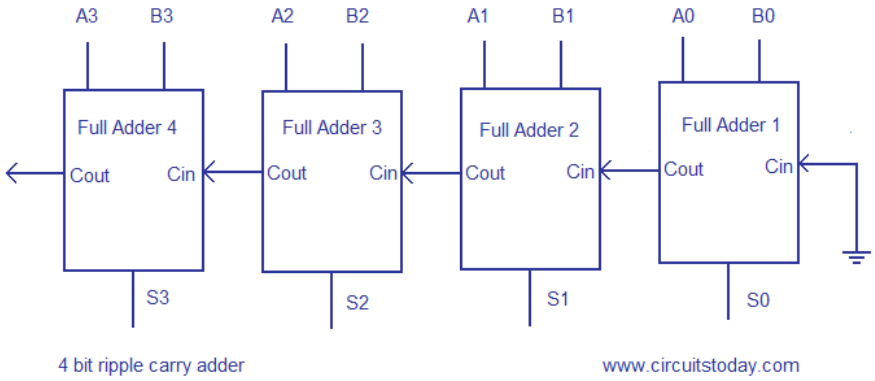
FRONT END VLSI TRAINING - 2015

ASSIGNMENT NO - 1

REPORT BY- Ajinkya Padwad

TOPIC - 4-BIT RIPPLE CARRY ADDER (STRUCTURAL APPROACH)



WORKING -

1. Since it is a structural approach, the top level module makes use of instances of more fundamental blocks to implement a higher level logic.
2. Ripple Carry Adder makes use of 4 Full Adder instances which have their independently defined module in a separate file location.
3. In the verilog code, two 4 bit inputs are considered -'A' and 'B' - with a single bit carry input, 'Cin' The final outputs are 4 bit 'sum' and 1 bit 'carry'.
4. The input and output pins of the 4 instances of Full Adder are correspondingly mapped on the input and output pins of the top module.
5. The first carry input is fed to the first Full Adder and the output carry is fed to the next Full Adder. Subsequently, the carry outputs are 'rippled' through the circuit and this gives the final carry output.
6. The intermediate carry signals are handled by defining three separate wires at the top module.
7. The 4 bit 'sum' is obtained bit by bit from each Full Adder.
8. Finally all the input and output signals are ported to the FPGA board for observing the output.

BOTTOM MODULE:

(Full Adder)

```
module FA1(
   input A,
   input B,
   input Cin,
   output sum ,
   output carry
   );
assign sum = A^B^Cin;                          //Logic for sum
assign carry = (A&B) | (B&Cin) | (Cin&A) ;     //Logic for carry
endmodule
```

TOP MODULE:

(Ripple Carry Adder)

```
module RCA2(
   input [3:0] A,
   input [3:0] B,
   input Cin,
   output [3:0] sum,
   output carry
   );
wire c1,c2,c3;                          //Three wires for intermediate carry outputs

FA1 w1( .A(A[0]), .B(B[0]), .Cin(Cin), .sum(sum[0]), .carry(c1)); //First instance

FA1 w2( .A(A[1]), .B(B[1]), .Cin(c1), .sum(sum[1]), .carry(c2));  //Second instance

FA1 w3( .A(A[2]), .B(B[2]), .Cin(c2), .sum(sum[2]), .carry(c3));  //Third instance

FA1 w4( .A(A[3]), .B(B[3]), .Cin(c3), .sum(sum[3]), .carry(carry));  //Fourth instance

endmodule
```