Implementação física de um microprocessador Risc de 32-bits usando tecnologia XFAB 600nm

Ramon Yago da Cruz Jacques Vieira
Universidade Federal do Rio Grande do Sul (UFRGS)
(rycjvieira@restinga.ifrs.edu.br)

Thaciaine Coelho Tavares
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
(tctabares@restinga.ifrs.edu.br)

Kelvin Rutsatz Costa
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
(krucosta@restinga.ifrs.edu.br)

Steffani Laurindo Silva
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
(slsilva@restinga.ifrs.edu.br)

Bruno Canal
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
(bruno.canal@restinga.ifrs.edu.br)

Alexsandro Cristovão Bonatto
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
(alexsandro.bonatto@restinga.ifrs.edu.br)

Resumo: Este artigo tem o propósito de relatar a implementação física de um processador de 32-bits de arquitetura do tipo RISC, de conjunto de instruções reduzidas, denominado de RISCO. Este processador é projetado para uma arquitetura simples com a capacidade reconfigurável, proporcionando uma fácil adaptação para os limites especificados dentro de um projeto de SoC. O processador foi implementado na tecnologia XFAB 600nm usando o conjunto de ferramentas EDA Cadence. O principal objetivo do projeto é obter e compartilhar experiências em projetos ASIC e no desenvolvimento de síntese física de circuitos integrados. A implementação física do processador em questão resultou em um circuito de 15,18 mm² com 9.247 células e uma potência estimada de 254 mW.

Palavras-chave: microeletrônica; microprocessador; implementação física.

Physical implementation of a 32-Bits Risc microprocessor using XFAB 600nm technology

Abstract: This paper relates the physical implementation of a 32-bits Reduced Instruction Set Architecture (RISC) processor. This processor is designed to be a simple architecture with reconfigurable capability, providing easy adaptation to the specified limits within a SoC project. The processor was implemented in XFAB 600 nm technology using the Cadence EDA Tools. The main aim of the paper is to obtain and share experience in ASIC projects and in the development integrated circuits physical synthesis. The physical implementation result in a circuit of 15.18 mm² with 9,247 cells and an estimated power of 254 mW.

Keywords: microelectronics; integrated circuit; microprocessor; physical implementation.

1. INTRODUÇÃO

Os circuitos integrados (CI) do tipo de sistemas em chip (System-on-Chip, ou SoC) modernos são projetados para diferentes tipos de aplicações, desde circuitos de processamento *high-end* até circuitos *low-power,* como aqueles usados nas tecnologias de Internet das Coisas (*Internet of Things*, ou IoT). Os atuais CIs produzidos utilizando nós de tecnologia de ponta, como o próximo passo da Lei de Moore de 10 nanômetros propostos pela Intel, permitem a integração de vários módulos de hardware no mesmo chip que microprocessadores heterogêneos, diferentes bancos de memória com grande capacidade e vários periféricos para comunicação e controle. Além do mais, o processamento e o controle remoto presentes nas aplicações de IoT, tomam proveito de CIs de baixa potência devido ao menor consumo de energia.

Como consequência da evolução do nó tecnológico, seguido pelo desenvolvimento de ferramentas EDA (*Electronic Design Automation*), a configuração dos CIs digitais aborda a integração de vários módulos IP (*Intellectual Property*) em um único chip. IPs, lógica de cola e componentes dos CIs são descritos usando Linguagem de Descrição de Hardware (HDL). Portanto, essas estruturas podem ser projetadas de um modo reconfigurável que fornece uma adaptação fácil a diferentes aplicativos ASIC. A descrição do HDL permite o uso do fluxo padrão de design da célula CI. O fluxo do projeto de célula padrão usa lógica pré-design padronizada em ferramentas de design auxiliado por computador (CAD) para construir o layout de um CI (RABAEY, CHANDRAKASAN e NIKOLIC, 2002). Assim, incluindo estruturas reconfiguráveis no fluxo padrão de projeto das células lógicas da biblioteca, fornece ao projetista uma boa relação de confiabilidade e tempo de colocação no mercado.

O domínio das etapas de processos de fabricação e verificação do CI, junto ao conhecimento completo das opções de configuração do conjunto de ferramentas, são fundamentais para alcançar o sucesso na fabricação de circuitos integrados. Essas etapas são exploradas neste trabalho, usando as ferramentas Cadence IC Design e uma conhecida tecnologia de fabricação fornecida pelo processo da *foundry* alemã XFAB denominado de XC06, através da empresa Brasileira CEITEC S.A. Este mesmo nó tecnológico foi utilizado para fabricar os processadores Intel DX4™ (1994), Intel® Pentium® de 75 a 120 MHz (1994) e Intel® Pentium® Pro de

150 à 200 MHz (1995). O processador Intel® Pentium® Pro foi introduzido em 1995 com velocidade de relógio iniciando em 200 MHz e contendo 5.500.000 transistores (INTEL, 2017). O processador Intel® Pentium® foi fabricado usando 3,3 milhões de transistores em uma área 163 mm².

Este artigo apresenta a implementação física do microcontrolador Risco, um microcontrolador reconfigurável de 32-bits de largura de palavra de dados e instruções, com conjunto reduzido de instruções (RISC). A arquitetura e o conjunto de instruções Risco foram apresentadas em JUNQUEIRA (1993). Foi elaborado pela primeira vez utilizando uma linguagem de descrição de hardware, conhecida como HDC, baseada em C e posteriormente implementada usando VHDL pelo grupo de pesquisa LaPSI (UFRGS-Brasil). Esta descrição VHDL é usada neste trabalho, com algumas estruturas de codificação que foram refeitas para a síntese em ferramentas ASIC. O fluxo de design ASIC tem como entrada uma descrição de hardware VHDL e gera um arquivo de banco de dados gráfico (GDS). Este trabalho apresenta os desafios da implementação física do microcontrolador Risco e faz uma discussão referente a isso para encontrar uma boa solução para o processador projetado.

Este artigo está organizado da seguinte forma: a Seção 2 apresenta a descrição arquitetônica do Risco; a Seção 3 apresenta a metodologia utilizada na implementação do microcontrolador Risco; a Seção 4 discute a análise dos resultados da implementação ASIC do microcontrolador Risco; a Seção 5 apresenta a metodologia de teste dos circuitos fabricados; finalmente, a Seção 6 apresenta as observações finais.

2. DESCRIÇÃO DA ARQUITETURA DO MICROPROCESSADOR RISCO

Esta seção apresenta as principais características da arquitetura do processador Risco. A implementação do Risco na VHDL tem um conjunto de parâmetros configuráveis que podem ser usados para definir suas características. Por exemplo, o número de registro e pinos IO (*Input/Output*) podem ser configurados. Tabela 1 mostra algumas das características do processador Risco usadas nesta implementação.

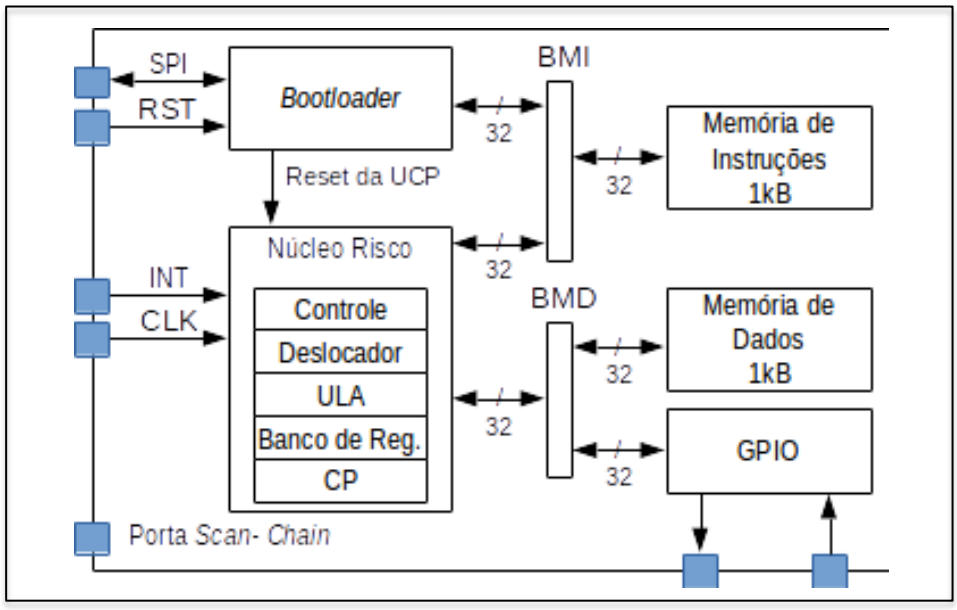
Tabela 1: Características do Processador.

Parâmetro	Tamanho	Parâmetro	Tamanho
Banco de registro	32	Pinos de saída de dados	8
Pinos de entrada de dados	8	Ciclos de acesso de memória	2

Fonte: Autores

Com uma arquitetura simples, o processador Risco introduz um pipeline de 3 estágios para a execução de instruções: busca na memória (1º estágio), execução da instrução (2º estágio) e gravação do resultado (3º estágio). É capaz de atingir um pico de um ciclo de máquina para a execução da maioria de suas instruções (JUNQUEIRA, 1993), e para instruções de salto é usado um ciclo extra. A Figura 1 mostra os módulos principais dentro do processador Risco.

Figura 1: Diagrama de blocos da arquitetura do processador Risco.



Fonte: Autores

O processador Risco está configurado conforme arquitetura Harvard, de forma que o núcleo interno faz acesso à duas memórias separadas, em barramentos distintos, sendo um para dados BMD (Barramento de Memória de Dados) e outra para instruções BMI (Barramento de Memória de Instruções). Tais memórias são voláteis, do tipo SRAM, com capacidade de 1 KiB cada. É necessário o uso de um *bootloader* para carregar o código executável diretamente de uma memória externa

(não volátil, do tipo Flash ou EEPROM) à memória de instruções. Como mostra a Figura 1, o *bootloader* está conectado a uma interface SPI (*Serial Peripheral Interface*) do tipo mestre[1]. Ambos os barramentos BMI e BMD possuem 32 bits de largura de dados bidirecional, 10 bits de largura de endereços e sinais de controle para escrita, habilitação e estado. Cada memória possui capacidade de armazenamento de 256 palavras de 32 bits.

O ciclo de inicialização do microprocessador ocorre da seguinte forma: após pulso de Reset ativo (Pino RST='1'), o módulo de *bootloader* reinicializa a UCP (Unidade Central de Processamento, ou núcleo Risco), mantendo-a em estado de bloqueio; a UCP faz o contador de programa (CP) igual a 0x000; em seguida, o *bootloader* carrega através da interface SPI o código executável para a memória de instruções, através do BMI; ao finalizar a carga da memória, o *bootloader* libera a UCP para executar as instruções de programa.

O microprocessador Risco também possui uma interface de uso geral para pinos de entrada e saída de dados, denominada de GPIO (*General Purpose Input/Output*). Essa interface está configurada para 8 pinos de saída e 8 pinos de entrada, sendo acessada pela UCP através das instruções LD (carrega da memória) e ST (armazena na memória), e suas variações, apontadas para o endereço 0x100.

3. METODOLOGIA DE PROJETO

A implementação física foi desenvolvida com base no fluxo de design de células padrão (*Standard Cell Design*). Esse fluxo recebe como entrada uma descrição do circuito à nível de transferência de registro (RTL, do inglês *Register Transfer Level*) e uma biblioteca células digitais padrões. O caso estudado usa uma biblioteca de células padrão comercial de uma tecnologia de 0,6 µm. Ao final, o processo de *backend* resulta em um arquivo GDS, que contém informações das máscaras de leiaute necessárias no processo de fabricação.

O fluxo de projeto do backend do circuito integrado digital está dividido em duas etapas: síntese lógica e síntese física. Este capítulo descreve o desenvolvimento do backend do processador Risco em um conjunto de ferramentas de EDA da Cadence.

[1] Existe no mercado diversas memórias não voláteis do tipo Flash, como o modelo SST25VF016B, do fabricante Microchip, que será usado na placa de testes com este microprocessador.

3.1 Descrição da síntese lógica

O projeto de síntese lógica é responsável por traduzir uma descrição RTL em um circuito mapeado em portas lógicas de uma dada tecnologia. O mapeamento leva em consideração o design do circuito, as restrições de tempo e as características da biblioteca de células padrão, a fim de fornecer um circuito que atenda às restrições de tempo do projeto buscando a otimização de área e potência do mesmo. Em outras palavras, por padrão, as ferramentas EDA buscam a melhor relação de área e energia que satisfaça as especificações de tempo requeridas. Esta restrição de tempo é definida no arquivo *Synopsys Design Constraints (SDC)*.

No caso estudado, uma série de sínteses lógicas foram desenvolvidas usando a ferramenta *RTL Compiler*. Essas sínteses foram feitas para explorar e entender o comportamento das ferramentas de síntese e do projeto em questão. Esta série de sínteses iniciou-se com uma síntese lógica levando em consideração as condições nominais de processo, tensão e temperatura (PVT), isto é, 3,3 V de tensão de alimentação e 25 ºC de temperatura e sem variações no processo de fabricação. Esta síntese também não considerou restrições de tempo para o circuito. A segunda síntese foi feita levando em consideração as piores condições PVT da biblioteca de células, neste caso todos os parâmetros PVT tendem a piorar o desempenho, em termos de atraso de propagação, das células lógicas utilizadas. Ou seja, neste caso considera-se 3,0 V de tensão de alimentação e 125 ºC de temperatura e variações de processo que penalizam o chaveamento e capacidade de corrente dos transistores. A diferença destas sínteses aparece principalmente nos resultados de tempo de propagação interno do circuito. Com base nessas análises, utilizou-se o resultado da síntese lógica de pior caso para definir o valor inicial da restrição de tempo. Depois disso, o projeto seguiu através de sucessivas análises para estudar o comportamento da síntese com maiores restrições de tempo. A restrição de tempo foi aumentada até alcançar uma frequência de relógio de 20 MHz.

Após alcançar a frequência requerida (20 MHz), incluiu-se estruturas de testes do tipo *Scan Chain* na síntese lógica e as restrições de tempo foram modificadas através da inserção do comando *adjust_path*. O comando *adjust_path* representa uma redução virtual nas restrições de tempo. A redução virtual é útil uma vez que, na síntese física, o atraso de propagação do caminho crítico tem um

incremento devido às capacitâncias parasitas adicionadas ao circuito. Portanto, este comando insere uma folga nas restrições de tempo que posteriormente, dentro do fluxo de implementação, será aproveitado pela síntese física.

3.2 Descrição da síntese física

Na síntese física, o circuito de portas lógicas gerado pela síntese lógica é convertido em um leiaute e por fim o resultado é o arquivo GDS. O arquivo GDS corresponde ao arquivo que é enviado para a *foundry* na qual o circuito integrado será fabricado. Neste arquivo seguem todas as informações sobre a construção das máscaras de fabricação.

A síntese física pode ser dividida em cinco etapas principais: *floorplanning*, *power planning*, posicionamento, síntese de árvore de relógio (CTS, do inglês *Clock Tree Synthesis*) e roteamento. Todas essas etapas foram feitas através da ferramenta de EDA *Encounter*. No *floorplanning*, o projetista define uma ideia preliminar de alocação de blocos. O *power planning* é a etapa em que o anel e trilhas de alimentação são definidos. O posicionamento responsável pela alocação das células lógica, e o CTS gera a árvore de relógio que é responsável pela distribuição do sinal de sincronismo (*clock*) dos flip-flops. Por fim, a etapa de roteamento gera os caminhos de interconexão entre as células lógicas do projeto.

No projeto desenvolvido, a limitação de área do circuito era uma especificação fixa, sem a opção de revisão, visto que esta área foi cedida pela *foundry* para a fabricação do CI. Portanto, o *floorplanning* é de fundamental importância neste projeto. Neste ponto a característica reconfigurável da descrição RTL do processador foi de suma importância. Essa característica permite que, de uma maneira rápida e eficiente seja possível determinar quantas conexões estarão disponíveis para a comunicação externa no processador, ou seja, o número de entradas e saídas de uso geral, GPIO. O número de GPIOs, no projeto em questão, pode determinar se a área do CI projetado será limitada pelo número de pinos ou devido à área ocupada pelo circuito de células lógicas. O projeto em questão requer diversas conexões externas. A característica reconfigurável da descrição RTL permite que este número de conexões seja alterado facilmente. Em uma breve análise, pode-se concluir que a melhor estratégia é inserir o máximo de conexões, tornando a área do projeto limitada pelo número de *pads*. Esta estratégia faz com

que seja necessária a utilização de *pads* com um fator de forma mais estreito e comprido, consequentemente haverá menos área interna disponível para o posicionamento das células lógicas. Portanto, para definir o número de *pads* GPIO, um estudo foi feito para encontrar a área central disponível levando em consideração os diferentes tipos de *pads* disponíveis.

A etapa de *power planning* leva em consideração a potência estimada na síntese lógica e a queda de tensão das camadas de roteamento. No caso estudado, a simulação da queda de tensão devido às características resistivas do roteamento não foi realizada devido à falta de arquivos da tecnologia usada neste projeto. Portanto, a simulação não estava disponível na ferramenta EDA. A metodologia utilizada para substituir a estimativa de queda de tensão foi considerar a capacidade máxima de corrente dos metais de roteamento dos anéis e trilhas de alimentação e encontrar o comprimento máximo que não resulta em uma queda de tensão prejudicial ao correto funcionamento do projeto.

A próxima etapa da síntese física é o posicionamento das células e o CTS. Note que antes e depois do CTS, o projetista deve otimizar o projeto através do comando "*optDesign [-preCTS -postCTS]"*. Este comando de otimização busca realocar e/ou redimensionar as células lógicas para que as restrições de tempo do projeto sejam alcançadas. A otimização leva em consideração as novas células de *buffer* adicionadas na árvore do relógio e a estimativa de roteamento do sinal. Este comando também é usado após rotear todo o projeto, neste caso utilizando o argumento "*-postRoute"*.

Ao final da síntese física, ainda na ferramenta *Encounter*, a dissipação de potência do projeto é reestimada. A estimativa de potência pode ser feita de duas formas: por meio de uma análise estática ou uma análise dinâmica. A análise estática leva em consideração apenas a atividade de comutação das entradas e, por meio da probabilidade de propagação de comutação do sinal, é determinada a dissipação de potência do projeto. A diferença entre esta estimativa e a estimativa de potência realizada na síntese lógica é a extração RC do leiaute. A estimativa de potência dinâmica é mais precisa, uma vez que esta análise usa como entrada um vetor com transições de sinal. Este vetor apresenta um sinal definido pelo usuário que pode representar uma atividade de entrada real. Com as características do sinal de entrada real, os tipos e quantidades de transições de nível lógico efetuadas no circuito são mais precisas, resultando em uma melhor estimativa de potência.

As últimas etapas do projeto na ferramenta *Encounter* é o roteamento das interconexões das células, as checagens de leiaute, DRC (do inglês, *Design Rules Check*), e a checagem de equivalência entre o leiaute e a descrição RTL. Encerradas estas etapas, é gerado o arquivo *Design Exchange Format (DEF)* que possibilita a exportação do leiaute para o ambiente de desenvolvimento *Virtuoso.* No ambiente *Virtuoso*, a exibição do leiaute das células é adicionado e uma nova verificação de DRC é feita. Uma vez que foi obtido sucesso em todas essas verificações, o arquivo GDS é gerado e enviado para a *foundry*.

4. ANÁLISE DE RESULTADOS E COMPARAÇÃO

A Tabela 2 mostra os resultados das análises de síntese lógica. Nesta tabela, a primeira coluna identifica as condições de PVT, a segunda mostra se existem ou não as estruturas de teste na síntese. A terceira e a quarta colunas exibem dados sobre as especificações de atrasos de propagação definidos no arquivo SDC. A quinta coluna mostra a resposta de temporização da síntese do *slack* do caminho crítico. Observe que o primeiro e o segundo valor desta coluna correspondem ao atraso de propagação do caminho crítico uma vez que não há restrições de tempo para essas sínteses. As últimas três colunas demonstram, respectivamente, a dissipação de potência, a área ocupada pelas células lógicas e memórias e o número de portas lógicas utilizadas na síntese.

Tabela 2: Análise da Síntese Lógica.

Condição PVT	Estrutura de Teste	Caminho Crítico (ns)	Valor de *adjust_path* (ps)	*Slack* time (ps)	Potência (mW)	Área do Núcleo (mm^2)	Número de Portas Lógicas
typ	NÃO	NÃO	0	45271*	151.22	7.97	5780
lento	NÃO	NÃO	0	112960*	134.39	7.97	5758
lento	NÃO	114	0	17931	41.05	8.16	6166
lento	NÃO	98	0	2047	8495	8.3	6392
lento	NÃO	96	0	1752	85.93	8.32	6449
lento	NÃO	90	0	135	77.22	8.37	6549
lento	NÃO	84	0	112	92.4	8.39	6664
lento	NÃO	78	0	3999	85.09	8.48	6961
lento	NÃO	62	0	1090	111.46	8.54	7247
lento	NÃO	50	0	16	147.86	8.75	7411
lento	NÃO	50	2000	48	191.33	8.56	7168
lento	NÃO	50	5000	16	147.96	8.77	7461
lento	SIM	50	5000	28	203.98	9.53	8536
lento	SIM	50	10000	4	243.98	9.64	8766
lento	SIM	50	11000	16	201.87	9.7	9008
lento	SIM	50	12000	1	201.01	9.75	9194
lento	SIM	50	13000	4	230.73	9.86	9407
lento	SIM	50	15000	4	221.7	9.97	9697
lento	SIM	50	18000	-815	200.97	10.37	10866

* Tempo de propagação do caminho crítico.

Fonte: Autores

A Tabela 2 demonstra alguns comportamentos importantes das ferramentas de síntese lógica. O primeiro fato é a diferença entre o uso das condições PVT da biblioteca, típico e pior caso. O pior caso resulta em um atraso de propagação que é quase 150% superior ao atraso de propagação resultante no caso típico. Em segundo lugar, a dissipação de potência estimada não necessariamente aumenta com restrições de tempo mais apertadas. A dissipação total de potência no caso

estudado pode ser aproximada pela equação (1), da potência dinâmica de um circuito digital.

$$P = \alpha \cdot C \cdot V_{DD}^{2} \cdot f \quad (1)$$

C é a capacitância total de saída de um nó do circuito, V_{DD} é a tensão de alimentação, *f* é a frequência eαé a atividade de chaveamento de um nó. Portanto, a potência é diretamente proporcional à frequência do circuito. Mas note que a ferramenta EDA pode encontrar diferentes estruturas de lógica para alcançar as restrições necessárias. Por isso, o circuito usado para a análise de dissipação de potência é diferente em cada síntese e a potência não necessariamente aumenta com especificações de frequência mais altas.

Nas sete últimas linhas da Tabela 2, é possível verificar que, com o uso do comando *adjust_path*, o comportamento crescente da estimativa de dissipação de potência desaparece. Nestes casos, a frequência utilizada na estimativa de potência é ao tempo definido apenas na coluna de restrições de atraso de propagação. Outro fato a ser percebido, é o aumento da área quando as estruturas de teste são adicionadas à síntese. As estruturas de teste, *scan chain,* substituem os *flip-flops* por *scan flip-flops* que possuem um multiplexador na entrada e apresentam uma área maior. No caso estudado, isto levou a um aumento em área de aproximadamente 8,5%.

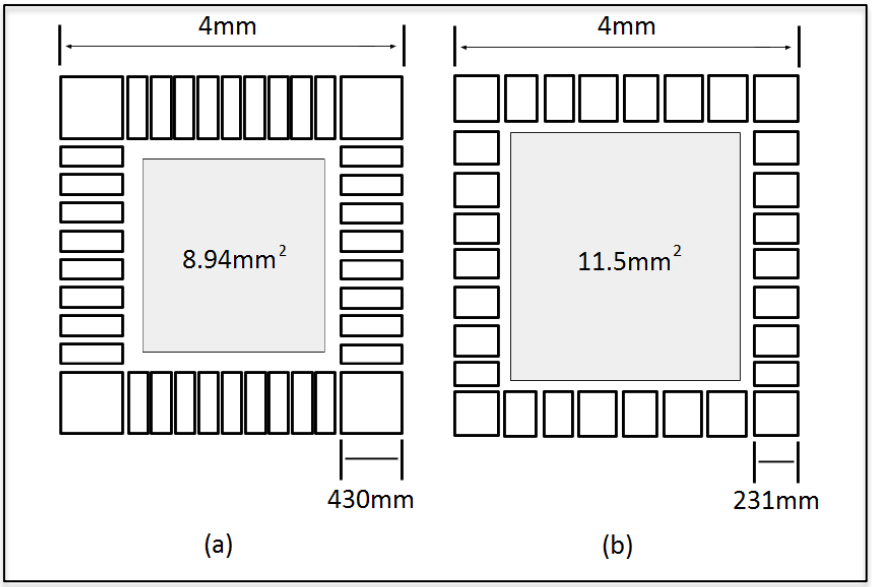
A métrica produto atraso potência, PDP (do inglês, *Power Delay Product*) é usada neste projeto para escolher as melhores restrições de projeto que atingem a frequência de relógio alvo de 20 MHz. Portanto, foi selecionada a síntese que oferece 50 ns de atraso de propagação e *adjust_path* de 12ns. Como resultado, a área estimada é 9.75 mm² com um total de 9.194 células lógicas.

Como o Risco possui uma descrição configurável, a escolha da quantidade de GPIOs leva em consideração a possibilidade de usar biblioteca com diferentes tipos de *pads*, estes *pads* são denominados *core_limited* ou *pad_limited* de acordo com a característica do projeto. Em um design *pad_limited*, os *pads* de IO têm um tamanho fixo de 110 x 430 µm. Por outro lado, a escolha de uma biblioteca de *pads core_limited*, resulta em *pads* de IO com um comprimento de 231 e largura variável, dependendo da funcionalidade do *pad*.

Assim, como ilustrado na Figura 2, se utilizados *pads* do tipo *pad_limited*, a área total do projeto, 16 mm², não é suficiente, pois a área disponível para as células lógicas e memórias fica abaixo da estimada na síntese lógica. Portanto, a escolha foi

utilizar uma configuração *core_limited* e disponibilizando no projeto 8 entradas e 8 saída de uso geral, GPIOs.

Figura 2: Análise de planejamento de planos: (a) limitado pelos *pads* (b) limitado pelo núcleo.

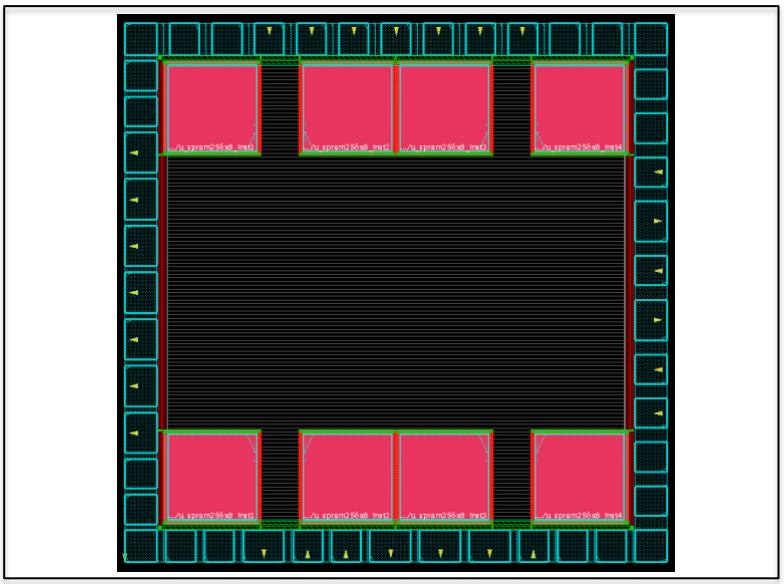


Fonte: Autores

Outra decisão tomada durante a etapa de *floorplanning* é a localização dos módulos de memória. Devido à limitação da *foundry*, existe apenas um módulo de memória disponível para uso neste projeto. Este módulo tem um tamanho de 256x8. Portanto, o projeto concatena oito módulos de memória, 4 memórias de 8 bits para resultar em uma memória de programa de 32 bits e outros 4 de 8 bits para criar a memória de dados. Como resultado, tem-se dois bancos de memórias de 256 x 32, um para dados e outro para programa, conectados ao processador Risco. Uma característica desse módulo disponibilizado é que os pinos de conexão estão localizados somente em dois lados perpendiculares da memória, formando um canto. Portanto, a opção utilizada foi colocar quatro módulos de memórias nos cantos da área do núcleo e os outros quatro no centro superior e inferior. Como resultado tem-se que todos os módulos de memórias possuem as faces de conexão facilmente acessadas pelos barramentos de dados, de endereço e de controle. Ao planejar a alocação dos *pads* do CI, levou-se em consideração que os *pads* localizados nas extremidades seriam prejudicados quanto a estratégia de conexão, em virtude do bloqueio de roteamento causado pelos módulos de memória. Deste modo os *pads* de alimentação foram alocados nos cantos do CI, pois os mesmos possuem conexão somente com o anel de alimentação não sendo prejudicados pelo

bloqueio de roteamento das memórias. O *floorplanning* resultante pode ser observado na Figura 3.

Figura 3: Planejamento do posicionamento dos módulos de memória dentro do núcleo do chip



Fonte: Autores

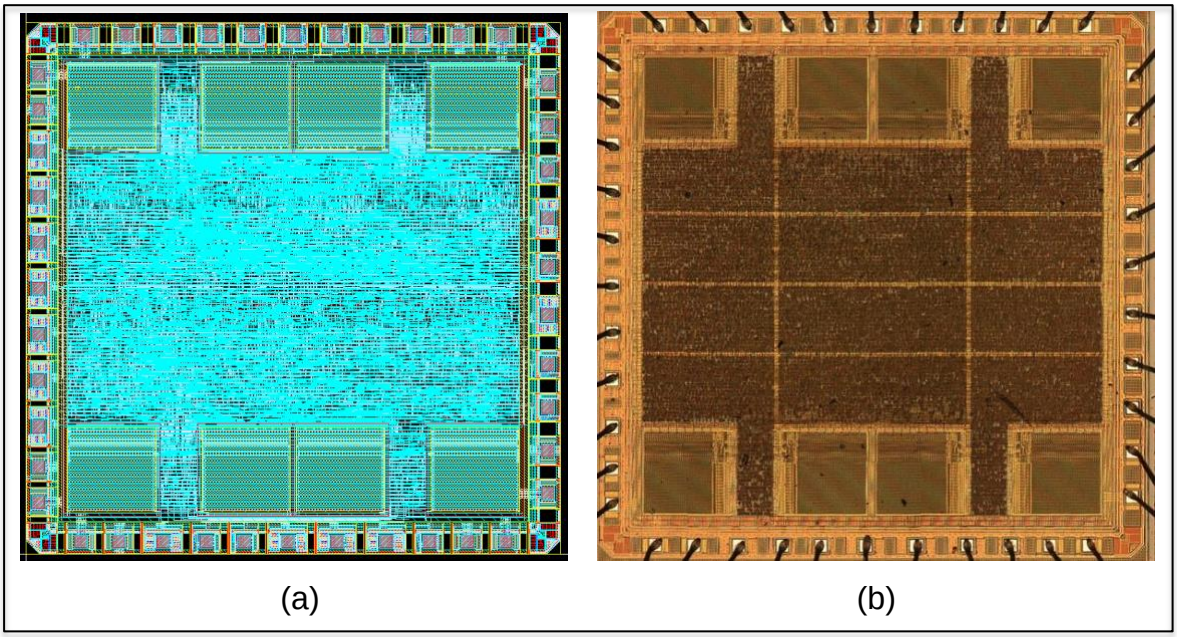
O projeto foi implementado com 8 pares de *pads* de alimentação, 4 pares para alimentação do núcleo e outros 4 pares para alimentar os anéis de *pads*. Cada *pad* de alimentação pode suprir 50 µA. Esta configuração dos *pads* de alimentação pode fornecer, portanto, até 660 mW para o núcleo do processador e mais 660 mW para os circuitos conectados ao processador. Ao se tratar da grade de alimentação interna, esta possui uma largura de trilha de 8 µm. Para o dimensionamento dos reforços na grade de alimentação, levou-se em consideração que nesta tecnologia a capacidade de corrente das conexões é de 1mA/µm, e a resistência de folha é equivalente à 110 mΩ/□ (miliohms por quadrado). Levando-se em consideração que o projeto possui reforços de alimentação a cada 1 mm, a queda de tensão máxima para esta configuração, considerando a capacidade máxima de corrente, é de cerca de 60 mV, ou seja, aproximadamente 2% da tensão de alimentação. Observe que este é um cenário muito pessimista pois esta corrente fornece uma potência total de 26,4 mW para cada linha de células. O circuito tem cerca de 150 linhas, portanto, o *power planning* executado atende aos requisitos do projeto, uma vez que a potência estimada na síntese lógica foi de 201 mW.

Uma vez que um processador é usado para várias aplicações, sua dissipação de potência depende do código que está sendo executado nele. Portanto, para fazer

a estimativa final de potência, a escolha foi usar uma análise estática com uma probabilidade de chaveamento de 50 % nas entradas. Esta análise é conservadora devido à alta probabilidade utilizada.

O leiaute final do circuito foi verificado no ambiente Virtuoso a partir do qual gerou-se o arquivo GDSII enviado para a *foundry* CEITEC. A Tabela 3 mostra as especificações finais alcançadas no projeto. O resultado final do projeto verificado e corrigido na ferramenta Virtuoso pode ser observado na Figura 4a e o circuito integrado resultante, após a fabricação pela empresa Ceitec S/A, pode ser observado na Figura 4b.

Figura 4: Projeto final do microprocessador Risco: (a) desenho final gerado pela ferramenta Virtuoso; (b) fotografia do circuito integrado fabricado



(a) (b)

Fonte: Autores

Tabela 3: Especificações resultantes do projeto do microprocessador Risco.

Nome do projeto	Risco ASIC	
Especificações	Área do núcleo	10.77 mm²
	Área total	15.18 mm
	Frequência	20 MHz
	Memória de instruções	4 módulos de 256 x 8
	Memória de dados	4 módulos de 256 x 8
	Total de pinos	43
	Pinos de alimentação	16
	Número de células	9.247
	Número de portas lógicas	42.349
	Potência estimada	254 mW

Fonte: Autores

5. PLANEJAMENTO DA ROTINA DE TESTE

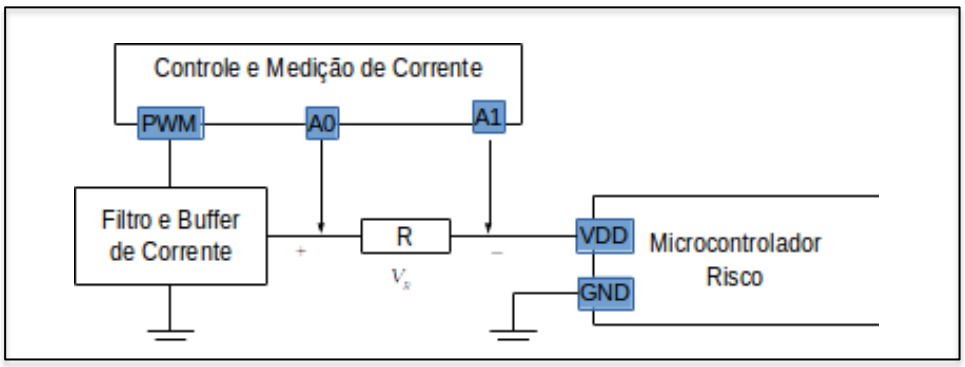
O teste do processador Risco está planejado para ser realizado em duas etapas. A primeira delas é a verificação estrutural, que envolve avaliar o resultado dos aspectos de fabricação e de encapsulamento do circuito integrado. A segunda etapa é a verificação do funcionamento da arquitetura interna. O teste de condições de funcionamento serve para verificar se não existem *pads* em curto-circuito, através da amarração de fio entre o pino do encapsulamento e o *chip*, e também se não existe curto-circuito entre os pinos de alimentação (VDD e GND) para o núcleo e para os *pads*. Além disso, é necessário verificar se o chip suporta a tensão de alimentação de 3,3 V, teste chamado de *bring-up*.

Para realizar o *bring-up* dos chips, de forma monitorada e controlada via computador, planejou-se um circuito regulador de tensão controlado digitalmente,

integrado a um sensor de corrente. Para isso, configurou-se uma plataforma microcontrolada para gerar um sinal PWM (*Pulse Width Modulation*) com amplitude de 0 a 5 V, utilizando um microcontrolador comercial Arduino Uno. O sinal PWM então é filtrado e inserido em um reforçador de corrente (amplificador operacional configurado como buffer). Este sinal de tensão é utilizado para energizar o circuito integrado.

Desta forma, é possível variar-se a razão de ciclo do PWM e controlar a tensão gerada para o circuito integrado a ser testado. Como meta proposta, pretende-se limitar a corrente elétrica em 200 mA e limitar a tensão elétrica em 3,6 V. Os pontos de teste A0 e A1 são conectados à duas entradas de conversores A/D do microcontrolador para medir-se a corrente elétrica de forma indireta, por meio do cálculo de V_R/R no resistor de 8,5 Ω. O diagrama de ligação da fonte controlada de tensão é observado na Figura 5.

Figura 5: Diagrama de blocos do circuito de fonte de tensão regulada e medição de corrente indireta



Fonte: Autores

Este circuito de controle de tensão de alimentação também é usado para monitorar a corrente elétrica fornecida para o circuito integrado. Também será utilizado nos testes posteriores para executar as rotinas de teste funcional do microprocessador variando-se a tensão de alimentação.

6. CONCLUSÕES

Um processador RISC de 32 bits, o Risco, foi inserido com sucesso no fluxo de projeto de CI, utilizando ferramentas Cadence e a tecnologia 0,6 µm da *foundry*

X-Fab. A característica reconfigurável do projeto VHDL em questão torna mais fácil a adaptação do projeto a requisitos específicos. O caso estudado foi altamente afetado por limitações na síntese física. Devido à característica reconfigurável de sua descrição, a iteração entre a síntese física e a definição da arquitetura foi rápida e fácil. No futuro, algumas soluções de verificações feitas após a síntese devem ser melhoradas, por exemplo, a estimativa de potência. Os trabalhos atuais estão sendo desenvolvidos para a criação do ambiente de teste e verificações de desempenho do circuito resultante.

REFERÊNCIAS

INTEL (2017). **The Evolution of a Revolution**, Intel Corporation. Disponível em: <http://download.intel.com/pressroom/kits/IntelProcessorHistory.pdf>. Acesso em: 27 ago. 2017.

JUNQUEIRA; A. A. (1993). **Risco – microprocessador risc cmos de 32 bits**, Dissertação, Instituto de Informática da Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, 1993.

RABAEY, J. M.; CHANDRAKASAN, A. P.; NIKOLIC, B. (2002), **Digital integrated circuits.** Prentice Hall Englewood Cliffs, 2002, vol. 2.