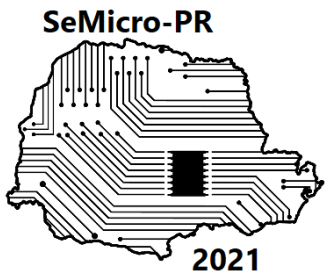

Implementação do controle digital de um ADC SAR de 8-bits

Edivania Ferreira Silva[1] e Paulo César C. de Aguirre.[1]
[1] Universidade Federal do Pampa, Alegrete, Brasil
edivaniaferreira.aluno@unipampa.edu.br e pauloaguirre@unipampa.edu.br

***Resumo*—Neste artigo é apresentado o desenvolvimento e os resultados obtidos de duas topologias de um controle digital para um Conversor Analógico-Digital de Aproximação Sucessiva (*ADC SAR)*. Estes circuitos foram elaborados utilizando duas máquinas de estados finitos e implementados na linguagem de descrição de *hardware SystemVerilog*. Os circuitos foram validados inicialmente no simulador ModelSim e depois em uma simulação AMS no Virtuoso Suite da Cadence. Além disso, foi feita a síntese lógica utilizando o *software* Cadence Genus Synthesis Solution e a síntese física com o s*oftware* Cadence Innovus Implementation System.**

I. INTRODUÇÃO

Os Conversores Analógico-Digitais (*ADC's*) são uma ligação entre o mundo digital e o analógico, pois desenvolvem um importante papel no processamento de sinais juntamente com o Conversores Digital para Analógico (*DAC's*) [1].

Esses conversores possuem várias topologias como: *Pipeline*, *Sigma-delta*, *Flash*, *Dual-Slope* e *SAR* [2]. Dentre essas topologias o *ADC SAR* é o mais utilizado em sistemas de baixo consumo de energia e velocidade. Ele é geralmente empregado em sistemas de controle de processos industriais, comunicação óptica e biomédica e também é um bom candidato para aplicativos de Internet das Coisas (IoT) e de voltagem ultra baixa [3].

O *ADC SAR* é composto por um circuito de amostragem e retenção (*S&H*), um Conversor Digital-Analógico (*DAC)* de *feedback*, um comparador e um circuito de controle digital que comanda a lógica *SAR* [4]. Com base nos resultados da comparação entre o sinal de amostragem e a realimentação do *DAC*, o circuito de controle digital determina o valor de cada bit de saída do *ADC* sucessiva e sequencialmente, do bit mais significativo (*MSB*) ao menos significativo *(LSB*) [5]. O controlador do *ADC SAR* é usualmente implementado em linguagem de descrição de *hardware (HDL*), e a sua implementação pode impactar significativamente no número de portas lógicas necessárias para a implementação do circuito.

Este trabalho apresenta duas topologias de um controle digital implementadas em *SystemVerilog*, para um *ADC SAR* de 8 bits. Este artigo está organizado da seguinte forma: a Seção II apresenta a lógica *SAR*, a Seção III apresenta os circuitos digitais de controle implementados neste trabalho, a Seção IV apresenta os resultados da síntese lógica, a Seção V apresenta a síntese física dos circuitos e a Seção VI apresenta algumas conclusões.

II. LÓGICA SAR

Um conversor analogico-digital por aproximação sucessiva é composto por quatro sub circuitos principais, um circuito de amostragem e retenção, um comparador de tensão analógico, um circuito de registro de aproximação e um *DAC*. Neste trabalho considera-se um ADC SAR por redistribuição de carga de 8 bits, conforme ilustrado na Fig. 1.

A metodologia de conversão por aproximações sucessivas permite obter-se uma maior resolução do que o método de conversão paralela. Esse tipo de conversão utiliza uma técnica de realimentação para comparar uma entrada de tensão analógica com um valor binário que corresponde ao número de bits de resolução do *ADC* e necessita de *N+2* ciclos de *Clock* para realizar a conversão. O algoritmo de aproximação sucessiva inicia zerando todo o registro de bits e em seguida atribui um valor de nível lógico alto para ao bit mais significativo, esse valor fica registrado e é direcionado ao *DAC* que o converterá em um sinal analógico, e então esse sinal é conduzido ao comparador. O comparador faz a comparação desse sinal com o sinal de entrada *Vin*, e dependendo do resultado, o valor que foi atribuído anteriormente ao *MSB* permanecerá em nível lógico alto ou irá para nível lógico baixo. Esse processo se repete até que todos os bits anteriores ao *MSB* tenham seus valores definidos. O controle digital comanda o início e fim de cada etapa de aproximação e o resultado só é validado quando a saída que pode ser denominada "status" for acionada em nível lógico alto, isso quando todo o processo for finalizado.

III. CIRCUITOS IMPLEMENTADOS

O controle digital apresentado foi implementado na linguagem de descrição de *hardware SystemVerilog* e elaborado de duas formas diferentes para que fosse observado em qual implementação obteria-se um melhor resultado. Ambas as versões denominadas V1 e V2 são de 8 bits e

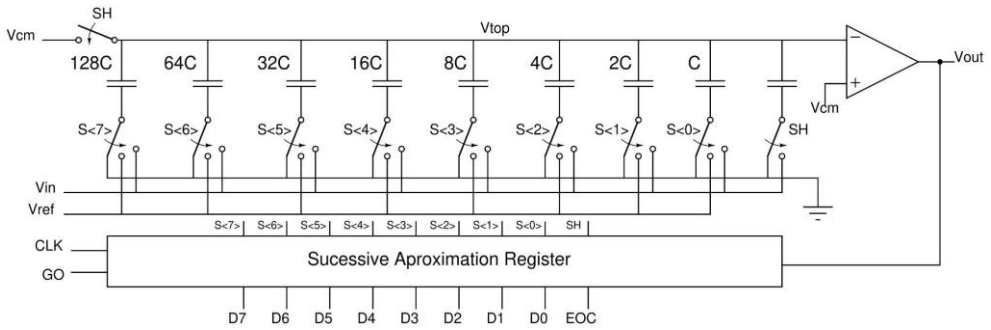

FIG. 1. ADC SAR DE 8 BITS

contam com as mesmas quantidades de entradas e saídas, três entradas e três saídas, apesar de possuírem lógicas distintas. Ambas implementações possuem uma entrada para o sinal de relógio (Clk), uma para o sinal do comparador (Comp) e uma para o reset (*Go*), e contam com um saída para o sinal de amostragem (*Sample)*, uma para o *DAC (Value*) e uma para o status da conversão (Conv_ok).

A. Versão 1

A versão V1, cuja máquina de estados está ilustrada na Fig. 2, possui uma variável N que determina a quantidade de bits. O valor dessa variável pode ser alterado a qualquer momento sem que seja necessário fazer grandes alterações no código.

A máquina de estados idealizada para esta versão possui quatro estados: *first*, *start*, *conversion e stop*. O seu funcionamento é iniciado quando ocorre a primeira subida de *Clock* e *Go* é colocado em nível lógico alto, e então, no estado *first*, todas as variáveis, as entradas e as saídas são iniciadas em zero, exceto *Value*, que recebe o seu próprio valor, porém "negado" (negação lógica), e *Sample*, que recebe "1". No estado *start* a variável que recebe o valores da máscara *(Mask*) e que possui tamanho igual a N recebe "1" no *LSB*, que é deslocado para o *MSB* através de um *shift left*. Esse valor atualizado de *Mask* é então guardado em uma variável auxiliar (Aux) e em *Value*, e *Sample* retorna a "0". O estado *Conversion* é responsável por fazer a conversão dos bits, sendo essa feita através de um loop utilizando *if else.* O primeiro bit a ser convertido é o *MSB*, sendo esse procedimento feito da seguinte maneira: se Comp for igual a um, o valor de *Mask* é atualizado com um deslocamento a direita e Aux recebe Aux "ou" (ou lógico) com *Mask*, se for igual a zero Aux recebe Aux "e" (e lógico) com *Mask* "negado" (negação lógica) e "ou" (ou lógico) com *Mask* deslocado à direita, neste caso *Mask* é atualizado com um deslocamento a direita. Em ambos os casos, após a conversão, *Value* recebe Aux e fornece esse valor ao *DAC*.

O loop de conversão é feito da seguinte maneira: em um *if* é colocada a seguinte notação i < N , onde i corresponde a uma variável de incremento que tem valor inicial igual a "1". A cada vez que uma conversão é efetivada, com Comp igual a um ou a zero, i tem seu valor incrementado até se igualar a N. Quando isso ocorre a máquina vai para o próximo estado, *stop*. Neste estado ocorre a conversão do *LSB*, que por ser o último bit não pode ser convertido utilizando a lógica anterior. Essa conversão ocorre desta forma: se Comp for um Aux recebe Aux "ou" (ou lógico) com *Mask*, e se for zero, Aux recebe Aux "e" (e lógico) com *Mask* "negado" (negação lógica). Com o *LSB* já convertido, Conv_ok recebe "1" sinalizando que todos os bits já possuem seus valores definidos. Por fim, o resultado final é fornecido por *Value* e a máquina retorna ao estado inicial *first.*

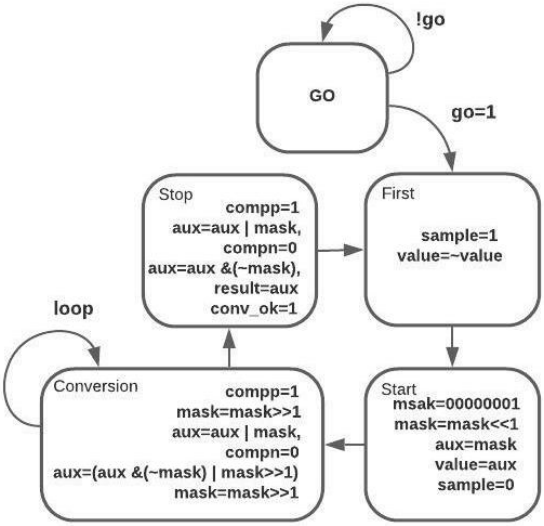

FIG. 2. MÁQUINA DE ESTADOS V1

B. Versão 2

Para V2, como está ilustrado na Fig. 3, a máquina de estados montada possui dez estados numerados de zero a nove. Os valores mascarados para serem convertidos foram organizados da seguinte maneira: B8=10000000, onde o número, no caso oito, corresponde a posição do bit que está em nível lógico alto, isso se repete até B1=00000001. Na primeira borda de subida de Clock, *Go* vai para nível lógico alto e então passa para o estado S0, nesse estado todas as entradas são iniciadas em zero, exceto *Sample* que recebe "1" e *Value* que recebe "11111111". Em S1, *Sample* retorna a "0" e *Value* recebe B8.

A partir do estado S2 começa o processo de comparação dos bits para que seus valores sejam definidos. Nesse estado,

S2, a comparação se dá da seguinte forma: se Comp for igual a um *Value* recebe *Value* "ou" (ou lógico) com B7, se for igual a zero, *Value* recebe B7. No estado S3, B7 é comparado pelo seguinte método: se Comp for igual a um *Value* recebe *Value* "ou" (ou lógico) com B6, se for igual a zero, *Value* recebe *Value* "e" (e lógico) com B7 negado e "ou" (ou lógico) com B6.

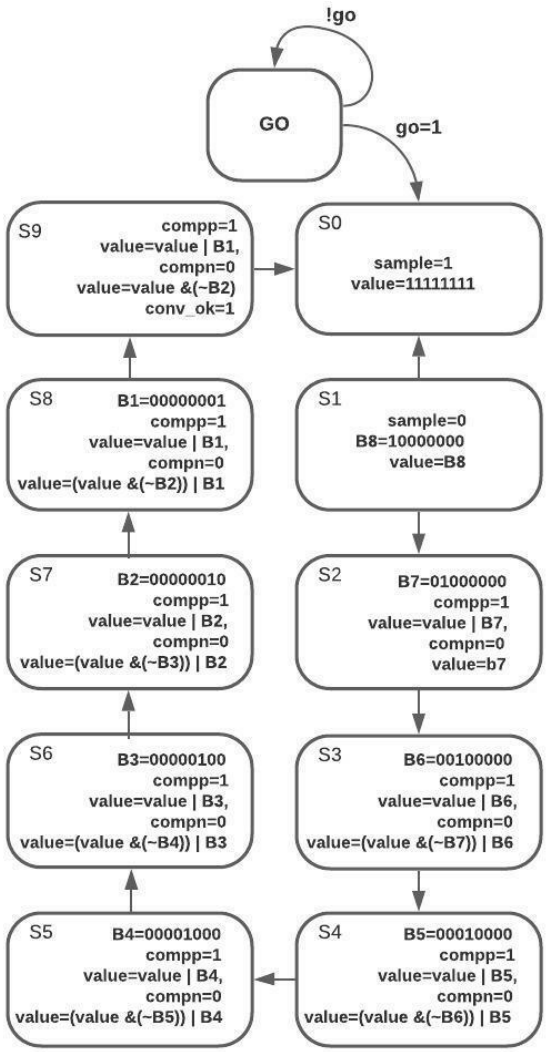

FIG. 3. MÁQUINA DE ESTADOS V2

A lógica do estado S3 se repete nos estados S4, S5, S6, S7, e S8 utilizados para comparar B6, B5, B4, B3 e B2, respectivamente. No estado S9 ocorre a comparação do B1, que não pode ser comparado utilizando a lógica anterior, da seguinte forma: se a saída do comparador for igual a um, *Value* recebe *Value* "ou" (ou lógico) B1, se for igual a zero, *Value* recebe Value "e" (e lógico) com B1 negado. Finalizada a comparação de todos os bits, Conv_ok recebe "1", *Value* fornece o resultado final e a máquina retorna para o estado S0.

Os dois controladores lógicos foram validados inicialmente no simulador ModelSim, e depois em uma simulação AMS no ambiente Virtuoso da Cadence, de modo análogo ao que foi apresentado em [6]. Nesta simulação, o ADC SAR considerado é ideal, ou seja, composto por capacitores e chaves ideais, além de um comparador implementado em verilogA, sem tensão de offset de entrada. O esquemático simplificado do ADC SAR em questão é apresentado na Fig. 1.

IV. SÍNTESE LÓGICA

Para comparar as diferentes implementações do controlador lógico para um ADC SAR de 8 bits, efetuou-se a síntese lógica de ambos os circuitos em tecnologia CMOS de 180 nm. A síntese lógica foi efetuada utilizando o software Cadence Genus Synthesis Solution. A Tabela 1 apresenta os resultados da síntese lógica, descrevendo o número de portas lógicas e a área total das portas lógicas. Nota-se que a versão V2 requereu apenas 33% do número de portas lógicas necessárias para implementar a versão 1, indicando que um design dedicado é mais otimizado.

Design	Portas Lógicas	Área (μm^2)
V1	197	10.683,2847
V2	65	3.553,088

TABELA 1. RESULTADO DAS SÍNTESES

V. SÍNTESE FÍSICA

Efetuou-se também a síntese física de ambos os circuitos utilizando o *software* Cadence Innovus Implementation System.

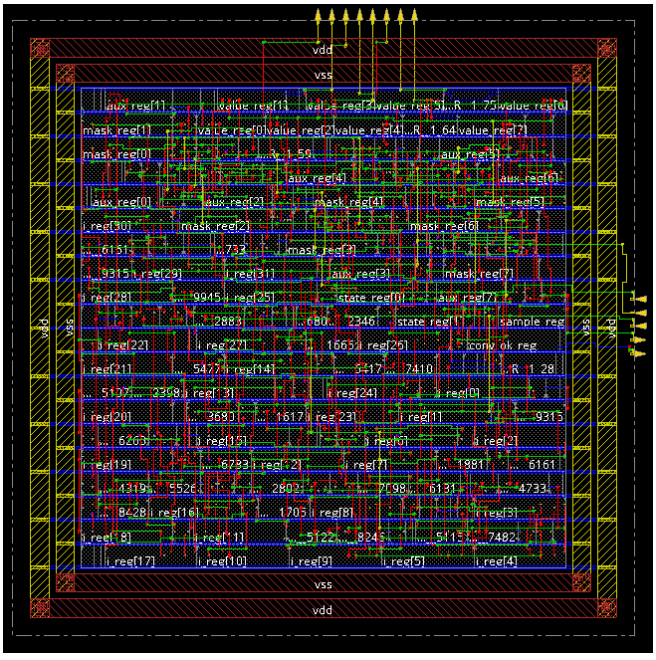

FIG. 4. LEIAUTE V1

Para efetuar uma comparação justa de área, optou-se por utilizar um *floorplan* com relação de aspecto de 1,0 e com a densidade de células utilizando 90% da área. As Figuras 4 e 5 apresentam o leiaute resultante das implementações V1 e V2, respectivamente. A área total de cada implementação está descrita na Tabela 1. É possível verificar que a área da

implementação V2 é aproximadamente um terço da área requerida para a implementação da versão V1. Devido às restrições do *process design kit* (PDK) empregado, não foi possível efetuar uma simulação AMS do circuito sintetizado no Virtuoso.

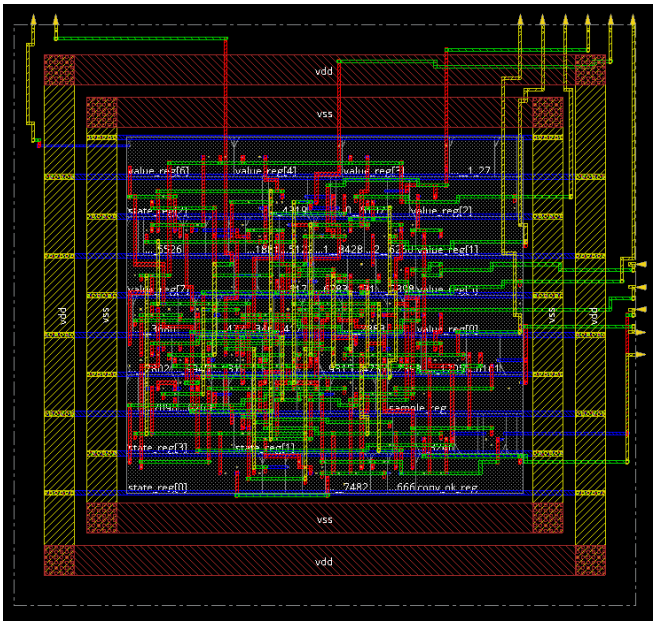

FIG. 5. LEIAUTE V2

VI. CONCLUSÃO

Este trabalho apresentou a implementação de dois circuitos distintos para o controle de um *ADC SAR* de 8 bits. A primeira versão foi otimizada para fácil reconfiguração referente ao número de bits do *ADC*, enquanto a outra versão foi desenvolvida exclusivamente para 8 bits. Evidenciou-se que a versão dedicada apresentou um número bastante reduzido de portas lógicas e área em relação à implementação versátil, de simples reconfiguração. Assim, conclui-se que deve-se otimizar a implementação do circuito digital de controle de *ADCs SAR* para reduzir a área de silício do circuito digital de controle.

AGRADECIMENTOS

Os autores gostariam de agradecer ao Programa de Desenvolvimento Acadêmico 2021 da UNIPAMPA.

REFERÊNCIAS

[1] R. Wittmann, F. Henkel, A. Ripp, A. Meyer, R. Wunderlich, S. Heinen,and M. Dietrich, “Innovative design methodology for analog-to-digital and digital-to-analog converters,” inANALOG 2020; 17th ITG/GMM-Symposium, 2020, pp. 1–6.

[2] F. Maloberti,Data Converters. Springer US, 2007.

[3] S.-H. Wang and C.-C. Hung, “A 0.3V 10B 3Ms/s SAR ADC with comparator calibration and kickback noise reduction for biomedical applications,”IEEE Transactions on Biomedical Circuits and Systems, vol. 14, no. 3,pp. 558–569, 2020.

[4] M. Pelgrom, Analog-to-Digital Conversion. Springer, 2017.

[5] A. Hassan, I. A. Halin, I. B. Aris, and M. K. Bin Hassan, “Design of 8-bit sar-adc cmos,” in2009 IEEE Student Conference on Research and Development (SCOReD), 2009, pp. 272–275.

[6] E. F. Silva, João L. J. Brum and P. C. C. de Aguirre, Digital Control of a Synchronous 8-bit SAR ADC, 21th Microelectronics Students Forum, August 23 to 27, 2021, VIRTUAL, BRAZIL