**HALF ADDRER**

entity half\_addrer is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

SUM : out STD\_LOGIC;

CARRY : out STD\_LOGIC);

end half\_addrer;

architecture Behavioral of half\_addrer is

begin

**SUM<= A XOR B;**

**CARRY<= A AND B;**

end Behavioral;

Test bench

stim\_proc: process

begin

**A<='0'; B<='0';**

**wait for 100 ns;**

**A<='0'; B<='1';**

**wait for 100 ns;**

**A<='1'; B<='0';**

**wait for 100 ns;**

**A<='1'; B<='1';**

**wait for 100 ns;**

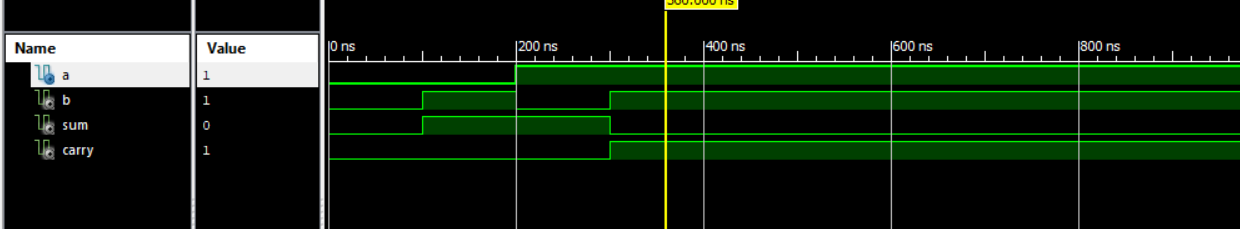
-- wait for <clock>\_period\*10;

-- insert stimulus here

wait;

end process;

END;



**Full ADDRER**

entity FA is

Port ( A : in STD\_LOGIC;

B : in STD\_LOGIC;

Cin : in STD\_LOGIC;

Y : out STD\_LOGIC;

Cout : out STD\_LOGIC);

end FA;

architecture Behavioral of FA is

begin

**Y<= A XOR B XOR Cin;**

**Cout<= ( A AND B) OR ( B AND Cin) OR ( Cin AND A);**

end Behavioral;

**Test Bench for Full Adder**

stim\_proc: process

begin

-- hold reset state for 100 ns.

A<='0'; B<='0'; Cin<='0';

wait for 100 ns;

A<='0'; B<='0'; Cin<='1';

wait for 100 ns;

A<='0'; B<='1'; Cin<='0';

wait for 100 ns;

A<='0'; B<='1'; Cin<='1';

wait for 100 ns;

A<='1'; B<='0'; Cin<='0';

wait for 100 ns;

A<='1'; B<='0'; Cin<='1';

wait for 100 ns;

A<='1'; B<='1'; Cin<='0';

wait for 100 ns;

A<='1'; B<='1'; Cin<='1';

wait for 100 ns;

**wait;**

**end process;**

**END;**

