Tuesday	**EE 671: VLSI Design**	Due on Saturday
Sep. 26, 2023	**Assignment 3**	Oct. 07, 2023

Q–1 Describe a 32 bit Brent Kung adder in VHDL and simulate it using a test bench. Your description should use `std_logic` types for various signals. You can use the public domain tool "ghdl" for VHDL simulation.

a) Logic functions AND, XOR, A + B.C and A.B + C.(A+B) are required for computing different orders of G, P and final sum and carry outputs. A template file with entity/architecture pairs for these functions is appended at the end. You should modify that code to implement delays for logic functions as given below:

Function	Delay in ps
AND	300 + last two digit of your roll number
A + B.C	400 + last two digits of your roll number
XOR	600 + 2*last two digits of your roll number
A.B + C.(A+B)	600 + 2*last two digits of your roll number

b) Using the above entities as components, write structural descriptions of each level of the tree for generating various orders of G and P values.
The rightmost blocks of all levels should use the already available value of C0 to compute the output carry directly using the logic block for A.B + C.(A+B) and use these instead of the G values for computation of P and G for the next level.

c) Using the outputs of the tree above, write structural VHDL code for generating the bit wise sum and carry values. Test the final adder with a test bench which reads pairs of 16 bit words and a single bit input carry value from a file, applies these to the adder and compares the result with the expected 16 bit sum and 1 bit carry values stored in the same file.
This test should be carried out for 10 randomly chosen input combinations.
It should use assert statements to flag errors if there is a mismatch between the computed sum/carry and the stored sum/carry.

```
-- simple gates with trivial architectures
library IEEE;
use IEEE.std_logic_1164.all;

entity andgate is
        port (A, B: in std_ulogic;
              prod: out std_ulogic);
end entity andgate;

architecture trivial of andgate is
begin
prod <= A AND B AFTER 300 ps;
end architecture trivial;
```

```
library IEEE;
use IEEE.std_logic_1164.all;

entity xorgate is
        port (A, B: in std_ulogic;
              uneq: out std_ulogic);
end entity xorgate;

architecture trivial of xorgate is
begin
uneq <= A XOR B AFTER 600 ps;
end architecture trivial;

library IEEE;
use IEEE.std_logic_1164.all;

entity abcgate is
        port (A, B, C: in std_ulogic;
              abc: out std_ulogic);
end entity abcgate;

architecture trivial of abcgate is
begin
abc <= A OR (B AND C) AFTER 400 ps;
end architecture trivial;

-- A + C.(A+B) with a trivial architecture
library IEEE;
use IEEE.std_logic_1164.all;

entity Cin_map_G is
port(A, B, Cin: in std_ulogic;
Bit0_G: out std_ulogic);
end entity Cin_map_G;

architecture trivial of Cin_map_G is
begin
Bit0_G <= (A AND B) OR (Cin AND (A OR B)) AFTER 600 ps;
end architecture trivial;
```