4.2. Parallel systhesis of distributed inner product processors

Inner product computation - fundamental operation in DSP

$$y = x^T h = \sum_{i=1}^{N} x_i h_i = \sum_{i=1}^{N} \sum_{j=0}^{B_x - 1} \sum_{k=0}^{B_h - 1} x_{i,j} h_{i,k} 2^{-(j+k)}$$

x **is the data and** ***h*** **is the coefficient vector**

Different evaluation methods

1. Processor like MAC architecture

2. Look-up table based DA architecture

3. Array of S/P multipliers followed by an adder tree

Area and time cost of the IP operation according to method 3

$$C_A = (N*B_h+N), \quad C_T = B_x+B_h+\log_2 N)$$

Cost reduction on bit level

$$y = x^t H = x^t \begin{vmatrix} h_1 \\ h_2 \\ . \\ . \\ h_n \end{vmatrix} = x^t \begin{vmatrix} h_{1,1}2^0 & h_{1,2}2^{-1} & . & h_{1,B_h-1}2^{B_h} \\ h_{2,0}2^0 & h_{2,1}2^{-1} & . & h_{2,B_h-1}2^{-B_h-1} \\ . & . & . & . \\ . & . & . & . \\ h_{N,0}2^0 & h_{N,1}2^{-1} & . & h_{N,B_h-1}2^{-B_h-1} \end{vmatrix}$$

1. Canonical Signed Digit representation of h_i $h_{i,k} \in \{-1, 0, +1\}$

2. Merging of identical subpolinomials of h_i

$$(h_{i,\alpha} + h_{i,\beta})x_n = h_{i,\alpha}x_n + h_{i,\beta}x_n$$

$$h_{j,\gamma}x_n + 2^c * h_{k,\gamma}x_m = h_{j,\gamma}(x_n + 2^c * x_m)$$

Numerical example of the method

h={115, -58, 152, 122} ; B_h=8 bit

$$y=x_1*115-x_2*58+x_3*152+x_4*122=$$

$$=x_1*(2^7-2^4+2^2-2^0) + x_2*(-2^6+2^3-2^1) + x_3*(2^7+2^5-2^3) + x_4*(2^7-2^3+2^1)$$

$$=-x_1*2^0 + x_3*2^5 + x_4*2^1 + (2^1*x_1-x_2)*(2^6-2^3+2^1) + (x_3+x_4)*(2^7-2^3)$$

First iteration step: $x_5=2*(2*x_1-x_2)$

$$y=[x_1\,x_2\,x_3\,x_4]\begin{bmatrix}1 & 0 & 0 & -1 & 0 & 1 & 0 & -1\\ 0 & -1 & 0 & 0 & 1 & 0 & -1 & 0\\ 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0\\ 1 & 0 & 0 & 0 & -1 & 0 & 1 & 0\end{bmatrix}$$

$$=[x_1\,x_3\,x_4\,x_5]\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & 0 & -1\\ 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0\\ 1 & 0 & 0 & 0 & -1 & 0 & 1 & 0\\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 1\end{bmatrix}$$

Second iteration step: $x_6=2*(x_3+x_4)$

$$y = [x_1\, x_3\, x_4\, x_5] \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}$$

$$= [x_1\, x_3\, x_4\, x_5\, x_6] \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

In other form

$$y = x^{0T} H^0 = x^{0T} P^0 H^1 = x^{1T} H^1$$

$$y = [x_1 x_2 x_3 x_4] \begin{bmatrix} 1 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}$$

For the full iteration process

$$y = x^{0T} H^0 = x^{0T} \prod_{m=0}^{M} P^m \quad H^{M+1}$$

where M **: number of iteration steps**

$M < N * B_h/2$ **(upper bound)**

x^m **: actual signal list after m iteration**

P^m **: preaddition matrices**

$P=[\ I\ p\ ]$, p **has only 2 nonzero element**

H^{m+1} **: temporary coefficient matrices**

$H^{M+1} < B_h + log_2\ N$

Algorithm implementation

Goal: find the best implementation of the IP operation for a given h coefficient vector, using serial arithmetic

Operators: serial adder/subtractor, bit delay cost ratio C=delay/adder

Cost function: total number of functional units

Reduction steps: based on statistics of non-zero digit coincidences in H^m matrices

The first iteration step and its statisticsl matrices

$$H^0=\begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 1 & 0 & -1 \\ 0 & -1 & 0 & 0 & 1 & 0 & -1 & 0 \\ 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \end{bmatrix}$$

$$S0^0=\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & -1 & -2 \\ 1 & -1 & 0 & 2 \\ 1 & -2 & 2 & 0 \end{bmatrix};\ S1^0=\begin{bmatrix} 0 & 2 & -2 & -2 \\ -3 & 0 & -1 & -1 \\ 1 & -1 & 0 & 0 \\ 2 & 0 & 0 & 0 \end{bmatrix};\ S2^0=\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & -2 & 0 \end{bmatrix};$$

The second iteration step and its statistical matrices

$$H^0=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}$$

$$S0^1=\begin{bmatrix} 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 \\ 0 & 0 & 2 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \end{bmatrix};\ S1^1=\begin{bmatrix} 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix};\ S2^1=\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix};$$

Maximal gain: G_{max} **= MAX{max** $|Si|$ **-** $i*C$**}**

G_{max} **assigns the two rows of** H^m

Number of statistics matrices determined by

$$n = MAX\{\frac{\lceil 0.5 B_h \rceil - \max|S0|}{0.5 + C} \; ; \; \frac{H^i{}_{SMNZ} - \max|S0|}{0.5 + C}\}$$

Notes on the algorithm

There could be more than one equivalent solution (same gain for different signals)

Technology dependence is controlled by C parameter

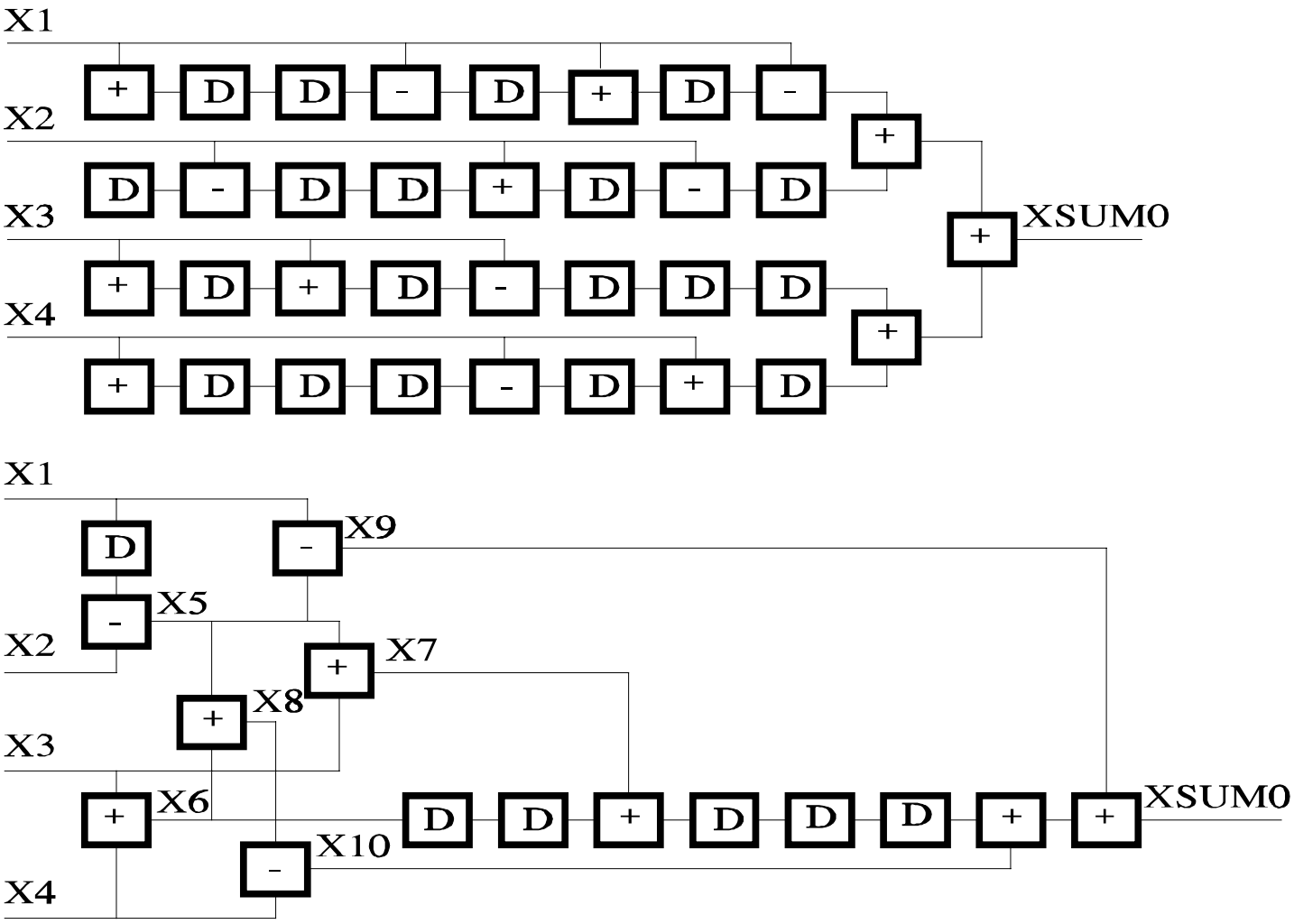
Practically only S0, S1, S2 are used

The result of the iteration process

X1 DFF -> X1D1	X6 DFF -> XS2
X1D1 SUB X2 -> X5	XS2 DFF -> XS3
X3 ADD X4 -> X6	XS3 ADD X7 -> XS4
X3 ADD X5 -> X7	XS4 DFF -> XS5
X5 ADD X6 -> X8	XS5 DFF -> XS6
X1 SUB X5 -> X9	XS6 DFF -> XS7
X4 SUB X8 -> X10	XS7 ADD -> XS8

XS8 ADD X9 -> XSUM0

The generated SFG



Conclusion

Bit level analyses of fixed coefficients provides significant hardware reduction (30%)

Original DA **arithmetic <<---->> memory**
New DA **arithmetic <<---->> topology**

Full synchron, pipe-line architecture built up from basic primitives

Optimal for FPGA implementations: logic blocks + routing balanced utilization

Typical applications: FIR, IIR filters, linear transformations, Neural networks