DEPARTMENT OF COMPUTER SYSTEM ENGINEERING

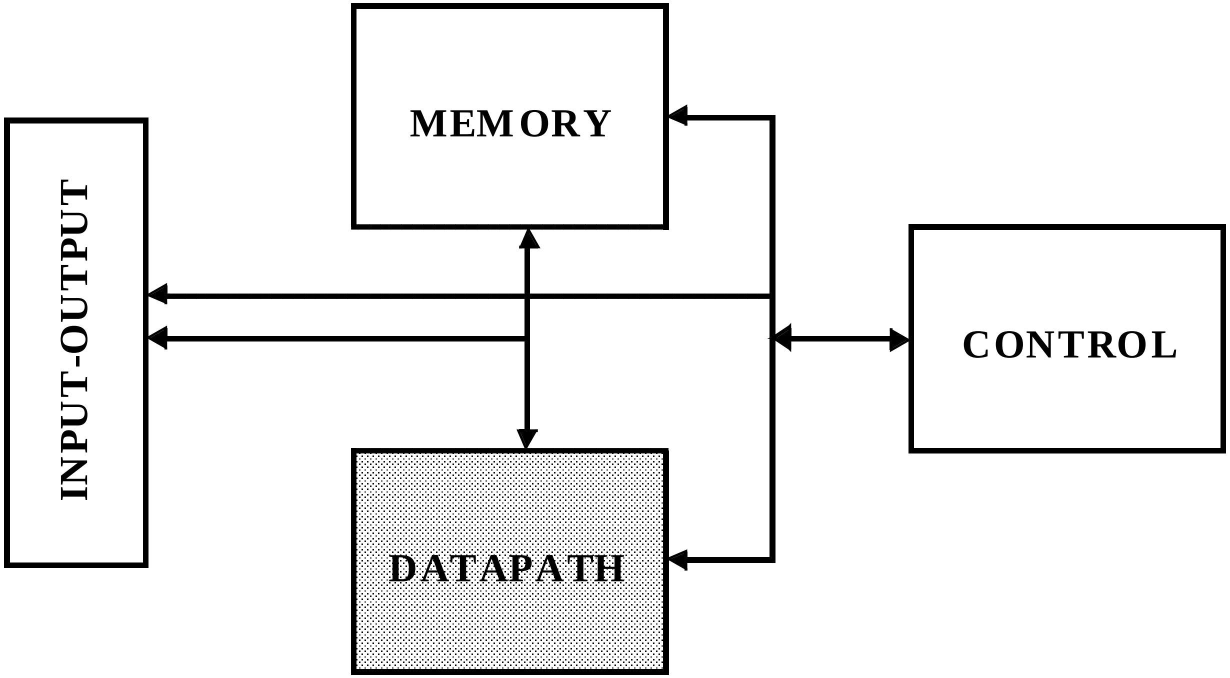
Digital Integrated Circuits - ENCS333

Dr. Khader Mohammad

Lecture #13 Adders

Integrated-Circuit Devices and Modeling

A Generic Digital Processor



Building Blocks for Digital Architectures

Arithmetic unit

- Bit-sliced datapath (adder, multiplier, shifter, comparator, etc.)

Memory

- RAM, ROM, Buffers, Shift registers

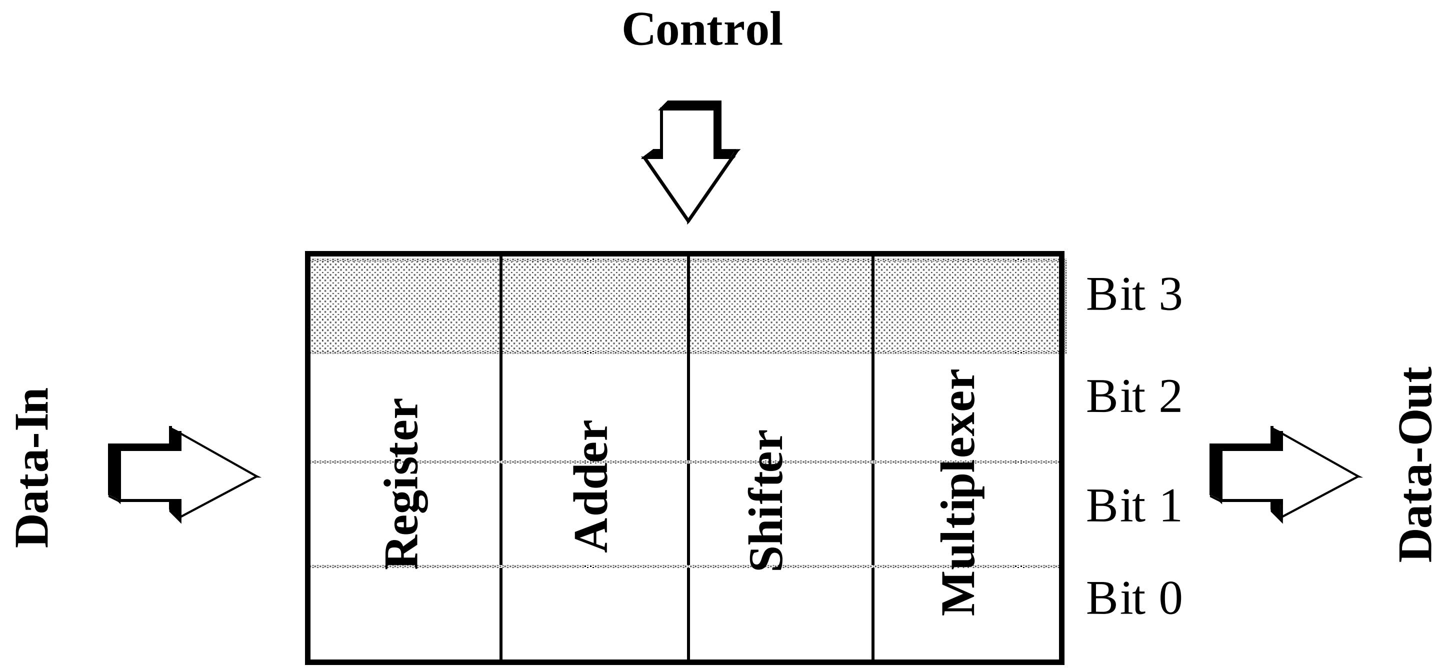
Control

- Finite state machine (PLA, random logic.)
- Counters

Interconnect

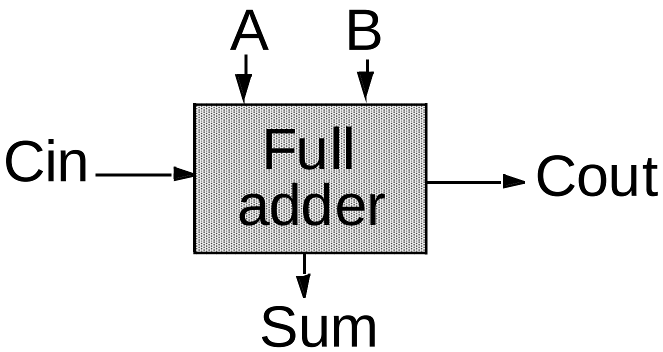
- Switches
- Arbiters
- Bus

Bit-Sliced Design



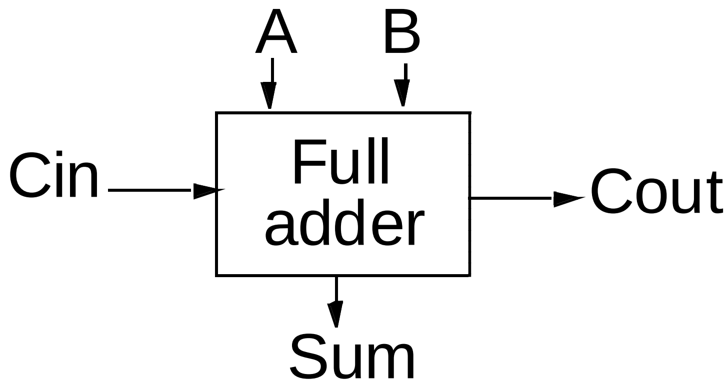
Tile identical processing elements

Full-Adder



A	*B*	C_i	*S*	C_o	*Carry status*
0	0	0	0	0	delete
0	0	1	1	0	delete
0	1	0	1	0	propagate
0	1	1	0	1	propagate
1	0	0	1	0	propagate
1	0	1	0	1	propagate
1	1	0	0	1	generate
1	1	1	1	1	generate

The Binary Adder



$$S = A \oplus B \oplus C_i$$

$$= A\bar{B}\bar{C}_i + \bar{A}B\bar{C}_i + \bar{A}\bar{B}C_i + ABC_i$$

$$C_o = AB + BC_i + AC_i$$

Express Sum and Carry as a function of P, G, D

Define 3 new variable which ONLY depend on A, B

Generate (G) = AB

Propagate (P) = $A \oplus B$

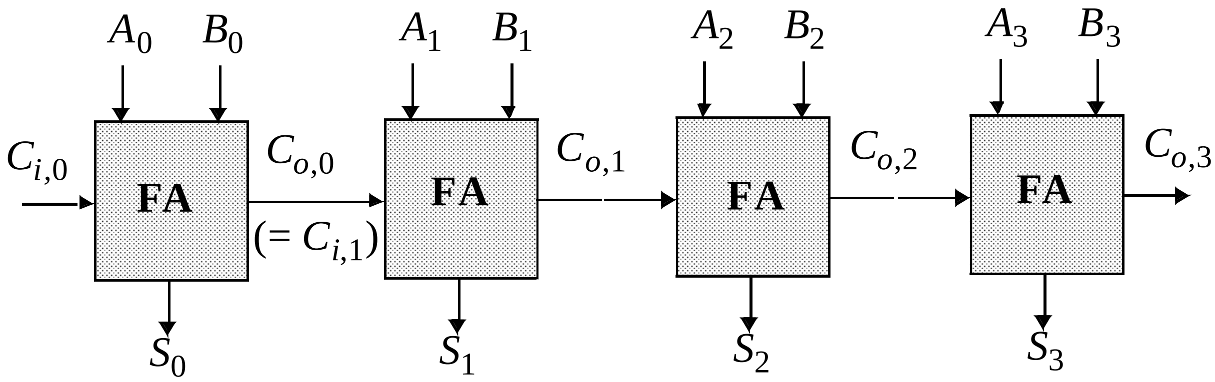
Delete = $\overline{A}\ \overline{B}$

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

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

Can also derive expressions for *S* and C_o based on *D* and *P*

The Ripple-Carry Adder



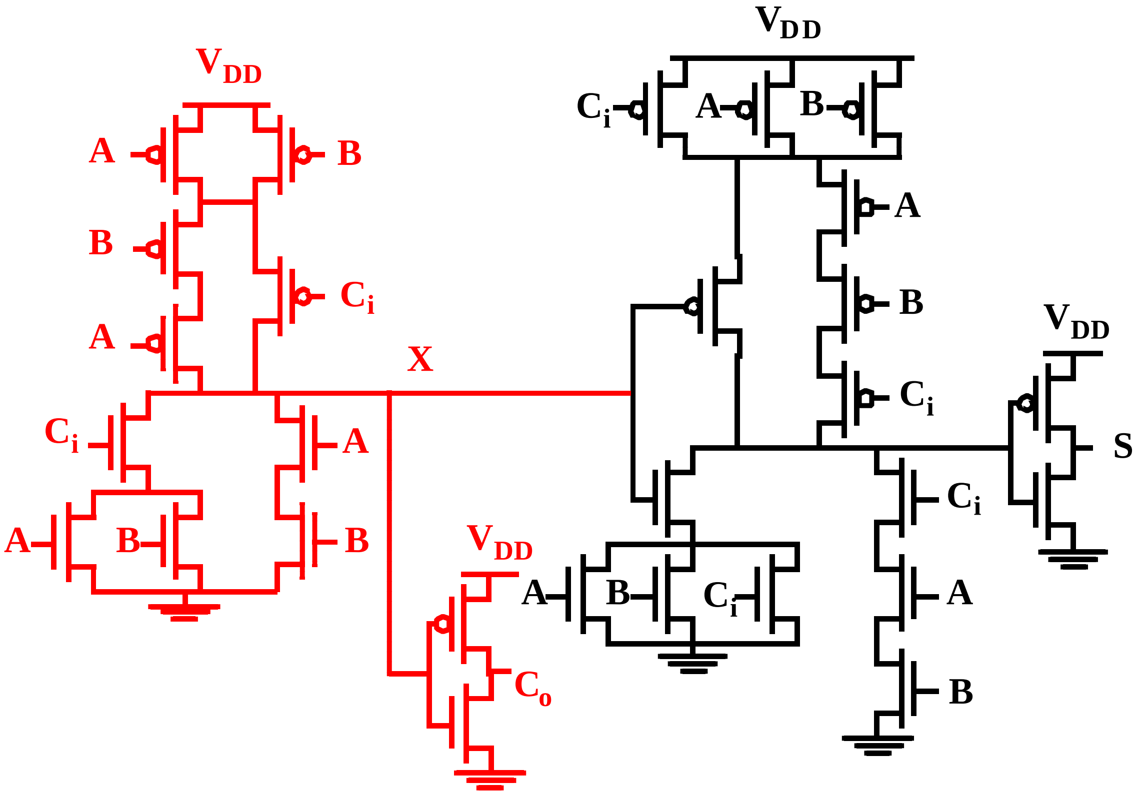
Worst case delay linear with the number of bits

$t_d = O(N)$

$$\mathbf{t_{adder} \approx (N-1)t_{carry} + t_{sum}}$$

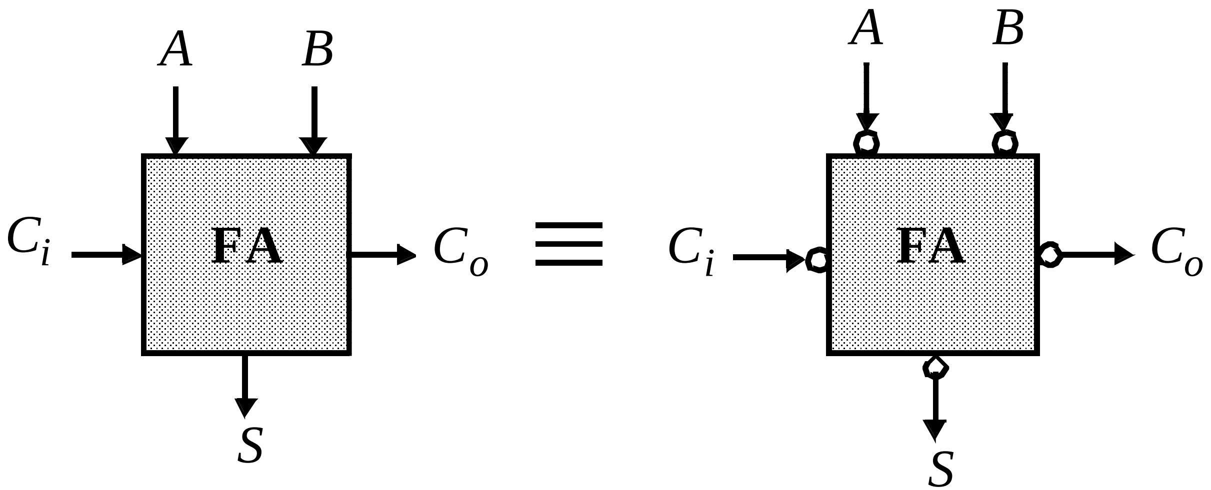
Goal: Make the fastest possible carry path circuit

Complimentary Static CMOS Full Adder



28 Transistors

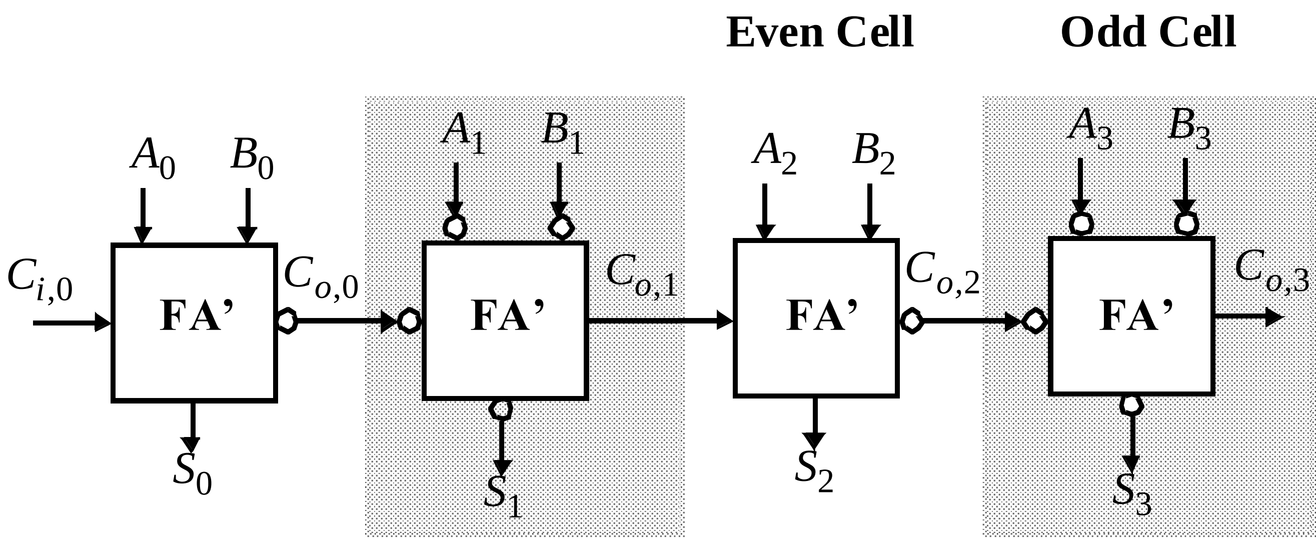
Inversion Property



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

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

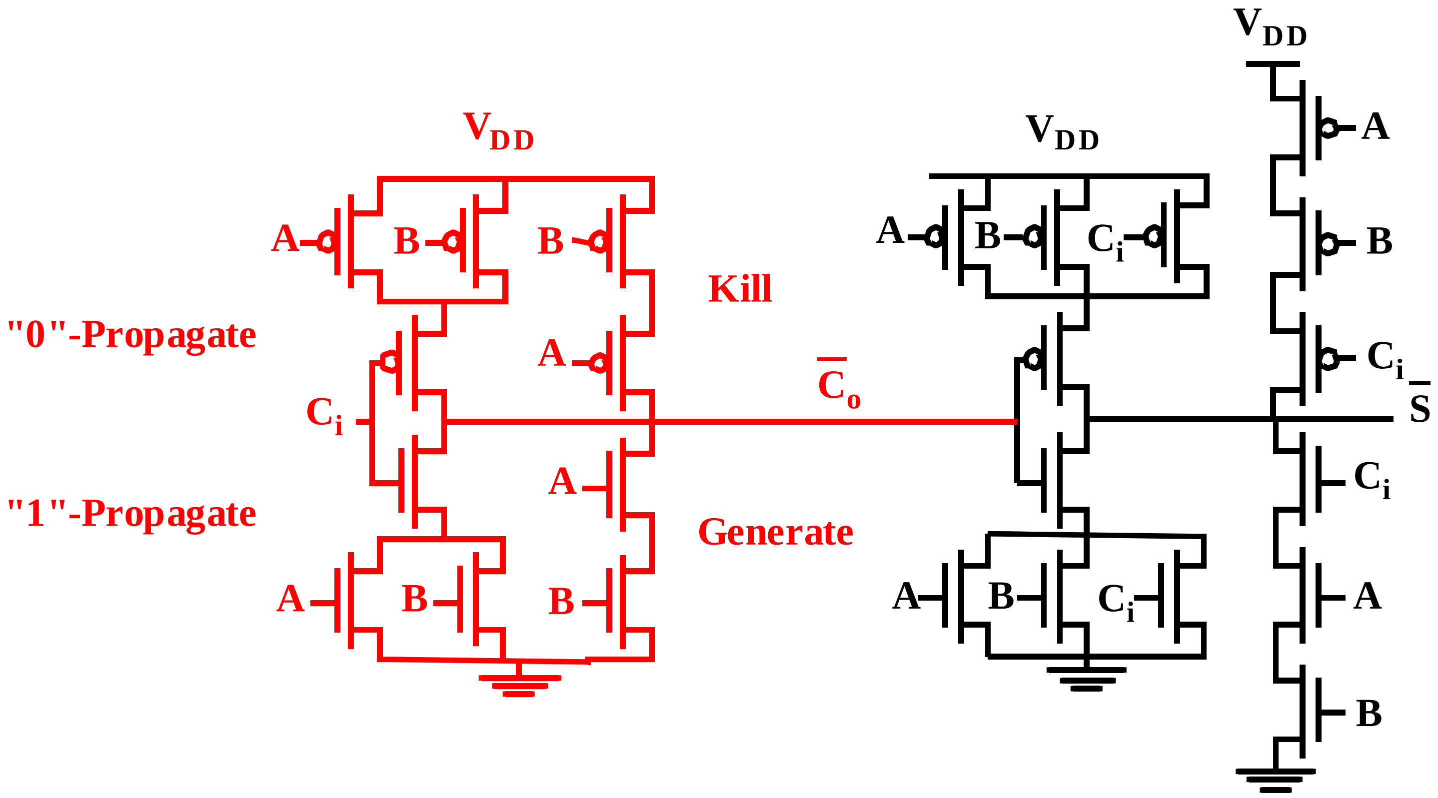
Minimize Critical Path by Reducing Inverting Stages



Exploit Inversion Property

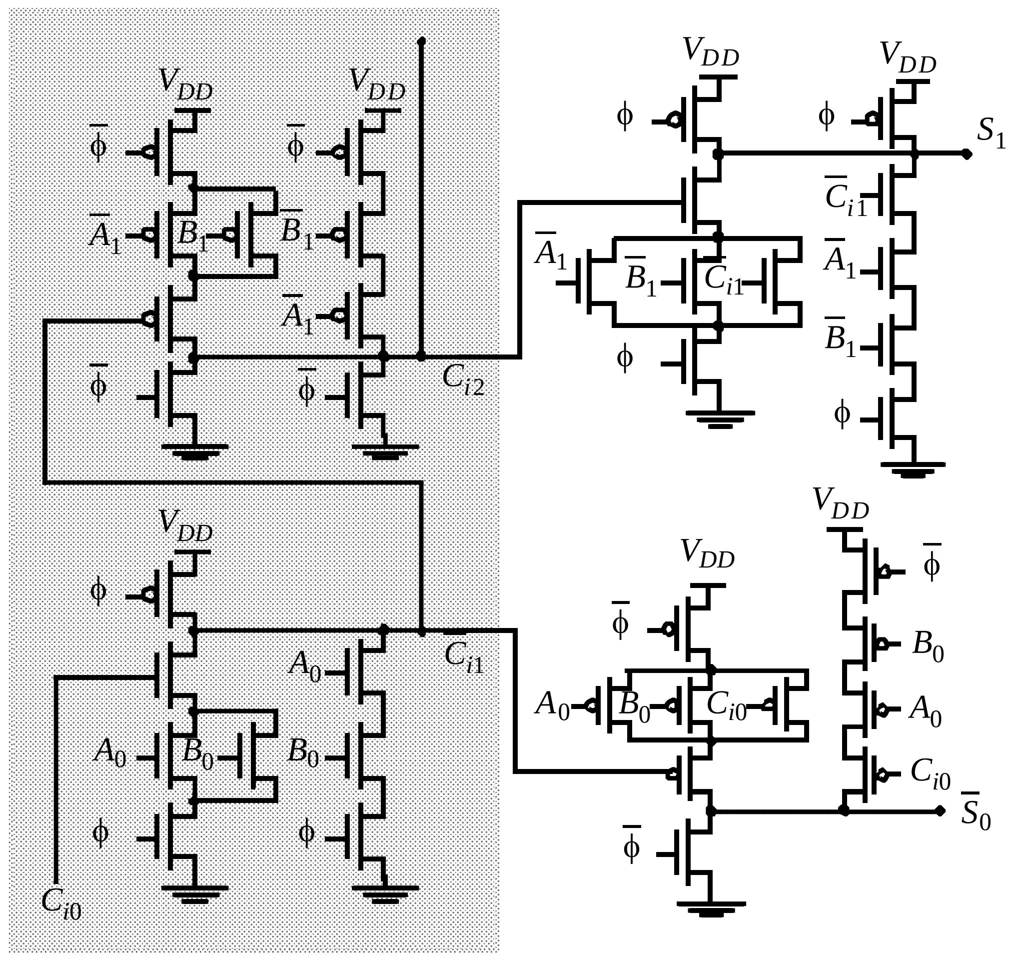
Note: need 2 different types of cells

The better structure: the Mirror Adder



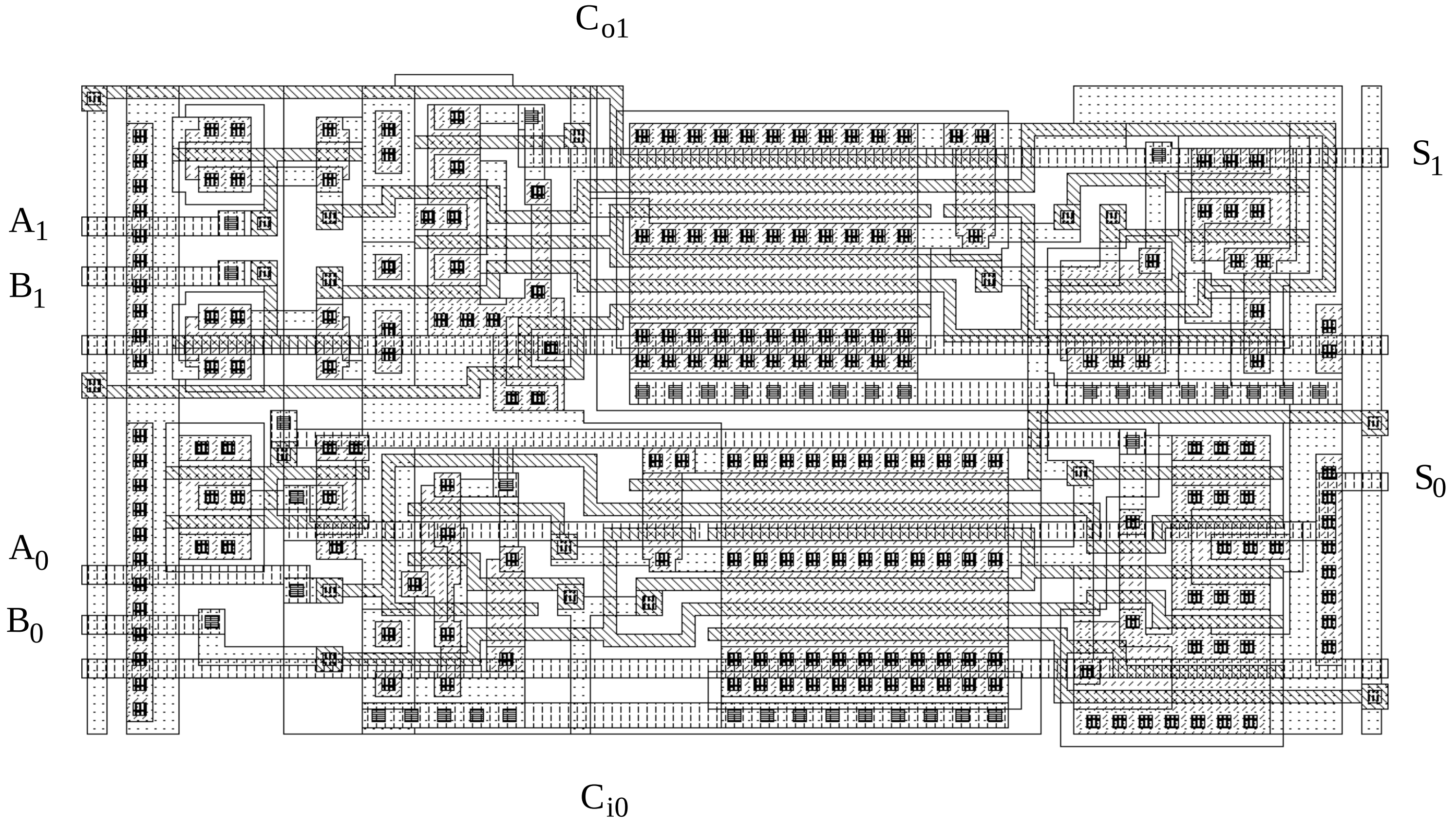
24 transistors

NP-CMOS Adder

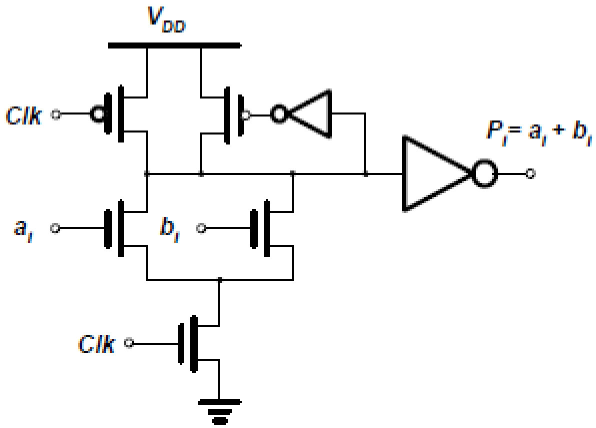


Carry Path

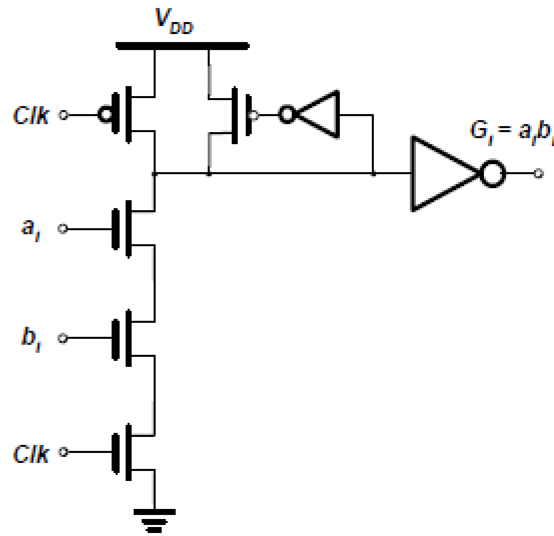
NP-CMOS Adder



Example: Domino Adder



Propagate



Generate

The Binary Multiplication

$$Z = \ddot{X} \times Y = \sum_{k=0}^{M+N-1} Z_k 2^k$$

$$= \left(\sum_{i=0}^{M-1} X_i 2^i\right)\left(\sum_{j=0}^{N-1} Y_j 2^j\right)$$

$$= \sum_{i=0}^{M-1}\left(\sum_{j=0}^{N-1} X_i Y_j 2^{i+j}\right)$$

with

$$X = \sum_{i=0}^{M-1} X_i 2^i$$

$$Y = \sum_{j=0}^{N-1} Y_j 2^j$$

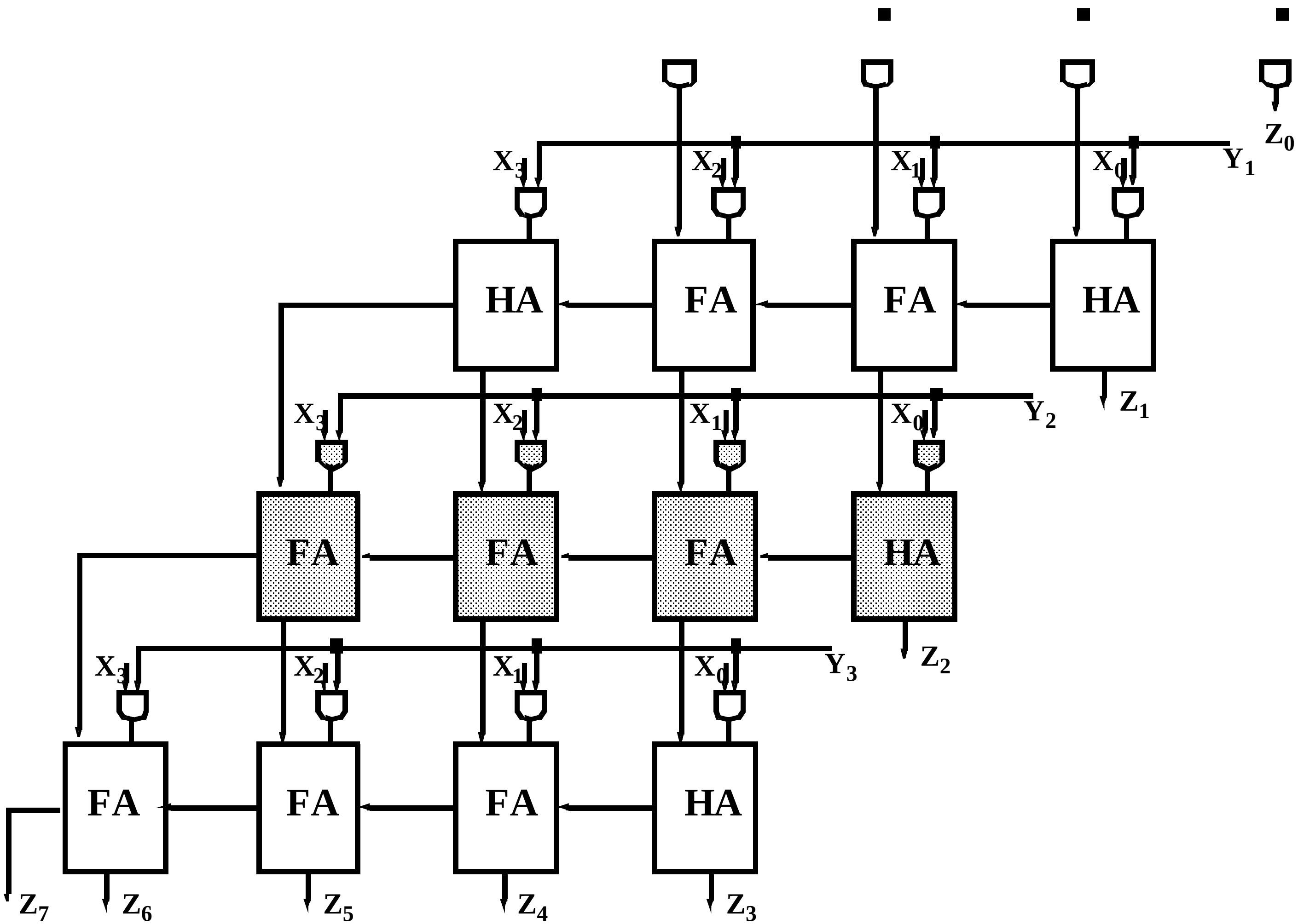
The Binary Multiplication

```
        1 0 1 0 1 0
×           1 0 1 1
-------------------
        1 0 1 0 1 0
      1 0 1 0 1 0
    0 0 0 0 0 0
+ 1 0 1 0 1 0
-------------------
1 1 1 0 0 1 1 1 0
```

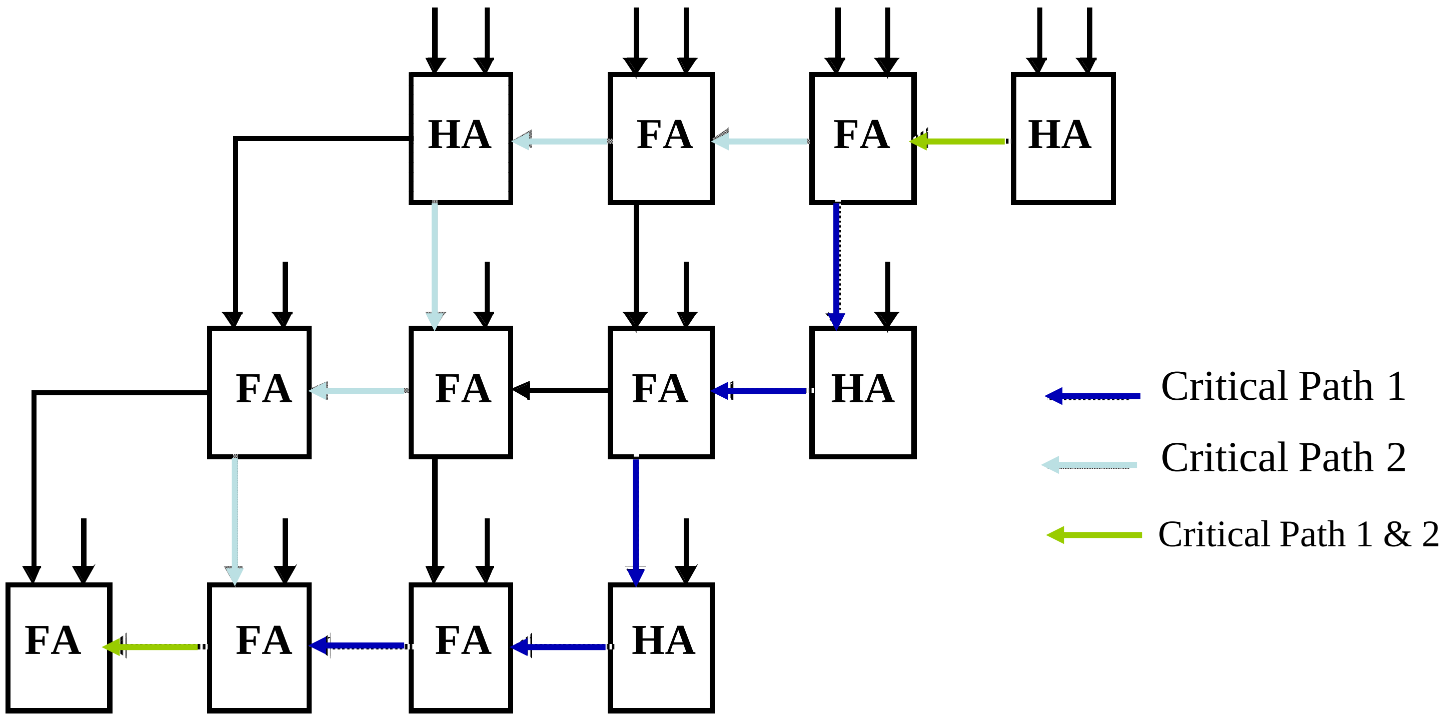
AND operation

Partial Products

The Array Multiplier

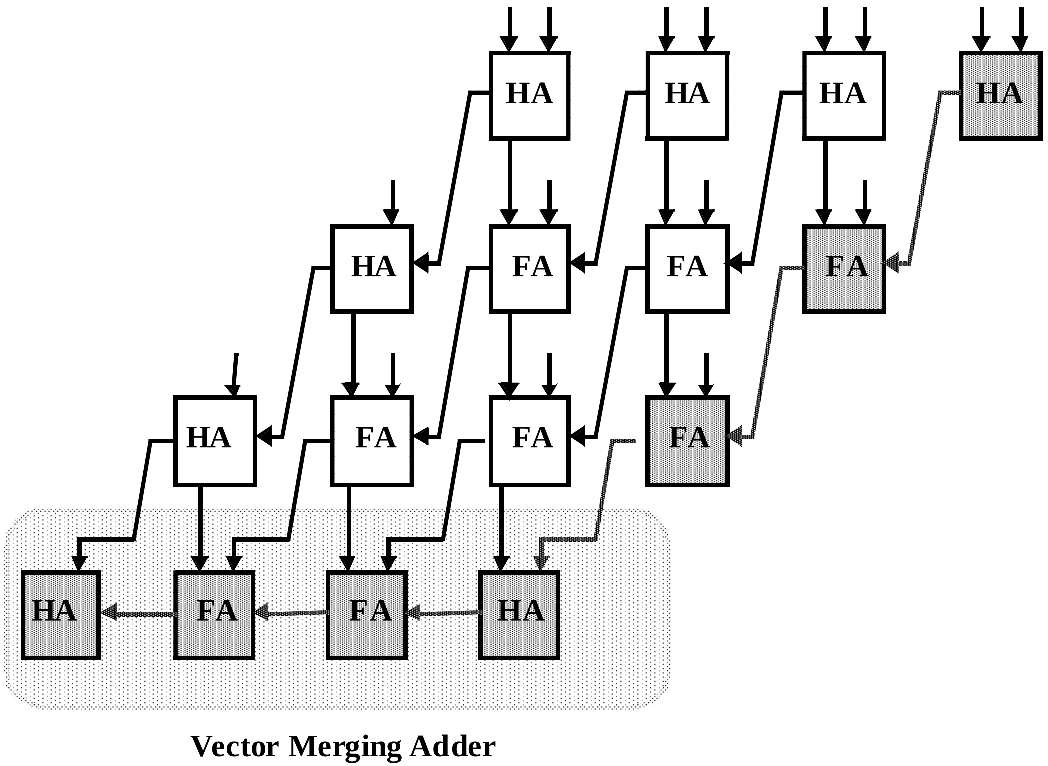


The MxN Array Multiplier — Critical Path



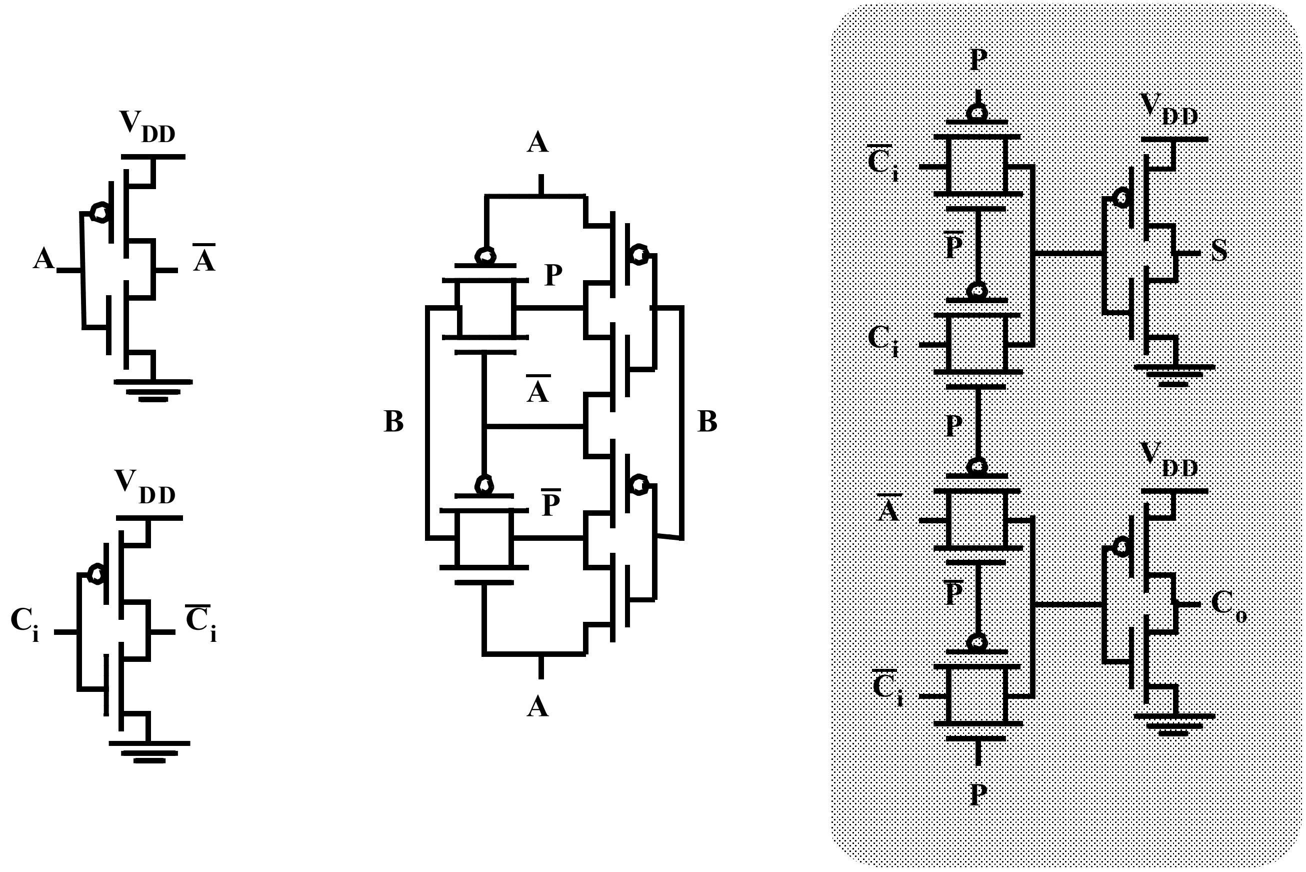
$$t_{mult} \approx [(M-1)+(N-2)]t_{carry}+(N-1)t_{sum}+(N-1)t_{and}$$

Carry-Save Multiplier



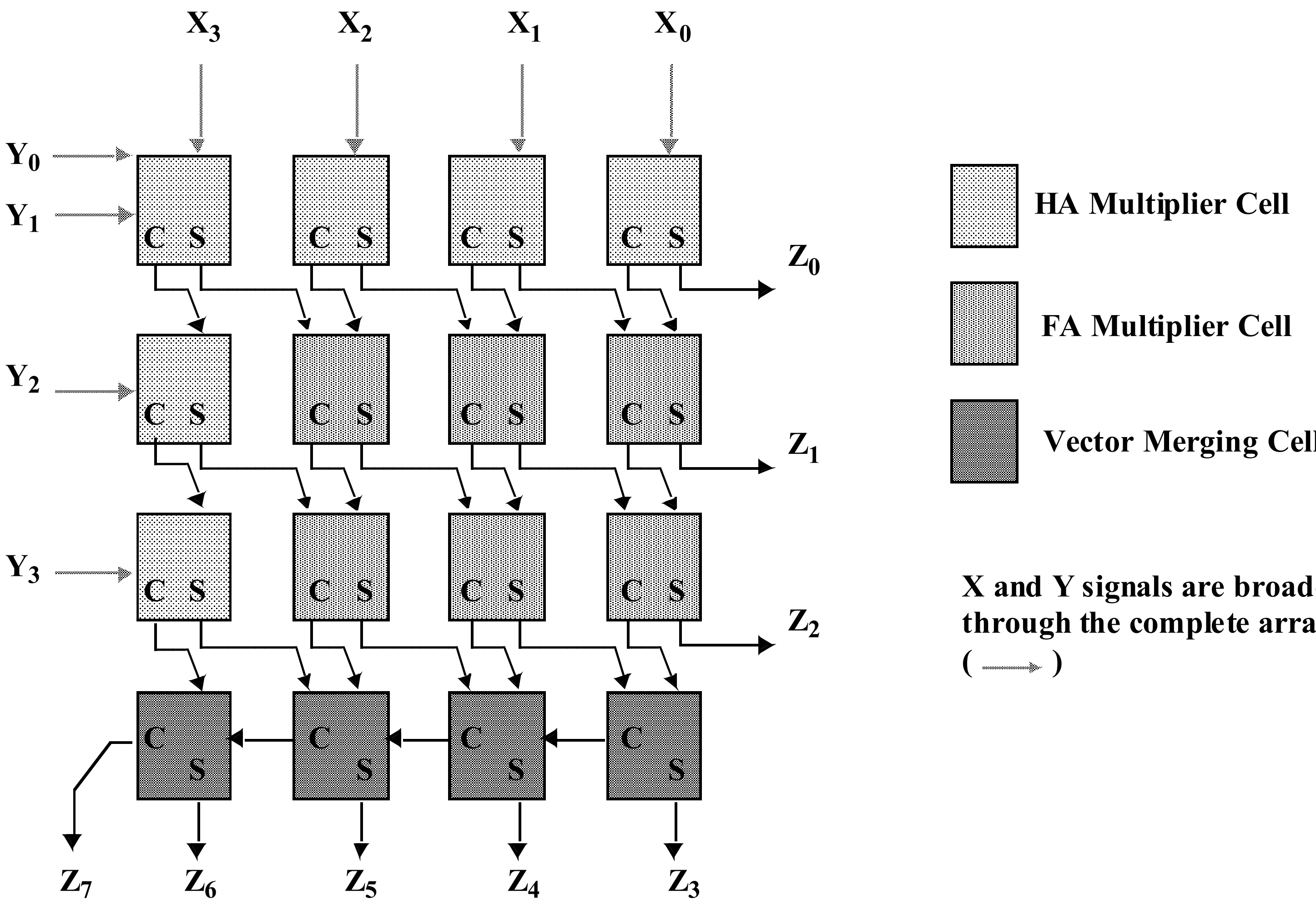
$$t_{mult} = (N-1)t_{carry} + (N-1)t_{and} + t_{merge}]$$

Adder Cells in Array Multiplier



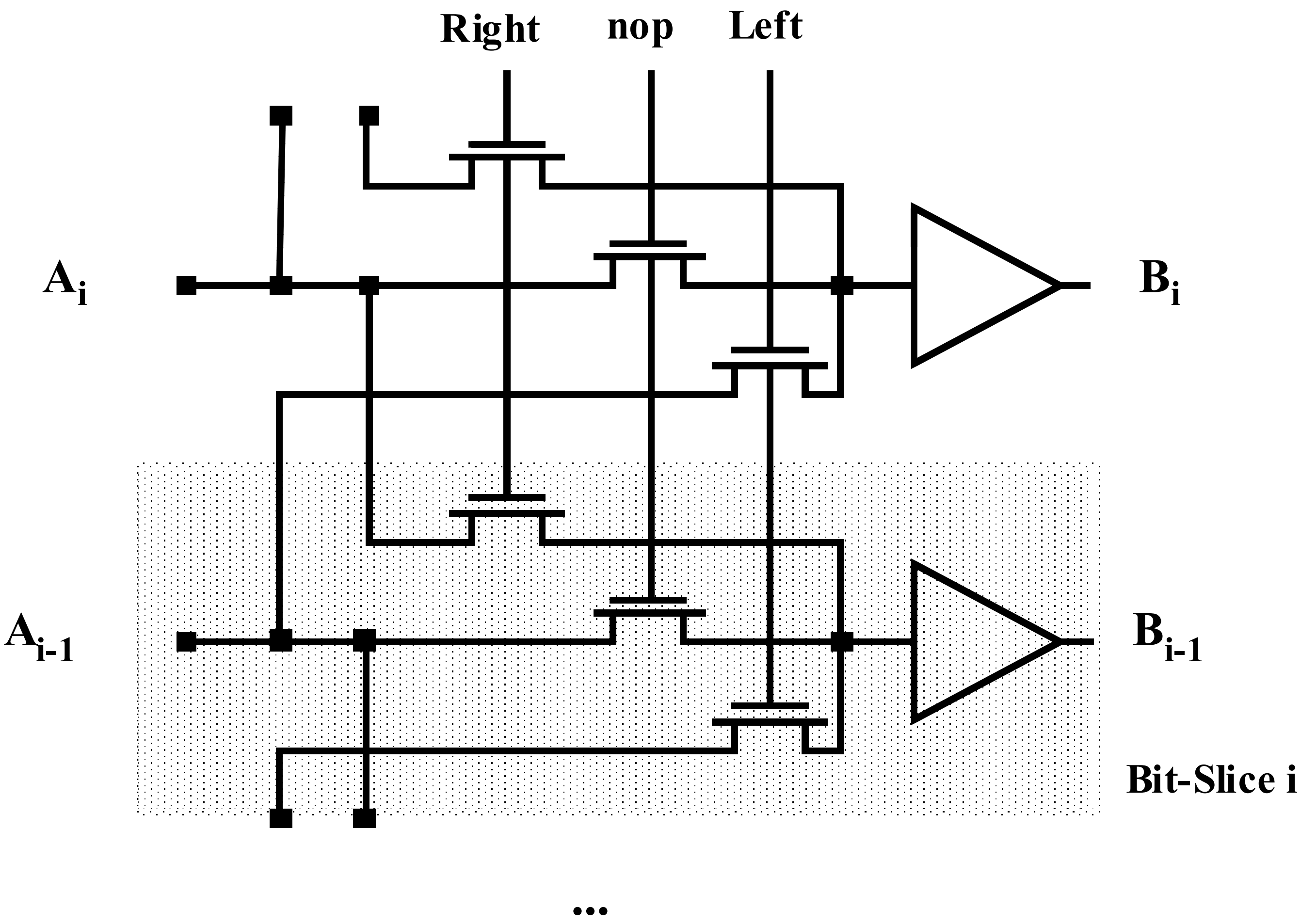
Identical Delays for Carry and Sum

Multiplier Floorplan

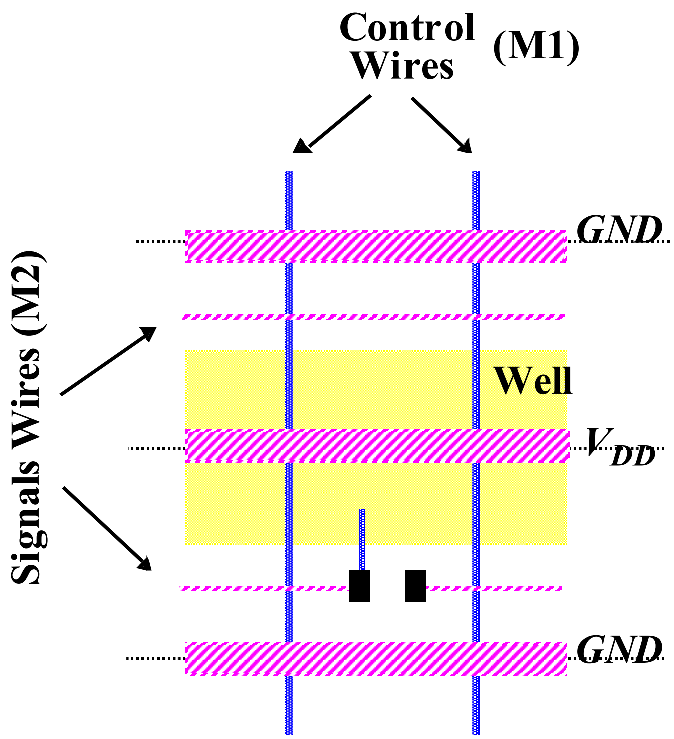


X and Y signals are broadcasted through the complete array.
( )

The Binary Shifter

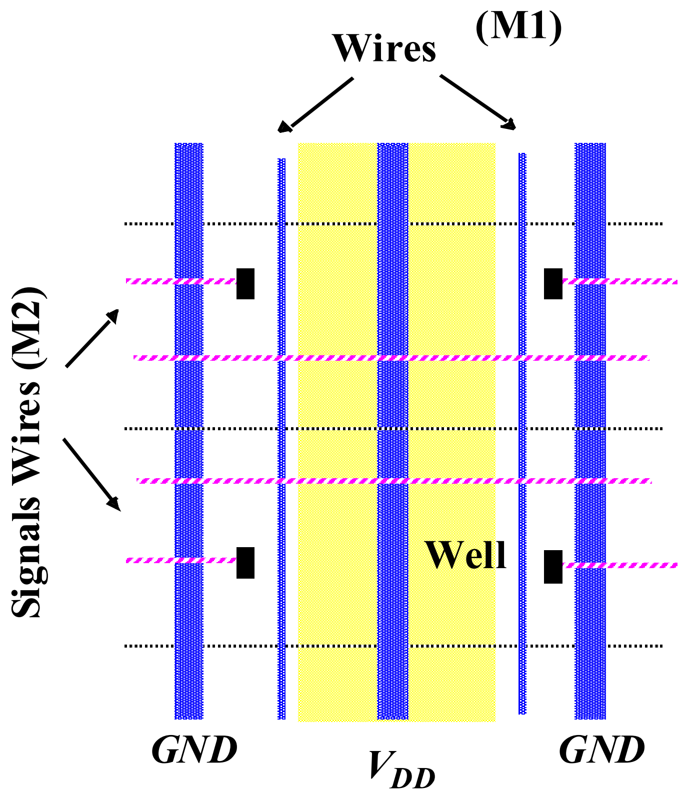


Layout Strategies for Bit-Sliced Datapaths



Approach I —

Signal and power lines parallel



Approach II —

Signal and power lines perpendicular