TÉCNICO LISBOA

Projeto de Sistemas Digitais

MEEC

ALU Simples [PT]

Autores:

João Marques Vinagre (93095) marquesvinagre@tecnico.ulisboa.pt
André Alexandre Costa Santos (96152) andresantos1.alf@gmail.com
Diogo Batista Araújo (96180) diogo.batista.araujo@tecnico.ulisboa.pt

Grupo 2

2022/2023 – 1º Semestre, P1

Conteúdo

1 Introdução

No o primeiro projeto, foi proposto o desenvolvimento de uma calculadora simples com capacidade para realizar quatro operações básicas: adição, multiplicação, NAND e "Shift-Right Arithmetic"a palavras de 12 bits.

Para o efeito, foi aproveitada a arquitetura estudada na primeira semana de aulas. A esta, foram efectuadas as devidas alterações de modo a obter o comportamento desejado. Nomeadamente, o aumento do tamanho dos registos de 8 para 12 bits, a reestruturação do ficheiro *datapath* de modo a acomodar as diferentes operações, bem como a reorganização do ficheiro *controlo* e a criação dos sinais extra necessários.

No final a arquitetura foi testada numa *FPGA* para validar o trabalho desenvolvido ao longo da semana.

2 Processo de design

A arquitetura desenhada é uma máquina de estados finita do tipo Moore com oito estados. Quatro destes estados correspondem às operações pedidas: adição, multiplicação, NAND e "Shift-Right Arithmetic". O estado "*INIT*"corresponde ao estado onde a máquina é inicializada. Os dois estados de "*LOAD*"são responsáveis por carregar os registos com os valores pretendidos pelo utilizador. Por fim, o estado "*END*", mostra o resultado da operação. Na figura seguinte é representado o diagrama de estados.

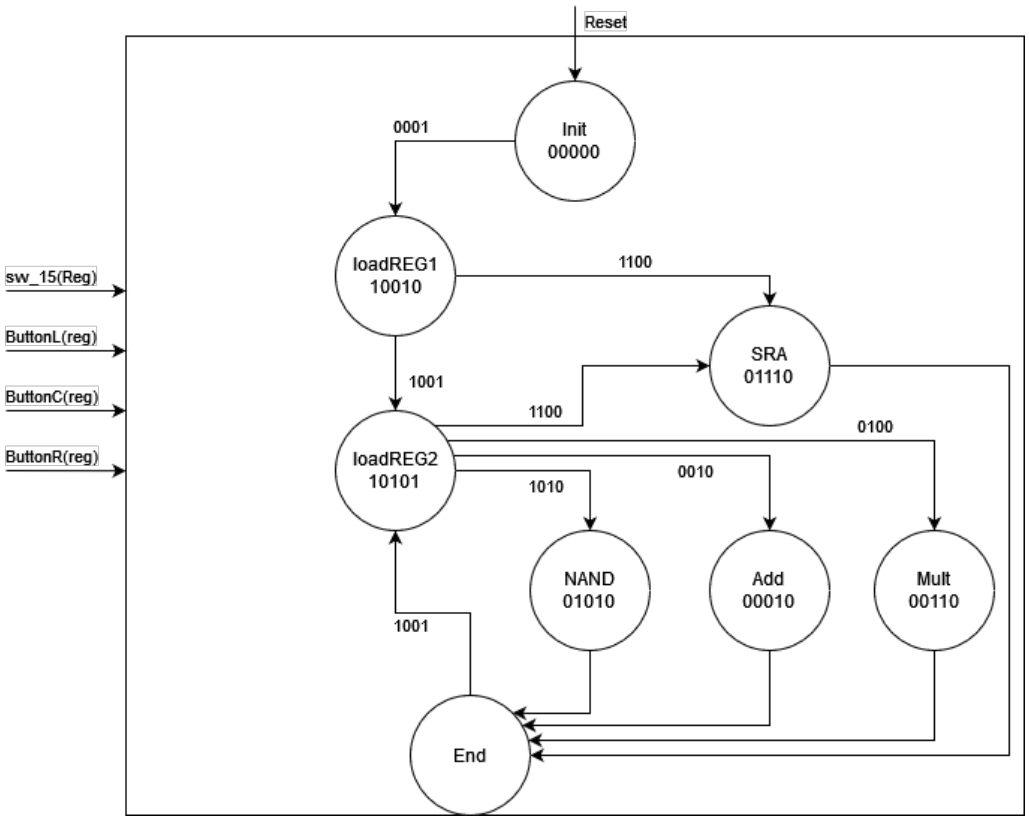

Figura 1: Diagrama de Estados ALU Simples

Como descrito no enunciado, a primeira operação que deve ser realizada é o load do valor no primeiro registo. Após isto ser realizado, é possível então realizar o Shift do valor introduzido, dado que não é necessário mais nenhum operando. É também possível prosseguir para a introdução do segundo operando e respetivo load, a partir do qual é possível realizar todas as operações programadas na calculadora. Após escolhida e realizada a operação, o utilizador tem a opção de voltar para a introdução do segundo valor, guardando o resultado obtido como primeiro operador, ou voltar para o início utilizando o botão de reset, reiniciando assim a calculadora.

2.1 Desafios

Inicialmente, foi ponderada uma máquina de estados que voltaria para o estado inicial após a realização de cada operação, ou após a realização de ambos os loads. No entanto, esta ideia não permitia a realização do Shift após introduzir apenas um operador, e era também incompatível com o requisito de voltar ao estado *"loadREG"* após pressionar Enter no estado final.

Ao desenvolver esta implementação, deparamo-nos com um problema: O facto de que, ao utilizar o mesmo botão para ambos os loads, o load do primeiro registo era "saltado", seguindo a máquina para o estado seguinte passado apenas um ciclo do relógio. Isto acontecia porque, ao entrar no estado *"loadREG1"*, o dedo do utilizador ainda estaria a premir o botão utilizado no ciclo de relógio seguinte. Consequentemente, a máquina de estados detetaria este input e interpretaria como o botão sendo premido de novo, saltando para o estado seguinte.

No desenvolvimento da operação de multiplicação, presente no "*datapath*", surgiu um novo problema: a máquina proposta realiza operações sobre valores com 12 bits, mas o resultado da multiplicação tem 24 (soma do número de bits de ambos os fatores da multiplicação). A solução utilizada foi a concatenação do número obtido após a multiplicação. É obtido um número de 24 bits após a operação, sendo descartados de seguida os 12 mais significativos. Foi também ponderada a utilização de 24 bits na saída da calculadora, mas a utilização deste método implicaria dificuldades com as operações lógicas, bem como na apresentação do resultado, já que o display da placa Basys3 tem apenas quatro algarismos. Por outro lado, realizando a concatenação do resultado é possível apenas multiplicar números entre 0 e 64 (2^6) já que valores superiores implicam uma perda de precisão no valor obtido.

3 Testes realizados

3.1 Simulação

Para a verificação do funcionamento da lógica na máquina de estados foi criado o ficheiro *circuito_tb.vhd* para estimular o circuito com vários imputs simulando a utilização final. Utilizando as funcionalidades de simulação do Vivado, é possível observar os valores gerados pelo circuito ao longo do tempo com os estímulos definidos no *circuito_tb.vhd*. Com esta ferramenta é possivel visualizar os sinais em todo o circuito, desta forma foi possível encontrar os desafios mencionados na secção anterior que foram prontamente corrigidos.

A primeira fase de implementação foi dada como concluída quando vários testes realizados foram passados com sucesso.

3.2 Testes na placa

Para preparar os testes na placa foi feita a síntese do circuito e a implementação da arquitetura para ser gerado o *BitStream* utilizado para programar a placa para que esta realize as operações desejadas. Os recursos da placa utilizados estão dispostos na seguinte tabela:

Recurso	Utilização	Utilização(%)
LUT	109	0,52%
FF	79	0,19%
DPS	1	1,11%
BUFG	3	9,38%

Apesar da baixa utilização de recursos relativamente aos disponíveis na placa, esta poderia ser otimizada. No entanto, este não era o foco do trabalho. Além disso, as possíveis vantagens oferecidas por essa optimização seriam desprezáveis dada a dimensão do projeto tanto na poupança de recursos como no possível ganho de velocidade. No caso de o circuito ter de partilhar espaço com outros, esta seria uma opção a considerar.

Com um clock de período 10ns, obtivémos um *"Worst Negative Slack"* de 2,735ns O que nos permitiria encurtar o período tornando a máquina mais responsiva. Porém, esta alteração nao seria notada podendo até ser considerado um *"underclock"* para poupar energia caso fosse necessário.

De seguida, foram realizados testes na placa Basys3 de modo a confirmar os resultados obtidos na simulação. A implementação funcionou como era esperado, comprovando-se assim a prontidão da calculadora desenvolvida para uma situação real. No entanto, a interface desenvolvida apresentou-se diferente da proposta: o valor era apenas guardado nos registos após sair do respetivo estado de load, e não ao entrar no dito estado. Consequentemente, a implementação apresenta um estado redundante (*Init*) que facilmente podia ser removido, adaptando assim a máquina de estados de forma mais eficiente.

4 Conclusões

Concluindo, foram obtidos resultados bastante satisfatórios no desenvolvimento da calculadora. A realização deste projeto facilitou a familiarização com o ambiente de desenvolvimento HDL do Vivado e a aclimatização às ferramentas disponibilizadas neste. Surgiram algumas dificuldades no desenvolvimento, mas foram resolvidas com relativa facilidade, e contribuiram também para a familiarização com as ferramentas disponibilizadas. Apesar da interface apresentada não corresponder à especificação, esta podia também facilmente ser alterada de modo a se tornar mais eficiente e intuitiva. A *performance* da calculadora corresponde também ao esperado de um sistema tão simples, sendo bastante rápida. Alterações no ciclo de relógio seriam insignificantes para um projeto desta escala.