Design and implementation of reversible integer quaternionic paraunitary filter banks on adder-based distributed arithmetic

Nick A. Petrovsky   Eugene V. Rybenkov   Alexander A. Petrovsky
nick.petrovsky@bsuir.by

September 20, 2017

Belarusian State University of Informatics and Radioelectronics,
Department of Computer Engineering
Minsk, Belarus

21st Conference SPA 2017
Signal Processing: Algorithms, Architectures, Arrangements, and Applications

Motivation

Recently, there has been increasing interest in designing filter banks with low implementation complexity. Approaches based on the sum-of-power-of-two (SOPOT) coefficients are particularly attractive because coefficients multiplications can be implemented with simple shifts and additions only which makes it possible to use the adder-based distributed arithmetic (DA) [1].

Idea

The adder-based DA ($DA_{\sum}$), in contrast to conventional DA (ROM-based DA), decomposes the fixed coefficients of the inner product into bit level, distributes the multiplication operation, and shares the common summation terms.

[1]T.-S Chang, C. Chen, C.-W. Jen, “New distributed arithmetic algorithm and its application to IDCT”, *IEE Proc. Circuits Devices and Systems*, vol.146.no.4, 1999, pp.159-163.

The purpose of the given paper develops a new family of the **integer-to-integer** invertible quaternionic Q-PUFB (Int-Q-PUFB) using multipliers based on the block-lifting structure with sum of-powers-of-two (SOPOT) coefficients.

Design examples show that SOPOT Int-Q-PUFB with a good frequency characteristic can be designed with low implementation complexity.

Quaternion algebra and orthogonal matrices

The quaternion algebra $\mathbb{H}$ is an associative non-commutative four-dimensional algebra

$$\mathbb{H} = \{\mathbf{q} = q_1 + q_2 i + q_3 j + q_4 k | q_1, q_2, q_3, q_4 \in \mathbb{R}\},$$

where the orthogonal imaginary numbers obey the following multiplicative rules:

$$i^2 = j^2 = k^2 = ijk = -1, \; ij = -ji = k, jk = -kj = i, \; ki = -ik = j.$$

There are two different multiplication matrices $\mathbf{M}^+(q)$ and $\mathbf{M}^-(q)$:

$$qx \Leftrightarrow \mathbf{M}^+(q)\,\mathbf{x}, \quad xq \Leftrightarrow \mathbf{M}^-(q)\,\mathbf{x}$$

which are related in the following way:

$$\mathbf{M}^{\mp}(q) = \mathbf{D}_C \mathbf{M}^{\pm}(q)^T \mathbf{D}_C,$$

where $\mathbf{D}_C = \mathrm{diag}(1, -\mathbf{I}_3)$ denotes the conjugate quaternion $\overline{q} = q_1 - q_2 i - q_3 j - q_4 k$ in the matrix representation, i.e. $\overline{q} = \mathbf{D}_C q$.

Thus $\mathbf{M}^{\pm}(\overline{q}) = \mathbf{M}^{\pm}(q)^T$ is equal to $\mathbf{M}^{\mp}(\overline{q}) = \mathbf{D}_C \mathbf{M}^{\pm}(q)\,\mathbf{D}_C$.

Every matrix belonging to $SO(4)$, can be represented as a product of left and right unit quaternions P and Q ($|P| = 1$ and $|Q| = 1$)

$$\underset{\mathbf{R}\in SO(4)}{\forall} \; \underset{P,Q\in\text{unit quat.}}{\exists} \quad \mathbf{R} = \mathbf{M}^+(P)\cdot\mathbf{M}^-(Q) = \mathbf{M}^-(Q)\cdot\mathbf{M}^+(P)$$

Structurally lossless lattice for Q-PUFB [2][3]

$$\mathbf{E}(z) = \mathbf{G}_{N-1}\mathbf{G}_{N-2}\ldots\mathbf{G}_1\mathbf{E}_0;$$

$$\mathbf{E}_0 = \frac{1}{\sqrt{2}}\mathbf{\Phi}_0\mathbf{W}\,\mathrm{diag}\left(\mathbf{I}_4, \mathbf{J}_4\right),\ \mathbf{G}_i = \frac{1}{2}\mathbf{\Phi}_i\mathbf{W}\ (z)\mathbf{W},\quad i = \overline{1, N-1},$$

$$\mathbf{W} = \begin{bmatrix} \mathbf{I}_2 & \mathbf{I}_2 \\ \mathbf{I}_2 & -\mathbf{I}_2 \end{bmatrix};\ \mathbf{\Lambda}(z) = \mathrm{diag}\left(\mathbf{I}_4, z^{-1}\mathbf{I}_4\right),$$

$$\mathbf{\Phi}_i = \mathrm{diag}\left(\mathbf{\Gamma}, \mathbf{I}_4\right)\mathrm{diag}\left(\mathbf{M}^-\left(Q_i\right), \mathbf{M}^-\left(Q_i\right)\right)\mathrm{diag}\left(\mathbf{M}^+\left(P_i\right), \mathbf{M}^+\left(P_i\right)\right)\mathrm{diag}\left(\mathbf{\Gamma}, \mathbf{I}_4\right)$$

$$\mathbf{\Phi}_{N-1} = \mathrm{diag}\left(\mathbf{J}_4, \mathbf{I}_4\right)\mathrm{diag}\left(\mathbf{M}^-\left(Q_i\right), \mathbf{M}^-\left(Q_i\right)\right)\mathrm{diag}\left(\mathbf{M}^+\left(P_i\right), \mathbf{M}^+\left(P_i\right)\right)\mathrm{diag}\left(\mathbf{\Gamma}, \mathbf{I}_4\right),$$

8-channel PMI LP Q-PUFB realized according to is one-regular if and only if:

$$Q_{N-1} = \pm\frac{1}{2}\overline{Q_{N-2}}\cdot\ldots\cdot\overline{Q_0}\cdot\overline{c_1}\cdot\overline{P_0}\cdot\ldots\cdot\overline{P_{N-1}}\cdot c_2,$$

where c_1 and c_2 are the quaternions: $c_1 = 1 + i + j + k$; $c_2 = k$; N is order of the factorization

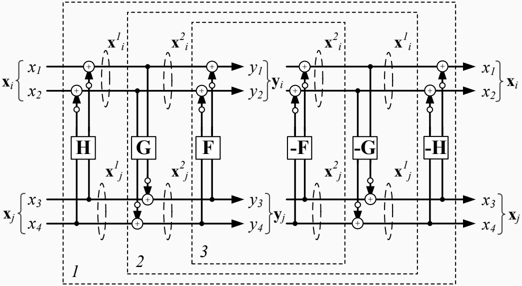
[2] M. Parfieniuk and A. Petrovsky, “Quaternionic lattice structures for four-channel paraunitary filter banks,” *EURASIP J. Adv. Signal Process., Special Issue on Multirate Systems and Applications*., vol. 2007, Article ID 37481.

[3] M. Parfieniuk and A. Petrovsky, “Inherently lossless structures for eight and six-channel linear-phase paraunitary filter banks based on quaternion multipliers,” *Signal Process.*, vol. 90, pp. 1755–1767, 2010.

The quaternion multiplication as integer-to-integer operator

$$\mathbf{M}^{+}(Q) = \begin{bmatrix} \mathbf{C}(Q) & -\mathbf{S}(Q) \\ \mathbf{S}(Q) & \mathbf{C}(Q) \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{I}_2 & \mathbf{F}(Q) \\ 0 & \mathbf{I}_2 \end{bmatrix}}_{\mathbf{U}(Q)} \underbrace{\begin{bmatrix} \mathbf{I}_2 & 0 \\ \mathbf{G}(Q) & \mathbf{I}_2 \end{bmatrix}}_{\mathbf{L}(Q)} \underbrace{\begin{bmatrix} \mathbf{I}_2 & \mathbf{H}(Q) \\ 0 & \mathbf{I}_2 \end{bmatrix}}_{\mathbf{V}(Q)}.$$

The block-lifting scheme for Q-MUL as integer-to-integer operator



$$\mathbf{F}(Q) = (\mathbf{C}(Q) - \mathbf{I}_2)\mathbf{S}(Q)^{-1},$$
$$\mathbf{G}(Q) = \mathbf{S}(Q),$$
$$\mathbf{H}(Q) = \mathbf{S}(Q)^{-1}(\mathbf{C}(Q) - \mathbf{I}_2).$$

$$\mathbf{M}^{+}(\overline{Q}) = \underbrace{\begin{bmatrix} \mathbf{I}_2 & -\mathbf{H}(Q) \\ 0 & \mathbf{I}_2 \end{bmatrix}}_{\mathbf{V}(\overline{Q})} \underbrace{\begin{bmatrix} \mathbf{I}_2 & 0 \\ -\mathbf{G}(Q) & \mathbf{I}_2 \end{bmatrix}}_{\mathbf{L}(\overline{Q})} \underbrace{\begin{bmatrix} \mathbf{I}_2 & -\mathbf{F}(Q) \\ 0 & \mathbf{I}_2 \end{bmatrix}}_{\mathbf{U}(\overline{Q})}.$$

Implementing Q-PUFB using quaternion multiplication

Controlling the dynamic range of lifting coefficients

The use of the ladder circuit parameterization increases the dynamic range of the matrix coefficients, and that is unacceptable for fixed-point arithmetic. Bringing the parameters of the multiplier to the required dynamic range can be achieved if the quaternion multiplication operator selected according to the following equation [4]:

$$\mathbf{M}^{+}(Q) = \begin{cases} \mathbf{P}_{post} \cdot \mathbf{M}^{+}\left(\widetilde{Q}\right) \cdot \mathbf{P}_{pre}, & \text{if } \det(\mathbf{P}) = 1, \\ \mathbf{P}_{post} \cdot \mathbf{M}^{-}\left(\widetilde{Q}\right) \cdot \mathbf{P}_{pre}, & \text{if } \det(\mathbf{P}) = -1, \end{cases}$$

$$Q\mathbf{x} = \mathbf{M}^{+}(Q)\,\mathbf{x} = \mathbf{P}_{post}\mathbf{M}^{\pm}\left(\widetilde{Q}\right)\mathbf{P}_{pre}\mathbf{x} =$$

$$= \mathbf{P}_{post} \underbrace{\begin{bmatrix} \mathbf{I}_2 & \mathbf{F}(\widetilde{Q}) \\ 0 & \mathbf{I}_2 \end{bmatrix}}_{\mathbf{U}(\widetilde{Q})} \underbrace{\begin{bmatrix} \mathbf{I}_2 & 0 \\ \mathbf{G}(\widetilde{Q}) & \mathbf{I}_2 \end{bmatrix}}_{\mathbf{L}(\widetilde{Q})} \underbrace{\begin{bmatrix} \mathbf{I}_2 & \mathbf{H}(\widetilde{Q}) \\ 0 & \mathbf{I}_2 \end{bmatrix}}_{\mathbf{V}(\widetilde{Q})} \mathbf{P}_{pre}\mathbf{x}.$$

[4] M. Parfieniuk and A. Petrovsky, “Quaternion multiplier inspired by the lifting implementation of plane rotations,” *IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications*, vol. 57, no. 10, pp. 2708–2717, Oct. 2010.

Adder-based distributed arithmetics (DA$_\Sigma$)

An inner product of length L is $y = \mathbf{A} \cdot \mathbf{X}$. Adder-based DA decomposes (by T. S. Chang et. al.) the fixed coefficients A_i into bit level (B is word length of vector $\mathbf{A}$ components):

$$y = \sum_{i=1}^{L} A_i X_i = \sum_{i=1}^{L} \left(\sum_{j=1}^{B} A_{ij} 2^{-j} \right) X_i = \sum_{j=1}^{B} \left(\sum_{i=1}^{L} A_{ij} X_i \right) 2^{-j} \quad \text{— bit-parallel.}$$

The term $\sum_{i=1}^{L} A_{ij} X_i = F_j$ is a combination of X_i since A_{ij} is only 0 or 1. If X_i is a serially input one can obtain F_j bit by bit via serial adders, that is

$$F_j = \sum_{m=1}^{M} F_{j,m} 2^{-m}.$$

And equation can be rewritten:

$$y = \sum_{j=1}^{B} F_j 2^{-j} = \sum_{j=1}^{B} \left(\sum_{m=1}^{M} F_{j,m} 2^{-m} \right) 2^{-j} = \sum_{m=1}^{M} \left(\sum_{j=1}^{B} F_{j,m} 2^{-j} \right) 2^{-m} \quad \text{— bit-serial.}$$

One can shift and then accumulate the term $\sum_{j=1}^{B} F_{j,m} 2^{-j}$ at each cycle m to obtain the inner product. Since F_j is computed using adders the proposed DA algorithm is called **adder-based DA (DA$_\Sigma$)**.

Problem statement of the designing Int-Q-PUFB

Design problem of a Int-Q-PUFB can be defined as: find a set of quaternions P_i and Q_i for a Q-PUFB and word length B of block-lifting coefficients $\mathbf{F}(q)$, $\mathbf{G}(q)$, and $\mathbf{H}(q)$, which provide high value of the coding gain (CG)

$$CG = 10\log_{10}\left(\frac{\frac{1}{M}\sum_{k=0}^{M-1}\sigma_{xk}^2}{\left(\prod_{k=0}^{M-1}\sigma_{xk}^2\right)^{\frac{1}{M}}}\right),$$

σ_{xk}^2 are the subband variances,

with the following constraints:

1. the maximum stopband attenuation (ε_{SBE}) measured on terms of energy;
2. the minimum reconstruction error ε_q, as a result of quantization of block-lifting coefficients: $\varepsilon_q = \max\left(|y(n) - x(n)|\right)$, where $y(n)$ is the output data of the synthesis filter bank; $x(n)$ is the input data of the analysis filter bank;
3. the maximum number of ONE bits K in binary code to represent the block-lifting coefficients of Int-Q-PUFB: filter coefficients are sum-of-power-of-two.

Polar form of quaternion: $Q = |Q| \cdot e^{i\phi} e^{j\psi} e^{k\chi}$

Target function

$$f(x) = -CG(x)$$

Constraints

$$g_1 = \varepsilon_{SBE}(x) - \varepsilon_{minSBE} \leq 0; \quad g_2 = \varepsilon_q(x) - \varepsilon_{maxq} \leq 0; \quad g_3 = K(x) - K_{max};$$
$$\phi \in [-\pi, \pi]; \ \psi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \ \chi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$$

Modified Lagrange function

$$P\left(x, \mu^k, r^k\right) = \frac{1}{2r^k} \sum_{j=1}^{\rho} \left\{ \left[\max\left(0, \mu_j^k + r^k g_j(x)\right)\right]^2 - \left(\mu_j^k\right)^2 \right\},$$

where

$P\left(x, \mu^k, r^k\right)$ – penalty function,
$\mu^k = \left(\mu_1^k, \ldots, \mu_\rho^k\right)$ –vector of Lagrange multipliers
r^k – penalty coefficients, k – iteration number

Obtains the SOPOT Int-Q-PUFB coefficients (bit-parallel DA$_\sum$)

Finding the SOPOT coefficients. Algorithm steps

1. set the initial values (initial point, penalty coefficients increment, B, K, order of factorization N)
2. construct modified Lagrange function $L\left(x, \mu^k, r^k\right)$
3. find point $x^*\left(\mu^k, r^k\right)$ of unconstrained minimum of function $L\left(x, \mu^k, r^k\right)$, in the same time determine parameters :
    - 3.1 transform vector x to the quaternions P_i, Q_i
    - 3.2 compute the coefficients of block-lifting structure: $\mathbf{F}(\widetilde{Q})$, $\mathbf{G}(\widetilde{Q})$, $\mathbf{H}(\widetilde{Q})$ and permutation matrix $\mathbf{P}_{pre}$, $\mathbf{P}_{post}$.
    - 3.3 compute the output $y(n)$ of analysis-synthesis system
    - 3.4 determine $CG(x)$; $\varepsilon_{SBE}(x)$, $\varepsilon_q(x)$, $K(x)$
4. **if** $\left|P\left(x^*(\mu^k, r^k), \mu^k, r^k\right)\right| \leq \varepsilon$ **then** return the minimum of a Lagrange function $x^*\left(\mu^k, r^k\right)$ and **goto** step **6 else** update penalty coefficients r^{k+1} and Lagrange multipliers μ_j^{k+1}
5. set $x^{k+1} = x^*\left(\mu^k, r^k\right)$; $k = k + 1$ and **goto** step **2**
6. end

Design example: block-lifting coefficients of LP PMI 8×24 Int-Q-PUFB for $N = 3$, $B = 12$, $K = 3$ (bit-parallel DA$_{\sum}$)

Analysis part

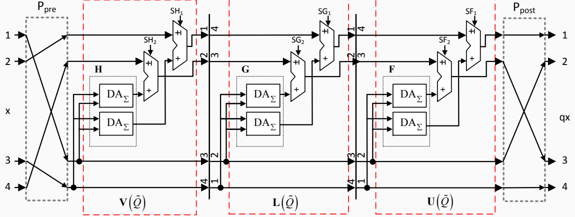
$\mathbf{M}^{\pm}$ ()	f_{11} , f_{12}	$SF_{1,2}$	g_{11} , g_{12}	$SG_{1,2}$	h_{11} , h_{12}	$SH_{1,2}$	$\mathbf{P}_{pre}$	$\mathbf{P}_{post}$
$\mathbf{M}^{-}$ (P_1)	$+(2^{-5}+2^{-4}+2^{-3})$ $-(2^{-3}+2^{-2}+2^{-1})$	$+,-$	$+(2^{-10}+2^{-9}+2^{-2})$ $+(2^{-8}+2^{-5}+2^{-3})$	$+,-$	$-(2^{-4}+2^{-3}+2^{-1})$ $-(2^{-6}+2^{-4}+2^{-1})$	$-,-$	[1 3 2 4]	[3 1 4 2]
$\mathbf{M}^{+}$ (P_2)	$-(2^{-8}+2^{-6})$ $-(2^{-7}+2^{-5}+2^{-2})$	$-,+$	$+(2^{-5}+2^{-4}+2^{-3})$ $+(2^{-5}+2^{-4}+2^{-3})$	$+,-$	$-(2^{-8}+2^{-5}+2^{-2})$ $+(2^{-8}+2^{-6}+2^{-5})$	$-,-$	[1 3 4 2]	[4 1 3 2]
$\mathbf{M}^{+}$ (P_3)	$+(2^{-4}+2^{-3}+2^{-2})$ $+(2^{-6}+2^{-3}+2^{-1})$	$-,-$	$-(2^{-7}+2^{-4}+2^{-3})$ $+(2^{-6}+2^{-3}+2^{-2})$	$-,-$	$-(2^{-5}+2^{-2}+2^{-1})$ $-(2^{-12}+2^{-10})$	$+,+$	[1 2 3 4]	[1 2 3 4]
$\mathbf{M}^{-}$ (Q_1)	$+(2^{-6}+2^{-4}+2^{-2})$ $-(2^{-6}+2^{-5}+2^{-2})$	$-,-$	$-(2^{-8}+2^{-6}+2^{-5})$ $+(2^{-8}+2^{-6}+2^{-4})$	$+,-$	$+(2^{-5}+2^{-3}+2^{-2})$ $+(2^{-7}+2^{-4}+2^{-3})$	$-,+$	[1 4 2 3]	[2 4 3 1]
$\mathbf{M}^{+}$ (Q_2)	$-(2^{-6}+2^{-5}+2^{-1})$ $+(2^{-5}+2^{-3}+2^{-2})$	$+,+$	$-(2^{-7}+2^{-6}+2^{-5})$ $-(2^{-8}+2^{-5}+2^{-4})$	$+,+$	$+(2^{-7}+2^{-3}+2^{-1})$ $-(2^{-8}+2^{-7}+2^{-2})$	$+,+$	[1 4 3 2]	[1 4 3 2]
$\mathbf{M}^{-}$ (Q_3)	$+(2^{-4}+2^{-2}+2^{-1})$ $-(2^{-7}+2^{-6}+2^{-3})$	$-,+$	$+(2^{-5}+2^{-4}+2^{-1})$ $-(2^{-8}+2^{-4}+2^{-1})$	$-,-$	$+(2^{-7}+2^{-4}+2^{-3})$ $-(2^{-4}+2^{-2}+2^{-1})$	$-,+$	[1 3 4 2]	[2 3 1 4]

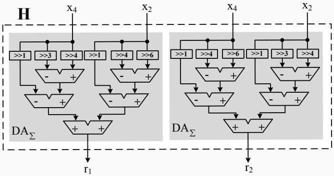
Synthesis part

$\mathbf{M}^{\pm}$ ()	h_{11} , h_{12}	$SH_{1,2}$	g_{11} , g_{12}	$SG_{1,2}$	f_{11} , f_{12}	$SF_{1,2}$	$\mathbf{P}_{pre}$	$\mathbf{P}_{post}$
$\mathbf{M}^{-}$ $(\overline{P_1})$	$-(2^{-5}+2^{-4}+2^{-3})$ $+(2^{-3}+2^{-2}+2^{-1})$	$+,-$	$-(2^{-10}+2^{-9}+2^{-2})$ $-(2^{-8}+2^{-5}+2^{-3})$	$+,-$	$-(2^{-4}+2^{-3}+2^{-1})$ $-(2^{-6}+2^{-4}+2^{-1})$	$-,-$	[2 4 1 3]	[1 3 2 4]
$\mathbf{M}^{+}$ $(\overline{P_2})$	$+(2^{-8}+2^{-6})$ $+(2^{-7}+2^{-5}+2^{-2})$	$+,-$	$+(2^{-5}+2^{-4}+2^{-3})$ $+(2^{-5}+2^{-4}+2^{-3})$	$+,-$	$-(2^{-8}+2^{-5}+2^{-2})$ $+(2^{-8}+2^{-6}+2^{-5})$	$+,+$	[2 4 3 1]	[1 4 2 3]
$\mathbf{M}^{+}$ $(\overline{P_3})$	$-(2^{-4}+2^{-3}+2^{-2})$ $-(2^{-6}+2^{-3}+2^{-1})$	$+,+$	$+(2^{-7}+2^{-4}+2^{-3})$ $-(2^{-6}+2^{-3}+2^{-2})$	$-,-$	$+(2^{-5}+2^{-2}+2^{-1})$ $+(2^{-12}+2^{-10})$	$-,-$	[1 2 3 4]	[1 2 3 4]
$\mathbf{M}^{-}$ $(\overline{Q_1})$	$-(2^{-6}+2^{-4}+2^{-2})$ $+(2^{-6}+2^{-5}+2^{-2})$	$+,+$	$-(2^{-8}+2^{-6}+2^{-5})$ $+(2^{-8}+2^{-6}+2^{-4})$	$+,-$	$+(2^{-5}+2^{-3}+2^{-2})$ $+(2^{-7}+2^{-4}+2^{-3})$	$+,-$	[4 1 3 2]	[1 3 4 2]
$\mathbf{M}^{+}$ $(\overline{Q_2})$	$+(2^{-6}+2^{-5}+2^{-1})$ $-(2^{-5}+2^{-3}+2^{-2})$	$+,+$	$+(2^{-7}+2^{-6}+2^{-5})$ $+(2^{-8}+2^{-5}+2^{-4})$	$+,+$	$-(2^{-7}+2^{-3}+2^{-1})$ $+(2^{-8}+2^{-7}+2^{-2})$	$+,+$	[1 4 3 2]	[1 4 3 2]
$\mathbf{M}^{-}$ $(\overline{Q_3})$	$-(2^{-4}+2^{-2}+2^{-1})$ $+(2^{-7}+2^{-6}+2^{-3})$	$+,-$	$-(2^{-5}+2^{-4}+2^{-1})$ $+(2^{-8}+2^{-4}+2^{-1})$	$-,-$	$+(2^{-7}+2^{-4}+2^{-3})$ $-(2^{-4}+2^{-2}+2^{-1})$	$+,-$	[3 1 2 4]	[1 4 2 3]

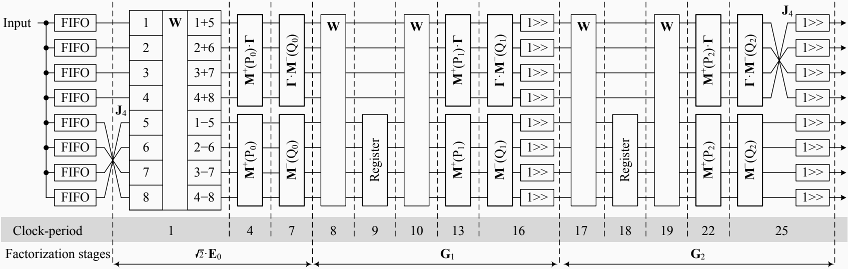
FPGA implementation of LP PMI Int-Q-PUFB for $N=3$ (bit-parallel DA_Σ)

FPGA architecture of given block-lifting based Q-MUL (Quaternion P_1)



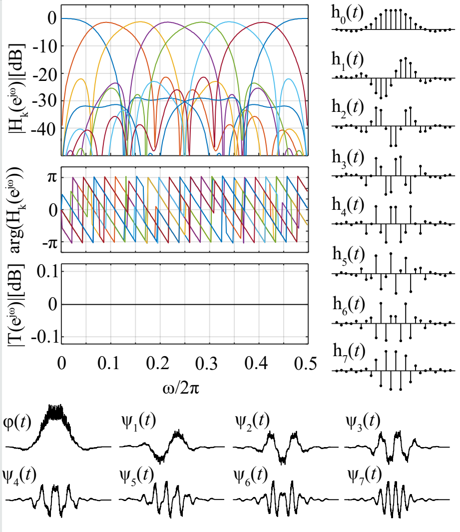


Filter bank structure



Design example: LP PMI Int-Q-PUFB for $B = 12$, $K = 3$

8-channel LP PMI 8×24 Int-Q-PUFB

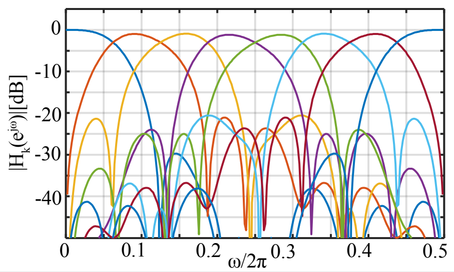


- $CG = 9.49$ dB, $\varepsilon_{SBE} = -21.45$ dB
- $DC\ att. = -49.21$ dB, $B = 16$, $K = 3$

Floating point precision Q-PUFB:

$\varepsilon_{SBE} = -20.56$ dB, $CG = 9.36$ dB,

$DC\ Att. = -313.0712$ dB



FPGA resource utilization:

Scheme	Occupied Slices	Slice Registers	Slice LUTs	LUT-FF pairs
Proposed Q-MUL	159	217	431	469
Int-Q-PUFB Analysis	1733	3562	5607	5977
Int-Q-PUFB Synthesis	1596	3623	5615	5977

Comparison of published LP PUFBs with Int-Q-PUFB

Transform format $M \times L$

M	L	Transform	CG	ε_{SBE}	$DC\ Att.$	precision
8	16	Tran et al.	9.22	−19.4	<-300	Floating p.
		Tran et al. *	9.26	−17.7	<-300	Floating p.
		DCT-2 *	9.27	−18.0	<-300	Floating p.
		WHT-2 *	9.27	−18.0	<-300	Floating p.
		Int-Q-PUFB	**9.44**	**−18.4**	**−40**	$\boldsymbol{B=16, K=3}$
8	24	Oraintara et al. *	9.36	−19.5	<−300	Floating p.
		DCT-3 *	9.38	−19.3	<-300	Floating p.
		WHT-2-3 *	9.38	−19.3	<-300	Floating p.
		Q-PUFB	**9.37**	**−21.1**	$\boldsymbol{<-316}$	**Floating p.**
		Int-Q-PUFB	**9.49**	**−21.3**	**−49**	$\boldsymbol{B=12, K=3}$
8	32	DCT-2-4	9.41	−23.8	<−300	Floating p.
		WHT-4 *	9.46	−18.9	<-300	Floating p.
		Int-Q-PUFB	**9.48**	**−24.8**	**−38**	$\boldsymbol{B=12, K=3}$

(*) Bodong Li; Xieping Gao, A method for initializing free parameters in lattice structure of linear phase perfect reconstruction filter bank, *Signal Processing*, Vol 98, pp 243-251, 2014/5/1.

Image coding results for LP PMI 8×24 Int-Q-PUFB for $N = 3$, $B = 16$, $K = 3$

Wavelet coefficients for "Lena" test image

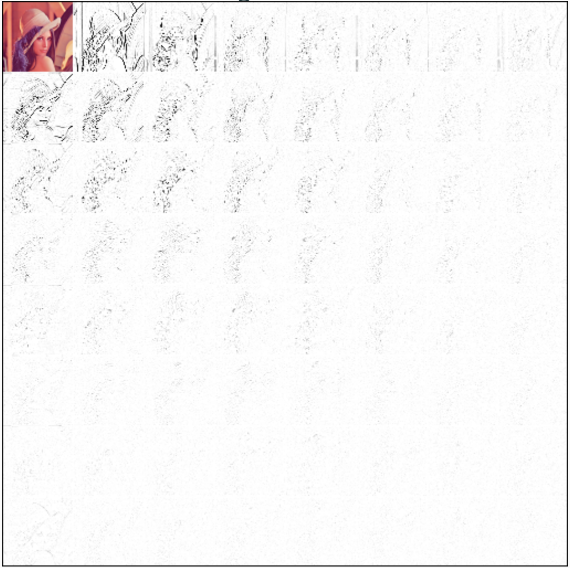


Image coding results for LP PMI 8×24 Int-Q-PUFB for $N = 3$, $B = 16$, $K = 3$

The Table shows the comparisons of PSNRs at various bitrates for two 512×512 8-bit test images, *Lena* and *Barbara*, between the given 8×24 Int-Q-PUFB ($CG =$ 9.61 dB) and published LP PUFBs [5]: 8×16 PUFB ($CG =$ 9.35 dB) and 8×16 BOFB ($CG =$ 9.62 dB) and 8×24 BOFB ($CG =$ 9.68 dB), and also 8-channel 16-tap PUFB based on lapped orthogonal transform (LOT).

Filter bank	"Lena" $[bpp]$			"Barbara" $[bpp]$		
	0.25	0.5	1.0	0.25	0.5	1.0
8×16 PUFB	33.17	36.57	39.73	29.20	33.31	$38,30$
8×16 LOT	32.91	36.13	39.28	29.05	33.04	$37,84$
8×24 PUFB	33.36	36.64	39.94	29.43	33.53	$38,34$
8×24 GenLOT	33.25	36.54	39.82	29.31	33.55	$38,31$
8×16 BOFB	33.43	36.67	39.73	29.31	33.33	$38,26$
8×16 GLBT	33.35	36.62	39.70	29.23	33.28	$38,19$
8×24 BOFB	33.53	36.82	39.84	29.66	33.63	$38,38$
8×24 GLBT	33.32	36.61	39.68	29.29	33.29	$38,18$
8×24 Q-PUFB	**34.65**	**37.15**	**39.41**	**30.58**	**34.51**	**$38,19$**
JPEG2000 $(9/7)$	33.25	36.29	39.25	27.73	31.41	$36,56$

Our designed 8×24 Int-Q-PUFB has a better PSNR performance than the corresponding filter banks, especially for image with relatively strong highpass components.

[5] T. Uto, T. Oka, and M. Ikehara, "M-channel nonlinear phase filter banks in image compression: Structure, design, and signal extension," *IEEE Trans. Signal Process.*, vol. 55, no. 4, pp. 1339–1351, April 2007.

Conclusions and future work

Conclusions

- in this paper we introduced a generalized block-lifting structure using adder-based $\mathrm{DA}_{\sum}$ as a block of quaternion multiplier in the Q-PUFB lattice structure
- possible to implement integer-to-integer transform in fixed point arithmetic with very short critical-path (in case $K = 3$ amounts to only three addition/subtraction operations)
- hardware constrained SOPOT coefficient optimization allows to reduce distortions in compare with direct quantization
- shown approach can be applied for the lossy-to-lossless (L2L) image coding

Future work

- Develop non-separable Q-PUFB transform to reduce image processing latency
- Generalize synthesis process for M-channel Int-Q-PUFB, where $M > 8$

Thank you for attention
Questions?