International Journal of Computer Architecture Education (IJCAE)

IPNoSys IDE

Ambiente de Desenvolvimento e Simulação Integrado para uma Arquitetura não Convencional

Lucas Vinicius Amaral de Oliveira
Universidade Federal do Rio de Janeiro – UFRJ
Rio de Janeiro, RJ, Brasil
loliveira@cos.ufrj.br

Sílvio Roberto Fernandes de Araújo
Universidade Federal Rural do Semi-Árido – UFERSA
Mossoró, RN, Brasil
silvio@ufersa.edu.br

***Abstract*—Neste artigo descrevemos a implementação do "IPNoSys IDE", um ambiente de desenvolvimento e simulação integrado baseado em Qt. O software é voltado para a plataforma IPNoSys. O "IPNoSys IDE" integrou ferramentas já consolidadas como o montador e o simulador SystemC dessa arquitetura. Além disso, incluiu diversas funcionalidades que auxiliam ao desenvolvedor e permite um uso didático dessa arquitetura não convencional.**

Keywords—IDE; IPNoSys; arquitetura não convencional

I. INTRODUÇÃO

Em qualquer disciplina de ciências, a prática e a experimentação são de fundamental importância no processo de aprendizagem. Nas disciplinas de arquitetura de computadores e software básico o processo de prática prescinde da existência de laboratórios equipados com *hardware* das arquiteturas estudadas. No entanto, nem sempre as universidades dispõem desse tipo de laboratório. Neste sentido, a abordagem abstrata do funcionamento da arquitetura é essencial para o bom entendimento dos alunos. Nesse cenário surge a importância dos simuladores.

Os simuladores representam uma poderosa ferramenta para o ensino de arquitetura de computadores [1]. Os estudantes aprendem detalhes das operações computacionais; tem maior facilidade de acesso aos simuladores em detrimento do *hardware* real, tem acesso ao *software* em qualquer lugar e qualquer momento com custo muito inferior; tem a possibilidade de melhor visualizar os processos que ocorrem internamente na arquitetura. Outras razões que suportam o uso de simuladores para o ensino de arquitetura de computadores são descritas em [1]. Os simuladores são de suma importância também no ensino de arquiteturas teóricas em fase de pesquisa, já que a manufatura do *hardware* se mostra muito mais custosa. E quando se fala de arquiteturas não convencionais, as quais podem apresentar características e paradigmas próprios, o uso de simulador para o aprendizado e a utilização da arquitetura pode ser indispensável.

Na literatura, existem muitos trabalhos relacionados ao uso de simuladores no auxilio ao ensino e aprendizagem de arquitetura e *software* básico [2, 3, 4]. Neste trabalho, descrevemos a implementação do "IPNoSys IDE", um ambiente de desenvolvimento e simulação que integra as ferramentas já consolidadas da arquitetura não convencional IPNoSys. O "IPNoSys IDE" tem como objetivo auxiliar o ensino dessa arquitetura, mas também servir como ferramenta de pesquisa relacionada à arquitetura IPNoSys e auxiliar no entendimento da execução das aplicações, uma vez que o processo de depuração pelas ferramentas existentes não é uma tarefa trivial.

O restante do artigo está organizado em três seções. Na seção II mostramos alguns trabalhos relacionados à implementação de ambientes de desenvolvimento integrado e a arquitetura IPNoSys. Na seção III descrevemos o "IPNoSys IDE", na seção IV mostramos aplicações desse IDE (*Integrated Development Environment*), por fim na seção V conclusões e trabalhos futuros.

II. TRABALHOS RELACIONADOS

A. Ambientes de Desenvolvimento e Simulação Didáticos

Na literatura podem ser encontrados vários trabalhos que trazem como proposta didática a implementação de um IDE incluindo um simulador da arquitetura. Analisamos três trabalhos: Bipide, MARS e Simulador SIC/XE.

O Bipide é um ambiente de desenvolvimento integrado (IDE) baseado na arquitetura dos processadores BIP (*Basic Instruction-set Processor*) [2]. A ferramenta integra um editor de código e um compilador desenvolvido para reconhecer algoritmos simples em Portugol (pseudolinguagem utilizada no ensino de lógica de programação). O Bipide é uma ferramenta voltada para a aprendizagem de programação e arquitetura. O processador BIP está na versão IV, que permite a entrada e saída de uma palavra de dados em binário. A entrada deve ser configurada na interface através de chaveamentos. O ambiente dispõe, ainda, de um simulador que exibe a execução passo-a-passo do programa com animação do processo interno da arquitetura. O simulador possibilita sete configurações de velocidade que vão do "Muito lento" ao "Ultrarrápido".

O MARS [3] é mais conhecido por suportar a arquitetura MIPS, provavelmente a mais utilizada no ensino de organização e arquitetura de computadores. A ferramenta possui um montador e um simulador da arquitetura. Esta arquitetura é muito utilizada na indústria e em vários cursos de graduação pelo mundo [3]. O software foi desenvolvido com o

objetivo de auxiliar o ensino de arquitetura de computadores, *software* básico e construção de compiladores. O simulador no MARS mostra o estado dos registradores e das memórias de dados e de instruções. É possível controlar a velocidade da simulação, alterar o conteúdo dos registradores e memórias, e possibilita ainda, a entrada de texto via teclado. O IDE não mostra visualmente os componentes da arquitetura MIPS, diferentemente do Bipide.

O simulador para plataforma SIC/XE desenvolvido em [4] tem uma proposta didática para ensino de *software* básico. O simulador é capaz de ler um programa em *assembly* para SIC/XE, traduzi-lo para código objeto, realizar o *link* entre vários códigos objetos e o *load* para execução, na qual é simulada passo-a-passo. A interface mostra a memória, com o código executável ligado e carregado, o conteúdo dos registradores, a instrução em execução e os dados que estão sendo operados. É possível realizar entrada e saída através de um arquivo de entrada/saída que deve ser carregado no simulador antes do início da simulação. O simulador funciona no modo passo-a-passo e no modo contínuo com opção de velocidade de passo. A ferramenta permite, como funcionalidade extra, a geração de código objeto para a arquitetura alvo no formato mif para FPGAs.

Os três IDE's tem objetivos muito similares: o auxilio no ensino e aprendizagem de disciplinas da ciência da computação, em especial, arquitetura e *software* básico mas todos são voltados para plataformas convencionais, que se baseiam na arquitetura de von Neumann.

B. Arquitetura IPNoSys

O ambiente de desenvolvimento e simulação apresentado neste artigo tem como arquitetura alvo a IPNoSys [5]. Esta é uma arquitetura não convencional baseada nas características de uma rede-em-chip ou NoC (*Network-on-Chip*) com topologia de grelha 2D. A grelha é formada pela ligação de várias RPUs (*Routing and Processing Unit*) com MAUs (*Memory Access Unit*) nos quatro cantos da rede, como mostra Fig. 1.

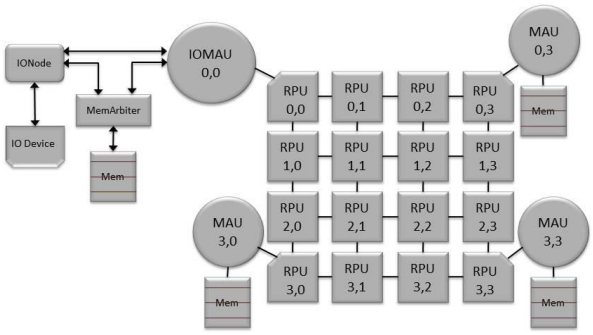

Fig. 1 - Arquitetura IPNoSys

As RPUs tem função de roteamento, como roteadores tradicionais de NoCs, e de executar instruções lógicas e aritméticas. Enquanto que as MAUs são responsáveis pela execução de instruções de acesso a memória (únicos componentes com este acesso) e de instruções de sincronização. Na Fig. 1 também se percebe um sistema de entrada/saída ligado ao IOMAU (uma MAU que gerencia o I/O, ou seja, a entrada/saída). Esse sistema funciona no modelo DMA (*Direct Memory Access*). Nessa arquitetura, os programas obedecem a um formato de pacote de NoC que lhe permite rotear e transmiti-lo por entre os componentes, e durante este processo as instruções são executadas [6]. Os pacotes são armazenados nas memórias e injetados na rede por meio das MAUs.

Com tal modelo de computação é possível explorar paralelismo em diversos níveis, seja por meio de pacotes consecutivos vindos de uma mesma MAU ou por pacotes simultâneos de MAUs diferentes. Esse é ponto forte da arquitetura, cujo desempenho pode chegar a 3,5 vezes maior que de um multiprocessador com capacidade de processamento equivalente [7].

Do ponto de vista experimental, podemos dizer que existe uma plataforma da qual é disponibilizado uma linguagem de programação simbólica (PDL – *Packet Description Language*), um montador com análise léxica, sintática e geração de código objeto, e um simulador do código objeto. Este simulador foi desenvolvido em SystemC com precisão de ciclo, o qual gera um relatório de simulação com diversos resultados e ainda possibilita a geração de arquivo de *log* com informações individuais de cada componente, quando habilitados as macros de *debug* antes da compilação do simulador.

III. IPNOSYS IDE

O "IPNoSys IDE" é um ambiente integrado de desenvolvimento e simulação para a plataforma IPNoSys. Este ambiente é uma GUI (*Graphical User Interface*), ou seja, uma camada gráfica que integra o montador e simulador originais [5], que chamaremos respectivamente de montador base e simulador base. Vale ressaltar que as ferramentas base foram desenvolvidas originalmente para uma interface de linha de comando e totalmente isoladas uma da outra. Assim, o usuário executava o montador passando um arquivo com código PDL como parâmetro e caso não houvesse erro era gerado um arquivo com o código objeto equivalente. Em seguida o usuário usava um comando no terminal para invocar o simulador passando o código objeto como parâmetro. O simulador poderia interromper a simulação reportando uma mensagem de erro ou concluir a execução escrevendo um arquivo texto com um relatório repleto de resultados da simulação. No código fonte do simulador o desenvolver/usuário poderia ainda ligar/desligar *flags* que ativam mensagens de depuração e configurar diversos parâmetros da arquitetura como quantidade de RPUs, profundidade dos *buffers* e etc. Entretanto, para estes parâmetros e *flags* terem efeito era necessário recompilar o simulador. Como o simulador foi escrito em SystemC com precisão de ciclo e cada unidade funcional (RPUs, MAUs, *buffers*, ALUs, árbitros, etc.) representa um thread de C++, quando as *flags* de depuração estão ligadas as mensagens são geradas no momento que a unidade funcional as emite e são salvas em um arquivo de *log*. Dessa forma, quando muitas unidades funcionais estavam em modo de depuração se tornava muito difícil acompanhar o fluxo de execução de todos os componentes.

Desse modo, "IPNoSys IDE" implementa uma interface de comunicação entre o montador base e o simulador base com

seu funcionamento original, e entre essas ferramentas base e o usuário. O IDE envia arquivos, parâmetros e dados de configurações para as ferramentas base, capta suas mensagens e as exibe em uma caixa de saída e recompila seu código quando necessário, tudo de forma transparente para o usuário. Além disso, acrescenta novas funcionalidades como simulação passo-a-passo controlada e exibição do código com destaque de sintaxe, como visto na Fig 2. A ferramenta é composta por dois módulos principais: módulo de edição e módulo de simulação.

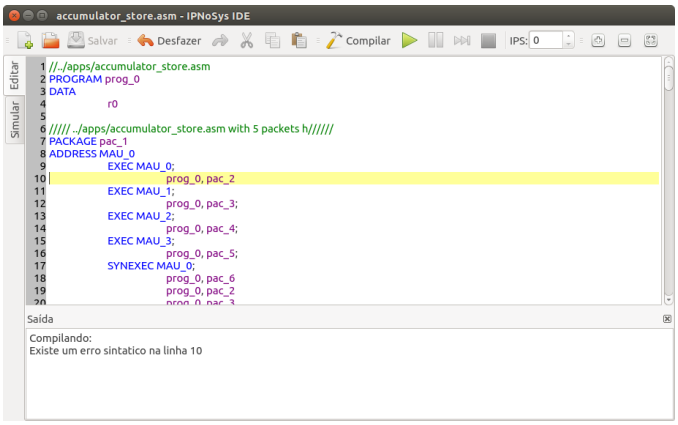

Fig. 2 - Módulo de edição do "IPNoSys IDE"

No modulo de edição, mostrado na Fig. 2, são fornecidas todas as ferramentas para criação e compilação de código PDL, assembly padrão definido para a plataforma. O editor de texto tem função de destaque de sintaxe, que facilita o entendimento visual do código. Exibe numeração de linha e destaque da linha sob o cursor de texto. Contempla, ainda, todas as operações básicas de um editor de texto simples. O módulo de edição está integrado ao montador base, capturando mensagens de sucesso ou erro de compilação e redirecionando o cursor neste último caso, como mostra a Fig. 2.

No módulo de simulação (Fig. 3), são fornecidas as ferramentas para o usuário (estudante ou pesquisador) acompanhar cada passo da simulação da execução do código na arquitetura. O painel central mostra a animação detalhada da simulação, um dos pontos principais da ferramenta. O painel lateral possibilita o acesso a todas as informações de estado de cada componente da arquitetura. Através da animação é possível notar quais RPUs estão com atividade na ULA, quais estão inativas e quais estão apenas roteando pacotes. É possível, ainda, visualizar a instrução em execução na ULA, a posição dos cabeçalhos dos pacotes injetados na rede, o uso dos *buffers* e uso dos canais de comunicação. A barra de ferramentas oferece a possibilidade de pausar, controlar a velocidade, parar a simulação e fazer zoom na animação. A velocidade da simulação é definida no controle *ips* (iterações por segundo), que quando é configurada em zero deixa o simulador no modo passo-a-passo. Nesse modo, o simulador espera que o usuário clique (ou use o atalho do teclado) para iniciar a próxima iteração. Cada iteração representa um ciclo de *clock* para o simulador base. A maioria das funcionalidades podem ser disparadas por atalhos de teclado que podem ser vistos no menu de ajuda.

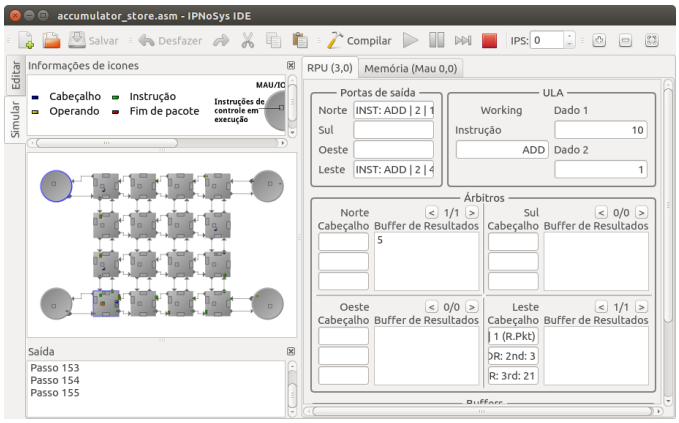

Fig. 3 - Módulo de simulação do "IPNoSys IDE"

O simulador básico descrito em [5] gera apenas um arquivo de *log* que detalha cada acontecimento na simulação de cada elemento da arquitetura do qual sua respectiva *flag* esteja ativada. O entendimento dos processos decorre da extração de informação desse arquivo, que não é intuitivo. A informação visual da simulação através da interface gráfica se mostra muito mais eficiente para o entendimento do funcionamento da arquitetura, se comparado com o método utilizado sem o auxilio do IDE. Todavia, o arquivo de *log* é exibido pelo IDE ao fim da simulação, já que contém informações não explicitadas no processo.

No IDE, ao clicar em uma RPU durante a simulação as informações presentes na mesma serão exibidas no painel lateral. Na Fig. 3, é possível visualizar o painel lateral com as caixas de informações de conteúdo dos buffers, dos pacotes presentes na RPU, das instruções em execução e dos cabeçalhos presentes. O conteúdo das memórias associadas a cada MAU pode ser visualizado na aba seguinte a das informações da RPU e clicando na MAU específica.

Também é possível, através da interface, configurar vários parâmetros da arquitetura para simulação. A janela de opções do menu ferramentas traz todos os parâmetros configuráveis do simulador e montador bases. Os parâmetros são tais como número de RPUs, tamanho dos *buffers* de entrada, tamanho dos *buffers* de resultados e quantidade de canais virtuais. Outra característica dessa GUI é sua auto-adaptação para exibir todos os componentes visuais da arquitetura quando se altera a quantidade de RPUs.

O "IPNoSys IDE" foi implementado em C++ utilizando Qt [8], um framework multiplataforma para programação em C++ padrão. O Qt dispõe de vários recursos pré-implementados, como é o caso da base para destaque de sintaxe através de expressões regulares. Possui ainda um sistema de comunicação entre objetos através de sinais e slots, que é mais seguro e robusto que as técnicas de *callback* (ponteiro para função) utilizadas em outros frameworks [9]. A ferramenta foi compilada na no sistema operacional Ubuntu 32 e 64 bits.

IV. Aplicações do IPNoSys IDE

A ferramenta representa um auxilio na programação de código PDL e na correção de erros. O destaque de sintaxe

ajuda no entendimento do código e resposta visual rápida na presença de erros sintáticos agiliza o processo de correção de erros. É possível visualizar facilmente, através da interface gráfica, as características do algoritmo de roteamento dos pacotes. A interface de simulação também ajuda a visualizar, de maneira mais intuitiva, fenômenos que ocorrem na arquitetura, tais como a injeção de pacotes em paralelo e a execução localizada. Esses fenômenos são mais difíceis de serem percebidos na análise do arquivo de *log*.

Por ter sido recém-concluída a integração desse ambiente (interface gráfica e ferramentas base), ainda não foi possível aplicá-lo na prática em sala de aula. Entretanto podemos indicar seu uso em várias disciplinas.

Pelas características da arquitetura IPNoSys, seu estudo, fazendo uso desse IDE, se aplica perfeitamente em disciplinas de Arquiteturas Não Convencionais ou Arquiteturas Avançadas, bem como em disciplinas de Processamento Paralelo, Programação Paralela, Arquitetura de Alto Desempenho e similares, fazendo uso da sua linguagem e seu modelo de computação que trabalha nativamente as ideias de barreira, distribuição e a junção da execução. Também é possível utilizar o ambiente em uma disciplina Organização e Arquitetura de Computadores para exemplificar o funcionamento de uma NoC e suas características bem como um modelo de pacote e algoritmo de roteamento, sem levar em consideração que os "roteadores" também executam instruções.

A arquitetura ainda tem um grande potencial de pesquisa investigatória. Mudanças em diversas características da arquitetura geram impactos no uso da memória, consumo de potência e principalmente em tempo de execução dos programas.

Já foi realizada uma comparação de diferentes algoritmos de roteamento na IPNoSys [10], e com o IDE é possível acompanhar visualmente o caminho de cada algoritmo, assim como contenções dos pacotes (levam a execução localizada) e balanceamento de carga. Além disso comparações de execuções de uma mesma aplicação variando as características da arquitetura (algoritmo de roteamento, quantidade de RPUs, quantidade de canais virtuais, profundidades dos *buffers*) com esse ambiente de fácil configuração tornam o processo da pesquisa mais rápido e claro.

O grupo de pesquisa dessa arquitetura tem criado vários novos modelos dessa arquitetura como: *software pipeline*, exploração de paralelismo em nível de instrução e hierarquia de memória. Cada novo modelo tem alterado apenas as ferramentas bases de modo que as experiências estão sendo realizadas no ambiente integrado de forma muito satisfatória. Podemos citar um exemplo desse uso como na criação de uma nova instrução (LOOP) que configura a quantidade de repetições que uma outra instrução deve ser executada em uma RPU. Durante a simulação é possível acompanhar uma contagem do número de vezes que a instrução está sendo executada em uma mesma RPU.

V. Conclusões e Trabalhos Futuros

Em comparação com os outros IDEs relatados nos trabalhos relacionados, o "IPNoSys IDE" se diferencia por abordar uma arquitetura não convencional e utilizada em diversas pesquisas correntes. Possibilita a visualização da arquitetura em funcionamento e se adapta a reconfigurações da arquitetura que alteram número de componentes visuais. Seu objetivo é auxiliar não apenas ao ensino, mas também as pesquisas para o desenvolvimento da arquitetura. Essa característica traz a perspectiva que a ferramenta estará em um constante processo de evolução e atualização. Como o IDE foi desenvolvido de forma modular, se forem realizadas alterações nos códigos das ferramentas base e forem mantidas as interfaces de comunicação com a GUI, os novos modelos da arquitetura também poderão se beneficiar desse ambiente integrado, como já vem acontecendo atualmente. Há possibilidade do uso dessa arquitetura, bem como suas ferramentas, em diversas disciplinas.

A ferramenta também vem se mostrando de grande valia nas pesquisas sobre a plataforma IPNoSys que permite acompanhar o comportamento dos componentes e dos pacotes em tempo de simulação. Essa facilidade já vem auxiliando estudantes e pesquisadores em suas pesquisas com a plataforma. Além disso, o IDE permite incluir facilmente novas ferramentas, como um compilador de C para PDL que está em desenvolvimento, que logo em breve poderá se integrar com o montador e a interface gráfica. Assim como um novo montador que inclui um *linker*, também em desenvolvimento.

Outros trabalhos futuros são: a inclusão de opções autocompletar no módulo editor, melhorar a reportagem de erros na compilação, incluir um módulo de depuração de modo o usuário possa incluir *breakpoints* no código PDL para indicar onde a simulação deve ser pausada, ferramentas de comparação das informações do relatório de simulação, entre outros.

References

[1] G. Wolffe, W. Yurcik, H. Osborne and M. Holliday, Teaching Computer Organization/Architecture With Limited Resources Using Simulators, ACM SIGCSE Bulletin 34, (1), 176 - 180, 2002.

[2] 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, Vol. 18, No 1, 2010.

[3] K. Vollmar, P. Sanderson, MARS: an education-oriented MIPS assembly language simulator. ACM SIGCSE Bulletin, pp. 239-243, March 2006.

[4] A. S. Neto, S.R. Fernandes "Um Simulador Didático para a Arquitetura SIC/XE" in Workshop Técnico-científico de computação, vol. I, A. F. Castro, F. E. Coelho, Y. Fernandes. Mossoró: Edufersa, 2014, p. 29-44.

[5] S. R. F. Araújo. Projeto de Sistemas Integrados de Propósito Geral Baseados em Redes em Chip – Expandindo as Funcionalidades dos Roteadores para Execução de Operações: A plataforma IPNoSys. 209 f.(Tese de Doutorado) - Universidade Federal do Rio Grande do Norte, 2012.

[6] S. R. Fernandes, et al. Processing while routing: a network-on-chip-based parallel system. IET Computers & Digital Techniques, v. 3, n. 5, p. 525 – 538, 2009. Disponível em: <http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=5200576>.

[7] S. R. Fernandes, et. al. Using NoC routers as processing elements. In: SBCCI, 2009. Natal, Brazil. ACM.

[8] Digia. Qt Project, 2014. Disponivel em: http://qt-project.org/.

[9] Digia. Signals & Slots, 2013. Disponivel em: http://qt-project.org/doc/qt-4.8/signalsandslots.html.

[10] V. Costa, S. R. Fernandes, D. F. Nunes, J. L. Silva, I. A. Carvalho, W. K. S. Dantas. Exploração de Espaço de Projeto do Roteamento na Arquitetura IPNoSys. Holos (Natal. Online), v. 4, p. 175-184, 2014.