Computer Organization and Architecture Laboratory

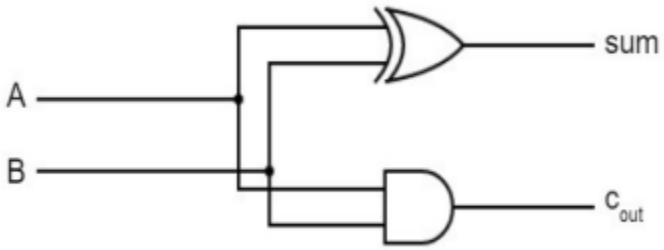
Assignment-3

Group members (Group - 56):

1. **Utsav Mehta (20CS10069)**
2. **Vibhu (20CS10072)**

Design of an Ripple Carry Adder (RCA) :-

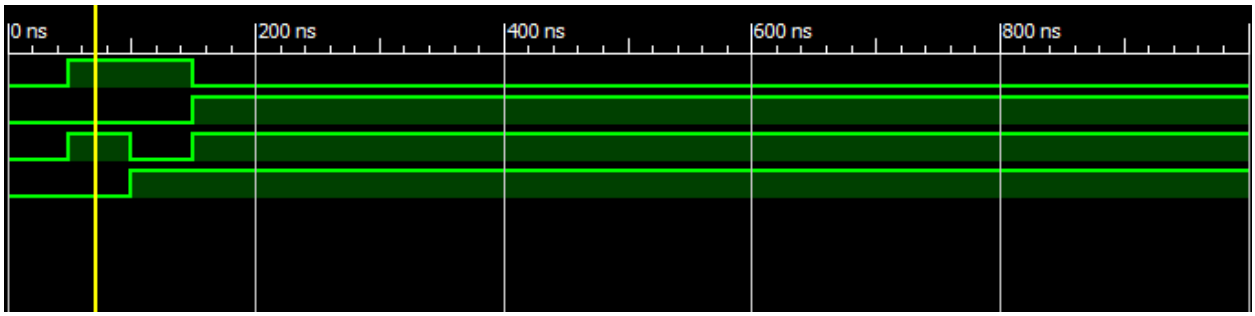
A. Half - adder:
   a. Takes two input bits ‘a’ and ‘b’
   b. Produces two output bits ‘sum’ and ‘carry’
   c. Values of the output bits can be derived from the input bits as -
      i. sum (or s) = a ^ b (bitwise - xor operation)
      ii. carry (or c) = a & b (bitwise - and operation)
   d. The logic circuit for half adder is:



   e. The truth table for half - adder is as shown -

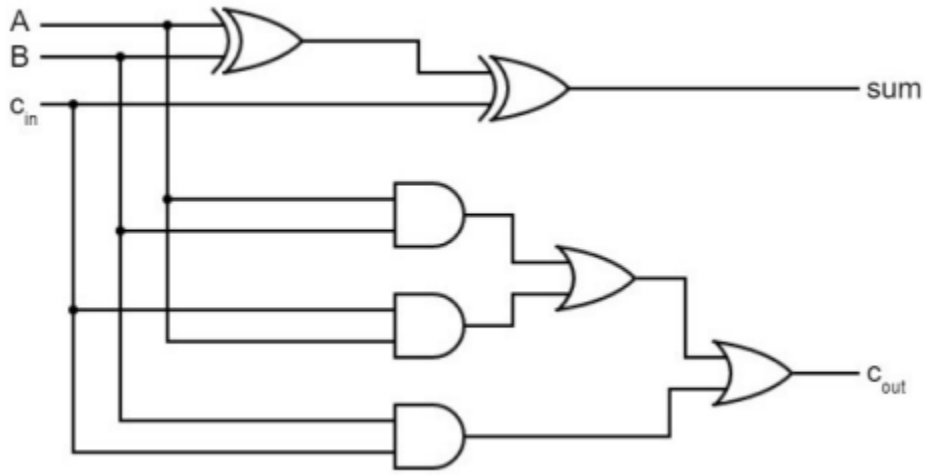
Truth Table (for half adder)			
Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

   e. The graph observed is as shown -



Relevant files: half_adder.v, tb_half_adder.v

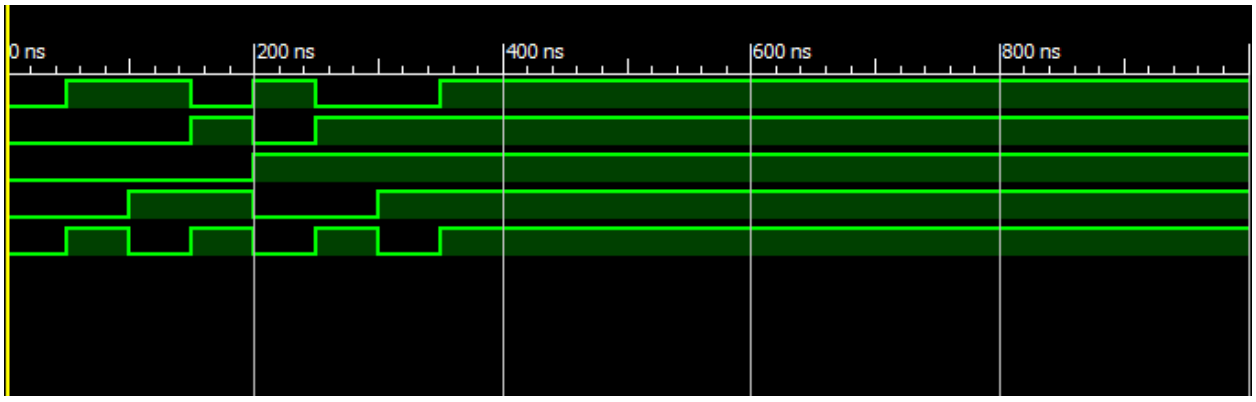
B. Full - adder:
   a. Takes three input bits ‘a’, ‘b’ and ‘carry0’
   b. Produces two output bits ‘sum’ and ‘carry1’
   c. Values of the output bits can be derived from the input bits as -
      i. temp_sum = a ^ b
      ii. temp_carry0 = a & b
      iii. sum = temp_sum ^ carry0
      iv. temp_carry1 = temp_sum & carry0
      v. carry1 = temp_carry0 | temp_carry1
   d. The logic circuit for full adder is as shown:



   e. The truth table for full - adder is as shown -

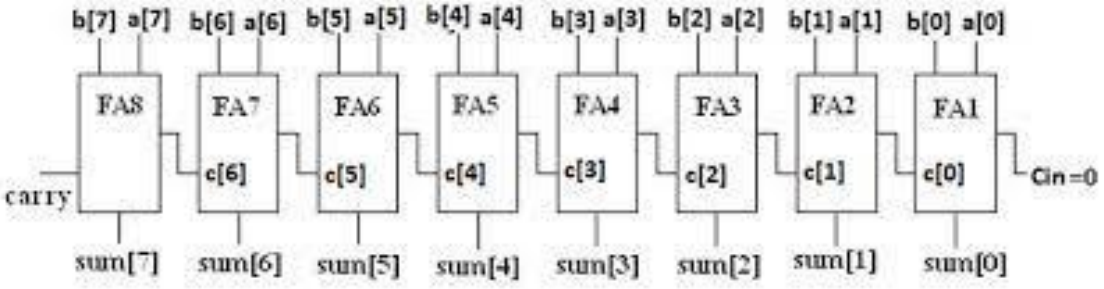
Truth Table (for Full Adder)				
Inputs			Outputs	
A	B	Carry0	Sum	Carry1
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

f. The graph observed is as shown -

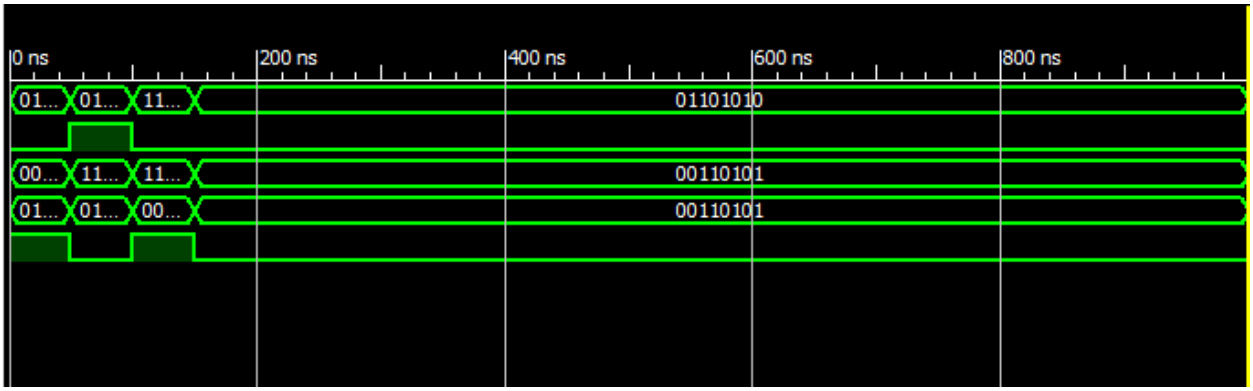


Relevant Files: full_adder.v, tb_full_adder.v

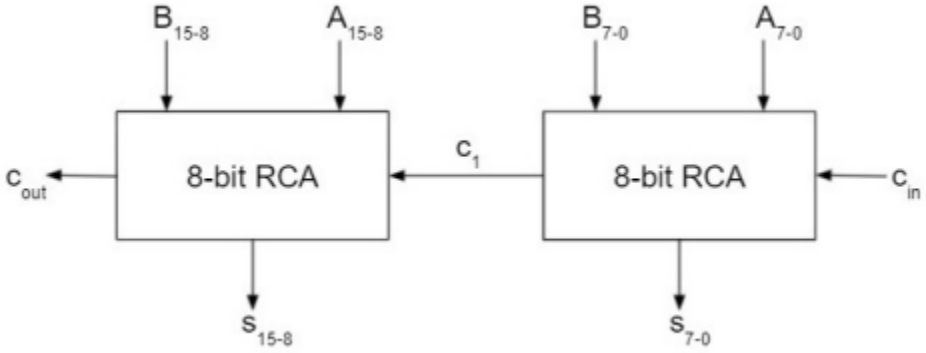
C. Ripple Carry Adders:
   a. 8-bit adder - Take 8 full adders and connect them in cascading fashion



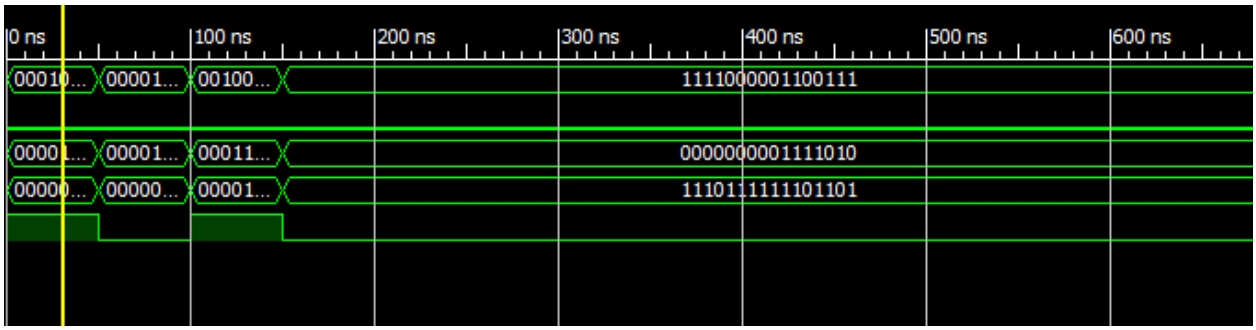
The graph observed for 8-bit ripple carry adder is as shown -



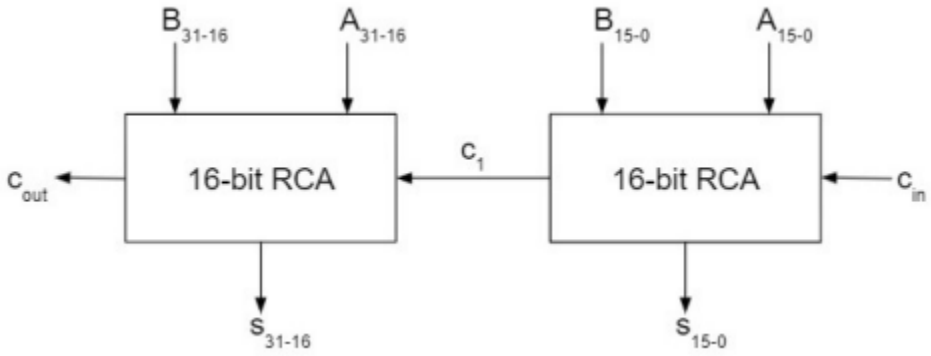
b. 16-bit adder - Take 2 eight bit adders and connect them in cascading fashion



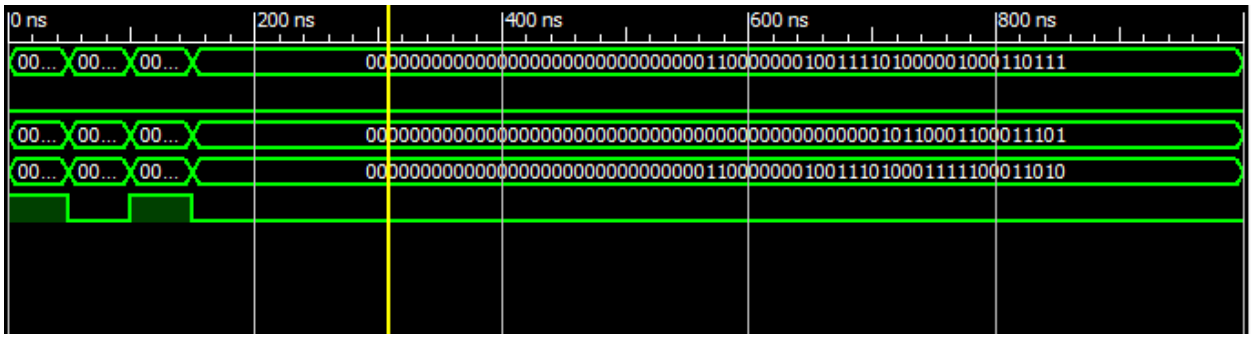
The graph observed for 16-bit ripple carry adder is as shown -



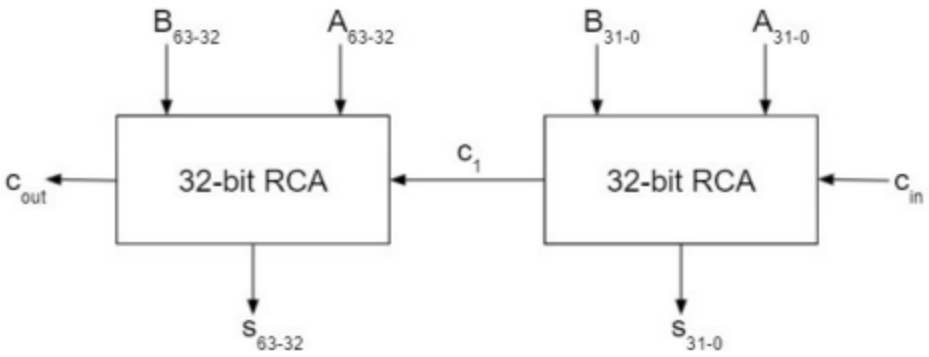
c. 32-bit adder - Take 2 sixteen bit adders and connect them in cascading fashion



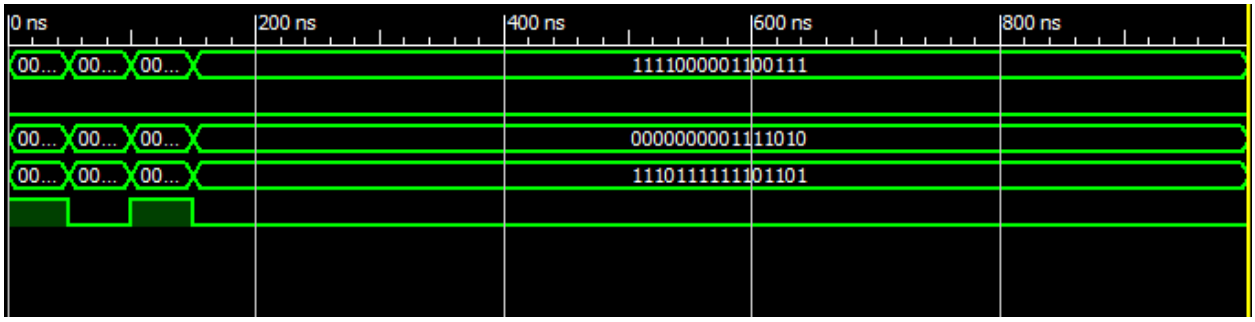
The graph observed for 32-bit ripple carry adder is as shown -



d. 64-bit adder - Take 2 thirty two bit adders and connect them in cascading fashion



The graph observed for 64-bit ripple carry adder is as shown -



The synthesis report (denoting the longest delays in the circuits is):-

Synthesis report	
Ripple carry adder	Longest delay
Half bit adder	1.066
Full bit adder	1.246
8-bit RCA	3.471
16-bit RCA	6.167
32-bit RCA	11.559
64-bit RCA	22.343

Relevant Files:

s8_bit_adder.v, tb_8_bit_adder.v

s16_bit_adder.v, tb_16_bit_adder.v
s32_bit_adder.v, tb_32_bit_adder.v
s64_bit_adder.v, tb_64_bit_adder.v

D. Using the above circuit for calculating differences:

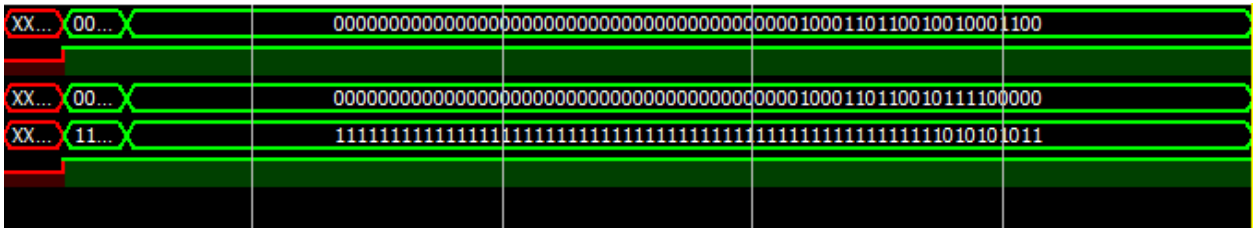
Let the input integers be A and B, and we want to calculate the value of A-B using the ripple carry adders generated above.

The equation can be re-interpreted as A + (-B).
The value of -B can be computed from B by the following steps:

1. Take the two's complement of B
2. Add 1 to the value computed above

Thus, we can add the generated value and A to compute the difference between A and B.

The graph observed for 64-bit ripple carry adder (used as a subtractor) is as shown -



Relevant Files:

substractor.v, s64_bit_adder.v

Design of an Carry-Look Ahead (CLA) adder :-

A. Design of a 4-bit CLA –

a. A Carry-Look Ahead adder improves the time taken by a Ripple Carry Adder by computing the carry for each subsequent step beforehand.
b. The hardware for CLA is more complex than that of RCA.
c. We introduce two variables, for pre-computation of the carry - Propagate (P) and Generate(G)

$$G_i = A_i \cdot B_i \text{ (carry generate)}$$
$$P_i = A_i \oplus B_i \text{ (carry propagate)}$$

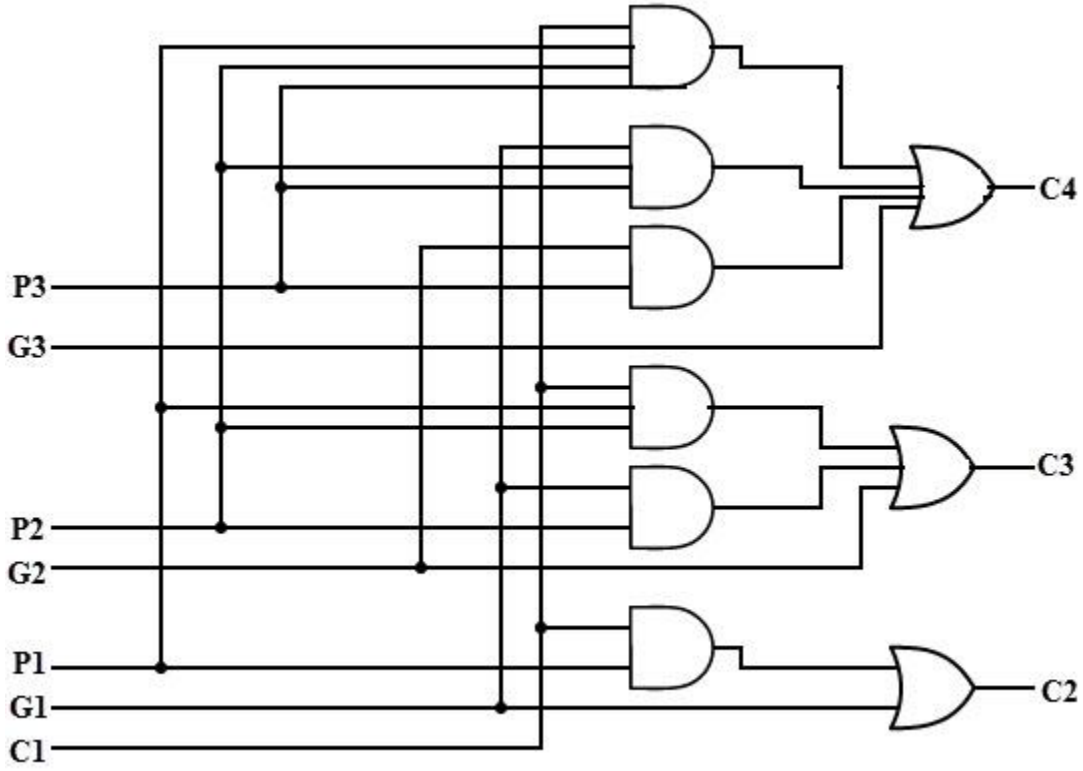
   i. G_i denotes the carry generation at the ith stage, computed independently from other bits.
   ii. P_i denotes the propagation of carry from this stage, to the next, thus we can write the following -

$$C_{i+1} = G_i + P_i \cdot C_i$$

d. The Boolean equations for Look Ahead carry generation are -

   i. c[0] = c_in
   ii. c[1] = g[0] | (p[0] & c_in)
   iii. c[2] = g[1] | (p[1] & g[0]) | (p[1] & p[0] & c_in)
   iv. c[3] = g[2] | (p[2] & g[1]) | (p[2] & p[1] & g[0]) | (p[2] & p[1] & p[0] & c_in)
   v. c[4] = g[3] | (p[3] & g[2]) | (p[3] & p[2] & g[1]) | (p[3] & p[2] & p[1] & g[0]) | (p[3] & p[2] & p[1] & p[0] & c_in)
   vi. c[4] = c_out = carry out bit

Circuit Diagram for a 4-bit CLA is as shown -



Relevant files:- CLA_4_bit.v, tb_CLA_4_bit.v

B. Comparison of a 4-bit CLA with 4-bit RCA –

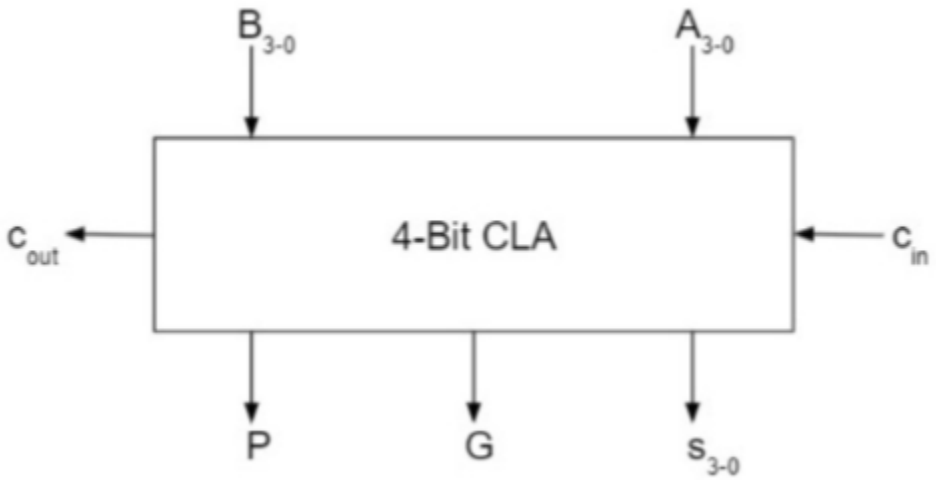
The speed (or time) required by a 4-bit CLA vs 4-bit RCA, for the same input bits, is as shown -

Synthesis Report	
4-bit CLA	4-bit RCA
2.123ns	2.115ns
0.249ns (logic)	0.249ns (logic)
1.874ns (route)	1.866ns (route)

The time required by a RCA is clearly greater than that required by a CLA, as the delay for shifting carry from (i)th bit to (i+1)th is minimized. Thus CLAs provide a faster method to implement bitwise summations, by decreasing the rippling costs.

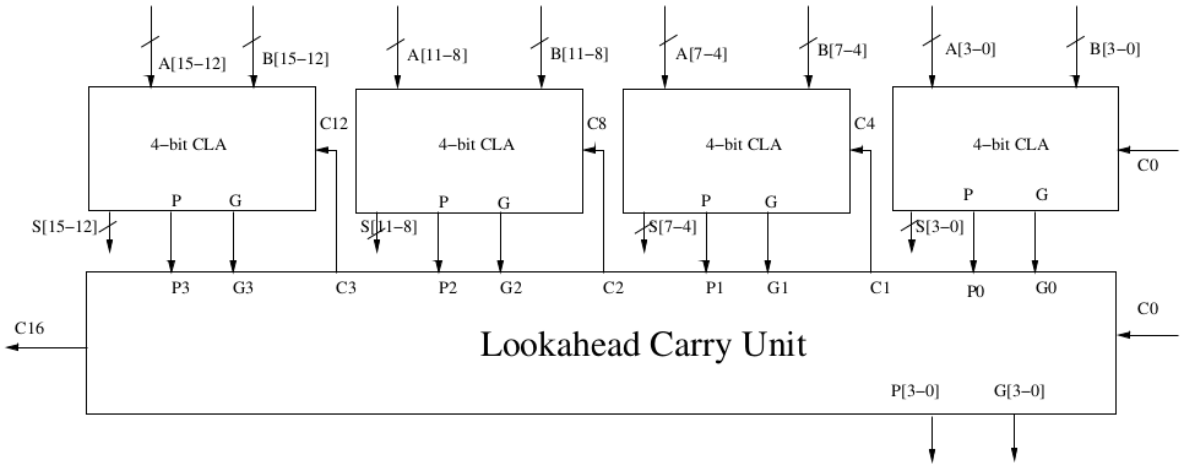
Relevant files:- CLA_4_bit.v, tb_CLA_4_bit.v
s4_bit_adder.v (4-bit RCA)
tb_4_bit_adder.v (test bench for 4bit RCA)
full_adder.v, half_adder.v

C. Design of a 16-bit CLA and it's comparison with 16-bit RCA –
   a. Augmenting the 4-bit CLA to generate Propagate and Generate Signals -
      i. Propagate (P) - block propagate for the next level of hierarchy
      ii. Generate (G) - block generate for the next level of hierarchy
      iii. The augmented 4-bit circuit looks like -



Relevant Files(s): CLA_4_bit.v, s4_bit_adder.v (4bit RCA)

b. Designing a 16-bit CLA using Look Ahead Carry Unit
    i. Look Ahead Carry Unit -
        1. Input - block propagates and generates from the previous level
        2. Output - carry bits and block propagate, generate for the next level
    ii. 4-bit CLA augmented circuit is integrated with Look Ahead Carry unit to create a 16-bit Carry Look Ahead adder



Relevant File(s): carry_look_ahead.v, tb_carry_look_ahead.v, CLA_16_bit.v, tb_CLA_4_bit.v

c. Comparison of the circuit created by 16-bit Carry Look Ahead adder (using carry look ahead unit, and four 4-bit Carry Look augmented adders) and 16-bit Ripple Carry Adders (using four 4-bit CLA adders)
    i. Time required by the 16-bit CLA is utilized in computation the sum of four 4-bits by the 4-bit CLA augmented and the time required by the Carry Look

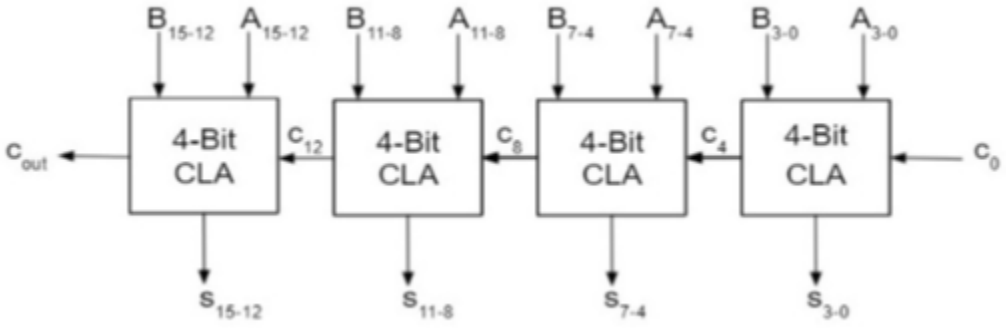
Ahead unit to transfer the carry between various 4-bit units.

ii. Time required by the 16-bit RCA is utilized in the computation of the sum required by four 4-bit CLA and the time required by the Ripple Carry Units to transfer the carry from one unit to the other.

iii. The time required to ripple the carry bits from one CLA to the other is the critical factor that results in the given time gaps –

Synthesis Report (for 16-bit RCA)	
Using Carry Look Ahead unit	Using Ripple Carry Adder
5.513ns	6.167ns
0.869ns (logic)	0.993ns (logic)
4.644ns (route)	5.174ns (route)

The circuit for 16-bit CLA using ripple carry adder unit:



Relevant File(s): CLA_16_bit.v, tb_CLA_16_bit.v, CLA_16_bit_ripple.v, tb_CLA_16_bit_ripple.v, carry_look_ahead.v, tb_carry_look_ahead.v

d. Result of Synthesis and the corresponding speed and LUTs
   i. Time required by the 16-bit CLA is utilized in computation the sum of four 4-bits by the 4-bit CLA augmented and the time required by the Carry Look Ahead unit to transfer the carry between various 4-bit units.
   ii. Time required by the 16-bit RCA is utilized in the computation of the sum required by four 4-bit CLA and the time required by the Ripple Carry Units to transfer the carry from one unit to the other.
   iii. Comparison of the cost required by the LUTs -
      1. Slice LUTs used for 16-bit CLA - 45 (without wrapper) and 52 (with wrapper)
      2. Slice LUTs used for 16-bit RCA - 24
   iv. The time required and Look Up Tables (LUTs) by various adders including the RCAs generated in Part-1, and various adders generated for Part-2 are summarized below.

Module	Total delay (in ns)	Logic Delay (in ns)	Route Delay (in ns)	LUTs used
s4_bit_adder.v (RCA)	2.115	0.249	1.866	5
s8_bit_adder.v (RCA)	3.471	0.497	2.974	12
s16_bit_adder.v (RCA)	6.167	0.993	5.174	24
s32_bit_adder.v (RCA)	11.559	1.985	9.574	48
s64_bit_adder.v (RCA)	22.343	3.969	18.374	96
CLA_4_bit.v	2.123	0.249	1.874	6

wrapper_CLA_4_bit.v	2.116	0.756	1.360	6
CLA_4_bit_augmented.v	3.209	0.373	2.836	10
wrapper_CLA_4_bit_augmented.v	2.130	0.756	1.374	10
CLA_16_bit.v	5.513	0.869	4.644	45
wrapper_CLA_16_bit.v	3.411	1.004	2.407	52
CLA_16_bit_ripple.v	6.167	0.993	5.174	24
wrapper_CLA_16_bit_ripple.v	3.344	0.880	2.464	47