Dispositivos Lógicos Programáveis II

Relatório 01 - Caminho crítico dos operadores

Arthur Cadore Matuella Barcella

12 de Março de 2024

Sumário

1 Descrição da atividade

1.1 Faça a implementação de um somador para valores de 16 bits sem sinal

Escreva 5 implementações em VHDL para as operações abaixo. Sintetize usando o Quartus e o dispositivo da DE2-115.

- a) a+b
- b) a + "0000000000000001"
- c) a + "0000000010000000"
- d) a + "1000000000000000"
- e) a + "1010101010101010"

1.2 Verifique a área (LE) e atraso (ns) para cada implementação

1.3 Discussão

- Qual implementação usou menos área? Por quê? Como o delay se comportou?
- Comente as diferenças entre os valores de área e delay obtidos nas operações a), b) e c).

2 Faça a implementação de um somador para valores de 16 bits sem sinal:

Para realizar a implementação criei o código demonstrado abaixo em VHDL, para realizar os cinco diferentes testes, basta comentar cada caso de uso, e descomentar o seguinte, no código abaixo, o caso dois está sendo utilizado.

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4
5  entity projeto1 is
6      port (
7          -- Input ports
8          a    : in  std_logic_vector(15 downto 0);
9          b    : in  std_logic_vector(15 downto 0);
10
11         -- Output ports
12         s    : out  std_logic_vector(15 downto 0)
13     );
14 end entity projeto1;
15
16 architecture v1 of projeto1 is
17 begin
18
19 -- Caso 1:
20 --    s <= std_logic_vector(unsigned(a) + unsigned(b));
21 -- Caso 2:
22    s <= std_logic_vector(unsigned(a) + "0000000000000001");
23 -- Caso 3:
24 --    s <= std_logic_vector(unsigned(a) + "0000000010000000");
25 -- Caso 4:
26 --    s <= std_logic_vector(unsigned(a) + "1000000000000000");
```

```
27 -- Caso 5:
28 --    s <= std_logic_vector(unsigned(a) + "1010101010101010");
29 end v1;
```

2.1 Verifique a área (LE) e atraso (ns) para cada implementação:

Os valores obtidos durante o uso do código estão descritos abaixo, para cada caso foi amostrado o tempo de delay total e o número de elementos necessários para compor o circuito.

N° do Caso	Atraso (ns)	N° de Elementos
1	11.026	16
2	10.452	16
3	14.245	8
4	9.954	0
5	10.641	15

Observação: No caso 4, é utilizado na verdade um componente lógico (imbutido no bloco de I/O), mas como não foi consumido nenhum circuito combinacional customizado da FPGA, o valor foi dito como zero (mais detalhes estão na sessão a seguir).

2.2 Discussão:

2.3 Qual implementação usou menos área? Por quê? Como o delay se comportou?

O menor valor de delay e ocupação de área obtido foi na operação "d", isso ocorreu por dois motivos:

Na questão "d", a configuração de soma entre a entrada "a"e a sequência binária estática é de apenas 1 bit, ou seja, apenas 1 bit será acrecentado ao valor de "a", porem, esse 1 bit já está na ultima casa do vetor de 16 bits, desta forma, o carry não se propagará para outras casas, permitindo que o vetor seja somado no próprio tratamento de I/O da placa. Isso ocorre pois na entrada de I/O, um circuito de inversão pode ser utilizado na entrada do bit antes de propaga-lo para os circuitos combinacionais da FPGA.

Como a soma do ultimo bit (mais significativo) sem olhar para o carry, se comporta como um circuito inversor, podemos utiliza-lo para resolver esse problema, tornando o tratamento mais rápido pois ocorre na própria entrada da FPGA.

Essa configuração permite atingir o menor tempo, mas também a menor área, visto que neste caso, nenhum circuito combinacional é exigido, pois o único tratamento no vetor "a"está na própria entrada, tornando essa questão a que consome o menor tempo de propagação e também a menor área de consumo nos circuitos da FPGA.

2.4 Comente as diferenças entre os valores de área e delay obtidos nas operações a), b) e c)

A principal diferença entre essa estrutura e as demais é a quantidade e posição de bits a serem somados com a entrada "a".

Ao analisarmos a questão "a"com a questão "b"por exemplo, podemos concluir que o circuito da "a"precisa ser maior tanto em espaço quanto em delay, isso devido ao circuito necessitar que todos os 16bits sejam somados, e portanto, é necessário seguir uma ordem para garantir que o carry seja

propagado corretamente do primeiro ao ultimo bit, dessa forma, aumentando o delay de progagação.

Agora ao compararmos a questão "c"com a questão "d", podemos notar que o número de bits "1"a serem somados é o mesmo, entretanto, a posição neste caso contribui para que o circuito da "c"seja maior. Ao somar um bit que está na posição "8"do vetor por exemplo, é necessário que todos os demais bits sejam verificados até chegar no bit "15". Isso ocorre pois ao realizar uma soma no meio do vetor, é necessário validar se todas as demais posições posteriores do vetor terão acressimo devido ao carry da soma no valor "8".

Assim, o circuito da questão "c"por mais que tenha a principio apenas 1 bit para ser somado, a posição deste bit irá tornar o circuito maior.

O mesmo ocorre para a questão "b", onde todos os bits serão validados, tornando-a igual a questão "a". Já para a questão "e", como o primeiro bit do vetor será sempre "0", a soma entre a entrada e a string pode ser reduzido para 15 casas não 16, tornando o circuito um pouco mais otimizado.

3 Referências

Slides da primeira aula DLP2