Multiplier-less Multiplication by Constants

Dr. Shoab A. Khan

Multiplication by Constant

- In many algorithms a large percentage of multiplications are by constants
- Complexity of a general purpose multiplier is not required
  - Generate Partial Products (PPs) only for 1s in the constant multiplier
- The number of PPs can be further reduced using canonic sign digit format

Example: FIR Filter

- In an FIR filter all coefficients are constant
- For a fully parallel implementation, general purpose multipliers are not required
- Coefficients are converted in canonic sign digit form

Canonic Sign Digit (CSD)

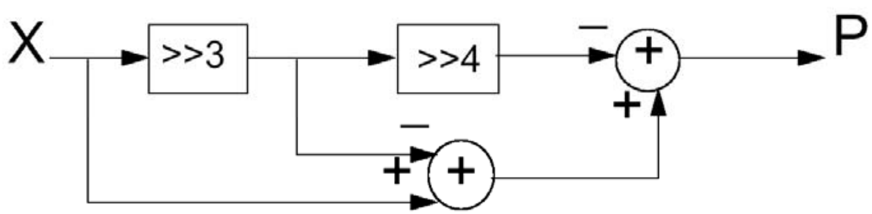
- No 2 consecutive bits are non-zero
- Contains minimum possible number of non-zero bits
- Representation is unique

$$C = \sum_{i=0}^{N-1} s_i \, 2^{\mathrm{i}} \text{ for } s_i \in \{-1, 0, 1\}$$

Canonic Sign Digit (CSD)

CSD is obtained using string property

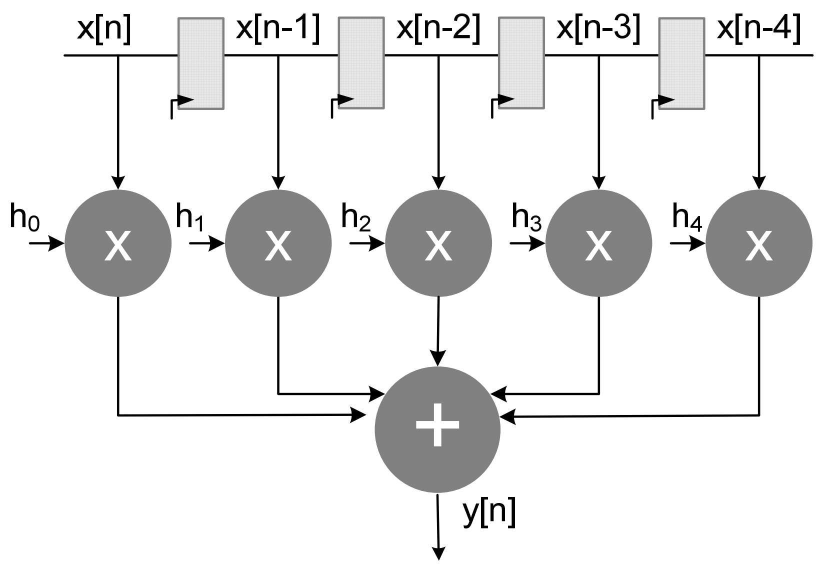
- Examples a Q1.7 format number
  - **01111111** $= 2^0 - 2^{-7} = 1000000\bar{1}$
  - $01101111 \rightarrow 0111000\bar{1} \rightarrow 100\bar{1}000\bar{1}$
    - $k = 2^0 - 2^{-3} - 2^{-7}$
    - $Kx = x2^0 - x2^{-3} - x2^{-7}$



FIR filter

- Convolution summation with constant coefficients h[k]

$$y[n] = \sum_{k=0}^{N-1} h[k]x[n-k]$$



Conversion of FIR Coefficient in CSD

- Only one nonzero CSD digit for approximately each 20 dB of stopband attenuation
- Four non-zero digits per coefficient for 80 dB stopband attenuation

Example: CSD Representation

- Let a coefficient is

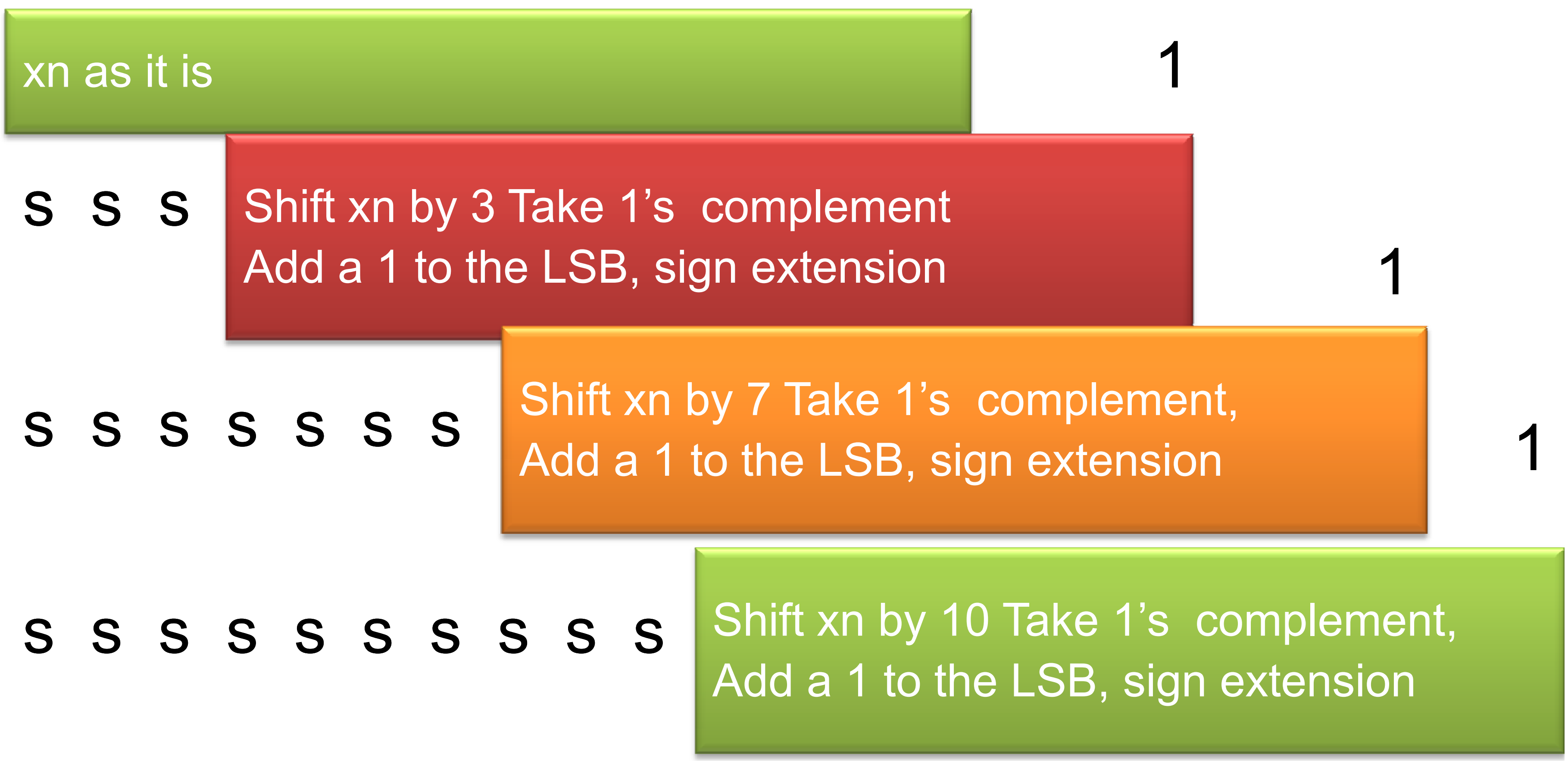
0 1 1 0 1 1 1 0 1 1 0 1 1

- Converting to CSD and keeping 4 non-zero digits:

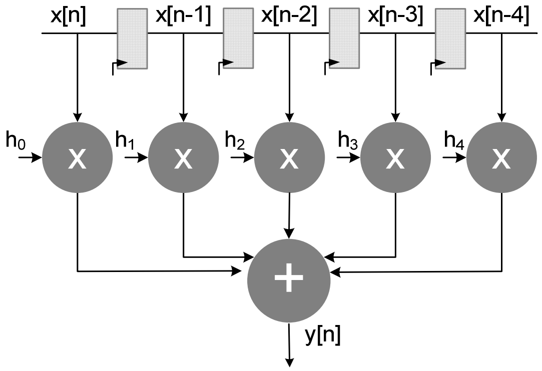
$1 \quad 0 \quad 0 \quad \bar{1} \quad 0 \quad 0 \quad 0 \quad \bar{1} \quad 0 \quad 0 \quad \bar{1}$

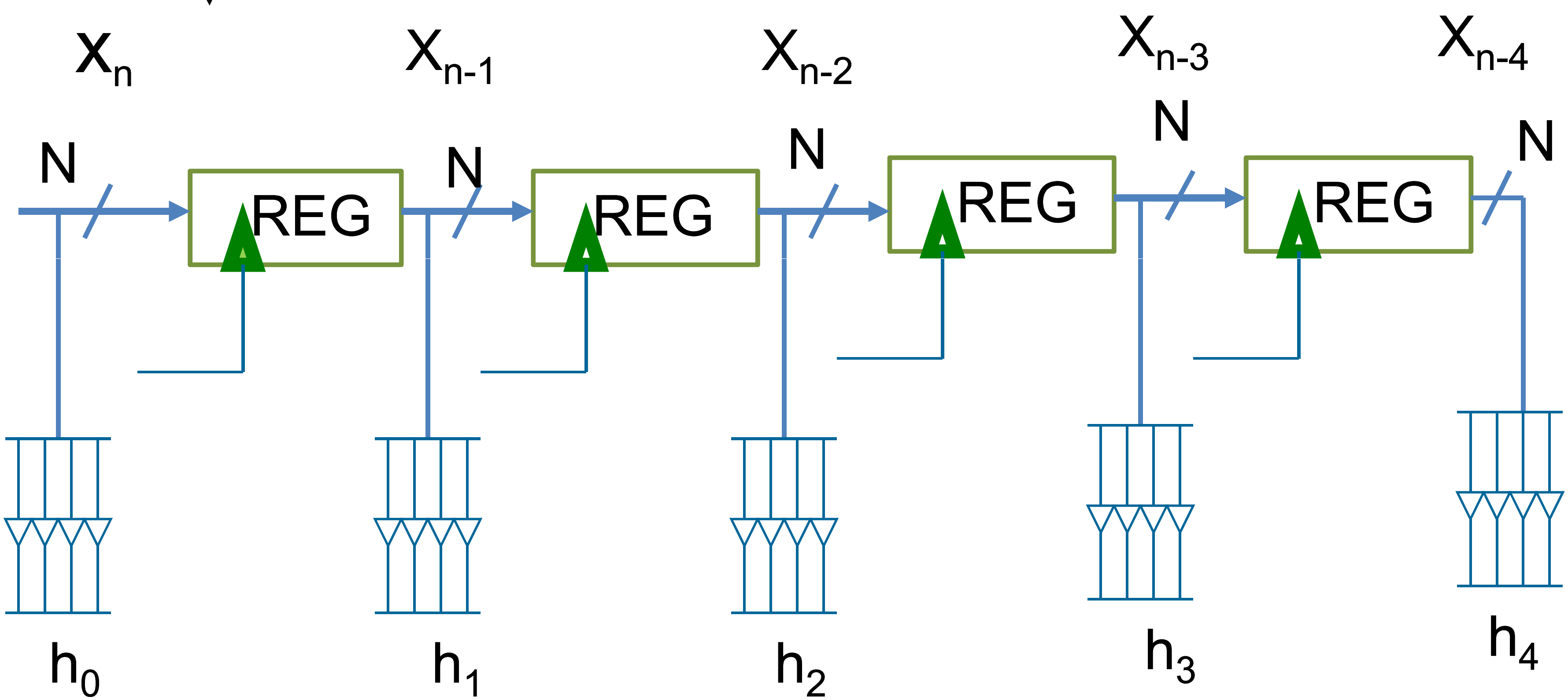
$2^{0} \quad 2^{-1} \quad 2^{-2} \quad 2^{-3} \quad 2^{-4} \quad 2^{-5} \quad 2^{-6} \quad 2^{-7} \quad 2^{-8} \quad 2^{-9} \quad 2^{-10}$

CSD multiplier

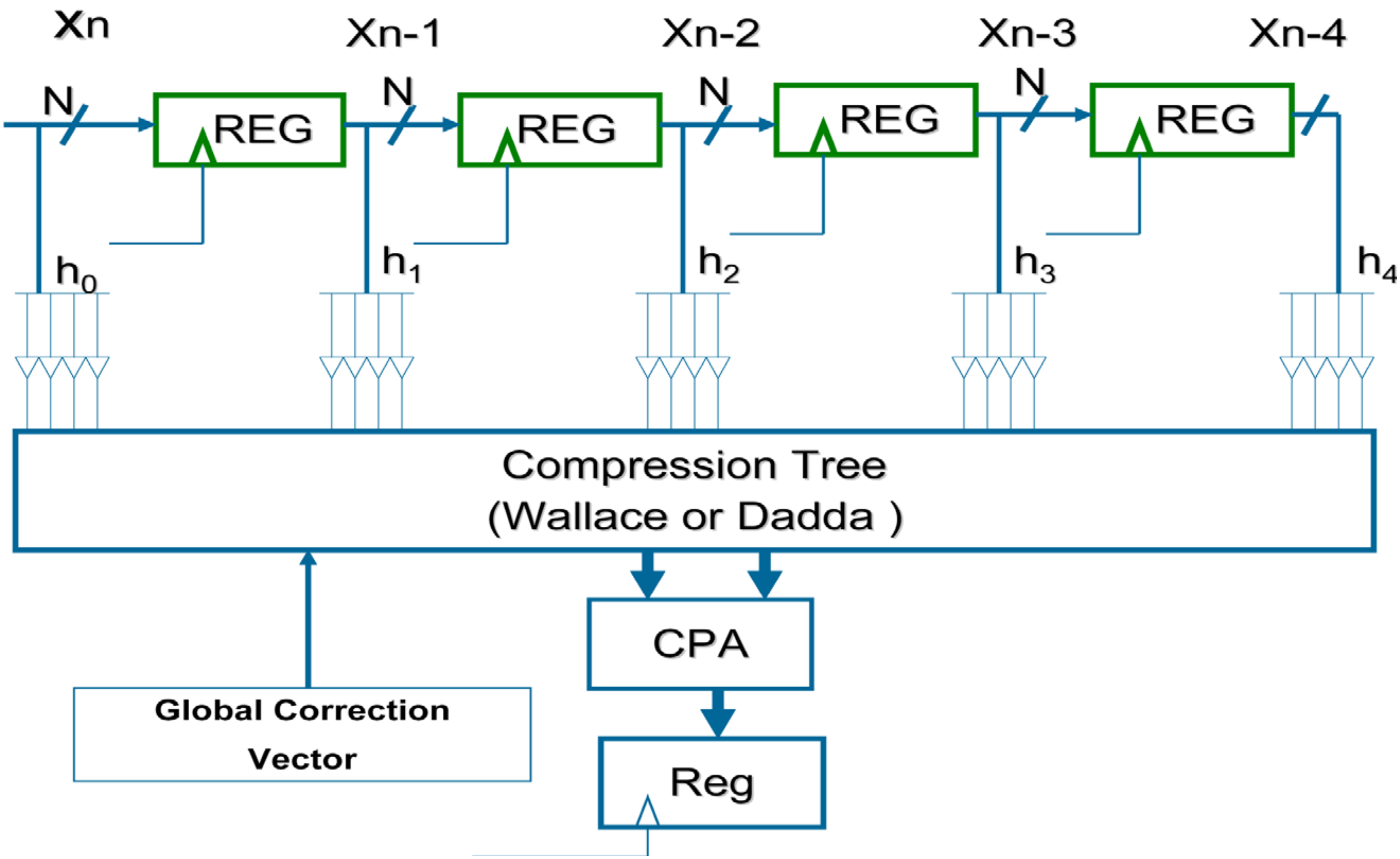


CSD Multiplier in 5-coeff FIR filter





An Optimal Direct Form FIR Filter Architecture



Example: CSD Representation

0110111011011101

$0110111011100\bar{1}01$

$0110111\,1\,00\bar{1}00\bar{1}01$

$0111000\bar{1}00\bar{1}00\bar{1}01$

$100\bar{1}000\bar{1}00\bar{1}00\bar{1}01$

CSD FIR paper

CANONICAL SIGNED DIGIT REPRESENTATION FOR FIR DIGITAL FILTERS

k	h(k)	h(k) Rounded	h(k) Canonical Signed Digit	Adds	Subtracts	Total
0	-0.0057534026	-575	0000 0010 0100 0001	1	2	3
1	0.00099026691	99	0000 0001 0100 0101	2	2	4
2	0.0075733471	757	0000 0101 0001 0101	3	2	5
3	-0.0065141204	-651	0000 0010 1001 0101	2	3	5
4	0.013960509	1396	0000 1010 1001 0100	2	3	5
5	0.0022951644	230	0000 0001 0010 1010	2	2	4
6	-0.019994041	-1999	0000 1000 0101 0001	2	2	4
7	0.0071369656	714	0000 0101 0100 1010	3	2	5
8	-0.039657373	-3966	0001 0000 1000 0010	2	1	3
9	0.011260066	1126	0000 0100 1010 1010	3	2	5
10	0.066233635	6623	0010 1010 0010 0001	2	3	5
11	-0.010497202	-1050	0000 0100 0010 1010	1	3	4
12	0.08513616	8514	0010 0001 0100 0010	4	0	4
13	-0.12024988	-12025	0101 0001 0000 1001	3	2	5
14	-0.2967858	-29679	1001 0100 0001 0001	3	2	5
15	0.30410913	30411	1000 1001 0101 0101	2	5	7
16	0.30410913	30411	1000 1001 0101 0101	2	5	7
17	-0.2967858	-29679	1001 0100 0001 0001	3	2	5
18	-0.12024988	-12025	0101 0001 0000 1001	3	2	5
19	0.08513616	8514	0010 0001 0100 0010	4	0	4
20	-0.010497202	-1050	0000 0100 0010 1010	1	4	5
21	0.066233635	6623	0010 1010 0010 0001	2	3	5
22	0.011260066	1126	0000 0100 1010 1010	3	2	5
23	-0.039657373	-3966	0001 0000 1000 0010	2	1	3
24	0.0071369656	714	0000 0101 0100 1010	3	2	5
25	-0.019994041	-1999	0000 1000 0101 0001	2	2	4
26	0.0022951644	230	0000 0001 0010 1010	2	2	4
27	0.013960509	1396	0000 1010 1001 0100	2	3	5
28	-0.0065141204	-651	0000 0010 1001 0101	2	3	5
29	0.0075733471	757	0000 0101 0001 0101	3	2	5
30	0.00099026691	99	0000 0001 0100 0101	2	2	4
31	-0.0057534026	-575	0000 0010 0100 0001	1	2	3
			TOTAL ADDS/SUBS			147

8 CSD Digits	7 CSD Digits	6 CSD Digits	5 CSD Digits	4 CSD Digits	3 CSD Digits	2 CSD Digits
-575	-575	-575	-575	-575	-575	-576
99	99	99	99	99	100	96
757	757	757	757	756	752	768
-651	-651	-651	-651	-652	-656	-640
1396	1396	1396	1396	1392	1408	1536
230	230	230	230	230	232	224
-1999	-1999	-1999	-1999	-1999	-2000	-2016
714	714	714	714	712	704	768
-3966	-3966	-3966	-3966	-3966	-3966	-3968
1126	1126	1126	1126	1128	1120	1152
6623	6623	6623	6623	6624	6656	6144
-1050	-1050	-1050	-1050	-1050	-1048	-1056
8514	8514	8514	8514	8514	8512	8448
-12025	-12025	-12025	-12025	-12024	-12032	-12288
-29679	-29679	-29679	-29679	-29680	-29696	-28672
30411	30411	30412	30416	30400	30464	30720

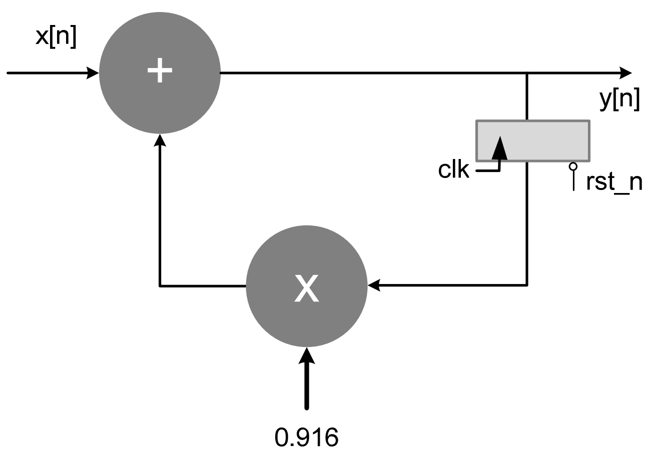
Optimized DFG Transformation

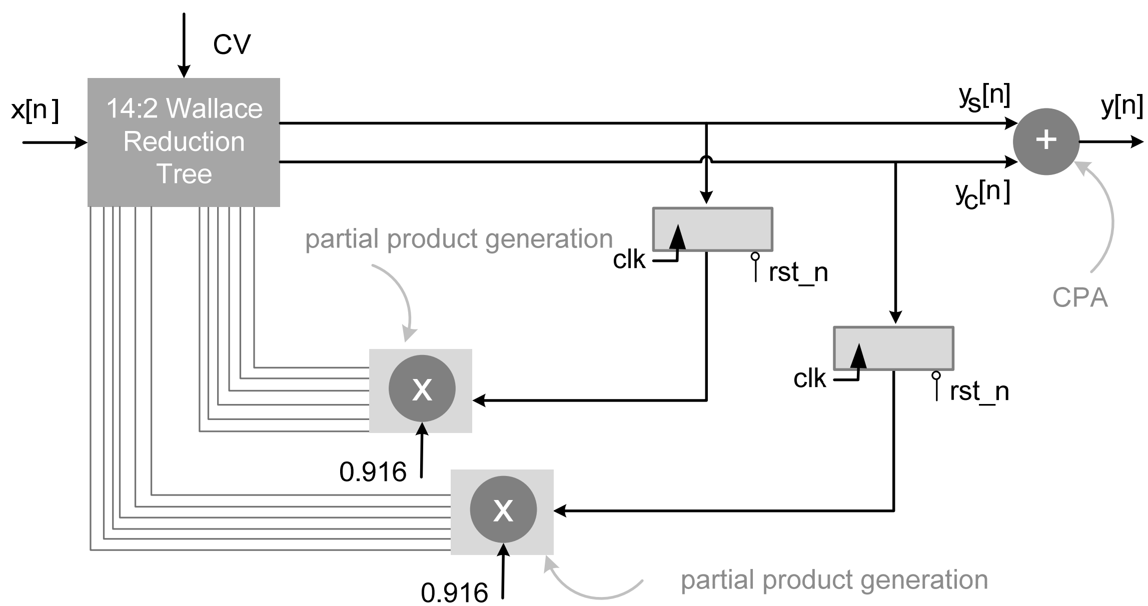
- Use compression tree and remove the use of CPA in a feedback loop
- The result is kept in partial sum and partial carry form
- The first order difference equation changes to

$$\{y_s[n], y_c[n]\} = 0.916y_s[n-1] + 0.916y_c[n-1] + x[n]$$

Example 1: First Order IIR Filter

DFG with one adder and one multiplier in the critical path.

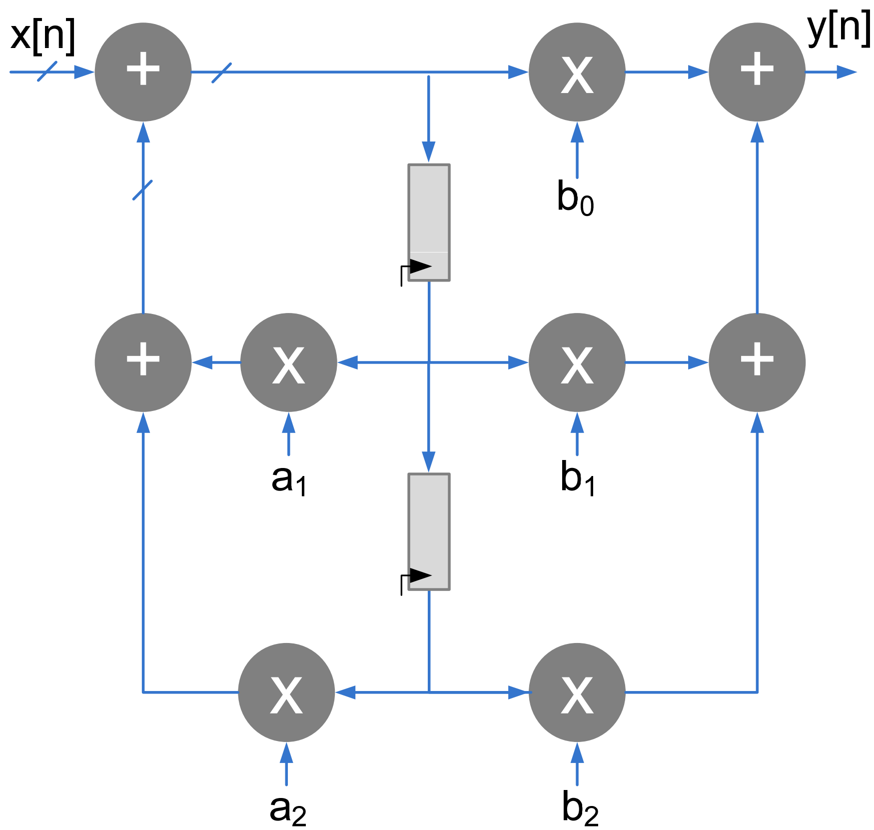




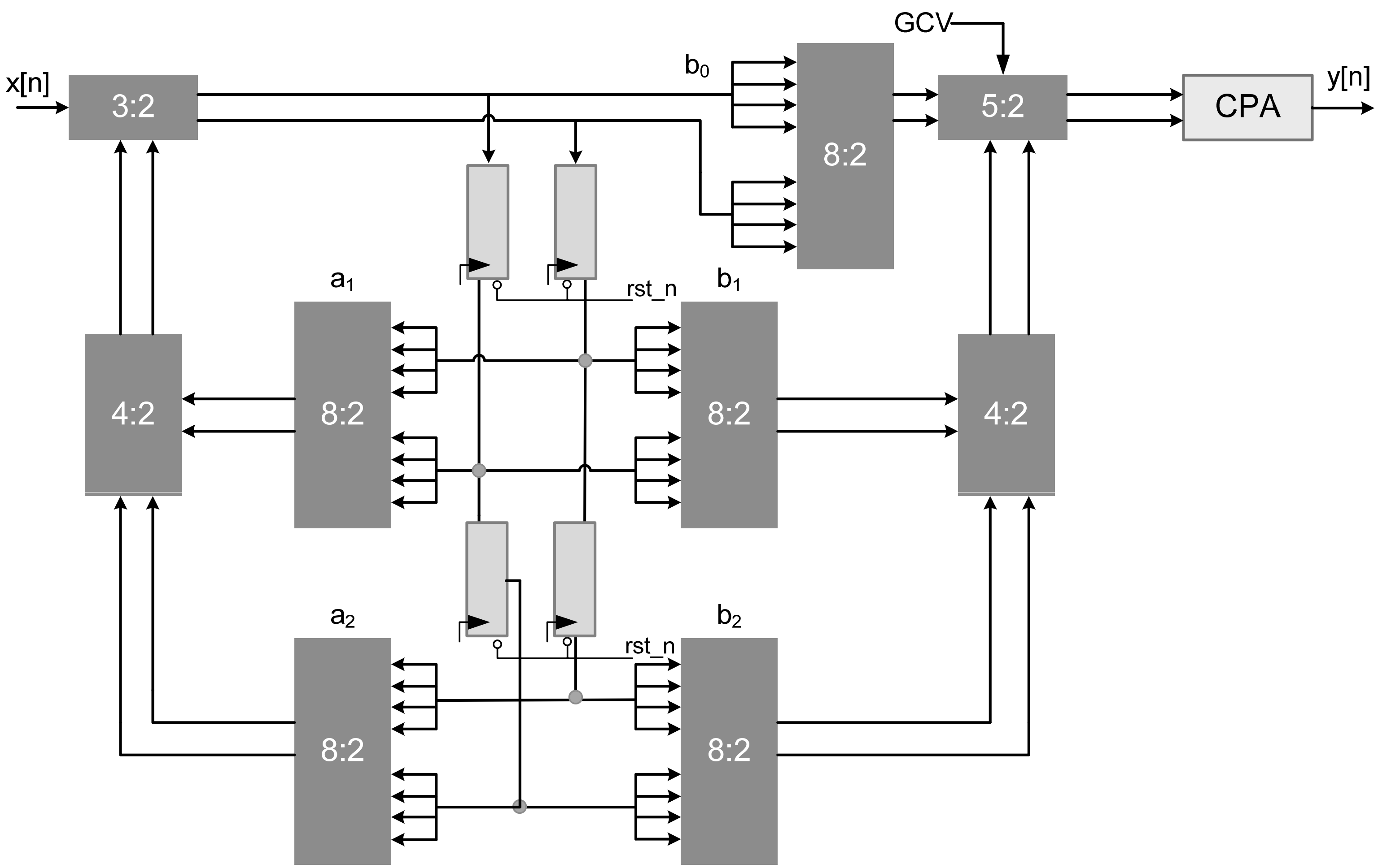
Transformed DFG with Wallace compression tree and CPA outside the feedback loop

Example 2: DFT 2nd Order IIR Filter

$$y[n] = a_1 y[n-1] + a_2 y[n-2] + b_0 x[n] + b_1 x[n-1] + b_2 x[n-2]$$

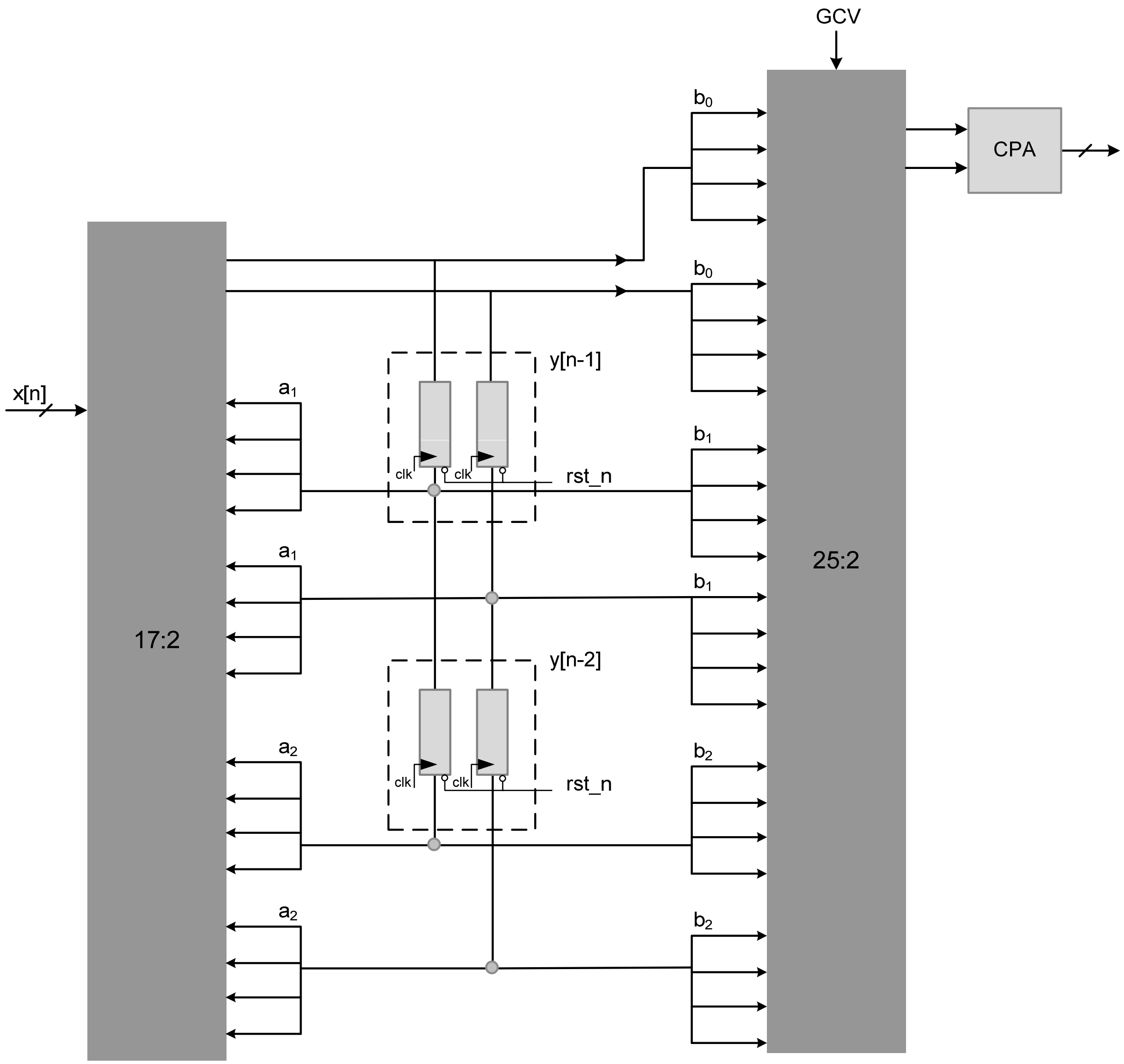


Example: Optimal Mapping: Design Option 1



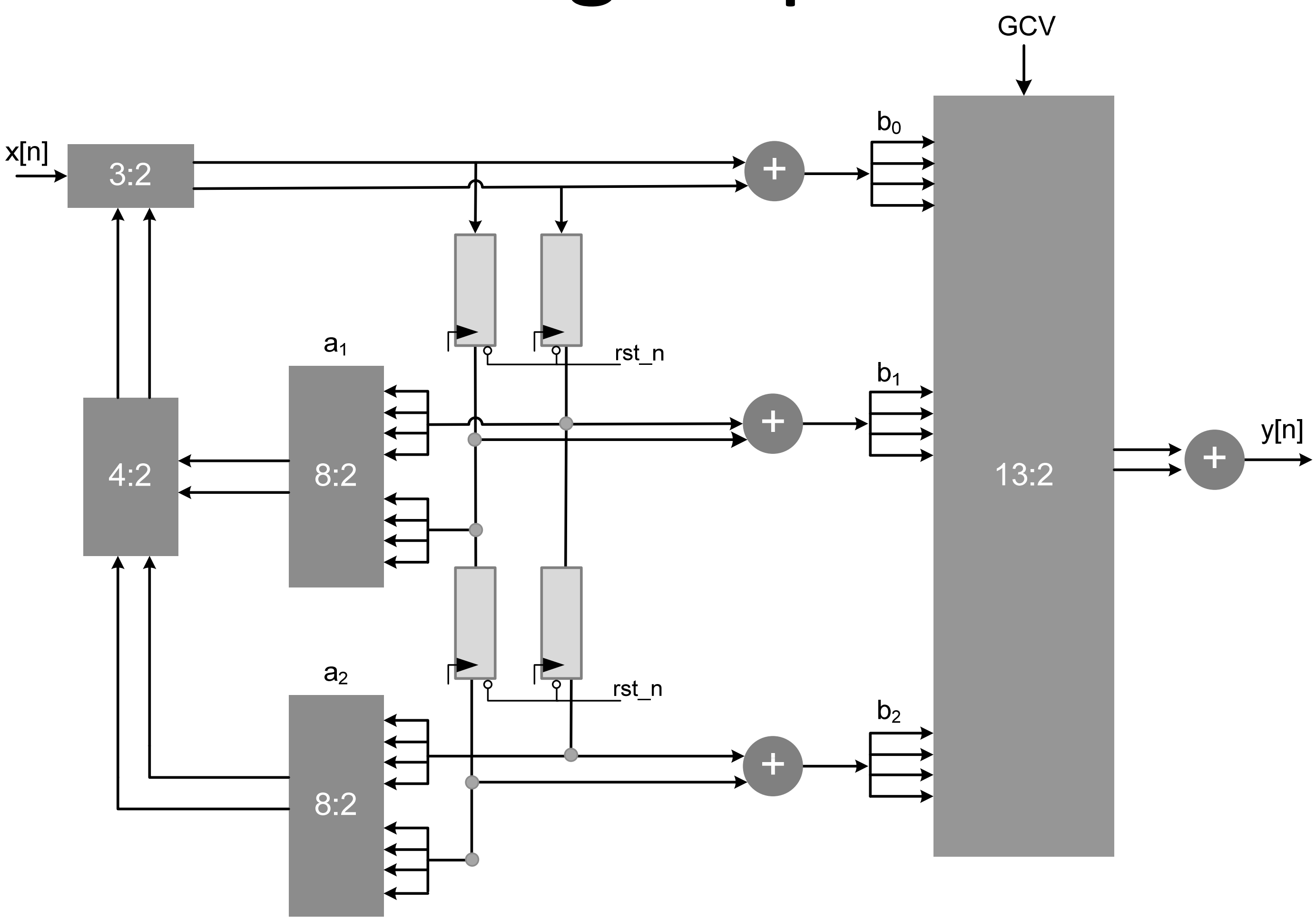
Optimized implementation with CSD multipliers, compression trees and CPA outside the IIR filter

Design Option 2



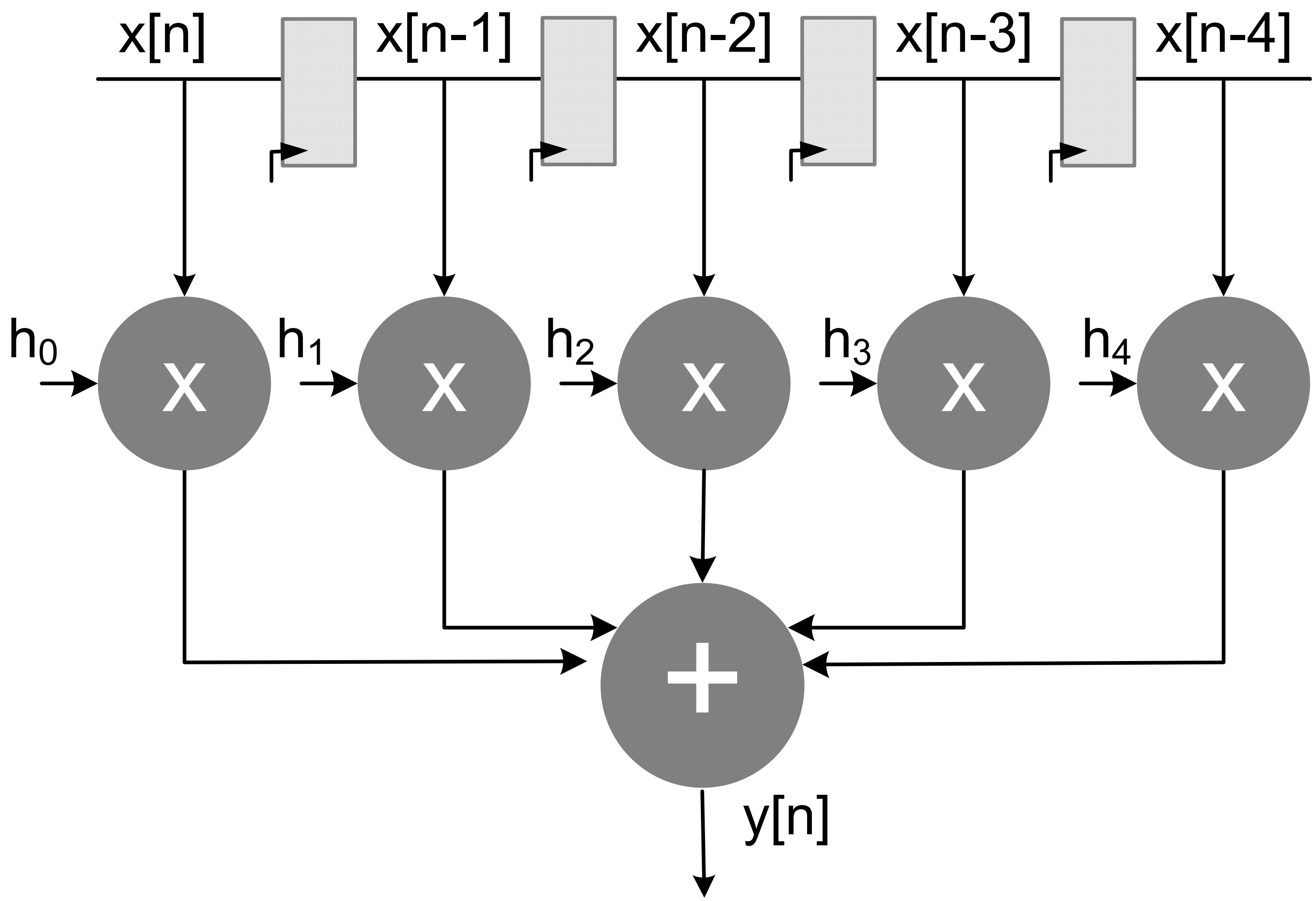
Using unified reduction trees for the feedforward and feedback computations and CPA outside the filter

Design Option 3

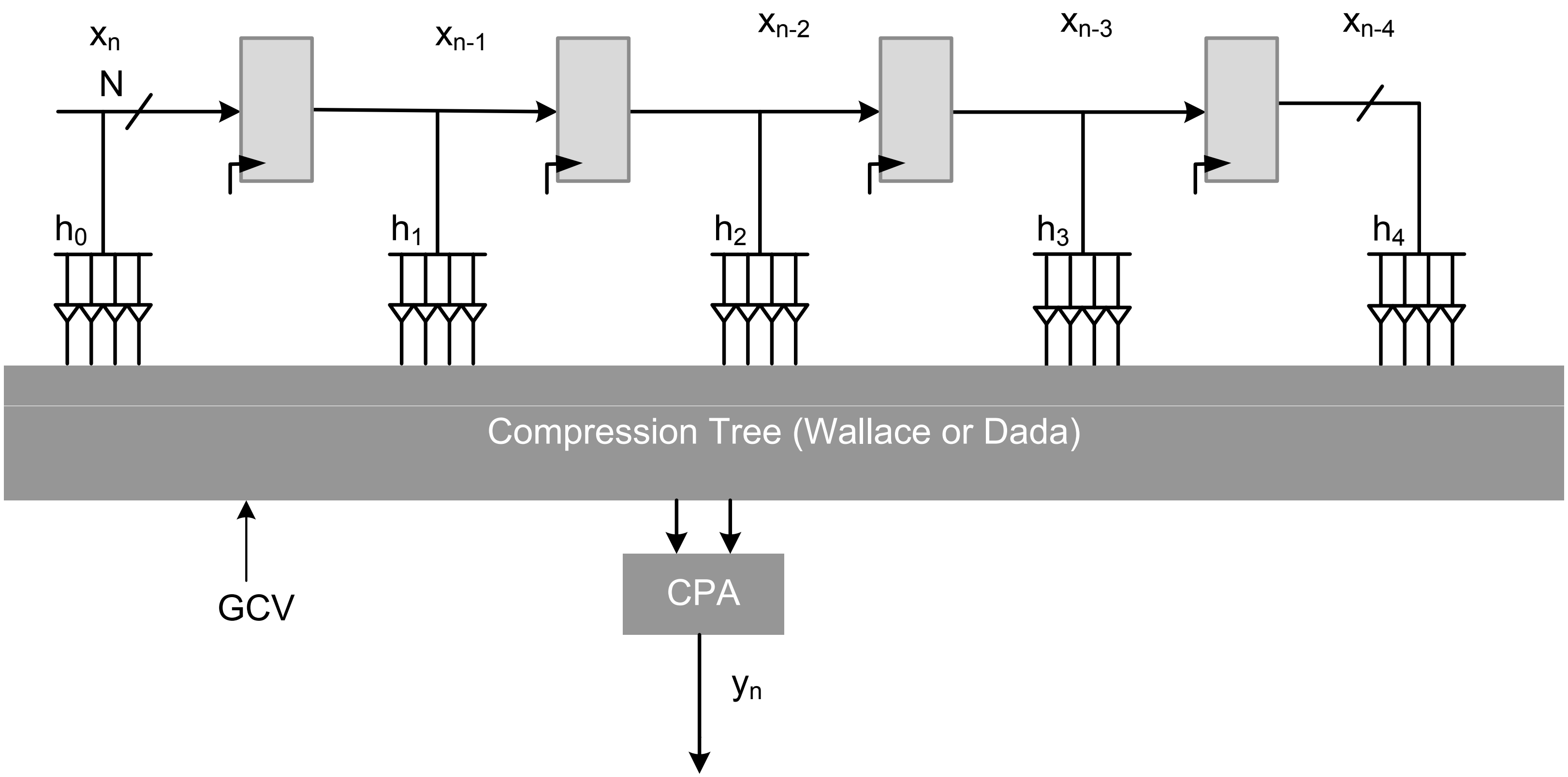


CPA outside the feedback loop

FIR Filter: Direct Form



All multiplications are implemented as one compression tree and a single CPA



Example: Conversion to Fixed-Point

h[n] = [0.0246 0.2344 0.4821 0.2344 0.0246]

h[n] = round(h[n]*2^{15}) =

[805 7680 15798 7680 805]

16’b0000_0110_0100_1010

16’b0011_1100_0000_0000

16’b0011_1101_1011_0110

16’b0011_1100_0000_0000

16’b0000_0110_0100_1010

Conversion in CSD

$$0000\_\bar{1}010\_0100\_1010$$

$$0100\_0\bar{1}00\_0000\_0000$$

$$0100\_00\bar{1}0\_0\bar{1}00\_\bar{1}0\bar{1}0$$

$$0100\_0\bar{1}00\_0000\_0000$$

$$0000\_\bar{1}010\_0100\_1010$$

Keeping maximum of 4 non-zero CSD in each coefficient results in

$$0000\_\bar{1}010\_0100\_1$$

$$0100\_0\bar{1}00\_0000\_0000$$

$$0100\_00\bar{1}0\_0\bar{1}00\_\bar{1}$$

$$0100\_0\bar{1}00\_0000\_0000$$

$$0000\_\bar{1}010\_0100\_1$$

Input to Compression Tree

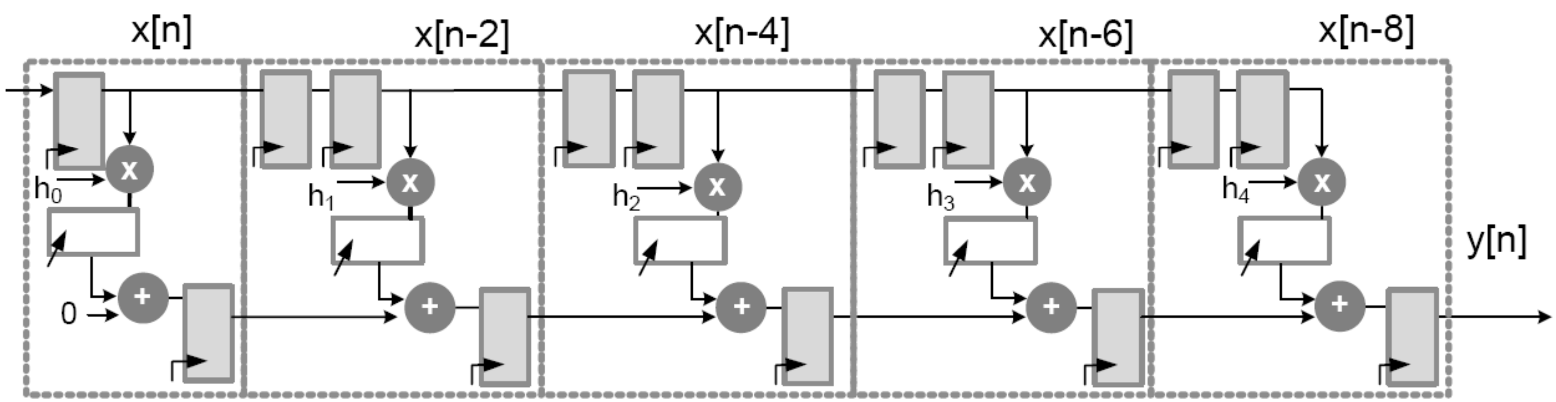
$$
\begin{aligned}
y[n] = (&-x[n]2^{-4} + x[n]2^{-6} + x[n]2^{-9} + x[n]2^{-12}) \\
&+ (x[n-1]2^{-1} - x[n]2^{-5}) \\
&+ (x[n-2]2^{-1} - x[n-2]2^{-6} - x[n-2]2^{-9} - x[n-2]2^{-12}) \\
&+ (x[n-3]2^{-1} - x[n-3]2^{-5}) \\
&+ (-x[n-4]2^{-4} + x[n-4]2^{-6} + x[n-4]2^{-9} + x[n-4]2^{-12})
\end{aligned}
$$

CV Computation for first CSD multiplier

$$0000010\bar{1}001001$$

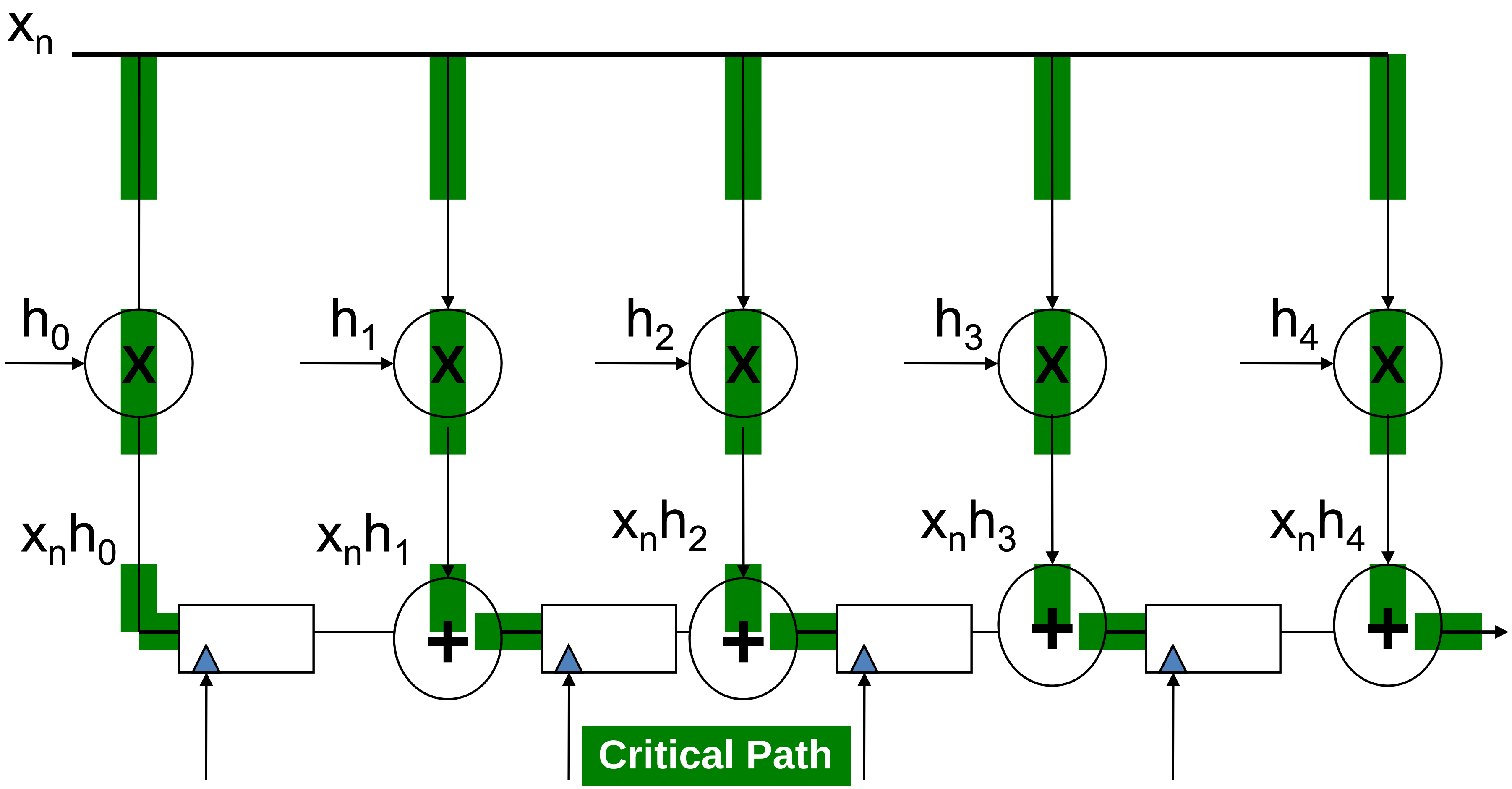
					1		1			1			1								
1	1	1	1	1	$\bar{x}_{15}$	x_{14}	x_{13}	x_{12}	x_{11}	x_{10}	x_9	x_8	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0	
1	1	1	1	1	1	1	x_{15}	$\bar{x}_{14}$	$\bar{x}_{13}$	$\bar{x}_{12}$	$\bar{x}_{11}$	$\bar{x}_{10}$	$\bar{x}_9$	$\bar{x}_8$	$\bar{x}_7$	$\bar{x}_6$	$\bar{x}_5$	$\bar{x}_4$	$\bar{x}_3$	$\bar{x}_2$	$\bar{x}_1$
1	1	1	1	1	1	1	1	1	1	$\bar{x}_{15}$	x_{14}	x_{13}	x_{12}	x_{11}	x_{10}	x_9	x_8	x_7	x_6	x_5	x_4
1	1	1	1	1	1	1	1	1	1	1	1	1	$\bar{x}_{15}$	x_{14}	x_{13}	x_{12}	x_{11}	x_{10}	x_9	x_8	x_7

Pipelined DF FIR Filter

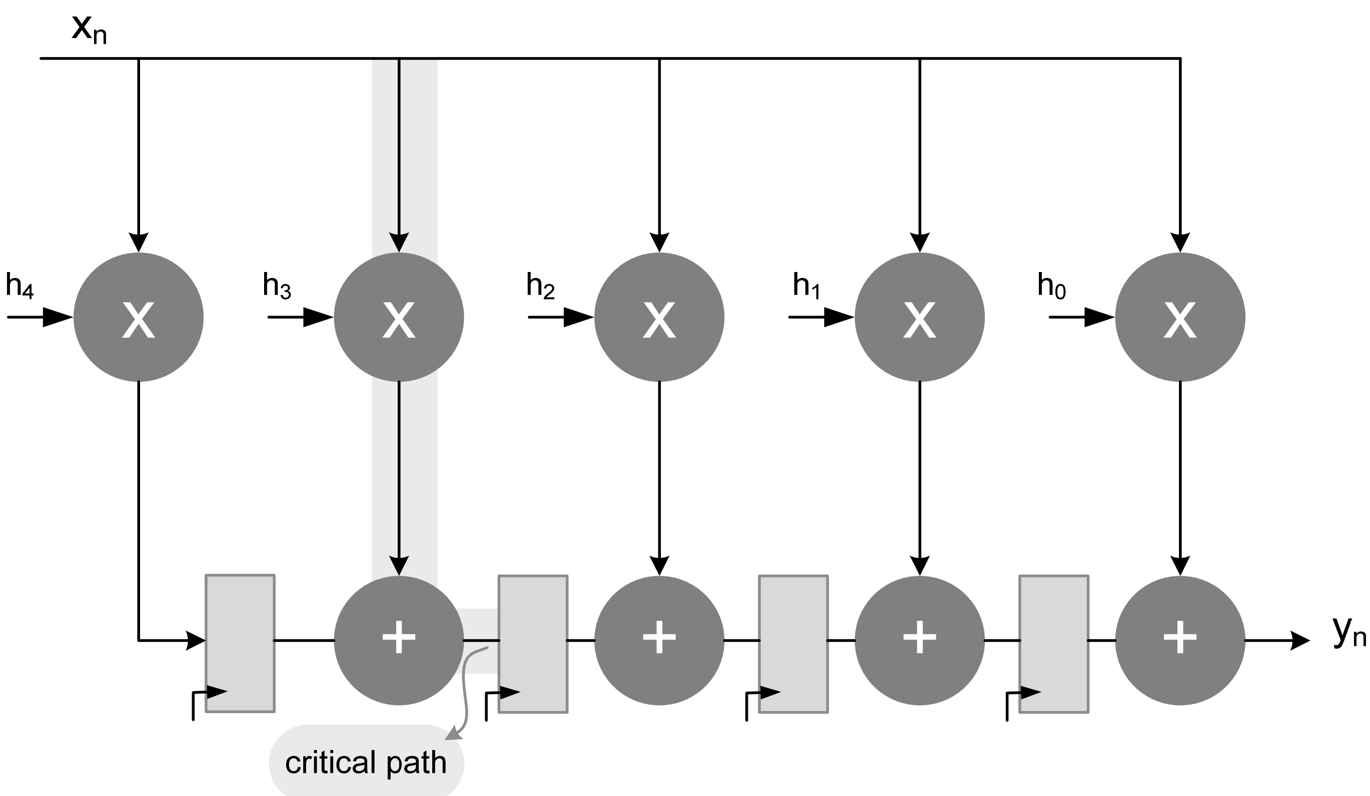


Pipeline direct form FIR filter for FPGAs with DSP48 blocks

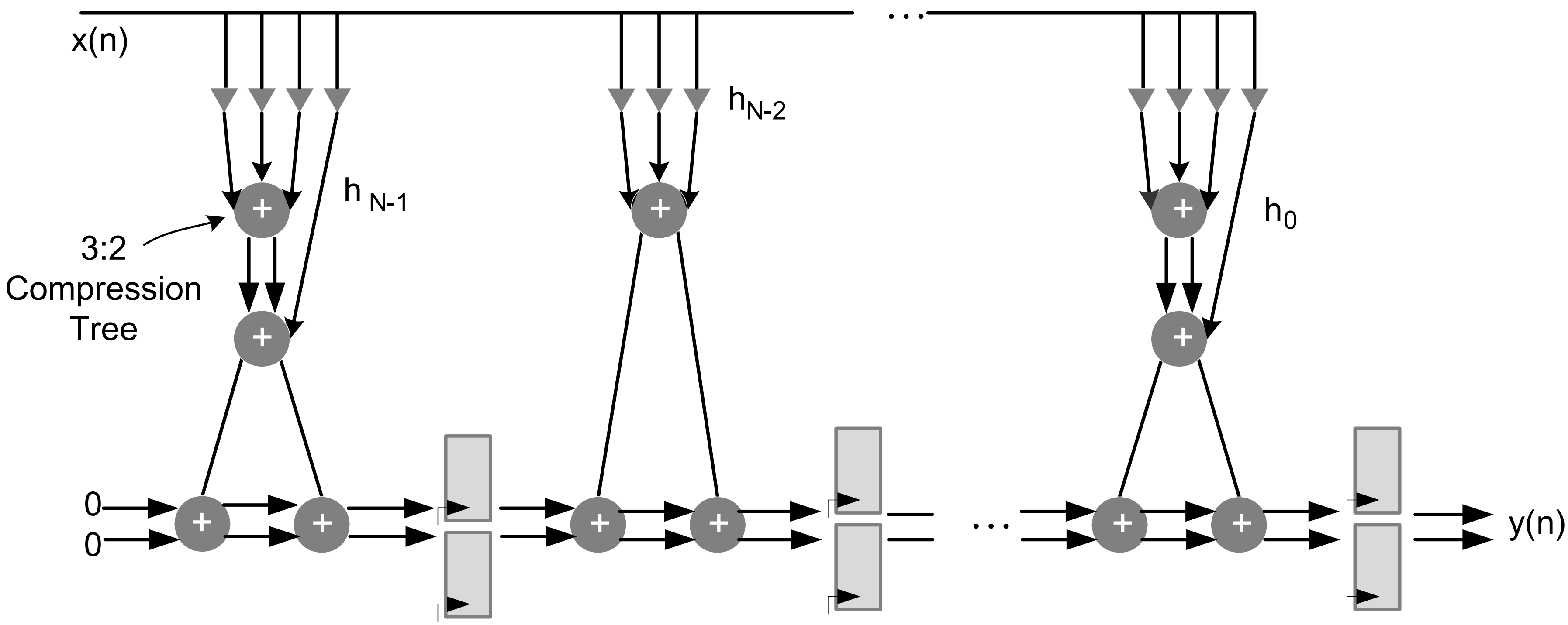
Transpose Direct Form FIR Filter



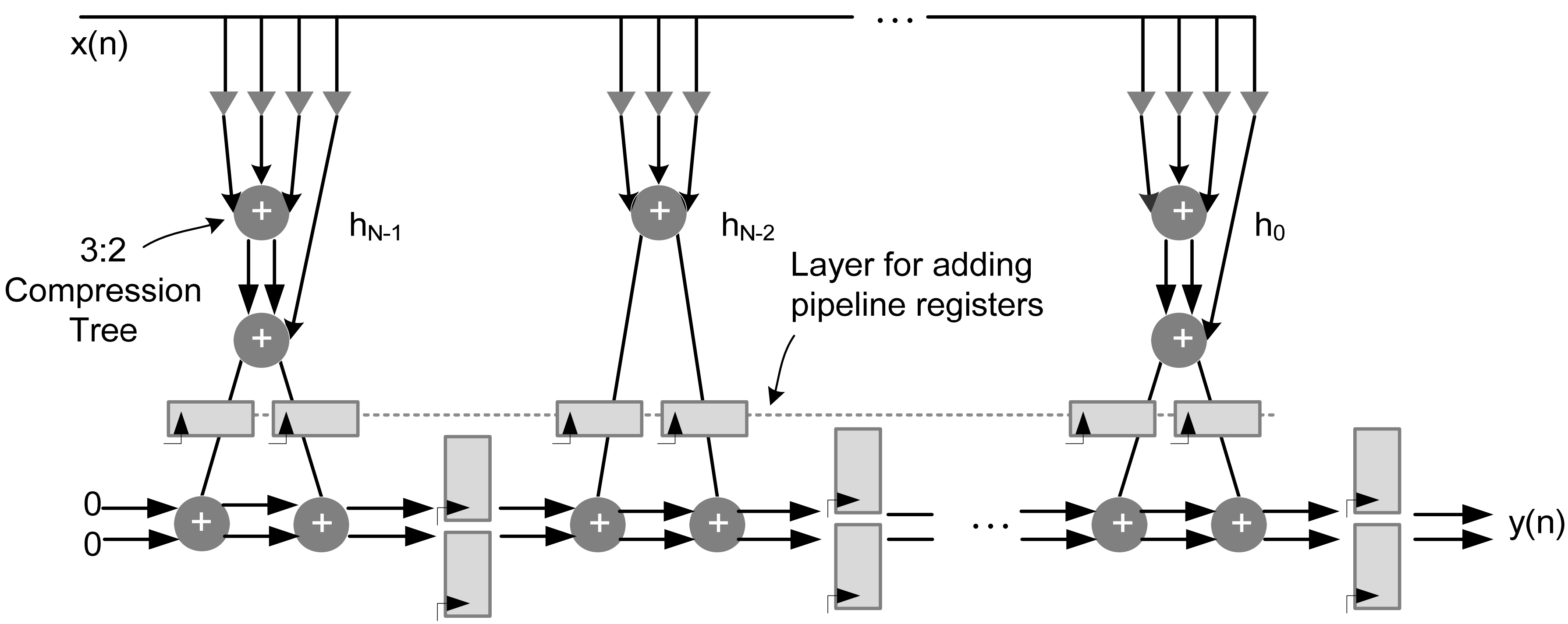
Critical Path



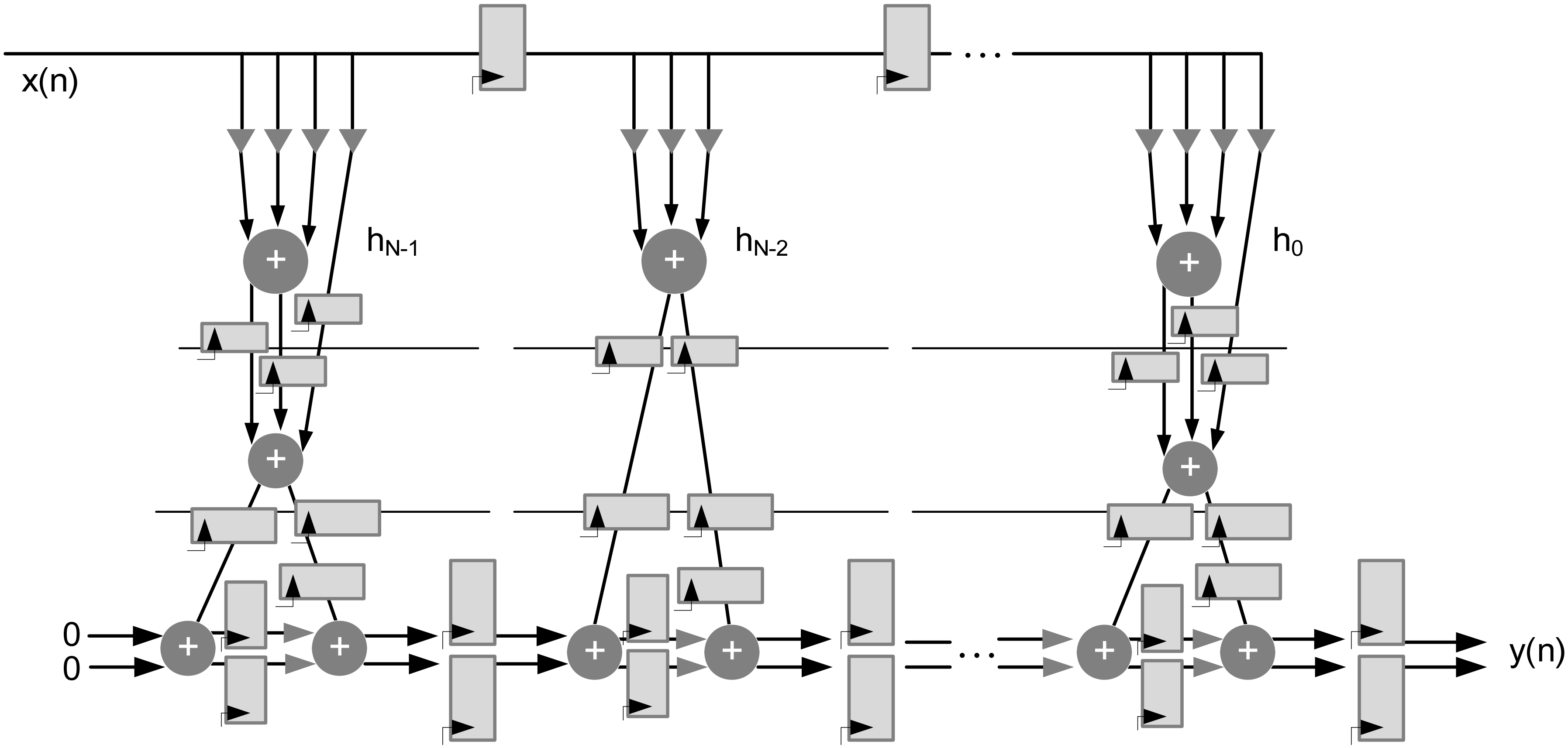
Filter Implementation



TD FIR with one stage of pipelining registers



Deeply pipelined TDF FIR filter with critical path equal to one full adder delay



Same Example

$$0000\_\bar{1}010\_0100\_1$$

$$0100\_0\bar{1}00\_0000\_0000$$

$$0100\_00\bar{1}0\_0\bar{1}00\_\bar{1}$$

$$0100\_0\bar{1}00\_0000\_0000$$

$$0000\_\bar{1}010\_0100\_1$$

TDF Implementation

$$M_4 = x[n]2^{5} - x[n]2^{7} + x[n]2^{10} + x[n]2^{13}$$

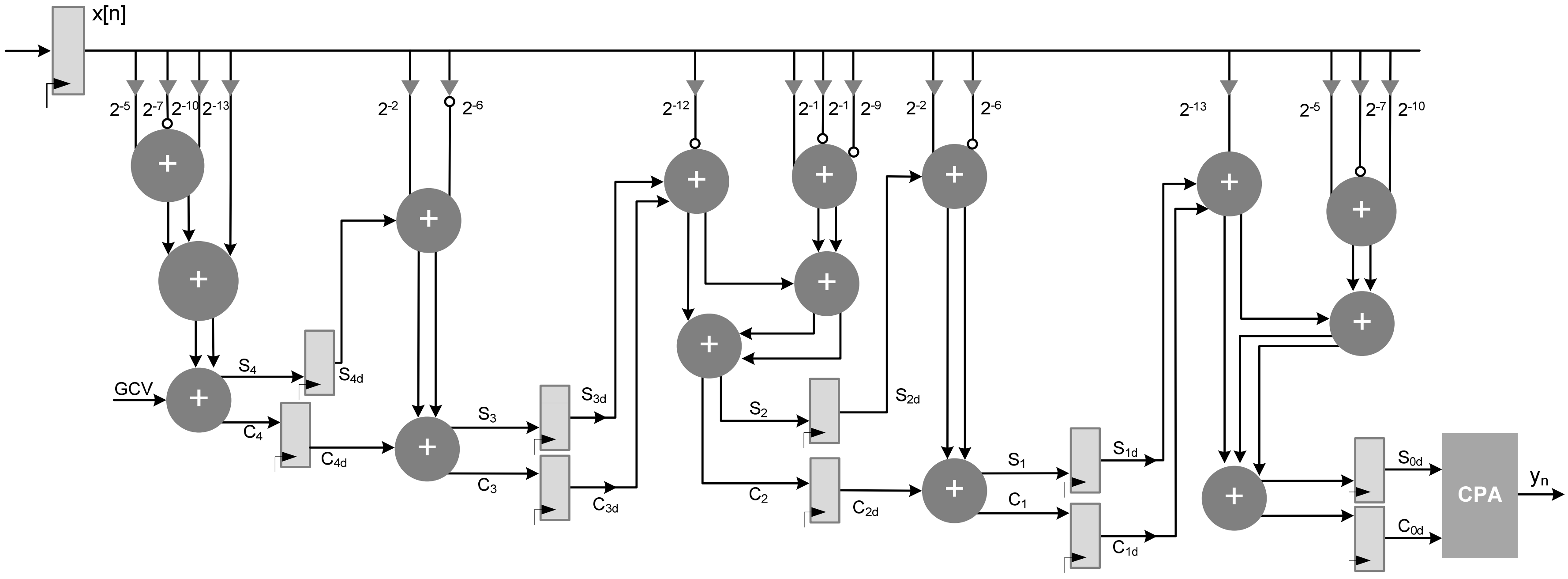
$$M_3 = x[n]2^{2} - x[n]2^{6}$$

$$M_2 = x[n]2^{1} - x[n]2^{6} - x[n]2^{9} - x[n]2^{12}$$

$$M_1 = x[n]2^{2} - x[n]2^{6}$$

$$M_0 = x[n]2^{5} - x[n]2^{7} + x[n]2^{10} + x[n]2^{13}$$

Example from the Book



$$\{c_4, s_4\} = x[n]2^{5} - x[n]2^{7} + x[n]2^{10} + x[n]2^{13} + 0 + 0$$

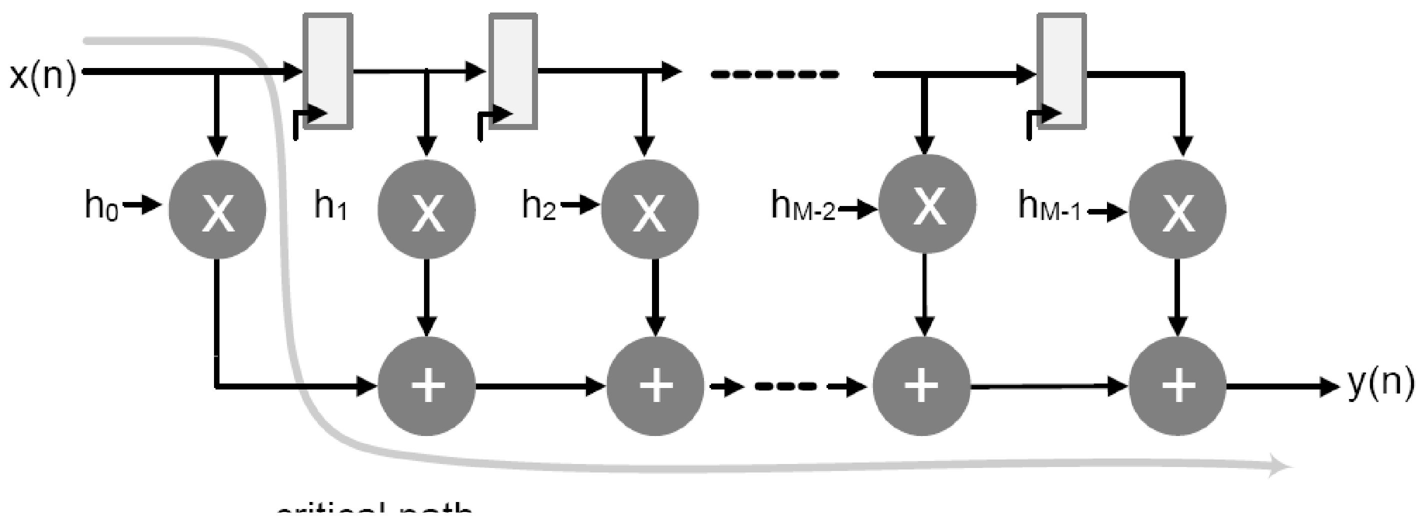
$$\{c_3, s_3\} = x[n]2^{2} - x[n]2^{6} + c_{4d} + s_{4d}$$

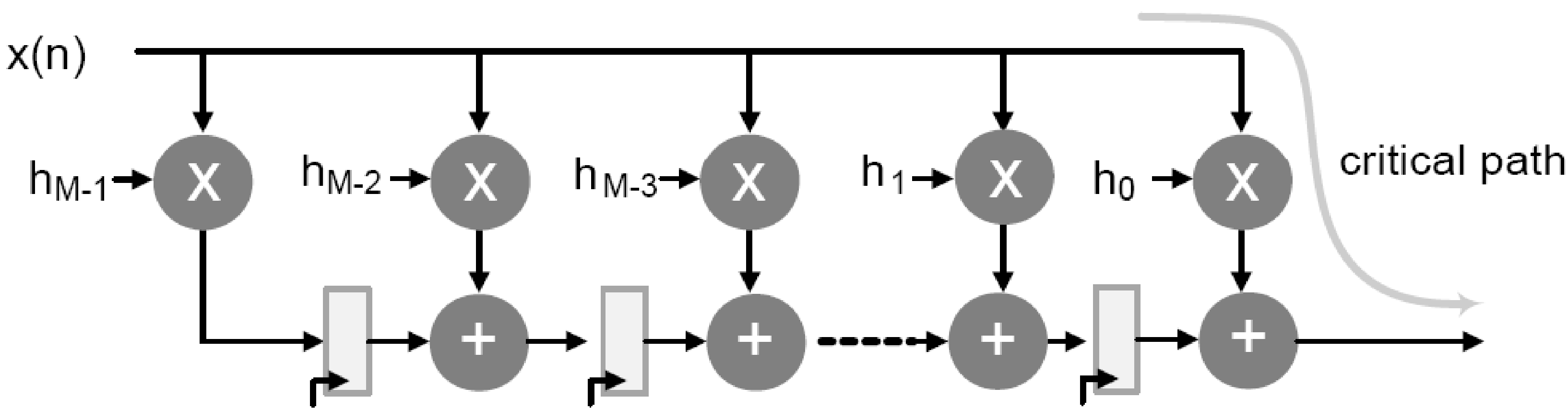
$$\{c_2, s_2\} = x[n]2^{1} - x[n]2^{6} - x[n]2^{9} - x[n]2^{12} + c_{3d} + s_{3d}$$

$$\{c_1, s_1\} = x[n]2^{2} - x[n]2^{6} + c_{2d} + s_{2d}$$

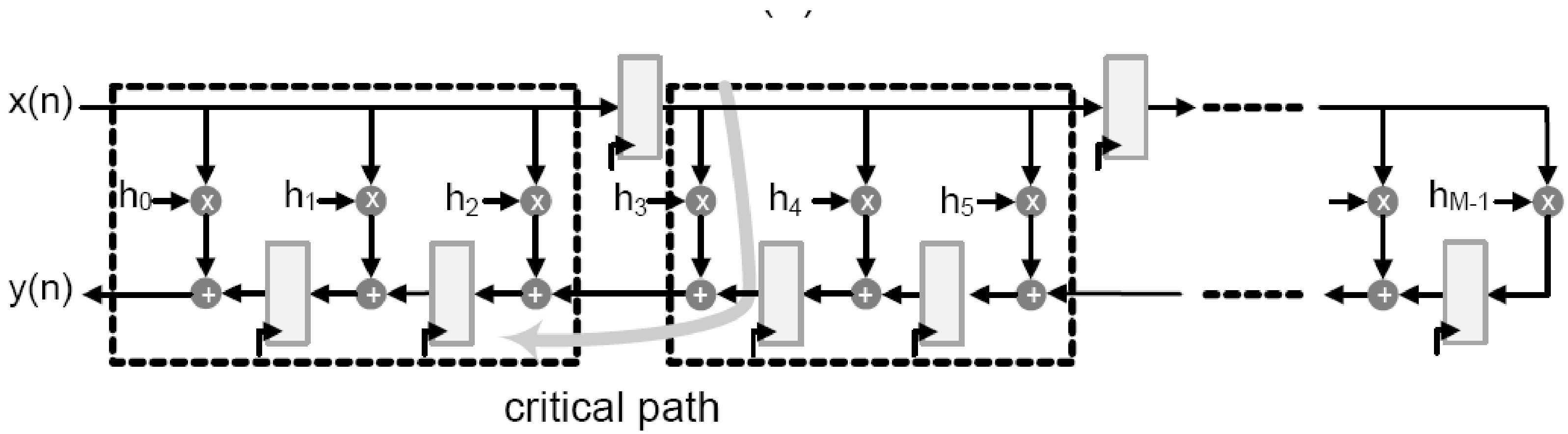
$$\{c_0, s_0\} = x[n]2^{5} - x[n]2^{7} + x[n]2^{10} + x[n]2^{13} + c_{1d} + s_{1d}$$

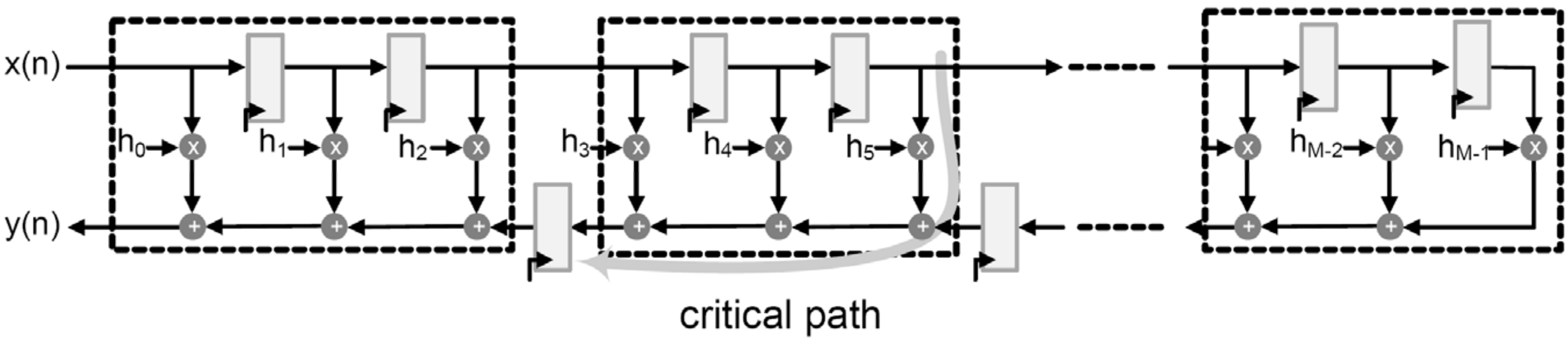
Hybrid FIR Filter Structure





Hybrid Designs





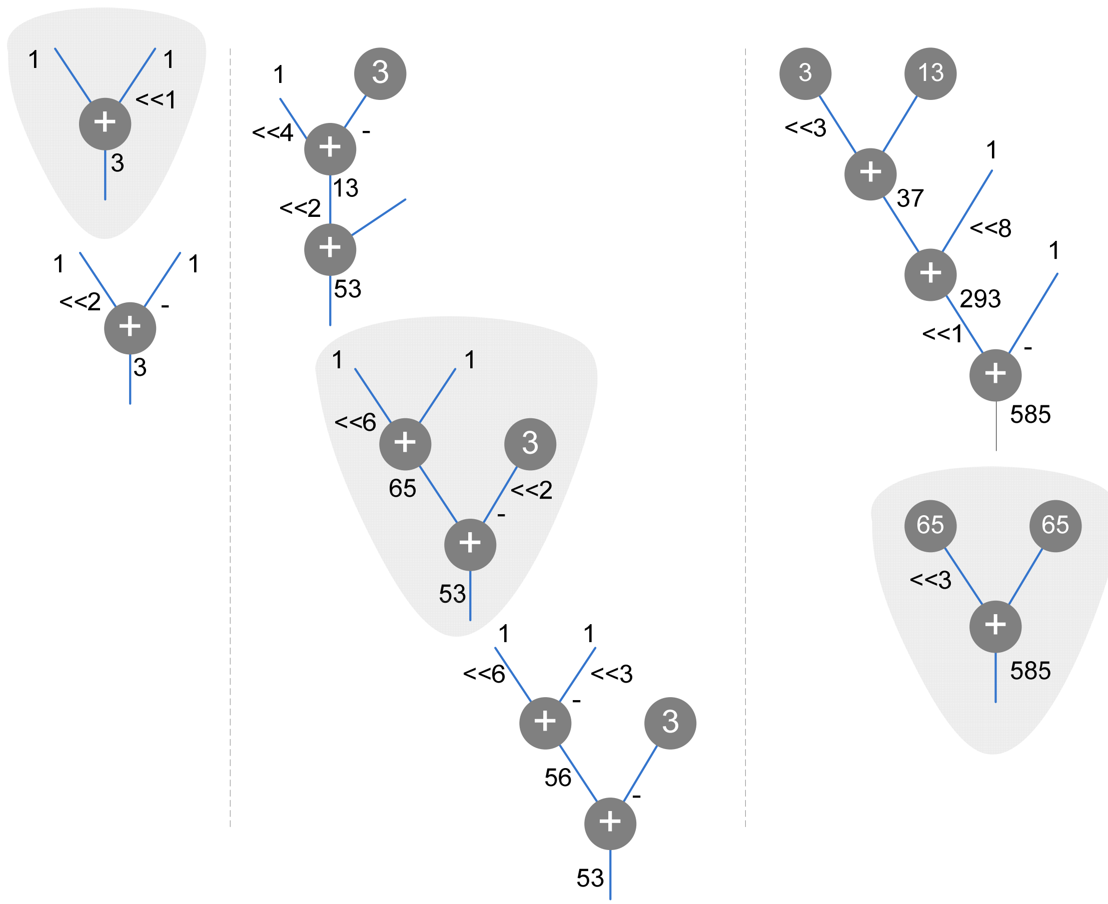
Complexity Reduction

ADV DSD CONTENTS

Complexity Reduction

- Constituent sub graphs that are shared in the original graph
- Example: three multipliers, 3, 53 and 585 with x

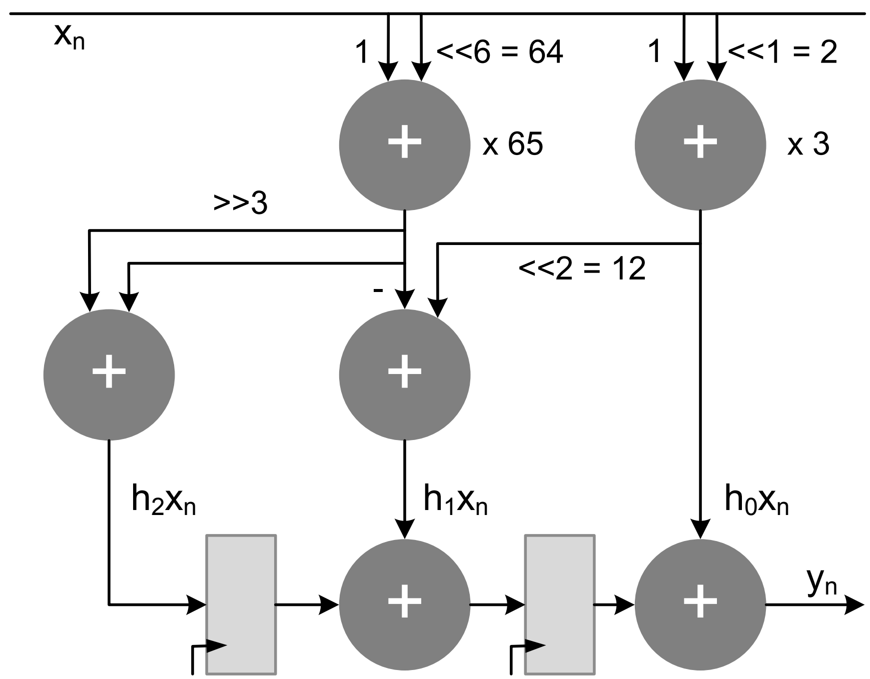
$$h_0 = 3 = 1 + 2^1$$
$$h_0 = 3 = 2^2 - 1$$
$$h_1 = 53 = 1 + 13 \times 2^2$$
$$13 = 2^4 - 3$$
$$h_1 = 53 = 65 - 3 \times 2^2$$
$$65 = 2^6 + 1$$
$$h_1 = 53 = 56 - 3$$
$$56 = 2^6 - 2^3$$
$$h_2 = 585 = 293 \times 2^1 - 1$$
$$293 = 2^8 + 37$$
$$37 = 3 \times 2^3 + 13$$
$$h_2 = 585 = 65 \times 2^3 + 65$$



$$
\begin{aligned}
h_0 &= 3 = 1 + 2^1 \\
h_0 &= 3 = 2^2 - 1 \\
h_1 &= 53 = 1 + 13 \times 2^2 \\
13 &= 2^4 - 3 \\
h_1 &= 53 = 65 - 3 \times 2^2 \\
65 &= 2^6 + 1 \\
h_1 &= 53 = 56 - 3 \\
56 &= 2^6 - 2^3 \\
h_2 &= 585 = 293 \times 2^1 - 1 \\
293 &= 2^8 + 37 \\
37 &= 3 \times 2^3 + 13 \\
h_2 &= 585 = 65 \times 2^3 + 65
\end{aligned}
$$

Optimized Implementation

- Selected sub-graphs from previous slide



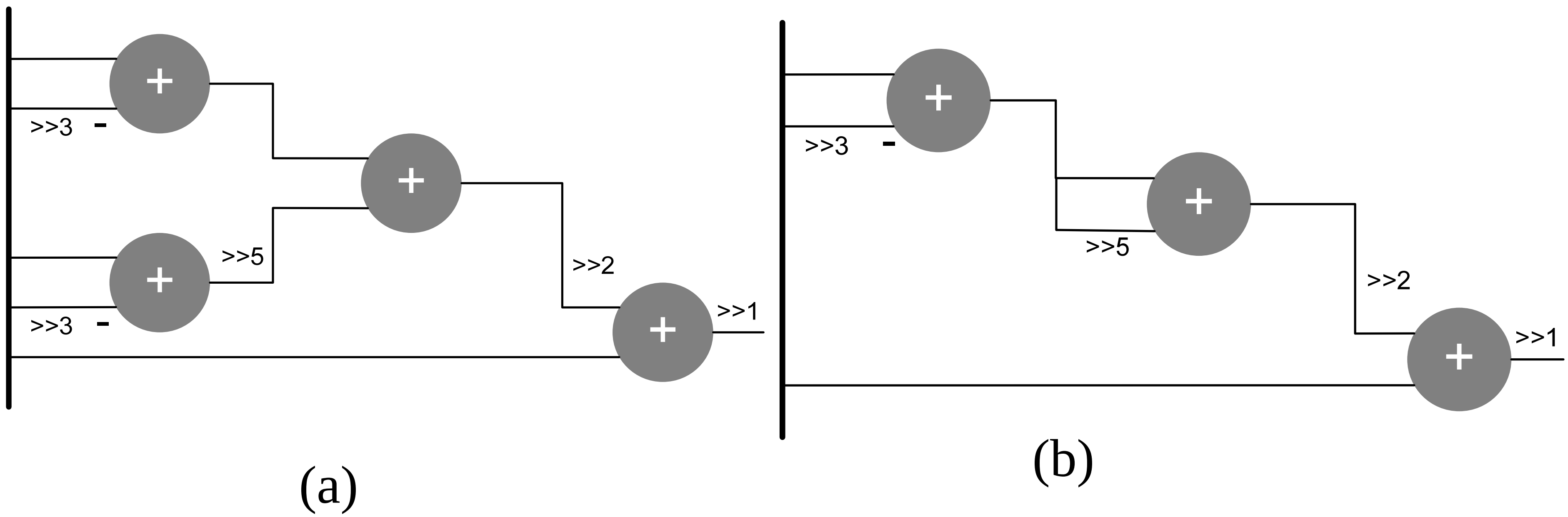
- Find common sub-expression
- Eliminate their re-use

$$h_0 x_n = (x_n \gg 1) + (x_n \gg 2) + (x_n \gg 3)$$
$$h_1 x_n = (x_n \gg 1) + (x_n \gg 3) + (x_n \gg 4)$$

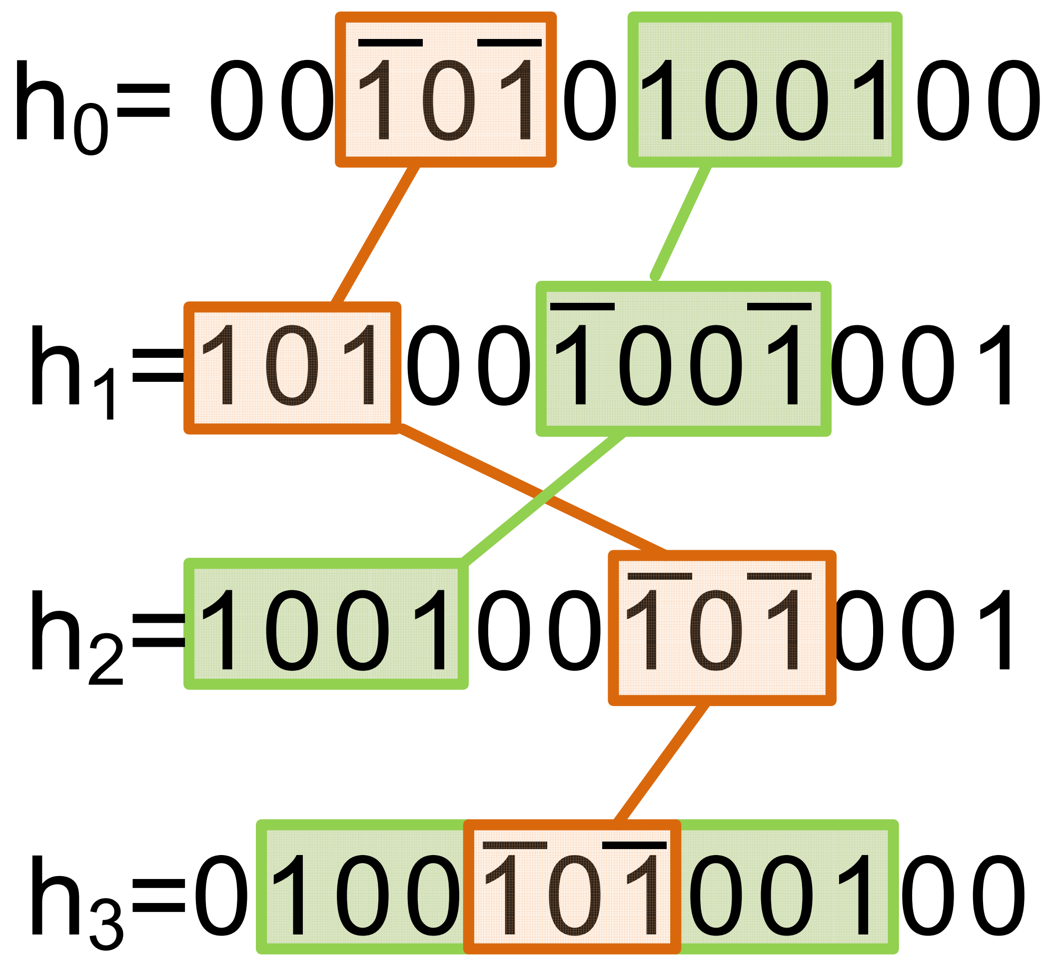
$$c_0 = (x_n \gg 1) + (x_n \gg 3)$$

$$h_0 x_n = c_0 + (x_n \gg 2)$$
$$h_1 x_n = c_0 + (x_n \gg 4)$$

Example: Common Sub-expression Elimination



Horizontal Common Sub-expressions for the example in the text



Vertical Sub-expressions Elimination

$$y_n = x_n z^{-3} h_3 + x_n z^{-2} h_2 + x_n z^{-1} h_1 + x_n h_0$$

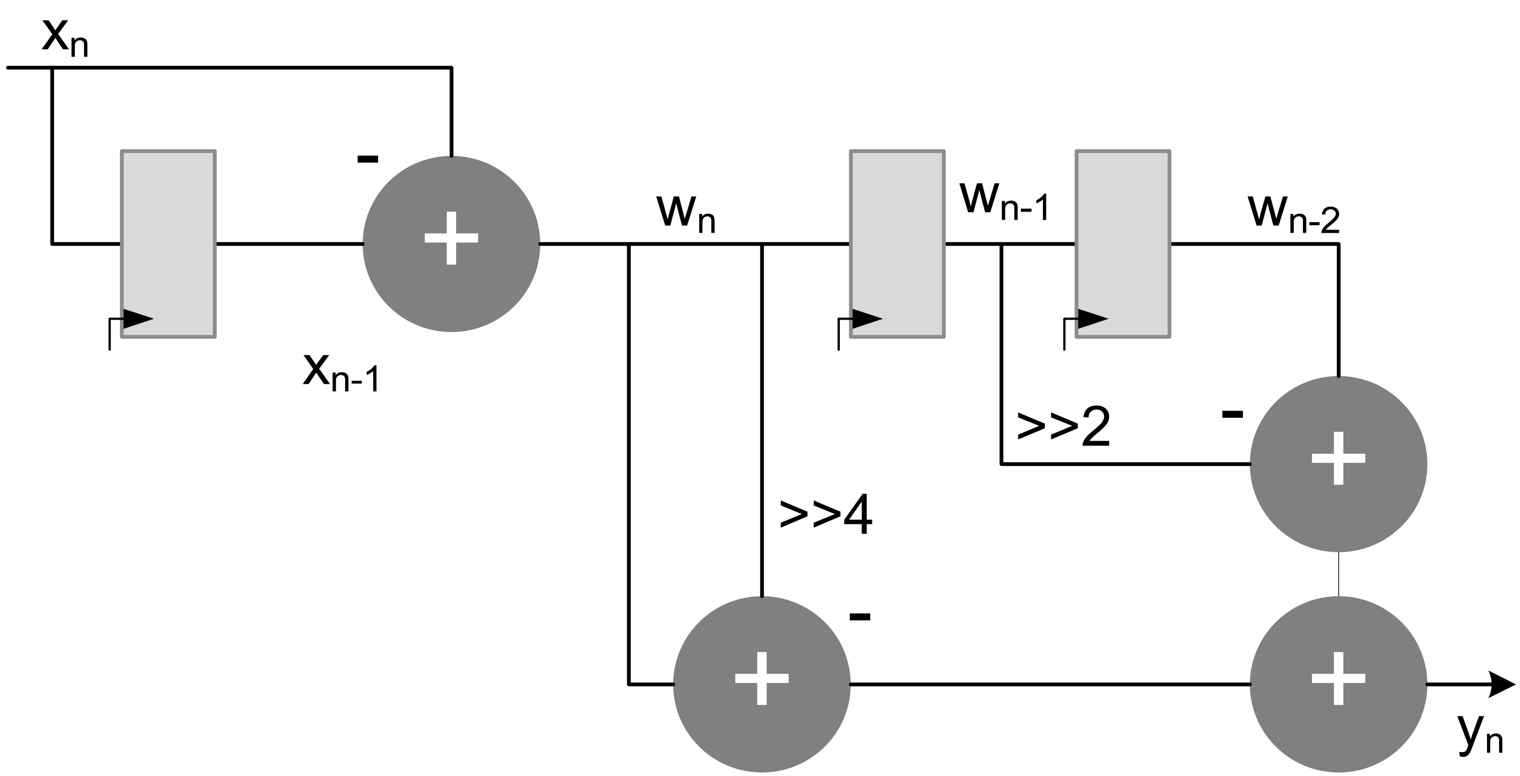
$$h_3 = 1 \quad 0 \quad 0 \quad 0 \quad 0$$
$$h_2 = \bar{1} \quad 0 \quad \bar{1} \quad 0 \quad 0$$
$$h_1 = 1 \quad 0 \quad 1 \quad 0 \quad \bar{1}$$
$$h_0 = \bar{1} \quad 0 \quad 0 \quad 0 \quad 1$$

$$\begin{aligned} y_n = \; & x_n z^{-3} \\ & -x_n z^{-2} \quad -x_n z^{-2} 2^{-2} \\ & +x_n z^{-1} \quad +x_n z^{-1} 2^{-2} \quad -x_n z^{-1} 2^{-4} \\ & -x_n \quad +x_n 2^{-4} \end{aligned}$$

Common Sub Expression

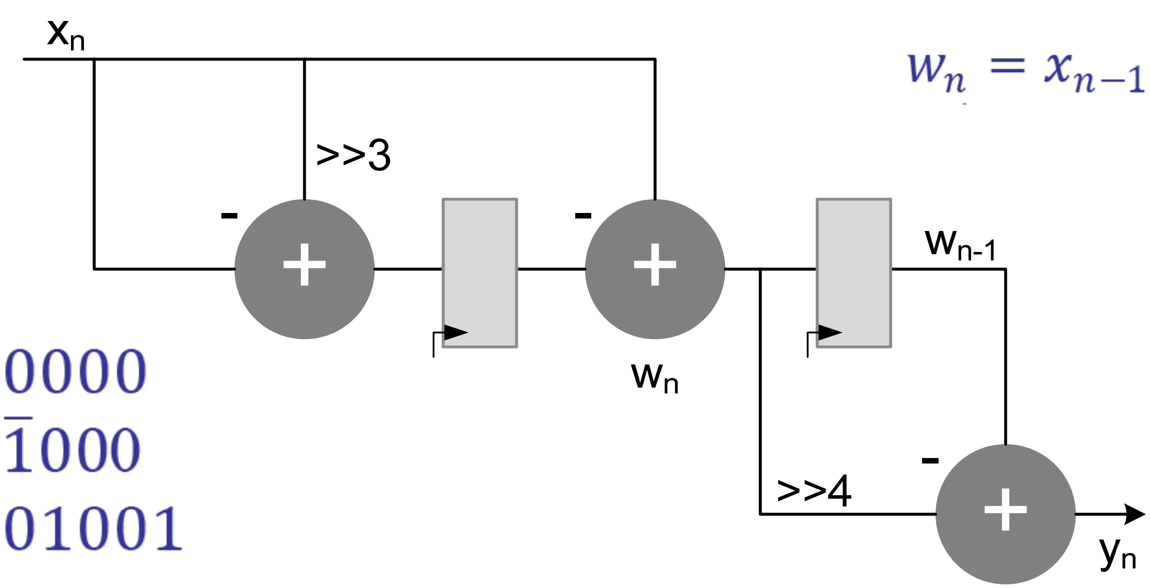
$$y_n = x_{n-1}z^{-2} - x_n z^{-2} + x_{n-1} - x_n - x_{n-1}z^{-1}2^{-2} + x_n z^{-1}2^{-2} - x_{n-1}2^{-4} + x_n 2^{-4}$$

Optimized implementation exploiting vertical common sub-expressions



Example of horizontal and vertical sub-expressions elimination

$$y_n = x_{n-1}z^{-1} - x_{n-1}z^{-1}2^{-3} - x_n z^{-1} - x_{n-1}2^{-4} + x_{n-1}2^{-7} + x_n 2^{-4}$$



$$w_n = x_{n-1} - x_{n-1}2^{-3} - x_n$$

$$h_2 = 100\bar{1}0000$$
$$h_1 = \bar{1}000\bar{1}000$$
$$h_0 = 000001001$$

Distributed Arithmetic Based Design

- Yet another way of looking at dot product design

$$y = \sum_{k\ \ 0}^{K\ \ 1} A_k x_k$$

$$y = \sum_{k\ \ 0}^{K\ \ 1} \left(-x_{k0} 2^0 + \sum_{b\ \ 1}^{N\ \ 1} x_{kb} 2^{\ \ b} \right) A_k$$

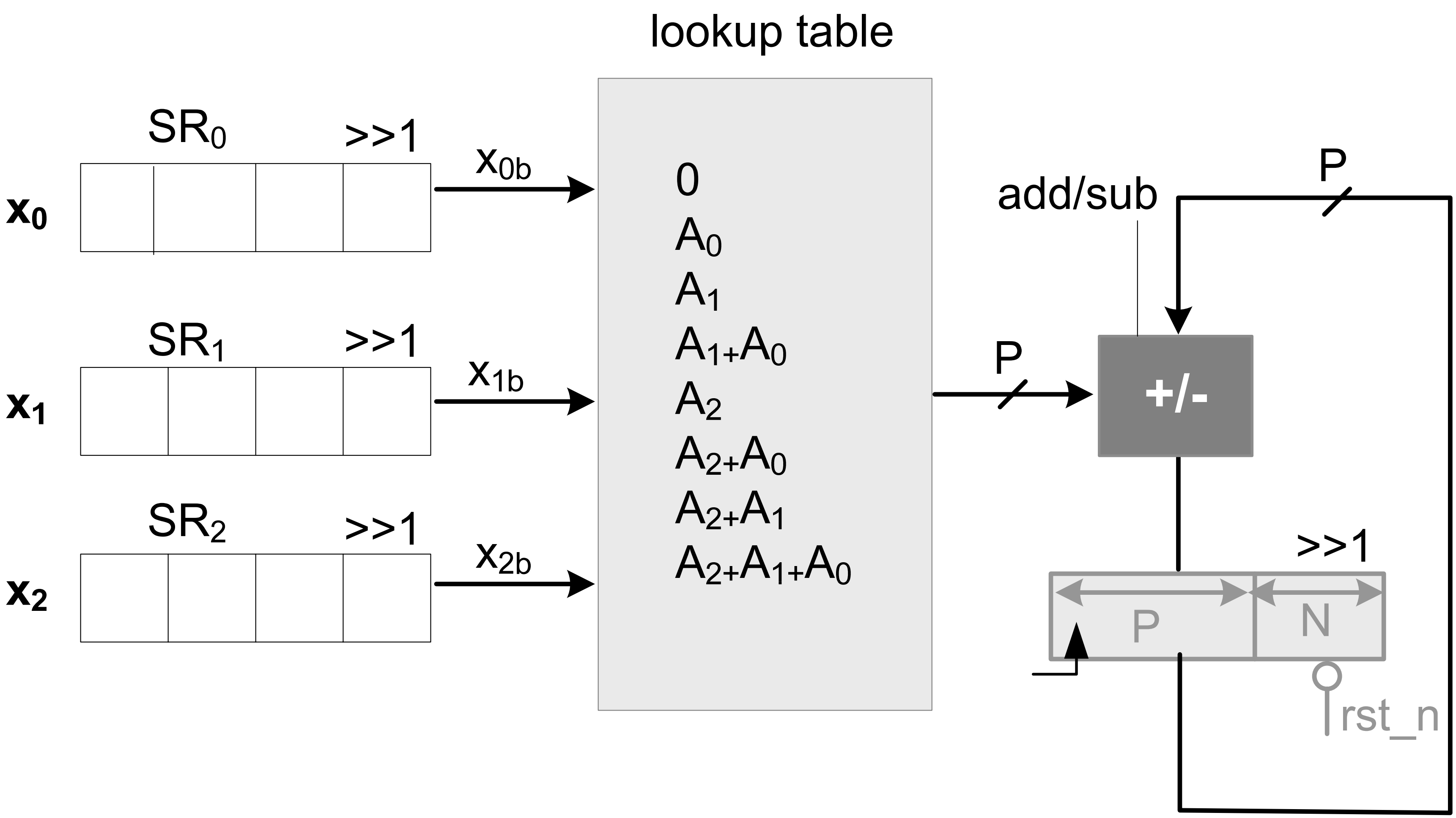
$$y = \sum_{k\ \ 0}^{K\ \ 1} (-x_{k0} 2^0 + x_{k1} 2^{\ \ 1} + \cdots x_{k(N\ \ 1)} 2^{N\ \ 1}) A_k$$

$$\begin{aligned} & -(x_{00}A_0 + x_{10}A_1 + x_{20}A_2)2^0 \\ & + (x_{01}A_0 + x_{11}A_1 + x_{21}A_2)2^{\ \ 1} \\ & + (x_{02}A_0 + x_{12}A_1 + x_{22}A_2)2^{\ \ 2} \\ & + (x_{03}A_0 + x_{13}A_1 + x_{23}A_2)2^{\ \ 3} \end{aligned}$$

ROM for Distributed Arithmetic

x_{2b}	x_{1b}	x_{0b}	Contents of ROM
0	0	0	0
0	0	1	A_0
0	1	0	A_1
0	1	1	$A_1 + A_0$
1	0	0	A_2
1	0	1	$A_2 + A_0$
1	1	0	$A_2 + A_1$
1	1	1	$A_2 + A_1 + A_0$

DA for computing the dot product of integer numbers for N=4 and K=3

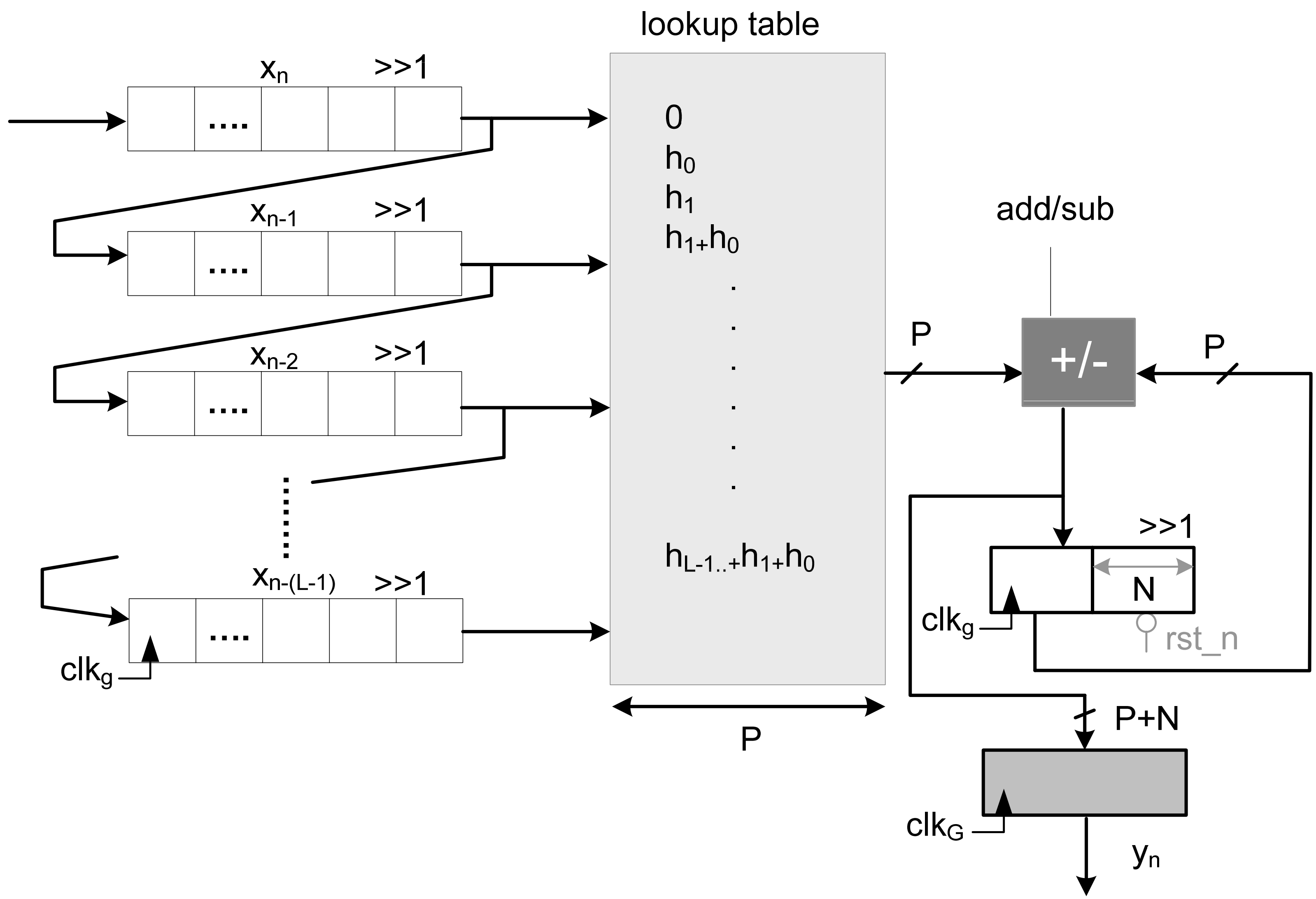


Look-up table

$A_0 = 3, A_1 = -1$ and $A_2 = 5$

x_{2b}	x_{1b}	x_{0b}	Contents of ROM	
0	0	0	0	0
0	0	1	A_0	3
0	1	0	A_1	-1
0	1	1	$A_1 + A_0$	2
1	0	0	A_2	5
1	0	1	$A_2 + A_0$	8
1	1	0	$A_2 + A_1$	4
1	1	1	$A_2 + A_1 + A_0$	7

DA-based architecture for implementing an FIR filter of length L and N-bit data samples

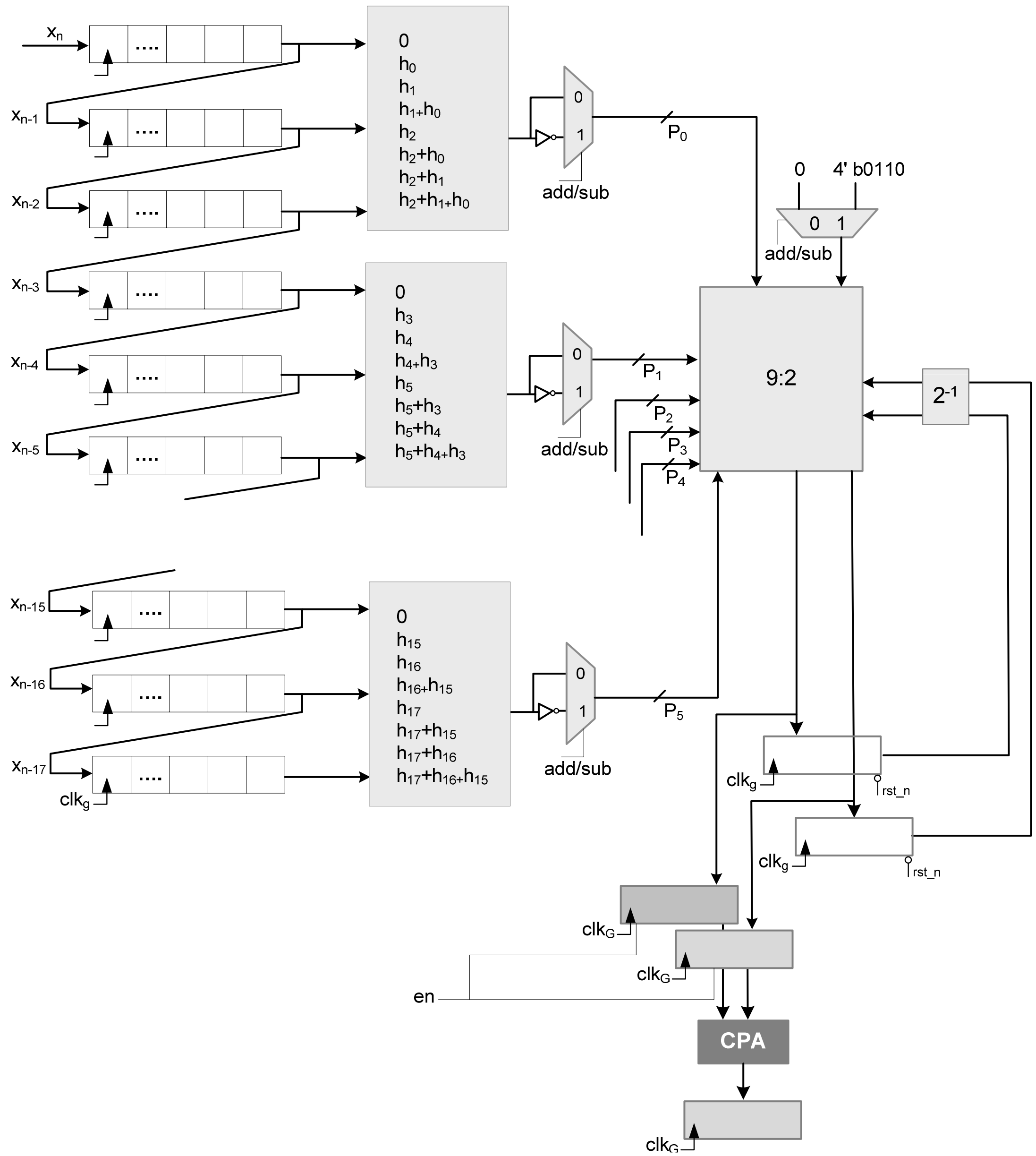


Cycle by cycle working of DA

$$x_0 = -6 = 4'b1010$$
$$x_1 = 6 = 4'b0110$$
$$x_2 = -5 = 4'b1011$$

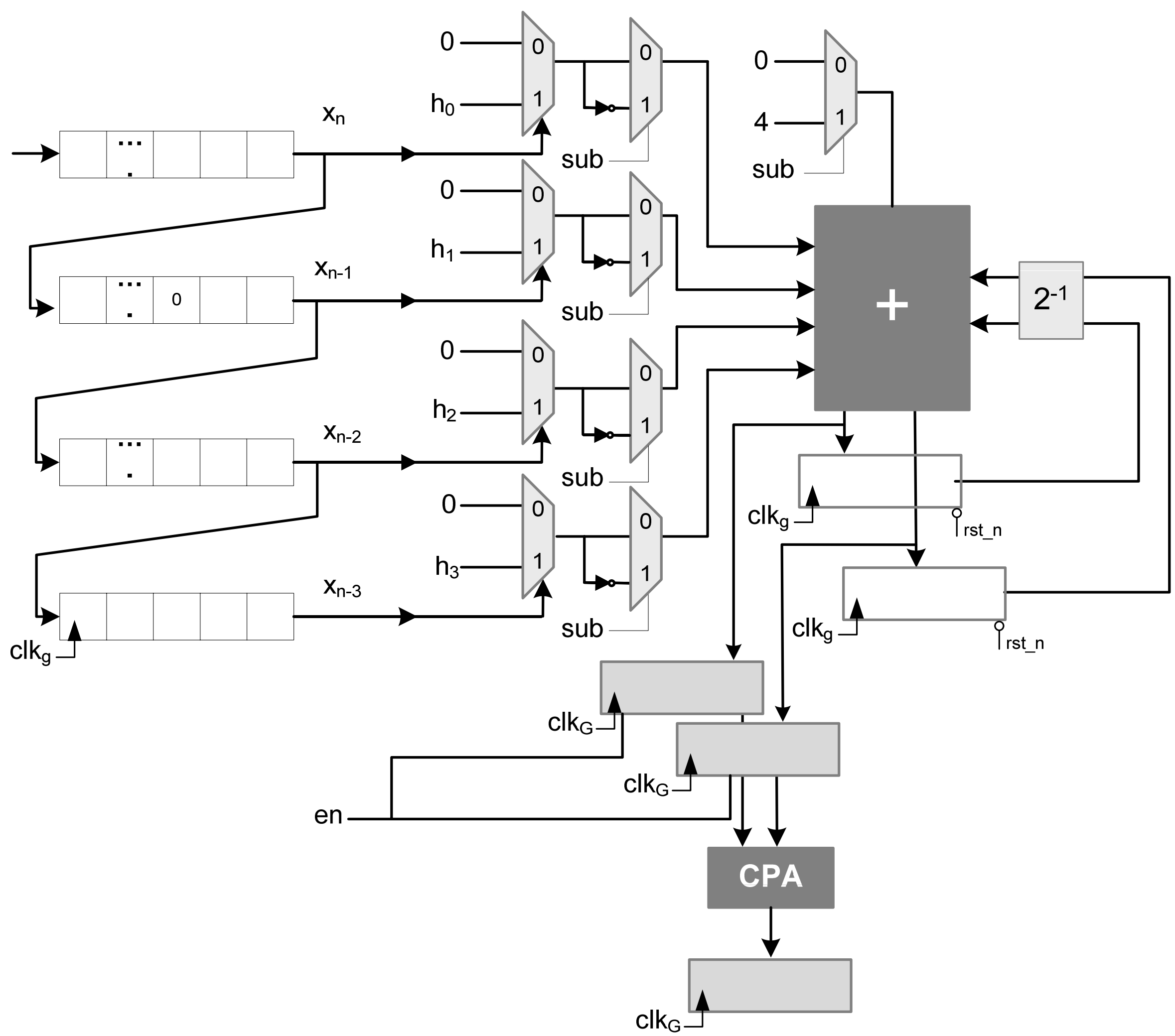
Cycle	Address	LUT	Accumulator
0	3’b100	5	000101_000
1	3’b111	7	001001_100
2	3’b000	-1	000011_110
3	3’b101	8	111001_111

DA-based parallel implementation of an 18-coefficient FIR filter setting L=3 and M=6

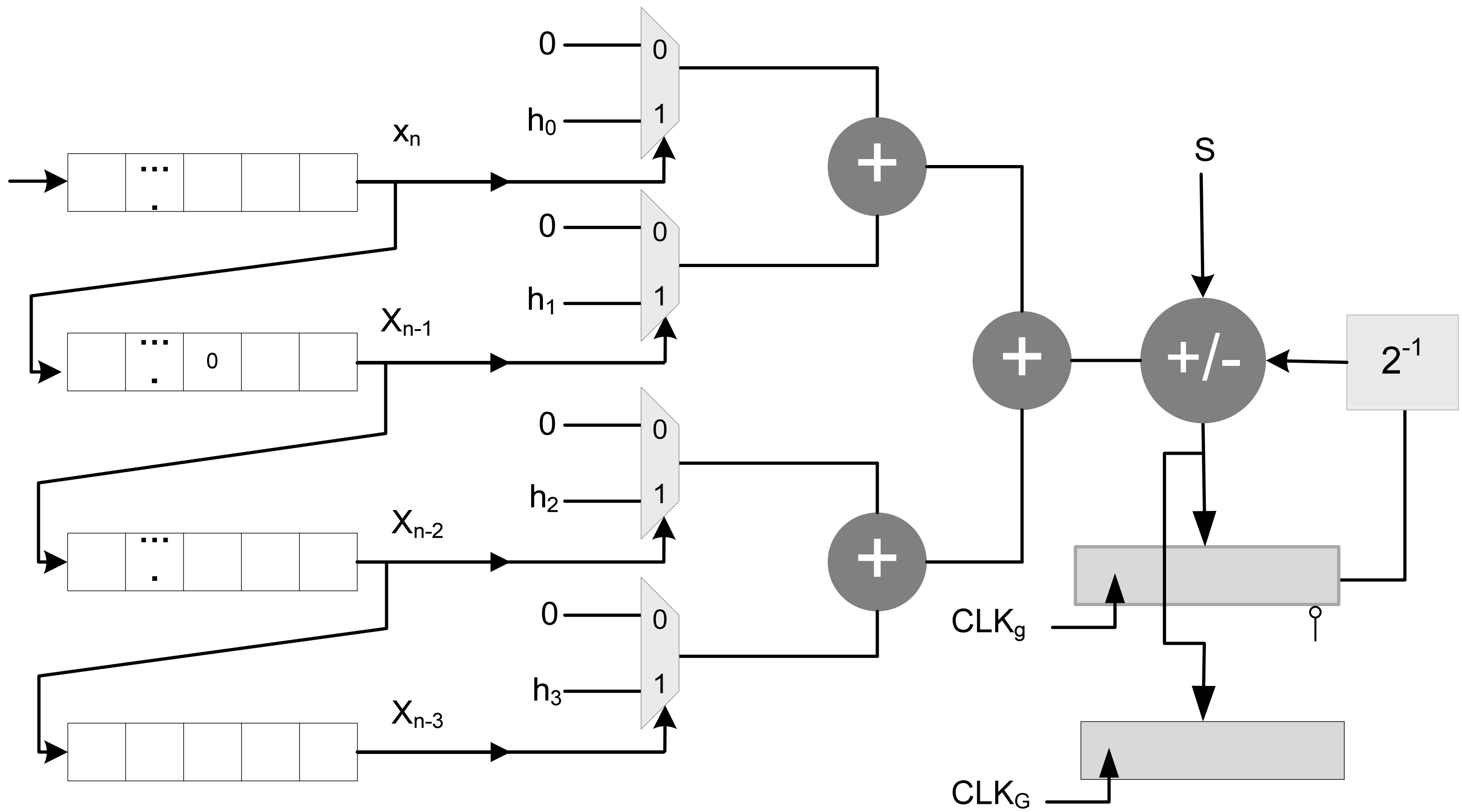


A LUT-less implementation of a DA-based FIR filter

A parallel implementation for M=K uses a 2:1 MUX, compression tree and a CPA

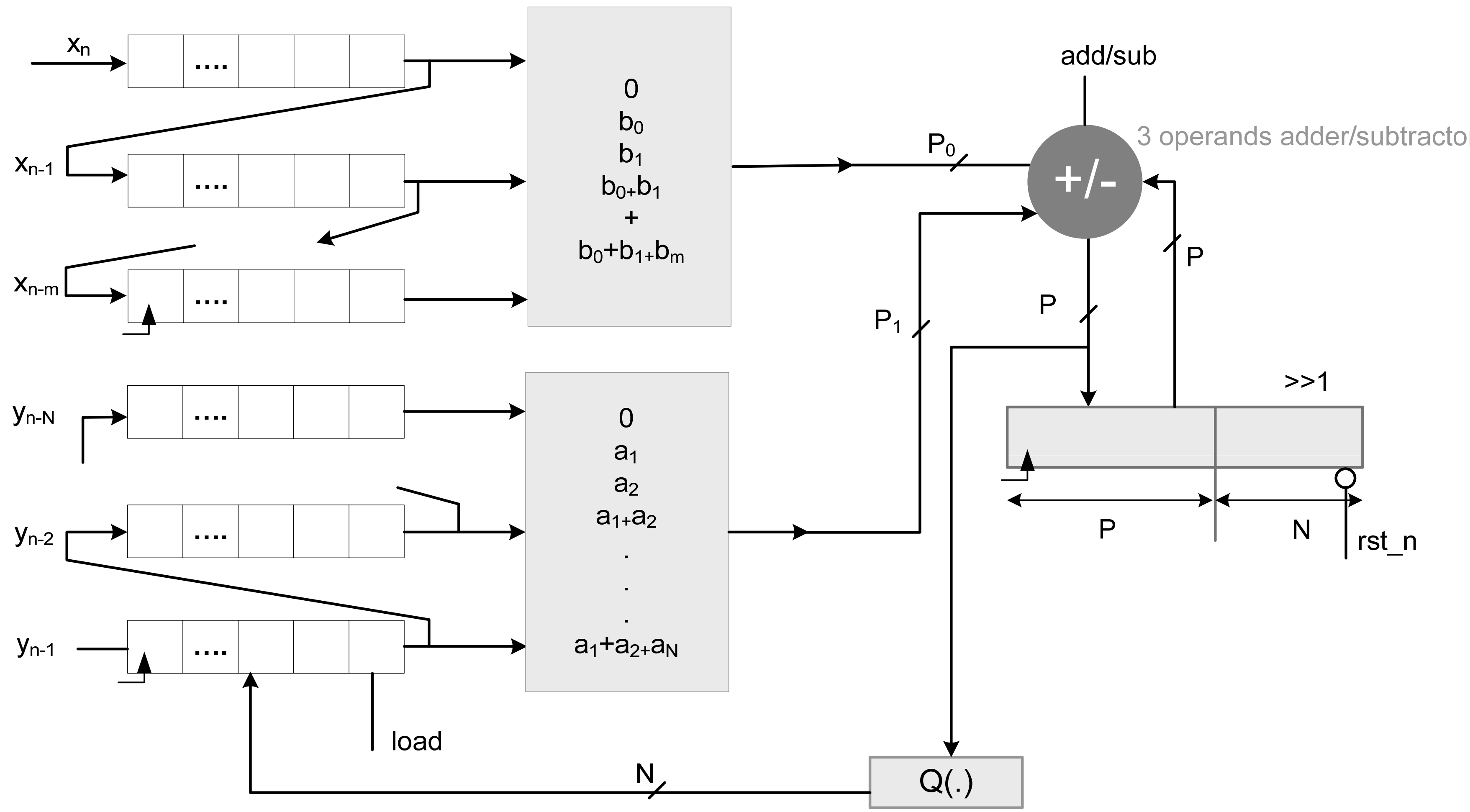


Reducing the output of the multiplexers using a CPA-based adder tree and one accumulator

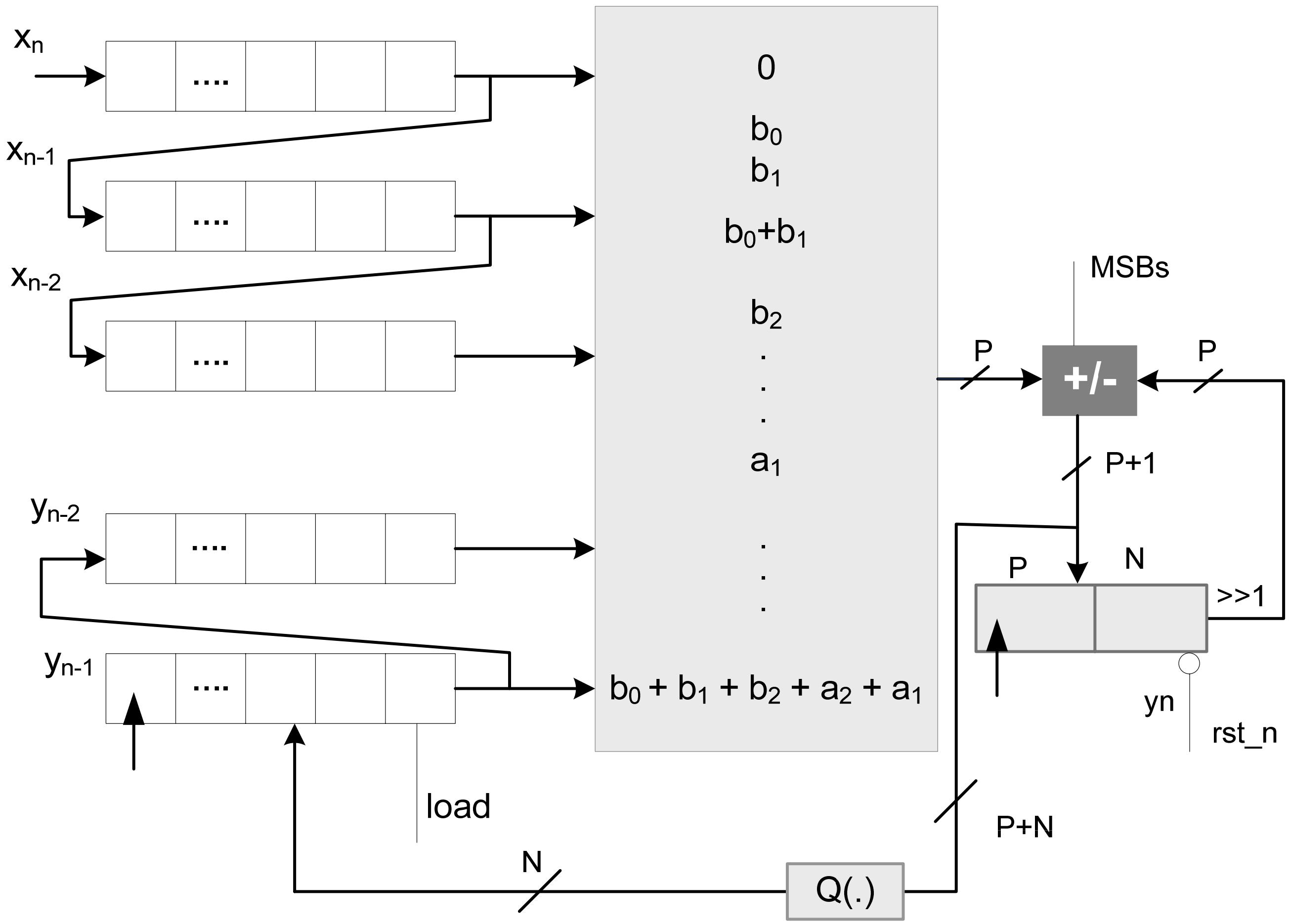


DA-based IIR filter design

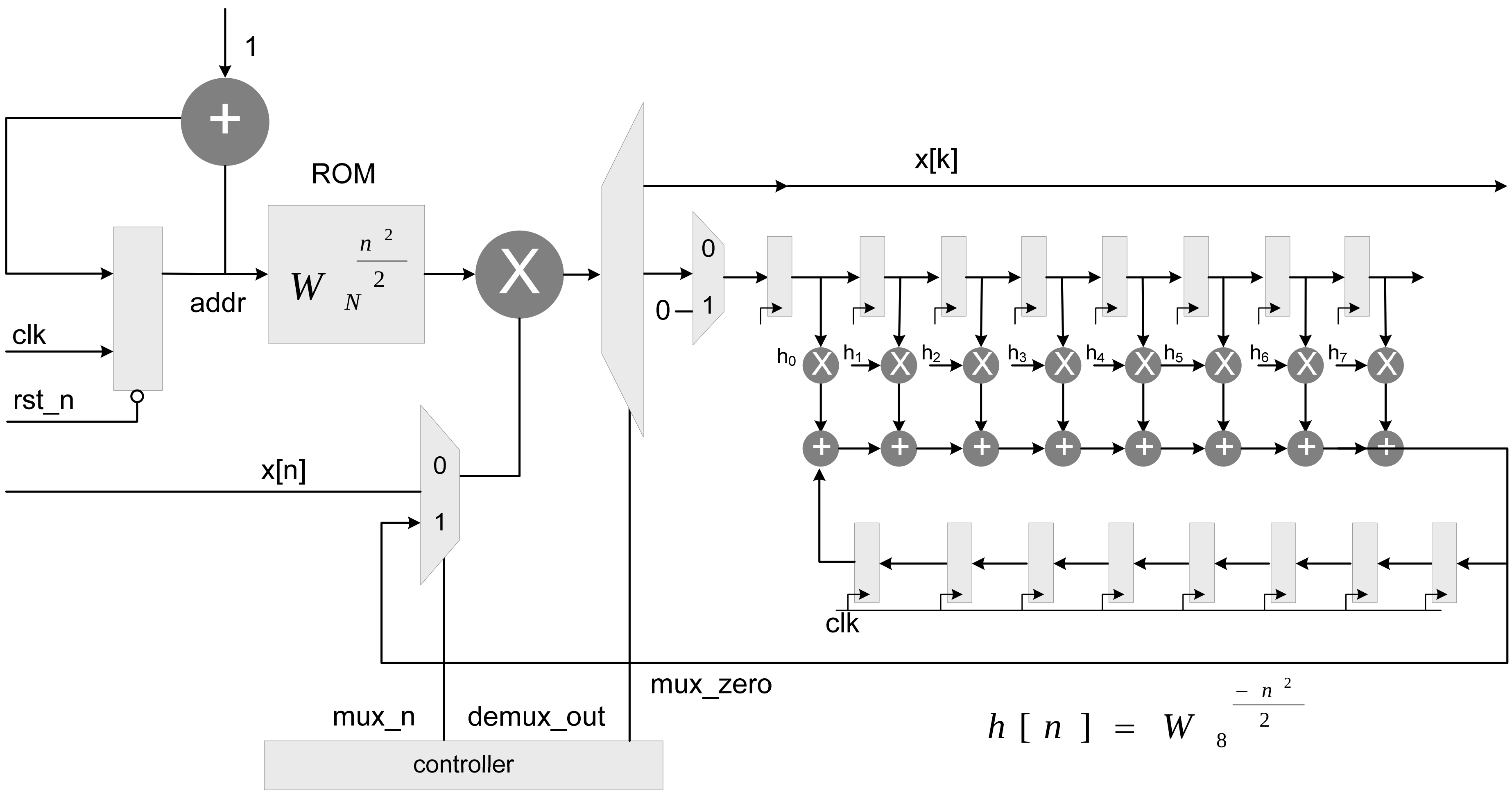
Two ROM-based design



One ROM-based design

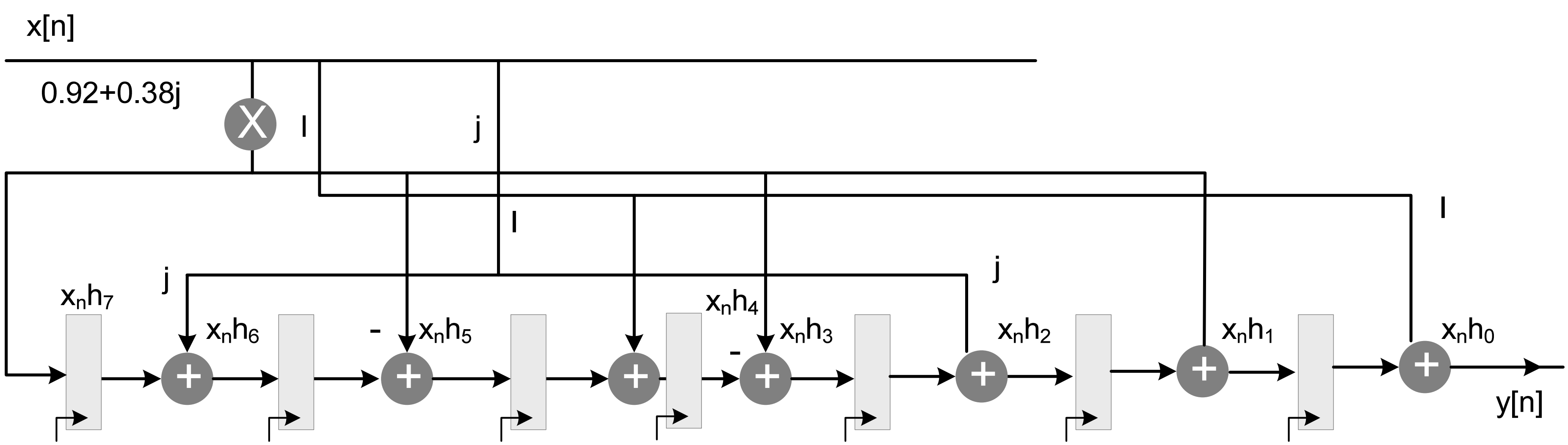


DFT implementation using circular convolution



$$h[n] = W_8^{-\frac{n^2}{2}}$$

Optimized TDF implementation of the DF implementation in previous figure



Questions/Feedback !!