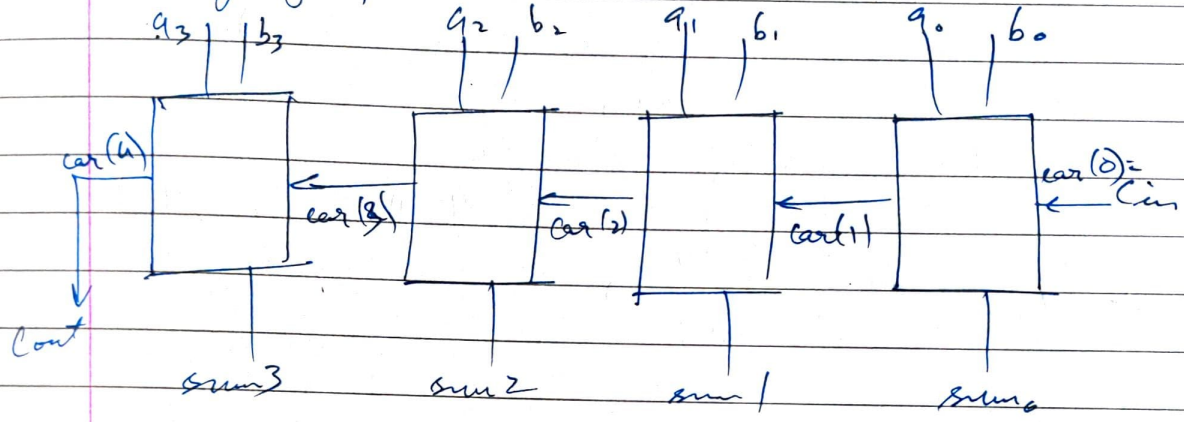
✡ Generate Statements

* Concurrent statements can be replicated a predetermined no. of times using "for-generate".
* Concurrent statements can be conditionally elaborated using "if-generate" statement.
* Eg: ~~4 Bit Bin Adder~~ ~~Full Adder~~ using "for-generate"

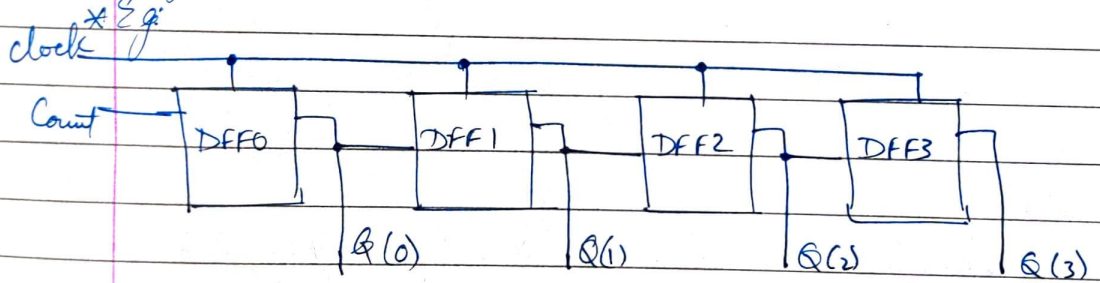
(1)

```
library ieee;
use ieee.std_logic_1164.all;

entity Bin_adder is
    port (A,B: in bit_vector(3 downto 0);
          Cin: in bit;
          sum: out bit_vector(3 downto 0);
          Cout: out bit);
end Bin_adder;

architecture forgen of Bin_adder is
    component fa
        port(a,b,c: in bit; cout, sum: out bit);
    end component fa;
    signal car: bit_vector(4 downto 0);
```

```
begin
    car(0) <= Cin;
    GK: for K in 3 downto 0 generate
        FAL: fa port map (car(k), a(k), b(k), car(K+1), sum(K));
    end generate GK;
    cout <= car(4);
end for_gen;
```



(2) If Generate statement




```
library ieee;
use ieee.std_logic_1164.all

entity counter4 is
    port (count, clock: in bit; Q: buffer bit_vector (0 to 3));
end counter4;
```

```
architecture if_gen of counter4 is
    component DFF
        port (D, clk: in bit; Que: out bit);
    end DFF;
begin
    GK: for K in 0 to 3 generate
        GK0: if K = 0 generate
            DFF1: DFF port map (count, clock, Q(K));
        end generate GK0;
        GK1_3: if K > 0 generate
            DFF1: DFF port map (Q(K-1), clock, Q(K));
        end generate GK1_3;
    end generate GK;
end if_gen;
```