceitos de arquitetura por alunos da primeira fase do curso de Ciência da Computação da UNIVALI; e
2. Viabilizar e facilitar o uso da arquitetura em disciplinas mais avançadas, promovendo uma integração interdisciplinar.

Para minimizar a complexidade do processador, foi adotada uma arquitetura orientada a acumulador com características derivadas da arquitetura dos microcontroladores PIC [19]. No entanto, algumas escolhas foram feitas no sentido de conferir uma regularidade arquitetural maior do que a desses microcontroladores, nos quais as palavras de dado e de instrução têm tamanhos diferentes e a largura do campo de código de operação pode variar para as diversas classes de instrução. Essas escolhas foram baseadas na assertiva apresentada por Patterson e Hennessy [18] de que quanto mais regular for a arquitetura de um processador, mais fácil será a sua implementação. Além disso, entende-se que a regularidade favorece o entendimento da arquitetura do processador pelos alunos.

Com base nas diretrizes de projeto discutidas previamente, foram definidos os seguintes atributos para a arquitetura da família BIP:

- Tamanho das palavras de instrução: 16 bits;
- Tamanho da palavra de dados: 16 bits;
- Registradores: PC (Program Counter), IR (Instruction Register), STATUS e ACC;
- Modelos de execução: Registrador-Registrador e Registrador-Memória;
- Modos de endereçamento: Imediato e Direto; e
- Formato de instruções: formato único composto por dois campos: o código de operação (de 5 bits) e o operando (de 11 bits).

Os processadores BIP apresentam uma organização monociclo do tipo Harvard com memórias separadas para dados e instruções. A Unidade Central de Processamento (CPU) é dividida em dois blocos: Controle e Caminho de Dados, sendo o Controle responsável por buscar e decodificar as instruções, gerando os sinais de comando para o Caminho de Dados, o qual é responsável por executar a operação correspondente à instrução corrente.

As diferentes versões dos processadores BIP representam níveis incrementais de complexidade de arquitetura e de organização, cada um com suporte incremental ao entendimento de conceitos de Algoritmos e Programação e oferecendo recursos adicionais para uso em outras disciplinas, em uma abordagem interdisciplinar. As seguintes versões foram disponibilizadas previamente:

- BIP I: inclui apenas instruções de aritmética 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 e de instruções, conjuntos de instruções, programação em linguagem de montagem e geração de código em linguagem de máquina [10];
- 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 [11];
- BIP III: acrescenta instruções de lógica, suportando operações de lógica bit-a-bit; e
- µBIP: inclui periféricos e funcionalidades típicas de microcontroladores, permitindo seu uso no ensino de Sistemas Embarcados e Distribuídos [12].

3. O ambiente integrado Bipide

O ambiente Bipide foi concebido pela necessidade de um simulador educacional que permitisse relacionar os conceitos de lógica de programação com aspectos definidos no hardware dos processadores BIP I e BIP II. Esse ambiente possibilita a criação de algoritmos em Portugol e sua execução passo a passo ou de forma contínua. A ferramenta permite visualizar o código correspondente ao programa em linguagem *assembly* e o estado dos componentes da organização do processador através de animações que ilustram o funcionamento interno do mesmo [9].

O Bipide está dividido em três módulos principais, que compreendem:

- Programação: módulo composto por um editor destinado ao desenvolvimento de programas em linguagem Portugol e um compilador capaz de traduzir a linguagem Portugol para a linguagem de montagem dos processadores BIP;

- Simulação: apresenta um simulador de arquitetura e de organização para os processadores BIP I e BIP II, o qual exibe graficamente o fluxo de execução dos programas por meio de animações; e
- Ajuda: módulo composto por informações sobre as funcionalidades da ferramenta e informações teóricas relacionadas à arquitetura e organização dos processadores BIP.

O subconjunto da linguagem Portugol utilizada no ambiente Bipide foi definido segundo as características arquiteturais dos processadores BIP I e BIP II. Foram disponibilizadas estruturas de desvio e de laços de repetição, além dos elementos básicos necessários para estruturar um algoritmo [9]. Para a visualização da execução dos programas implementados na ferramenta, o módulo de simulação do Bipide exibe simultaneamente o programa em linguagem Portugol, o *assembly* correspondente e a organização do processador, o que auxilia na redução das abstrações apresentadas nos conceitos de programação [9].

A Figura 1 apresenta a interface dos módulos de programação e simulação do Bipide, com seus principais recursos descritos a seguir:

1. Editor: possibilita a edição de algoritmos em Portugol;
2. Mensagens de Erros: exibe mensagens de erro ocorridas durante a compilação do programa;
3. Exportação: possibilita exportar os algoritmos nos diferentes formatos: (*i*) *assembly*; (*ii*) binário; (*iii*) VHDL (VHSIC hardware description language); e (*iv*) MIF (Memory Initialization File);
4. Simulação: disponibiliza botões que permitem controlar a simulação, incluindo a execução passo a passo ou de forma contínua;
5. Velocidade: permite controlar a velocidade da simulação do programa;
6. Portugol: exibe o programa Portugol que está sendo simulado destacando a linha em execução;
7. Assembly: exibe o código *assembly* gerado pelo compilador, onde a linha em execução é destacada assim como todo o conjunto de instruções que correspondem à linha do código em alto nível, em destaque na janela Portugol;
8. Organização do Processador: mostra a organização do processador (BIP I ou BIP II), e as animações que representam as instruções em execução;
9. Registradores: exibe os valores dos registradores do processador durante a simulação do programa; e
10. Descrição: mostra o nome e a descrição de cada instrução executada.

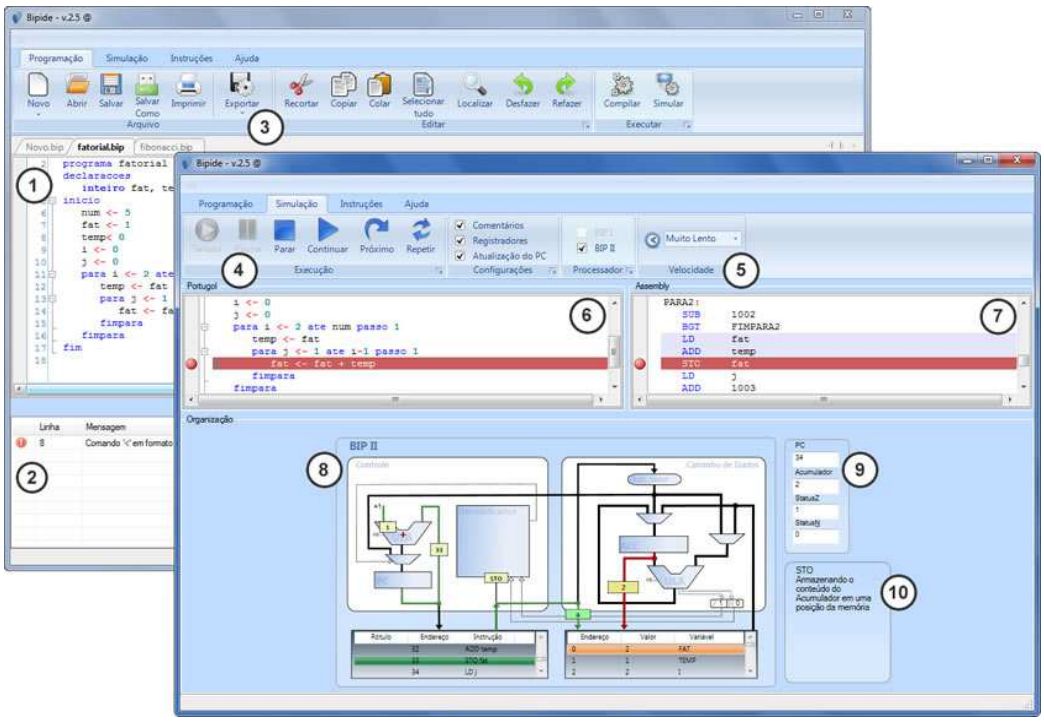


Figura 1. Interface do Bipide

4. BIP e Bipide no contexto educacional

Sabe-se que a utilização de processadores simplificados no ensino contribui para a compreensão das abstrações envolvidas nos conceitos relacionados à aprendizagem inicial de programação, o que pode facilitar a aprendizagem dos alunos que apresentam problemas em lidar com abstrações [1].

No entanto, as iniciativas de criação de processadores com objetivo didático normalmente buscam apoiar apenas as disciplinas diretamente envolvidas, como "Circuitos Digitais" e "Arquitetura e Organização de Computadores". A criação da família de processadores BIP, com suas diferentes versões, buscou transcender esse enfoque, induzindo um leque de possibilidades de exploração dos conceitos envolvidos em diferentes disciplinas da Ciência da Computação em uma abordagem interdisciplinar.

As três versões iniciais do processador BIP permitem trabalhar níveis de complexidade diferentes e com isso envolver disciplinas do primeiro, segundo, terceiro e sétimo semestres do curso de Ciência da Computação da UNIVALI, além de envolver trabalhos de conclusão de curso.

Com base nessa proposta já foram realizados diversos trabalhos, disponibilizando novos recursos e ferramentas, destacando-se: (*i*) desenvolvimento da IDE Bipide para programação, depuração e simulação da arquitetura e da organização dos processadores BIP [9]; (*ii*) avaliação do seu uso e do impacto no aprendizado dos alunos da disciplina de Algoritmos e Programação [9]; e (*iii*) o desenvolvimento de um microcontrolador sintetizado em lógica programável baseado na arquitetura BIP (o µBIP) [12].

Através de um projeto de pesquisa buscou-se avaliar a viabilidade de utilização dos processadores BIP I e BIP II em uma disciplina inicial de Algoritmos e Programação. Para isso, foi selecionado um conjunto de problemas comumente utilizados em disciplinas introdutórias de programação, incluindo algoritmos seqüenciais e algoritmos com desvios condicionais e laços de repetição.

Foram descartados os algoritmos que apresentavam saída em forma de texto ou exigiam recursos indisponíveis nos processadores BIP, o que resultou em um conjunto de 30 algoritmos (cerca de 80% dos problemas propostos inicialmente). Os algoritmos que utilizavam entrada-e-saída de dados numéricos foram adaptados com a substituição da entrada do usuário pela atribuição do valor a uma constante e a saída atribuída a uma variável do programa.

Em seguida, os algoritmos foram implementados na ferramenta Bipide e executados pelo processador. Como resultado da implementação, obteve-se um levantamento das instruções *assembly* utilizadas por cada algoritmo, apresentado na Tabela 1. Estes dados podem ser utilizados para futuras análises como, por exemplo, cálculos de desempenho e otimizações na geração de código *assembly*.

A análise realizada demonstra a capacidade dos processadores BIP em executar desde algoritmos simples até algoritmos mais complexos. Com isso, percebe-se que a família de processadores BIP pode ser adotada como modelo em disciplinas da primeira fase de um curso de computação.

Tabela 1. Instruções utilizadas pelos algoritmos implementados

Instruções	Algoritmos																															Uso
	Sequenciais										Desvio Condicional										Laços de Repetição											
HLT	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	X
STO	24	1	2		4	25	37	35	6	11	13	26	5	4	5					22	5	14	7	22	18	32	16	27	16	14	X
LD	21	1	2	18	4	16	25	29	6	10	10	20	3	1	1					15	3	9	4	18	5	12	7	18	6	5	X
LDI	2			5		9	10	6		2	5	9	3	4	5					5	3	7			5		3	3	2	7	X
ADD				2		4	2	1		1	2	9								2	1	3	2	3	2		3	9		4	X
ADDI				1		1	4					1												3				4		1	X
SUB	1			3		3		5		2	2	5	1	1	1					3	1	2	1		2		1	1		2	X
SUBI	2																			2										2	X
BEQ															2																X
BNE												1		1	1	1	2	3	4	1			2		3					2	X
BGT	2			2		2	2	2		1	1	2								2	1	1	1	2	1	3	2		1	2	X
BGE											1																	2			X
BLT																															-
BLE													1																		X
JMP	2			2		2	2	2		1	3	3								2	1		1	4	3	2	1	2	1	2	X
RETURN											3	3		2						2		1		2	2	3	3	1	2	1	X
CALL											4	3		2						3		1		1	2	2	3	1	2	1	X

No entanto, essa análise evidenciou as limitações impostas pela arquitetura das versões iniciais dos processadores BIP, o que restringe a quantidade de problemas possíveis de serem exemplificados com seu uso e dificulta sua adoção em disciplinas de semestres posteriores. Dessa forma, o desenvolvimento do BIP IV, descrito nas seções seguintes deste trabalho, busca suprir essa necessidade, integrando à família BIP um novo processador com funcionalidades intermediárias aos processadores BIP III e µBIP. Com essa versão pretende-se suportar operações de E/S, manipulação de vetores e o uso de sub-rotinas e, com isto, superar as limitações apresentadas pelas versões disponibilizadas anteriormente e ampliar as possibilidades de uso da família BIP em um contexto educacional.

5. Arquitetura e organização do BIP IV

A especificação do BIP IV estende a arquitetura do BIP III e aproveita algumas instruções presentes no µBIP. Além de possibilitar operações de entrada-e-saída e chamada de procedimentos, foram incluídas nessa versão instruções de deslocamento e de manipulação de vetores.

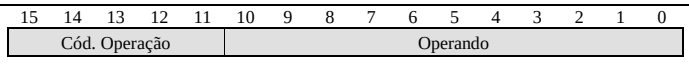
Para suportar as operações de entrada-e-saída, foi adotado o método de E/S mapeada em memória, utilizado no µBIP, onde os registradores IN_PORT (entrada) e OUT_PORT (saída) são utilizados para acesso a interfaces de E/S. A Tabela 2 resume a arquitetura do BIP IV, identificando as classes de instrução suportadas pelos diferentes processadores da família.

O código de operação dos processadores BIP é composto por 5 bits, permitindo identificar até 32 instruções. Deste total, 28 códigos são utilizados pelas instruções do BIP IV. Os quatro códigos restantes são reservados para futuras instruções, sendo que um deles é utilizado no µBIP para representar a instrução de retorno de interrupção [12].

O conjunto de instruções do BIP IV é uma extensão do conjunto de instruções do BIP III acrescido das instruções herdadas do µBIP, responsáveis pela manipulação de vetores e chamadas de sub-rotinas.

A organização do BIP IV foi baseada na organização do processador BIP III, à qual foram adicionados alguns componentes presentes no µBIP, necessários para suportar manipulação de vetores e chamada de sub-rotinas. Esses componentes compreendem uma unidade de manipulação de vetores e uma pilha de suporte a procedimentos, onde é salvo o endereço de memória da instrução seguinte à chamada da sub-rotina.

Tabela 2. Resumo da Arquitetura do BIP IV

Tamanho da palavra de dados	16 bits	
Tipos de dados	Inteiro de 16 bits com sinal (–32768 a +32767)	
Tamanho da palavra de instrução	16 bits	
Formato de instrução		
Modos de endereçamento	Direto: o Operando é um endereço da memória Imediato: o Operando é uma constante Indireto: o Operando é um endereço base de um vetor que é somado ao INDR para o cálculo de um endereço efetivo da memória de dados	
Registradores	ACC: acumulador STATUS: registrador de Status SP: apontador do topo da pilha PC: contador de programa INDR: registrador de índice	
Classes de instução	BIP I – BIP IV	Controle: HLT Armazenamento: STO Carga: LD e LDI Aritmética: ADD, ADDI, SUB e SUBI
	BIP II – BIP IV	Desvio: BEQ, BNE, BGT, BGE, BLT, BLE e JMP
	BIP III – BIP IV	Lógica booleana: AND, OR, XOR, ANDI, ORI, XORI e NOT
	BIP IV	Deslocamento Lógico : SLL e SRL Manipulação de vetor: LDV e STOV Suporte a procedimentos : RETURN e CALL

Foram incluídos ainda dois pinos para as operações de E/S, representados pelos registradores IN_PORT e OUT_PORT. A Figura 2 ilustra a organização do BIP IV, na qual é possível observar os componentes adicionados ao processador, como a pilha (representada como Stack) e a unidade de manipulação de vetores (representada por Vector Access), além dos controles adicionados ao decodificador de instruções.

6. Suporte ao BIP IV no Bipide

Para implementar o suporte ao BIP IV na ferramenta Bipide, foram levantadas características relacionadas às interfaces de E/S de simuladores de arquitetura similares. Essas características permitiram definir aspectos que orientaram a construção da interface de E/S utilizada no Bipide. A seguir descreve-se a análise realizada e a implementação do suporte ao BIP IV na ferramenta Bipide.

6.1. Análise de trabalhos similares

A análise de trabalhos similares considerou simuladores de arquitetura de processadores com interfaces de E/S. A Tabela 3 apresenta uma comparação entre esses sistemas e a proposta implementada no Bipide. Observa-se que os sistemas Bipide e MipsIt apresentam as mesmas características, sendo os únicos a permitir programação em linguagem de alto nível. No entanto, a arquitetura do processador MIPS, ilustrada no MipsIt, possui complexidade acima do desejável para a apresentação de conceitos iniciais de programação. Já a arquitetura do BIP, tendo sido criada para propósito didático, possui a simplicidade necessária para ilustração desses conceitos.

Percebe-se também que o ambiente Bipide possibilita a simulação da organização dos processadores BIP, permitindo que o aluno visualize graficamente, por meio de animações, o funcionamento dos componentes do processador. Considera-se essa característica um grande diferencial da ferramenta, contribuindo para o aprendizado.

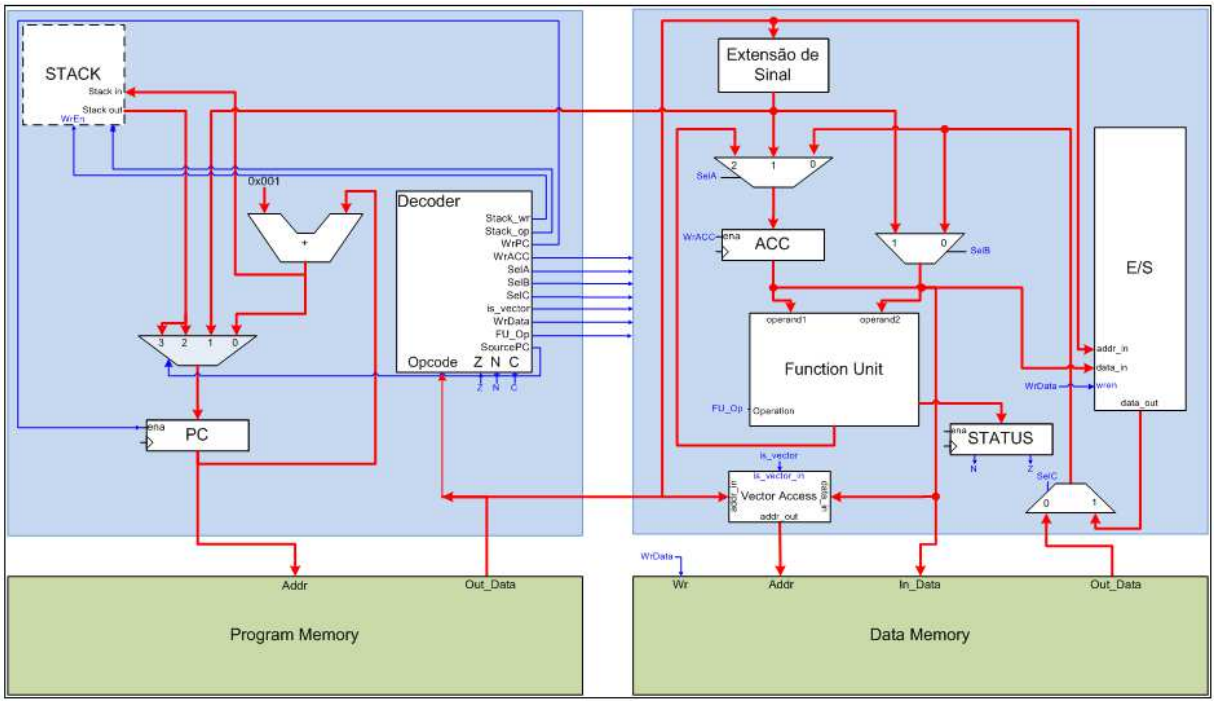


Figura 2. Organização do BIP IV

Tabela 3. Características dos Sistemas Analisados

Simulador	Processador Alvo	Simula a Organização	Suporta linguagem de alto nível	Interface de E/S
NeanderWin [20]	Neander-X	-	-	Chaves e Visor
GNUSim8085 [21]	Intel 8085	-	-	Campo Editável
MipsIt [22]	MIPS32	SIM	SIM	Chaves e LEDs
ABACUS [23]	Intel 8085	-	-	Chaves e LEDs
WinMIPS64 [24]	MIPS64	-	-	Terminal
Bipide	Família BIP	SIM	SIM	Chaves e LEDs / Campo Editável

Entre os simuladores analisados, alguns apresentam interfaces de E/S baseadas em controles simples, com representações de chaves e LEDs, outros incluem visores ilustrando valores em decimal ou em hexadecimal. Tais interfaces atendem problemas simples como, por exemplo, algoritmos onde a entrada e a saída são números inteiros; ou ainda, problemas mais complexos, como a simulação de algum dispositivo acoplado ao processador e que necessite ler ou escrever em determinados bits da porta. Devido às essas possibilidades, adotou-se no Bipide uma interface de entrada-e-saída baseada em chaves e LEDs. Para facilitar a compreensão por alunos que ainda não estão familiarizados com a notação binária, foi adicionado um campo de edição e leitura em decimal.

6.2. Implementação

Para contemplar o suporte ao BIP IV, o ambiente Bipide foi alterado de modo a suportar as seguintes características dos processadores BIP III e BIP IV: (*i*) operações de E/S; (*ii*) utilização de vetores; (*iii*) chamada de sub-rotinas com passagem de parâmetros; e (*iv*) operações de lógica. Para isso, as alterações feitas no Bipide incluem mudanças na gramática Portugol e novas verificações nas ações semânticas, assim como alterações na geração de código *assembly* e mudanças no simulador.

O compilador do Bipide foi construído através da IDE ANTLRWorks, um ambiente de desenvolvimento de gramáticas para o ANTLR3 (ANother Tool for Language Recognition V.3) [25], na qual foram implementados os analisadores léxico e sintático e definidas as ações semânticas para geração do código em linguagem de montagem e tratamento de erros.

As classes que compõem o Bipide foram desenvolvidas na linguagem C# sobre a IDE Visual Studio 2010 [26]. Para o desenvolvimento das interfaces gráficas e animações do simulador, foram utilizadas as tecnologias WPF (Windows Presentation Foundation) [27] e a ferramenta Expression Blend [28]. Por possibilitarem a criação de aplicações com interfaces ricas, tais tecnologias agregaram ao Bipide recursos de animação e funcionalidades mais intuitivas, oferecendo uma maior interatividade com o usuário. A Figura 3 ilustra o fluxo básico em que o Bipide foi desenvolvido.

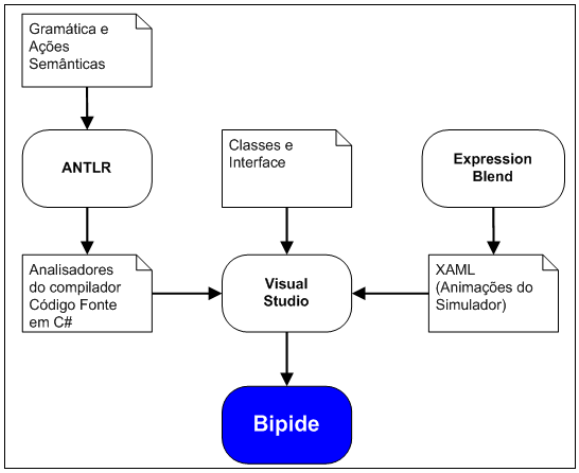


Figura 3. Fluxo de implementação do Bipide

Em sua versão anterior, o simulador do Bipide permitia escolher entre os processadores BIP I e BIP II, refletindo em mudanças nas instruções disponíveis, ilustrações e animações do processador. Para manter esse conceito, o módulo de simulação da nova versão do Bipide foi modificado incluindo as alterações necessárias para representar, além do BIP I e BIP II, os processadores BIP III e BIP IV.

Entre as alterações feitas no módulo de simulação do Bipide, destaca-se ainda a adição de uma interface de E/S, em realce na Figura 4. Foram utilizados LEDs como periférico de saída e chaves binárias como periférico de entrada. Estes periféricos representam os valores de posições mapeadas em memória para esta finalidade. A interface de entrada-e-saída inclui ainda a opção de leitura e escrita em notação decimal, feita através de campos editáveis.

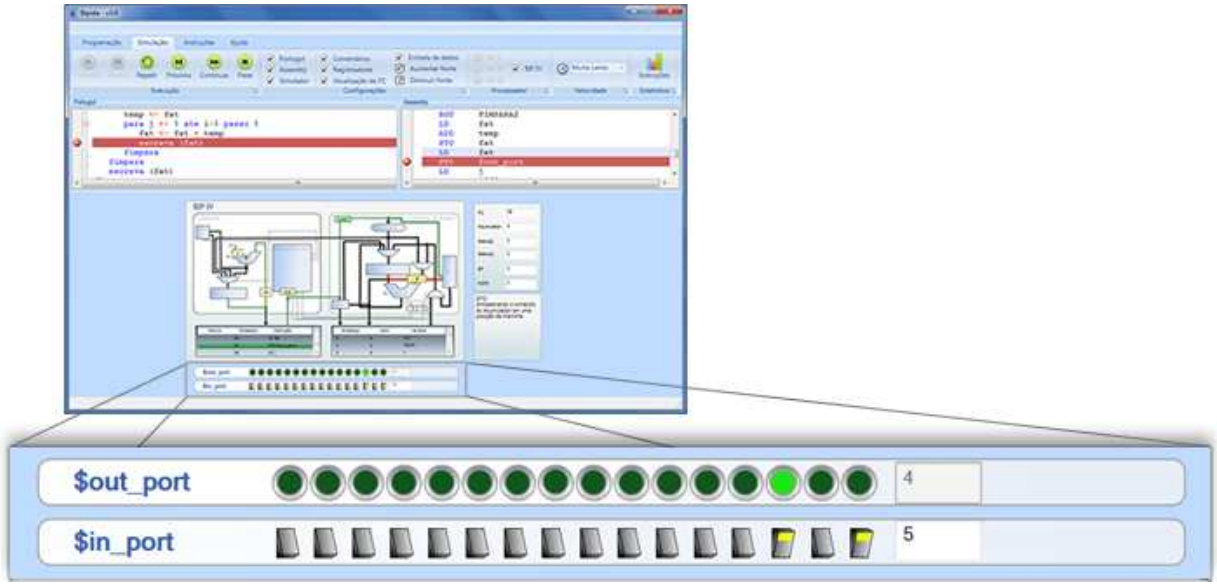


Figura 4. Interface de entrada e saída adotada no Bipide

Essa solução, baseada em LEDs e chaves, foi seguida por alguns dos simuladores apresentados na Seção 6.1, e foi adotada no Bipide por refletir um nível de abstração médio na qual fica transparente ao utilizador o real funcionamento do hardware. Além disso, a interface adotada permite resolver problemas onde a entrada e a saída são números inteiros, ou ainda simular o controle de dispositivos periféricos que realizam leitura e/ou escrita em determinados bits da porta. Dessa forma, imagina-se um conjunto de problemas que podem ser ilustrados por este ambiente, os quais vão desde o desenvolvimento de um algoritmo que realiza a soma de dois números inteiros até o controle da rotação de um motor de passo (onde a ativação de determinados bits em uma sequência determinada de passos são necessários para produzir o movimento de rotação do dispositivo).

As alterações realizadas no ambiente Bipide são recentes, dessa forma, a avaliação dos benefícios obtidos com a nova versão da ferramenta deverá ser feita ao longo dos próximos semestres, com a realização de experimentos com alunos de disciplinas do curso de Ciência da Computação da UNIVALI.

7. Conclusões

Este artigo apresentou o desenvolvimento do processador BIP IV e as alterações realizadas na IDE Bipide para suportar esse processador. Com isso, pretende-se ampliar as possibilidades de uso da família BIP e do Bipide em um contexto educacional interdisciplinar, colaborando para a redução do nível de abstração envolvido em conceitos de programação e permitindo o desenvolvimento de algoritmos mais complexos que os suportados anteriormente.

A análise de algoritmos apresentada na Seção 4 evidencia as limitações das versões anteriores dos processadores BIP e demonstra que os recursos incluídos no BIP IV ampliam a quantidade de problemas possíveis de serem exemplificados com sua utilização. Isso favorece a adoção da família BIP e do ambiente Bipide como ferramentas de apoio em diferentes disciplinas dos cursos da área de computação, que até então encontravam dificuldades na utilização da ferramenta em virtude das restrições impostas pela arquitetura dos processadores BIP disponíveis anteriormente.

Com as novas funcionalidades e possibilidades de exploração disponibilizadas, espera-se obter novas evidências sobre a redução dos problemas de aprendizagem associados à abstração dos conceitos abordados. Pretende-se ainda promover o uso dos processadores BIP e da ferramenta Bipide em outras disciplinas do curso, como Compiladores e Circuitos Digitais.

Outros trabalhos futuros incluem o suporte a diferentes linguagens de alto nível, como C e Java; uma nova análise com os algoritmos suportados pela nova versão do processador e a realização de experimentos práticos que permitam avaliar os benefícios obtidos com as novas funcionalidades disponibilizadas.

A nova versão do Bipide pode ser obtida em: http://sourceforge.net/projects/bipide/.

Referências

[1] C. A. Zeferino, A. L. A. Raabe, P. V. Vieira, M. C. Pereira, Um Enfoque Interdisciplinar no Ensino de Arquitetura de Computadores. In: Arquitetura de Computadores: educação, ensino e aprendizado. Martins, C; Navaux P.; Azevedo, R.; Kofuji, S. (Org.). No Prelo, 2012.

[2] A. Robins, J. Rountree, N. Rountree, Learning and Teaching Programming: A Review and Discussion. Computer Science Education, v. 13, n. 2, p. 137-172, 2003.

[3] B. Haberman, O. Muller, Teaching abstraction to novice pattern-based and ADT-based problems-solving processes. In: 38th Annual Frontiers in Education Conference, p. F1C-7 - F1C-12, Saratoga Springs, 2008.

[4] J. T. Khalife, Threshold for the introduction of programming: providing learners with a simple computer model. In: Proceedings of 28th International Conference on Information Technology Interfaces, p. 71-76, Cavtat, 2006.

[5] N. L. V. Calazans, F. G. Moraes, Integrating the Teaching of Computer Organization and Architecture with Digital Hardware Design Early in Undergraduate Courses. IEEE Transactions on Education, v. 44, n. 2, p. 109-119, 2001.

[6] V. G. Renumol, D. Janakiram, S. Jayaprakash, Identification of Cognitive Processes of Effective and Ineffective Students During Computer Programming. ACM Transactions on Computing Education, v. 10, n. 3, p. 1-21. New York: ACM. 2010

[7] M. McCracken, V. Almstrum, D. Diaz, M. Guzdial, D. Hagan, Y. B. Kolikant, et al., A Multi-National, Multi-Institutional Study of Assessment of Programming Skills of First-year CS Students. SIGCSE Bulletin, USA, n. 33, v. 4, p. 125-140, 2001.

[8] R. Lister, E. S. Adams, S. Fitzgerald, W. Fone, J. Hamer, M. Lindholm, et al., A multi-national study of reading and tracing skills in novice programmers. ACM SIGCSE Bulletin, USA, v. 36, n. 4, p. 119-150, dez. 2004.

[9] P. V. Vieira, A. L. A. Raabe, C. A. Zeferino, Bipide – ambiente de desenvolvimento integrado para a arquitetura dos processadores BIP. Revista Brasileira de Informática na Educação, v. 18, n. 1, 2010.

[10] D. Morandi, M. C. Pereira, A. L. A. Raabe, C. A. Zeferino, Um processador básico para o ensino de conceitos de arquitetura e organização de computadores. Hífen, Uruguaiana, v. 30, p. 73-80, 2006.

[11] D. Morandi, A. L. A. Raabe, C. A. Zeferino, Processadores para Ensino de Conceitos Básicos de Arquitetura de Computadores. In: Workshop sobre Educação em Arquitetura de Computadores (WEAC 2006), Proceedings... Porto Alegre: SBC, 2006. p. 17-24.

[12] M. C. Pereira, C. A. Zeferino, uBIP: a simplified microcontroller architecture for education in embedded systems design. In: IP Based Electronic System Conference & Exhibition - IP 08, 2008, Grenoble. Proceedings.... Grenoble: Design and Reuse, 2008. p. 193-197.

[13] J. A. N. G. Manzano, J. F. Oliveira, Algoritmos: lógica para desenvolvimento de programação de computadores. 17. ed. São Paulo, SP: Érica, 2005.

[14] N. Ziviani, Projeto de algoritmos: com implementações em Java e C++. São Paulo, SP: Thomson, 2007.

[15] P. R. M. Rech, P. V. Vieira, C. A. Zeferino, A. L. A. Raabe, BIP IV: especificação e suporte na ferramenta Bipide. In: Workshop sobre Educação em Arquitetura de Computadores (WEAC 2011), Proceedings of the 23rd International Symposium on Computer Architecture and High Performance Computing. Vitória, 2011.

[16] A. Clements, Selecting a processor for teaching computer architecture. Microprocessor and Microsystems, USA, v. 23, n. 5, p. 281-290, 1999.

[17] B. Nikolic, Z. Radivojevic, J. Djordjevic, V. Milutinovic, A Survey and Evaluation of Simulators Suitable for Teaching Courses in Computer Architecture and Organization. IEEE Transactions on Education, USA, v. 52, n. 4, p. 449-458, 2009.

[18] D. A. Patterson, J. L. Hennessy, Organização e projetos de computadores: a interface hardware/software. 3. Ed. Rio de Janeiro: Campus, 2005.

[19] Microchip.PIC16F62X Data Sheet: FLASH-Based 8-Bit CMOS Microcontroller. Arizona. 2003.

[20] J. A. S. Borges, G. P. Silva, NeanderWin – um simulador didático para uma arquitetura do tipo Acumulador. In: WEAC 2006. Proceedings... Porto Alegre: SBC, 2006.

[21] GNUSim8085. GNUSim8085, 2003.

[22] M. Brorsson, MipsIt: A Simulation and Development Environment Using Animation for Computer Architecture Education. In: Workshop on Computer Architecture Education, 2002, Anchorage, Alaska. Proceedings… New York, NY: ACM, 2002.

[23] R. M. Ziller, ABACUS, 1999.

[24] M. Scott, WinMIPS64, 2010.

[25] P. Terence, The Definitive ANTLR Reference, The Pragmatic Bookshelf, Dallas, Texas, 1997.

[26] Microsoft Corporation. Microsoft Visual Studio 2010, 2011.

[27] Microsoft Corporation. Windows Presentation, 2011.

[28] Microsoft Corporation. Expression Blend, 2011.