MULTI-OPERAND ADDITION

- Bit-arrays for unsigned and signed operands
  - simplification of sign extension
- Reduction by rows and by columns
  - [p:2] modules and [p:2] adders for reduction by rows
  - (p:q] counters and multicolumn counters for reduction by columns
- Sequential implementation
- Combinational implementation
  - Reduction by rows: arrays of adders (linear arrays, adder trees)
  - Reduction by columns: (p:q] counters
  - systematic design method for reduction by columns with (3:2] and (2:2] counters
- Pipelined adder arrays
- Partially combinational implementation

BIT ARRAYS FOR UNSIGNED AND SIGNED OPERANDS

$$
\begin{array}{l}
a_0\, a_0\, a_0\, a_0\,.\,a_1 a_2 \,\ldots\, a_n \\
b_0\, b_0\, b_0\, b_0\,.\,b_1 b_2 \,\ldots\, b_n \\
c_0\, c_0\, c_0\, c_0\,.\,c_1 c_2 \,\ldots\, c_n \\
d_0\, d_0\, d_0\, d_0\,.\,d_1 d_2 \,\ldots\, d_n \\
\underbrace{e_0\, e_0\, e_0}_{\text{sign extension}}\, e_0\,.\,e_1 e_2 \,\ldots\, e_n
\end{array}
$$

Figure 3.1: SIGN-EXTENDED ARRAY FOR $m = 5$.

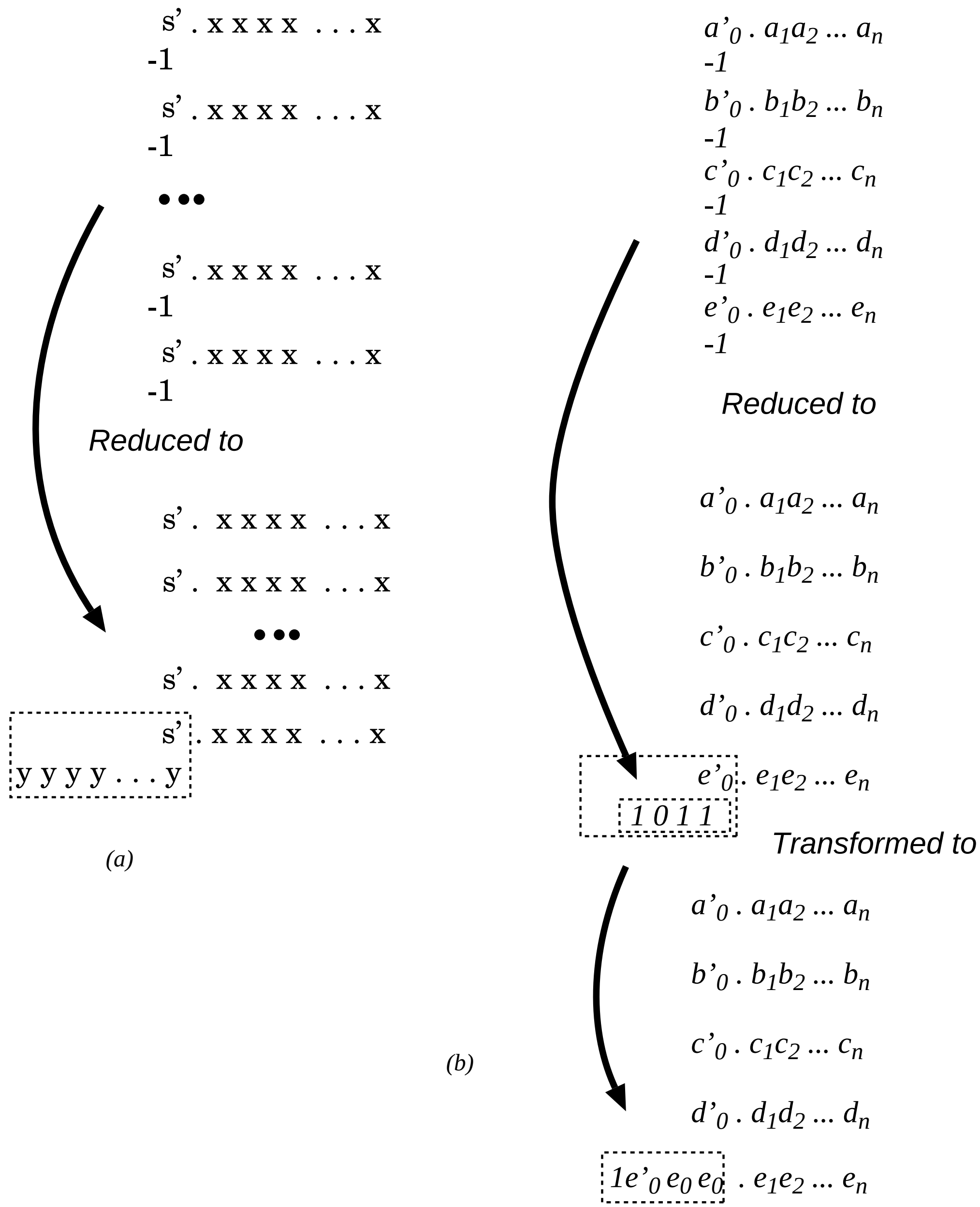

Figure 3.2: SIMPLIFYING SIGN-EXTENSION: (a) GENERAL CASE. (b) EXAMPLE OF SIMPLIFYING ARRAY WITH $m = 5$.

REDUCTION

- By rows
- By columns

[p:2] ADDERS FOR REDUCTION BY ROWS

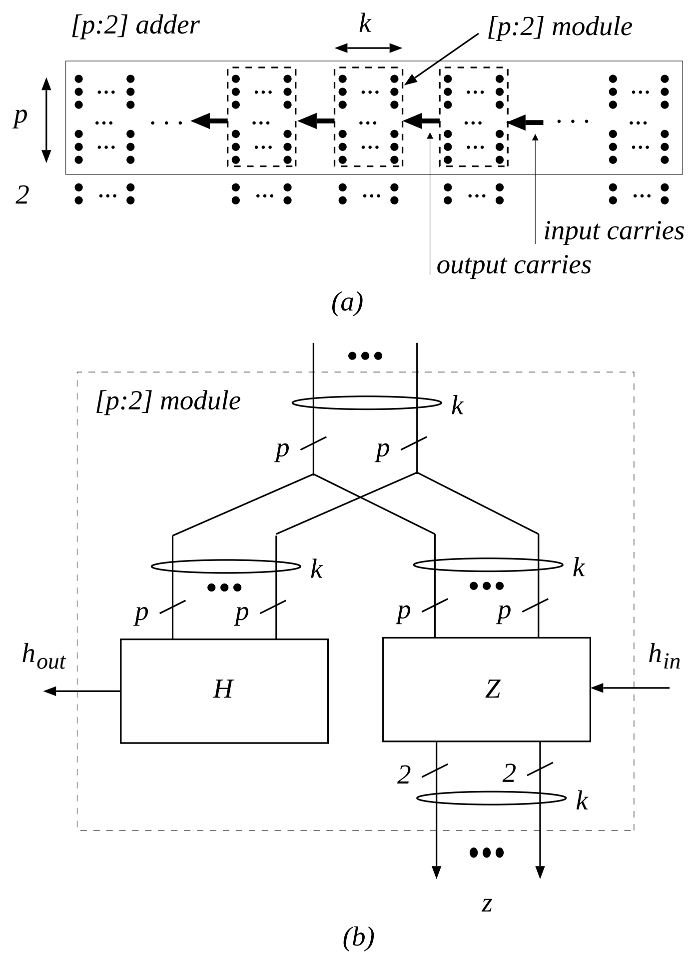

Figure 3.3: A [p:2] adder: (a) Input-output bit-matrix. (b) k-column [p:2] module decomposition.

MODEL OF [p:2] MODULE

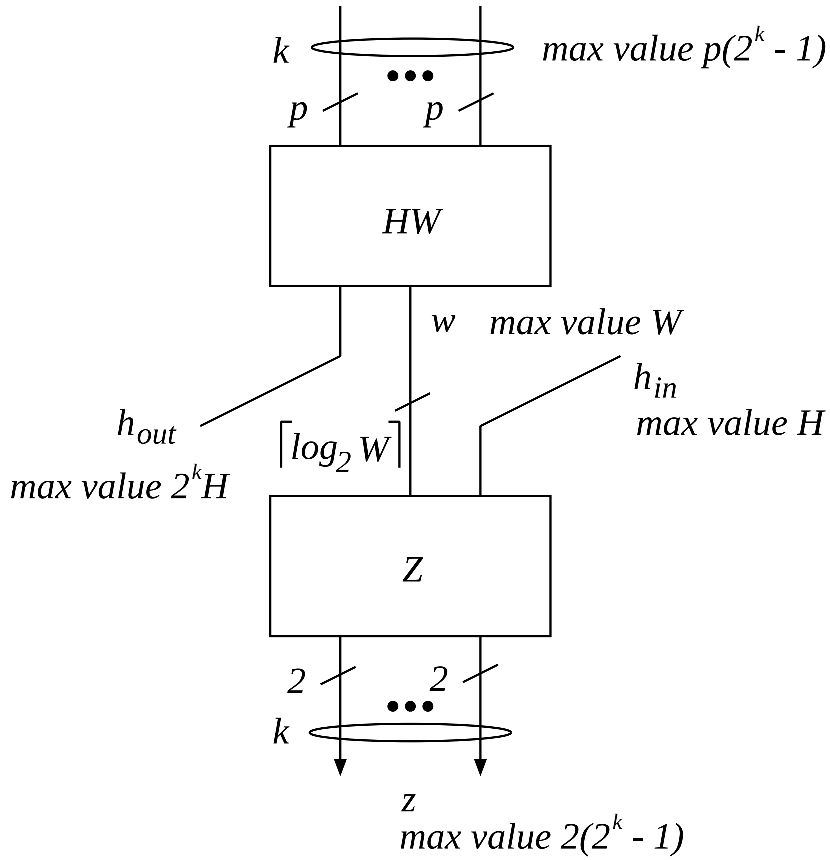

Figure 3.4: A model of a [p:2] module.

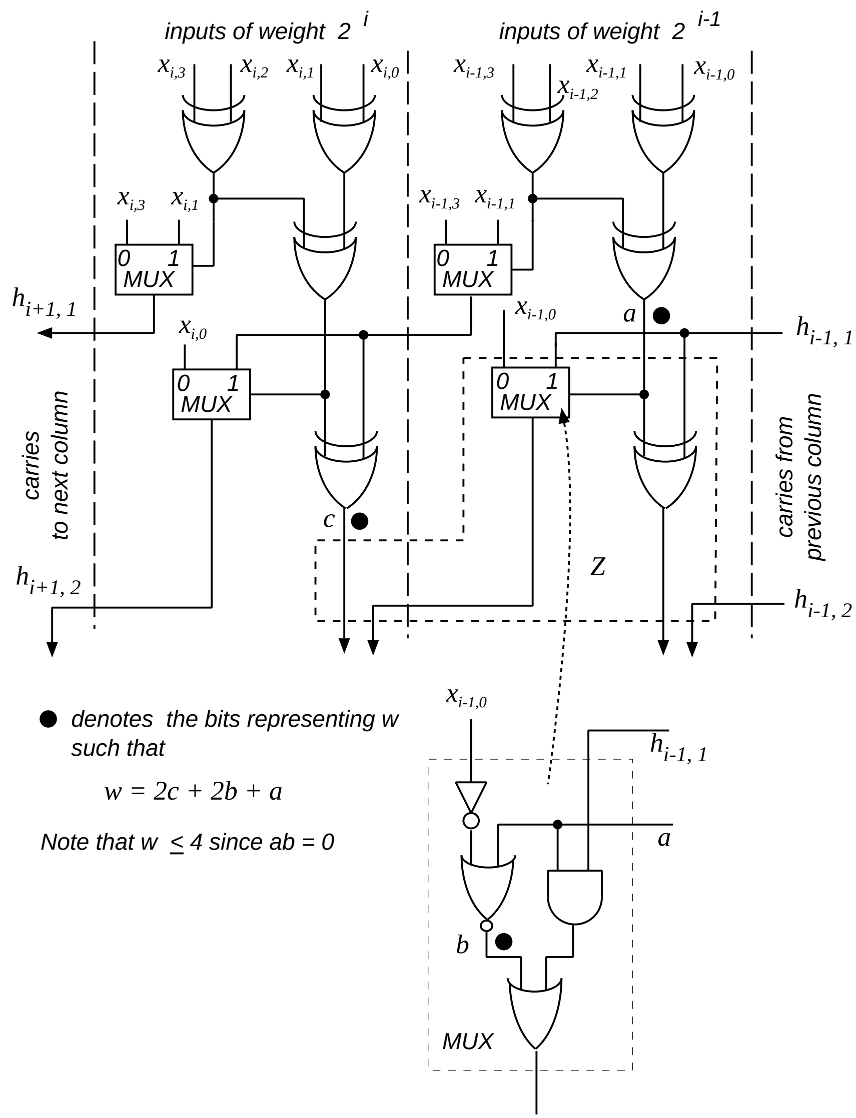

Figure 3.5: Gate network implementation of [4:2] module.

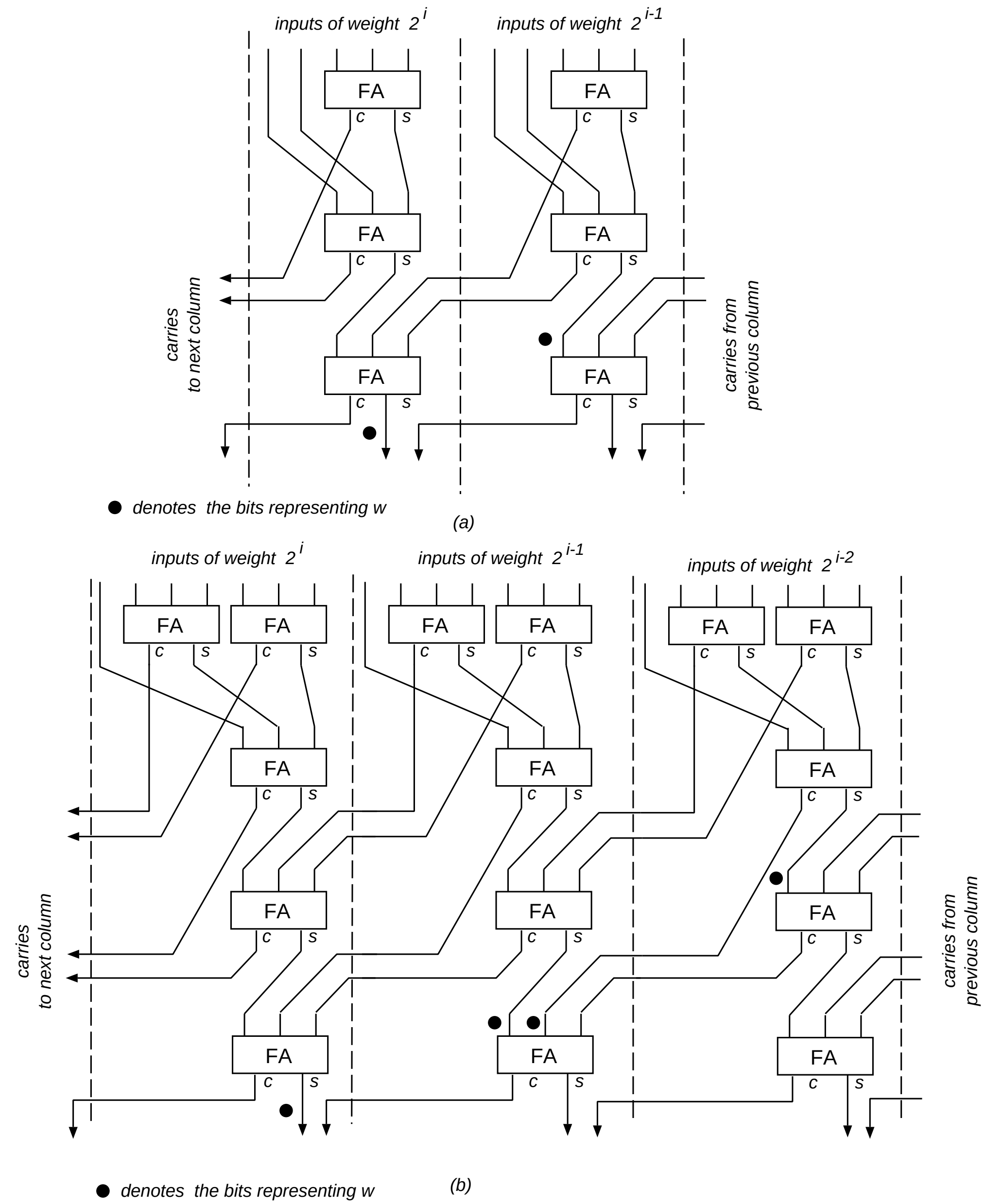

Figure 3.6: (a) [5:2] module. (b) [7:2] module.

$(p{:}q]$ COUNTERS FOR REDUCTION BY COLUMNS

$$\sum_{i=0}^{p-1} x_i = \sum_{j=0}^{q-1} y_j 2^j$$

$$2^q - 1 \geq p, \text{ i.e., } q = \lceil log_2(p+1) \rceil$$

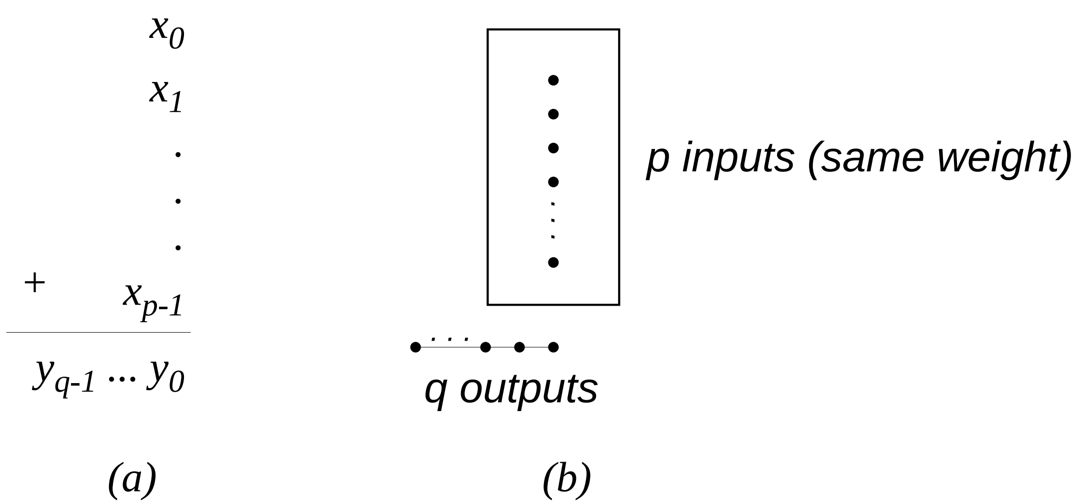

Figure 3.7: (a) $(p{:}q]$ reduction. (b) Counter representation.

IMPLEMENTATION OF $(p{:}q]$ COUNTERS

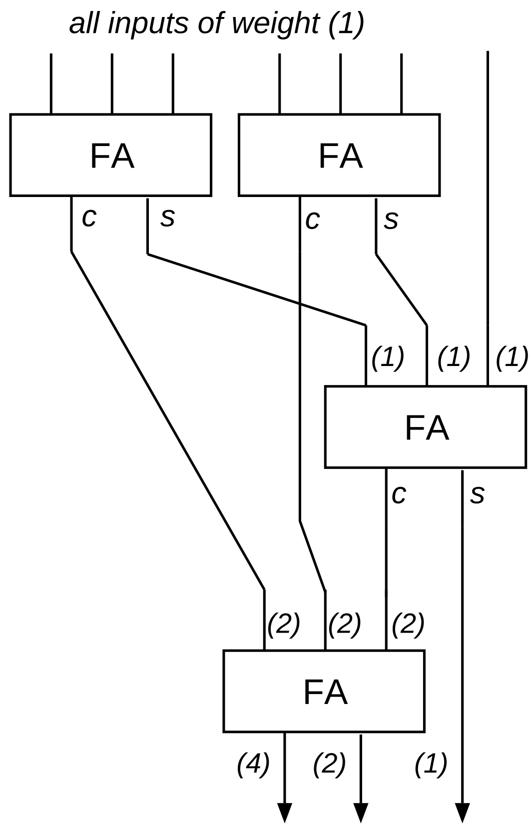

Figure 3.8: Implementation of (7:3] counter by an array of full adders.

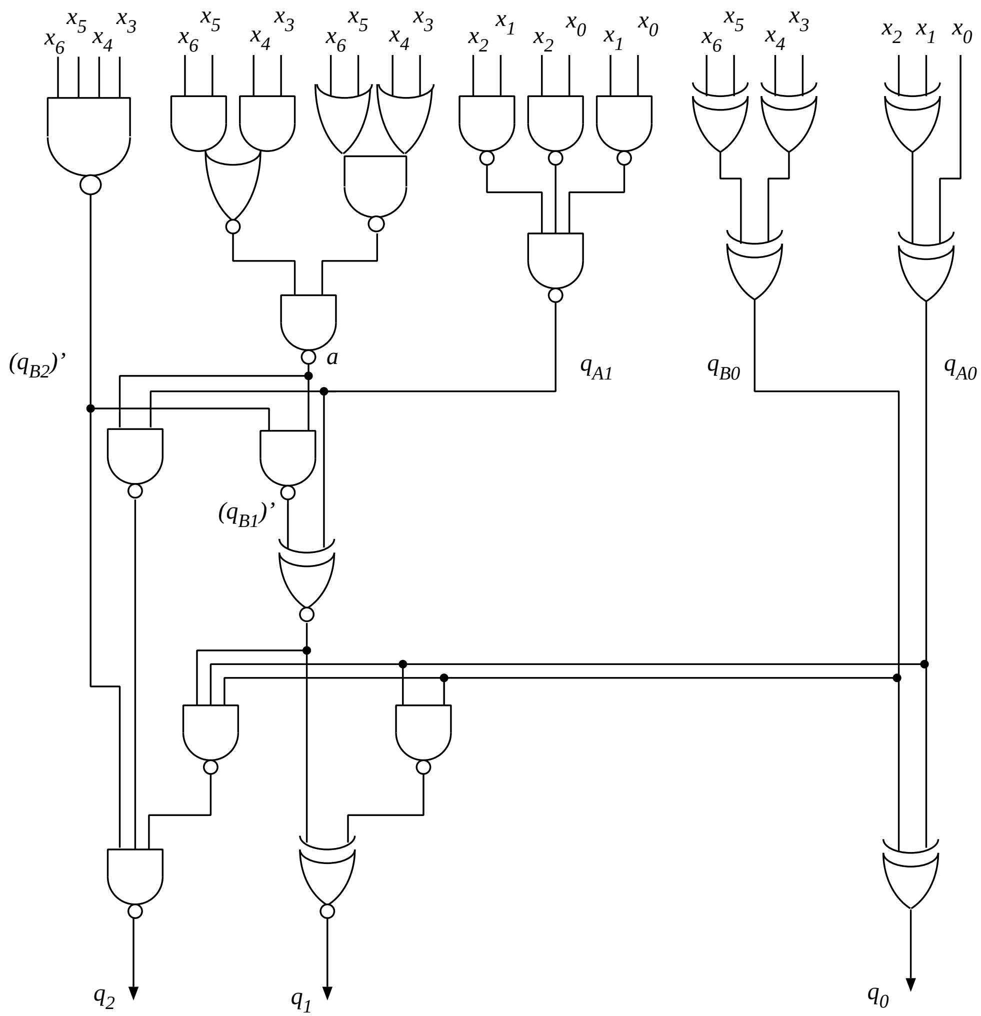

Figure 3.9: Gate network of a (7:3) counter.

MULTICOLUMN COUNTER

$$(p_{k-1}, p_{k-2}, \ldots, p_0 : q]$$

$$v = \sum_{i=0}^{k-1} \sum_{j=1}^{p_i} a_{ij} 2^i \leq 2^q - 1$$

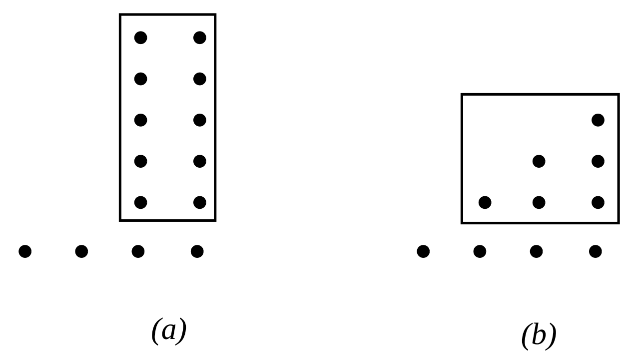

Figure 3.10: (a) (5,5:4] counter. (b) (1,2,3:4] counter.

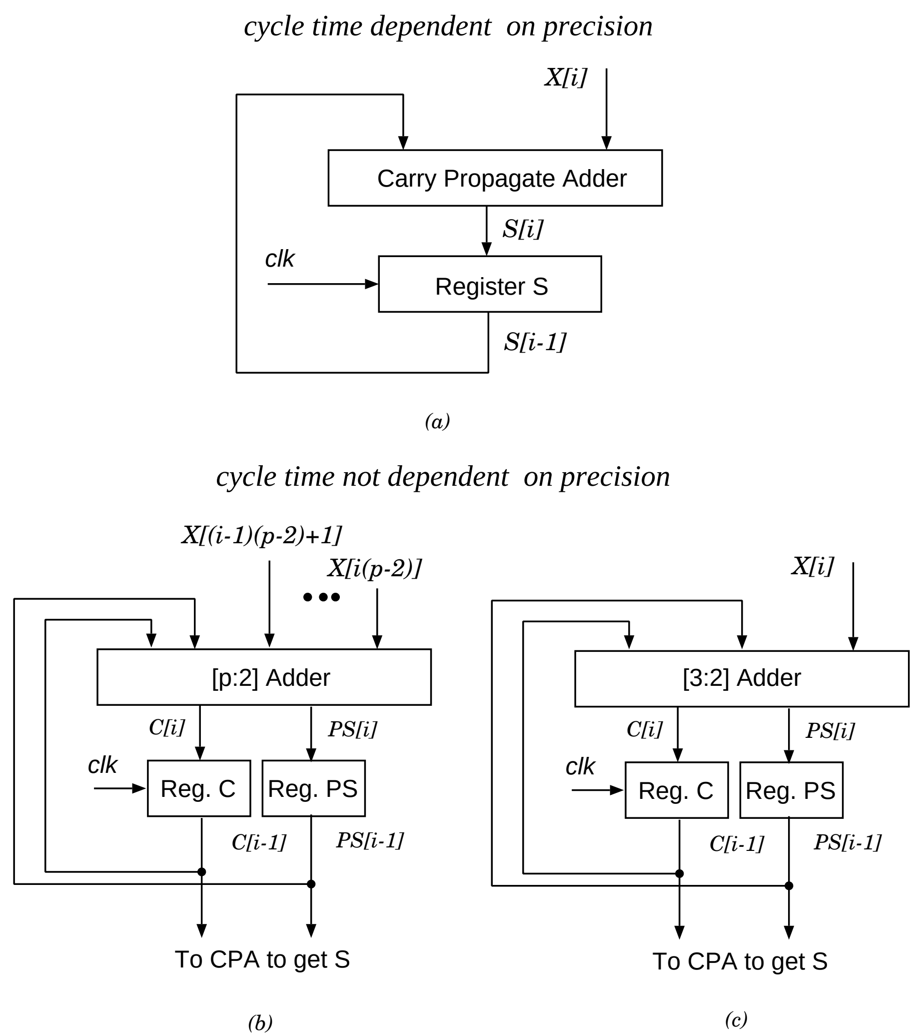

Figure 3.11: SEQUENTIAL MULTIOPERAND ADDITION: a) WITH CONVENTIONAL ADDER. b) WITH [p:2] ADDER. c) WITH [3:2] ADDER.

COMBINATIONAL IMPLEMENTATION

- Reduction by rows: array of adders
  - Linear array
  - Adder tree
- Reduction by columns with (p:q] counters

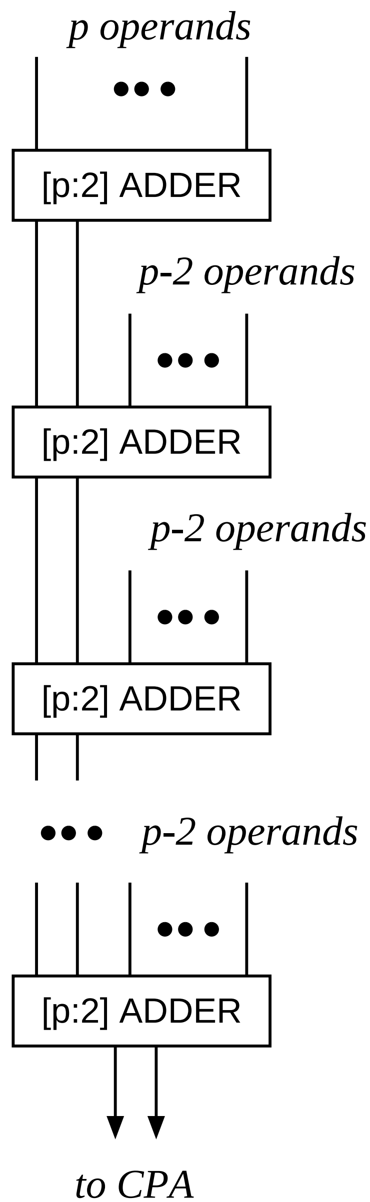

Figure 3.12: LINEAR ARRAY OF [p:2] ADDERS FOR MULTIOPERAND ADDITION.

ADDER TREE

- k - the number of [p:2] CS adders for m operands:

$$pk = m + 2(k-1)$$

$$k = \left\lceil \frac{m-2}{p-2} \right\rceil \qquad \text{[p:2] carry-save adders}$$

- The number of adder levels

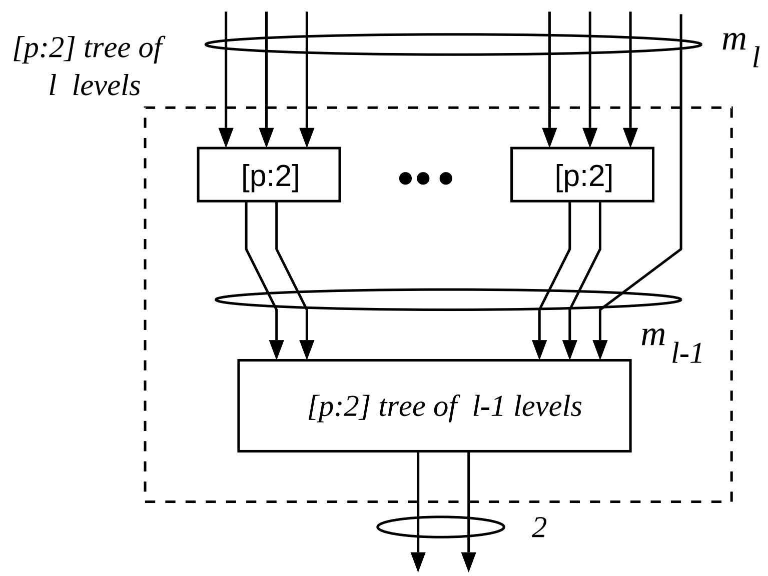

Figure 3.13: Construction of a [p:2] carry-save adder tree.

$$m_l = p \left\lfloor \frac{m_{l-1}}{2} \right\rfloor + m_{l-1} mod\ 2$$

NUMBER OF LEVELS (cont.)

Table 3.1: [3:2] Reduction sequence.

l	1	2	3	4	5	6	7	8	9
m_l	3	4	6	9	13	19	28	42	63

$$m_l \approx \frac{p^l}{2^{l-1}}$$

$$l \approx log_{p/2}(m_l/2)$$

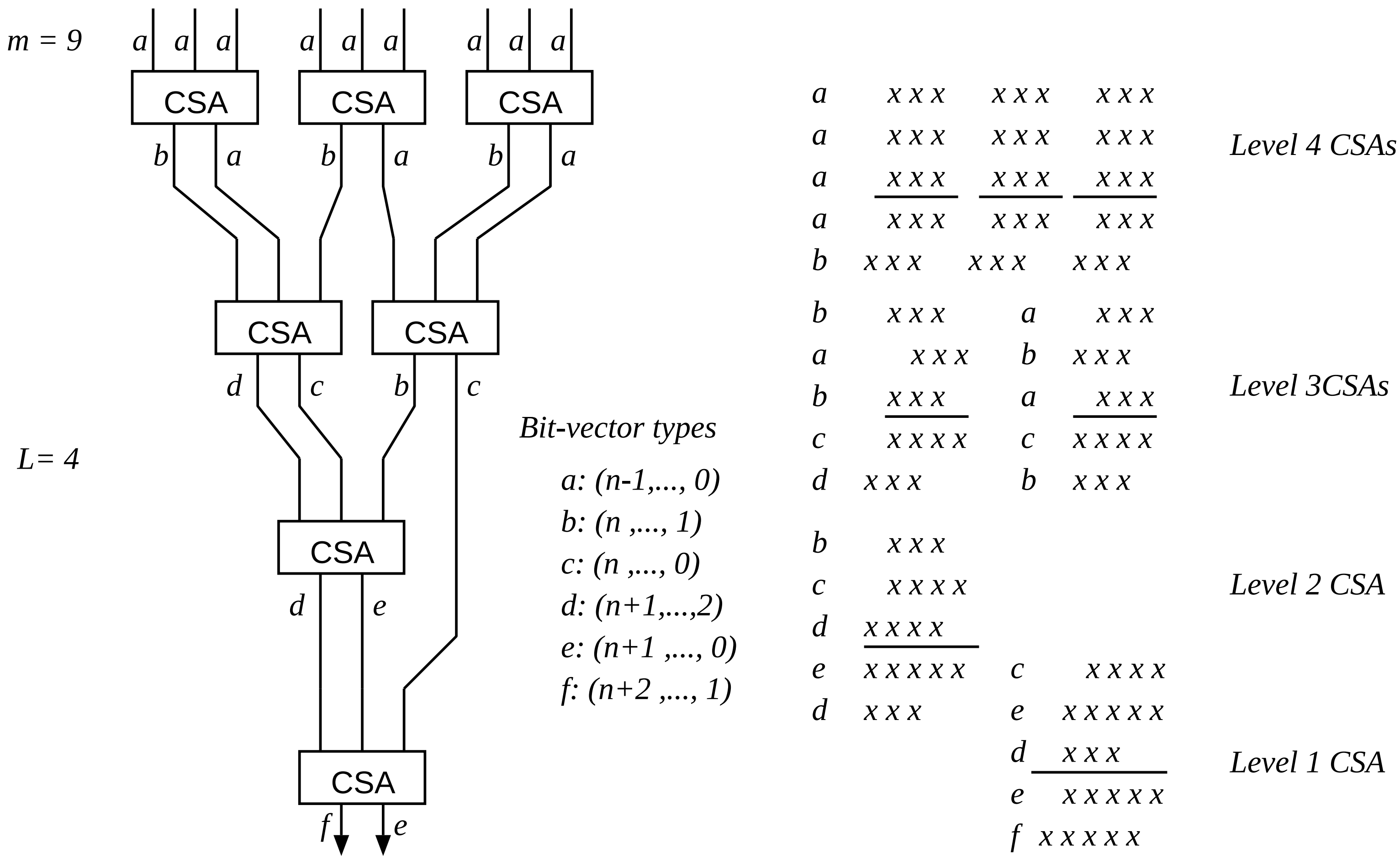

Figure 3.14: [3:2] adder tree for 9 operands (magnitudes with $n = 3$) .

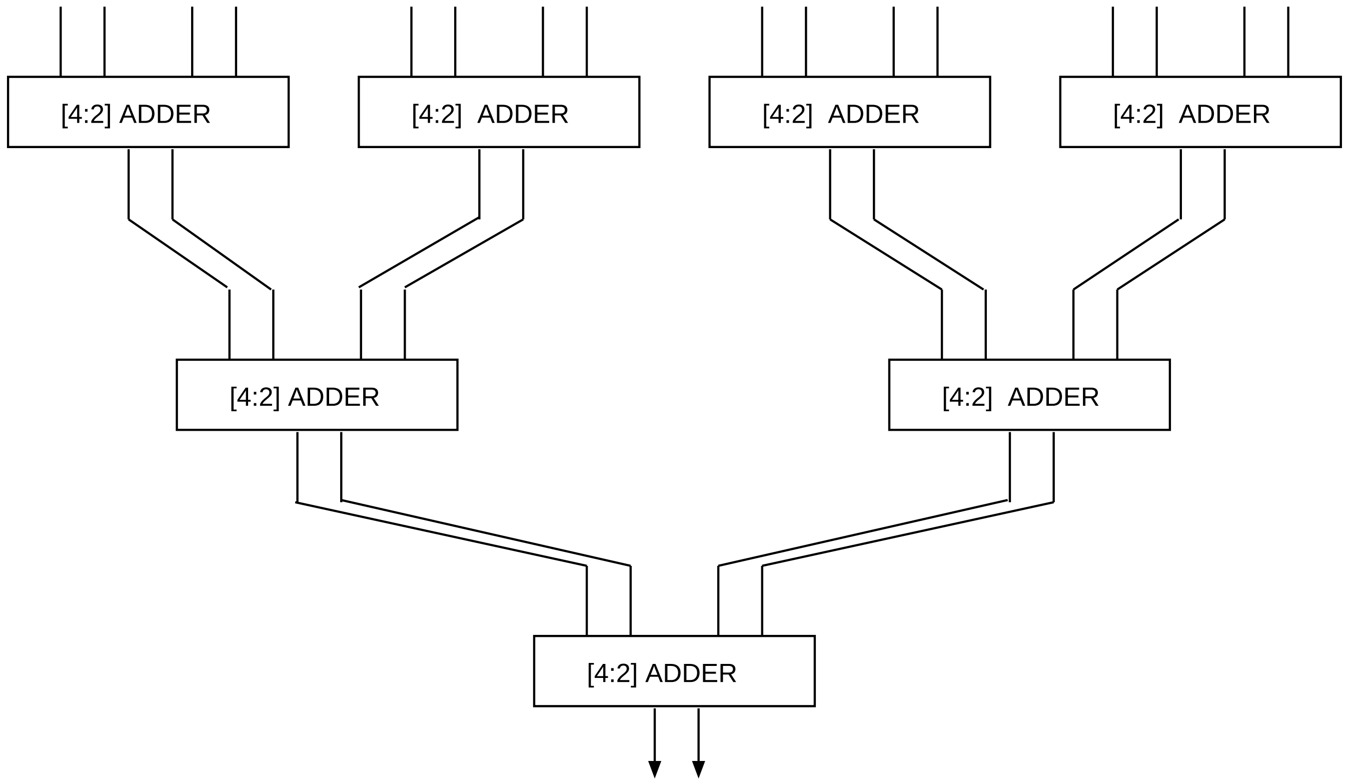

Figure 3.15: Tree of [4:2] adders for $m = 16$.

REDUCTION BY COLUMNS WITH (p:q] COUNTERS

		1	**0**	1	1
		0	**0**	1	0
		1	**0**	0	1
		0	**1**	1	0
		1	**0**	1	0
		1	**1**	1	1
		0	**1**	1	0
		0	**1**	0	1
	0	**1**	1	1	
1	**0**	1	0		

Figure 3.16: Example of reduction using (7:3] counters.

NUMBER OF COUNTER LEVELS

$$m_1 = p$$
$$m_l = p\left\lfloor \frac{m_{l-1}}{q} \right\rfloor + m_{l-1}\ mod\ q$$

$$l \approx log_{p/q}(m_l/q)$$

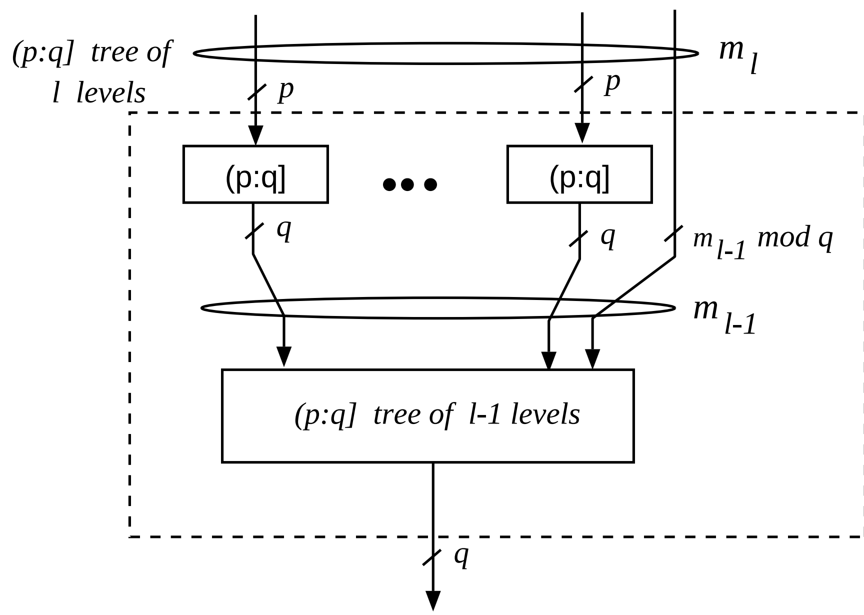

Figure 3.17: Construction of (p:q] reduction tree.

Table 3.2: Sequence for (7:3] counters

Number of levels	1	2	3	4	...
Max. number of rows	7	15	35	79	...

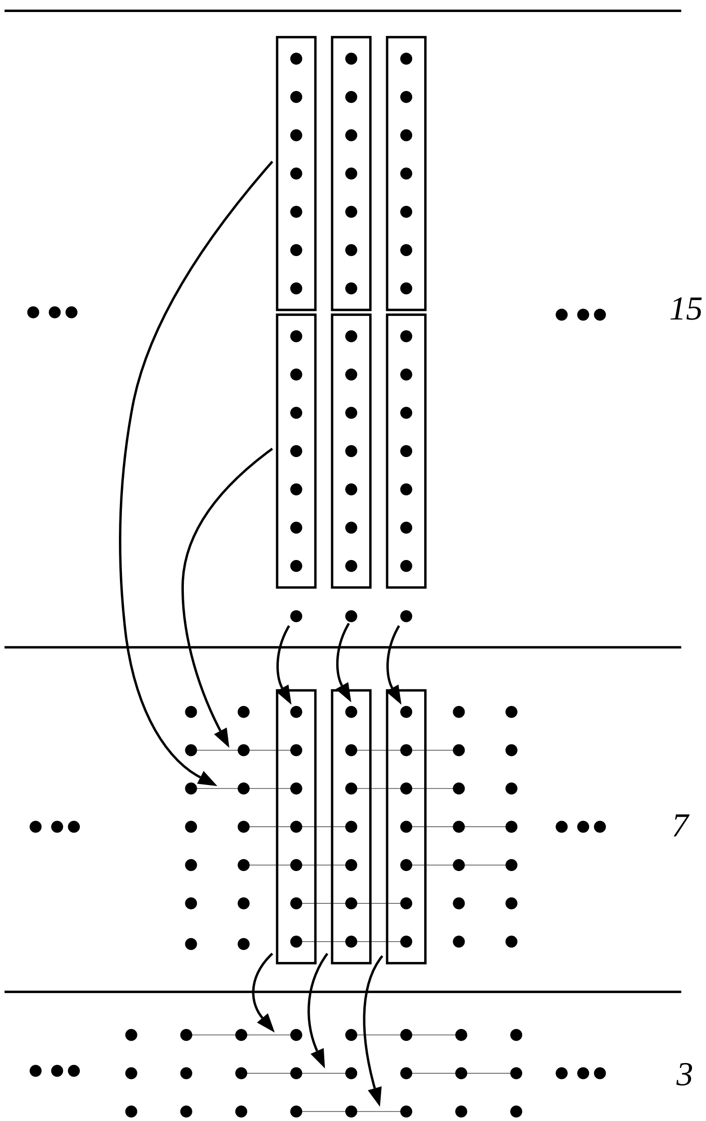

Figure 3.18: Multilevel reduction with (7:3) counters

SYSTEMATIC DESIGN METHOD

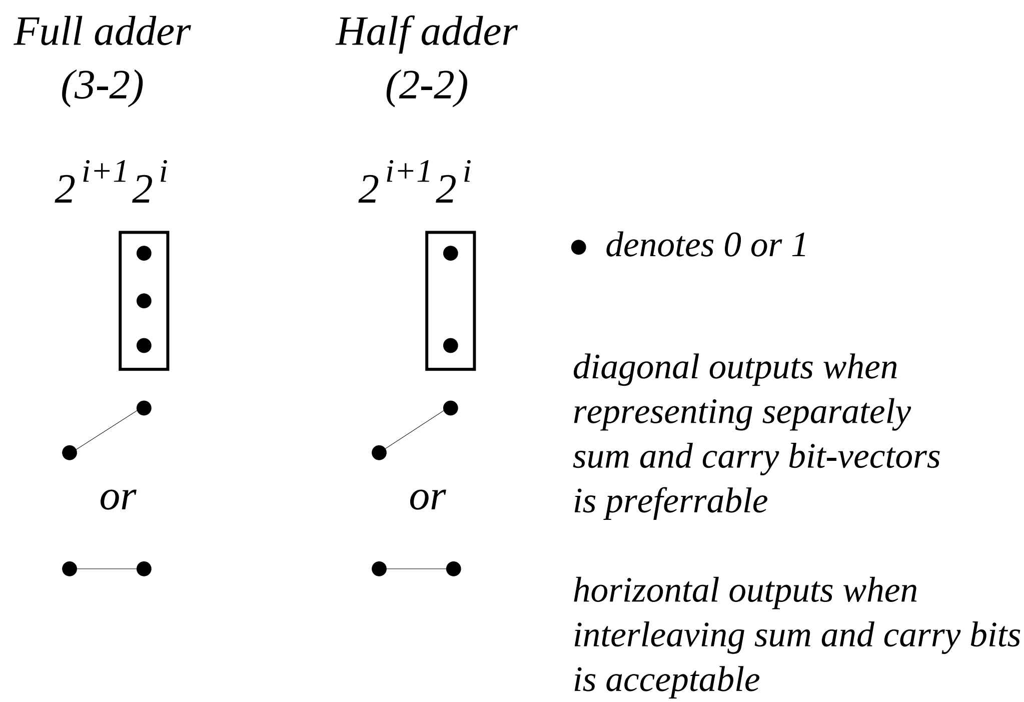

Figure 3.19: Full adder and half adder as (3:2] and (2:2] counters.

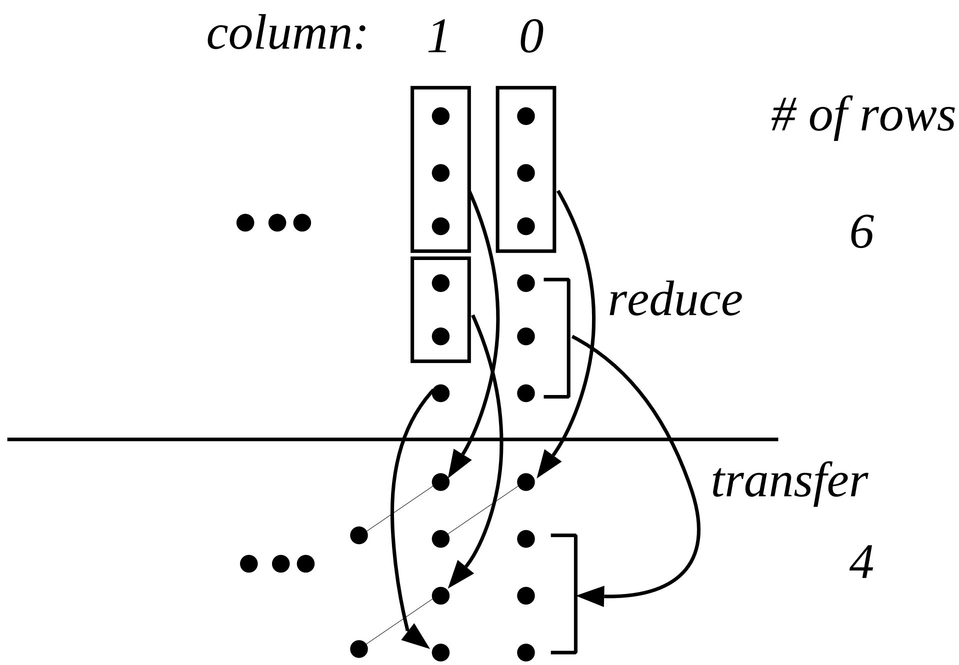

Figure 3.20: Reduction process.

RELATION AT LEVEL l

e_i – number of bits in column i

f_i – number of full adders in column i

h_i – number of half adders in column i

$$e_i - 2f_i - h_i + f_{i-1} + h_{i-1} = m_{l-1}$$

resulting in

$$2f_i + h_i = e_i - m_{l-1} + f_{i-1} + h_{i-1} = p_i$$

Solution producing min number of carries:

$$f_i = \lfloor p_i/2 \rfloor \qquad h_i = p_i \; mod \; 2$$

	i						
	6	5	4	3	2	1	0
$l = 4$							
e_i			8	8	8	8	8
m_3			6	6	6	6	6
h_i			0	0	0	1	0
f_i			2	2	2	1	1
$l = 3$							
e_i		2	6	6	6	6	6
m_2		4	4	4	4	4	4
h_i		0	0	0	0	1	0
f_i		0	2	2	2	1	1
$l = 2$							
e_i		4	4	4	4	4	4
m_1		3	3	3	3	3	3
h_i		0	0	0	0	0	1
f_i		1	1	1	1	1	0
$l = 1$							
e_i	1	3	3	3	3	3	3
m_0	2	2	2	2	2	2	2
h_i	0	0	0	0	0	0	1
f_i	0	1	1	1	1	1	0

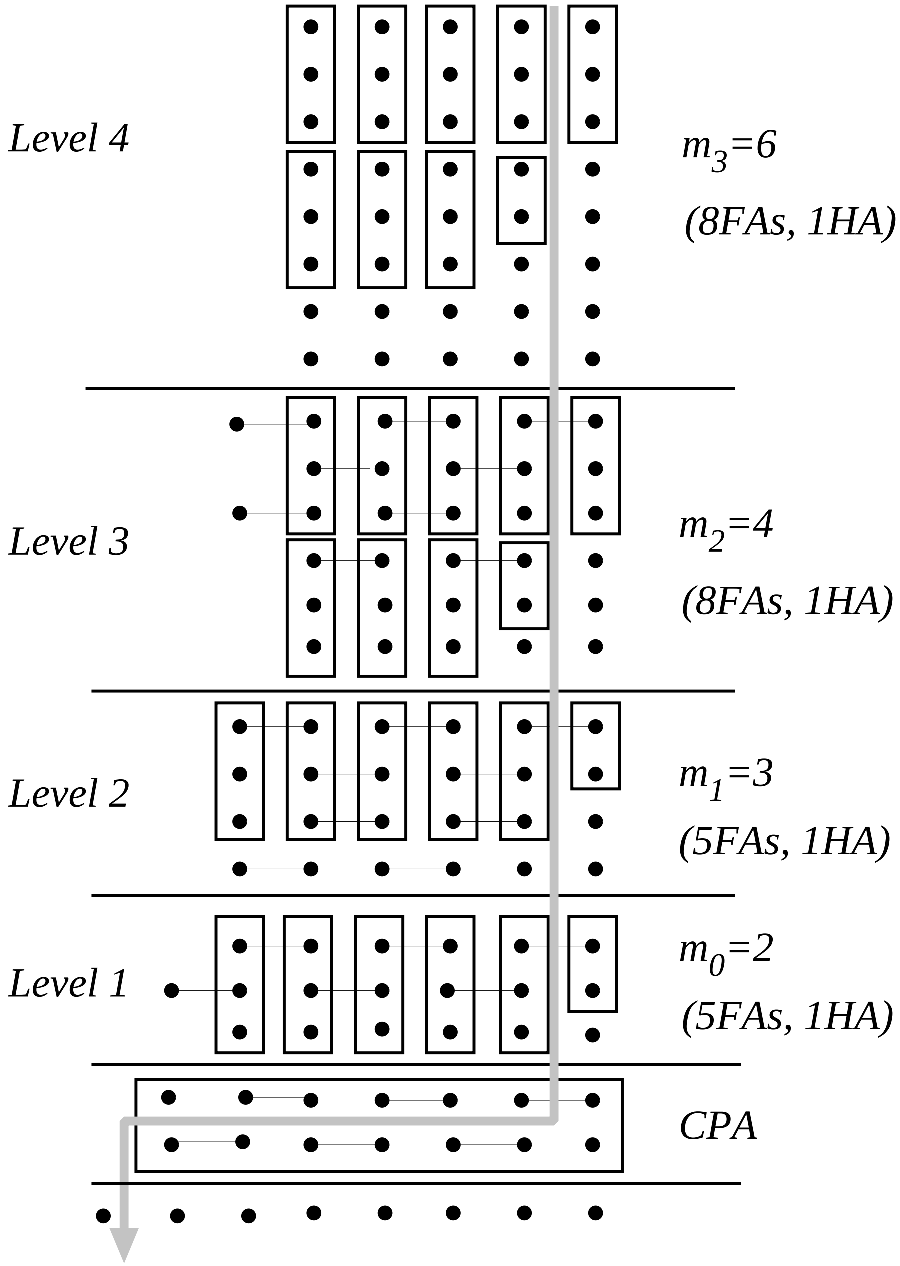

Figure 3.21: Reduction by columns of 8 5-bit magnitudes. Cost of reduction: 26 FAs and 4 HAs.

EXAMPLE: ARRAY FOR $f = a + 3b + 3c + d$

Operands in [-4,3). Result range:

$$-4 + (-12) + (-12) - 4 = -32 \leq f \leq 3 + 9 + 9 + 3 = 24$$

$$
\begin{array}{r|cccccc}
a & a_2 & a_2 & a_2 & a_2 & a_1 & a_0 \\
b & b_2 & b_2 & b_2 & b_2 & b_1 & b_0 \\
2b & b_2 & b_2 & b_2 & b_1 & b_0 & 0 \\
c & c_2 & c_2 & c_2 & c_2 & c_1 & c_0 \\
2c & c_2 & c_2 & c_2 & c_1 & c_0 & 0 \\
d & d_2 & d_2 & d_2 & d_2 & d_1 & d_0
\end{array}
$$

transformed into

$$
\begin{array}{r|cccc}
a & & a_2' & a_1 & a_0 \\
 & & \text{-1} & & \\
b & & b_2' & b_1 & b_0 \\
 & & \text{-1} & & \\
2b & b_2' & b_1 & b_0 & \\
 & \text{-1} & & & \\
c & & c_2' & c_1 & c_0 \\
 & & \text{-1} & & \\
2c & c_2' & c_1 & c_0 & \\
 & \text{-1} & & & \\
d & & d_2' & d_1 & d_0 \\
 & & \text{-1} & &
\end{array}
$$

FINAL BIT MATRIX

$$
\begin{array}{|cccccc}
1 & 0 & b_2' & a_2' & a_1 & a_0 \\
 & & c_2' & b_2' & b_1 & b_0 \\
 & & & b_1 & b_0 & \\
 & & & c_2' & c_1 & c_0 \\
 & & & c_1 & c_0 & \\
 & & & d_2' & d_1 & d_0
\end{array}
$$

	i					
	5	4	3	2	1	0
$l = 3$						
e_i	1	0	2	6	6	4
m_2	4	4	4	4	4	4
h_i	0	0	0	1	0	0
f_i	0	0	0	1	1	0
$l = 2$						
e_i	1	0	4	4	4	4
m_1	3	3	3	3	3	3
h_i	0	0	0	0	0	1
f_i	0	0	1	1	1	0
$l = 1$						
e_i	1	1	3	3	3	3
m_0	2	2	2	2	2	1*
h_i	0	0	0	0	0	0
f_i	0	0	1	1	1	1

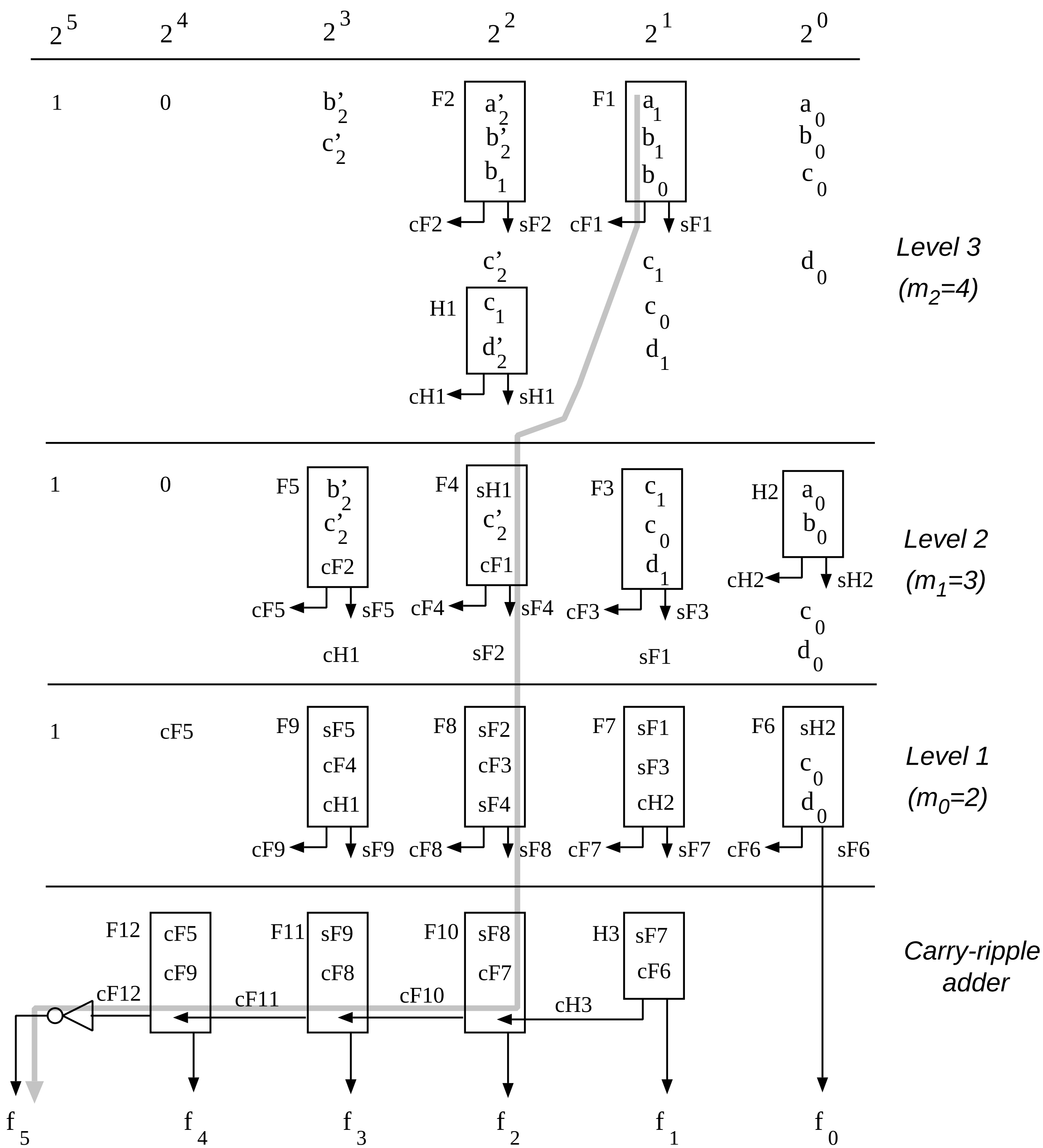

Figure 3.22: Reduction array $f = a + 3b + 3c + d$.

PIPELINED LINEAR ARRAY

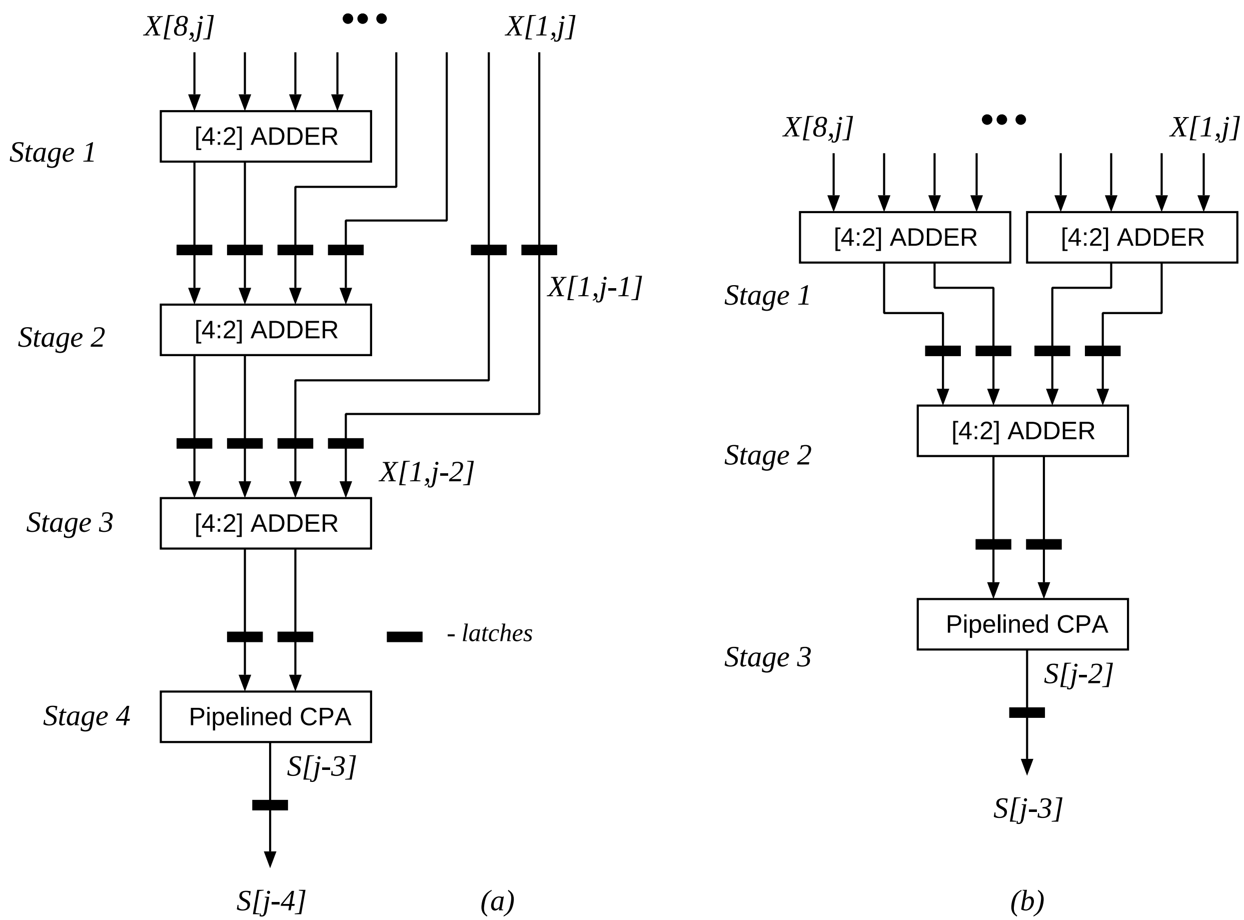

Figure 3.23: Pipelined arrays with [4:2] adders for computing $S[j] = \sum_{i=1}^{8} X[i,j],\ j = 1, \ldots, N$: (a) Linear array. (b) Tree array.

PARTIALLY COMBINATIONAL IMPLEMENTATION

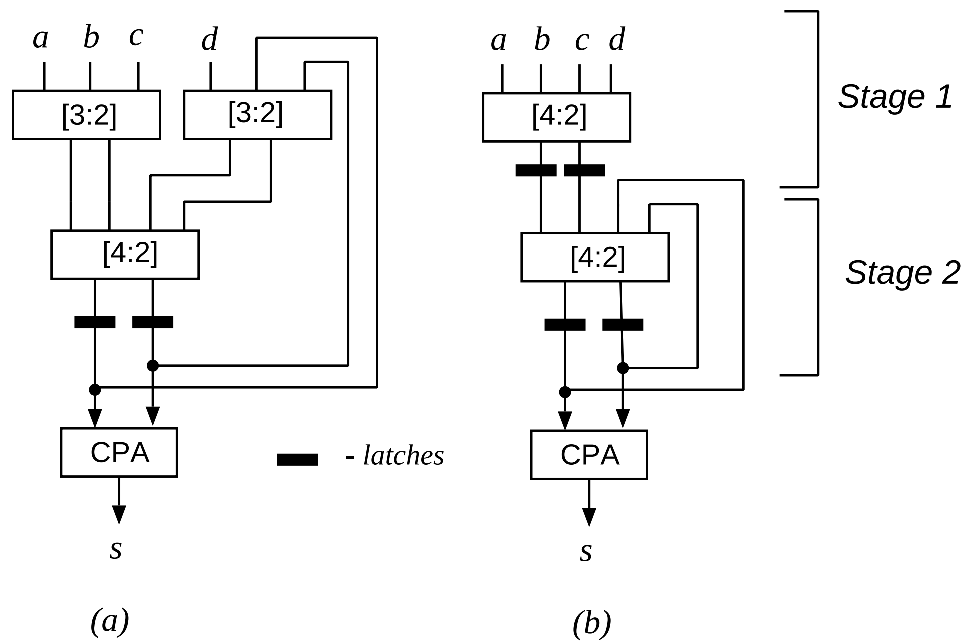

Figure 3.24: Partially combinational scheme for summation of 4 operands per iteration: (a) Nonpipelined. (b) Pipelined.

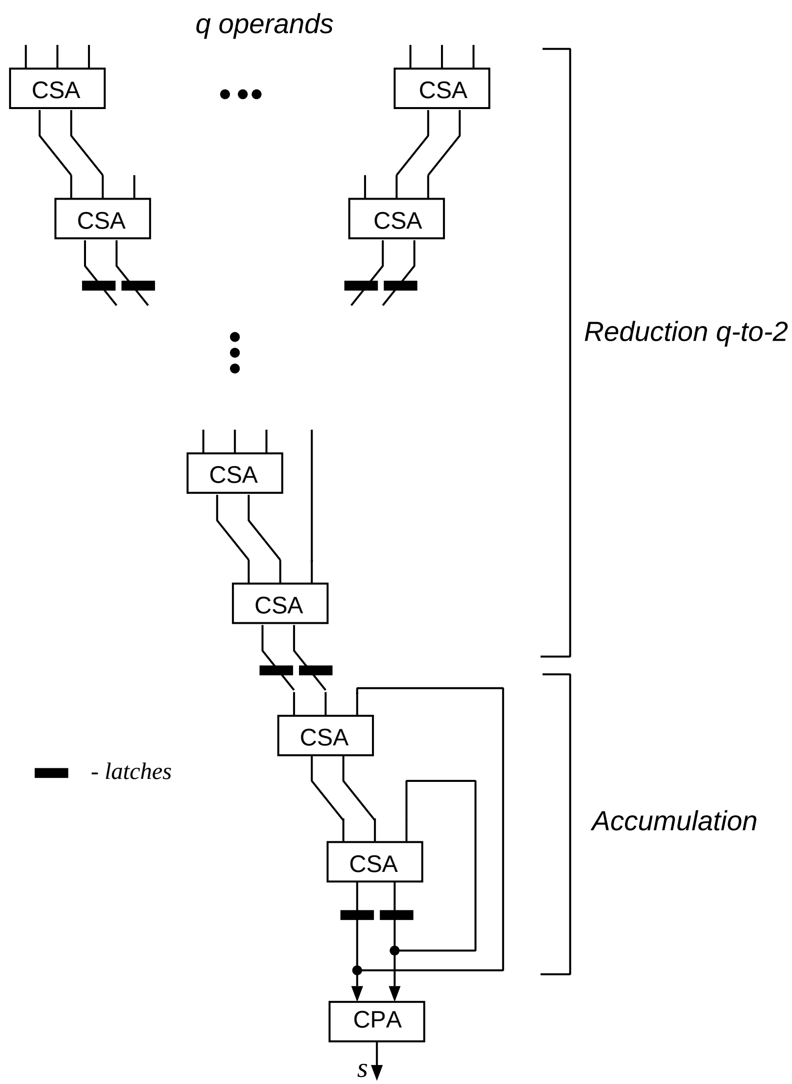

Figure 3.25: Scheme for summation of q operands per iteration.