COA Lab Report

Assignment No. 1

Student 1: Aryan Singh Roll No: 19CS30007
Student 2: Abhinandan De Roll No: 19CS10069

OBJECTIVES:

- Ripple Carry Adder
    - Half Adder
    - Full Adder
    - N-bit Ripple Carry Adder
    - N-bit Subtractor

- Carry Look-Ahead Adder
    - Design of 4 bit CLA
    - HDL design of 4 bit CLA
    - 16 Bit CLA (with ripple and CLA design)

Ripple Carry Adder

1a) Half Adder

Half Adder has two input pins a and b, and two output pins sum and C_{out}(carry).

Following is the truth table of Half Adder

Truth Table

a	b	Sum	Cout
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Circuit Diagram

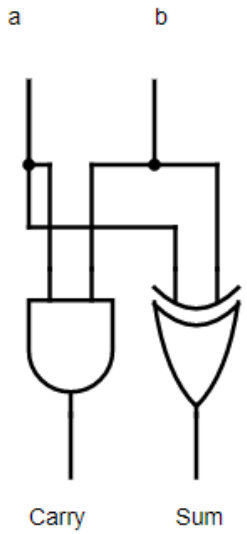


Fig 1.1: Half Adder

1b) Full Adder

The corresponding truth table is as follows:-

Full Adder has three input pins a, b and cin(C_0), and two output pins sum and carry(C_{out}).

Truth Table

a	b	Cin	Sum	Cout
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

Circuit Diagram

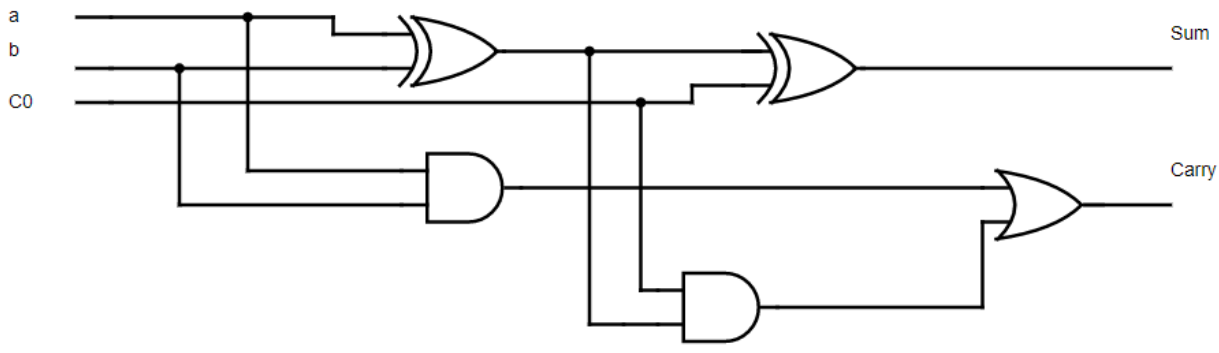


Fig 1.2: Full Adder

1c) N Bit RCA

We first design an 8 bit adder by cascading 8 Full adders, then we shall cascade two 8 bit adders to make a 16 bit adder.

Name	Longest Delay(ns)
4 Bit RCA	3.424
8 Bit RCA	6.088
16 Bit RCA	11.416
32 Bit RCA	22.072
64 Bit RCA	43.384

The delays are extracted from the synthesis report generated through Xilinx ISE 14.7 .

Circuit Diagram

- 8 Bit RCA

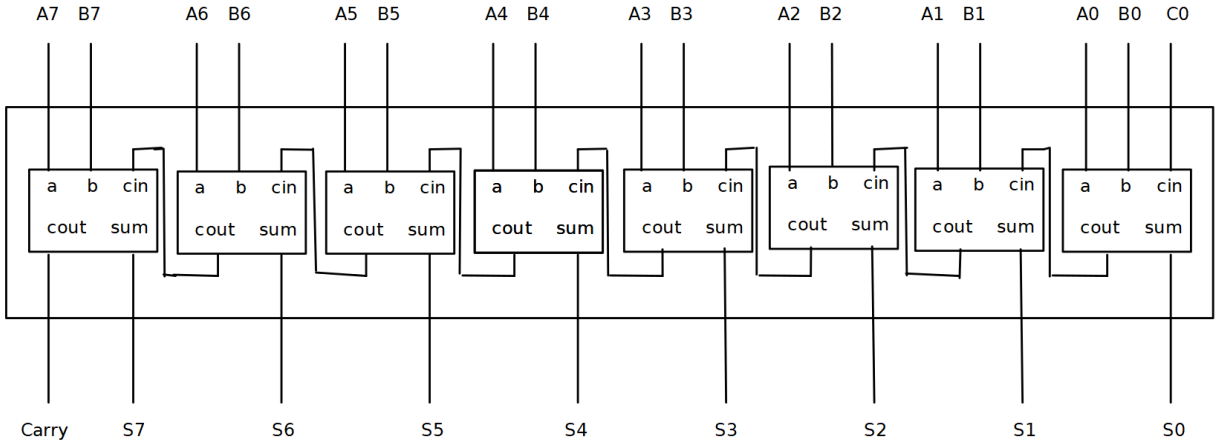


Fig 1.3: 8-bit RCA

- 16 Bit RCA

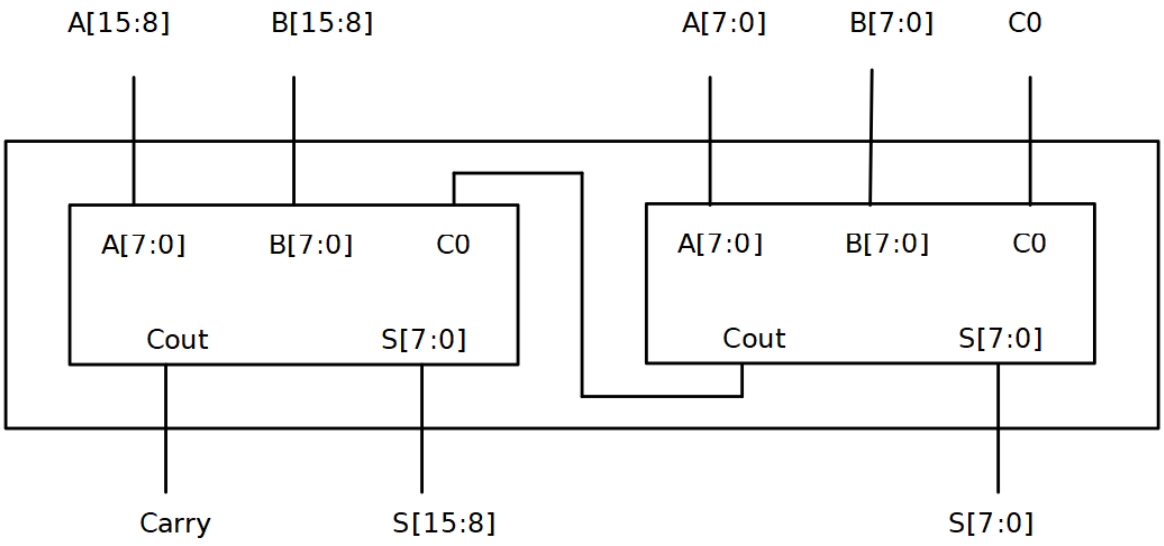


Fig 1.4: 16-bit RCA

- 32 Bit RCA

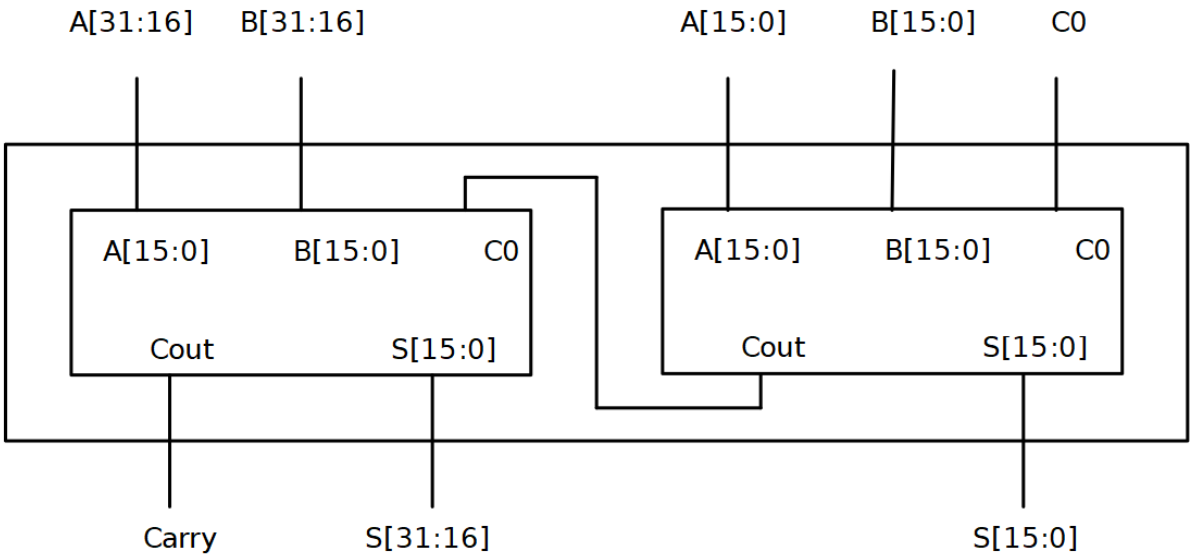


Fig 1.5: 32-bit RCA

- 64 Bit RCA

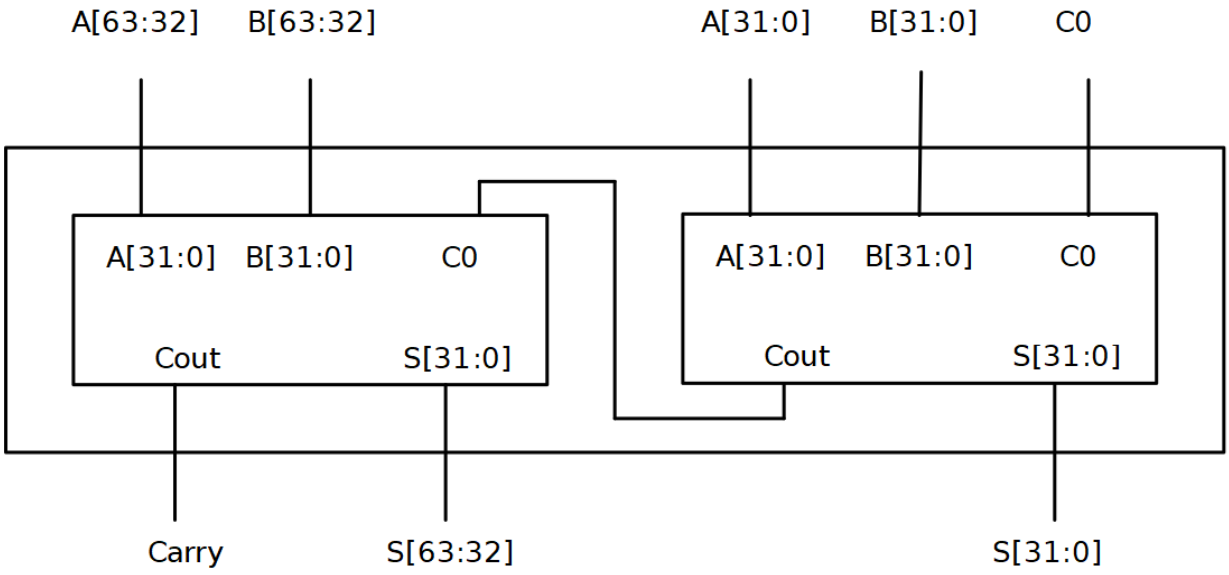


Fig 1.6: 64-bit RCA

1d) Subtractor

The subtractor can be made by altering the design of RCA. Here we introduce a 2-1 Mux taking in B(for input 0) and B′ (for input 1) with mode as input selector and we assign the carry input as mode.

Thus when we set mode as 0 the RCA works as an RCA with $C_0 = 0$. But when we change the mode to 1, the B(input) becomes B′ and C_0 goes to 1. Thus we are effectively making a 2's complement of B.

First we negate B and add 1 through Carry C_0. Therefore A + B(2's complement) = A - B. Here we can also use xor gates instead of Mux.

Circuit Diagram

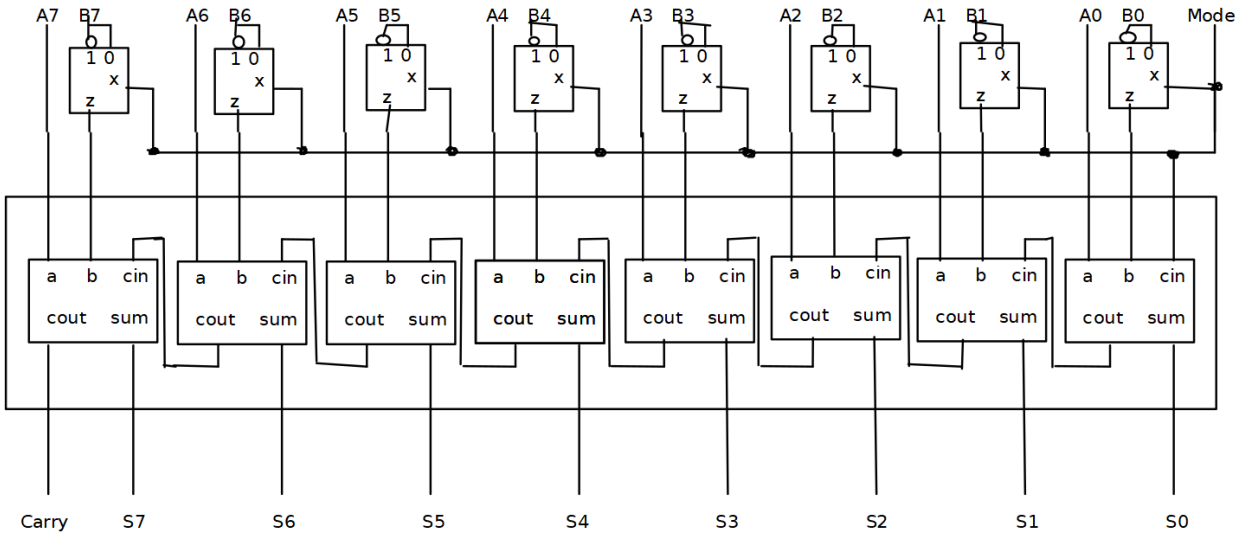


Fig 1.7: 8-bit subtractor

Carry Lookahead Adder

2a) Our previously designed ripple carry adder introduces a lot of delay due to the rippling of the carry. This is also known as a *carry chain.* However, if we complicate our logic design, we can achieve a lesser delay. Instead of waiting for the Cin from the previous full adder, we can pre calculate the carry signals in terms of A and B using a separate *Lookahead Carry Unit.*

Lets define Propagate(P) and Generate(G) signals for the ith bit as follows (Fig 2.1):

$$P_i = A_i \oplus B_i$$
$$G_i = A_i B_i$$

The new expressions for Sum(S) and Carryout(C) for the ith bit are (Fig 2.2 & 2.3):

$$S_i = P_i \oplus C_{i-1}$$
$$C_{i+1} = G_i + P_i C_i$$

For a 4-bit CLA the pre computed carry bits come out to be (Fig 2.4):

$$C_1 = G_0 + P_0 C_0$$

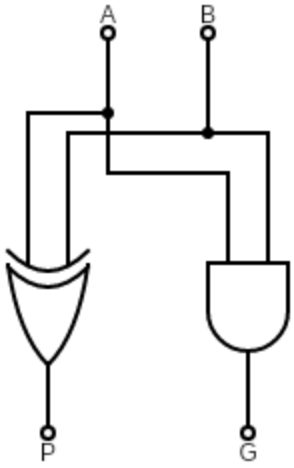
$$C_2 = G_1 + P_1 C_1 = G_1 + P_1(G_0 + P_0 C_0) \qquad \textit{[applying recursively]}$$
$$= G_1 + P_1 G_0 + P_1 P_0 C_0$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

$$C_4 = G_3 + P_3 C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G, + P_3 P_2 P_1 P_0 C_0$$

Hence C_i is clearly independent of C_{i-1} {for all i > 1} Thus the carry chain is avoided and we get a faster computation.

Circuit Diagrams:



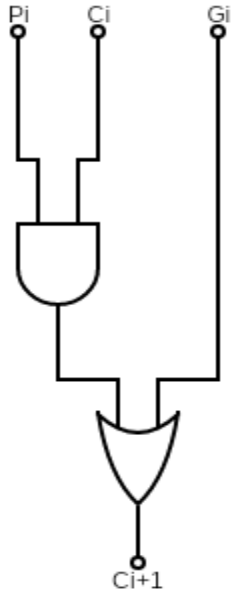


Fig 2.1 Propagate and generate Fig 2.2: Sum evaluator Fig 2.3: Carry Evaluator

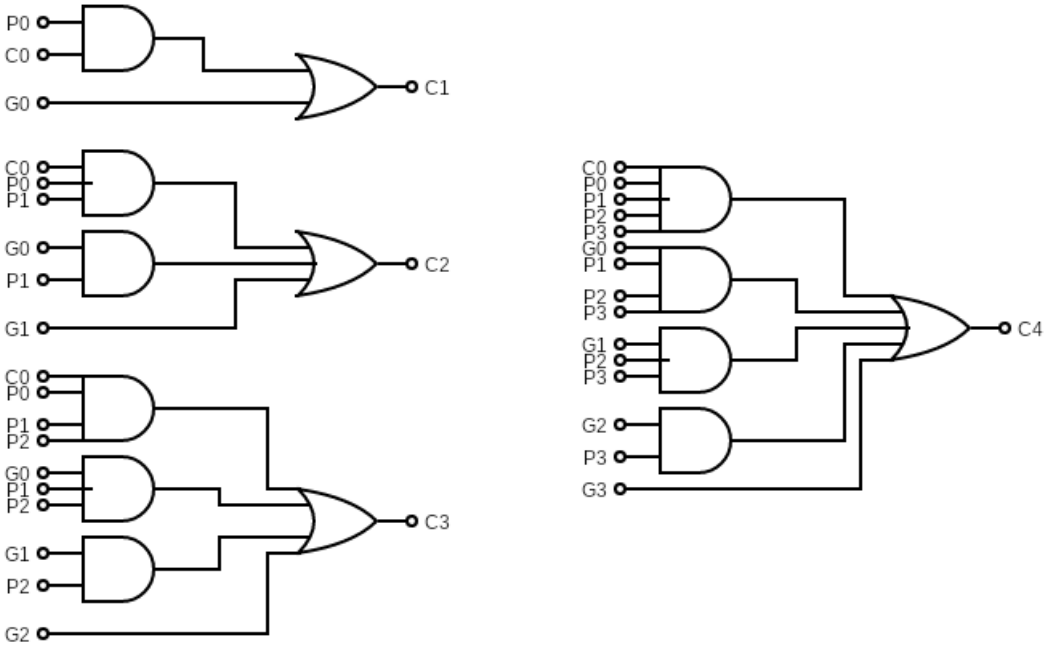


Fig 2.4: The Lookahead Carry Generator

2b) Delay outputs:

4 bit RCA: *3.424ns*
4 bit CLA: *3.252ns*

The CLA shows a lesser delay because we pre compute the delays using a separate lookahead carry generator. Using this we can avoid the carry chain and hence obtain a lesser delay. Fig 2.5 shows us the optimal design of the 4 bit CLA.

However the delay is not very different as the pre-computation part contributes to the delay as well. The difference becomes noticeable as we increase our input size. The RCA works in $O(n)$ and the CLA works in $O(log(n))$ in case of an n-bit adder. We also find considerable difference if we compare two 16-bit adders.

16 bit RCA: *11.416ns*
16 bit CLA: *5.838ns*

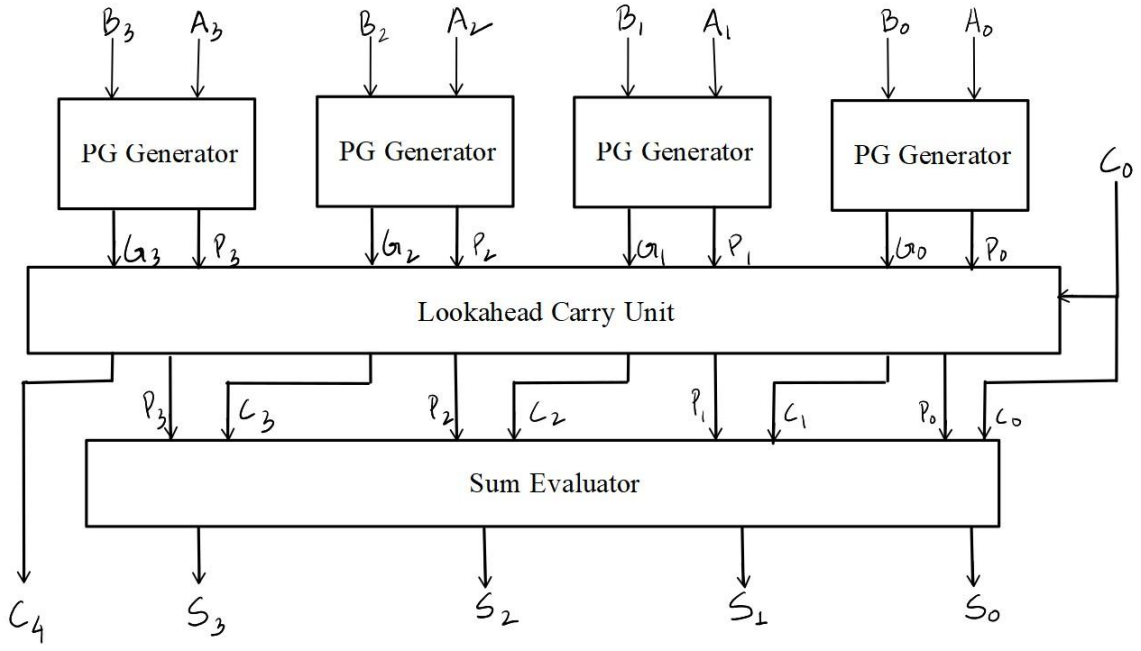


Fig 2.5: 4-bit Lookahead Carry Adder

2c)

i) Augmented 4 bit adder circuit:

Here P_bG_b gen uses the following boolean logic:-

$$P_b = P_3P_2P_1P_0$$
$$G_b = G_3 + P_3G_2 + P_3P_2G_3 + P_3P_2P_1G_0$$

This additional logic is used to output the *propagate* and *generate* signals from a 4-bit adder.

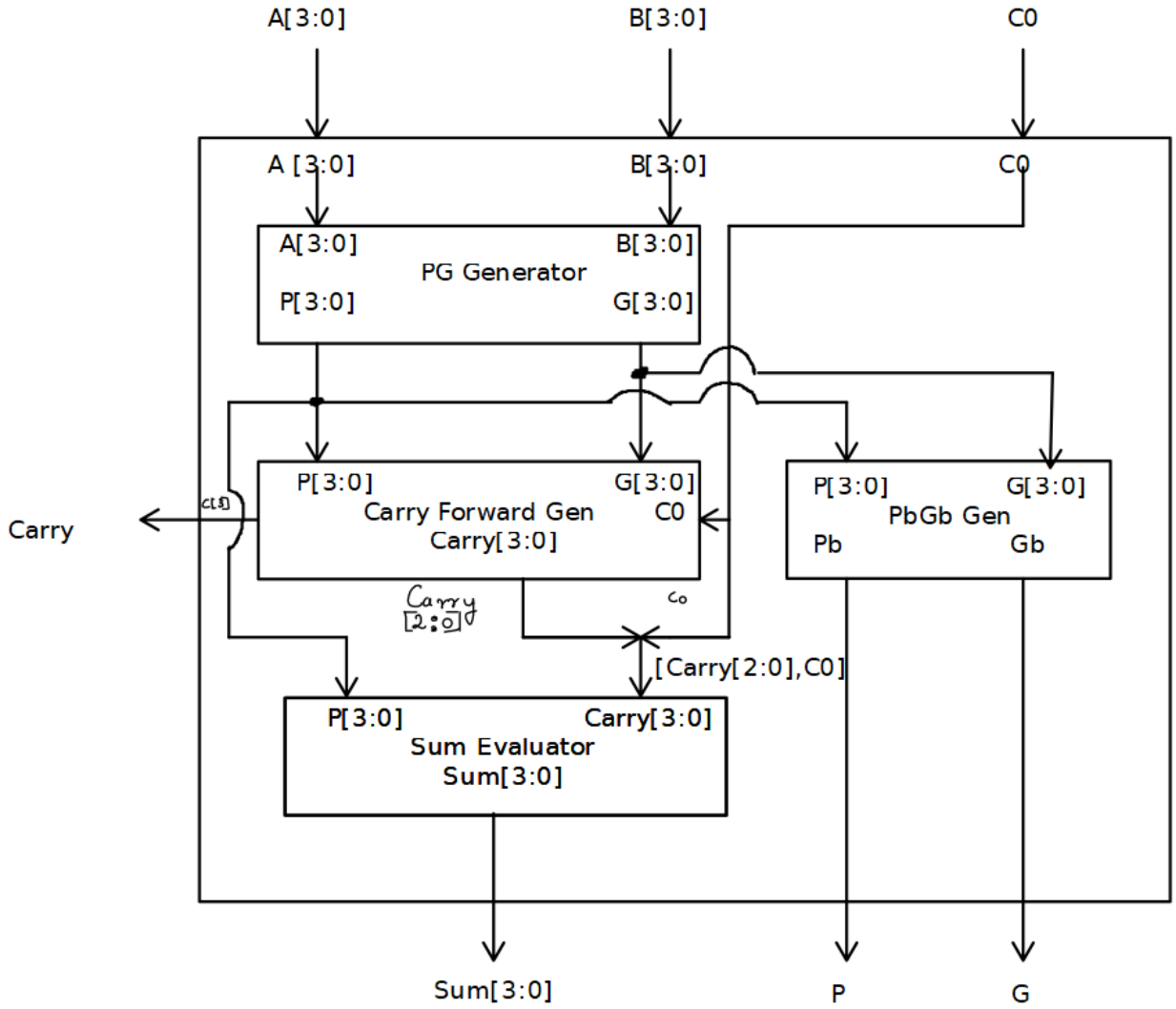


Fig 2.7 CLA with block propagate and generate signals

ii) Integration of the 4-bit lookahead carry unit. The circuit below shows a hierarchical implementation of the CLA using a Lookahead carry unit.

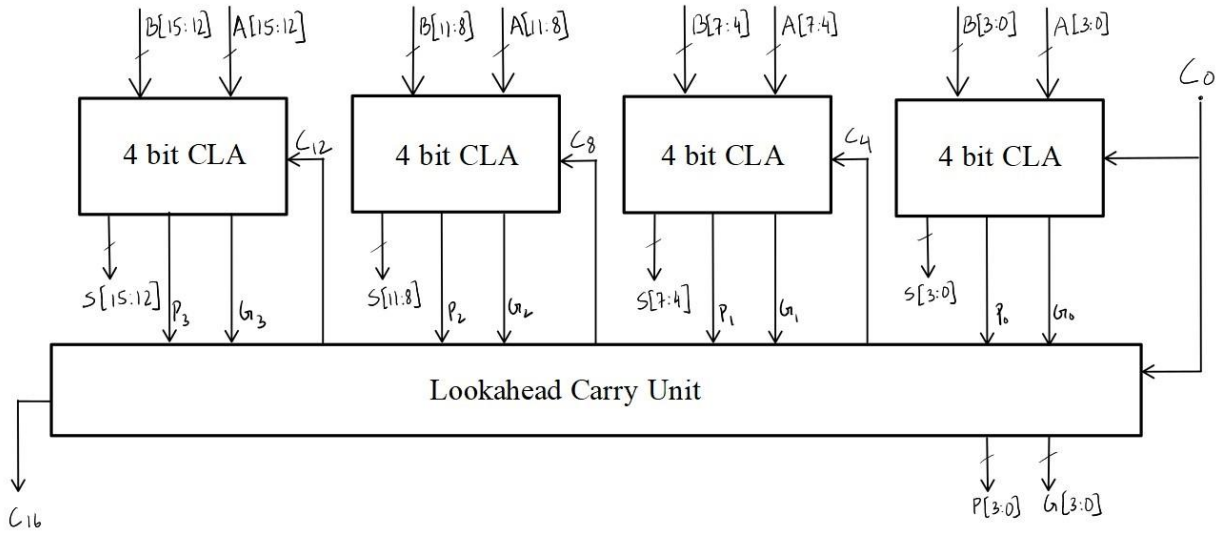


Fig 2.8: 16-bit Carry Lookahead Adder (with lookahead carry unit)

iii) Comparing the delays between the ripple design(Fig 2.6) and the one with the lookahead carry unit(Fig 2.7).

Ripple design : 9.270ns
With Carry unit: 5.838ns

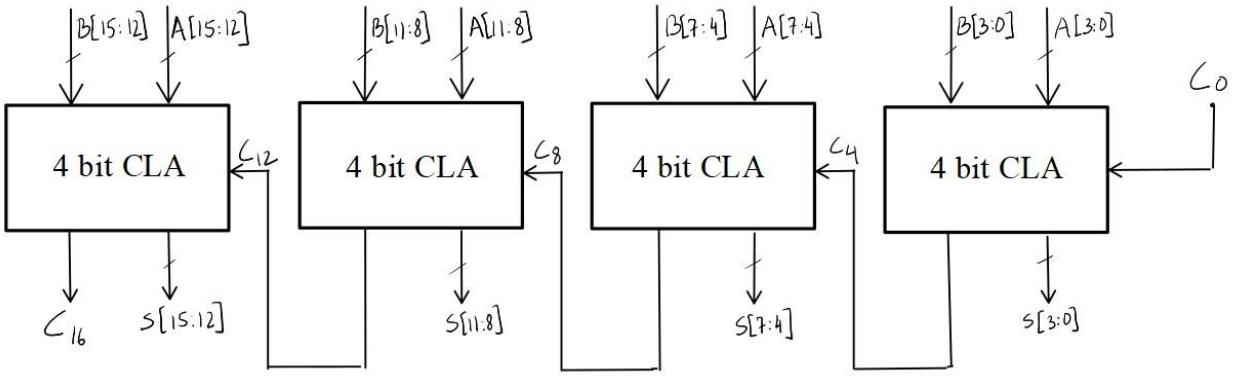


Fig 2.6: 16-bit Carry Lookahead Adder (Ripple design)

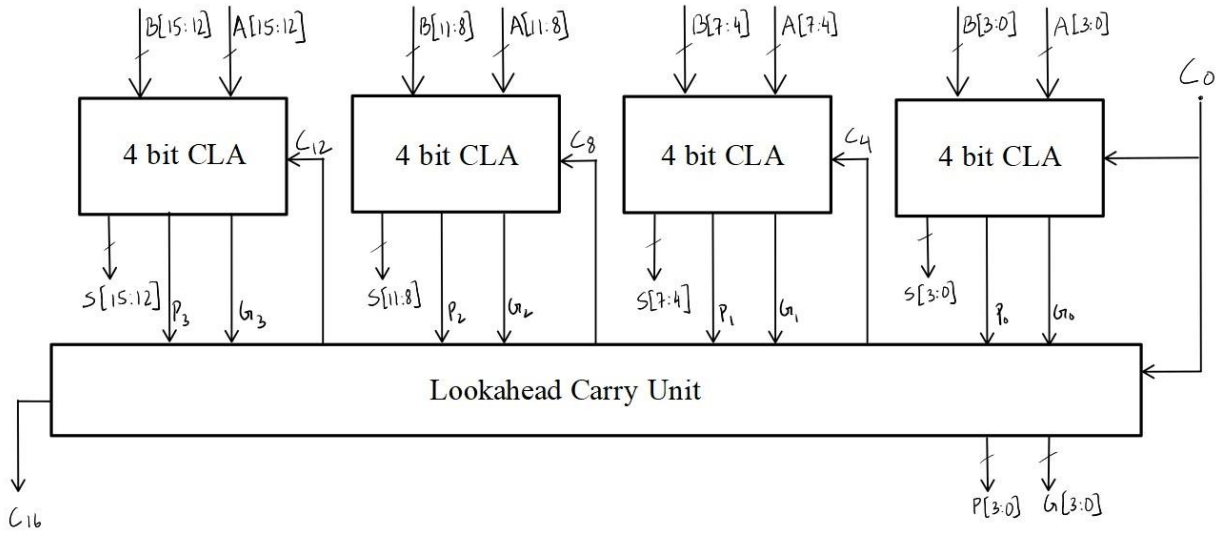


Fig 2.7: 16-bit Carry Lookahead Adder (with lookahead carry unit)

Critical Path and Delay Analysis

Clearly, the CLA with a lookahead carry unit runs faster because it precomputes the carry inputs. Roughly, the total time for computation is the time taken to generate the Propagate and Generate signals, the carry generation and finally the 4-bit CLA. Since the operation is concurrent, it takes less time.

On the other hand, the ripple design (in spite of being faster than an ordinary RCA) runs slower because each 4-bit CLA will operate only when the preceding unit finishes its operation. This introduces a kind of carry chain which accounts for the observed delay. Thus, the critical path is the one that contains all the carry signals{C_4, C_8, C_{12} and C_{15}} and passes through all the 4 CLAs. Hence it takes more time.

The detailed paths can also be found in the synthesis reports submitted along with this report.

iv) From the synthesis reports we can find the following data:

For 16-bit RCA

 # BELS : 32 {Lookup table cost}
 # Levels of logic : 36
LUT3 : 32
IO Buffers : 50
IBUF : 33
OBUF : 17

For 16-bit CLA

 # BELS : 76 {Lookup table cost}
 # Levels of logic : 16
LUT2 : 52
LUT3 : 6
LUT4 : 4
LUT5 : 10
LUT6 : 4
IO Buffers : 58
IBUF : 33
OBUF : 25

The Levels of logic are higher for a 16-bit RCA as compared to the LCA. This clearly explains the difference in the delay that has been observed between the two 16-bit adders and verifies that our results are correct. Moreover, the circuit for the 16-bit LCA is more complicated as compared to the RCA, hence the Look Up Table cost is much more for the 16-bit LCA.