VINÍCIUS DE BARROS SILVA 10335913

Aula 2:

g) É necessário criar uma biblioteca (subdiretório) de trabalho no contexto do Modelsim. Esta
será denominada work. Selecione a opção File -> New -> Library. Na caixa de diálogo,
selecione “new library and a logical mapping to it” e digite work em library name e
work em library physical name. Novamente, observe o que acontece na janela
"Transcript".
Pergunta: qual é o comando-linha correspondente?

-# reading C:\modeltech64_10.1d\win64/../modelsim.ini
vlib work
vmap work work
Copying C:\modeltech64_10.1d\win64/../modelsim.ini to modelsim.ini
Modifying modelsim.ini
** Warning: Copied C:\modeltech64_10.1d\win64/../modelsim.ini to modelsim.ini.
Updated modelsim.ini.

h) Selecione o arquivo full_adder_1.vhd e clique no botão Compile.
Clique em Done.
Observação: Devem aparecer na janela "Transcript" as mensagens a seguir, indicando a
correta compilação sem erros:
a. # -- Compiling entity
b. # -- Compiling architecture
c. # Errors: 0, Warnings: 0
Pergunta: qual é o comando-linha correspondente?

vcom -reportprogress 300 -work work
C:/Users/Vinícius/Desktop/USP/PSI3451_wang/aula_2/fa_1/full_adder_1.vhd
Model Technology ModelSim SE-64 vcom 10.1d Compiler 2012.11 Nov 1 2012
-- Loading package STANDARD
-- Loading package TEXTIO
-- Loading package std_logic_1164
-- Compiling entity full_adder_1
-- Compiling architecture dataflow of full_adder_1

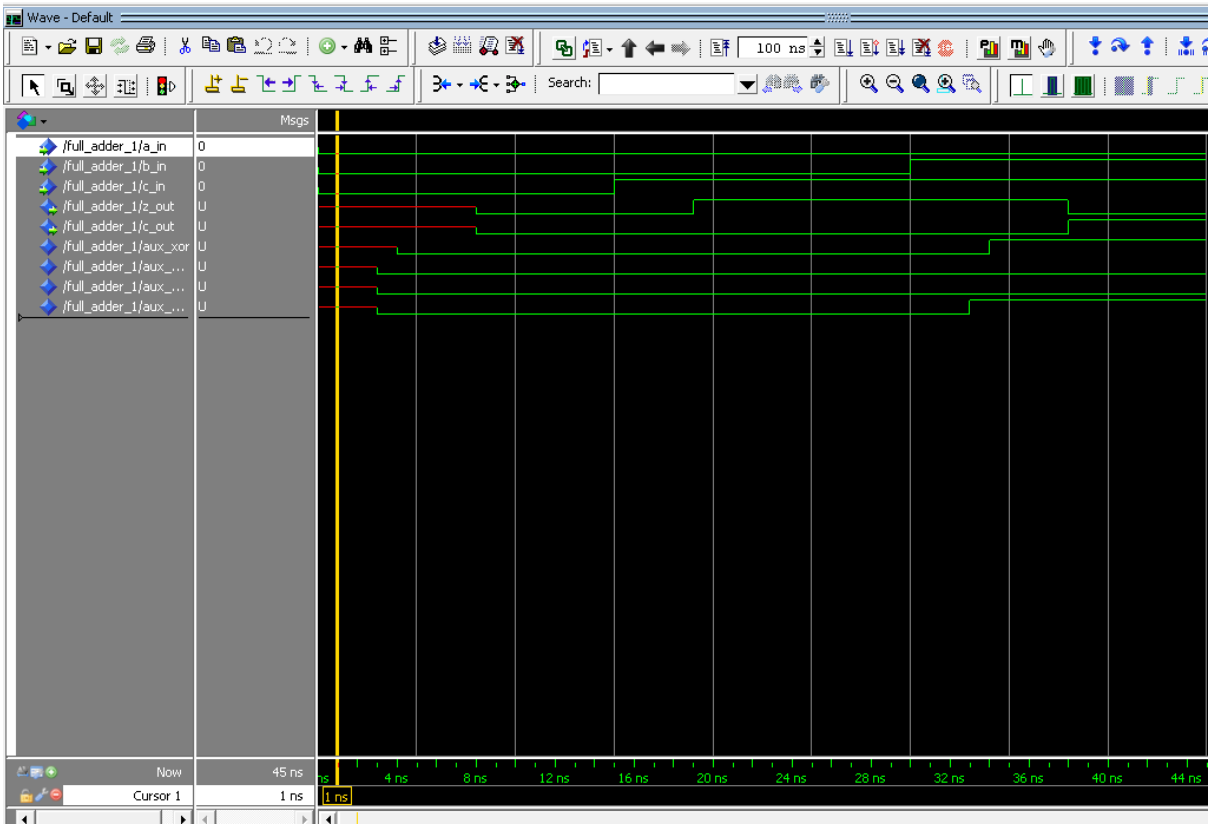
k) Selecione a aba Visibility → Apply full visibility to all modules (full debug mode) →
OK. Carregue a simulação clicando novamente em OK na janela Start Simulation.
Pergunta: qual é o comando-linha correspondente?

vsim -gui -voptargs=+acc work.full_adder_1
vsim -gui -voptargs=+acc work.full_adder_1
** Note: (vsim-3812) Design is being optimized...
#
Loading std.standard
Loading std.textio(body)
Loading ieee.std_logic_1164(body)
Loading work.full_adder_1(dataflow)#1

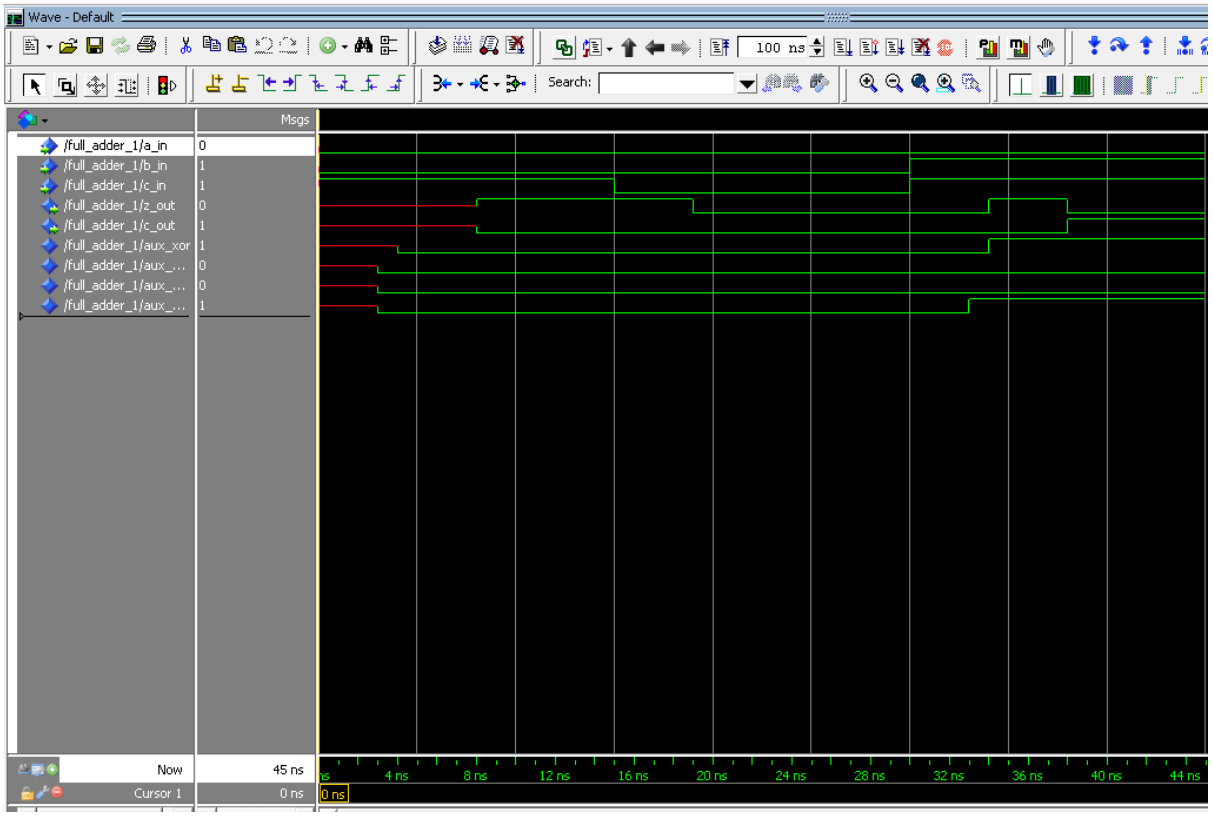
Pergunta: como é o comportamento do circuito em termos de tempos de atraso? De que
forma são as dependências entre as operações lógicas na descrição VHDL e como os atrasos
são transmitidos?
Item 1) Captura e simulação do somador completo full_adder_1 (com atrasos inerciais)

Os atrasos foram ditados na descrição VHDL, para and2 2ns para or3 4ns e para xor 4ns. Assim, o circuito para z_out apresenta um atraso de 8ns devido a duas portas xor com atraso 4ns,o c_out também apresenta 8ns de atraso, 3ns devido aos and2 e 5ns do or3. A dependência é :
z_out fica pronto após o sinal aux_xor (4ns) estar pronto e passar pela xor2 (4ns).
c_out fica pronto após os sinais aux_ands (3ns) ficarem prontos paralelamente e passarem pela porta or3 (5ns).
O atraso é propagado em série nas portas lógicas ligadas em série.

Hamming 1



Hamming 2

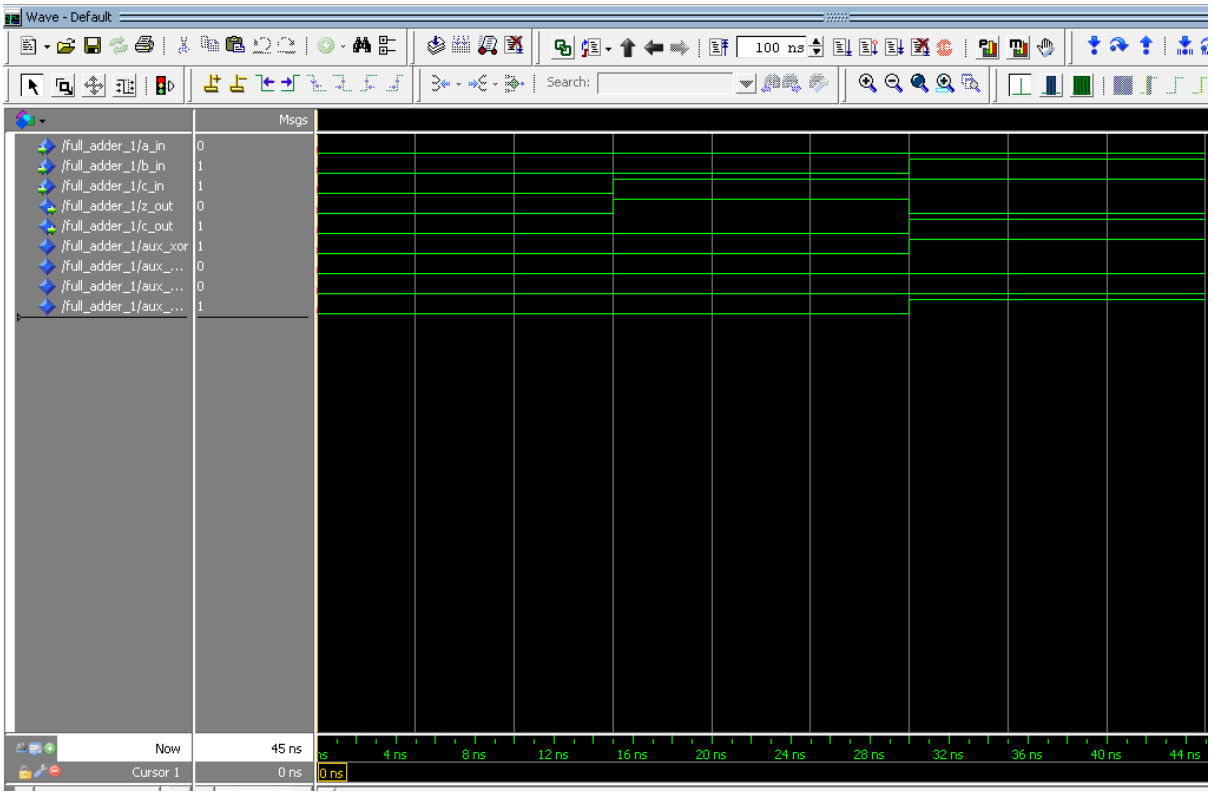


Pergunta: como é o novo comportamento do circuito em termos de tempos de atraso? De
que forma as dependências entre as operações lógicas na descrição influenciam os
atrasos? Confirme que a simulação de atraso DELTA corresponde à simulação do item
1), porém com os tempos de atraso retraídos.

Item 2) Captura e simulação do somador completo full_adder_1 com atraso delta

a) Fazer upload de curva de simulação com as transições de distância de Hamming 1
(jpg ou bmp). Anotar as dependências nas transições de atualização dos sinais z_out,
c_out, aux_and_1, aux_and_2, aux_and_3 e aux_xor (estudar nas expressões como a
transição de um sinal afeta um outro).

Dessa vez praticamente todas as portas responderam imediatamente em 0ns, e aparentemente não houve propagação do atraso, isso se deve ao fato de quando o atraso não é especificado o simulador coloca um atraso infinitesimal, esse atraso infinitesimal pode ser observado em vermelho no início da simulação. Também não aparece a propagação do atraso pois a multiplicação de infinitesimal por um número é infinitesimal.

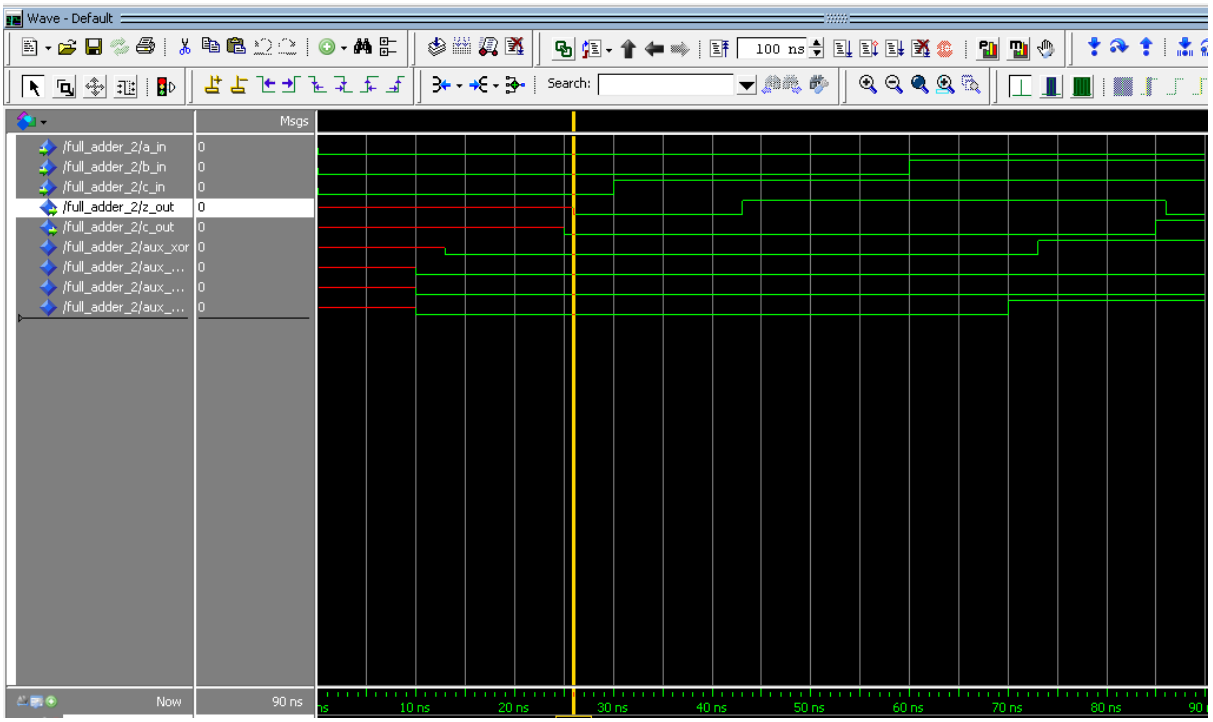


d) Guarde os resultados do Wave para comparação.
Perguntas: seguindo as recomendações, a simulação mostrou o comportamento
esperado do circuito como descrito na apostila de conceitos? Quais foram os tempos
de atraso verificados para as portas lógicas? Foi o esperado? Por que?
Pergunta: como se compara os resultados com os da seção 1?
Item 3) Captura e simulação do somador completo full_adder_2 no modelo estrutural

A simulação não mostrou o comportamento apresentado na apostila de conceitos, isso se deve ao fato de as portas lógicas não terem sido instanciadas com os valores da apostila. No caso, os atrasos instanciados na descrição vhdl foram: xor2 com 13ns, and2 com 10n e or3 com 15ns. assim para z_out temos: and2 + or3 = 25ns

c_out temos: 2 x xor2 = 26ns.

Na seção 1 tinhamos atrasos menores devido ao atraso descrito no código, porém o comportamento do circuito foi o mesmo.

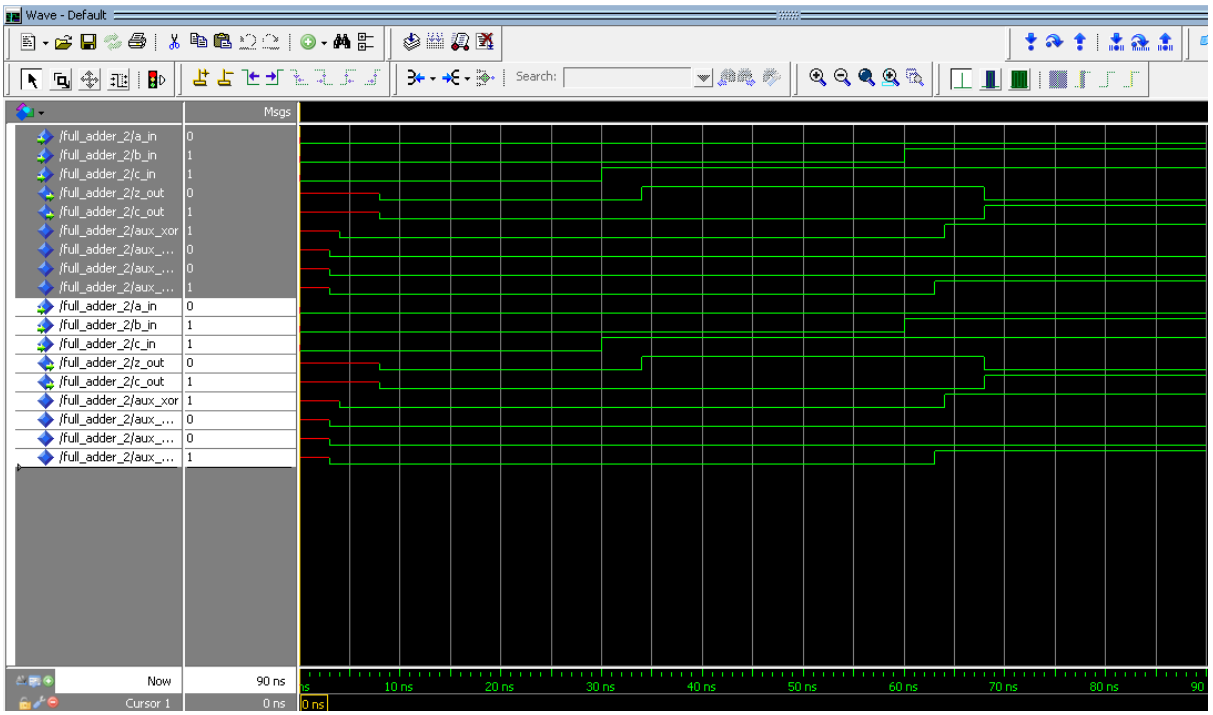


c) Repita o procedimento de captura, compilação e simulação do item 3)
Perguntas: quais foram os tempos de atraso verificados para as portas lógicas? Por que?
Pergunta: como se compara os resultados com os das seções 2 e 3?
Item 4) Captura, compilação e simulação do somador no modelo estrutural com alterações
nas instanciações (sem generic)

and2= 3ns
xor2 = 4ns

or3 = 5ns
Esse tempo de atraso se deve ao instanciamento das portas lógicas no código, como não há especificação para o generic, é usado o valor definido como padrão, no caso os valores descritos acima.

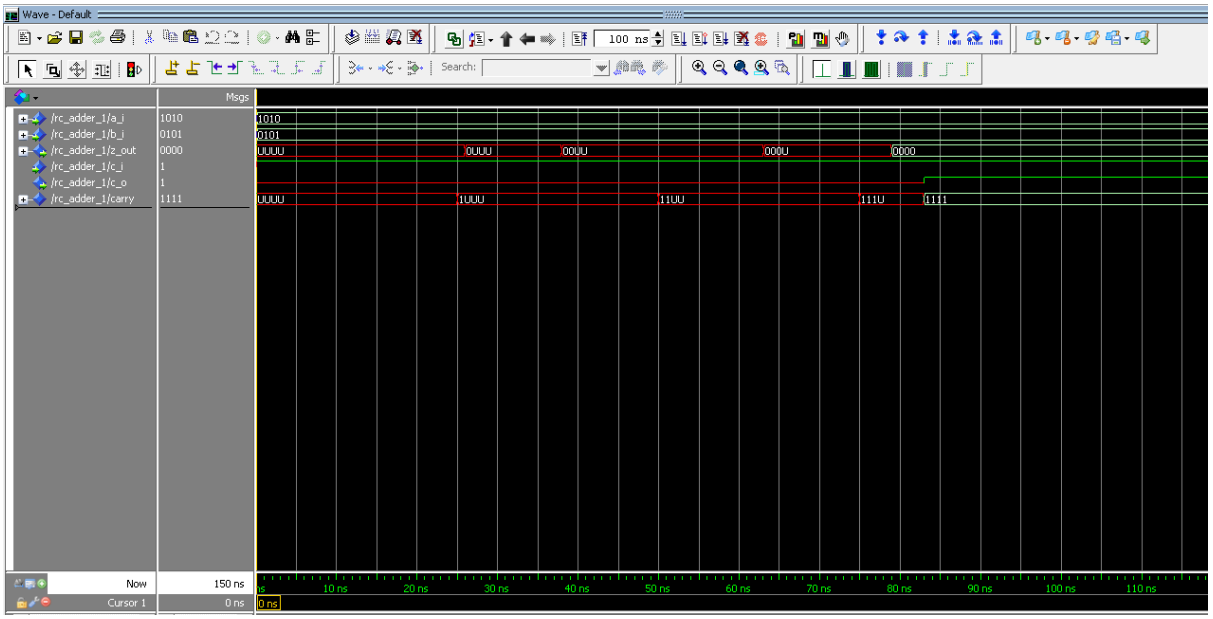
diferente da seção 2 aqui há um atraso determinado como padrão na criação dos componentes, assim ocorre um atraso. Já na seção 3 foi instanciado usando generic valores maiores do que os padrões para os atrasos das portas lógicas, assim o circuito demorou mais para chegar no resultado.



Item 5) Projeto, captura, compilação e simulação do somador de 4 bits no modelo estrutural

d) Faça um esboço do ripple-carry adder com todos os seus sinais externos e internos.

Pode-se observar que o resultado da soma foi obtido em 79ns enquanto o valor do carry só pode ser apresentado em 83ns isso é devido ao atraso de propagação do carry ao longo dos somadores.


Wave - Default
Msgs
/rc_adder_1/a_i
/rc_adder_1/b_i
/rc_adder_1/z_out
/rc_adder_1/c_i
/rc_adder_1/c_o
/rc_adder_1/carry
1010
0101
0000
1
1
1111
UUUU
0UUU
00UU
000U
0000
UUUU
1UUU
11UU
111U
1111
Now
150 ns
Cursor 1
0 ns
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
100 ns
110 ns

```vhdl
entity rc_adder_1 is
    generic
    (
    WIDTH   : natural := 4
    );
    port
    (
    a_i, b_i        :   in STD_LOGIC_VECTOR (WIDTH-1 downto 0);
    z_out       :   out STD_LOGIC_VECTOR (WIDTH-1 downto 0);
    c_i         :   in STD_LOGIC;
    c_o         :   out STD_LOGIC
    );

end rc_adder_1;


architecture structural of rc_adder_1 is

component full_adder_2
port
    (
    a_in, b_in, c_in        :   in STD_LOGIC;
    z_out, c_out        :   out STD_LOGIC
    );

end component;

component full_adder_1
    port
    (
    a_in, b_in, c_in        :   in STD_LOGIC;
    z_out, c_out        :   out STD_LOGIC
    );

end component;
```

```vhdl

signal carry: STD_LOGIC_VECTOR (WIDTH-1 downto 0);

begin


        f1 :full_adder_2
        PORT MAP(a_i(WIDTH-1),b_i(WIDTH-1),c_i,z_out(WIDTH-1),carry(WIDTH-1));


        f2 :full_adder_2
        PORT MAP(a_i(WIDTH-2),b_i(WIDTH-2),carry(WIDTH-1),z_out(WIDTH-2),carry(WIDTH-2));


        f3 :full_adder_2
        PORT MAP(a_i(WIDTH-3),b_i(WIDTH-3),carry(WIDTH-2),z_out(WIDTH-3),carry(WIDTH-3));

        f4 :full_adder_1
        PORT MAP(a_i(0),b_i(0),carry(WIDTH-3),z_out(0),carry(0));

        c_o <= carry(0);


end structural;
```