Carry-Skip Adder
Carry-LookAhead Adder

Review

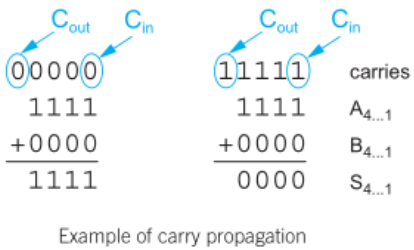
Chaitanya Tejaswi
(214102408)

Indian Institute of Technology, Guwahati
April 15, 2022

Overview

Ideas

1. Carry-Propagate Adders: C_{in} influences all subsequent bits. C_{out} is "Majority Gate".



Example of carry propagation

$$
\begin{aligned}
S: &= A\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}\bar{B}C + ABC \\
&= (A \oplus B) \oplus C = P \oplus C \\
C_{out} &= AB + AC + BC \\
&= AB + C(A + B) \\
&= \overline{\bar{A}\bar{B} + \bar{C}(\bar{A} + \bar{B})} \\
&= \mathrm{MAJ}(A,\ B,\ C)
\end{aligned}
$$

$$
\begin{aligned}
G &= AB \\
P &= A \oplus B \\
K &= \overline{AB} \\
&= \overline{A + B}
\end{aligned}
$$

full adder

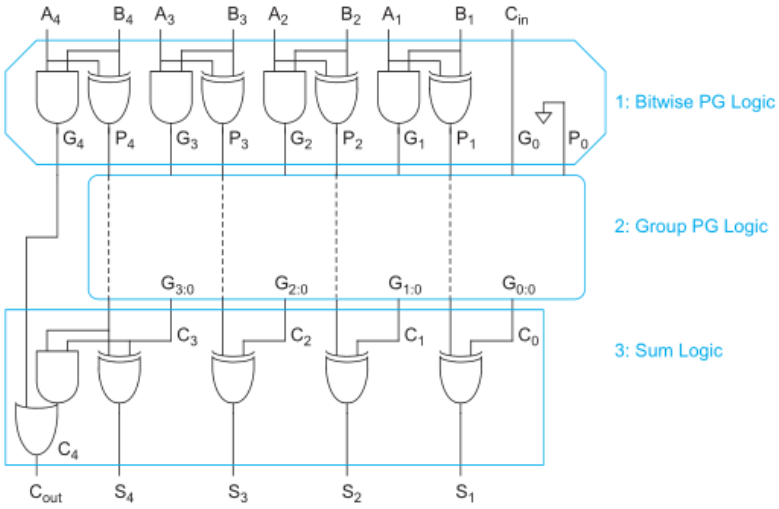
2. Propagate, Generate or Kill (PGK)
   A group of bits:
   generates a carry if its C_{out} is true independent of the C_{in}
   propagates a carry if its C_{out} is true when there is a C_{in}
   kills a carry if its C_{out} is false independent of the C_{in}

Ideas

3. Abstracting ckt implementation using Group PGK signals

For a full adder:

$$G = AB$$
$$P = A \oplus B$$
$$K = \overline{AB}$$
$$= \overline{A + B}$$



Addition with generate and propagate logic

$$G_{i:j} = G_{i:k} + P_{i:k} \cdot G_{k-1:j}$$
$$P_{i:j} = P_{i:k} \cdot P_{k-1:j}$$

where

$$G_{i:i} \equiv G_i = A_i \cdot B_i$$
$$P_{i:i} \equiv P_i = A_i \oplus B_i$$

Sum:

$$S_i = P_i \oplus G_{i-1:0}$$

Carry-Ripple Adder

C_{out} of previous stage is connected to C_{in} of next stage.

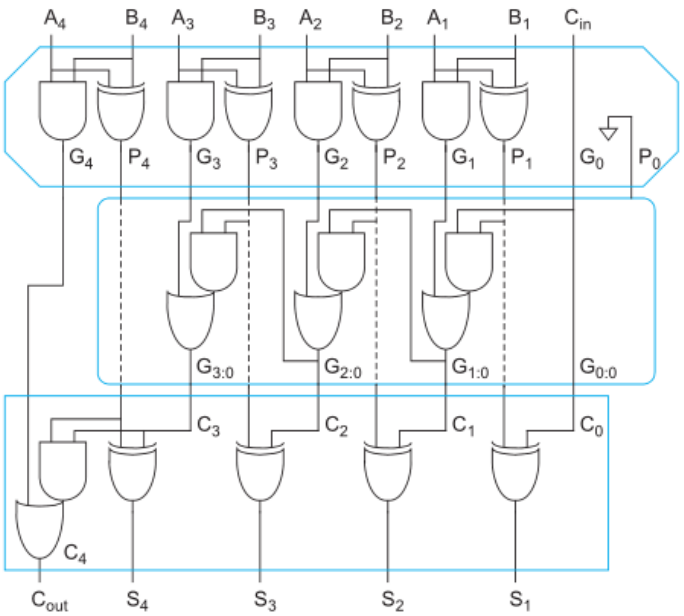


FIGURE 11.14 4-bit carry-ripple adder using PG logic

where

$$C_i = A_i B_i + (A_i + B_i) C_{i-1} \qquad G_{i:0} = G_i + P_i \cdot G_{i-1:0}$$
$$= A_i B_i + (A_i \oplus B_i) C_{i-1}$$
$$= G_i + P_i C_{i-1}$$

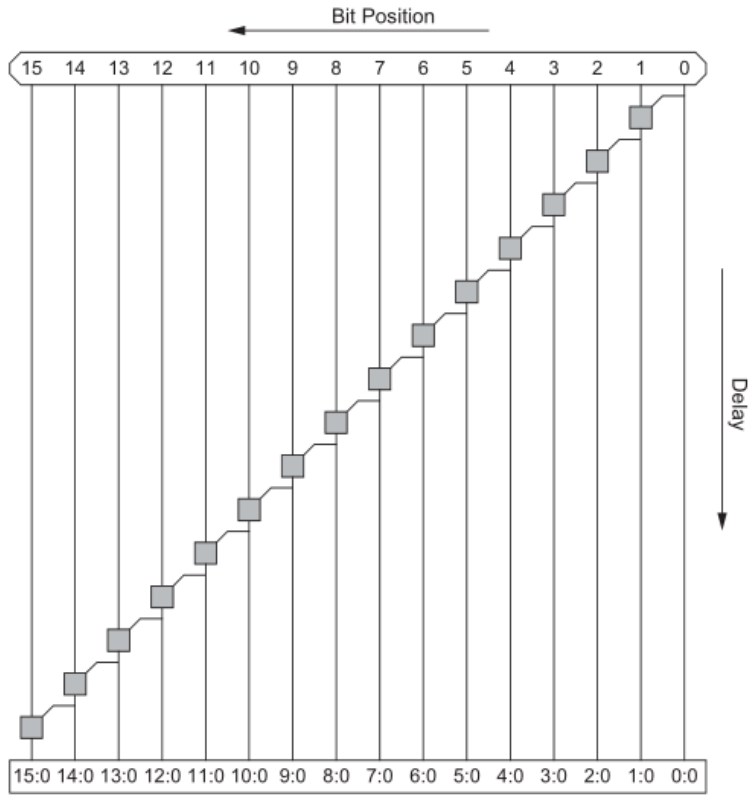


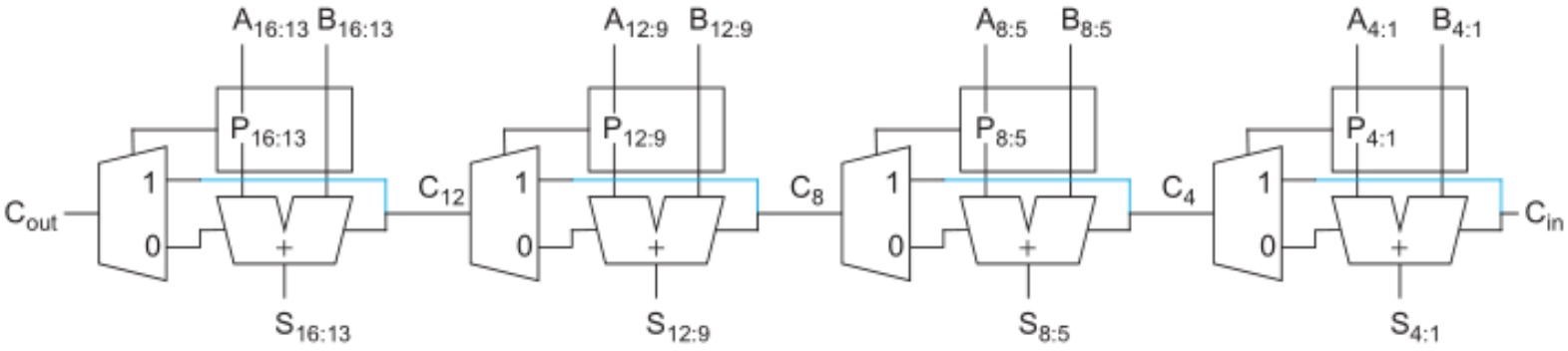
FIGURE 11.15 Carry-ripple adder group PG network

$$t_{\text{ripple}} = t_{pg} + (N-1)\, t_{AO} + t_{\text{xor}}$$

Figure: Carry-Ripple Adder: Block Diagram, CriticalPath Delay (1)

Carry-Skip Adder

Computes group propagation-delay (t_p) for each Carry chain, and uses this to skip long carry ripples.
Skip MUX selects group C_{in} if $P = 1$, C_{out} if $P = 0$.

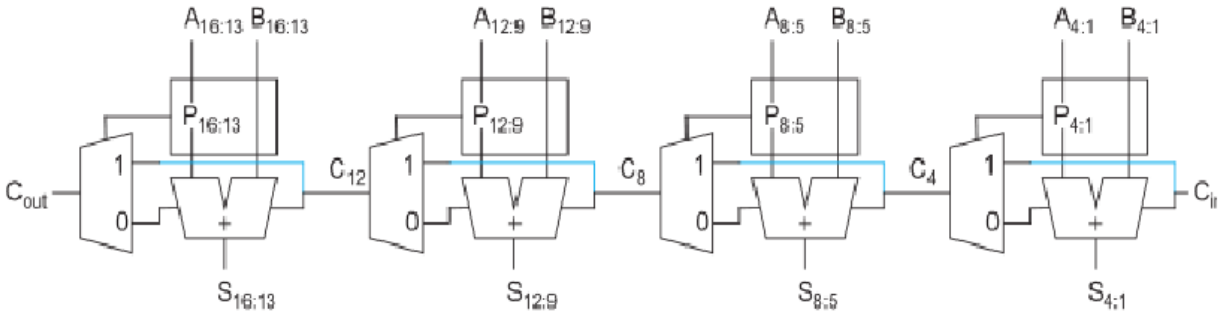


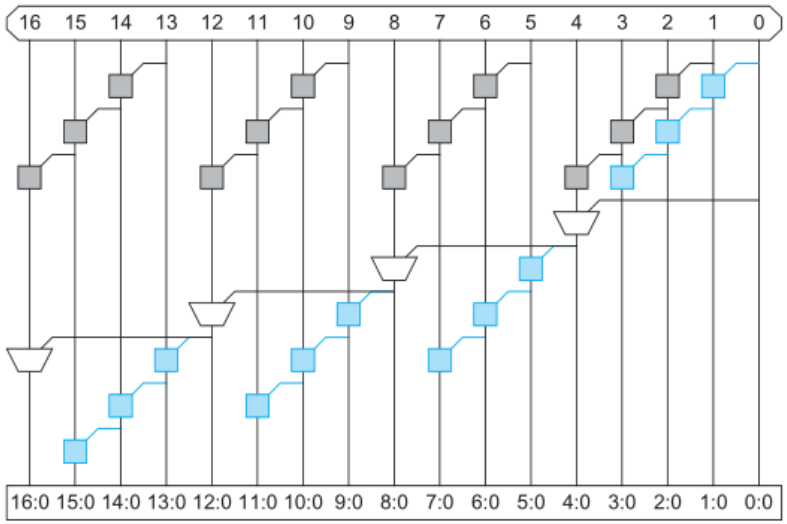
$$t_{\text{skip}} = t_{pg} + 2(n-1)t_{AO} + (k-1)t_{\text{mux}} + t_{\text{xor}}$$

Figure: Carry-Skip Adder: Block Diagram, CriticalPath Delay (1)

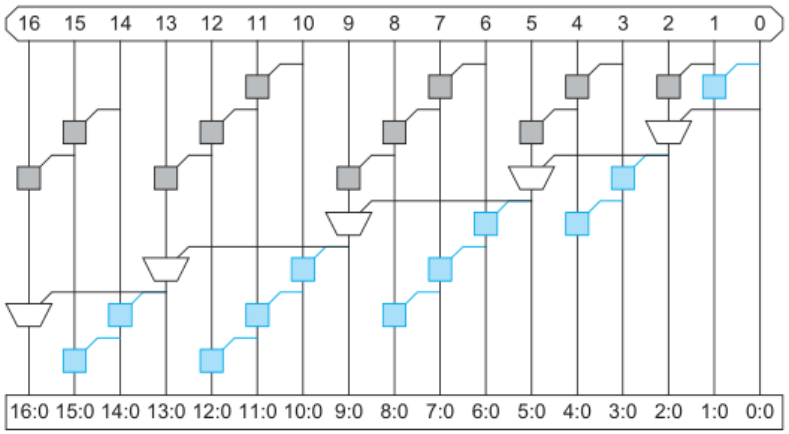
Carry-Skip Adder

Pro: Low/High Weights





Carry-skip adder PG network

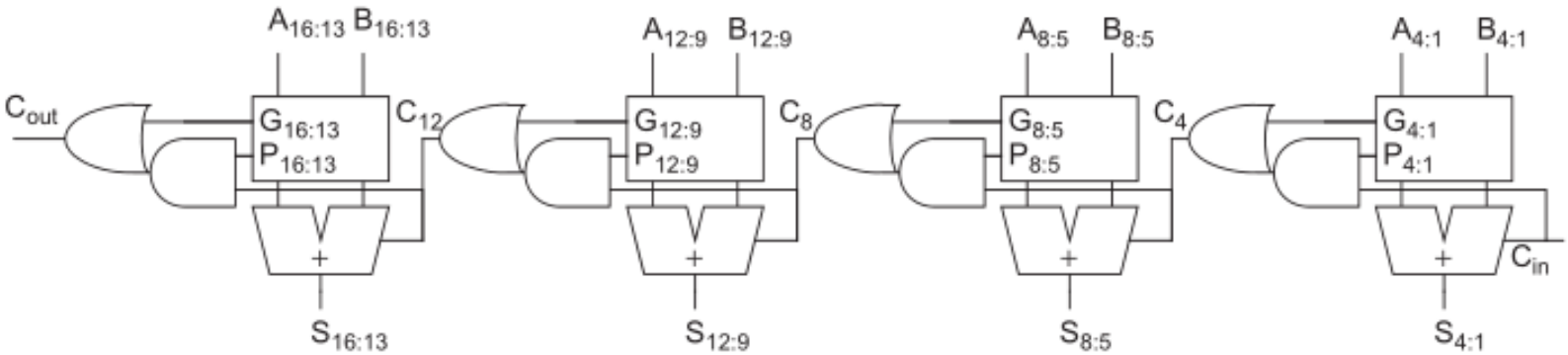


Variable group size carry-skip adder PG network

Figure: Carry-Skip Adder: Propagate/Generate (1)

Carry-LookAhead Adder

Computes group propagation & generation delays (t_p, t_g) for each Carry chain, and uses this to skip long carry ripples.
Skip MUX selects group C_{in} if $P = 1$, C_{out} if $P = 0$.



$$t_{\text{cla}} = t_{pg} + t_{pg(n)} + \left[\left(n-1\right)+\left(k-1\right)\right] t_{AO} + t_{\text{xor}}$$

Carry-LookAhead Adder

Pro: Parallelization

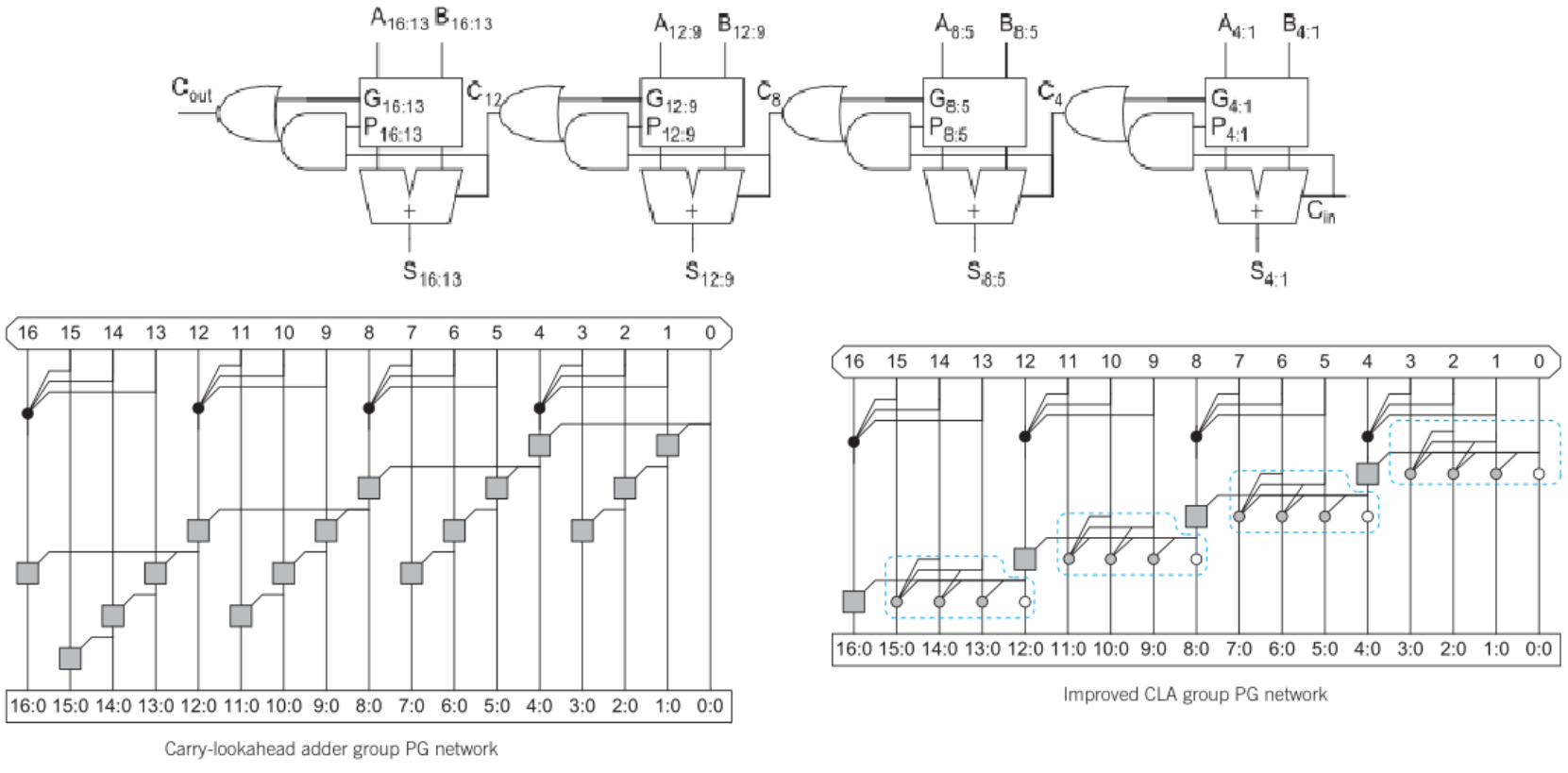


Figure: Carry-LookAhead Adder: Propagate/Generate (1)

References

[1] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*. USA: Addison-Wesley Publishing Company, 4th ed., 2010.

Thank You!