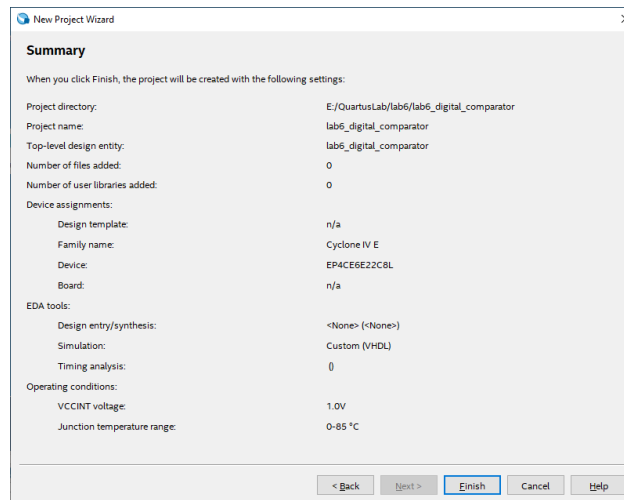


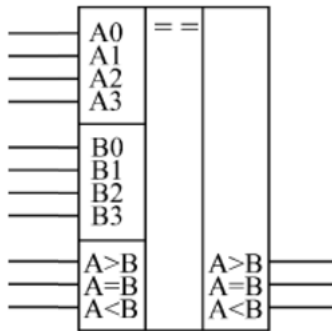
# Лабораторная работа 6 (Цифровой компаратор)

☒ Done ☐

1. Создаем пустой проект с такими же параметрами как и лабораторная №1



2. Выбираю свой вариант



3. Создаем Verilog HDL File, Block Diagram/Schematic File, VHDL File и в разделе Verification/Debugging File выбираем University Program VWF и заполняем их кодом

```
module comparator (  
    input A0, A1, A2, A3,      // 4 разделенных входа A  
    input B0, B1, B2, B3,      // 4 разделенных входа B  
    input GT_in,               // Управляющий вход: A > B предыдущего разряда  
    input LT_in,               // Управляющий вход: A < B предыдущего разряда  
    input EQ_in,               // Управляющий вход: A = B предыдущего разряда  
    output reg GT_out,          // Результат: A > B  
    output reg LT_out,          // Результат: A < B  
    output reg EQ_out           // Результат: A = B  
);
```

```

always @(*) begin
    // Сборка значений A и B из отдельных входов
    reg [3:0] A, B;
    A = {A3, A2, A1, A0};
    B = {B3, B2, B1, B0};

    // Условие равенства
    if (EQ_in && A == B) begin
        GT_out = 0;
        LT_out = 0;
        EQ_out = 1;
    end
    // Условие A > B
    else if ((GT_in || EQ_in) && A > B) begin
        GT_out = 1;
        LT_out = 0;
        EQ_out = 0;
    end
    // Условие A < B
    else if ((LT_in || EQ_in) && A < B) begin
        GT_out = 0;
        LT_out = 1;
        EQ_out = 0;
    end
    // По умолчанию
    else begin
        GT_out = 0;
        LT_out = 0;
        EQ_out = 0;
    end
end
endmodule

```

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity comparator_VHDL is
    Port (
        A0, A1, A2, A3 : in STD_LOGIC;  -- 4 отдельных входа A
        B0, B1, B2, B3 : in STD_LOGIC;  -- 4 отдельных входа B
        GT_in : in STD_LOGIC;           -- Управляющий вход: A > B предыдущего разряда
        LT_in : in STD_LOGIC;           -- Управляющий вход: A < B предыдущего разряда
        EQ_in : in STD_LOGIC;           -- Управляющий вход: A = B предыдущего разряда
        GT_out : out STD_LOGIC;          -- Результат: A > B
        LT_out : out STD_LOGIC;          -- Результат: A < B
        EQ_out : out STD_LOGIC;          -- Результат: A = B
    );
end comparator_VHDL;

architecture Behavioral of comparator_VHDL is
begin
    process (A0, A1, A2, A3, B0, B1, B2, B3, GT_in, LT_in, EQ_in)
        variable A : STD_LOGIC_VECTOR(3 downto 0);

```

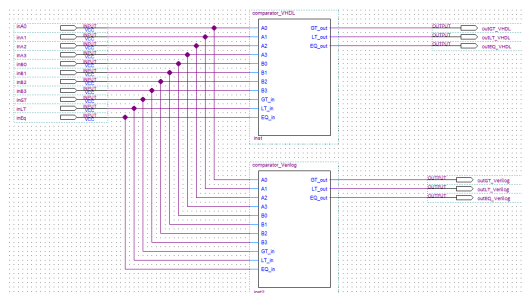
```

variable B : STD_LOGIC_VECTOR(3 downto 0);
begin
  -- Сборка значений A и B из отдельных входов
  A := A3 & A2 & A1 & A0;
  B := B3 & B2 & B1 & B0;

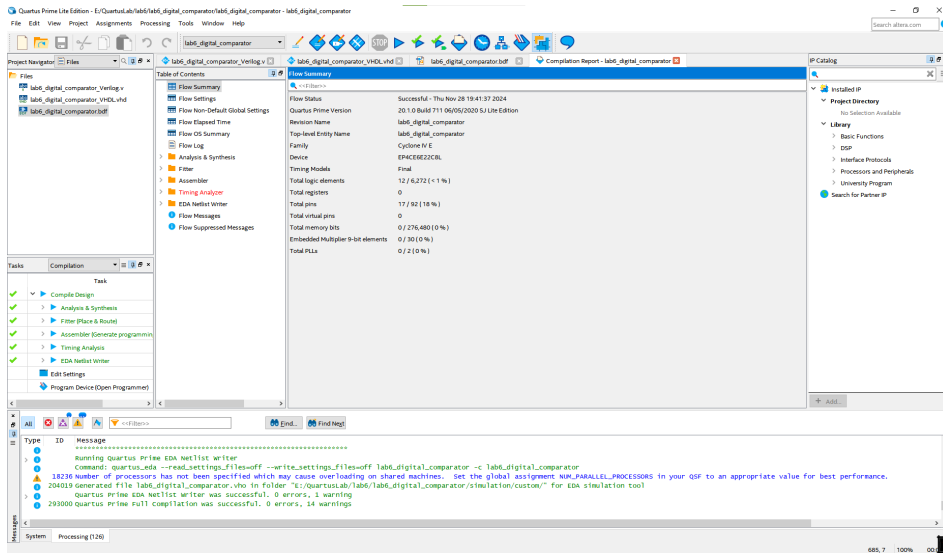
  -- Условие равенства
  if (EQ_in = '1' and A = B) then
    GT_out <= '0';
    LT_out <= '0';
    EQ_out <= '1';
  -- Условие A > B
  elsif ((GT_in = '1' or EQ_in = '1') and A > B) then
    GT_out <= '1';
    LT_out <= '0';
    EQ_out <= '0';
  -- Условие A < B
  elsif ((LT_in = '1' or EQ_in = '1') and A < B) then
    GT_out <= '0';
    LT_out <= '1';
    EQ_out <= '0';
  else
    -- По умолчанию
    GT_out <= '0';
    LT_out <= '0';
    EQ_out <= '0';
  end if;
end process;
end Behavioral;

```

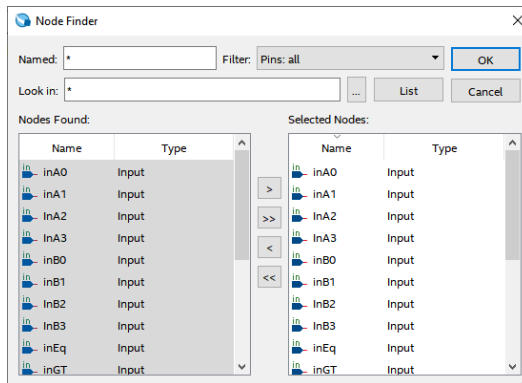
#### 4. Компилируем их и добавляем на Block файл



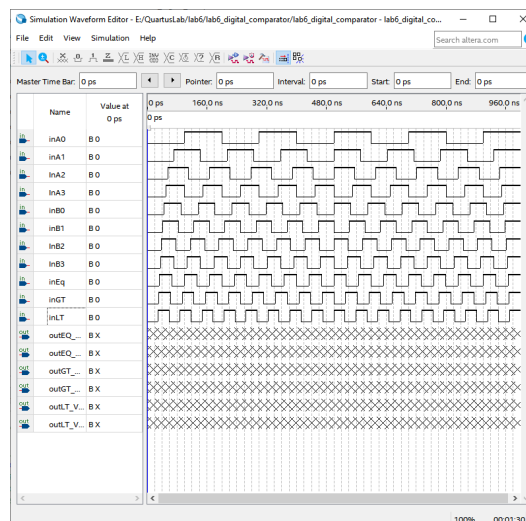
#### 5. Отправляем нашу схему на верхний уровень и запускаем компиляцию проекта, дожидаясь успешного завершения.



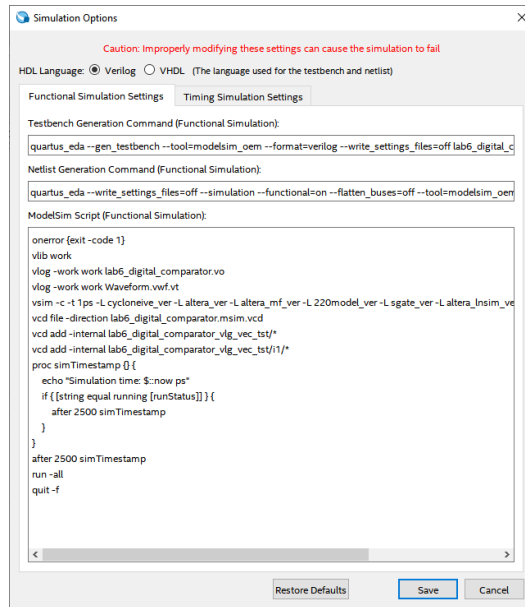
6. На странице добавления узлов в модуляцию ищем все наши узлы и добавляем их



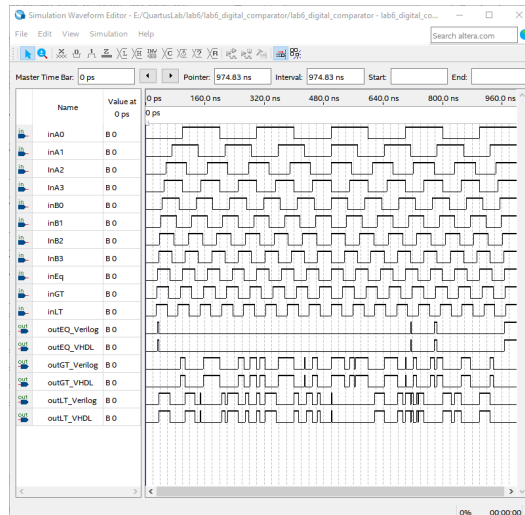
8. Для каждого узла выставляем разную частоту от 15 до 5 MHz



9. Настраиваем параметры симуляции и запускаем ее



## 9. Проверяем результат



Входы сравниваемых кодов				Входы наращивания			Выходы
A3,B3	A2,B2	A1,B1	A0,B0	A>B	A<B	A=B	A>B
A3>B3	X	X	X	X	X	X	1
A3<B3	X	X	X	X	X	X	0
A3=B3	A2>B2	X	X	X	X	X	1
A3=B3	A2<B2	X	X	X	X	X	0
A3=B3	A2=B2	A1>B1	X	X	X	X	1
A3=B3	A2=B2	A1<B1	X	X	X	X	0
A3=B3	A2=B2	A1=B1	A0>B0	X	X	X	1
A3=B3	A2=B2	A1=B1	A0<B0	X	X	X	0

Входы сравниваемых кодов				Входы наращивания			Выходы
A3=B3	A2=B2	A1=B1	A0=B0	1	0	0	1
A3=B3	A2=B2	A1=B1	A0=B0	0	1	0	0
A3=B3	A2=B2	A1=B1	A0=B0	X	X	1	0
A3=B3	A2=B2	A1=B1	A0=B0	1	1	0	0
A3=B3	A2=B2	A1=B1	A0=B0	0	0	0	1