Adder Architectures

Dr. Shubhajit Roy Chowdhury,

Centre for VLSI and Embedded Systems Technology,

IIIT Hyderabad, India

Email: src.vlsi@iiit.ac.in

Outline

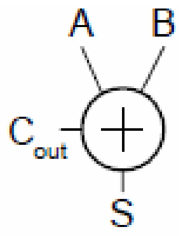
- Single bit addition
- Carry ripple adder
- Carry skip adder
- Carry look ahead adder
- Carry select adder
- Carry increment adder
- Tree adder
- Higher valency trees

Single bit addition

Half Adder

$S =$

$C_{out} =$

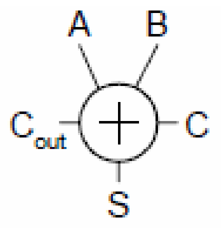


A	B	C_{out}	S
0	0		
0	1		
1	0		
1	1		

Full Adder

$S =$

$C_{out} =$



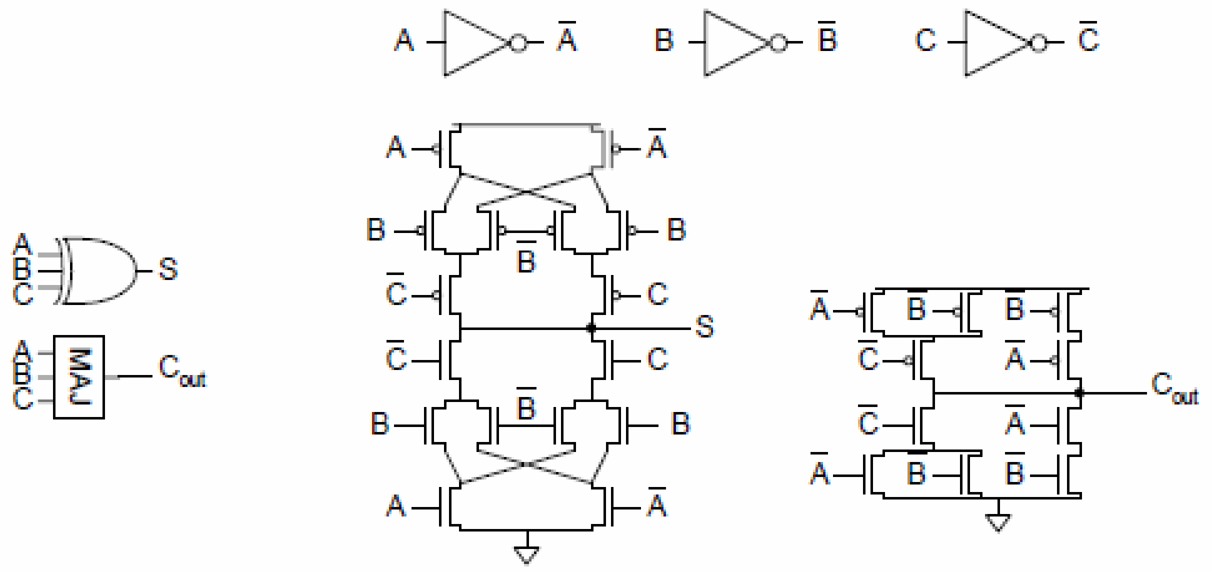
A	B	C	C_{out}	S
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Full Adder Design I

- Brute force implementation from eqns

$$S = A \oplus B \oplus C$$

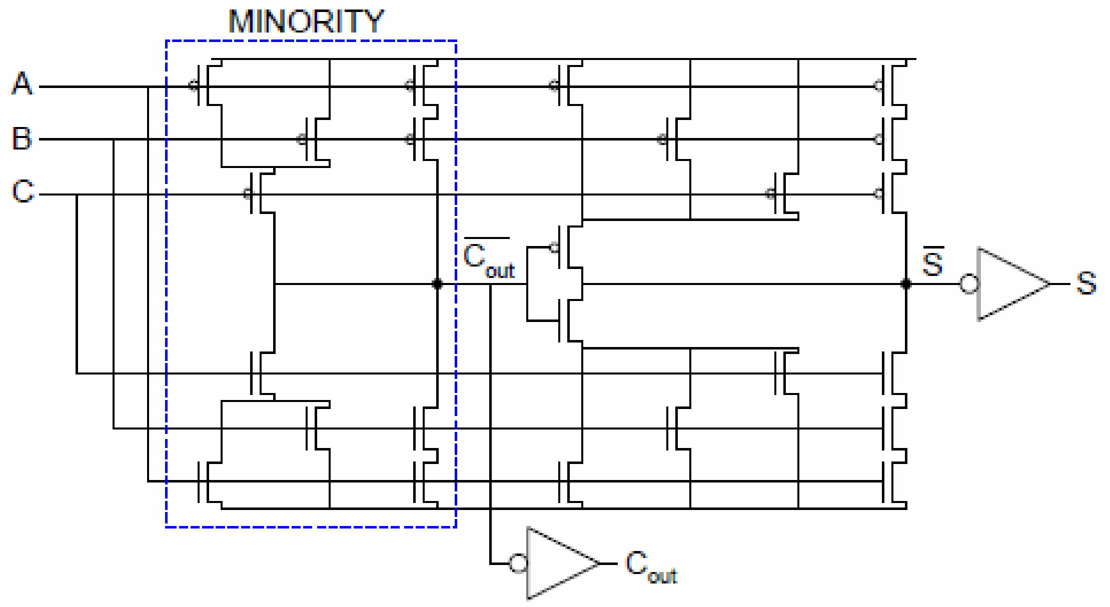
$$C_{out} = MAJ(A, B, C)$$



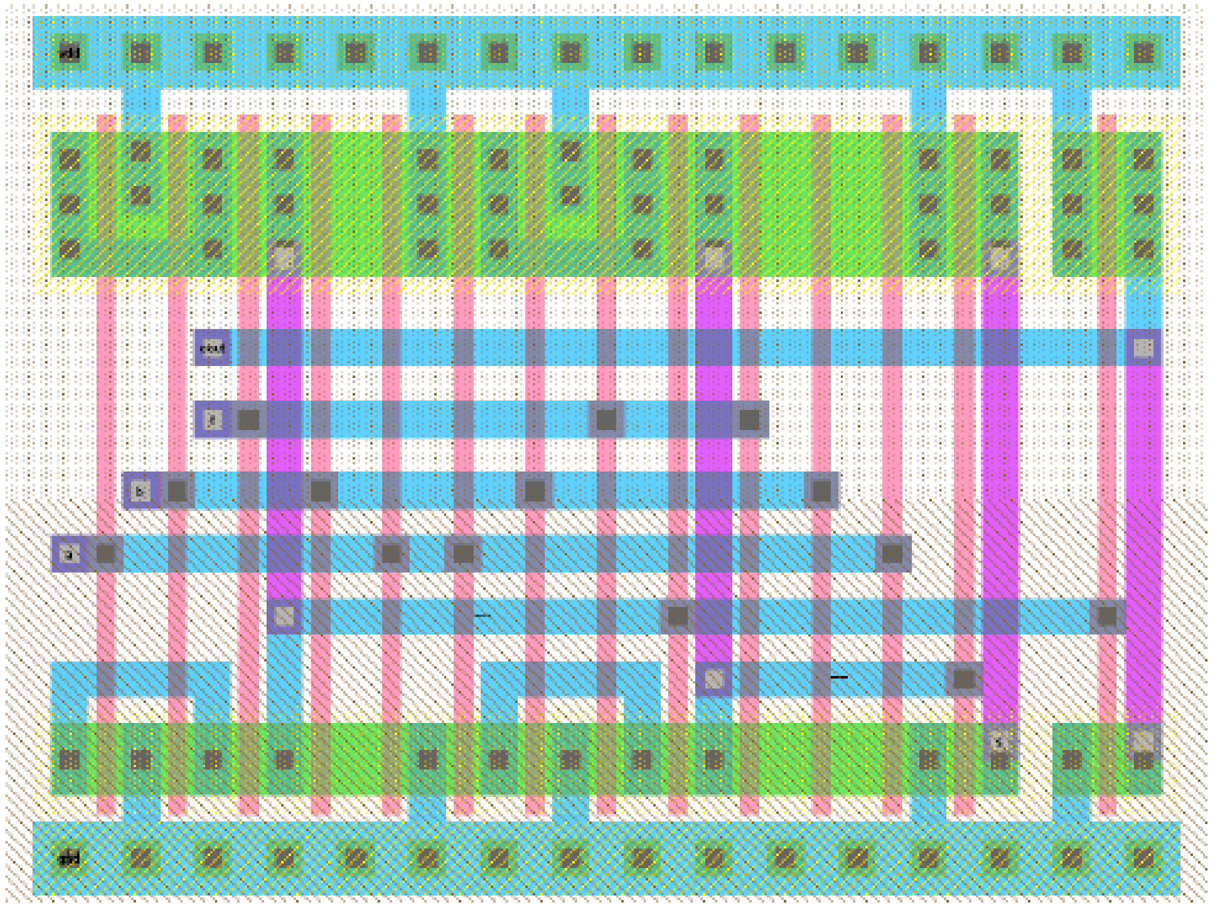
Full Adder Design II

- Factor S in terms of C_{out}

  $S = ABC + (A + B + C)(\sim C_{out})$
- Critical path is usually C to C_{out} in ripple adder

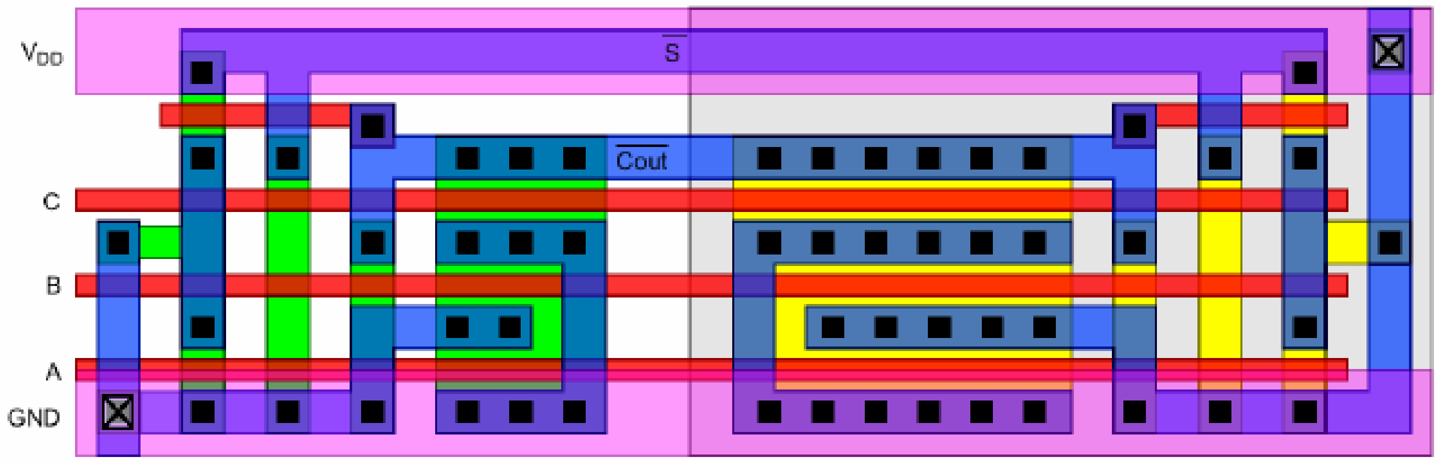


Layout of the adder



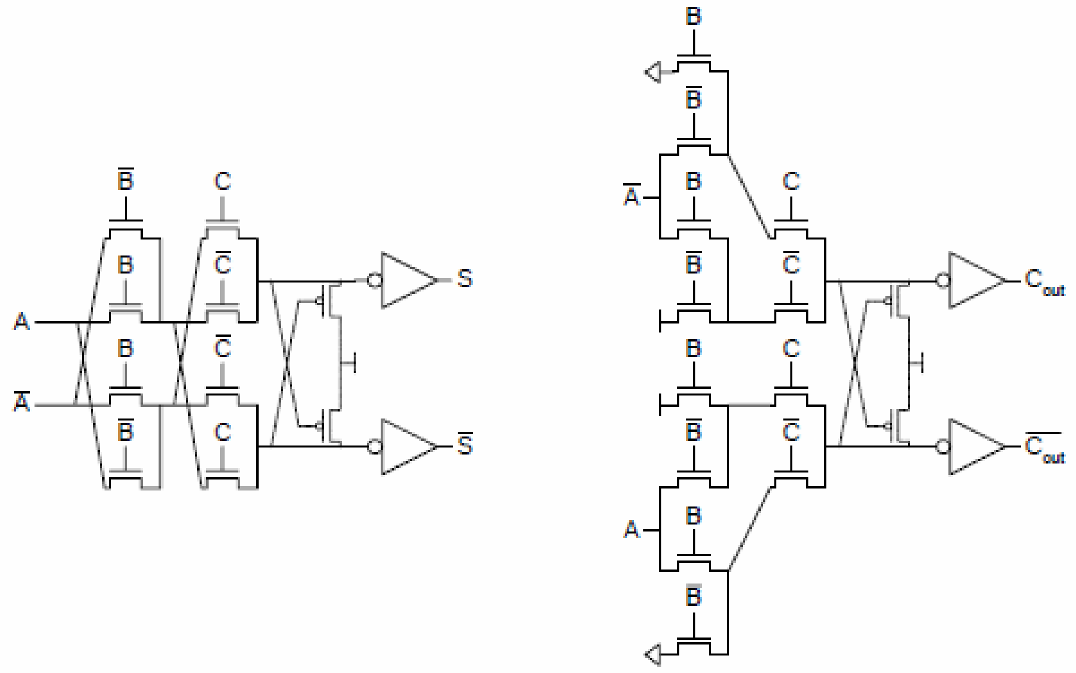
Modified Layout

- Clever layout circumvents usual line of diffusion
  - Use wide transistors on critical path
  - Eliminate output inverters

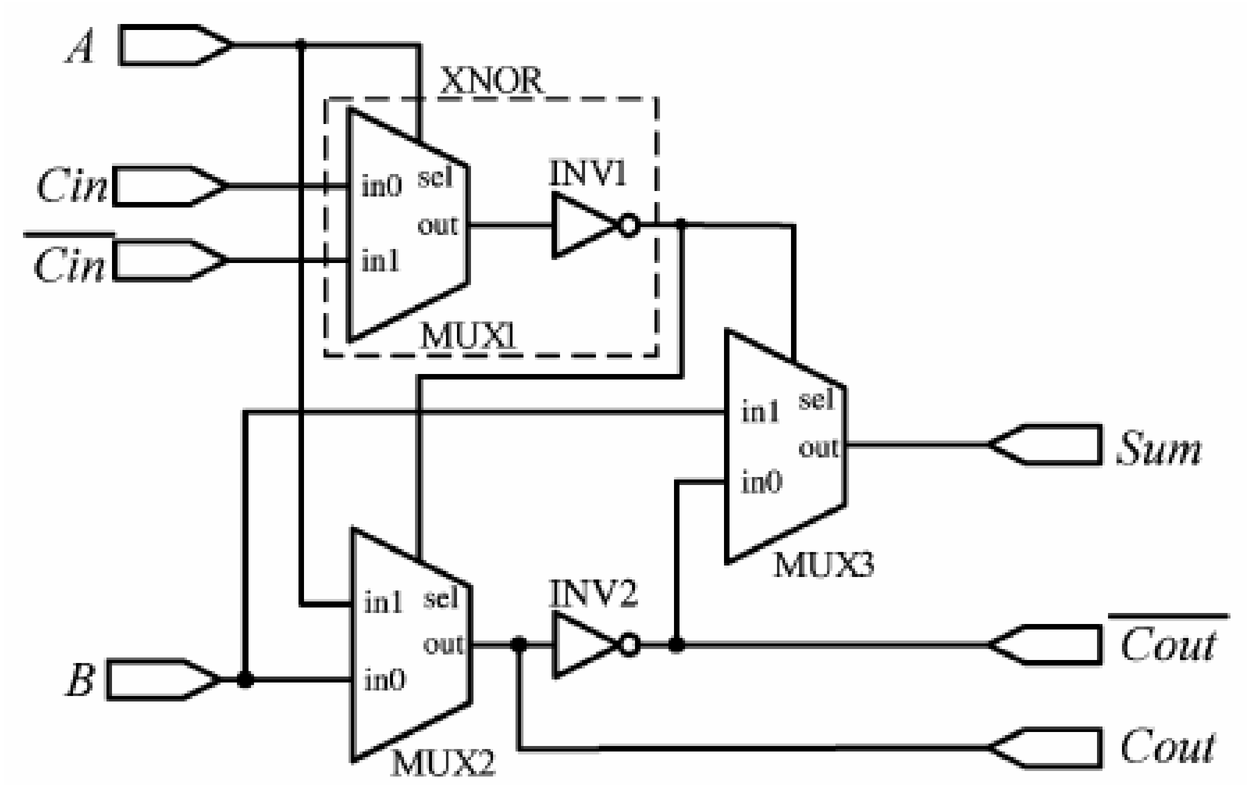


Full Adder Design III

- Complementary Pass Transistor Logic (CPL)
  - Slightly faster, but more area



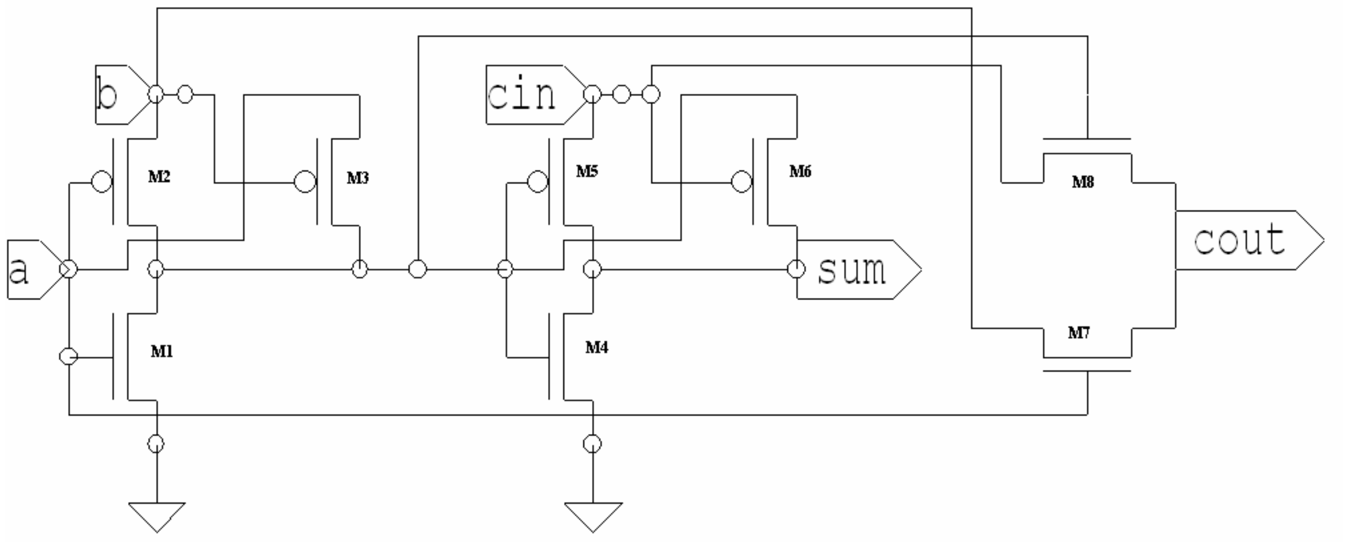
Full Adder Design IV



10 Transistor Adder

J.F. Lin et al (2007)

Full Adder Design V

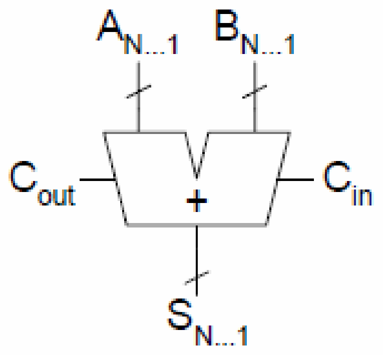


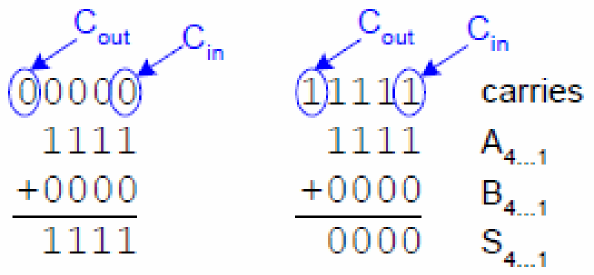
8 Transistor Adder

S. Roy Chowdhury et al (2008)

Carry Propagate Adder

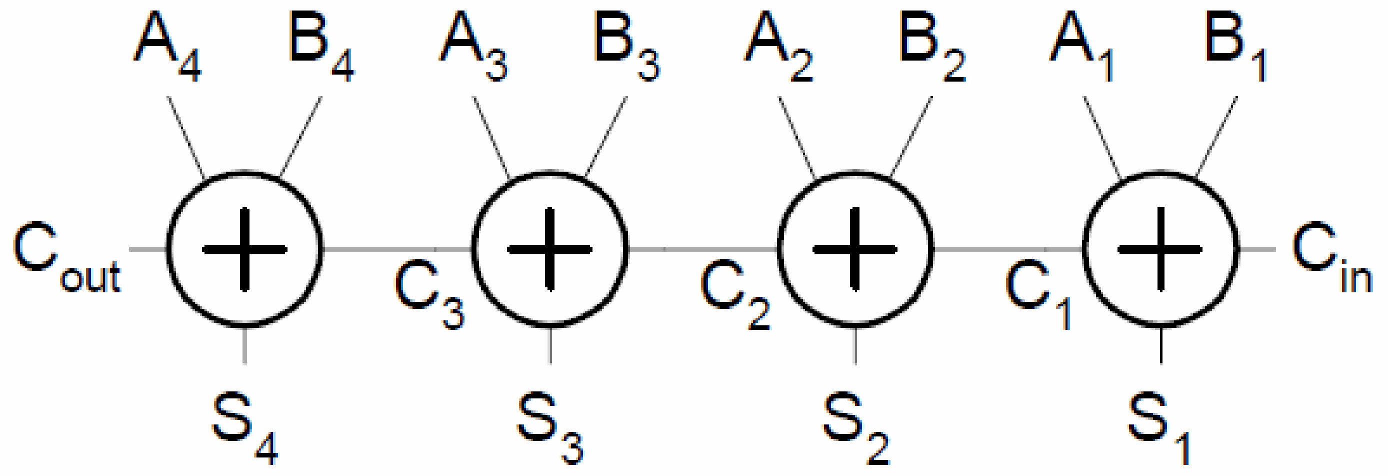
- N-bit adder called CPA
  - Each sum bit depends on all previous carries
  - How do we compute all these carries quickly?





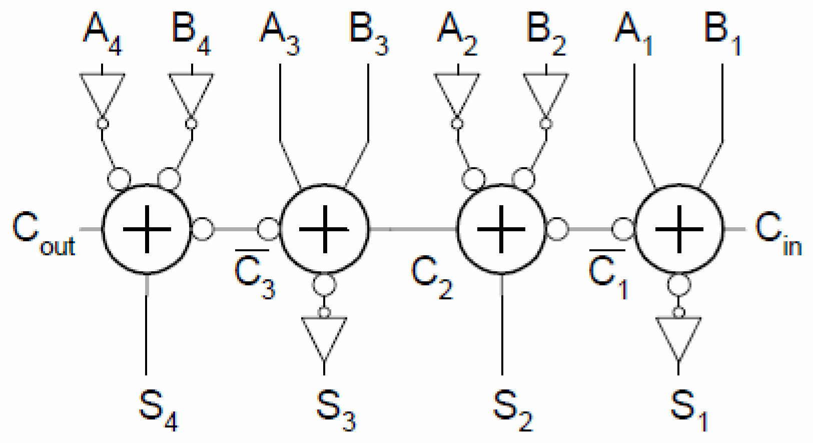
Carry Ripple Adder

- Simplest design: cascade full adders
  - Critical path goes from C_{in} to C_{out}
  - Design full adder to have fast carry delay



Inversions

- Critical path passes through majority gate
  - Built from minority + inverter
  - Eliminate inverter and use inverting full adder



Carry propagate, generate and kill

- For a full adder, define what happens to carries
  - Generate: C_{out} = 1 independent of C
    - G =
  - Propagate: C_{out} = C
    - P =
  - Kill: C_{out} = 0 independent of C
    - K =

Generate and Propagate

- Equations often factored into G and P
- Generate and propagate for groups spanning i:j

$$G_{i:j} =$$

$$P_{i:j} =$$

- Base case

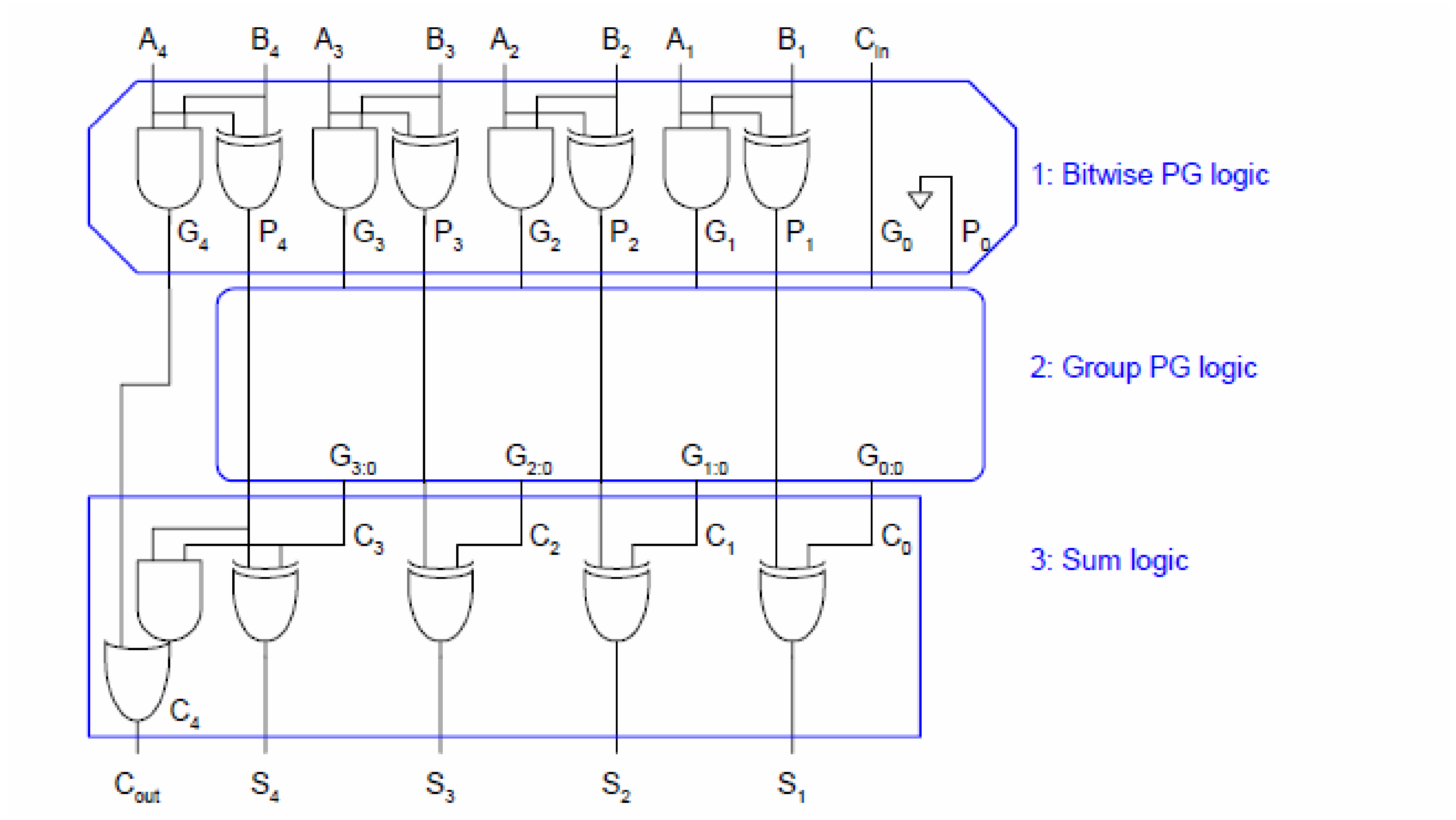
$$G_{i:i} \equiv G_i = \qquad G_{0:0} \equiv G_0 =$$

$$P_{i:i} \equiv P_i = \qquad P_{0:0} \equiv P_0 =$$

- Sum:

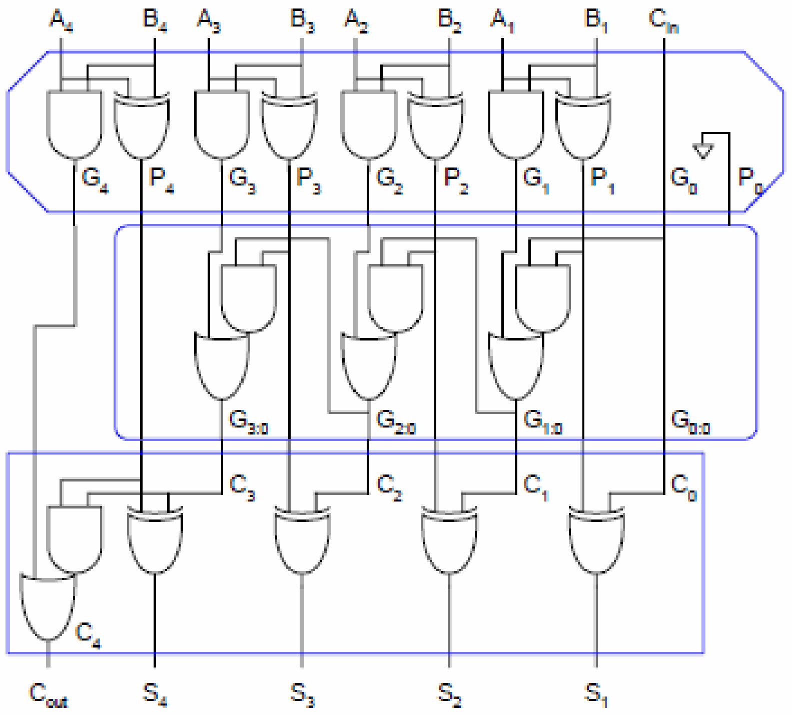
$$S_i =$$

Propagate Generate Logic



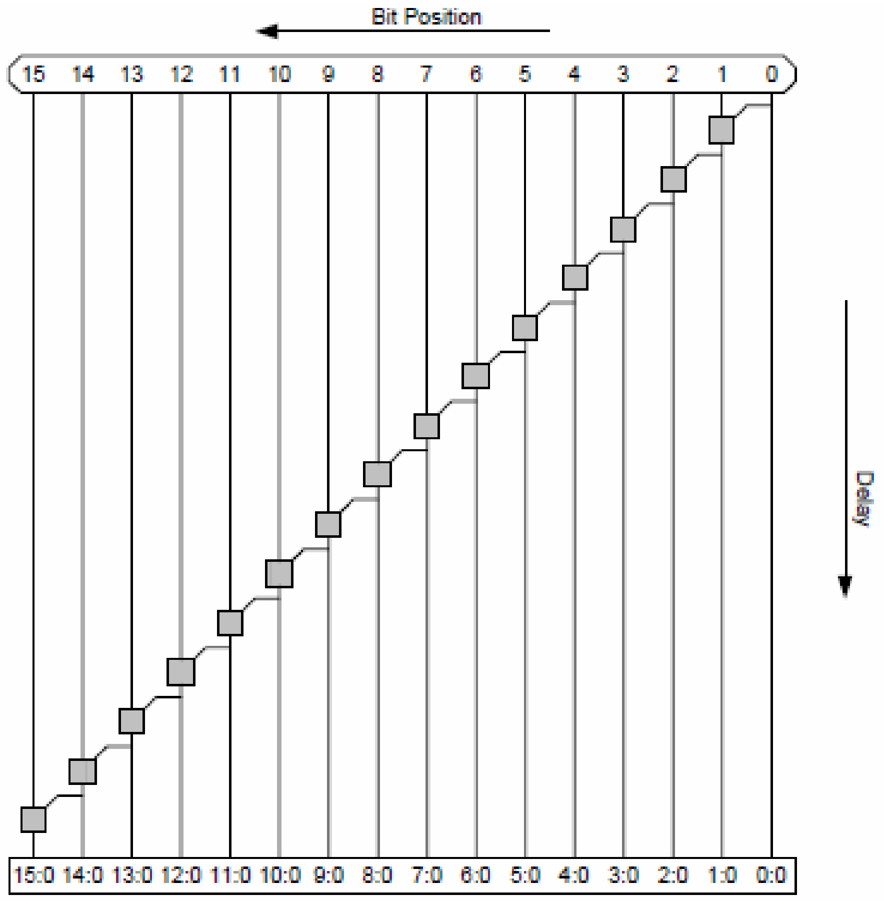
Carry Ripple Revisited

$$G_{i:0} = G_i + P_i \bullet G_{i-1:0}$$

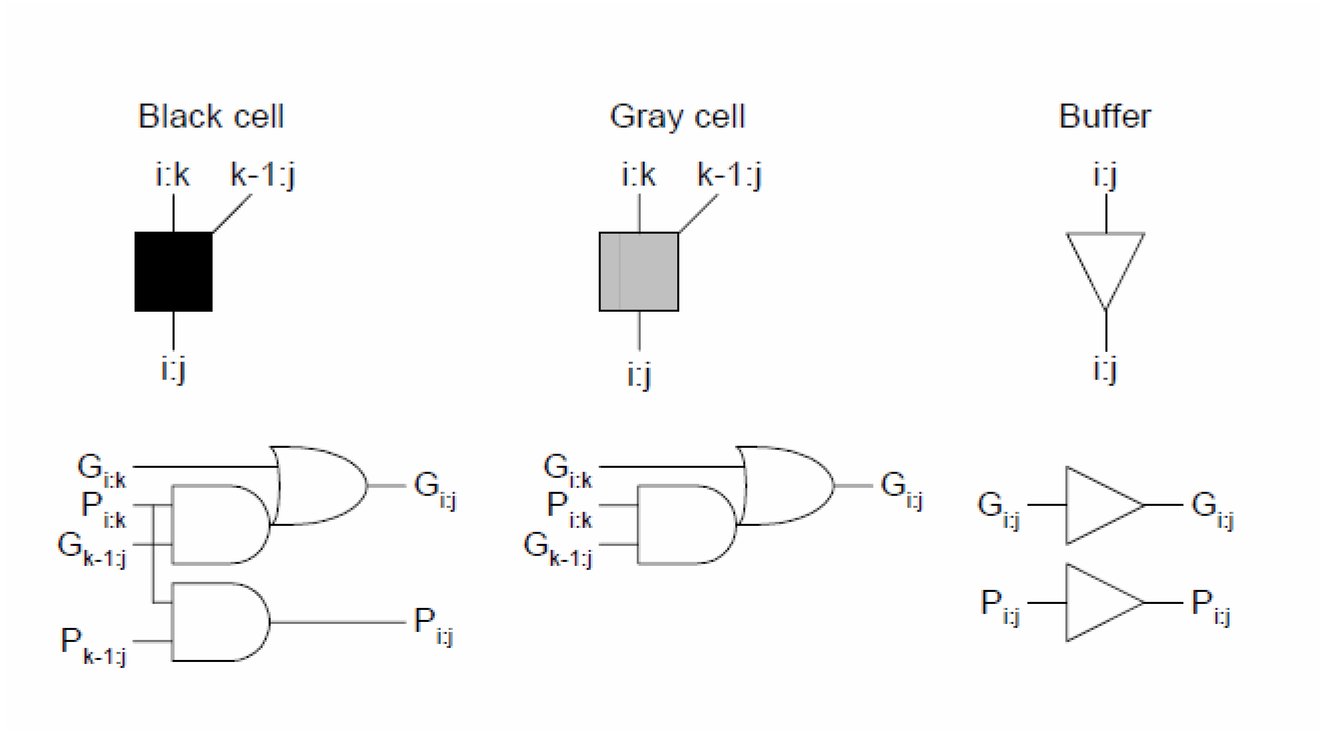


Carry ripple PG diagram

$t_{\text{ripple}} =$

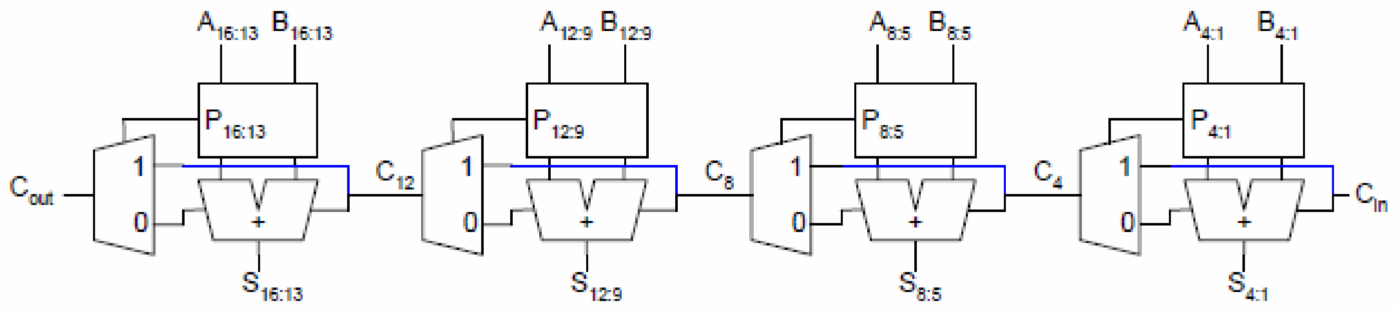


PG diagram notation

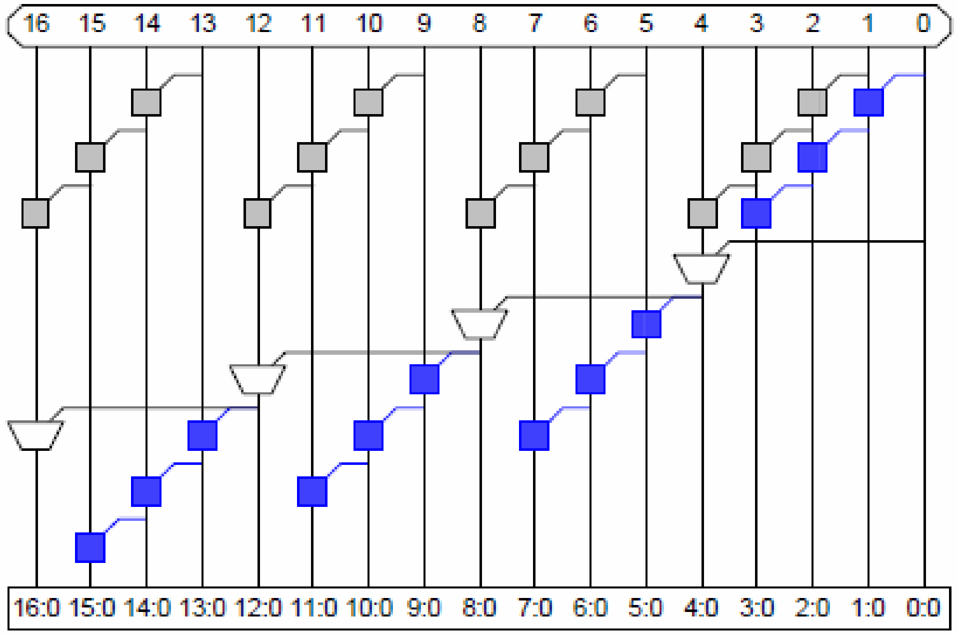


Carry Skip Adder

- Carry-ripple is slow through all N stages
- Carry-skip allows carry to skip over groups of n bits
  - Decision based on n-bit propagate signal



PG diagram of Carry Skip Adder

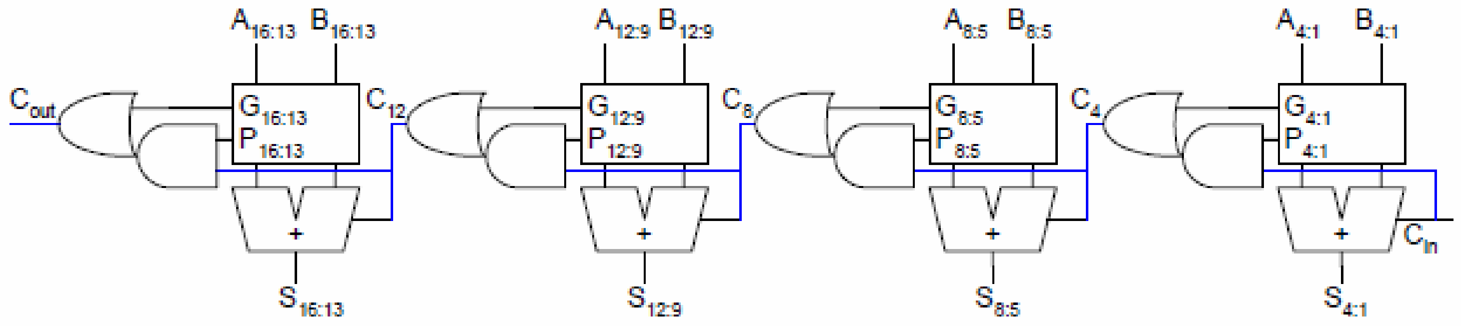


For k n-bit groups (N = nk)

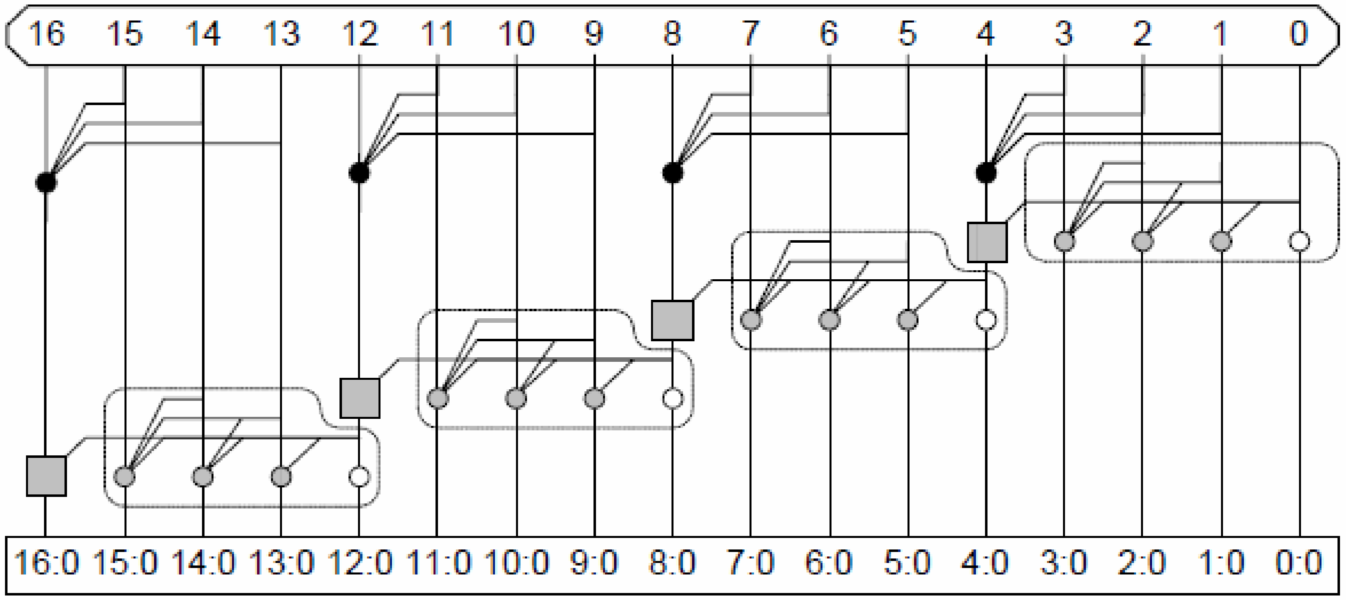
$t_{\text{skip}} =$

Carry Look Ahead Adder

- Carry-lookahead adder computes $G_{i:0}$ for many bits in parallel.
- Uses higher-valency cells with more than two inputs.

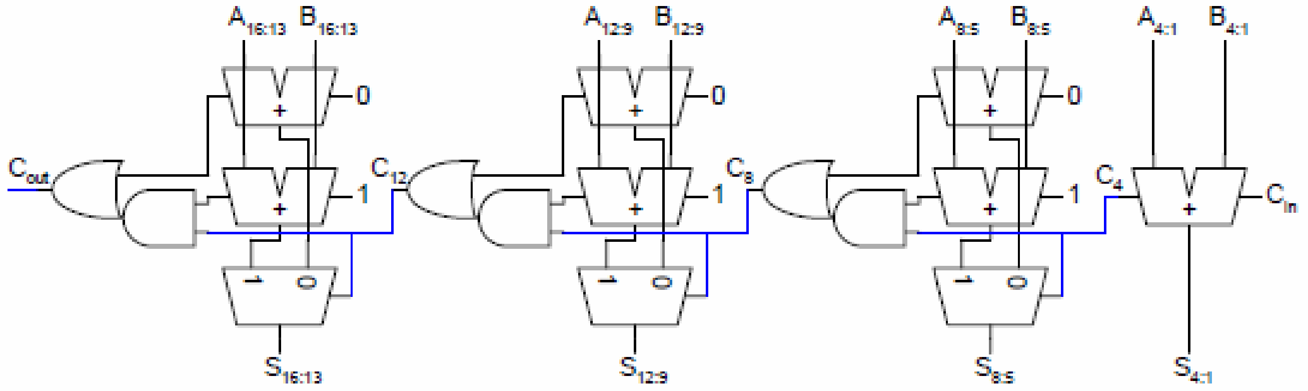


Carry Look Ahead Adder PG diagram



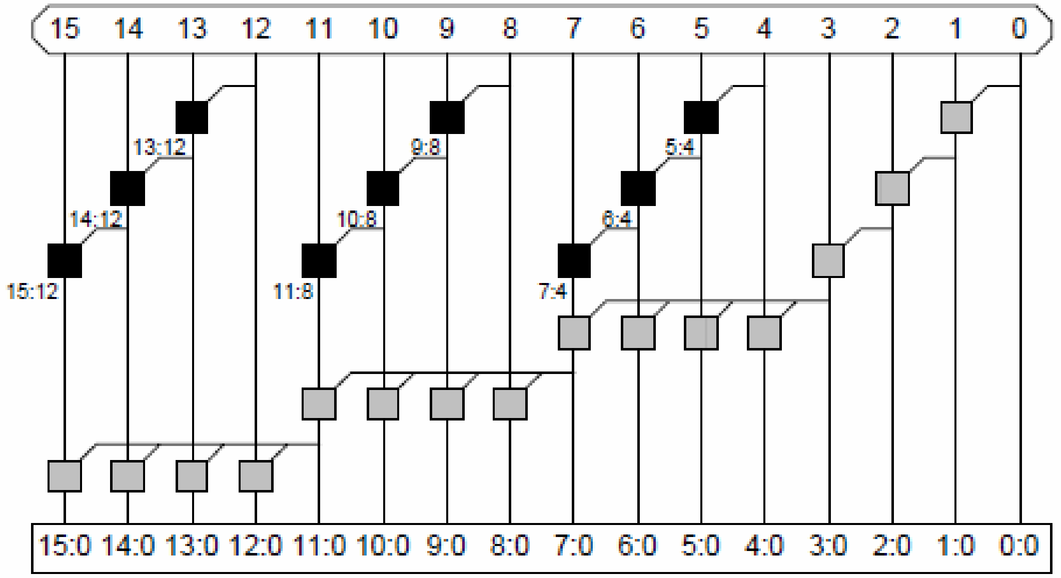
Carry Select Adder

- Trick for critical paths dependent on late input X
  - Precompute two possible outputs for X = 0, 1
  - Select proper output when X arrives
- Carry-select adder precomputes n-bit sums
  - For both possible carries into n-bit group



Carry Increment Adder

- Factor initial PG and final XOR out of carry-select

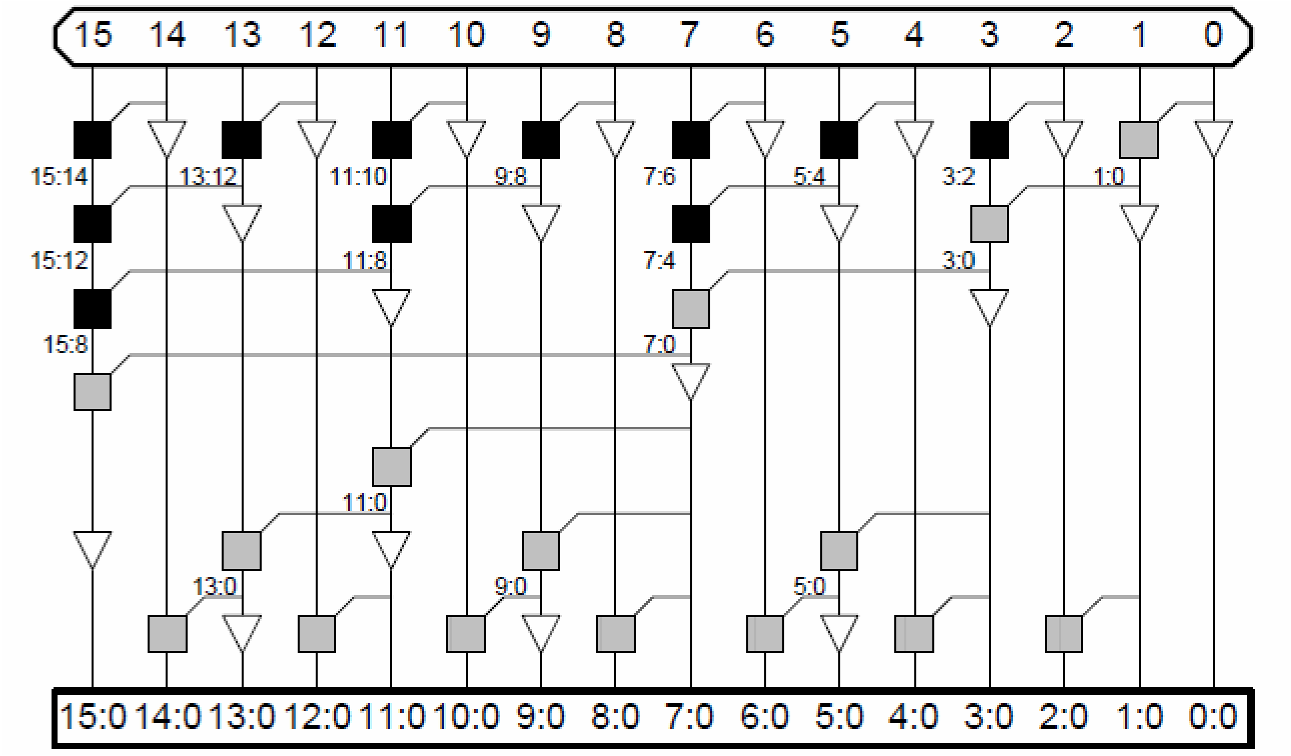


$$t_{\text{increment}} = t_{pg} + \left[(n-1) + (k-1) \right] t_{AO} + t_{\text{xor}}$$

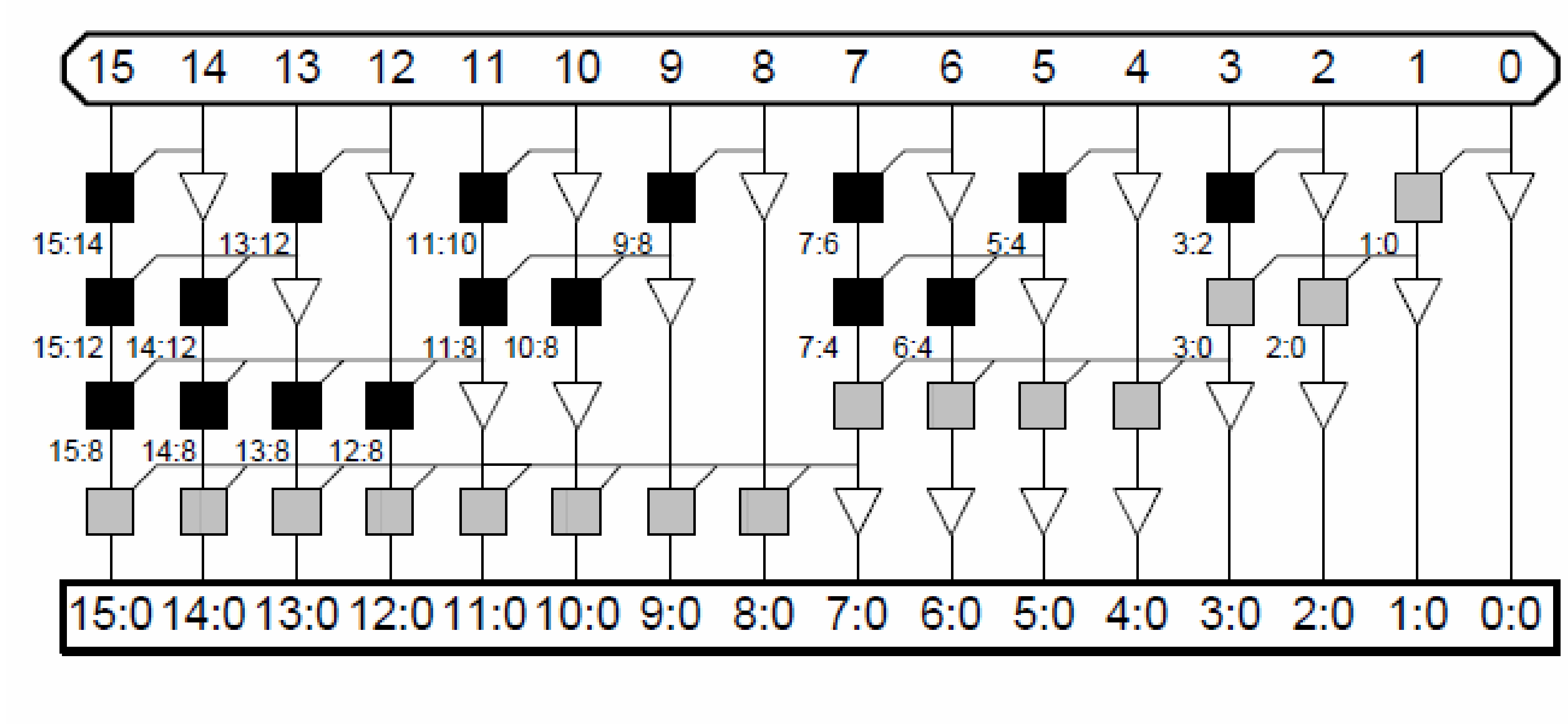
Tree Adders

- If lookahead is good, lookahead across lookahead!
  - Recursive lookahead gives O(log N) delay
- Many variations on tree adders

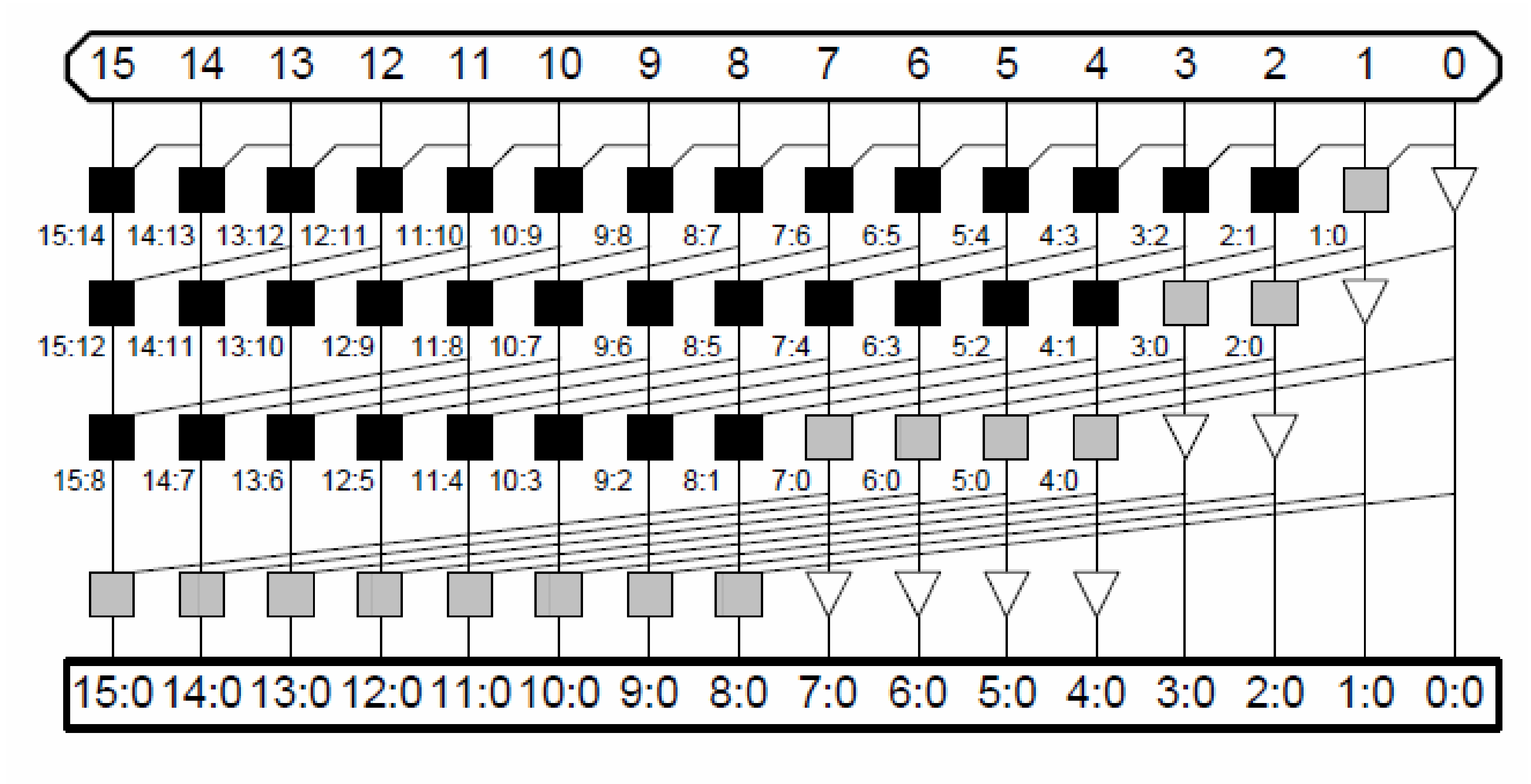
Brent Kung Adder



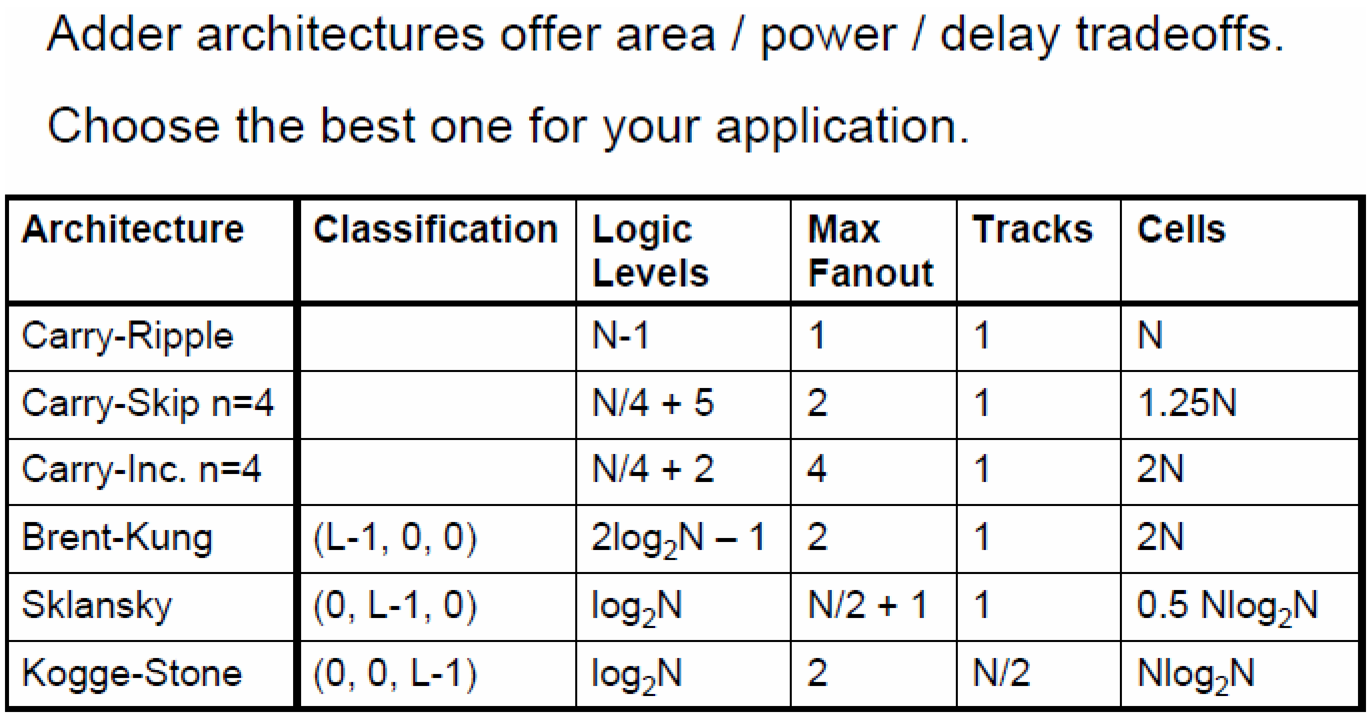
Sklansky Adder



Kogge Stone Adder



Comparison of the adders



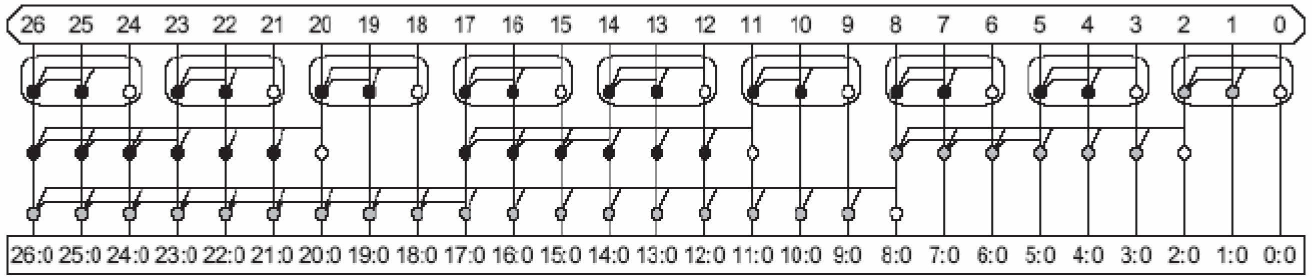
Adder architectures offer area / power / delay tradeoffs.

Choose the best one for your application.

Architecture	Classification	Logic Levels	Max Fanout	Tracks	Cells
Carry-Ripple		N-1	1	1	N
Carry-Skip n=4		N/4 + 5	2	1	1.25N
Carry-Inc. n=4		N/4 + 2	4	1	2N
Brent-Kung	(L-1, 0, 0)	$2\log_2 N - 1$	2	1	2N
Sklansky	(0, L-1, 0)	$\log_2 N$	N/2 + 1	1	$0.5\ N\log_2 N$
Kogge-Stone	(0, 0, L-1)	$\log_2 N$	2	N/2	$N\log_2 N$

Higher Valency Trees

- ❑ Combine 3 or 4 groups at each level
- ❑ High fan-in gates better suited to domino circuits



Thank You