Simulação Quântica em VHDL: Um Estudo de Caso Baseado no Algoritmo de Deutsch*

E.R. Monteiro[1]**, D.P. Jaccottet[1], R.H.S. Reiser[2], E.A.C. Costa[2], and M.L. Pilla[3]

[1] Ciência da Computação - Centro Politécnico, Universidade Católica de Pelotas
Rua Felix da Cunha 412, 96010-000 Pelotas, Brasil
[2] Programa de Pós-Graduação em Informática, Universidade Católica de Pelotas
[3] Departamento de Informática – IFM, Universidade Federal de Pelotas
Campus Universitário s/n, Capão do Leão, Brasil
{eduardam,diegopj,reiser,ecosta}@ucpel.tche.br,
pilla@ufpel.edu.br

Abstract. This work presents a methodology for hardware simulation of quantum algorithms which applies digital systems and uses the established standards of the classic circuits. The methods are implemented and synthesized in VHDL. Thus, for the verification of the generated descriptions, the resultant circuit is executed in FPGA. This work considers the quantum circuit models as a universal language for modeling quantum algorithms. Methods for description of states and the set of universal quantum gates have been implemented. In addition, the main operators related to measurement and control operations associated to quantum circuits are also considered. For validation and tests, this work considers a case study regarding the Deutsch's algorithm, a special case Deutsch-Jozsa algorithm which is known as the first true quantum algorithm. The results include an error analysis of simulated quantum circuits.

Keywords: Architecture of Computing Systems; Quantum Simulation; Quantum Algorithms, Deutsch's Algorithm, VHDL

1 Introdução

Apesar das restrições tecnológicas que ainda tornam indisponíveis os computadores quânticos, a computação quântica tem desenvolvido e testado novos algoritmos através de simulação [12,15,5,8] via *software* e via *hardware*, possibilitando análise do paralelismo quântico e das propriedades como interferência e superposição de estados. Algoritmos quânticos oferecem maior *speedup* na execução de tarefas específicas, tais como a encriptação, fatoração, amplificação, busca em banco de dados e correção do erro quântico. Muitos são os estudos direcionados à simulação por *software*, mas os resultados são obtidos de forma lenta e custosa, consumindo recursos computacionais que incluem a programação paralela e/ou distribuída e, frequentemente, baseados em ambientes de computação em grade interconectados por rede de alta velocidade.

Com o advento da nanotecnologia, novas condições são ofertadas para uma futura implementação física dos computadores quânticos. Recentemente, pesquisadores

* Projeto parcialmente financiado pelo CNPq, referente aos processos: Edital Universal(476933/2007-2) e Apoio Técnico (502999/2008-0).

** Bolsista de Iniciação Científica PIBIC/CNPq

da Universidade de Yale criaram o primeiro processador quântico em estado sólido, consistindo em um chip para processamento simples de algoritmos básicos, mas cuja execução ocorre em um único dispositivo eletrônico, semelhante aos processadores atuais. Tais fatos são motivadores, pois simulações paralelas em *hardware* clássico podem incentivar tanto o desenvolvimento de novos algoritmos quânticos como sua aplicação no processamento da informação com base nestas novas tecnologias

Neste contexto, considera-se a continuidade no trabalho introduzido em [7], com a construção do protótipo da biblioteca *qExVHDL*. Busca-se o desenvolvimento de métodos de descrição baseados nas HDLs (*Hardware Description Languages*) para manipulação de dados e de controle quântico. Com base nos padrões clássicos estabelecidos, o principal objetivo é desenvolver componentes básicos no nível de *hardware* para simular o comportamento de circuitos quânticos. Uma implementação inicial em dispositivos reconfiguráveis, FPGA (*Field Programmable Gate Array*) é apresentada, permitindo a exploração do paralelismo intrínseco do *hardware*.

Na modelagem e na implementação dos métodos para manipulação de dados e controle quânticos, este trabalho aplica a linguagem VHDL, significando VHSIC (*Very High Speed Integrated Circuits*) *Hardware Description Language*, padrão estabelecido pela IEEE para realização de simulações e sínteses com portabilidade de código, permitindo a reutilização de componentes em projetos de *hardware* [10]. Os componentes quânticos em desenvolvimento constituem uma base para a especificação e simulação de componentes mais complexos.

Na simulação dos bits quânticos, os valores reais que compõem os coeficientes complexos foram obtidos utilizando um sistema de ponto flutuante com precisão de 32 bits. Para verificação das descrições geradas, o trabalho considera o modelo de circuitos quânticos [11,9], consistindo numa linguagem universal para modelagem de algoritmos. Assim, foram implementados métodos para descrição de estados, do conjunto de portas universais e principais operadores associados a circuitos quânticos, os quais podem ser automaticamente gerados através da aplicação da ferramenta *Quantum-RAMAGT*. Também foi desenvolvido um estudo de caso, em nível de linguagem de descrição de *hardware*, para o circuito quântico do algoritmo de Deutsch [14].

O artigo continua com uma breve descrição de trabalhos relacionados à simulação quântica na Seção 2. O modelo de circuito quânticos é considerado na Seção 3. Seguem-se as etapas do algoritmo de Deutsch, descritas na Seção 4. As características básicas da modelagem da computação quântica em VHDL são apresentadas na Seção 5, incluindo a simulação de bits e operadores quânticos. O estudo de caso, com a aplicação desta metodologia na descrição do algoritmo de Deutsch é introduzida na Seção 5.2. A Seção 5.3 descreve os resultados alcançados na implementação e execução dos métodos, incluindo a análise do erro computacional. Por fim, tem-se as conclusões e continuidade do trabalho na Seção 6.

2 Trabalhos Relacionados

A simulação é uma metodologia amplamente usada para o desenvolvimento e depuração de circuitos quânticos, a qual utiliza massivamente operações de ponto flutuante. Neste contexto, vários trabalhos científicos tem contribuído para a implementação de algoritmos quânticos e sua correspondente execução em dispositivos lógicos programáveis, facultando sua simulação em *hardware* via construção de circuitos integrados.

Em *FPGA Emulation of Quantum Circuits* [5], considera-se o modelo de circuitos quânticos para descrever os principais algoritmos e as correspondentes analogias com o modelo de circuitos digitais. O projeto introduz um emulador de algoritmos quânticos em FPGA, concentrando seus experimentos sobre novas técnicas de modelagem de circuitos quânticos que incluem a manipulação do emaranhamento e a computação probabilística, considerando as questões críticas de precisão.

O projeto *Evolving quantum circuits and an FPGA-based Quantum Computing Emulator* [8], desenvolve metodologias e ferramentas com interesse na síntese lógica para circuitos quânticos e no *design* de sistemas quânticos, introduzindo o emulador de circuitos quânticos FPGA-*based Evolvable Quantum Hardware*.

Em *Massively parallel quantum computer simulator* [12], tem-se um componente de *software*, portátil, para simulação de computadores quânticos universais com uso massivo de computadores paralelos. A simulação em *software* é ilustrada pela execução de vários algoritmos quânticos em diferentes arquiteturas de computadores (IBM BlueGene/L, IBM Regatta p690, Hitachi SR11000/J1, Cray X1E, SGI Altix 3700) e *clusters* de computadores rodando em Windows XP.

No artigo *Synthesis of Quantum Logic Circuits* [15], é analisada a eficiência lógica de circuitos quânticos que desempenham ambas funcionalidades: (i) implementação de computações quânticas genéricas e (ii) inicialização de registradores quânticos.

3 Modelo de Circuitos Quânticos

O modelo de circuitos quânticos constitui-se em uma linguagem universal para descrição de computadores quânticos [2], utilizado em aplicações práticas para análise dos atuais algoritmos quânticos [9,11]. Seguem-se algumas das principais notações e convenções usadas nos circuitos quânticos (veja Figura 1):

1. *entrada*, indicada por um q-bit ou um registrador de q-bits;
2. *linhas horizontais*, representando a evolução temporal (da esquerda para a direita) do fluxo de dados (q-bits) entre duas portas lógicas;
3. *linhas verticais*, mostrando a atuação síncrona em dois ou mais q-bits;
4. *controle*, representado por um círculo no q-bit controle;
5. *portas lógicas*, indicadas por caixas com uma letra de identificação;
6. *medida* de um q-bit, geralmente ocorrendo na saída do circuito e alterando o estado do q-bit, o qual retorna 0 e 1 com uma distribuição de probabilidade associada;
7. *estados* φ_i como transformações em instantes de tempo (t_i).

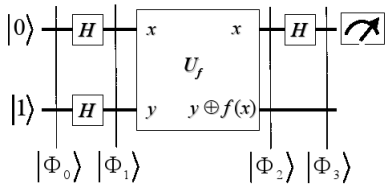

Figura 1. Circuito de Deutsch.

No computador quântico, um q-bit de informação além de poder armazenar 0 ou 1 também pode estar em superposição e armazenar ambos 0 e 1. Por conseqüência, a capacidade de informação em um registrador de n q-bits corresponde a 2^n valores por vez.

Ou seja, se um registrador quântico de n q-bits pode armazenar 2^n números ao mesmo tempo, um computador quântico deverá processar toda essa entrada de dados, simultaneamente, caracterizando o paralelismo quântico. A transformação dos possíveis estados de registradores quânticos pode ser modelada por operadores unitários, referidos como portas quânticas (dispositivos que executam uma operação unitária fixada, sobre q-bits selecionados, em período determinado no tempo).

Circuitos quânticos podem ser definidos considerando-se apenas conjuntos finitos de portas quânticas de um q-bit denominadas *portas elementares*, neste caso, constituem conjuntos de *portas quânticas universais*.

Um q-bit é interpretado por um vetor do espaço de Hilbert $\mathcal{H}_2$, definido como um espaço vetorial complexo munido do produto interno $a \cdot b = \sum_i a_i^* \cdot b_i$ [4]. Um operador unitário U é interpretado como uma matriz unitária ($UU^* = U^*U = I$)

Além de portas universais, um circuito quântico pode conter outras operações que não são unitárias, como, por exemplo, operações de medida (neste caso, interpretadas por projeções do vetor correspondente ao q-bit sobre um par de subespaços ortogonais) e ainda operações de controle. Cada porta de um q-bit descreve informação de um sistema quântico de uma dimensão, indicada na notação de Dirac pela expressão $|\psi\rangle = a|0\rangle + b|1\rangle$, parametrizado pelos números complexos a e b, os quais satisfazem a condição de normalidade $|a|^2 + |b|^2 = 1$. Sendo assim, qualquer matriz unitária de dimensão 2 representa uma porta lógica quântica sobre um q-bit [11], e as sucessivas aplicações descrevem transformações sobre um sistema quântico unidimensional. A seguir, consideram-se as operações quânticas sobre um q-bit definidas por Hadamard (H), Fase (P), e porta $\frac{\pi}{8}$ (T), dadas pelas matrizes:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, P = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, T = \begin{bmatrix} 1 & 0 \\ 0 & \exp(i\frac{\pi}{4}) \end{bmatrix}$$

A porta H, quando aplicada no estado clássico $|0\rangle$, gera como resultado uma superposição, indicada por $|\phi\rangle = H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. A porta P quando aplicada sobre o estado $|\phi\rangle$ resulta em $P|\phi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)$, introduzindo uma fase relativa igual a $\exp(i\frac{\pi}{2})$. Quando da operação de medida sobre $P|\psi\rangle$, resultam as mesmas probabilidades de obter os estados clássicos $|0\rangle$ ou $|1\rangle$ que ocorrem se $|\psi\rangle$ for medido. Assim ocorre com os estados de entrada e saída na aplicação da porta *Hadamard*.

Considerando-se a porta T e um estado genérico $|\psi\rangle = a|0\rangle + b|1\rangle$, tem-se que $T|\psi\rangle = a|0\rangle + exp(i\pi/4)b|1\rangle$. A exemplo do que ocorre com a porta P, as probabilidades de se obter o estado $|0\rangle$ ou o estado $|1\rangle$ são as mesmas tanto para o estado genérico $|\psi\rangle$ quanto para o estado modificado $T|\psi\rangle$.

Devido à interpretação geométrica dos q-bits por pontos sobre a superfície da esfera unitária (Esfera de Bloch), portas quânticas podem ser pensadas como rotações. Assim, rotações em torno de cada eixo $\overrightarrow{OX}$, $\overrightarrow{OY}$, $\overrightarrow{OZ}$ podem ser geradas pelas correspondentes matrizes de Pauli X, Y e Z:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

A representação de portas lógicas quânticas atuando sobre n *q-bits* envolve matrizes unitárias de dimensão $2^n \times 2^n$. A aplicação simultânea de uma porta U sobre o q-bit $|\phi\rangle$

[4] Operação * é a composição das operações matemáticas de transposição e conjugação

e de outra V sobre o q-bit $|\psi\rangle$ é representada pelo produto tensorial $(U \otimes V)(|\phi\otimes\rangle|\psi\rangle)$ entre as correspondentes matrizes [9]. Além destas operações, consideram-se ainda uma classe de portas lógicas que atuam sobre dois ou mais *q-bits* sem serem fatoradas como produtos tensoriais de matrizes 2×2. Casos particulares de operações lógicas atuantes em dois q-bits são as portas *controladas* e os *swaps* (associadas à troca de valores) [3,4].

CNOT, como é indicado o operador não-controlado, recebe dois q-bits na sua entrada, um q-bit de controle e um q-bit alvo, se o q-bit de controle receber $|1\rangle$, o estado do q-bit alvo é trocado, senão o q-bit alvo permanece inalterado. Genericamente, se U é um operador unitário de um q-bit, uma operação *U-controlada* é uma operação a dois q-bits, sendo um o controle e o outro, o alvo da ação do operador U.

Tem-se que $\mathcal{U} = \{H, P, CNOT, T\}$ é um conjunto universal de portas lógicas. Assim, toda aproximação de operações unitárias arbitrárias pode ser obtida por composições definidas sobre operadores de $\mathcal{U}$.

4 Algoritmo de Deutsch

Utilizando o algoritmo de Deutsch, é possível verificar se uma função binária é constante $(f(0) = f(1))$ ou equilibrada $(f(0) \neq f(1))$, realizando apenas uma operação com a função. Uma tentativa de analogia com alguma operação clássica seria como saber se uma moeda possui de um lado uma cara e do outro um valor (coroa), com apenas uma observação. Ou seja, é possível observar ambos os lados da moeda de uma só vez. O circuito quântico que descreve este algoritmo está ilustrado na Figura 1. Neste podemos ver que a o estados de entrada dos q-bits pode ser descrito como na Eq. (1).

$$|\Phi_0\rangle = |0\rangle \oplus |1\rangle = |01\rangle \tag{1}$$

O passo seguinte do algoritmo é aplicar a porta Hadamard em ambos q-bits. O estado $|\Phi_1\rangle = H_b H_a |\Phi_0\rangle = H_a|0\rangle H_b|1\rangle$ será então descrito pela Eq. (2).

$$|\Phi_1\rangle = \frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) \ = \frac{1}{2}[|0\rangle(|0\rangle - |1\rangle) + |1\rangle(|0\rangle - |1\rangle)] \tag{2}$$

A operação unitária U_f efetua a transformação $|x, y\rangle \rightarrow |x, y \oplus f(x)\rangle$. Considerando-se $|x, y\rangle = \frac{1}{\sqrt{2}}|x\rangle[|0\rangle - |1\rangle]$ como estado inicial e a seguintes condições:

$$\begin{aligned} &|0 \oplus f(x)\rangle = |0\rangle \text{ e } |1 \oplus f(x)\rangle = |1\rangle \ \text{ se} f(x) = 0, \ e \\ &|0 \oplus f(x)\rangle = |1\rangle \text{ e } |1 \oplus f(x)\rangle = |0\rangle \ \text{ se } f(x) = 1. \end{aligned}$$

tem-se que $U_f(\frac{1}{\sqrt{2}}|x\rangle[|0\rangle - |1\rangle])$ é descrito pela Eq. (3):

$$U_f(\frac{1}{\sqrt{2}}|x\rangle[|0\rangle - |1\rangle]) = \frac{1}{\sqrt{2}}|x\rangle[|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle] = (-1)^{f(x)}\frac{1}{\sqrt{2}}|x\rangle[|0\rangle - |1\rangle] \tag{3}$$

e, portanto, após a aplicação de U_f o estado do sistema será o descrito pela Eq.(4):

$$|\Phi_2\rangle = U_f|\Phi_1\rangle = \frac{1}{2}[(-1)^{f(0)}|0\rangle(|0\rangle - |1\rangle) + (-1)^{f(1)}|1\rangle(|0\rangle - |1\rangle)], \tag{4}$$

Verifica-se ainda que o estado Φ_2 pode ser reescrito de acordo com a Eq. (5):

$$|\Phi_2\rangle = \begin{cases} \pm\frac{1}{2}[(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)] & \text{se } f(0) = f(1); \\ \pm\frac{1}{2}[(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)] & \text{se } f(0) \neq f(1). \end{cases} \quad (5)$$

Assim, o próximo passo é aplicar a chave Hadamard ao primeiro q-bit. Lembrando que: $H(|0\rangle + |1\rangle) = |0\rangle$ e $H(|0\rangle - |1\rangle) = |1\rangle$, tem-se que o estado final do sistema será descrito pela Eq.(6):

$$|\Phi_3\rangle = \begin{cases} \pm\frac{1}{\sqrt{2}}[(|0\rangle - |1\rangle)] & \text{se } f(0) = f(1); \\ \pm\frac{1}{\sqrt{2}}[(|1\rangle - |1\rangle)] & \text{se } f(0) \neq f(1). \end{cases} \quad (6)$$

Finalmente, ao medirmos o estado do primeiro q-bit saberemos se a função é constante ($|0\rangle$) ou balanceada ($|1\rangle$). Resumindo, é necessário aplicar a função somente uma vez, e medir um único q-bit.

5 Modelagem de Computação Quântica usando VHDL

A simulação de circuitos quânticos em *hardware* integra conceitos e tecnologias da física quântica e da física clássica, objetivando uma simulação computacional tão flexível e eficiente quanto a simulação em *software*. Como a maioria dos algoritmos exige uma amostra exponencial de recursos quando da simulação via tecnologias clássicas, o gerenciamento de recursos torna-se uma estratégia importante no projeto.

A extensão da linguagem VHDL para aplicação na descrição do comportamento e a estrutura de circuitos quânticos inclui também o desenvolvimento de bibliotecas de portas quânticas em VHDL. Neste trabalho, os circuitos quânticos são construídos a partir de métodos, dados e componentes providos pela linguagem. Aplicam-se tecnologias clássicas na construção de componentes fundamentais dos circuitos quânticos e nas regras que governam as computações quânticas.

A descrição do circuito pode ser verificada ou por simulação em *software*, geralmente utilizada para alcançar performance e tornar o processo de validação mais prático. Optou-se no projeto, pela simulação em *hardware*, visando uma análise do comportamento do circuito a partir da síntese em FPGA.

5.1 Simulação de bits quânticos

Na especificação de um bit quântico, q-bit, $|\psi\rangle = a|0\rangle + b|1\rangle$, consideram-se pares de números reais: $|\psi\rangle = (\alpha_1, \alpha_2)|0\rangle + (\beta_1, \beta_2)|1\rangle$, onde α_1 e α_2 correspondem à parte real e imaginária do complexo a, coeficiente da primeira componente da base, e β_1 e β_2 correspondem à parte real e imaginária do complexo b, coeficiente da segunda componente da base.

Esta representação de quantidades contínuas por precisão finita introduz erro nos dados de entrada do sistema modelado. Na implementação dos estados de um ou mais bits quânticos, os coeficientes são arredondados de acordo com o sistema de ponto flutuante (*IEEE Standard for Floating-Point Arithmetic*, IEEE 754 single-precision). Assim, a implementação dos valores reais $\alpha_1, \alpha_2, \beta_1, \beta_2$ pode ser obtida utilizando um sistema de ponto flutuante, com precisão de 32 bits, apresentando 1 bit de sinal, expoentes de 8 bits e mantissas de 23 bits. Os coeficientes a e b de $|\psi\rangle$ foram obtidos considerando registradores de estados quânticos com quatro partes, seguindo a arquitetura na Figura 2.

Figura 2. Arquitetura do q-bit na Biblioteca **qExVHDL**.

5.2 Aplicando VHDL na Descrição do Algoritmo de Deutsch

A partir da descrição em VHDL do circuito correspondendo ao algoritmo de Deutsch, pela aplicação da ferramenta ISE desenvolvida pela empresa Xilinx [17], pode-se estabelecer o fluxo de projeto para a correspondente prototipação em FPGA. Ou seja, a metodologia proposta considera a geração automática de descrições em VHDL, a nível comportamental, na etapa de implementação das portas e estados do corrente circuito. Posteriormente, o código resultante é sintetizado em FPGA, veja Figura 3.

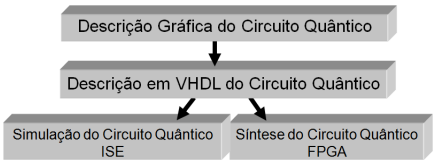

Figura 3. Método Gerador de Código.

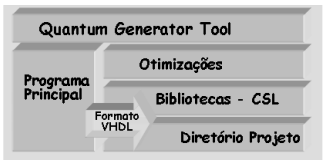

Figura 4. Ferramenta *Quantum-RAMAGT*.

Para geração automática de código VHDL, customizou-se a ferramenta RAMAGT (*Radix-*2^m *Array Multipliers Automatic Generation Tool*), proposta em [1], sendo desenvolvida pela integração de duas linguagens: (i) a linguagem C++, responsável pela construção da interface gráfica; e (ii) a linguagem CSL (*C Scripting Language*) [6], cuja sintaxe semelhante às linguagens de programação clássicas (C e C++) facilita a programação e a manipulação dos dados. O código desenvolvido em CSL é interpretado gerando a descrição em VHDL referente aos parâmetros propostos pelo usuário e correspondendo a aplicação em desenvolvimento.

A arquitetura da ferramenta denominada *Quantum-RAMAGT*, na Figura 4, está brevemente descrita em quatro etapas caracterizadas pela suas funcionalidades: (i) interface do programa, disponibilizando ao usuário a seleção do tipo de descrição em VHDL; (ii) a seleção das otimizações disponíveis, que correspondem aos arquivos CSL que geram código VHDL otimizado; (iii) seleção dos parâmetros que definem a precisão dos dados, a dimensão do sistema e o número de operadores para a aplicação corrente; (vi) finalização do código VHDL para a aplicação modelada e geração do arquivo descritor do projeto.

A descrição do circuito ocorre através de transformação envolvendo multiplicadores e somadores e componentes de ponto flutuante disponibilizados pela ferramenta Xilinx, que podem então ser sintetizados ao FPGA diretamente. As otimizações desenvolvidas se concentraram na simplificação dos coeficientes, no tratamento de números complexos, na modelagem de acumulador para armazenamento de somas, na aplicação de componentes conversores da precisão dos dados e na possibilidade de redução da precisão (16 e 8 bits). A Figura 5 apresenta as interfaces da ferramenta *Quantum-RAMAGT* para a geração do código VHDL para um sistema bidimensional correspondendo à descrição em VHDL do algoritmo de Deutsch. Para este estudo de caso, o programa gerou operadores específicos: o operador unitário U_f e a porta Hadamard (H), cujos coeficientes estão indicados na Figura. 6, indexados pela expressão `Op0` (em 32 bits).

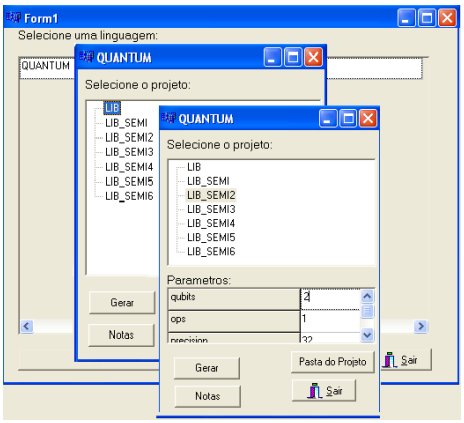

Figura 5. Gerador de Código.

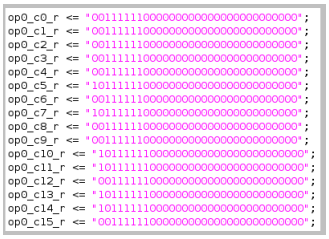

Figura 6. Porta H dada por Op0 (32 bits).

5.3 Resultados

A validação da descrição em VHDL do circuito correspondente ao algoritmo de Deutsch foi executada na placa FPGA desenvolvida pela empresa *Xilinx*: Placa $XC2VP30$, Família *Virtex-II PRO*, Pacote $FF896$, com frequência de operação igual a 151.871 Mhz e uma área de 12.555 elementos lógicos, aproximadamente 42% da àrea total disponibilizada (30.000 elementos lógicos). Pelas otimizações desenvolvidas na ferramenta *Quantum-RAMAGT*, pode-se observar que tais resultados se mantiveram relativamente iguais ao estudo de caso abordado em [7]. Isto representa um ganho significativo considerando que o circuito correspondente ao algoritmo de Deutsch modela um sistema bidimensional e com maior número de operadores que o circuito que implementa o interferômetro de Mach-Zender. Para validação dos resultados utilizou-se a interface gráfica disponibilizada pela ferramenta *Chipscope*, a qual possibilita a manipulação de entrada e visualização dos resultados, apresentada na Figura 7.

Na análise do erro na execução da porta *Hadamard*, considera-se a expressão para o número real $\frac{\sqrt{2}}{2}$: $(0,707106)_{10} = (00111111001101010000010011100110)_2$, gerada pela configuração do conversor *IEEE-754 Floating-Point Conversion* customizado afim de manipular os dados binário em ponto flutuante, padrão da IEEE-754 desenvolvido em [16]. Neste caso, o erro absoluto e relativo podem ser obtidos pelas expressões:
$E_A \cong |0,707106 - 0,7071059942245483| = 0,5 * 10^{-8}$, $E_R \cong 0,82 * 10^{-8}$,

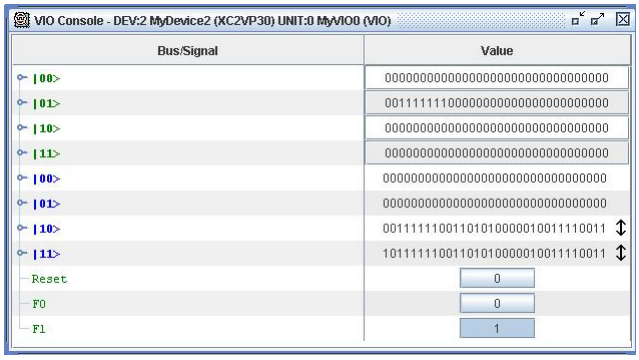

Figura 7. Interface Gráfica da Ferramenta *Chipscop*e.

respectivamente, comprovando-se que as aproximações estão dentro do limite de erro ($E_R \leq 0,5 * 10^{-5}$). Tem-se interpretação análoga para a porta *Phase*.

Dentre as portas unitárias que compõem o conjunto $\mathcal{U}$ na Seção 3, justifica-se a análise de erro referente a porta Hadamard, cuja execução gera manipulação de dados de ponto flutuante incluindo operações de somas e multiplicações sobre coeficientes complexos. A Tabela 1 resume uma comparação do erro para a porta *Hadamard* considerando as simulações descritas em [5] (usando ponto fixo e precisão de 16 bits).

Tabela 1. Tabela Comparativa de Erros

Porta Hadamard	*Erro Absoluto*
qExVHDL	$0,5 * 10^{-8}$
FPGA Emulator [5]	$3,05 * 10^{-5}$

Por exemplo, o estado inicial $|\phi_0\rangle = 1|0\rangle + 0|1\rangle$, tem $\alpha_0 = (1, 0)$ como parte real do primeiro coeficiente. Na estrutura de ponto flutuante, considerando-se a base binária, o coeficiente $(\alpha_1)_2 = +m * b^{+e}$ tem expressão binária da forma:

$$\underbrace{0}_{I}\ \underbrace{0}_{II}\ \underbrace{1111111}_{III}\ \underbrace{10000000000000000000000}_{IV}$$

onde (I) indica o sinal do número real; (II) indica o sinal do expoente, (III) indica o expoente e; e (IV) indica a expressão binária da mantissa m.

6 Conclusão

Neste trabalho, apresentam-se os resultados obtidos com simulação em *hardware* visando uma análise do paralelismo quântico, baseada no estudo de casos de algoritmos básicos da computação quântica. Para tal, considera-se uma metodologia que provê a geração automática de código e sua prototipação em dispositivos reconfiguráveis. Com base no modelo de circuitos quânticos, esta metodologia é aplicada ao circuito bidimensional referente ao algoritmo de Deutsch, obtendo sua implementação em VHDL e correspondente síntese em FPGA. A ferramenta *Quantum-RAMAGT* foi desenvolvida para prover a geração automática do código VHDL, disponibilizando uma biblioteca de otimizações. Estas otimizações são referentes à simplificação dos coeficientes complexos e visam a redução na utilização de recursos computacionais gerados pelo aumento

exponencial referente à dimensão do circuito. Neste estudo de caso, implementam-se os operadores U_f e a porta Hadamard (H), na precisão de 32 bits. Em continuidade, estuda-se a aplicação da metodologia aos algoritmos de Grover e Shor. A análise da metodologia da programação quântica em sistemas digitais incluindo o processamento de sinais é outro grande desafio, que visa o desenvolvimento de aplicações em sistemas de comunicação, para aplicações específicas de circuitos integrados na prototipação em ASIC (*Application-Specific Integrated Circuit*).

Agradecimentos: Este trabalho foi parcialmente financiado pelo CNPq, Edital Universal e de Apoio Técnico, Processos 476933/2007-2 e 502999/2008-0.

Referências

1. Almeida, S.M., Costa, E., Jacoottet, D.P., Pieper, L.Z., Roschild, J.: RAMAGT-Radix-2^m Array Multipliers Automatic Generation Tool. XXIII South Symp. on Microelectronics (2008)
2. Barenco, A., Bennett, C. H., Cleve, R., DiVincenzo, D. P., Margolus, N. H., Shor, P. W., Sleator, T., Smolin, J. A., Weinfurter, H.: Elementary gates for quantum computation. Physical Review A. (1195) 3457–3467
3. Bettelli, S., Catarco, T., Serafini, L.: Toward an architecture for quantum programming. The European Physical Journal D - Atomic, Molecular and Optical Physics. (2003)
4. Emberson, P.: Quantum Algorithm Designe. 4th Year MMath Research Project. (2002) http://www-users.cs.york.ac.uk/~sok/QAD/
5. Khalid, A. U., Zilic, Z., Radecka, K.: FPGA Emulation of Quantum Circuits. ICCD '04: Proceedings of the IEEE International Conference on Computer Design. (2004) 310–315
6. Koch, P.: C Scripting Language - Reference Manual V. 4.4.0. (2002) http://csl.sourceforge.net/csldoc/index.htm
7. Monteiro, E.R., Jaccottet, D.P., Reiser, R.H.S., Costa, A.C.R., Costa, E.A.C., Pilla, M.L.: Aplicação da Biblioteca qExVHDL na Descrição do Interferômetro de Mach-Zehnder. Jornadas Chilenas de Computación. (2008)
8. Negovetic, G., Perkowski, M., Lukac, M., Buller, A.: Evolving quantum circuits and an FPGA-based Quantum Computing Emulator. Proc. Intern. Workshop on Boolean Problems. (2002) 15–22
9. Nielsen, M. A., Chuang, I. L.: Quantum Computation and Quantum Information. Cambridge University Press. (2000)
10. Pedroni, V.A.: Circuit Design with VHDL. MIT Press. (2004)
11. Portugal, R., Lavor, C., Maculan, N.: Uma introdução à Computação Quântica. Notas em Matemática Aplicada - SBMAC. (2004)
12. Raedt, K., Michielsen, K., Raedt, H., Trieu, B., Arnold, G., Richter, M., Lippert, T., Watanabe, H., Ito, N.: Massively parallel quantum computer simulator. Computer Physics Communications. (2007) 121–136
13. Ricci, T.F., Ostermann, F., Prado, S.D.: O tratamento clássico do interferômetro de Mach-Zehnder: uma releitura mais moderna do experimento da fenda dupla na introdução da física quântica. Revista Brasileira de Ensino de Física. (2007) 79–88
14. Rieffel, E.G, Polak, W.: An Introduction to Quantum Computing for Non-Physicists. ACM Computing Surveys. (2000) 300–335
15. Shende, V. V., Bullock, S. S., Markov, I.: Synthesis of Quantum Logic Circuits. IEEE Transactions on Computer-Aided Design. (2006) 1000–1010
16. Wan, Q., Brewer, K.J.: IEEE-754 Floating-Point Conversion from Floating-Point to Hexadecimal. City University of New York. (2003) http://babbage.cs.qc.edu/IEEE-754/Decimal.html
17. Xilinx:Xilinx Manuals Online. Xilinx Inc. (1998) http://toolbox.xilinx.com/docsan/data/html/home.htm