



**INSTITUTO
FEDERAL**

Santa Catarina

Câmpus
São José

Relatorio 01

DISPOSITIVOS LÓGICOS PROGRAMÁVEIS II (DLP029007)

Rhenzo Hideki Silva Kajikawa

20 de Setembro de 2023

Sumário

1. Caminho crítico dos operadores	3
2. Resultados	3
2.1. Qual implementação usou menos área? Por quê? Como o delay se comportou?	
3	
2.2. Comente as diferenças entre os valores de área e delay obtidos nas operações a), b) e c)	3

1. Caminho crítico dos operadores

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) $a + \text{"0000000000000001"}$

c) $a + \text{"0000000010000000"}$

d) $a + \text{"1000000000000000"}$

e) $a + \text{"1010101010101010"}$

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

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. Resultados

	Área(LE)	Delay(ns)
a) $a+b$	16 / 114,480 (< 1 %)	9.24
b) $a + \text{"0000000000000001"}$	15 / 114,480 (< 1 %)	6.966
c) $a + \text{"0000000010000000"}$	8 / 114,480 (< 1 %)	5.803
d) $a + \text{"1000000000000000"}$	0 / 114,480 (0 %)	3.455
e) $a + \text{"1010101010101010"}$	14 / 114,480 (< 1 %)	7.515

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

A implementação com menos área e delay é a implementação *D*. Isso ocorre pois ela é uma implementação que pode acontecer com somadores localizados nos buffers de IO da placa. Por esse motivo , não aparece no relatório gerado pelo Quartus nenhum elemento lógico sendo utilizado e também por não depender de nenhum somador seu delay acaba sendo baixo.

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

A diferença das operações de *A*, *B* e *C* se devem ao fato das diferentes posições possíveis com o valor de bit 1. A operação *A* tem a maior área pois é necessária garantir que a soma cubra todos os valores de bits. A operação *B* tem o valor de *b* fixo , dessa forma foi utilizado 1 somador a menos. Por fim a operação *c* como os bits que precedem

o 1 são 0 não irão gerar nenhum Carry para a soma então é apenas necessário garantir somadores a partir do bit 1.