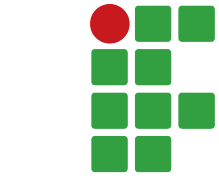


INSTITUTO FEDERAL
Santa Catarina

Câmpus
São José

AE1

Conhecendo os dispositivos lógicos programáveis

Curso: Engenharia de Telecomunicações
Disciplina: ELD129003 - Eletrônica Digital 2
Professor: Marcos Moecke

Aluna
Luiza Kuze Gomes

02 de março de 2024

Sumário

1 Introdução

Podemos dividir a atividade em três grandes partes: Análise comparativa, tempos de propagação e simulação funcional.

A análise comparativa consiste em analisar a estrutura interna dos dispositivos lógicos programáveis ao realizar a comparação entre duas famílias de dispositivos diferentes: Max II e Cyclone IV E. Para a família Max II, foi selecionado o dispositivo com código "EPM240F100C4". Já para a família Cyclone IV E, o código é "EP4CE6E22C7". Inicialmente é inserido um circuito para programação e feita uma análise da estrutura interna, abrindo o Chip Planner e Node Properties. Ao final, observamos o Pin Planner e o circuito através do RTL Viewer e Technology Map.

A parte dos tempos de propagação consite em realizar medições dos tempos de propagação em circuitos combinacionais. É inserido um código em VHDL, diferentes configurações no compilador e medições para verificar os tempos de propagação, analisando os diferentes caminhos críticos a dependerem dessas configurações.

Nessas partes da atividade, foram feitas análises do tempo de compilação no Quartus II, verificação do número de elementos lógicos e número de pinos.

A simulação funcional é a parte da atividade que consiste em efetuar a simulação do código em VHDL no software ModelSim, fazendo um arquivo com extensão .do para executar o script e outro arquivo Wave.do para configurar a apresentação da simulação.

2 Análise Comparativa

É a primeira grande parte da atividade e nela realizei uma análise das diferenças entre as famílias de dispositivos Max II e Cyclone IV E, de acordo com os passos 1 e 2 descritos na Wiki da disciplina.

2.1 Circuito para programação

Essa etapa segue igual para ambos os dispositivos, uma vez que é apenas inserir o circuito para programação. O circuito colocado em estudo é um contador 74161.

É importante destacar que encontrei 3 warnings após compilar o projeto. Esses warnings foram discutidos em aula, então comentarei brevemente sobre cada um deles.

O primeiro aviso é decorrente a falta do procedimento de pinagem, não definindo os pinos de entrada e saída. Como a atividade não previa essa ação, não temos que nos preocupar com isso. O warning tem a seguinte sentença: *"Critical Warning (169085): No exact pin location assignment(s) for 14 pins of 14 total pins. For the list of pins please refer to the I/O Assignment Warnings table in the fitter report."*.

O segundo aviso é referente a falta de um arquivo .sdc no projeto. Esses arquivos .sdc definem restrições de tempo para otimizar o processo de compilação e podem, por exemplo, melhorar o desempenho. Nessa etapa, também não iremos nos preocupar com isso. A sentença do warning: *"Critical Warning (332012): Synopsys Design Constraints File file not found: Ckt74161.sdc. A Synopsys Design Constraints File is required by the Timing Analyzer to get proper timing constraints. Without it, the Compiler will not properly optimize the design."*

O terceiro e último aviso é algo já esperado. Por padrão, o Quartus II assume um clock de 1 GHz como requisito. Porém, para dispositivos mais antigos, nesse caso a família MAX II, um clock de 1 GHz é incompatível. A sentença do warning: *"Critical Warning (332148): Timing requirements not met"*.

2.2 Observando a estrutura interna do PLD

Aqui foram utilizadas duas ferramentas para observação: Chip Planner e Node Properties.

2.2.1 Chip Planner

O Chip Planner trás uma visão dos tipos de recurso presentes, informando cada funcionalidade com uma respectiva cor.

A diferença entre a quantidade de recursos disponíveis em cada um é nítida. Comparando as figuras 1 e 2, podemos visualizar que o dispositivo Max II apresenta menos áreas de divisões de recursos em relação ao dispositivo da família Cyclone IV E.

A quantidade de áreas em azul (Logic Array Blocks) foi o que mais me chamou atenção, pois significa que a Cyclone IV E pode ser utilizada para desenvolver projetos com mais funcionalidades do que a Max II.

É possível visualizar onde o circuito anteriormente inserido para programação está localizado nos dispositivos, ele é representado em vermelho.

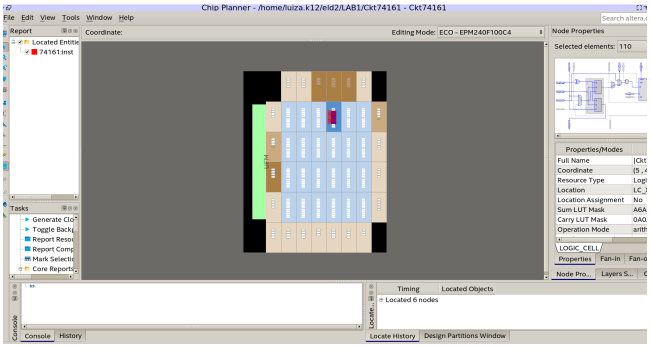


Figura 1: Chip Planner - Max II

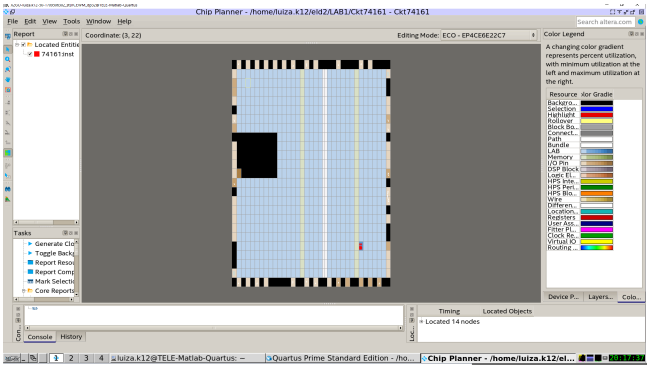


Figura 2: Chip Planner - Cyclone IV E

2.2.2 Node Properties

O Node Properties possibilita contemplar o circuito de um elemento lógico com mais detalhes, destacando os componentes em seu interior.

Durante esse procedimento tive dificuldade de interpretar os resultados e comparações em cada um dos dispositivos.

Nas figuras 3 e 4, a princípio encontrei somente a diferença entre alguns dos terminais de conexão em azul.

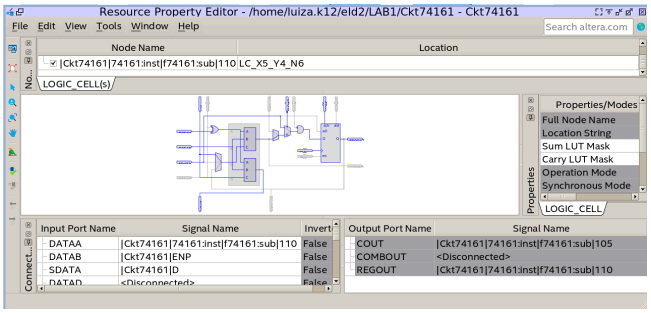


Figura 3: Circuito do elemento lógico - Max II

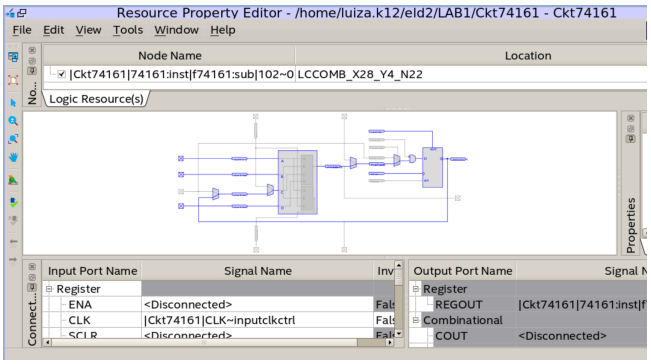


Figura 4: Circuito do elemento lógico - Cyclone IV E

2.3 Pin Planner

O Pin Planner é o local para distribuir os sinais de entrada e saída aos pinos dos dispositivos. Nesse procedimento, fica evidente a diferença física entre os dispositivos estudados na atividade.

A diversidade de pinos da Cyclone IV E difere da Max II. Na figura 6, a aba de tipos de símbolos dos pinos da Cyclone IV E permite aos usuários uma personalização muito superior ao dispositivo Max II na figura 5.

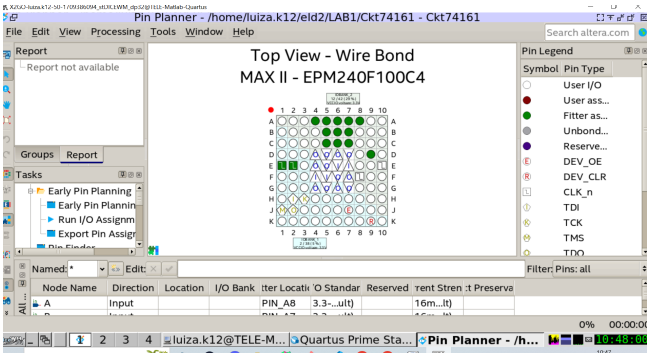


Figura 5: Pin Planner - Max II

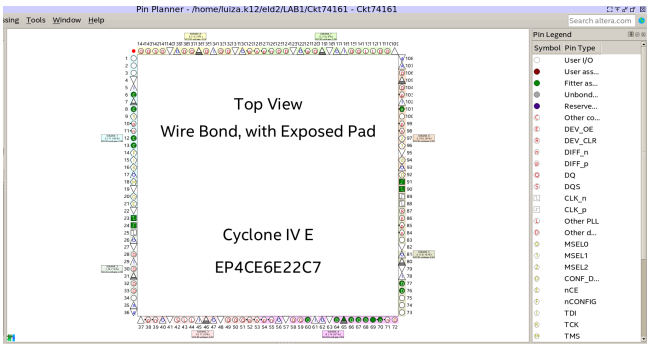


Figura 6: Pin Planner - Cyclone IV E

2.4 Tempo de processamento

O tempo de compilação de cada um dos dispositivos segue nas figuras 7 e 8, assim como os números de elementos lógicos e pinos nas figuras 9 e 10.

Em relação ao número de elementos lógicos e número de pinos utilizados, bem com o percentual em relação ao número total do dispositivo, houveram algumas mudanças as quais farei destaque.

De acordo com a figura 9, para a Max II estão sendo utilizados 6 elementos lógicos e 14 dos pinos totais, eles representam respectivamente 3% e 18% do total presente. Já para a Cyclone IV E, na figura 10, estão sendo utilizados 10 elementos lógicos e 14 dos pinos totais, eles representam respectivamente < 1% e 15% do total presente.

Outro detalhe, na figura 10, são os novos itens no painel da Cyclone IV E em relação a Max II, são eles: "Total memory bits", "Embedded Multiplier 9-bit elements"e "Total PLLs".

Task	Time
Compile Design	00:00:14
Analysis & Synthesis	00:00:08
Fitter (Place & Route)	00:00:02
Assembler (Generate programming files)	00:00:01
Timing Analysis	00:00:02
EDA Netlist Writer	00:00:01
Edit Settings	

Figura 7: Tempo de processamento - Max II

Task	Time
Compile Design	00:00:19
Analysis & Synthesis	00:00:10
Fitter (Place & Route)	00:00:04
Assembler (Generate programming files)	00:00:02
Timing Analysis	00:00:02
EDA Netlist Writer	00:00:01
Edit Settings	

Figura 8: Tempo de processamento - Cyclone IV E

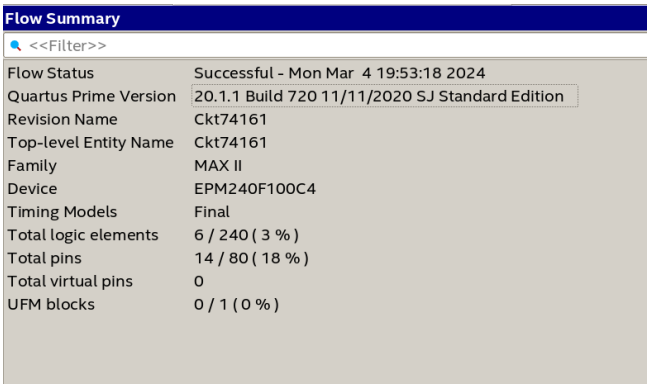


Figura 9: Relatório de compilação - Max II

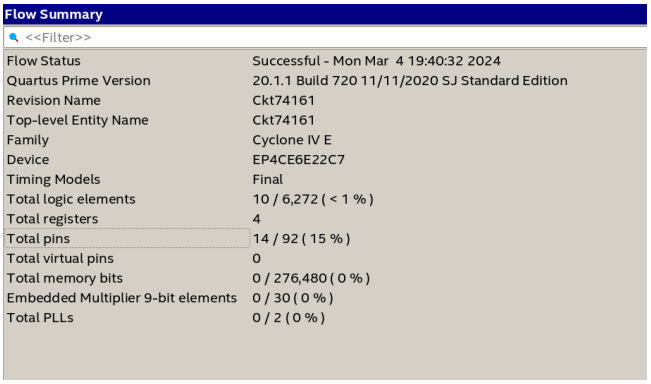


Figura 10: Relatório de compilação - Cyclone IV E

2.5 RTL Viewer

O RTL Viewer é uma forma de abrir o circuito inserido para programação e visualizar seus elementos constituintes. Ele é apresentado na figura 11 e ele se mantém igual em ambos os dispositivos do estudo.

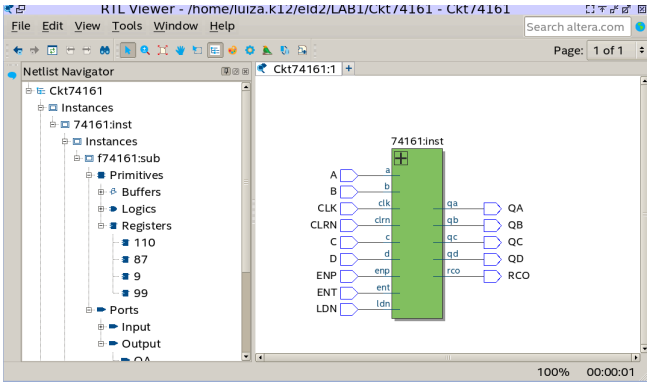


Figura 11: RTL Viewer

2.6 Technology Map

O Technology Map permite uma representação física do projeto e difere em cada dispositivo.

Seguindo a ideia das etapas anteriores, a Cyclone IV E mantém uma estrutura mais complexa que a Max II.

Uma das diferenças presentes na figura 13, Technology Map da Cyclone IV E, é a presença de novas células de lógica combinacional que não estão presentes na figura 12, Technology Map da Max II.

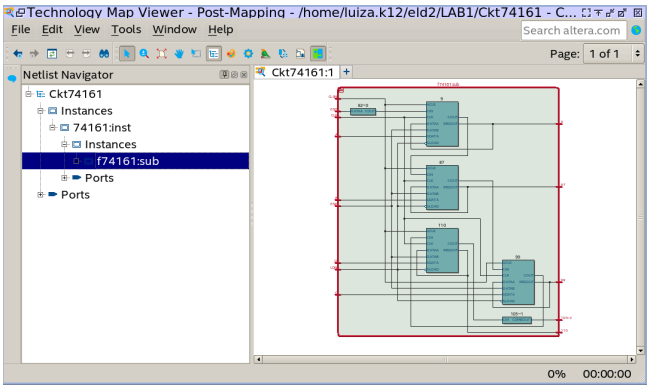


Figura 12: Technology Map - Max II

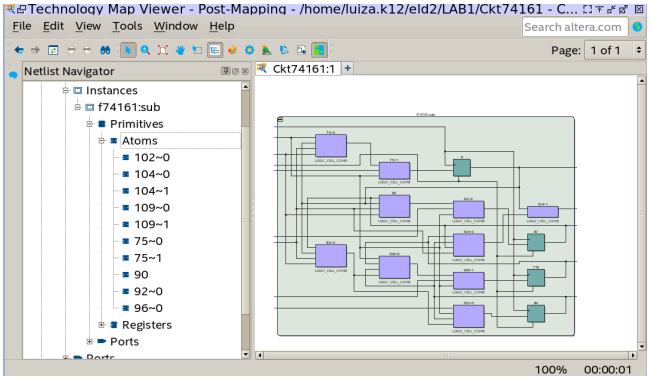


Figura 13: Technology Map - Cyclone IV E

3 Tempos de Programação

A seguir, vou proceder com o passo 3 descrito no roteiro da atividade na Wiki, o qual proprõe a medição de tempos de propagação em circuitos combinacionais. A única alteração que difere do roteiro é realização da atividade com a família Cyclone IV E com código de dispositivo "EP4CE6E22C7"e não com a família Max V, como foi comentado em aula.

3.1 Circuito exemplo

Anteriormente, inserimos um circuito de um contador 74161 para programação. Agora, vamos utilizar o código em VHDL disposto na Wiki de forma genérica para permitir o estudo de diferentes tempos de programação e análise da área do chip que está sendo utilizada.

O código representa o cálculo da distância de Hamming e está disposto abaixo:

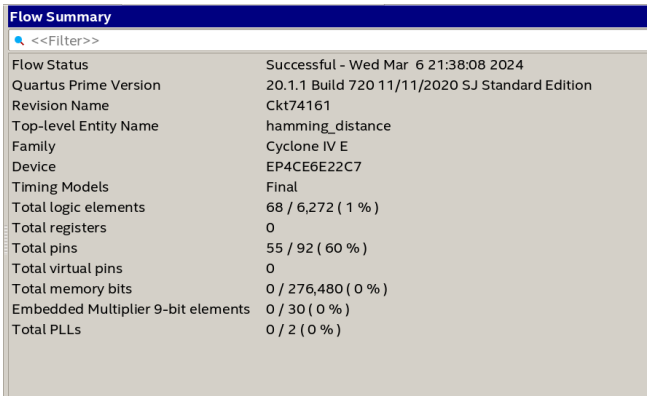
```
1 library ieee;
2 use ieee.std_logic_1164.all;
3 use ieee.numeric_std.all;
4 entity hamming_distance is
5   generic (
6     N:  natural := 25;
7     M:  natural := 5); -- M = ceil(log2(N))
8   port(
9     a, b :  in  std_logic_vector (N-1 downto 0);
```

```
10      y : out std_logic_vector (M-1 downto 0));
11  end entity;
12
13  architecture ifsc_arch_gen of hamming_distance is
14    signal diff: unsigned (N-1 downto 0);
15    signal sum: unsigned (M-1 downto 0);
16  begin
17    diff <= unsigned(a xor b);
18    process (diff)
19      variable tmp : integer range 0 to N;
20    begin
21      tmp := 0;
22      for i in diff'range loop
23        tmp := tmp + to_integer(unsigned'('0' & diff(i)));
24      end loop;
25      sum <= to_unsigned(tmp,M);
26    end process;
27    y <= std_logic_vector(sum);
28  end architecture;
```

3.2 Área ocupada no DLP

Nessa etapa, vamos verificar a área ocupada no DLP em questão ao compilar o projeto e conferir o Chip Planner.

Na figura 14, visualizamos o uso de 68 elementos lógicos que representam 1% do total. Além disso, também há o dado da quantidade de pinos totais utilizados que nesse caso são 55 dos 92 disponíveis, ou seja, 60% do total.



Flow Summary

<<Filter>>

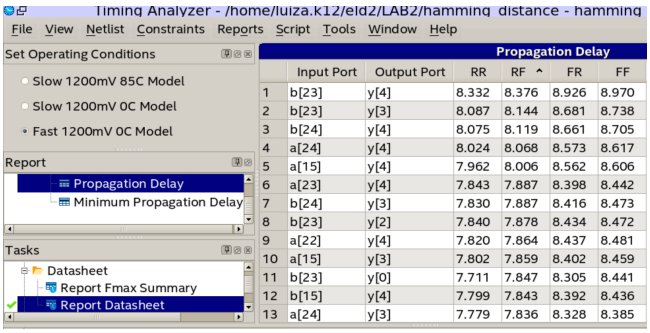
Flow Status	Successful - Wed Mar 6 21:38:08 2024
Quartus Prime Version	20.1.1 Build 720 11/11/2020 SJ Standard Edition
Revision Name	Ckt74161
Top-level Entity Name	hamming_distance
Family	Cyclone IV E
Device	EP4CE6E22C7
Timing Models	Final
Total logic elements	68 / 6,272 (1 %)
Total registers	0
Total pins	55 / 92 (60 %)
Total virtual pins	0
Total memory bits	0 / 276,480 (0 %)
Embedded Multiplier 9-bit elements	0 / 30 (0 %)
Total PLLs	0 / 2 (0 %)

Figura 14: Relatório de compilação do projeto

3.3 Propagation Delay

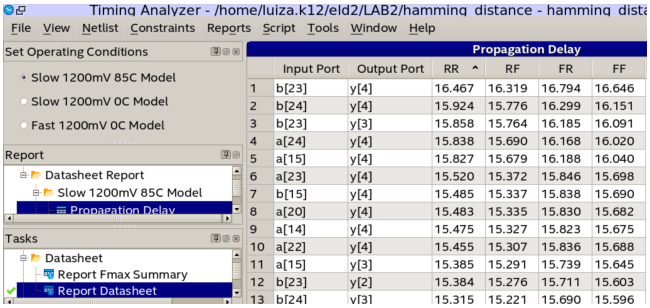
Com a ferramenta "Propagation Delay"do Quartus II, percebemos o atraso de um sinal ao percorrer o interior de um componente lógico e ver o tempo de propagação da entrada até a saída.

Utilizando esta ferramenta, é possível verificar que o caminho crítico ("Critical Path") é entre a entrada b[23] e a saída y[4]. Na figura 16, é exibido o tempo de propagação para o caminho crítico de 16.794 ns no Slow Model e, na figura 15, 8.926 ns no Fast Model. O caminho crítico representa a pior condição analisada de PVT (Processo, Voltagem e Temperatura) para o funcionamento do dispositivo.



	Input Port	Output Port	RR	RF	FR	FF
1	b[23]	y[4]	8.332	8.376	8.926	8.970
2	b[23]	y[3]	8.087	8.144	8.681	8.738
3	b[24]	y[4]	8.075	8.119	8.661	8.705
4	a[24]	y[4]	8.024	8.068	8.573	8.617
5	a[15]	y[4]	7.962	8.006	8.562	8.606
6	a[23]	y[4]	7.843	7.887	8.398	8.442
7	b[24]	y[3]	7.830	7.887	8.416	8.473
8	b[23]	y[2]	7.840	7.878	8.434	8.472
9	a[22]	y[4]	7.820	7.864	8.437	8.481
10	a[15]	y[3]	7.802	7.859	8.402	8.459
11	b[23]	y[0]	7.711	7.847	8.305	8.441
12	b[15]	y[4]	7.799	7.843	8.392	8.436
13	a[24]	y[3]	7.779	7.836	8.328	8.385

Figura 15: Tempos de Propagação Fast Model



	Input Port	Output Port	RR	RF	FR	FF
1	b[23]	y[4]	16.467	16.319	16.794	16.646
2	b[24]	y[4]	15.924	15.776	16.299	16.151
3	b[23]	y[3]	15.858	15.764	16.185	16.091
4	a[24]	y[4]	15.838	15.690	16.168	16.020
5	a[15]	y[4]	15.827	15.679	16.188	16.040
6	a[23]	y[4]	15.520	15.372	15.846	15.698
7	b[15]	y[4]	15.485	15.337	15.838	15.690
8	a[20]	y[4]	15.483	15.335	15.830	15.682
9	a[14]	y[4]	15.475	15.327	15.823	15.675
10	a[22]	y[4]	15.455	15.307	15.836	15.688
11	a[15]	y[3]	15.385	15.291	15.739	15.645
12	b[23]	y[2]	15.384	15.276	15.711	15.603
13	b[24]	y[3]	15.315	15.221	15.690	15.596

Figura 16: Tempos de Propagação Slow Model

3.4 Chip Planner

Uma outra maneira de visualizar os tempos de propagação é utilizando o Chip Planner. Nessa outra ferramenta, podemos identificar cada área colorida de elemento lógico utilizado acoplado ao seu respectivo tempo de propagação.

Selecionando um dos caminhos presentes do sinal, encontramos uma situação semelhante a figura 17 para um determinado caminho presente, a depender do seu tempo de propagação e conexões.

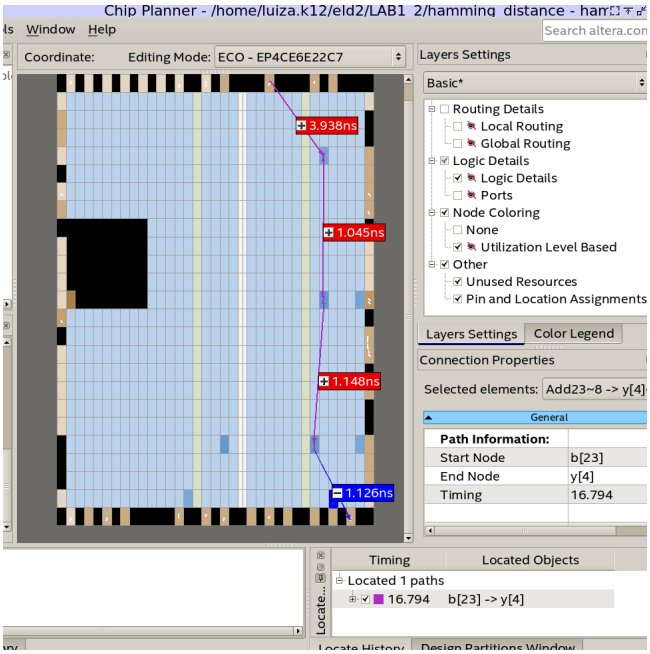


Figura 17: Tempos de Propagação - Chip Planner

Além disso, ao aumentar o zoom é possível obter um maior detalhamento de cada parte do elemento lógico, conforme a figura 18.

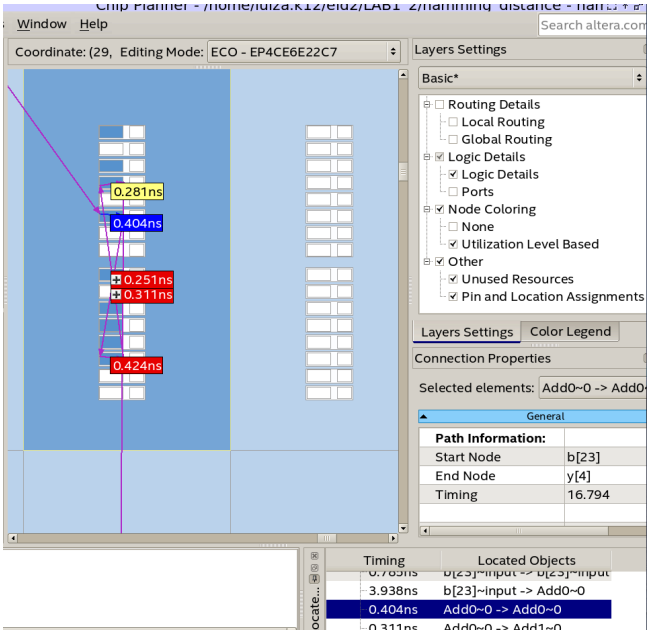


Figura 18: Detalhamento dos Tempos de Propagação - Chip Planner

3.5 Configurando o compilador

O Quartus permite que o próprio usuário faça configurações no compilador para obter melhores resultados no projeto, como por exemplo, melhorias no desempenho. Existem duas operações principais para essa melhora: A mudança de *Optimization Technique* e alteração do valor de *Fitter Initial Placement Seed*. O objetivo principal nesse momento foi de alguma forma diminuir o tempo do caminho crítico para melhorar o desempenho.

Para um melhor aproveitamento da prática, realizei a troca entre todas as opções disponíveis das técnicas de otimização. Os resultados estão descritos na tabela 1 e contém uma variância muito baixa dentre os valores de caminho crítico obtidos, pois é um projeto pequeno, ou seja, existem poucos elementos lógicos sendo utilizados.

Tabela 1: Tempos de Propagação para Caminho Crítico

Métrica	**Tempo de Propagação (ns)**	
	Slow Model	**Fast Model**
Balanced (Normal Flow)	16.794	8.926
Performance (High effort)	16.108	8.595
Performance (Aggressive)	16.108	8.595
Power (High effort)	17.627	9.686
Power (Aggressive)	16.547	8.820
Area (Aggressive)	16.794	8.970

Além disto, vale destacar que somente com essa alteração de otimização e as demais configurações anteriores do projeto, o melhor caminho crítico foi com a otimização *Power (Agressive)* em Fast Model, de acordo com a figura 19

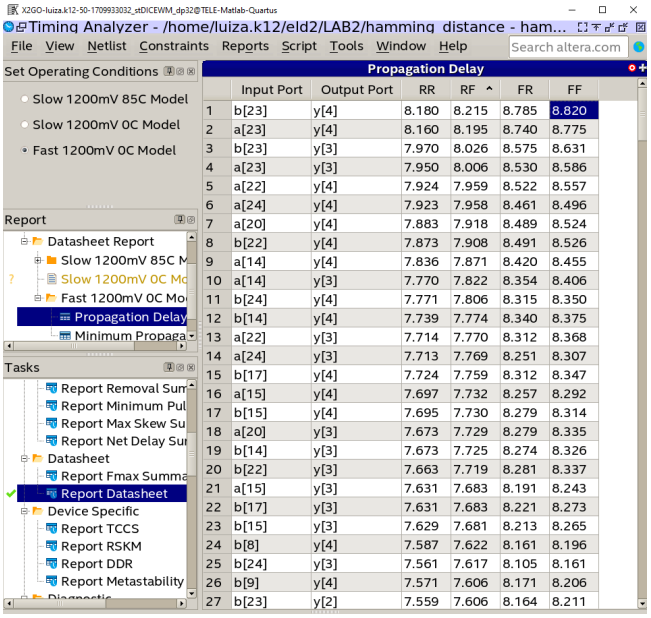


Figura 19: Tempos de Propagação - Opção Power (Aggressive)

Agora, será feita a configuração do valor de *Fitter Initial Placement Seed*, o qual a função é modificar o "sorteio"das áreas utilizadas por elementos lógicos e fazer diferentes arranjos.

Abrindo o Chip Planner, na figura 20, podemos nitidamente perceber essas mudanças ao alterar o valor do *Fitter*. Anteriormente, na figura 17, outras áreas em azul escuro estavam alocadas em uso, pois estava com um outro *Fitter*.

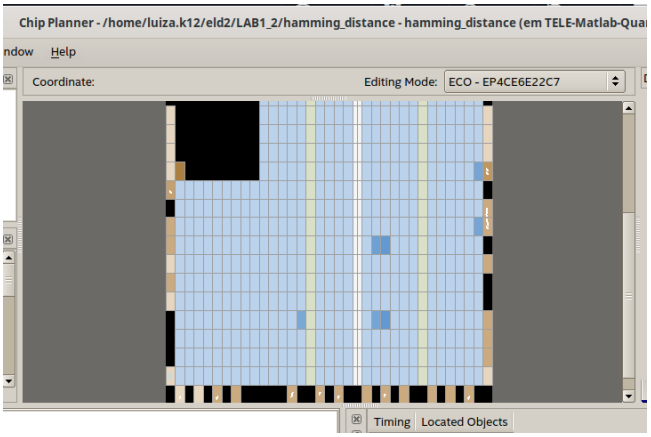


Figura 20: Chip Planner após mudança da semente inicial do Fitter

Para complementar a prática, desenvolvi a tabela 2, em que apliquei novos valores no Fitter e obtive novos caminhos críticos, mantendo otimização *Power (Agressive)* em Fast Model:

Tabela 2: Relação entre o Valor do Fitter e o Tempo de Propagação

Fitter	Caminho Crítico (ns)
1	8.820
5	8.872
10	8.929
20	9.673
30	10.072

Para prosseguir com a prática, mantive o *Fitter* em 1, já que foi o caminho crítico com menor tempo de propagação.

3.6 Atraso máximo

Uma forma de melhorar o tempo de propagação é restringir esse tempo entre entrada(s) e saída(s), colocando um arquivo .sdc.

Na atividade, foi inserida uma restrição de máximo atraso entre todas as portas de entradas para todas as portas de saída. O código foi disponibilizado inicialmente na Wiki e está apresentado abaixo:

```
1 set_max_delay -from [get_ports *] -to [get_ports *] 50
```

Realizei alguns testes referentes ao atraso máximo, os quais disponibilizei os resultados na tabela 3.

Tabela 3: Relação entre o Delay Máximo e o Tempo de Propagação

Delay Máximo	Caminho Crítico (ns)
4	6.576
9	6.562
15	7.248

Com isso, conforme a figura 21 foi possível chegar em um melhor tempo de propagação no caminho crítico com um valor de 6.562 ns, com as ferramentas de otimização *Power (Aggressive)* e Fast Model, *Fitter Initial Placement Seed* com valor 1 e atraso máximo de 9 ns.

Vale destacar que o campo "Set Operating Conditions"com a opção *"Fast 1200mV 0℃ model"* fez a maior redução de tempo no caminho crítico. Essa opção faz com que o circuito represente uma condição de operação mais rápida e em uma menor temperatura.

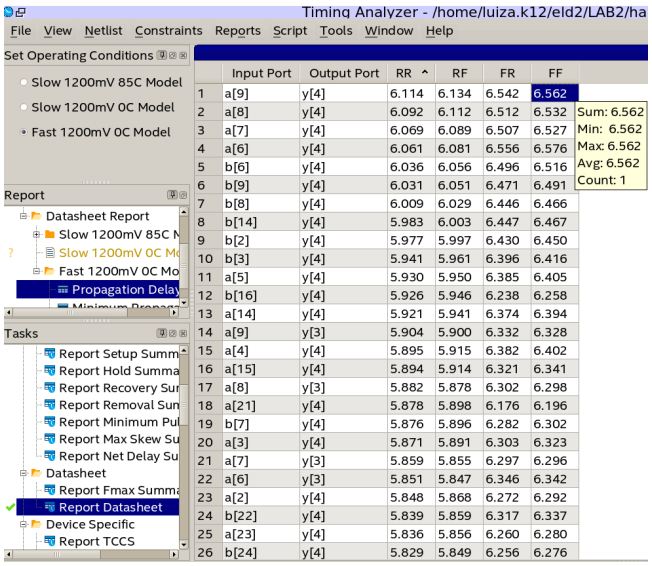


Figura 21: Tempo de Propagação inserir Delay Máximo

3.7 Detalhamento dos tempos de propagação

Vale destacar que podemos encontrar informações interessantes no Chip Planner que precisam de um pouco de mais atenção para serem visualizadas.

Setando valores para M e N no código hamming distance com M = 25 e N = 5, e max _delay = 50 ns. Vou realizar três destaques ao detalhar os tempos de propagação no Chip Planner.

Primeiro, na figura 22, verifiquei o tempo de propagação do pino de entrada até o primeiro elemento lógico que é de 0,786 ns.

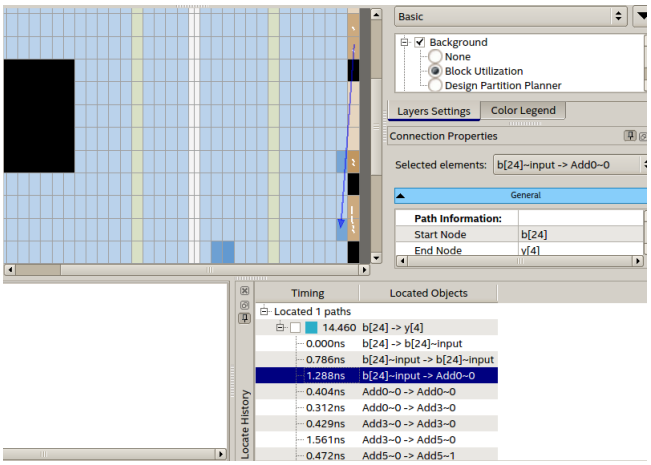


Figura 22: Tempo de propagação da entrada até o primeiro elemento lógico

Segundo, na figura 23, a soma dos tempos apresentados representa o tempo de propagação do primeiro elemento lógico até o último elemento lógico.

Figura 23: Tempo de propagação do primeiro ao último elemento lógico

Terceiro, na figura 24, o tempo de propagação do último elemento lógico até o pino de saída que é de 1,772 ns.

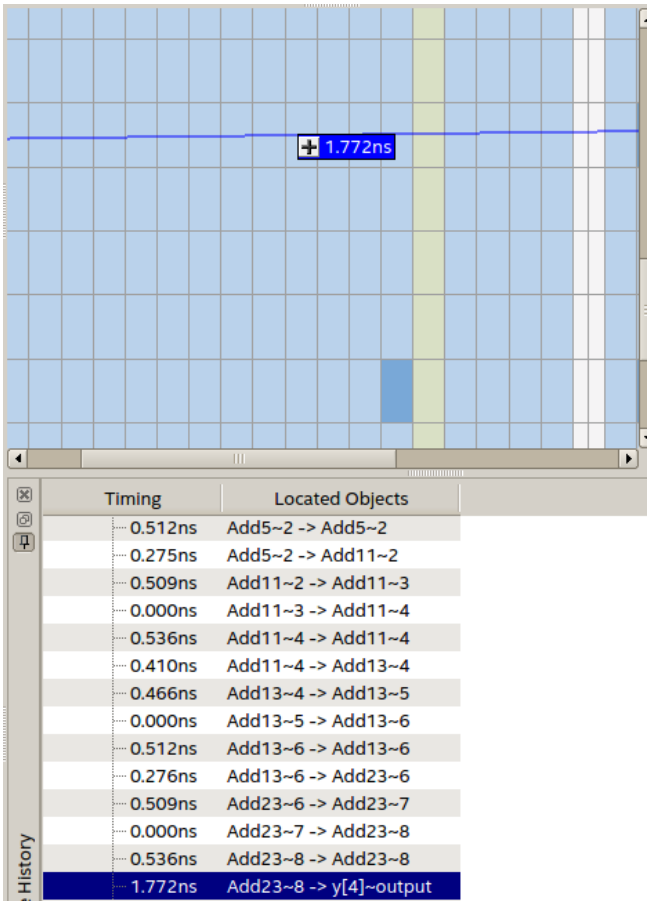


Figura 24: Tempo de propagação do último elemento lógico até a saída

4 Simulação Funcional

A simulação funcional no Modelsim é o último passo do roteiro da atividade experimental. Optei por realizar a simulação com o "Hamming Distance", código em VHDL utilizado no passo 3 da Wiki.

Na figura 25, mostro os resultados da simulação. Foi feito um arquivo chamado "tb _hamming _distance.do"para automatizar a simulação e o arquivo "Wave.do"para guardar as configurações do sinal no simulador.

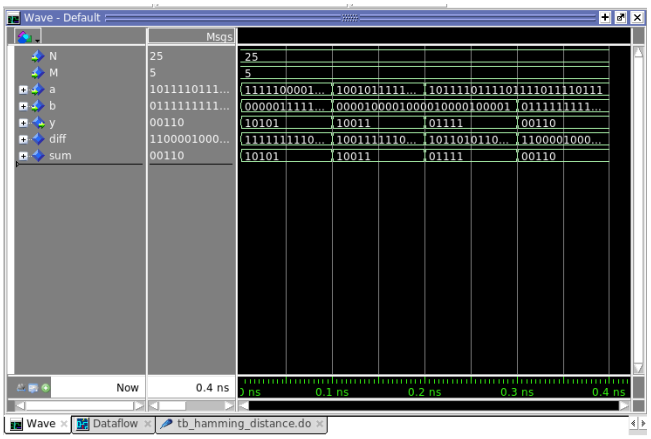


Figura 25: Simulação no Modelsim

5 Conclusão

Portanto, os dispositivos dispositivos lógicos programáveis (DLP) de diferentes famílias possuem grandes diferenças internas. Dependendo da família, um dispositivo pode proporcionar a um projeto um novo grau de complexidade e, além disso, pode ser um grande limitador de recursos ou não, e, consequentemente, nas implementações do usuário também.

A família Max II mostrou ser mais compacta ao dispor de uma quantidade menor de tipos de recursos para desenvolvimento. Uma vantagem que pode ser colocada é a simplicidade de trabalhar com ela com projetos pequenos, já que ela trabalhou minimamente mais veloz que a Cyclone IV E. Entretanto, trabalhar minimamente veloz pode não compensar o custo da falta de bons novos recursos que estão presentes na outra família.

Já a família Cyclone IV E oferece uma capacidade de lógica muito maior, sendo mais robusta e com uma maior quantidade de recursos. A vantagem dessa família são as novas possibilidades na implementação que se abrem ao dispor de novos recursos. Como os tempos de compilação foram extremamente próximos em ambos os dispositivos, o mínimo tempo a mais que a Cyclone IV E apresentou no processo de compilação pode ser facilmente compensado com suas novas funcionalidades.

Conclui também que as configurações no compilador são extremamente importantes para o desempenho, área e potência em um projeto. Esses ajustes consistem em um refinamento proporcionado pelo próprio usuário e que possibilitam uma grande redução no tempo de propagação.