Lecture 5

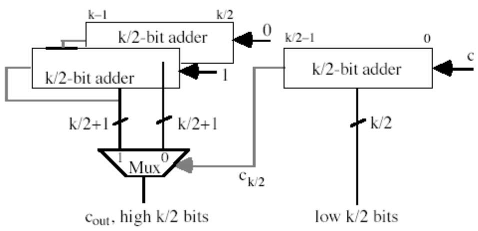
Conditional-Sum Adder

Hybrid Adders

Parallel Prefix Network Adders

Carry-Select Adders

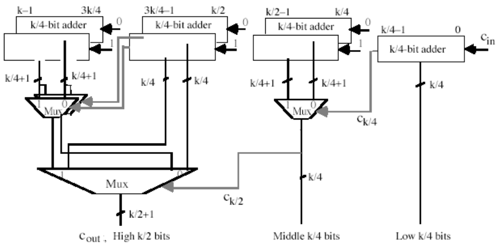
One-level k-bit Carry-Select Adder



One-level k-bit Carry-Select Adder
Cost & Latency

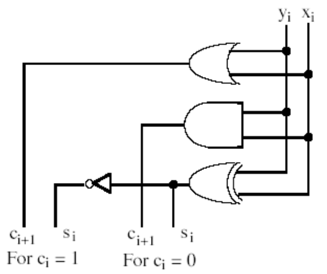
$$C_{select\text{-}add}(k) = 3C_{add}(k/2) + k/2 + 1$$
$$T_{select\text{-}add}(k) = T_{add}(k/2) + 1$$

Two-level k-bit Carry Select Adder



Conditional-Sum Adder

Top-level Block of Conditional-Sum Adder



Operation of a 16-bit Conditional-Sum Adder

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

Block sum and block carry-out; c_{out}

Conditional-Sum Adder

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

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

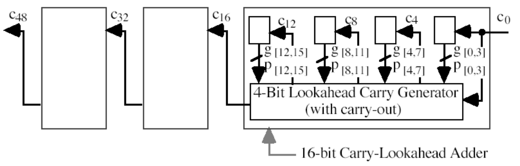
$$T(k) = T(k/2) + 1 = \log_2 k + T(1)$$

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

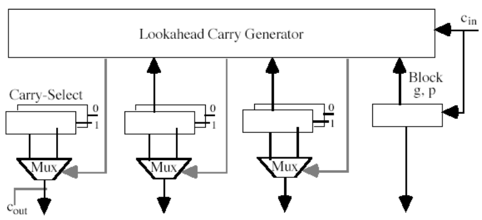
The term k + 2 in the first recurrence represents an upper bound on the number of single-bit 2-to-1 multiplexers needed for combining two k/2-bit adders into a k-bit adder

Hybrid Adders

A Hybrid Ripple-Carry/Carry-Lookahead Adder



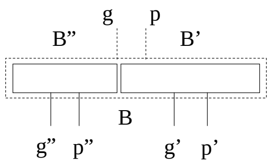
A Hybrid Carry-Lookahead/Carry-Select Adder



Parallel Prefix Network Adders

Parallel Prefix Network Adders

Basic component - Carry operator (1)

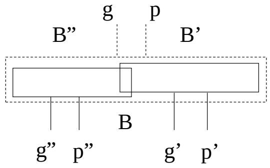


g = g” + g’p”
p = p’p”

(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’) ¢ (g”, p”) = (g” + g’p”, p’p”)

Properties of the carry operator ¢

Associative

$$[(g_1, p_1) ¢ (g_2, p_2)] ¢ (g_3, p_3) = (g_1, p_1) ¢ [(g_2, p_2) ¢ (g_3, p_3)]$$

Not commutative

$$(g_1, p_1) ¢ (g_2, p_2) \neq (g_2, p_2) ¢ (g_1, p_1)$$

Parallel Prefix Network Adders

Major concept

Given:

(g_0, p_0) (g_1, p_1) (g_2, p_2) …. (g_{k-1}, p_{k-1})

Find:

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

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

Parallel Prefix Network Adders

Similar problem

Parallel Prefix Sum Problem

Given:

x_0 x_1 x_2 … x_{k-1}

Find:

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

Parallel Prefix Adder Problem

Given:

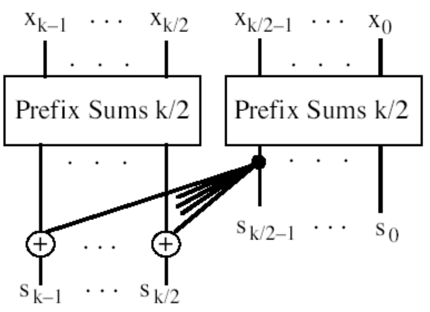
x_0 x_1 x_2 … x_{k-1}

Find:

x_0 $x_0 ¢ x_1$ $x_0 ¢ x_1 ¢ x_2$ … $x_0 ¢ x_1 ¢ x_2 ¢ \ldots ¢ x_{k-1}$

where $x_i = (g_i, p_i)$

Parallel Prefix Sums Network I



Parallel Prefix Sums Network I
Cost Analysis

Cost = **C(k)** = 2 C(k/2) + k/2 =
= 2 [2C(k/4) + k/4] + k/2 = 4 C(k/4) + k/2 + k/2 =
= …. =
= $2^{\log_2 k-1}$C(2) + k/2 ($\log_2$k-1) =
= **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 = **32** = (16/2) $\log_2$ 16

Parallel Prefix Sums Network I
Delay Analysis

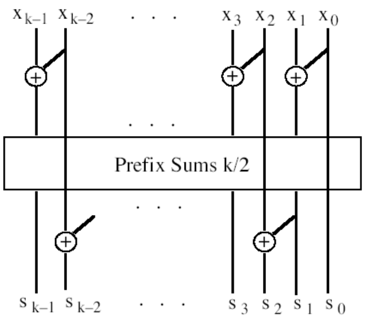
Delay = **D(k)** = D(k/2) + 1 =
= [D(k/4) + 1] + 1 = D(k/4) + 1 + 1 =
= …. =
= $\log_2$k

| 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



Parallel Prefix Sums Network II
Cost Analysis

Cost = **C(k)** = C(k/2) + k-1 =

= [C(k/4) + k/2-1] + k-1 = C(k/4) + 3k/2 - 2 =

= …. =

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

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

C(2) = 1

Example:

C(16) = C(8) + 16-1 = [C(4) + 8-1] + 16-1 =

= C(2) + 4-1 + 24-2 = 1 + 28 - 3 = **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 =

= …. =

= $\mathbf{2 \log_2 k - 1}$

D(2) = 1

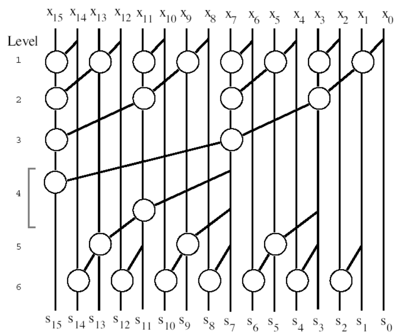
Example:

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

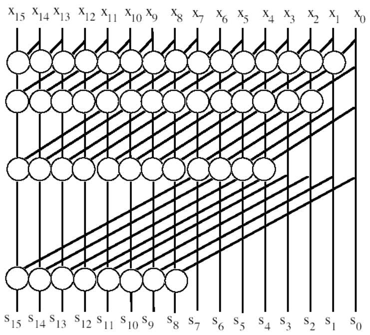
= D(4) + 4 = [D(2) + 2] + 4 =

= **7** = $2 \log_2 16 - 1$

Brent-Kung parallel prefix graph for 16 inputs



Kogge-Stone parallel prefix graph for 16 inputs

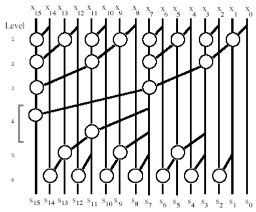


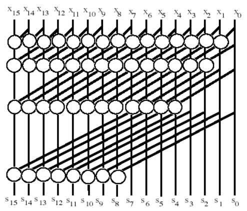
Parallel Prefix Network Adders

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-Cost Trade-off





B-K: Six levels, 26 cells   K-S: Four levels, 49 cells

A Hybrid Brent-Kung/Kogge-Stone parallel prefix graph for 16 inputs

Hybrid: Five levels, 32 cells