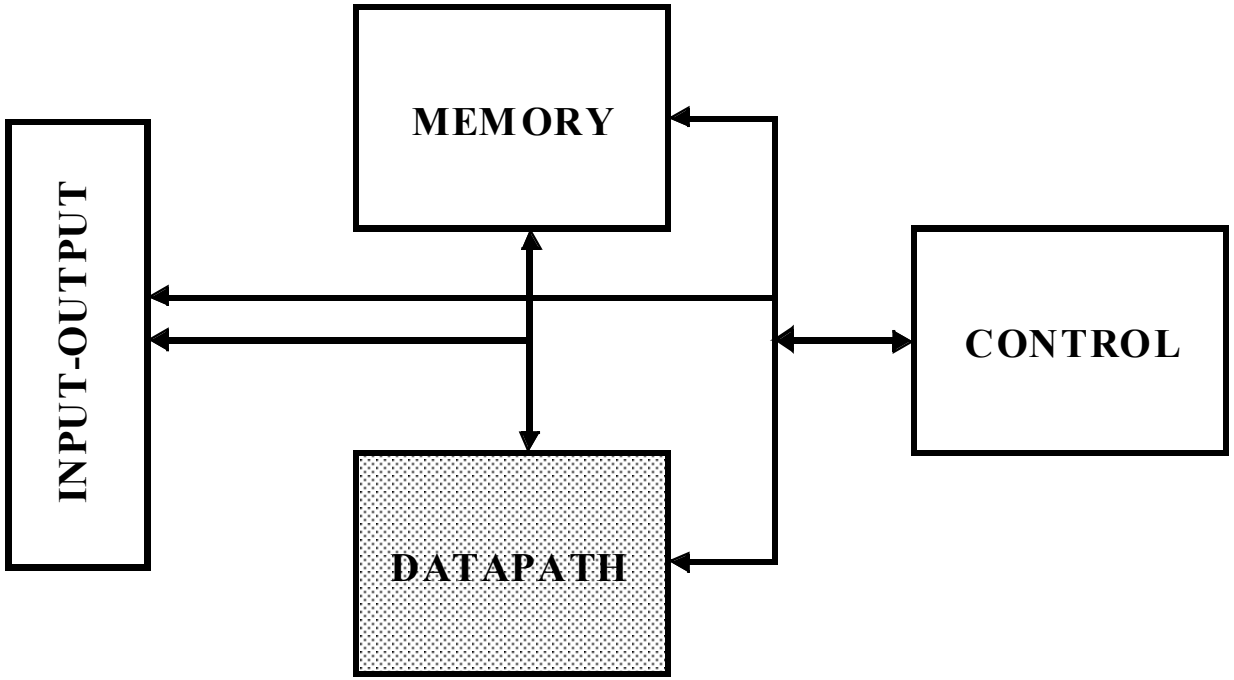
Digital Integrated Circuits
A Design Perspective

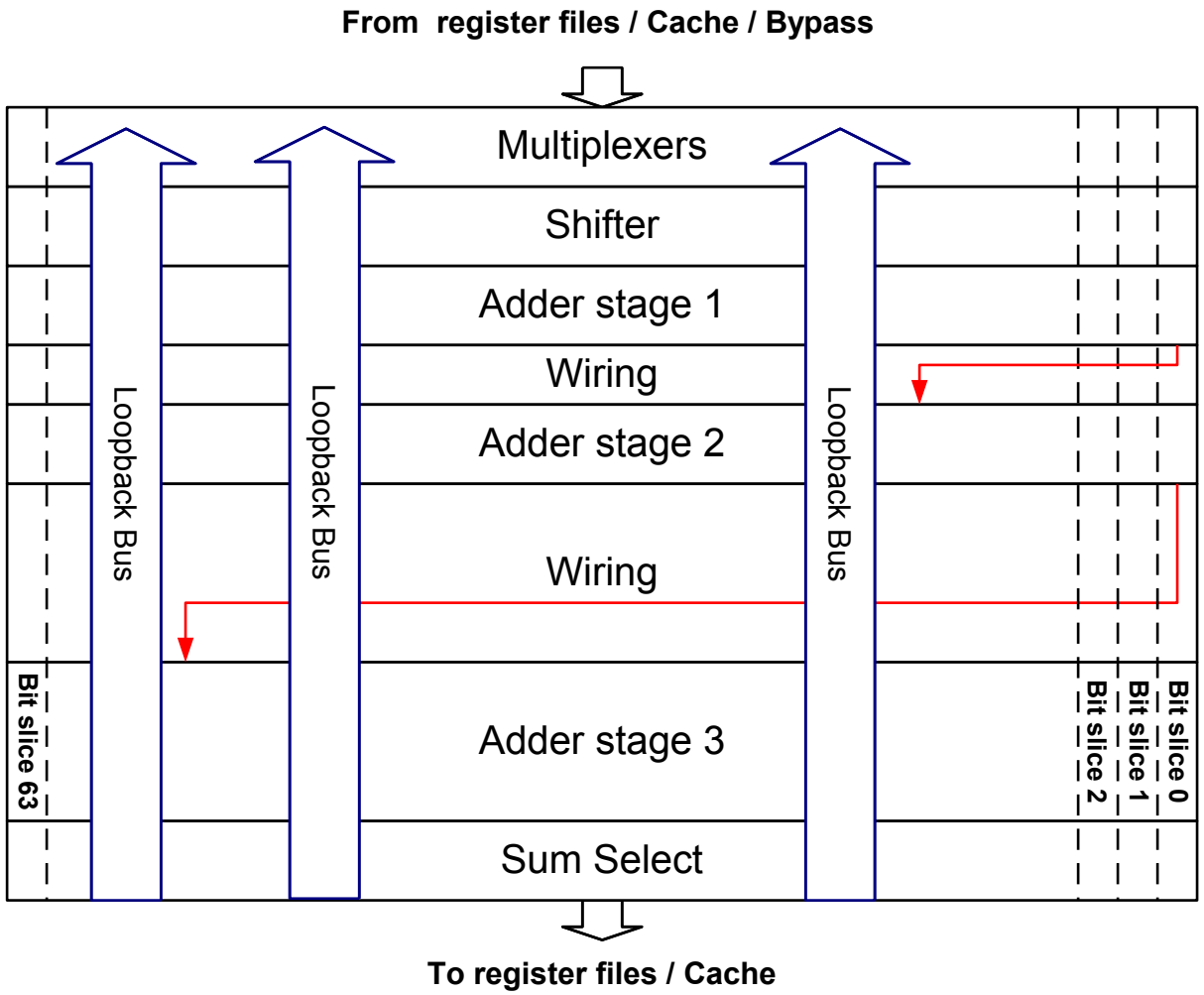
Chapter 5

Arithmetic Circuits

A Generic Digital Processor



Bit-Sliced Datapath



outline

- Adder
- Datapath functional unit
  - Comparators
  - Shifters
  - Multi-input Adders
- Multipliers

Adders

Multitudes of contrivances were designed,and almost endless drawings made, for the purpose of economizing the time and simplifying the mechanism of carriage

__charles babbage, on difference engine No.1,1864

Outline

- Single-bit Addition
- Carry-Ripple Adder
- Carry-Skip Adder
- Carry-Lookahead Adder
- Carry-Select Adder
- Carry-Increment Adder
- Tree Adder

PGK

- 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 =

PGK

- For a full adder, define what happens to carries
  - Generate: C_{out} = 1 independent of C
    - G = A • B
  - Propagate: C_{out} = C
    - P = A ⊕ B
  - Kill: C_{out} = 0 independent of C
    - K = ~A • ~B
- Note that we will be sometimes using an alternate definition for (only for carry)

$$C_o(G,P) = G + PC_i$$

$$S(G,P) = P \oplus C_i$$

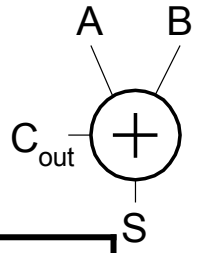
Propagate (P) = A+ B

Single-Bit Addition

Half Adder

$$S = A \oplus B$$

$$C_{out} = AgB$$



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

$$S = A\overline{B}\overline{C} + \overline{A}B\overline{C} + \overline{A}\overline{B}C + ABC$$

$$= A \oplus B \oplus C = P \oplus C$$

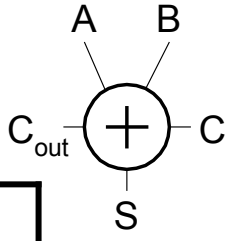
$$C_{out} = AB + (A + B)C$$

$$= \overline{\overline{A}\overline{B} + (\overline{A} + \overline{B})\overline{C}} = MAJ(A, B, C)$$

Full Adder

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

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



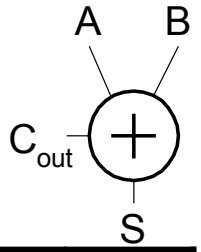
A	B	C	C_{out}	S
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Single-Bit Addition

Half Adder

$$S = A \oplus B$$

$$C_{out} = AgB$$



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

$$S = A\overline{B}\overline{C} + \overline{A}B\overline{C} + \overline{A}\overline{B}C + ABC$$

$$= A \oplus B \oplus C = P \oplus C$$

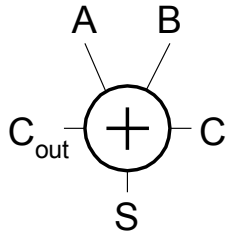
$$C_{out} = AB + (A + B)C$$

$$= \overline{\overline{A}\overline{B} + (\overline{A} + \overline{B})\overline{C}} = MAJ(A, B, C)$$

Full Adder

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

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



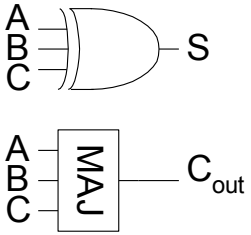
A	B	C	C_{out}	S	
0	0	0	0	0	delete
0	0	1	0	1	
0	1	0	0	1	propagate
0	1	1	1	0	
1	0	0	0	1	
1	0	1	1	0	
1	1	0	1	0	generate
1	1	1	1	1	

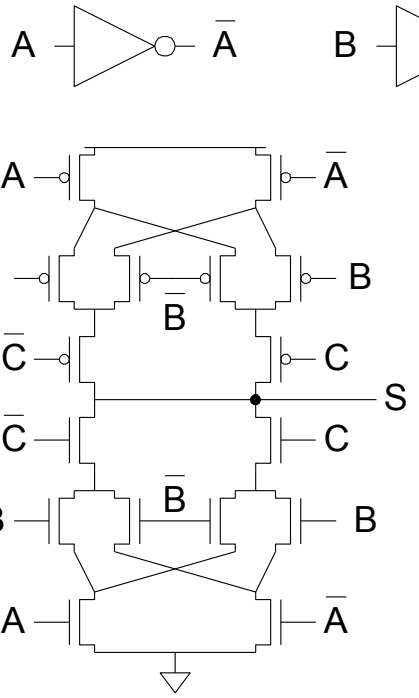
Full Adder Design I

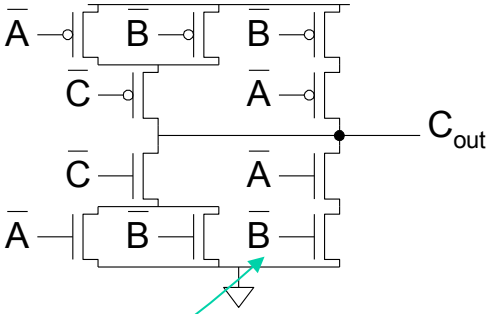
- implementation from eqns

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

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







$$S = A\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}\bar{B}C + ABC$$

$$= A \oplus B \oplus C = P \oplus C$$

$$C_{out} = AB + (A+B)C$$

$$= \overline{\bar{A}\bar{B} + (\bar{A}+\bar{B})\bar{C}} = MAJ(A,B,C)$$

Fewer than mentioned above because of sharing some transistors for XOR gate

Full Adder Design II

- Factor S in terms of C_{out}

  S = ABC + (A + B + C)(~C_{out})

  Cout=MAJ(A,B,C)

- Critical path is usually C to C_{out} in ripple adder

$$C_o = AB + BC_i + AC_i$$

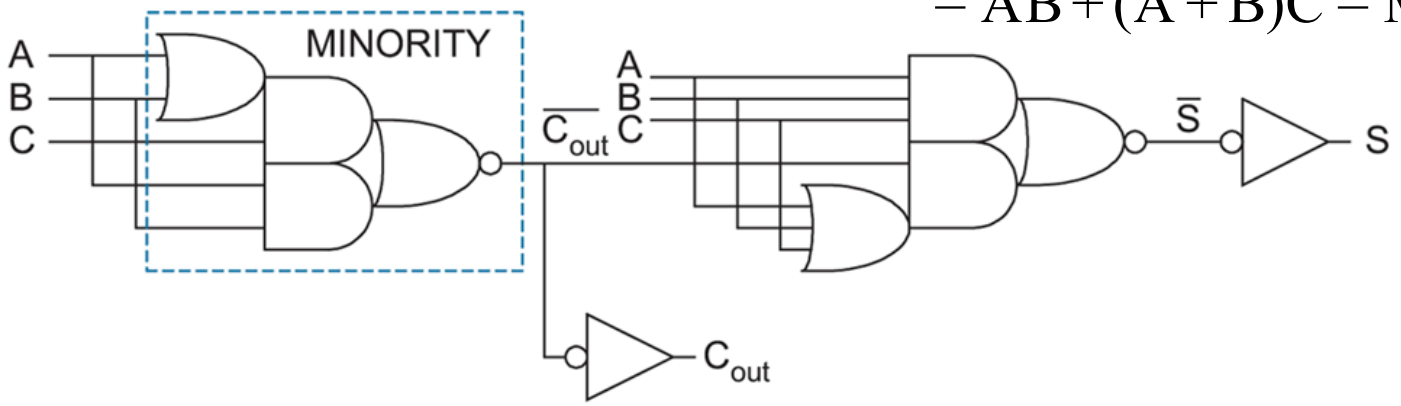
$$S = ABC_i + \overline{C_o}(A + B + C_i)$$

$$S = A\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}\bar{B}C + ABC$$

$$= A \oplus B \oplus C = P \oplus C$$

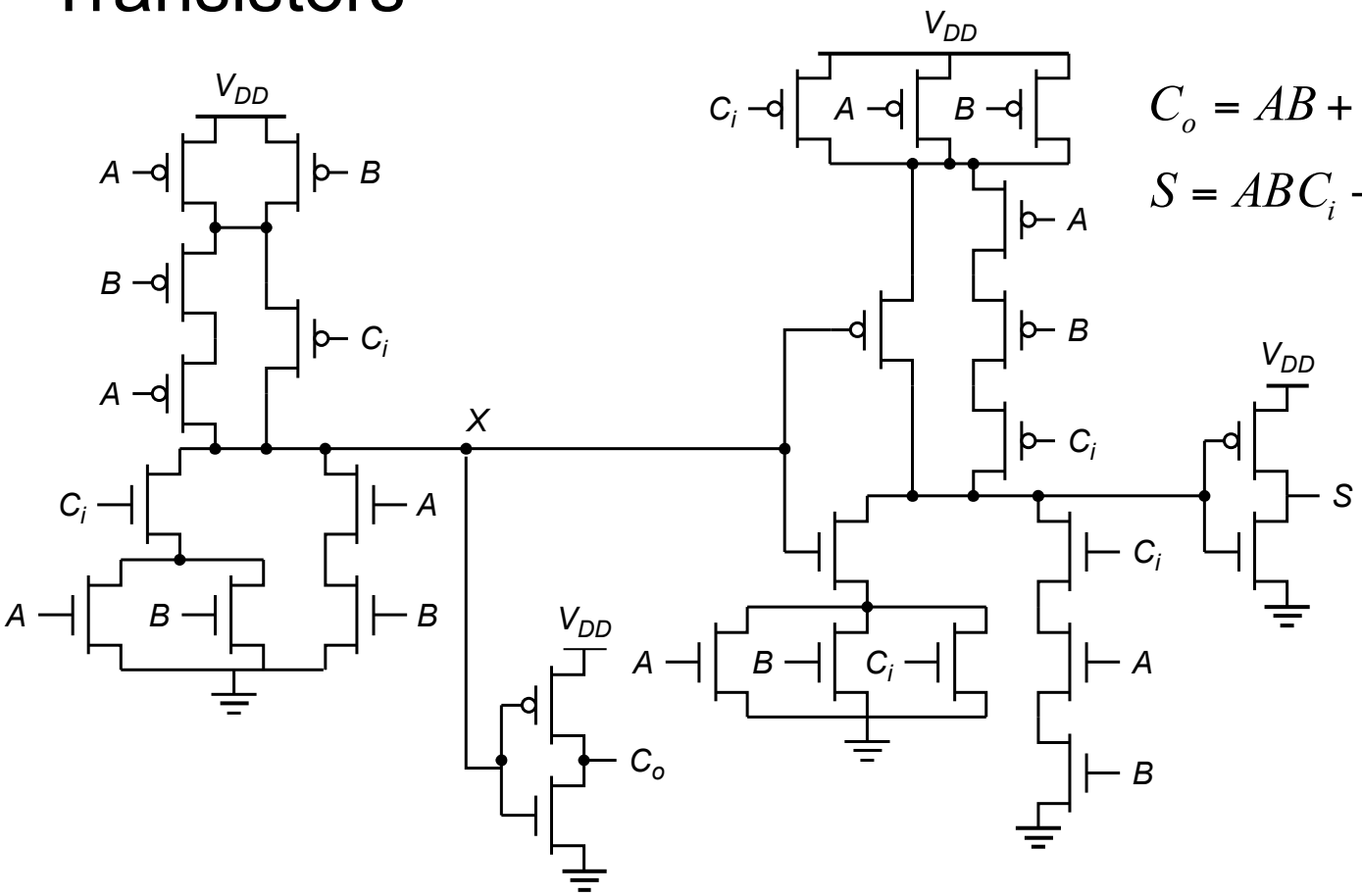
$$C_{out} = AB + (A + B)C$$

$$= \overline{\bar{A}\bar{B} + (\bar{A} + \bar{B})\bar{C}} = MAJ(A, B, C)$$



Full Adder Design II

Complimentary Static CMOS Full Adder__28 Transistors



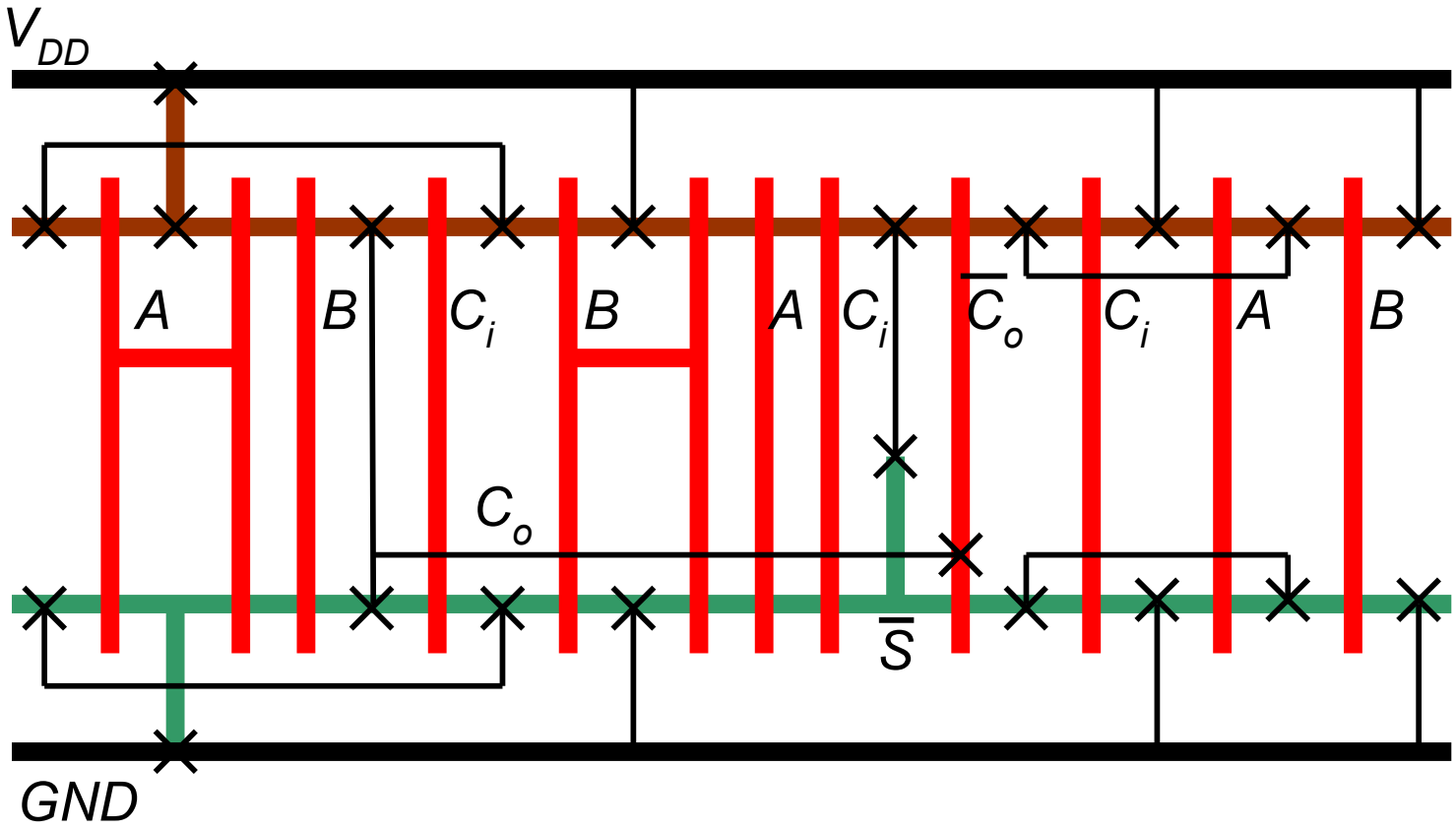
$$C_o = AB + BC_i + AC_i$$

$$S = ABC_i + \overline{C_o}(A + B + C_i)$$

Mirror Adder-Stick Diagram

$$C_o = AB + BC_i + AC_i$$

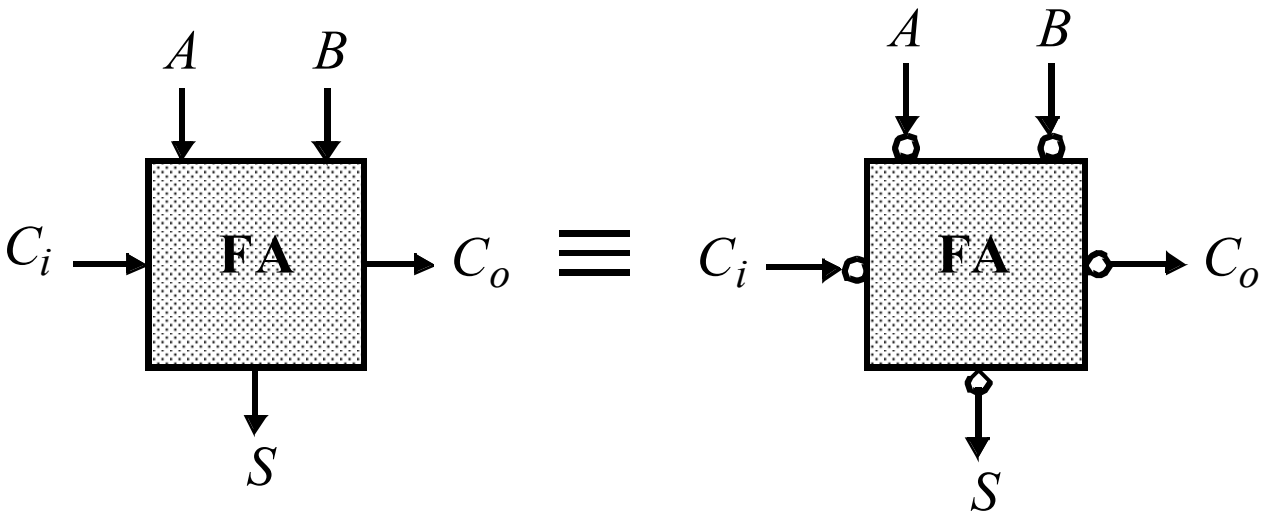
$$S = ABC_i + \overline{C_o}(A + B + C_i)$$



The Mirror Adder

- The NMOS and PMOS chains symmetrical. A maximum of two series transistors arranged in the carry-generation
- the most critical issue is the minimization of the capacitance at node Co. The reduction of the diffusion capacitances is particularly important.
- The capacitance at node Co is composed of 4 diffusion capacitances, 2 internal gate capacitances, and 6 gate capacitances in the connecting adder cell .
- The transistors connected to Ci are closest to the output.
- Only the transistors in the carry stage have to be optimized for optimal speed. All transistors in the sum stage can be minimal size.

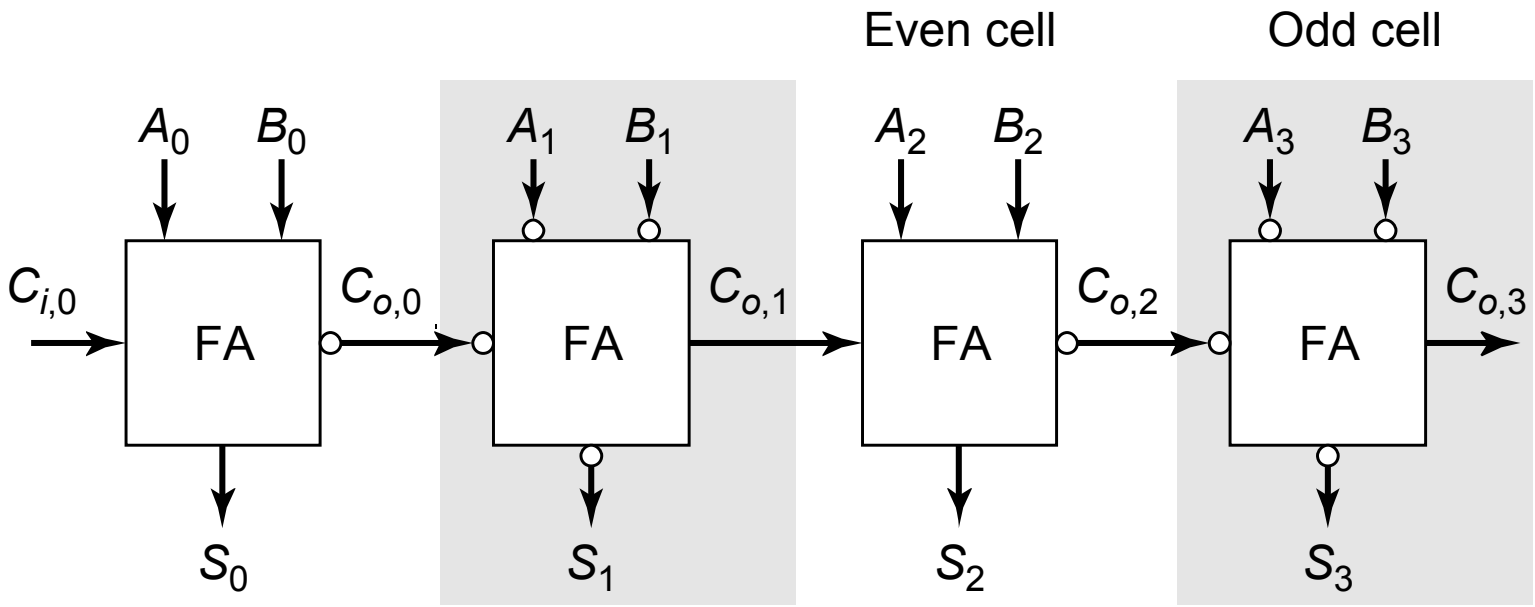
Inversion Property



$$\bar{S}(A,B,C_i) = S(\bar{A},\bar{B},\bar{C}_i)$$

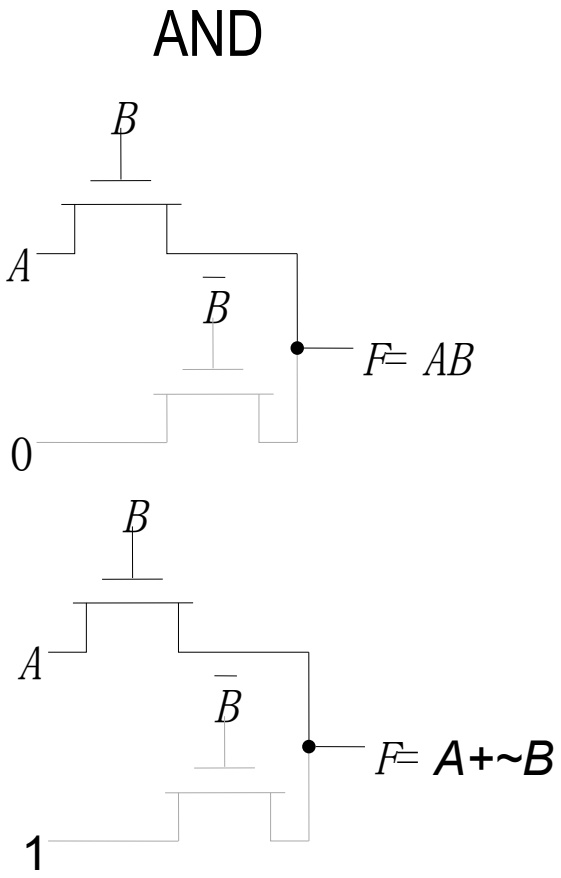
$$\overline{C_o}(A,B,C_i) = C_o(\bar{A},\bar{B},\bar{C}_i)$$

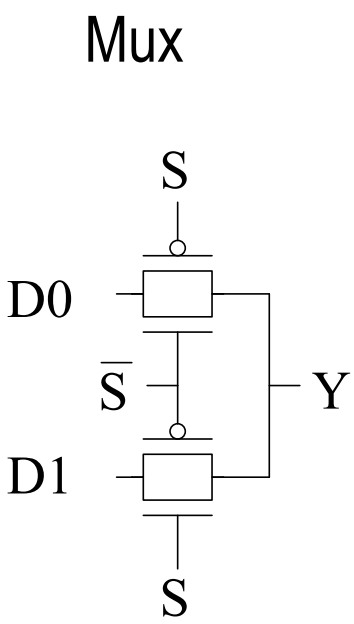
Minimize Critical Path by Reducing Inverting Stages



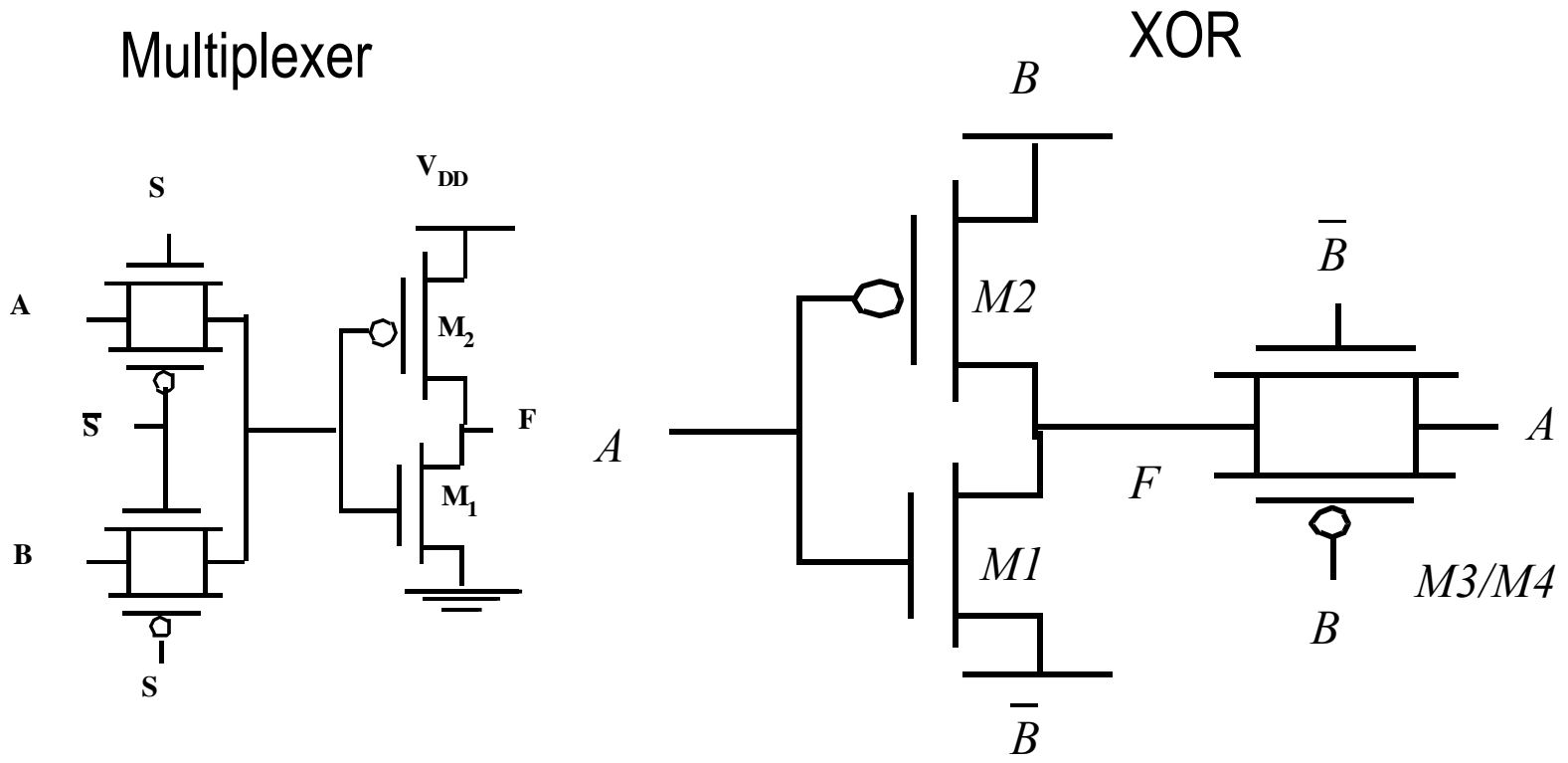
Exploit Inversion Property

Transmission Gate

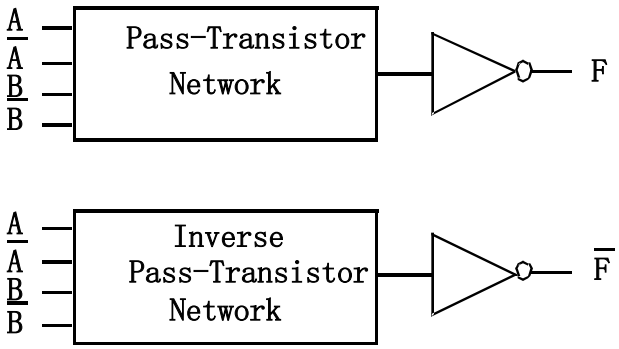




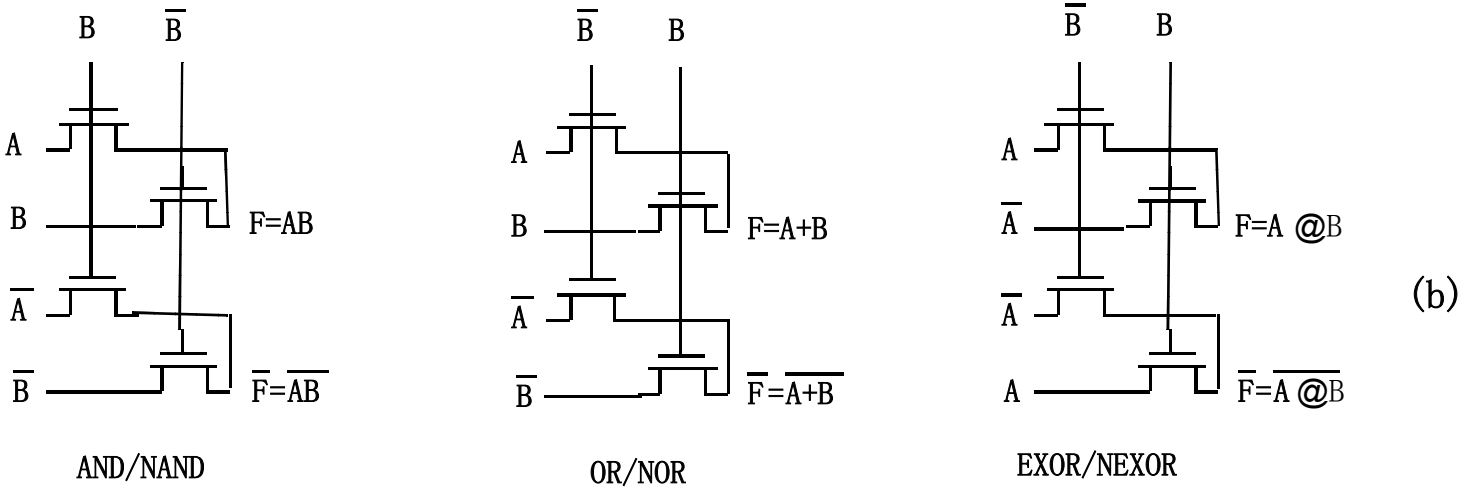
Pass-Transistor Based



Complementary Pass Transistor Logic



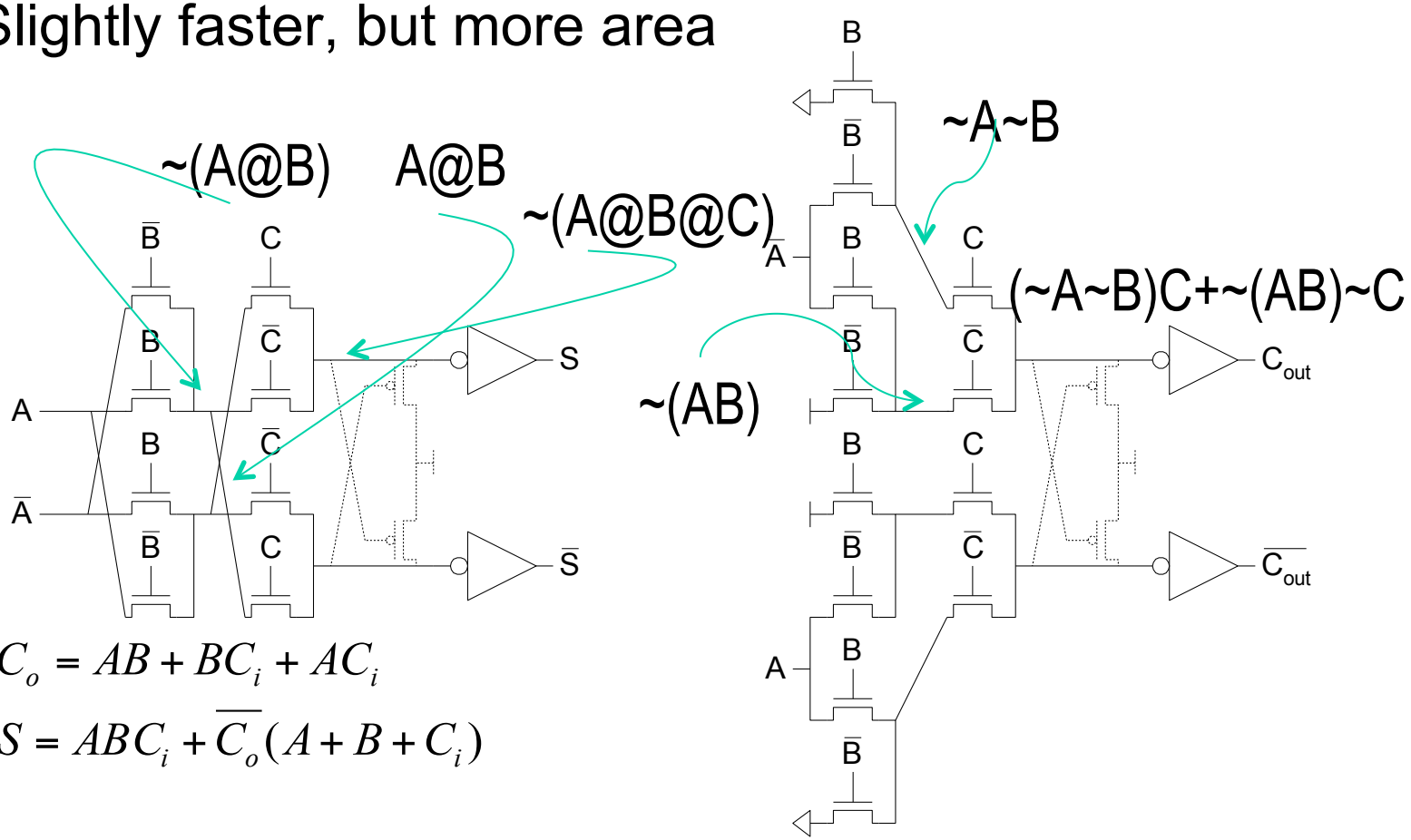
(a)



(b)

Full Adder Design III

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



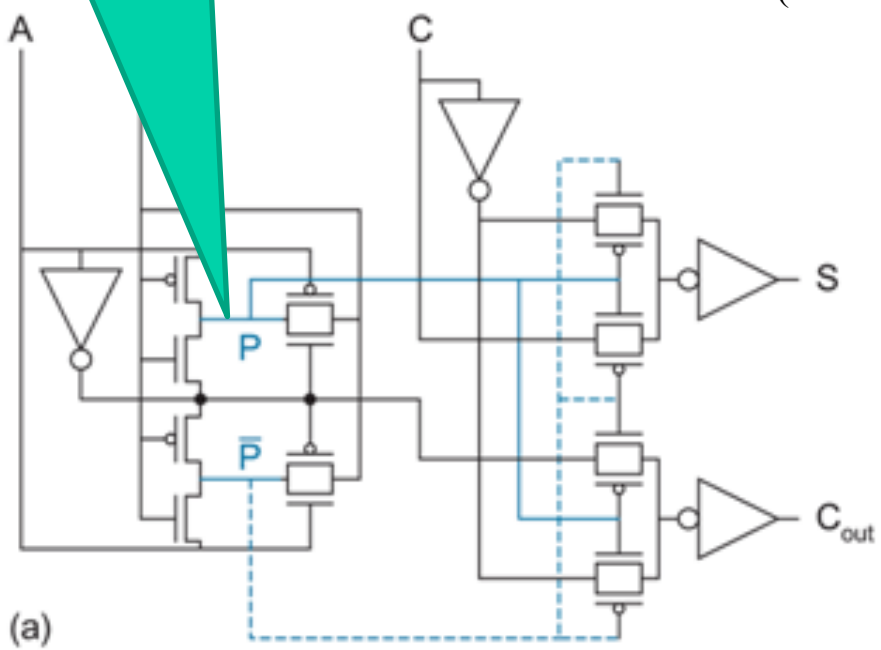
$$C_o = AB + BC_i + AC_i$$

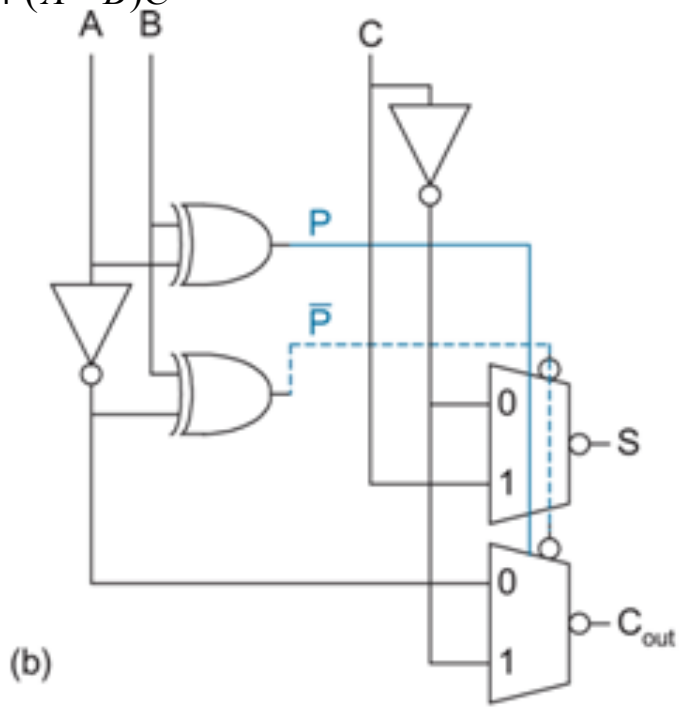
$$S = ABC_i + \overline{C_o}(A + B + C_i)$$

Full Adder Design IV

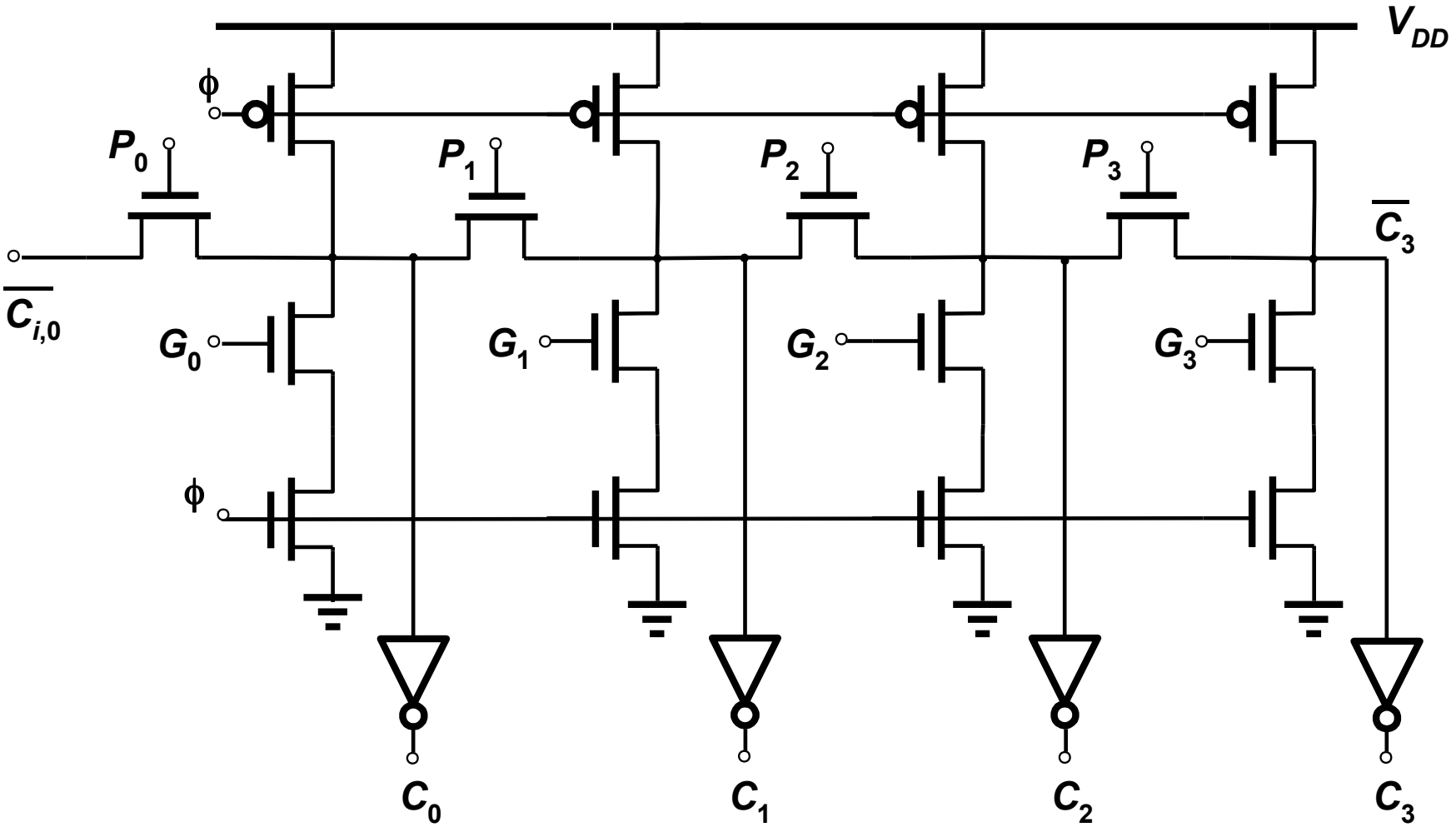
A@B

$$S = A\overline{B}\overline{C} + \overline{A}B\overline{C} + \overline{A}\overline{B}C + ABC$$
$$= A \oplus B \oplus C = P \oplus C$$
$$C_{out} = AB + (A \oplus B)C = AAB + A\overline{A}\overline{B} + (A \oplus B)C$$
$$= A(\overline{A \oplus B}) + (A \oplus B)C$$





Manchester Carry Chain



Generate / Propagate

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

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

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

- Base case

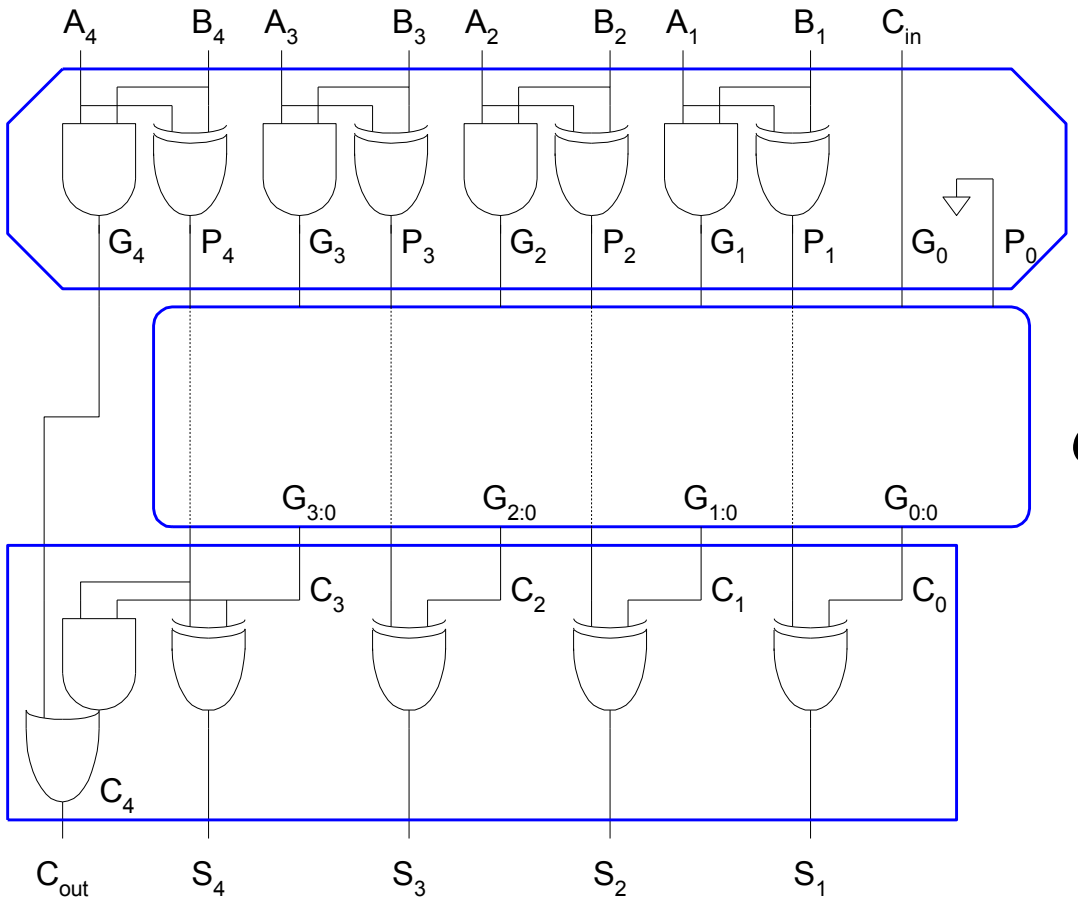
- Sum:

$$G_{i:i} \equiv G_i = A_i \cdot B_i \qquad G_{0:0} \equiv G_0 = C_{in}$$

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

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

PG Logic



$$G_{0:0} \equiv G_0 = C_{in}$$

$$P_{0:0} \equiv P_0 = 0$$

1: Bitwise PG logic

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

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

2: Group PG logic

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

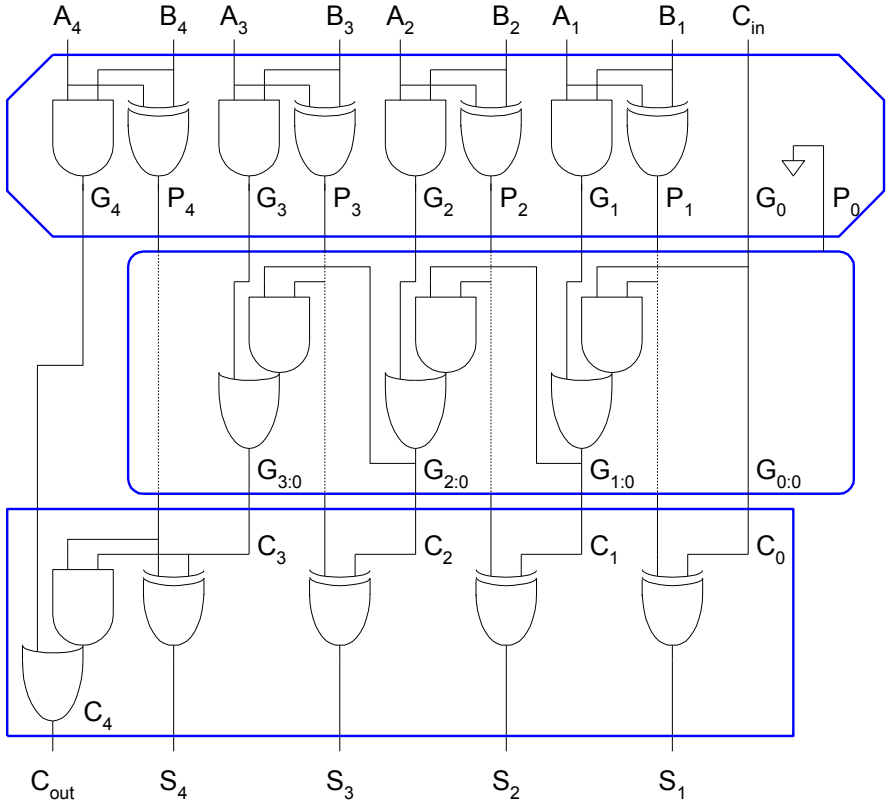
$$P_{i:j} = P_{i:k} \cdot P_{k-1:j}$$

3: Sum logic

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

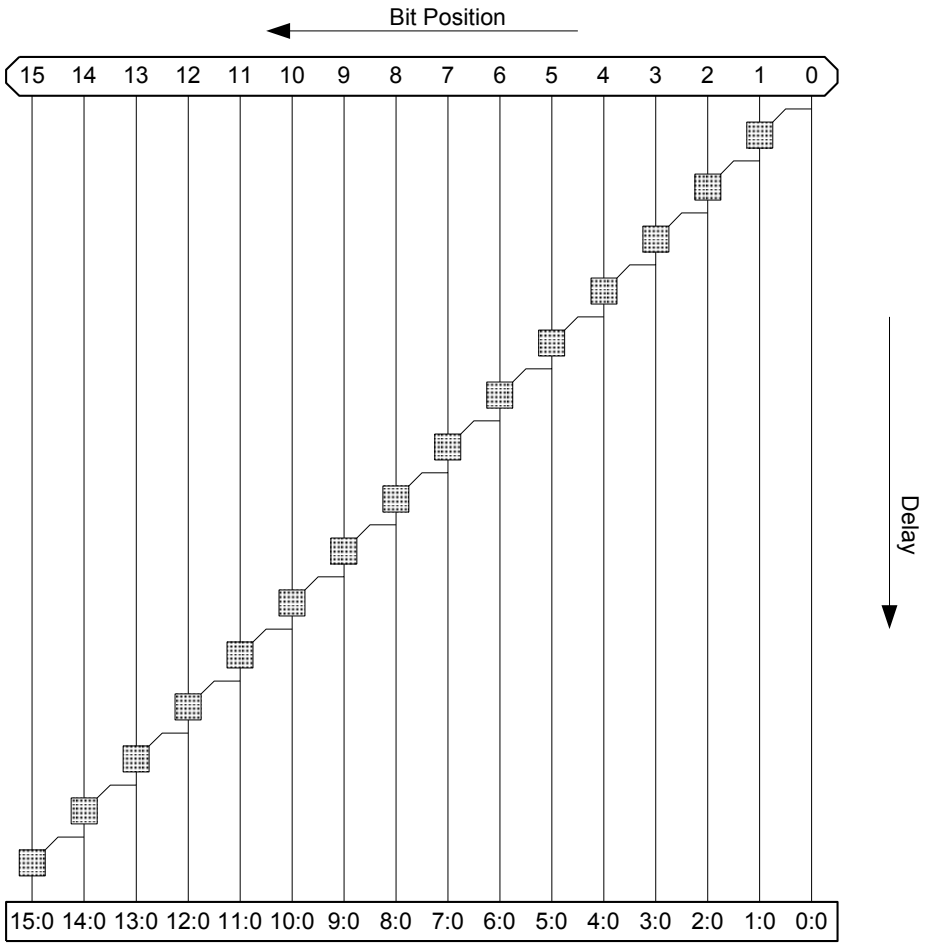
Carry-Ripple Revisited

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

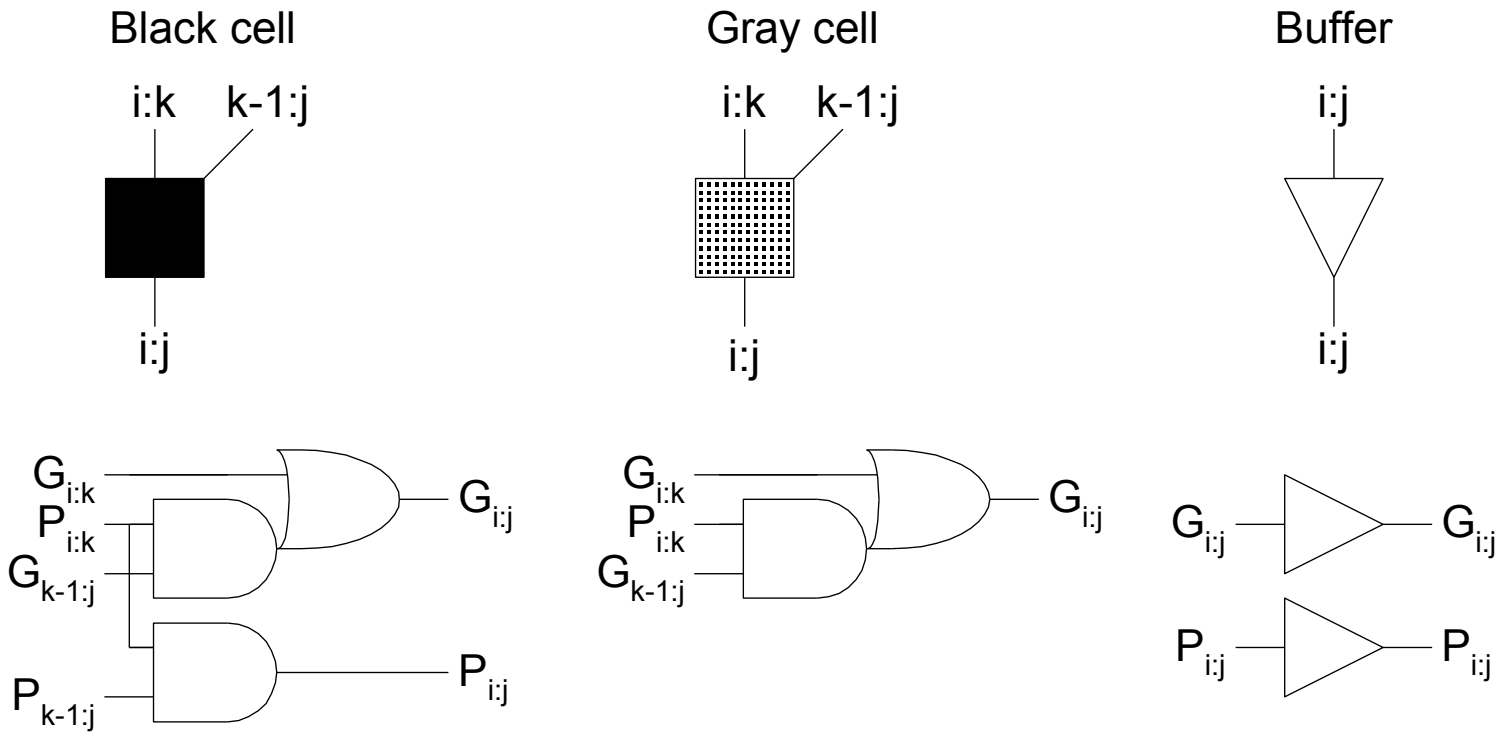


Carry-Ripple PG Diagram

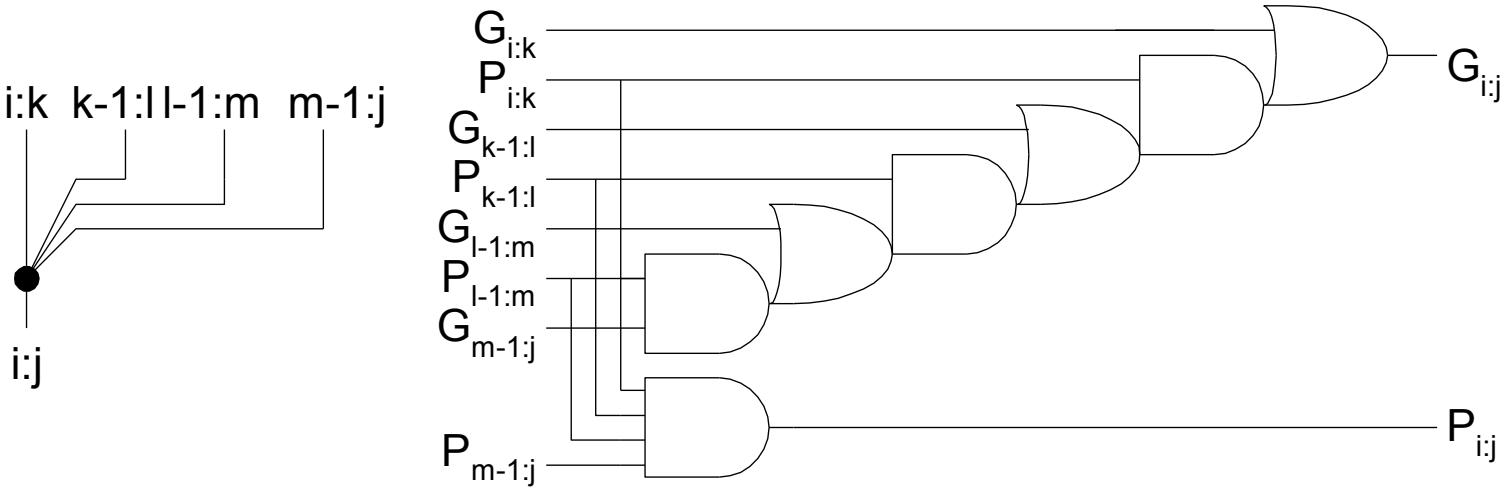
$$t_{\text{ripple}} = t_{pg} + (N-1)t_{AO} + t_{\text{xor}}$$



PG Diagram Notation

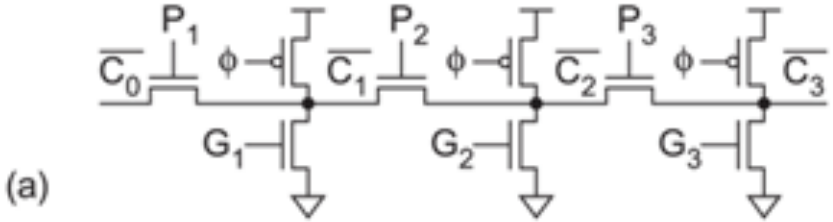


Higher-Valency Cells

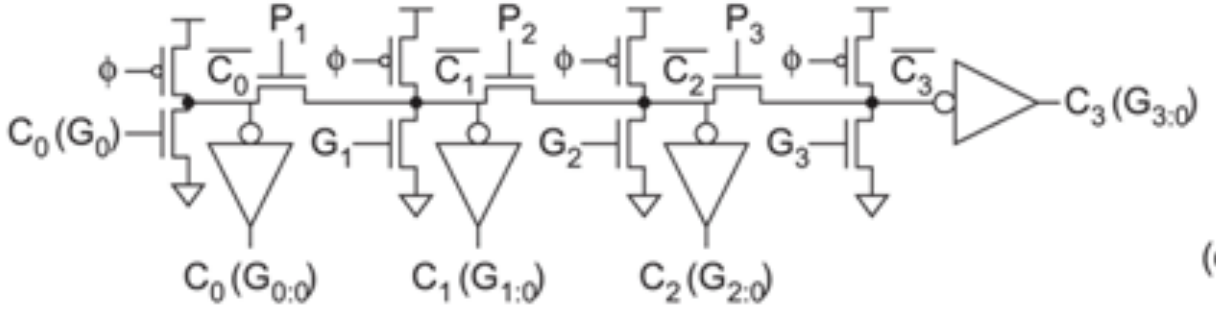


Manchester Carry Chain

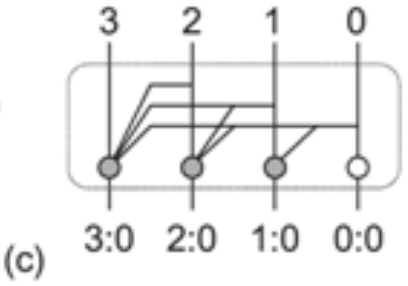
$$C_0 = G_{0:0} = C_0$$
$$C_1 = G_{1:0} = G_1 + P_1 C_0$$
$$C_2 = G_{2:0} = G_2 + P_2(G_1 + P_1 C_0)$$
$$C_3 = G_{3:0} = G_3 + P_3(G_2 + P_2(G_1 + P_1 C_0))$$



(a)



(b)



(c)

Manchester Carry Chain