**Project name**: Sum of three numbers using VHDL

**Dr.name**: Dr.Anas Mellhem

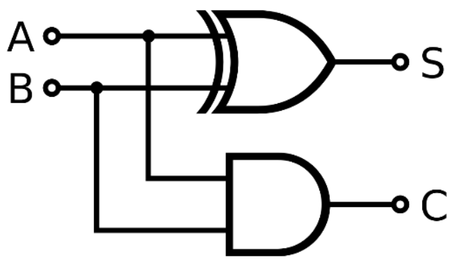
| NAME: | ID: |
|---|---|
| **Ahmad Osama Al-hajqasem** | 201810048 |
| **Huthaifa Jamal Salman** | 201810018 |
| **Ahmad Sameer** | 201810983 |

## Introduction:

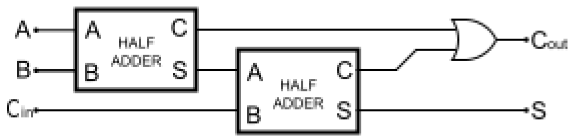
In our previous lecture, we learned how to add one bit to one bit using a Half adder, figure (1.1)

After that learned how to increase the bit number from adding two-bit to adding a three-bit number using a full adder figure (1.2), after that, we learned to add two numbers with different bit numbers using Ripple Carry adder and we are here to learn how to add

Three numbers each number 4 bit

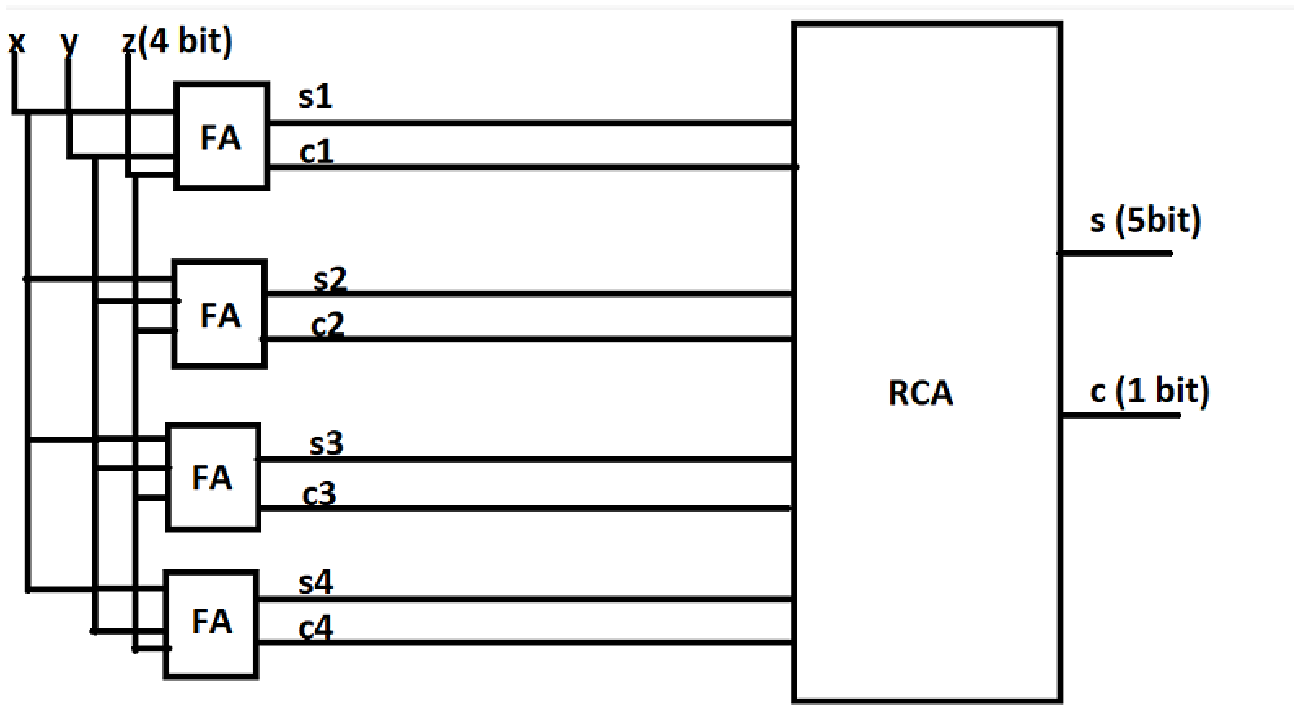


1.1



1.2

## Design:



Let's talk about the design, the design contains two-component

The first component is FA we need in this fourth separate Full Adder

The input of this FA is (x,y,z) each input has 4 bits each bit of each input

Enter in each FA after that the output of half adder its sum and carry

The sum is 4 bits and the carry also 4 bits the bits number is 8 bits

So this output enters the RCA and we know in our previous lecture

The RCA added two numbers with different lengths (n-bits).after that

We get the output we have here six bits output the sum 5 bit and the carry

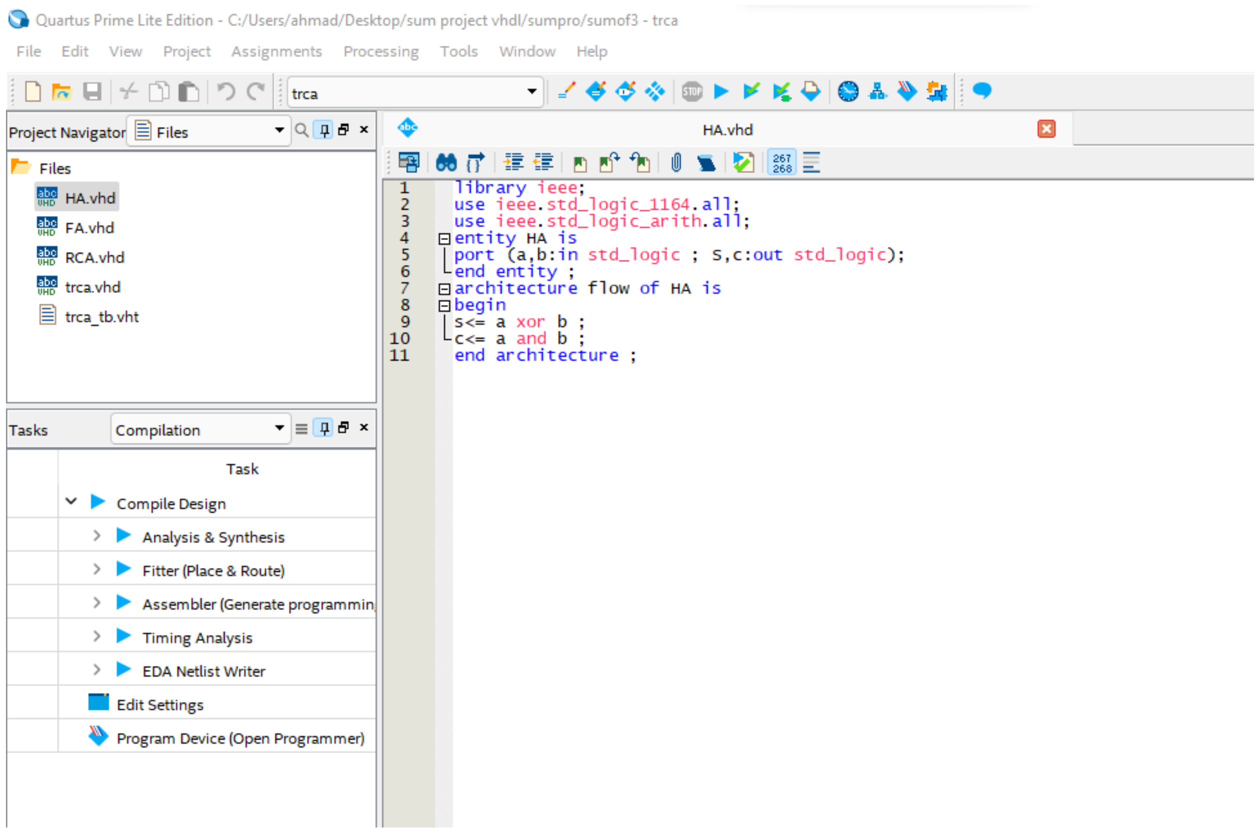
1 bit.

## Project description:

Step 1: open Quartus program to start writing code and select a new project

And start to write

Step 2: create Ha component

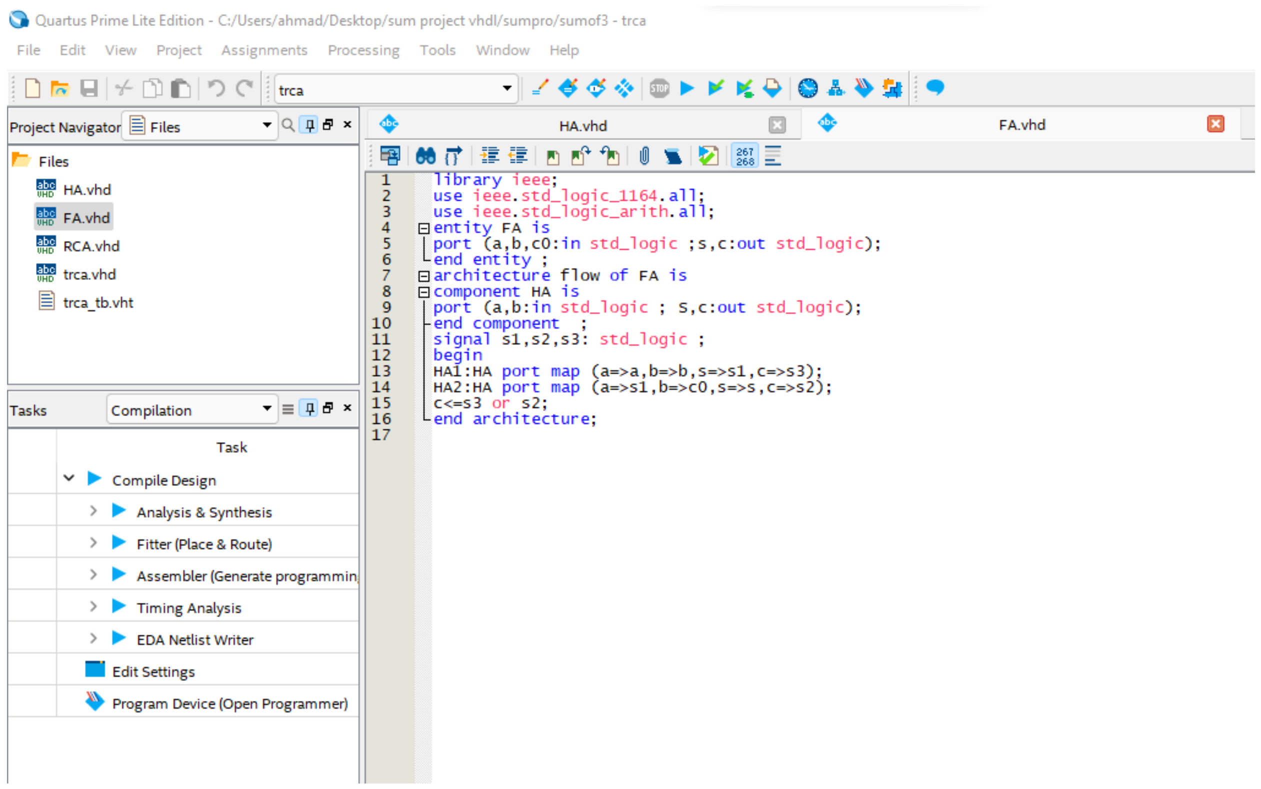

Quartus Prime Lite Edition - C:/Users/ahmad/Desktop/sum project vhdl/sumpro/sumof3 - trca
File Edit View Project Assignments Processing Tools Window Help
trca
Project Navigator
Files
HA.vhd
FA.vhd
RCA.vhd
trca.vhd
trca_tb.vht
Tasks
Compilation
Task
Compile Design
Analysis & Synthesis
Fitter (Place & Route)
Assembler (Generate programmin
Timing Analysis
EDA Netlist Writer
Edit Settings
Program Device (Open Programmer)

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
entity HA is
port (a,b:in std_logic ; s,c:out std_logic);
end entity ;
architecture flow of HA is
begin
s<= a xor b ;
c<= a and b ;
end architecture ;
```

So in HA, we defined the port input and output and then write the architecture

This architecture study it in a previous lecture

## Step 3: create FA component

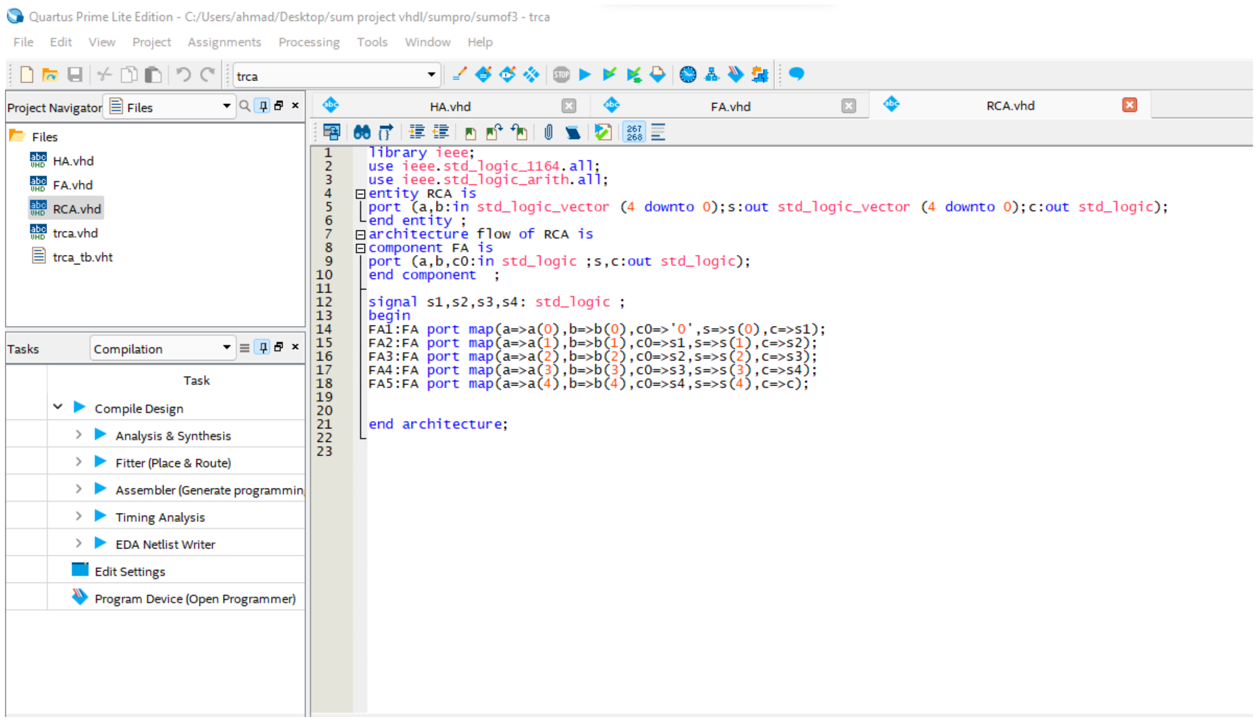

Quartus Prime Lite Edition - C:/Users/ahmad/Desktop/sum project vhdl/sumpro/sumof3 - trca
File Edit View Project Assignments Processing Tools Window Help
trca
Project Navigator
Files
HA.vhd
FA.vhd
RCA.vhd
trca.vhd
trca_tb.vht
Tasks
Compilation
Task
Compile Design
Analysis & Synthesis
Fitter (Place & Route)
Assembler (Generate programmin
Timing Analysis
EDA Netlist Writer
Edit Settings
Program Device (Open Programmer)

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
entity FA is
port (a,b,c0:in std_logic ;s,c:out std_logic);
end entity ;
architecture flow of FA is
component HA is
port (a,b:in std_logic ; s,c:out std_logic);
end component  ;
signal s1,s2,s3: std_logic ;
begin
HA1:HA port map (a=>a,b=>b,s=>s1,c=>s3);
HA2:HA port map (a=>s1,b=>c0,s=>s,c=>s2);
c<=s3 or s2;
end architecture;
```

As we learned in the previous lecture we dined the input and output and write the architecture as we learned in the previous lecture

Step 4: create the RCA component as we learned in the previous lecture

Defined the input and output and the architecture


Quartus Prime Lite Edition - C:/Users/ahmad/Desktop/sum project vhdl/sumpro/sumof3 - trca
File Edit View Project Assignments Processing Tools Window Help
trca
Project Navigator
Files
HA.vhd
FA.vhd
RCA.vhd
trca.vhd
trca_tb.vht
Tasks
Compilation
Task
Compile Design
Analysis & Synthesis
Fitter (Place & Route)
Assembler (Generate programmin
Timing Analysis
EDA Netlist Writer
Edit Settings
Program Device (Open Programmer)

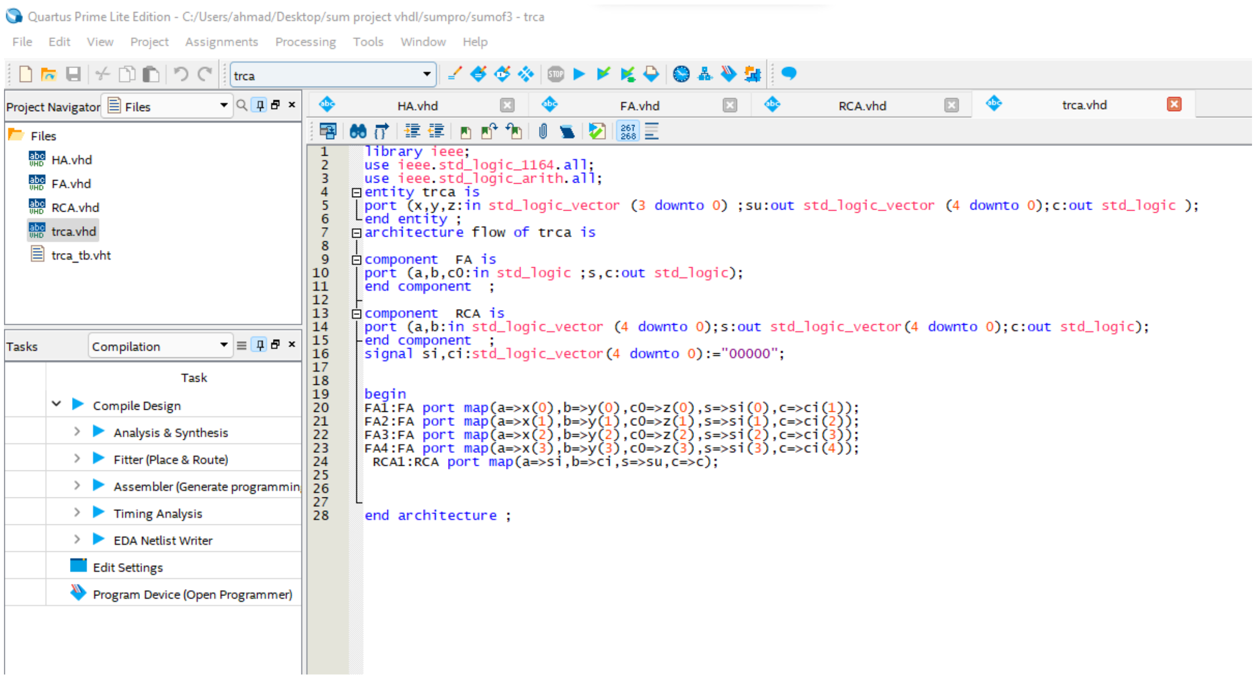
```vhdl
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
entity RCA is
port (a,b:in std_logic_vector (4 downto 0);s:out std_logic_vector (4 downto 0);c:out std_logic);
end entity ;
architecture flow of RCA is
component FA is
port (a,b,c0:in std_logic ;s,c:out std_logic);
end component  ;

signal s1,s2,s3,s4: std_logic ;
begin
FA1:FA port map(a=>a(0),b=>b(0),c0=>'0',s=>s(0),c=>s1);
FA2:FA port map(a=>a(1),b=>b(1),c0=>s1,s=>s(1),c=>s2);
FA3:FA port map(a=>a(2),b=>b(2),c0=>s2,s=>s(2),c=>s3);
FA4:FA port map(a=>a(3),b=>b(3),c0=>s3,s=>s(3),c=>s4);
FA5:FA port map(a=>a(4),b=>b(4),c0=>s4,s=>s(4),c=>c);


end architecture;
```

Step 5 : the main  in this step write the stage one and two the project

As we see in this screenshot


Quartus Prime Lite Edition - C:/Users/ahmad/Desktop/sum project vhdl/sumpro/sumof3 - trca
File Edit View Project Assignments Processing Tools Window Help
trca
Project Navigator
Files
HA.vhd
FA.vhd
RCA.vhd
trca.vhd
trca_tb.vht
Tasks
Compilation
Task
Compile Design
Analysis & Synthesis
Fitter (Place & Route)
Assembler (Generate programmin
Timing Analysis
EDA Netlist Writer
Edit Settings
Program Device (Open Programmer)

```vhdl
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
entity trca is
port (x,y,z:in std_logic_vector (3 downto 0) ;su:out std_logic_vector (4 downto 0);c:out std_logic );
end entity ;
architecture flow of trca is

component  FA is
port (a,b,c0:in std_logic ;s,c:out std_logic);
end component  ;

component  RCA is
port (a,b:in std_logic_vector (4 downto 0);s:out std_logic_vector(4 downto 0);c:out std_logic);
end component  ;
signal si,ci:std_logic_vector(4 downto 0):="00000";


begin
FA1:FA port map(a=>x(0),b=>y(0),c0=>z(0),s=>si(0),c=>ci(1));
FA2:FA port map(a=>x(1),b=>y(1),c0=>z(1),s=>si(1),c=>ci(2));
FA3:FA port map(a=>x(2),b=>y(2),c0=>z(2),s=>si(2),c=>ci(3));
FA4:FA port map(a=>x(3),b=>y(3),c0=>z(3),s=>si(3),c=>ci(4));
 RCA1:RCA port map(a=>si,b=>ci,s=>su,c=>c);



end architecture ;
```

Create the entity in the entity defined the input

We have here 3 inputs the type of this input std-logic_vector

Each input length is 4 bits

Defined the output the sum and the carry the sum length is 5 bits

And the carry length is one bit

As we see in the screenshot

```vhdl
entity trca is
port (x,y,z:in std_logic_vector (3 downto 0) ;su:out std_logic_vector (4 downto 0);c:out std_logic );
end entity ;
```

Defined the architecture

In the architecture, we defined the component we need in this project

The first component is FA :

```vhdl
component  FA is
port (a,b,c0:in std_logic ;s,c:out std_logic);
end component  ;
```

Then, Defined the component RCA

```
component  RCA is
port (a,b:in std_logic_vector (4 downto 0);s:out std_logic_vector(4 downto 0);c:out std_logic);
end component  ;
signal si,ci:std_logic_vector(4 downto 0):="00000";
```

Defined the signal the point of creating signals is to connect between full adders

And RCA

```
RCA1:RCA port map(a=>si,b=>ci,s=>su,c=>c);
```

After that, we create 4 FA

And connect it with inputs and signals

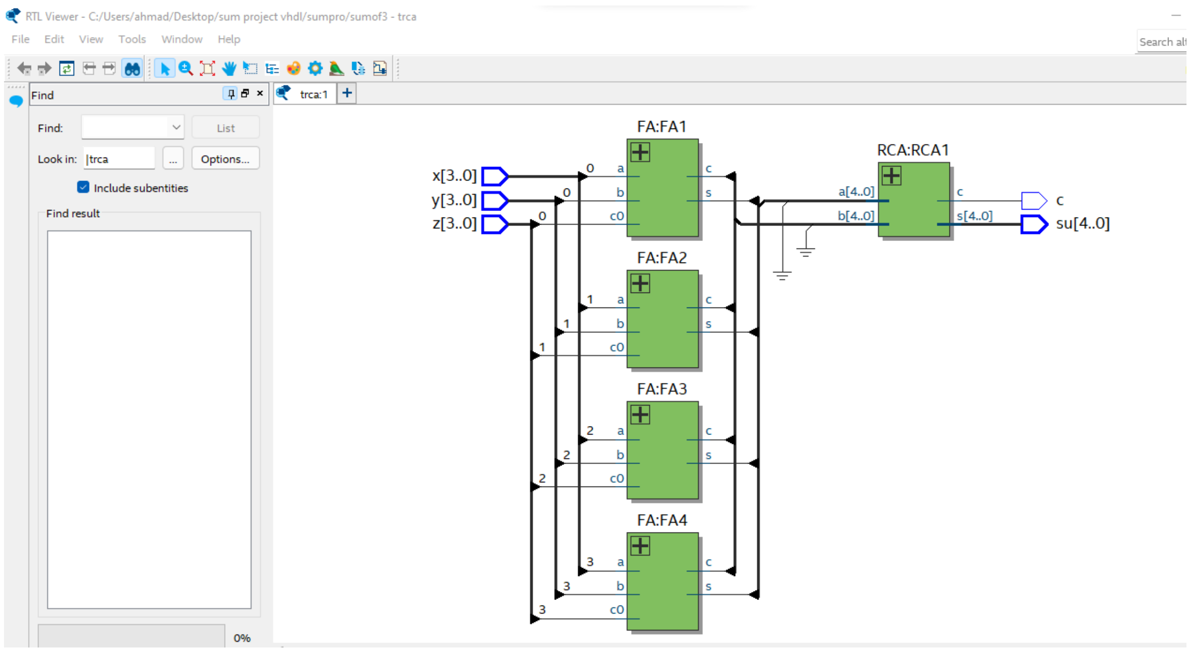
```
FA1:FA port map(a=>x(0),b=>y(0),c0=>z(0),s=>si(0),c=>ci(1));
FA2:FA port map(a=>x(1),b=>y(1),c0=>z(1),s=>si(1),c=>ci(2));
FA3:FA port map(a=>x(2),b=>y(2),c0=>z(2),s=>si(2),c=>ci(3));
FA4:FA port map(a=>x(3),b=>y(3),c0=>z(3),s=>si(3),c=>ci(4));
```

After that, we create RCA component

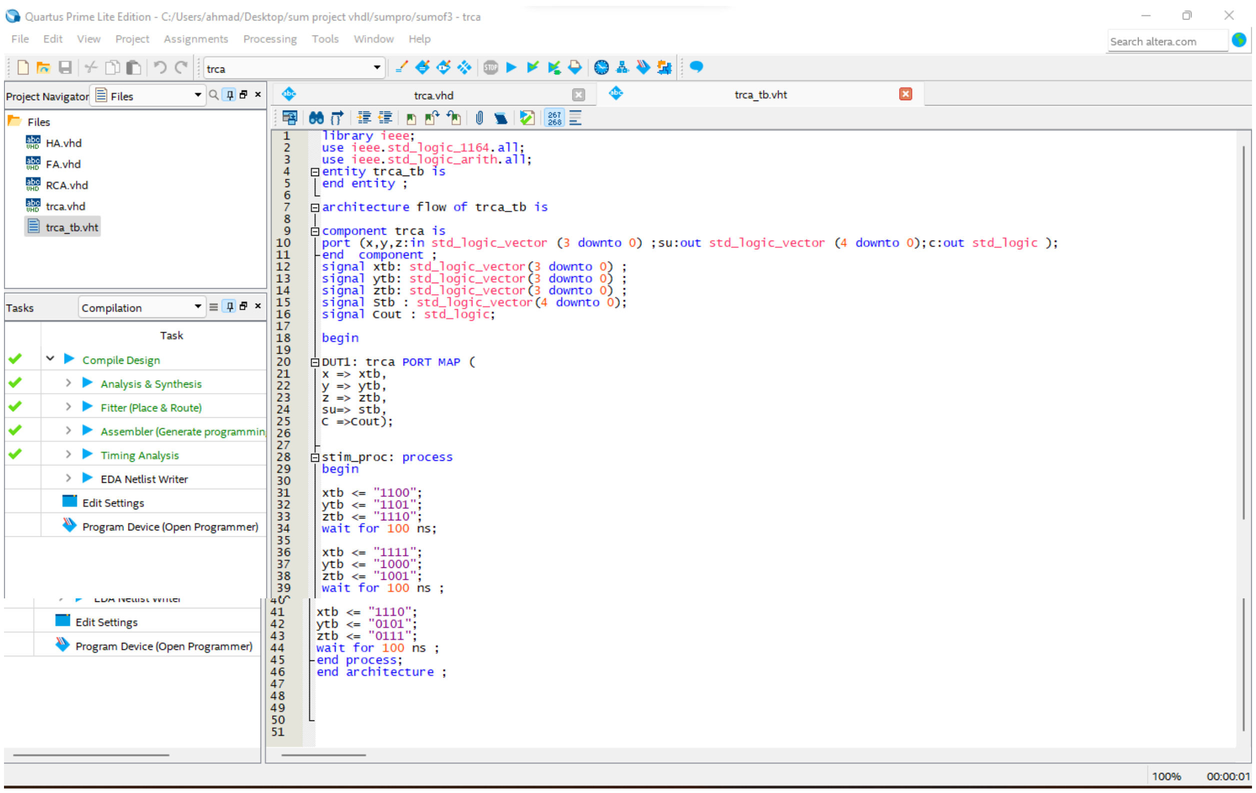
And connect it with signals and output

```
RCA1:RCA port map(a=>si,b=>ci,s=>su,c=>c);
```

We are finished here writing coding



Now we need to write a test bench to test this project


Quartus Prime Lite Edition - C:/Users/ahmad/Desktop/sum project vhdl/sumpro/sumof3 - trca

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
entity trca_tb is
end entity ;

architecture flow of trca_tb is

component trca is
port (x,y,z:in std_logic_vector (3 downto 0) ;su:out std_logic_vector (4 downto 0);c:out std_logic );
end  component ;
signal xtb: std_logic_vector(3 downto 0) ;
signal ytb: std_logic_vector(3 downto 0) ;
signal ztb: std_logic_vector(3 downto 0) ;
signal Stb : std_logic_vector(4 downto 0);
signal Cout : std_logic;

begin

DUT1: trca PORT MAP (
x => xtb,
y => ytb,
z => ztb,
su=> stb,
C =>Cout);


stim_proc: process
begin

xtb <= "1100";
ytb <= "1101";
ztb <= "1110";
wait for 100 ns;

xtb <= "1111";
ytb <= "1000";
ztb <= "1001";
wait for 100 ns ;

xtb <= "1110";
ytb <= "0101";
ztb <= "0111";
wait for 100 ns ;
end process;
end architecture ;
```

First, we create a new text file and save it as vht

Defined an entity without put and input

```
entity trca_tb is
end entity ;
```

Defined the architecture and defined the project we made as component

```
port (x,y,z:in std_logic_vector (3 downto 0) ;su:out std_logic_vector (4 downto 0);c:out std_logic );
end  component ;
```

Defined the signal, these signals use as input and output in this test

Defined the component and connect the input and output with signals

```
DUT1: trca PORT MAP (
x => xtb,
y => ytb,
z => ztb,
su=> stb,
C =>Cout);
```

After that, we defined a process and enter the value u want to test

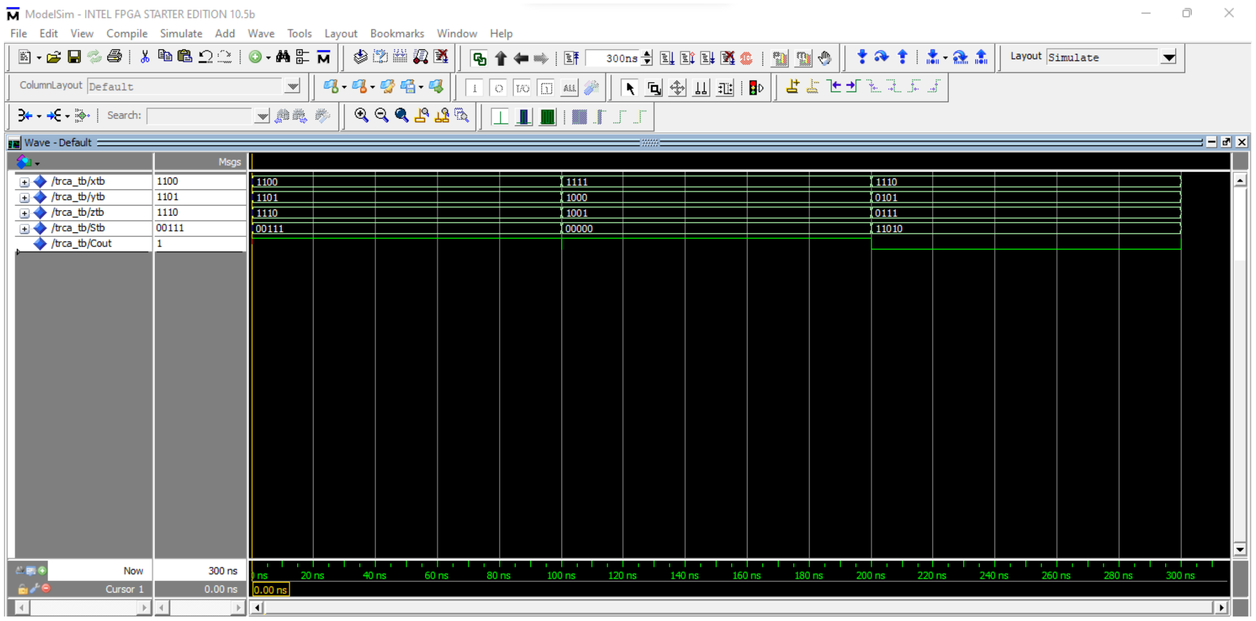
```
stim_proc: process
begin

xtb <= "1100";
ytb <= "1101";
ztb <= "1110";
wait for 100 ns;

xtb <= "1111";
ytb <= "1000";
ztb <= "1001";
wait for 100 ns ;

xtb <= "1110";
ytb <= "0101";
ztb <= "0111";
wait for 100 ns ;
end process;
end architecture ;
```

Final we have the test bench result :



```
/trca_tb/xtb    1100     1100    1111    1110
/trca_tb/ytb    1101     1101    1000    0101
/trca_tb/ztb    1110     1110    1001    0111
/trca_tb/Stb    00111    00111   00000   11010
/trca_tb/Cout   1

Now       300 ns
Cursor 1  0.00 ns
```

**Thanks for reading**