ECE 645: Lecture 5

Fast Adders:
Parallel Prefix Network Adders,
Conditional-Sum Adders,
& Carry-Skip Adders

Required Reading

Behrooz Parhami,
Computer Arithmetic: Algorithms and Hardware Design

Chapter 6.4, Carry Determination as Prefix Computation
Chapter 6.5, Alternative Parallel Prefix Networks

Chapter 7.4, Conditional-Sum Adder
Chapter 7.5, Hybrid Adder Designs

Chapter 7.1, Simple Carry-Skip Adders

Note errata at:
http://www.ece.ucsb.edu/~parhami/text_comp_arit_1ed.htm#errors

Recommended Reading

J-P. Deschamps, G. Bioul, G. Sutter,
Synthesis of Arithmetic Circuits: FPGA, ASIC and Embedded Systems

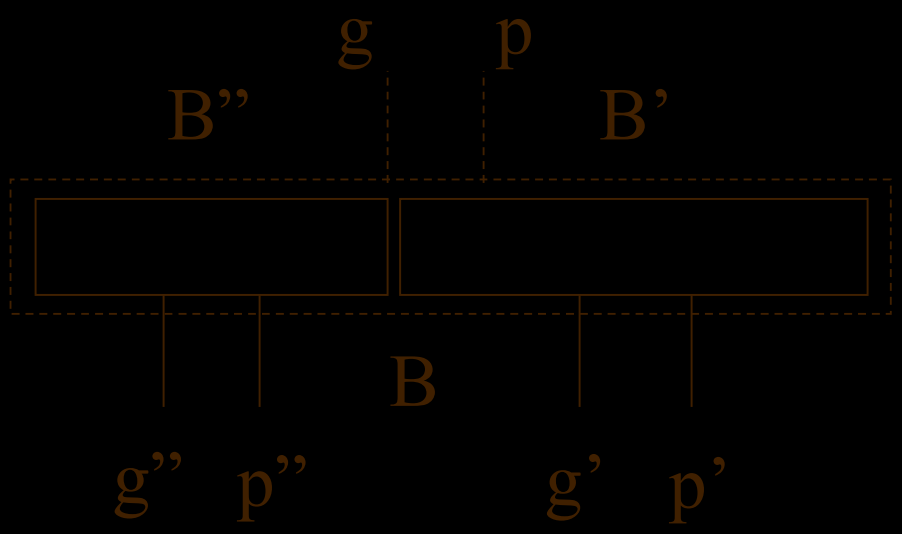
Chapter 11.1.9, Prefix Adders

Chapter 11.1.3, Carry-Skip Adder
Chapter 11.1.4, Optimization of Carry-Skip Adders
Chapter 11.1.10, FPGA Implementations of Adders
Chapter 11.1.11, Long-Operand Adders

Parallel Prefix Network Adders

Parallel Prefix Network Adders

Basic component - Carry operator (1)

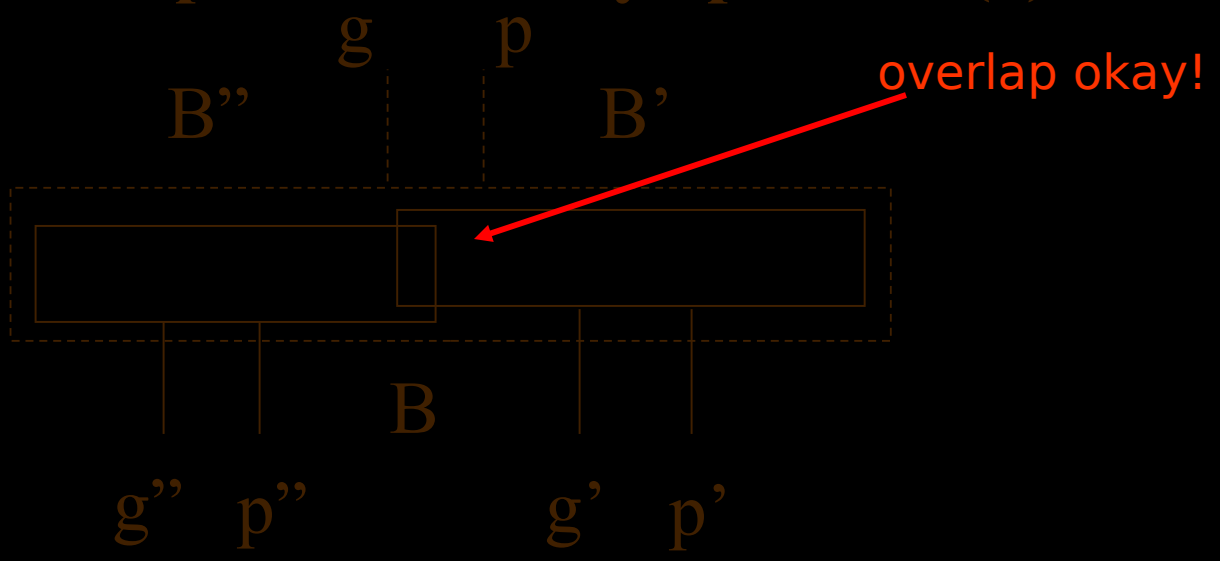


$$g = g'' + g'p''$$
$$p = p'p''$$

$$(g, p) = (g', p') ¢ (g'', p'') = (g'' + g'p'', p'p'')$$

Parallel Prefix Network Adders

Basic component - Carry operator (2)



$$g = g'' + g'p''$$
$$p = p'p''$$

$$(g, p) = (g', p') \not\circ (g'', p'') = (g'' + g'p'', p'p'')$$

Properties of the carry operator ¢

Associative

$$[(g_1, p_1) \not\cent (g_2, p_2)] \not\cent (g_3, p_3) = (g_1, p_1) \not\cent [(g_2, p_2) \not\cent (g_3, p_3)]$$

Not commutative

$$(g_1, p_1) \not\cent (g_2, p_2) \neq (g_2, p_2) \not\cent (g_1, p_1)$$

Parallel Prefix Network Adders

Major concept

Given:

(g_0, p_0) (g_1, p_1) (g_2, p_2) $\cdots$ (g_{k-1}, p_{k-1})

Find:

$(g_{[0,0]}, p_{[0,0]})$ $(g_{[0,1]}, p_{[0,1]})$ $(g_{[0,2]}, p_{[0,2]})$ $\cdots$ $(g_{[0,k-1]}, p_{[0,k-1]})$

block generate from index 0 to k-1

$$c_i = g_{[0,i-1]} + c_0 p_{[0,i-1]}$$

Similar to Parallel Prefix Sum Problem

Parallel Prefix Sum Problem

Given:

$x_0 \quad x_1 \quad x_2 \quad \cdots \quad x_{k-1}$

Find:

$x_0 \quad x_0+x_1 \quad x_0+x_1+x_2 \quad \cdots \quad x_0+x_1+x_2+\ \ldots +\ x_{k-1}$

Parallel Prefix Adder Problem

Given:

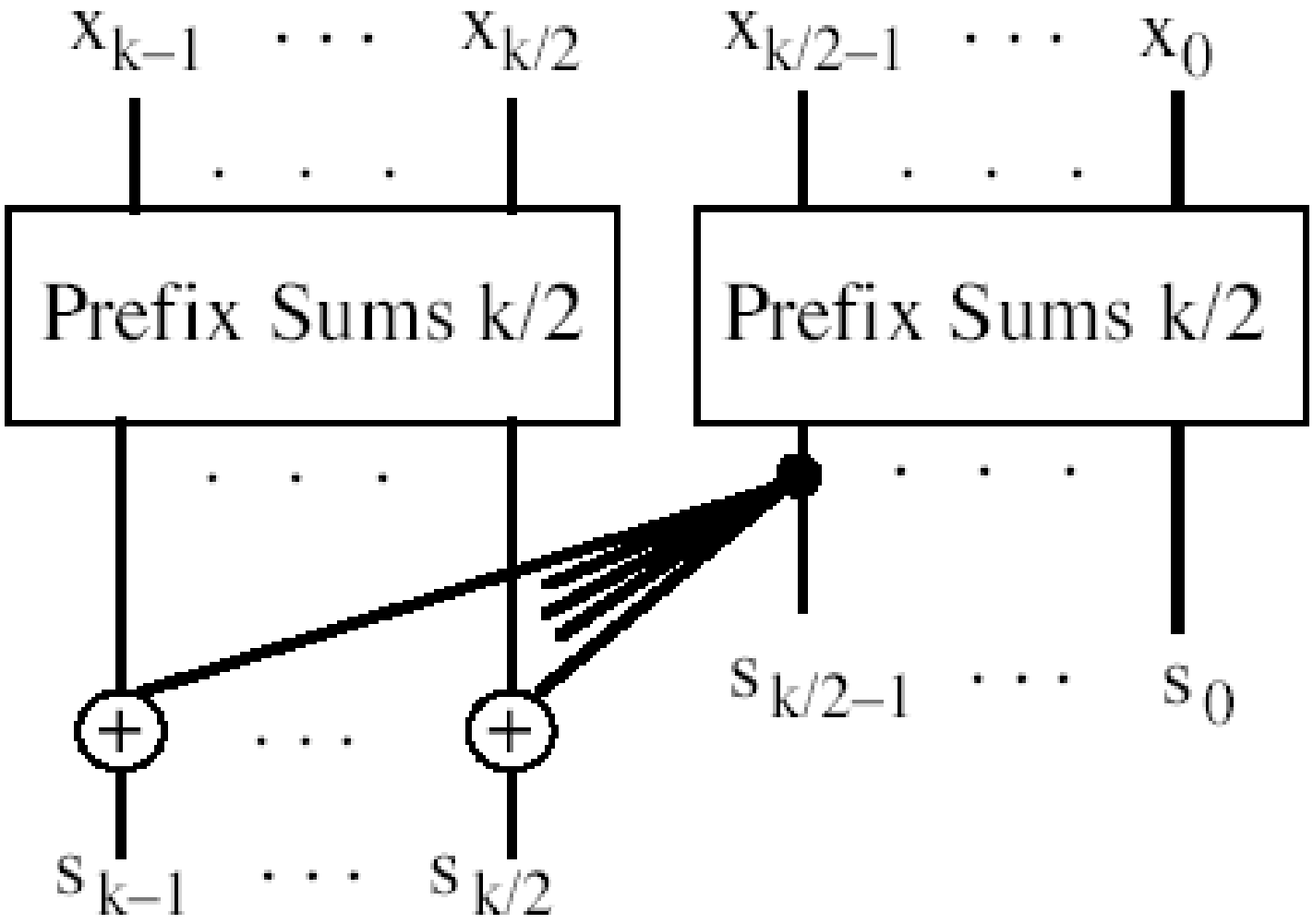
$x_0 \quad x_1 \quad x_2 \quad \cdots \quad x_{k-1}$

Find:

$x_0 \quad x_0 \not\subset x_1 \quad x_0 \not\subset x_1 \not\subset x_2 \quad \cdots \quad x_0 \not\subset x_1 \not\subset x_2 \not\subset \ldots \not\subset x_{k-1}$

where $x_i = (g_i, p_i)$

Parallel Prefix Sums Network I



Parallel Prefix Sums Network I – Cost (Area) Analysis

Cost = C(k) $= 2\,C(k/2) + k/2 =$

$= 2\,[2C(k/2) + k/2] + k/2 = 4\,C(k/2) + k/2 + k/2 =$

$= \ldots. =$

$= 2^{\log_2 k-1} C(2) + k/2\,(\log_2 k - 1) =$

$= \mathbf{k/2 \log_2 k}$

$C(2) = 1$

Example:

C(16) $= 2\,C(8) + 8 = 2[2\,C(4) + 4] + 8 =$

$= 4\,C(4) + 16 = 4\,[2\,C(2) + 2] + 16 =$

$= 8\,C(2) + 24 = 8 + 24 = \mathbf{32} = (16/2) \log_2 16$

Parallel Prefix Sums Network I – Delay Analysis

Delay = D(k) = D(k/2) + 1 =

= [D(k/4) + 1] + 1 = D(k/4) + 1 + 1 =

= …. =

= $\log_2 k$

$D(2) = 1$

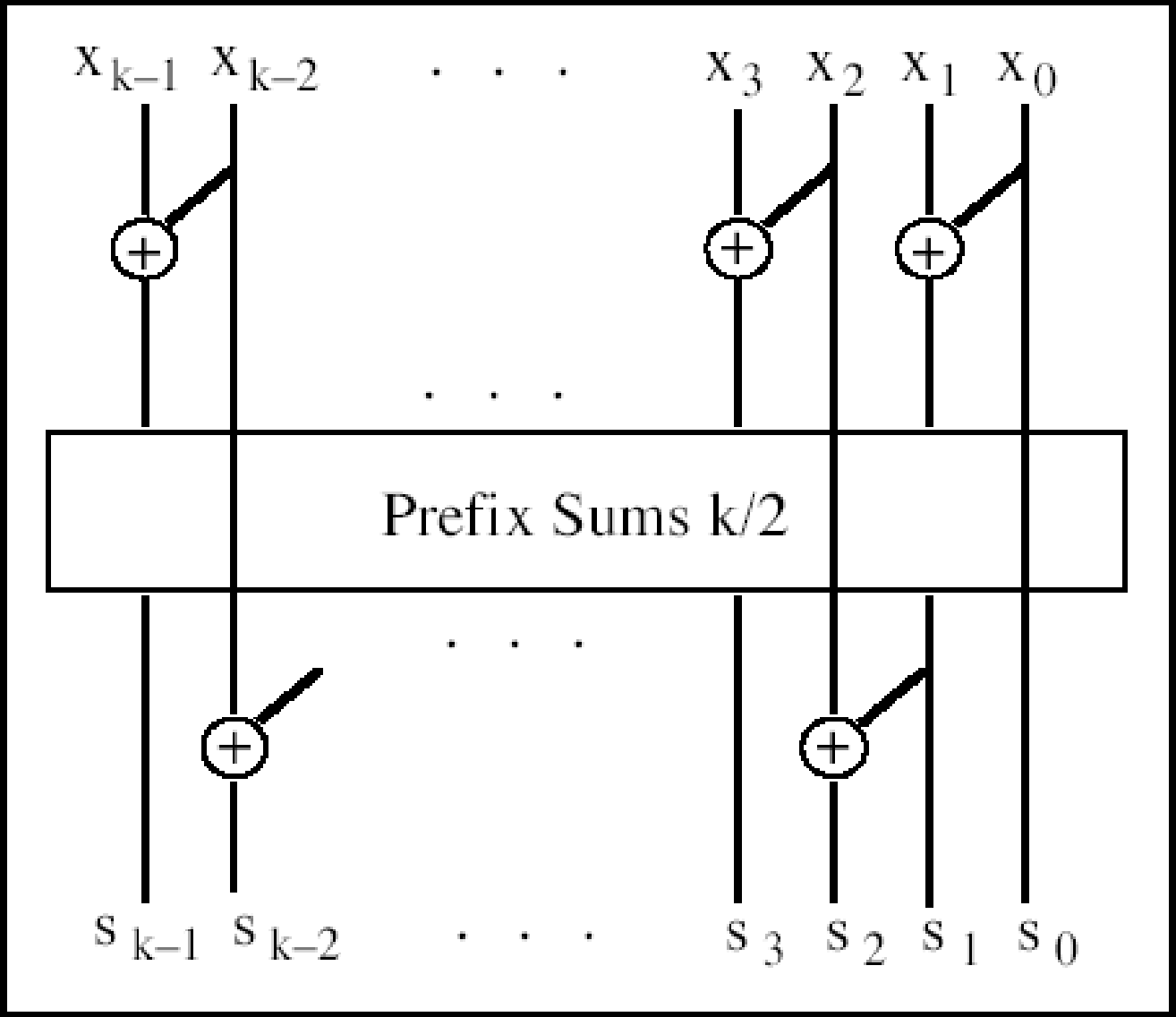
Example:

D(16) = D(8) + 1 = [D(4) + 1] + 1 =

= D(4) + 2 = [D(2) + 1] + 2 =

= **4** = $\log_2 16$

Parallel Prefix Sums Network II (Brent-Kung)



Parallel Prefix Sums Network II – Cost (Area) Analysis

$$\textbf{Cost} = \textbf{C(k)} = C(k/2) + k-1 =$$
$$= [C(k/4) + k/2-1] + k-1 = C(k/4) + 3k/2 - 2 =$$
$$= \ldots. =$$
$$= C(2) + (2k - 2k/2^{(\log_2 k-1)}) - (\log_2 k-1) =$$
$$= \mathbf{2k - 2 - \log_2 k}$$

$$\left| C(2) = 1 \right|$$

Example:

$$\mathbf{C(16)} = C(8) + 16-1 = [C(4) + 8-1] + 16-1 =$$
$$= C(2) + 4-1 + 24-2 = 1 + 28 - 3 = \mathbf{26}$$
$$= 2 \cdot 16 - 2 - \log_2 16$$

Parallel Prefix Sums Network II – Delay Analysis

Delay = D(k) = D(k/2) + 2 =

= [D(k/4) + 2] + 2 = D(k/4) + 2 + 2 =

= …. =

= 2 $\log_2$k - 1

D(2) = 1

Example:

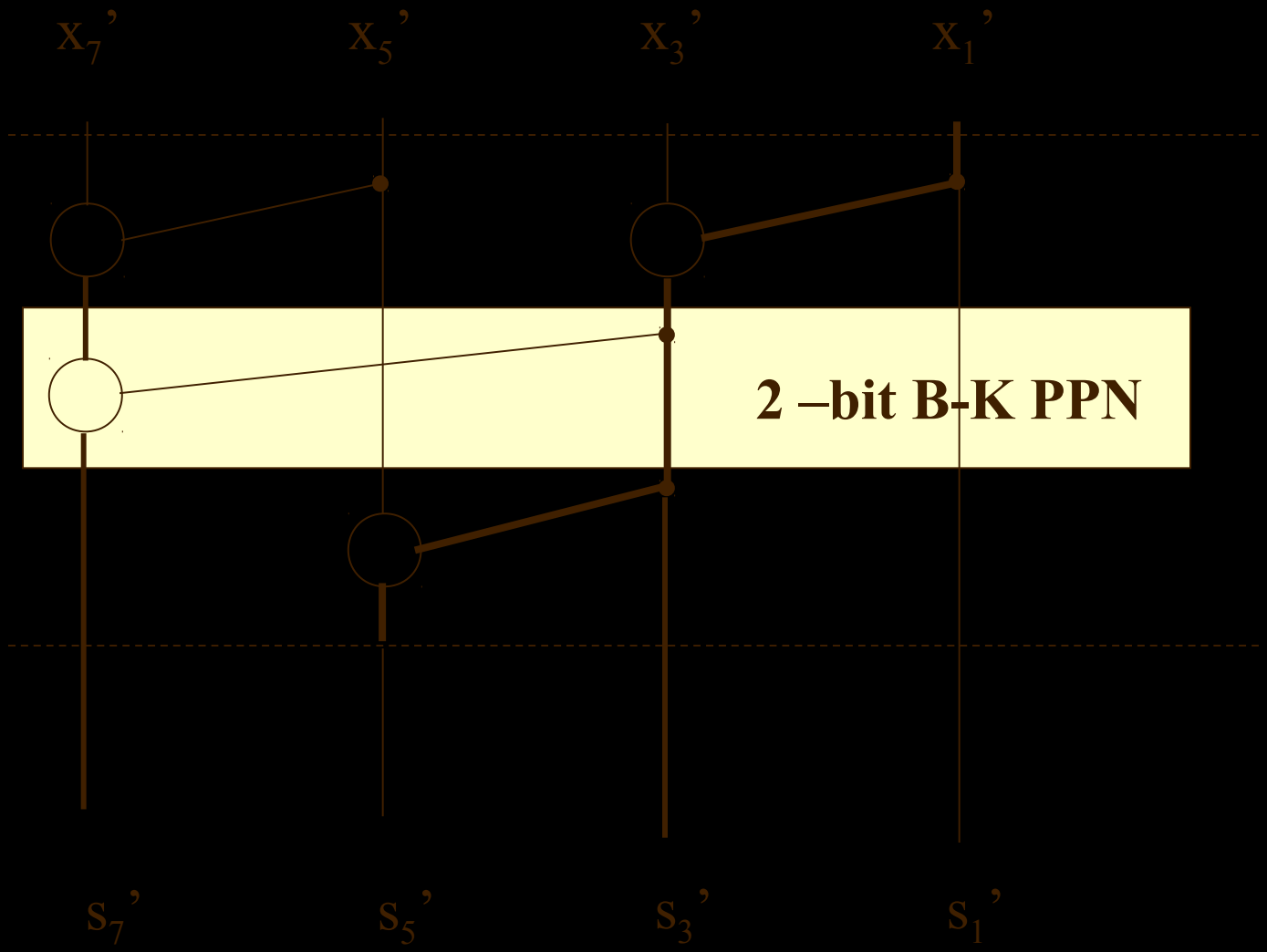
D(16) = D(8) + 2 = [D(4) + 2] + 2 =

= D(4) + 4 = [D(2) + 2] + 4 =

= 7 = 2 $\log_2$ 16 - 1

8-bit Brent-Kung Parallel Prefix Network

4-bit Brent-Kung Parallel Prefix Network



8-bit Brent-Kung Parallel Prefix Network Critical Path

Critical Path

GP

$g_i = x_i \; y_i$

$p_i = x_i \oplus y_i$

1 gate delay

¢

$g = g'' + g' \; p''$

$p = p' \; p''$

2 gate delays

C

$c_{i+1} = g_{[0,i]} + c_0 \, p_{[0,i]}$

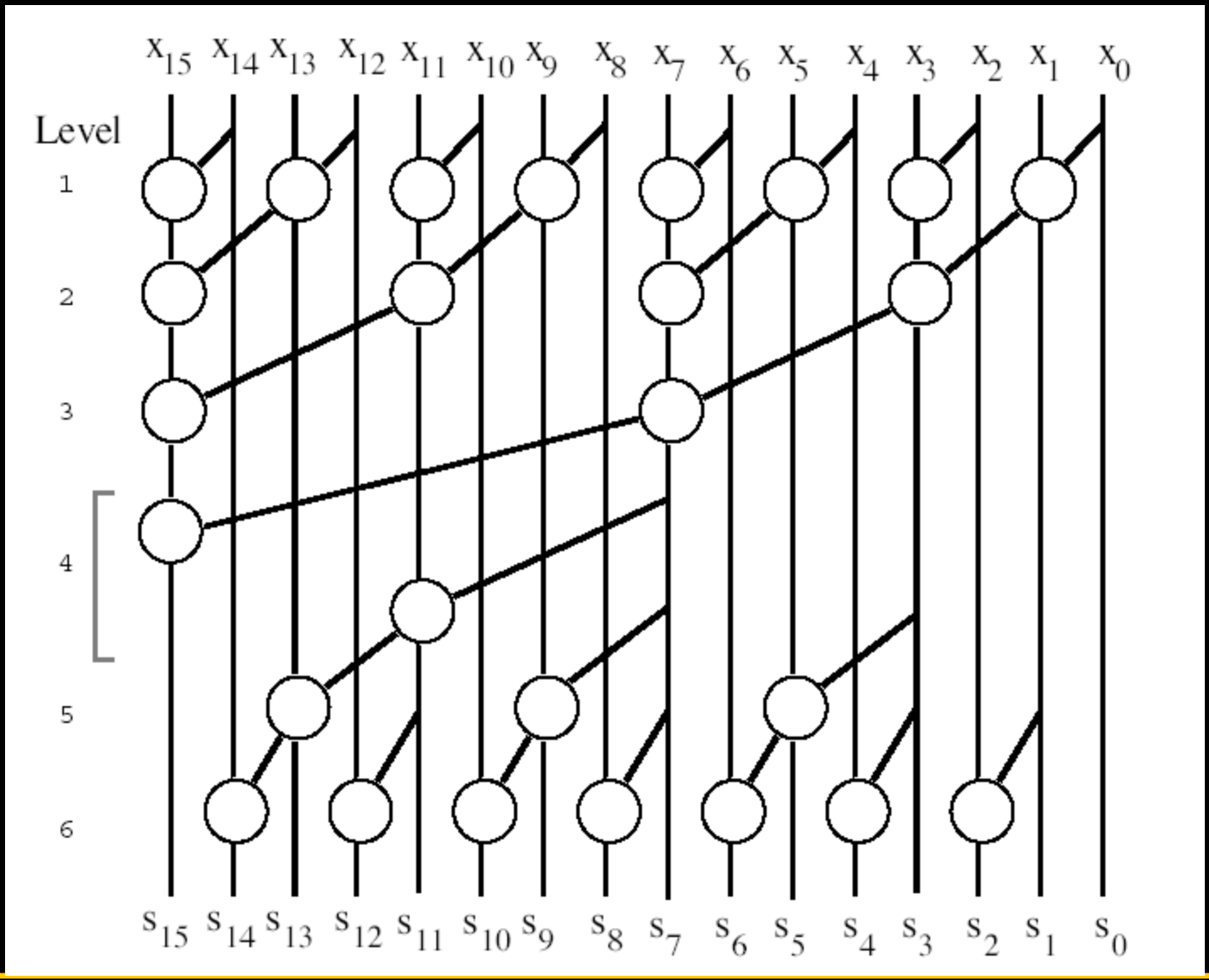
2 gate delays

S

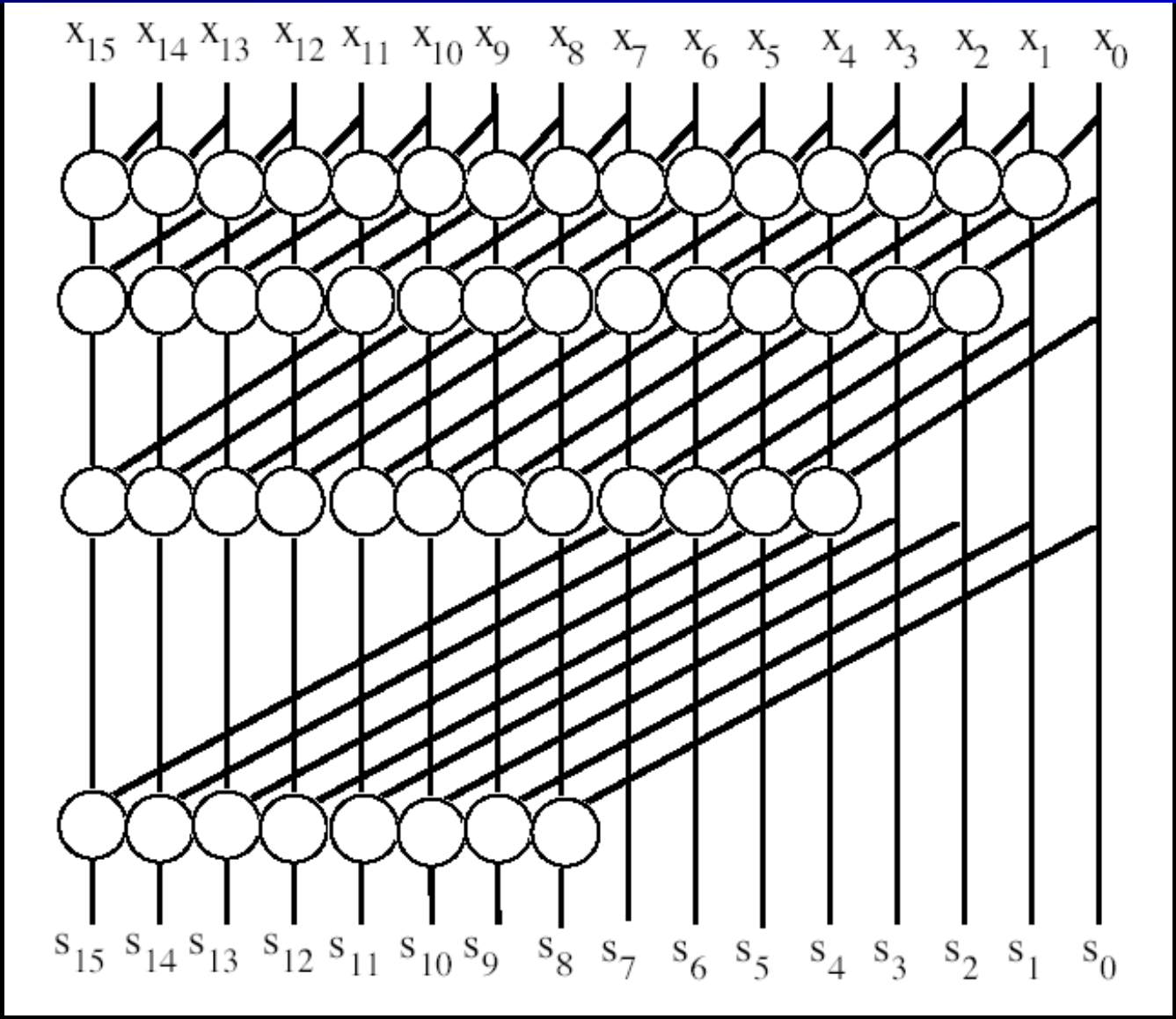
$s_i = p_i \oplus c_i$

1 gate delay

Brent-Kung Parallel Prefix Graph for 16 Inputs



Kogge-Stone Parallel Prefix Graph for 16 Inputs

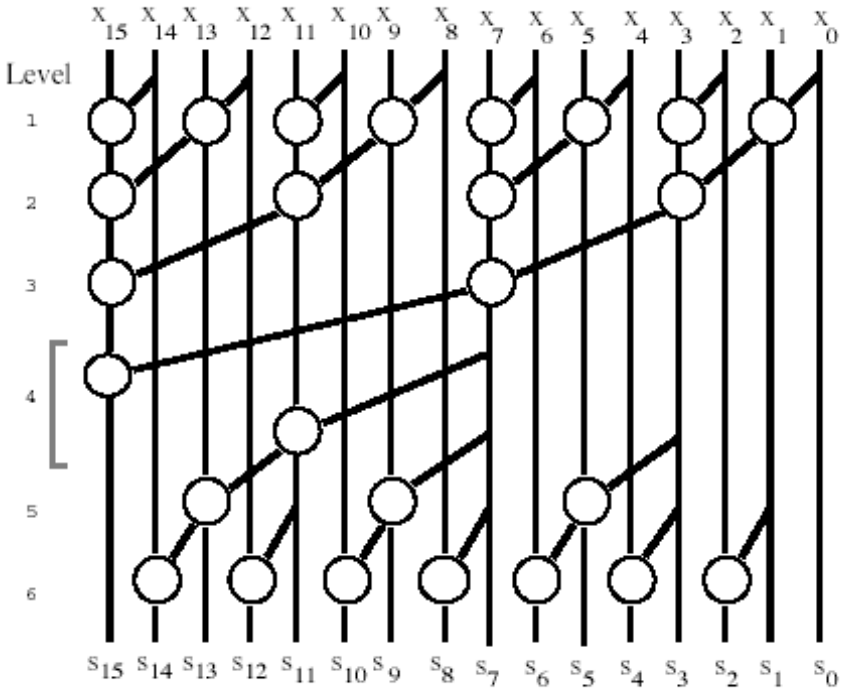


Parallel Prefix Network Adders

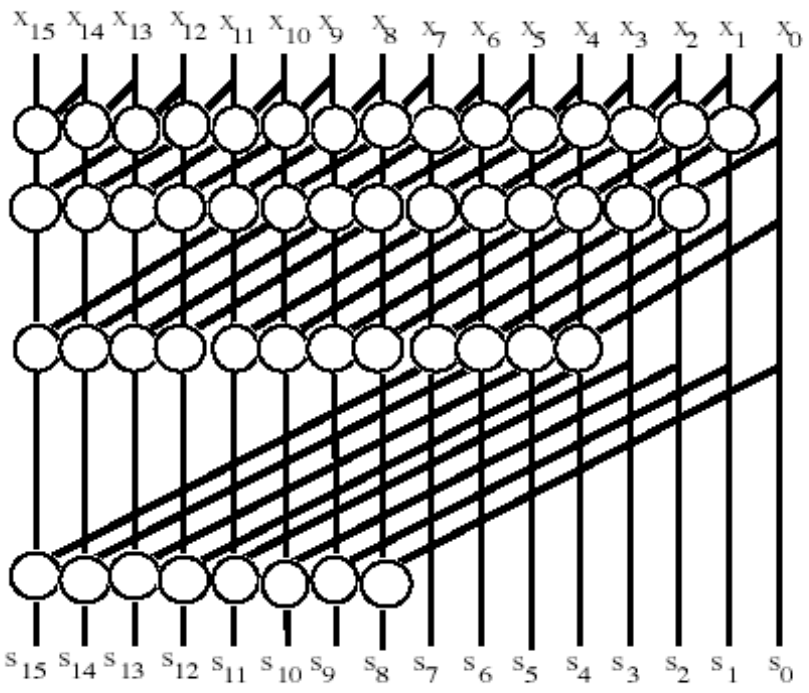
Comparison of architectures

	Network 2 Brent-Kung	Hybrid	Kogge-Stone
Delay(k)	$2 \log_2 k - 2$	$\log_2 k + 1$	$\log_2 k$
Cost(k)	$2k - 2 - \log_2 k$	$k/2 \log_2 k$	$k \log_2 k - k + 1$
Delay(16)	6	5	4
Cost(16)	26	32	49
Delay(32)	8	6	5
Cost(32)	57	80	129

Latency vs. Area Tradeoff

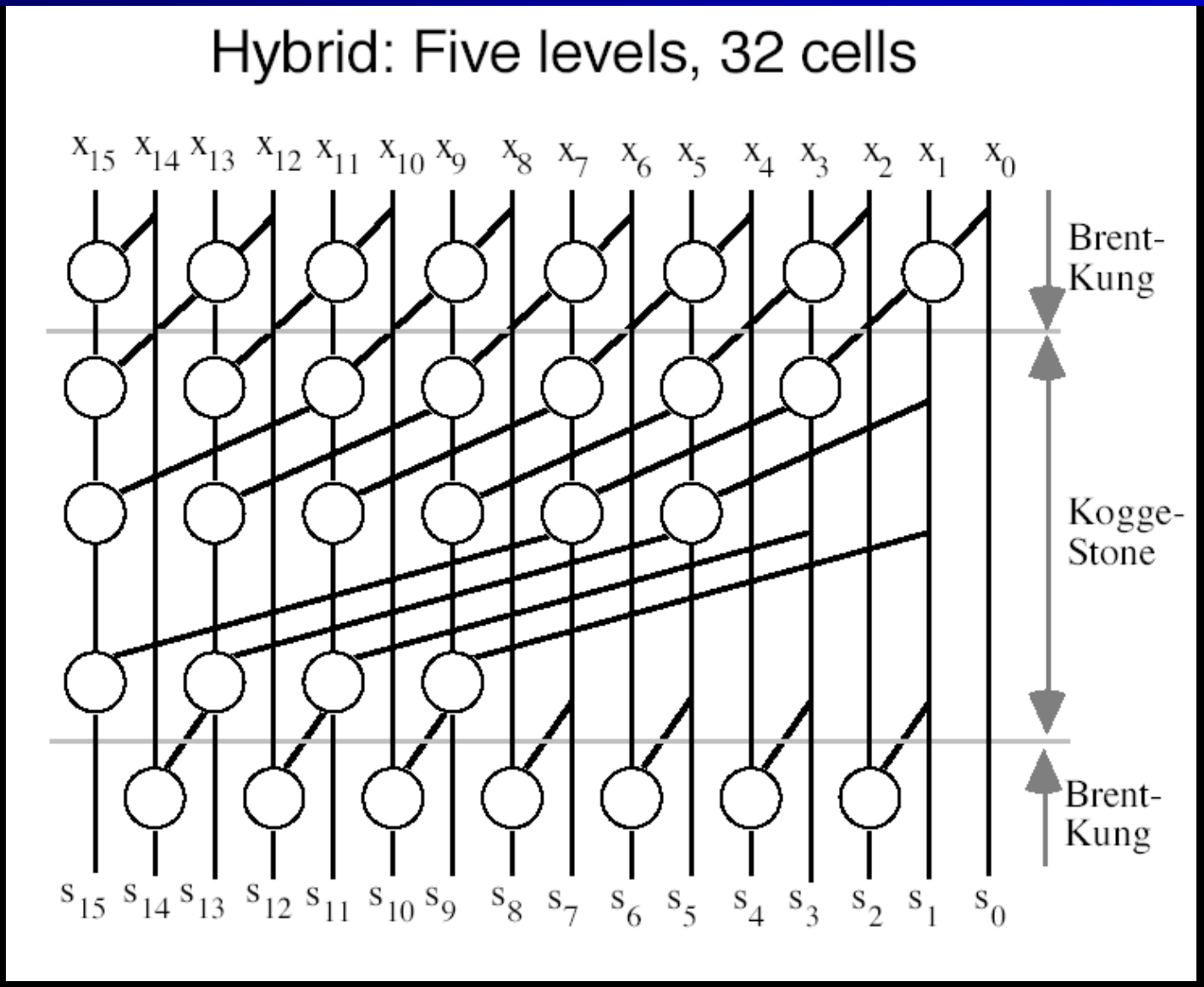


B-K: Six levels, 26 cells



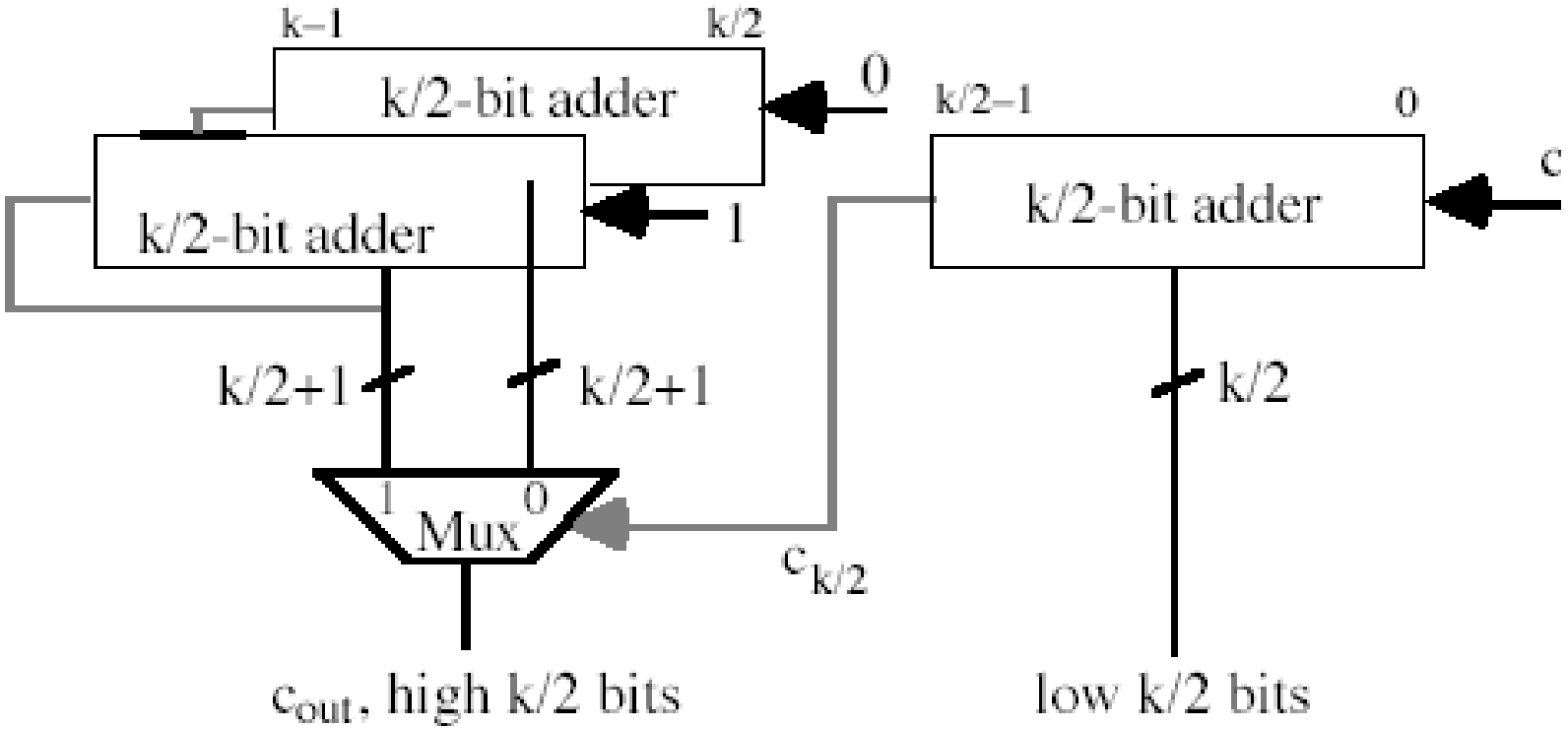
K-S: Four levels, 49 cells

Hybrid Brent-Kung/Kogge-Stone Parallel Prefix Graph for 16 Inputs

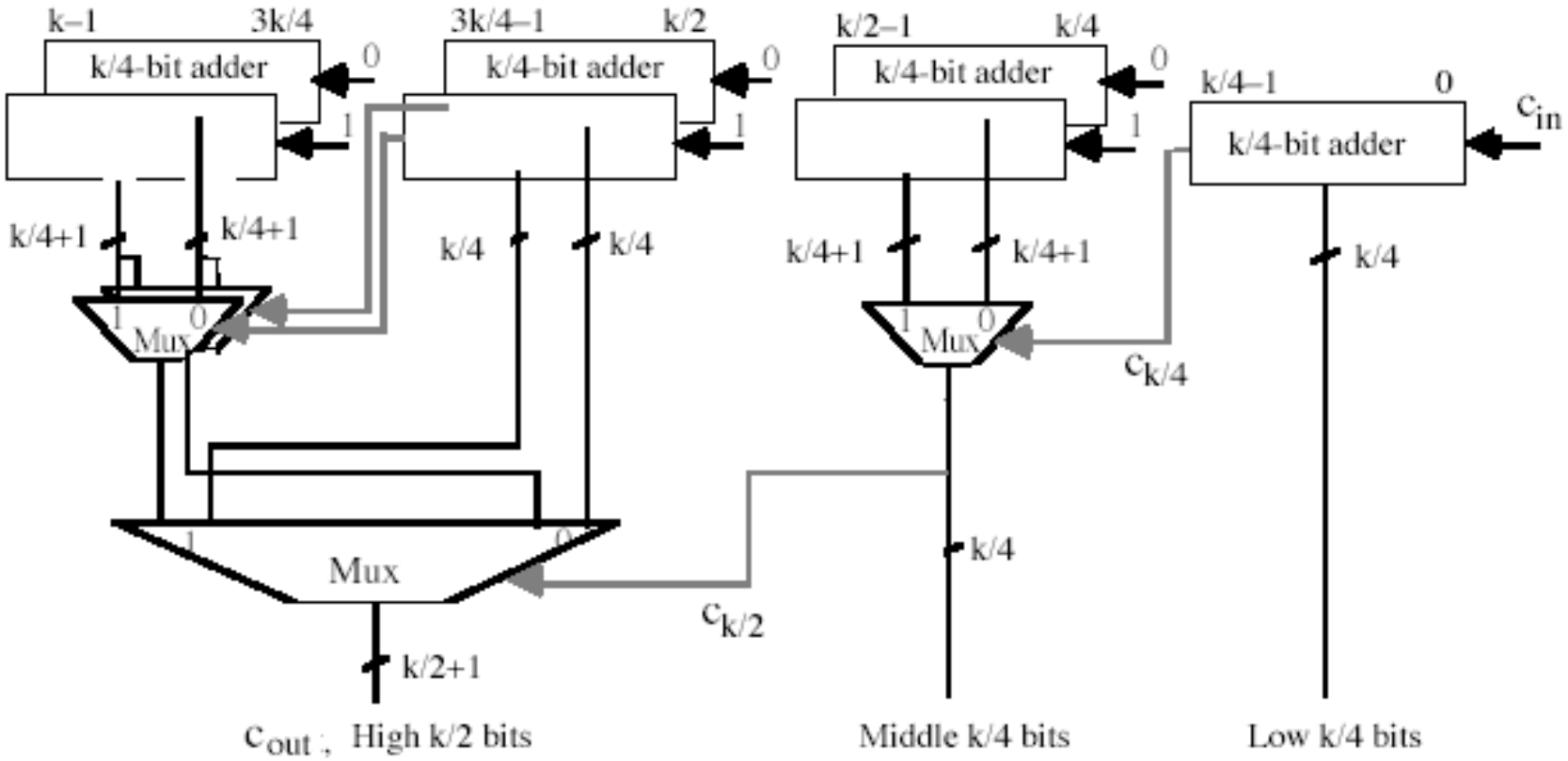


Conditional-Sum Adders

One-level k-bit Carry-Select Adder



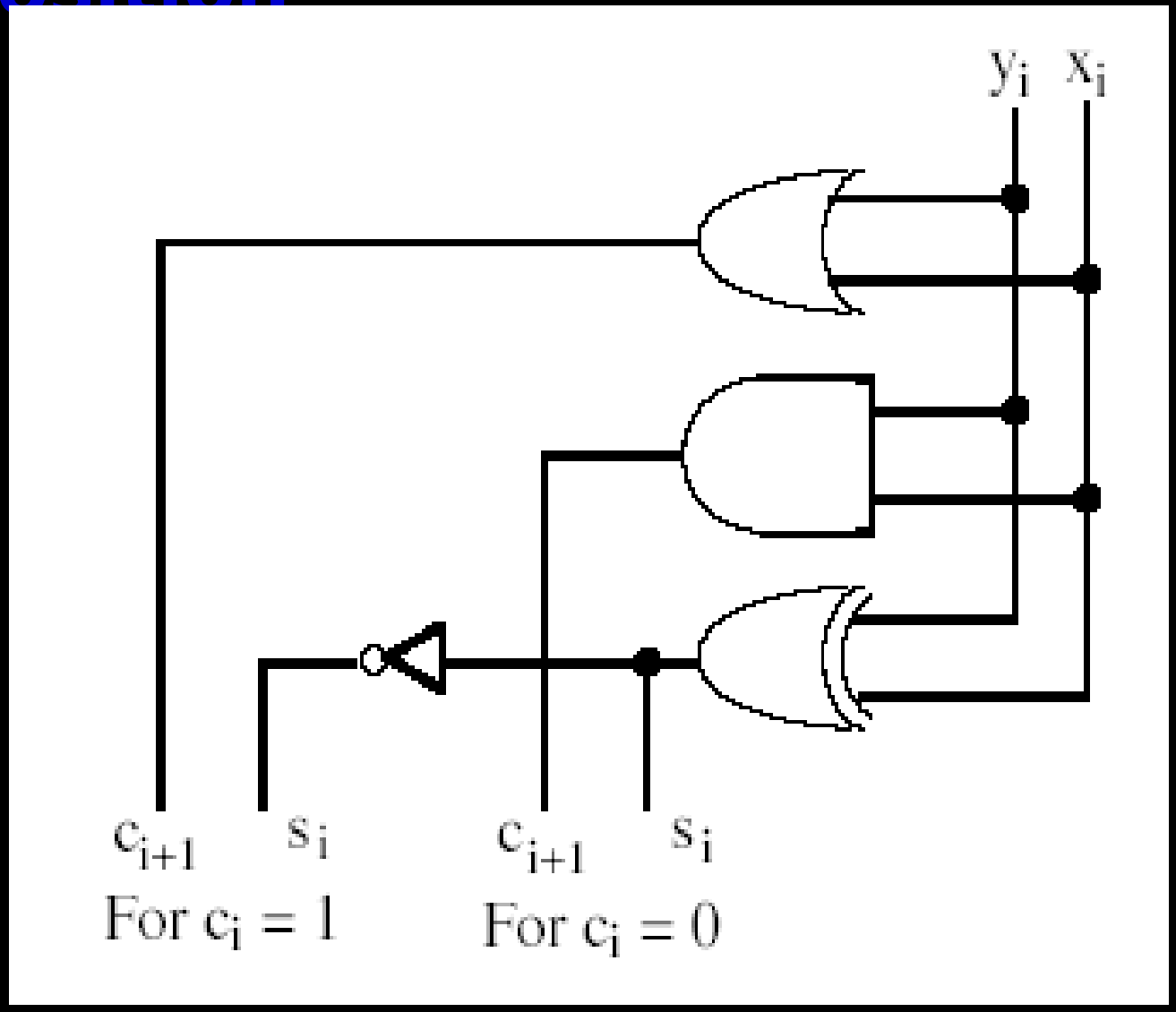
Two-level k-bit Carry Select Adder



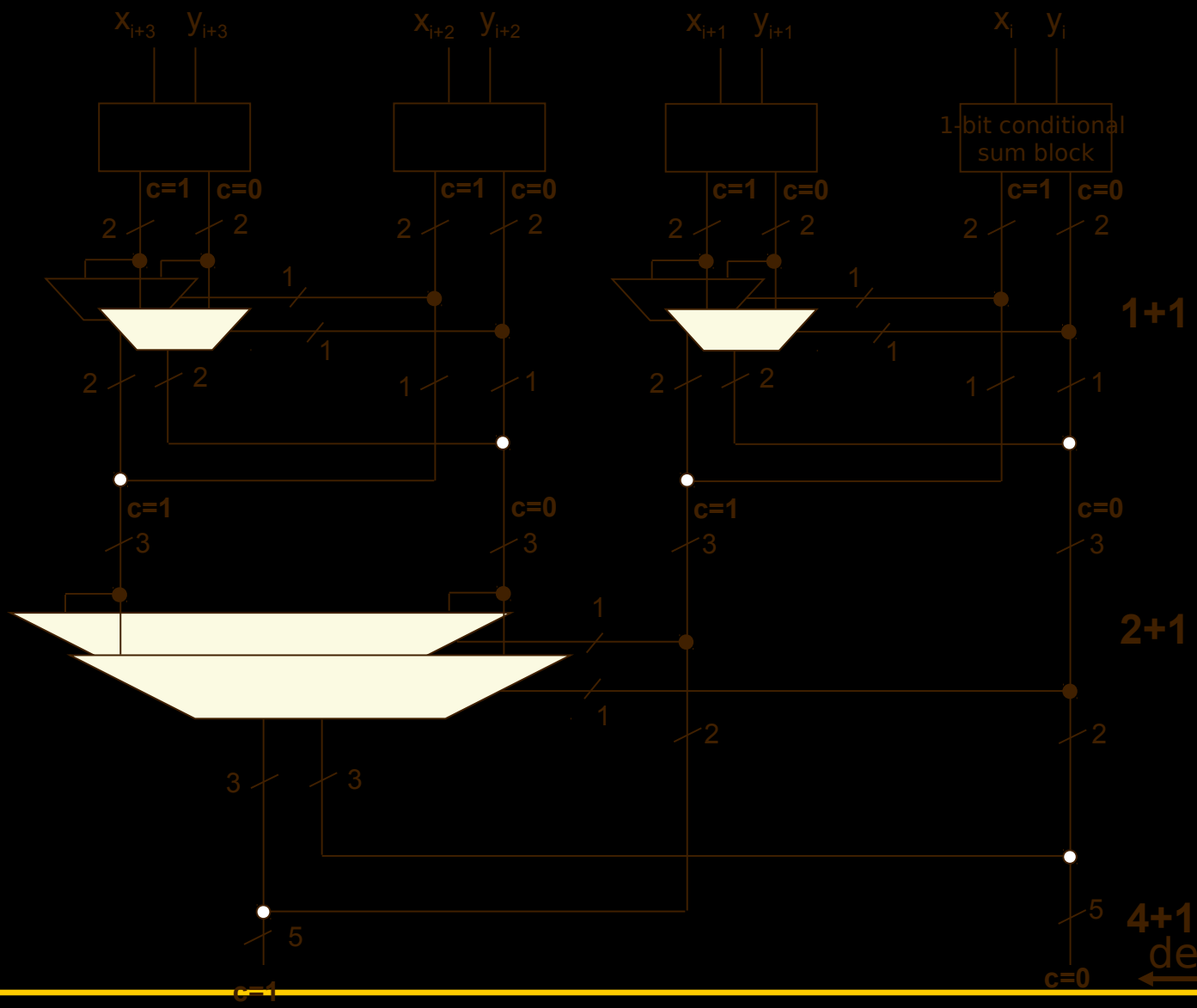
Conditional Sum Adder

- Extension of carry-select adder
- Carry select adder
  - One-level using k/2-bit adders
  - Two-level using k/4-bit adders
  - Three-level using k/8-bit adders
  - Etc.
- Assuming k is a power of two, eventually have an extreme where there are $\log_2 k$-levels using 1-bit adders
  - This is a conditional sum adder

Conditional Sum Adder: Top-Level Block for One Bit Position



Three Levels of a Conditional Sum Adder



16-Bit Conditional Sum Adder Example

			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
		x	0	0	1	0	0	1	1	0	1	1	1	0	1	0	1	0	
		y	0	1	0	0	1	0	1	1	0	1	0	1	1	1	0	1	
Block width	Block carry-in	Block sum and block carry-out																	c_{in}
1	0	s	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	1	
		c	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0
	1	s	1	0	1	1	0	1	0	1	0	1	1	0	1	0	1		
		c	0	1	0	1	1	1	1	0	1	1	0	1	1	1	1		
2	0	s	0	1	1	0	1	1	0	1	0	0	1	1	0	1	1	1	
		c	0	0	1	0	1	0	0	1	0	1	1	0	1	1	1	0	
	1	s	1	0	1	1	0	0	1	0	0	1	0	0	1	0			
		c	1	0	1	0	0	1	1	1	0	1	0	1	1	0			
4	0	s	0	1	1	0	0	0	0	1	0	0	1	1	0	1	1	1	
		c	0	0	1	0	0	1	0	0	0	1	1	0	1	1	1	0	
	1	s	0	1	1	1	0	0	1	0	0	1	0	0					
		c	0	0	1	1	0	1	1	1	0	1	0	1					
8	0	s	0	1	1	1	0	0	0	1	0	1	0	0	0	1	1	1	
		c	0	0	1	1	0	1	1	1	0	1	0	1	1	0	1	0	
	1	s	0	1	1	1	0	0	1	0									
		c	0	0	1	1	0	1	1	1									
16	0	s	0	1	1	1	0	0	1	0	0	1	0	0	0	1	1	1	
		c	0	0	1	1	0	1	1	1	0	1	0	1	1	0	1	0	
	1	s																	
		c																	

c_{out}

Conditional Sum Adder Metrics

Multilevel carry-select idea carried out to the extreme, until we arrive at single-bit blocks.

$$C(k) \approx 2C(k/2) + k + 2 \approx k\,(\log_2 k + 2) + k\,C(1)$$

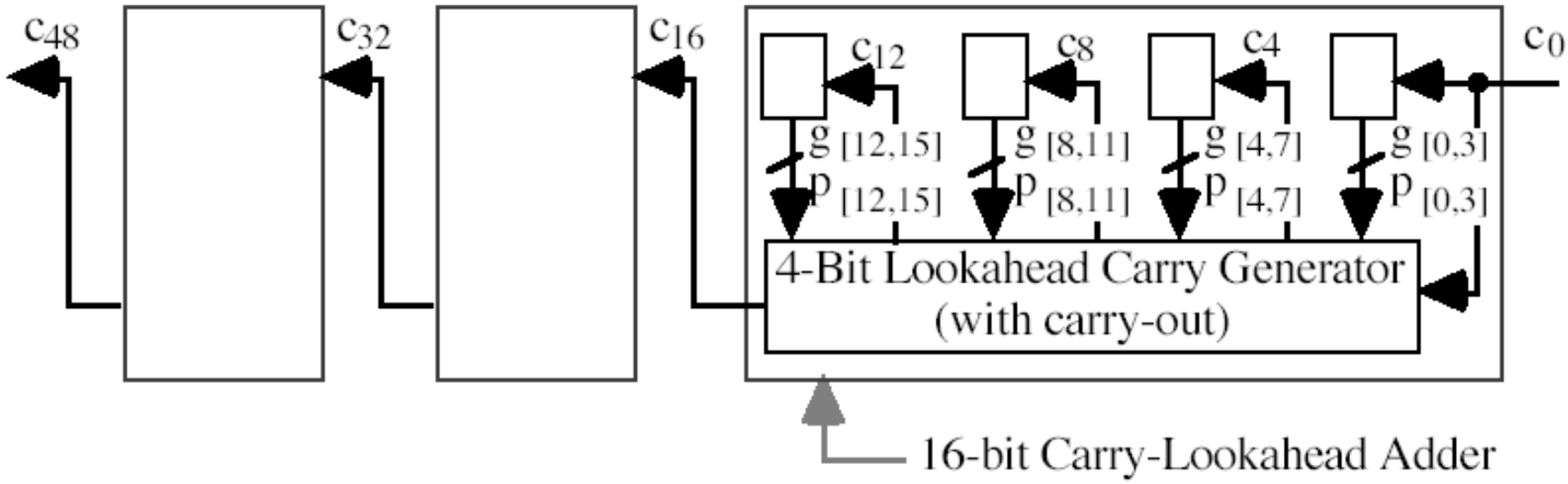
$$T(k) = T(k/2) + 1 = \log_2 k + T(1)$$

where C(1) and T(1) are the cost and delay of the circuit of Fig. 7.11 used at the top to derive the sum and carry bits with a carry-in of 0 and 1

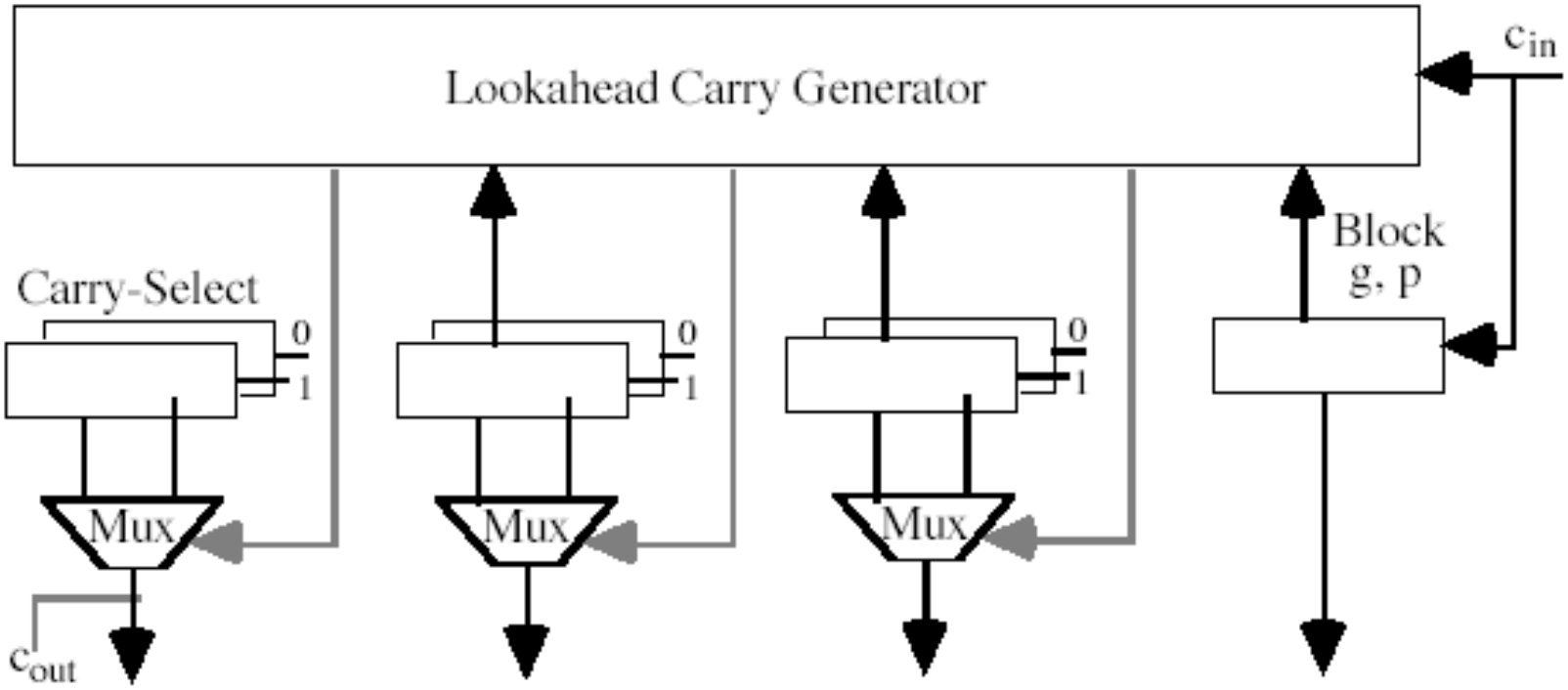
The term k + 2 in the first recurrence represents an upper bound on the number of single-bit 2-to-1 multiplexers needed for combining two k/2-bit adders into a k-bit adder

Hybrid Adders

A Hybrid Ripple-Carry/Carry-Lookahead Adder



A Hybrid Carry-Lookahead/Carry-Select Adder



Carry-Skip Adders
Fixed-Block-Size

7.1 Simple Carry-Skip Adders

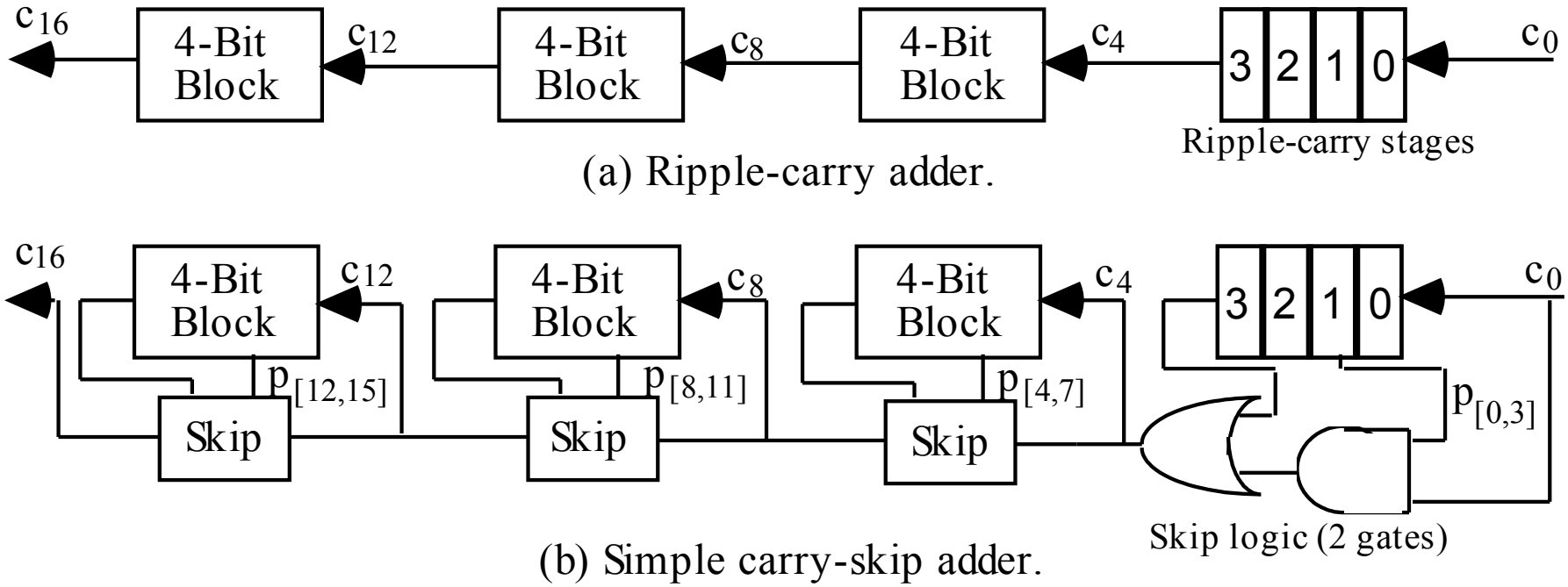
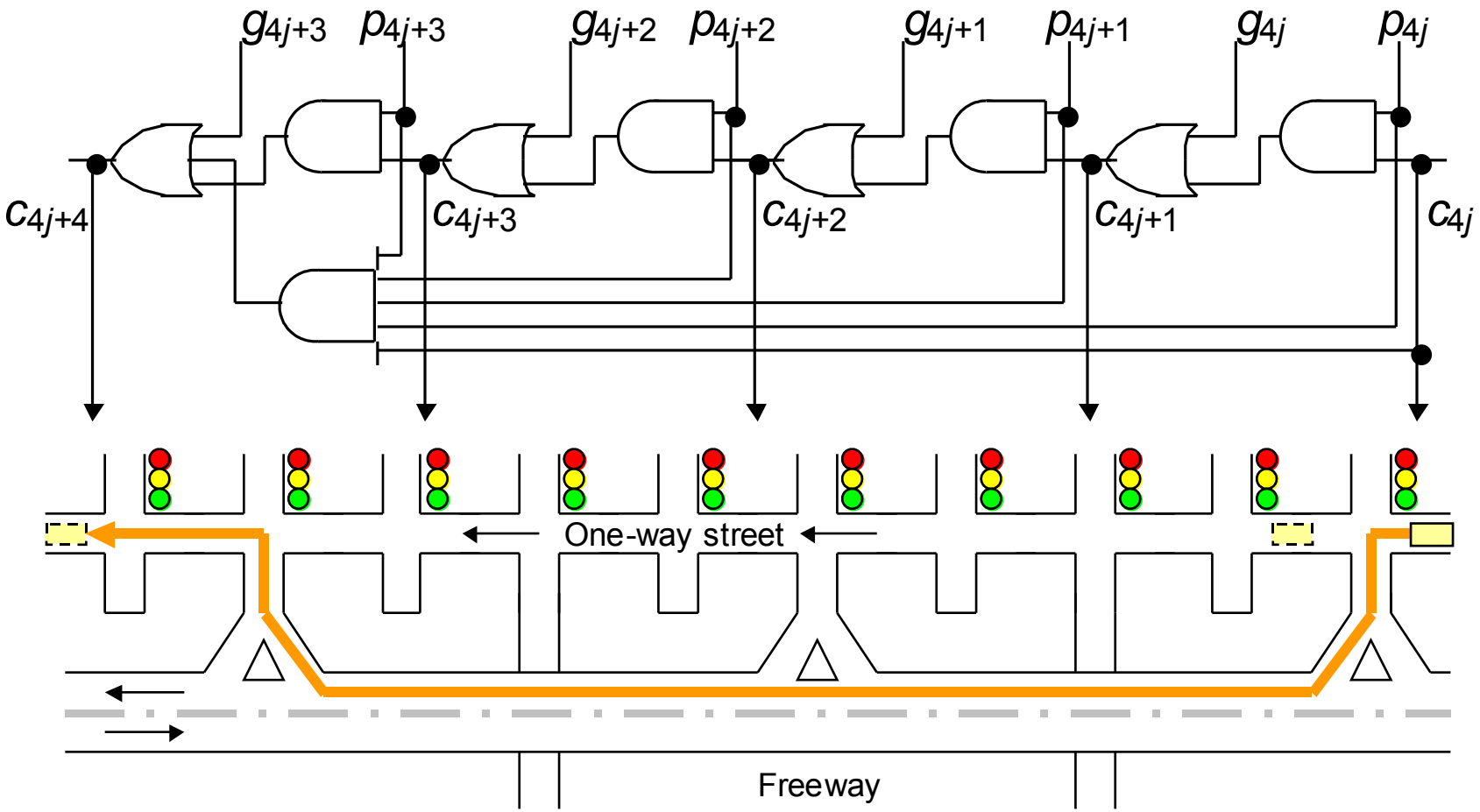


Fig. 7.1 Converting a 16-bit ripple-carry adder into a simple carry-skip adder with 4-bit skip blocks.

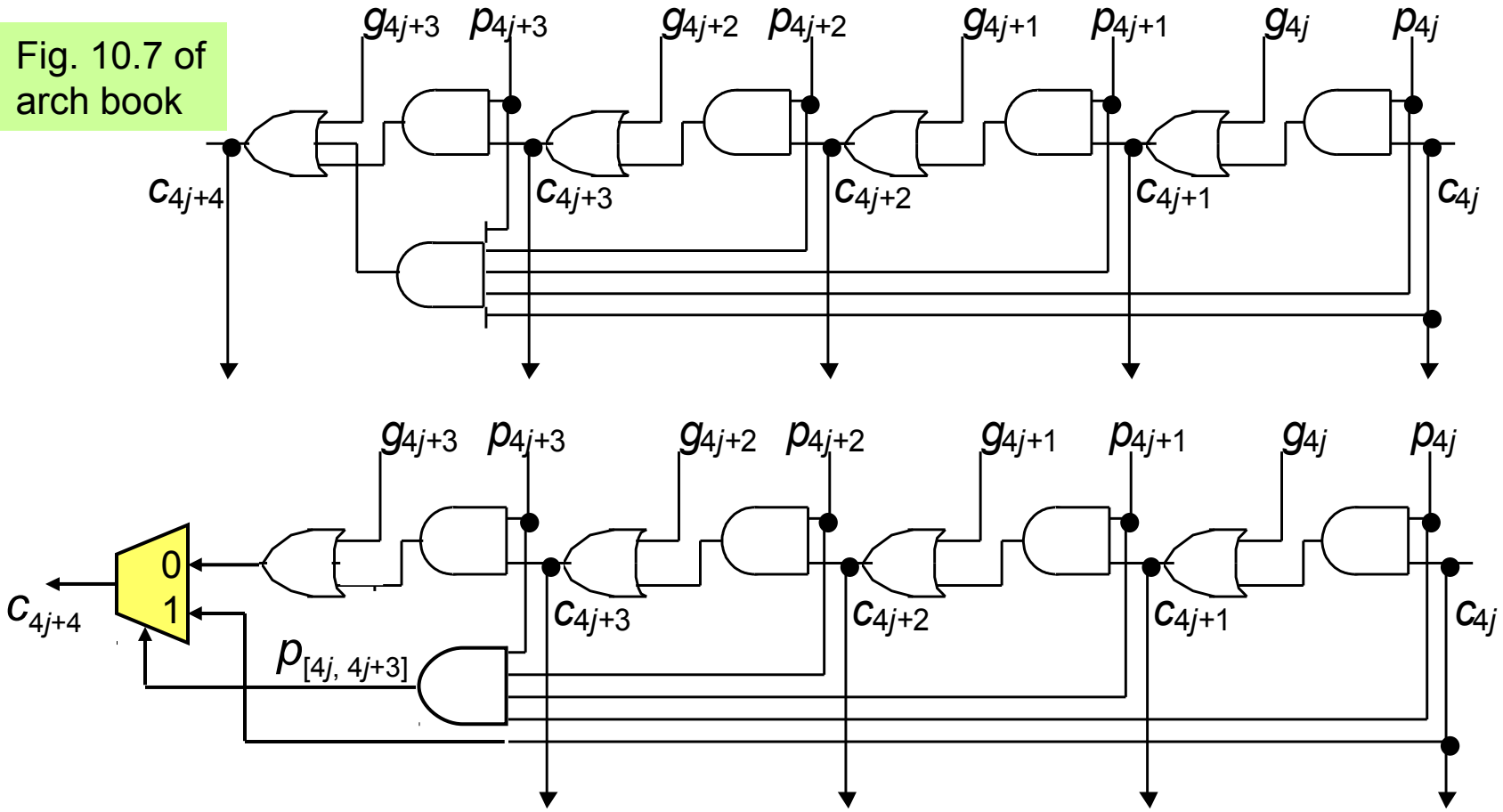
Another View of Carry-Skip Addition



Street/freeway analogy for carry-skip adder.

Mux-Based Skip Carry Logic

Fig. 10.7 of arch book



The carry-skip adder with "OR combining" works fine if we begin with a clean slate, where all signals are 0s at the outset; otherwise, it will run into problems, which do not exist in mux-based version

Carry-Skip Adder with Fixed Block Size

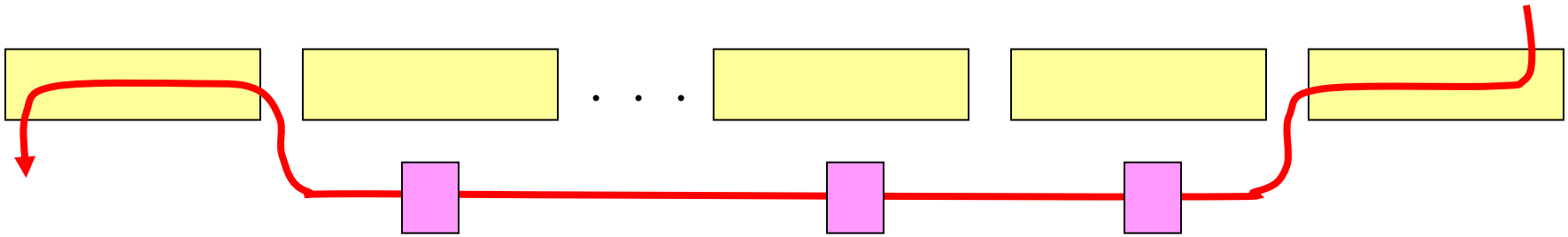
Block width b; k/b blocks to form a k-bit adder (assume b divides k)

$$T_{\text{fixed-skip-add}} = \underbrace{(b-1)}_{\text{in block 0}} + \underbrace{0.5}_{\text{OR gate}} + \underbrace{(k/b-2)}_{\text{skips}} + \underbrace{(b-1)}_{\text{in last block}}$$

$$\cong 2b + k/b - 3.5 \text{ stages}$$

$$dT/db = 2 - k/b^2 = 0 \quad \Rightarrow \quad b^{\text{opt}} = \sqrt{k/2}$$

$$T^{\text{opt}} = 2\sqrt{2k} - 3.5$$



Example: $k = 32$, $b^{\text{opt}} = 4$, $T^{\text{opt}} = 12.5$ stages
(contrast with 32 stages for a ripple-carry adder)

Fixed-Block-Size Carry-Skip Adder (1)

Notation & Assumptions

Adder size - k-bits

Fixed block size - b bits

Number of stages - t

Delay of skip logic = Delay of one stage of ripple-carry adder
= 1 delay unit

Latency of the carry-skip adder with fixed block width

$$\text{Latency}_{\text{fixed-carry-skip}} = \underbrace{(b-1)}_{\text{in block 0}} + \underbrace{0.5}_{\text{OR gate}} + \underbrace{\frac{k}{b} - 2}_{\text{skips}} + \underbrace{(b-1)}_{\text{in last block}}$$

$$= 2b + \frac{k}{b} - 3.5$$

Fixed-Block-Size Carry-Skip Adder (2)

Optimal fixed block size

$$\frac{dLatency_{fixed\text{-}carry\text{-}skip}}{db} = 2 - \frac{k}{b^2} = 0$$

$$b_{opt} = \sqrt{\frac{k}{2}} \qquad t_{opt} = \left\lceil \frac{k}{b_{opt}} \right\rceil = \sqrt{2k}$$

$$Latency^{opt}_{fixed\text{-}carry\text{-}skip} = 2\sqrt{\frac{k}{2}} + \frac{k}{\sqrt{\frac{k}{2}}} - 3.5 =$$

$$= \sqrt{2k} + \sqrt{2k} - 3.5 = \boxed{2\sqrt{2k} - 3.5}$$

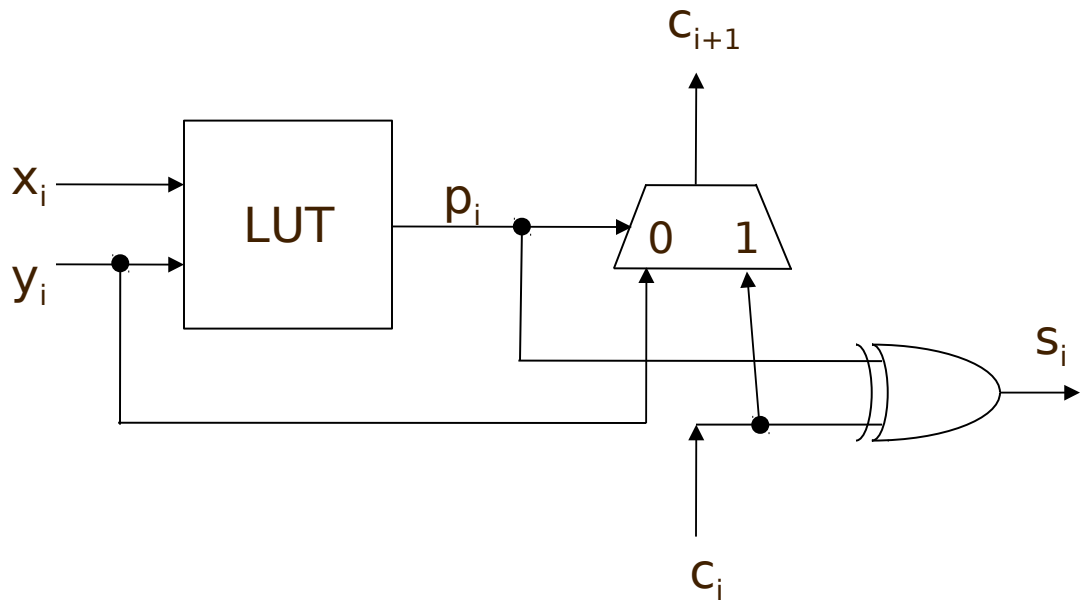
Fixed-Block-Size Carry-Skip Adder (3)

k	b_{opt}	t_{opt}	$Latency_{fixed\text{-}carry\text{-}skip}$	$Latency_{ripple\text{-}carry}$	$Latency_{look\text{-}ahead}$
32	4	8	12.5	32	6.5
128	8	16	28.5	128	8.5
16	2	8	8.5	16	4.5
	3	5	7.5		
64	5	13	18.5	64	6.5
	6	11	18.5		

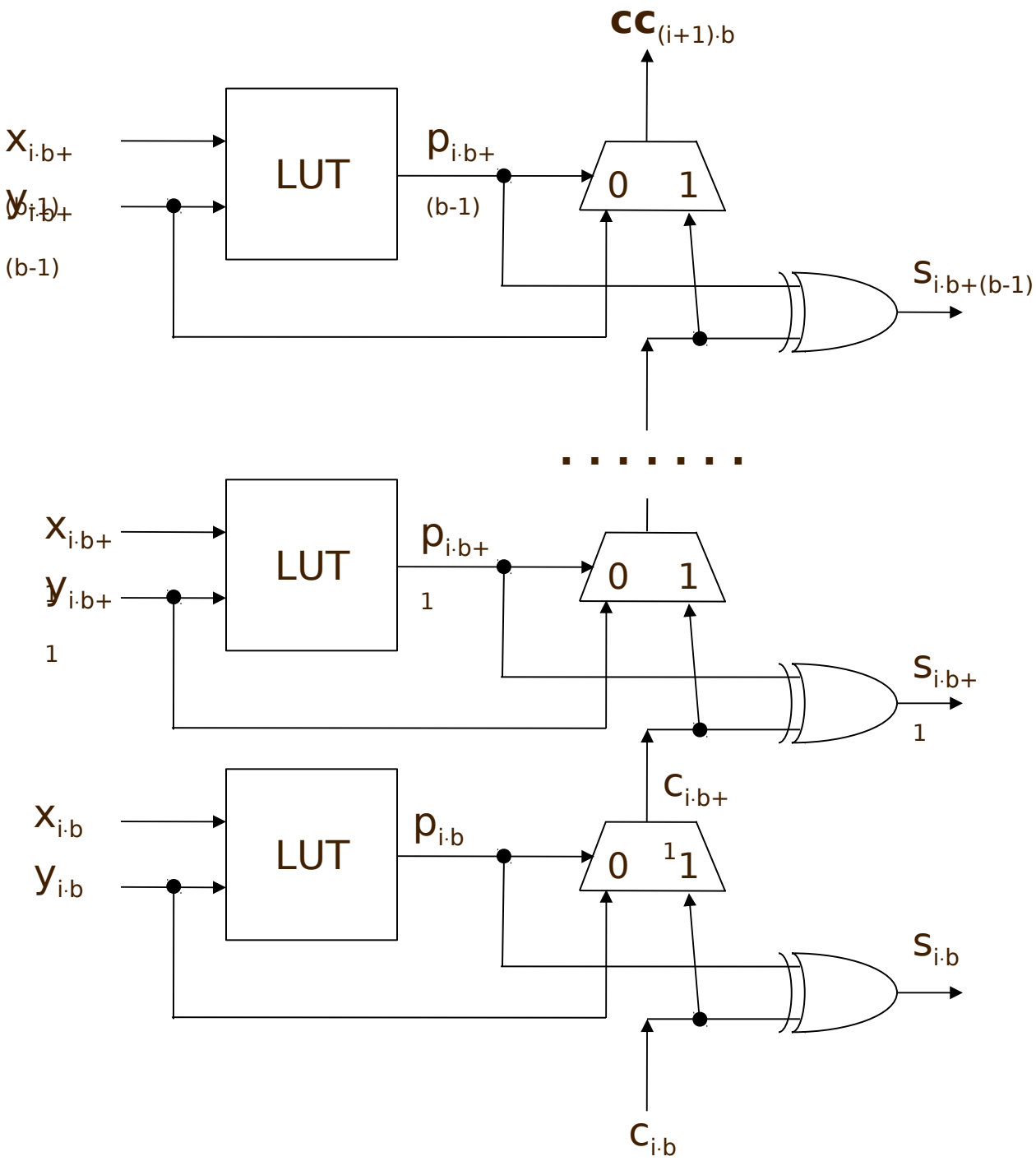
Carry-Chain & Carry-Skip Adders in Xilinx FPGAs

Basic Cell of a Carry-Chain Adder in Xilinx FPGAs

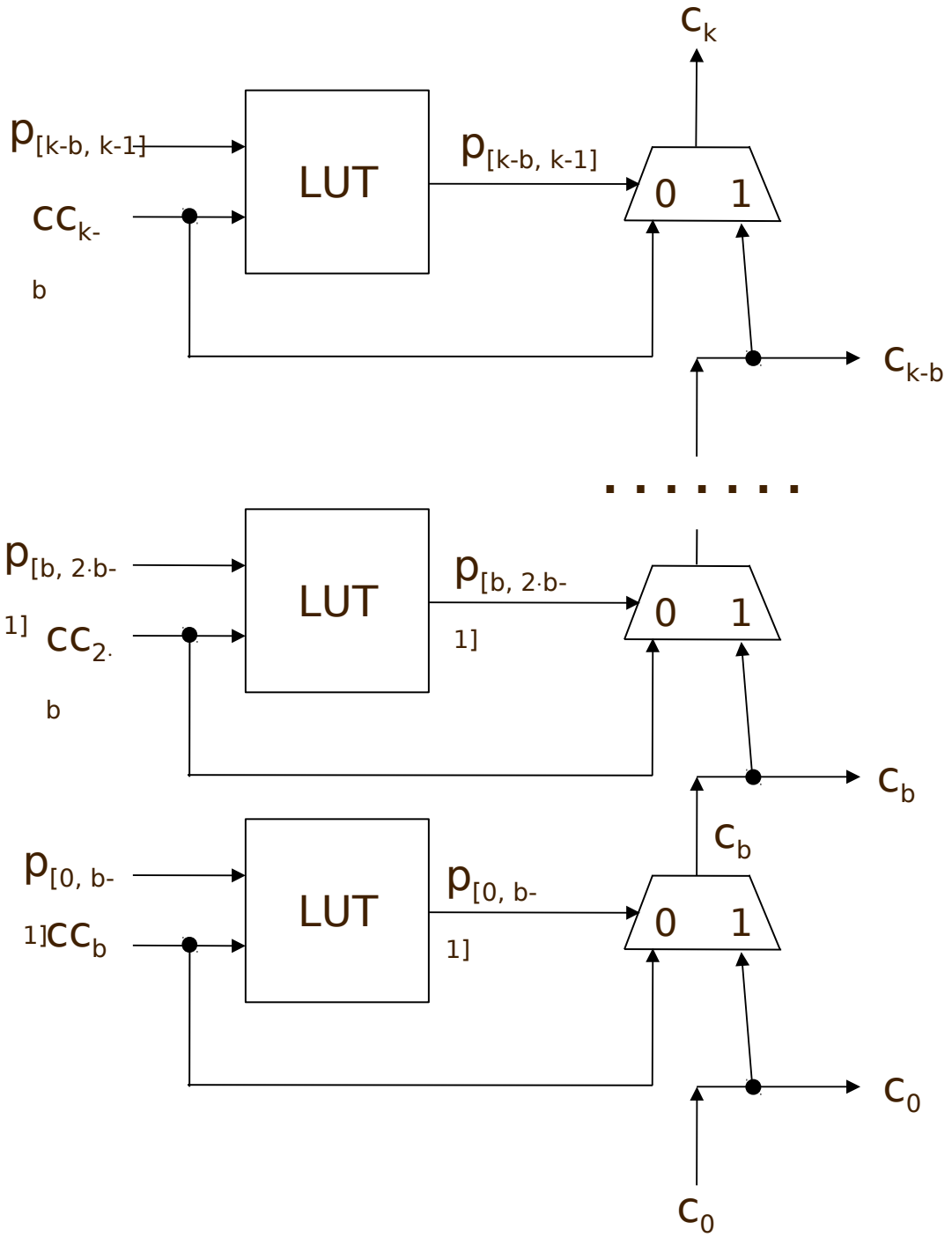
x_i	y_i	c_{i+1}
0	0	y_i
0	1	c_i
1	0	c_i
1	1	y_i



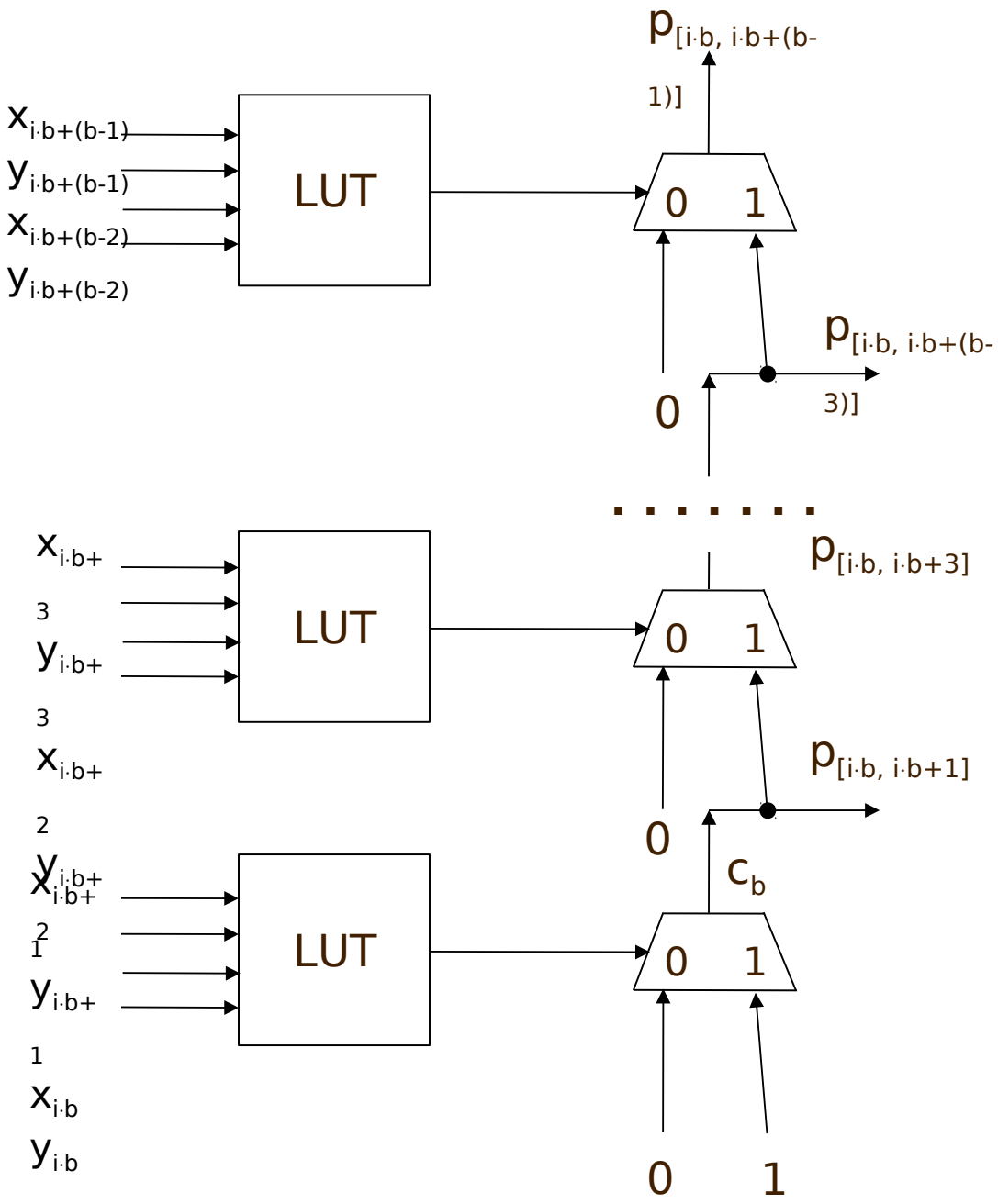
Carry-Skip Adder b-bit block



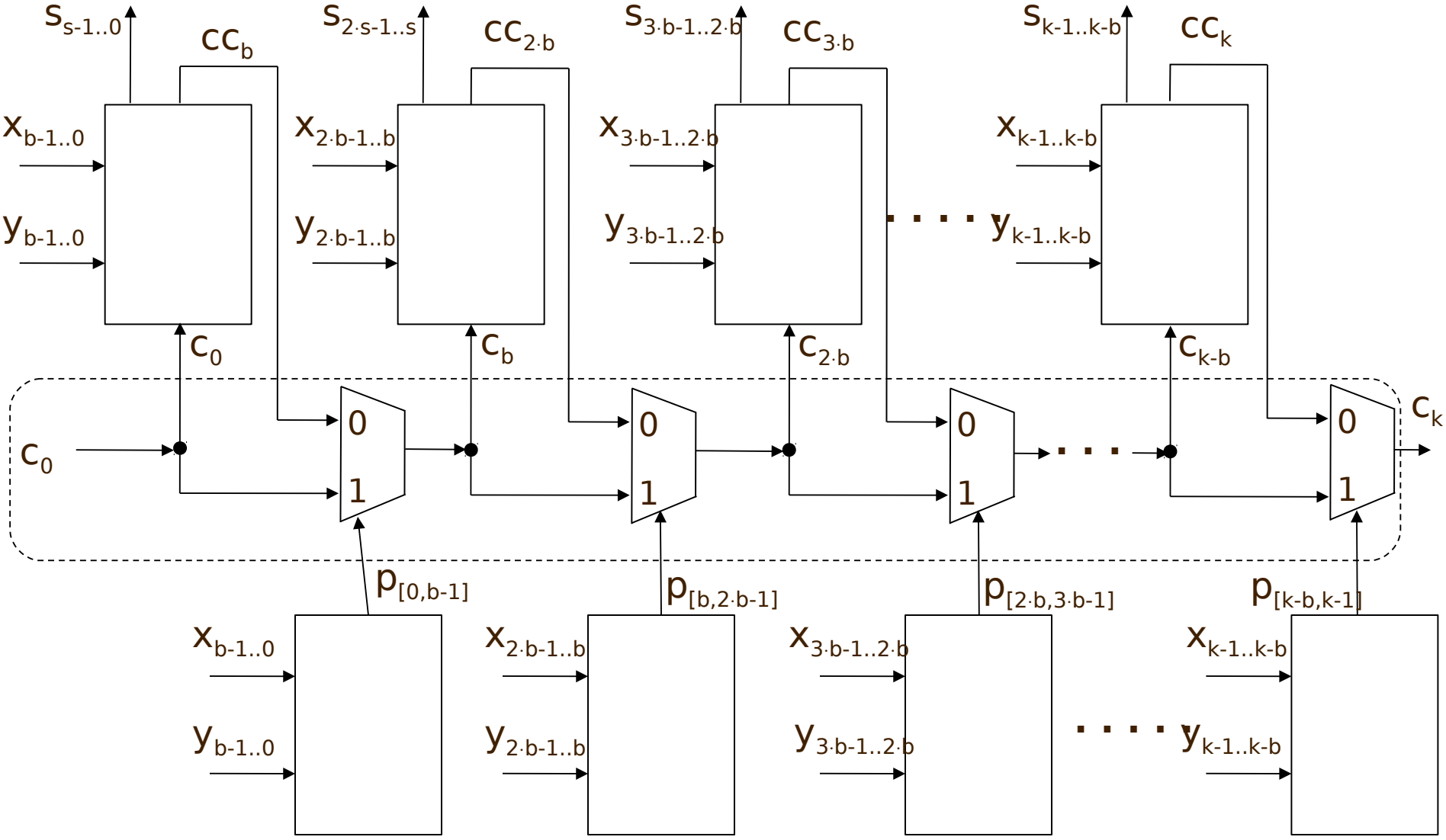
Carry-Skip Adder: Carry Skip Multiplexers



Carry-Skip Adder: Computation of the Block Propagate Signals



Complete Carry-Skip Adder



Carry-Skip Adders in Xilinx Spartan II FPGAs

by J-P. Deschamps, G. Bioul, G. Sutter

k	Delay in ns				Max. Frequency
	b=k	b=8	b=16	b=32	Increase
14	13		12		13
16	14		13		21
23	14		14		63
38	-		16	17	141
77	-		20	20	29
159	-		28	25	53

Carry-Skip Adders in Xilinx Spartan II FPGAs

by J-P. Deschamps, G. Bioul, G. Sutter

k	Area in CLB slices				Area Overhead		
	b=k	b=8	b=16	b=32	b=8	b=16	b=32
32	47		41	-	47%		28%
48	73		66	-	52%		38%
64	99		91	-	55%		42%
128	-		191	179	-		49%
256	-	391		375	-	53%	46%
512	-		791	767	-		54%

Carry-Skip Adders
Variable-Block-Size

Carry-Skip Adder with Variable-Width Blocks

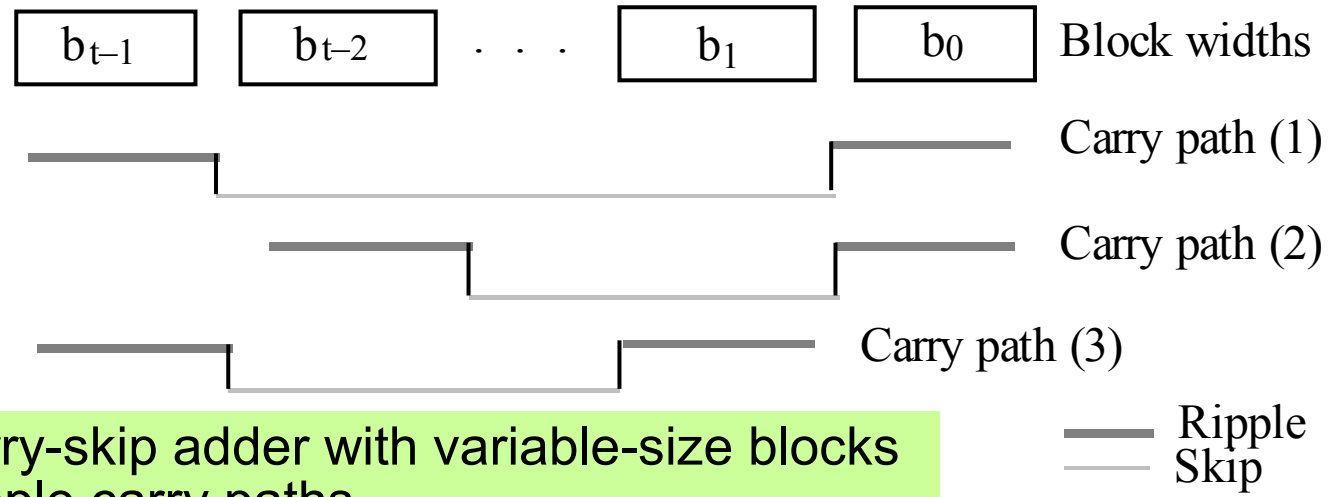


Fig. 7.2 Carry-skip adder with variable-size blocks and three sample carry paths.

The total number of bits in the t blocks is k:

$$2[b + (b + 1) + \ldots + (b + t/2 - 1)] = t(b + t/4 - 1/2) = k$$

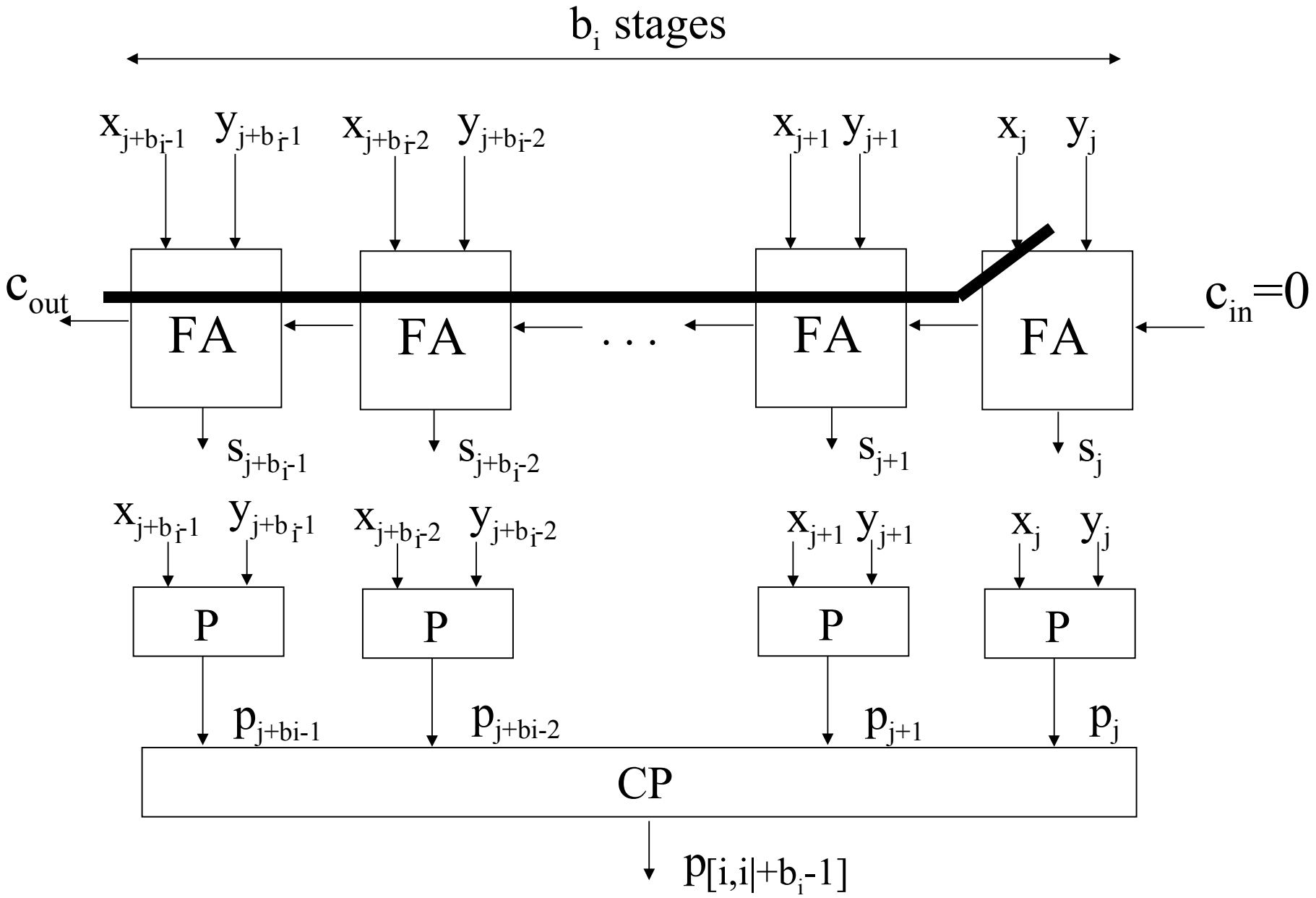
$$b = k/t - t/4 + 1/2$$

$$T_{\text{var-skip-add}} = 2(b - 1) + 0.5 + t - 2 = 2k/t + t/2 - 2.5$$

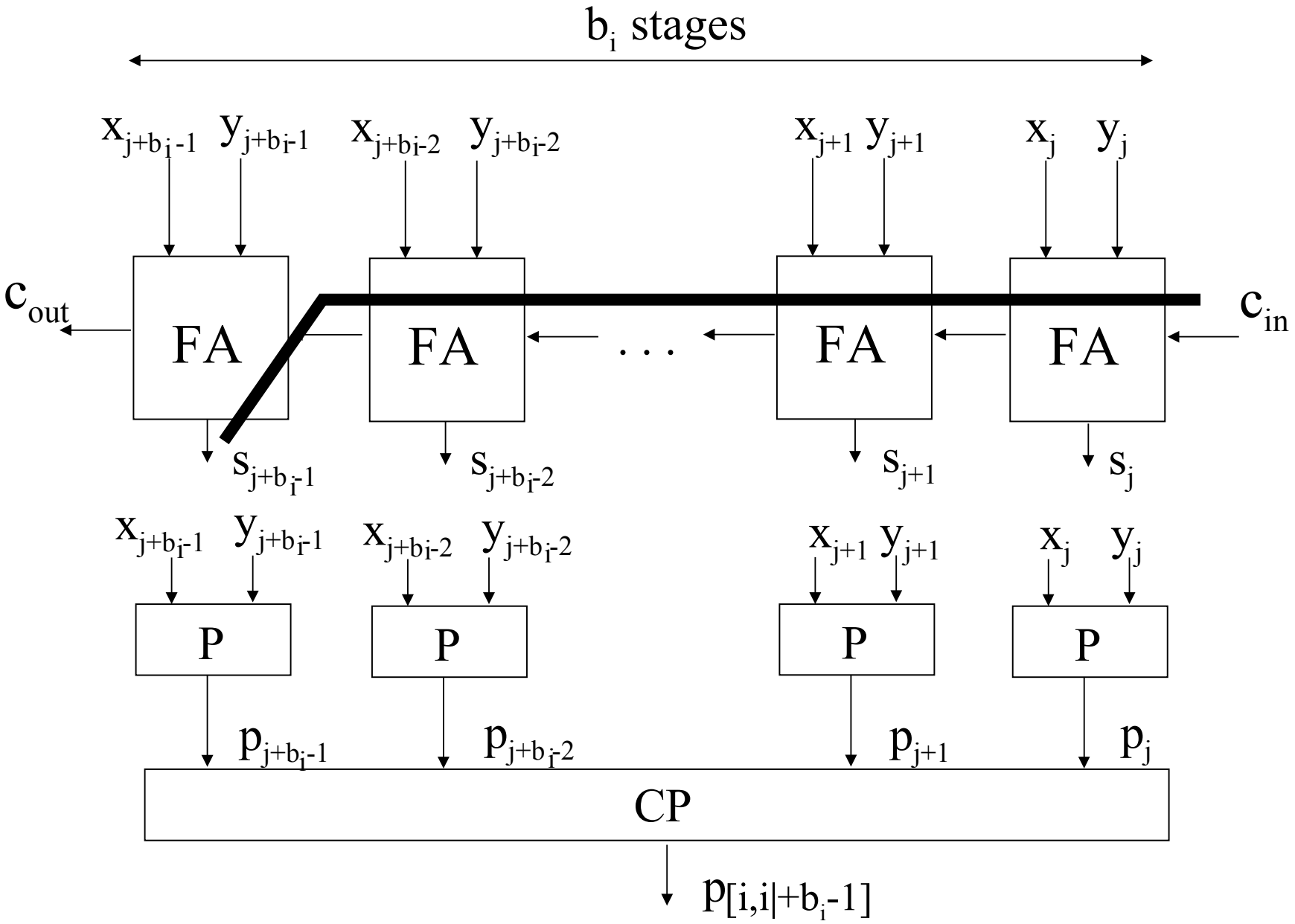
$$dT/db = -2k/t^2 + 1/2 = 0 \quad \Rightarrow \quad t^{\text{opt}} = 2\sqrt{k}$$

$$T^{\text{opt}} = 2\sqrt{k} - 2.5 \quad \text{(a factor of } \sqrt{2} \text{ smaller than for fixed-block)}$$

Most critical path to produce carry



Most critical path to assimilate carry



Variable-Block-Size Carry-Skip Adder (1)

Notation & Assumptions

Adder size - k-bits

Number of stages - t

Block size - variable

First and last block size - b bits

Delay of skip logic = Delay of one stage of ripple-carry adder
= 1 delay unit

Variable-Block-Size Carry-Skip Adder (2)

Optimum block sizes

$$b_{t-1} \quad b_{t-2} \quad b_{t-3} \quad \ldots \quad b_{t/2+1} \quad b_{t/2-1} \quad \ldots \quad b_2 \quad b_1 \quad b_0$$

$$b \quad b+1 \quad b+2 \quad \ldots \quad b+\frac{t}{2}-1 \quad b+\frac{t}{2}-1 \quad b+2 \quad b+1 \quad b$$

Total number of bits

$$k = 2\left[b + (b+1) + (b+2) + \ldots + \left(b + \frac{t}{2} + 1\right) \right] =$$

$$= t\left(b + \frac{t}{4} - \frac{1}{2} \right)$$

Variable-Block-Size Carry-Skip Adder (3)

Number of bits in the first and last block

$$b = \frac{k}{t} - \frac{t}{4} + \frac{1}{2}$$

Latency of the carry-skip adder with variable block width

$$\text{Latency}_{\text{fixed-carry-skip}} = \underbrace{(b-1)}_{\text{in block 0}} + \underbrace{0.5}_{\text{OR gate}} + \underbrace{t-2}_{\text{skips}} + \underbrace{(b-1)}_{\text{in last block}}$$

$$= 2b + t - 3.5 = 2\left(\frac{k}{t} - \frac{t}{4} + \frac{1}{2}\right) + t - 3.5 =$$

$$= \frac{2k}{t} + \frac{1}{2}t - 2.5$$

Variable-Block-Size Carry-Skip Adder (4)

Optimal number of blocks

$$\frac{dLatency_{variable\text{-}carry\text{-}skip}}{dt} = -\frac{2k}{t^2} + \frac{1}{2} = 0$$

$$\Downarrow$$

$$t_{opt} = \sqrt{4k} = 2\sqrt{k}$$

$$b_{opt} = \frac{k}{t_{opt}} - \frac{t_{opt}}{4} + \frac{1}{2} =$$

$$= \frac{k}{2\sqrt{k}} - \frac{2\sqrt{k}}{4} + \frac{1}{2} = \frac{1}{2}$$

$$\Rightarrow \quad b_{opt} = 1$$

Variable-Block-Size Carry-Skip Adder (5)

Optimal latency

$$\text{Latency}^{opt}_{\text{variable-carry-skip}} = \frac{2k}{t} + \frac{1}{2}t - 2.5 =$$

$$= \frac{2k}{2\sqrt{k}} + \frac{2\sqrt{k}}{2} - 2.5 =$$

$$= 2\sqrt{k} - 2.5$$

$$\text{Latency}^{opt}_{\text{variable-carry-skip}} \approx \frac{\text{Latency}^{opt}_{\text{fixed-carry-skip}}}{\sqrt{2}}$$