Design of 32-Bit Brent Kung Adder

Self Project

Muhammad Shoaib Iqbal
Mtech (Electronics System)
IIT-B

Date : 25 july, 2021

Introduction

Truth table for addition of 3-bits is given in the following table

A	B	C_{in}	Sum	C_{out}
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

The equations for Sum and C_{out} are

$$Sum = A \oplus B \oplus C$$

$$C_{out} = AB + BC + CA$$

- The worst case of ripple carry adder is linear in number of bits to be added
- Carry propagation is the critical path for a multi-bit adder
- To speed up the adder, an architecture is used where logic term are classified as those dependent on carry and those independent of carry bit

The sum and carry can also be written as

$$C_{i+1} = G_i + P_i.C_i \text{ , where}$$
$$G_i = A.B$$
$$P_i = A + B$$

Here computation of G_i and P_i are independent of carry, and can be generated as soon as input comes.

Tree Adder

Tree adders use the idea of carry look ahead addition. Due to tree structure, the delay is the order of $log\ n$ for n-bit adder.

$$C_{i+1} = G_i + P_i.C_i$$
$$C_{i+1} = G_i + P_i.(G_{i-1} + P_{i-1}C_{i-1})$$

Labelling the single bit G and P value as G_i^1 and P_i^1, the higher order G and P values can be written as

$$G_{i,i-1}^{2} = G_{i}^{1} + P_{i}^{1}G_{i-1}^{1}$$

$$P_{i,i-1}^{2} = P_{i}^{1}P_{i-1}^{1}$$

Hence, we can write carry as

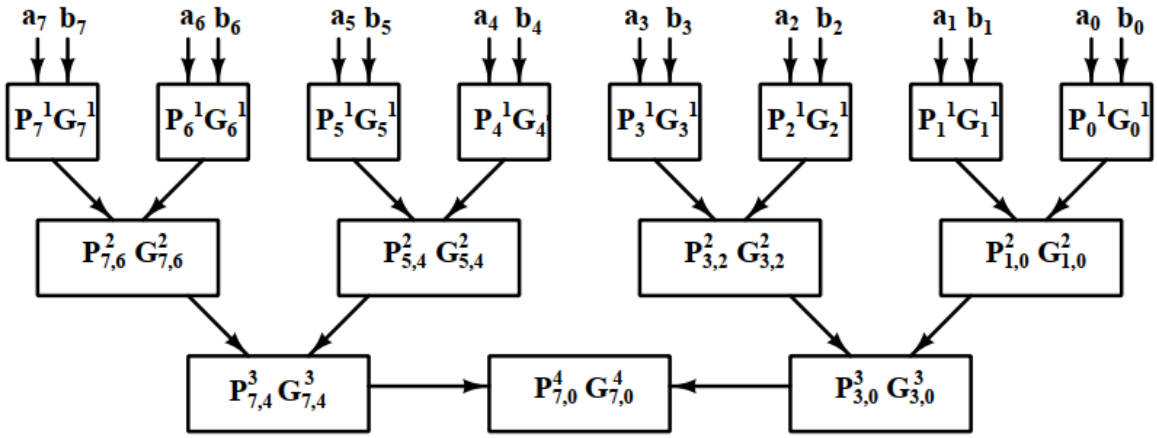
$$C_{i+1} = G_{i,i-1}^{2} + P_{i,i-1}^{2}C_{i-1}$$

Continuing the same process, we can write C_{i+1} from C_{i-3}, as

$$C_{i+1} = G_{i,i-3}^{3} + P_{i,i-3}^{3}C_{i-3}$$

Brent Kung Adder

- The brent kung adder is a logarithmic adder of low complexity
- Values of G and P are computed in a tree fashion
- The figure shows the generation of P and G for an 8-bit adder



Once P and G values are generated, carry bits can be calculated as

$$C_{1} = G_{0}^{1} + P_{0}^{1}C_{0}$$

$$C_{2} = G_{1,0}^{2} + P_{1,0}^{2}C_{0}$$

$$C_{4} = G_{3,0}^{3} + P_{3,0}^{3}C_{0}$$

$$C_{8} = G_{7,0}^{4} + P_{7,0}^{4}C_{0}$$

C_0 is the input carry and is available at t = 0. Rest all carry are calculated as

$$C_{3} = G_{2}^{1} + P_{2}^{1}C_{2}$$

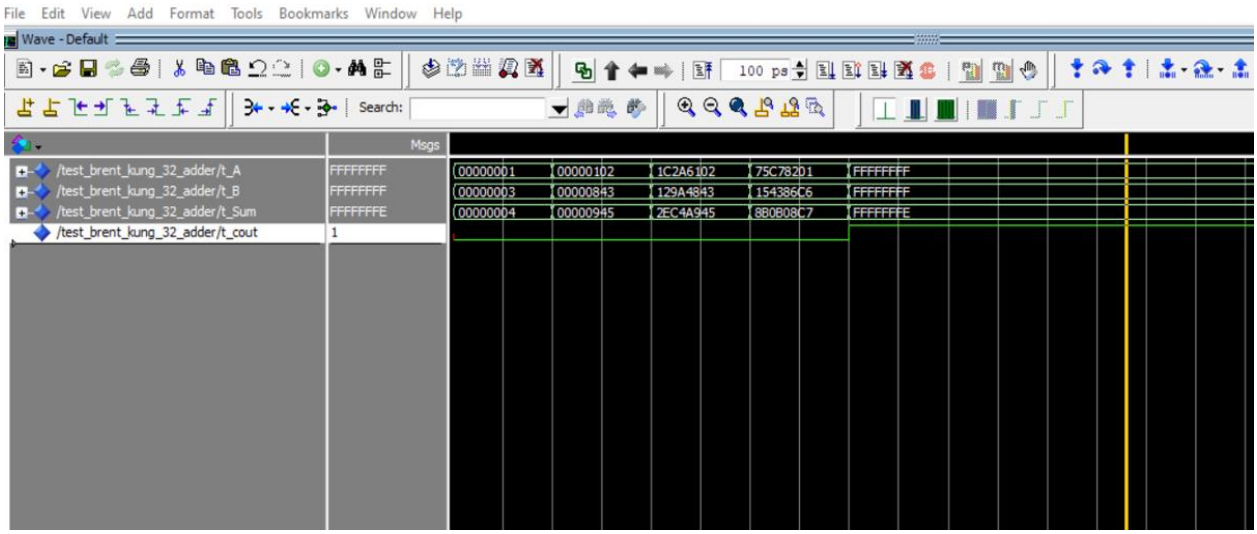
$$C_{5} = G_{4}^{1} + P_{4}^{1}C_{4}$$

$$C_6 = G_{5,4}^2 + P_{5,4}^2 C_4$$

$$C_7 = G_6^1 + P_6^1 C_6$$

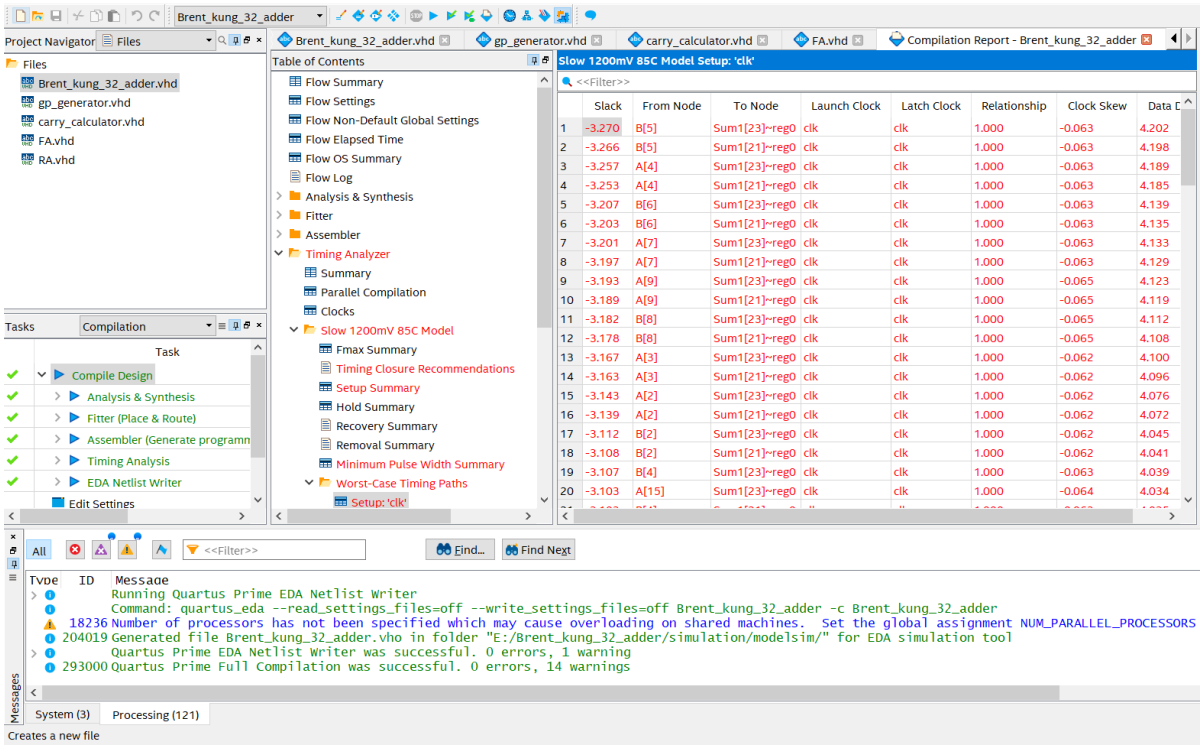
Simulation and Result

A VHDL code is written in Quartus and simulated in ModelSim. Following figure shows the result of two 32-bit numbers in ModelSim.

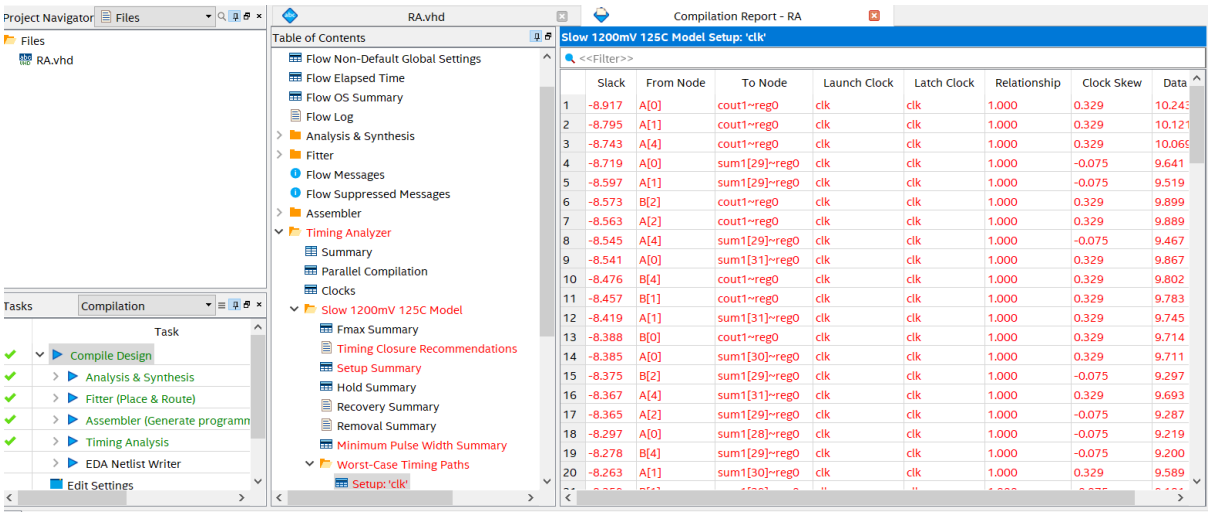


Comparison with Ripple Carry Adder using Timing Analysis

Timing analysis for 32-bit Brent Kung and 32-bit Ripple adder is done in Quartus. The result for worst case timing path for both the adders is shown in following figures.



Timing Analysis for Brent Kung Adder



Timing Analysis for Ripple Carry Adder

Conclusion

The clock-slack for Brent Kung adder is lower than that of ripple carry adder. Hence Brent Kung is faster than ripple carry adder.

References

Digital VLSI lectures by Prof. Dinesh Sharma, IIT-bombay