



Universidade do Estado do Rio de Janeiro

Centro de Tecnologia e Ciências

Faculdade de Engenharia


Sérgio de Souza Raposo

Coprocessador para operações quânticas

Rio de Janeiro
2012

Sérgio de Souza Raposo

Coprocessador para operações quânticas



Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Orientadora: Prof^a. Dr^a. Nadia Nedjah
Co-orientadora: Prof^a. Dr^a. Luiza de Macedo Mourelle

Rio de Janeiro
2012

CATALOGAÇÃO NA FONTE
UERJ/REDE SIRIUS/CTC/B

R586 Raposo, Sérgio de Souza
Coprocessador para operações quânticas/Sérgio de
Souza Raposo. – 2012.
100f. : il.

Orientadora: Nadia Nedjah.

Co-orientadora: Luiza de Macedo Mourelle.

Dissertação (Mestrado) – Universidade do Estado do
Rio de Janeiro, Faculdade de Engenharia.

Bibliografia: f. 97 – 100.

1. Coprocessador. 2. Computação quântica - Dissertação. 3. Emuladores. I. Nedjah, Nadia. II. Mourelle, Luiza de Macedo. III. Universidade do Estado do Rio de Janeiro. Faculdade de Engenharia. IV. Título.

CDU 510.5

Autorizo, apenas para fins acadêmicos e científicos, a reprodução total ou parcial desta dissertação, desde que citada a fonte.

Assinatura

Data

Sérgio de Souza Raposo

Coprocessador para operações quânticas

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Aprovado em 27 de Fevereiro de 2012

Banca Examinadora:

Prof.^a. Dr.^a. Nadia Nedjah (Orientadora)
Faculdade de Engenharia, UERJ

Prof.^a. Dr.^a. Luiza de Macedo Mourelle (Co-orientadora)
Faculdade de Engenharia, UERJ

Prof. Dr. Maurício Lima Pilla
Departamento de Computação, UFPEL

Prof. Dr. Renato Portugal
Laboratório Nacional de Computação Científica, LNCC

Rio de Janeiro
2012

Dedico este trabalho àqueles dotados de espírito empreendedor que, não obstante os obstáculos e limitações objetivas e subjetivas que se lhes apresentam, perseveram no alcance de seu objetivo, ainda que não logrem o ideal auto-estabelecido, pois um empreendimento é composto não apenas de elementos tangíveis e dos imateriais, mas da alma do autor.

AGRADECIMENTOS

A Deus pelas pessoas, oportunidades, recursos e saúde que me possibilitaram chegar a esta etapa do Mestrado.

À Elza Regiani, Alessandra e Nathália, minha família, que compreenderam e resignaram-se com minha menor disponibilidade enquanto precisei dedicar-me aos estudos.

Às Professoras Dr^a. Nadia Nedjah (Orientadora) e Dr^a. Luiza de Macedo Mourelle (Co-orientadora) pela preciosa cessão de seus conhecimentos e experiência, disponibilidade e paciência com minhas limitações, sem as quais não poderia alcançar tal estágio neste curso.

Aos Professores(as) Dr. Jorge Luís Machado do Amaral, Dr. José Franco Machado do Amaral, Dr. Luiz Biondi Neto, Dr^a. Luiza de Macedo Mourelle, Dr^a. Nadia Nedjah e Dr. Nival Nunes de Almeida pelos ensinamentos nas disciplinas necessárias ao cumprimento dos créditos e pela amizade.

Ao Professor Dr. Renato Portugal (LNCC - Laboratório Nacional de Computação Científica), pelas inúmeras vezes em que, atenciosa e simpaticamente, atendeu-me quando solicitei-lhe ajuda em Mecânica Quântica e Computação Quântica.

Aos Colegas do Mestrado Marcos Santana, Edgar Garcia, Luneque Silva, Rogério Moraes, Fábio Pessanha e Rafael Mathias, pedindo perdão pelas involuntárias omissões, pela amizade e incontáveis vezes que, dedicadamente, disponibilizaram seus tempos e conhecimentos para eliminar as dúvidas que encontrei nos mais diversos assuntos.

À FAPERJ, pelo concessão da bolsa de estudo.

A todas as pessoas listadas acima e as que imperdoavelmente não foram citadas, faço aqui o reconhecimento da contribuição inestimável e indispensável prestada, sem a qual os méritos que esta Dissertação venha a conquistar, incondicionalmente compartilhados com vocês, não seriam possíveis ou o seriam em menor grau!

RESUMO

RAPOSO, Sérgio de Souza. *Coprocessador para operações quânticas*. 2012. 100f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2012.

A demanda crescente por poder computacional estimulou a pesquisa e desenvolvimento de processadores digitais cada vez mais densos em termos de transistores e com *clock* mais rápido, porém não podendo desconsiderar aspectos limitantes como consumo, dissipação de calor, complexidade fabril e valor comercial. Em outra linha de tratamento da informação, está a computação quântica, que tem como repositório elementar de armazenamento a *versão quântica* do *bit*, o *q-bit* ou *quantum bit*, guardando a superposição de dois estados, diferentemente do *bit* clássico, o qual registra apenas um dos estados. Simuladores quânticos, executáveis em computadores convencionais, possibilitam a execução de algoritmos quânticos mas, devido ao fato de serem produtos de *software*, estão sujeitos à redução de desempenho em razão do modelo computacional e limitações de memória. Esta Dissertação trata de uma versão implementável em *hardware* de um coprocessador para simulação de operações quânticas, utilizando uma arquitetura dedicada à aplicação, com possibilidade de explorar o paralelismo por replicação de componentes e *pipeline*. A arquitetura inclui uma memória de estado quântico, na qual são armazenados os estados individuais e grupais dos *q-bits*; uma memória de rascunho, onde serão armazenados os operadores quânticos para dois ou mais *q-bits* construídos em tempo de execução; uma unidade de cálculo, responsável pela execução de produtos de números complexos, base dos produtos tensoriais e matriciais necessários à execução das operações quânticas; uma unidade de medição, necessária à determinação do estado quântico da máquina; e, uma unidade de controle, que permite controlar a operação correta dos componente da via de dados, utilizando um microprograma e alguns outros componentes auxiliares.

Palavras-chave: Coprocessador. Computação quântica. Emuladores.

ABSTRACT

The growing demand for computational power has pushed the research and development of digital processors that are even more dense in terms of transistor number and faster clock rate, without ignoring concerning constraints such as energy consumption, heat dissipation, manufacturing complexity and final market costs. Another approach to deal with digital information is quantum computation, that relies on a basic storage entity that keeps a superposition of the two possible states, in contrast with of a bit of a conventional computer, that stores only one of these two states. Simulators for quantum computation can run quantum algorithms on conventional computers. However, since these are developed using a software implementation, performance limitation occur due to the classical computational model used. This dissertation presents an implementable hardware architecture of a specialized coprocessor that simulates quantum operations, employing an application-specific design that allows parallel processing based on component replication and pipelining. The proposed architecture includes a quantum state memory, where individual and joined states of q -bits are stored; a scratch memory, dedicated to storing quantum operators that are built at runtime; the arithmetic unit, that performs complex numbers multiplications, to allow the full computation of tensorial and scalar products of matrices, required to implement quantum operators; the measurement unit, that is required to perform quantum state observation; and the control unit, that controls proper operation of the datapath components using a microprogram and some other auxiliary components.

Keywords: Coprocessor. Quantum computing. Emulators

LISTA DE FIGURAS

1	Sistema esférico de coordenadas	22
2	Esfera de Bloch	23
3	Circuito clássico para determinação de voto majoritário	25
4	Circuito quântico para determinação de voto majoritário	26
5	Circuito de um operador Swap construído com 3 operadores CNOT	36
6	Comunicação entre processador e coprocessador	39
7	Macro-arquitetura do coprocessador	39
8	Interface da memória de estado quântico	41
9	Interface da memória de q -bits	41
10	Palavra da memória MQ - parte dos coeficientes	41
11	Diagrama de tempos de leitura e gravação simultâneas em MQ	42
12	Palavra da memória MQ - parte do controle	43
13	Memória de controle de q -bits	43
14	Diagrama de tempos de leitura e gravação simultâneas em MCQ	44
15	Interface da memória de operadores	45
16	Diagrama de tempos de leitura de MOp	46
17	Interface da memória de rascunho	47
18	Diagrama de tempos de leitura e gravação simultâneas em MR	48
19	Micro-arquitetura da unidade de medição do estado quântico	50
20	Arquitetura da unidade de cálculo	52
21	Multiplicador de números complexos	54
22	Unidade de controle	64
23	Instrução descritiva de operação quântica	65
24	Circuito de endereçamento da memória de controle	67
25	Inicialização dos q -bits	82
26	Produto matricial com operador com 1 q -bit	83
27	Produto matricial de uma operação NOT sobre 2 q -bits não-entrelaçados	85
28	Visão ampliada das duas operações NOT sobre 1 q -bit cada	85
29	Visão ampliada da primeira operação NOT	86
30	Visão ampliada da segunda operação NOT	86
31	Construção do operador para 2 q -bits e limite de 4 q -bits por operador	87
32	Construção do operador para 3 q -bits e limite de 4 q -bits por operador	88
33	Detalhe da 1ª parte do diagrama de tempos da construção do operador	88
34	Detalhe da 2ª parte do diagrama de tempos da construção do operador	88
35	Detalhe da 3ª parte do diagrama de tempos da construção do operador	89
36	Produto tensorial entre dois q -bits	90
37	Produto matricial: multiplicações com a 1ª linha do operador	91
38	Produto matricial: multiplicações com a 2ª linha do operador	92

39	Produto matricial: multiplicações com a 3 ^a linha do operador	92
40	Produto matricial: multiplicações com a 4 ^a linha do operador	92
41	Produto matricial: gravação do resultado da operação na memória MQ	93
42	Produto matricial: medição e gravação do estado colapsado na memória MQ . . .	93

LISTA DE TABELAS

1	Direção e sentido representativos do q -bit para alguns valores.	23
2	Tabela-verdade do circuito de voto majoritário	25
3	Tabela-verdade para o operador CNOT	34
4	Tabela-verdade para o operador Toffoli com 2 q -bits de controle.	35
5	Estados nas entradas e saídas do operador Swap: CNOT ₁	36
6	Estados nas entradas e saídas do operador Swap: CNOT ₂	36
7	Estados nas entradas e saídas do operador Swap: CNOT ₃	36
8	Tabela verdade para o operador Fredkin	37
9	Ocupação dos endereços da memória MOp.	46
10	Coprocessador com 2 q -bits e probabilidades hipotéticas.	49
11	Coprocessador com 2 q -bits e subintervalos hipotéticos.	49
12	Códigos da operações quânticas	66
13	Operador NOT para 2 q -bits: distribuição dos coeficientes aos pares.	74
14	Endereços da memória MR dimensionada para 2 q -bits	74
15	Endereços da memória MR dimensionada para 3 q -bits	75
16	Endereços da memória MR dimensionada para 4 q -bits	75
17	Endereços ocupados por operador para 2 q -bits - MR dimensionada para 3 q -bits	76
18	Endereços ocupados por operador para 2 q -bits - MR dimensionada para 4 q -bits	76
19	Endereços ocupados por operador para 3 q -bits - MR dimensionada para 4 q -bits	77
20	Valores dos coeficientes de q -bits antes e depois de um produto tensorial	89
21	Alguns valores reais e a representação hexadecimal do formato IEEE754	91
22	Probabilidades associadas a cada estado quântico possível	93

SUMÁRIO

	INTRODUÇÃO	12
1	TRABALHOS RELACIONADOS	15
1.1	Visão geral	15
1.2	Considerações relativas aos trabalhos relacionados.	18
1.3	Proposta desta dissertação	18
2	COMPUTAÇÃO QUÂNTICA	19
2.1	Computação clássica	19
2.2	Computação Quântica.	20
2.2.1	<u>O bit quântico</u>	20
2.2.2	<u>Paralelismo quântico</u>	24
2.2.3	<u>Medição do estado quântico</u>	26
2.2.4	<u>Emaranhamento</u>	27
2.2.5	<u>Não-clonagem</u>	27
2.2.6	<u>Produto tensorial</u>	28
2.2.7	<u>Operadores quânticos</u>	31
2.2.7.1	Operador NOT quântico ou X.	31
2.2.7.2	Operador Y.	32
2.2.7.3	Operador Z	32
2.2.7.4	Operador Hadamard ou H	32
2.2.7.5	Operador de fase ou S	33
2.2.7.6	Operador $\frac{\pi}{8}$ ou T	33
2.2.7.7	Operador Identidade ou I.	33
2.2.7.8	Operador CNOT	34
2.2.7.9	Operador Toffoli.	34
2.2.7.10	Operador Swap	35
2.2.7.11	Operador Fredkin ou CSwap.	36
2.3	Considerações finais do capítulo	37
3	ARQUITETURA DO COPROCESSADOR	38
3.1	Visão geral	38
3.2	Macro-arquitetura do coprocessador	38
3.2.1	<u>Memória de estado quântico</u>	40
3.2.1.1	Memória de <i>q-bits</i>	41
3.2.1.2	Memória de controle de <i>q-bits</i>	42
3.2.2	<u>Memória de operadores</u>	44
3.2.3	<u>Memória de rascunho</u>	46
3.2.4	<u>Unidade de medição</u>	47
3.2.5	<u>Unidade de cálculo</u>	50
3.2.6	<u>Unidade de controle</u>	50

3.3	Considerações finais do capítulo	50
4	UNIDADE DE CÁLCULO	51
4.1	Micro-arquitetura da unidade	51
4.2	Multiplicação de números complexos.	53
4.3	Produto tensorial	54
4.3.1	<u>Produto tensorial de q-bits</u>	54
4.3.2	<u>Produto tensorial de operadores</u>	55
4.4	Produto matricial	60
4.5	Considerações finais do capítulo	61
5	UNIDADE DE CONTROLE	63
5.1	Micro-arquitetura da unidade	63
5.2	Gerenciador de instruções	65
5.3	Memória de controle	67
5.3.1	<u>Registrador de endereço de microprograma</u>	67
5.3.2	<u>Tratamento de desvio incondicional ou condicional</u>	68
5.4	Microprograma	68
5.4.1	<u>Micro-ordens</u>	69
5.4.2	<u>Flags</u>	70
5.4.3	<u>Operando</u>	70
5.5	Controladores de produto tensorial	71
5.5.1	<u>Controlador de produto tensorial de q-bits</u>	71
5.5.2	<u>Controlador de produto tensorial de operadores quânticos básicos</u>	73
5.5.2.1	Controlador de gravação de produto tensorial em MR	76
5.5.2.2	Registrador auxiliar de quantidade de q -bits do operador quântico	77
5.5.2.3	Conversores de endereço dos coeficientes na memória MR.	77
5.5.2.4	Registradores de endereço	79
5.6	Considerações finais do capítulo	80
6	RESULTADOS DE SIMULAÇÃO	81
6.1	Inicialização dos q-bits	81
6.2	Operações quânticas	82
6.2.1	<u>Operação quântica sobre 1 q-bit</u>	82
6.2.2	<u>Operação quântica sobre 2 ou mais q-bits não-emaranhados</u>	84
6.2.3	<u>Operação quântica sobre 2 ou mais q-bits emaranhados</u>	87
6.2.3.1	Operador para 2 q -bits com memória MR limitada a 4 q -bits	87
6.2.3.2	Operador para 3 q -bits com memória MR limitada a 4 q -bits	87
6.2.4	<u>Produto tensorial de q-bits</u>	89
6.2.5	<u>Produto matricial sobre dois q-bits emaranhados</u>	90
6.3	Considerações finais do capítulo	94
7	CONCLUSÃO E TRABALHOS FUTUROS	95
7.1	Conclusões	95
7.2	Trabalhos futuros	96
	REFERÊNCIAS	97

INTRODUÇÃO

O COMPUTADOR eletrônico convencional é uma máquina que executa operações lógicas e aritméticas sequencialmente sobre dados construídos sobre uma base binária. Este modelo Cartesiano para a unidade elementar de informação, o bit, é bastante conveniente para o tratamento dos dados em um circuito eletrônico, pois os dois estados possíveis podem ser representados de maneira muito bem controlada e segura por circuitos digitais.

De acordo com a versão atualizada da *Lei de Moore*, originalmente cunhada por Roger Moore em 1965, a capacidade de processamento dos computadores dobra a cada período de vinte e quatro meses (BARBOSA, 2007). A aceleração da execução das instruções pode ser conseguida de diversas formas, como o aumento da frequência de *clock*, emprego de periféricos inteligentes para reduzir a demanda sobre processador principal, utilização de processadores múltiplos e/ou mais de um processador, arquiteturas mais avançadas de processador (Harvard × Von Neumann convencional), processadores com instruções mais simples (tipicamente RISC), uso de memórias mais rápidas entre outras. Nos últimos quase 50 anos a Lei de Moore tem se cumprido, mas caso não surja uma nova tecnologia, a validade desta Lei está ameaçada devido à aproximação dos limites físicos (BARBOSA, 2007). O aumento da performance dos processadores por meio do aumento da quantidade e densidade de transistores está se aproximando do limite exequível para a tecnologia atual, não apenas pela complexidade de desenvolver processadores com precisão nanométrica que possam ser produzidos em escala industrial, com as implicações em consumo de energia, dissipação de calor e precisão no processo de fabricação de dispositivos diminutos, mas também pela aproximação de um contexto quântico. Em outras palavras, as reduzidas dimensões dos componentes de um processador tendem a se afastar das previsíveis e bem conhecidas regras da Física clássica. Em (SCIENCEDAILY, 2008) comenta-se da disponibilidade da tecnologia de 45nm e já é citada a de 22nm, apresentada atualmente pela Intel[®] como disponível para produção no final de 2011 (INTEL, 2011).

A Computação Quântica tem sido alvo de pesquisas diversas devido ao potencial de aumento na velocidade de processamento por meio de uso de algoritmos dotados de paralelismo intrínseco, acenando com a possibilidade de solução em tempo polinomial de algoritmos do tipo

NP-Completo (BRUMATTO, 2010).

A tecnologia dos computadores quânticos ainda busca formas de controlar elétrons, o que já é conseguido com poucos ainda e por pouco tempo. À época da conclusão desta Dissertação, o maior número de *q-bits* emaranhados alcançado em condições especiais era quatorze (MATSON, 2011; MONZ, T. et al., 2011).

Investimentos de empresas já resultam em dispositivos comerciais, como é o caso da D-Wave, que anuncia um computador dispondo de 128 *q-bits* (D-WAVE, 2012), porém não emaranháveis em sua totalidade e voltado à execução de algoritmos para computação adiabática (GUIZZO, 2010; JOHNSON, M. W. et al, 2011; AARONSON, 2011; CAMPBELL, 2011).

Paralelamente, a tecnologia para produção de *chips* está chegando ao limite da miniaturização, aproximando-se da fronteira que separa o mundo da abstração clássica da Física quântica. Enquanto não existirem processadores quânticos comerciais, a programação quântica está sendo testada por meio de emuladores e bibliotecas de rotinas para operações quânticas, dos quais pode-se elencar exemplos como QCL(OMER, 1998), QCS ,QuaSi, Fraunhofer Quantum Computing Simulator, QuCalc, QDensity, OpenQuacs, QML, JaQuzzi, Senko's Quantum Computer, Shornuf, SimQ-bit e QHaskell (BARBOSA, 2007; VIZZOTTO; COSTA, 2006; MARQUEZINO, 2006). Destaca-se o *Jülicher Supercomputer Jugene simuliert*, um computador IBM BlueGene/P de 825.5 Teraflops (Linpack), 144TB de memória principal (JUGENE... , 2010), capaz de simular operações com 42 *q-bits*, desenvolvido pelo grupo de pesquisa Jülich e o grupo de Física Computacional da Universidade de Groningen, na Holanda, neste momento ainda invicto em termos de quantidade de *q-bits* emaranhados (MICHELSEN, 2010).

Simuladores, sendo produtos de *software*, implicam em maior tempo de processamento devido à execução sequencial, eventualmente acelerada pelo uso de processadores múltiplos, e ao compartilhamento de recursos com outros processos, do que se fosse executado puramente por *hardware*, que entregaria os resultados das operações quânticas em menor tempo, graças à maior rapidez que um circuito eletrônico processa informações e ao paralelismo na execução das operações. Tal *hardware* poderia assumir a forma de um processador especializado, um *coprocessador quântico*, que faria parte de um computador maior com um processador convencional. Esta Dissertação apresenta a arquitetura e simulações de um Coprocessador quântico que poderá vir a ter uma implementação em *hardware* em um trabalho futuro.

Esta dissertação está organizada em sete capítulos, além desta introdução. Primeiramente, o Capítulo 1 aborda alguns trabalhos relativos à implementação em *hardware* de circuitos quânticos. Seguindo-se, o Capítulo 2 apresenta os aspectos essenciais e conceitos bá-

sicos da computação quântica, assim como definições dos operadores quânticos mais usuais. Na sequência, o Capítulo 3 descreve a macro-arquitetura do Coprocessador proposto junto com a estrutura das diferentes memórias incluídas. Subsequentemente, o Capítulo 4 detalha a unidade de cálculo, um dos componentes-chave da macro-arquitetura. Prosseguindo, o Capítulo 5 discorre sobre a unidade de controle, componente da macro-arquitetura que gerencia todos os outros componente da micro-arquitetura; Depois, o Capítulo 6 trata dos resultados de simulação obtidos. Finalmente, esta dissertação encerra-se com as conclusões no Capítulo 7.

Capítulo 1

TRABALHOS RELACIONADOS

Diversas pesquisas no meio acadêmico têm investido na simulação de operações quânticas em *software* e *hardware*, encontrando-se trabalhos que implementam bibliotecas, circuitos quânticos em dispositivos programáveis ou reconfiguráveis, iniciativas não limitadas ao uso de linguagens descritoras de *hardware*, como o VHDL, havendo também uso de ferramentas de modelagem que empregam outras linguagens de programação.

1.1 Visão geral

(MARON et al., 2011) descreve a otimização da biblioteca de execução do ambiente VPE-qGM (Visual Programming Environment for the qGM Model - QGM: Quantum Geometric Machine), na qual foi utilizada recursividade de funções matemáticas e métodos para geração dinâmica dos valores que definem transformações quânticas, obtendo considerável diminuição do consumo de memória. O modelo QGM substitui o conceito de portas quânticas pelo paradigma de *sincronizações de processos elementares*(PEs). A demanda por espaço em memória é uma questão relevante devido ao grande número de coeficientes que um operador pode conter, pois o crescimento das dimensões do operador quântico cresce exponencialmente com a quantidade de *q-bits*-alvo. Os Autores consideraram que os resultados alcançados revelam um avanço importante para o suporte de algoritmos quânticos com quantidade expressiva de *q-bits* no ambiente VPE-qGM, tendo sido vislumbrado potencial continuidade que otimizaria simulação quântica distribuída e a biblioteca de execução e a representação da estrutura da memória de estado quântico.

(MONTEIROY et al., 2008) aborda conceitos, tecnologias e aplicações da simulação quântica em *hardware* e sua importância para o desenvolvimento tecnológico e científico, introduzindo a definição de métodos descritores de circuitos quânticos com base no modelo de circuitos clássicos. Por meio da extensão de bibliotecas de VHDL (VHSIC Hardware Description

Language) que contempla controle e fluxo de dados quânticos e clássicos, viabilizou a implementação de algoritmos quânticos em FPGA (Field Programmable Gate Array), tendo o uso daquela linguagem motivado pela possibilidade de descrever os circuitos quânticos a partir do modelo empregado em circuitos clássicos. A simulação em *hardware* de algoritmos quânticos, de acordo com os Autores, desperta grande interesse aos pesquisadores de computação quântica devido à possibilidade de análise de propriedades como o paralelismo quântico, emaranhamento e superposição de estados. O trabalho ainda apresenta um exemplo de validação calcada na implementação da versão quântica do *Interferômetro de Mach-Zehnder*, utilizando diferentes configurações de FPGA's, permitindo ainda a análise de desempenho e dimensão dos circuitos quânticos simulados. Os Autores consideram um grande desafio para o desenvolvimento de aplicações em sistemas de comunicação, incluindo aplicações particulares de circuitos integrados em ASIC's, a análise da metodologia da programação quântica em sistemas digitais e o processamento de sinais. Entendem os Autores que contribuição relevante do trabalho é a simulação do paralelismo quântico associado às portas (operadores) unitárias e anteveem como continuação da pesquisa o tratamento da medição do estado quântico por processo probabilístico, análise de erro e otimização da implementação.

(MONTEIRO, 2009) faz um estudo de linguagens de descrição de *hardware* (HDL: *Hardware Description Language*), especialmente a VHDL, na modelagem de circuitos quânticos, empregando a metodologia qExVHDL (Quantum Extension for VHDL), para estudo de caso do interferômetro de Mach-Zehnder, algoritmo de Deutsch e o algoritmo de Grover, simulação e implementação em *hardware* (FPGA) dos mesmos, testes, verificação de área requerida e frequência de operação, comprovando a possibilidade de construção de circuitos quânticos dedicados. Constatou-se neste trabalho a dificuldade de manipular dados de ponto flutuante, requerendo a conversão para uma base binária com 32 ou 64 bits por meio de ferramentas específicas. Observada também a demanda exponencialmente crescente de memória em função da quantidade de *q-bits*. A Autora relata dificuldade na alocação de memória RAM na etapa de síntese pelo *software* ISE da Xilinx. Nas implementações relativas aos Algoritmos de Deutsch e Grover, uma máquina com memória RAM de 2GB não foi suficiente para realizar as sínteses dos circuitos, sendo requeridos 4 GB, exigindo o emprego de um computador com processador Intel Xeon 2.5 GHz e 8GB de memória RAM e otimizações no tratamento de números complexos do gerador de código Quantum-RAMAGT (*Radix^{2^m} Array Multipliers Automatic Generation Tool*).

(MONTEIRO et al., 2009) trata da implementação em FPGA do Algoritmo de Deutsch

utilizando a ferramenta Quantum-RAMAGT para gerar a descrição em VHDL do código a ser sintetizado. Neste estudo de caso, implementam-se os operadores unitários e também a porta Hadamard (H). Utilizou-se o padrão IEEE754 com precisão simples (32 bits). Apresentam-se os resultados da simulação voltada à análise do paralelismo quântico, incluindo análise de erro.

(MONTEIRO et al., 2009) apresenta a metodologia qExVHDL para simulação de algoritmos quânticos utilizando VHDL e fazendo a síntese para implementação em FPGA. Neste trabalho, considerou-se um modelo de circuitos quânticos como base para modelagem de algoritmos quânticos. A validação da proposta utilizou estudo de caso referente do algoritmo de Grover para análise do paralelismo quântico. A simulação em *hardware* permitiu fazer análise do paralelismo quântico calcada no estudo de casos de algoritmos básicos da computação quântica, utilizando a ferramenta Quantum-RAMAGT para geração automática de código e para prototipação de dispositivos reconfiguráveis. Construiu-se uma biblioteca com recursos para simplificação dos coeficientes complexos, modelagem de acumulador de somas, redução da precisão dos dados, tratamento de operações com números 0 e 1. Implementou-se os operadores Hadamard, operador para busca e operador de Grover com precisão de 8 bits. O esforço para redução no consumo de recursos computacionais devido ao aumento exponencial dos circuitos em função da quantidade de *q-bits* tratada. A descrição dos resultados inclui análise da área de silício requerida. Os Autores apresentam a intenção de estudar a aplicação da metodologia ao Algoritmo de Shor e a Teleportação Quântica, a otimização dos circuitos de modo a reduzir a área de silício requerida e o aumento da frequência de *clock*.

(KHALID; ZILIC; RADECKA, 2004) considera 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, inclusive com tratamento do emaranhamento de *q-bits*, computação probabilística e das questões críticas de precisão.

Em *Massively parallel quantum computer simulator* (RAEDT et al., 2007) utiliza componente de *software* com característica portátil para simulação de computadores quânticos universais com emprego de computadores paralelos. A simulação em *software* abarca vários algoritmos quânticos em diferentes arquiteturas de computadores (IBM Blue-Gene/L, IBM Regatta p690, Hitachi SR11000/J1, Cray X1E, SGI Altix 3700) e *clusters* de computadores executados em Windows XP.

(SHENDE; BULLOCK; MARKOV, 2006) analisa a eficiência lógica de circuitos quânticos que desempenham computações quânticas genéricas e inicialização de registradores quânticos.

1.2 Considerações relativas aos trabalhos relacionados

Os trabalhos citados em 1.1 abordam aspectos relacionados a simulações de circuitos quânticos em *hardware*, eventualmente com implementações em dispositivos programáveis com FPGA's. Encontram-se iniciativas que fazem experimentações com circuitos quânticos básicos e as que implementam algoritmos quânticos completos, como o de Grover, apenas para citar um exemplo. A confecção de dispositivos especializados (FPGA ou ASIC) para o tratamento de um problema específico possibilita a redução do tempo de processamento necessário à disponibilização do(s) resultados(s) e do consumo de recursos. O uso de ferramentas de prototipagem específicas para circuitos quânticos proporcionam aumento de produtividade no desenvolvimento de simuladores quânticos pela possibilidade de reuso e extensão de código já validados. É recorrente o investimento em pesquisa ou interesse em reduzir o consumo de memória de estado quântico e especialmente a memória de rascunho, pois o crescimento exponencial da quantidade de coeficientes de um operador quântico em função do número de q -bits simultaneamente representa um problema importante quando se considera o emprego de FPGA's. A precisão também se mostra um desafio nas implementações, pois a representação de números em ponto flutuante na base binária, formatados conforme IEEE754, implica no uso de barramento de dados com largura múltipla de 32, se usada a precisão simples, ou 64, se precisão dupla. Levando-se em conta que um coeficiente de operador quântico e a amplitude de cada um dos dois coeficientes de um *bit quântico* pertence ao conjunto dos números complexos, há de se utilizar uma faixa de vias do barramento de dados para representar a parte inteira e igual quantidade para a parte imaginária.

1.3 Proposta desta dissertação

Propõe-se nesta Dissertação um coprocessador especializado em operações quânticas, descrito em código VHDL parametrizável e escalável, capaz de executar qualquer algoritmo quântico, tratar simultaneamente em uma operação quântica qualquer quantidade de q -bits e qualquer número de q -bits definidos para o coprocessador, sendo limitado pelos recursos computacionais de simulação e síntese e dos recursos físicos do dispositivo programável. O coprocessador contará com paralelismo de operações matemáticas utilizando números complexos codificados no padrão IEEE754 com precisão simples.

Capítulo 2

COMPUTAÇÃO QUÂNTICA

PRINCIPALMENTE , neste capítulo serão introduzidos alguns conceitos básicos de computação quântica. Além disso, serão definidos, em termos práticos, os operadores quânticos, juntamente com a operação de produto tensorial de matrizes, que forma a base de qualquer computação quântica.

2.1 Computação clássica

Pode-se descrever genericamente um computador clássico como uma máquina que lê dados, realiza cálculos e gera saída de dados codificados binariamente, comumente representados por 0s e 1s, valores estes associados a nível baixo e alto de tensão, respectivamente. A computação clássica tem como unidade de informação o bit, o qual pode assumir um entre dois estados a cada vez ou, em termos numéricos, o valor 1 ou 0, tomando-se uma base binária. Até que se comande o armazenamento de um estado (ou valor binário) diferente, o bit manterá seu estado anterior *ad infinitum* enquanto o *hardware* se mantiver energizado (WATANABE, 2003). A energia presente às entradas não iguala-se à energia na saída, portanto a diferença é convertida em calor. Um bit é um repositório de dado independente de outros bits, de modo que a alteração do seu conteúdo não afeta os outros bit e vice-versa. Um bit pode ser lido infinitas vezes sem que seu estado seja modificado. Os dados de entrada são processados por unidades lógicas ou aritméticas que resultam em uma saída, o que equivale a dizer que tais unidades correspondem à funções. Estas unidades realizam operações lógicas mais ou menos complexas, construídas por circuitos apropriados de portas elementares tais como NOT, AND e OR (WATANABE, 2003). Nesta Dissertação, o termo *operador* será empregado como sinônimo de “porta”. Alguns aspectos da computação clássica merecem destaque quando se pretende considerar as particularidades da computação quântica. Na computação clássica, a característica sequencial da execução de instruções armazenadas em uma memória também é

intrínseca ao citado modelo computacional. Ainda que se utilize paralelismo de processadores, periféricos ou subsistemas com processadores próprios, a característica sequencial da execução do programa armazenado em memória permanece. Com exceção da porta NOT, as portas AND, OR e derivadas não permitem conhecer os estados presentes nas suas entradas tomando-se apenas o estado presente na saída. Por exemplo, uma porta OR com duas entradas, quando sua saída está com valor 1, não se pode afirmar se somente uma das entradas ou ambas estão no nível 1. Por isto, as portas clássicas são ditas *irreversíveis*. (PORTUGAL; LAVOR; CARVALHO, 2004; SILVA, 2002)

2.2 Computação Quântica

A Mecânica Quântica é a parte da Física que estuda e descreve o comportamento das partículas nos níveis atômico e subatômico, complementando a Física Clássica, que trata da realidade concreta. Esta apoia-se na Mecânica Newtoniana e no Eletro-magnetismo, calcada na mecânica ondulatória e na matricial, posteriormente unificadas pela descrição de Paul Dirac (OLIVEIRA, 2007). Um computador quântico executa cálculos utilizando as propriedades da mecânica quântica, um contexto probabilístico, diferentemente da condição determinística da computação clássica, embora também possa executar as mesmas operações desta. Na computação quântica existe um paralelismo intrínseco, pois o estado quântico é uma superposição de estados básicos (PORTUGAL; LAVOR; MACULAN, 2004; CARVALHO; LAVOR; MOTTA, 2007; SILVA, 2002). Os repositórios de dados podem estabelecer um relacionamento com interferência mútua, conhecido como *emaranhamento* (OLIVEIRA, 2007).

2.2.1 O bit quântico

A unidade de informação em um computador quântico é o *quantum bit* ou, mais usualmente, *q-bit*¹, que pode ser representado por notação matricial, um vetor-coluna 2×1 , conforme Equação 1, onde α e β são números complexos e que denotam a amplitude de cada vetor da base vetorial ortonormal² de dimensão 2, de acordo com a Equação 2, chamada de *base computacional* na computação quântica. A Equação 3 mostra um *q-bit* na forma de combinação linear (PORTUGAL; LAVOR; MACULAN, 2004; WATANABE, 2003; OLIVEIRA; SARTHOU, 2004). Um *q-bit*, diferentemente do bit clássico, é capaz de armazenar não apenas os estados 0 e 1, mas ambos, cada um com uma magnitude ou *amplitude* própria (OMER, 2009, 2000;

¹*qbit*, *qubit* *quibit* são outras formas encontradas. Nesta Dissertação, será utilizada a forma *q-bit*

²Base ortonormal é um conjunto de vetores linearmente independentes, ortogonais dois a dois, cada um com norma 1, e que geram um espaço vetorial.

OLIVEIRA; SARTHOU, 2004), caracterizando uma *superposição* (PORTUGAL; LAVOR; MACULAN, 2004). Portanto, um *q-bit* pode assumir infinitos valores com característica probabilística, e não determinística. Na Mecânica Quântica, vetor também é referido como *estado* e estes dois termos serão utilizados ao longo desta Dissertação (PORTUGAL; LAVOR; MACULAN, 2004).

$$|v\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \quad (1)$$

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (2)$$

$$|v\rangle = \alpha|0\rangle + \beta|1\rangle \quad (3)$$

Assim, é permitido dizer que um *q-bit* pode representar simultaneamente os estados 0 e 1, porém com coeficientes próprios para cada um. O quadrado da amplitude fornece a probabilidade no intervalo $[0, 1]$ de o *q-bit* encontrar-se naquele estado (WATANABE, 2003), segundo a Regra de Born (LORENZEN, 2009; LIMA, A. F. de; LULA JÚNIOR, B., 2006).

A soma dos quadrados das amplitudes de cada estado possível totaliza 1, conforme a Equação 4:

$$|\alpha|^2 + |\beta|^2 = 1 \quad (4)$$

sendo preservada a norma do vetor (VIGNATTI; NETO; BITTENCOURT, 2004). O estado de um computador quântico passa a ser representável por um vetor que é combinação linear dos vetores da base, possuindo a base dimensão igual a 2^n , sendo n a quantidade de *q-bits*. Logo, um vetor representativo do estado do computador quântico não guarda apenas tantos estados quanto a quantidade de *q-bits*, mas a potência n de 2. O modo comumente adotado para representar um *q-bit* é a *notação de Dirac*, conhecida também como “braket”, nome em Inglês dos caracteres “<” e “>”. A notação inclui o caracter “|” entre os brackets, chamando-se de “bra” o *q-bit* entre o “<” e o “|” e “ket” o *q-bit* entre “|” e “>” (THIBES, 2010; VIGNATTI; NETO; BITTENCOURT, 2004). Os “kets” são utilizados para representar um estado possível do sistema quântico, estado este composto por um ou mais *q-bits*. Um *q-bit* no chamado estado colapsado, isto é, com 100% de probabilidade de estar em estado 0 ou 1, é representado como $|0\rangle$ ou $|1\rangle$, respectivamente. Um estado composto por mais de 1 *q-bit* seria escrito com tantos algarismos do conjunto $\{0, 1\}$ quanto os de *q-bits*, cuja ordem cresce da direita para a esquerda, como em uma representação binária (RIEFFEL; POLAK, 2000; VIGNATTI; NETO; BITTENCOURT, 2004). Este agrupamento de *a-bits* é chamado de *registro quântico* (WATANABE, 2003; OMER, 2000; BERTHIAUME, 1996). Por exemplo,

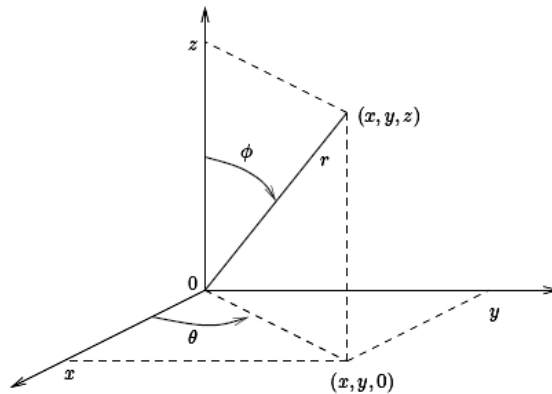


Figura 1: Sistema esférico de coordenadas

- 2 *q-bits*: $|0\rangle|1\rangle$ ou $|01\rangle$: primeiro *q-bit* no estado 0 e o segundo *q-bit* no estado 1;
- 3 *q-bits*: $|1\rangle|1\rangle|0\rangle$ ou $|110\rangle$: primeiro *q-bit* no estado 0 e o outros *q-bit* no estado 1;

Uma forma alternativa de representar um conjunto de *q-bits* utiliza a base decimal ao invés da binária. Ilustrando, o exemplo acima com o conjunto de *q-bits* $|110\rangle$ poderia ser representado como $|6\rangle$. A base para representar um conjunto de n *q-bits* possui 2^n vetores ou *estados*. Logo um conjunto de n *q-bits* pode ser representado conforme Equação 5

$$|v\rangle = \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle + \dots + \omega|2^n - 1\rangle \quad (5)$$

Sendo um vetor unitário, a soma do quadrados dos módulos das amplitudes deve resultar 1, como mostra a Equação 6:

$$|\alpha|^2 + |\beta|^2 + |\gamma|^2 + \dots + |\omega|^2 = 1 \quad (6)$$

Um sistema esférico de coordenadas, como se observa na Figura 1, serve de base para a representação gráfica do estado de um *q-bit*, conhecida como *Esfera de Bloch* (CARVALHO; LAVOR; MOTTA, 2007), ilustrada na Figura 2 (PORTUGAL; LAVOR; MACULAN, 2004; OMER, 2009). Tal representação polar em \mathbb{C}^3 possibilita visualizar o estado quântico e as infinitas possibilidades que poderiam resultar em uma probabilidade p de uma medição resultar $|0\rangle$ ou probabilidade $1 - p$ de resultar $|1\rangle$. As projeções nos eixos x , y e z são podem ser obtidas pela Equação 7

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos\phi \cdot \text{sen}\theta \\ \text{sen}\phi \cdot \text{sen}\theta \\ \cos\theta \end{bmatrix} \quad (7)$$

Sendo que $0 \leq \theta \leq \pi$ e $0 \leq \phi \leq 2\pi$ Desta forma, a base para representação em \mathbb{C}^3 é definida em na Equação 8 (PORTUGAL; LAVOR; MACULAN, 2004)

$$|0\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \quad (8)$$

Assim, considerando a representação esférica de coordenadas, vê-se que o estado colapsado $|0\rangle$ resulta em um vetor apontando para o norte e o estado colapsado $|1\rangle$ para o sul (CARVALHO; LAVOR; PORTUGAL, 2005). Como o vetor é aplicado no centro de uma esfera, sua norma (comprimento) mantém-se constante e vale 1, condizente com a soma dos quadrados das amplitudes dos dois estados (vetores da base) possíveis (CARVALHO; LAVOR; PORTUGAL, 2005).

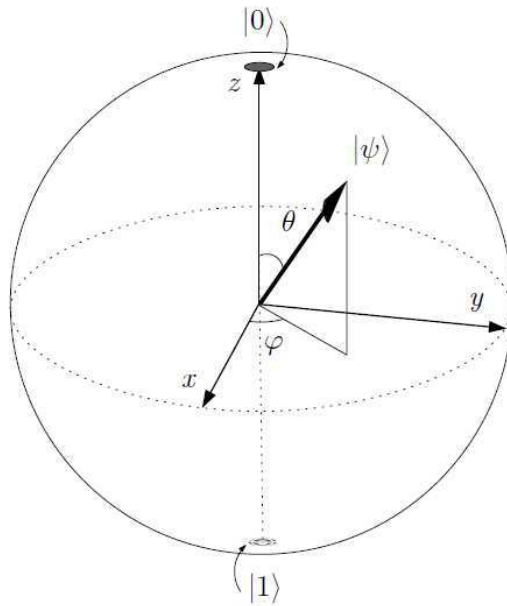


Figura 2: Esfera de Bloch

Tome-se as igualdades apresentadas na Equação 9 (PORTUGAL; LAVOR; MACULAN, 2004):

$$\alpha = \cos(\theta/2), \beta = e^{i\phi} \sin(\theta/2) \quad (9)$$

A representação do estado de um q -bit pode também assumir a forma trigonométrica, mostrada na Equação 10:

$$|v\rangle = \cos(\theta/2)|0\rangle + e^{i\phi} \sin(\theta/2)|1\rangle \quad (10)$$

A Tabela 1 apresenta a direção e sentido do vetor representativo do q -bit para alguns estados (OLIVEIRA; SARTHOU, 2004; CARVALHO; LAVOR; MOTTA, 2007).

Tabela 1: Direção e sentido representativos do q -bit para alguns valores

θ	ϕ	$ v\rangle$	observações
0	0	$ 0\rangle$	polo norte
π	0	$ 1\rangle$	polo sul
$\frac{\pi}{2}$	0	$\frac{ 0\rangle+ 1\rangle}{\sqrt{2}}$	equador - sobre eixo x
$\frac{\pi}{2}$	$\frac{\pi}{2}$	$\frac{ 0\rangle+i 1\rangle}{\sqrt{2}}$	equador - sobre eixo y

2.2.2 Paralelismo quântico

A Equação 5 é uma combinação linear cujos termos são os estados colapsados (vetores da base) multiplicados pela amplitude associada, formando o estado quântico da máquina (BARROS, 2011). Uma operação sobre um vetor resulta igual à soma da operação sobre cada termo da combinação linear, portanto configura-se um paralelismo intrínseco (SILVA, 2002; RIEFFEL; POLAK, 2000). Sendo T um operador hipotético, o desenvolvimento leva à Equação 11

$$T|v\rangle = T(\alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle + \dots + \omega|2^n - 1\rangle)$$

$$T|v\rangle = \alpha T|0\rangle + \beta T|1\rangle + \gamma T|2\rangle + \dots + \omega T|2^n - 1\rangle \quad (11)$$

Uma porta lógica permite a avaliação de mais de um estado simultaneamente (OMER, 2000), conforme poderá ser observado no exemplo a seguir. Seja $|\phi\rangle$ um registrador quântico conforme Equação 12,

$$|\phi\rangle = |a\rangle|b\rangle \quad (12)$$

e T um operador unitário que, operado sobre $|\phi\rangle$, resulta na Equação 13

$$T(|a\rangle, |b\rangle) = T(|\phi\rangle) = |a\rangle|b \oplus f(a)\rangle \quad (13)$$

sendo \oplus uma soma com módulo 2, $a, b \in \{0, 1\}$, $f(a) : \{0, 1\} \rightarrow \{0, 1\}$. Supondo que $|a\rangle$ esteja em estado superposto tal que (Equação 14):

$$|a\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}, |b\rangle = |0\rangle \quad (14)$$

resulta na Equação 15

$$T(|\phi\rangle) = |a\rangle|0 \oplus f(a)\rangle \quad (15)$$

Sendo $|b\rangle = |0\rangle$, $|0 \oplus f(n)\rangle = |f(n)\rangle$, assim obtém-se a Equação 16

$$T(|\phi\rangle) = |a\rangle|f(a)\rangle =$$

$$T(|\phi\rangle) = \frac{|0\rangle|f(0)\rangle + |1\rangle|f(1)\rangle}{\sqrt{2}} \quad (16)$$

Neste exemplo, ilustra-se a possibilidade em que o q -bit a está em estado superposto, ou seja, dois estados fundamentais ($|0\rangle$ e $|1\rangle$) presentes simultaneamente, de modo que uma porta quântica atua sobre todos os estados simultaneamente.

Tabela 2: Tabela-verdade do circuito de voto majoritário

A	B	C	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

A Figura 4 (MARQUEZINO, 2006) mostra um exemplo de aplicação desta atuação simultânea em um circuito de voto majoritário quântico, cuja versão com porta clássicas pode ser visto na Figura 3. Sendo um circuito com portas clássicas, a avaliação dos oito estados possíveis (considerando-se a existência de três entradas: a , b e c), somente poderia ocorrer em momentos diferentes, demandando mais de uma iteração do algoritmo ou a atuação de mais processadores (MARQUEZINO, 2006).

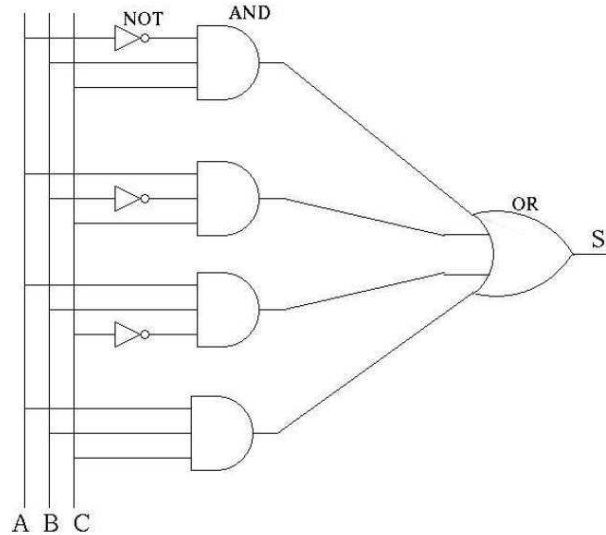


Figura 3: Circuito clássico para determinação de voto majoritário

Este circuito apresenta em sua saída o mesmo estado lógico da maioria das entradas A , B e C , conforme tabela-verdade mostrada na Tabela 2. A versão quântica do circuito de voto majoritário encontra-se na Figura 4 (NIELSEN; CHUANG, 2000). Os quadrados identificados com “X” representam portas NOT quânticas, o equivalente quântico da porta inversora clássica. Em cada uma das linhas de controle a , b e c existe uma segunda porta X com o propósito de restabelecer o estado original do q -bit de controle. As conexões representadas por pontos pretos referem-se aos q -bits de controle e a conexão representada por \oplus refere-se ao q -bit alvo de uma

porta Toffoli generalizada, explicada em detalhe na Seção 2.2.7.9. Por ora, é suficiente entender que uma porta Toffoli funciona como uma porta quântica inversora, que somente atua sobre o q -bit-alvo quando os q -bits de controle estão no estado 1. As entradas a , b e c recebem os q -bits que representam os “votos” e atuam como controle das portas Toffoli. As cinco entradas recebem um dado quântico colapsado em $|0\rangle$ e são os alvos das portas Toffoli. Cada uma das três primeiras portas Toffoli, contadas da esquerda para a direita, trata um caso de “duplo voto $|1\rangle$ ”, respectivamente “ b e c ”, “ a e c ” e “ a e b ”. A quarta porta Toffoli resolve o caso em que todas as entradas de controle estão com estado $|1\rangle$. A quinta e última porta Toffoli inverte quanticamente o estado da última linha de estado $|0\rangle$ de acordo com a votação, sendo invertida quanticamente este resultado e entregue na saída $|s\rangle$. Utilizando como exemplo a situação em que b e c estão com estado $|1\rangle$ e a com estado $|0\rangle$, a primeira porta Toffoli, estando com valor $|1\rangle$ em todas as suas entradas de controle, irá inverter o estado $|0\rangle$ provido como “ q -bit-alvo”, passando a $|1\rangle$. A porta X na mesma linha inverterá o estado para $|0\rangle$, que impedirá a última porta Toffoli do circuito de inverter o estado de seu “ q -bit-alvo”, permanecendo em $|0\rangle$. Finalmente, a porta X na lista deste “ q -bit-alvo” encarregar-se-á de aplicar à saída $|0\rangle$ o estado $|0\rangle$. Analogamente, a análise é aplicável às outras possibilidades em que 2 ou mais entradas de controle encontram-se no estado $|1\rangle$.

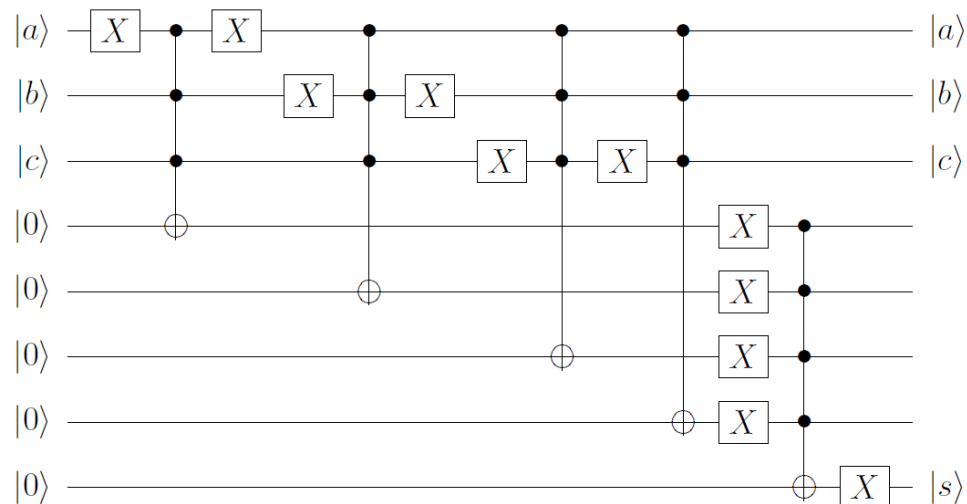


Figura 4: Circuito quântico para determinação de voto majoritário

2.2.3 Medição do estado quântico

Segundo a interpretação da Física, um q -bit está nos estados $|0\rangle$ e $|1\rangle$ ao mesmo tempo, significando que a quantidade de informação no estado possa ser infinita. Todavia, esta infinidade de informação encontra-se no nível quântico. A maneira de fazer a informação tornar-se disponível

no nível clássico é realizando-se uma *medição*, ação esta que, segundo a Mecânica Quântica, faz o *q-bit* assumir um dos estados de acordo com as probabilidades associadas a cada um (PORTUGAL; LAVOR; MACULAN, 2004). Um conjunto de n *q-bits* ($n \geq 1$) após uma medição na base computacional pode assumir um entre 2^n estados possíveis, havendo uma probabilidade associada a cada estado, dada pelo quadrado da amplitude relativa ao estado (WATANABE, 2003). Como o ato de medição, ou *observação*, interfere no sistema quântico, sendo o próprio observador também um sistema quântico, o *q-bit* alvo da observação perde sua condição probabilística e assume um dos estados colapsados ($|0\rangle$ ou $|1\rangle$). Em um sistema quântico real, as probabilidades relativas a cada *q-bit* não podem ser conhecidas em tempo de execução, pois isso significaria observar o *q-bit* e, por conseguinte, cessar a condição quântica. Já em um simulador ou emulador para computação quântica, o acesso às probabilidades é possível, já que se trata de um registro em memória das duas amplitudes representativas do estado do *q-bit*.

2.2.4 Emaranhamento

Um *q-bit* pode relacionar-se com um ou mais *q-bits* de forma a estabelecer uma correlação, chamada *emaranhamento*, e não meramente um agrupamento de *q-bits* estanques (PORTUGAL; LAVOR; MACULAN, 2004). Embora *q-bits* emaranhados não deixem de existir individualmente, o emaranhamento estabelece uma característica *grupal* (OLIVEIRA, 2007) que é de interesse na Computação Quântica, como a codificação superdensa e teleporte (OLIVEIRA; SARTHOU, 2004; RIGOLIN, 2008). A leitura de um *q-bit* emaranhado resulta no *colapso* de correlação, levando-o à assunção solidária de um valor 0 ou 1 (OLIVEIRA, 2007). A condição emaranhada de 2 ou mais *q-bits* não é fatorável, ou seja, não há estados individuais de *q-bits* que por meio de um produto tensorial entre eles resulte no estado grupal. Desta forma, *q-bits* emaranhados são operados por portas apropriadas para atuar sobre o conjunto de *q-bits* (LIMA, A. F. de; LULA JÚNIOR, B., 2006).

2.2.5 Não-clonagem

Diferentemente da computação clássica, o *q-bit* não pode ser copiado impunemente, pois isto representaria uma *medição* e, portanto resultaria na mudança de condição do *q-bit*. Em (WOOTERS; ZUREK, 1982), o Teorema da Não-Clonagem pode ser explicado da seguinte forma: suponha uma máquina com entrada para 2 *q-bits*, sendo o primeiro um estado $|\phi\rangle$ desconhecido e $|v\rangle$ um estado “inerte”, como uma folha de papel virgem na bandeja de alimentação de uma máquina de fotocópia. O estado inicial desta máquina é $|\phi\rangle \otimes |v\rangle$, sendo \otimes a operação

produto tensorial. Deseja-se descobrir um operador unitário T tal que o estado na saída da máquina seja $|\phi\rangle \otimes |\phi\rangle$, ou seja, o segundo q -bit assuma o mesmo estado do primeiro, sem alterar este. O hipotético operador T teria que ser capaz de reproduzir em $|v\rangle$ o estado de $|\phi\rangle$, qualquer que fosse ele. O produto interno usual³ entre $|\phi\rangle$ e $|v\rangle$, $\langle\phi|v\rangle^2$, somente seria útil quando resultasse em 0 ou 1, respectivamente quando os estados forem ortogonais ou iguais. Para qualquer outro resultado, o operador T falharia (WOOTERS; ZUREK, 1982).

2.2.6 Produto tensorial

Quando uma operação quântica tiver que ser realizada sobre um estado com n q -bits, $n \geq 2$, eles precisam ser operados por um produto tensorial, resultando em um vetor-coluna com 2^n linhas (RIEFFEL; POLAK, 2000). O operador quântico para tal conjunto de q -bits é uma matriz quadrada com dimensão $2^n \times 2^n$, o qual pode ser construído a partir de operadores quânticos básicos (dimensão 2×2) (YANOFSKY, 2007). O produto tensorial entre duas matrizes de quaisquer dimensões se dá pelo produto de cada coeficiente da primeira matriz com cada um da segunda (PORTUGAL; LAVOR; MACULAN, 2004; RIEFFEL; POLAK, 2000). Sejam duas matrizes, $A_{m \times n}$ e $B_{p \times q}$, definidas pela Equação 17, cujo produto tensorial resulta em uma matriz C como dimensão $(m.p) \times (n.q)$, da Equação 2.2.6.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}, B = \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1q} \\ b_{21} & b_{22} & \dots & b_{2q} \\ \dots & \dots & \dots & \dots \\ b_{p1} & b_{p2} & \dots & b_{pq} \end{bmatrix} \quad (17)$$

$$C = A \otimes B = \begin{bmatrix} a_{11}.b_{11} & a_{11}.b_{12} & \dots & a_{11}.b_{1q} & \dots & a_{1n}.b_{11} & a_{1n}.b_{12} & \dots & a_{1n}.b_{1q} \\ a_{11}.b_{21} & a_{11}.b_{22} & \dots & a_{11}.b_{2q} & \dots & a_{1n}.b_{21} & a_{1n}.b_{22} & \dots & a_{1n}.b_{2q} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{11}.b_{p1} & a_{11}.b_{p2} & \dots & a_{11}.b_{pq} & \dots & a_{1n}.b_{p1} & a_{1n}.b_{p2} & \dots & a_{1n}.b_{pq} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{m1}.b_{11} & a_{m1}.b_{12} & \dots & a_{m1}.b_{1q} & \dots & a_{mn}.b_{11} & a_{mn}.b_{12} & \dots & a_{mn}.b_{1q} \\ a_{m1}.b_{21} & a_{m1}.b_{22} & \dots & a_{m1}.b_{2q} & \dots & a_{mn}.b_{21} & a_{mn}.b_{22} & \dots & a_{mn}.b_{2q} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{m1}.b_{p1} & a_{m1}.b_{p2} & \dots & a_{m1}.b_{pq} & \dots & a_{mn}.b_{p1} & a_{mn}.b_{p2} & \dots & a_{mn}.b_{pq} \end{bmatrix}$$

O produto tensorial sobre n q -bits inicia-se com os 2 primeiros q -bits e, ao final de todos os produtos tensoriais entre os n q -bits, resulta em vetor-coluna com 2^n linhas (PORTUGAL; LAVOR; MACULAN, 2004; OLIVEIRA; SARTHOU, 2004; VIGNATTI; NETO; BITTENCOURT, 2004).

Nas Equações 18 encontram-se q -bits genéricos que serão multiplicados tensorialmente nos

³Sejam dois vetores $x = (x_1, x_2, x_3, \dots, x_n)$ e $y = (y_1, y_2, y_3, \dots, y_n)$ vetores em \mathbb{R}^n . Produto interno usual é a operação definida por $\langle x, y \rangle = x_1y_1 + x_2y_2 + x_3y_3 + \dots + x_ny_n$ e atende às propriedades da positividade, aditividade, homogeneidade e simetria, cuja descrição não fazem parte do escopo deste trabalho. O Produto interno usual também conhecido como *produto interno canônico* e *produto interno euclidiano*. (THIBES, 2010)

exemplos a seguir.

$$|\psi_0\rangle = \begin{bmatrix} \alpha_0 \\ \beta_0 \end{bmatrix}, |\psi_1\rangle = \begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix}, |\psi_2\rangle = \begin{bmatrix} \alpha_2 \\ \beta_2 \end{bmatrix}, \dots, |\psi_n\rangle = \begin{bmatrix} \alpha_{n-1} \\ \beta_{n-1} \end{bmatrix} \quad (18)$$

Na Equação 19 observa-se o produto tensorial dos dois primeiros q -bits, produzindo um vetor-coluna com 4 linhas (2^2 linhas) (PORTUGAL; LAVOR; MACULAN, 2004).

$$|\psi_{01}\rangle = |\psi_0\rangle \otimes |\psi_1\rangle = \begin{bmatrix} \alpha_0 \\ \beta_0 \end{bmatrix} \otimes \begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} \alpha_0.\alpha_1 \\ \alpha_0.\beta_1 \\ \beta_0.\alpha_1 \\ \beta_0.\beta_1 \end{bmatrix} \quad (19)$$

Continuando, o resultado anterior será multiplicado tensorialmente com o terceiro q -bit, gerando um vetor-coluna com 8 linhas (2^3 linhas) (PORTUGAL; LAVOR; MACULAN, 2004).

$$|\psi_{012}\rangle = |\psi_{01}\rangle \otimes |\psi_2\rangle = \begin{bmatrix} \alpha_0.\alpha_1 \\ \alpha_0.\beta_1 \\ \beta_0.\alpha_1 \\ \beta_0.\beta_1 \end{bmatrix} \otimes \begin{bmatrix} \alpha_2 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} \alpha_0.\alpha_1.\alpha_2 \\ \alpha_0.\alpha_1.\beta_2 \\ \alpha_0.\beta_1.\alpha_2 \\ \alpha_0.\beta_1.\beta_2 \\ \beta_0.\alpha_1.\alpha_2 \\ \beta_0.\alpha_1.\beta_2 \\ \beta_0.\beta_1.\alpha_2 \\ \beta_0.\beta_1.\beta_2 \end{bmatrix}$$

E assim sucessivamente, até o último vetor-coluna construído ser multiplicado com o n -ésimo q -bit, produzindo um vetor-coluna com 2^n linhas, conforme se vê na Equação 20.

$$|\psi_{012\dots n-1}\rangle = |\psi_{012\dots n-2}\rangle \otimes |\psi_{n-1}\rangle = \begin{bmatrix} \alpha_0.\alpha_1.\alpha_2 \dots \alpha_{n-3}.\alpha_{n-2} \\ \alpha_0.\alpha_1.\alpha_2 \dots \alpha_{n-3}.\beta_{n-2} \\ \alpha_0.\alpha_1.\alpha_2 \dots \beta_{n-3}.\alpha_{n-2} \\ \alpha_0.\alpha_1.\alpha_2 \dots \beta_{n-3}.\beta_{n-2} \\ \dots \\ \beta_0.\beta_1.\beta_2 \dots \beta_{n-3}.\beta_{n-2} \end{bmatrix} \otimes \begin{bmatrix} \alpha_{n-1} \\ \beta_{n-1} \end{bmatrix}$$

$$|\psi_{012\dots n-1}\rangle = |\psi_{012\dots n-2}\rangle \otimes |\psi_{n-1}\rangle = \begin{bmatrix} \alpha_0.\alpha_1.\alpha_2 \dots \alpha_{n-2}.\alpha_{n-1} \\ \alpha_0.\alpha_1.\alpha_2 \dots \alpha_{n-2}.\beta_{n-1} \\ \alpha_0.\alpha_1.\alpha_2 \dots \beta_{n-2}.\alpha_{n-1} \\ \alpha_0.\alpha_1.\alpha_2 \dots \beta_{n-2}.\beta_{n-1} \\ \dots \\ \beta_0.\beta_1.\beta_2 \dots \beta_{n-2}.\beta_{n-1} \end{bmatrix} \quad (20)$$

Na construção de um operador para 2 ou mais q -bits, inicia-se o produto tensorial entre 2 operadores quânticos básicos 2×2 . Nas Equações 21 encontram-se dois operadores básicos hipotéticos A e B :

$$A = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}, B = \begin{bmatrix} b_1 & b_2 \\ b_3 & b_4 \end{bmatrix} \quad (21)$$

O produto tensorial AB_{\otimes} entre A e B , resulta em um operador para 2 q -bits como o da Equação 22.

$$AB_{\otimes} = A \otimes B = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \otimes \begin{bmatrix} b_1 & b_2 \\ b_3 & b_4 \end{bmatrix}$$

$$AB_{\otimes} = \begin{bmatrix} a_1.b_1 & a_1.b_2 & a_2.b_1 & a_2.b_2 \\ a_1.b_3 & a_1.b_4 & a_2.b_3 & a_2.b_4 \\ a_3.b_1 & a_3.b_2 & a_4.b_1 & a_4.b_2 \\ a_3.b_3 & a_3.b_4 & a_4.b_3 & a_4.b_4 \end{bmatrix} \quad (22)$$

No caso de um operador para 3 q -bits, realiza-se o produto tensorial $ABC_{\otimes} = A \otimes B \otimes C = AB_{\otimes} \otimes C$, resultando na Equação 23.

$$ABC_{\otimes} = AB_{\otimes} \otimes C = \begin{bmatrix} a_1b_1 & a_1b_2 & a_2b_1 & a_2b_2 \\ a_1b_3 & a_1b_4 & a_2b_3 & a_2b_4 \\ a_3b_1 & a_3b_2 & a_4b_1 & a_4b_2 \\ a_3b_3 & a_3b_4 & a_4b_3 & a_4b_4 \end{bmatrix} \otimes \begin{bmatrix} c_1 & c_2 \\ c_3 & c_4 \end{bmatrix}$$

$$ABC_{\otimes} = \begin{bmatrix} a_1b_1c_1 & a_1b_1c_2 & a_1b_2c_1 & a_1b_2c_2 & a_2b_1c_1 & a_2b_1c_2 & a_2b_2c_1 & a_2b_2c_2 \\ a_1b_1c_3 & a_1b_1c_4 & a_1b_2c_3 & a_1b_2c_4 & a_2b_1c_3 & a_2b_1c_4 & a_2b_2c_3 & a_2b_2c_4 \\ a_1b_3c_1 & a_1b_3c_2 & a_1b_4c_1 & a_1b_4c_2 & a_2b_3c_1 & a_2b_3c_2 & a_2b_4c_1 & a_2b_4c_2 \\ a_1b_3c_3 & a_1b_3c_4 & a_1b_4c_3 & a_1b_4c_4 & a_2b_3c_3 & a_2b_3c_4 & a_2b_4c_3 & a_2b_4c_4 \\ a_3b_1c_1 & a_3b_1c_2 & a_3b_2c_1 & a_3b_2c_2 & a_4b_1c_1 & a_4b_1c_2 & a_4b_2c_1 & a_4b_2c_2 \\ a_3b_1c_3 & a_3b_1c_4 & a_3b_2c_3 & a_3b_2c_4 & a_4b_1c_3 & a_4b_1c_4 & a_4b_2c_3 & a_4b_2c_4 \\ a_3b_3c_1 & a_3b_3c_2 & a_3b_4c_1 & a_3b_4c_2 & a_4b_3c_1 & a_4b_3c_2 & a_4b_4c_1 & a_4b_4c_2 \\ a_3b_3c_3 & a_3b_3c_4 & a_3b_4c_3 & a_3b_4c_4 & a_4b_3c_3 & a_4b_3c_4 & a_4b_4c_3 & a_4b_4c_4 \end{bmatrix} \quad (23)$$

Portanto, um operador para n q -bits, $n \geq 3$, é construído por meio do produto tensorial do operador para $n - 1$ q -bits com um operador quântico básico. Logo, a construção de um operador quântico para n q -bits necessariamente requer os produtos tensoriais entre dois operadores quânticos básicos seguidos pelos produtos tensoriais de um operador para m q -bits, $m = \{3, \dots, n-1\}$, com um operador quântico básico. Quando se trata de operação quântica sobre dois ou mais q -bits, o produto tensorial entre operadores básicos e entre os q -bits participantes deve ser realizado previamente, executando-se por fim o produto matricial entre o operador e o vetor-coluna assim preparados. O produto tensorial possui a propriedade mostrada na Equação 24.

$$(A \otimes B).(u \otimes v) = (Au) \otimes (Bv) \quad (24)$$

onde A e B são operadores e u e v vetores-coluna. Para operações que não causem emaranhamento e nem envolvam q -bits emaranhados, e enquanto não for requisitada medição do estado quântico, o uso desta propriedade evita a construção do operador para os n q -bits, construção do vetor-coluna com os n q -bits e na fatoração do resultado, de forma a gravar na memória de estado quântico os novos dados de cada q -bit participante, já que os mesmos permanecem não-emaranhados. Quando da requisição da medição do estado quântico do Coprocessador, o produto tensorial entre todos os q -bits seria realizado.

2.2.7 Operadores quânticos

Operadores quânticos realizam operações sobre um ou mais q -bits. Possuem quantidade de entradas igual à de saídas, mantendo a igualdade entre a energia presente nas entradas e nas saídas, não ocorrendo dissipação de calor, portanto. Permitem conhecer as condições das entradas, pois estas informações são preservadas. Sendo reversíveis (WATANABE, 2003; RIEFFEL; POLAK, 2000), possibilitam retornar o sistema ao estado anterior. Operadores quânticos têm representação matricial como operadores unitários (WATANABE, 2003; OLIVEIRA, 2007). Operadores quânticos básicos possuem dimensão 2×2 e processam 1 q -bit, com exceção do operador CNOT (Subseção 2.2.7.8), cuja dimensão é 4×4 e atua sobre 2 q -bits, e operador Toffoli (Subseção 2.2.7.9), com dimensão $2^i \times 2^i$, onde i é a quantidade de entradas de controle mais 1. Operadores quânticos podem atuar sobre 2 ou mais q -bits simultaneamente, alguns sendo construídos por meio de produto tensorial entre operadores. Desta forma, alguns operadores quânticos podem ser construídos a partir de produto(s) tensorial(is) entre operadores quânticos básicos. Todos têm a característica de reversibilidade de estado, isto é, a partir de um certo estado quântico, pode-se voltar ao estado anterior pela aplicação do operador inverso do último operador utilizado (OMER, 2000). Eventualmente, alguns operadores poderão ser auto-adjuntos ⁴, como o operador *Hadamard*, descrito na Subseção 2.2.7.4. Como a medição leva o conjunto de q -bits a assumir valores colapsados irremediavelmente, finda-se a reversibilidade de estados. Como qualquer operação sobre um vetor equivale a operar sobre a combinação linear da base deste mesmo vetor, uma operação quântica sobre um certo estado opera simultaneamente sobre todos os q -bits que compõem o sistema (THIBES, 2010). Ao longo desta seção serão exemplificadas operações sobre um ou mais q -bits. Além da representação como combinação linear (Equação 5), um q -bit é representável como um vetor-coluna com duas linhas, como se vê na Equação 25.

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (25)$$

2.2.7.1 Operador NOT quântico ou X

Considerando o sistema esférico de coordenada da Figura 1, o operador NOT quântico promove uma rotação de 180 graus em torno do eixo x , invertendo as amplitudes associadas aos vetores da base. Se aplicado a um q -bit em estado colapsado, resulta no outro estado colapsado, como no caso de uma porta NOT clássica (PORTUGAL; LAVOR; MACULAN, 2004)

⁴Operador adjunto T^* de um operador T é aquele que satisfaz à condição $\langle T(u), v \rangle = \langle u, T^*(v) \rangle$. O operador adjunto é a matriz transposta conjugada do operador, assim $T^* = \overline{T}^T$ (WATANABE, 2003). Operador auto-adjunto é aquele em que $T = T^*$, ou seja, a adjunta do operador é o próprio operador.

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad (26)$$

Assim

$$X|v\rangle = \beta|0\rangle + \alpha|1\rangle$$

Por exemplos, aplicando o operador X sobre dois estados colapsados, vemos na Equação 27:

$$X|0\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle \quad X|1\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle \quad (27)$$

2.2.7.2 Operador Y

Com base no sistema esférico de coordenada da Figura 1, o operador Y realiza uma rotação de 180 graus em torno do eixo y (PORTUGAL; LAVOR; MACULAN, 2004).

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad (28)$$

Aplicando-se o operador Y ao $|v\rangle$, temos na Equação 29.

$$Y|v\rangle = i(-\beta|0\rangle + \alpha|1\rangle) \quad (29)$$

2.2.7.3 Operador Z

Considerando o sistema esférico de coordenada da Figura 1, o operador Z promove uma rotação de 180 graus em torno do eixo z (PORTUGAL; LAVOR; MACULAN, 2004).

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad (30)$$

Na Equação 31, vê-se a operação sobre $|v\rangle$.

$$Z|v\rangle = \alpha|0\rangle - \beta|1\rangle \quad (31)$$

2.2.7.4 Operador Hadamard ou H

O operador Hadamard transforma um q -bit no estado colapsado ($|0\rangle$ ou $|1\rangle$) em uma superposição de ambos estados com igual amplitude (PORTUGAL; LAVOR; MACULAN, 2004; RIEFFEL; POLAK, 2000). Este é um operador auto-adjunto (YANOFKY, 2007).

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (32)$$

$$H|0\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}, H|1\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \end{bmatrix}$$

A aplicação do operador Hadamard no vetor que sofreu uma operação Hadamard volta ao estado original, conforme Equação 33

$$H(H.|1\rangle) = H. \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (33)$$

2.2.7.5 Operador de fase ou S

A matriz correspondente ao operador S está apresentada na Equação 34, onde i é a unidade imaginária $i^2 = -1$ (PORTUGAL; LAVOR; MACULAN, 2004).

$$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \quad (34)$$

Assim, aplicando-se o operador S ao $|v\rangle$, temos a Equação 35

$$S|v\rangle = \alpha|0\rangle + i\beta|1\rangle \quad (35)$$

A execução de um operador S preserva as probabilidades de se obter os estados $|0\rangle$ ou $|1\rangle$, o que não aconteceria com um operador Hadamard, por exemplo (PORTUGAL; LAVOR; MACULAN, 2004).

2.2.7.6 Operador $\frac{\pi}{8}$ ou T

O operador T realiza o deslocamento da fase relativa do ket $|1\rangle$ (PORTUGAL; LAVOR; MACULAN, 2004).

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{bmatrix} \quad (36)$$

Assim, aplicando-se o operador T ao $|v\rangle$, temos a Equação 37.

$$T|v\rangle = \alpha|0\rangle + \frac{1}{\sqrt{2}}\beta(1+i)|1\rangle \quad (37)$$

2.2.7.7 Operador Identidade ou I

Este operador preserva o estado do q -bit, como ocorre com um operador identidade convencional (PORTUGAL; LAVOR; MACULAN, 2004; MARQUEZINO, 2006).

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (38)$$

2.2.7.8 Operador CNOT

O operador CNOT, acrônimo de *Controlled NOT*, é uma dos principais na computação quântica, pois tem a capacidade de emaranhar q -bits e seu *modus operandi* é extensível a outros operadores quânticos. (PORTUGAL; LAVOR; MACULAN, 2004; PORTUGAL; COSME; GONÇALVES, 2006). Este operador possui dois argumentos: o q -bit de controle e o q -bit alvo. O operador CNOT inverte o estado do q -bit alvo quando o q -bit de controle está no estado 1. A magnitude da inversão depende do estado do q -bit de controle. Pode-se dizer que o operador CNOT é um operador NOT dependente de um segundo q -bit (OMER, 2009). A representação matricial do operador CNOT é mostrada na Equação 39 e sua tabela-verdade é apresentada na Tabela 3, onde a coluna a representa o q -bit de controle e a coluna b o q -bit-alvo antes da operação. As colunas a' e b' referem-se, respectivamente, ao q -bit de controle e ao q -bit-alvo após a operação.

Tabela 3: Tabela-verdade para o operador CNOT

a	b	a'	b'
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (39)$$

2.2.7.9 Operador Toffoli

O operador Toffoli utiliza três q -bits: dois q -bits para controle e um q -bit alvo. O operador Toffoli inverte o estado do q -bit alvo quando os dois q -bits de controle estão no estado $|1\rangle$. O operador Toffoli pode ser visto como uma extensão da porta CNOT. A tabela-verdade deste operador encontra-se na Tabela 4 (SILVA, 2002). a e b são os q -bits de controle e c o q -bit alvo. a' , b' e c' são as saídas (PORTUGAL; LAVOR; MACULAN, 2004).

$$Toffoli = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad (40)$$

Tabela 4: Tabela-verdade para o operador Toffoli com 2 q -bits de controle

a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

2.2.7.10 Operador Swap

O operador Swap troca o estado quântico entre 2 q -bits (PORTUGAL; COSME; GONÇALVES, 2006).

$$Swap = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (41)$$

$$Swap|a, b\rangle = |b, a\rangle$$

O operador Swap pode ser construído com o auxílio de 3 operadores CNOT encadeados (BARROS, 2011), o segundo operador CNOT tendo como q -bit alvo e q -bit de controle o q -bit de controle e q -bit alvo do primeiro operador, respectivamente. Analogamente, o terceiro operador CNOT utiliza como q -bit alvo e q -bit de controle o q -bit de controle e o q -bit alvo do segundo operador. Nas Equações 42, 43 e 44 mostram-se o progresso dos estados no arranjo com três operadores CNOT. O operador \oplus é uma soma com módulo 2.

$$CNOT_1|a, b\rangle = |a, a \oplus b\rangle \quad (42)$$

$$CNOT_2|a \oplus b, a\rangle = |a \oplus b, (a \oplus b) \oplus a\rangle \quad (43)$$

$$CNOT_3|(a \oplus b) \oplus a, a \oplus b\rangle = |(a \oplus b) \oplus a, [(a \oplus b) \oplus a] \oplus (a \oplus b)\rangle \quad (44)$$

O circuito de um operador Swap construído com 3 operadores CNOT está apresentado na Figura 2.5(a) (MARQUEZINO, 2006). A Figura 2.5(b) apresenta legendas que auxiliam na compreensão da mudança de estados que cada operador CNOT realiza. As legendas $a..a$, $b'..b'$ e $a'..a'$, relativas aos q -bits de controle dos operadores CNOT, indicam que o estado não foi modificado. As legendas $a..a'$, $b..b'$ e $b'..b''$, associadas ao q -bit-alvo dos operadores CNOT, mostram que o estado é passível de alteração.

A Tabelas 5, 6 e 7 mostram as entradas e saídas dos operador $CNOT_1$, $CNOT_2$ e $CNOT_3$, respectivamente. Pode-se notar que, comparando-se as duas primeiras colunas da

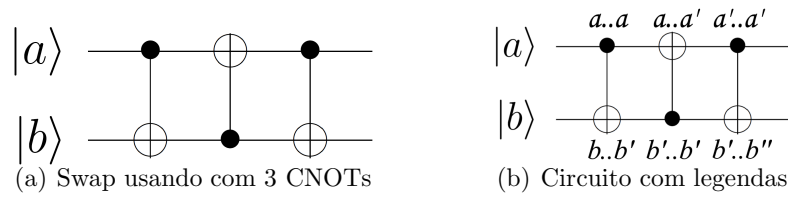


Figura 5: Circuito de um operador Swap construído com 3 operadores CNOT

Tabela 5 com a primeira e a terceira coluna da Tabela 7, os estados entre os q -bits a e b foram trocados entre si.

Tabela 5: Estados nas entradas e saídas do operador Swap: CNOT₁

controle	alvo	saída
a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Tabela 6: Estados nas entradas e saídas do operador Swap: CNOT₂

controle	alvo	saída
$a \oplus b$	a	$(a \oplus b) \oplus a$
0	0	0
1	0	1
1	1	0
0	1	1

Tabela 7: Estados nas entradas e saídas do operador Swap: CNOT₃

controle	alvo	saída
$(a \oplus b) \oplus a$	$a \oplus b$	$[(a \oplus b) \oplus a] \oplus (a \oplus b)$
0	0	0
1	1	0
0	1	1
1	0	1

2.2.7.11 Operador Fredkin ou CSwap

O operador Fredkin ou CSwap, acrônimo de *Controlled Swap*, troca o estado quântico entre 2 q -bits em função de um q -bit de controle. A tabela-verdade apresenta-se na Tabela 8, sendo a e b os q -bits que terão seus estados trocados entre si e c o q -bit de controle. a' , b' e c' são as saídas (LEE; HUANG; ZHU, 2010; RIEFFEL; POLAK, 2000). A representação matricial deste

Tabela 8: Tabela verdade para o operador Fredkin

a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	1	1

operador vê-se na Equação 45.

$$CSwap = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (45)$$

2.3 Considerações finais do capítulo

Neste capítulo apresentou-se aspectos básicos da computação quântica a fim de formar um arcabouço mínimo para a compreensão dos capítulos vindouros, em especial a arquitetura, memória de estados e unidade de cálculo. O produto tensorial é a operação mais utilizada no curso de um algoritmo quântico, conforme mais q -bits vão emaranhando-se entre si. Embora o produto tensorial seja uma operação-meio em uma operação quântica, possui importância significativa devido à quantidade de multiplicações com números complexos que demanda e que aumenta exponencialmente com a quantidade de q -bits operados, requerendo espaço em memória de rascunho, gerenciamento de mais de uma memória e de outros componentes necessários ao controle.

Capítulo 3

ARQUITETURA DO COPROCESSADOR

NESTE capítulo serão apresentados a macro-arquitetura do Coprocessador e seus componentes, seus interrelacionamentos e funcionalidades principais.

3.1 Visão geral

O Coprocessador em discussão implementa um emulador de máquina quântica capaz de realizar operações sobre um conjunto de q -bits. Trata-se de um sistema isolado e que interage com um processador que lhe envia instruções formatadas que especificam operações quânticas sobre um ou mais q -bits. Sendo uma operação para 2 ou mais q -bits, o operador quântico é construído em tempo de execução. O estado quântico permanece hermético na máquina quântica até que seja solicitada pelo processador principal (PROC) a leitura do estado da mesma, resultando no *estado colapsado* de cada q -bit.

3.2 Macro-arquitetura do coprocessador

Como pode ser visto na Figura 6, o Coprocessador de Operações Quânticas (COPROC) comunica-se com o processador PROC por meio de um canal *half-duplex*, configuração esta escolhida por considerar a característica sequencial do algoritmo e, portanto, a potencial dependência do resultado de uma operação anterior. As operações quânticas são solicitadas por PROC por meio de *instruções*, blocos descritivos que especificam a operação quântica básica e o q -bit-alvo. Caso a operação seja sobre dois ou mais q -bits, dois ou mais descritores são requeridos para especificar a operação e o q -bits-alvo. Quando solicitado pelo processador PROC, o coprocessador COPROC realiza a leitura do estado quântico.

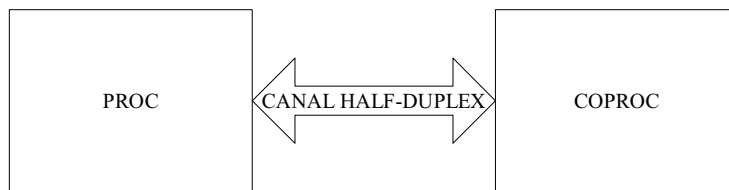


Figura 6: Comunicação entre processador e coprocessador

A macro-arquitetura do coprocessador está apresentada na Figura 7.

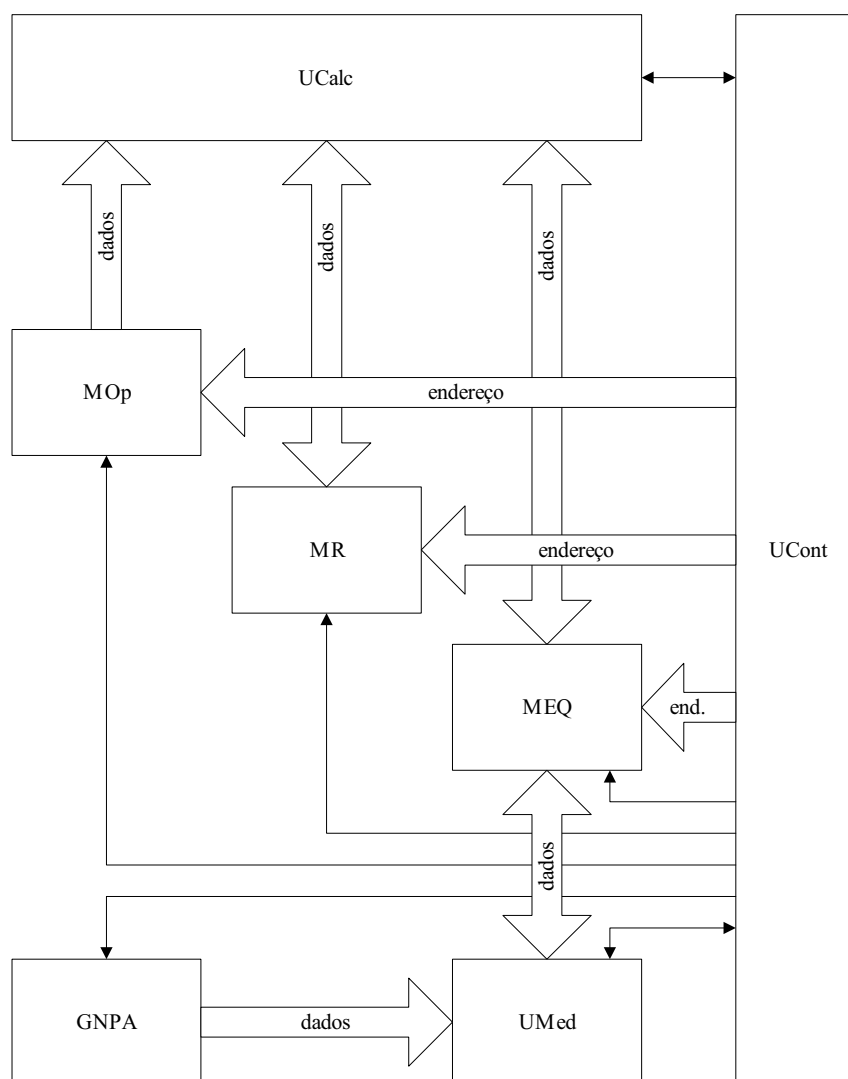


Figura 7: Macro-arquitetura do coprocessador

A unidade de controle UCont gerencia, por meio de um microprograma e componentes auxiliares, os demais elementos da arquitetura. UCont registra, decodifica e interpreta as instruções que contém o código da operação quântica corrente e o(s) q -bit(s)-alvo. A memória MEQ, do tipo *read-write*, guarda o estado da máquina quântica. MOp é a memória *read-only*

destinada aos coeficientes dos operadores básicos. MR é memória de rascunho do tipo *read-write* e serve para armazenar os coeficientes dos operadores quânticos para dois ou mais *q-bits*, calculados a partir dos operadores quânticos básicos. A unidade de cálculo UCalc realiza o produto tensorial entre *q-bits*, produto tensorial entre operadores básicos e produto tensorial entre operador calculado e operador básico e o produto matricial entre operador e *q-bit(s)*. A unidade UMed realiza a medição do estado quântico com o auxílio do gerador de números pseudo-aleatórios GNPA. Quando solicitado por PROC, UCont fornece o estado quântico do Coprocessador, resultante da medição realizada por UMed.

3.2.1 Memória de estado quântico

A memória de estado quântico MEQ é *read-write* e *dual-port*, com restrição para leitura e escrita simultânea no mesmo endereço. É composta por duas partes: a memória de estado de *q-bits* MQ e a memória de controle de *q-bits* MCQ. Ambas têm a mesma quantidade de endereços, sendo seus conteúdos vinculados entre si para cada endereço, como se uma fosse extensão da outra. Tal separação deveu-se a possibilidade de apenas uma destas memórias serem gravadas em certas situações. A quantidade de endereços de MEQ atende à necessidade de representar todos os estados possíveis da máquina quântica. Os primeiros *ad* endereços são reservados ao armazenamento dos valores dos kets dos *q-bits*, ainda não emaranhados ou já emaranhados. Os $2^{ad} - ad$ endereços restantes são destinados ao armazenamento dos dados do vetor-coluna que representa um conjunto de *q-bits* emaranhados, complementando o conjunto de coeficientes do vetor-coluna nos *ad* endereços iniciais, quando o *q-bit* está emaranhado. A memória MEQ está organizada de forma a comportar em cada endereço os 2 kets de cada *q-bit* não emaranhado da máquina quântica, mais os registros complementares para descrever as amplitudes de cada estado possível relativo ao conjunto de 2 ou mais *q-bits* emaranhados. Em cada endereço de MQ são guardados os dois coeficientes do *q-bit*, assim. Na Figura 8, o parâmetro *ad*, referenciado nos sinais *addr rd* e *addr wr*, indica a quantidade de bits do barramento de endereços, respectivamente de leitura e de gravação, cujo valor é igual ao número de *q-bits* da máquina quântica. O parâmetro *dad* é igual à quantidade de endereços de MQ e MCQ, ou seja $dad = 2^{ad}$. A condição não-emaranhada de um *q-bit* é potencialmente transitória em um algoritmo quântico. Assim, o endereço originalmente destinado ao *q-bit* não-emaranhado, após o emaranhamento, será utilizado para armazenamento de dois dos coeficientes do vetor-coluna que representa o conjunto de *q-bits* emaranhados ao qual pertence, um relativo à linha ímpar, outro à par. Estas referências “ímpar” e “par” serão úteis na compreensão do funcionamento de UCalc quando

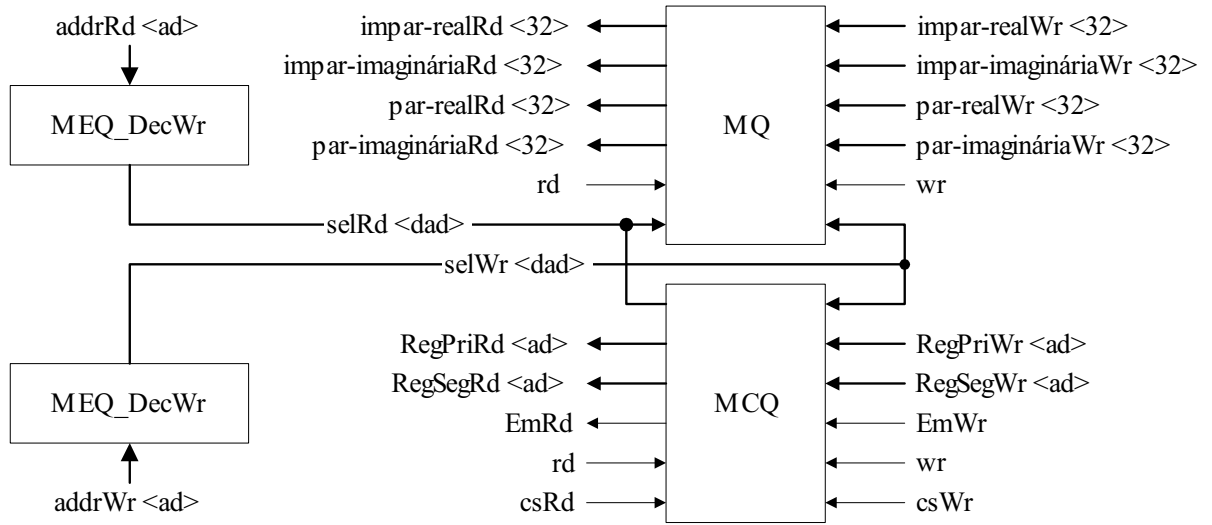


Figura 8: Interface da memória de estado quântico

realizando produtos tensoriais entre q -bits e produto matricial entre um operador quântico e registro de q -bits.

3.2.1.1 Memória de q -bits

A memória MQ possui dois barramentos de dados com 128 bits cada, representando no formato IEEE754 com precisão simples (WAN; BREWER, 2003), dois números complexos, 32 bits para a parte real e 32 bits para a parte imaginária de cada um. A memória MQ está apresentada na Figura 9. Conforme ilustrado na Figura 10 cada endereço de MQ armazena a parte

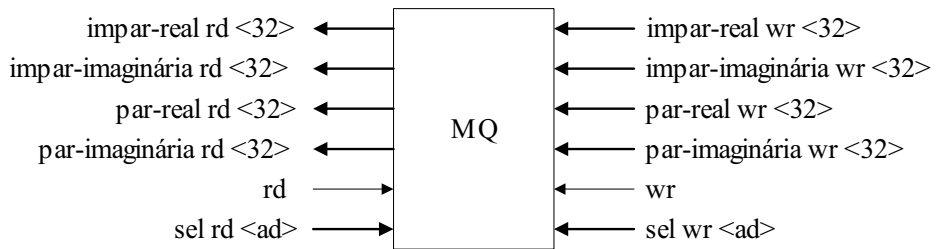


Figura 9: Interface da memória de q -bits

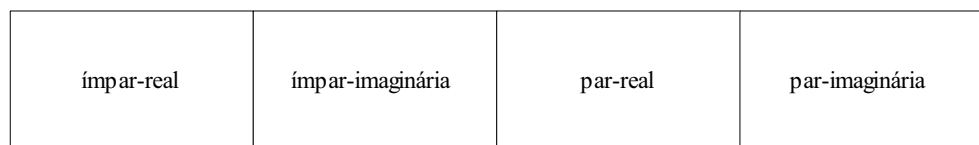


Figura 10: Palavra da memória MQ - parte dos coeficientes

real e a imaginária do ket $|0\rangle$ (32 bits cada) e a parte real e a imaginária do ket $|1\rangle$ (32 bits

cada). Quando de um *reset* da máquina quântica, cada posição de memória MQ é inicializado de forma que o *ket* $|0\rangle$ tenha na parte real o valor 1,000 e 0,000 na parte imaginária, ambos no padrão IEEE754, enquanto que o *ket* $|1\rangle$ possua valor 0,000 (IEEE754) nas partes real e imaginária (OMER, 2000). A quantidade de endereços de MEQ comporta as duplas de coeficientes necessárias à representação do vetor-coluna com todos os *q-bits* emaranhados entre si. Na Figura 11 ilustra-se o diagrama de tempos de leitura e gravação simultânea na memória MQ. Inicialmente, os endereços 0 a 3 não estão inicializados. Os sinais de controle *rd* e *wr* são ativos quando no nível alto. O sinal *rd* é mantido ativo durante todo o diagrama para demonstrar a correção do funcionamento da memória quando há ou não escrita. Realiza-se a gravação (sinal *wr* ativo) do valor 65 no endereço 1 e depois o valor 85 no endereço 2, valores estes que são obtidos aos serem relidos tais endereços. O endereço 3, não tendo sido objeto de gravação em nenhuma vez, apresenta-se com valor não inicializado.

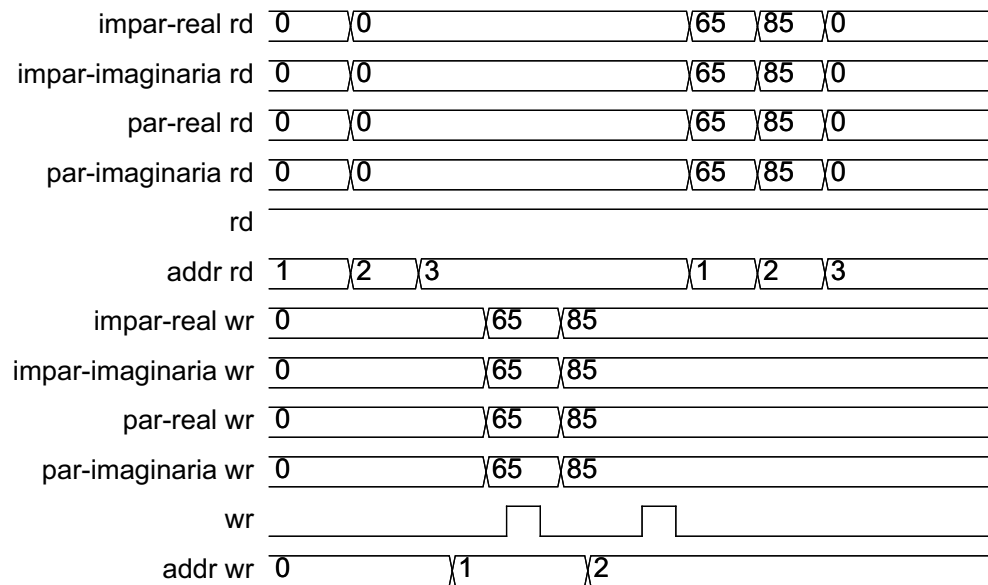


Figura 11: Diagrama de tempos de leitura e gravação simultâneas em MQ

3.2.1.2 Memória de controle de *q-bits*

A memória MCQ, apresentada na Figura 13, serve ao controle de emaranhamento de *q-bits*. *Q-bits* emaranhados são representados por uma quantidade de posições de memória igual a 2^{n-1} , sendo n a quantidade de *q-bits* emaranhados entre si. Em uma máquina quântica pode ocorrer, em dado instante, a existência dois ou mais conjuntos de *q-bits* emaranhados, porém sem emaranhamento entre conjuntos. Está mostrada na Figura 12 o formato da palavra de MCQ. Os campos *RegPr* e *Regseg* fazem parte de uma lista parcialmente encadeada, ou encadeada apenas em um sentido, que é utilizada no controle de emaranhamento de *q-bits*,

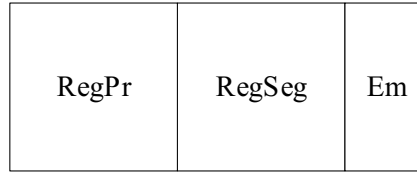


Figura 12: Palavra da memória MQ - parte do controle

sendo tais campos utilizados somente quando o campo *Em* contiver valor 1. O campo *RegPr* armazena o endereço do *q-bit* inicial do registro de *q-bits* emaranhados. O campo *Regseg* informa o endereço do *q-bit* seguinte ao endereço corrente. A necessidade de formar uma lista parcialmente encadeada, e não encadeada, deve-se à inexistência da necessidade de conhecer o registro imediatamente anterior, já que toda operação quântica que se fizer sobre um conjunto de *q-bits* emaranhados deverá iniciar-se com os coeficientes do primeiro endereço do conjunto, obrigatoriamente. Se o *q-bit* for o primeiro dos *q-bits* emaranhados no conjunto ao qual pertence, o campo *RegPr* contém seu próprio endereço. Se o *q-bit* for o último da sequência de *q-bits* do registro, é armazenado o próprio endereço. Os campos *RegPr* e *Regseg* têm dimensão $\lceil \log_2 n \rceil$ bits, onde n é a quantidade de *q-bits* do coprocessador. O campo *Em* comporta 1 bit. Os *q-bits* emaranhados requerem a alocação de 2^e (e , a quantidade de *q-bits* emaranhados) posições de memória porque formam registros resultantes de produto tensorial entre eles. Como uma operação quântica dá-se sobre conjunto de *q-bits*, é necessário conhecer-se o número do primeiro registro, já que a operação quântica (entenda-se produto matricial) iniciar-se-á pela primeira linha do vetor-coluna que representa os *q-bits* emaranhados. O campo *Em* indica se o *q-bit* está emaranhado (valor 1) ou não (valor 0) com outro. A operação quântica será descrita por tantas instruções quantos forem os *q-bits* participantes da operação, sendo a última instrução identificada pelo campo *T*. Quando de um *reset* da máquina quântica, o campo *Em* de cada posição da memória MCQ é inicializado com valor 0, enquanto que os outros campos não requerem colocação de algum valor. Na Figura 13, os sinais *RegPr*, *RegSeg* e *Em* guardam

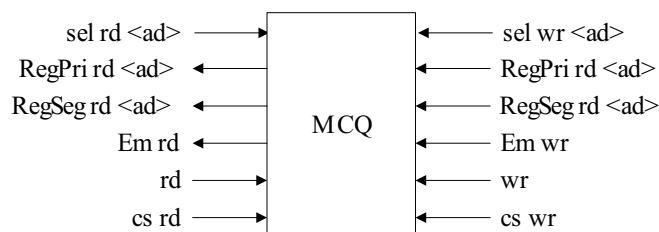


Figura 13: Memória de controle de *q-bits*

correspondência direta com os campos de mesmo nome da palavra de MCQ. O parâmetro

ad referenciado nos sinais $RegPr$ e $RegSeg$ indica a quantidade de bits do barramento de endereços, respectivamente de leitura e de gravação. A Figura 14 mostra um teste de leitura e gravação simultânea na memória MCQ.

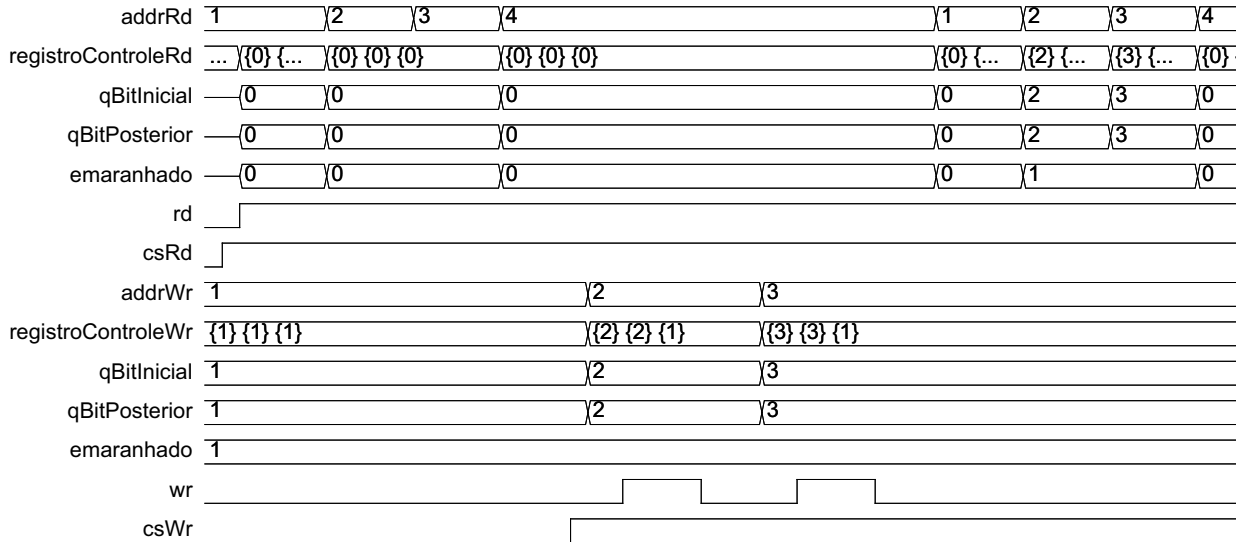


Figura 14: Diagrama de tempos de leitura e gravação simultâneas em MCQ

Os campos $registroControleRd$ e $registroControleWr$ são do tipo registro com 3 campos: $qBitInicial$, $qBitPosterior$ e $emaranhado$. Os sinais $addrRd$ e $addrWr$ determinam o endereço de leitura e de gravação, respectivamente. Os sinais $csRd$ e $csWr$ são $chipSelect$ de leitura e escrita, respectivamente. O sinal rd habilita leitura e o sinal wr , a gravação. Em um período inicial, vê que as posições de memória 1 a 4 encontram-se sem dados gravados no $registradorControleRd$. A seguir executa-se gravação do endereço 2 com valor 0x2 nos campos $qBitInicial$ e $qBitPosterior$ e valor 0x1 no campo $emaranhado$. Após o quê, grava-se no endereço 3 com o valor 0x3 nos dois primeiros campos de $registroControleWr$, mantido o valor 0x1 no campo $emaranhado$. Por fim, leem-se os endereços 1 a 4, comprovando a gravação dos conteúdos somente nos endereços 2 e 3. São usados dois decodificadores, MEQ_DecRd e MEQ_DecWr , respectivamente o decodificador de endereço para a leitura e gravação de MQ e MCQ (vide Figura 8). Cada um destes decodificadores possui n bits na entrada e 2^n bits na saída, sendo n a quantidade de q -bits na máquina quântica.

3.2.2 Memória de operadores

A memória de operadores (MOp) armazena os coeficientes dos operadores quânticos básicos que, conforme Subseção 2.2.7, representável por uma matriz quadrada 2×2 , exceto o operador CNOT, uma matriz quadrada 4×4 . A memória MOp é somente de leitura com barramento

de dados de 128 bits, cada 64 bits representando um número complexo no formato IEEE754 de precisão simples, sendo 32 bits contíguos para a parte real e outros 32 para a imaginária. Cada endereço de MOp compreende dois coeficientes de uma linha, coluna ímpar e coluna par. Portanto, 2 endereços contíguos de MOp armazenam os coeficientes de um operador quântico básico, um endereço contendo os coeficientes da linha ímpar e o endereço seguinte com os da par. Os bits 0 ... 31, 32 ... 63, 64 ... 95 e 96 ... 127 respectivamente estão associados à parte real do coeficiente da coluna ímpar, parte imaginária do coeficiente da coluna ímpar, parte real do coeficiente da coluna par e parte imaginária do coeficiente da coluna par. A Figura 15 ilustra os sinais desta memória.

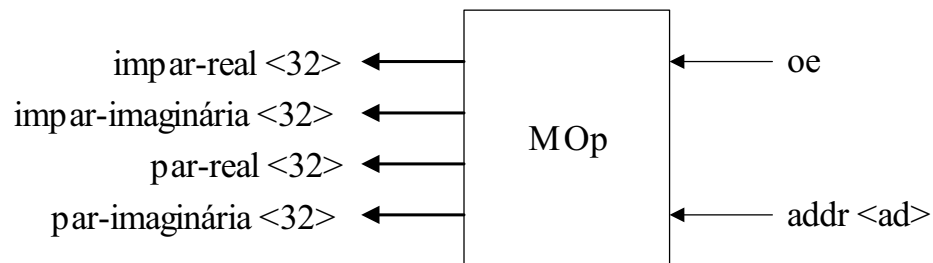


Figura 15: Interface da memória de operadores

Este arranjo que estabelece a guarda de dois números complexos por endereço, os dois coeficientes contíguos de uma mesma linha do operador, está em acordo compatibiliza-se com o arranjo da memória MQ, na qual guardam-se dois kets do q -bit por endereço. Desta maneira, os coeficientes de um operador são lidos aos pares (coluna ímpar e coluna par de uma mesma linha do operador quântico), de forma que cada um abastece uma das entradas de cada um dos multiplicadores de número complexo, objeto da Subseção 4.4 do Capítulo 4. A memória MOp contém os coeficientes dos seguintes operadores: I, X, Y, Z, H, S, T e CNOT, e os endereços que seus coeficientes ocupam são mostrados na Tabela 9. As instruções passadas pelo processador PROC contém em seu campo OP um código que representa o endereço inicial da área ocupada pelo operador quântico. Os operadores quânticos são referenciados no programa quântico por um código único que é associado ao endereço inicial da faixa reservada ao operador quântico por meio de uma *look-up table*. Os endereços subsequentes são sequencialmente acessados quando da execução do produto tensorial ou matricial. Na Figura 16 ilustra-se um teste de leitura da memória MOp nos endereços 0 e 1, respectivamente contendo os coeficientes da primeira e segunda linha do operador identidade. O valor hexadecimal 0x3F800000 e o 0x00000000 correspondem aos valores 1,000 e 0,000 no formato IEEE754. A parte real do coeficiente (1,1) e (2,2) da matriz identidade possui valor 1,000, enquanto que a parte imaginária de todos os

Tabela 9: Ocupação dos endereços da memória MOp

ENDEREÇO	COLUNA ÍMPAR	COLUNA PAR	OPERADOR
21	1	0	CNOT
20	0	0	CNOT
19	0	1	CNOT
18	0	0	CNOT
17	0	0	CNOT
16	0	1	CNOT
15	0	0	CNOT
14	1	0	CNOT
13	0	$e^{\frac{i\pi}{4}}$	T
12	1	0	T
11	1	0	S
10	0	i	S
9	$\frac{1}{\sqrt{2}} * (1 + 0i)$	$\frac{1}{\sqrt{2}} * (0 - i)$	H
8	$\frac{1}{\sqrt{2}} * (1 + 0i)$	$\frac{1}{\sqrt{2}} * (1 + 0i)$	H
7	0	-1	Z
6	1	0	Z
5	$-i$	0	Y
4	0	$-i$	Y
3	1	0	X
2	0	1	X
1	0	1	I
0	1	0	I

coeficientes e a parte real dos coeficiente (1,2) e (2,1) são iguais a 0,000. O barramento de dados de MOp é compartilhado com o barramento de dados de leitura da memória MR.

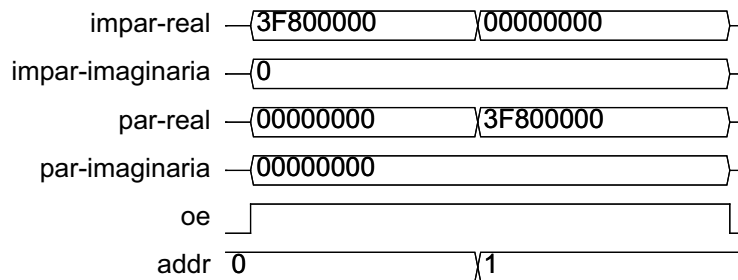


Figura 16: Diagrama de tempos de leitura de MOp

3.2.3 Memória de rascunho

A memória de rascunho (MR) destina-se ao armazenamento dos coeficientes de operadores quânticos para 2 ou mais q -bits, construídos em tempo de execução por meio de produto tensorial de operadores quânticos básicos. A memória MR é *read-write* e *dual-port*, com restrição para leitura e escrita simultânea no mesmo endereço, e é apresentada na Figura 17. Possui

barramento de dados com 128 bits, onde cada 64 bits formam as partes real e imaginária de um número complexo no formato IEEE754: Os bits 0 ... 31, 32 ... 63, 64 ... 95 e 96 ... 127 respectivamente estão associados à parte real do coeficiente da coluna ímpar, parte imaginária do coeficiente da coluna ímpar, parte real do coeficiente da coluna par e parte imaginária do coeficiente da coluna par. A memória MR compartilha o barramento de dados com a memória MOp. Os barramentos de endereço de leitura e de gravação são, respectivamente, *addr rd* e *addr wr*, que são decodificados por *MR_DecRd* e *MR_DecWr*. Analogamente a MOp, cada

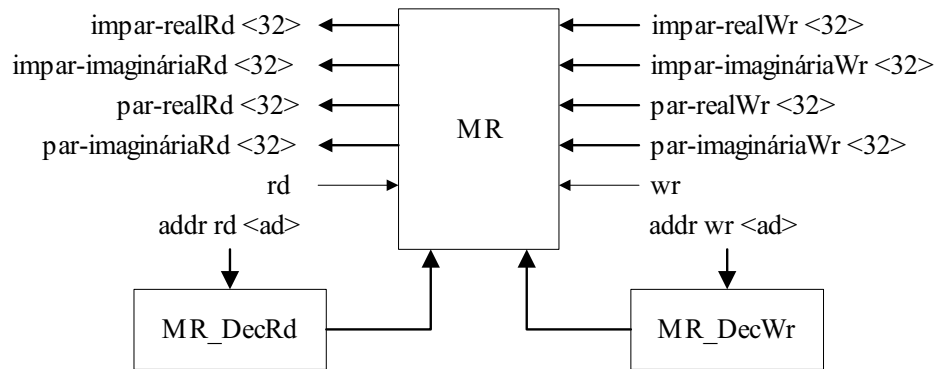


Figura 17: Interface da memória de rascunho

endereço de MR compreende dois coeficientes de uma linha, coluna ímpar e coluna par, abastecendo uma das entradas dos 2 multiplicadores de números complexos de UCalc durante o produto tensorial ou matricial. A quantidade de endereços *qtdEnd* de MR é igual a

$$qtdEnd = 2^{2(q-1)}, \quad (46)$$

onde q é a máxima quantidade de q -bits operados simultaneamente na máquina quântica. A memória MR somente é utilizada na construção e armazenagem temporária do operador quântico para dois ou mais q -bits, conforme instruções do programa quântico. A Figura 18 ilustra o diagrama de tempos de leitura e gravação simultâneas em MR. Na parte inicial realizou-se a leitura das posições de memória 0 a 3, seguindo-se duas escritas nos endereços 1 e 2, finalizando com a releitura dos endereços 0 a 3. O barramento de dados de gravação de MR é compartilhado com o barramento de dados de gravação da memória MQ, enquanto que o barramento de dados de leitura de MR é compartilhado com o barramento de dados da memória MOp.

3.2.4 Unidade de medição

A unidade de medição UMed tem a missão de disponibilizar o estado da máquina quântica quando solicitado pelo processador PROC. Conforme explanado na Subseção 2.2.3 do Capítulo

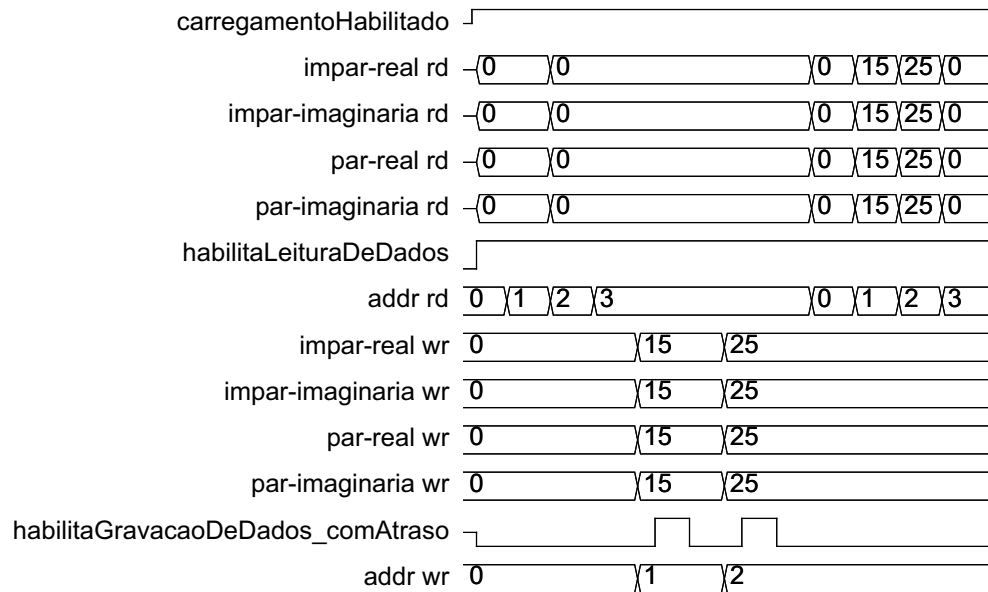


Figura 18: Diagrama de tempos de leitura e gravação simultâneas em MR

2, a medição do estado quântico resulta no colapso de cada q -bit do Coprocessador para um dos estados $|0\rangle$ ou $|1\rangle$. Não foi encontrada na literatura um modelo de medição do estado do q -bits em sistemas simulados de computação quântica. Considerando que para cada um dos 2^q estados possíveis para o(s) q q -bit(s) do Coprocessador, $q \geq 1$, existe uma probabilidade associada, entendeu-se que o processo de medição poderia utilizar o modelo de *seleção proporcional* ou *roleta*, também utilizado em Algoritmos Genéticos na fase de seleção dos indivíduos de uma geração (AMORIM, 2006). Na seleção proporcional, cada um dos n indivíduos da geração corrente tem associada uma pontuação que é utilizada para determinar a probabilidade deste indivíduo ser selecionado. Um intervalo $[0, 1)$, faixa coberta por um gerador de números pseudo-aleatórios, é dividido em n partes, cada uma cobrindo um subintervalo sem interseção com outro, sendo a extensão de cada subintervalo proporcional à pontuação atribuída a cada indivíduo (AMORIM, 2006; DEB, 2001). Transportando tal entendimento para os estados quânticos possíveis no Coprocessador, um deles resultante da leitura da memória MEQ, a probabilidade associada a cada estado quântico estaria atrelada a um subintervalo. Por exemplo, um Coprocessador que possuísse 2 q -bits e apresentasse em dado instante as probabilidades apresentadas na Tabela 10, teria os subintervalos associados a cada estado conforme a Tabela 11.

Quando o gerador de números pseudo-aleatórios fosse solicitado, este apresentaria um valor que seria confrontado com os citados subintervalos, resultando na seleção do estado quântico associado ao intervalo onde se inclui o número pseudo aleatório fornecido pelo gerador. Na macro-arquitetura proposta, GNPA é o gerador de números pseudo-aleatórios que fornece

Tabela 10: Coprocessador com 2 q -bits e probabilidades hipotéticas

Estado	Probabilidade (%/100)
$ 00\rangle$	0, 1
$ 01\rangle$	0, 2
$ 10\rangle$	0, 3
$ 11\rangle$	0, 4

Tabela 11: Coprocessador com 2 q -bits e subintervalos hipotéticos

Estado	Probabilidade
$ 00\rangle$	$[0..0, 1)$
$ 01\rangle$	$[0, 1..0, 3)$
$ 10\rangle$	$[0, 3..0, 6)$
$ 11\rangle$	$[0, 6..1)$

um valor real no intervalo $[0..1)$, baseado no princípio LFSR (*Linear Feedback Shift Register*). O LFSR é um registrador de deslocamento cujo bit de entrada é uma função linear de seu estado anterior. A única função linear de bits é um *XOR*, portanto é um registrador de deslocamento cujo bit de entrada é impulsionado pela função ou-exclusiva de alguns bits do registrador para a mudança de valor. As posições de bit que afetam o próximo estado são chamadas de *Taps*. O período máximo do ciclos sobre o qual a sequência se repete é igual a $2^n - 1$ para um registro LFSR com n bits de largura. Realizando a operação sobre os 8 bits mais significativos da mantissa e nos 4 bits menos significativos do expoente, garante-se que os números gerados permaneçam no intervalo $[0,1]$ (CALAZAN; NEDJAH; MOURELLE, 2012). A Unidade de Medição UMed calcula a distribuição de subintervalos e suas extensões a partir do produto tensorial de todos os q -bit do Coprocessador a partir das amplitudes de cada estado, e realiza a busca do valor provido por GNPA nos subintervalos citados, exteriorizando o estado quântico pertinente. Como consequência da medição, os q -bits assumem estado colapsado ($|0\rangle$ ou $|1\rangle$) e cessam-se os eventuais emaranhamentos.

O componente UMed, cuja micro-arquitetura é apresentada na Figura 19, possui um componente de controle local *uCMed*. Quando o sinal *comandaMedicao* é ativado, UMed recebe os coeficientes de cada endereço da memória MQ (sinais *impar-real rd*, *impar-imaginária rd*, *par-real rd* e *par-imaginária rd*) e calcula por meio de *FCalc* os valores inicial e final de cada subfaixa. Os 2^q números relativos às faixas, onde q é a quantidade de q -bits do Coprocessador, são armazenados na memória local *Mem*, que atua como uma *look-up table*. O número pseudo-aleatório *npa* provido por GNPA é comparado às posições de *Mem* por meio de *Comp*, identificado qual estado quântico será selecionado para ser gravado na memória MEQ, após o quê é ativado o sinal *fimDaMedicao*.

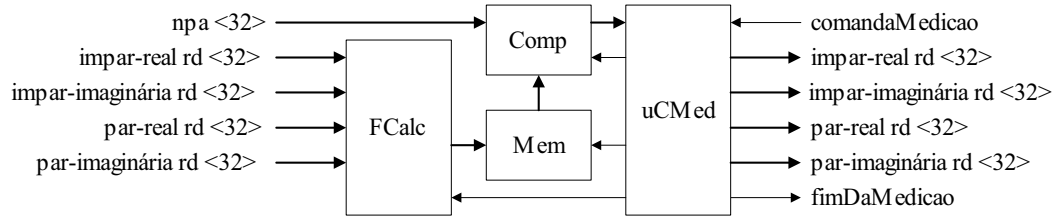


Figura 19: Micro-arquitetura da unidade de medição do estado quântico

3.2.5 Unidade de cálculo

A Unidade de Cálculo UCalc é o componente da macro-arquitetura que realiza produto entre números complexos, necessário à execução do produto tensorial de operadores quânticos, produto tensorial de q -bits, produto matricial entre operador e registro de q -bits e soma de números complexos. UCalc está conectada aos barramentos de dados de leitura e de gravação da memória MR e memória MQ e ao barramento de dados da memória MOp. O Capítulo 4 detalhará este componente.

3.2.6 Unidade de controle

A unidade de controle (UCont) faz o gerenciamento dos componentes do Coprocessador por meio de microprograma e componentes auxiliares. A partir das instruções oriundas do processador PROC, UCont orquestrará por meio de micro-ordens e controladores especializados os outros componentes da macro-arquitetura bem como os componentes internos a UCont. UCont será descrita em detalhes no Capítulo 5.

3.3 Considerações finais do capítulo

Neste capítulo apresentou-se a macro-arquitetura e detalhou-se os componentes, exceto a unidade de cálculo (UCalc) e a de controle (UCont), elementos da arquitetura com maior complexidade funcional e maior número de componentes internos, requerendo capítulos dedicados. O Coprocessador possui memórias separadas para a guarda do estado quântico (MEQ), dos coeficientes dos operadores quânticos básicos (MOp) e dos coeficientes do operador quântico construído (MR). A unidade UMed executa a medição do estado quântico utilizando o algoritmo de seleção *roleta* e gerador de números pseudo-aleatórios. O próximo capítulo apresenta em detalhes a Unidade de Cálculo.

Capítulo 4

UNIDADE DE CÁLCULO

A UNIDADE de Cálculo é o componente da macro-arquitetura dedicado aos produtos e somas de números complexos que compõem os operadores quânticos e agrupamentos de q -bits. Tais multiplicações e somas serão as operações elementares no produto tensorial e produto matricial necessários às operações quânticas. Sob a gerência de UCont, a unidade de cálculo UCalc realiza o produto tensorial de operadores quânticos, produto tensorial de q -bits e o produto matricial entre operador e conjunto de q -bits. Recebe e entrega dados que representam números complexos no padrão IEEE754 com precisão simples (32 bits para a parte real e 32 para a imaginária).

4.1 Micro-arquitetura da unidade

O diagrama de UCalc é apresentado na Figura 20. UCalc está conectada a dois barramentos de dados de entrada e dois de saída com 128 bits cada: um barramento de dados de entrada recebe uma dupla de coeficientes do vetor-coluna representativo do q -bit ou conjunto de q -bits, oriunda de MQ. No segundo barramento de dados de entrada, aportam duplas de coeficientes originárias de MOp ou de MR. MOp fornece os coeficientes dos operadores quânticos básicos, enquanto que MR provê os coeficientes de um operador construído para 2 ou mais q -bits. UCalc conecta-se aos barramentos de dados para gravação das memórias MR e MQ, separadamente. Os dados destinados ao barramento de dados gravação de MR são aqueles relativos ao produto tensorial de operadores; os que seguem ao barramento de dados de gravação de MQ são os resultantes do produto tensorial de q -bits ou do produto matricial entre operador quântico e q -bit ou registro de q -bits. RMC1 e RMC2 são registradores que armazenam o valor do coeficiente do q -bit ou do operador no curso de uma multiplicação tensorial ou matricial. RMCPT1 e RMCPT2 registram os coeficientes da segunda linha de operadores quânticos básicos quando do produto tensorial. RMCEX1n e RMCEX2n representam um conjunto de registradores destinados a

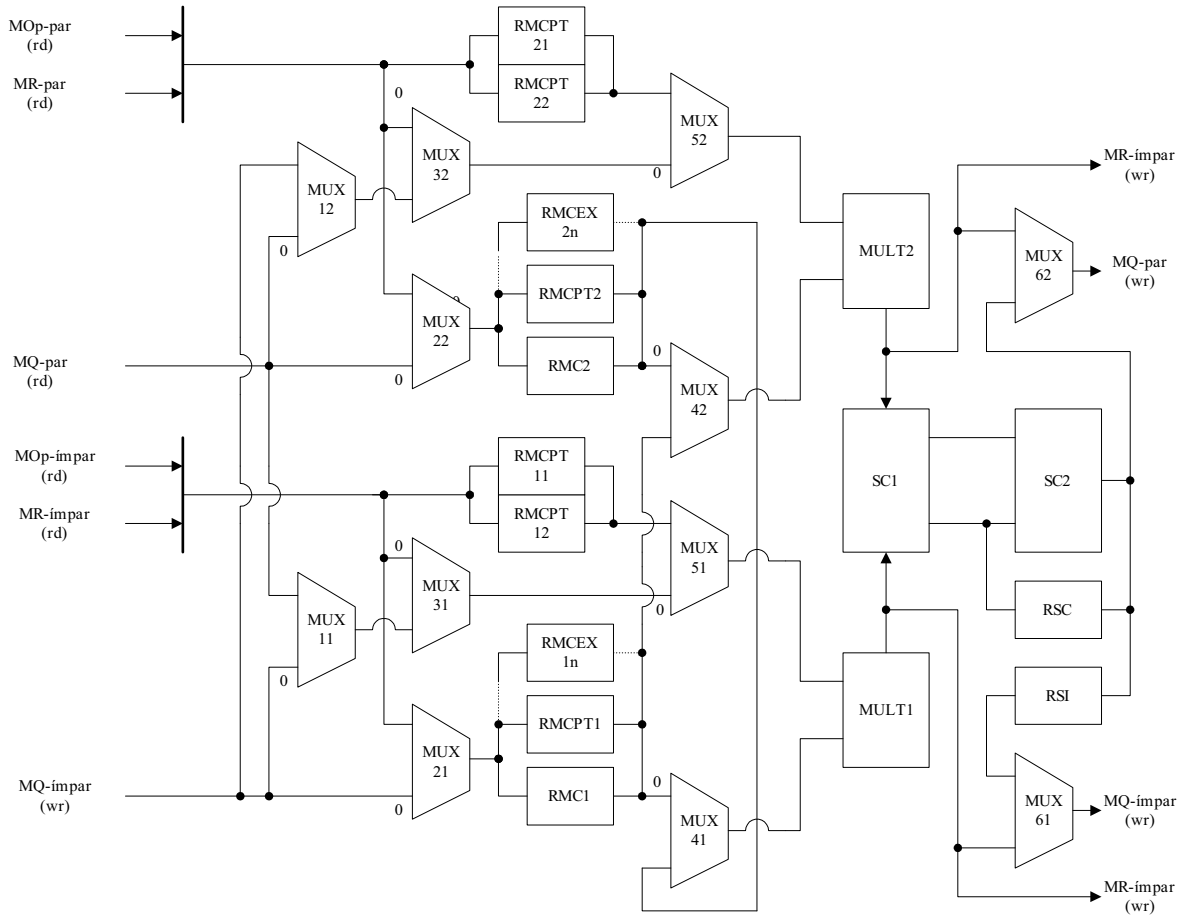


Figura 20: Arquitetura da unidade de cálculo

armazenar os valores das posições de memória de MQ envolvidas em uma operação quântica antes da operação ocorrer. A quantidade de registradores no conjunto RMCEX1n e RMCEX2n é $2^p - 2$, onde p é a quantidade máxima de q -bits sobre os quais um operador pode atuar, logo o sufixo n em RMCEX1n e RMCEX2n pertence a $\{1 \dots 2^p - 2\}$. RMCPT11 e RMCPT21 armazenam coeficientes da primeira linha de operador quântico básico no produto tensorial de operadores, tendo RMCPT12 e RMCPT22 função análoga para a segunda linha de coeficientes. MULT1 e MULT2 são multiplicadores de números complexos. SC1 e SC2 são somadores de números complexos, o primeiro necessário à soma de 2 produtos parciais em uma multiplicação matricial entre um operador e conjunto de q -bits, e o segundo atuando na totalização dos produtos parciais de uma linha da matriz de operador com o vetor-coluna representado por 1 q -bit ou mais q -bits emaranhados. A totalização dos produtos da linha é iterativamente realizada por RSC, o registrador de soma de números complexos. RSI, o registrador de soma de número complexo intermediária, é o registrador da soma dos produtos a ser gravado no campo da linha ímpar de um endereço de MQ, até que o resultado relativo na próxima linha

par esteja disponível para que ambos sejam gravados no devido endereço de MQ. Observe no diagrama acima que os multiplexadores possuem uma das entradas assinaladas com número zero, que é a entrada selecionada por *default*. Tal notação será útil no entendimento do funcionamento da Unidade de Controle, tema do próximo capítulo. A entrada de controle dos multiplexadores, quando não está em nível alto, resulta na seleção da referida entrada 0.

4.2 Multiplicação de números complexos

Os multiplicadores de número complexo (MULT1 e MULT2) são constituídos por IPs de unidades aritméticas (UA) de ponto flutuante e precisão simples parametrizáveis. Utilizam-se 4 multiplicadores, 1 somador e 1 subtrator. Representando-se números complexos como pares ordenados, (a, b) e (c, d) , a e c sendo os coeficientes da parte real e b e d os coeficientes da parte imaginária, uma multiplicação de números complexos tem a forma da Equação 47 ou da Equação 48, onde $ac - bd$ a parte real e $ad + bc$ a parte imaginária. As 4 multiplicações (ac , ad , bc e bd) são feitas primeiramente e ao mesmo tempo. Aprontados os produtos, ocorrem a soma ($ad + bc$) e a subtração ($ac - bd$) concomitantemente.

$$(a, b).(c, d) = (ac - bd, ad + bc) \quad (47)$$

$$(a + bi).(c + di) = ac - bd + (ad + bc)i, \quad (48)$$

Na Figura 21, é apresentado a arquitetura do multiplicador de números complexos. Como estas UAs operam continuamente, não dependendo de um comando para iniciar a operação, o multiplicador de número complexo pode ser sequencialmente abastecido com dados e os resultados serão sequencialmente disponibilizados.

Na Figura 21, cada UA considera os dados presentes nas suas entradas na transição positiva do *clock* e o resultado é entregue também na transição positiva. A quantidade de ciclos de *clock* para a apresentação do resultado é determinado pelo parâmetro *latency* da UA. Para os multiplicadores, adotou-se o valor 3 e para o somador e subtrator o valor 2 e os sinais de *clock* de multiplicadores e de somadores/subtratores são diferentes, de forma que os produtos pudessem ser utilizados pelos somadores/subtratores antes da próxima transição do sinal de *clock* do multiplicador. Se tal abordagem não fosse utilizada, introduzir-se-ia atraso de 1 ciclo de *clock* em um produto entre números complexos, podendo resultar em um atraso considerável quando de uma operação quântica envolvendo vários *q-bits*. A quantidade de multiplicações com números complexos em uma operação quântica, *qtdMultComp*, é dada conforme Equação

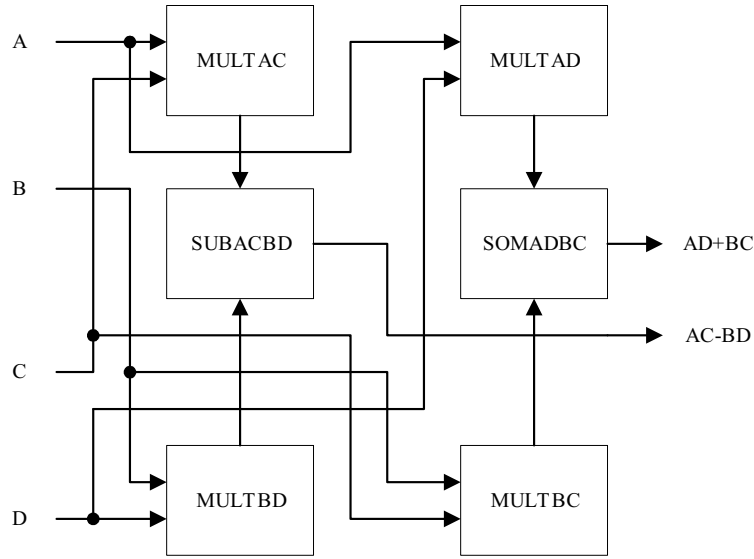


Figura 21: Multiplicador de números complexos

49, onde q é a quantidade de q -bits envolvidos na operação quântica, para $q \geq 2$.

$$qtdMultComp = 2^q + 4^q + 4^q = 2^q + 4^{q+1} \quad (49)$$

4.3 Produto tensorial

Quando da realização de uma operação quântica sobre dois ou mais q -bits em que pelo menos um esteja emaranhado, o produto tensorial entre q -bits será realizado, bem como o produto tensorial entre operadores, tratado nas Subseções que se seguem.

4.3.1 Produto tensorial de q -bits

A construção do registro de q -bits inicia-se como o produto tensorial entre os q -bit especificados nas duas primeiras instruções da operação quântica. O q -bit que fosse especificado na terceira instrução seria multiplicado tensorialmente com o registro de dois q -bits que foi preparado na iteração anterior. Para os produtos tensoriais entre quatro ou mais q -bits, procede-se analogamente, multiplicando-se o vetor-coluna para n q -bits com o próximo q -bit definido na instrução. Logo, a construção do produto tensorial de n q -bits necessita do produto tensorial entre $n - 1$ q -bits anteriores. A quantidade de linhas do produto tensorial de q q -bits é 2^q , requerendo 2^{q-1} posições da memória MEQ. O produto tensorial entre q -bits estende a condição emaranhada a todos os q -bits participantes. A memória MQ, inicialmente comportando em cada endereço os 2 coeficientes do vetor-coluna 2×1 do q -bit que representa, passa a ocupar 2^{q-1} endereços (2^q coeficientes), q sendo a quantidade de q -bits emaranhados, cabendo aos campos da memória

MCQ a guarda dos ponteiros que remontam a ordem dos dados de cada endereço participante de MQ. Por exemplo, se os q -bits 1, 4 e 2 participassem de uma operação quântica, estando o q -bit 4 emaranhado, ao final da operação os endereços 1 e 2 também guardariam a marca de emaranhamento (campo *Em*). Os endereços 1, 4 e 2 seriam necessários mas não suficientes para acomodar o resultado (vetor 8×1) de tal produto tensorial, pois um endereço extra (o quarto endereço deste grupo) armazenaria os 2 últimos coeficientes do vetor-coluna resultante. O produto tensorial de n q -bits, $n \geq 2$, é precedido pela armazenagem dos 2^{n-1} coeficientes do conjunto de coeficientes em RMC/RMC2, RMCPR1/RMCPT2, e no conjunto de registradores RMCEX1n/RMCEX2n, apenas os necessários, nesta ordem, ou seja, os coeficientes das 2 primeiras linhas do vetor-coluna representativo do conjunto de q -bits ocupa os registradores RMC1 e RMC2, seguindo-se as outras linhas nos registradores restantes. Em seguida, os 2 coeficientes do q -bit que será operado com o conjunto de coeficientes são colocados no barramento de entrada de dados específico, mantendo-se presentes durante todo o curso do produto tensorial de q -bits. O produto tensorial, a esta altura, pode iniciar-se: RMC1 tem sua saída habilitada e, por intermédio de MUX41 e MUX42, MULT1 e MULT2 recebem o coeficiente da primeira linha em uma das entradas. Por meio de MUX11, MUX31 e MUX51, MULT1 recebe em sua outra entrada o coeficiente da linha ímpar do segundo q -bit participante deste produto tensorial. Analogamente, MULT2 recebe o coeficiente da linha par que transitou por MUX12, MUX32 e MUX52. Assim, entregues estão os fatores que gerarão as duas primeiras linhas do novo vetor-coluna. Sucessivamente, o produto dos coeficientes restantes do conjunto de q -bits (armazenados em RMCPT1/RMCPT2, RMCEX11/RMCEX21 e assim por diante) vão sendo apresentados nas entradas de MULT1 e MULT2, respectivamente intermediados por MUX41 e MUX42. Os produtos tensoriais disponibilizados nos devidos barramentos de dados de saída já podem ser armazenados em MQ, já que MEQ é uma memória dual-port e o endereço de MQ que está sendo lido é diferente daquele onde o resultado deve ser gravado. Como o segundo q -bit participante da operação fará parte do novo conjunto de q -bits emaranhados, seu conteúdo será sobre-escrito. Desde que a *latência* do multiplicador de número complexo (MULT1 ou MULT2) seja de pelo menos 1 ciclo de *clock*, a escrita do resultado neste segundo q -bit acontecerá quando ele não mais estiver sendo lido.

4.3.2 Produto tensorial de operadores

No operador quântico que deve atuar sobre 2 ou mais q -bits simultaneamente, cada dimensão do operador possui 2^q , onde q é a quantidade de linhas e de colunas do operador. O Coprocessador

guarda na memória MOp os coeficientes dos operadores quânticos básicos, todos com dimensão 2×2 , com exceção do operador CNOT, cuja dimensão é 4×4 . Desta forma, um operador quântico, com exceção dos operadores CNOT, Toffoli, Swap e Fredkin, pode ser construído a partir de produto(s) tensorial(is) entre operadores quânticos básicos, como exemplificado na Equação 50, com A representando um operador quântico básico genérico.

$$A = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \quad (50)$$

O produto tensorial entre 3 operadores quânticos básicos 2×2 resulta em uma matriz com dimensão 8×8 . Analogamente ao produto tensorial entre q -bits, a construção de um operador para q ou mais q -bits, $q \geq 3$, é construído por meio do produto tensorial do operador para $n-1$ q -bits com o operador quântico básico. Uma operação quântica é especificada por um ou mais descritores, cada descritor definindo a operação quântica e o q -bit alvo. Quando tratar-se de uma operação sobre 2 ou mais q -bits, a construção do operador quântico e do vetor-coluna segue a ordem dos descritores. O produto tensorial de operadores e de q -bits somente ocorre quando a instrução quântica for relativa a mais de um q -bit e exista pelo menos 1 q -bit emaranhado. Sendo a instrução quântica sobre dois ou mais q -bits e nenhum deles emaranhado, a execução do produto tensorial (símbolo \otimes) é evitada pelo uso da propriedade descrita na Equação 51:

$$(A.B) \otimes (u.v) = (A.u) \otimes (B.v) \quad (51)$$

Por exemplo, a construção de um operador para 2 q -bits tomando-se dois operadores quânticos com dimensão 2×2 para serem multiplicados tensorialmente, requer 16 produtos de números complexos, resultando em uma matriz 4×4 . O produto tensorial de 2 q -bits demanda 4 multiplicações. O produto matricial deste operador 4×4 com o vetor-coluna 4×1 implica em 4 produtos e 3 adições por linha do operador, totalizando 16 produtos e 12 adições. Desta forma, necessitam-se de $16 + 4 + 16 = 36$ produtos e 12 somas. O resultado desta operação matricial é um vetor-coluna 4×1 , o qual representa o produto tensorial dos 2 q -bits com seus novos coeficientes individuais. No entanto, o resultado não disponibiliza os 2 coeficientes de cada q -bit para que eles possam ser armazenados na devida posição de memória, sendo necessário o desmembramento deste vetor-coluna 4×1 em 2 vetores-coluna 2×1 . Utilizando-se da propriedade representada na Equação 51, o produto matricial de um operador básico (matriz 2×2) com um q -bit (vetor-coluna 2×1) requer 2 multiplicações e 1 adição por linha de operador, totalizando 4 multiplicações e 2 somas. Assim, a operação quântica sobre 2 q -bits realizada com base na mencionada propriedade do produto tensorial resulta em $2 \times 4 = 8$ multiplicações e $2 \times 2 = 4$ somas. O resultado de cada produto matricial é um vetor-coluna

2×1 , já pronto para ser gravado em MQ. Enquanto não houver q -bit emaranhado na operação, o uso da dita propriedade do produto tensorial propicia redução no custo em processamento. Na execução do produto tensorial de operadores para 2 q -bits, uma linha de coeficientes do operador (2×2) é lida a cada vez de MOp. Os endereços de MOp que serão lidos são providos por UCont de acordo com a instrução quântica definida pelo programador. O produto tensorial de dois operadores inicia-se com o carregamento dos coeficientes da primeira linha do primeiro operador em RMC1 e RMC2, seguindo-se o carregamento dos coeficientes da segunda linha em RMCPT1 e RMCPT2, com a intermediação de MUX21 e MUX22. A seguir e analogamente, carregam-se os coeficientes da primeira linha do segundo operador em RMCPT11 e RMCPT21 e os da segunda linha em RMCPT12 e RMCPT22. Carregados os coeficientes dos operadores básicos, os mesmos são encaminhados às entradas de MULT1 e MULT2 na seguinte ordem de habilitação de saídas dos registradores, gerando os coeficientes do operador para 2 q -bits:

1. RMC1, RMCPT11 e RMCPT21: $a_1.b_1$ e $a_1.b_2$; RMC2, RMCPT11 e RMCPT21: $a_2.b_1$ e $a_1.b_2$; Com estes produtos, obtêm-se os coeficientes da primeira linha do novo operador para 2 q -bits.
2. RMC1, RMCPT12 e RMCPT22: $a_1.b_3$ e $a_1.b_4$; RMC2, RMCPT12 e RMCPT22: $a_2.b_3$ e $a_2.b_4$. Com estes produtos, obtêm-se os coeficientes da segunda linha.
3. RMCPT1, RMCPT11 e RMCPT21: $a_3.b_1$ e $a_3.b_2$; RMCPT2, RMCPT11 e RMCPT21: $a_4.b_1$ e $a_4.b_2$. Com estes produtos, obtêm-se os coeficientes da terceira linha.
4. RMCPT1, RMCPT12 e RMCPT22: $a_3.b_3$ e $a_3.b_4$; RMCPT2, RMCPT12 e RMCPT22: $a_4.b_3$ e $a_4.b_4$. Com estes produtos, obtêm-se os coeficientes da quarta e última linha.

Para se fazer o produto tensorial de 3 q -bits, parte-se do produto tensorial de dois operadores básicos o qual deverá ser multiplicado tensorialmente com os coeficientes do terceiro operador. O operador construído para 2 q -bits possui 16 coeficientes, consumindo 8 posições de memória de MR. O operador para n q -bits possui 4 vezes mais coeficientes do que o operador para $n - 1$ q -bits, ou seja, 2 vezes mais endereços ocupados em MR. A memória MR é dual-port para permitir que o produto tensorial de operadores ocorra continuamente, no entanto a mesma posição de memória não pode ser lida e escrita simultaneamente. Assim, a construção do produto tensorial para 3 ou mais q -bits inicia-se pelo último coeficiente do operador de origem, neste caso o operador para 2 q -bits, como que percorrendo cada coeficiente da última para a primeira coluna, da última para a primeira linha. A ordem reversa

para leitura dos coeficientes (da última para a primeira coluna, da última para a primeira linha) foi adotada para prevenir conflito no endereçamento de MR em implementações capazes de produzir o resultado em menor quantidade de ciclos de *clock*. Para implementações menos ágeis, a ordem crescente ou decrescente na leitura dos pares de coeficientes é indiferente. Para realizar o produto tensorial de operador de 3 *q-bits*, o coprocessador deve contar com 16 registradores, 8 para os coeficientes da coluna ímpar (RMC1, RMCPT1, RMCEX11, RMCEX12, RMCEX13, RMCEX14, RMCEX15 e RMCEX16) e 8 para a par (RMC2, RMCPT2, RMCEX21, RMCEX22, RMCEX23, RMCEX24, RMCEX25 e RMCEX26) do operador para 2 *q-bits*, a serem registrados na seguinte ordem: a_4b_3 e a_4b_4 são carregados respectivamente em RMC1 e RMC2; a_3b_3 e a_3b_4 em RMCPT1 e RMCPT2, respectivamente; a_4b_1 e a_4b_2 bem RMCEX11 e RMCEX21; a_3b_1 e a_3b_2 em RMCEX12 e RMCEX22; a_2b_3 e a_2b_4 em RMCEX13 e RMCEX23; a_1b_3 e a_1b_4 em RMCEX14 e RMCEX24; a_2b_1 e a_2b_2 em RMCEX15 e RMCEX25; e finalmente a_1b_1 e a_1b_2 em RMCEX16 e RMCEX26. A seguir e analogamente, carregam-se os coeficientes da primeira linha do terceiro operador em RMCPT11 e RMCPT21 e os da segunda linha em RMCPT12 e RMCPT22. Carregados os coeficientes dos operadores, os mesmos são encaminhados às entradas de MULT1 e MULT2 na seguinte ordem de habilitação de saídas dos registradores, produzindo os coeficientes do novo operador para 3 *q-bits*:

1. RMC2, RMCPT12 e RMCPT22: $a_4.b_4.c_3$ e $a_4.b_4.c_4$; RMC1, RMCPT12 e RMCPT22: $a_4.b_3.c_3$ e $a_4.b_3.c_4$; RMCPT2, RMCPT12 e RMCPT22: $a_3.b_4.c_3$ e $a_3.b_4.c_4$; RMCPT1, RMCPT12 e RMCPT22: $a_3.b_3.c_3$ e $a_3.b_3.c_4$. Com estes produtos, obtêm-se os coeficientes da oitava e última linha do novo operador.
2. RMC2, RMCPT11 e RMCPT21: $a_4.b_4.c_1$ e $a_4.b_4.c_2$; RMC1, RMCPT11 e RMCPT21: $a_4.b_3.c_1$ e $a_4.b_3.c_2$; RMCPT2, RMCPT11 e RMCPT21: $a_3.b_4.c_1$ e $a_3.b_4.c_2$; RMCPT1, RMCPT11 e RMCPT21: $a_3.b_3.c_1$ e $a_3.b_3.c_2$. Com estes produtos, obtêm-se os coeficientes da sétima linha.
3. RMCEX21, RMCPT12 e RMCPT22: $a_4.b_2.c_3$ e $a_4.b_2.c_4$; RMCEX11, RMCPT12 e RMCPT22: $a_4.b_1.c_3$ e $a_4.b_1.c_4$; RMCEX22, RMCPT12 e RMCPT22: $a_3.b_2.c_3$ e $a_3.b_2.c_4$; RMCEX12, RMCPT12 e RMCPT22: $a_3.b_1.c_3$ e $a_3.b_1.c_4$. Com estes produtos, obtêm-se os coeficientes da sexta linha.
4. RMCEX21, RMCPT11 e RMCPT21: $a_4.b_2.c_1$ e $a_4.b_2.c_2$; RMCEX11, RMCPT11 e RMCPT21: $a_4.b_1.c_1$ e $a_4.b_1.c_2$; RMCEX22, RMCPT11 e RMCPT21: $a_3.b_2.c_1$ e $a_3.b_2.c_2$; RMCEX12,

- RMCPPT11 e RMCPPT21: $a_3.b_1.c_1$ e $a_3.b_1.c_2$. Com estes produtos, obtêm-se os coeficientes da quinta linha.
5. RMCEX23, RMCPPT12 e RMCPPT22: $a_2.b_4.c_3$ e $a_2.b_4.c_4$; RMCEX13, RMCPPT12 e RMCPPT22: $a_2.b_3.c_3$ e $a_2.b_3.c_4$; RMCEX24, RMCPPT12 e RMCPPT22: $a_1.b_4.c_3$ e $a_1.b_4.c_4$; RMCEX14, RMCPPT12 e RMCPPT22: $a_1.b_3.c_3$ e $a_1.b_3.c_4$. Com estes produtos, obtêm-se os coeficientes da quarta linha.
6. RMCEX23, RMCPPT11 e RMCPPT21: $a_2.b_4.c_1$ e $a_2.b_4.c_2$; RMCEX13, RMCPPT11 e RMCPPT21: $a_2.b_3.c_1$ e $a_2.b_3.c_2$; RMCEX24, RMCPPT11 e RMCPPT21: $a_1.b_4.c_1$ e $a_1.b_4.c_2$; RMCEX14, RMCPPT11 e RMCPPT21: $a_1.b_3.c_1$ e $a_1.b_3.c_2$. Com estes produtos, obtêm-se os coeficientes da terceira linha.
7. RMCEX25, RMCPPT12 e RMCPPT22: $a_2.b_2.c_3$ e $a_2.b_2.c_4$; RMCEX15, RMCPPT12 e RMCPPT22: $a_2.b_1.c_3$ e $a_2.b_1.c_4$; RMCEX26, RMCPPT12 e RMCPPT22: $a_1.b_2.c_3$ e $a_1.b_2.c_4$; RMCEX16, RMCPPT12 e RMCPPT22: $a_1.b_1.c_3$ e $a_1.b_1.c_4$. Com estes produtos, obtêm-se os coeficientes da segunda linha.
8. RMCEX25, RMCPPT11 e RMCPPT21: $a_2.b_2.c_1$ e $a_2.b_2.c_2$; RMCEX15, RMCPPT11 e RMCPPT21: $a_2.b_1.c_1$ e $a_2.b_1.c_2$; RMCEX26, RMCPPT11 e RMCPPT21: $a_1.b_2.c_1$ e $a_1.b_2.c_2$; RMCEX16, RMCPPT11 e RMCPPT21: $a_1.b_1.c_1$ e $a_1.b_1.c_2$. Com estes produtos, obtêm-se os coeficientes da primeira linha.

Em cada um dos 8 ciclos de *clock* em que ocorre a disponibilização dos resultados por MULT1 e MULT2 (cada resultado é aprontado 4 ciclos de *clock* após o ciclo em que os dados foram coletados na transição positiva), ou seja, do quinto ao décimo-terceiro, a memória de rascunho MR é gravada com a dupla de resultados (1 dupla de resultados por endereço). Embora a construção de um operador para 2 ou mais *q-bits* requeira a execução de 4^q multiplicações, onde q é a quantidade de *q-bits* do operador em construção, existe a possibilidade de o produto matricial entre o operador e o vetor-coluna resultante do produto tensorial de q -bits ser iniciado antes da finalização da construção do operador. Para isto, seria necessário que o produto tensorial de q -bits estivesse pronto e fosse a última iteração de produto tensorial entre o operador anterior para $q-1$ *q-bits* e o operador quântico básico. UCalc precisaria contar com mais dois multiplicadores para números complexos para realizar o produto tensorial, enquanto que MULT1, MULT2, SC1, SC2, RSC e RS1 realizariam o produto entre os coeficientes do operador que foram recém-calculados e os devidos coeficientes do vetor-coluna representativo

do produto tensorial dos q -bits. Registradores adicionais seriam requeridos para armazenar os coeficientes do vetor-coluna antes da operação, para que o produto matricial pudesse ser consumado adequadamente. Uma vez que os coeficientes de uma linha do novo operador estivessem calculados, o produto matricial poderia ser executado, cujos resultados matricial seriam escritos em MQ conforme fossem disponibilizados.

4.4 Produto matricial

O produto matricial é a própria operação quântica. Eventualmente antecedido pelo(s) produto(s) tensorial(is) entre operadores e produto(s) tensorial(is) entre q -bits, o produto matricial é a última etapa do conjunto de cálculos a ser realizado por UCalc. Do produto matricial resultarão os novos coeficientes do q -bit ou conjunto de q -bits, configurando o novo estado quântico. O produto matricial é antecedido pela armazenagem dos coeficientes do q -bit ou conjunto de coeficientes de q -bits em RMC/RMC2, RMCPR1/RMCPT2, e da série de registradores RMCEX1n/RMCEX2n. Este armazenamento faz-se necessário para que os valores dos coeficientes antes da operação estejam disponíveis para os repetidos produtos com os coeficientes de cada linha do operador, liberando MQ para receber os novos coeficientes (resultado do produto matricial) do q -bit ou conjunto de q -bits. O armazenamento prévio apresenta a utilidade de dispor localmente à UCalc destes coeficientes, evitando a repetida leitura dos mesmos da memória MQ, eventualmente mais custosa em termos de número de ciclos de *clock*. O armazenamento em tais registradores é feito atendendo à ordem:

1. Os coeficientes do q -bit especificado no primeiro descritor de instrução quântica são carregados em RMC1 e RMC2;
2. Os coeficientes do q -bit especificado no segundo descritor é armazenado em RMCPT1 e RMCPT2;
3. Os coeficientes do q -bit especificado no terceiro descritor e os seguintes são copiados para RMCEX1n e RMCEX2n, $n \geq 1$, conforme a quantidade de endereços de MQ necessários para armazenar os coeficientes do conjunto de q -bits.
4. Uma vez carregados, os coeficientes do operador básico em MOp ou MR serão lidos e aportados diretamente nas entradas de MULT1 e MULT2 por intermédio de MUX31/MUX32 e MUX51/MUX52.

5. Com dois multiplicadores para número complexo, o produto matricial pode ser feito de 2 em 2 coeficientes (2 coluna de uma linha do operador, 2 linhas do vetor-coluna que representa o(s) q -bit(s)). A dupla de produtos é aprontada simultaneamente e somada por SC1.
6. Quando da execução dos produtos relativos aos primeiros 2 coeficientes de cada linha do operador, RSC tem seu conteúdo inicializado com valor 0 (padrão IEEE754), de modo que SC2 repasse o mesmo valor aportado por SC1.
7. O conteúdo de SC2 é registrado em RSC, que passa a funcionar como um acumulador de somas de produtos, até que os dois últimos coeficientes da linha corrente do operador sejam multiplicados e RSC disponha do valor do coeficiente a ser armazenado no devido endereço de MQ.
8. Como cada endereço da memória MQ guarda 2 coeficientes (linha ímpar e linha par do vetor-coluna), quando o produto matricial for de uma linha ímpar do operador, o resultado (final) dos produtos contido em RSC é armazenado em RSI. Quando o resultado dos produtos de uma linha par do operador estiver pronto, o conteúdo de RSI e o de RSC é gravado no apropriado endereço de MQ.

Findas as multiplicações dos últimos 2 coeficientes do operador e gravados os novos coeficientes do vetor-coluna, o coprocessador está disponível para atender à próxima solicitação de PROC.

4.5 Considerações finais do capítulo

Neste Capítulo apresentou-se UCalc, o componente da macro-arquitetura que executa todas as multiplicações e somas de números complexos, operações-meio e operações-fim de uso massivo nas operações quânticas, especialmente quando envolve q -bits emaranhados e/ou operações que emaranham q -bits. A unidade UCalc foi concebida de forma que viabilizasse o paralelismo das multiplicações entre coeficientes de operadores e q -bits. UCalc (vide diagrama na Figura 20) é constituída por dois conjuntos idênticos de componentes (multiplexadores e registradores) mais os multiplicadores, somadores e registradores associados. A capacidade de tratamento simultâneo de mais coeficientes é possível, com a replicação dos citados conjuntos e devidas modificações na quantidade de entradas dos correspondentes aos atuais MUX41 e MUX42. A quantidade de registradores e multiplexadores antes do estágio das UAs deve ser sempre em quantidade que seja potência de 2, limitado a 2^q , onde q é a quantidade máxima de q -bits

operáveis simultaneamente pelo coprocessador, de UCalc. A quantidade de etapas relativas às UAs é igual à metade da etapa anterior, pois cada etapa de UAs opera os dados fornecidos pelos dois conjuntos pré-UAs. O aumento da capacidade de multiplicações simultâneas de UCalc também exigiria da Unidade de Controle UCont o escalonamento das gravações dos resultados em MQ, já que haveria a disponibilização de dados a gravar em mais de um endereço de MQ. No capítulo seguinte, a Unidade de Controle será descrita, também complementando a compreensão do funcionamento de UCalc.

Capítulo 5

UNIDADE DE CONTROLE

A UNIDADE de Controle (UCont) representa um conjunto diversificado e amplo de componentes e funcionalidades para o microcontrole dos demais componentes da macro-arquitetura do Coprocessador e também dela própria. Utiliza microprograma e componentes especializados para gerenciar desde a recepção das solicitações do processador principal PROC até o retorno do estado quântico.

5.1 Micro-arquitetura da unidade

A unidade de Controle UCont é apresentada na Figura 22, onde se vê os componentes que serão detalhados nas Subseções seguintes:

1. InstGer é o gerenciador de instruções;
2. MC é a memória de controle;
3. MIR é o registrador de micro-instrução;
4. MPC registrador de endereço da memória de controle;
5. DesvGer faz o tratamento de desvio incondicional ou condicional;
6. OpAdReg é o registrador de endereço de coeficientes da memória de operadores MOp;
7. OpPTCont controla o produto tensorial de operadores quânticos básicos;
8. OpPTRecCont gerencia a gravação de produto tensorial em MR;
9. MRQbCont armazena a quantidade de q -bits corrente do operador quântico em construção;

10. MRRdCont e MRWrCont são, respectivamente, os registradores de endereço de leitura e de gravação da memória MR;
11. MRPTRdConv e MRPTWrConv são conversores de endereço de leitura e de gravação dos coeficientes na memória MR, respectivamente;
12. QbPTCont é o controlador de produto tensorial de q -bits;
13. MQRdReg e MQWrReg são, respectivamente, registradores para armazenamento do endereço de leitura e de escrita em MEQ.
14. MRWrAd: barramento de endereço de gravação de MR;
15. MRRdAd: barramento de endereço de leitura de MR;
16. OpAd: barramento de endereço de MOp;
17. MEQWrAd: barramento de endereço de gravação de MEQ;
18. MEQRdAd: barramento de endereço de leitura de MEQ;
19. MCQdata: barramento de endereço de dados de leitura e gravação de MCQ.

A unidade UCont recebe dois sinais de clock e um vetor de *flags*, externando um vetor de sinais *tri-state* que representam as micro-ordens.

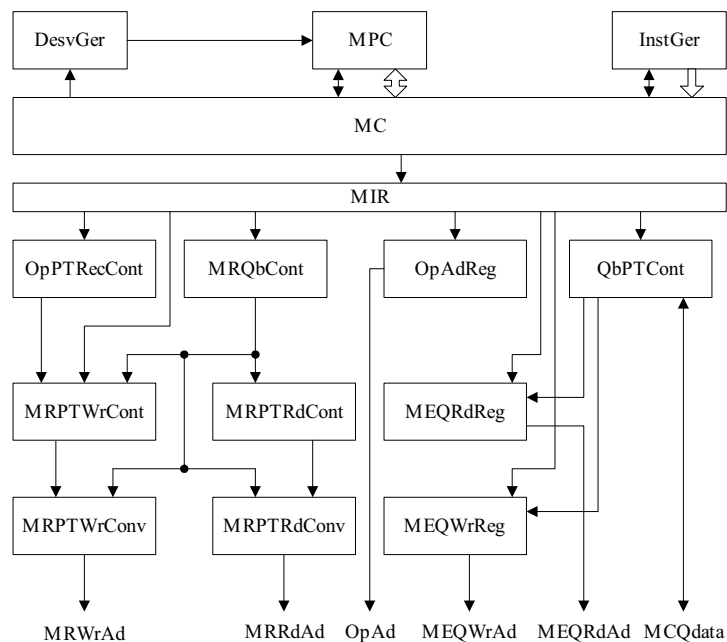


Figura 22: Unidade de controle

5.2 Gerenciador de instruções

PROC apresenta instruções de tamanho fixo que são recepcionadas e registradas por InstGer, as quais descrevem o(s) operador(es) e q -bits que participa(m) da operação quântica. Uma operação quântica é especificada por tantas instruções quantas forem os q -bits a operar. O formato da instrução da operação quântica é apresentado na Figura 23.

OP	QB	T
----	----	---

Figura 23: Instrução descritiva de operação quântica

OP é o código da operação quântica, QB o número do q -bit-alvo e T o terminador. A quantidade de bits necessária para representar OP é dada por $OP_{ext} = \lceil \log_2(M) \rceil$, M a quantidade de operadores quânticos básicos implementados no coprocessador e $\lceil \cdot \rceil$ delimitam o alvo da função de arredondamento para cima. A quantidade de bits requerida para representar QB é igual a $QB_{ext} = N$, onde QB_{ext} quantidade de bits requerida para representar QB e N a quantidade de q -bits que o coprocessador possui. O campo T possui 1 bit e indica, quando está com valor 1, que a instrução é a última da série que especifica a operação quântica.

Se a operação quântica solicitada a COPROC for sobre um q -bit, a única instrução já define a operação quântica. Um operador quântico para n q -bits, $n \geq 2$, é construído com base nas n instruções, cada uma definindo o operador quântico básico (campo OP) e o número do q -bit (campo QB) que participarão do produto tensorial de operadores quânticos básicos e do produto tensorial de q -bits. A operação quântica *medição* dispensa o preenchimento dos campos QB e T . O *status* presente do Coprocessador elenca 9 operações quânticas: oito operações unitárias e uma para realizar a medição do estado quântico, conforme Tabela 12, sendo 4 a quantidade de bits necessária para representar a operação quântica.

Tomando-se como exemplo um coprocessador dotado de 4 q -bits, a memória de estado quântico teria que possuir 16 posições, logo 4 bits seriam necessários para endereçar esta faixa (0...15). Segundo a Tabela 12, uma operação Not (X) tem o código 1. Se tivesse como alvo o q -bit 2, os campos seriam preenchidos da seguinte forma:

- instrução única: $OP = 0001$, $QB = 0010$ e $T = 1$.

Supondo-se uma operação Hadamard (código 4) sobre os q -bits 1 e 2, as instruções seriam:

- instrução 1: $OP = 0100$, $QB = 0001$ e $T = 0$

Tabela 12: Códigos da operações quânticas

OPERAÇÃO	CÓDIGO DA OPERAÇÃO	QUANTIDADE DE <i>Q-BITS</i> REQUERIDA
I	0	1
X	1	1
Y	2	1
Z	3	1
H	4	1
S	5	1
T	6	1
CNOT	7	2
MEDIÇÃO	8	0

- instrução 2: $OP = 0100$, $QB = 0010$ e $T = 1$

No caso da solicitação de uma medição do estado quântico (código 8):

- instrução única: $OP = 1000$, $QB = X$ e $T = X$

onde X representa um valor qualquer e não será considerado.

InstGer possui os seguintes sinais de controle:

1. *DescReq*, solicitação de instrução;
2. *PntRst*, reinicializa ponteiro da instrução.

InstConst externa os sinais de *status* a seguir:

1. *NoDesc*, lista de instruções vazia;
2. *Desc1+*, existe mais de um instrução na operação quântica;
3. *DescUlt*, alcançado a última instrução;
4. *DescQtd*, quantidade de instruções na operação corrente;
5. *MOpAdUlt*, endereço final a ler de MOp;
6. *QbPri*, número do *q-bit* na primeira instrução da operação quântica corrente;
7. *DescCor*, instrução atualmente selecionada.

5.3 Memória de controle

A memória de controle MC é do tipo *read-only* e contém o microprograma, possuindo um barramento de dados e um de endereços. Cada endereço desta memória é composto por um conjunto de bits cujos estados são *hard-coded*, determinados em tempo de simulação ou síntese. A quantidade de endereços de MC é a suficiente para comportar todas as micro-instruções, pois também é determinada em tempo de simulação ou síntese. Cada bit desta memória está vinculado a uma linha *tri-state*, interpretada como ativa quando está no nível alto e como inativa quando não estiver em nível alto (alta impedância ou nível baixo). A escolha da característica *tri-state* deveu-se à existência de alguns sinais de controle possuírem mais de um *driver*, justamente aqueles que relacionados aos componentes auxiliares de controle.

5.3.1 Registrador de endereço de microprograma

O registrador de endereço MPC é formado por um circuito composto por contador crescente com *preset*, *buffers* e multiplexador, conforme se vê na Figura 24. A característica crescente do contador CNT automatiza o sequenciamento de micro-instruções que não seja do tipo desvio. Quando da ocorrência de um desvio incondicional ou um desvio condicional em que a condição seja satisfeita, o conteúdo do campo *operando* da micro-instrução é carregado em neste contador.

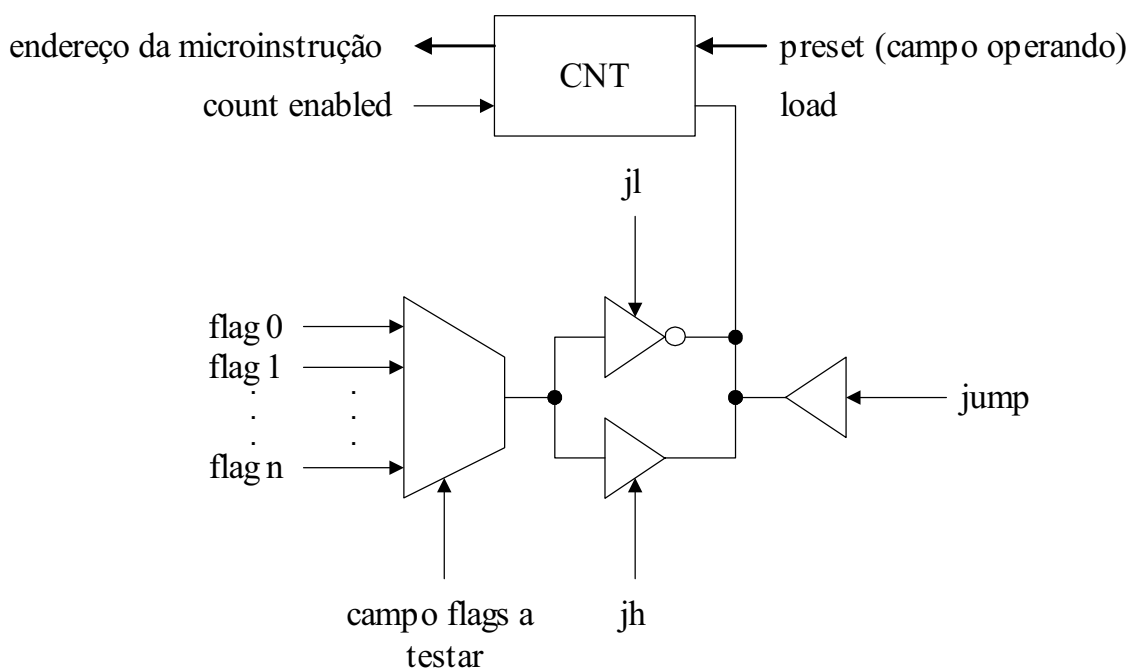


Figura 24: Circuito de endereçamento da memória de controle

O sinal *countEnable* habilita o incremento do valor contido em CNT. Sendo um contador, o sequenciamento das micro-instruções pode se dar sem a necessidade de micro-instruções para realizar a soma do conteúdo de um registrador com um valor 1 utilizando uma ULA. Outros aspectos da Figura 24 serão explanados na Seção 5.3.2

5.3.2 Tratamento de desvio incondicional ou condicional

DesvGer é um circuito combinacional que utiliza os sinais do vetor de *flags* para habilitar ou não o carregamento de MPC com o conteúdo do campo de *operando* da micro-instrução. A Figura 24 apresenta os componentes acessórios ao contador CNT que participam do carregamento do mesmo quando de um *jump* incondicional ou *jump* condicional quando o *flag* em teste corresponde ao estado lógico esperado. Os *buffers* não-inversores e o *buffer* inversor têm saída *tri-state*.

5.4 Microprograma

O microprograma residente na memória MC comanda os sinais de todos os componentes da macro-arquitetura e também os componentes auxiliares de controle de UCont. A micro-instrução constitui-se de três partes: micro-ordens, *flag* e operando. Na atual versão deste Coprocessador, a micro-instrução tem característica horizontal. A faixa de bits do *flag* é a representação binária do código do *flag* a ser verificado em uma instrução de desvio condicional. *Operando* é uma faixa de bits destinada à guarda de um numeral que é requerida em certas operações de carregamento de registrador. O conjunto de micro-ordens gerencia diretamente componentes elementares como registradores, contadores, multiplexadores e *buffers*, bem como componentes especializados que atuam cooperativamente com o microprograma. O registrador MIR armazena os sinais da última micro-instrução lida de MC (micro-ordens) e serve também para estabilizar os sinais evitando que ocorram acionamentos espúrios na transição do *clock*. Em paralelo com algumas das saídas de MIR, existem alguns componentes auxiliares de UCont que incorpora algum automatismo e atuam sobre uma linha de sinal de micro-ordem, atendendo a determinadas sequências constantes e repetidas de controle. As ações comandadas pelo microprograma podem ser conhecidas no Algoritmo 1.

Algoritmo 1 Microprograma

```

1: _inicializaQbits:
2: Inicialize registrador de endereço da memória MEQ com endereço do último q-bit
3: repeat
4:   Escreva valor 1,000 no ket  $|0\rangle$  e 0,000 no ket  $|1\rangle$  do q-bit selecionado
5:   Incremente registrador de endereço da memória MEQ
6: until que o primeiro q-bit seja inicializado
7: _esperaInstrução:
8: repeat
9:   Leia entrada
10: until que PROC apresente instrução
11: _recepcionaInstrução:
12: repeat
13:   Registra instrução
14:   Verifica se operador causa emaranhamento
15:   Verifica se q-bit especificado está emaranhado (campo  $Em = 1$ )
16: until encontrar a última instrução (campo  $T = 1$ )
17: _testaseProdutoTensorialEhNecessario:
18: Se operador causa emaranhamento e/ou existe q-bit emaranhado Então
19:   Desvie para _produtoTensorial;
20: Senão
21:   Desvie para _produtoMatricial
22: Fim Se
23: _produtoTensorial:
24: Execute produto tensorial entre q-bits participantes, criando um registro de q-bits
25: Se não for operação quântica que cause emaranhamento Então
26:   repeat
27:     execute produto tensorial entre operadores participantes
28:   until encontrar a última instrução
29: Fim Se
30: _produtoMatricial:
31: Se operador causa emaranhamento e/ou existe q-bit emaranhado Então
32:   obtenha o operador da memória MR e multiplique-o pelo registro de q-bits
33:   repeat
34:     Multiplique cada operador pelo q-bit especificados na instrução
35:   until encontrar a última instrução
36: Fim Se
37: Retorne a _esperaInstrução

```

5.4.1 Micro-ordens

O conjunto de bits de micro-ordem abarca sinais de controle dos componentes básicos, sinais que determinam o desvio do fluxo condicional ou incondicional, carregamento de registrador com um numeral e o comando de componentes com funções complexas que contém algoritmo próprio. Setenta e quatro bits conformam um arranjo horizontal de micro-ordens. Parte das micro-ordens são utilizadas para gerenciar componentes de UCont, inclusive no controle de

componentes que desempenham função elementares automaticamente.

5.4.2 Flags

Este campo do vetor de *bits* determina o código que representa o *flag*, utilizado nas instruções do microprograma que fazem desvio condicional, que são apenas duas: a que testa se o *flag* especificado está no nível alto e a que verifica se o estado do *flag* está diferente de alto. A quantidade de bits necessária para representar os *flags* é dada pela Equação 52. Atualmente, são usados 10 *flags* descritos a seguir:

$$qtdBits = \lceil \log_2 qtdFlags \rceil \quad (52)$$

1. *endZeroQbRd*: Endereço de leitura em MEQ é igual a zero;
2. *endZeroQbWr*: Endereço para gravação em MEQ é igual a zero;
3. *semDesc*: Lista de instruções está vazia;
4. *op + 1*: Há mais de um operador quântico especificado na operação quântica corrente;
5. *ultDesc*: Seleção da última instrução da operação quântica atual;
6. *bitEm*: Sinaliza que no conjunto de *q-bits* especificados existe pelo menos um emaranhado;
7. *endZeroMRRd*: Endereço de leitura em MR é igual a zero;
8. *endZeroMRWr*: Endereço para gravação em MR é igual a zero;
9. *fimPTOp*: Término do produto tensorial entre os operadores elencado nas instruções;
10. *fimPTQb*: Término do produto tensorial entre os *q-bits* especificados nas instruções.

5.4.3 Operando

Representa o endereço do registro do *q-bit* ou posição da memória do microprograma para onde desviar ou um numeral a ser carregado. A quantidade de bits reservada para este campo é aquela suficiente para comportar o maior entre os três tipos de conteúdo. Este campo também serve ao armazenamento de um valor no registrador de endereço da memória MOp, quando se necessita acessar os coeficientes de um operador quântico básico.

5.5 Controladores de produto tensorial

UCont possui componentes especializados para prover controle auxiliar na realização de produtos tensoriais entre q -bits e produtos tensoriais entre operadores quânticos.

5.5.1 Controlador de produto tensorial de q -bits

O componente QbPTCont gerencia o endereçamento de leitura e de gravação na memória MQ os dados de controle dos registros e os multiplexadores e registradores de MCalc. O produto tensorial de q -bits, além da operação matemática, requer o tratamento do registro de controle dos q -bits participantes da operação, pois tais registros formam um lista encadeada em apenas um sentido, lista tal que abarca todos os registros pertinentes à representação matricial como vetor-coluna dos q -bits emaranhados. Todos os registros de q -bits mais os registros das posições de memória complementares, caso sejam necessárias, ostentarão o estado *ativado* no campo *emaranhado*. O campo *qBitInicial* de todos os registros participantes conterão o número do q -bit que primeiro emaranhou-se com outro. O campo *qBitPosterior* de cada registro guardará o número do registro que o segue, exceto o último, que armazenará um valor que indica fim da lista. Uma operação que envolva q -bits emaranhados, seja produto tensorial ou matricial, tratará o conjunto de registros encadeados sempre a partir do primeiro e lerá cada um seguindo a ordem de encadeamento até o último. Os resultados parciais do produto tensorial de um grupo de registros associados a n q -bits emaranhados com o q -bit seguinte são disponibilizados às saídas de MULT1 e MULT2 e colocados no barramento de dados para gravação de MQ. QPTCont comanda a multiplicação tensorial entre os coeficientes de um conjunto de registros de q -bits com os coeficientes do registro de um q -bit. Um conjunto de registros de q -bits refere-se a 1 ou mais q -bits. A quantidade necessária de registros na memória MQ para armazenar os coeficientes do vetor-coluna que representa os q -bits emaranhados é dada pela Equação 53.

$$qtdEnd = 2^{q-1}, \quad (53)$$

sendo q a quantidade de q -bits descritos pelo conjunto de registros. Os números dos q -bits participantes do produto tensorial são providos pela instrução quântica, composta por 2 ou mais instruções. As instruções são interpretados consecutivamente a partir do primeira instrução da operação quântica corrente. A cada iteração do produto tensorial, o conjunto de registros dos q -bits tensorialmente multiplicados até a iteração anterior serve como fator para o produto tensorial corrente, no qual o outro fator é dado pelo registro especificado na instrução corrente. QbPTCont recebe o sinal *inicieProdutoTensorial*, que quando no nível alto,

faz QbPTCont assumir o controle dos componentes do coprocessador. QbPTCont recebe os seguintes parâmetros:

1. *qtdDeOperacoesNaInstrucao*: quantidade de operadores descritos na operação quântica corrente;
2. *numeroDoQbitDoPrimeiroOperador*: número do *q-bit* definido na primeira instrução da operação quântica corrente;
3. *qBitDesc*: número do *q-bit* da instrução corrente;
4. *controleDeQbitsDeLeitura*: dados do registro de controle do *q-bit* referenciada pela instrução corrente;
5. *microInstrucao*: via de dados das micro-ordens;
6. *controleDeQbitsDeGravacao*: dados do registro de controle do *q-bit* onde será gravado o novo conteúdo dos dados de controle;
7. *numeroDeQBitDeGravacao*: endereço do *q-bits* onde os dados serão gravados;
8. *enderecoACarregar*: o endereço a carregar no registrador de numero de *q-bit*.

QbPTCont recebe o *flag fimProdutoTensorial* e externa *mqWrLoad*, que habilita a gravação em MEQ; O registrador MEQRdReg guarda o endereço a ser acessado na memória MEQ, sendo constituído por um contador crescente com *preset*, servindo ao automatismo de endereço sequencial em procedimento de inicialização, leitura sequencial na inicialização dos *q-bits*, para a determinação do estado da máquina quântica. Já a capacidade de *preset* permite que o registrador armazene o endereço alvo de uma escrita ou leitura isolada ou, se em grupo de 2 ou mais registradores de *q-bit*, que não obedeça uma sequência consecutiva de endereços, como pode ocorrer em um produto tensorial entre *q-bits* e com produto matricial envolvendo dois ou mais *q-bits*. Os sinais de controle são:

1. *countEnabled*: habilita a contagem interna utilizada no endereçamento;
2. *reset*: retorna o contador interno ao valor inicial, preparando-no para uma nova sequência de endereçamentos;
3. *load*: carrega um endereço específico.

MEQRdReg recebe o parâmetro *numeroDoQBit*, o número relativo do *q-bit* a ler, e exterioriza *endQBit*, o valor a ser colocado no barramento de endereço de MEQ. O sinal *outEnabled* habilita a colocação do dados desta memória no barramento.

Analogamente a MEQRdReg, o registrador MEQWrReg armazena o endereço da memória de estados onde gravar. MEQWrReg é um contador crescente com *preset*, viabilizando um sequenciamento de endereços quando da inicialização de *q-bits*, leitura sequencial na inicialização dos *q-bits*, para a determinação do estado da máquina quântica. Já a capacidade de *preset* permite que o registrador armazene o endereço alvo de uma escrita ou leitura isolada ou, se em grupo de 2 ou mais registradores de *q-bit*, que não obedeça uma sequência consecutiva de endereços, como pode ocorrer em um produto tensorial entre *q-bits* e com produto matricial envolvendo dois ou mais *q-bits*. Os sinais de controle são: *countEnabled*; *reset*; *load*. Recebe o parâmetro *numeroDoQBit*; Externa *endQBit*, o valor a ser colocado no barramento de endereço de MEQ; e *outEnabled*.

5.5.2 Controlador de produto tensorial de operadores quânticos básicos

O componente OpPTCont controla o sequenciamento de endereço de leitura e de escrita da memória MR e a temporização destas operações. Comandado o início, este componente utiliza endereços relativos, do último para o primeiro par de coeficientes que representa o operador quântico armazenado na memória MR. Tal endereço relativo é controlado por OpPTCont, cabendo ao conversor de leitura a determinação do endereço real. O endereço relativo com o qual se inicia a leitura depende da quantidade de *q-bits* para a qual o operador quântico atualmente em MR. A quantidade de endereços ocupados em MR pelo operador corrente *qtdEnd* é dada por $qtdEnd = 4^q$, onde q é a quantidade de *q-bits* que o operador corrente trata. Desta forma, este controlador endereça (endereço relativo) na memória MR desde $4^q - 1$ até 0. Quando da construção do operador para dois *q-bits*, os dois operadores multiplicados tensorialmente são oriundos da memória MOp, sendo o resultado gravado na memória MR. Na construção de operador para três ou mais *q-bits*, o operador lido da memória MR é multiplicado tensorialmente com o operador provido pela memória MOp requer, cujo resultado é gravado novamente em MR. A leitura e escrita em MR eventualmente ocorre de forma simultânea, porém sempre em endereços diferentes. Durante a construção do operador para dois ou mais *q-bits*, o controlador de produto tensorial de operadores quânticos OpPTCont referencia os coeficientes do operador seguindo uma sequência de endereço que inicia na primeira dupla de coeficientes da primeira linha e termina na última dupla de coeficientes da última linha.

Como exemplo, toma-se um operador NOT para operar dois q -bits (matriz 4×4), mostrado na Equação 54.

$$NOT \otimes NOT = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad (54)$$

Devido ao fato de os coeficientes das colunas ímpar e par ocuparem um mesmo endereço em MR (bem como em MOp), a tabela própria para conter uma dupla de coeficientes em cada célula, reduz-se à dimensão 2×4 , como se vê na Tabela 13.

Tabela 13: Operador NOT para 2 q -bits: distribuição dos coeficientes aos pares

0 0	0 1
0 0	1 0
0 1	0 0
1 0	0 0

Desta forma, numera-se a partir de zero a primeira célula da primeira linha, percorrendo-se as células da direita para a esquerda e de cima para baixo. Identicamente, operadores para três ou mais q -bits seguiriam a mesma regra de numeração. A memória MR pode ser abstraída como uma tabela cujos endereços obedecem a mesma regra explicada acima. Nas tabelas que se seguem, mostrar-se-ão exemplos da organização de endereços da memória MR dimensionada para 2, 3 e 4 q -bits. Uma memória MR dimensionada para dois q -bits teria os endereços organizados como se vê na Tabela 14.

Tabela 14: Endereços da memória MR dimensionada para 2 q -bits

0	1
2	3
4	5
6	7

Se dimensionada para armazenar operadores para até três q -bits, a abstração tabular da memória MR condiz com a Tabela 15. Operadores para três q -bits requerem 32 endereços. Operadores até 4 q -bits requerem que a memória MR disponha de 128 endereços.

A memória MR preparada para armazenar operadores para até quatro q -bits, teria os endereços seguindo a ordem mostrada na Tabela 16.

Quando o operador armazenado na memória MR é próprio para atuar sobre uma quantidade de q -bits menor do que aquela que o maior operador comportável por MR, os endereços alocados para este operador para menos q -bits ocupam uma faixa não contígua de endereços

Tabela 15: Endereços da memória MR dimensionada para 3 q -bits

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31

Tabela 16: Endereços da memória MR dimensionada para 4 q -bits

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127

absolutos da memória MR, conforme se apresenta nas tabelas a seguir. Supondo-se uma memória MR dimensionada para operadores para três q -bits, os coeficientes de um operador para 2 q -bits ocuparia os endereços absolutos de MR marcados em negrito na Tabela 17.

No caso de uma memória MR dimensionada para operadores para quatro q -bits, os coeficientes de um operador para dois e para três q -bits, respectivamente estariam presentes nos endereços absolutos de MR marcados em negrito na Tabela 18 e 19.

Cabe a OpPTCont ativar os devidos sinais de controle de UCalc (multiplexadores e registradores) para que os coeficientes sejam colocados nas entradas de MULT1 e MULT2. OpPTCont controla o sinal de habilitação de gravação em MR dos resultados parciais do produto tensorial, de modo que somente fique no nível alto 4 ciclos de clock após o início do produto tensorial, que é a latência do multiplicador de números complexos. OpPTcont fornece ao conversor de gravação dois sinais que participam da determinação do endereço de gravação em MR: ordem da coluna do coeficiente selecionado do primeiro operador e ordem da linha do

Tabela 17: Endereços ocupados por operador para 2 q -bits - MR dimensionada para 3 q -bits

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31

Tabela 18: Endereços ocupados por operador para 2 q -bits - MR dimensionada para 4 q -bits

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127

coeficiente selecionado no segundo operador. OpPTCont fornece um dos sinais de controle que participam da habilitação da contagem dos registradores de endereço de leitura e de gravação de MR.

5.5.2.1 Controlador de gravação de produto tensorial em MR

O controlador de gravação de produto tensorial em MR, OpPTRecCont, comanda o sequenciamento dos endereços de MR onde deverão ser gravados os resultados dos produtos tensoriais de operadores. Utiliza o maior endereço relativo do operador quântico seguinte, provido por OPPTMaxReg, e do contador de endereço relativo de gravação, MRWrCont. O sinal *countEnable* é uma micro-ordem fornecida pelo microprograma para ativar este controlador, iniciando a contagem ciclos de *clock* para que a gravação dos resultados em MR seja habilitada, atendendo a latência do multiplicador de números complexos para disponibilização do produto. O sinal *maxCount* é colocado no nível alto pela micro-instrução. Finda a contagem regressiva, o sinal *gravEmCurso* é posto no nível alto, sinalizando que a fase de gravação de

Tabela 19: Endereços ocupados por operador para 3 q -bits - MR dimensionada para 4 q -bits

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127

resultados iniciou-se. Alcançado o endereço relativo 0, *zeroCount* é ativado por MRWrCont e *gravEmCurso* é inicializado.

5.5.2.2 Registrador auxiliar de quantidade de q -bits do operador quântico

MRQbCont é um contador crescente que desempenha a função de um registrador e que provê a quantidade de q -bits do operador corrente para MRRdCont, MRWrCont, MRPTRdConv e MRPTWrConv, para que possa ser determinado o maior endereço relativo deste operador em construção. Ao fim de cada iteração da série de produtos tensoriais entre operadores, MRQbCont é incrementado. Quando da início da construção do operador para 2 ou mais q -bits, este componente tem seu valor de saída inicializado, isto é, valor 2. MRQbCont recebe dois sinais de controle: *reset*, que estabelece o valor mínimo 2, e *incLimite*, que incrementa de uma unidade o valor à saída deste componente.

5.5.2.3 Conversores de endereço dos coeficientes na memória MR

MRPTRdConv é o componente responsável pelo cálculo do endereço real de leitura em MR a partir do endereço relativo provido por MRRdCont utilizado na construção do operador quântico por meio do produto tensorial.

Em cada endereço de MR existe uma dupla de coeficientes, correspondendo a 2 colunas adjacentes da mesma linha da matriz quadrada que representa o operador. A numeração das duplas de coeficientes inicia-se nas colunas 1 e 2 da linha 1 do operador, percorrendo todas as duplas de coeficientes desta linha e continuando na linha subsequente, até os coeficientes

da n -ésima - 1 e n -ésima coluna da n -ésima linha, sendo n a quantidade de linhas e colunas da matriz quadrada, cujo valor depende da quantidade de q -bits a ser tratada pelo operador quântico em construção, dada pela Equação 55.

$$dim = 2^{qtdBitsSim} \quad (55)$$

A quantidade de endereços de MR a serem lidos para construir o operador quântico é $n^2/2$. O endereço em MR de onde a dupla de coeficientes deve ser lida é dada por $endDest$, calculado por:

$$endDest = ((endOrig - (endOrig \bmod(2^{qBOp-1}))) \cdot (2^{maxQB-qBOp})) + (endOrig \bmod(2^{qBOp-1})) \quad (56)$$

MRPTRdConv recebe os parâmetros $qtdDeBitsDoOperadorCorrente$ e $endereçoDeOrigem$, o endereço relativo a ser convertido. À saída, $endereçoDestino$ é o endereço absoluto de MR a ser lido. Este conversor trabalha em paralelo com o conversor MRPTWrConv, pois o produto tensorial de operadores prevê a leitura e gravação simultânea em MR.

Análogo a MRPTRdConv, o conversor de endereço de gravação MRPTWrConv calcula o endereço de MR. O endereço destino $endDest$ é calculado conforme mostra Equação 57.

$$endDest = ((endOrig - (endOrig \bmod(2^{qBOp-1}))) \cdot (2^{maxQB-qBOp})) + (endOrig \bmod(2^{qBOp-1})) \quad (57)$$

sendo $endOrig$ o endereço de origem, $qBOp$ a quantidade de q -bits do operador corrente e $maxQB$ a máxima quantidade de q -bits tratável por um operador no coprocessador. Os sinais de controle de MRPTWrConv são:

- $qtdDeQBitsDoOperador$: Informa a quantidade de *bits* do operador;
- $paridadeDaLinha$: Informa a paridade da linha;
- $paridadeDaColunaImpar$: Indica a paridade da coluna;
- $endereçoRelativoCorrenteDeGravacao$: Informa o endereço relativo corrente de gravação;
- $primeiroProdutoTensorial$: informa que está em curso o primeiro produto tensorial de operadores, ativando a compensação do cálculo de endereços de gravação em MR.

Este componente externa $endereçoAbsolutoDeGravacao$, que será aplicado ao barramento de endereço de gravação de MR. Este conversor opera simultaneamente com o conversor MRPTRdConv, pois o produto tensorial de operadores com 3 ou mais q -bits implica na leitura e gravação simultâneas em MR.

5.5.2.4 Registradores de endereço

O registrador de endereço OpAdReg da memória de operadores (MOp) armazena o endereço relativo da dupla de coeficientes do operador quântico em construção. Seu conteúdo é utilizado pelos conversores MRPTRdConv e MRPTWrConv para determinar o endereço da dupla de coeficientes a ler e escrever em MR, respectivamente. Em realidade, OpAdReg é um contador crescente com *preset*, contribuindo para o automatismo de endereçamento sequencial em procedimento de produto matricial de operador para 1 *q-bit* e produto tensorial de operadores. Já a capacidade de *preset* possibilita o endereçamento em MOp dos coeficientes da primeira linha do operador básico, a partir da qual ocorrerão as leituras de coeficientes. OpAdReg recebe os sinais de controle *countEnable* e *loadEnable*, possui entrada de *preset* para carregar o endereço inicial do operador quântico básico e gera *flag* que sinaliza quando atingiu o último endereço do operador quântico selecionado.

O registrador de endereço de leitura da memória MR, MRRdCont, é um contador decrescente com *preset*. O fato de ser um contador em vez de um mero registrador permite embutir um automatismo básico, simplificando o microprograma. O endereçamento de MR ocorre no sentido decrescente a partir do endereço onde encontra-se o último coeficiente do operador quântico corrente. MRRdCont provê o endereço relativo do operador quântico a ser convertido em endereço real de MR pelo componente ConvRdPT. MRRdCont recebe os sinais de controle *countEnable*, habilitando a contagem interna a ser utilizada nos endereços de leitura, e *reset*, este estabelecendo o maior endereço relativo do operador a ser lido em MR. MRRdCont recebe o parâmetro *qtdDeQBitsDoOperador*, necessário ao cálculo do maior endereço relativo do operador a ser lido de MR. MRRdCont externa o parâmetros *endereçoRelativoCorrente*, o valor do endereço relativo atual, e os *flags fimDeContagem* e *primeiroProdutoTensorial*, este sinalizando que o primeiro produto tensorial de operadores está em curso, compartilhado com MRWrCont.

Análogo a MRRdCont, MRWrCont é o registrador de endereço de gravação da memória MR é um contador *down* com *preset*, possibilitando desempenhar um automatismo simples de decremento, livrando o microprograma desta incumbência. O endereçamento de MR durante a gravação também se dá no sentido decrescente a partir dos últimos coeficientes do operador quântico seguinte, ou seja o operador que está em construção, que trata uma quantidade de *q-bits* 1 unidade superior ao operador quântico que está sendo lido. MRWrCont provê o endereço relativo do operador quântico a ser convertido em endereço real de MR pelo componente ConvWrPT. MRWrCont recebe os sinais de controle *countEnable* e *reset*, este estabelecendo o maior

endereço relativo do operador a ser escrito em MR, e o parâmetro *qtdDeQBitsDoOperador*, necessário ao cálculo do maior endereço relativo do operador a ser gravado em MR. Este componente externa o parâmetro *enderecoRelativoCorrente*, o endereço relativo onde gravar na memória MR, e os *flags fimDeContagem*, indicando conclusão do processo de endereçamento, e *primeiroProdutoTensorial*, que sinaliza o curso do primeiro produto tensorial entre operadores, compartilhado por MRdCont.

5.6 Considerações finais do capítulo

Apresentou-se neste capítulo as operações da Unidade de Controle, o elemento da arquitetura que gerencia todos os outros componentes e a si próprio. Junto com a Unidade de Cálculo, UCont formam o coração do coprocessador. UCont realiza o controle por meio de micro-instruções e componentes especializados, ora o controle sendo delegado a estes, ora sendo feito um gerenciamento paralelo.

Capítulo 6

RESULTADOS DE SIMULAÇÃO

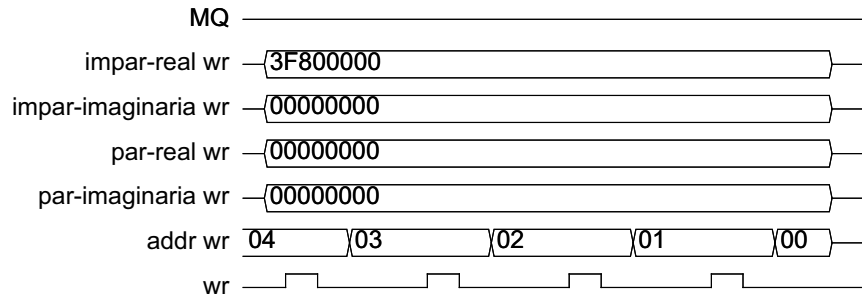
SIMULAÇÕES apresentadas neste capítulo mostram o funcionamento dos algoritmos de inicialização, produto matricial entre um operador quântico básico para 1 q -bit e um q -bit e o produto tensorial entre operadores. Foi arbitrado que a configuração do Coprocessador utilizado nesta simulações possuísse 4 q -bits e capaz de realizar operações ora sobre 3, ora sobre 4 q -bits pois entendeu-se que desta forma poderia-se mostrar adequadamente as operações e a escalabilidade do projeto sem resultar na geração de diagramas de tempos extensos em demasia. Nas diversas figuras deste Capítulo veem-se diagramas de tempos com vias de dados apresentando recorrentemente valores hexadecimais “3F800000” e “00000000”, respectivamente correspondendo aos valores 1,000 e 0,0000 no padrão IEEE754.

6.1 Inicialização dos q -bits

No início do funcionamento do Coprocessador e na eventualidade de um *reset*, os q -bits do Coprocessador são inicializados com o ket $|0\rangle$ ou estado 0 quântico, sendo representado matricialmente por:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (58)$$

Na Figura 25 vê-se o diagrama de tempos da inicialização das posições da memória MQ. O sinal *wr* habilita a gravação na memória MQ e, durante sua asserção, o sinal *impar-real wr* apresenta o valor $1 + 0i$ e os sinais *impar-imaginaria wr*, *par-real wr* e *par-imaginaria wr* o valor $0 + 0i$, bem como o sinal mostra o endereço selecionado.

Figura 25: Inicialização dos q -bits

6.2 Operações quânticas

As operações quânticas são solicitadas pelo processador PROC e podem ser feitas sobre 1 ou mais q -bits, conforme especificação contida no descritor(es) da instrução. Conforme a quantidade de q -bits participantes e a presença ou não de pelo menos 2 q -bits emaranhados, ou se a operação quântica é do tipo que causa emaranhamento, o tratamento dispensados às solicitações de PROC são diferenciados. Cada uma das instruções é codificada no sinal *instrucaoCorrente*, que ver-se-á em diversos diagramas de tempos. Uma instrução é mostrada no gráfico como três valores agrupados por chaves. Por exemplo: {02}{01}{1}. O primeiro campo ({02} neste exemplo) especifica o código da operação quântica; o segundo campo ({01}), o número do q -bit-alvo da operação quântica; finalmente, o último campo ({01}) define se a instrução corrente é a última da operação, assumindo o valor 0, ou se não é a última, ostentando o valor 1.

6.2.1 Operação quântica sobre 1 q -bit

Operação quântica sobre 1 q -bit é o produto matricial entre um operador quântico básico e o vetor-coluna que representa o q -bit alvo. Os coeficientes do operador quântico são lidos de MOp e os do q -bit são oriundos de MQ. Neste exemplo será aplicada a operação NOT quântica (Equação 59) sobre um q -bit ainda colapsado no estado $|0\rangle$ (Equação 58). A representação matricial desta operação é apresentada na Equação 60. A Figura 26 exhibe o diagrama de tempos desta operação. O sinal *solicitacaoDeInstrucao* comanda o componente *InstGer* a disponibilizar os dados da instrução que, no caso deste exemplo com operador NOT sobre 1 q -bit, apenas existe uma instrução. O sinal *listaDeInstrucaoVazia* assume o nível baixo, pois a instrução corrente ainda está em curso. O sinal *encontrouUltimoOperadorDaInstrucao* nível alto indica que a instrução corrente é a última da operação quântica. O sinal *instrucaoCorrente* é um registro com três campos. Neste exemplo, o primeiro possui valor 2, o código da operação NOT neste Coprocessador; o segundo campo contém o valor 2, o número do q -bit-alvo; e o último campo

é um *flag* cujo valor 0 indica que é a última instrução da operação quântica. Da memória MQ são lidos os coeficientes do citado *q-bit* que encontra-se no estado $|0\rangle$ (vetor-coluna com $1 + 0i$ na primeira linha e $0 + 0i$ na segunda). Da memória MOp são obtidos os coeficientes do operador NOT: na primeira leitura, leem-se os coeficientes da primeira linha do operador ($0 + 0i$ e $1 + 0i$). Na segunda leitura, os coeficientes da segunda linha ($1 + 0i$ e $0 + 0i$). Primeiramente é executado o produto matricial da primeira linha do operador com o vetor-coluna: $[(0 + 0i).(1 + 0i) + (0 + 0i).(1 + 0i)]$, resultando em $0 + 0i$. Em seguida, realiza-se o produto matricial da segunda linha do operador com o vetor-coluna: $[(1 + 0i).(1 + 0i) + (0 + 0i).(0 + 0i)]$, resultando em $1 + 0i$. Os resultados são gravados na memória MQ.

$$NOT = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{59}$$

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{60}$$

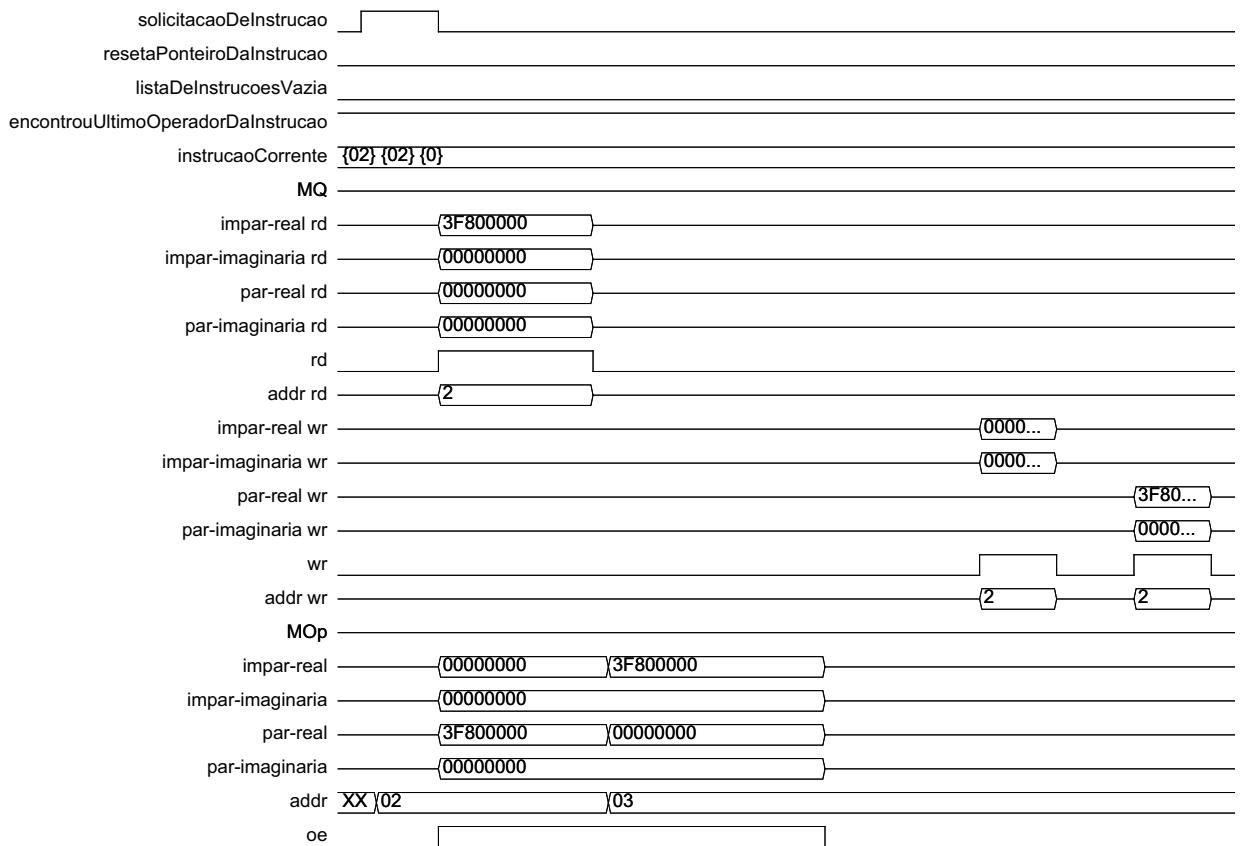


Figura 26: Produto matricial com operador com 1 *q-bit*

6.2.2 Operação quântica sobre 2 ou mais q -bits não-emaranhados

Operação quântica sobre n q -bits não-emaranhados, $n \geq 2$ é resolvida como n operações quânticas sobre 1 q -bit cada, realizadas sequencialmente, com funcionamento análogo àquela descrita na Subseção 6.2.1. Neste exemplo será aplicada a operação NOT quântica sobre dois q -bits não emaranhados e colapsados no estado $|0\rangle$ (Equação 58). O operador NOT para 2 q -bits é mostrado na Equação 61. O vetor-coluna formado pelo produto tensorial de 2 q -bits, cada um no estado $|0\rangle$, é apresentado na Equação 62. A representação matricial desta operação aparece na Equação 63.

$$NOT \otimes NOT = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad (61)$$

$$|0\rangle \otimes |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (62)$$

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (63)$$

Utilizando a propriedade do produto tensorial apresentada na Equação 24 do Capítulo 2, faz-se a operação NOT sobre 2 q -bits em duas etapas, ou seja, duas operações NOT (Equação 60), sobre cada um dos q -bits, separadamente. O produto tensorial entre os vetores-coluna resultantes de cada operação NOT sobre 1 q -bit produz o vetor-coluna mostrado na Equação 62, em concordância com a citada propriedade do produto tensorial. Na Figura 27 apresenta-se o diagrama de tempos das 2 operações NOT sobre 1 q -bit desde a solicitação de instruções. A Figura 28 mostra de forma ampliada o período de tempo relativo às 2 operações. As Figuras 29 e 30 exibem, respectivamente, o diagrama de tempos da primeira e da segunda operação NOT. Cada uma das duas instruções descritivas da operação NOT, codificada no sinal *instrucaoCorrente*, apresenta-se como na lista abaixo:

1. {02}{01}{1}: operação NOT sobre o q -bit 1. Instrução não é a última desta operação.
2. {02}{02}{0}: operação NOT sobre o q -bit 2. Instrução é a última desta operação.

Inicialmente, O sinal *listaDeInstrucaoVazia* mostra-se em nível alto até que uma instrução seja apresentada por PROC. O sinal *solicitacaoDeInstrucao* é ativado tantas vezes até que

se encontre a última instrução na operação quântica corrente, verificando se na instrução corrente existe pelo menos 1 q -bit emaranhado. Neste exemplo, ocorre por 2 vezes. Não existindo q -bit emaranhado, o sinal *resetaPonteiroDaInstrucao* é ativado de forma que InstGer possa fornecer os dados da primeira instrução na próxima vez que o sinal *solicitacaoDeInstrucao* for comandado. A cada vez que a última instrução da operação quântica estiver selecionada, o sinal *encontrouUltimoOperadorDaInstrucao* apresenta nível alto. Na primeira ativação de *solicitacaoDeInstrucao* depois de o ponteiro de instruções ter sido inicializado, assim como ocorreu na operação quântica sobre 1 q -bit descrito na Subseção 6.2.1. Como no presente exemplo utilizou-se operadores NOT sobre q -bits no estado $|0\rangle$, os cálculos e resultados apresentados na operação sobre 1 q -bit se repetem.

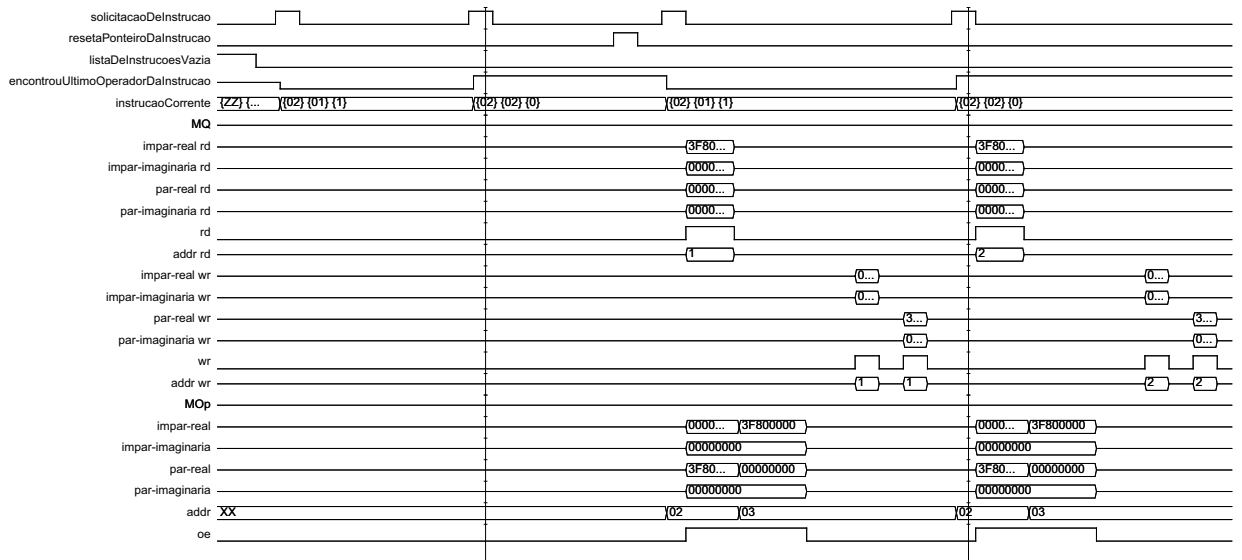


Figura 27: Produto matricial de uma operação NOT sobre 2 q -bits não-emaranhados

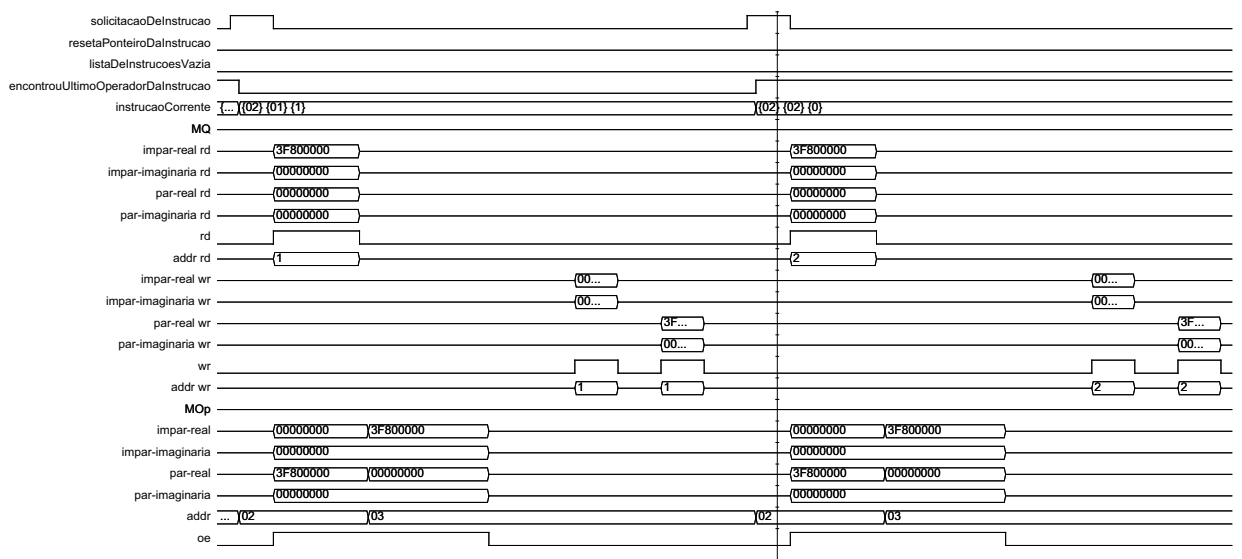


Figura 28: Visão ampliada das duas operações NOT sobre 1 q -bit cada

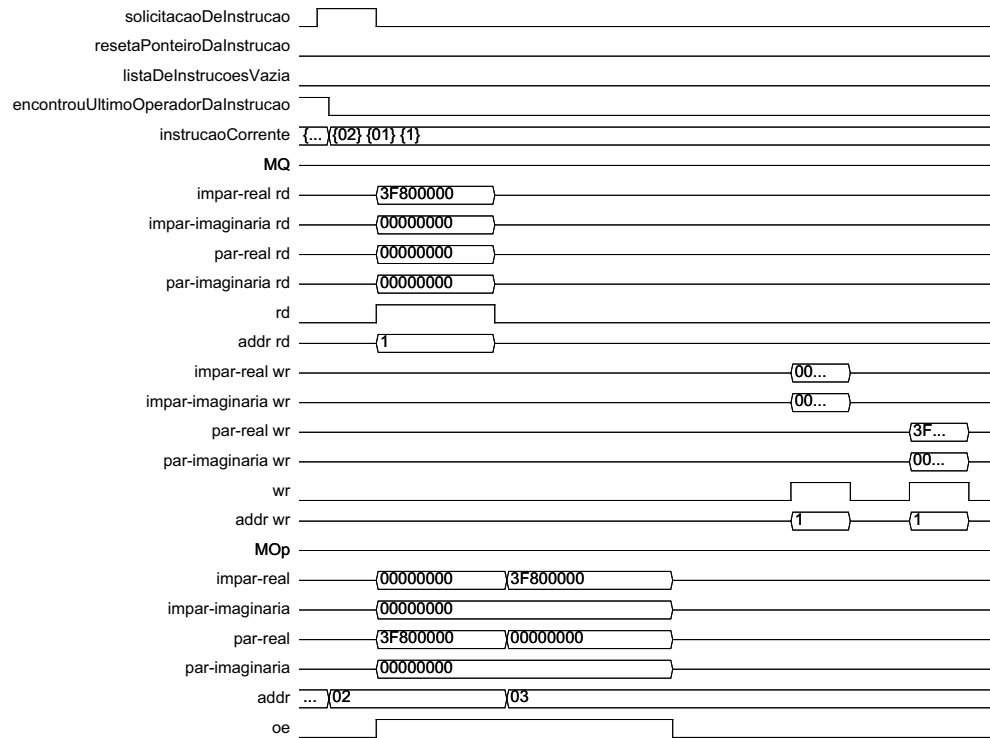


Figura 29: Visão ampliada da primeira operação NOT

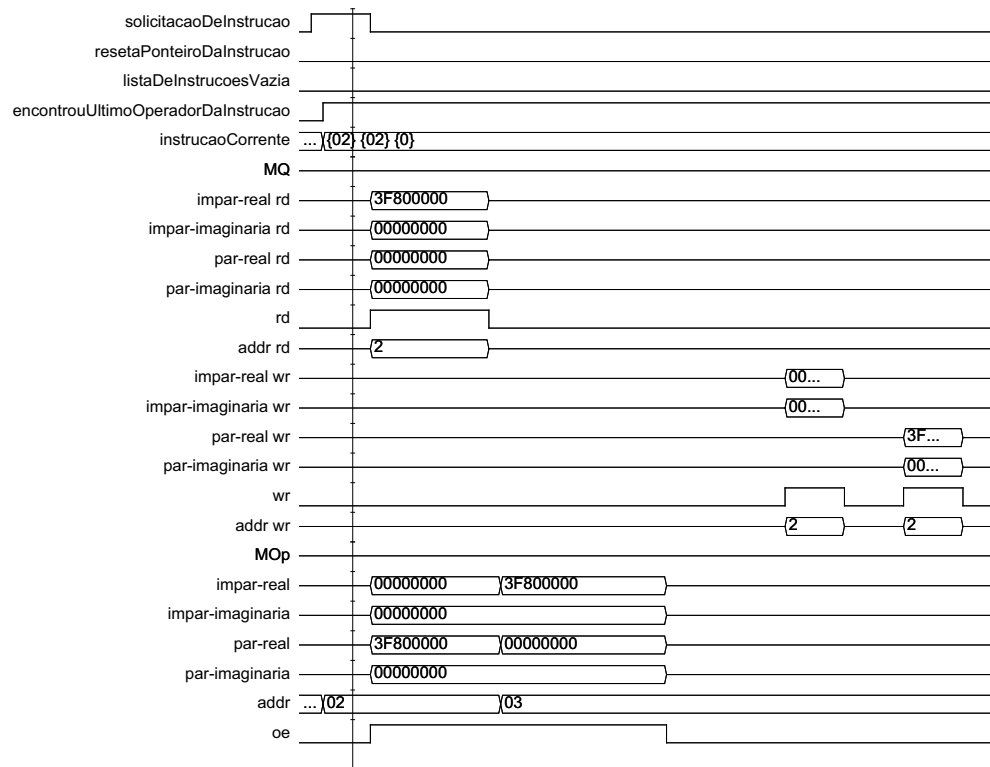


Figura 30: Visão ampliada da segunda operação NOT

6.2.3 Operação quântica sobre 2 ou mais q -bits emaranhados

Quando da execução de uma operação quântica sobre 2 ou mais q -bits em que pelo menos 2 estejam emaranhados e/ou quando a operação quântica causar emaranhamento, a construção do operador quântico por meio de produto tensorial faz-se necessária neste Coprocessador. Nesta seção serão apresentados exemplos de construção de operador para 2 ou mais q -bits a partir de operadores quânticos básicos.

6.2.3.1 Operador para 2 q -bits com memória MR limitada a 4 q -bits

Dispondo de 128 endereços para comportar os 256 coeficientes de um operador para 4 q -bits, a memória MR no presente exemplo somente terá 8 endereços ocupados pelos 16 coeficientes de um operador NOT para 2 q -bits. construído a partir de dois operadores NOT para 1 q -bit. Inicialmente, são obtidos da memória MOp os coeficientes da primeira linha, seguidos dos coeficientes da segunda linha do operador quântico NOT para 1 q -bit, sendo armazenados nos registradores de UCalc. Em sequência, os coeficientes do segundo operador quântico básico são lidos de MOp e armazenados temporariamente por UCalc. O produto tensorial entre estes operadores gerarão os 8 coeficientes a serem armazenados em MR. Como se trata de leitura e escrita em dispositivos diferentes, o endereçamento regressivo de MR não é indispensável, como o que pode ser necessário na geração de operadores para 3 ou mais q -bits, situação em que a memória MR é fonte e destino dos dados. Na Tabela 18 no Capítulo 5, os endereços absolutos em negrito são aqueles ocupados pelos coeficientes calculados para o operador que trata 4 q -bits. O diagrama de tempos desta operação encontra-se na Figura 31.

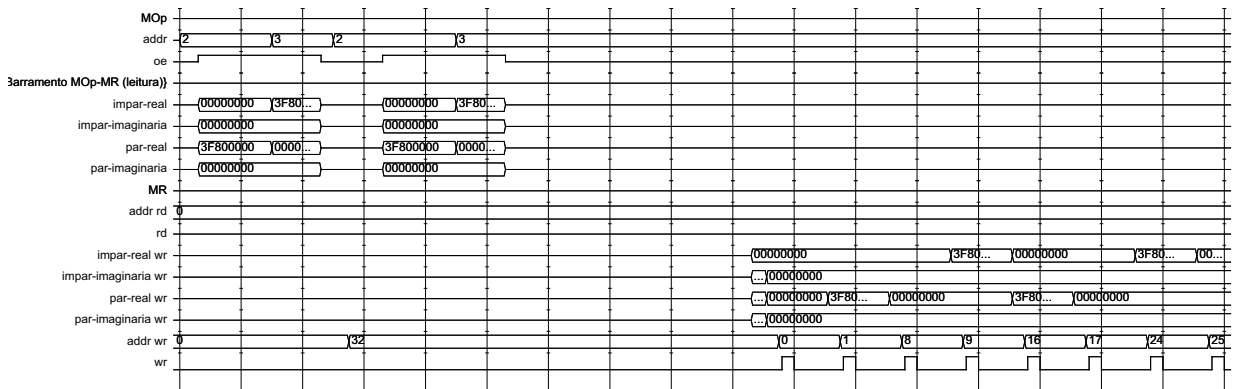


Figura 31: Construção do operador para 2 q -bits e limite de 4 q -bits por operador

6.2.3.2 Operador para 3 q -bits com memória MR limitada a 4 q -bits

A construção do operador para 3 q -bits com a memória MR dimensionada para um operador de 4 q -bits deve ser antecedida pela construção do operador para 2 q -bits, como descrito na Sub-

seção 6.2.3.1. Diferentemente do procedimento anterior, os dados serão oriundos das memórias MR e MOp. O operador para 3 q -bits possui 64 coeficientes, ocupando apenas 32 endereços da memória MR. A Figura 32 mostra o diagrama de tempos desta operação e as Figuras 33, 34 e 35 apresentam trechos expandidos do primeiro diagrama de tempos. Os endereços selecionados para leitura e gravação simultâneas na memória MR são diferentes e condizentes com a Tabela 19 no Capítulo 5.

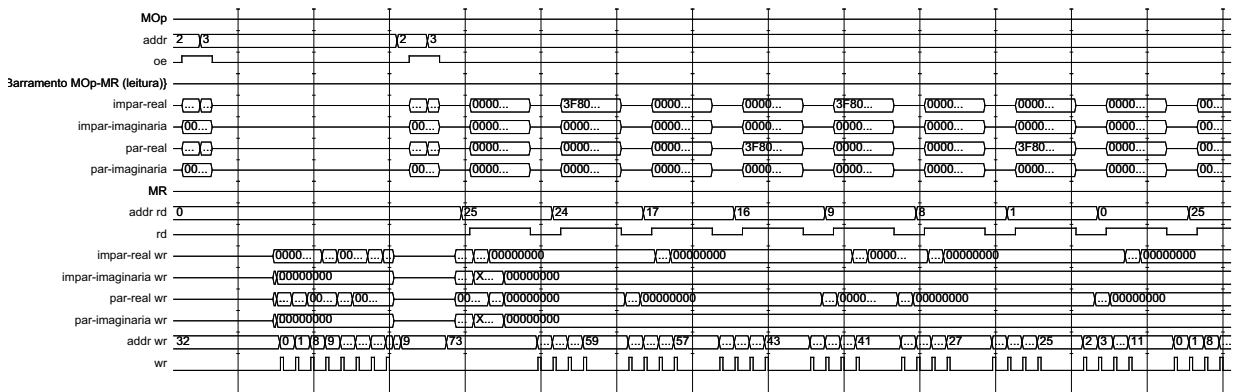


Figura 32: Construção do operador para 3 q -bits e limite de 4 q -bits por operador

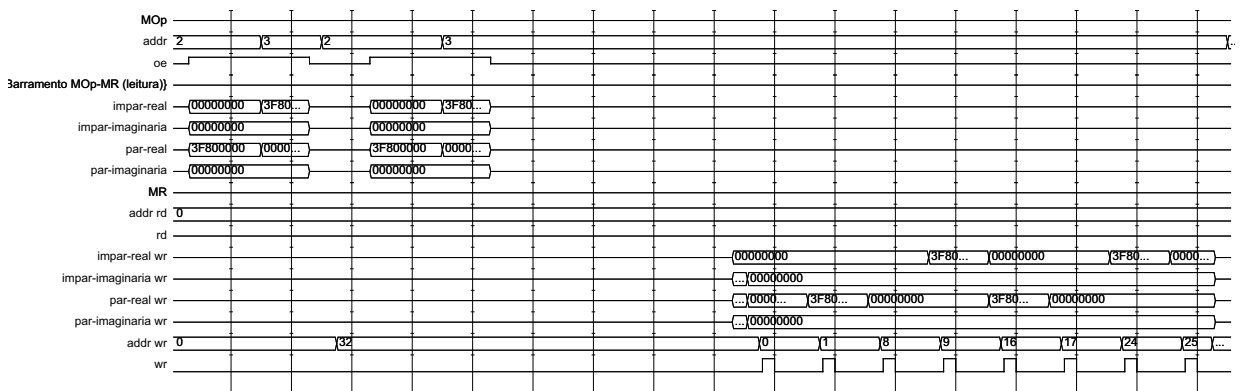


Figura 33: Detalhe da 1ª parte do diagrama de tempos da construção do operador

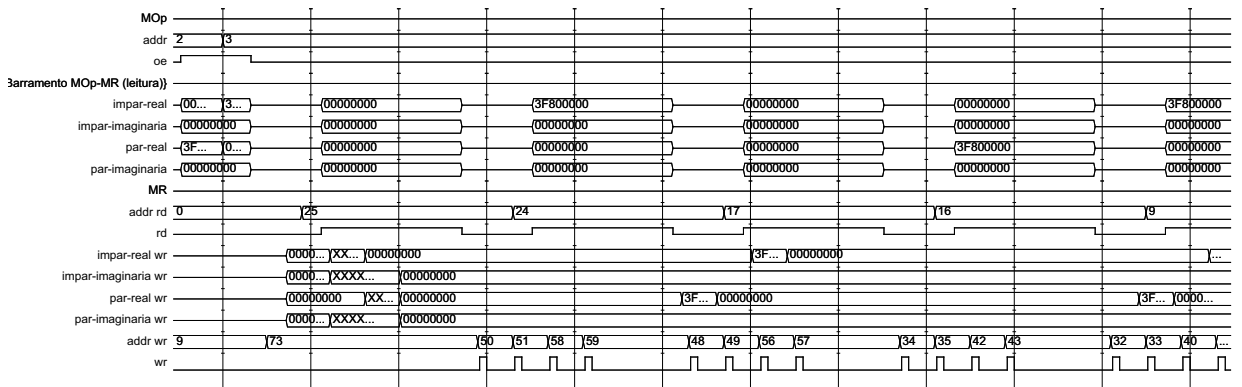


Figura 34: Detalhe da 2ª parte do diagrama de tempos da construção do operador

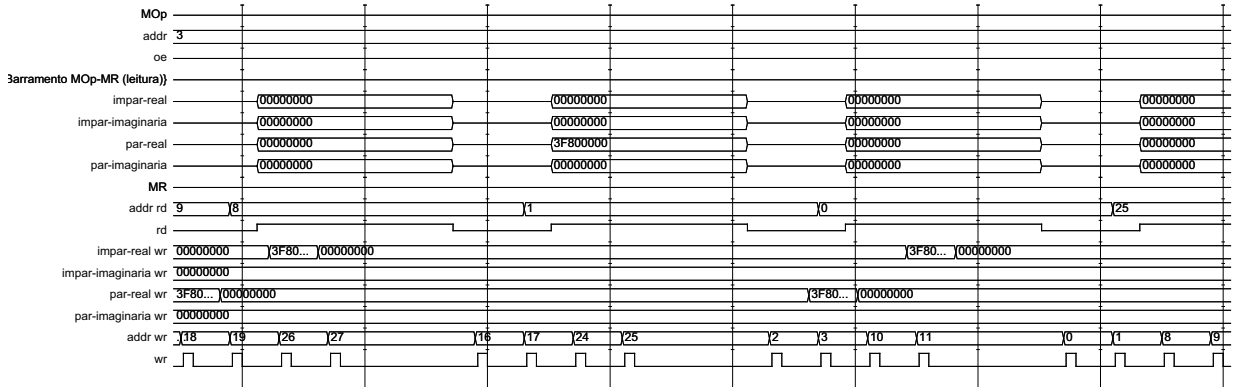


Figura 35: Detalhe da 3ª parte do diagrama de tempos da construção do operador

6.2.4 Produto tensorial de q -bits

Quando q -bits estão emaranhados ou serão alvo de uma operação que causa emaranhamento, as 2^q (onde q é a quantidade de q -bits participantes da operação) posições da memória alocadas os coeficientes do vetor-coluna que representa os q -bits emaranhados, dois coeficientes por endereço. As posições de memória de MEQ que originalmente destinavam-se à guarda dos $kets$ de cada q -bit não emaranhado, passam a guardar duplas de coeficientes que compõem o vetor-coluna de q -bits emaranhados, com o complemento de mais posições da memória MEQ se tratar-se de 3 ou mais q -bits. A Equação 64 mostra na forma matricial o produto tensorial sobre dois q -bits que estão no estado colapsado $|0\rangle$ antes da operação e formam um vetor-coluna com 4 linhas após a mesma.

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (64)$$

Tabela 20: Valores dos coeficientes de q -bits antes e depois de um produto tensorial

Endereço do q -bit	Paridade da linha do coeficiente	Valor antes antes da operação	Valor depois depois da operação
0	ímpar	1,0000	1,0000
0	par	0,0000	0,0000
1	ímpar	1,0000	0,0000
1	par	0,0000	0,0000

A Tabela 20 exhibe os valores das duplas de coeficientes em cada endereço de memória MEQ antes e depois do produto tensorial. Observa-se que neste exemplo as mesmas duas posições de memória são suficientes para armazenar os 4 coeficientes do vetor coluna que representa a dupla de q -bits. A Figura 36 mostra o diagrama de tempos refere-se ao citado

produto tensorial. $\{MQ\ rd\}$ e $\{MQ\ wr\}$ são separadores dos grupos de sinais, respectivamente associados à de leitura e de gravação da memória MQ.

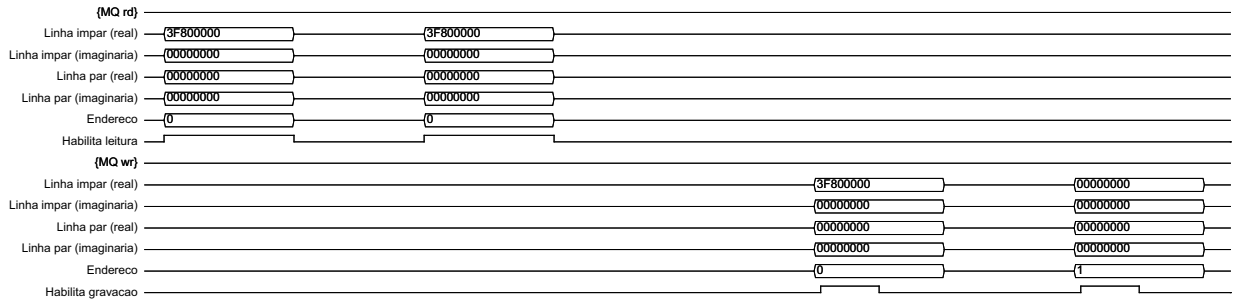


Figura 36: Produto tensorial entre dois q -bits

6.2.5 Produto matricial sobre dois q -bits emaranhados

Uma operação sobre dois ou mais q -bits emaranhados é realizada pelo produto matricial entre um operador construído em tempo de execução e já escrito na memória MR e um vetor-coluna representado pelas posições de memória MQ associadas aos q -bits-alvo. O exemplo utilizado para esta simulação emprega um operador NOT construído (Equação 65) e um registrador $q01$ formado por dois q -bits emaranhados (q -bit 0 e q -bit 1), cujo estado hipotético aparece na Equação 66. A Equação 67 mostra na forma matricial operação NOT sobre $q01$ e o resultado da mesma.

$$NOT = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad (65)$$

$$q01 = \begin{bmatrix} 0,2237 \\ 0,4251 \\ 0,1039 \\ 0,8709 \end{bmatrix} \quad (66)$$

$$NOT(q01) = \begin{bmatrix} 0,8709 \\ 0,1039 \\ 0,4251 \\ 0,2237 \end{bmatrix} \quad (67)$$

Nos diagramas de tempos apresentados nesta Subseção, encontrar-se-ão valores na base hexadecimal que representam aqueles em ponto flutuante no padrão IEEE754, cuja correspondência é apresentada na Tabela 21.

Os diagramas de tempos mostrados nas Figuras 37, 38, 39 e 40 exibem a leitura dos coeficientes dos endereços 0 e 1 da memória MQ e, respectivamente, a leitura dos coeficientes do operador gravado na memória MR nos endereços relativos 0/1, 2/3, 4/5 e 6/7.

Tabela 21: Alguns valores reais e a representação hexadecimal do formato IEEE754

Valor real	IEEE754 em base hexadecimal
0,0000	0x00000000
0,2237	0x3E65119C
0,4251	0x3ED9A6B5
0,1039	0x3DD4C985
0,8709	0x3F5EF34D
1,0000	0x3F800000

$\{MR\ rd\}$, $\{MQ\ rd\}$ e $\{Saida\ de\ UCalc\}$ são separadores dos grupos de sinais, respectivamente relacionados aos barramentos e sinais de controle relativos à leitura da memória MR, barramentos e sinais de controle para leitura da memória MQ e valores nas saídas da Unidade de Cálculo UCalc.

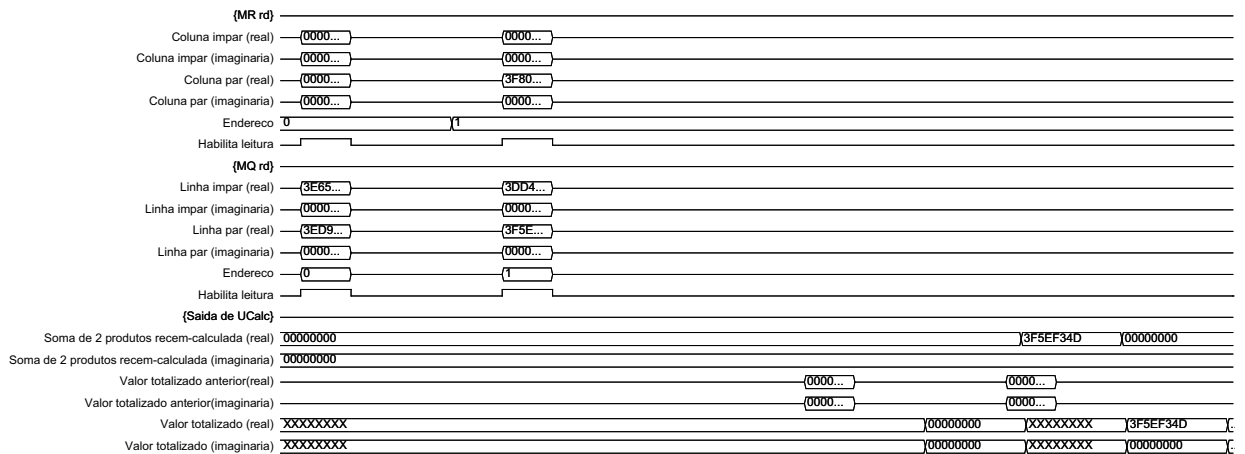


Figura 37: Produto matricial: multiplicações com a 1ª linha do operador

Ao fim da operação quântica, o registrador $q01$ compõe-se dos coeficientes ilustrados na Equação 67, que são escritos na memória MQ, conforme se vê na Figura 41 os sinais do relacionados à gravação. Os quatro coeficientes do registrador $q01$ após a operação NOT estão associados a cada um dos quatro estados possíveis obtíveis como resultado de uma medição do estado quântico. A soma dos quadrados das amplitudes totaliza 1,0000, conforme e Equação 68.

$$0,8709^2 + 0,1039^2 + 0,4251^2 + 0,2237^2 = 0,7585 + 0,0108 + 0,1807 + 0,05 = 1,0000 \quad (68)$$

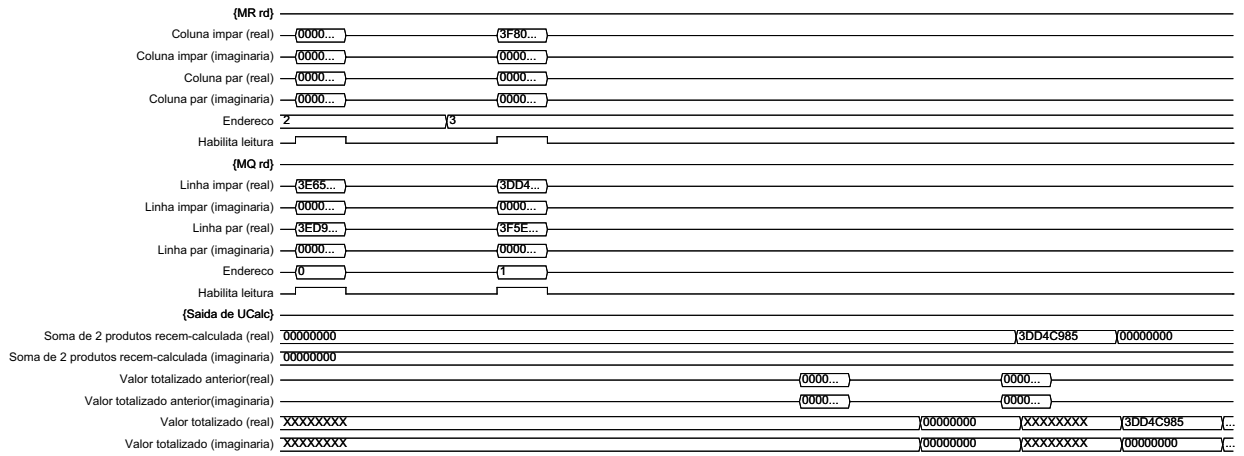


Figura 38: Produto matricial: multiplicações com a 2ª linha do operador

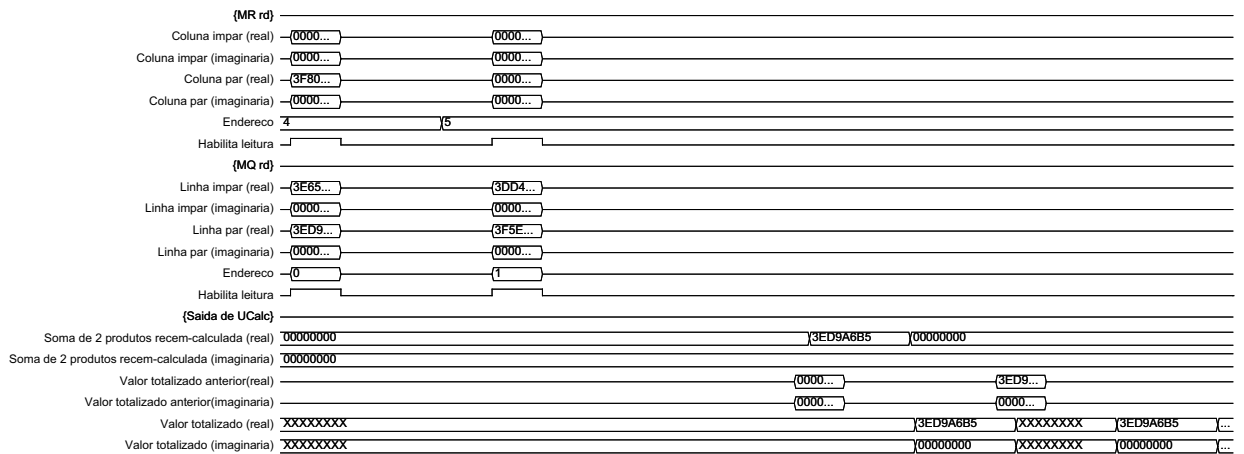


Figura 39: Produto matricial: multiplicações com a 3ª linha do operador

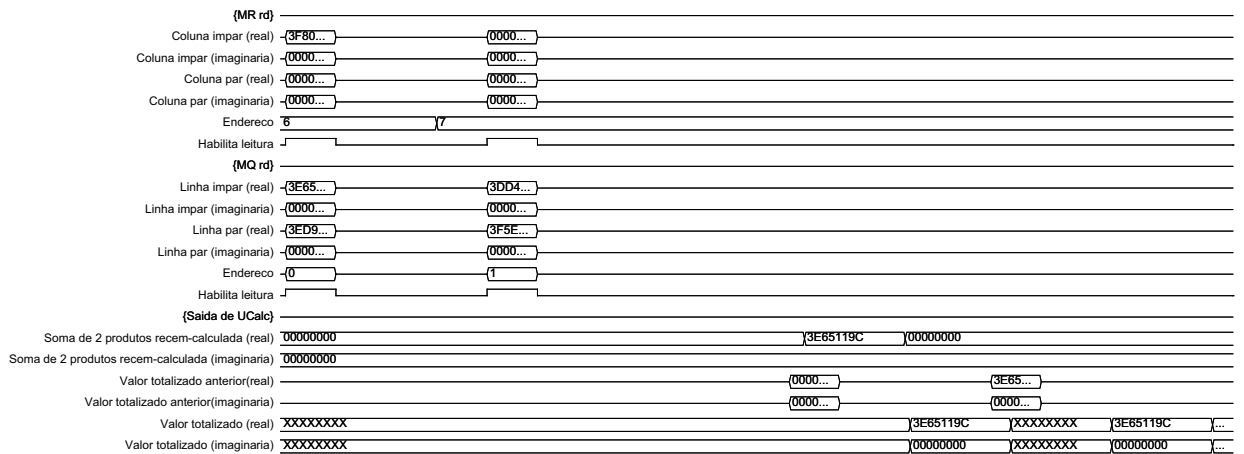


Figura 40: Produto matricial: multiplicações com a 4ª linha do operador

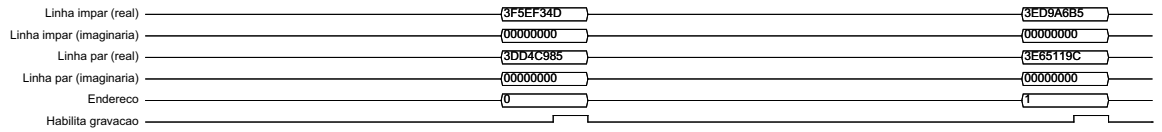


Figura 41: Produto matricial: gravação do resultado da operação na memória MQ

A Tabela 22 apresenta a inter-relação entre faixa de valores, probabilidade e estado quântico.

Tabela 22: Probabilidades associadas a cada estado quântico possível

Ordem da faixa de probabilidade	Início da faixa (inclusivo)	Fim da faixa (exclusivo)	Probabilidade (%)	Estado quântico
1ª	0,0000	0,7585	75,85	00⟩
2ª	0,7585	0,7693	1,08	01⟩
3ª	0,7693	0,9500	18,07	10⟩
4ª	0,9500	1,0000	5	11⟩

Neste exemplo, o medidor de estado quântico UMed recebeu de GNPA o valor 0,754657, pertencente à primeira subfaixa de probabilidades, resultando no estado $|00\rangle$, ou seja, q -bits 0 e 1 com o estado $|0\rangle$, conforme se pode observar na Figura 42, o diagrama de tempos que mostra a aquisição do valor fornecido pelo GNPA e dos coeficientes dos q -bits do Coprocessador, seguida da gravação do resultado na memória MEQ. $\{MQ\ rd\}$, $\{MQ\ wr\}$ e $\{Medidor\ de\ estado\ quântico\ (UMed)\}$ são separadores dos sinais pertencentes a, respectivamente, barramentos e sinais de controle para leitura da memória MQ, barramentos e sinais de controle para gravação da memória MQ e sinais de controle e de dados da Unidade de Medição UMed. A medição do estado quântico encerra um ciclo de operações de um algoritmo quântico, resultando em um estado colapsado para cada q -bit do Coprocessador, estabelecendo um estado equiparável ao de um computador.

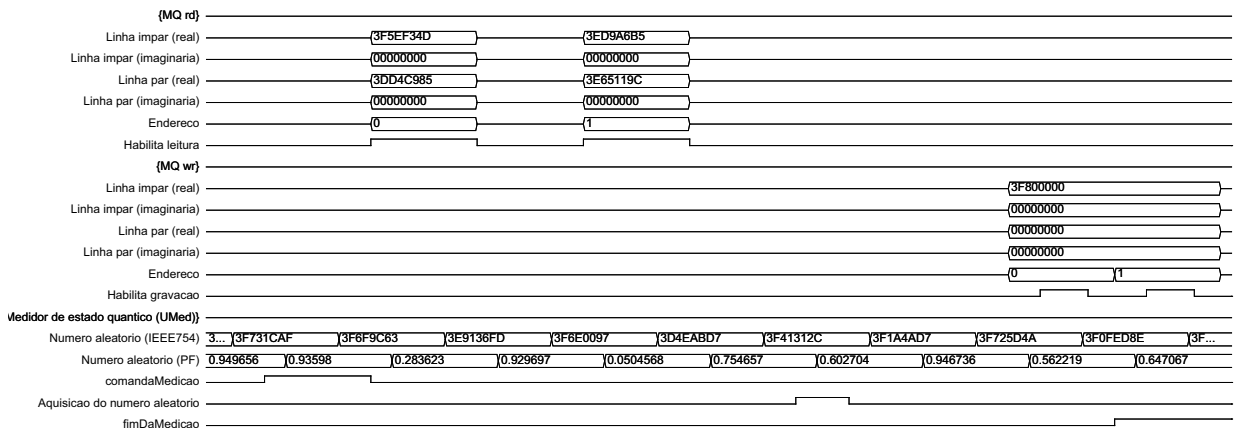


Figura 42: Produto matricial: medição e gravação do estado colapsado na memória MQ

6.3 Considerações finais do capítulo

Apresentaram-se neste capítulo as simulações envolvendo produtos tensoriais entre operadores, produto tensorial entre q -bits, produtos matriciais entre operador e vetor-coluna representativo de q -bits e medição do estado quântico. Mostrou-se como os coeficientes do operador quântico em construção são calculados, utilizando inicialmente os coeficientes dos operadores quânticos básicos, oriundos de MOp, e depois os coeficientes do operador construído na iteração anterior, residente em MR, e os coeficientes do próximo operador básico. Os endereços lidos e escritos em MR, para uma mesma quantidade de q -bits do operador em construção, dependem da quantidade máxima de q -bits que o Coprocessador pode operar simultaneamente. Por intermédio do algoritmo do tipo *roleta* e gerador de números pseudo-aleatórios, pôde-se reproduzir a aleatoriedade do resultado respeitando as amplitudes de cada estado possível.

Capítulo 7

CONCLUSÃO E TRABALHOS FUTUROS

NESTA dissertação apresentaram-se elementos básicos da computação quântica para auxiliar o Leitor na compreensão da arquitetura e das funcionalidades dos componentes do Coprocessador proposto. O emaranhamento de q -bits é um procedimento utilizado nos algoritmos quânticos. Este requer a realização massiva de produtos tensoriais entre operadores e q -bits as multiplicações matriciais entre operadores e vetores-coluna representativos dos q -bits envolvidos, operações estas que, em última instância, são multiplicações com números complexos. Maior a quantidade de q -bits participantes da operação, mais cresce e de forma exponencial a quantidade de multiplicações com números complexos. Este trabalho mostrou uma abordagem implementável em *hardware* com possibilidade de escalabilidade e explorações do paralelismo, contribuindo para a redução do tempo de execução de operação quânticas.

7.1 Conclusões

A macro-arquitetura proposta apresenta número de componentes que possivelmente poderia ser mantido em uma versão do Coprocessador com maior capacidade de processamento. A Unidade de Cálculo, responsável pela execução dos produtos e somas de números complexos do Coprocessador que são realizados em grande número, é o componente da macro-arquitetura que tem maior impacto no desempenho do Coprocessador. A característica modular da Unidade de Cálculo permite que ela possa ser expandida, mas implicando no aumento de complexidade da Unidade de Controle, alargamento da via de dados e, eventualmente, de barramentos.

A Unidade de Controle é o componente que gerencia todos os componentes da via de dados do Coprocessador, utilizando memória de controle e componentes auxiliares, estes para automatizar algumas tarefas elementares e contribuem para simplificar o microprograma.

As simulações validam o funcionamento de algumas operações de exemplo que utilizam o produto matricial e o produto tensorial, baseado em um código escalável e parametrizável. Nos computadores utilizados na simulação, gerenciados por Windows XP[®] ou Windows7[®], dotados de memória RAM de 4GB e utilizando o programa ModelSim[®], foi possível simular um coprocessador com capacidade de operar até 6 q -bits simultaneamente.

O desenvolvimento deste Coprocessador alcançou um nível de desenvolvimento que permite realizar operações quânticas com tantos q -bits não-entrelaçados quanto os recursos computacionais permitam definir. O produto tensorial para mais de dois q -bits e o produto matricial para mais de dois q -bits entrelaçados não puderam ser finalizados ao tempo de apresentação desta dissertação.

7.2 Trabalhos futuros

Além da finalização da implementação das funcionalidades citadas, várias direções em termo de trabalhos futuros poderão ser investigadas:

- o Coprocessador poderá contar com mais multiplicadores na Unidade de cálculo, sempre em potência de 2, de modo a executar mais produtos simultâneos em produto matricial e produto tensorial.
- A organização em *pipeline* da arquitetura para agilizar a operação quântica pelo uso de multiplicadores de números complexos auxiliares para a realização do produto matricial das linhas já calculadas do novo operador em construção, dispensando a espera da conclusão de todos os cálculos requeridos para a construção do operador, como definido no atual projeto do Coprocessador.
- A utilização de uma memória de rascunho com mais portas de leitura e gravação, juntamente com um maior número de multiplicadores de números complexos, viabilizaria a construção do operador quântico em menor tempo.
- A inclusão de uma nova memória de rascunho auxiliar para armazenar operadores construídos para mais de 1 q -bit poderá poupar do recálculo de operadores maiores que fossem utilizados recorrentemente.

REFERÊNCIAS

AARONSON, S. blog, *Quantum-Effect-Demonstrating Beef*. 2011. Disponível em: <http://www.scottaaronson.com/blog/>.

AMORIM, E. de A. *Fluxo de Potência Ótimo em Sistemas Multimercados Através de um Algoritmo Evolutivo Multiobjetivo*. Tese (Doutorado) — Universidade Estadual Paulista, 2006.

BARBOSA, A. de A. *Um Simulador Simbólico de Circuitos Quânticos*. Dissertação (Mestrado) — Universidade Federal de Campina Grande, 2007.

BARROS, P. S. N. *Reconhecimento de padrões quânticos aplicados à sequências de DNA*. Dissertação (Mestrado) — Universidade Federal Rural de Pernambuco, 2011.

BERTHIAUME, A. mini course, *Quantum Computation*. 1996. (BRICS Notes Series).

BRUMATTO, H. J. Introdução à computação quântica. 2010. Disponível em: <http://www.ic.unicamp.br/~ducatte/mo401/1s2010/T2/096389-t2.pdf>.

CALAZAN, R. M.; NEDJAH, N.; MOURELLE, L. de M. A massively parallel reconfigurable co-processor for computationally demanding particle swarm optimization. *LASCAS (2012) - International Symposium of IEEE Circuits and Systems in Latin America*, 2012.

CAMPBELL, M. Quantum computer sold to high-profile client. *New Scientist Tech*, n. 2815, Junho 2011.

CARVALHO, L. M.; LAVOR, C. C.; PORTUGAL, R. Representações de um q-bit: propriedades e visualização. 2005.

CARVALHO, L. N. M.; LAVOR, C.; MOTTA, V. S. Caracterização matemática e visualização da esfera de bloch: Ferramentas para computação quântica. *Sociedade Brasileira de Matemática Aplicada e Computacional*, v. 8, n. 3, p. 351–360, 2007.

- D-WAVE. *The D-Wave One system*. 2012. Disponível em: <<http://www.dwavesys.com/en/products-services.html>>.
- DEB, K. *Multi-objective optimization using evolutionary algorithms*. [S.l.]: Wiley and Sons, 2001. 518 p. ISBN ISBN-13 978-0471873396.
- GUIZZO, E. D-wave does not quantum compute. *IEEE Spectrum*, january 2010. Disponível em: <spectrum.ieee.org/computing/hardware/loser-dwave-does-not-quantum-compute/0>.
- INTEL. *Intel® 22nm Technology*. 2011. Disponível em: <<http://www.intel.com.br/content/www/us/en/silicon-innovations/intel-22nm-technology.html>>.
- JOHNSON, M. W. et al. Quantum annealing with manufactured spins. *Nature*, *Nature*, n. 473, p. 194–198, 2011.
- JUGENE - Configuration. 2010. Disponível em: <<http://www.fz-juelich.de/ias/jsc/EN/Expertise/Supercomputers/JUGENE/Configuration/Configuration-node.html>>.
- KHALID, A. U.; ZILIC, Z.; RADECKA, K. Fpga emulation of quantum circuits. *ICCD 04: Proceedings of the IEEE International Conference on Computer Design*, p. p. 310–315, 2004.
- LEE, J.; HUANG, X.; ZHU, Q. sheng. Decomposing fredkin gate into simple reversible elements with memory. *International Journal of Digital Content Technology and its Applications*, v. 4, n. 5, 2010.
- LIMA, A. F. de; LULA JÚNIOR, B. Circuitos quânticos: uma introdução. *WECIQ*, 2006.
- LORENZEN, F. *Extensão dos Be-ables de Bell e Competição Atenuação×Amplificação. Dissertação (Mestrado) — Universidade de São Paulo, 2009.*
- MARON, A. et al. Introduzindo uma abordagem para simulação quântica com baixa complexidade espacial. *10a Conferência Brasileira de Dinâmica, Controle e Aplicações, Agosto 2011.*
- MARQUEZINO, F. d. L. *A Transformada de Fourier Quântica Aproximada e sua simulação. Dissertação (Mestrado) — LNCC - Laboratório Nacional de Computação Científica, 2006.*
- MATSON, J. Physicists entangle a record-breaking 14 quantum bits. *Scientific American*, 2011. Disponível em: <<http://blogs.scientificamerican.com/observations/2011/04/05/physicists-entangle-a-record-breaking-14-quantum-bits/>>.

- MICHIELSEN, K. Jülicher supercomputer simuliert Quantencomputer. Março 2010.
Disponível em: <<http://www.fz-juelich.de/SharedDocs/Pressemitteilungen/UK/DE/2010/index7d86htm.html>>.
- MONTEIRO, E. et al. Simulação quântica em vhdl: Um estudo de caso baseado no algoritmo de deutsch. v. 1, p. p. 13–27, 2009.
- MONTEIRO, E. R. *qExVHDL: Uma Modelagem de Circuitos Quânticos Utilizando VHDL. 2009.*
- MONTEIRO, E. R. et al. Simulação quântica em vhdl: Um estudo de caso baseado no algoritmo quântico de grover. *In: Workshop de Iniciação Científica, p. p. 1–10, 2009.*
- MONTEIROY, E. et al. Aplicação da biblioteca q-exvhdl na descrição do interferômetro de mach-zehnder. *In Jornadas Chilenas de Computación, p. p. 1–10, 2008.*
- MONZ, T. et al. 14-qubit entanglement: Creation and coherence. *Physical Review Letters, APS, v. 106, n. 106, 2011.*
- NIELSEN, M. A.; CHUANG, I. L. *Quantum Computation and Quantum Information. [S.l.: s.n.], 2000. 708 p. ISBN ISBN-13: 978-1107002173.*
- OLIVEIRA, I. S.; SARTHOU, R. S. Computação quântica e informação quântica. *V Escola do CBPF, 2004.*
- OLIVEIRA, N. A. *A Utilização do Algoritmo Quântico de Busca em Problemas da Teoria da Informação. Dissertação (Mestrado) — Universidade Federal de Campina Grande, 2007.*
- OMER, B. *A Procedural Formalism for Quantum Computing. Dissertação (Mestrado) — Technical University of Vienna, Department of Theoretical Physics, 1998.*
- OMER, B. *Quantum Programming in QCL. Dissertação (Mestrado) — Technical University of Vienna, Institute of Information Systems, 2000.*
- OMER, B. *Structured Quantum Programming. Tese (Doutorado) — Vienna University of Technology, Institute of Information Systems, 2009.*
- PORTUGAL, R.; COSME, C. M. M.; GONÇALVES, D. N. Algoritmos quânticos. 2006.
- PORTUGAL, R.; LAVOR, C. C.; CARVALHO, N. M. L. M. Uma introdução aos algoritmos quânticos. 2004.

- PORTUGAL, R.; LAVOR, C. C.; MACULAN, L. M. e Carvalho e N. Uma introdução à computação quântica. *Notas em Matemática Aplicada, SBMAC, São Carlos, v. 8, 2004.*
- RAEDT, K. et al. Massively parallel quantum computer simulator. *Computer Physics Communications*, p. p. 121–136, 2007.
- RIEFFEL, E.; POLAK, W. An introduction to quantum computing for non-physicists. *ACM Comput. Surveys, v. 32, p. 300–335, 2000.*
- RIGOLIN, G. G. Emaranhamento quântico. *Revista Phisicae 7, p. 1, 2008.*
- SCIENCEDAILY. Pushing the limits of computer chip miniaturization. *Science Daily, 2008.*
Disponível em: <<http://www.sciencedaily.com/releases/2008/01/080112083626.htm>>.
- SHENDE, V. V.; BULLOCK, S. S.; MARKOV, I. Synthesis of quantum logic circuits. *IEEE Transactions on Computer-Aided Design, p. p. 1000–1010, 2006.*
- SILVA, F. L. S. da. *Processamento de informação quântica usando um sistema de íons aprisionados e cavidades. Dissertação (Mestrado) — Universidade Estadual de Campinas, 2002.*
- THIBES, R. S. Álgebra linear e mecânica quântica. *V Bienal da SBM, 2010.*
- VIGNATTI, A. L.; NETO, F. S.; BITTENCOURT, L. F. *Uma Introdução à Computação Quântica. 2004.*
- VIZZOTTO, J. K.; COSTA, A. C. da R. Linguagens de programação quântica - um apanhado geral. *WECIQ, 2006.*
- WAN, Q.; BREWER, K. Ieee-754 floating-point conversion from floating-point to. 2003.
- WATANABE, M. S. M. *O Algoritmo Polinomial de Shor para Fatoração em um Computador Quântico. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2003.*
- WOOTERS, W. K.; ZUREK, W. H. A single quantum cannot be cloned. *Letters to Nature, n. 299, p. 802–803, 1982.*
- YANOFSKY, N. S. Proof, computation and agency. v. 352, p. 145–180, 2007.