# COMPUTER ORGANIZATION AND ARCHITECTURE LAB

# INTRODUCTION TO VERILOG PROGRAMMING

# VERILOG ASSIGNMENT-1

# PART-02


**GROUP 09**

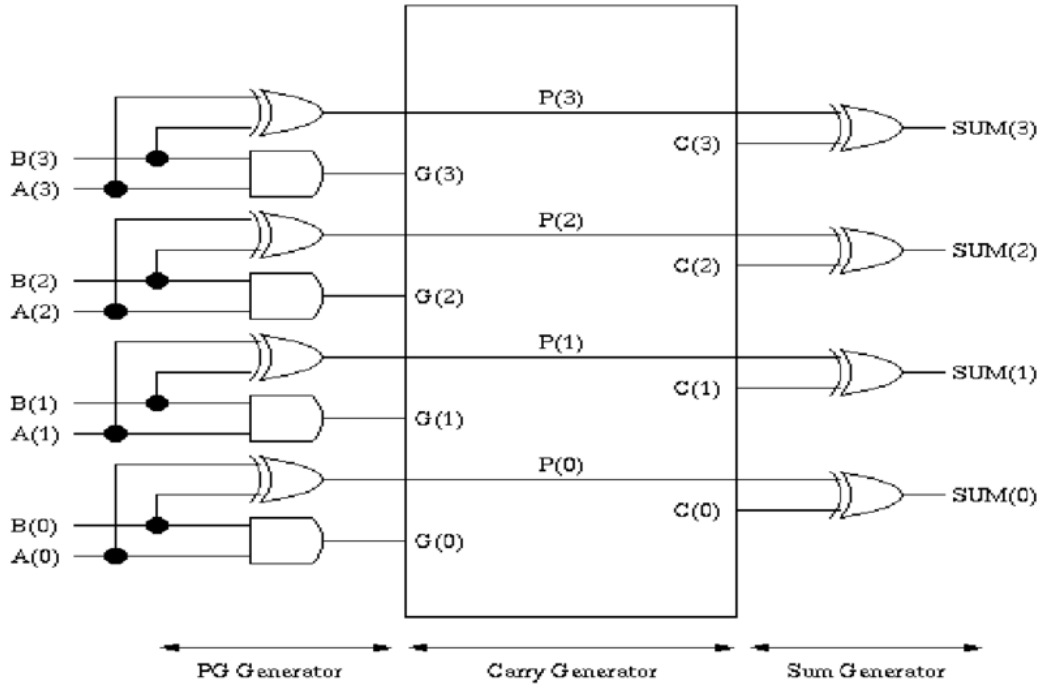
**Atishay Jain (20CS30008)**

**Abothula Suneetha (20CS10004)**


## *CARRY LOOK AHEAD ADDERS*

In the previous part of this assignment, we have designed Ripple Carry Adders but there is a large delay due to rippling of the carry. So, we wish to design high-speed adder, i.e. Carry Look-ahead Adders in this part. A carry look-ahead adder reduces the propagation delay by calculating the carry signals based on the input signals in advance.

(a) 4 bit Carry Look-ahead Adder:

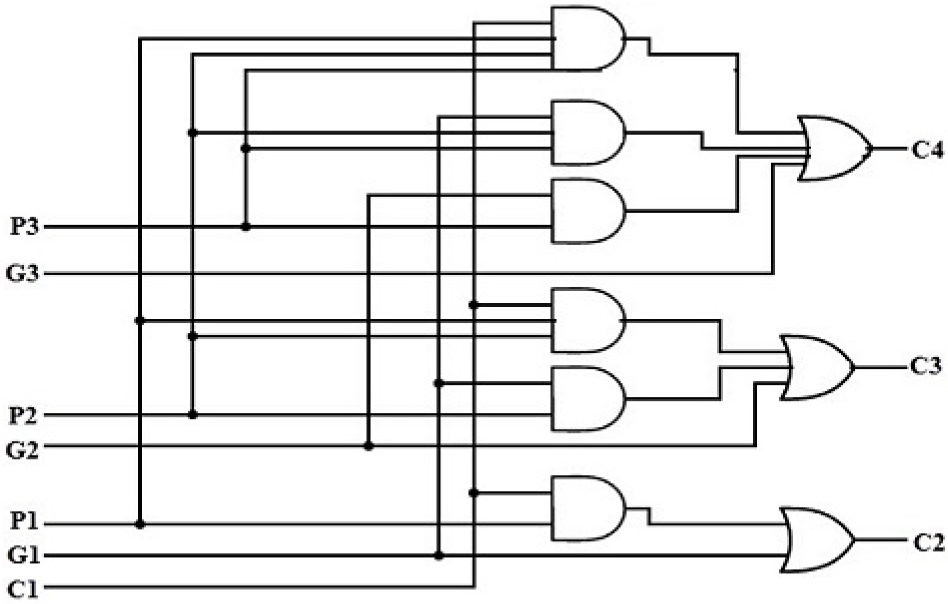



The carry lookahead adder reduces the delay in computing the sum by calculating the input carry for each full adder before hand without waiting for the previous block for to compute first.

Let us say,
A[3] A[2] A[1] A[0] and B[3] B[2] B[1] B[0] are the two 4 bit inputs
cin is the input carry
P[i] and G[i] are the carry propagate and generate signals respectively for i = 0 to 3

The equations for propagate and generate signals:
P[i] = A[i] XOR B[i];
P[i] = A[i] ^ B[i], 0 <= i <= 3
and
G[i] = A[i] AND B[i]
G[i] = A[i] & B[i], 0 <= i <= 3

Boolean equations of the Look-ahead carry generation for the 4 carry bits, C[1], C[2], C[3], and C[4] in terms of the carry generate and propagate signals are:
C[0] = cin
C[i+1] = G[i] | (P[i] & C[i]), for i = 0 to 3




On expanding, we will get

C1 = G[0] | (P[0] & C[0])

C2 = G[1] | (P[1] & G[0]) | (P[1] & P[0] & C[0])

C3 = G[2] | (P[2] & G[1]) | (P[2] & P[1] & G[0]) | (P[2] & P[1] & P[0] & C[0])

C4 = 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[0])

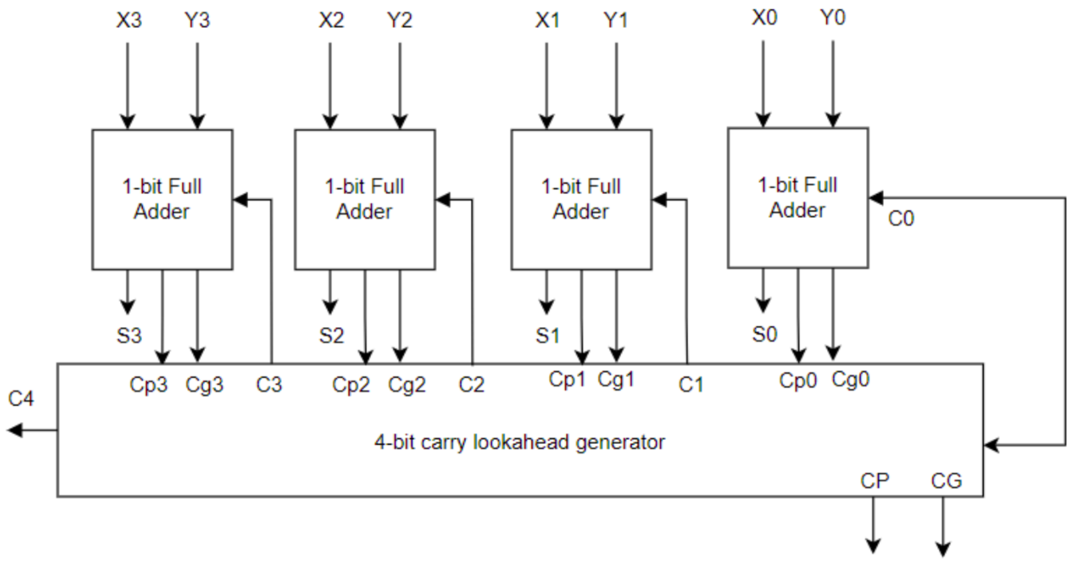
(b) Comparing 4-bit CLA with 4-bit RCA:

KEEP HIERARCHY option in the synthesizer is set TRUE

| | Time delay(in ns) | Levels of logic |
|---|---|---|
| 4-bit RCA | 2.153 | 4 |
| 4-bit CLA | 2.116 | 2 |

We observe that the 4-bit CLA is faster than the 4-bit RCA as we have eliminated the rippling of carry using generate and propagate signals.

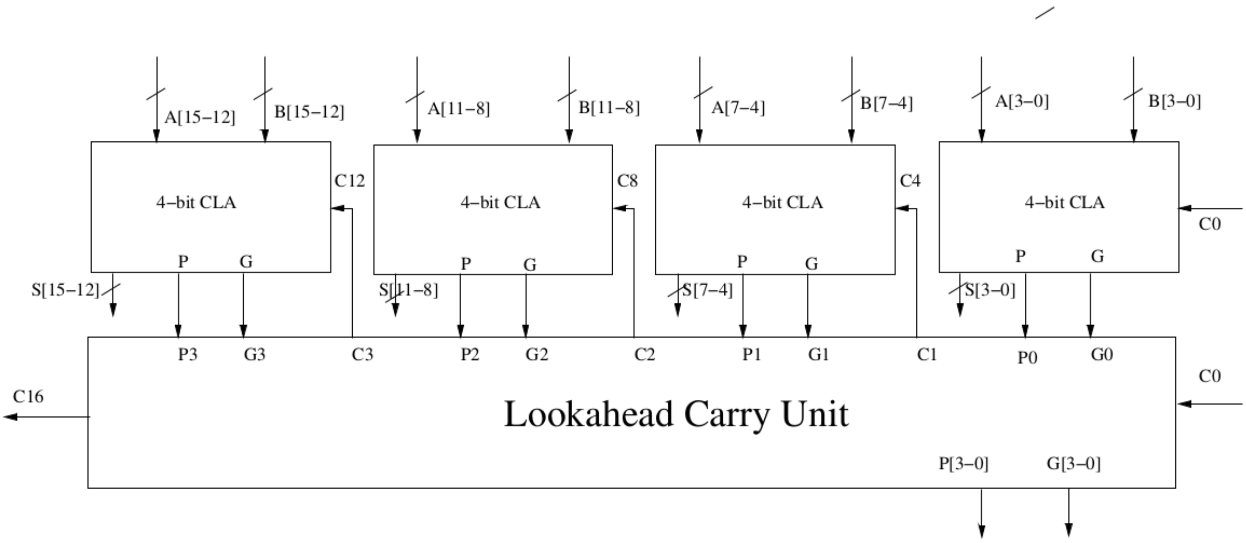
(c) Augmented 4-bit CLA:




Here, instead of carry out we give the block propagate and generate as output which are then used by the Lookahead carry unit. This design is helpful for making 16-bit CLA by combining the block propagate and generate from the lower levels instead of waiting for the carry to ripple out every time.
Let P be the bock propagate and G be the block generate

P = P[3] & P[2] & P[1] & P[0]
G = G[3] | (P[3]&G[2]) | (P[3]&P[2]&G[1]) | (P[3] &P[2]&P[1]&G[0])

(d) LookAhead Carry Unit




(e) 16 bit adder with Lookahead carry unit and with Ripple

| | No. of LUT | Time Delay (in ns) |
|---|---|---|
| With LCU | 50 | 4.207 |
| With Rippling | 31 | 4.215 |
| 16 bit RCA | 24 | 6.167 |

16bit LCU

| Constraint | Check | Worst Case Slack | Best Case Achievable | Timing Errors | Timing Score |
|---|---|---|---|---|---|
| TS_clk = PERIOD TIMEGRP "clk" 4.5 ns HIGH 50% | SETUP | 0.122ns | 4.378ns | 0 | 0 |
| | HOLD | 0.302ns | | 0 | 0 |

16bit Ripple

| Constraint | Check | Worst Case Slack | Best Case Achievable | Timing Errors | Timing Score |
|---|---|---|---|---|---|
| TS_clk = PERIOD TIMEGRP "clk" 4.5 ns HIGH 50% | SETUP | 0.216ns | 4.284ns | 0 | 0 |
| | HOLD | 0.378ns | | 0 | 0 |

4bit CLA

| | Met | Constraint | Check | Worst Case Slack | Best Case Achievable | Timing Errors | Timing Score |
|---|---|---|---|---|---|---|---|
| 1 | Yes | TS_clk = PERIOD TIMEGRP "clk" 2.2 ns HIGH 50% | SETUP | 0.327ns | 1.873ns | 0 | 0 |
| | | | HOLD | 0.228ns | | 0 | 0 |
| | | | MINPERIOD | 0.045ns | 2.155ns | 0 | 0 |

4bit CLA Augmented

| | Met | Constraint | Check | Worst Case Slack | Best Case Achievable | Timing Errors | Timing Score |
|---|---|---|---|---|---|---|---|
| 1 | Yes | TS_clk = PERIOD TIMEGRP "clk" 2.2 ns HIGH 50% | SETUP | 0.285ns | 1.915ns | 0 | 0 |
| | | | HOLD | 0.159ns | | 0 | 0 |
| | | | MINPERI... | 0.045ns | 2.155ns | 0 | 0 |