UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS BLUMENAU
DEPARTAMENTO DE ENG. DE CONTROLE, AUTOMAÇÃO E COMPUTAÇÃO
CURSO DE ENG. DE CONTROLE, AUTOMAÇÃO E COMPUTAÇÃO

João Paulo do Nascimento

Roteador memresistivo para redes-em-chip de sistema bioinspirados

Blumenau
2023

João Paulo do Nascimento

Roteador memresistivo para redes-em-chip de sistema bioinspirados

Trabalho de Conclusão de Curso submetido ao curso de Eng. de Controle e Automação do Campus Blumenau da Universidade Federal de Santa Catarina como requisito parcial para a obtenção do título de Bacharel em Eng. de Controle e Automação.

Orientador(a): Profa. Dra. Janaína G. Guimarães.

Blumenau

2023

Nascimento, João Paulo do
Roteador memresistivo para redes-em-chip de sistema bioinspirados / João Paulo do Nascimento ; orientadora, Janaína Gonçalves Guimarães, 2024.
94 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Blumenau, Graduação em Engenharia de Controle e Automação, Blumenau, 2024.

Inclui referências.

1. Engenharia de Controle e Automação. 2. Memristor. 3. Roteador. 4. Memória SRAM. 5. Redes-em-chips. I. Guimarães, Janaína Gonçalves. II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Controle e Automação. III. Título.

João Paulo do Nascimento

Roteador memresistivo para redes-em-chip de sistema bioinspirados

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do título de engenheiro de controle e automação e aprovado em sua forma final pelo Curso de Engenheiro de Controle e Automação.

Blumenau, 24 de novembro de 2023.

Coordenação do Curso

Banca examinadora

Documento assinado digitalmente
Janaina Goncalves Guimaraes
Data: 08/12/2023 09:55:02-0300
CPF: ***.475.831-**
Verifique as assinaturas em https://v.ufsc.br

Prof.(a). Janaína G. Guimarães, Dr.(a).

Orientador(a)

Documento assinado digitalmente
Adao Boava
Data: 08/12/2023 10:18:02-0300
CPF: ***.640.969-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Adão Boava, Dr.

Universidade Federal de Santa Catarina

Documento assinado digitalmente
BEATRIZ OLIVEIRA CAMARA DA FE
Data: 08/12/2023 13:42:37-0300
Verifique em https://validar.iti.gov.br

Prof.(a) Beatriz Oliveira Câmara, Me.(a)

Universidade de Brasília

Blumenau, 2023.

AGRADECIMENTOS

Aos meus pais, João Batista do Nascimento e Elisângela Alves do Nascimento, cujo amor e dedicação moldaram quem sou hoje; ao meu irmão, João Matheus do Nascimento, pelo apoio incondicional e inspiração mútua ao longo dos anos; gostaria de expressar minha sincera gratidão. A estes três pilares devo grande parte do meu sucesso acadêmico, pois, sem seu apoio, nada disso seria possível.

À minha estimada orientadora, Janaína Guimarães, expresso minha profunda gratidão por sua inestimável paciência e generosidade ao me orientar. Pois, na medida que percorri os desafios deste trabalho, sua orientação e apoio desempenharam um papel crucial. Suas valiosas orientações e insights enriqueceram o desenvolvimento desta pesquisa, tornando-a mais significativa e enriquecedora. Sua dedicação e comprometimento são verdadeiramente inspiradores, e sou grato por ter tido a honra de tê-la como minha orientadora.

À minha amada namorada e futura esposa, Maria Gabriela, que se revelou um bálsamo para as turbulências que enfrentei. Tua presença trouxe calma às minhas crises de ansiedade e luz ao pior período da minha vida. Seu apoio constante e amor incondicional foram a força que me impulsionou a superar cada obstáculo, e sou imensamente grato por ter você ao meu lado, tornando cada desafio mais suportável e cada vitória mais doce.

Por fim, expresso minha gratidão a todos meus amigos e colegas, que se tornaram fonte de conhecimento, diversão e estresse ao longo do curso. Sem eles, a jornada certamente não teria sido tão enriquecedora e divertida.

“E, no final, meu sonho é igual
Família, churrasco no quintal
Ver meu time na final
Agradecendo a vida longe dos funeral
E fé pra isso” - BK', Amanhecer.

RESUMO

Este estudo introduz o projeto de um roteador digital genérico voltado para sistemas bioinspirados, baseado na tecnologia dos *memristors*. O objetivo principal é explorar as propriedades dos *memristors* e avaliar sua viabilidade em circuitos digitais, particularmente em roteadores e memórias. Os *memristors*, dispositivos teorizados por Leon Chua em 1971, possuem a capacidade única de ajustar sua resistência com base no fluxo de carga mais recente que atravessa seus terminais. Neste projeto, utilizou-se *memristors* para armazenar informações e realizar operações lógicas, aproveitando-os no desenvolvimento de uma biblioteca de circuitos memresistiva. Durante a pesquisa demonstrou-se que o dispositivo idealizado por Chua (1971) não responde bem a aplicações multiníveis mais complexas, no entanto, para circuitos mais ‘leves’, apresentaram resultados satisfatórios, mas ainda inferiores em termos de ocupação de área e dissipação de energia quando comparados a tecnologias recentes, como QCA e SET. Ao final, ressaltou-se que o uso de *memristors* na criação de memórias SRAM se mostrou eficaz e vantajoso em relação às abordagens que dependem exclusivamente de transistores. Portanto, embora haja bastante espaço para melhorias, os *memristors* já demonstram um potencial para aprimorar a eficiência dos circuitos de memória num futuro próximo.

Palavras-chave: *Memristor*, roteador, SRAM, transistor.

ABSTRACT

This study introduces the design of a generic digital router targeted for bioinspired systems, based on memristor technology. The main objective is to explore the properties of memristors and assess their feasibility in digital circuits, particularly in routers and memories. Memristors, devices theorized by Leon Chua in 1971, possess the unique capability to adjust their resistance based on the most recent flow of charge passing through their terminals. In this project, memristors were used to store information and perform logical operations, leveraging them in the development of a memristive circuit library. Throughout the research, it was demonstrated that the device envisioned by Chua (1971) does not respond well to heavier multilevel applications; however, for lighter circuits, they yielded satisfactory results, albeit still inferior in terms of footprint and energy dissipation compared to recent technologies like QCA and SET. In conclusion, the use of memristors in SRAM memory creation proved effective and advantageous in comparison to approaches relying solely on transistors. Therefore, while there is ample room for improvement, memristors already demonstrate potential for enhancing memory circuit efficiency in the near future.

Keywords: Memristor, router, SRAM, transistor.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS E SÍMBOLOS

CMOS	*Complementary Metal-Oxide-Semiconductor*;
MOS	*Metal-Oxide-Semiconductor*;
q	Carga elétrica;
Φ	Fluxo magnético;
M	Memresistência;
m	Metros;
V	Tensão;
i	Corrente;
TiO_2	Dióxido de titânio;
μ_v	Mobilidade média de íons;
D	Espessura da camada semicondutora;
W ou w	Largura;
L	Comprimento;
w_0 ou ω_0	Frequência de alimentação do memristor;
TE	Lado dopado do *memristor*;
BE	Lado não dopado do *memristor*;
XSV	Estado interno do *memristor*;
FET	*Fiel Effect Transistor*;
S, G e D	*Source*, *Gate* e *Drain*;
NoC	*Network-on-Chip*
SoC	*System-on-Chip*
SR	*Registrador de deslocamento*
FIFO	*First-in, First-out Register*
PIPO	*Parallel-in, Parallel-out Register*
PISO	*Serial-in, Serial-out Register*
SIPO	*Serial-in, Parallel-out Register*
RAM	*Random-Access Memory*
DRAM	*Dynamic Random-Access Memory*
SRAM	*Static Random-Access Memory*
r_n	*Request lines*;
g_n	*Grant lines*;
h_n	*Hold lines*;

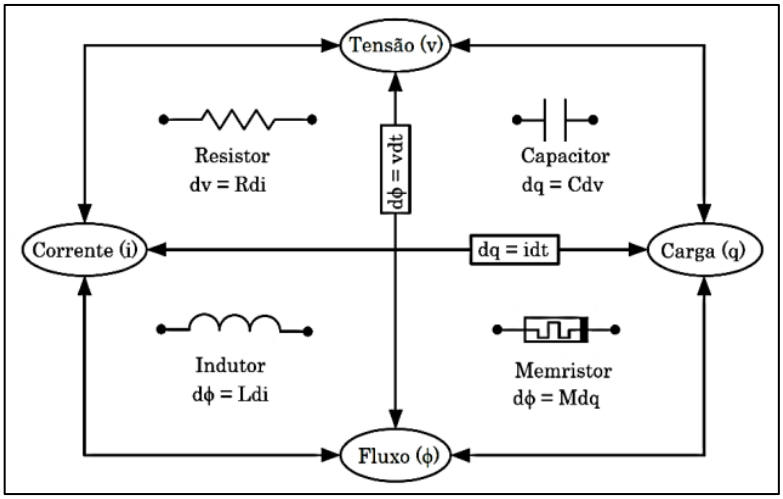
V_{IN}	Tensão de entrada;
V_{OUT}	Tensão de saída;
V_{DD}	Tensão de alimentação;
R_T	Resistência de condução;
MRL	Margem de ruído baixa;
MRH	Margem de ruído alta;
VIL	Tensão de entrada baixa;
VOL	Tensão de saída baixa;
VIH	Tensão de entrada alta;
VOH	Tensão de entrada baixa;
MSB	*Bit* mais significativo;
E	*Enable*;
CLK	*Clock*;
nW/S	Sinal de controle conversão paralelo-serial;
WL	Word-Line
BL	Bit-Line
OA	*Oblivious Arbiter*
RoR	*Round-Robin Arbiter*
QCA	*Quantum-dot cellular automata*;
SET	*Single-electron transistor*;
EB	*Ellastic Buffer*;
6T2M	*Six Transistors – Two Memristors*;
6T	*Six Transistors*;
SW	*Select Write*;
WEn	*Write Enable*;
REn	*Read Write*;
TCC	Trabalho de conclusão de curso;
SR	*Shift-Register*;
SRCLK	*Shift-Register Clock*;
SRwOR	*Shift-Register with Output Register*;
RCLK	*Register Clock*;
mem	*Memresistive*;
MIMO	Multiple-input, multiple-output;
MISO	Multiple-input, single-output;

CAPÍTULO 1 – INTRODUÇÃO:

Nos últimos anos, avanços notáveis têm moldado o campo da eletrônica e da computação de maneira surpreendente. Um desses avanços é a emergência dos *memristors*, nomeados pela contração das palavras "*memory*" e "*resistor*", dispositivos eletrônicos que abrem portas para uma nova era de armazenamento de dados, processamento de informações e inteligência artificial.

A concepção teórica do *memristor* remonta a 1971, quando Leon Chua fez uma observação crucial no âmbito da teoria dos circuitos. Ele discerniu que os três elementos fundamentais de dois terminais conhecidos até então (resistor, capacitor e indutor) eram definidos por relações entre duas das quatro grandezas elétricas fundamentais: corrente, tensão, carga elétrica e fluxo magnético (CHUA, 1971), tal qual evidenciado na figura 1.1.

Figura 1.1 – Relação entre as grandezas elétricas fundamentais.



Fonte: DIAS (2018).

Intrigantemente, de todas as combinações possíveis entre essas quatro variáveis, cinco delas levavam a relações já conhecidas. Inspirado por essa constatação, Chua aventurou-se a postular a necessidade de um quarto componente para preencher a lacuna na relação entre fluxo magnético e carga elétrica. Surgia assim o "*memristor*", cujo comportamento distinto não poderia ser reproduzido por circuitos baseados apenas nos outros três elementos fundamentais da eletrônica.

Esse elemento singular, cujo comportamento se assemelha ao de um resistor não linear, possui a capacidade de manter dados de forma estável por um longo

período mesmo na ausência de alimentação de energia. Ele se consolidou como o quarto elemento essencial, representando um avanço significativo na evolução da eletrônica. Ademais, essa característica do dispositivo de reter informações se mostra atrativa para aplicações de armazenamento não volátil, onde a estabilidade e a retenção de dados são essenciais.

No âmbito da pesquisa em circuitos digitais, o uso de *memristors* tem sido amplamente explorado. Uma série de circuitos digitais fundamentais, como decodificadores, codificadores, travas, multiplexadores e flip-flops, tem sido desenvolvida com base na tecnologia do *memristor*. Esses avanços evidenciam a viabilidade prática da incorporação do *memristor* no campo dos circuitos digitais, e até mesmo a perspectiva de substituição dos circuitos digitais tradicionais baseados em tecnologia CMOS (WASER, 2009).

Embora o desenvolvimento do *memristor* ainda esteja em estágios iniciais, a concepção de circuitos que combinam a tecnologia do *memristor* com a tecnologia CMOS tem sido considerada uma abordagem segura e promissora. Essa integração possibilita o aproveitamento das vantagens individuais de ambos os componentes, resultando em sistemas híbridos que podem explorar eficientemente as propriedades únicas do *memristor* enquanto mantêm uma base estável e comprovada de CMOS (YANG, 2013).

Sob esse contexto, o presente trabalho pretende recriar, utilizando *memristors*, o comportamento de um roteador para redes-em-chip aplicado em sistemas bioinspirados. Para tal, idealiza-se, após uma ampla revisão teórica, desenvolver uma biblioteca de circuitos memresistivos, explorando o potencial sinérgico entre o memristor e os conceitos provenientes de portas lógicas para a construção de um circuito roteador abrangente.

1.1 MOTIVAÇÃO

A integração sinérgica de *memristors* em conjunção com transistores MOS no delineamento de circuitos lógicos denota a perspectiva de uma série de vantagens latentes. *Memristors*, como dispositivos de memória, manifestam um perfil de resistência variável intrínseca, cujo comportamento é modulado em resposta à corrente elétrica que os atravessa. Esses componentes têm sido alvo de indagações devido à sua capacidade de efetuar o armazenamento de informações em um estado

não volátil, bem como a habilidade de desempenhar operações computacionais transcendentais em comparação aos transistores convencionais (DIAS, 2018).

Desta forma, ao serem conectados aos transistores CMOS, os quais constituem os pilares fundamentais da arquitetura eletrônica contemporânea, é possível vislumbrar contribuições substanciais no contexto da exigência de circuitos lógicos. Algumas das prerrogativas engendradas abarcam:

- **Menor Consumo de Energia:** Os *memristors* têm a capacidade de reter seu estado resistivo mesmo quando a alimentação de energia é desligada. Isso permite a construção de circuitos não voláteis que consomem menos energia, uma vez que não é necessário constantemente manter uma fonte de alimentação para reter informações (VENTRA, PERSHIN, 2013).
- **Menor Área de Chip:** A implementação de *memristors* junto com transistores MOS pode levar a uma redução no tamanho físico dos circuitos. *Memristors* têm a vantagem de ocupar menos espaço em comparação com elementos de memória convencionais, como as células de memória baseadas em transistores (DIAS, 2018).
- **Operações de Computação Avançadas:** Os *memristors* exibem características de sinapse, o que significa que podem ser usados para implementar funções de aprendizado e plasticidade sináptica, semelhantes ao funcionamento das sinapses biológicas no cérebro. Isso torna os *memristors* potencialmente úteis para aplicações em redes neurais artificiais e computação neuromórfica (MENDES, 2016).
- **Maior Densidade de Integração:** A capacidade de fabricar circuitos que combinam *memristors* e transistores MOS pode levar a uma maior densidade de integração em um chip. Isso permite a construção de sistemas mais complexos em um espaço menor (PERSHIN, VENTRA, 2014b).
- **Maior Velocidade e Eficiência:** *Memristors* podem ser projetados para operar em frequências mais altas do que os transistores tradicionais. Isso pode resultar em operações de computação mais rápidas e eficientes em termos de energia (PISSARDINI, 2017).
- **Flexibilidade de projeto:** A combinação de *memristors* e transistores MOS oferece flexibilidade de projeto, permitindo a criação de circuitos híbridos que

aproveitam as vantagens de ambos os tipos de dispositivos (PISSARDINI, 2017).

Com base nas considerações expostas, inicialmente foi proposto implementar, através de *memristors* e inversores CMOS, um circuito lógico (roteador) para sistemas bioinspirados baseados em redes-em-chip. Esse sistema usaria como blocos básicos portas lógicas NAND, NOR e NOT. Especificamente, a decisão de incorporar tais circuitos digitais é fundamentada na inerente versatilidade que essas lógicas oferecem. Como dispositivos lógicos universais, as portas NOT, NAND e NOR viabilizam a realização de uma vasta gama de operações lógicas, o que se traduz em um benefício substantivo para a concepção de circuitos complexos.

1.2 OBJETIVOS

O objetivo geral desta pesquisa é implementar um circuito digital de um roteador para sistemas bionspirados (redes-em-chip) baseado em *memristors* tendo como base a tecnologia memresistiva. Adicionalmente, busca-se desenvolver uma biblioteca abrangente de circuitos digitais que utilizem *memresistors* como elemento base. Essa biblioteca desempenhará um papel crucial no planejamento e desenvolvimento do roteador.

1.2.1 Objetivos específicos

- Realizar uma revisão bibliográfica direcionada sobre os princípios fundamentais dos *memristors*, incluindo suas propriedades, características de funcionamento e aplicações em circuitos eletrônicos.
- Investigar e simular o desempenho de portas lógicas que se baseiam na tecnologia do *memristor*, analisando aspectos como velocidade de operação, consumo de energia, estabilidade e capacidade de processamento.
- Projetar, simular e avaliar experimentalmente um conjunto diversificado de circuitos digitais utilizando *memristors* como componentes-chave, demonstrando sua viabilidade prática e seu potencial para superar as limitações das tecnologias convencionais.

- Desenvolver uma biblioteca de circuitos digitais abrangente, que abarque desde os circuitos mais simples até estruturas mais complexas, utilizando o *memristor* como base. Essa biblioteca deve ser estruturada de forma modular e de fácil reutilização.
- Explorar as oportunidades oferecidas pela tecnologia memresistiva para a criação de um roteador digital aplicado a redes-em-chip.
- Analisar os benefícios do uso de *memristors* nos circuitos digitais projetados, considerando aspectos como eficiência energética, adaptação e perspectivas futuras.

1.3 ORGANIZAÇÃO DO TRABALHO

A estrutura deste trabalho segue uma organização por capítulos, conforme delineado a seguir. No primeiro capítulo, intitulado "Introdução", é apresentada uma visão abrangente dos modelos memresistivos, explorando conceitos fundamentais que estabelecem as bases teóricas. O segundo capítulo, intitulado "Dispositivos Eletrônicos", dedica-se a uma análise introdutória dos transistores MOS e inversores CMOS, destacando os princípios físicos e elétricos relevantes. Além disso, aprofunda-se no tópico de simulação e replicação das propriedades reais do *memristor*. No terceiro capítulo, intitulado "Redes-em-Chip e Circuitos Digitais", são abordados tópicos introdutórios às redes-em-chip, incluindo conceitos da arquitetura digital aplicada ao roteador idealizado. O quarto capítulo, "Projeto e Análise de Portas Lógicas Básicas Baseadas em *Memristors*", concentra-se na implementação e avaliação de um modelo SPICE das portas lógicas elementares. O quinto capítulo, intitulado "Biblioteca Memresistiva", aborda a criação de uma biblioteca de circuitos lógicos memresistivos, reinterpretando diversos circuitos comuns da indústria eletrônica. O sexto capítulo, "O Roteador", detalha as etapas metodológicas para a construção e análise do roteador. Além disso, realiza uma análise *top-down* do dispositivo, descrevendo seus circuitos e apresentando resultados de simulações para validação e comparação com tecnologias existentes. O último capítulo, "Conclusão", encerra o trabalho consolidando as informações discutidas ao longo do documento.

CAPÍTULO 2 – DISPOSITIVOS ELETRÔNICOS:

2.1 MESRISTORES

2.1.1 **Aspectos gerais:**

Considere a imagem de um tubo por onde água está em circulação. Quanto mais estreito for o tubo, maior será a obstrução ao fluxo da água. De maneira análoga, pode-se começar por observar que as resistências convencionais presentes nos circuitos elétricos atuais possuem um "diâmetro" fixo. No entanto, o *memristor* contrasta com essa característica ao demonstrar a capacidade notável de ajustar o seu "diâmetro", isto é, sua resistência, de acordo com a quantidade e direção da corrente elétrica que flui através dele (MENDES, 2016).

O *memristor* tem a habilidade de "memorizar" seu próprio diâmetro, ou seja, sua resistência, com base na última vez que a carga passou pelo circuito. Quando a circulação cessa, da mesma forma que a interrupção da corrente elétrica, o diâmetro do "tubo" (*memristor*) mantém-se na configuração mais recente até que a circulação seja reestabelecida (MENDES, 2016).

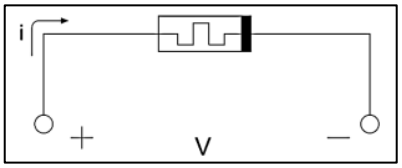
Apresentado à comunidade científica há mais de quatro décadas através do artigo "*The missing circuit element*" (CHUA, 1971), este elemento eletrônico passivo de dois terminais possui uma memresistência M. Essa memresistência cria uma conexão entre a carga elétrica (q) e o fluxo magnético (Φ), representada da seguinte forma:

$$\boldsymbol{d\emptyset = M \cdot dq} \tag{2.1}$$

Sendo assim, como tanto o fluxo magnético quanto a carga elétrica podem ser expressos como integrais da tensão e corrente ao longo do tempo, respectivamente, a equação (2.1) pode ser modificada de forma a comprovar que o *memristor* introduz uma resistência que é influenciada pelo histórico da corrente que o percorre, assim como pela variação de tensão entre seus terminais (ASCOLI et al., 2013).

$$\boldsymbol{v(t) = M \cdot i(t)} \tag{2.2}$$

Figura 2.1 – Símbolo elétrico do *memristor*.



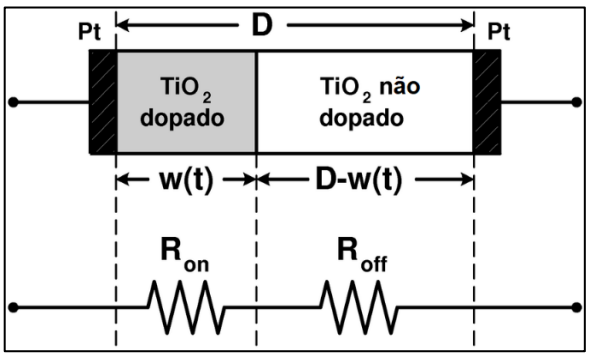
Fonte: DIAS (2018).

Ademais, a depender do fluxo da corrente i(t), a memresistência pode ser elevada ou decrescida. Contudo, sempre reterá, após a interrupção de estímulo elétrico, o valor final da resistência assumido pelo componente (GARC et al., 2014).

2.1.2 *Memristor* de dióxido de titânio:

A proposta inicial de aplicação prática e materialização de um *memristor* foi apresentada em 2008 por uma equipe de pesquisa liderada pelo cientista Stanley Williams nos laboratórios da Hewlett Packard, conforme indica YAKOPIC et al (2011). Estruturado por uma fina camada de TiO_2, com espessura *D*, posicionada entre dois contatos de platina, o *memristor* de TiO_2 (figura 2.2) tem sua camada semicondutora dividida em duas regiões: uma delas, com largura *w*, apresenta uma elevada concentração de dopantes (TiO_2 puro), resultando em um valor baixo de resistência (R_{ON}); e outra região que exterioriza uma baixa concentração de dopantes, com menor quantidade de oxigênio (TiO_{2-X}), isto é, um valor muito maior de resistência (R_{OFF}) (DIAS, 2018).

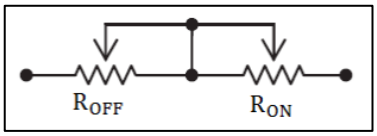
Figura 2.2 – Estrutura física do *memristor* de TiO_2.



Fonte: Scientific Figure on ResearchGate.

O circuito equivalente do *memristor* também pode ser ilustrado conforme a figura 2.3, onde as resistências da região dopada e da região não dopada são interligadas em série (DIAS, 2018).

Figura 2.3 – Circuito equivalente com resistores variáveis em série.



Fonte: DIAS (2018).

Nesse sistema, ao ser aplicada uma tensão externa *v(t)* entre os terminais do dispositivo, responsável por polarizar o *memristor*, as lacunas de oxigênio no material respondem ao campo elétrico aplicado, resultando no deslocamento da fronteira entre as camadas de TiO_2 e TiO_{2-X}. Dessa forma, ocorrerá o deslocamento da fronteira entre as duas regiões, provocando o movimento dos dopantes carregados (BLANC; STAEBLER, 1971). Conforme exemplifica STRUKOV et al. (2008) (apud DIAS, 2018), no cenário mais simples de condução eletrônica ôhmica e deriva iônica linear sob um campo uniforme, com uma mobilidade média de íons μ_v, a formulação matemática do elemento quando submetido a uma tensão *v(t)* é expressa através das seguintes equações:

$$\boldsymbol{v(t) = \left(R_{ON} \frac{w(t)}{D} + R_{OFF} \left(1 - \frac{w(t)}{D} \right) \right) i(t)} \tag{2.3}$$

Onde:

$$\boldsymbol{\frac{dw(t)}{dt}} = \mu_V \boldsymbol{\frac{R_{ON}}{D} i(t) \quad \rightarrow \quad w(t) = \mu_V \frac{R_{ON}}{D} q(t)} \tag{2.4}$$

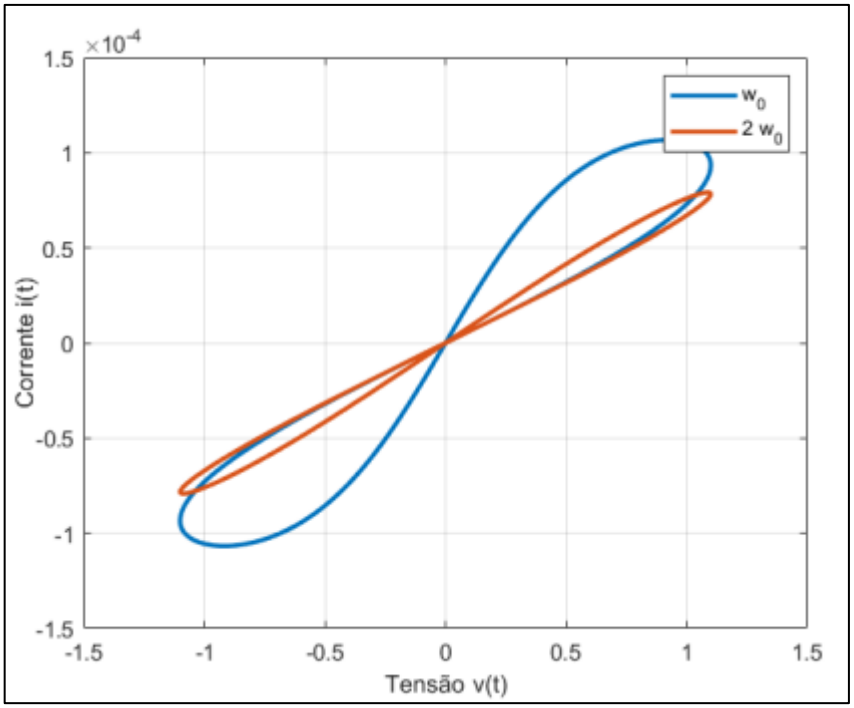
Sendo a variável de estado *w* diretamente relacionada à carga elétrica que passa pelo *memristor*, tem-se que seu valor é situado próximo às extremidades do intervalo [0, D]. Tal situação é conhecida como "*switching*", que ocorre devido a variações significativas de tensão entre os terminais do componente ou exposição prolongada à polarização (DIAS, 2018).

Ao manipular (2.3) e (2.4), é possível obter a memresistência desse sistema, considerado $R_{ON} \ll R_{OFF}$:

$$\boldsymbol{M(q) = R_{OFF} \left(1 - \mu_V \frac{R_{ON}}{D^2} q(t) \right)} \tag{2.5}$$

Na qual observa-se que *q(t)* cresce à medida que a mobilidade dos dopantes (μ_v) aumenta e a espessura da camada semicondutora D diminui. Sendo que, para qualquer tensão alternada simétrica de polarização, a curva I x V do *memristor* apresenta um comportamento de duplo ciclo de histerese e, na medida que a frequência (w_0) aumenta, elevando a taxa de comutação do *mesmristor*, o gráfico se aproxima de uma linha reta, como ilustrado na Figura 2.4.

Figura 2.4 – Curva de histerese do *memristor* ideal.



Fonte: Berrios, Couto, Cury e Rezende (2019).

2.1.3 **Modelo de simulação:**

Como demonstrado por DIAS (2018), além do *memristor* de TiO2, abordado no tópico anterior, diversos trabalhos apresentando modelos construtivos de *memristors* foram publicados desde a sua descoberta em 1971, tais como o modelo Spintrônico de IKEDA et al. (2010), o ferroelétrico apresentado por CHANTHBOUALA et al. (2012), o orgânico dos autores EROKHIN e FONTANA (2008), etc. No entanto, a disponibilidade limitada de amostras reais de *memristors* tem dificultado a capacidade da maioria dos pesquisadores de conduzir experimentos práticos com esses dispositivos. Diante dessa situação, a adoção de modelos elétricos computacionais surge como uma alternativa de destaque para viabilizar a análise de comportamento e a elaboração de aplicações relacionadas a esses componentes por meio de simulações.

Simultaneamente, há diversas abordagens de modelos SPICE, entre as quais destacam-se as publicações de BIOLEK et al. (2009), PERSHIN e DI VENTRA (2012), Simmons Tunnel Barrier de PICKETT et al. (2009). Cada uma delas possui singularidades de aplicação, com vantagens e desvantagens específicas.

Considerando isto, neste trabalho, escolheu-se adotar uma adaptação do modelo de Biolek através da função janela proposta por JOGLEKAR e WOLF (2009).

2.1.3.1 *Modelo SPICE de BIOLEK adaptado por JOGLEKAR E WOLF (2009):*

Conforme apontado por DIAS (2018), certos modelos virtuais existentes, como o desenvolvido por BIOLEK et al. (2008), também baseado no *memristor* de TiO_2, apresentam desafios de convergência. Isso é particularmente evidente nos limites de R_{ON} e R_{OFF}. Ao comutar o *memristor* para esses estados terminais, observa-se que nenhum estímulo externo seria capaz de modificar essa condição. Consequentemente, o *memristor* ficaria permanentemente preso a um desses estados, resultando em funções matemáticas complexas e implementações computacionais ineficazes.

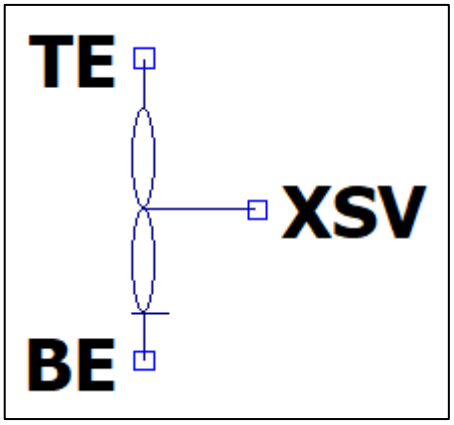
A janela de Joglekar é uma função matemática proposta por Joglekar e Wolf em 2009 para contornar este comportamento do modelo de Biolek. Ela descreve a relação entre a carga elétrica que passa pelo *memristor* e a posição da fronteira entre as camadas dopadas do material.

$$\boldsymbol{q(x) = 1 - (2x - 1)^{2p}} \qquad (2.6)$$

Onde *p* é um inteiro positivo e *x* é a posição da fronteira entre as camadas dopadas do material. Essa função garante que a velocidade da coordenada *x* seja zero ao se aproximar de cada limite do intervalo no qual está compreendido. Além disso, as diferenças entre os modelos com deriva linear e não linear desaparecem à medida que *p* aumenta. A função janela é importante porque permite que o *memristor* "lembre" a posição da fronteira entre as camadas, o que é fundamental para o seu comportamento como um elemento de memória (DIAS, 2018).

O modelo de Biolek corrigido com a janela de Joglekar é implementado em um simulador de circuitos elétricos, como o SPICE, e permite que sejam realizadas análises comportamentais do *memristor* em diferentes condições de operação. Esse modelo é importante porque permite que o *memristor* seja utilizado como um elemento de memória em circuitos eletrônicos, com aplicações em diversas áreas, como computação, processamento de sinais e sistemas de controle.

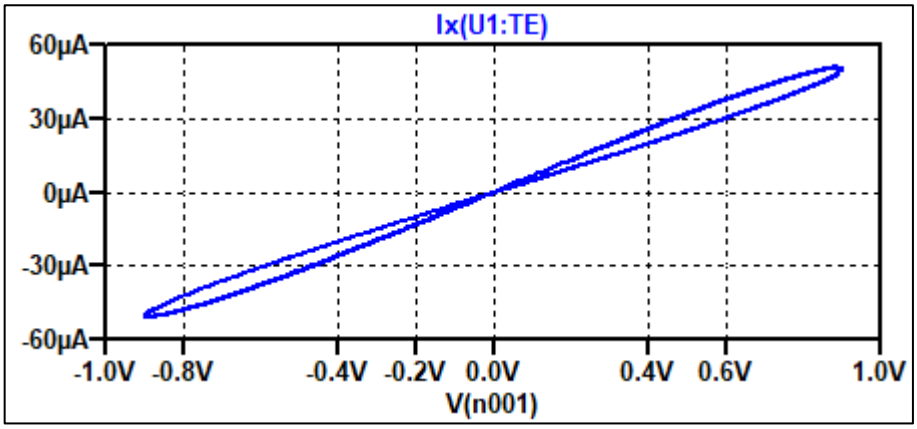
Figura 2.5 – Representação do *memristor* em LTSpice.



Fonte: Autor.

Na figura 2.5, TE representa o lado dopado do *mesmristor*, BE o lado não dopado e o terceiro terminal do dispositivo, denominado '*xsv*', utilizado para representar o estado interno do *memristor*. Não há necessidade de conexão a este terminal e o mesmo não será utilizado nesta pesquisa. O comportamento do *memristor* de Joglekar, nome dado a adaptação do modelo de Biolek realizado pela função janela de Joglekar e Wolf, quando submetido à uma tensão senoidal, é demonstrado na figura 2.6, na qual é possível observar a variação da corrente de saída de acordo com a tensão de entrada (ou seja, resistência visto *R=V/i*) numa faixa extremamente linear.

Figura 2.6 – Comportamento do *memristor* de Joglekar.



Fonte: Autor.

2.2 TRANSITORES MOS

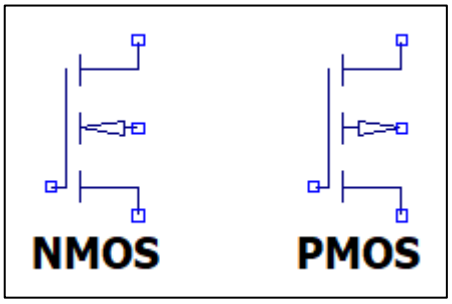
Os Transistores MOS (Metal-Oxide-Semiconductor) são blocos essenciais de semicondutores que desempenham um papel vital na eletrônica moderna, sendo amplamente empregados em diversas aplicações, como circuitos integrados, eletrônica digital, amplificadores e fontes de corrente. Como uma variante dos Transistores de Efeito de Campo (FET), os MOS são notáveis por sua eficiência energética, tamanho compacto e facilidade de produção (RASHID, 2014).

A estrutura básica de um Transistor MOS consiste em três partes: dreno (D), fonte (S) e *gate* (G), todas construídas sobre um substrato de silício. Um isolante de óxido de metal, frequentemente óxido de silício, separa o *gate* do canal semicondutor. Os MOS se dividem em duas principais categorias: NMOS (canal n) e PMOS (canal p) (RASHID, 2014), como evidenciado na figura 2.7.

Os Transistores MOS operam ao controlar a corrente entre dreno e fonte por meio do campo elétrico gerado pela tensão no *gate*, permitindo um controle preciso da corrente no circuito. Em um NMOS, uma tensão positiva no *gate* diminui o canal condutor entre fonte e dreno, enquanto em um PMOS, uma tensão negativa tem o mesmo efeito. O MOS pode operar em modos distintos: corte, saturação e triodo (região linear). Em corte, não há corrente entre dreno e fonte, enquanto em saturação, a corrente flui controlada pela tensão no *gate*. O MOSFET consome pouca energia quando está estável, sendo ótimo para baixo consumo, como em circuitos integrados (RASHID, 2014).

A combinação de NMOS e PMOS forma circuitos CMOS, essenciais em circuitos digitais. Fabricados por litografia, dopagem seletiva e deposição de materiais, eles tiveram um papel crucial na evolução da eletrônica (RASHID, 2014).

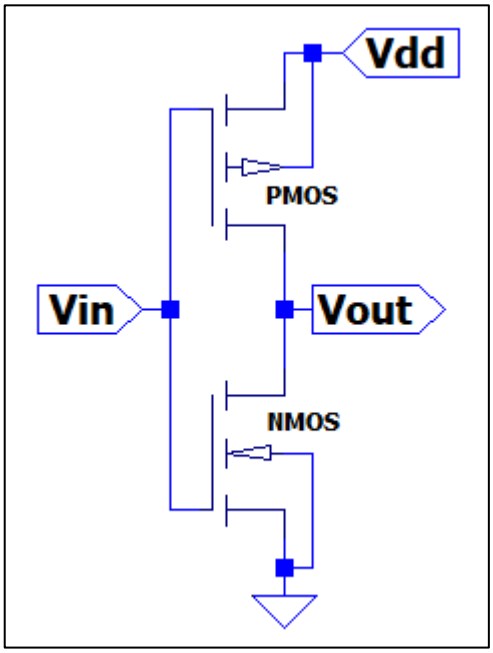
Figura 2.7 – Transistores NMOS e PMOS.



Fonte: Autor.

2.3 INVERSOR CMOS

Figura 2.8 – Inversor CMOS.



Fonte: Autor.

O inversor CMOS (Complementary Metal-Oxide-Semiconductor), mostrado na figura 2.8, é um circuito lógico básico que inverte um sinal de entrada, produzindo sua lógica oposta na saída. Consistindo em um par de transistores MOSFET, um NMOS e um PMOS, o inversor CMOS opera de forma complementar, quando um transistor está ligado, o outro está desligado. Essa abordagem otimiza o consumo de energia, especialmente durante as mudanças de estado lógico. A operação básica do inversor CMOS ocorre da seguinte forma (RASHID, 2014):

1. **Entrada (V_{IN}):** Uma entrada logicamente baixa (zero) ativa o transistor PMOS e desativa o NMOS, resultando em uma saída logicamente alta (V_{DD} - a fonte de alimentação).
2. **Transição:** À medida que a entrada aumenta, o PMOS desliga e o NMOS conduz, diminuindo a saída.
3. **Saída (V_{OUT}):** Quando a entrada é logicamente alta (V_{DD}), o NMOS conduz totalmente, e o PMOS fica desligado, levando a uma saída logicamente baixa.
4. **Transição inversa:** Quando a entrada diminui, o NMOS desliga e o PMOS conduz, restaurando a saída a um estado logicamente alto.

Essa alternância controlada dos transistores NMOS e PMOS garante que a saída do inversor seja sempre o inverso lógico da entrada. Isso torna o inversor CMOS essencial para a construção de circuitos digitais complexos, como portas lógicas e circuitos integrados. Devido à sua natureza complementar, os inversores CMOS são amplamente usados para economia de energia e são uma escolha popular na indústria de semicondutores (VALAVALA et al., 2018).

2.4 TRANSISTOR 45nm

Um transistor MOS de 45nm é um componente eletrônico fabricado por meio de uma tecnologia de processo com dimensões críticas na ordem de 45 nanômetros. Isso significa que o tamanho do canal e a largura do *gate* desse transistor estão próximos a essa escala nanométrica. Esses transistores são frequentemente encontrados em circuitos integrados, como microprocessadores e chips de memória.

É importante ressaltar que a tecnologia de 45 nm é considerada uma geração mais antiga, tendo sido introduzida pela primeira vez em 2007. Em comparação com os processos de fabricação mais modernos, que utilizam tamanhos menores, como 5nm ou 3nm, os transistores menores permitem uma densidade mais alta de componentes em um chip. Isso geralmente resulta em um melhor desempenho e maior eficiência energética (TSMC, 2022).

No entanto, um ponto importante a destacar é que tecnologias mais recentes, como as inferiores a 16nm, podem apresentar limitações, como a falta de suporte em bibliotecas virtuais para software de simulação, como é o caso do LTSpice.

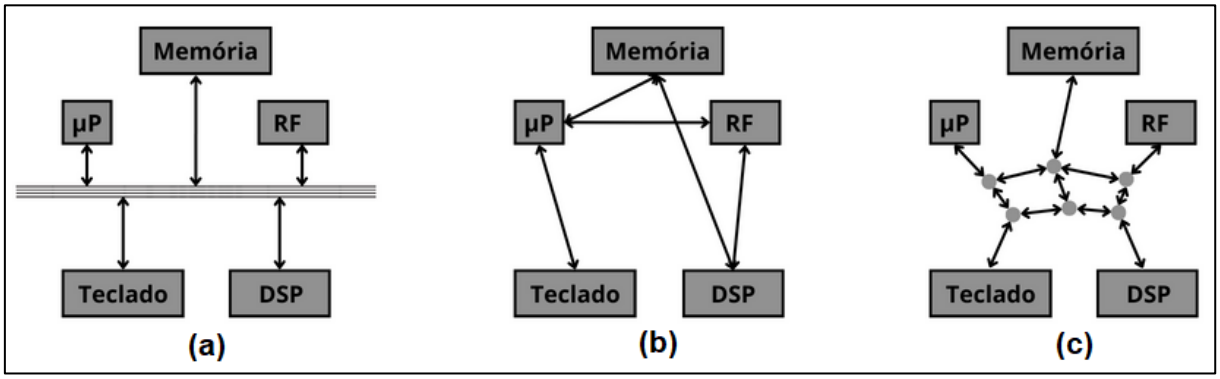
CAPÍTULO 3 – REDES EM CHIP E CIRCUITOS DIGITAIS

3.1 REDES-EM-CHIPS

Nos dispositivos eletrônicos contemporâneos, como smartphones e tablets, encontra-se um extenso conjunto de elementos de processamento compactamente integrados em um único chip de silício, denominado *System on Chip* (SoC) (DIMITRAKOPOULOS, 2016). Dentro de um espaço extremamente restrito, estabelecer conexões físicas e lógicas eficazes entre todos esses componentes representa um desafio considerável. Abordagens convencionais de conexão, como trilhas dedicadas (ponto-a-ponto), que ligam somente dois elementos, tornam-se impraticáveis à medida que o número de componentes aumenta de forma exponencial.

Adicionalmente, de acordo com Jantsch et al. (2003), barramentos compartilhados demonstram restrições em termos de largura de banda, elevado consumo energético, produção de ruído, e carecem de escalabilidade para atender às demandas dos sistemas contemporâneos. Uma resposta a esse desafio é a adoção de redes-em-chip (NoC – *Network-on-Chips*), inspiradas nos princípios bem estabelecidos das redes de computadores convencionais, aplicando-os ao nível do chip de silício (SWAPNA et al., 2012). O desenvolvimento de uma NoC representa um desafio multidimensional que engloba aspectos tanto de hardware quanto de software, abrangendo a seleção de topologias, o design de interfaces de rede, a implementação de roteadores e a definição de algoritmos de roteamento (CÂMARA, 2017). Na figura 3.1 é ilustrado os três tipos de conexão evidenciados.

Figura 3.1 - Conexões barramento (a), ponto-a-ponto (b) e NoC (c).

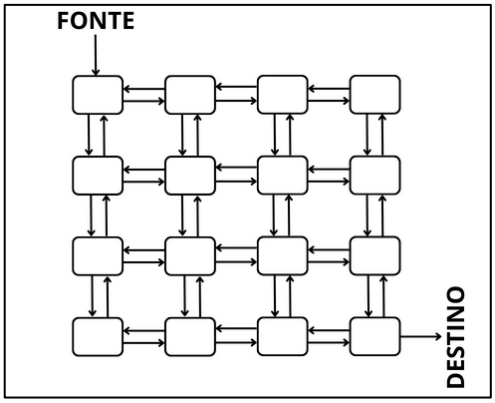


Fonte: SWAPNA et al (2012).

3.2 TOPOLOGIAS DE REDE

As topologias de rede definem como os elementos se conectam através de enlaces, determinando a configuração física da rede. Existem várias topologias para Redes-em-Chip (NoCs), incluindo *SPIN*, *Mesh*, *Torus*, *Ring* e *Butterfly*. A topologia *Mesh* é particularmente relevante, pois cada IP está diretamente conectado a um roteador (vide figura 3.2), proporcionando eficiência de layout, boas propriedades elétricas e simplicidade no endereçamento de recursos on-chip (HOLSMARK (2008) apud CÂMARA, 2017).

Figura 3.2 – Esquemática arquitetura MESH.



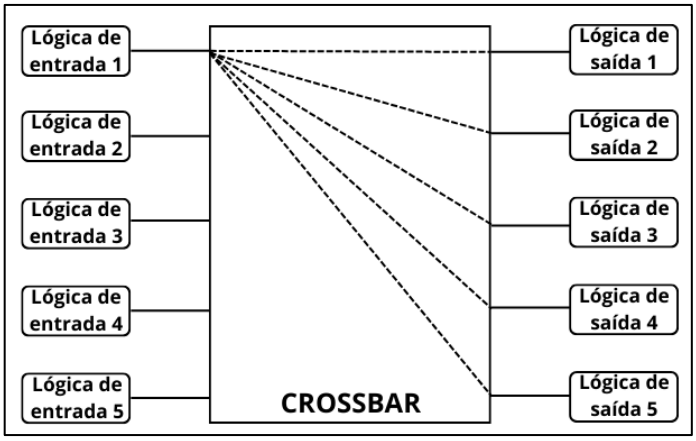
Fonte: PUJARI et al (2017).

Portanto, este trabalho adota a topologia *Mesh* como base para o desenvolvimento do roteador memresistivo, embora seja importante notar que essa abordagem é modular e pode ser adaptada para outras topologias.

3.3 ROTEADOR

Os roteadores desempenham um papel central nas Redes-em-Chip, sendo responsáveis por transmitir dados de uma fonte para um destinatário, conectando várias entradas e saídas para viabilizar diferentes topologias de rede (DIMITRAKOPOULOS, 2016). Ou seja, possuem relação direta na velocidade, taxa de transferência, área ocupada e consumo de energia do sistema (CÂMARA, 2017). Em termos gerais, um roteador consiste em três componentes principais (SERPANOS, 2011), conforme mostra a figura 3.3:

Figura 3.3 – Modelo de roteador genérico.



Fonte: CÂMARA (2017).

1. **Lógica de entrada:** Esta etapa é encarregada de extrair os endereços dos dados de entrada. Inclui, frequentemente, registradores para a paralelização dos bits e a extração do endereçamento.
2. **Estrutura de interconexão (*crossbar*):** Essencial para conectar corretamente as entradas aos destinos de saída, esta fase, tipicamente, engloba buffers para armazenamento temporário de dados e faz uso de registradores de deslocamento (PISO) para a serialização dos bits de informação provenientes de entradas paralelas. Além disso, são utilizados demultiplexadores para direcionar corretamente a entrada para a saída apropriada.
3. **Lógica de saída:** Fundamental para a distribuição eficiente de dados, a lógica de saída, com seu árbitro especializado, gerencia as solicitações das entradas conectadas. Isso assegura uma alocação ordenada e sem conflitos dos canais de saída, otimizando o desempenho do sistema.

3.4 REGISTRADOR DE DESLOCAMENTO DE 8 BITS COM REGISTRADOR DE SAÍDA

Composto de dois ou mais flip-flops D que compartilham o mesmo sinal de *clock*, o registrador tem a finalidade de armazenar um único bit (WAKERLY, 2000). Por outro lado, o registrador de deslocamento (SR) é um registrador de n-bits com capacidade de deslocar a posição do dado armazenado em um bit a cada pulso de *clock* (CÂMARA, 2017). Essa operação de deslocamento pode ser realizada de forma unidirecional, ou seja, somente para a direita ou para a esquerda, ou de forma

bidirecional, onde uma entrada de controle é utilizada para determinar a direção do deslocamento.

Entre os tipos mais utilizados de registradores de deslocamento, destacam-se os registradores FIFO (First-In-First-Out). Eles processam dados na ordem em que foram adicionados à fila, priorizando os dados mais antigos. Esses registradores são frequentemente empregados na transferência de dados entre diferentes domínios de clock, garantindo uma ordem de processamento consistente.

Contrastando com o FIFO, tem-se os registradores PISO (Paralelo em Série) e SIPO (Série em Paralelo). O PISO aceita dados em paralelo e os desloca sequencialmente para fora em uma única saída, de maneira serial. O SIPO opera de maneira inversa, recebendo dados em série e apresentando-os em várias saídas em paralelo.

Além disso, o registrador SISO (Série em Série) opera sequencialmente, recebendo e transmitindo dados bit a bit em série. Em um sistema digital, a interconexão estratégica de registradores FIFO, PISO, SIPO e SISO proporciona uma flexibilidade abrangente para inúmeras operações, desde a organização ordenada de dados até a conversão eficiente entre formatos paralelos e seriais, adaptando-se às necessidades específicas de cada aplicação.

Em cenários particulares, os registradores de deslocamento se unem aos registradores de armazenamento. Esses últimos possuem um sinal de clock independente do registrador de deslocamento principal, assegurando a persistência dos dados. Cada bit no registrador de deslocamento requer um registrador de armazenamento, comumente assumindo a forma de um flip-flop D. Essa configuração, conhecida como registrador de deslocamento com registrador de saída, é então replicada na forma SIPO de 8 bits para a construção do roteador idealizado neste trabalho. Dessa maneira, os dados são inseridos de forma serial, um bit de cada vez, mas podem ser lidos de forma paralela, permitindo o acesso simultâneo aos 8 bits nas saídas. Essa abordagem visa otimizar tanto a manipulação sequencial quanto a recuperação eficiente dos dados armazenados.

3.5 MEMÓRIA SRAM

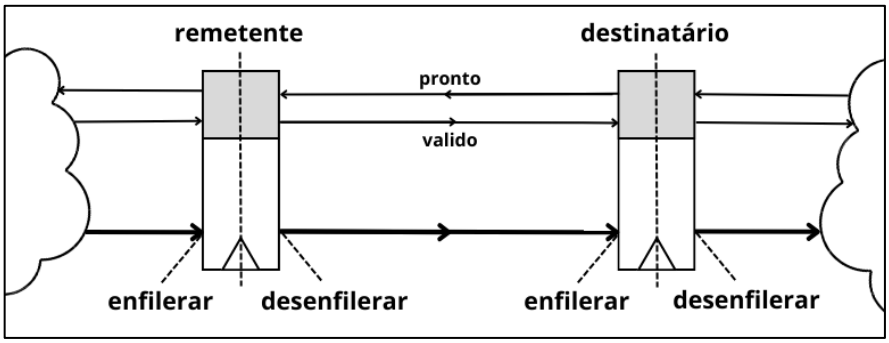
Memórias RAM são dispositivos de armazenamento que permitem leitura e escrita rápidas e aleatórias, independentemente da localização dos dados. Existem

dois tipos principais: *DynamicRAM* (DRAM), que requer leitura e reescrita periódica devido à perda gradual de carga em capacitores, e *StaticRAM* (SRAM), que mantém dados enquanto alimentada com energia (REDMOND, 2023). A SRAM é mais rápida e eficiente em energia, mas geralmente mais cara e tem menor capacidade. Ela suporta apenas leitura e escrita, com cada célula sendo um latch ou flip-flop, acrescidos de uma lógica para possibilitar as operações de leitura e escrita quando tal célula for selecionada (CÂMARA, 2017). A SRAM pode ser single-port (uma operação de cada vez) ou dual-port (duas operações simultâneas), útil para sistemas com múltiplos acessos à RAM ao mesmo tempo.

3.6 BUFFER ELÁSTICO

O *buffer* elástico (*elastic buffer* - EB) é um tipo fundamental de registrador que pode ser usado para implementar o protocolo de aperto de mão "ready/valid" (pronto/válido) (DIMITRAKOPOULOS, 2016). Para isso, é essencial que o EB esteja presente em ambas as interfaces, como mostra a figura 3.4, tanto no remetente quanto no destinatário.

Figura 3.4 – Buffer elástico nas interfaces do remetente e destinatário.



Fonte: DIMITRAKOPOULOS (2016).

O EB é projetado para aceitar novos dados e transferir dados disponíveis somente quando os sinais “*ready*” (pronto) e “*valid*” (válido) estiverem ambos em estado alto, ou seja, quando ambos forem iguais a 1.

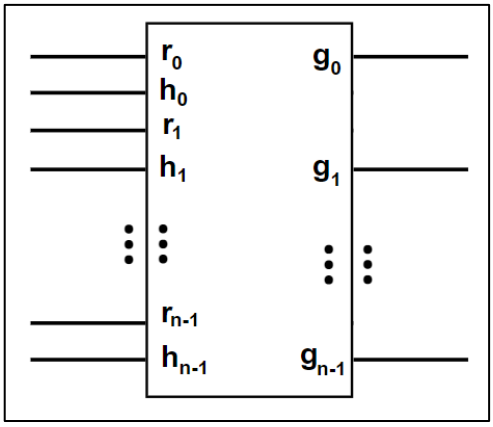
Este tipo de protocolo *ready/valid* é frequentemente utilizado em sistemas de comunicação e transferência de dados para garantir que a transmissão de informações ocorra de forma sincronizada e segura, mesmo quando as partes envolvidas operam em velocidades diferentes (CÂMARA, 2017). O remetente só envia

dados quando está pronto (*ready*) para fazê-lo, e o destinatário só recebe dados quando está pronto para processá-los (*valid*). Isso ajuda a evitar problemas como a perda de dados ou a sobrecarga do sistema.

3.7 ÁRBITRO ROUND ROBIN

Visto que cada porta de saída de um roteador é compartilhada por todas as portas de entrada, surge a necessidade de organizar múltiplos pedidos para um mesmo recurso. Esta tarefa é atribuída aos árbitros (DALLY, 2003), responsáveis por distribuir o acesso a esse recurso para uma porta de entrada de cada vez.

Figura 3.5 – Símbolo de um árbitro.



Fonte: DALLY (2003).

As entradas identificadas como r_n na figura 3.5 são denominadas “*request lines*" (linhas de solicitação) e são utilizadas pelo agente para requisitar o acesso ao recurso sob arbitragem. O árbitro, por sua vez, faz a seleção entre essas solicitações por meio das "grant lines" (linhas de concessão) g_n. Já as entradas h_n são as "hold lines" (linhas de retenção) e têm a responsabilidade de permitir que uma concessão seja mantida pelo tempo que for necessário (CÂMARA, 2017).

O funcionamento do árbitro baseia-se em sua justiça, geralmente categorizada como:

- **Justiça fraca:** Garante que todos os pedidos serão atendidos em algum momento.
- **Justiça forte:** Assegura que os pedidos sejam atendidos frequentemente de forma equitativa.

- **Justiça FIFO:** Os pedidos são atendidos na ordem em que são realizados.

De acordo com CÂMARA (2017, p. 34) os árbitros podem, também, ser categorizados quanto à sua prioridade, isto é:

> "Árbitros de prioridade fixa são considerados injustos. Árbitros de prioridade variável iterativa mudam sua prioridade a cada ciclo e são considerados justos. Seu tipo de justiça depende do gerador do bit de prioridade p_n. Um árbitro com gerador de prioridade cujo pricípio de operação seja o pedido que acabou de ser servido deverá ter a prioridade mais baixa no próximo ciclo é chamado de árbitro *RoundRobin* (RoR). Árbitros *round-robin*, para uma demanda igualmente distribuída, apresentam justiça forte."

E, ainda que existam árbitros mais eficazes capazes de proporcionar uma justiça aprimorada, a autora conclui informando que os árbitros do tipo *round-robin* são comumente adotados em roteadores para redes em chip.

CAPÍTULO 4 – PROJETO E ANÁLISE DE PORTAS LÓGICAS BÁSICAS BASEADAS EM *MEMRISTORS*:

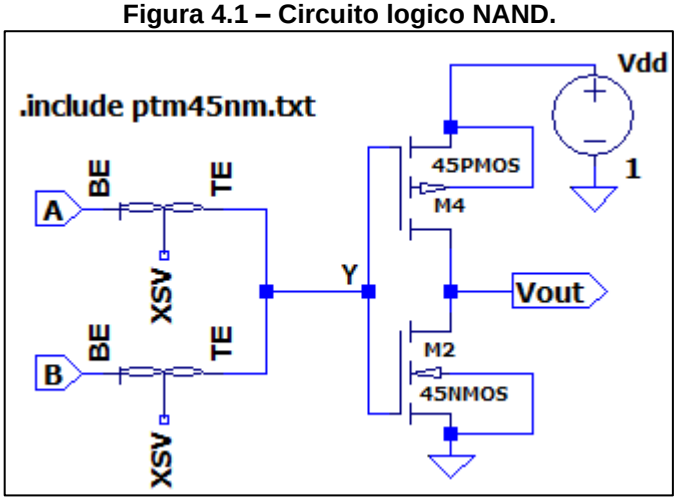
Para a elaboração do presente documento, as simulações foram conduzidas empregando o software LTspice IV, que é um simulador SPICE de alto desempenho amplamente adotado na esfera acadêmica. Reconhecido como um dos principais simuladores de circuitos freeware, o LTspice oferece uma plataforma robusta. Seu editor de esquemas facilita a criação e adaptação de circuitos de maneira ágil e descomplicada, permitindo inclusive a inclusão de modelos macro para simplificar a montagem de circuitos intrincados.

4.1 PORTA NAND

A fundamentação do circuito NAND utilizando *memristors* e transistores MOS tem início na compreensão do funcionamento da fase inicial do sistema. No esquemático mostrado na figura 4.1, os *memristors* associados as entradas A e B colaboram para formar a lógica AND, resultando na definição do nó Y como:

$$\boldsymbol{Y = A \cdot B} \tag{4.1}$$

Figura 4.1 – Circuito logico NAND.



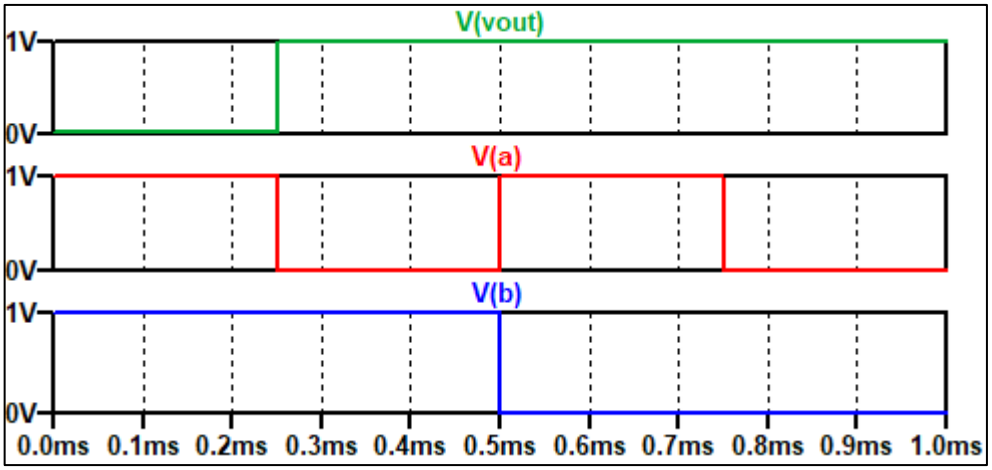
Fonte: Autor.

A partir desse ponto, o circuito incorpora um conjunto CMOS de uma porta inversora. O funcionamento dessa configuração foi detalhado na seção 3. Essa estrutura permite que Y = 1 leve a V_{OUT} = 0, e Y = 0 leve a V_{OUT} = 1. Dessa forma, a

formulação completa do circuito expresso na figura 4.1, cujo resultado é evidenciado na figura 4.2, é vista como:

$$V_{OUT} = \overline{A \cdot B} \tag{4.2}$$

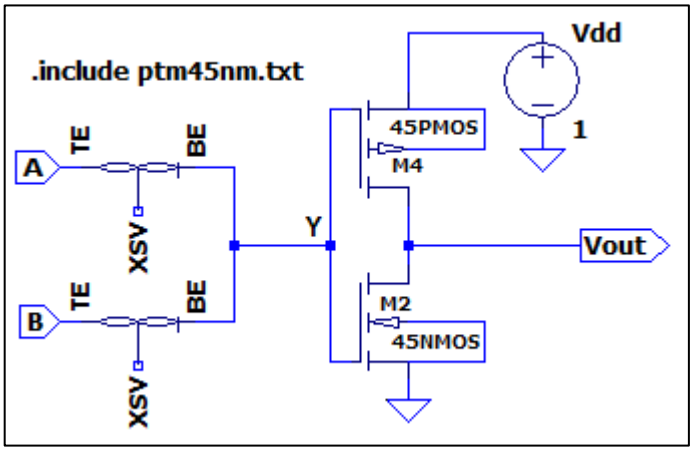
Figura 4.2 – Comportamento da lógica NAND implementada.



Fonte: Autor.

4.2 PORTA NOR

Figura 4.3 – Circuito logico NOR.



Fonte: Autor.

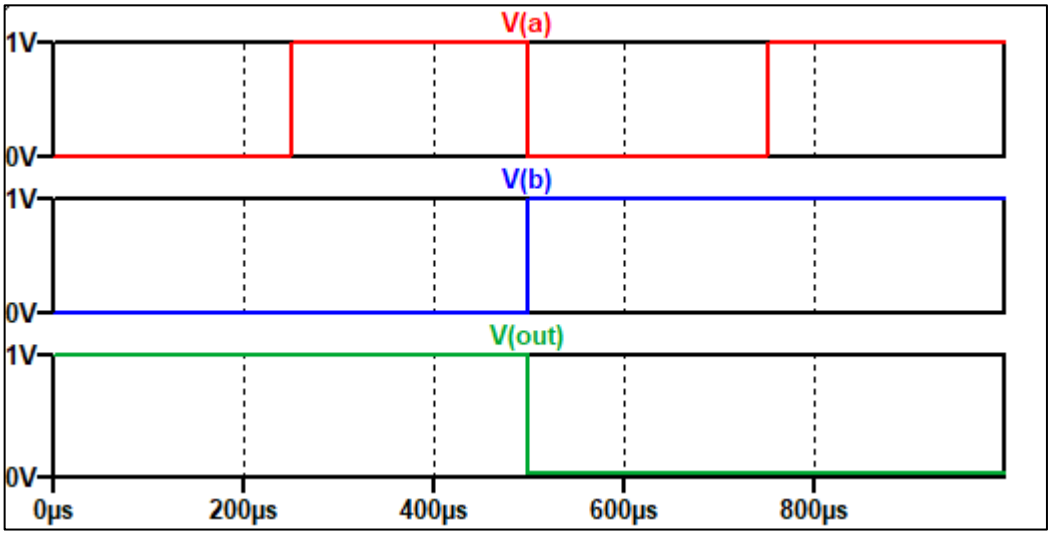
A principal diferença entre as portas NAND e NOR reside na polaridade oposta dos *memristors* em relação às suas entradas (CHO et al. 2015). Sendo assim, a implementação da lógica NOR (evidenciada na figura 4.3) através de *memristors* e transistores MOS segue o mesmo princípio que foi demonstrado para a lógica NAND, no entanto, devido a polaridade dos *memristors*, agora:

$$Y = A + B \tag{4.3}$$

$$V_{OUT} = \overline{A + B} \tag{4.4}$$

O resultado de um teste prático da porta NOR pode ser visualizado na figura 4.4:

Figura 4.4 – Comportamento da lógica NOT implementada.

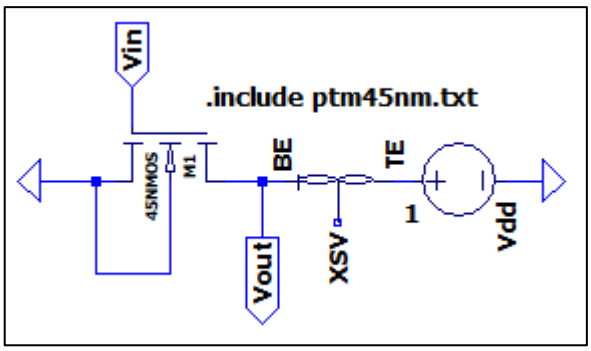


Fonte: Autor.

4.3 PORTA NOT

No estudo de SINGH (2017, apud Liu et al. 2021), foi proposto um projeto de circuito para uma porta NOT que utiliza um *memristor* em conjunto com um transistor NMOS. A configuração é ilustrada na Figura 4.5. Nesse arranjo, a porta NOT é alimentada por uma tensão contínua V_{DD}, aplicada na entrada frontal do *memristor*. Onde, V_{IN} denota o sinal de entrada, enquanto V_{OUT} representa o sinal de saída correspondente.

Figura 4.5 – Circuito logico NOT_mem[1].



Fonte: Autor.

No cenário em que V_{IN} = 1 (indicando uma alta tensão de entrada), o transistor entra em condução, resultando na polarização progressiva do *memristor* com uma memristência de valor R_{ON}. Nessa condição, o transistor NMOS está saturado, com a resistência de condução nula ($R_T \approx 0$), e a saída do circuito é conforme demonstra a equação (4.5):

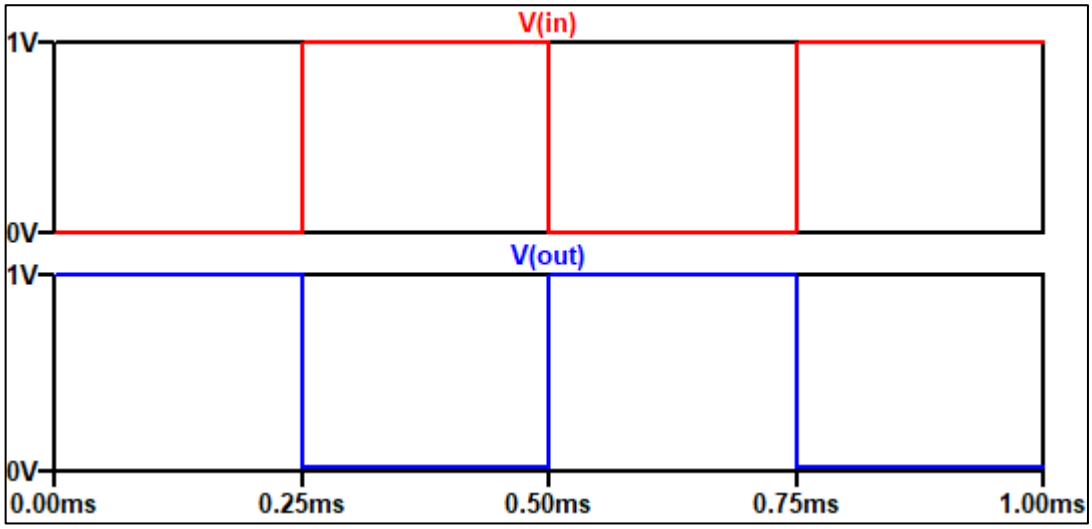
$$V_{OUT} = \frac{R_T}{R_{ON} + R_T} * V_{DD} \approx 0 \tag{4.5}$$

Quando, por outro lado, V_{IN} = 0, o transistor é levado ao estado de corte. Isso faz com que a resistência $R_T \approx \infty$ e a saída assume o seguinte formato:

$$V_{OUT} = \frac{R_T}{R_{ON} + R_T} * V_{DD} \approx V_{DD} \tag{4.6}$$

O comportamento da logica NOT, implementada na figura 4.5, segue tal qual evidenciado na figura 4.6:

Figura 4.6 – Comportamento da lógica NOT implementada.



Fonte: Autor.

Além da metodologia demonstrada na figura 4.5, é possível desenvolver uma porta inversora utilizando-se apenas inversores CMOS tradicionais. Essa configuração foi explicitada no capitulo 2.3, figura 2.8, e é aplicada na construção das lógicas NAND e NOR memresistivas. Sendo assim, por possuir um inversor "embutido" em sua configuração, as portas NAND e NOR conseguem, sozinhas,

[1] Os circuitos lógicos desenvolvidos, em grande maioria, responderam melhor à lógica NOT implementada utilizando a porta NAND, excetuando-se o circuito CONTADOR, que fui implementado utilizando a logica NOT_mem.

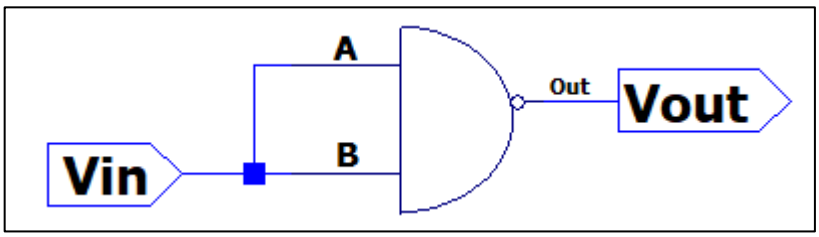
replicar o funcionamento de um inversor. Para isto, basta replicar a entrada (V_{IN}) nos terminais A e B destas lógicas, conforme segue-se abaixo:

$$V_{OUT} = \overline{V_{IN} \cdot V_{IN}} = \overline{V_{IN}} \quad (4.7)$$

Ou, igualmente:

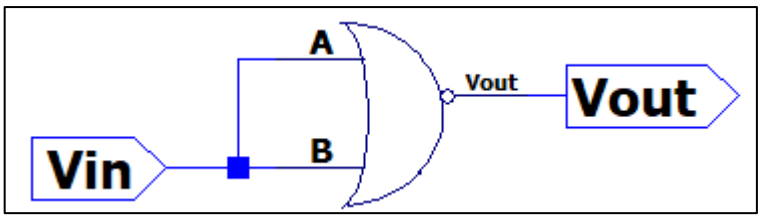
$$V_{OUT} = \overline{V_{IN} + V_{IN}} = \overline{V_{IN}} \quad (4.8)$$

Figura 4.7 – Modelo alternativo (NAND) da lógica NOT.



Fonte: Autor.

Figura 4.8 – Modelo alternativo (NOR) da lógica NOT.



Fonte: Autor.

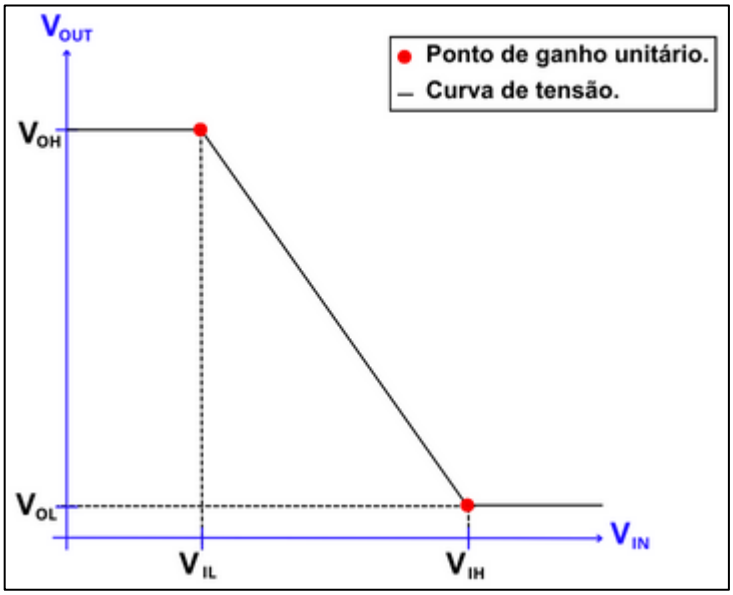
4.4 MARGEM DE RUÍDO

A margem de ruído em uma porta lógica é uma medida que indica a diferença máxima permitida entre as tensões de entrada (níveis lógicos alto e baixo) para garantir que a porta funcione corretamente (SEDRA et al., 2023). Em outras palavras, a margem de ruído representa a faixa de variação de sinal de entrada na qual a porta lógica ainda reconhece o nível de entrada como alto ou baixo de forma confiável.

Sendo uma característica importante para garantir a operação estável de circuitos digitais, a margem de ruido leva em consideração a variação de tensão causada por ruído elétrico e outros fatores que podem afetar o sinal de entrada. Quanto maior a margem de ruído, mais robusta é a porta lógica em relação às flutuações de sinal e ao ruído elétrico. Isso significa que a porta pode tolerar variações maiores nas tensões de entrada antes de começar a interpretar erroneamente o sinal.

Considerando que as portas lógicas utilizadas neste trabaho foram ajustadas (ver tabela 4.1) de acordo com os parâmetros R_{on} e R_{off} do *memristor* e largura (W) e comprimento (L) dos transistores para fornecerem a maior robustez possível, a análise da margem de ruído divide-se em dois parâmetros, a alta margem de ruído (MR_H) e a baixa margem de ruído (MR_L). MR_B é definida pela diferença entre a tensão de entrada baixa máxima (V_{IL}) reconhecida pela porta lógica e a tensão de saída baixa máxima (V_{OL}) produzida por ela. MR_H é a diferença entre a tensão de saída alta mínima (V_{OH}) da porta lógica e a tensão de entrada alta mínima (V_{IH}) reconhecida por ela (FARIAS et al., 2018). Esses níveis lógicos de tensão são definidos no ponto de ganho unitário, onde a inclinação é -1, conforme ilustrado na figura 4.9 (HILL, 1968 apud FARIAS et al., 2018).

Figura 4.9 – Curva de transferência de tensão de uma porta lógica.



Fonte: Autor.

A curva mostrada na figura 4.9 é obtida através da simulação do tipo *DC_sweep* presente no software LTSpice. Esta simulação tem por objetivo aumentar gradativamente o valor da fonte de tensão entre os níveis lógicos alto e baixo, 0V à 1V respectivamente, exibindo, de forma gráfica, a variação de tensão na saída da porta lógica analisada, afim de revelar com quais valores de entrada a porta lógica comuta o nível de tensão na saída.

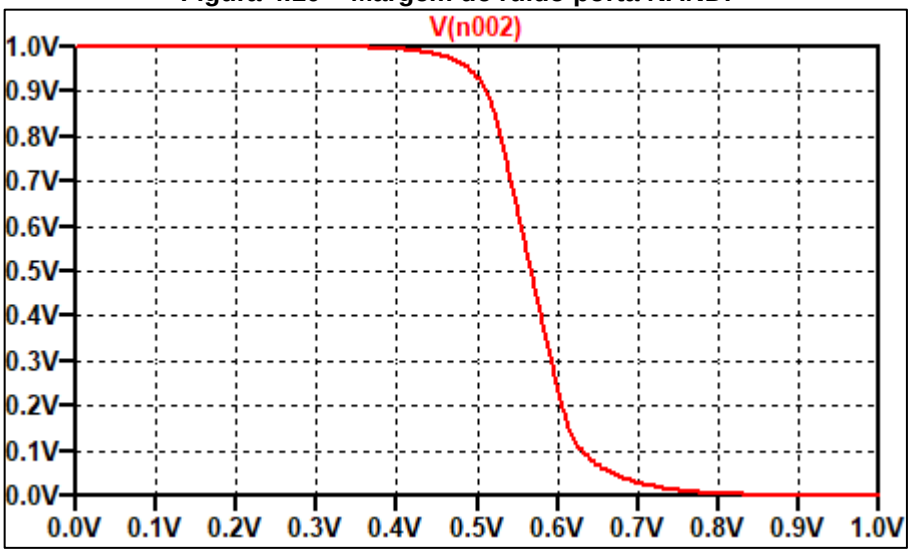
Tabela 4.1 – Valores de cada parâmetro das portas lógicas NAND e NOR.

PARAMETRO	VALOR
R_{on}	100Ω
R_{off}	100kΩ
L – PMOS	45nm
W – PMOS	90nm
L – NMOS	90nm
W – NMOS	45nm

Fonte: Autor.

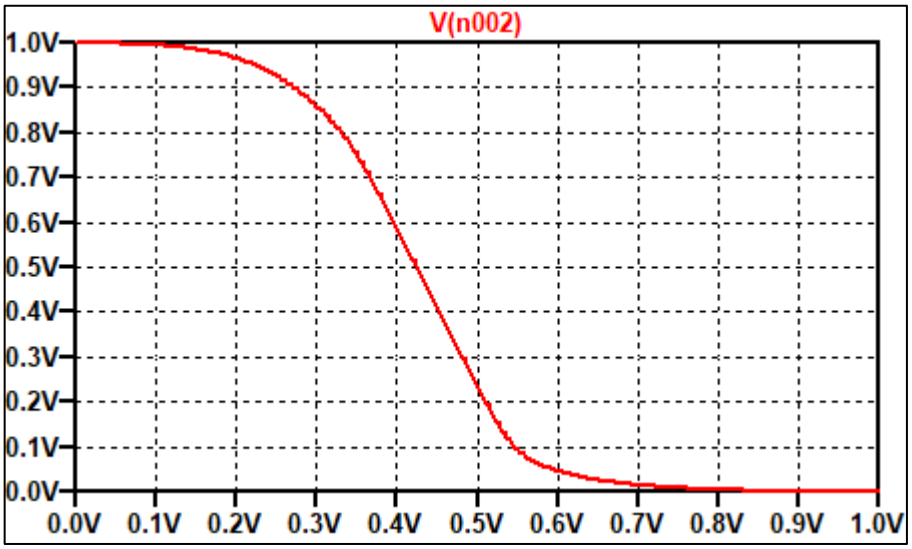
As figuras 4.10, 4.11 e 4.12 exibem as curvas de transferência de tensão para as portas NAND, NOR e NOT_mem, respectivamente. Destas curvas é possível extrair os valores de V_{OL}, V_{OH}, V_{IL} e V_{IH} e, consequentemente, calcular MR_H e MR_L.

Figura 4.10 – Margem de ruido porta NAND.



Fonte: Autor.

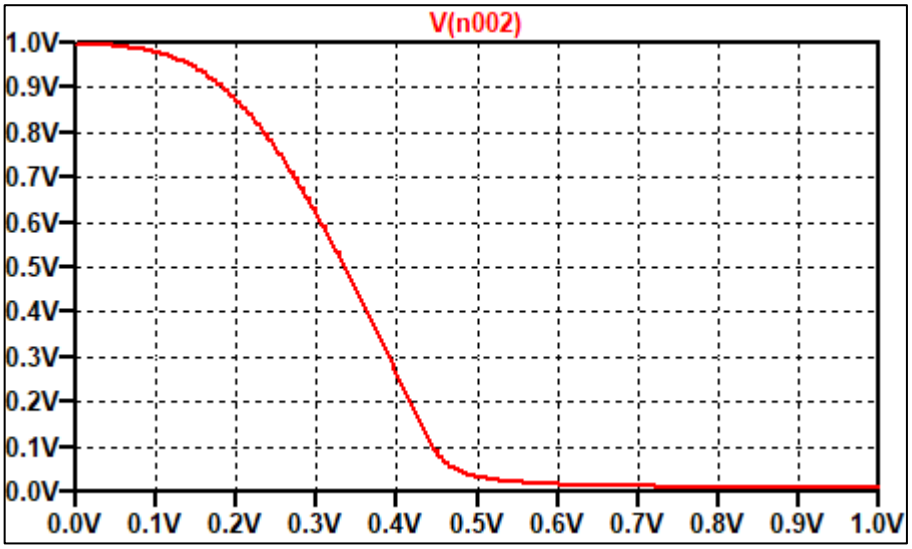
Figura 4.11 – Margem de ruido porta NOR.



Fonte: Autor.

É importante salientar que na concepção da porta NOT_mem adotou-se um valor para o parâmetro W igual à três vezes a largura do transistor PMOS evidenciado na tabela 4.1. Isto foi feito objetificando dar mais robustez a porta lógica inversora, visto que possui uma configuração singular se comparada as demais. Os resultados dos cálculos de MR_H e MR_L, para cada porta lógica fundamental, são exibidos na tabela 4.2.

Figura 4.12 – Margem de ruido porta NOT_mem.



Fonte: Autor.

Tabela 4.2 – Parâmetros e resultados do cálculo da margem de ruido.

PORTA	VI_L [mV]	VO_L [mV]	MR_L [mV]	VO_H [mV]	VI_H [mV]	MR_H [mV]
NOT	151,87	51,54	100,32	940,86	471,18	469,67
NOR	204,87	72,78	132,09	962,00	565,42	396,58
NAND	479,18	56,87	422,31	963,67	659,86	303,81

Fonte: Autor.

Em termos comparativos, uma porta inversora tradicionalmente construída com transistores MOS de 45nm, possui valores de MR_L e MR_H iguais à 480mV e 650mV (FARIAS et al., 2018). Tal informação sugere que circuitos inversores tradicionais são menos sensíveis a ruido do que os construídos com memristors, indicando que os resistores de memória, nesta topologia, não são adequados para projetos de circuitos multiníveis, com várias conexões em série, o que de fato é verídico.

Essa última afirmação é pautada no fato de que circuitos mais complexos, tal qual a SRAM hierarquizada, não funcionaram corretamente nos testes realizados no desenvolvimento deste trabalho, como será discutido no capitulo 6. Uma solução

factível seria aumentar o tamanho dos transistores utilizados, visto que isto aumentaria a margem de ruido (FARIAS et al., 2018). No entanto, visto que o tamanho dos transistores é diretamente proporcional a velocidade de comutação (FARIAS et al., 2018), o aumento necessário para causar uma diferença notável no desempenho dos circuitos idealizados os deixariam lentos demais, impossibilitando o correto funcionamento dos mesmos. Logo, como o ruído pode ser contornado de outras formas, aplicando filtros por exemplo, decidiu-se dar prosseguimento com a configuração W e L que proporcionaria maior velocidade e menor área ao roteador.

Ao fim, os testes realizados nesta seção inferem a importância de, ao iniciar o projeto de circuitos digitais, ter um profundo entendimento das especificações das portas lógicas particulares que estão sendo empregadas, a fim de assegurar que a margem de ruído seja apropriada e atenda às exigências específicas da aplicação em questão.

4.5 ATRASO DE COMUTAÇÃO

O atraso de comutação em portas lógicas digitais é o tempo necessário para que a saída de uma porta lógica responda a uma mudança na entrada, podendo ser dividido em atraso de subida (tempo para a saída mudar de alto para baixo) e atraso de descida (tempo para a saída mudar de baixo para alto). Esse parâmetro, influenciado pela tecnologia da porta e pela carga conectada, desempenha um papel crucial na temporização e estabilidade de sistemas digitais, sendo essencial para projetos que envolvem alta velocidade, onde os atrasos de comutação de múltiplas portas ao longo de um caminho são somados para calcular o atraso total do sistema.

No presente caso de aplicação, tendo em vista que o tempo de comutação das fontes no LTSpice pode chegar a 10^{-15} segundos, o atraso de comutação depende somente das capacidades inerentes ao modelo de *memristor* utilizado e, portanto, é significativamente baixo, ficando numa faixa inferior a 40ps para a tensão V_{DD} utilizada, mas ainda acima dos 20ps observados em inversores CMOS 45nm tradicionais.

CAPÍTULO 5 – BIBLIOTECA SPICE MEMRESISTIVA:

O roteador é um dispositivo de natureza predominantemente digital, composto por módulos de complexidade crescente, que podem ser decompostos até o nível das portas lógicas, consideradas os blocos fundamentais da arquitetura digital. Na subseção anterior, foram abordadas as portas NOT, NAND e NOR memresistivas. Todos os outros elementos, como latches e flip-flops, serão desenvolvidos com base nessas portas, seguindo uma abordagem de projeto hierárquico. É relevante destacar que, embora a arquitetura digital dos componentes desenvolvidos aqui esteja bem documentada na literatura, a eletrônica utilizada difere daquela empregada na validação dos modelos clássicos. Portanto, torna-se essencial validar esses modelos específicos que utilizam *memristors*.

Esta seção apresenta tanto a arquitetura quanto os resultados de simulação desses elementos, com o objetivo de realizar a correspondente validação. Todas as simulações foram conduzidas utilizando o software LTspice IV, com uma tensão de alimentação (V_{DD}) de 1V e um stop time de até 8ms, em uma máquina equipada com um processador AMD Ryzen 5600G de 6 núcleos operando a 3,9 GHz, 16 GB de memória RAM e um SSD NMVe de 1 TB. A despeito da velocidade e eficácia desse computador na execução de programas mais exigentes, a simulação de alguns circuitos mais robustos demandou alguns minutos para sua conclusão.

Grande parte dos 38 circuitos lógicos recriados nesta biblioteca possui sua lógica de funcionamento e construção amplamente difundidas em fontes confiáveis da internet. Contudo, lógicas mais complexas requerem uma pesquisa mais ampla e testes diversos de funcionamento. Sendo assim, nos circuitos Contador, Demux, PISO e Divisor de frequência, optou-se por adotar a metodologia proposta por CÂMARA (2017), onde a autora propõe uma série de circuitos lógicos baseados na hierarquização no software LTSpice.

5.1 PORTA OR

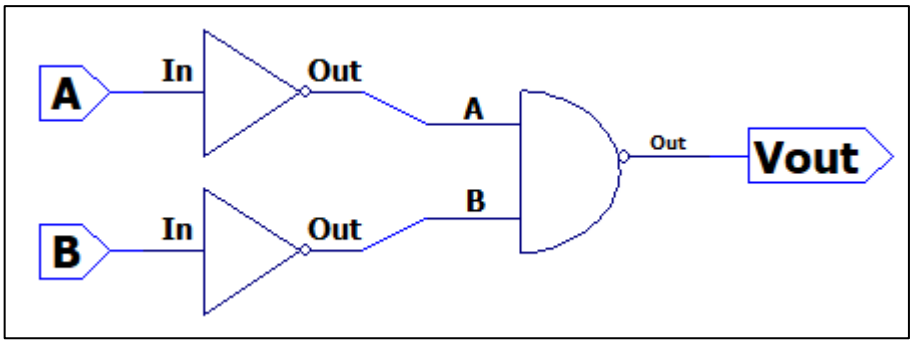
5.1.1 **Porta OR de duas entradas**

Para criar uma porta OR de duas entradas, é possível utilizar uma porta NAND de duas entradas e duas portas NOT, conforme evidenciado na figura 5.1. A função

booleana resultante desta combinação fica descrita pela equação (5.1) e o resultado de um teste virtual pode ser visualizado na figura 5.2:

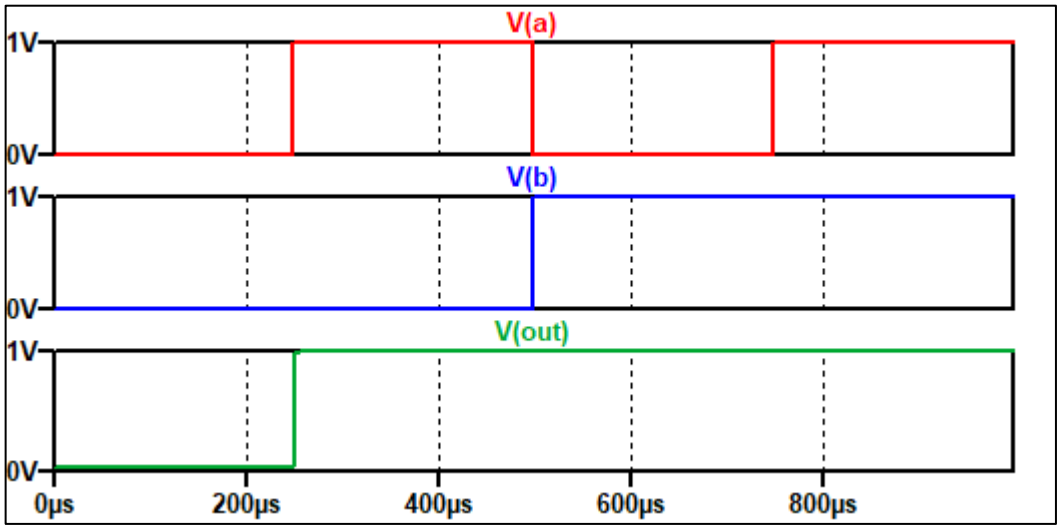
$$V_{OUT} = \overline{\overline{A} \cdot \overline{B}} = A + B \tag{5.1}$$

Figura 5.1 – Modelo SPICE da lógica OR memresistiva com duas entradas.



Fonte: Autor.

Figura 5.2 – Resultado da lógica OR memresistiva com duas entradas.



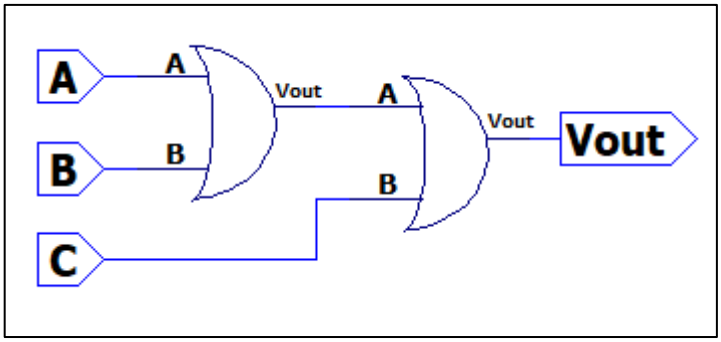
Fonte: Autor.

5.1.2 **Porta OR de três entradas**

Uma das formas simples de implementar uma lógica OR de três entradas é utilizando-se duas portas OR de duas entradas, vide figura 5.3, onde:

$$V_{OUT} = (A + B) + C = A + B + C \tag{5.2}$$

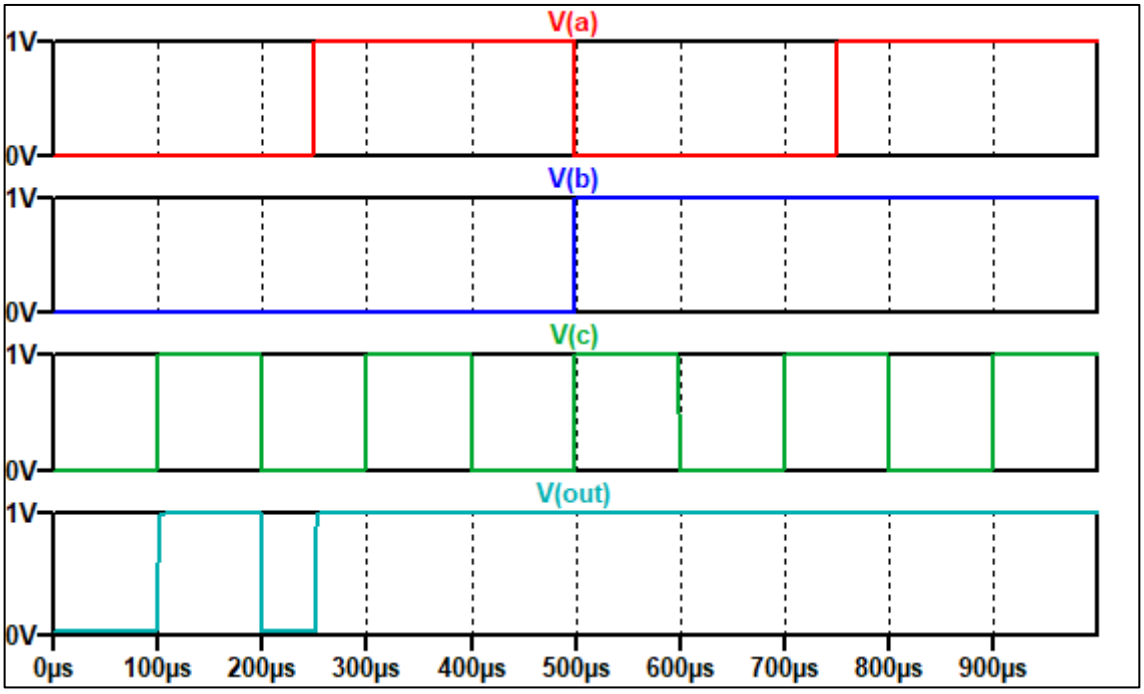
Figura 5.3 – Modelo SPICE da lógica OR memresistiva com três entradas.



Fonte: Autor.

Na figura 5.4, mostrada a seguir, é ilustrado o resultado de um teste simples utilizando a porta OR com três entradas.

Figura 5.4 – Resultado da lógica OR memresistiva com três entradas.



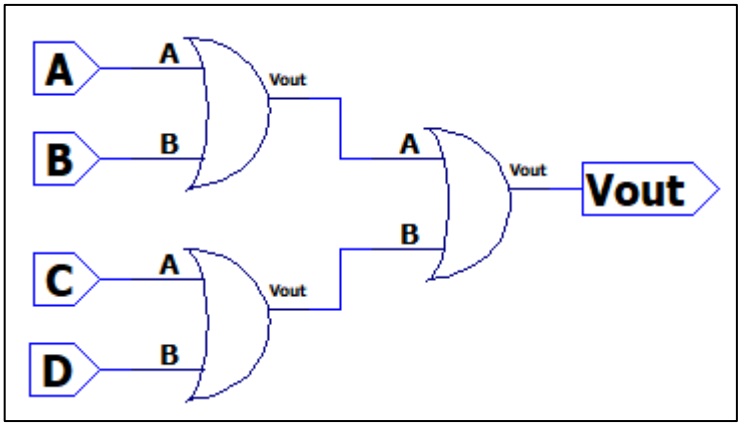
Fonte: Autor.

5.1.3 **Porta OR de quatro entradas**

Semelhante a implementação anterior, uma lógica OR de quatro entradas pode ser realizada via três portas OR de duas entradas seguindo o esquemático da figura 5.5. A função booleana desta lógica é destacada na equação (5.3) e ilustrada na figura 5.6:

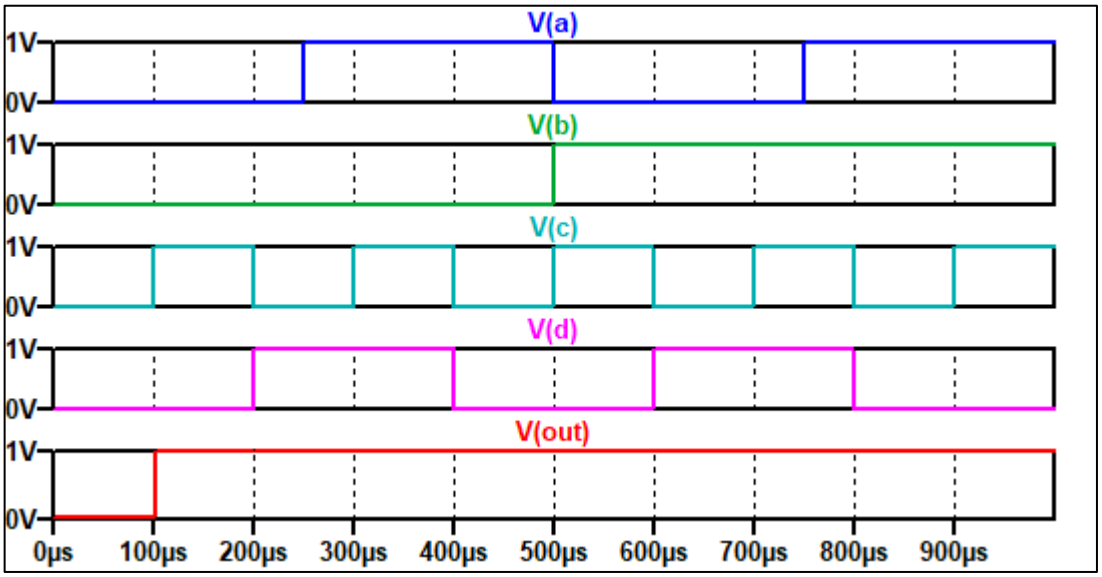
$$\boldsymbol{V_{OUT} = (A + B) + (C + D) = A + B + C + D} \qquad (5.3)$$

Figura 5.5 – Modelo SPICE da lógica OR memresistiva com quatro entradas.



Fonte: Autor.

Figura 5.6 – Resultado da lógica OR memresistiva com quatro entradas.



Fonte: Autor.

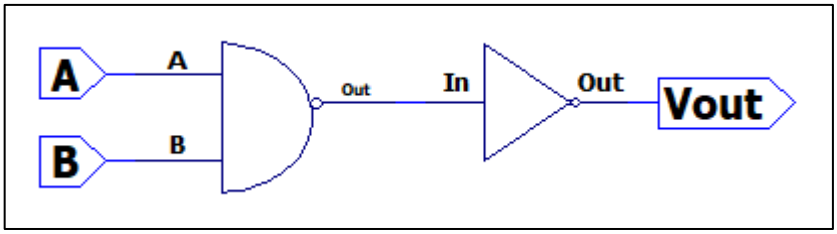
5.2 PORTA AND

5.2.1 **Porta AND de duas entradas**

Uma porta AND de duas entradas é obtida através da combinação de uma porta NOT e uma porta NOR de duas entradas seguindo o esquema proposto na figura 5.7. Sua expressão booleana é definida vide equação (5.4), da qual resulta-se o exemplo mostrado na figura 5.8:

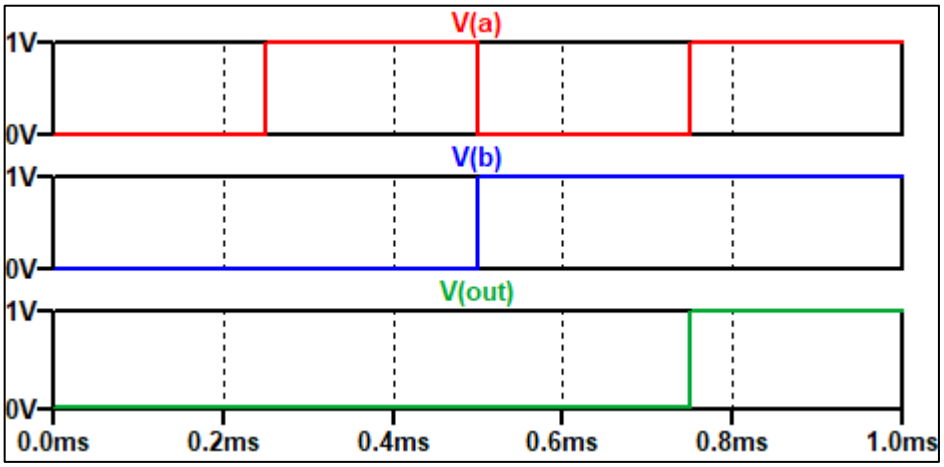
$$\boldsymbol{V_{OUT} = \overline{\overline{A} + \overline{B}} = A \cdot B} \tag{5.4}$$

Figura 5.7 – Modelo SPICE da lógica AND memresistiva com duas entradas.



Fonte: Autor.

Figura 5.8 – Resultado da lógica AND memresistiva com duas entradas.



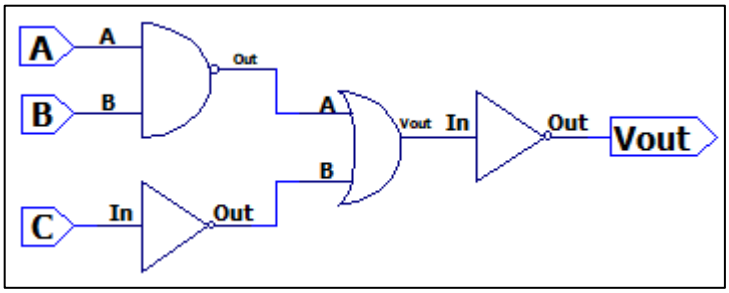
Fonte: Autor.

5.2.2 **Porta AND de três entradas**

A porta AND de três entradas pode ser construída usando duas portas NAND de duas entradas e duas portas NOT obedecendo o esquema da figura 5.9. Sua função booleana, expressa na equação (5.5), é exemplificada na figura 5.10:

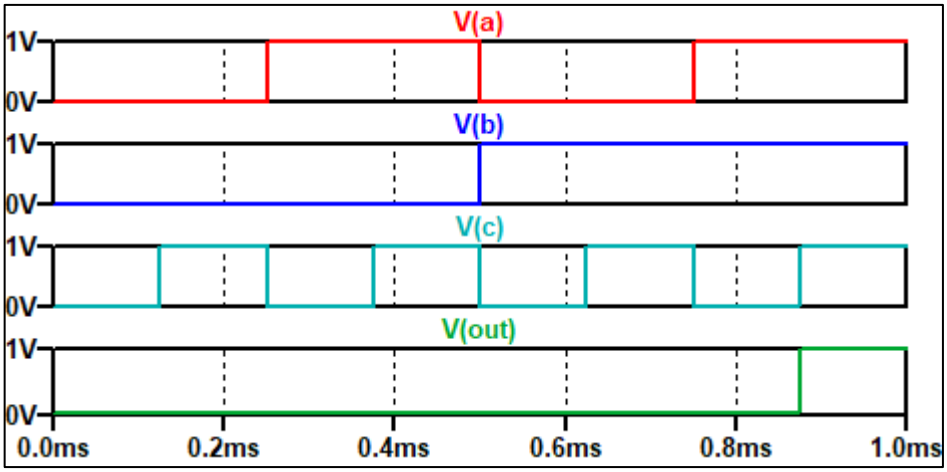
$$V_{OUT} = \overline{\left(\left(\overline{A \cdot B}\right) + \overline{C}\right)} = A \cdot B \cdot C \tag{5.5}$$

Figura 5.9 – Modelo SPICE da lógica AND memresistiva com três entradas.



Fonte: Autor.

Figura 5.10 – Resultado da lógica AND memresistiva com três entradas.



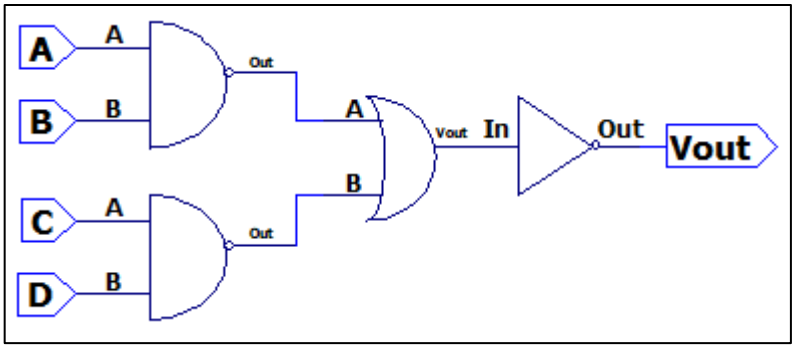
Fonte: Autor.

5.2.3 **Porta AND de quatro entradas**

Finalmente, a porta AND de quatro entradas pode ser criada, conforme ilustra a figura 5.11, combinando duas portas NAND de duas entradas, uma porta OR de duas entradas e uma porta NOT. Sua função booleana é expressa na equação (5.6) e exemplificada na figura 5.12:

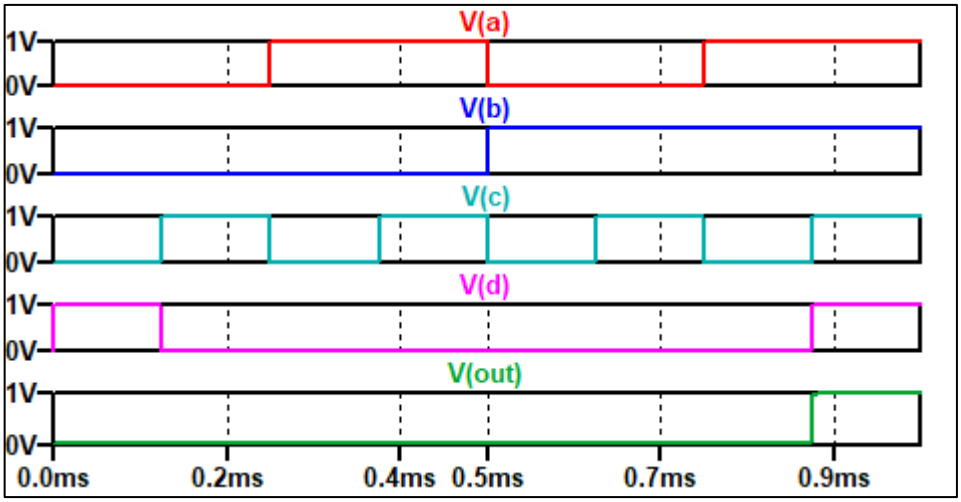
$$V_{OUT} = \overline{\overline{(A \cdot B)} + \overline{(C \cdot D)}} = A \cdot B \cdot C \cdot D \tag{5.6}$$

Figura 5.11 – Modelo SPICE da lógica AND memresistiva com quatro entradas.



Fonte: Autor.

Figura 5.12 – Resultado da lógica AND memresistiva com quatro entradas.



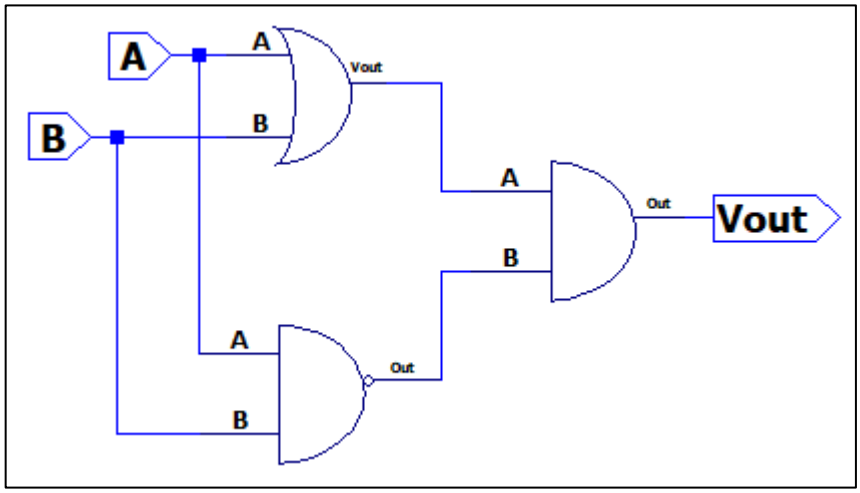
Fonte: Autor.

5.3 PORTA XOR

A implementação da lógica XOR de duas entradas pode ser realizada de várias maneiras, sendo uma delas a combinação das portas NAND, NOR e NAND, conforme ilustrado na figura 5.13 e comprovado na equação (5.7):

$$\boldsymbol{V_{OUT} = \overline{(A + B) \cdot (\overline{A \cdot B})} = A \oplus B} \tag{5.7}$$

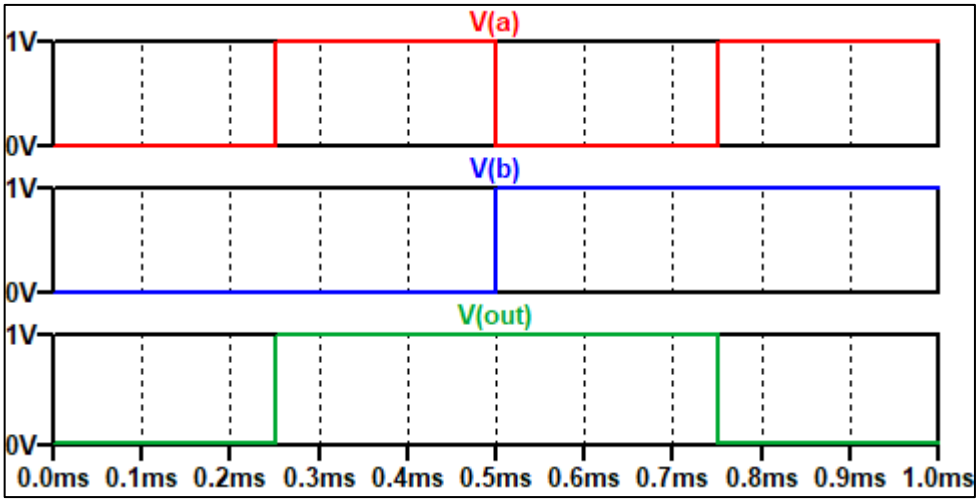
Figura 5.13 – Modelo SPICE da lógica XOR memresistiva com duas entradas.



Fonte: Autor.

Como intuito de comprovar o funcionamento da lógica desenvolvida, na figura 5.14 é demonstrado o resultado do teste realizado na porta XOR construída via *memristors*.

Figura 5.14 – Resultado da lógica XOR memresistiva com duas entradas.

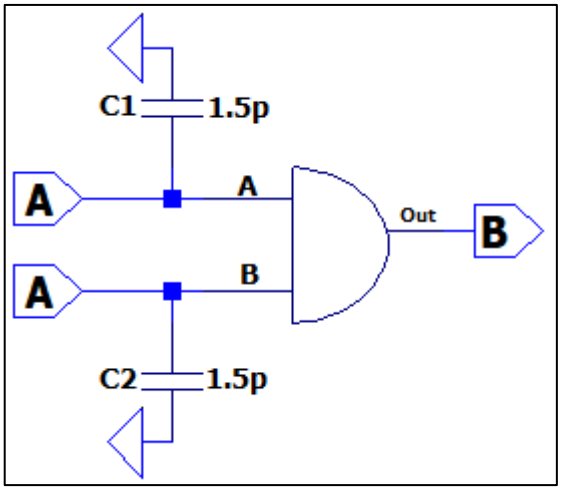


Fonte: Autor.

5.4 FILTRO CAPACITIVO

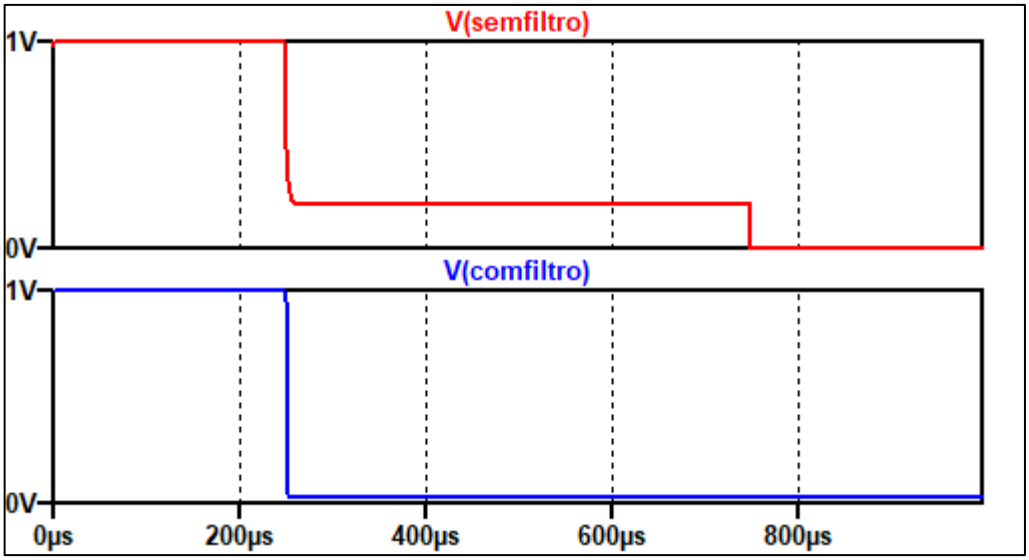
Com o objetivo de corrigir algumas falhas de simulação decorrentes das aproximações matemáticas inerentes aos modelos de *memristors* e transistores utilizados, foi desenvolvido um filtro capacitivo equipado com um capacitor de 1,5pF em cada entrada de uma porta NAND, vide figura 5.15. Os capacitores têm a finalidade de eliminar qualquer disparo indesejado de tensão, enquanto a porta NAND, com uma alimentação V_{DD} de 1V, serve para padronizar os níveis lógicos alto e baixo como um e zero, respectivamente, tal qual evidenciado na figura 5.16.

Figura 5.15 – Modelo SPICE do filtro capacitivo.



Fonte: Autor.

Figura 5.16 – Sinal sem e com filtragem, respectivamente.

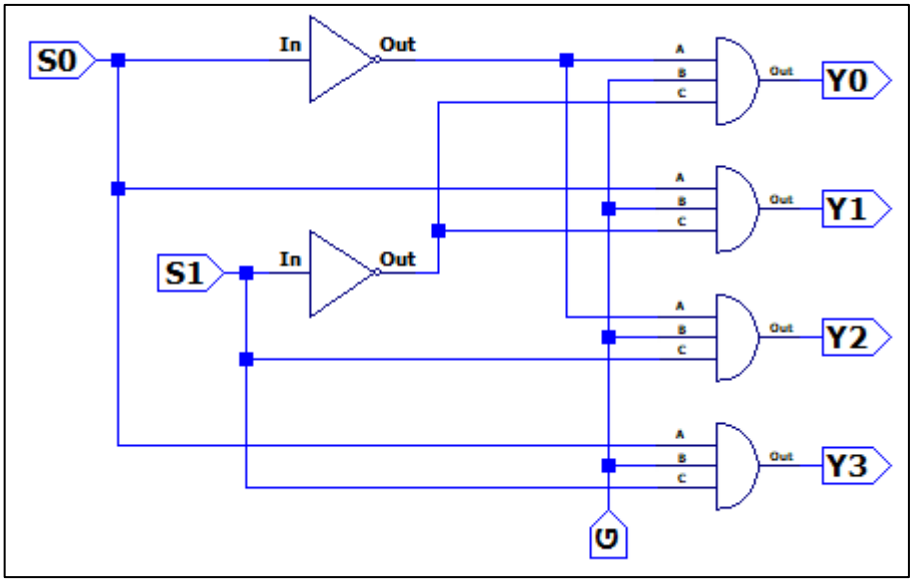


Fonte: Autor.

5.5 DECODIFICADOR BINÁRIO

O processo de decodificação envolve a conversão de uma entrada codificada em uma saída que utiliza um código diferente. Em geral, a saída codificada possui um número maior de bits do que a entrada. Há uma correspondência direta entre as palavras codificadas na entrada e as palavras codificadas na saída (WAKERLY apud CÂMARA, 2017). O decodificador mais comumente utilizado é o decodificador binário, também conhecido como decodificador $n{:}\,2^n$. Esse decodificador recebe um código binário de n bits como entrada e gera uma das n saídas correspondentes ao código binário (CÂMARA, 2017). Uma ilustração do decodificador binário mais simples, com duas entradas e quatro saídas, é apresentada na figura 5.17.

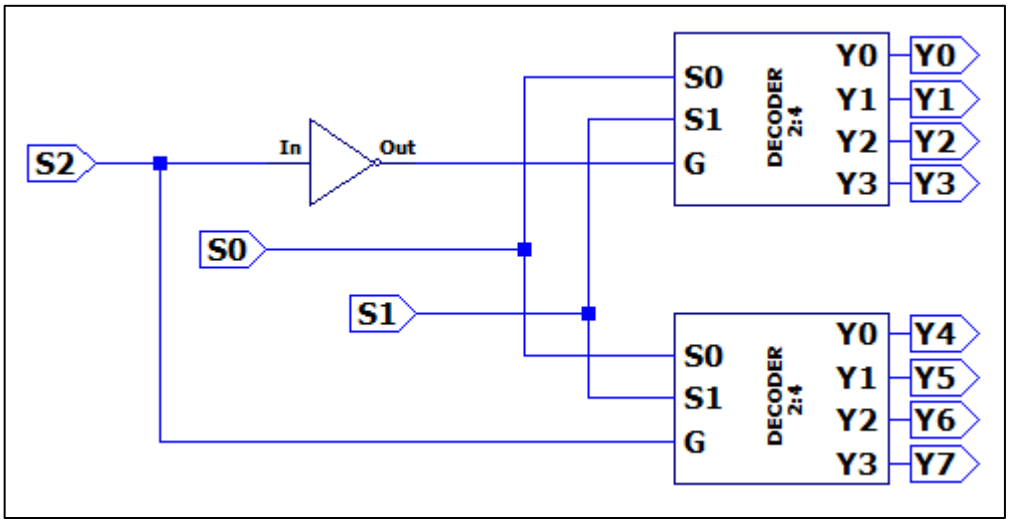
Figura 5.17 – Decodificador SPICE 2:4 memresisitivo.



Fonte: Autor.

Neste decodificador, S1 representa o bit mais significativo (MSB) do código de entrada, e a numeração das saídas corresponde diretamente à conversão decimal do código binário de entrada. O sinal G controla a habilitação do circuito. É possível, ainda, criar um decodificador de três entradas e oito saídas cascateando decodificadores 2:4, conforme descreve a figura 5.18.

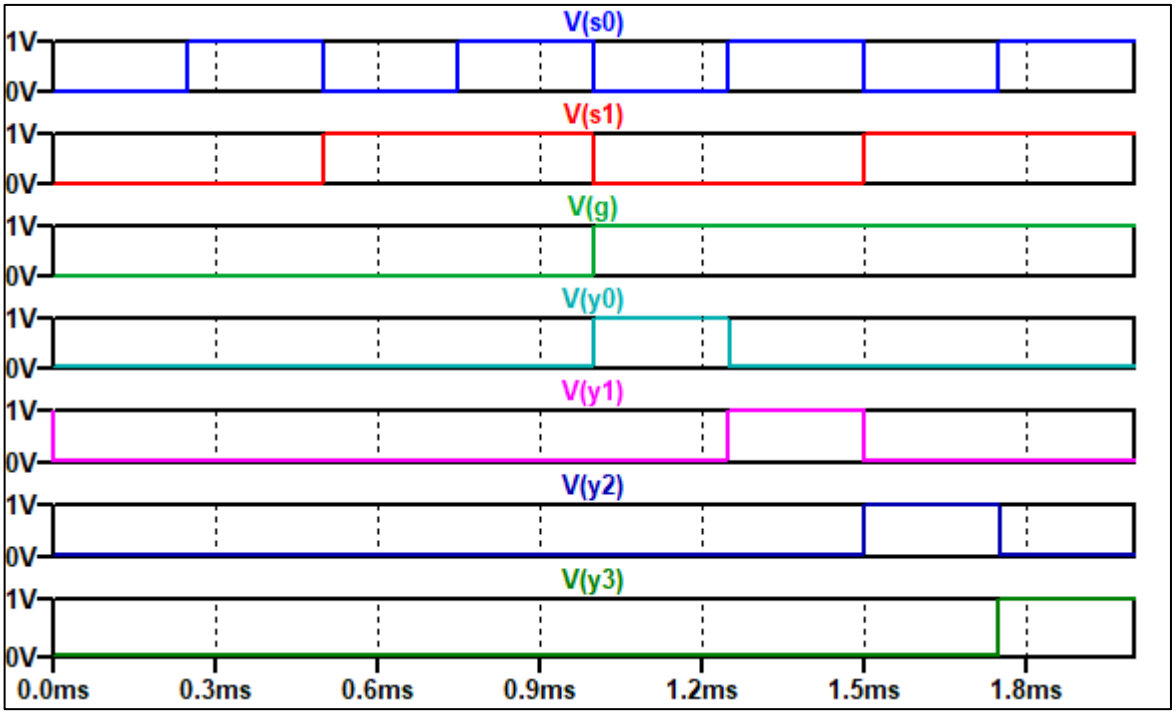
Figura 5.18 – Decodificador SPICE 3:8 memresisitivo.



Fonte: Autor.

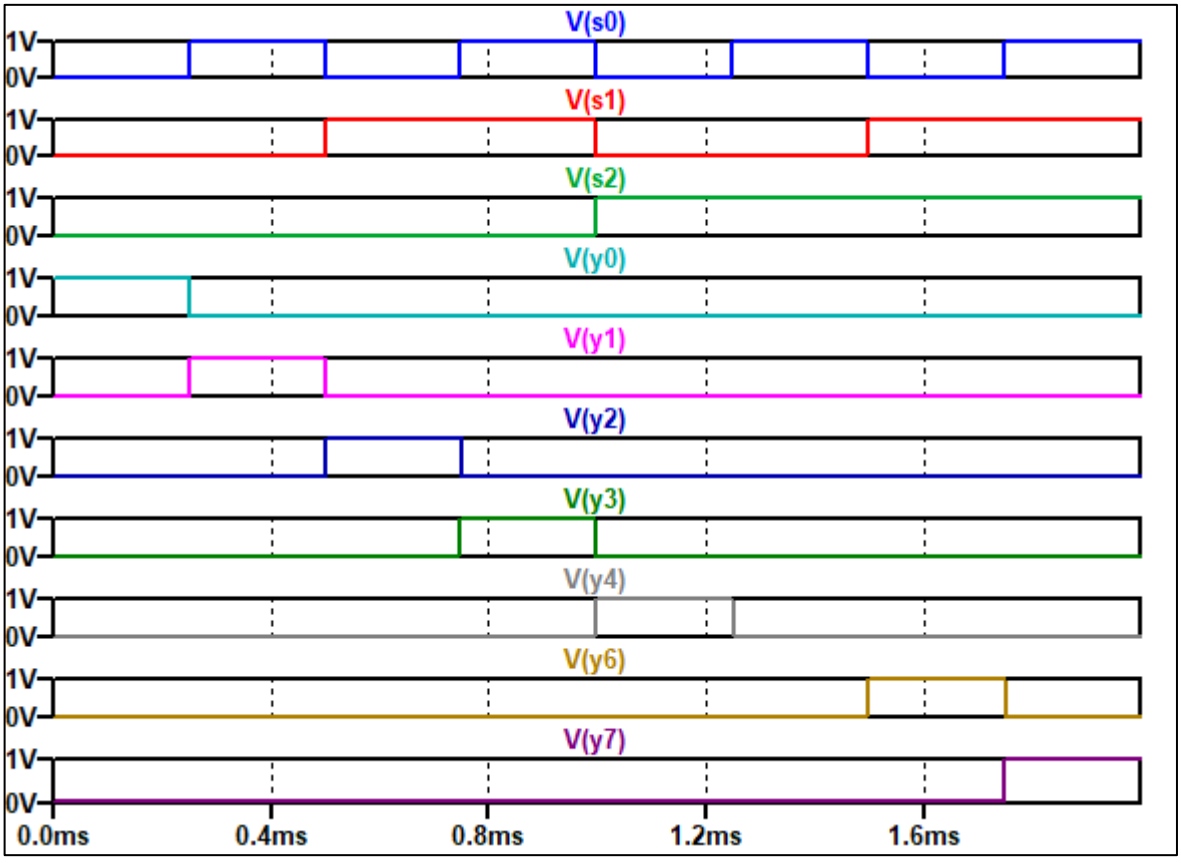
O decodificador 3:8 é apresentado acima, onde S2 é o MSB do código de entrada. As figuras 5.19 e 5.20 exibem os resultados das simulações para os decodificadores 2:4 e 3:8, respectivamente.

Figura 5.19 – Entradas e saídas DECODER 2:4.



Fonte: Autor.

Figura 5.20 – Entradas e saídas DECODER 3:8.

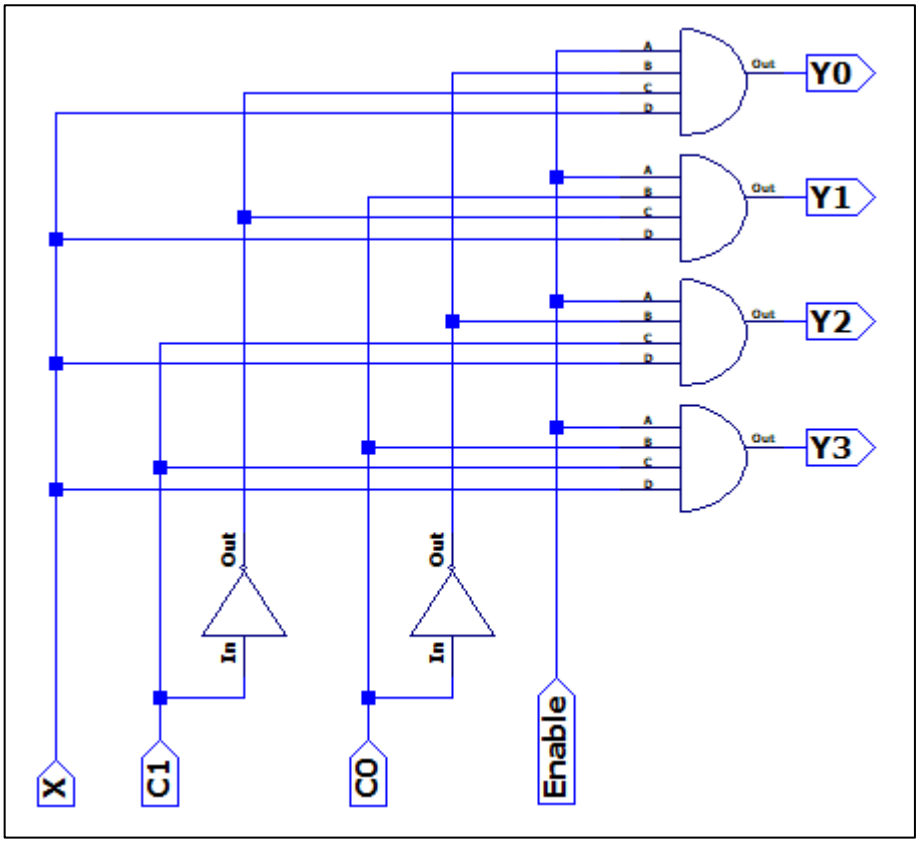


Fonte: Autor.

5.6 DEMULTIPLEXADOR

Usado para dividir um sinal em múltiplos canais ou rotas, um demultiplexador (DEMUX) é um dispositivo eletrônico que recebe um único sinal de entrada e o direciona para uma das várias saídas disponíveis, baseando-se em um sinal de controle de seleção. Na figura 5.21 é apresentado um exemplo de um DEMUX 1:4, o que significa que uma entrada é distribuída para quatro saídas distintas. Neste cenário, a entrada de dados é representada por X, enquanto C1 e C0 são os sinais de controle de seleção, com C1 sendo o bit mais significativo (MSB). O funcionamento do DEMUX é habilitado por um sinal de ativação chamado *Enable*.

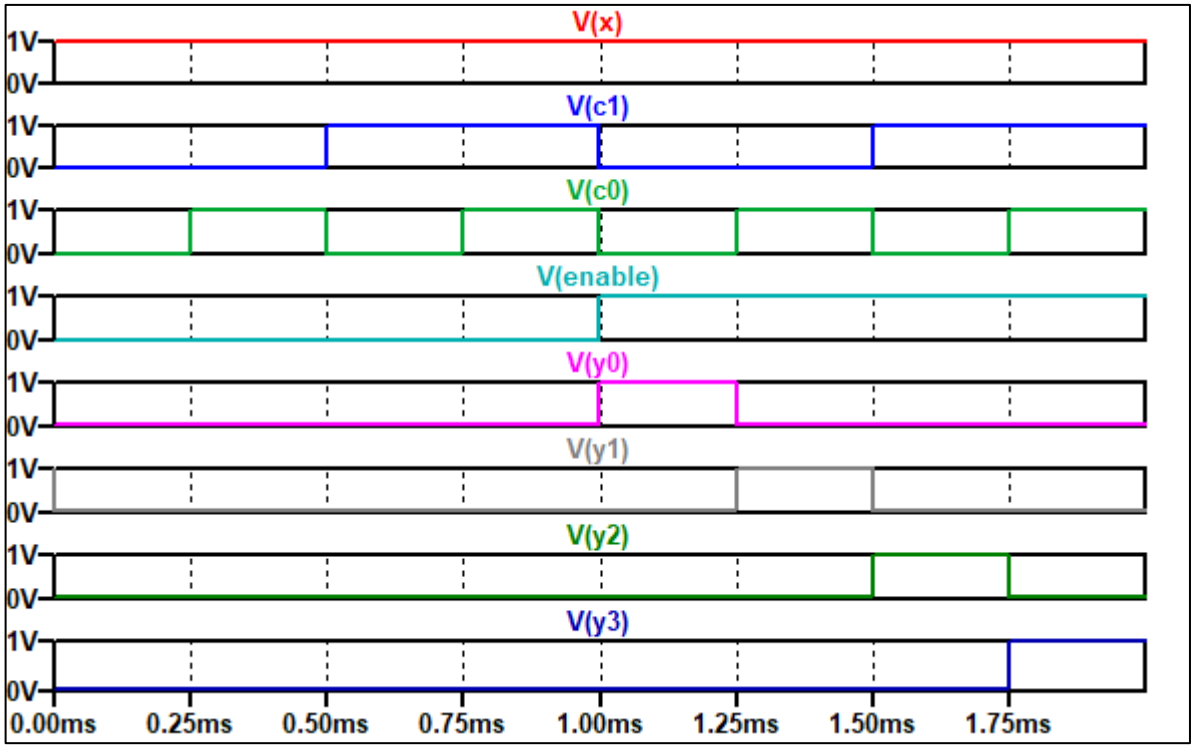
Figura 5.21 – SPICE DEMUX 1:4 memresistivo.



Fonte: Autor.

A figura 5.22, mostrada a seguir, evidencia o resultado de um teste simples realizado no DEMUX 1:4, exibindo todos os sinais de entrada e saída.

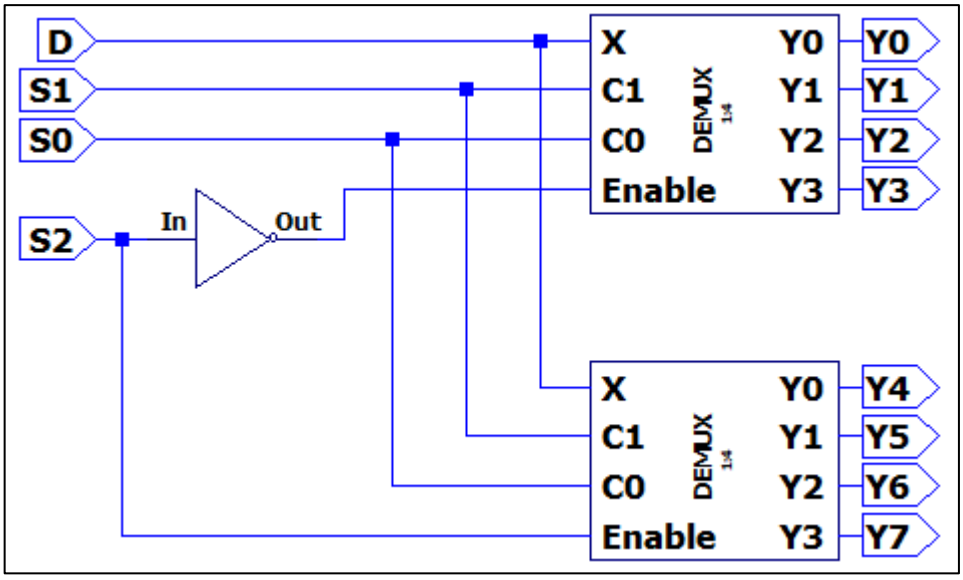
Figura 5.22 – Entradas e sápidas DEMUX 1:4 memresistivo.



Fonte: Autor.

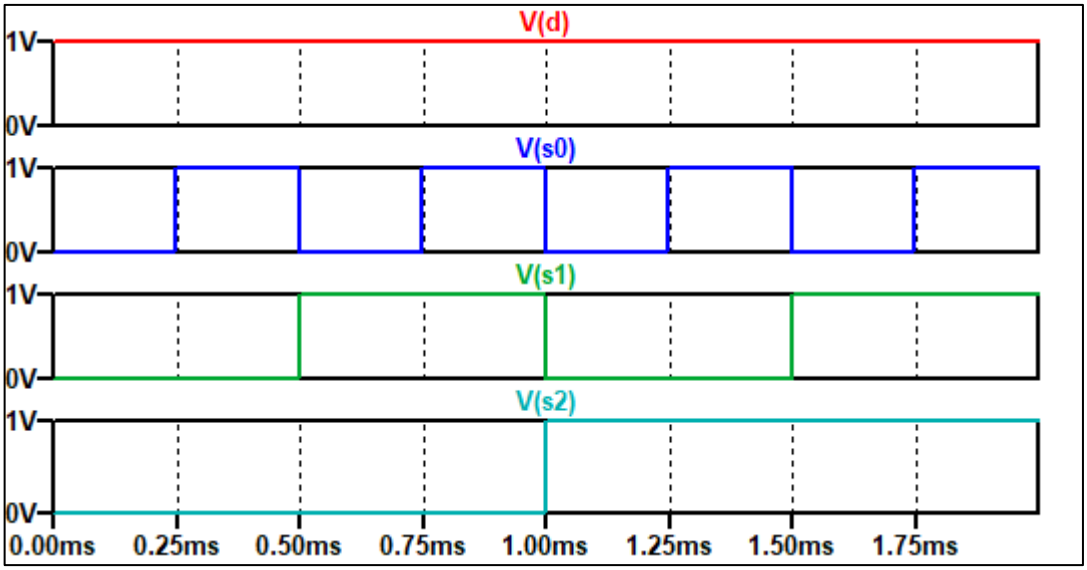
Demultiplexadores maiores podem ser obtidos cascateando-se dois ou mais DEMUX 1:4, como ilustrado na figura 5.23. Neste caso, a entrada é denotada por D e o controle de seleção é composto por S2, S1 e S0, onde S2 representa o bit mais significativo. Os resultados de um teste de funcionalidade realizado no DEMUX 1:8 são demonstrados nas figuras 5.24 e 5.25.

Figura 5.23 – DEMUX SPICE 1:8 memresisitivo.



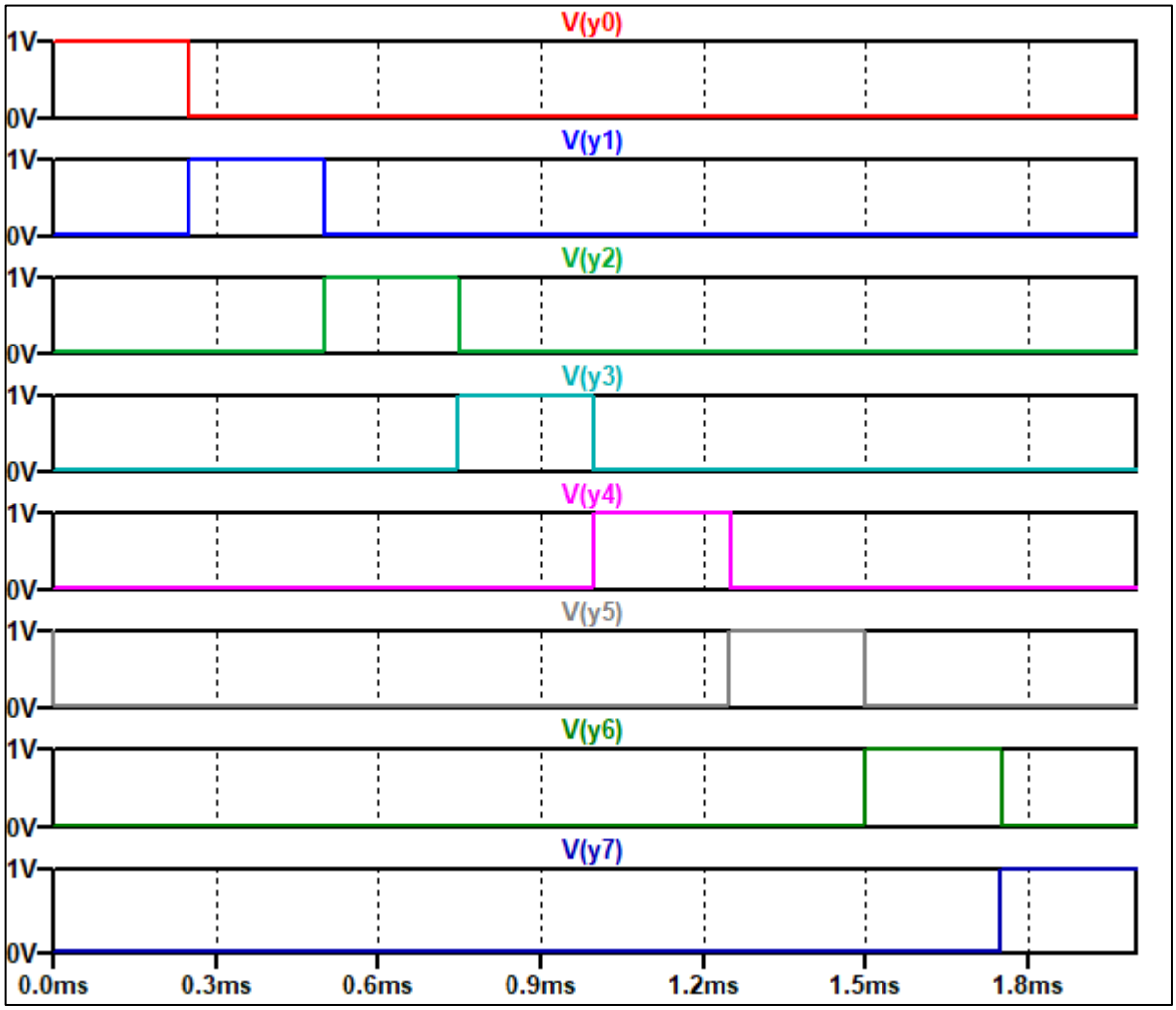
Fonte: Autor.

Figura 5.24 – Entradas DEMUX 1:8.



Fonte: Autor.

Figura 5.25 – Saídas DEMUX 1:8.



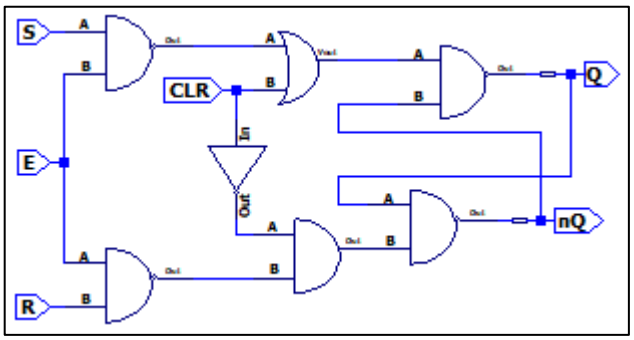
Fonte: Autor.

5.7 LATCH SR

O latch SR, implementado na figura 5.26, possui duas entradas, S (*SET*) e R (*RESET*), que controlam o valor da saída Q. Quando S é acionado, a saída Q se torna 1, quando R é acionado, a saída Q se torna 0. Se ambas as entradas S e R estiverem em 0, o estado imediatamente anterior em Q é mantido, tal operação recebe o nome de "hold". No entanto, quando S e R assumem ambos o valor 1, tanto Q quanto nQ tornam-se 1, o que é matematicamente inadmissível. Portanto, essa operação é proibida, uma vez que resulta em instabilidade no circuito.

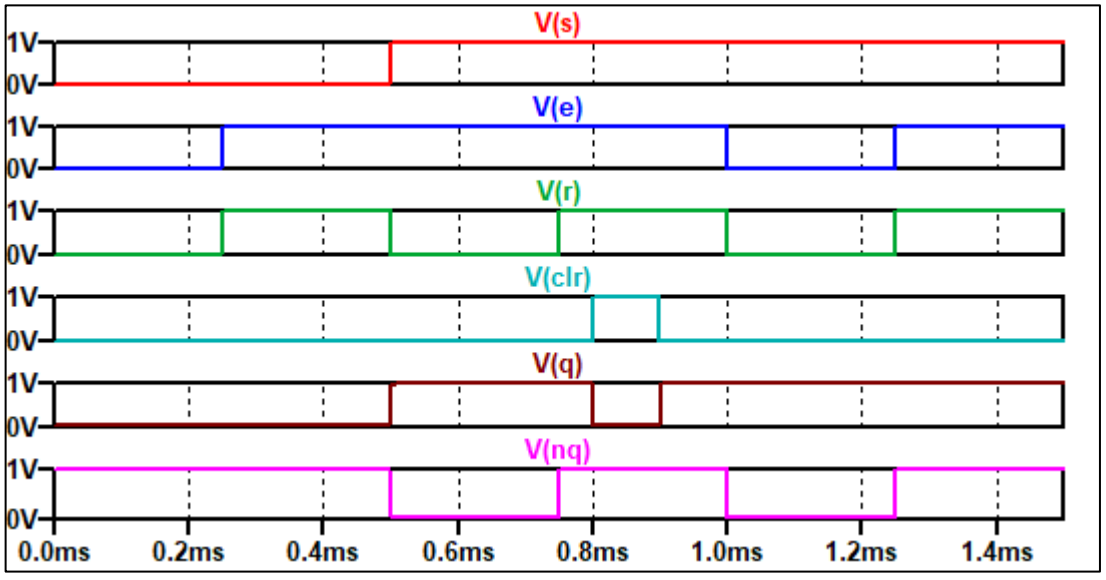
Para evitar a ativação simultânea de S e R, inclui-se uma terceira entrada chamada *enable*. Essa entrada permite que o latch funcione normalmente quando ativada e mantenha o estado atual quando desativada. Ademais, o comando *clear* permite "limpar" o estado momentâneo de Q e nQ, conforme ilustra o teste realizado exibido na figura 5.27.

Figura 5.26 – Latch SR memresistivo.



Fonte: Autor.

Figura 5.27 – Simulação Lacth SR memresistivo.

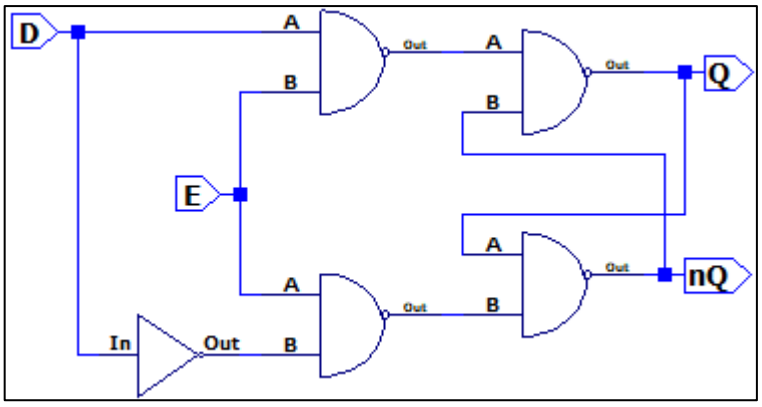


Fonte: Autor.

5.8 LATCH D COM BORDA DE SUBIDA

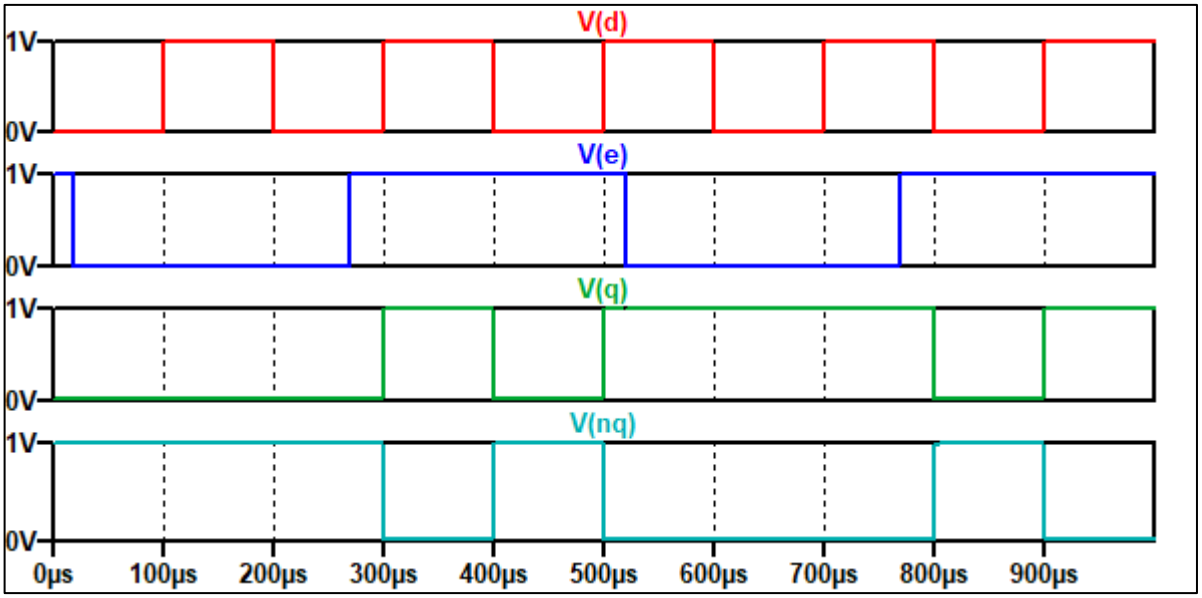
A figura 5.28 ilustra a estrutura do latch D. Este dispositivo é construído com base em um latch SR, onde as entradas *SET* e *RESET* são conectadas por meio de uma porta NOT controlada pelo sinal de *enable* (E). Enquanto o *enable* estiver em nível lógico 0, o latch manterá o último valor armazenado. No entanto, quando o *enable* for alterado para nível lógico 1, a saída refletirá diretamente o valor presente na entrada D, conferindo a esse dispositivo a característica de "latch transparente", tal como é possível visualizar no teste realizado, exibido na figura 5.29.

Figura 5.28 – Latch D memresistivo.



Fonte: Autor.

Figura 5.29 – Entradas e saídas Lach D.

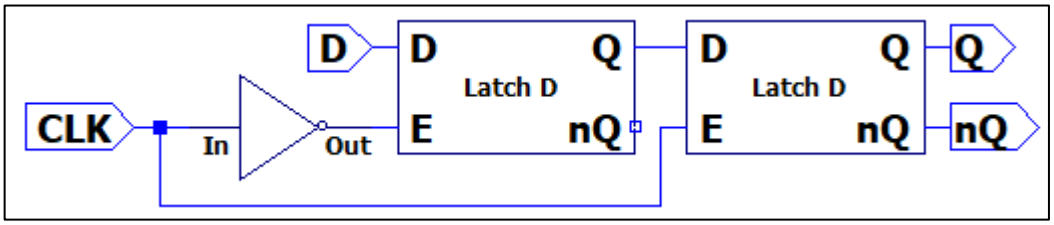


Fonte: Autor.

5.9 FLIP-FLOP D COM BORDA DE SUBIDA

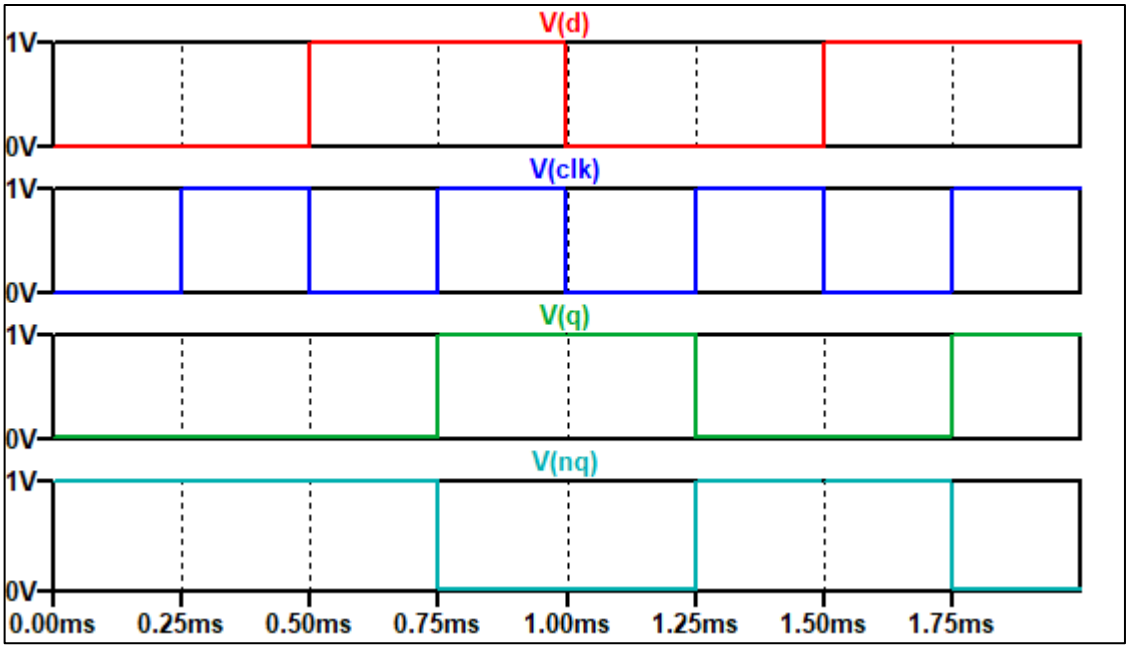
O flip-flop D concebido opera com base na transição de subida do sinal de *clock*. Ele é composto por dois latches D, sendo o primeiro denominado "mestre" e o segundo "escravo". Quando o sinal de *clock* está em nível baixo (0), o latch mestre replica a entrada. No entanto, quando o *clock* sobe para o nível alto (1), o mestre entra em estado de espera, e sua saída é então copiada para o latch escravo. Apesar do latch escravo estar pronto para aceitar mudanças em sua saída durante todo o período em que o *clock* permanece em nível alto, qualquer alteração só ocorre na próxima transição de subida do *clock*, uma vez que, durante esse intervalo, o mestre permanece em estado de espera. A figura 5.30 ilustra a estrutura do flip-flop D. A saída seguirá a entrada D sempre que ocorrer uma transição de subida no *clock*. Os resultados da simulação do flip-flop estão apresentados na figura 5.31.

Figura 5.30 – Flip-Flop D memresistivo



Fonte: Autor.

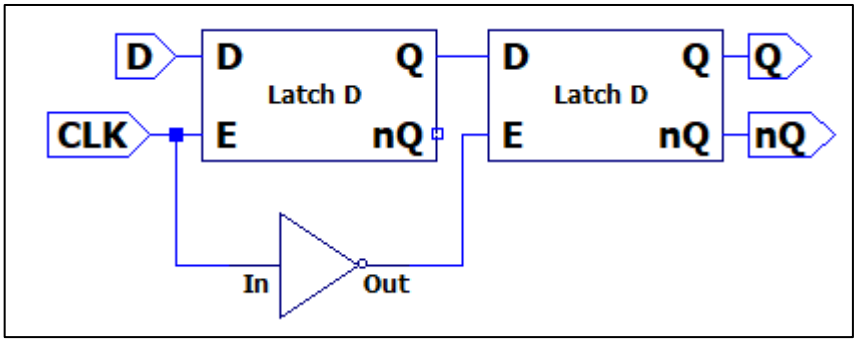
Figura 5.31 – Entradas e saídas Flip-Flop D.



Fonte: Autor.

Salienta-se ainda que é possível criar um flip-flop D que seja ativado somente na transição de descida do sinal de *clock*. Para isso, basta inverter o sinal de *clock* na entrada do latch escravo, mantendo-o não invertido no latch mestre, conforme mostrado na figura 5.32.

Figura 5.32 – Flip-Flop D ativado na descida.

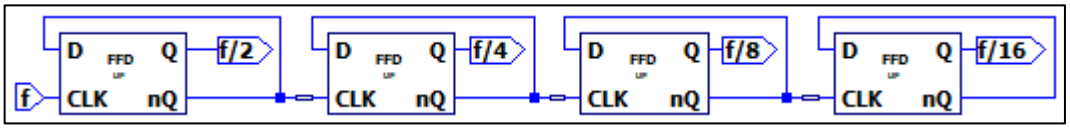


Fonte: Autor.

5.10 DIVISOR DE FREQUÊNCIA

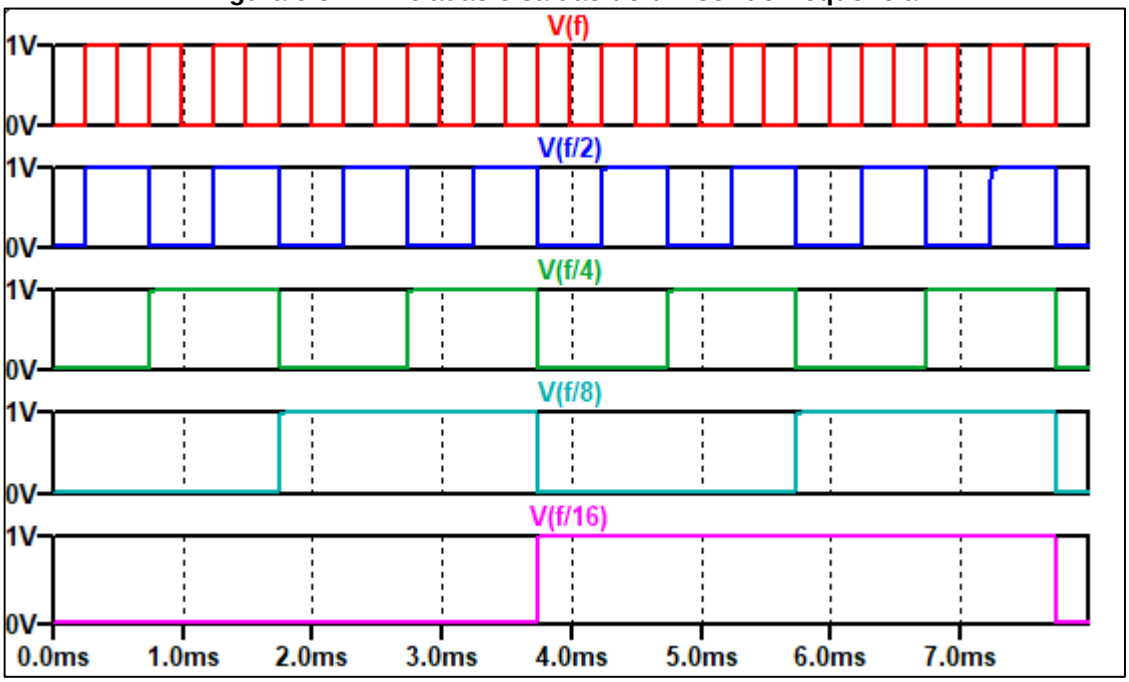
Ao fazer um loop de feedback conectando a saída nQ de volta à entrada D, em um flip-flop D, é possível multiplicar o período do sinal periódico que é adicionado à entrada D por um fator de 2. Sendo assim, torna-se facilmente cabível a criação de um circuito divisor de frequência utilizando flip-flops. Para obter frequências superiores a D*2, basta adicionar mais flip-flops em cascata. Um circuito que contenha um total de n flip-flops será capaz de produzir divisores de frequência que abrangem um intervalo de 2^1 até 2^n vezes a frequência original. As figuras 5.33 e 5.34 ilustram o esquemático de construção e o resultado do funcionamento, respectivamente, do divisor de frequência idealizado, onde f representa o sinal periódico a ser dividido.

Figura 5.33 – Divisor de frequência memresistivo.



Fonte: Autor.

Figura 5.34 – Entradas e saídas do divisor de frequência.

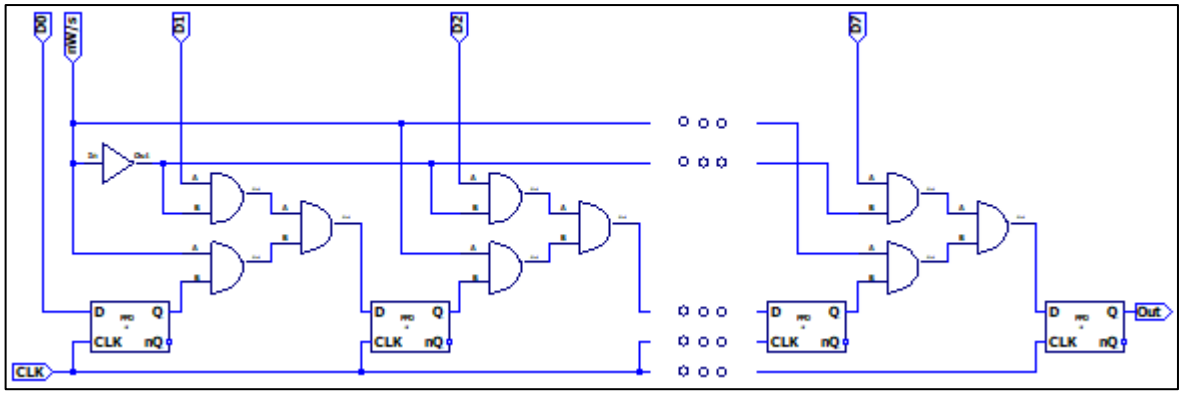


Fonte: Autor.

5.11 PISO

O registrador de deslocamento (PISO), é um dispositivo eletrônico capaz de converter uma entrada de bits paralelos em uma saída de formato serial. Ele é equipado com um sinal de entrada chamado nW/S, que controla o processo de carregamento dos bits em paralelo e a ativação do modo de registro de deslocamento para a conversão em série. Quando este sinal de entrada está em nível baixo (0), os valores dos bits de entrada (D0 a D7) são armazenados nos flip-flops D durante o momento em que o sinal de *clock* sobe. Agora, quando o sinal nW/S é ativado (1), os bits armazenados são deslocados uma posição a cada subida do *clock*.

Figura 5.35 – PISO memresistivo.



Fonte: Autor.

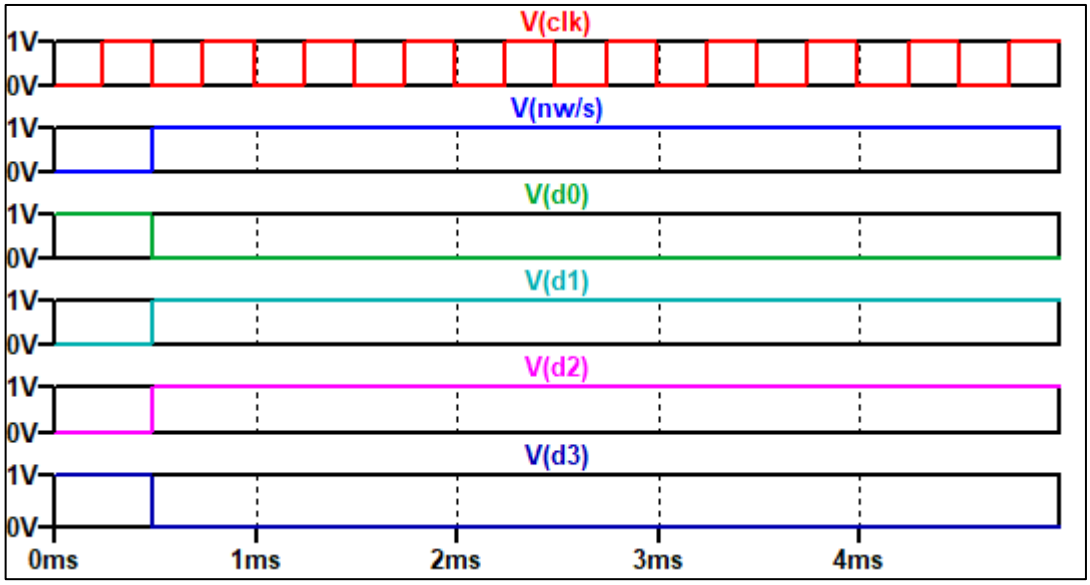
A figura 5.35 exibe o esquemático do PISO construído com a tecnologia memresistiva. Já na figura 5.36 é mostrado o teste de validação realizado no PISO, onde utilizou-se o conjunto binário da tabela 5.1 como entrada. Com o teste é possível comprovar que a saída segue o formado do conjunto binário de entrada serializado.

Tabela 5.1 – Conjunto binário de entrada do PISO.

D0	D1	D2	D3	D4	D5	D6	D7
1	0	0	1	1	1	0	1

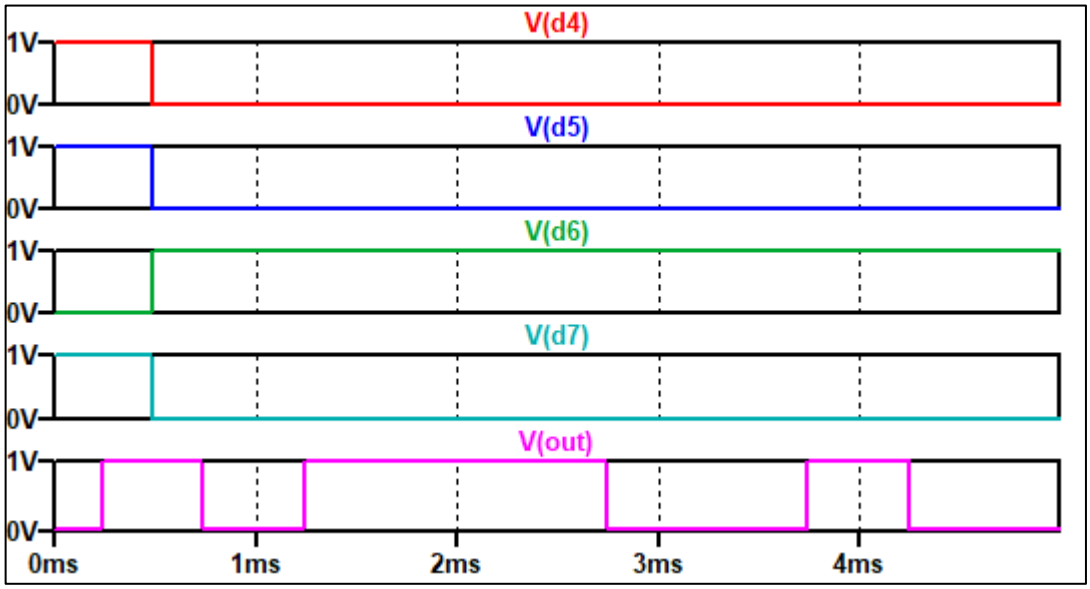
Fonte: Autor.

Figura 5.36 – Entradas PISO memresistivo.



Fonte: Autor.

Figura 5.37 – Entradas e saída PISO memresistivo.



Fonte: Autor.

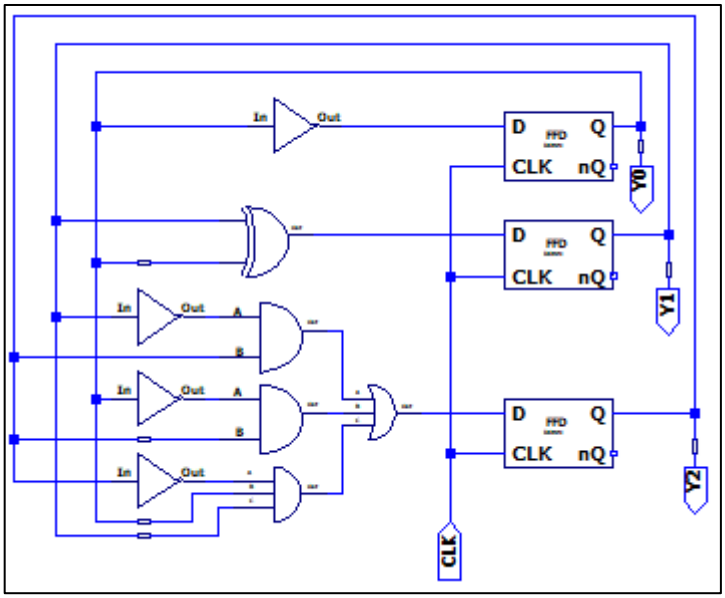
Salienta-se que apenas um ciclo de *clock* é usado para armazenar o conjunto de entrada e outros oitos ciclos para que este conjunto seja completamente convertido em formato serial na saída. Ou seja, a serialização completa consume nove ciclos de *clock*. Ademais, é importante observar que a ativação e desativação do sinal nW/S devem ocorrer antes de ocorrer uma subida no sinal de *clock*.

5.12 CONTADOR BINÁRIO

Um contador é um componente eletrônico que opera em resposta a um sinal de *clock* e tem a capacidade de contar sequencialmente. Em essência, ele segue um padrão específico de contagem que geralmente forma um ciclo, retornando ao seu estado inicial após atingir um valor máximo. Um contador binário de n-bits é construído usando n flip-flops e isso permite que ele tenha 2^n estados distintos possíveis.

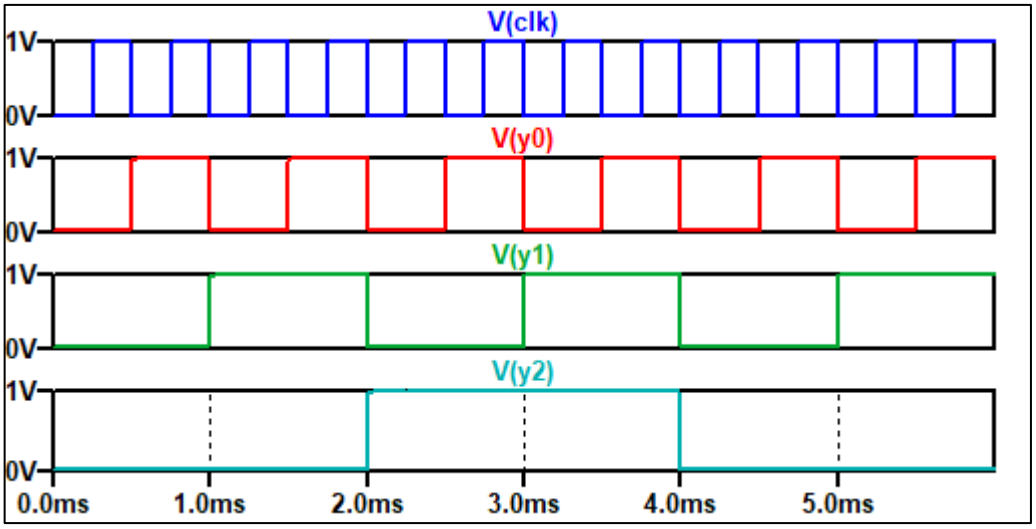
As figuras 5.38 e 5.39 apresentam o esquemático construtivo e o resultado do teste de validação, respectivamente, do contador binário de 3 bits idealizado, capaz de contar de 0 a 7. As saídas desse contador representam o valor atual em formato binário, com Y2 sendo o bit mais significativo (MSB). É importante notar que o contador é acionado na descida do sinal de *clock*.

Figura 5.38 – Contador memresistivo.



Fonte: Autor.

Figura 5.39 – Entrada e saídas do contador memresistivo.

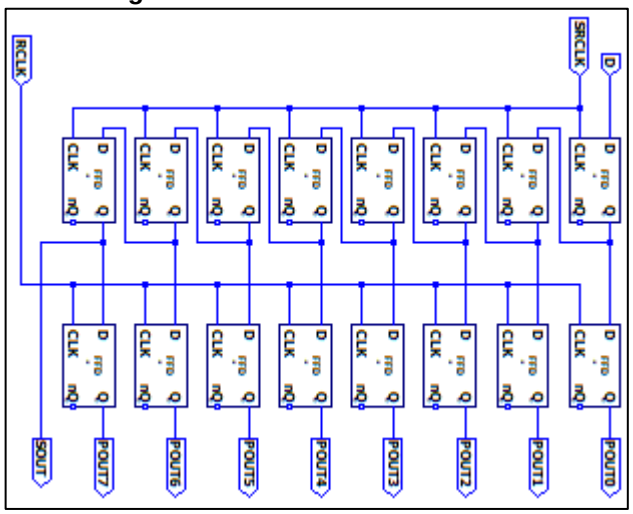


Fonte: Autor.

5.13 REGISTRADOR DE DESLOCAMENTO DE 8 BITS COM REGISTRADOR DE SAÍDA

O SRwOR, cujo esquemático de construção é evidenciado na figura 5.40, é um tipo de registrador de deslocamento com uma arquitetura específica que não é comumente descrita em livros convencionais. Ele consiste em um SIPO (*Serial-In*, *Parallel-Out*) de 8 bits à esquerda e um registrador de armazenamento de 8 bits à direita. Sua função é paralelizar uma palavra de 8 bits na entrada sem perda de bits após 8 períodos de *clock*. Cada registrador dentro do SRwOR possui sinais de *clock* independentes entre si, no entanto, é possível sincroniza-los utilizando um divisor de frequência, como um contador.

Figura 5.40 – Esquemático do registrador de deslocamento de 8 bits com registrador de saída.



Fonte: Autor.

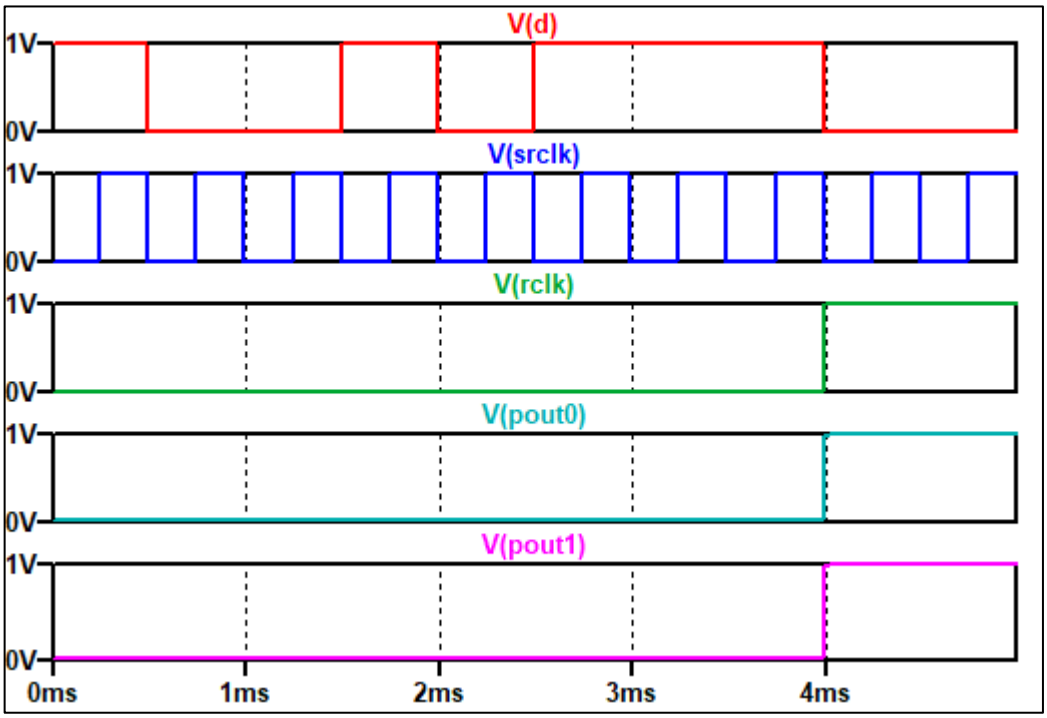
A saída paralela do SIPO alimenta os bits do registrador de armazenamento. O bit POUT7 é o MSB da entrada do SRwOR e os três primeiros MSB (POUT7, POUT6, POUT5) representam o endereço de destino. Para paralelizar uma palavra de 8 bits na entrada sem perda, são necessários 8 períodos do SRCLK. O RCLK deve estar sincronizado. Na simulação, evidenciada nas figuras 5.41 e 5.42, gera-se uma palavra de 8 bits (vide tabela 5.2) na entrada D_{IN}, e após 8 períodos do SRCLK, os bits estão nos registradores. Com 1 período do RCLK equivalente a 8 períodos do SRCLK, em 4ms de simulação os bits da palavra aparecem nas saídas do SRwOR.

Tabela 5.2 – Palavra de teste do SRwOR.

D0	D1	D2	D3	D4	D5	D6	D7
1	1	1	0	1	0	0	1

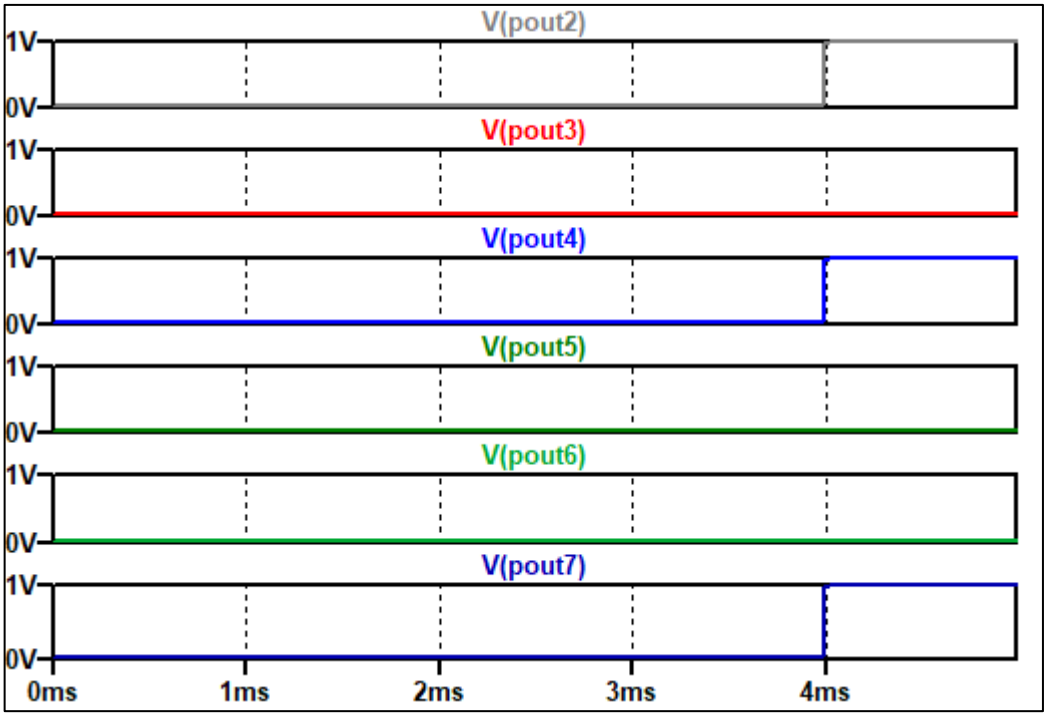
Fonte: Autor.

Figura 5.41 – Entradas e saídas do SRwOR de 8 bits.



Fonte: Autor.

Figura 5.42 – Saídas do SRwOR de 8 bits.



Fonte: Autor.

5.14 MEMÓRIA SRAM

Conhecida por sua rápida velocidade de acesso, permitindo a leitura e escrita de dados em ciclos de *clock* muito curtos, a SRAM (*Static Random Access Memory*) é um tipo de memória de acesso aleatório que armazena dados em circuitos integrados (MUSTAQUEEM et al, 2022). Ela é constituída por células que funcionam como unidades de armazenamento de um único bit de informação, sendo cada célula composta por um conjunto de transistores que formam uma espécie de latch capaz de armazenar um bit de dados. Essas células SRAM são organizadas em matrizes, com cada célula sendo identificada por um endereço específico composto por linhas de endereço e colunas de endereço. Quando um endereço é selecionado, a célula SRAM correspondente é ativada, permitindo a leitura ou escrita do valor armazenado.

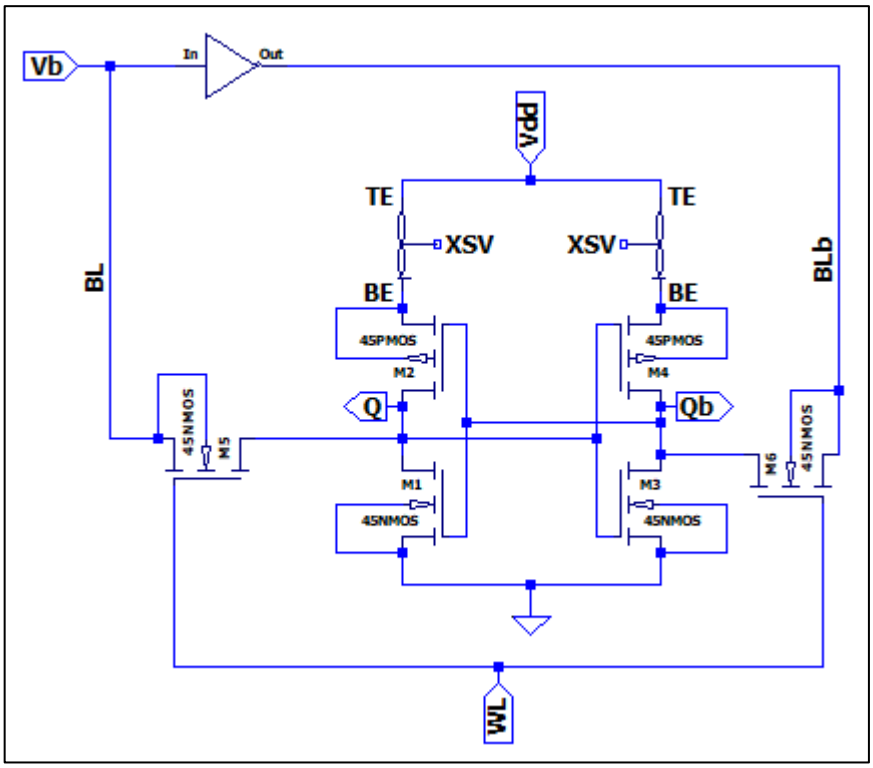
Apesar da célula tradicional de SRAM, composta unicamente por transistores, possuir uma metodologia estabelecida e comprovada, ela contempla um consumo de energia significativamente superior ao de outros tipos de memória, como a DRAM (Dynamic Random Access Memory). Em resposta a esta última indagação, Mustaqueem et al. (2022) apresentou uma inovadora célula SRAM 6T2M não volátil,

que incorpora dois *memristors* em conjunto com seis transistores para armazenar informações. Os *memristors* desempenham um papel fundamental, permitindo o armazenamento persistente de dados mesmo em situações de queda de energia, além de restaurar os dados previamente armazenados com sucesso.

De acordo com Mustaqueem et al. (2022), a principal vantagem de incorporar *memristors* na construção de células SRAM reside no fato de serem elementos não voláteis, ou seja, eles são capazes de manter informações ou dados anteriores mesmo quando a energia é desligada, sem necessidade de tensão ou corrente constante. Ademais, a célula SRAM 6T2M oferece uma notável estabilidade em comparação com a convencional célula SRAM 6T e exibe uma corrente de vazamento muito baixa.

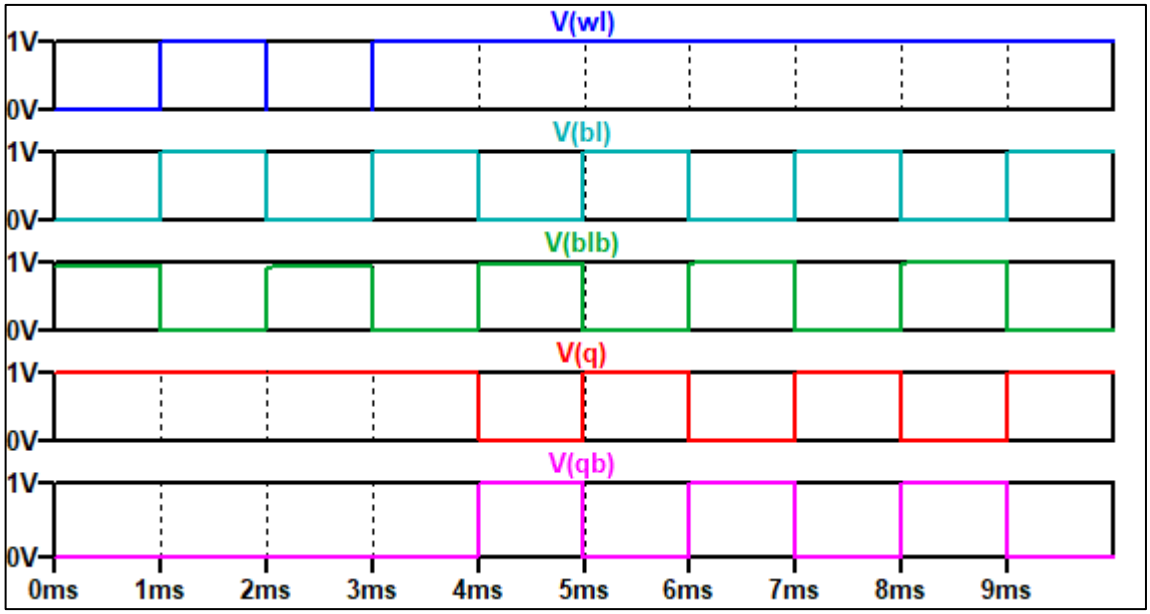
Uma adaptação do modelo proposto por Mustaqueem et al. (2022) é exibida na Figura 5.43, na qual evidencia-se os parâmetros de entrada (BL - *Bit Line*, WL - *Word Line*, BLb - *Bit Line* barrado), alimentação (V_{DD}) e a saída (Q e Qb):

Figura 5.43 – SRAM 6T2M.



Fonte: Autor.

Figura 5.44 – Funcionamento SRAM 6T2M.



Fonte: Autor.

Em resumo, o funcionamento da célula SRAM 6T2M, mostrada na figura 5.44, pode ser comparado a um "Latch" que armazena o valor de BL quando ocorre uma transição de subida em WL. Portanto, as saídas Q e Qb podem ser influenciadas pelas mudanças em BL e WL. Em uma simulação da SRAM, é comum que as saídas Q e Qb mantenham o último valor armazenado até que ocorra uma transição em WL. Assim, a alteração no valor das saídas só acontecerá quando WL realizar a transição, capturando o valor de BL nesse momento.

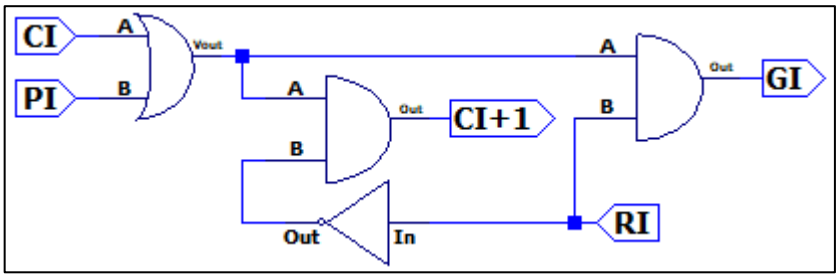
5.15 ÁRBITRO ROUND-ROBIN

Um árbitro RoR estrutura-se em três componentes distintos, sendo eles, um árbitro de nível subconsciente com uma prioridade iterativa que pode ser ajustada (OA), contando com entradas para os bits de prioridade; um gerador de prioridade estilo *Round-Robin*; e, por fim, um mecanismo de controle *Grant-Hold* responsável por manter a atribuição do *grant* por um tempo definido pelo comando de *hold*, evitando que o recurso seja alocado a outro agente durante esse intervalo (CÂMARA, 2017).

O esquemático do OA, mostrado na figura 5.45, aceita como entrada os bits RI, PI e CI e gera os bits CI+1 e GI como saída. Aqui é importante esclarecer que o índice ‘I’ em RI, PI e GI representa um índice no intervalo [0, n-1], onde *n* é o total de bits utilizados na aplicação. O bit RI sinaliza a solicitação de uso do recurso, enquanto o

bit CI indica que o recurso ainda não foi alocado para nenhuma solicitação de prioridade superior. Quando o bit PI é ativado, a solicitação correspondente recebe alta prioridade. Se tanto RI quanto CI estiverem ativados, e a respectiva célula possuir a maior prioridade, o pedido é concedido. Em seguida, o bit CI+1 é desativado, indicando que o recurso foi alocado e não está mais disponível.

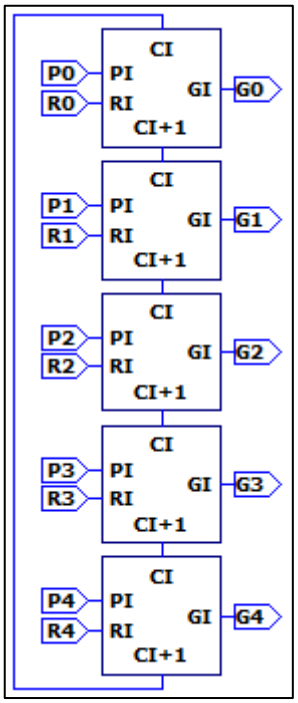
Figura 5.45 – Esquemático de 1 bit do árbitro inconsciente com prioridade variável iterativa.



Fonte: Autor.

Entendendo-se que a metodologia guia, ou seja, o roteador idealizado por CÂMARA (2017), possui 5 portas de entrada, o árbitro completo utilizando *memristors* deve ser composto por 5 bits, conforme indicado na figura 5.46:

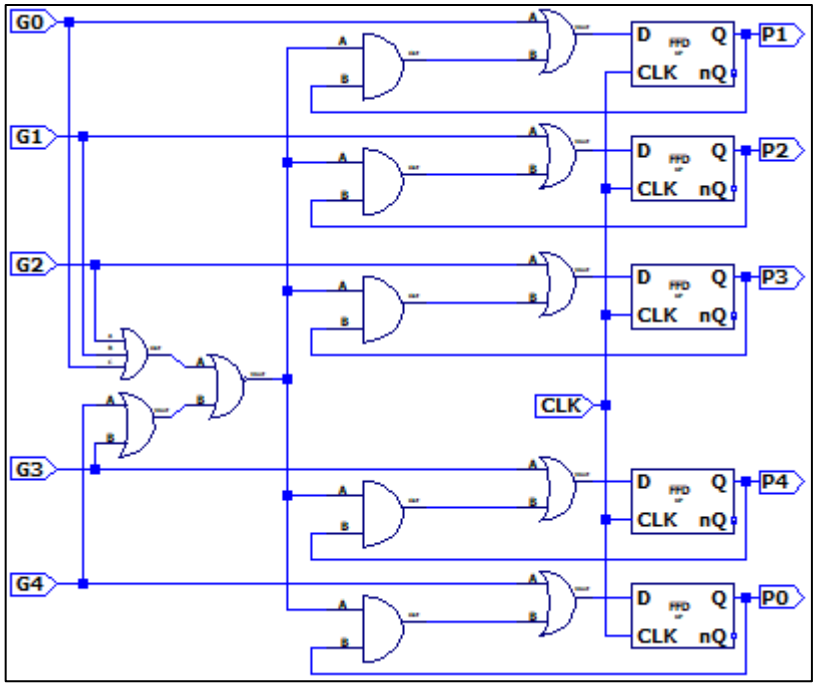
Figura 5.46 – Esquemático do árbitro inconsciente com prioridade variável iterativa.



Fonte: Autor.

O esquemático 5.47 ilustra o gerador de prioridade no estilo Round-Robin. Quando uma solicitação é atendida (representada por GI), o próximo bit de prioridade, PI+1, é elevado no ciclo subsequente. Isso resulta na promoção da solicitação mais próxima à que foi atendida, recebendo prioridade máxima, enquanto a solicitação atendida é movida para o final da fila.

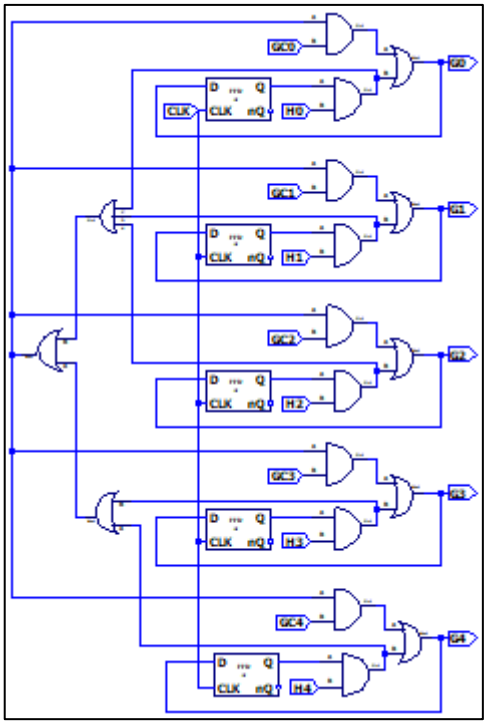
Figura 5.47 – Esquemático do gerador de prioridade Round-Robin.



Fonte: Autor.

Na figura 5.48 é apresentado um *Grant-Hold* de 5 bits que recebe *grants* gerados pelo OA (GCI) e o pedido de *hold* (HI), que determina o tempo durante o qual o *grant* atual deve ser mantido, bloqueando novas solicitações de *grant* durante esse intervalo.

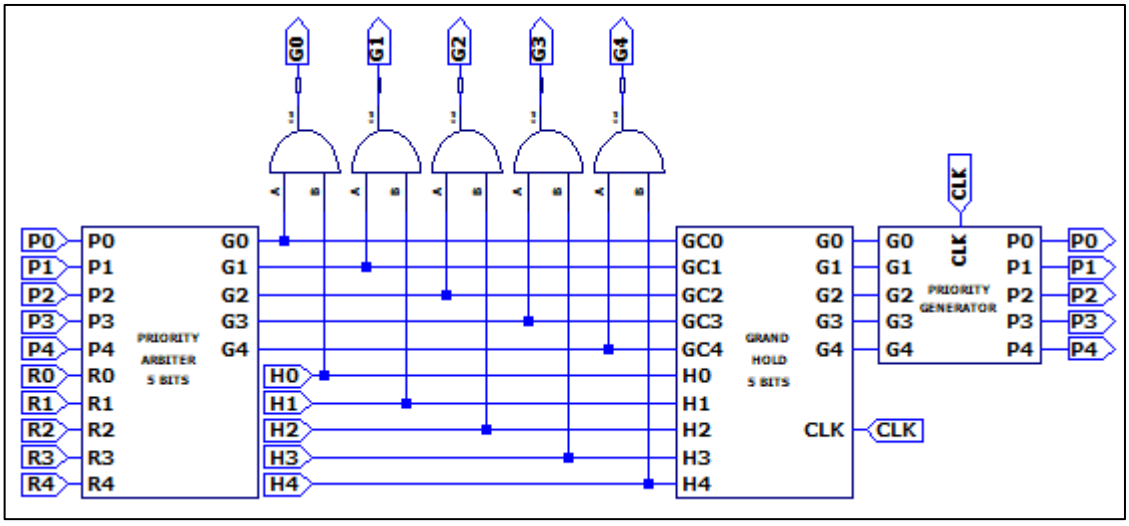
Figura 5.48 – Esquemático do circuito de *Grant-Hold*.



Fonte: Autor.

Na simulação do RoR completo (figura 5.49) três agentes (R1, R2 e R4) solicitam o uso do recurso. O agente R2 é o primeiro a fazer a solicitação e é imediatamente atendido. No entanto, no ciclo seguinte, R2 faz um novo pedido junto com os outros agentes, contudo, como seu pedido foi atendido no ciclo anterior, ele é movido para o final da fila e só será atendido novamente quando todos os outros agentes que solicitaram o recurso forem atendidos. A tabela 5.3 mostra a ordem das solicitações e o resultado esperado, apresentado na figura 5.50.

Figura 5.49 – Esquemático do árbitro Round-Robin.



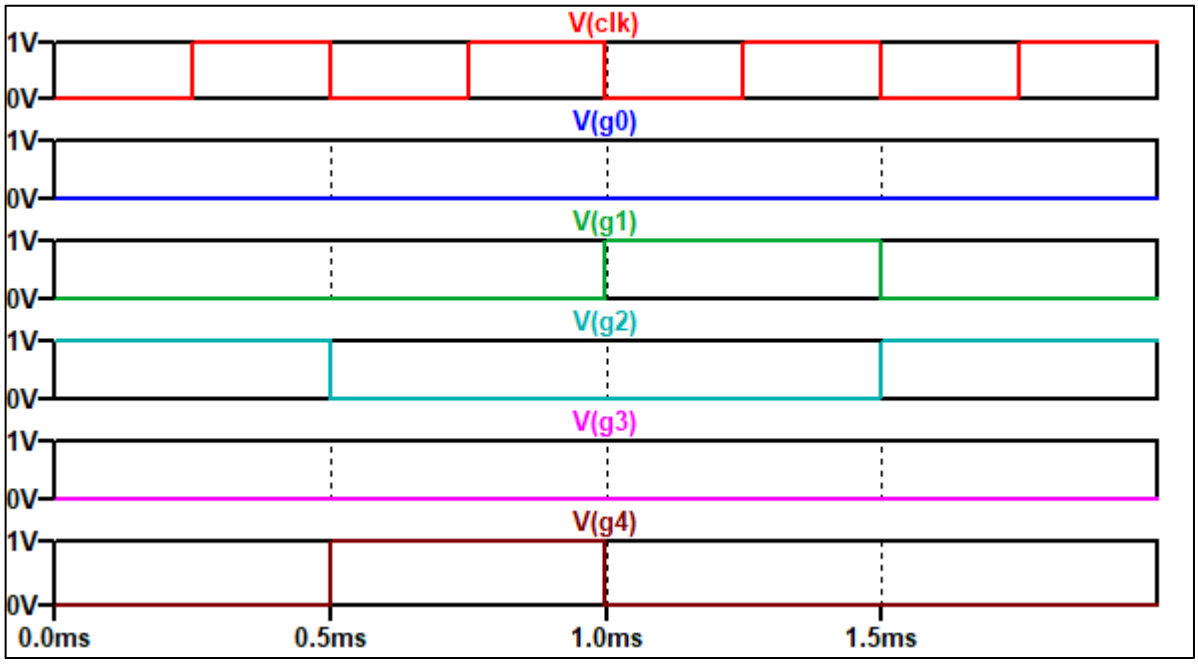
Fonte: Autor.

Tabela 5.3 – Pedidos e resultados do teste do RoR.

T(ms)	R1	R2	R4	G1	G2	G4
0 → 0.5	0	1	0	0	1	0
0.5 → 1	1	1	1	0	0	1
1 → 1.5	1	1	0	1	0	0
1.5 → 2	0	1	0	0	1	0

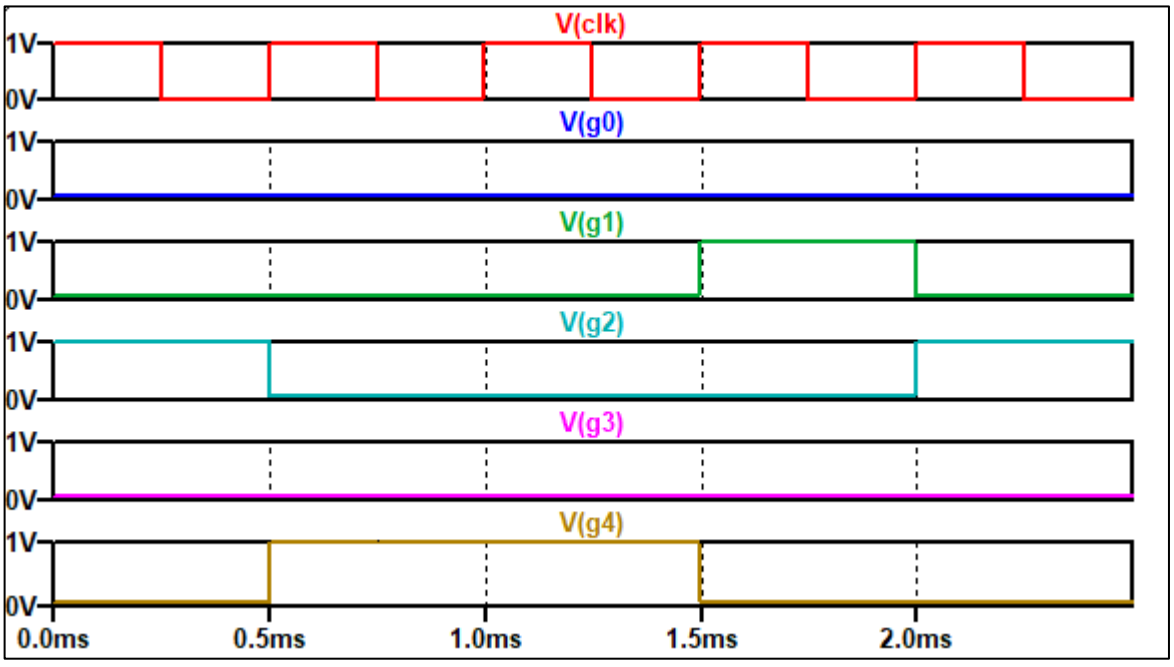
Fonte: Autor.

Figura 5.50 – 1ª simulação do árbitro Round-Robin.



Fonte: Autor.

Figura 5.51 – 2ª simulação do árbitro Round-Robin.



Fonte: Autor.

Durante a primeira simulação, cada sinal de *hold* foi mantido por apenas um período de *clock*. Se, em vez de manter o *hold* por apenas um período de *clock* em todas as solicitações, o R4 tiver seu *hold* mantido por dois períodos de *clock*, os pedidos subsequentes dos R1 e R2 sofrerão um atraso de um período em relação à simulação anterior, conforme ilustrado figura 5.51.

CAPÍTULO 6 – ROTEADOR:

Objetificando criar um roteador funcional para uma Rede em Chip utilizando a tecnologia memresistiva, uma investigação foi conduzida na literatura científica para avaliar arquiteturas de roteadores previamente implementadas em outras tecnologias, as quais serviram como base referencial. A partir dessas arquiteturas, os módulos digitais que compõem o roteador foram identificados e estudados de maneira detalhada.

A metodologia empregada para o projeto desses módulos foi hierárquica, tomando como base as portas NOT, NOR e NAND baseadas em memresistores. Para suportar esse processo, foi desenvolvida uma coleção de circuitos digitais fundamentais. Todos os circuitos básicos criados para a implementação do roteador memresistivos foram documentados na seção anterior. A arquitetura final será minuciosamente examinada ao decorrer do capítulo 6.

6.1 AVALIAÇÃO

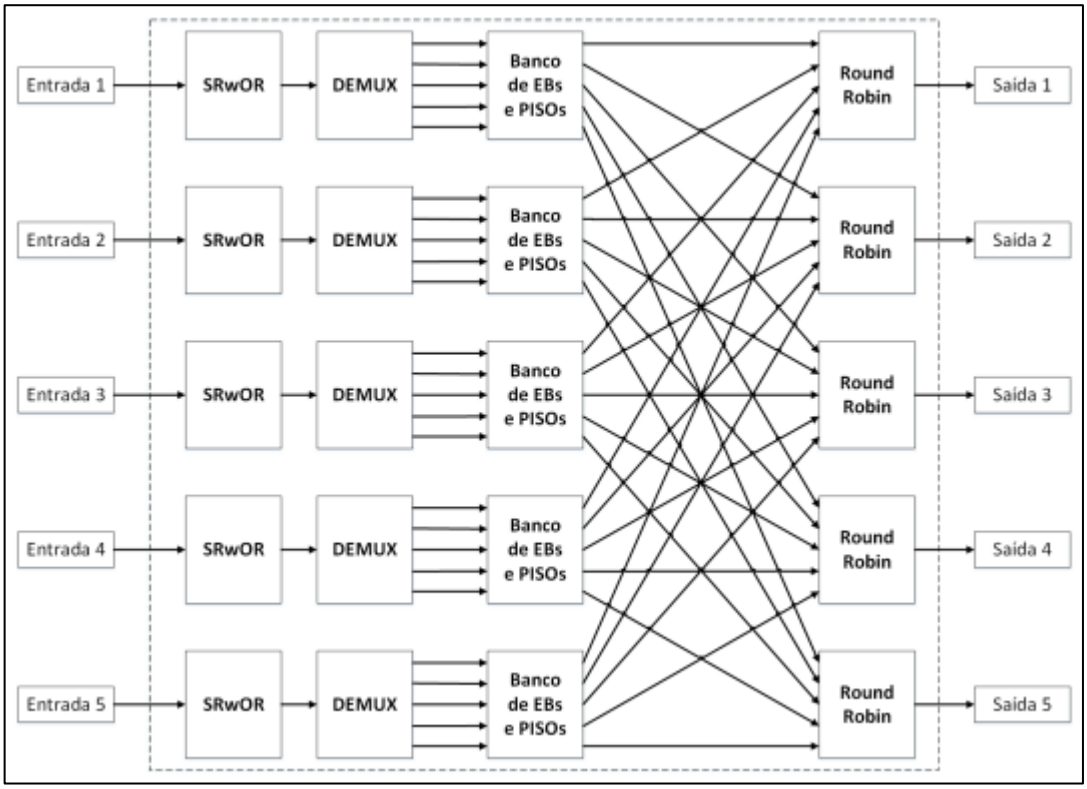
Durante o desenvolvimento da estrutura final do sistema, foram elaborados um total de 38 circuitos, abrangendo desde os mais simples até os mais complexos. Devido à progressão da arquitetura a partir de designs mais elementares, a ênfase não foi centrada primordialmente na capacidade de realização de testes. Para os módulos mais básicos, cujo funcionamento é completamente definido por tabelas verdade, a validação consistiu em testar combinações possíveis de entradas e comparar as saídas resultantes com os valores teóricos fornecidos nas respectivas tabelas de verdade. Para dispositivos mais complexos, os quais não possuem uma tabela verdade comprovadamente validada, foram aplicados testes específicos para assegurar que estavam operando em conformidade com o conceito de design estabelecido.

6.2 ANÁLISE

Inicialmente, pretendia-se construir o roteador memresistivo seguindo a metodologia elucidada por CÂMARA (2017), onde a autora desenvolve um roteador nanoeletrônico baseado em transistores monoelétron. Sendo assim, devido a

orientação para topologia Mesh, o roteador em questão foi idealizado contendo 5 entradas e 5 saídas. Um diagrama de blocos, presente na figura 6.1, serve como representação do circuito completo, onde cada entrada inclui um conjunto EB + PISO para cada uma das 5 saídas, ou seja, um “banco” contendo 5 EBs e 5 PISOs para cada saída. Isso indica que cada entrada é capaz de armazenar 8 bytes para uma mesma saída ou um total de 40 bytes.

Figura 6.1 - Esquemático do roteador completo.



Fonte: CÂMARA (2017).

Um pacote de dados consiste em uma palavra de 8 bits, dos quais os 3 bits mais significativos contêm o endereço de destino (CÂMARA, 2017). Quando um pacote chega em qualquer uma das entradas, ele passa pelo SRwOR para paralelização dos bits e extração do endereço, após isto o DEMUX seleciona a saída apropriada. O pacote é, então, encaminhado para o buffer correspondente, responsável pelo armazenamento temporário dos dados, seguindo o protocolo ready/valid. Quando armazenado no EB, ele envia uma solicitação de uso do recurso de saída ao árbitro RoR da saída correspondente. Após a aprovação da solicitação, o pacote é novamente serializado por meio do PISO de 8 bits e, finalmente, transmitido pelo enlace de saída.

Contudo, ao desenvolver o circuito completo, percebeu-se que o *memristor* não respondia muito bem a hierarquização profunda, de forma que, por se tratar de uma tecnologia recente (no quesito de bibliotecas virtuais), alguns *bugs* ocorriam com frequência, como a distorção e atenuação de sinais quando múltiplos circuitos eram conectados. A principal fonte de erro centrou-se no desenvolvimento de uma memória SRAM hierarquizada, equivalente a demonstrada por CÂMARA (2017), que não funcionava como deveria nos inúmeros testes realizados, isto é, as saídas e sinais intermediários presentes no circuito não retratavam o que deveria estar acontecendo de fato. Poderia argumentar-se que tal situação seria ocasionada pela baixa margem de ruído tratada com detalhes no capitulo 4. No entanto, como as lógicas de Registrador e Árbitro, que possuem um certo grau de complexidade e hierarquização, responderam bem, com a aplicação do filtro capacitivo, aos ensaios realizados, acredita-se que o erro esteja centrado na configuração do *memristor* ou na organização da estrutura SRAM idealizada. O circuito SRAM hierarquizado proposto é mostrado no apêndice A, acompanhado das lógicas de FIFO e Buffer Elástico, ambos sem resultados de simulação, visto que o não funcionamento da SRAM hierarquizada impossibilita os testes nestes circuitos.

Outra sugestão poderia ser a de desenvolver uma SRAM equivalente a hierarquizada, contudo, sem múltiplas camadas de circuitos lógicos. No entanto, tal concepção seria complexa ao ponto de necessitar, ao mínimo, mais seis meses de pesquisa.

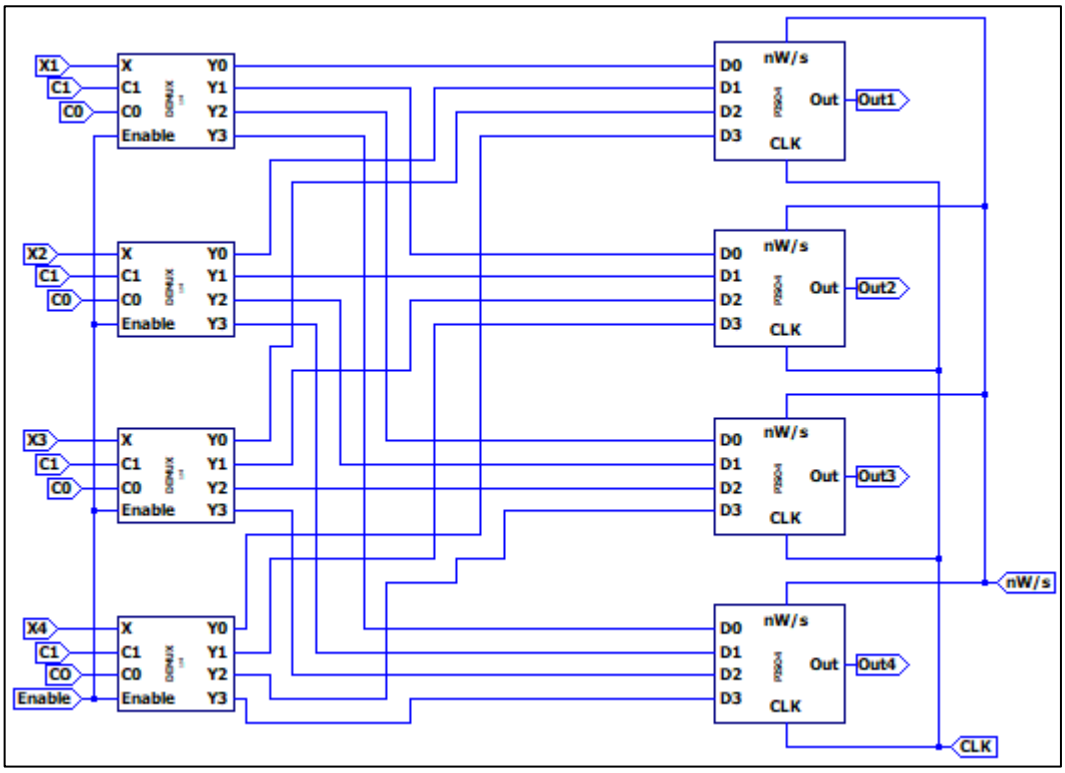
Contudo, afim de comprovar a possibilidade de utilizar-se *memristors* no desenvolvimento de roteadores, escolheu-se replicar duas arquiteturas com complexidade inferior à mostrada por CÂMARA (2017), concentrando-se exclusivamente na função básica de roteamento entre as entradas e saídas.

6.3 ROTEADOR 4x4

Um roteador genérico pode ser conceitualmente dividido em dois planos: o plano de controle, responsável pelos protocolos de roteamento e pela gestão da tabela de roteamento, e o plano de dados, que se encarrega do roteamento efetivo do tráfego de dados (Câmara et al, 2016). O roteador alternativo proposto se concentra no plano de dados, sendo um circuito totalmente memresistivo, construído com base na biblioteca de circuitos criada neste trabalho.

O roteador genérico segue a lógica evidenciada na figura 3.3 com três principais componentes: Logica de Entrada, Estrutura de Interconexão e Lógica de Saída. A Lógica de Entrada recebe dados e determina a saída com base na tabela de consulta. A Estrutura de Interconexão conecta as placas de entrada e saída. A Lógica de Saída armazena e transmite os dados. Cada logica de entrada possui um demultiplexador (demux) que direciona os bits de entrada de acordo com a tabela de consulta, controlada pelas linhas de seleção do demux. Enquanto que a lógica de saída, que inclui um conversor paralelo-serial (PISO), armazena todos os bits paralelos recebidos e os encaminha através da ligação de forma serial (Câmara et al, 2016). A figura 6.2 demonstra o circuito desenvolvido.

Figura 6.2 – Roteador memresistivo genérico.



Fonte: Autor.

6.4 TESTES MIMO E MISO

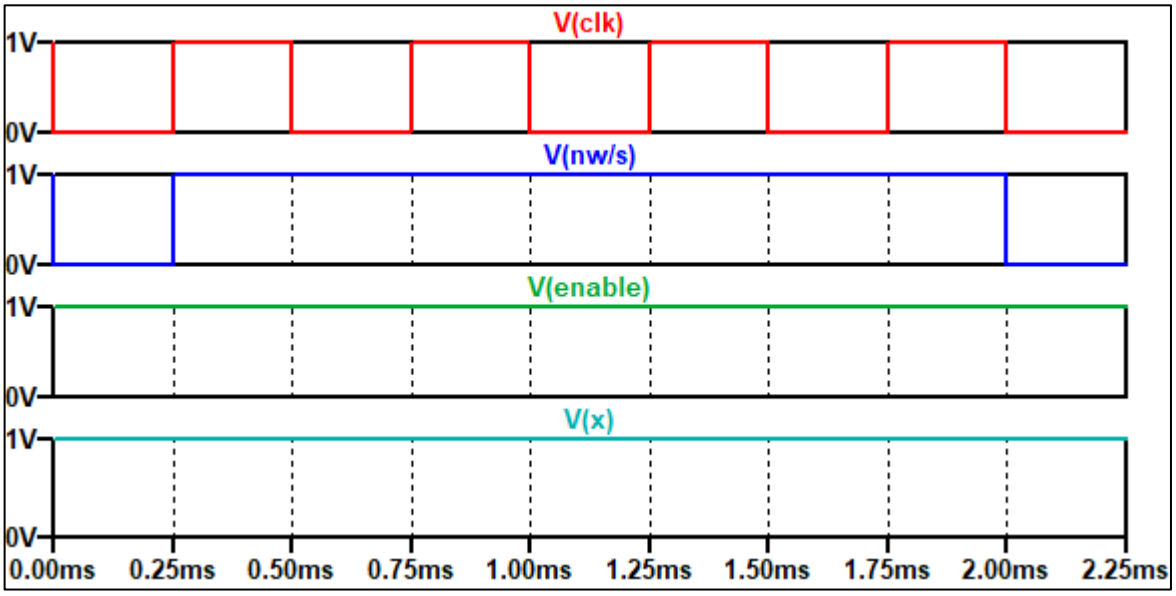
Com o objetivo de comprovar a eficácia do roteador, foram realizados dois testes. No primeiro teste (figura 6.3), múltiplas entradas de nível alto foram direcionadas para saídas de acordo com a tabela 6.1. A comparação entre o resultado obtido, conforme mostrado na figura 6.4, e a tabela 6.1 revela que os dados chegaram corretamente na saída selecionada pela lógica de consulta.

Tabela 6.1 – Tabela de consulta do roteador.

	D1	D2
Input 1	1	0
Input 2	0	0
Input 3	1	1
Input 4	0	1

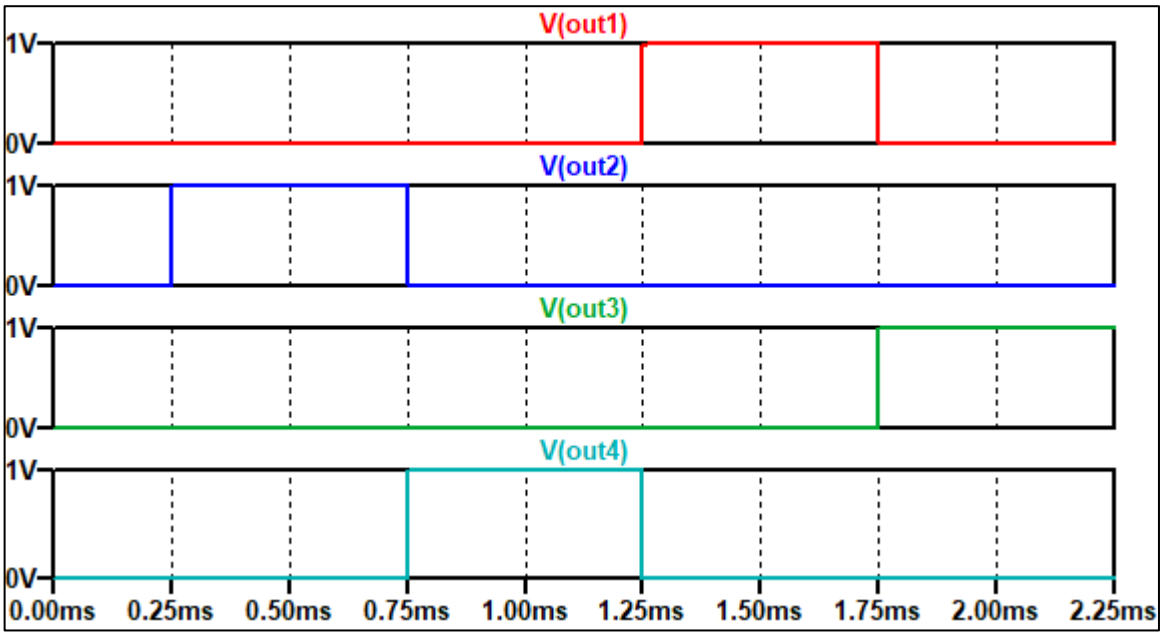
Fonte: Autor.

Figura 6.3 – Entradas do roteador durante o teste MIMO.



Fonte: Autor.

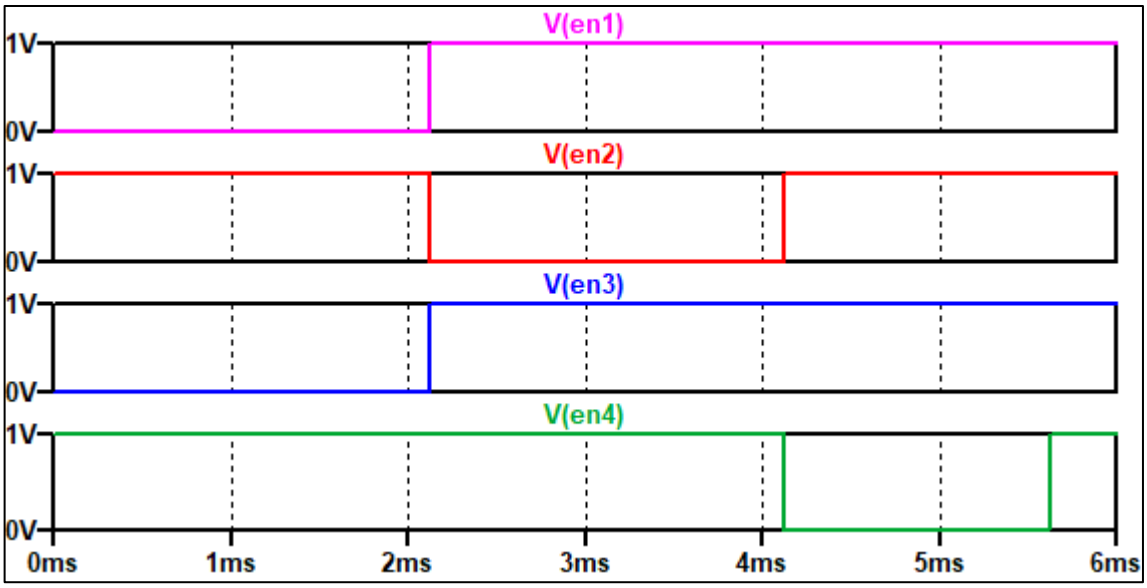
Figura 6.4 – Saídas do roteador durante o teste MIMO.



Fonte: Autor.

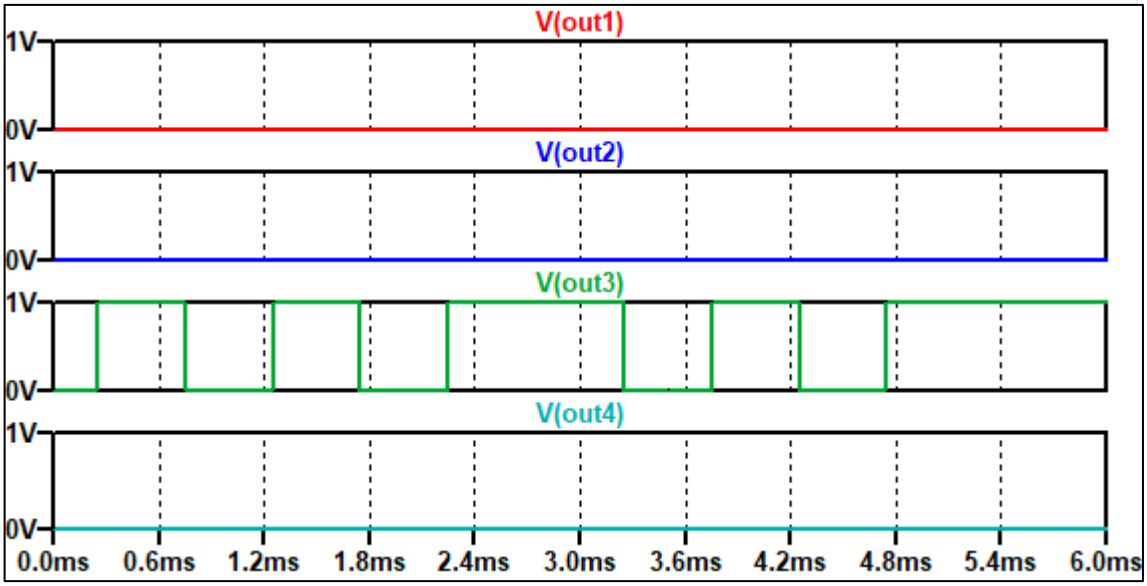
No segundo teste, todas as entradas foram direcionadas exclusivamente para a saída 3. A sequência de consulta permaneceu a mesma para todas as entradas, sendo a função *enable* responsável por organizar as saídas do Demux, como evidenciado na Figura 6.5. A Figura 6.6 mostra a sequência correta na saída 3, enquanto as demais saídas permanecem em 0, conforme o esperado.

Figura 6.5 – *Enables* durante o teste MISO.



Fonte: Autor.

Figura 6.6 – Saídas durante o teste MISO.

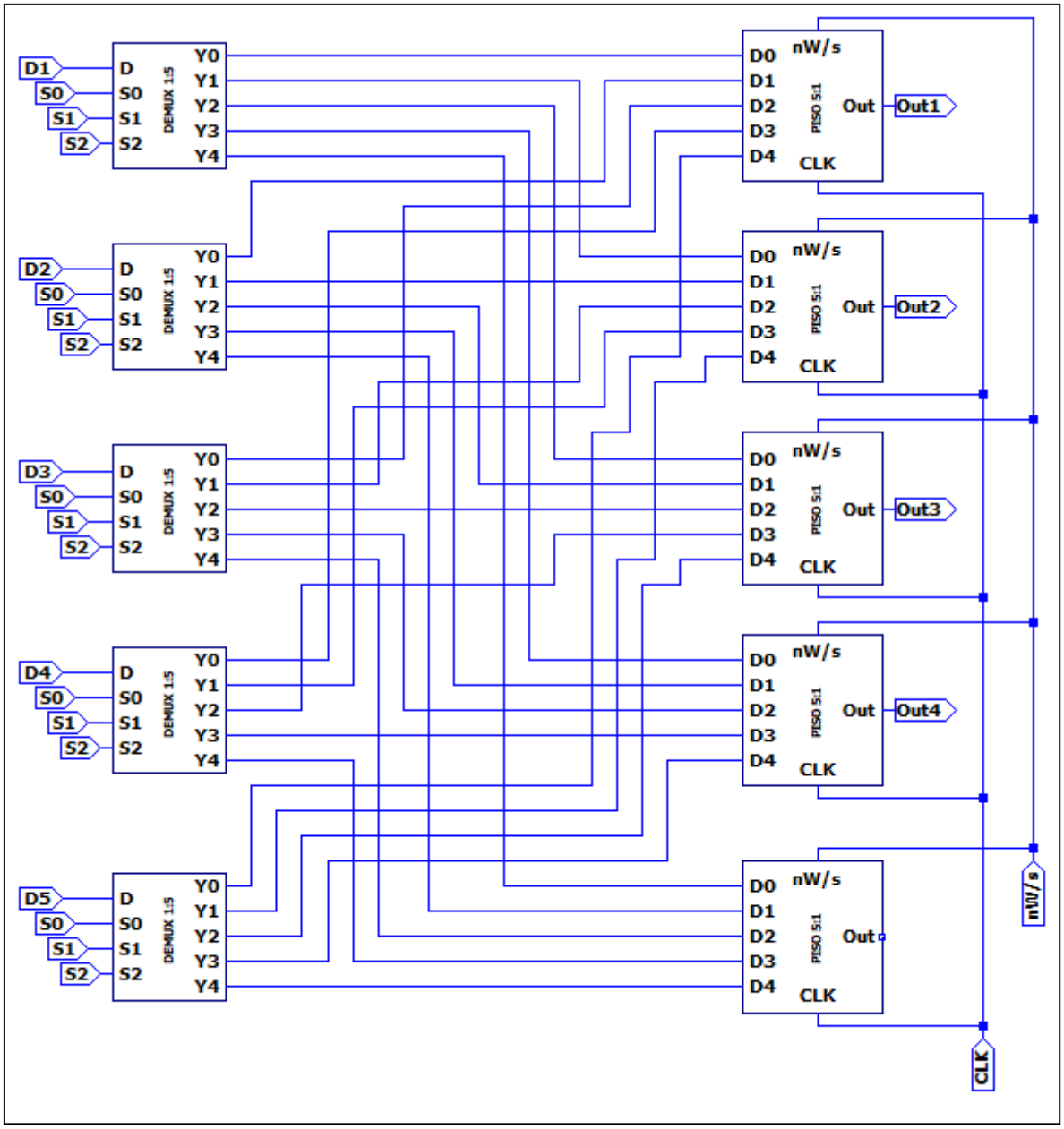


Fonte: Autor.

6.5 ROTEADOR PARA TOPOLOGIA MESH

A arquitetura genérica desenvolvida pode ser adaptada afim de operar na topologia Mesh, tal qual o objetivo inicial desse trabalho. Para isto, basta apenas adequar o roteador para possuir 5 entradas e 5 saídas. Logo, devido a biblioteca modular e hierárquica, a adaptação é realizada substituindo os componentes das linhas de entrada e de saída por DEMUX's 1:5 e PISO's 5:1. O resultado é exibido a seguir, na figura 6.7, onde o funcionamento dar-se-á de forma semelhante ao roteador 4x4 genérico.

Figura 6.7 – Roteador memresistivo 5x5 para topologia Mesh.



Fonte: Autor.

6.6 RESULTADOS

6.6.1 **Área e potência**

Para calcular a área do circuito utiliza-se o transistor como ponto de partida. A área de um transistor NMOS ou PMOS é o resultado da multiplicação do comprimento (W) pela largura (L) do canal. Em outras palavras, é o produto do comprimento pela profundidade do canal. De acordo com a abordagem inicialmente proposta por Farooq, Aslam e Usman (2018), a área ocupada pelo *memristor* pode ser estimada como aproximadamente metade da área do transistor MOS utilizado. Portanto, considerando que o circuito idealizado do roteador genérico possui um total de 320 portas NAND, cada uma contendo dois *memristors* e dois transistores (um NMOS e um PMOS com características W*L distintas), tem-se os valores mostrados na tabela 6.2:

Tabela 6.2 – Área e dissipação de potência do roteador genérico.

COMPONENTE	QUANT. NANDs	ÁREA	P. ESTÁTICA	P. DINÂMICA
MEMNAND	1	12150 nm^2	5,47 nW	6,15 µW
PISO	26	315900 nm^2	142,21 nW	160,10 µW
DEMUX	54	656100 nm^2	295,37 nW	332,53 µW
ROTEADOR	320	3888000 nm^2	1750,34 nW	1970,57 µW
SRAM[2]	1	52650 nm^2	5271,35 nW	1,07 µW

Fonte: Autor.

Potência estática é a potência constante consumida por um componente em estado estável, relacionada à corrente e tensão. Ela pôde ser visualizada utilizando a simulação do tipo *operational point* no LTSpice, que informa o valor em Watts dissipados. Somando-se o valor consumido em cada elemento de uma NAND, pode-se atribuir a potência estática consumida por essa porta lógica memresistiva. Então, realizando um procedimento semelhantes ao cálculo de área, ao relacionar a dissipação estática de uma porta logica NAND com a quantidade de NANDs presentes no sistema, calcula-se a potência estática dissipada pelo circuito roteador.

Tratando-se de componentes que mudam de estado durante o funcionamento do sistema, transistores e *memristors* terão também uma potência dinâmica, que se

[2] Embora o circuito de SRAM não tenha sido utilizado na concepção do roteador genérico, optou-se por também o analisar quanto a área e potência, visto que a construção de memórias é uma das principais áreas de estudo dos memristors. O cálculo de área e potência foi efetuado de maneira singular para cada elemento, portanto, a somatória dos valores resultou nos dados exibidos na tabela.

refere à potência variável durante mudanças de estado ou operação transitória. Para o cálculo desta grandeza, aplica-se a simulação do tipo transiente e utiliza-se a mesma metodologia enunciada para a potência estática, relacionando os valores consumidos por uma NAND com a quantidade de NANDs presentes no circuito. Os valores de energia dissipada em cada nível de hierarquização, utilizando 1V como nível logico alto, são exibidos no quadro 6.2. Um demonstrativo do cálculo de área e potência dissipada pode ser visualizado no Apêndice B.

6.6.2 **Comparativo entre tecnologias**

Ao realizar comparações entre tecnologias, é essencial garantir uma base justa para avaliação, o que geralmente envolve a análise de circuitos que sejam preferencialmente idênticos ou muito semelhantes em termos de construção lógica. No entanto, devido à escassez de referências que abordem o desenvolvimento de roteadores semelhantes com análises de área e potência, optou-se por utilizar a lógica NAND como foco de análise. Vale ressaltar que, como mencionado anteriormente, é possível implementar o circuito de um roteador genérico em sua totalidade utilizando somente essa porta lógica, logo, o comparativo pode ser expandido para todos os circuitos idealizados neste trabalho.

Para realizar uma comparação abrangente, escolheu-se duas tecnologias de ponta, QCA e SET, que apresentam características distintas. A tecnologia QCA, ou *Quantum-dot Cellular Automata*, baseia-se na organização de células que utilizam o estado de polarização para transferir informações. Essa abordagem promete altas frequências de *clock* da ordem de THz e baixo consumo de energia, tornando-a uma alternativa promissora aos circuitos tradicionais (SARDINHA et al, 2013).

Por outro lado, a tecnologia SET, ou *Single-Electron Transistor*, utiliza junções de túnel para criar ilhas e controlar o fluxo de elétrons individuais, resultando em um consumo de energia muito baixo. O controle é alcançado por meio da manipulação do nível de Fermi por meio da tensão de porta. Outrossim, os SETs podem operar em temperaturas mais elevadas, até 350 K (CÂMARA et al, 2016).

Além dessas tecnologias avançadas, a comparação também inclui o método convencional que utiliza transistores MOS de 22nm. Todos esses métodos já estão sendo empregados na eletrônica atual e servem como ponto de partida na avaliação do desempenho dos circuitos lógicos construídos com *memristors*.

A tabela 6.3 exibe os dados obtidos de Kavitha & Kaulgud (2017), Jaiswal & Sasamal (2017), CÂMARA (2016) e Zaidi (2013) referentes a área e potência dissipada das portas NAND construídas nas tecnologias mencionadas, todas equiparadas para um V_{DD} = 0,9V.

Tabela 6.3 – Comparativo entre tecnologias.

TECNOLOGIA	*MEMRISTOR*	QCA	SET	CMOS 22nm
ÁREA [nm²]	12150	100210	172	14800
POTÊNCIA [µW]	4,79	0,01554	0,00022	0,47

Fonte: Autor.

Embora os dispositivos deste trabalho tenham usado transistores de 45nm, ao fazer uma análise crítica da tecnologia memristiva, nota-se que ela representa um avanço notável em termos de economia de espaço, mesmo quando comparada aos circuitos construídos com transistores de 22nm. No entanto, a dissipação de energia é uma desvantagem, seja em comparação com tecnologias QCA, SET ou transistores de 22nm. Isso é razoável, pois o *memristor* possui certas características do resistor padrão, dissipando energia na forma de calor, como uma resistência convencional.

Sendo uma tecnologia relativamente recente, ainda há espaço para avanços e descobertas adicionais. Atualmente, tecnologias mais consolidadas academicamente, como QCA e SET, oferecem vantagens significativas em termos de eficiência de área e consumo de energia. No entanto, o *memristor* se destaca por sua capacidade de armazenar dados mesmo quando desligado, devido à memorização de sua resistência após a última carga passar por ele.

Apesar do *memristor* não ter respondido bem a uma hierarquização profunda, como na concepção de um roteador mais robusto, acredita-se que com o desenvolvimento de novos estudos e bibliotecas sobre os resistores de memória isto poderá ser contornado. Mesmo assim, comprova-se a possibilidade de sua utilização na criação de circuitos lógicos amplamente utilizados na indústria, como flip-flops, contadores e portas lógicas. No entanto, seu principal campo de atuação deve-se centrar na construção de memórias, como proposto por Mustaqueem (2022). O trabalho desse autor investigou com sucesso uma célula SRAM 6T2M, reconstruída

[3] Entre os vários artigos consultados sobre a tecnologia QCA, nenhum apresentou uma análise de potência dissipada para a porta NAND, no entanto, Kavitha & Kaulgud (2017) e Jaiswal & Sasamal (2017) expõem uma análise de energia dissipada. Com isso, é possível encontrar a potência dissipada relacionando a energia gasta, dada em Joules. com o tempo de simulação e a quantidade de células QCA empregadas numa porta NAND.

neste documento, na região sublimiar com uma tensão de alimentação de 0,3V. Essa configuração superou a célula SRAM tradicional 6T, melhorando eficiência energética, estabilidade e velocidade do circuito em baixa tensão. A célula 6T2M apresentou um aumento significativo na potência de leitura e escrita, com melhorias de 40% e 90%, respectivamente, em comparação com a célula convencional. Isso resultou em menor corrente de vazamento e melhor margem de ruído estático de leitura (RSNM).

Em resumo, a célula SRAM baseada em *memristor* mostra grande promessa para aplicações em circuitos de memória em comparação com as SRAM tradicionais. No entanto, ainda requer mais estudos e pesquisa para equiparar-se às tecnologias QCA e SET em termos de eficiência energética e ocupação de área em circuitos lógicos tradicionais.

CAPÍTULO 7 – CONCLUSÃO

Este estudo introduziu um projeto de roteador voltado para sistemas bioinspirados, utilizando *memristors* como componente central. O objetivo primordial foi explorar as propriedades dos *memristors* e avaliar sua aplicabilidade em circuitos digitais, particularmente em roteadores e memórias. Os *memristors*, dispositivos concebidos por Leon Chua em 1971, destacam-se pela capacidade singular de ajustar sua resistência com base no fluxo de carga mais recente que atravessa seus terminais.

Neste projeto, os *memristors* foram empregados para armazenar informações e realizar operações lógicas, contribuindo para o desenvolvimento de uma biblioteca de circuitos memresistiva. Embora os resultados iniciais tenham revelado um desempenho inferior, em termos de ocupação de área e dissipação de energia, quando comparados a tecnologias mais recentes, como QCA e SET, ressalta-se que o uso de *memristors* na criação de memórias SRAM demonstrou eficácia e vantagens significativas em relação à abordagens baseadas apenas em transistores.

Portanto, apesar das áreas de aprimoramento identificadas, este estudo demonstrou claramente o potencial dos *memristors* para aprimorar a eficiência dos circuitos de memória e a sua viabilidade na concepção de circuitos digitais mais amplos, desde que não haja uma hierarquização significativamente pesada. A pesquisa realizada neste trabalho espera contribuir para o avanço da tecnologia de circuitos digitais e sistemas bioinspirados, inspirando novas pesquisas e desenvolvimentos, afim de auxiliar a difundir a aplicabilidade do *memristor*.

REFERÊNCIAS

CHUA, L. **Memristor-The missing circuit element**. IEEE Transactions on Circuit Theory, v. 18, n. 5, p. 507-519, 1971. ISSN 0018-9324

ASCOLI, A. et al. **Memristor Model Comparison**. IEEE Circuits and Systems Magazine, v. 13, n. 2, p. 89-105, 2013. ISSN 1531-636X

Câmara, Beatriz Oliveira. **ROTEADOR NANOELETRÔNICO PARA REDES-EM-CHIP BASEADO EM TRANSISTORES MONOELÉTRON**. 2017. 89 f. Dissertação (Mestrado) - Curso de Engenharia de Sistemas Eletrônicos e Automação, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, 2017.

BERRIOS, Lesly Viviane Montúfar; COUTO, Cibelly Cristina Rodrigues; CURY, Yasmin Delbany; REZENDE, Paulo Henrique Oliveira. **FUNDAMENTOS FÍSICO-QUÍMICOS E MATEMÁTICOS DE UM MEMRISTOR.** In: CEEL - ISSN, 2596-2221., 2019, Uberlândia. UFU - Universidade Federal de Uberlândia, 2019. p. Disponível em: https://www.peteletricaufu.com.br/static/ceel/artigos/artigo_482.pdf. Acesso em: 23 ago. 2023.

GARC, F. et al. **Building Memristor Applications:** From Device Model to Circuit Design. IEEE Transactions on Nanotechnology, v. 13, n. 6, p. 1154-1162, 2014. ISSN 1536-125X

Liu, G., Shen, S., Jin, P. et al. **Design of Memristor-Based Combinational Logic Circuits**. Circuits Syst Signal Process 40, 5825–5846 (2021). https://doi.org/10.1007/s00034-021-01770-1

KVATINSKY, S. et al. **TEAM: Threshold Adaptive Memristor Model**. IEEE Transactions on Circuits and Systems I: Regular Papers, v. 60, n. 1, p. 211-221, 2013b. ISSN 1549- 8328.

PICKETT, M. D. et al. **Switching dynamics in titanium dioxide memristive devices**. Journal of Applied Physics, v. 106 n. 7, p. 074508, 2009. ISSN 0021-8979.

LEHTONEN, E.; POIKONEN, J.; LAIHO, M. **Implication logic synthesis methods for memristors**. Circuits and Systems (ISCAS), 2012 IEEE International Symposium on, 2012 IEEE. p.2441-2444.

PERSHIN, Y. V.; DI VENTRA, M. **Practical approach to programmable analog circuits with memristors**. IEEE Transactions on Circuits and Systems I: Regular Papers, v. 57, n. 8, p. 1857-1864, 2010. ISSN 1549-8328.

PRODROMAKIS, T. et al. **A versatile memristor model with nonlinear dopant kinetics**. IEEE transactions on electron devices, v. 58, n. 9, p. 3099-3105, 2011. ISSN 0018-9383.

BIOLEK, D.; BIOLKOVA, V.; BIOLEK, Z. **SPICE model of memristor with nonlinear dopant drift**. Radioengineering, 2009. ISSN 1210-2512.

EROKHIN, V.; FONTANA, M. P. **Electrochemically controlled polymeric device: a memristor (and more) found two years ago**. arXiv preprint arXiv:0807.0333, 2008.

CHANTHBOUALA, A. et al. **A ferroelectric memristor**. Nature materials, v. 11, n. 10, p. 860-864, 2012. ISSN 1476-1122.

IKEDA, S. et al. **A perpendicular-anisotropy CoFeB-MgO magnetic tunnel junction**. Nature materials, v. 9, n. 9, p. 721-724, 2010. ISSN 1476-1122.

BLANC, J.; STAEBLER, D. L. **Electrocoloration in SrTiO: Vacancy drift and oxidation- reduction of transition metals**. Physical Review B, v. 4, n. 10, p. 3548, 1971.

STRUKOV, D. B. et al. **The missing memristor found. nature**, v. 453, n. 7191, p. 80-83, 2008. ISSN 0028-0836.

LAIHO, M.; LEHTONEN, E. **Cellular nanoscale network cell with memristors for local implication logic and synapses**. Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on, 2010, IEEE. p.2051-2054.

Yang, J. J., Strukov, D. B., & Stewart, D. R. (2013). **Memristive devices for computing**. Nature nanotechnology, 8(1), 13-24.

WASER, R. et al. **Redox-based resistive switching memories-nanoionic mechanisms, prospects, and challenges**. Advanced materials, v. 21, n. 25-26, p. 2632-2663, 2009. ISSN 1521-4095.

PISSARDINI, Rodrigo de Sousa. **MEMCOMPUTAÇÃO: características e aplicações em computação paralela.** Escola Politécnica da Universidade de São Paulo, São Paulo, 2017.

L. T. Valavala, K. Munot and R. T. Karri Babu, "**Design of CMOS Inverter and Chain of Inverters Using Neural Networks**," 2018 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS), Hyderabad, India, 2018, pp. 269-274, doi: 10.1109/iSES.2018.00065.

SEDRA, Adel S.; SMITH, Kenneth C.; CARUSONE, Tony Chan; GAUDET, Vincent. **Circuitos Microeletronicos**. 8. ed. Brasil: Elsevier, 2023. 776 p.

Descripcion del modelo electrico del memristor - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Estructura-fisica-del-memristor-sintetizado-por-HP_fig1_230731606 [accessed 23 Aug, 2023]

RASHID, Muhammad H. **Eletrônica de Potência**: dispositivos, circuitos e aplicações. 4. ed. Brasil: Pearson, 2014.

C. Yakopcic, A. Sarangan, J. Gao, T. M. Taha, G. Subramanyam and S. Rogers, "**TiO2 memristor devices**," Proceedings of the 2011 IEEE National Aerospace and Electronics Conference (NAECON), Dayton, OH, USA, 2011, pp. 101-104, doi: 10.1109/NAECON.2011.6183085.

C. Yakopcic, T. M. Taha, G. Subramanyam, R. E. Pino and S. Rogers, "**A Memristor Device Model**," in IEEE Electron Device Letters, vol. 32, no. 10, pp. 1436-1438, Oct. 2011, doi: 10.1109/LED.2011.2163292.

Pujari, Ruturaj & Rakheja, Shaloo. (2017). **Performance evaluation of copper and graphene nanoribbons in 2-D NoC structures**. 353-359. 10.1109/ISQED.2017.7918341.

C. Yakopcic, T. M. Taha, G. Subramanyam and R. E. Pino, "**Generalized Memristive Device SPICE Model and its Application in Circuit Design**," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 32, no. 8, pp. 1201-1214, Aug. 2013, doi: 10.1109/TCAD.2013.2252057.

A. Singh, **Memristor based XNOR for high speed área efcient 1-bit Full Adder**, in 2017 IEEE International Conference on Computing, Communication and Automation (2017), pp. 1549–1553

DIMITRAKOPOULOS, G. **Microarchitecture of network-on-chip routers.** [S.l.]: SPRINGERVERLAG NEW YORK, 2016

CHO, Kyoungrok; LEE, Sang-Jin; ESHRAGHIAN, Kamran. Memristor - **CMOS logic and digital computational components.** *Microelectronics Journal*, v. 46, n. 3, p. 214-220, 2015. ISSN 0026-2692. DOI: https://doi.org/10.1016/j.mejo.2014.12.006. Disponível em: https://www.sciencedirect.com/science/article/pii/S0026269214003553.

WAKERLY, J. F. **Digital design: principles and practices.** [S.l.]: Prentice Hall, 2000. 949 p.

REDMOND, Joanna. **SRAM vs DRAM: Difference Between SRAM & DRAM Explained**. 2023. Disponível em: https://www.enterprisestorageforum.com/hardware/sram-vs-dram/. Acesso em: 12 set. 2023.

MOLTER, Tim. **Memristor Models in LTSpice**: knowm. Knowm. 2017. Disponível em: https://knowm.org/memristor-models-in-ltspice/. Acesso em: 23 ago. 2023.

JANTSCH, A.; TENHUNEN, H. **Networks on chip.** [S.l.]: Kluwer Academic Publishers, 2003. 303 p.

HOLSMARK, R.; PALESI, M.; KUMAR, S. **Deadlock free routing algorithms for irregular mesh topology NoC systems with rectangular regions. Journal of Systems Architecture**, v. 54, n. 3, p. 427–440, 2008.

SERPANOS, D. N.; WOLF, T. **Architecture of network systems.** Morgan Kaufmann, 2011. 320 p.

SWAPNA, S.; SWAIN, A. K.; MAHAPATRA, K. K. **Design and analysis of five port router for network on chip.** In: 2012 Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics. IEEE, 2012. p. 51–55.

45nm Transistor - TSMC (Taiwan Semiconductor Manufacturing Company) Disponível em: https://www.tsmc.com/english/dedicatedFoundry/technology/45nm.htm

NIMO. **Predictive Technology Model (PTM).** [S.l.: s.n.], 2012. http://ptm.asu.edu/. Acesso em 16 de Setembro de 2023.

DALLY, W. J.; TOWLES, B. P. **Principles and Practices of Interconnection Networks.** [S.l.]: Elsevier, 2003. 581 p.

FRANÇA, Luiz Augusto Scheuermann. **Computação neuromórfica em nanoescala**: desenvolvimento de arquitetura noc baseada em tecnologia mos para redes neurais. 2023. 77 f. TCC (Graduação) - Curso de Curso de Graduação em Engenharia de Controle e Automação, Departamento de Eng. de Controle, Automação e Computação, Universidade Federal de Santa Catarina, Blumenau, 2023. Disponível em: https://repositorio.ufsc.br/bitstream/handle/123456789/248787/TCC_Luiz_Scheuermann.pdf?sequence=1#page=67&zoom=100,113,276. Acesso em: 16 set. 2023.

MUSTAQUEEM, Zeba; ANSARI, Abdul Quaiyum; AKRAM, Md Waseem. Low Leakage and Robust Sub-threshold SRAM Cell Using Memristor. In: INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 4., 2022. **JOURNAL.** Pan, 2021. v. 68, p. 667-676.

CÂMARA, B. O.; GUIMARÃES, J. G.; COSTA, J. C. **Proposal of a router circuit based on nanoelectronic devices.** In: Advanced Manufacturing, Electronics and Microsystems TechConnect Briefs 2016. [S.l.]: TechConnect, 2016. p. 183–187. Disponivel em: < https://briefs.techconnect.org/wp-content/volumes/TCB2016v4/pdf/379.pdf>.

FAROOQ, Umer; ASLAM, M. Hassan; USMAN, Muhammad. Computer and Information Sciences: on the comparison of memristor-transistor hybrid and transistor-only heterogeneous fpgas. **Journal Of King Saud University.** Riade. 6 abr. 2018. Disponível em: http://www.sciencedirect.com/. Acesso em: 30 out. 2023.

PARAMASIVAM, K.; PRIYA, R. Sathiya; SAMINATHAN, V.. Design and Analysis of Memristor Memory Cell Using Different Windowing Functions. **International Journal Of Innovative Technology And Exploring Engineering (Ijitee).** Dez. 2018. Disponível em: https://www.ijitee.org/wp-content/uploads/papers/v8i2s2/BS2041128218.pdf. Acesso em: 23 ago. 2023.

L. H. B. Sardinha, A. M. M. Costa, O. P. V. Neto, L. F. M. Vieira and M. A. M. Vieira, "**NanoRouter: A Quantum-dot Cellular Automata Design**" in IEEE Journal on Selected Áreas in Communications, vol. 31, no. 12, pp. 825-834, December 2013, doi: 10.1109/JSAC.2013.SUP2.12130015.

S. S. Kavitha and N. Kaulgud, "Quantum dot cellular automata (QCA) design for the realization of basic logic gates," 2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT), Mysuru, India, 2017, pp. 314-317, doi: 10.1109/ICEECCOT.2017.8284519.

JAISWAL, Ramanand; SASAMAL, Trailokya Nath. Efficient Design of Exclusive-Or Gate using 5-Input Majority Gate in QCA. **Iop Conference Series: Materials Science and Engineering.** mar. 2013. Disponível em: https://iopscience.iop.org/article/10.1088/1757-899X/225/1/012143/pdf. Acesso em: 02 nov. 2023.

ZAIDI, Adil; VERMA, Ankit; RAHEJA, Ashish. Design & Simulation of CMOS Inverter at Nanoscale beyond 22nm. **International Journal Of Emerging Science And Engineering.** maio 2013. Disponível em: https://www.ijese.org/wp-content/uploads/Papers/v1i5/E0224031513.pdf. Acesso em: 02 nov. 2023.

FARIAS, Clayton; MEINHARDT, Cristina; BUTZEN, Paulo F. **Investigating CMOS Inverters Noise Margins at Different Technologic Nodes**. 2018. Disponível em: https://sbmicro.org.br/sforum-eventos/sforum2018/Investigating%20CMOS%20Inverters%20Noise%20Margins%20at%20Different%20Technologic%20Nodes.pdf. Acesso em: 08 nov. 2023.

DIAS, Cesar de Souza. **Aplicação da tecnologia memresistiva ao projeto de sistemas digitais através da exploração de circuitos híbridos e implicação material.** 2018. 150 f. Dissertação (Mestrado) - Curso de Pós-Graduação em Computação, Centro de Ciências Computacionais, Universidade Federal do Rio Grande, Rio Grande, 2018. Disponível em: https://sucupira.capes.gov.br/sucupira/public/consultas/coleta/trabalhoConclusao/viewTrabalhoConclusao.jsf?popup=true&id_trabalho=6318358. Acesso em: 20 set. 2023.

MENDES, Pedro. **MEMRISTOR:** a resistência com memória. Braga - Portugal, 1p, 25 jun. 2016

APÊNDICE A

A.1 SRAM HIERARQUIZADA

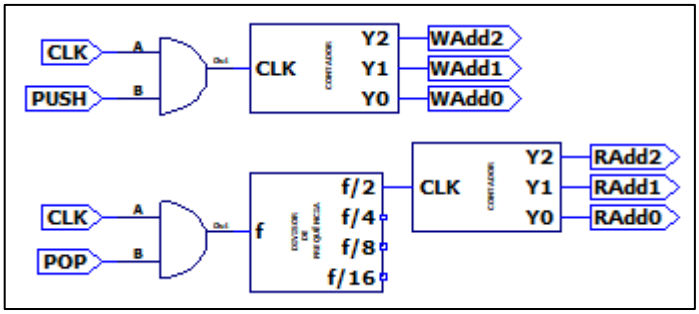
A memória desenvolvida por CÂMARA (2017), exibida na figura A.1, opera como uma SRAM dual-port falsa, com dois endereçamentos independentes, mas compartilhando controles de leitura e escrita. Cada controle possui um endereço separado, permitindo o controle individual e simultâneo das operações em uma capacidade de 8 palavras de 8 bits. Cada bit é representado por um flip-flop D e é numerado de b7 a b0. Para escrever novos dados, o *array* deve ser selecionado no modo de escrita com SW (*Select Write*) e um comando de escrita é emitido por meio de WEn (*Write Enable*). Para ler os dados, é necessário selecionar o *array* no modo de leitura usando SR (*Select Read*) e ativar REn (*Read Enable*). A seleção do *array* é realizada por meio de decodificadores 3:8, um para escrita e outro para leitura, com entradas S2, S1 e S0 representando os bits de endereço da SRAM.

A.2 FIFO

A.2.1 **Lógica de ponteiro**

Neste FIFO circular, o ponteiro percorre os endereços de 0 a 7, retornando a 0 quando atinge o final, criando a sensação de memória contínua. Tanto o ponteiro de escrita como o de leitura são incrementados após cada operação correspondente. Para realizar isso, utiliza-se uma lógica de ponteiro que consiste em um contador binário de 3 bits acionado na borda de descida do *clock*. O ponteiro de escrita, evidenciado na figura A.2, é controlado pelo sinal de *clock* geral do FIFO e pelo sinal de push, enquanto o ponteiro de leitura é controlado pelo dobro do período do sinal de *clock* do sistema e pelo sinal de *pop*. Essa duplicação do período de *clock* assegura que os dados de saída do FIFO permaneçam disponíveis por um ciclo completo do *clock* do sistema.

Figura A.2 – Logica de ponteiro.

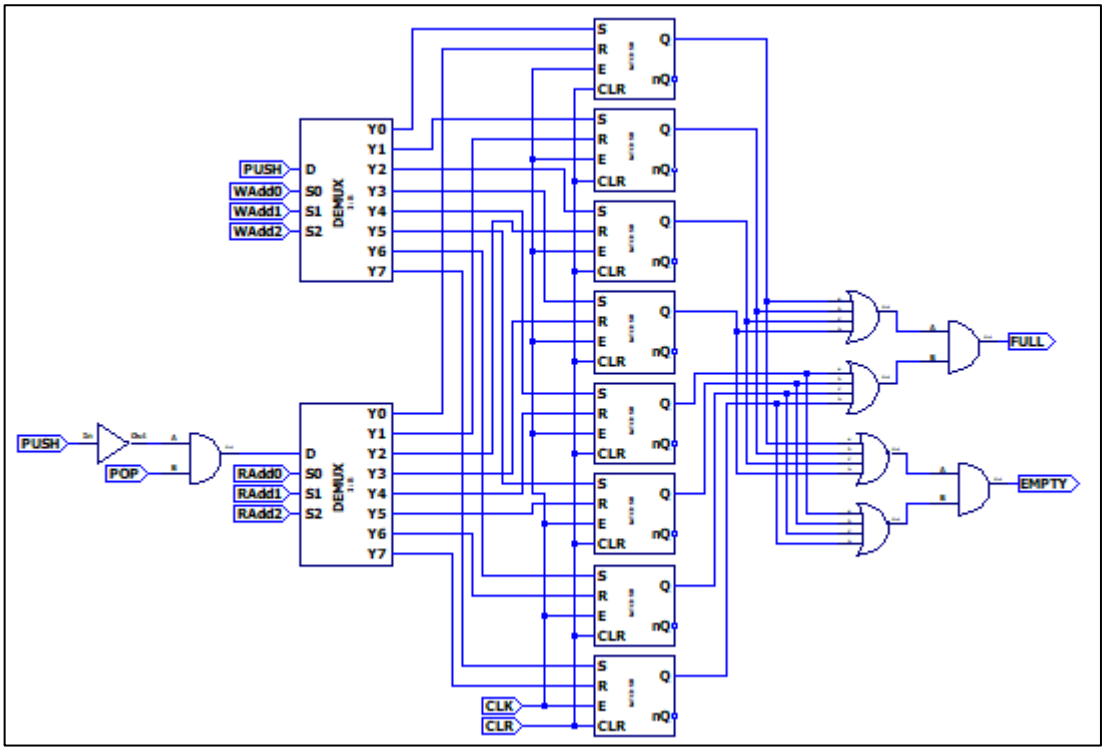


Fonte: Autor.

A.2.2 **Lógica de flag**

O FIFO em questão possui duas flags, *full* que indica a ocupação completa da memória e *empty* que indica a ausência de dados. O gerenciamento das flags é realizado através de latches SR, um para cada *array* na memória, que são acionados por operações de escrita (*SET*) ou leitura (*RESET*). São necessários 8 latches SR, endereçados por dois DEMUX, um para escrita (Wadd) e outro para leitura (Radd), priorizando a operação de escrita em relação à leitura. A flag *full* indica que todos os *arrays* foram escritos, mas ainda não lidos, e a flag *empty* indica que todos os *arrays* foram lidos ou ainda não receberam dados. O esquemático utilizado para a lógica de flag é mostrado na figura A.3.

Figura A.3 – Esquemático da lógica de flag.

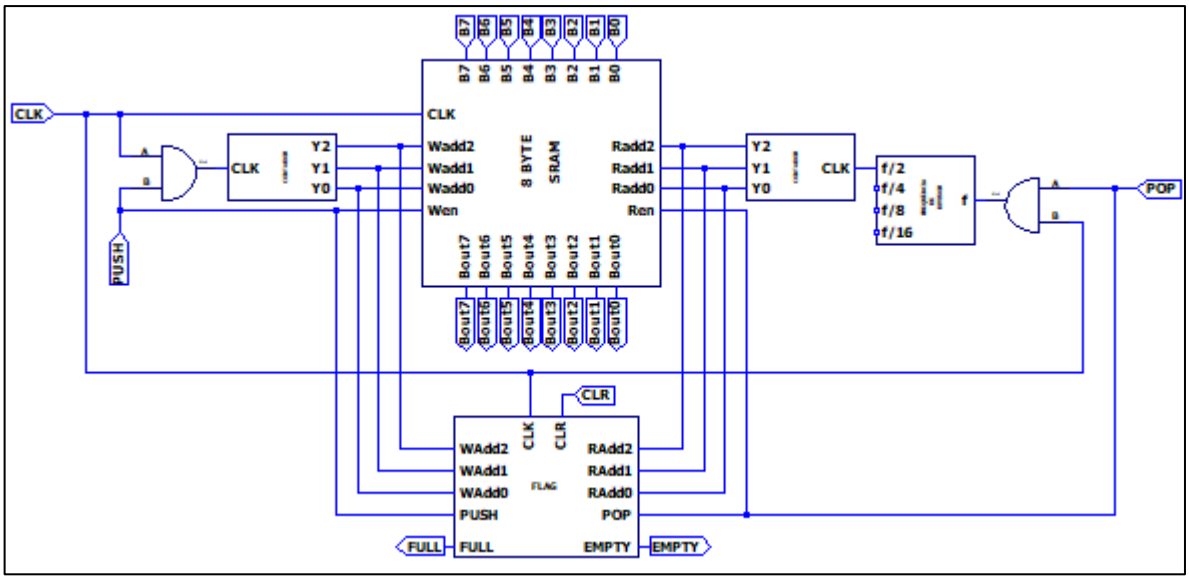


Fonte: Autor.

A.2.3 **FIFO**

O FIFO possui lógicas de flag *full* e *empty* para indicar quando todas as posições de memória estão preenchidas ou vazias, respectivamente. Latches SR são usados para indicar operações de escrita e leitura em cada *array* da memória SRAM. O FIFO utiliza ponteiros de escrita e leitura, que são conectados aos endereços de escrita e leitura, bem como às lógicas de flag. Os sinais de *push* e *pop* estão ligados aos comandos de escrita e leitura na SRAM e nas lógicas de flag. A figura A.4 demonstra o esquemático do registrador FIFO memresistivo completo idealizado neste trabalho.

Figura A.4 – Esquemático do registrador FIFO.



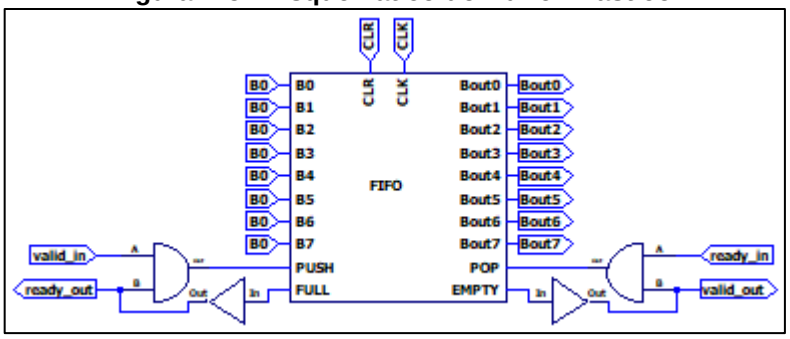
Fonte: Autor.

A.3 BUFFER ELÁSTICO

O protocolo *Event-Based* é uma maneira eficaz de obter dados de um FIFO. Ele utiliza sinais como *valid_in* e *valid_out* para indicar a disponibilidade de dados para envio ou recebimento. Além disso, *ready_out* e *ready_in* indicam se o FIFO está pronto para receber dados ou se o receptor está preparado para recebê-los. Para evitar problemas de gerenciamento quando o FIFO está cheio ou vazio, o EB utiliza portas AND externas. Isso garante que um envio, ou *push*, só ocorra quando há espaço disponível na memória do FIFO e dados disponíveis para envio. Da mesma forma, um *pull* (recepção) só ocorre quando o FIFO contém dados para o receptor e o receptor

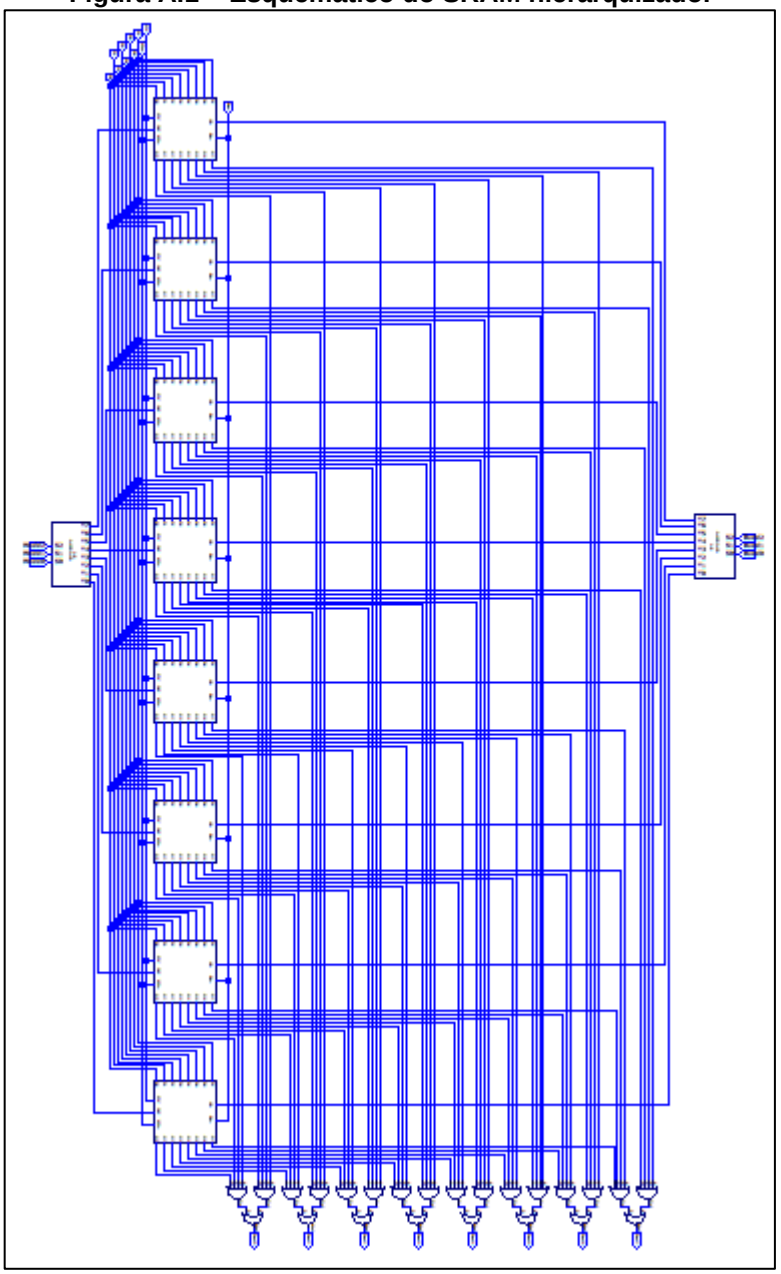
está pronto para recebê-los. A figura A.5 mostra o esquemático do EB memresistivo idealizado neste trabalho.

Figura A.5 – Esquemático do Buffer Elástico.



Fonte: Autor.

Figura A.1 – Esquemático do SRAM hierarquizado.



Fonte: Autor.

APÊNDICE B

Para o cálculo da área ocupada pelo circuito, utilizou-se da seguinte logica:

$$\boldsymbol{A_T = W \cdot L} \quad \text{(B.1)}$$

$$A_M = \frac{A_T}{2} \quad \text{(B.2)}$$

$$A_c = N \cdot (A_T + A_M) \quad \text{(B.3)}$$

Onde:

- $A_T \rightarrow$ Área do transistor;
- $A_M \rightarrow$ Área do *memristor*;
- $A_T + A_M \rightarrow$ Área de uma NAND memresistiva;
- $N \rightarrow$ Quantidade de transistores ou *memristors* presentes no circuito;
- $A_c \rightarrow$ Área total do circuito.

Quanto ao cálculo da potência, estática ou dinâmica, dissipada pelo circuito, utilizou-se da seguinte logica:

$$\boldsymbol{P_T = P_{T_{NMOS}} + P_{T_{PMOS}}} \quad \text{(B.4)}$$

$$P_M = P_{M_1} + P_{M_2} \quad \text{(B.5)}$$

$$P = N \cdot (P_T + P_M) \quad \text{(B.6)}$$

Onde:

- $P_T \rightarrow$ Potência dissipada pelos transistores;
- $P_M \rightarrow$ Potência dissipada pelos *memristors*;
- $P_T + P_M \rightarrow$ Potência total dissipada por uma NAND memresistiva;
- $N \rightarrow$ Quantidade de *memristors* presentes no circuito;
- $P_c \rightarrow$ Potencia total dissipada pelo circuito.