UM PADRÃO DE FORMATO DE ARQUIVO TEXTO PARA DESCREVER FUNÇÕES THRESHOLD

MICAEL MARTINS POPPING; RENATO SOUZA DE SOUZA;
FELIPE DE SOUZA MARQUES; LEOMAR SOARES DA ROSA JR.

Universidade Federal de Pelotas – {mmpopping; rsdsouza; felipem; leomarjr}@inf.ufpel.edu.br

1. INTRODUÇÃO

A indústria de microeletrônica apresenta um enorme avanço, durante as últimas décadas, no desenvolvimento de circuitos VLSI (*Very-Large-Scale Integration*). Essa evolução, basicamente, se deve ao grande avanço na escala da tecnologia MOSFET (*Metal-Oxide-Semiconductor Field Effect Transistor*). Embora a redução do canal dos transistores venha contribuindo para o aumento da densidade e do desempenho dos circuitos integrados, a agressiva redução na escala dos transistores tem apresentado consequências físicas e elétricas com impacto no funcionamento dos dispositivos. Um dos principais desafios foi reduzir o consumo de potência estática (*leakage*) em um circuito integrado. Pois a estrutura planar do gate dos transistores não permite um controle total e eficaz da movimentação dos elétrons pelo canal do transistor. Isto se deve ao fato de que, o transistor MOSFET esta atingindo seus limites físicos de construção, aproximando-se das dimensões atômicas do silício(Si), no qual é a base de sua produção. Sendo assim, estima-se que nos próximos anos, o transistor MOSFET será substituído por novos dispositivos que permitem a continuação de progresso dos circuitos integrados, previsto pela Lei de Moore (ITRS, 2015).

Além do desenvolvimento na redução do transistor MOSFET, novas buscas por alternativas em diferentes tecnologias estão sendo realizadas para o contínuo avanço tecnológico, fomentando a criação de tecnologias emergentes.

A redução das dimensões do canal do transistor permitiu que a indústria de semicondutores fosse capaz de fabricar sistemas digitais cada vez mais complexos. Sendo assim, as ferramentas de EDA (*Electronic Design Automation*) foram de suma importância, pois, tiveram que se adaptar a trabalhar com projetos com um número crescente de componentes. No entanto, a abstração lógica dos transistores não mudou, mesmo com a inserção da tecnologia FinFET (*Field Effect Transistor*), não apresentando efeitos dentro da síntese lógica. Porém com o surgimento de novas nanotecnologias, novos paradigmas estão sendo descobertos. Pois esses dispositivos apresentam características diferentes do transistor tradicional, desta forma, originando
novas abstrações lógicas.

Memristores, *Spintronic*, *RTD* (diodos de tunelamento ressonante) apresentam como tecnologia, onde o gate básico pode ser um *Threshold Logic Gate* (TLG). *Threshold Logic* é um poderoso paradigma alternativo para implementar funções Booleanas em projetos digitais. Uma função de *threshold* (TLF), basicamente, pode ser como uma função Booleana no qual a saída depende dos pesos das entradas e um valor limiar. Teoricamente, uma função de *threshold* precisa satisfazer as seguintes condições: cada entrada contém um peso específico e o gate possui um valor limiar; onde a soma dos pesos das entradas for igual ou maior que o valor limiar, a função avalia para 1 lógico, caso

contrário, avalia para 0 lógico. Essa definição pode ser expressa pela seguinte forma: (MUROGA, 1971)

$$f = \{1, \ se \ \sum_{i=1}^{n} x_i \, w_i \geq t; \ 0, \ caso \ contrário$$

onde x_i representa os valores Booleanos de cada entrada {0,1}, w_i os pesos de cada entrada, e T é o valor limiar. Uma TLF é representada por um vetor w_1, w_2 , ... , w_n ; T . Por exemplo, os vetores TLF para as seguintes funções f = x_1*x_2*x_3 e g = x_1+x_2+x_3 são [1,1,1;3] e [1,1,1;1], respectivamente (NEUTZLING, 2014).

Uma propriedade importante de um TLG, é a possibilidade de implementar diferentes funções Booleanas com a simples mudança dos pesos das entradas e/ou o valor limiar. Sendo assim, diversos trabalhos vem ganhando destaque em apresentar soluções que aumentam as possibilidades do uso de dispositivos baseado em lógica de *Threshold*. Em 2005, Zhang, propôs a primeira metodologia para síntese e otimização de lógica de *Threshold* multinível (ZHANG; GUPTA; ZHONG; JHA, 2005). O primeiro método heurístico para identificar um TLF foi proposto por Gowda em (GOWDA, et al ., 2011). Em (NEUTZLING, et al. 2017) foi proposto um método heurístico direto e efetivo, baseado em Composição Funcional, para identificar um TLF. Tornando-se o primeiro algoritmo heurístico que é capaz de encontrar todos os TLGs com até seis variáveis, apresentando resultados melhores em comparação com outros métodos heurísticos e mantendo um tempo de execução similar. Sendo assim, é possível notar que estudos voltados a funções de Threshold vêm ganhando força nos últimos anos, desta forma estabelecendo novas lacunas a serem estudadas neste meio. Uma destas lacunas é a ausência de um formato padrão, dentre as ferramentas atuais, que represente uma estrutura de dados na qual complemente todas as informações necessárias para um Threshold Logic Gate. Sendo assim, este trabalho, tem por objetivo propor um padrão de formato de arquivo texto para descrever circuitos que utilizam funções baseada em lógica de *threshold.*

2. METODOLOGIA

A descrição proposta é baseada no formato AIGER (Armin Biere, 2011). Em um arquivo texto, são inseridas todas as informações necessárias para representação de uma porta/circuito lógico. Dentre essas informações são: descrição das entradas e saída, quantidade de *gates* , pesos das entradas e o valor de *threshold* de cada *gate*. *A* descrição precisa seguir a seguinte ordem: (i) cabeçalho; (ii)definição das variáveis; (iii) definição da saída; (iv) declaração dos gates; (v) descrição do porta/circuito. Essa ordem de descrição é apresentada pela Figura 1, onde é possível notar que ela define o funcionamento de uma porta lógica *AND*.

```
1. and3 5 3 1 1
2. 10
3. 12
4. 14
5. 20
```

```
6. 20 1 1 1 3
7. 20 10 12 14
```

Figura 1. Descrição de uma porta AND de três entradas com Threshold Logic usando o formato proposto.

2.1 Cabeçalho

A primeira linha do arquivo texto precisa seguir uma certa ordem. Primeiro uma identificação/nome para a descrição. Como podemos notar, na Figura 1, na linha 1, o nome “and3”, representa o nome para a descrição. Em segundo, a quantidade de nodos, representado pelo valor “5”, linha 1 da Figura 1. Após, em terceiro, a informação de quantas entradas existem. Na descrição da Figura 1, é possível notar que existem três variáveis, informado pelo “3” presente na linha 1. Depois, ainda na linha 1 da Figura 1, tem a informação da quantidade de saída, representado pelo “1”. E por fim, a informação de quantos *gates* fazem parte do circuito, no caso da Figura 1, existe apenas um *gate,* representado pelo último “1” presente na linha 1.

2.2 Declaração de entradas

É feita a declaração de cada uma das entradas. A quantidade de variáveis de ser a mesma na qual foi informada no cabeçalho. Cada declaração deve ser realizada em linhas diferentes. Sendo assim, é possível notar nas linhas 2, 3 e 4 da Figura 1 a declaração das três variáveis. Por padrão, cada entrada é representada por um número par. Caso essa variável tenha sua polaridade invertida, ela deve ser representado por um valor ímpar.

2.3 Declaração da saída

A saída é descrita de forma análoga a declaração das entradas. Assim, deve-se utilizar um número par para representar uma saída direta, ou um número ímpar para representar esta saída negada. A linha 5, da Figura 1, representa esta declaração.

2.4 Declaração do gate

Na declaração de cada *gate*, é necessário seguir um padrão. São essenciais cinco informações. A primeira, é a identificação do *gate* , ou seja, é utilizado um valor qualquer para identificá-lo. As próximas três informações, são os valores de peso de cada variável. Lembrando que cada *gate threshold* utilizado neste trabalho, é composto por três entradas. E por fim, a última informação é o valor limiar(valor de *threshold*) do *gate*. Sendo assim, a linha 6 da Figura 1, apresenta esta declaração. Onde, o valor “20” é a identificação do *gate*, os três valores “1”, são os pesos de cada entrada e o valor “3” é o valor limiar.

2.5 Descrição do circuito

Por fim, o circuito é descrito informando as relações entre cada *gate* e as entradas. Para isso, é preciso informar qual *gate* será usado e quais as entradas serão ligadas. Desta forma, esta descrição é utilizada quatro informações. Primeiro o número que identifica o *gate* já declarado. E as próximos três informações são os valores que identificam as entradas. Na Figura 1, a linha 7 ilustra esta declaração. O valor “20” é referencia ao *gate* declarado anteriormente e os valores, “10”, “12” e “14” são as entradas que serão ligadas ao *gate* “20”.

3. RESULTADOS E DISCUSSÃO

O formato de descrição proposto já vem sendo utilizado em algoritmos, dedicados a Threshold Logic, desenvolvidos dentro do grupo de pesquisa.

4. CONCLUSÕES

Este trabalho apresentou um nova proposta para um formato de arquivo texto para descrição de circuitos/portas lógicas dedicadas a lógica baseada em *Threshold.* A descrição proposta foi baseada em uma outra descrição de circuitos bem consolidada dentro da academia, que éo formato AIGER, na qual descreve circuitos no formato de AIG (MISHCHENKO; CHATTERJEE; BRAYTON, 2005). O formato proposto apresenta uma forma bem simples de ser descrita. Como trabalhos futuros, pretende-se desenvolver um método automático para geração do formato, a partir da análise de uma estrutura de dados com lógica de *Threshold.* Por fim, o método automático desenvolvido para geração do formato proposto, será incorporado na ferramenta ABC (Berkeley Logic Synthesis and Verification Group) para fazer parte do pacote de algoritmos dedicados a lógica de *Threshold*.

5. REFERÊNCIAS BIBLIOGRÁFICAS

AIGER. The AIGER And-Inverter Graph (AIG). Avaliable in: http://fmv.jku.at/aiger/, Accessed in: 2019.

Berkeley Logic Synthesis and Verification Group. ABC: A Sys- tem for Sequential Synthesis and Verification. Disponível em: https://people.eecs.berkeley.edu/ alanmi/abc/. Acesso em: 08-2019.

GOWDA, T. et al. Identification of threshold functions and synthesis of threshold networks. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, vol.30, 2011. 665-677.

ITRS. Semiconductor Industries Association Roadmap. Avaliable in: http://public.itrs.net, Accessed in: 2019.

MISHCHENKO, A.; CHATTERJEE, S.; BRAYTON, R. FRAIGs: A unifying representation for logic synthesis and verification. [S.l.]: ERL Technical Report, EECS Dept., UC Berkeley, 2005.

MUROGA, S. Threshold Logic and Its Applications. New York: Wiley-Interscience, 1971.

NEUTZLING, A. “Synthesis of Threshold Logic Based Circuits”. Dissertação do Programa de Pós-Graduação em Computação, UFRGS. Porto Alegre. 2014. p. 71.

NEUTZLING, A. et al. “A Simple and Effective Heuristic Method for Threshold Logic Identification”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), 2017.

ZHANG, R. GUPTA, P. ZHONG, L. JHA, N. K. “Threshold Network Synthesis and Optimization and Its Application to Nanotechnologies”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), 2005, pp. 107 - 118.