Universidade Federal de Pernambuco
Centro de Informática

Graduação em Engenharia da Computação

Desenvolvimento de um modelo de energia para o processador NIOS II da Altera

Henrique Figueirôa Lacerda

Trabalho de Graduação

Recife
27 de Fevereiro de 2014

Universidade Federal de Pernambuco
Centro de Informática

Henrique Figueirôa Lacerda

Desenvolvimento de um modelo de energia para o processador NIOS II da Altera

Trabalho apresentado ao Programa de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Engenharia da Computação

Orientador: *Abel Guilhermino da Silva Filho*

Recife
27 de Fevereiro de 2014

AGRADECIMENTOS

Primeiramente, a meus pais Jediel e Laura que sempre estiveram presentes e sempre confiaram e acreditaram nas minhas escolhas. Foram eles que sempre me demonstraram que o futuro é a primeira coisa que você deve garantir.

Em seguida, gostaria de agradecer a minha irmã Camila, juntamente com meu cunhado Radu, que apesar de não estarem presente pessoalmente durante esta etapa, sempre acreditaram em mim e me motivaram.

Gostaria também de agradecer a minha namorada Rafaela. Esta que, apesar de não estar presente desde o início, em muito me motivou nesta reta final.

Gostaria também de agradecer a todos os meus amigos que estiveram presentes por toda essa jornada. Primeiro, àqueles que eu já conhecia antes de entrar na graduação. Apesar deles nunca acreditarem que a vida de um estudante de engenharia não é fácil e de sempre duvidarem quando eu dizia que estava na faculdade nos finais de semana, sempre acreditaram e apoiaram meus esforços.

E, por fim, mas não menos importante, a todos aqueles que conheci durante a faculdade. Tanto aos de Ciência da Computação como aos Engenharia da Computação, vários amigos de várias turmas. Depois de diversos projetos, trabalhos, provas e estudos juntos, grandes amizades foram criadas.

Todos estes presentes durante esse tempo foram vitais para a obtenção deste grau de engenheiro.

Engineers like to solve problems.
If there are no problems handily available, they will create their own problems.
-Scott Adams

RESUMO

Com o crescente mercado de sistemas embarcados, cresce também a necessidade de sistemas mais leves, com maior eficiência térmica e com uma maior duração para a bateria. Esses fatores estão diretamente ligados à redução do consumo de energia. Nesses sistemas, o componente principal é o processador embarcado. Atualmente, o processador embarcado NIOS II da *Altera* é o processador *softcore* mais utilizado na indústria de FPGAs. Para criação de um sistema com este processador, há um conjunto de parâmetros que precisam ser definidos. Sabendo-se que cada conjunto desses gera um configuração diferente para o processador (o que implica num consumo de energia diferente), simulações foram feitas utilizando as ferramentas *Altera Quartus*, *Qsys*, *ModelSim-Altera* e *PowerPlay Power Analyzer* e o consumo de energia do processador para uma simulação do sistema pôde ser obtido. Em seguida, estes valores foram comentados e avaliados através de gráficos, permitindo assim a criação de um modelo de energia genérico. A partir deste modelo, é possível determinar uma estimativa de consumo de energia para o processador num sistema, apenas definindo os parâmetros escolhidos na criação do módulo do processador NIOS II.

Palavras-chave: Altera NIOS II, Sistemas embarcados, consumo de energia, estimativa de energia, modelo de energia.

SUMÁRIO

LISTA DE FIGURAS

1 INTRODUÇÃO

Este capítulo está subdividido em quatro partes, como descritas a seguir: a primeira apresenta uma visão geral sobre os sistemas embarcados e seu consumo de energia; a segunda apresenta uma motivação para o desenvolvimento deste trabalho; a terceira descreve os objetivos a serem alcançados ao final deste trabalho, tanto gerais como específicos; e a quarta e última demonstra a estrutura deste trabalho como um todo.

1.1 Considerações iniciais

Cada vez mais, cresce o número de sistemas embarcados presentes nos utensílios do dia-a-dia. Em geral, sistemas embarcados são pequenos dispositivos, os quais possuem todos os principais elementos de um computador, capazes de executar uma ou mais tarefas.

Com o crescente mercado de dispositivos portáteis, cresce também a necessidade de sistemas mais leves, com maior eficiência térmica e com uma maior duração para a bateria. Todos estes fatores são efeitos da redução do consumo de energia neste tipo de sistema [1].

Uma das etapas que merece mais atenção na engenharia de sistemas embarcados é a etapa de especificação. Após definidos os requisitos, a estrutura do sistema, os objetivos e

todos os outros detalhes importantes, encontrar uma configuração ideal para os componentes de um sistema embarcado, torna-se uma tarefa bastante custosa.

Do mesmo modo, a definição dos parâmetros implica em alterações em todos os requisitos do sistema, incluindo o consumo de energia. Como em geral são várias as características e cada uma destas possui diversos parâmetros, escolher os parâmetros a partir de testes com cada configuração poderia levar centenas de anos [1].

1.2 Motivação

O processador embarcado é um dos principais componentes de um sistema embarcado. Trata-se de um tipo de processador limitado baseado em processadores de propósito geral que executa as mesmas tarefas. O seu consumo de energia implica em grande parte do consumo do sistema como um todo. Assim, este trabalho visa a redução do consumo de energia dos processadores embarcados.

Segundo a *Gartner Research*[1] o processador embarcado NIOS II da Altera é o processador *softcore*[2] mais utilizado na indústria de FPGA [2]. Além disso, é conhecido como o processador mais versátil do mundo, por possuir três versões distintas para se adequar às mais diversas necessidades.

---

[1] *Gartner Research* é uma empresa americana de consultoria e pesquisa em tecnologia da informação.
[2] Um processador que pode ser totalmente implementado a partir de síntese lógica.

Porém, após a escolha da versão desejada, existem ainda diversas características, cada uma com diversos parâmetros para serem escolhidos, resultando numa quantidade muito grande de possíveis configurações.

Deste modo, a escolha dos parâmetros certos no projeto deste processador embarcado é uma tarefa não-trivial que deve ser otimizada de alguma maneira visando otimização do tempo de projeto e redução do consumo de energia.

1.3 Trabalhos relacionados

Tanto em [20] como em [21] o autores propõem métodos para estimar o consumo de potência do processador NIOS II. Porém eles propõem uma estimativa de potência do processador após ele ser carregador na FPGA mas não em função dos parâmetros de projetos definidos na criação do sistema.

1.4 Objetivos

Este trabalho tem como objetivo a definição de um modelo de energia para o processador embarcado NIOS II da Altera com o intuito de tornar possível a estimativa de consumo de energia do processador apenas a partir da escolha dos parâmetros de projeto.

Objetivos Específicos

• Criar simulações para uma arquitetura usando este processador e variar os parâmetros deste, analisando a variação no consumo de energia.

• Definição de um modelo de energia dependente destes parâmetros que seja capaz de estimar um valor de consumo de energia.

1.5 Estrutura do documento

Este documento está estruturado da seguinte maneira:

1 – Introdução: descrição introdutória do tema deste trabalho seguido de uma motivação para desenvolvimento deste projeto e finalizando com esta estrutura do documento em geral.

2 – NIOS II: este capítulo inicia-se com uma descrição sobre a companhia *Altera*. Em seguida, são detalhadas as informações necessárias para o desenvolvimento e embasamento deste trabalho acerca do processador embarcado NIOS II.

3 – Ferramentas: neste capítulo, são apresentadas e detalhadas as ferramentas que serão utilizadas neste trabalho: *Altera Quartus*, *Qsys*, *PowerPlay Power Analyzer*, *ModelSim-Altera*.

4 – Cálculo da energia: este capítulo inicia-se definindo o sistema que será simulado e usado como base para o processador para estimar o seu consumo de energia. Em seguida, são detalhados todos os passos necessários para efetuar a simulação do sistema e, assim, obter o consumo de energia referente à esta simulação.

5 – Resultados obtidos: no início deste capítulo, é explicado como foram feitas as várias simulações para visualização da variação do consumo de energia. Em seguida, os valores obtidos são mostrados e comentados a partir de gráficos. Por fim, um modelo de energia baseado nos resultados obtidos é definido e detalhado.

6 – Conclusão: Neste último capítulo os resultados obtidos são comentados e comparados aos objetivos iniciais.

2 NIOS II

Neste capítulo inicialmente fala-se sobre a *Altera Corporation*, produtora do processador foco deste projeto. Em seguida este processador é descrito em detalhes, o NIOS II.

2.1 Altera

A *Altera Corporation* é uma empresa pioneira no mercado de soluções lógicas programáveis. Estas, basicamente, constituem-se de dispositivos CPLDs e dispositivos FPGAs [3][4]. Os primeiros são dispositivos lógicos programáveis designados para aplicações mais simples. Possuem uma arquitetura menos complexa, com poucos –e maiores – blocos lógicos. No entanto, por esta ser mais simples, provêm operações mais rápidas de entrada e saída [6].

Do mesmo modo, as FPGAs são dispositivos lógicos programáveis que também podem ser programados após serem fabricados. Porém, possuem uma arquitetura mais complexa com diversos minúsculos blocos lógicos. Podem ser usados para implementar

qualquer função lógica que um ASIC[3] possa realizar, além de possuir a vantagem de permitir a atualização das suas funcionalidades após o produto ser entregue ao cliente [5].

A *Altera* provê uma gama variada de FPGAs que supre diversos mercados e aplicações. As 3 principais séries são *Cyclone*, *Arria* e *Stratix*. A série *Stratix* é a série que possui as maiores e mais velozes FPGAs, direcionadas para aplicações de alta qualidade. Por outro lado, as FPGAs da série *Cyclone*, são dispositivos de baixo custo, que consomem menos energia. São indicadas para aplicações em larga escala e com custo limitado. Entre estas está a série *Arria*. Esta compõe os dispositivos de melhor custo benefício para aplicações que buscam um meio-termo entre desempenho e consumo de energia [5].

O processador NIOS II foi desenhado especificamente para a linha de FPGAs da Altera. Deste modo, as ferramentas da Altera devem ser usadas para criação de sistemas usando este processador e para, em seguida, carregar estes sistemas nas placas FPGAs.

2.2 NIOS

O NIOS II é uma evolução do primeiro processador embarcado configurável NIOS da *Altera*. Este, lançado em 2001, é um processador de 16 bits e foi o primeiro processador comercialmente viável criado especificamente para ser utilizados em sistemas embarcados embutidos em FPGAs [7].

O NIOS II, lançado em 2004, é um processador com arquitetura *RISC*[4] de 32 bits que pode ser completamente programável utilizando-se a síntese lógica das *FPGAs* da *Altera* [8].

---

[3] ASIC - Circuito integrado construído para executar uma tarefa específica
[4] RISC – Computador com um conjunto de instrução reduzido. Arquitetura que provê um conjunto de instruções simples e reduzido.

Por possuir esta característica de ser totalmente configurável, permite aos desenvolvedores do sistema total liberdade de escolha e personalização deste processador, adequando-se assim melhor aos requisitos do sistema.

Na prática, a maioria das FPGAs implementam uma lógica extra além da lógica do processador. Porém, as FPGAs da *Altera* permitem que elementos possam ser adicionados com o intuito de melhorar a experiência no desenvolvimento do processador NIOS II [9], assim como permitem que alguns componentes e lógicas desnecessárias ou que não estão sendo utilizadas possam ser removidas, permitindo-se assim reduzir custos e consumo de potência do sistema, por exemplo.

Basicamente uma implementação de um NIOS II consiste na tomada de decisões durante a etapa de projeto e na especificação dos parâmetros que formarão o núcleo do NIOS II desejado. Decisões essas que podem ser bastante importantes para o sistema, como por exemplo se deve-se aumentar ou diminuir o tamanho da memória cache, gerando um sistema menor e menos custoso ou gerando um sistema mais rápido. Do mesmo modo, decidir se deve-se incluir ou não algum componente no projeto do processador, como por exemplo um módulo de divisão. Este pode tornar mais custoso e mais complexa a implementação, porém pode diminuir bastante o tempo de execução de aplicações que efetuem muitas divisões. Removendo-se este módulo implica na simplificação do sistema, mas implica no aumento do tempo de execução de aplicações que envolvam aritmética complexa [9].

A arquitetura básica deste processador é demonstrada na Figura 2.1. Os módulos em azul são os módulos obrigatórios numa implementação. Já os itens em cinza, são módulos opcionais que podem ser adicionados, a depender da versão do núcleo.

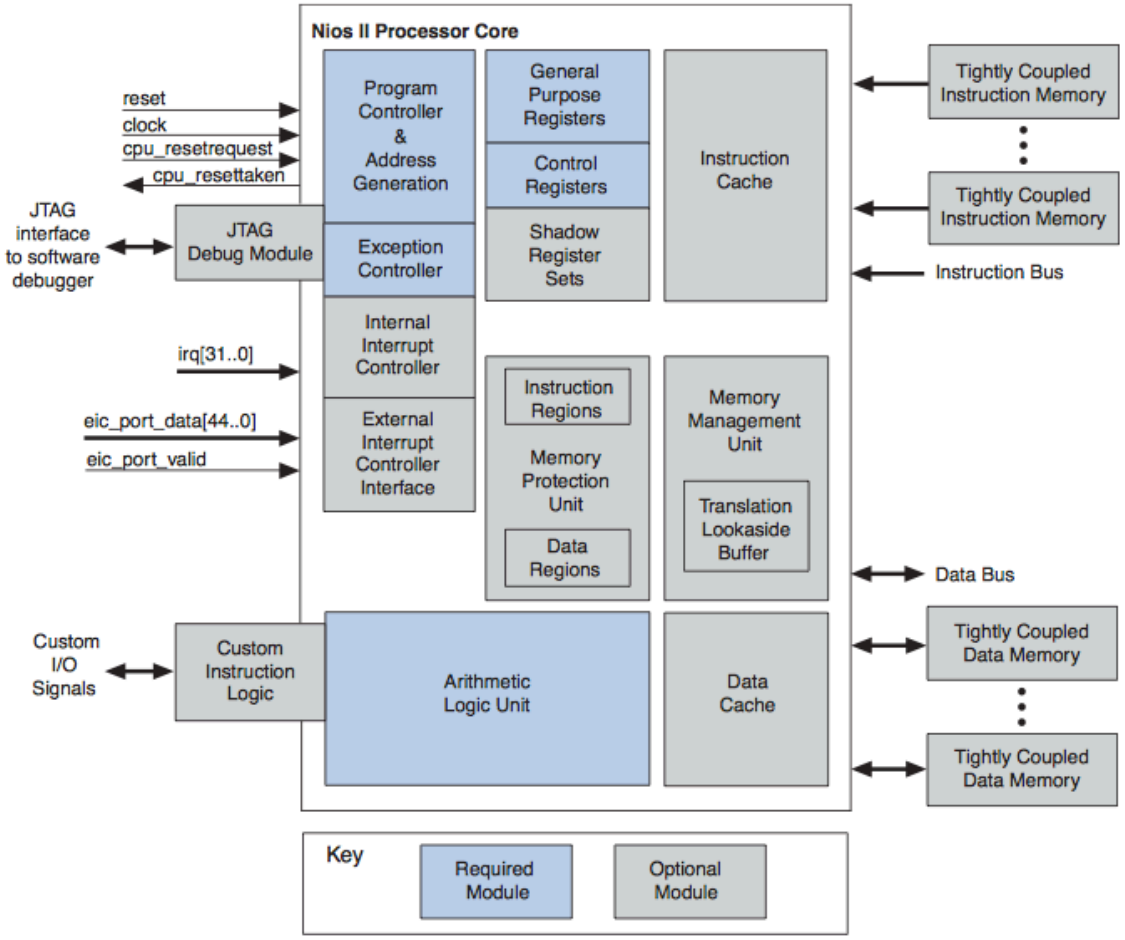

Figura 2.1 – Arquitetura do Processador NIOS II (Altera, 2011).

O Banco de Registradores é um dos módulos obrigatórios e é composto por 32 registradores de 32 bits de propósito geral e até 32 registradores de 32 bits de controle. A Unidade Lógica e Aritmética (ALU) opera em dados que estão salvos nos registradores de propósito geral. Esta arquitetura dispõe de um simples e não-vetorizado controlador de exceções que lida com todos os tipos de exceções lançadas, inclusive interrupções internas de hardware. E, por último, existe um módulo responsável pelos sinais de controle e geração de endereços dos programas.

Dentre os módulos opcionais, existe por exemplo módulos para cache de instruções e de dados que são memórias internas de alta velocidade do NIOS II. Outro módulo é o módulo de depuração JTAG. Este provê uma emulação no chip que permite que o processador seja

controlado por um PC via USB. Este PC pode enviar sinais para este módulo e assim controlar o processador para efeitos de depuração.

O processador NIOS II é conhecido como o processador mais versátil do mundo por ser oferecido em 3 diferentes configurações. Primeiramente, uma versão mais econômica e mais sensível ao custo e consumo de potência. Por outro lado há uma versão mais indicada para sistemas que necessitem alto desempenho, com características em tempo real. E, por fim, uma versão intermediária direcionada para manter o balanço entre desempenho e consumo de potência.

A versão mais rápida do NIOS II, conhecida como NIOS II/f ou *fast core*[5], tem como objetivo principal o alto desempenho. É dito como ótimo para aplicações que necessitam alta performance, como sistemas rodando sistemas operacionais completos, por exemplo [10]. Além disso, possui mais componentes que as outras versões, cujas escolhas são opcionais, tais como memória cache de instruções e de dados separadas, hardware específico para multiplicação, divisão e operações de deslocamento para aprimorar a performance de operações aritméticas, dentre outros componentes. Além destes, possui dois módulos referentes a memória. O primeiro, chamado de MMU (*Memory Management Unit* ou Unidade de Gerenciamento de Memória), gerencia todos os acessos à memória inclusive a tradução de endereços virtuais para endereços físicos, proteção de memoria, controle de cache e alocação de memoria para processos [19]. Este módulo é útil para sistemas que desejam rodar um sistema operacional completo, pois essa unidade é responsável pela proteção de memoria e controle da memoria virtual do sistema operacional. Já o segundo, chamado de MPU (*Memory Protection Unit* ou Unidade de Proteção de Memória), quando ativado e propriamente configurado, monitora todos os dados e acessos de memória e dispara exceções

[5] Versão rápida.

quando acessos ilegais feitos [18]. Este módulo é indicado para casos onde apenas uma unidade de proteção é desejada, sem necessidade do gerenciamento de memoria virtual.

Já a versão intermediária do NIOS II, conhecida como NIOS II/s ou *standard core*[6], tem como objetivo principal propor um tamanho reduzido sem afetar tanto o desempenho. Este usa aproximadamente 20% menos lógica que o NIOS II/f, porém o desempenho chega a cair em até 40% em comparação com esta versão mais rápida [10]. O NIOS II/s possui também alguns componentes opcionais que podem ser escolhidos a critério do projetista, tais como hardware para multiplicação, divisão e operações de deslocamento, módulo JTAG para permitir a depuração de aplicações, entre outros. Porém possui apenas cache de dados e não de instruções, não possui módulos de proteção e gerenciamento da memória, nem alguns outros componentes que a versão NIOS II/f possui.

Por fim, existe a versão mais econômica, conhecida como NIOS II/e ou *economy core*[7], esta tem como objetivo principal possuir o menor núcleo possível. Esta versão foi fabricada com o intuito de reduzir ao máximo a utilização dos recursos mas mantendo a compatibilidade com a arquitetura do conjunto de instruções da família NIOS II [10]. Além disso, não possui a maioria dos componentes que as versões maiores possuem, como memórias cache, hardware para instruções que não foram implementadas (instruções personalizadas), hardware extra para operações aritméticas, sendo assim incapaz de executar instruções de salto, por exemplo.

---

[6] Versão padrão.
[7] Versão econômica.

3 FERRAMENTAS

Neste capítulo são descritas as ferramentas que serão utilizadas para o desenvolvimento deste projeto. Inicialmente o ambiente *Altera Quartus* é descrito, em seguida outras ferramentas utilizadas em conjunto com o Quartus são detalhadas. São elas: *Qsys*, *PowerPlay Power Analyzer* e o *ModelSim-Altera*.

3.1 *Altera Quartus*

Uma das ferramentas propostas pela *Altera* para auxiliar os projetistas no desenvolvimento utilizando suas ferramentas é o *Quartus*. Trata-se de um software que provê além de um ambiente para desenvolvedores de circuitos digitais em *VHDL*[8], *Verilog*[9] e outras linguagens de descrição de hardware, provê também uma forma prática de desenvolvimento e visualização de circuitos digitais, além de permitir ao projetista efetuar simulações e visualizar as formas de onda resultantes (*waveforms*) [11].

Além disto, esta ferramenta possui diversas extensões e outros aplicativos que podem ser acoplados à ela com a finalidade de aumentar suas funcionalidades. Ferramentas que, por

[8] *VHSIC Hardware Description Language* – Linguagem utilizada em designs de circuitos digitais.
[9] Linguagem de descrição de hardware também utilizada para design e verificação de circuitos digitais.

exemplo, estimam o consumo de energia, simulam *FPGAs*, provêm interfaces para auxiliar o projetista a montar e visualizar o circuito, entre outros.

3.2 *Qsys*

Uma das ferramentas que podem ser adicionadas ao *Altera Quartus*, é uma ferramenta de integração de sistemas da *Altera* chamado de *Qsys*. Este permite que o projetista economize tempo e esforços no processo de projeto de um circuito integrado para *FPGAS*, automaticamente gerando toda a lógica de interconexão para conexão de *IP Cores*[10] e subsistemas [12]. Através de uma interface gráfica, permite que o projetista rapidamente crie o seu sistema e ele automaticamente gera todos as conexões de barramentos, lógicas de largura de barramento, decodificação de endereços, entre outros. Além disso, gera automaticamente o código em linguagem de descrição de hardware do sistema, que pode ser útil para etapas futuras de síntese, validação e testes.

3.3 *PowerPlay Power Analyzer*

Uma outra ferramenta que o *Altera Quartus* possui é o *PowerPlay Power Analyzer*. Esta permite ao projetista estimar o consumo de potência do sistema com uma alta precisão.

[10] São unidades lógicas, células ou layout de design de chip reusáveis que pertencem a uma entidade registrada.

Utilizando-se como entrada um arquivo do tipo *.vcd*[11] gerado após uma simulação do sistema, é capaz de estimar o consumo de potência precisamente, utilizando-se de dados inferidos de atividades de sinais, condições do ambiente, requisitos de *clock*, entre outros [13]. A utilização deste tipo de arquivo não é obrigatória porém, quando é feita, a acurácia da potência estimada aumenta, pois este tipo de arquivo contém informações das atividades de chaveamento e probabilidades estáticas para todos os pinos e registradores do sistema [13].

Esta ferramenta gera como resultado uma estimativa da potência consumida pelo sistema, resultado esse que não deve ser usado como uma medição real, mas que pode ser usado para fins de estimativa e otimização do consumo de potência do sistema funcional.

3.4 *ModelSim-Altera*

ModelSim é uma ferramenta de simulação lógica para verificação e depuração de circuitos digitais da empresa *Mentor Graphics*. A *Altera* provê uma versão do *ModelSim* chamado de *ModelSim-Altera* que já inclui as bibliotecas necessárias para a simulação de *FPGAs* da *Altera*. Assim, esta ferramenta suporta simulações comportamentais e em nível de porta lógica para todos os dispositivos da *Altera* [15].

---

[11] Arquivo *Value Change Dump* – padrão de arquivos que contém informações de formas de onda de uma determinada simulação.

4 CÁLCULO DA ENERGIA

Neste capítulo inicialmente é definido o sistema que será utilizado como base para o processador NIOS II para as simulações. Em seguida, são descritos todos os passos necessários para simulação e estimativa do consumo de energia.

4.1 Definindo o sistema

Para definição do modelo de energia do processador NIOS II, foi utilizado como base um tutorial para criação de um sistema utilizando este processador disponibilizado pela própria *Altera*, o *Nios II Hardware Development Design Example*[13]. Este demonstra os passos básicos para criação um sistema simples, utilizando as ferramentas da *Altera* (*Altera Quartus* e *Qsys*), com o processador NIOS II. O sistema criado tem como funcionalidade executar uma aplicação que controla os LEDs da placa FPGA onde for carregado baseado em um timer.

A figura 4.1 mostra cada elemento presente no sistema sugerido. Primeiramente ao sistema é adicionado um chip de memória RAM (*On-chip* RAM) que deverá ser utilizado como memória de dados e de instruções. Como é para uma simples aplicação, o tutorial sugere uma memória de apenas 20KBytes. Em seguida, é adicionado ao sistema um núcleo do processador NIOS II (NIOS II/s *Core*). Este é configurado de uma maneira que seja do tipo NIOS II/s, sem o módulo de divisão, sem o módulo de multiplicação e com uma cache de

instruções de 2Kbytes. Logo após, é adicionado um módulo JTAG UART, capaz de provê uma maneira conveniente de controlar a execução da aplicação enquanto o processador a executa através de um computador utilizando um cabo USB. Após isso, é adicionado um temporizador de intervalo (*Timer*) para prover um pulso de *clock* periódico. Em seguida, um periférico de identificação do sistema (*System Peripheral ID*) é adicionado ao sistema. Este tem como funcionalidade adicionar uma proteção para evitar que aplicações compiladas para outros sistemas com versões diferentes do NIOS II sejam carregadas neste sistema. Após, é adicionado um módulo (PIO – *Parallel* I/O) que provê ao processador a capacidade de facilmente acessar pinos de entrada e saída de sinais da placa. Como este sistema controla os LEDs da placa onde será carregado, este módulo vem com a finalidade de facilitar o acesso a eles.

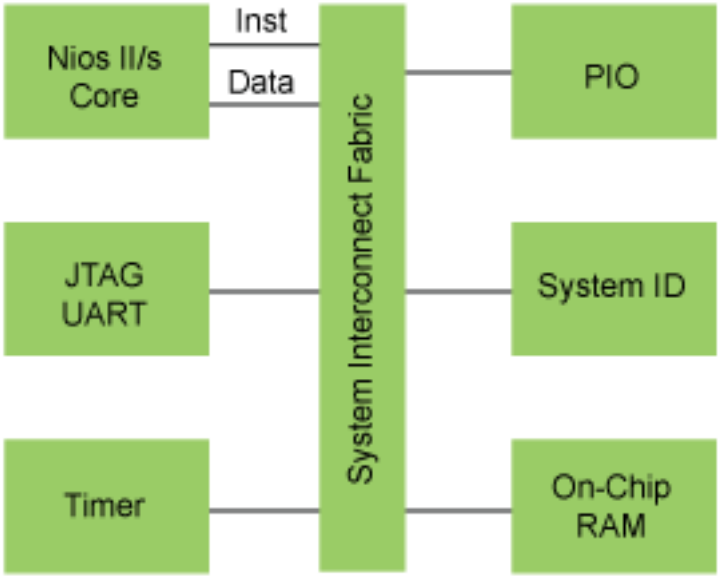

Figura 4.1 – Arquitetura utilizada para obtenção do modelo proposto *(Altera, 2011*).

Após a adição de todos os elementos ao sistema proposto, o sistema enfim pode ser gerado pela ferramenta *Qsys* da *Altera*. A figura 4.2 mostra o sistema definido na ferramenta *Qsys*. Após a geração deste, ele deve ser adicionado a um projeto na ferramenta *Quartus* para em seguida este projeto ser compilado e assim carregado na placa proposta.

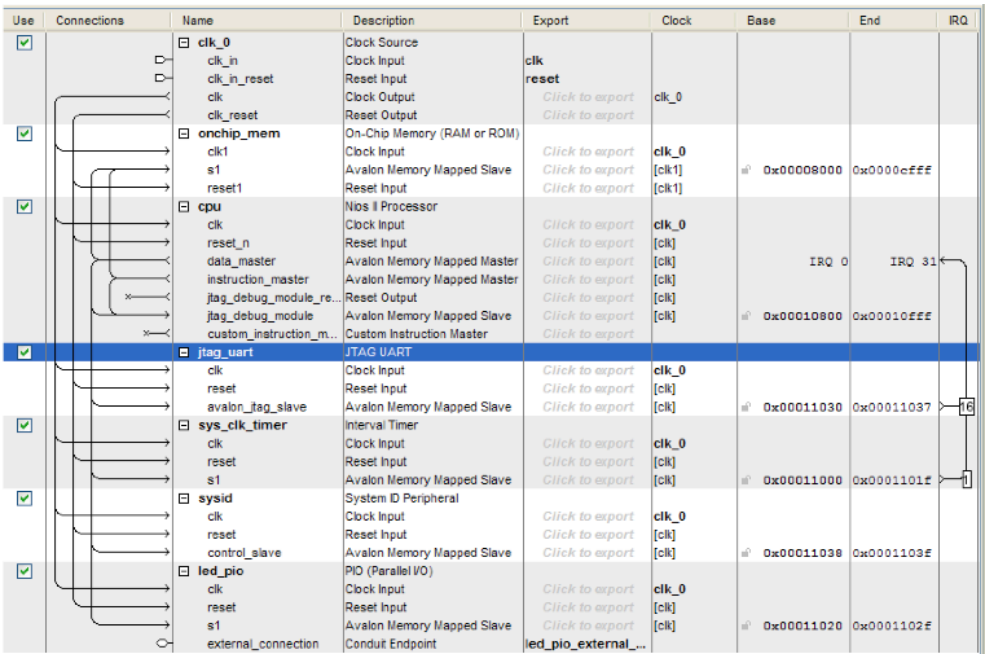

Figura 4.2 – Sistema utilizado para obtenção do modelo proposto.

A figura 4.3 mostra o projeto no Quartus após a adição do sistema gerado pelo *Qsys*. Nesta etapa as saídas do sistemas são ligados aos pinos da placa onde o sistema será carregado, para assim permitir que o sistema funcione corretamente e tenha controle sobre os LEDs da placa.

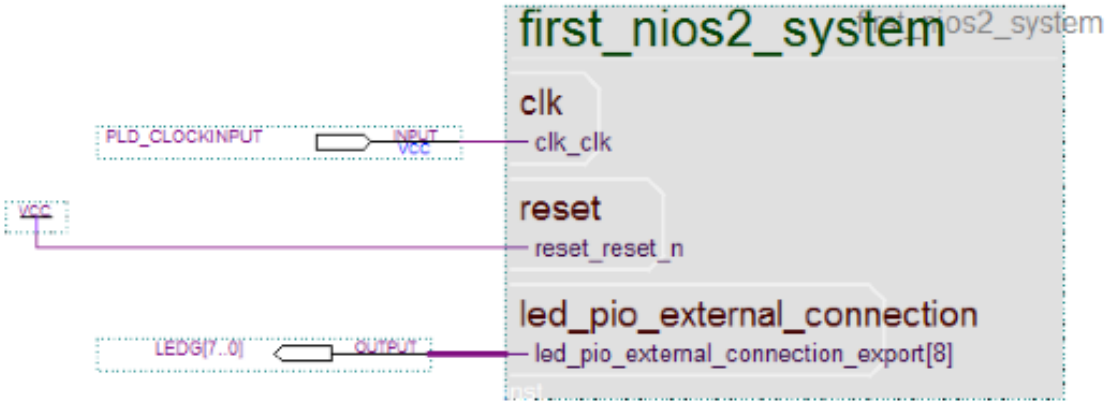

Figura 4.3 – Diagrama final do sistema no *Quartus* (*Altera, 2011*).

Como o foco deste trabalho está apenas na parametrização e nos valores de consumo de potência do processador NIOS II, o sistema acima citado será utilizado como base, porém a

escolha dos parâmetros dos outros componentes do sistema é algo que não afetará nos resultados deste trabalho. Sendo assim, serão utilizados os mesmos valores definidos para o sistema proposto pelo tutorial. Além disso, o sistema não será carregado em uma placa. Ao invés disso, será simulada na ferramenta *ModelSim-Altera*.

4.2 Estimando a energia

Utilizando-se como base o tutorial anteriormente citado, o escopo deste trabalho se resume às atividades descritas no diagrama da figura 4.4, para obtenção de valores de consumo de potência do processador:

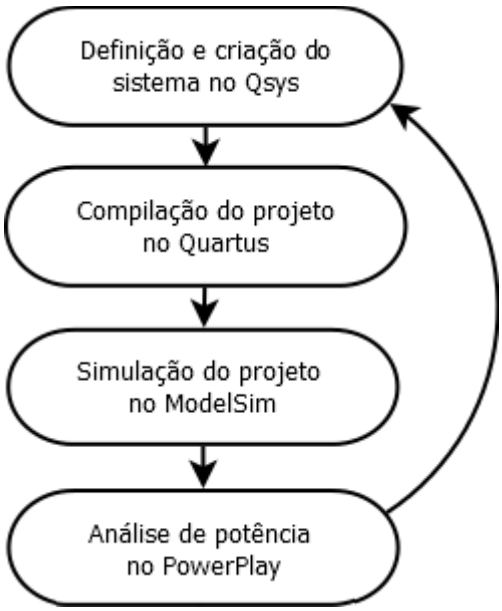

Figura 4.4 – Diagrama de execução para obtenção dos valores de potência.

Inicialmente, uma configuração para o processador NIOS II é escolhida junto com o restante do sistema na ferramenta *Qsys*. Em seguida, o sistema e os arquivos para síntese do

sistema são gerados na mesma ferramenta. O sistema gerado é adicionado ao projeto no *Quartus*. Após a compilação completa do projeto no *Quartus*, deve-se simular o sistema em nível RTL[12] na ferramenta *ModelSim*. Esta etapa será importante para criação do arquivo contendo as informações das atividades dos sinais do sistema após a simulação. Quando o *Quartus* chama o *ModelSim* alguns arquivos são gerados automaticamente para serem utilizados no *ModelSim*. Assim que esta ferramenta é aberta, um *script* é executado e o sistema em si já é compilado na ferramenta. A seguir, a figura 4.5 mostra os passos que devem ser executados para criação da saída desejada.

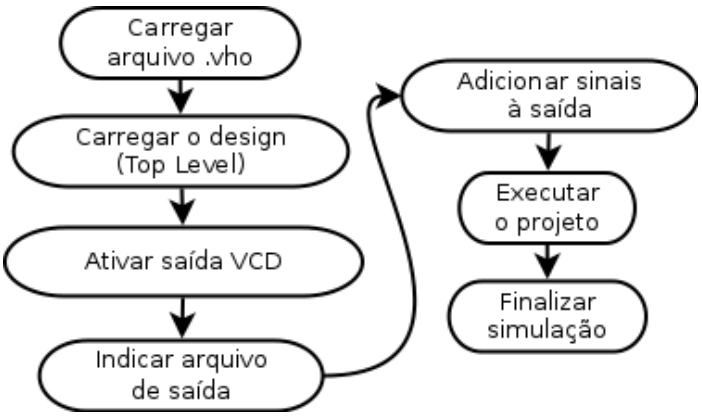

Figura 4.5 – Passo-a-passo para simulação na ferramenta *ModelSim*.

Inicialmente o arquivo .vho gerado após a simulação do *Quartus* é carregado na ferramenta. Este arquivo contém informações de *netlist* do sistema, que nada mais é que as informações de interconexão dos circuitos internos do sistema. Em seguida, o *design* do sistema deve ser definido para o *ModelSim* poder estabelecer a hierarquia do sistema. Após isso, deve-se ativar para que a saída seja escrita num arquivo .vcd. Depois, deve-se indicar o nome deste arquivo. Logo após, deve-se adicionar todas os sinais desejados à saída. Em seguida, deve-se executar a simulação durante o tempo desejado. E, por fim, finalizar a

[12] RTL – Nível de transferência de registradores – comportamento do sistema é descrito em função dos fluxos de sinais [16].

simulação. Após esta etapa, no arquivo .vcd escolhido constará toda a atividade de sinal para esta simulação.

Para facilitar a execução dos passos acima citados, o *ModelSim* permite a criação de um arquivo *script* com extensão .do que permite ao usuário escrever todos os comandos neste arquivo e chamá-lo uma só vez. O *ModelSim* se encarrega de chamar um comando após o outro sequencialmente. O *script* mostrado na figura 4.6 a seguir é um exemplo do passo-a-passo para simulação nesta ferramenta.

```
vcom -reportprogress 300 -work work C:/.../modelsim/nios2_project.vho
vsim +altera –do run_msim_rtl_vhdl.do -l msim_transcript -gui work.nios2_project
vcd on
vcd file my_vcd.vcd
vcd add -r /*
run 1ms
quit –sim
```

Figura 4.6 – Exemplo de script para simulação no ModelSim.

Em seguida, de posse do arquivo .vcd, deve-se importá-lo na ferramenta *PowerPlay Power Analyzer* e assim estimar o consumo de potência relativo ao projeto. Esta última ferramenta basicamente pega a atividade de chaveamento dos sinais do sistema contida no arquivo e estima a potência consumida pelo sistema baseado em dados prévios conhecidos pela ferramenta. Apesar da estimativa gerada pela ferramenta não ser totalmente precisa, pode ser levada em consideração para a geração do modelo de potência proposto.

A execução desta ferramenta no projeto produz como saída um relatório de potência, como o mostrado na figura 4.7 a seguir. Este relatório demonstra as condições que foram usadas para estimar o consumo, como por exemplo a voltagem usada e a temperatura ambiente. Além disso, contém tanto dados da potência térmica total consumida pelo sistema e como também por cada bloco que compõe o sistema, separados por tipo de blocos ou por

hierarquia, diferenciando também potências estáticas e dinâmicas. Potência térmica é a potência que dissipa como calor da FPGA quando o sistema é executado.

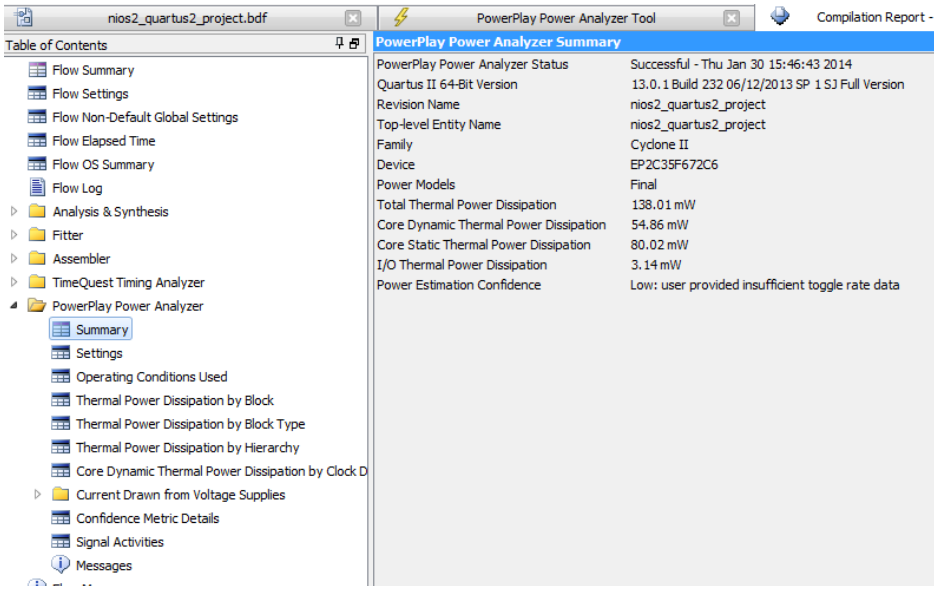

Figura 4.7 – Relatório gerado pelo *PowerPlay Power Analyzer*.

A princípio, a parte importante para este projeto do relatório gerado se encontra no relatório de dissipação de potência por hierarquia, mas precisamente na potência dissipada pela CPU. Este relatório é mostrado na figura 4.8 a seguir.

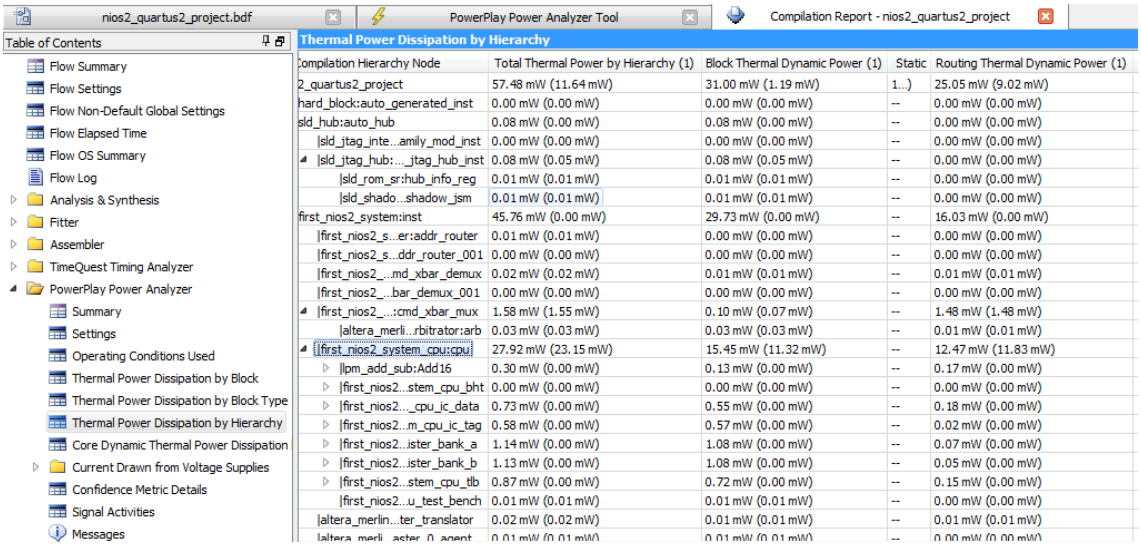

Figura 4.8 – Relatório de potência dissipada por hierarquia.

Neste relatório é mostrada a potência consumida por cada elemento da hierarquia do sistema. Na primeira coluna deste relatório, é mostrada a potência térmica total consumida por aquele nível da hierarquia. O valor mostrado entre parênteses é referente a apenas aquele nível da hierarquia. Enquanto que o valor fora dos parênteses é referente à potência consumida por aquele nível de hierarquia somada a todos os níveis que estão abaixo dele. Sendo assim, o valor correspondente ao componente da CPU e que se encontra fora dos parênteses é o valor que importa para os fins deste trabalho.

Pela definição de energia, sabe-se que:

$$E = P\ x\ t$$

Como já é conhecido o valor da potência e sabe-se que esse valor foi o valor obtido para o tempo de simulação que foi definido na ferramenta *ModelSim*, pode-se assim obter o valor de energia referente àquela simulação.

5 - RESULTADOS OBTIDOS

Neste capítulo inicialmente são descritas as várias simulações feitas para visualização da variação do consumo de energia. Em seguida, os resultados obtidos são comentados e comparados através de gráficos.

5.1 Simulações

Utilizando-se os fluxos anteriormente descritos, foram efetuadas diversas simulações nas quais algum parâmetro do processador NIOS II era alterado e todos os outros eram fixados. Deste modo, foi possível notar variações no consumo de energia para variação de parâmetros específicos. Para todas as simulações efetuadas, foi fixado o tempo de simulação para 1ms.

5.2 Resultados obtidos

O parâmetro principal na definição do NIOS II para o sistema é a definição de qual versão do processador utilizar. Há um aumento de consumo de energia considerável entre as

três versões do processador: NIOS II/e, NIOS II/s e NIOS II/f. O gráfico mostrado na figura 5.1 demonstra a variação do consumo de energia entre as versões mais básicas dos três modelos do processador. Vale ressaltar que o consumo da versão mais rápida deste processador chega a ser duas vezes maior que o consumo da versão mais básica do processador, em iguais condições.

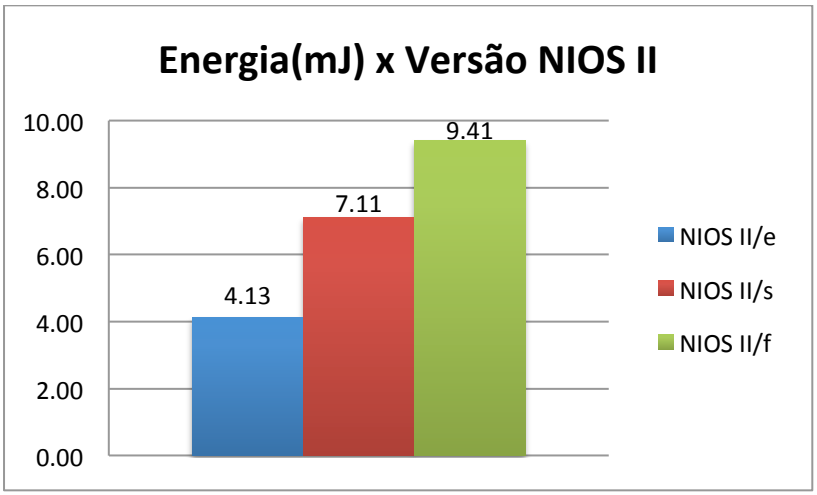

Figura 5.1 – Variação de energia entre as versões básicas do NIOS II.

Do mesmo modo, foi possível perceber que existe um aumento notável quando se adiciona componentes de hardwares à arquitetura do processador. Componentes estes como módulo multiplicador, módulo divisor, memória cache de dados, entre outros. Entretanto, notou-se que a variação ao adicionar os componentes é comum a todas as versões. Ou seja, adicionar um módulo multiplicador à versão de intermediária do NIOS II (NIOS II/s) provoca um aumento no consumo equivalente a adicionar este módulo à versão mais rápida do processador (NIOS II/f).

Sendo assim, os parâmetros a seguir mostrados foram individualmente alterados enquanto que os outros eram mantidos fixados e o consumo de energia foi estimado, como mostrado posteriormente.

- **Presença do módulo depurador JTAG e seu nível de depuração**. Simulações foram efetuadas com e sem esse módulo e, quando existente, foram variados os níveis de depuração. Quanto maior o nível, maior a quantidade de recursos lógicos. Em níveis mais altos, novos pontos de paradas (*breakpoints*) e pontos de controle (*watchpoints*) são adicionados para permitir uma depuração mais precisa e detalhada do código [17]. Os níveis 0 (não-presente), 1, 2, 3 foram variados para as versões NIOS II/s e NIOS II/f e apenas os possíveis níveis 0 e 1 (únicas configurações permitidas) foram variados para a versão NIOS II/e. A figura 5.2 indica um gráfico comparativo do consumo de energia obtido variando-se os parâmetros desse módulo de depuração. Vale ressaltar o quanto aumenta o consumo quando aumenta-se este nível do depurador. No maior nível existente o consumo cresce significativamente devido à grande quantidade de lógica adicionada. Além disso, pode-se perceber que a adição deste módulo provocou praticamente o mesmo aumento de consumo em todas as versões.

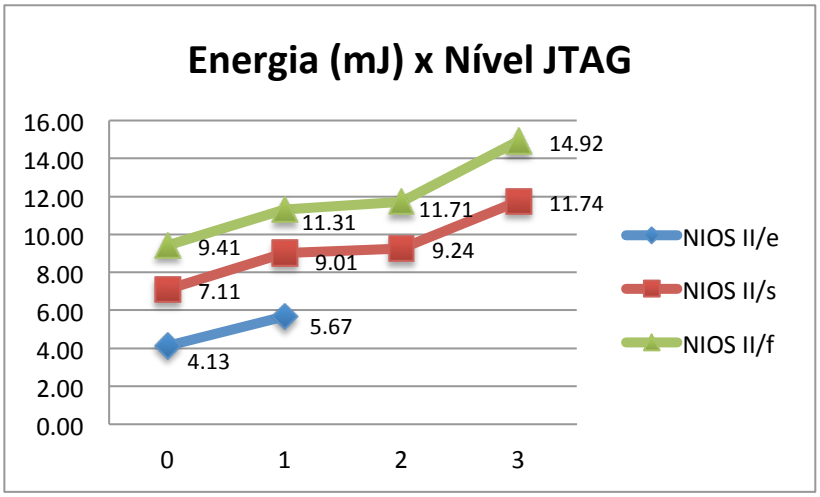

Figura 5.2 - Variação de energia em função do nível do depurador JTAG.

- **Elementos de hardware de multiplicação e divisão**. Estes módulos apenas estão presentes nas versões padrão (NIOS II/s) e rápida (NIOS II/f) do NIOS II. O módulo

de multiplicação pode ser adicionado de duas maneiras: *Embedded Multipliers* e *Logic Elements*. A primeira, chamada de *Embedded Multipliers* ou Multiplicadores embarcados, simplesmente adiciona um módulo multiplicador à Unidade lógica e aritmética do processador. A outra opção, chamada de *Logic Elements* ou Elementos Lógicos, adiciona multiplicadores baseados em elementos lógicos à Unidade lógica e aritmética do processador. Esta última opção consegue um alto desempenho de multiplicação sem a necessidade de um multiplicador embarcado porém afeta a frequência máxima do processador. Por outro lado, para adicionar-se o módulo de divisão, apenas está presente a opção de adicionar um módulo de divisão baseado em elementos lógicos à Unidade Lógica e Aritmética (ALU). A figura 5.3 a seguir mostra um gráfico referente ao aumento de consumo de energia quando adicionados os módulos de multiplicação e divisão. É importante perceber que o aumento de energia é mais significativo para o hardware multiplicador que para o hardware divisor. Além disso, não é significativa a diferença de energia entre as opções de hardware multiplicador.

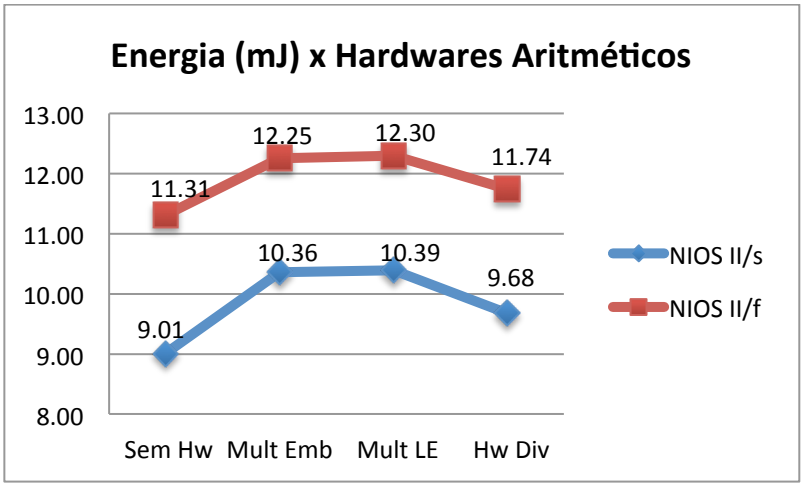

Figura 5.3 – Variação de energia quando adicionados hardwares aritméticos.

- **Memórias Cache de instruções**. Assim como os módulos de multiplicação e divisão, a memória cache de instruções só está disponível para as versões padrão (NIOS II/s) e rápida (NIOS II/f) do processador. Primeiramente, para ambas as versões, foram simuladas arquiteturas com o tamanho da memória cache de instruções variando entre 512 *bytes*, 1 *Kbyte*, 2 *Kbytes*, 4 *Kbytes*, 8 *Kbytes* e 16 *Kbytes*. A figura 5.4 mostra os valores de energia obtidos para estas simulações. Embora o aumento do tamanho da memória cache implique num aumento de desempenho para executar uma aplicação, pois ocorrerá uma diminuição na quantidade de acessos à memória principal, deve-se levar em consideração no projeto que este aumento em muito influencia no consumo de energia do sistema.

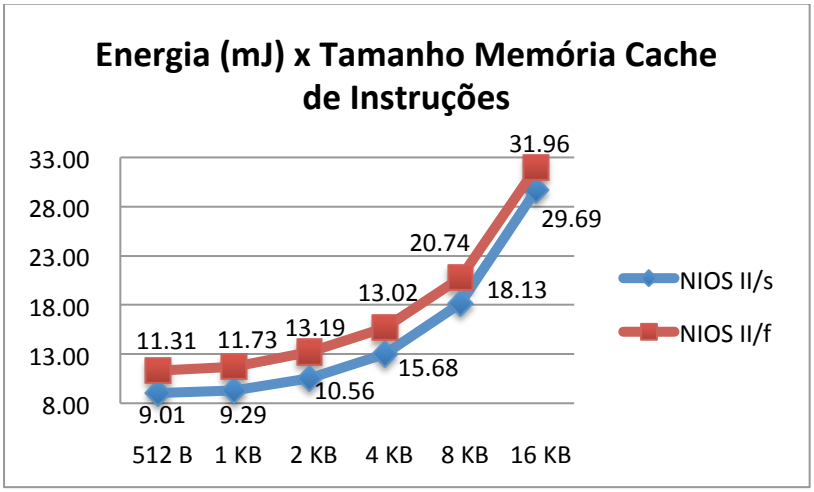

Figura 5.4 – Variação da energia em função do tamanho da cache de instruções.

- **Memória cache de dados**. Este tipo de memória está disponível apenas para a versão mais rápida do processador, a versão NIOS II/f. Para esta versão, é possível definir o tamanho da memória cache de instruções em si e o tamanho da linha desta cache. Sendo assim, nas simulações efetuadas foram utilizados os valores de 512 *Bytes*, 1 *Kbyte*, 4 *Kbytes* e 16 *Kbytes* para o tamanho da memória cache e os valores de 4 *Bytes*, 16 *Bytes* e 32 *Bytes* para o tamanho da linha. A figura 4.5 exibe os valores de energia

obtidos para tais simulações. É interessante notar que para memórias cache de maior tamanho o consumo é maior quando o tamanho da linha é menor, por possuir uma quantidade maior de linhas.

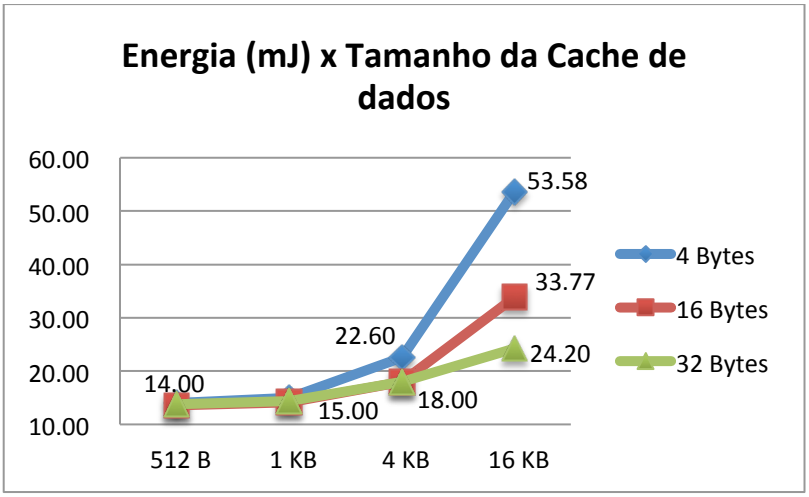

Figura 5.5 – Variação de energia em função do tamanho da linha e da cache de dados.

- **Módulos de gerenciamento e proteção de memória**. Como já dito anteriormente, a versão mais rápida do NIOS II (NIOS II/f) possui ainda módulos adicionais para gerenciamento e proteção de memória. A presença destes indica um aumento no consumo de energia do processador, como mostrado na figura 5.6 a seguir. Nesta, é mostrada a comparação entre o consumo do processador sem nenhum módulo e com cada módulo separadamente.

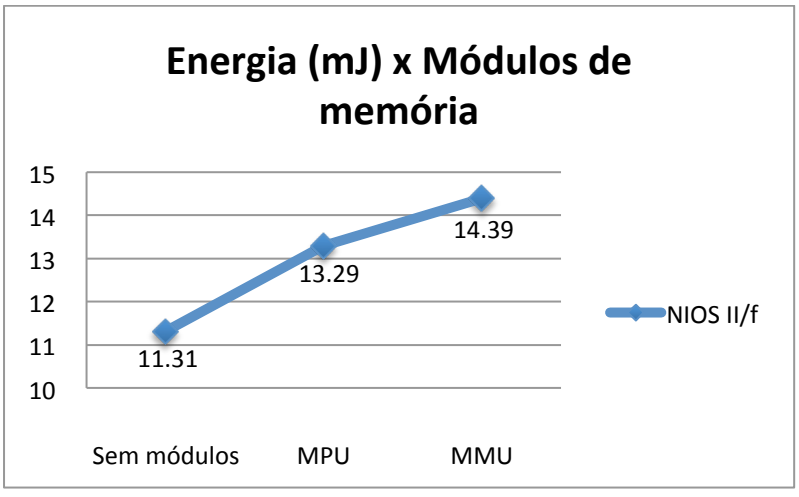

Figura 5.6 – Variação de energia quando adicionados módulos gerenciais da memória.

5.3 Modelo de energia

A partir dos valores anteriormente encontrados, é possível determinar um modelo de energia para o processador citado, com base nos parâmetros escolhidos e nos valores de energia simulados. Com esse modelo é possível obter um valor aproximado sem a necessidade de simular novamente todo o sistema e, assim, tornar possível uma estimativa prévia durante a etapa de projeto da arquitetura.

A partir dos valores anteriormente obtidos, foi possível criar aproximações e definir padrões nos valores, utilizando os gráficos gerados e proporções. E, deste modo, definir um modelo genérico para estimativa do consumo de energia do processador NIOS II. Modelo este definido a seguir, onde *t* é o tempo de simulação do sistema:

$$E = t * \{4.13 * Ne + 7.11 * Ns + 9.41 * Nf + 1.54 * JTAGLevel +$$

$$(Nf + Ns) * [1.3 * Mult + 0.55 * Div + CD(2.36 * TCD + 7 + 2 * Nf)] +$$

$$Nf * (MPU * 2 + MMU * 4) +$$

$$Nf * CI * [TL4 * (8.7 * TCI + 5.3) + TL16 * (6.7 * TCI + 7.3) + TL32$$

$$* (3.3 * TCI + 10.7)]\}$$

Os valores *Ne*, *Ns* e *Nf* são referentes à versão escolhida para o processador, entre econômica, padrão e rápida. Caso a versão escolhida seja a versão econômica, *Ne* deve ter

valor definido igual a 1 e os outros valores (*Ns* e *Nf*) devem ser definidos como 0, e assim sucessivamente.

O valor de *JTAGLevel* é referente ao nível do depurador JTAG e deve variar entre 0 e 1 (0 para a ausência do modulo JTAG), se a versão escolhida do processador for a econômica e entre 0 e 3, se outra versão.

Mult deve ser igual a 1, caso algum módulo multiplicador seja desejado e 0, caso contrário. O mesmo acontece para *Div*, no caso de possuir ou não um módulo divisor.

CD deve ser 1, se existir uma memória cache de dados, e 0, caso contrário. O valor de TCD é referente ao tamanho dessa memória cache de dados. Ele deve variar entre 1 e 6, valores estes associados respectivamente aos valores de 512 *bytes*, 1 *Kbyte*, 2 *Kbytes*, 4 *Kbytes*, 8 *Kbytes* e 16 *Kbytes*.

MPU deve ser igual a 1, caso exista o módulo de proteção de memória da versão mais rápida do NIOS II e 0, caso contrário. O mesmo acontece para o valor de MMU, o módulo de gerenciamento da memória desta mesma versão.

Por fim, CI indicará se existe, valor igual a 1, ou não, valor igual a 0, memória cache de instruções. TL4, TL16 e TL32 são também valores binários e indicam, respectivamente, se o tamanho da linha desta memória cache de instruções é de 4 *Bytes*, 16 *Bytes* ou 32 *Bytes*. O valor de TCI deve variar entre 1 e 4, valores estes associados aos valores de 512 *bytes*, 1 *Kbyte*, 4 *Kbytes* e 16 *Kbytes* para o tamanho desta memória cache.

6 - CONCLUSÃO

Neste capítulo final, os resultados obtidos são comentados e comparados aos objetivos iniciais do projeto.

Após a identificação dos parâmetros que poderiam ser alterados na criação do módulo do processador NIOS II e execução de várias simulações variando os valores destes, foi possível estimar o consumo de energia para diversas configurações deste processador. A partir daí, foi possível determinar padrões nos valores encontrados, a partir de proporções e gráficos gerados. Em seguida, com base nestes padrões, criou-se um modelo de energia para o dado processador que permite que um valor de energia seja gerado partindo-se apenas da escolha dos parâmetros para a criação deste processador.

Os valores obtidos a partir do modelo de energia não devem ser usados para efeitos de comparação real. Porém, estes servem como base para etapa de projeto de uma arquitetura, se esta for feita visando consumo de energia, economizando-se assim, tempo de projeto.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] NORTHERN III, J.; SHANBLATT, M. An Evolutionary Approach to Configuring an Embedded System Based on Power Consumption. **Proceedings of The 3rd IEEE International Workshop on System-on-Chip for Real-Time Applications**, p. 201-204, jun.-jul. 2003.

[2] **Nios II Processor: The World's Most Versatile Embedded Processor**. Altera. Disponível em: <http://www.altera.com/devices/processor/nios2/ni2-index.html>. Acesso em: 16 janeiro 2014.

[3] **About Us**. Disponível em: <http://www.altera.com/corporate/about_us/abt-index.html>. Acesso em: 17 janeiro 2014.

[4] ZACKS, Len/Zacks.com. **Altera Shipping 28-nm FPGAs - Analyst Blog**. NASDAQ. 2012. Disponível em <http://www.nasdaq.com/article/altera-shipping-28nm-fpgas-analyst-blog-cm133782>. Acesso em: 17 janeiro 2014.

[5] **FPGAs**. Altera. Disponível em: <http://www.altera.com/products/fpga.html>. Acesso em: 17 janeiro 2014.

[6] **Difference Between FPGA and CPLD**. DifferenceBetween.net. Disponível em: <http://www.differencebetween.net/technology/difference-between-fpga-and-cpld/>. Acesso em: 17 janeiro 2014.

[7] **Nios Embedded Processor**. Altera. Disponível em: <http://www.altera.com/products/ip/processors/nios/nio-index.html>. Acesso em: 21 janeiro 2014.

[8] **Nios II**. Wikipedia. Disponível em: <http://en.wikipedia.org/wiki/Nios_II>. Acesso em: 21 janeiro 2014.

[9] **Nios II Processor Reference Handbook**. Altera. Disponível em: <http://www.altera.com/literature/hb/nios2/n2cpu_nii5v1.pdf>. Acesso em: 21 janeiro 2014.

[10] **Nios II Core Implementation Details**. Altera. Disponível em: <http://www.altera.com/literature/hb/nios2/n2cpu_nii51015.pdf>. Acesso em: 21 janeiro 2014.

[11] **Altera Quartus**. Wikipedia. Disponível em <http://en.wikipedia.org/wiki/Altera_Quartus>. Acesso em: 27 janeiro 2014.

[12] **Qsys - Altera's System Integration Tool**. Altera. Disponível em: <http://www.altera.com/products/software/quartus-ii/subscription-edition/qsys/qts-qsys.html>. Acesso em: 27 janeiro 2014.

[13] PowerPlay Power Analysis. Altera. Disponível em: <http://www.altera.com/literature/hb/qts/qts_qii53013.pdf>. Acesso em: 30 janeiro 2014.

[14] **Nios II Hardware Development Design Example**. Altera. Disponível em: <http://www.altera.com/support/examples/nios2/exm-hardware-tutorial.html>. Acesso em: 30 janeiro 2014.

[15] **ModelSim-Altera Software**. Altera. Disponível em: <http://www.altera.com/products/software/quartus-ii/modelsim/qts-modelsim-index.html>. Acesso em: 20 fevereiro 2014.

[16] **Register transfer level**. Wikipedia. Disponível em> <http://pt.wikipedia.org/wiki/Register_transfer_level>. Acesso em: 20 fevereiro 2014.

[17] **Debugging Nios II Designs**. Altera. Disponível em: <http://www.altera.com/literature/hb/nios2/edh_ed51003.pdf>. Acesso em: 20 fevereiro 2014.

[18] **Nios II MPU Usage**. Altera. Disponível em: <http://www.altera.com/literature/an/an540.pdf>. Acesso em: 20 fevereiro 2014.

[19] **Programming Model**. Altera. Disponível em: <http://www.altera.com/literature/hb/nios2/n2cpu_nii51003.pdf>. Acesso em: 20 fevereiro 2014.

[20] HOLANDA, J. A. M. **Projeto de um Estimador de Potência para o Processador Nios II da Altera**. USP – São Carlos, março 2007.

[21] SENN, L.; SENN, E.; SAMOYEAU, C. **Modelling the power and energy consumption of NIOS II softcores on FPGA**. Proceeding CLUSTERW '12 Proceedings of the 2012 IEEE International Conference on Cluster Computing Workshops. p. 179-183. 2012.