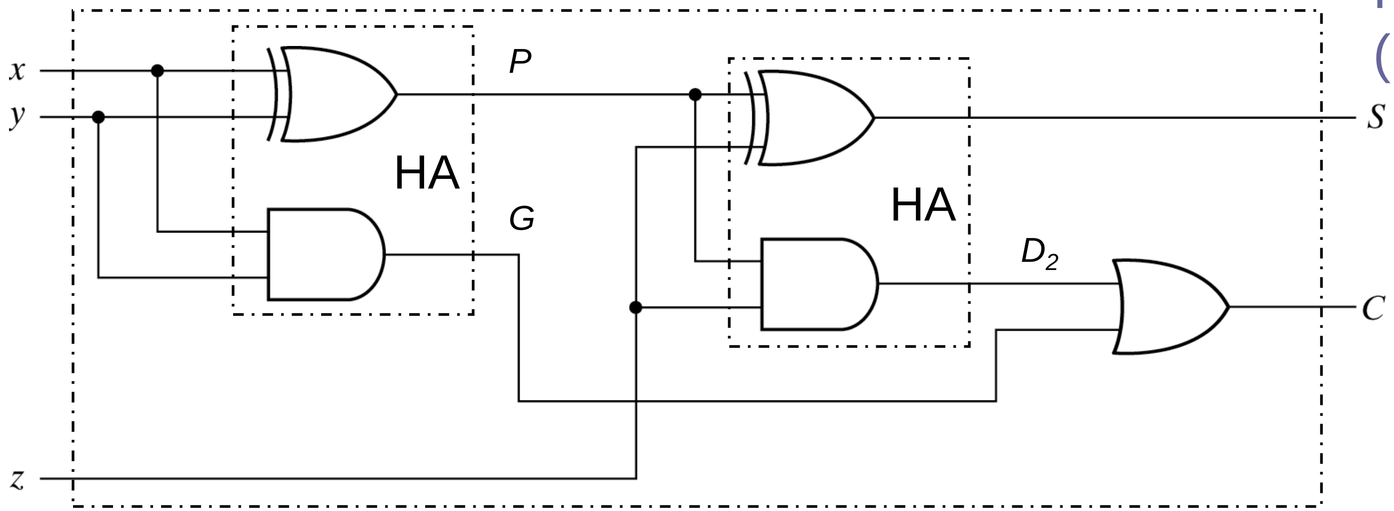
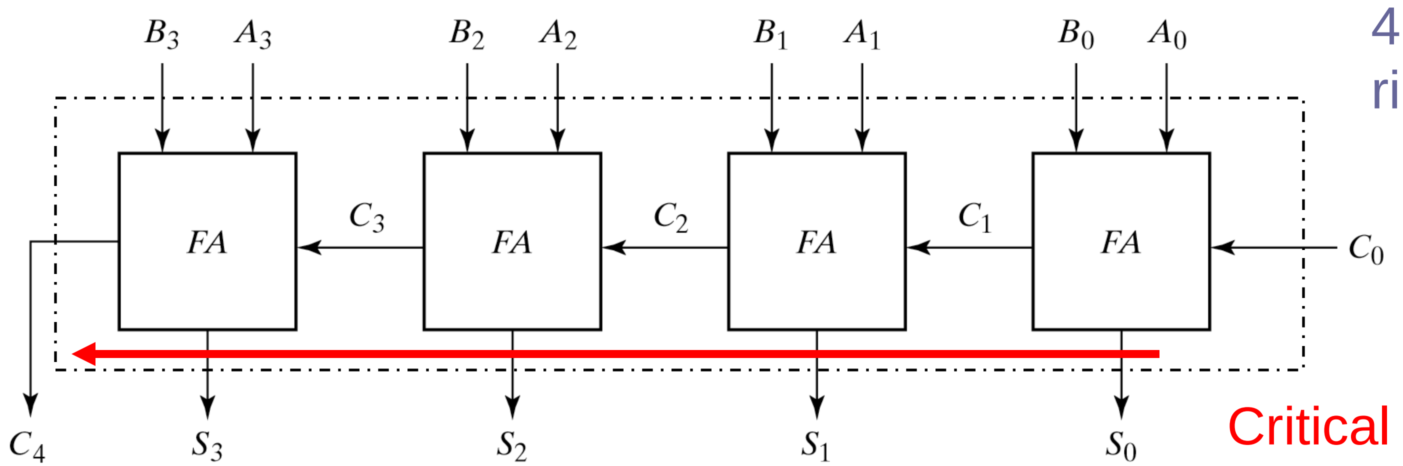
EECS 3201: Digital Logic Design Lecture 17

Ihab Amer, PhD, SMIEEE, P.Eng.

Flash Back!



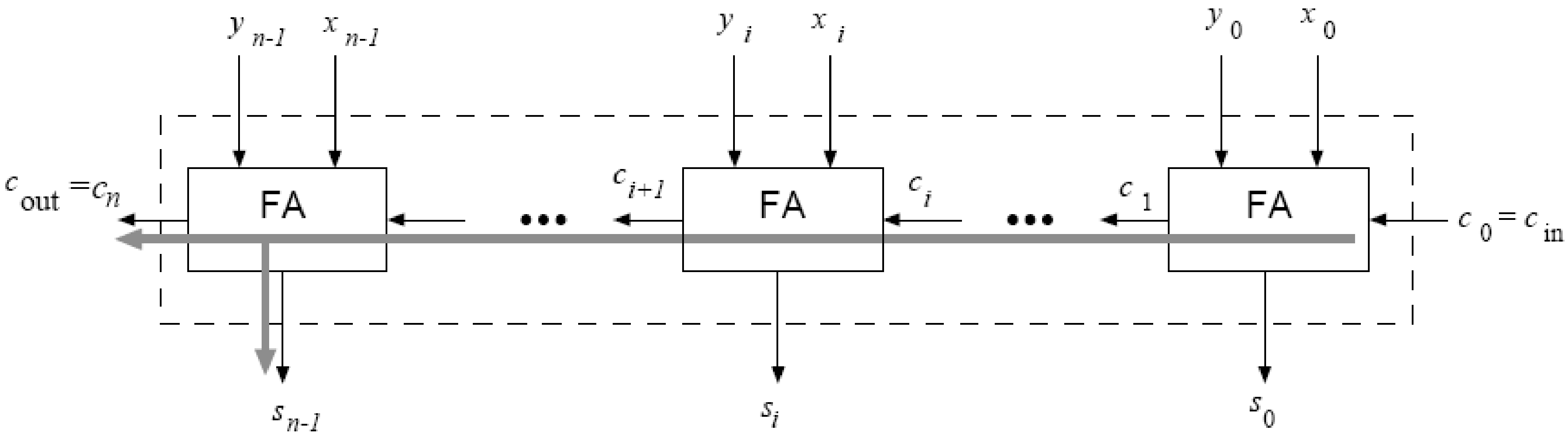
Full Adder (FA)



4-bit binary ripple-adder

Critical Path

Basic Carry-Ripple Adder (CRA)



Carry-ripple adder.

$$T_{CRA} = (n-1)t_c + \max(t_c, t_s)$$

About Carries

- At position i of the addition, consider the relation between c_{i+1} and c_i. There are three mutually exclusive cases. Each of those cases rely on x_i and y_i only, and hence, can be performed in parallel (for all i)

Carry-Out Cases

Case	x_i	y_i	$x_i + y_i$	c_{i+1}	Comment
1	0	0	0	0	kill (stop) carry-in
2	0	1	1	c_i	propagate carry-in
	1	0	1	c_i	propagate carry-in
3	1	1	2	1	generate carry-out

Case 1 (Kill): $k_i = x_i' y_i' = (x_i + y_i)'$
Case 2 (Propagate): $p_i = x_i \oplus y_i$
Case 3 (Generate): $g_i = x_i y_i$
Then

$$c_{i+1} = g_i + p_i c_i = x_i y_i + (x_i \oplus y_i) c_i$$

Alternative (simpler) expression:

$$c_{i+1} = g_i + a_i c_i$$

Since $a_i = k_i'$ we call it "alive"

Carry Chains

Two types:

1-carry chain consisting of carry=1
0-carry chain consisting of carry=0

i	9	8	7	6	5	4	3	2	1	0
x_i	1	0	1	0	1	1	1	1	0	0
y_i	0	0	0	1	0	1	0	0	1	0
	p	k	p	p	p	g	p	p	p	k
	a		a	a	a	a	a	a	a	
c_{i+1}	0 $\leftarrow$	0	1 $\leftarrow$	1 $\leftarrow$	1 $\leftarrow$	1	0 $\leftarrow$	0 $\leftarrow$	0 $\leftarrow$	0

Generalization to Group of Bits

$$c_{j+1} = g_{(j,i)} + p_{(j,i)}c_i = g_{(j,i)} + a_{(j,i)}c_i$$

or, for $i = 0$

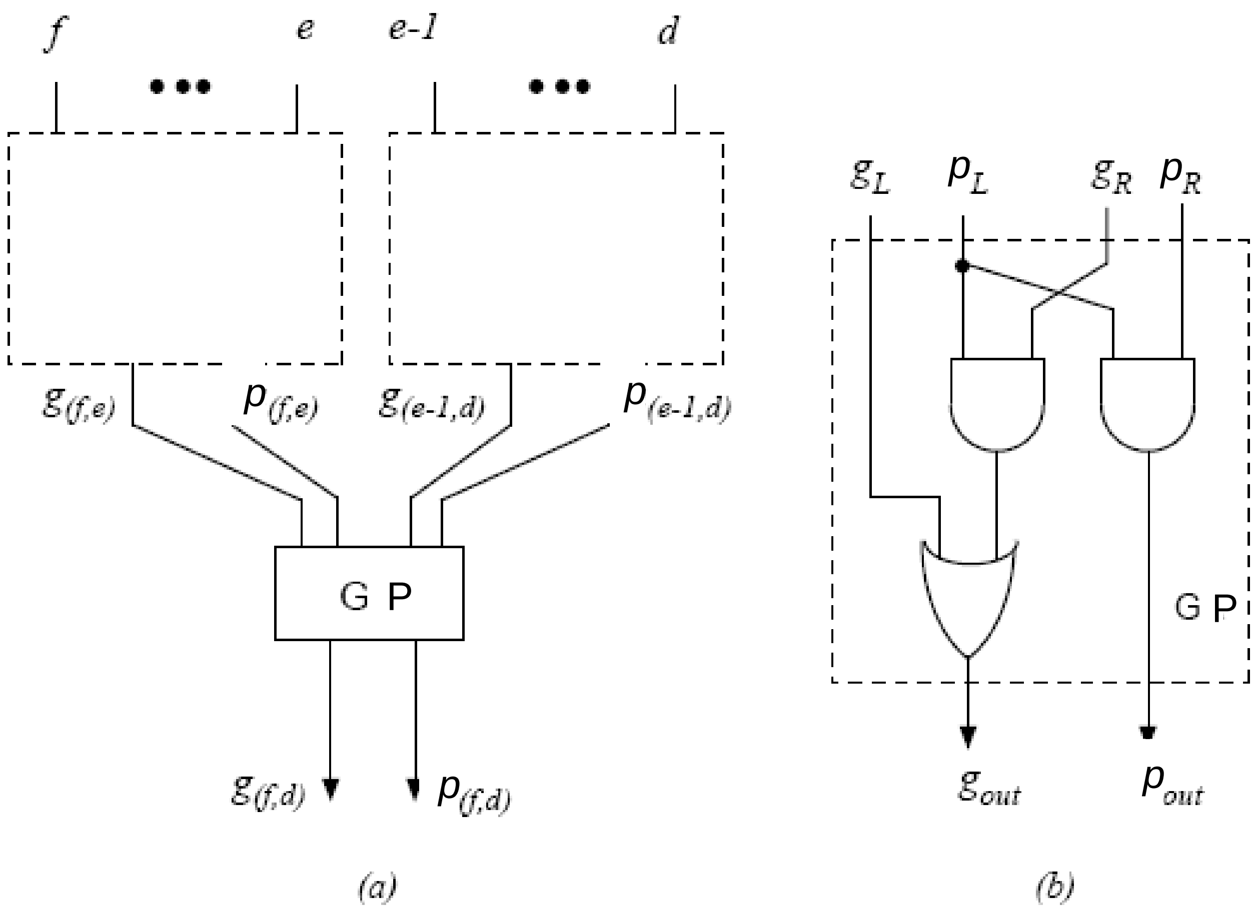
$$c_{j+1} = g_{(j,0)} + p_{(j,0)}c_0 = g_{(j,0)} + a_{(j,0)}c_0$$

Recursive combining of subranges of variables:

$$g_{(f,d)} = g_{(f,e)} + p_{(f,e)}g_{(e-1,d)} = g_{(f,e)} + a_{(f,e)}g_{(e-1,d)}$$

$$p_{(f,d)} = p_{(f,e)}p_{(e-1,d)}$$

Generalization (Cont'd)



Computing $(g_{(f,d)}, p_{(f,d)})$.

Example

- Obtain bit #13 of the sum of the following 16-bit operands ($c_{in} = 0$)

x = 0110|0010|1100|0011

y = 1011|1101|0001|1110

$p_{(12,12)} = 1$, $p_{(11,8)} = 1$, $k_{(7,4)} = 1$, $g_{(3,0)} = 1$

→ $p_{(12,8)} = 1$, $k_{(7,0)} = k_{(7,4)} + p_{(7,4)}k_{(3,0)} = 1$

→ $k_{(12,0)} = 1$

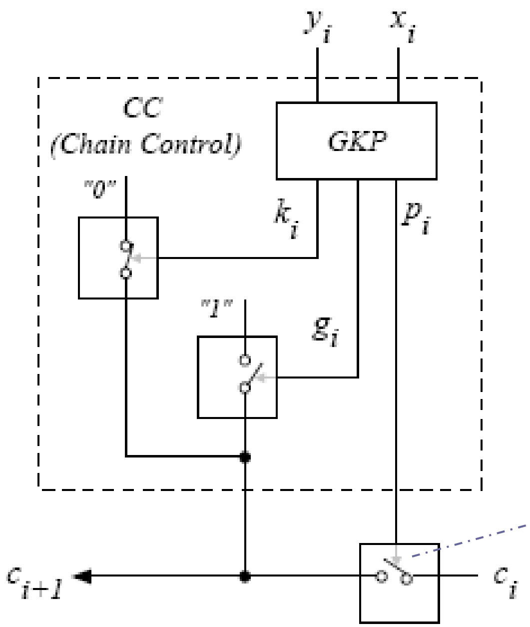
→ $c_{13} = g_{(12,0)} + p_{(12,0)}c_{in} = 0$

→ $s_{13} = x_{13} \oplus y_{13} \oplus c_{13} = 0$

Fast Two-Operand Addition

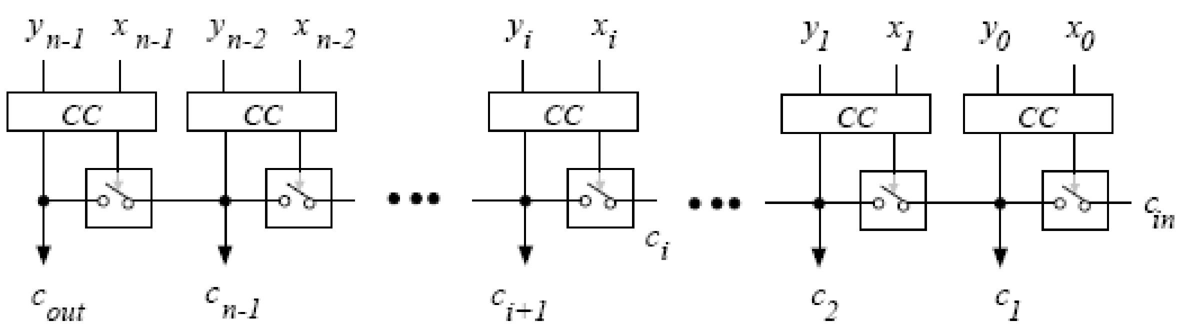
- Conventional Number System (Carry Propagate Adders – CPA)
  - Switched Carry-Ripple Adder
  - Carry-Skip Adder
  - Carry Lookahead Adder
  - Prefix Adder
  - Carry-Select Adder and Conditional-Sum Adder
  - Variable-Time Adder
- Redundant Number System (Totally Parallel Adders – TPA); Adders with limited carry propagation
  - Carry-Save Adder
  - Signed Digit Adder

Switched Carry-Ripple (Manchester) Adder

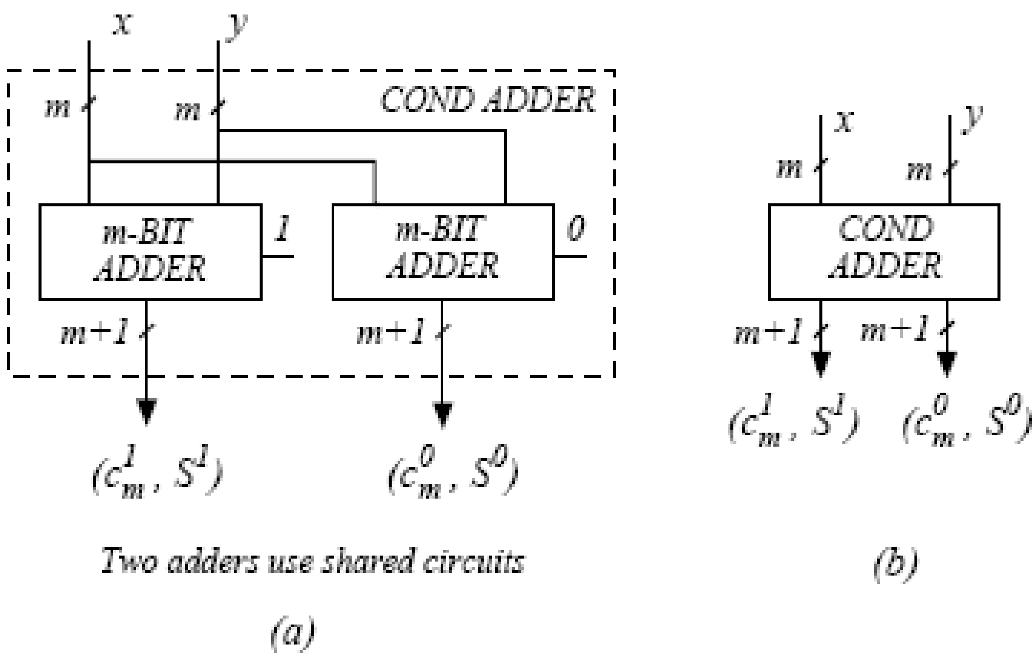


$$T_{SRA} = t_{sw} + (n-1)t_p + (n/m)t_{buf} + t_s$$

These maybe special transistors

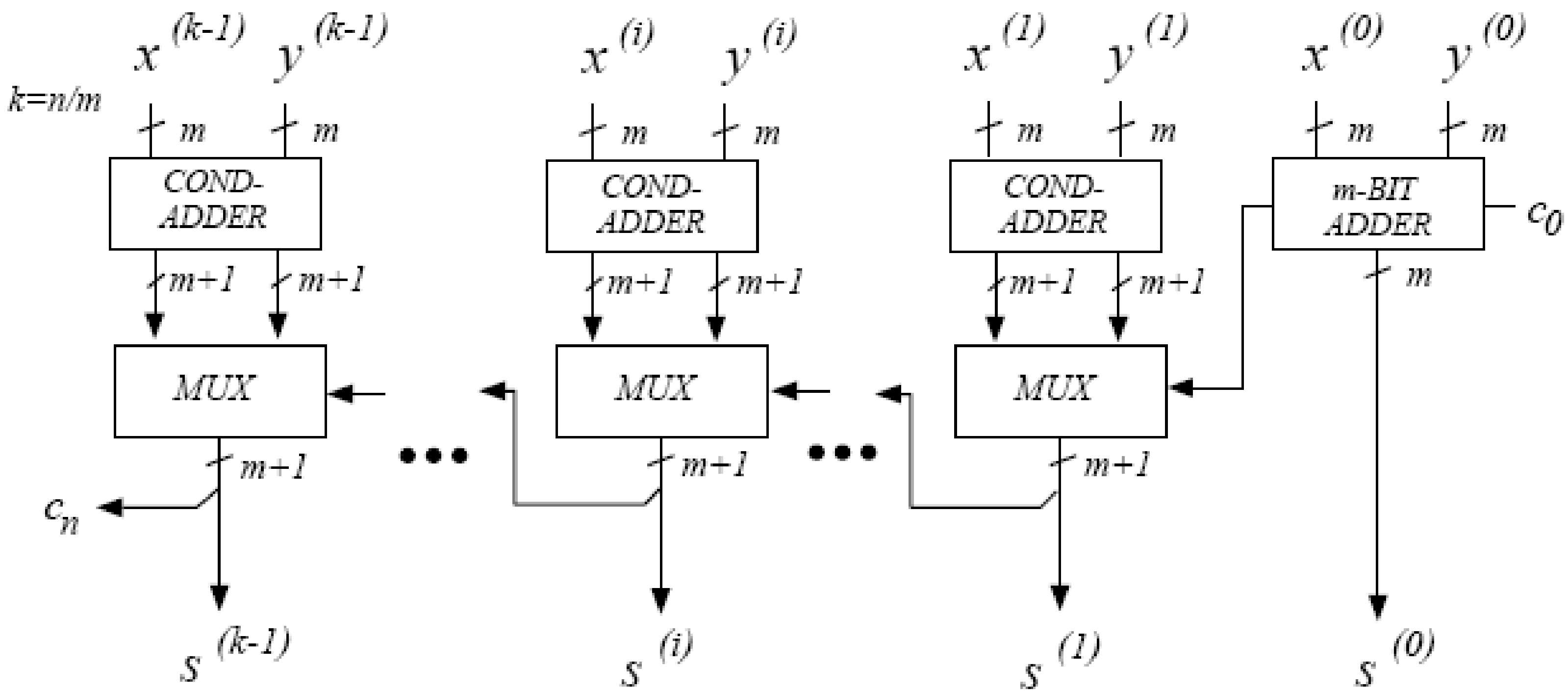


Conditional Adder



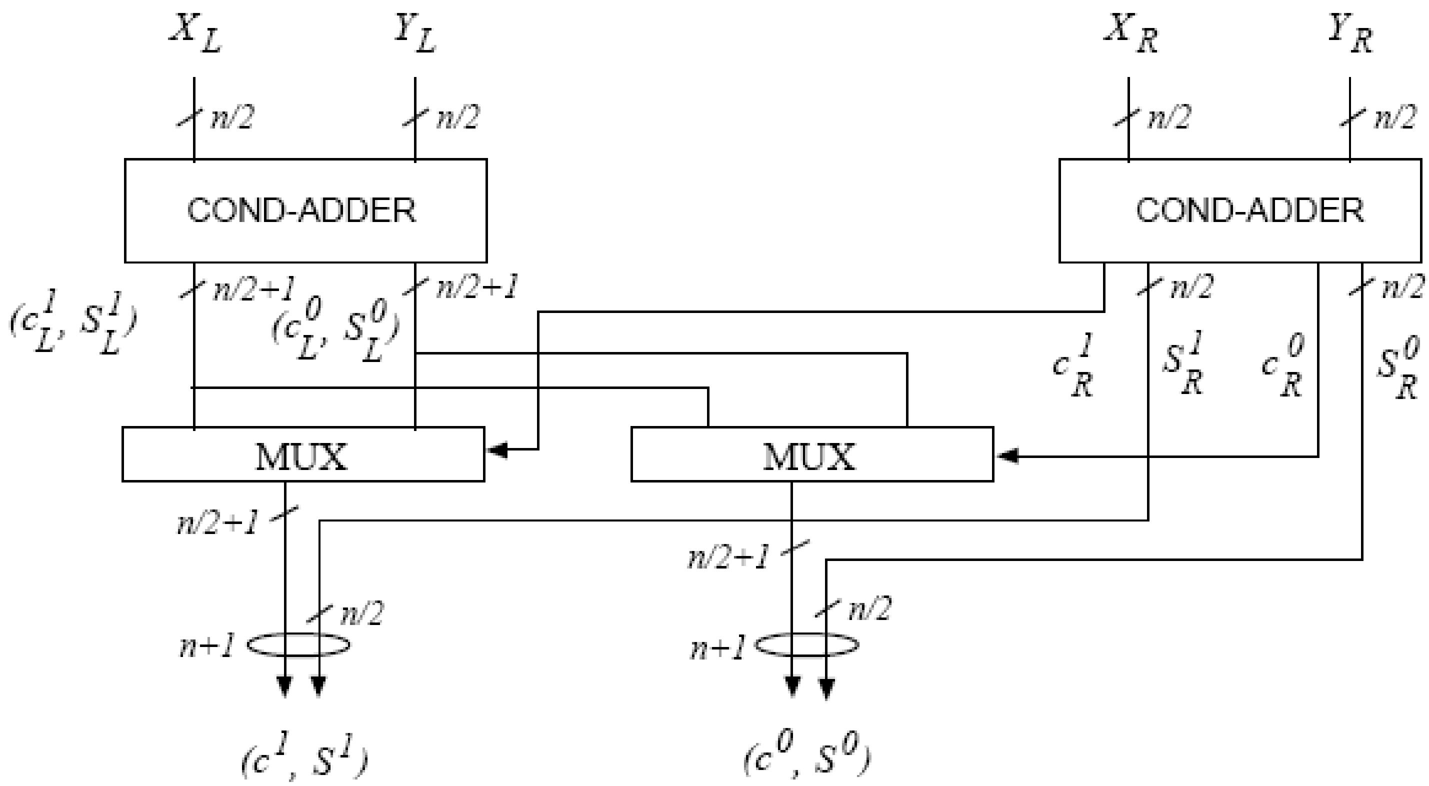
(a) Obtaining conditional outputs. (b) Combined conditi onal adder.

Carry-Select Adder



$$T_{CSEL} = t_{add,m} + (\frac{n}{m} - 1)t_{mux}$$

Conditional-Sum Adder



$$T_{cond-sum} = t_{add,m} + (log_2(n/m))t_{mux}$$

Numerical Example

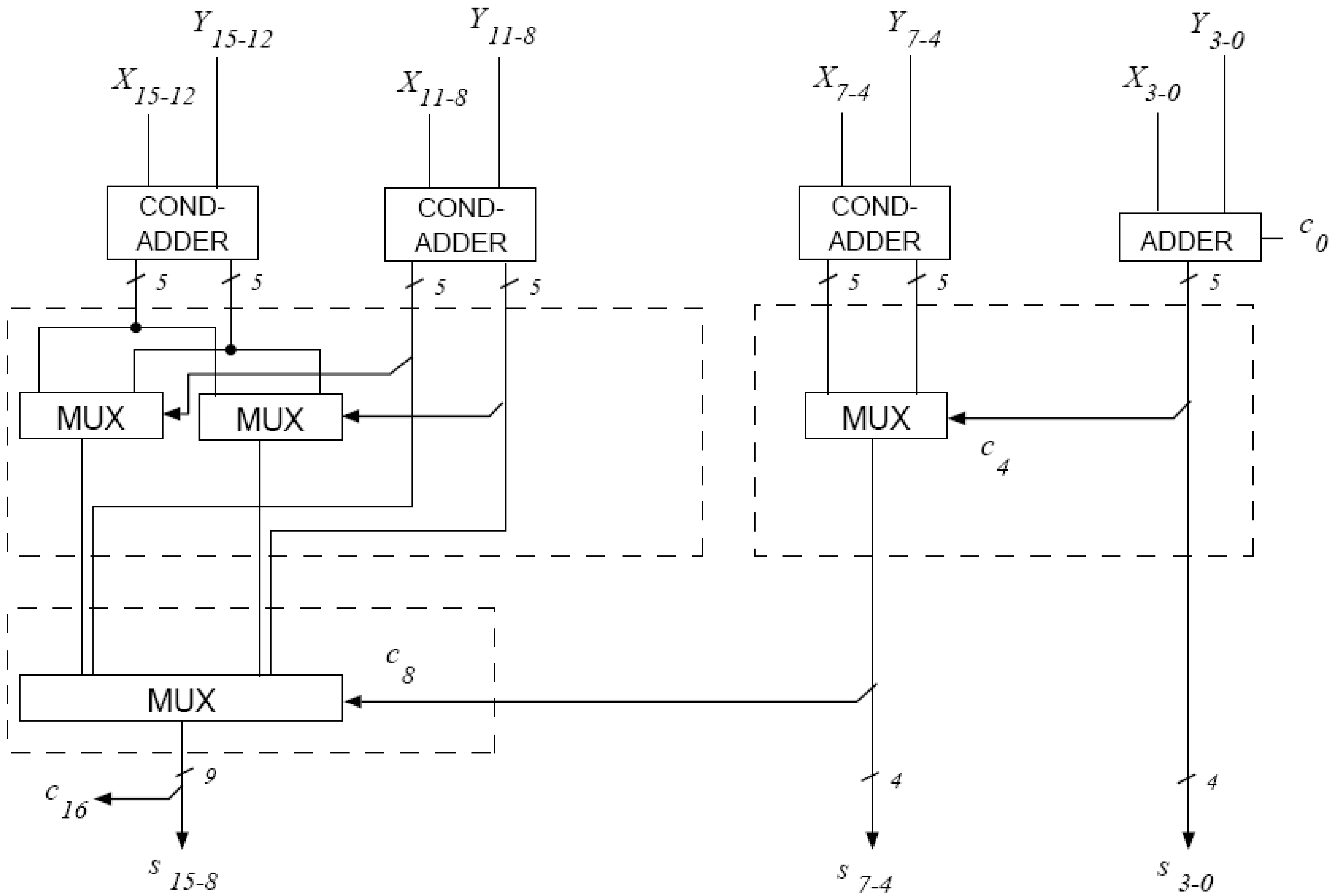
$$
\begin{aligned}
X_L &= 0011 & X_R &= 0111\\
Y_L &= 1010 & Y_R &= 1001\\
(c_L^0, S_L^0) &= (0{,}1101) & (c_R^0, S_R^0) &= (1{,}0000)\\
(c_L^1, S_L^1) &= (0{,}1110) & (c_R^1, S_R^1) &= (1{,}0001)
\end{aligned}
$$

Combining we obtain

$$(c^0, S^0) = (0, 11100000)$$

$$(c^1, S^1) = (0, 11100001)$$

Example: 16-bit Conditional-Sum Adder (m = 4)



Example

- Conditional-sum for eight bits with *m = 2*

X	01	01	01	11
Y	10	10	11	11
S^0	11	11	00	10
c^0	0	0	1	1
S^1	00	00	01	11
c^1	1	1	1	1
S^0	11	11	01	10
c^0	0		1	
S^1	00	00	01	11
c^1	1		1	
S^0	00	00	01	10
c^0	1			
S^1	00	00	01	11
c^1	1			

References

- Milos D. Ercegovac and Tomas Lang, “Digital Arithmetic”, Morgan Kaufmann Publishers, an imprint of Elsevier Science, 2004