Christian-Albrechts-Universität zu Kiel

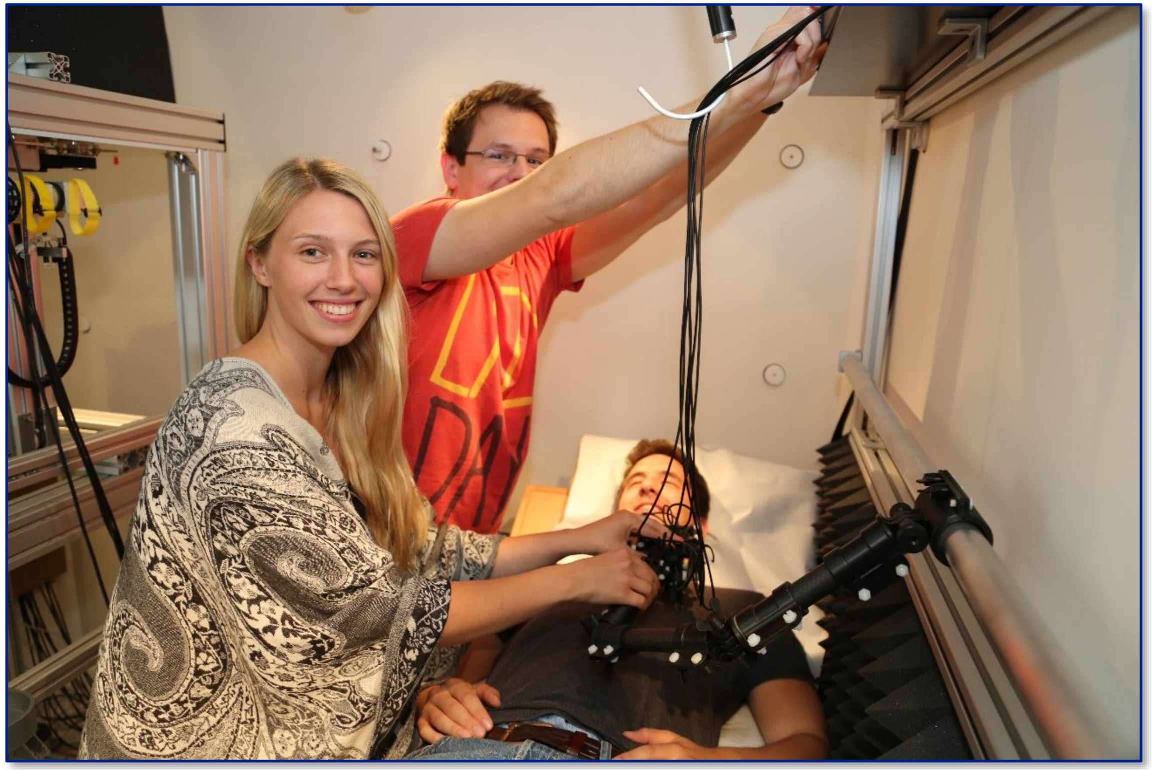
Advanced Digital Signal Processing

Part 3: Efficient FIR Structures

Gerhard Schmidt

Christian-Albrechts-Universität zu Kiel
Faculty of Engineering
Institute of Electrical and Information Engineering
Digital Signal Processing and System Theory

Contents



Derivation – Part 1

Single output of an FIR filter

Basic formula:

$$y(n) \;=\; \sum_{i=0}^{N-1} v(n-i)\,h_i.$$

In vector notation:

$$\begin{aligned}
\boldsymbol{v}(n) &= \left[v(n),\, v(n-1),\, ...,\, v(n-N+1)\right]^{\mathrm{T}},\\
\boldsymbol{h} &= \left[h_0,\, h_1,\, ...,\, h_{N-1}\right]^{\mathrm{T}},\\
y(n) &= \boldsymbol{v}^{\mathrm{T}}(n)\,\boldsymbol{h}.
\end{aligned}$$

Two outputs of an FIR filter (block processing)

About 2N multiplications and additions in order to compute two output samples.

$$\begin{bmatrix} y(n) \\ y(n-1) \end{bmatrix} = \begin{bmatrix} v(n), & \dots, & v(n-N+1) \\ v(n-1), & \dots, & v(n-N) \end{bmatrix} \begin{bmatrix} h_0 \\ \vdots \\ h_{N-1} \end{bmatrix}.$$

Derivation – Part 2

"Even" and "odd" signal and filter vectors

Definitions:

$$\begin{aligned}
\boldsymbol{h}_{\text{even}} &= \left[h_0,\, h_2,\, \ldots,\, h_{N-2}\right]^{\mathrm{T}}, \\
\boldsymbol{h}_{\text{odd}} &= \left[h_1,\, h_3,\, \ldots,\, h_{N-1}\right]^{\mathrm{T}}, \\
\boldsymbol{v}_{\text{even}}(n) &= \left[v(n),\, v(n-2),\, \ldots,\, v(n-N+2)\right]^{\mathrm{T}}, \\
\boldsymbol{v}_{\text{odd}}(n) &= \left[v(n-1),\, v(n-3),\, \ldots,\, v(n-N+1)\right]^{\mathrm{T}}.
\end{aligned}$$

Relations:

$$\begin{aligned}
\boldsymbol{v}_{\text{odd}}(n) &= \boldsymbol{v}_{\text{even}}(n-1)\,, \\
\boldsymbol{v}_{\text{even}}(n) &= \boldsymbol{v}_{\text{odd}}(n+1)\,.
\end{aligned}$$

Derivation – Part 3

Two outputs of an FIR filter (continued)

Once again:

$$\begin{bmatrix} y(n) \\ y(n-1) \end{bmatrix} = \begin{bmatrix} v(n), & \ldots, & v(n-N+1) \\ v(n-1), & \ldots, & v(n-N) \end{bmatrix} \begin{bmatrix} h_0 \\ \vdots \\ h_{N-1} \end{bmatrix}$$

… inserting the abbreviations …

$$= \begin{bmatrix} \boldsymbol{v}_{\text{even}}^{\text{T}}(n) & \boldsymbol{v}_{\text{odd}}^{\text{T}}(n) \\ \boldsymbol{v}_{\text{even}}^{\text{T}}(n-1) & \boldsymbol{v}_{\text{odd}}^{\text{T}}(n-1) \end{bmatrix} \begin{bmatrix} \boldsymbol{h}_{\text{even}} \\ \boldsymbol{h}_{\text{odd}} \end{bmatrix}$$

… multiplying the matrix elements with the subvectors …

$$= \begin{bmatrix} \boldsymbol{v}_{\text{even}}^{\text{T}}(n)\,\boldsymbol{h}_{\text{even}} + \boldsymbol{v}_{\text{odd}}^{\text{T}}(n)\,\boldsymbol{h}_{\text{odd}} \\ \boldsymbol{v}_{\text{even}}^{\text{T}}(n-1)\,\boldsymbol{h}_{\text{even}} + \boldsymbol{v}_{\text{odd}}^{\text{T}}(n-1)\,\boldsymbol{h}_{\text{odd}} \end{bmatrix}$$

… adding appropriate zeros …

$$= \begin{bmatrix} \boldsymbol{v}_{\text{even}}^{\text{T}}(n)\,\boldsymbol{h}_{\text{even}} + \boldsymbol{v}_{\text{odd}}^{\text{T}}(n)\,\boldsymbol{h}_{\text{odd}} \overbrace{-\boldsymbol{v}_{\text{odd}}^{\text{T}}(n)\,\boldsymbol{h}_{\text{even}} + \boldsymbol{v}_{\text{odd}}^{\text{T}}(n)\,\boldsymbol{h}_{\text{even}}}^{=0} \\ \boldsymbol{v}_{\text{even}}^{\text{T}}(n-1)\,\boldsymbol{h}_{\text{even}} + \boldsymbol{v}_{\text{odd}}^{\text{T}}(n-1)\,\boldsymbol{h}_{\text{odd}} \underbrace{-\boldsymbol{v}_{\text{odd}}^{\text{T}}(n)\,\boldsymbol{h}_{\text{odd}} + \boldsymbol{v}_{\text{odd}}^{\text{T}}(n)\,\boldsymbol{h}_{\text{odd}}}_{=0} \end{bmatrix}$$

Derivation – Part 4

Two outputs of an FIR filter (continued)

… result from the previous slide …

$$\begin{bmatrix} y(n) \\ y(n-1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{v}_{\mathrm{even}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{even}} + \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{odd}} - \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{even}} + \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{even}} \\ \boldsymbol{v}_{\mathrm{even}}^{\mathrm{T}}(n-1)\,\boldsymbol{h}_{\mathrm{even}} + \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n-1)\,\boldsymbol{h}_{\mathrm{odd}} - \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{odd}} + \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{odd}} \end{bmatrix}$$

… inserting $\boldsymbol{v}_{\mathrm{odd}}(n) = \boldsymbol{v}_{\mathrm{even}}(n-1)$ ***…***

$$= \begin{bmatrix} \boldsymbol{v}_{\mathrm{even}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{even}} + \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{odd}} - \overbrace{\boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)}^{\boldsymbol{v}_{\mathrm{even}}^{\mathrm{T}}(n-1)}\,\boldsymbol{h}_{\mathrm{even}} + \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{even}} \\ \underbrace{\boldsymbol{v}_{\mathrm{even}}^{\mathrm{T}}(n-1)}_{\boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)}\,\boldsymbol{h}_{\mathrm{even}} + \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n-1)\,\boldsymbol{h}_{\mathrm{odd}} - \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{odd}} + \boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\mathrm{odd}} \end{bmatrix}$$

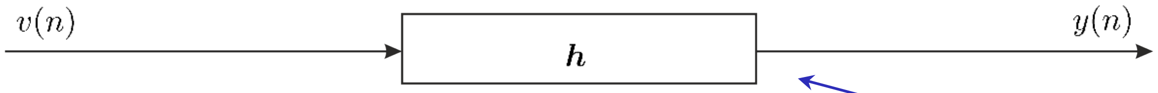
… rearranging the terms …

$$= \begin{bmatrix} \big[\boldsymbol{v}_{\mathrm{even}}(n) - \boldsymbol{v}_{\mathrm{even}}(n-1)\big]^{\mathrm{T}}\,\boldsymbol{h}_{\mathrm{even}} + \overbrace{\boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\big[\boldsymbol{h}_{\mathrm{odd}} + \boldsymbol{h}_{\mathrm{even}}\big]}^{y_{\mathrm{tmp}}(n)} \\ \underbrace{\boldsymbol{v}_{\mathrm{odd}}^{\mathrm{T}}(n)\,\big[\boldsymbol{h}_{\mathrm{even}} + \boldsymbol{h}_{\mathrm{odd}}\big]}_{y_{\mathrm{tmp}}(n)} - \big[\boldsymbol{v}_{\mathrm{odd}}(n) - \boldsymbol{v}_{\mathrm{odd}}(n-1)\big]^{\mathrm{T}}\,\boldsymbol{h}_{\mathrm{odd}} \end{bmatrix}.$$

About 1,5 N multiplications and additions in order to compute two output samples.

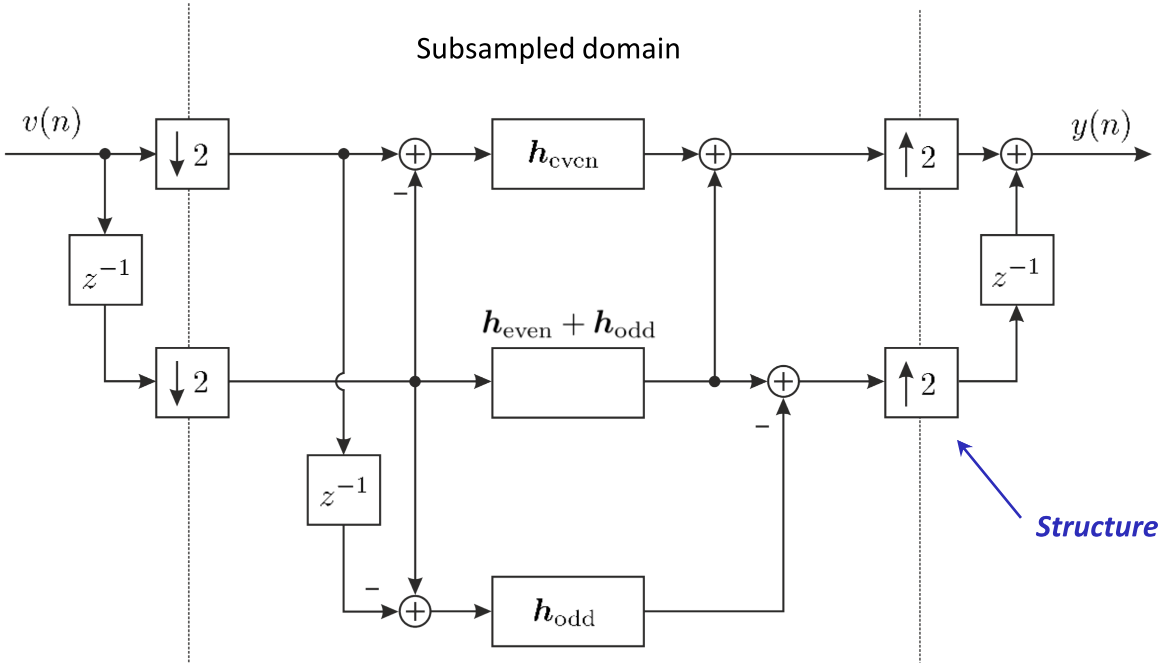
Derivation – Part 5

Basic structure



Structure has to be computed twice to produce two output samples!

Efficient structure

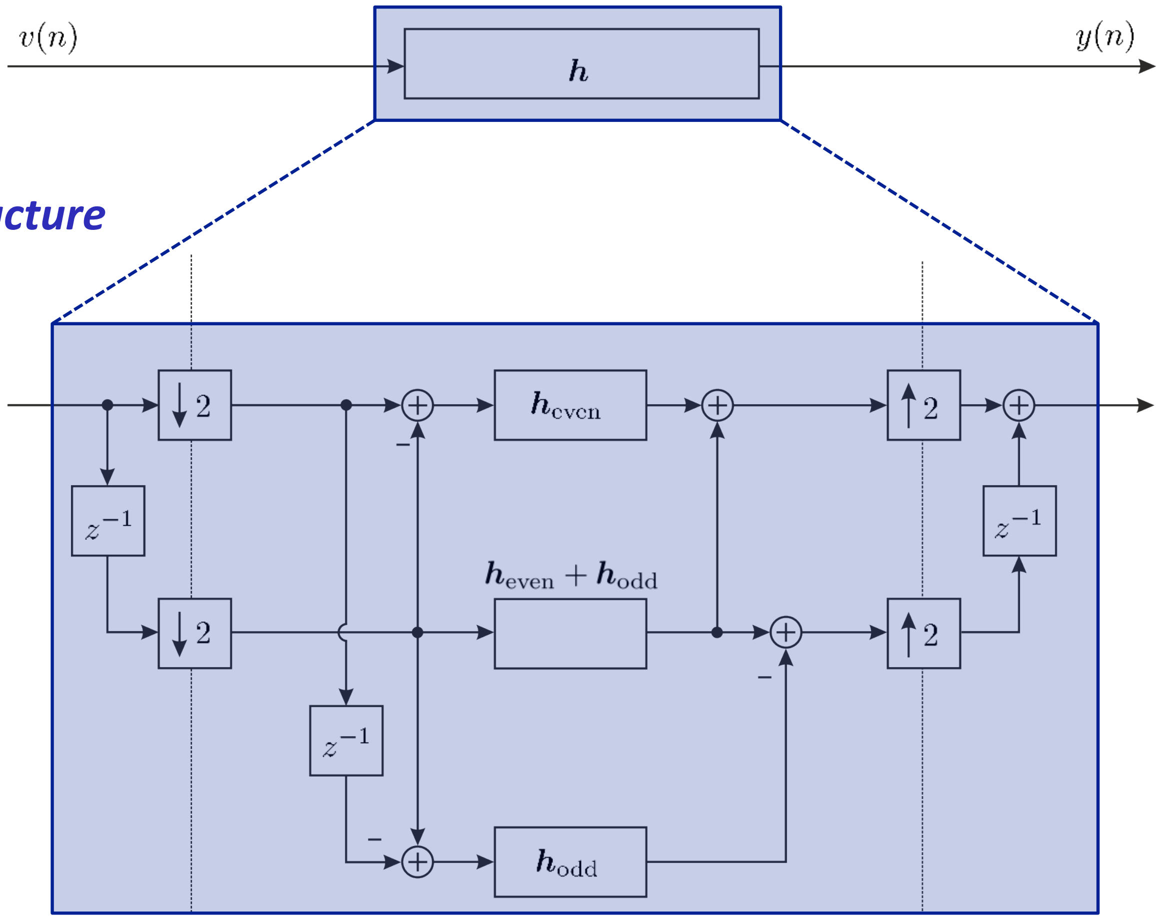


Structure has to be computed once to produce two output samples!

Derivation – Part 6

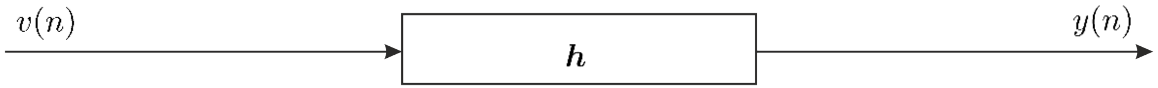
Basic structure

Efficient structure

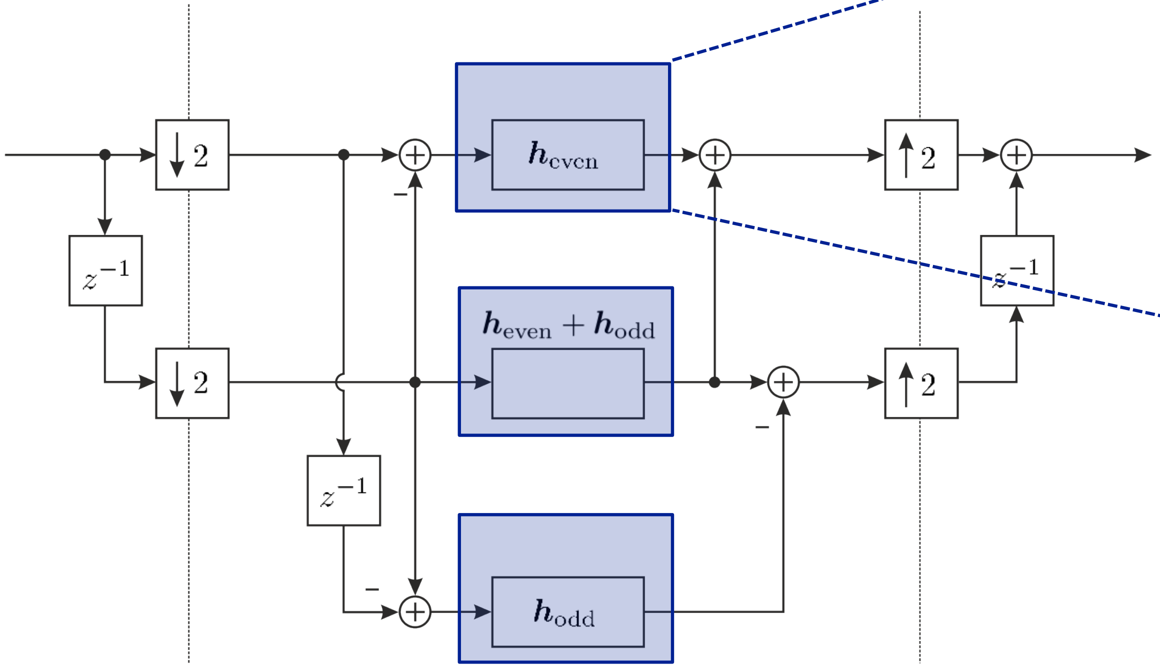


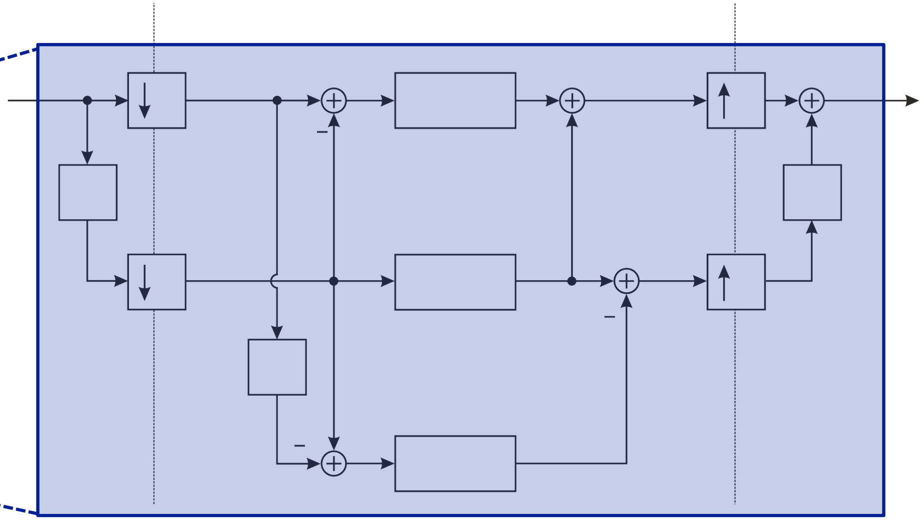
Derivation – Part 7

Basic structure



Efficient structure





Efficient structure in efficient structure …

Derivation – Part 8

Reduction in computational complexity (for large filter orders)

Number of samples per frame	Reduction
2	25,00 %
4	43.75 %
8	57.81 %
16	68.36 %
32	76.27 %
64	82.20 %
128	86.65 %

Derivation – Part 8

Reduction in computational complexity (for large filter orders)

Number of samples per frame	Reduction
2	25,00 %
4	43.75 %
8	57.81 %
16	68.36 %
32	76.27 %
64	82.20 %
128	86.65 %

Attention: while reducing the numerical complexity often the delay is increased!

Contents