

Reconfigurable Computing for DSP... Algorithms, Architectures, Applications and Power Modeling on FPGAs

Dr. Abbes Amira

**Senior Lecturer in Parallel and Reconfigurable Computing
for Computer Vision**

CMOS Emerging Technologies Workshop

July 19-21, 2006, Banff, Alberta, Canada

- ❑ Digital Image and Signal Processing: Design and Implementation
- ❑ Custom/Reconfigurable Computing using FPGAs, Hardware/Software Co-Design, System on Chip, Embedded Systems
- ❑ Medical Imaging: Segmentation, Registration and Compression
- ❑ Image and Video Processing based on Statistical, Multiresolution and Hybrid Approaches: Noise removal, Segmentation, Compression and Face Recognition, Computer Vision for Robotics
- ❑ Low Power Architectures for Image and Mobile Applications
- ❑ Information Retrieval

- ❑ **Reconfigurable Computing**
 - ❑ **Definitions**
 - ❑ **FPGA's- The Basis of Current RC**
- ❑ **FMAT Project**
 - ❑ **Motivations and Objectives**
 - ❑ **System Architecture**
- ❑ **Implementation Strategies for FMAT Cores**
 - ❑ **Algorithms**
 - ❑ **Architectures**
 - ❑ **FPGA Implementations**
 - ❑ **Applications**
- ❑ **Power Modeling on FPGAs**
- ❑ **Conclusions**

Part 1

Reconfigurable Computing

- What is it?
 - Compute by building a circuit rather than executing instructions.
 - Efficient for long running computations
 - Video and image processing
 - DSP
 - Network processing

Example: $Z[i] = a.X[i] + b.Y[i]$

//program

Load rx, X

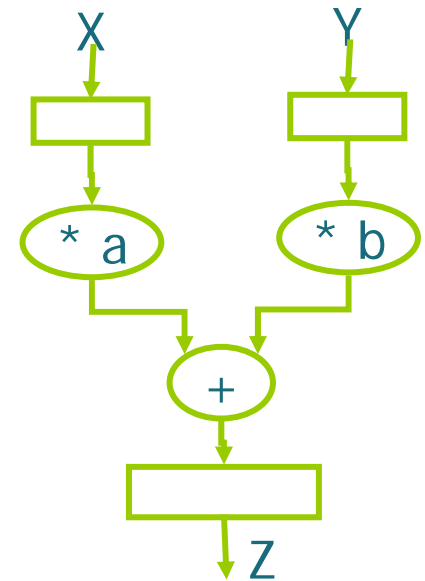
Mpy r1, rx, ra

Load ry, Y

Mpy r2, ry, rb

Add r3, r1, r2

Store r3, Z



Using *reconfigurable electronics* to build systems with advantages over *conventional technology* ** in any of the areas of:

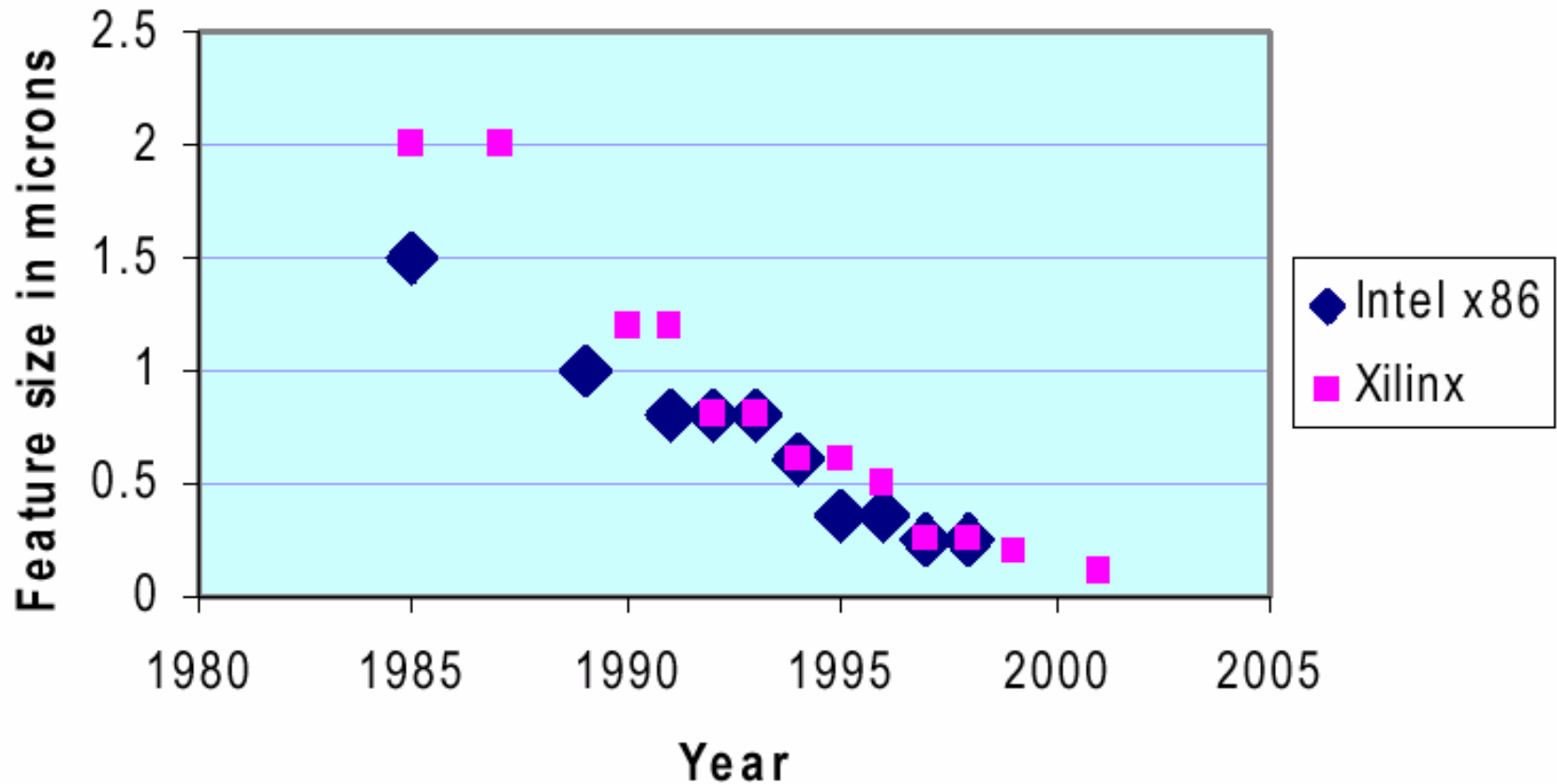
- Time- to- market
- Performance
- Power
- Size/ weight
- Flexibility
- Life cycle cost

**** ASIC's, CPU's, and DSP's**

- ❑ Cost-Effective
- ❑ High-performance
- ❑ Shorter/Simpler Design Cycles
- ❑ Faster time-to-market
- ❑ Longer time in-the-market
- ❑ Backed with a Vast Array of Hard/Soft IP

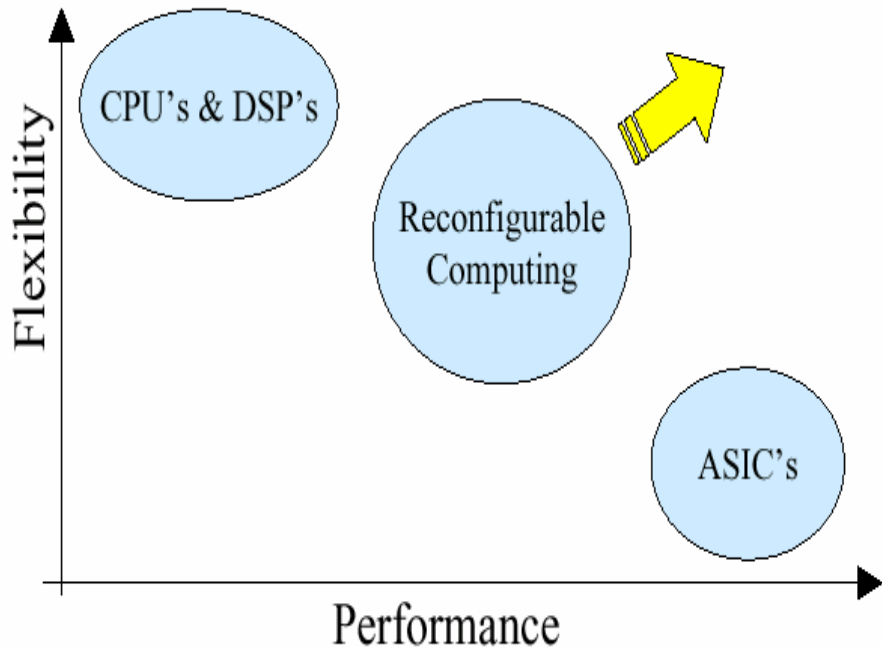
Within the next 5-10 years, you will find programmable logic devices used in almost every piece of electronic equipment...Xilinx

Xilinx FPGA Minimum Features Sizes



Sources: Xilinx data books, WWW pages
[http:// infopad. eecs. berkeley. edu/ CIC/ summary/ local/](http://infopad.eecs.berkeley.edu/CIC/summary/local/)

Performance and Flexibility

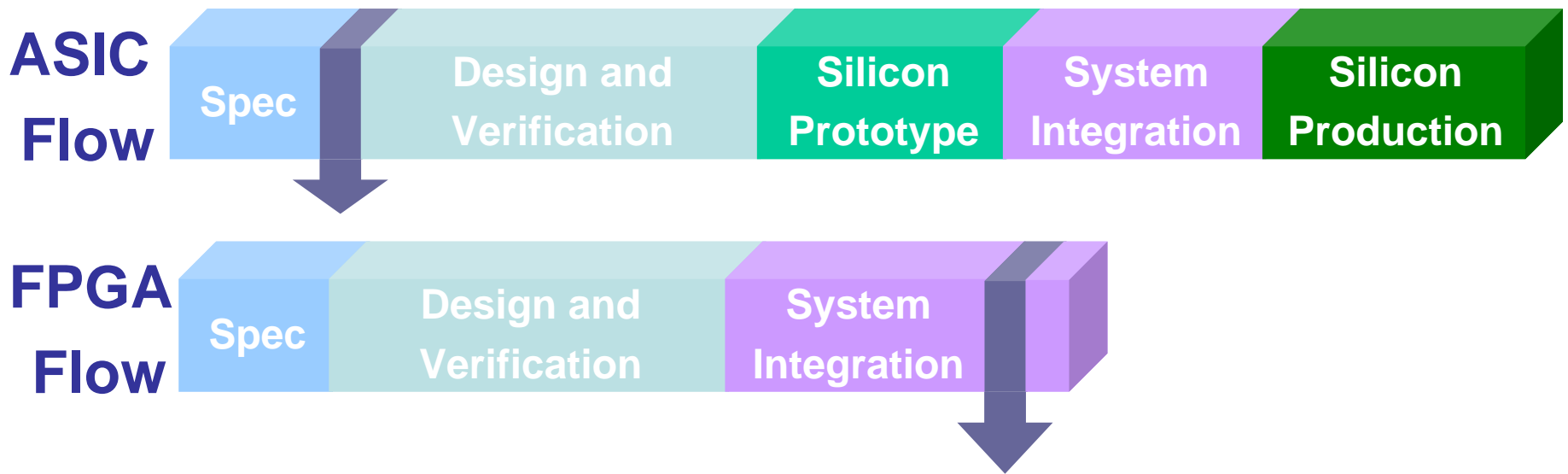


Current research is working to quantify:

- **performance**
- **power**
- **size/ weight**
- **life cycle cost**
- **applicability of RC**
- **Heterogeneous platforms/SoC**

“The performance of ASIC's with the flexibility of programmable processors.”

Simpler/Faster Design Flows



- 2:1 proven Time-to-Market Advantage
- No silicon design or verification steps
- More design flexibility through later design freeze

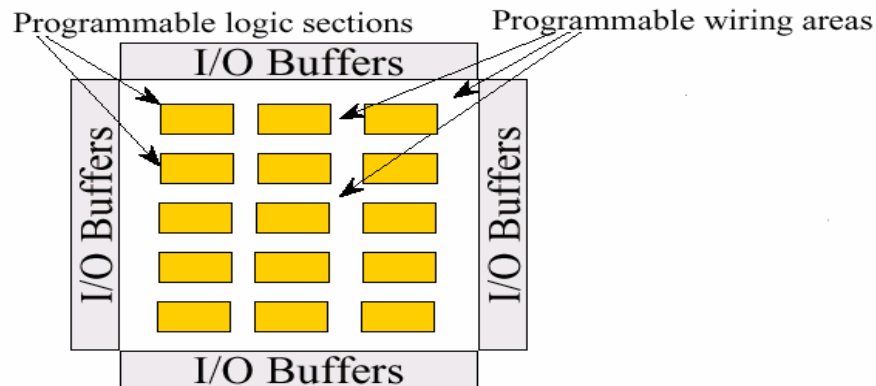
•“By going with *Virtex-II*, we were able to ship our product months earlier than we would have with a traditional ASIC design.”

Brian Ramelson,

Lucent Technologies.. Virtex 5!!!

What is an FPGA?

- ❑ An FPGA is a silicon chip device
- ❑ The FPGA is made up of an array of logic blocks

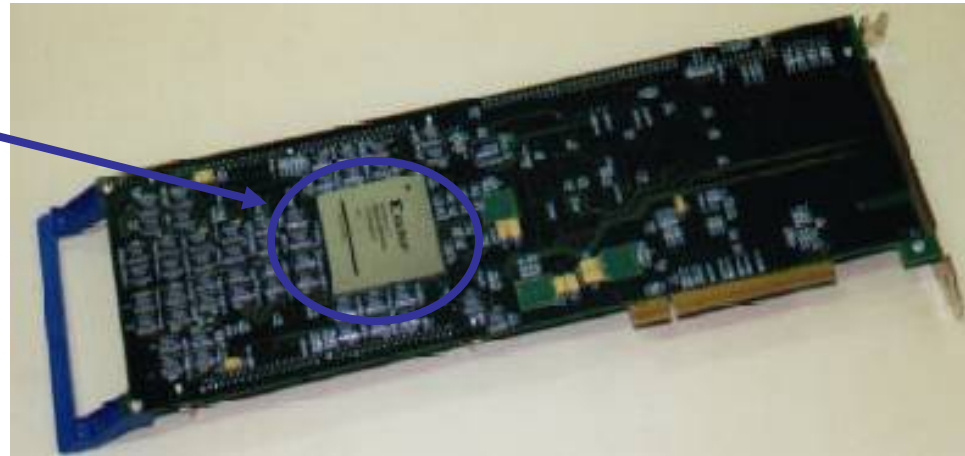


- ❑ The blocks can be configured and connected together by a user to create an electronic circuit
- ❑ Recent FPGAs have extra features...i.e. Virtex II Pro, Virtex 5
 - ❑ Up to four IBM PowerPC RISC processor blocks
 - ❑ Dedicated 25 x 18-bit multipliers enable increased precision, single-precision floating-point, and wide filters from fewer slices for lower cost
 - ❑ Embedded BRAM

What is an FPGA prototyping board?

- ❑ An FPGA prototyping board is a platform that includes the components required for a working FPGA system
- ❑ These boards are used as:
 - ❑ A starting point for quickly prototyping a design in an FPGA
 - ❑ A teaching platform for students who are learning about the technology

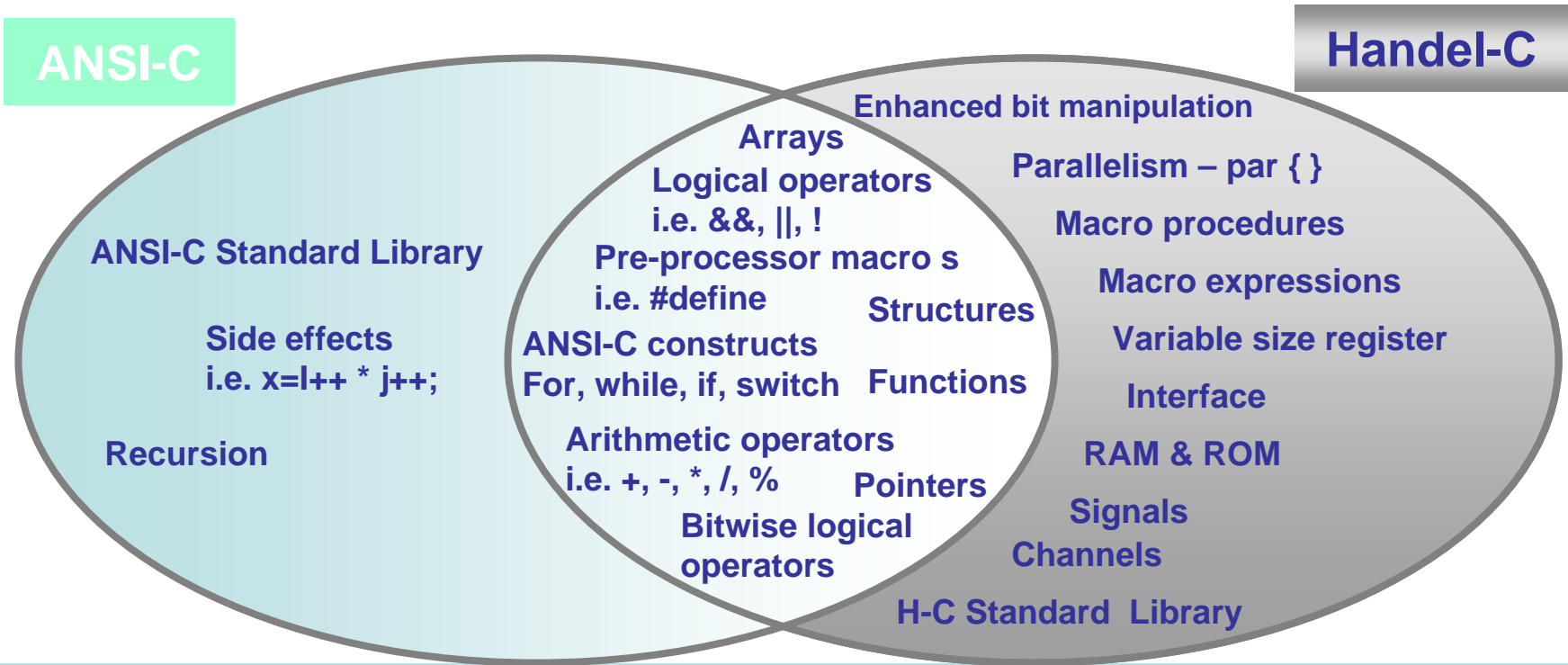
Xilinx V2000E FPGA



RC1000 Board

What is Handel-C?

- ❑ Design to describe software algorithms compiled to hardware
- ❑ ANSI-C extended for hardware



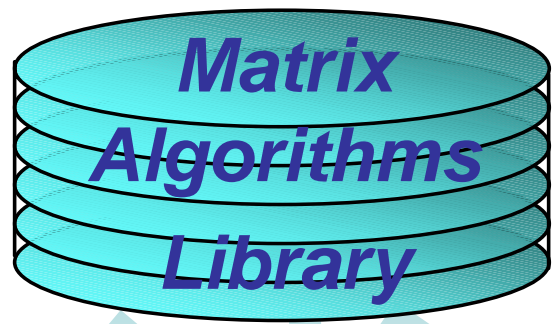
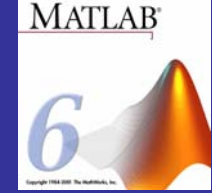
Part 2

FMAT Project

- ❑ To use FPGAs as low cost accelerator for image and signal processing applications based matrix algorithms
- ❑ To design a new heterogeneous environment encompassing Handel-C, MATLAB and HDL design environments.
- ❑ To integrate the new systems with those currently accepted as de facto industry standard High Level Design Languages (VHDL, ...) including the IP cores provided by such environments (example, Xilinx Coregen library)

- ❑ To apply the proposed system for the acceleration and processing of very large images (segmentation, compression, ... etc).
- ❑ To develop novel, efficient and scalable architectures for matrix operations, matrix transforms and matrix decompositions, where both the area and the speed can be estimated for any matrix algorithm with any specific design parameters and design methodology
- ❑ To address the problem of processing large matrix multiplications, transforms and decompositions

VHDL/Schematic Components



Generator

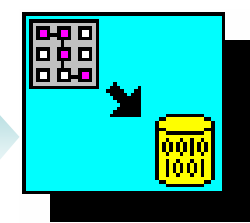
Compile

Simulate



EDIF 1 + EDIF 2 + EDIF 3

Handel-C



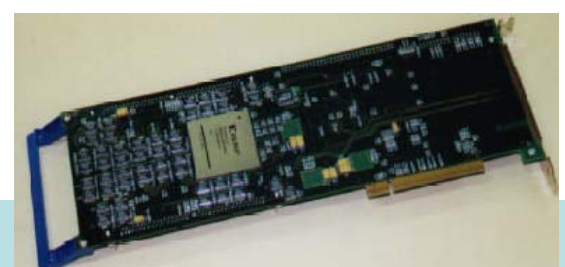
XST

System Architect



- Design Parameters
- Design Methodologies

Configure 11111010010...1000

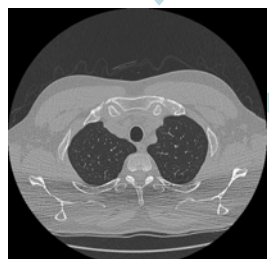


FMAT System

Application User System Architect



- Estimating Performance Measures (Power, Area, Max Frequency...etc)
- Capturing Platform Features at higher level



CSC

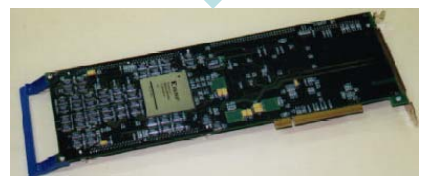
DWT

RLC

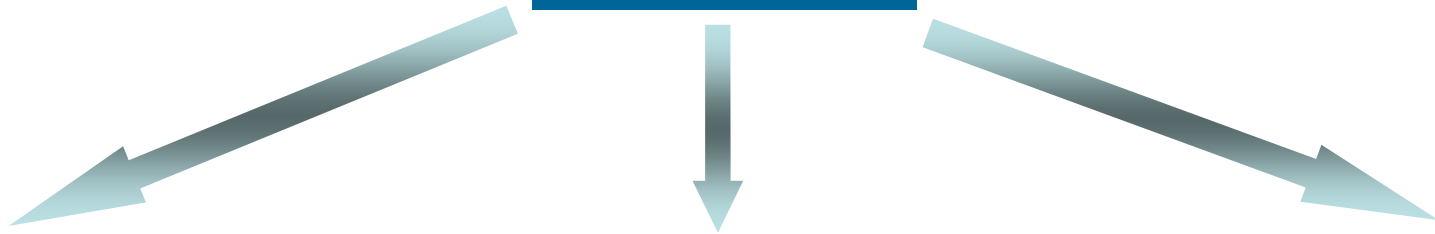
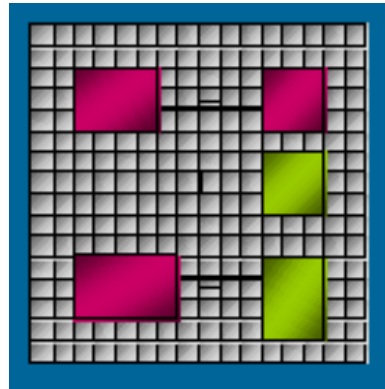


- VHDL
- Handel-C
- Schematic
- Hybrid
- EDIF
- Bitstream

- FPGA Configuration?
- Implementation?
- Reconfiguration?
- Compilation?



- MM
- DCT (1D,2D)
- FFT (1D, 2D)
- DWT (1D, 2D)
- FRAT (1D, 2D)
- SVD
- QR



Array Blocks Routines
Platform

Matrix multiplication
Matrix inversion

DSP, IP
Platform

Matrix transforms:
DCT, DWT, FHT,
DHT, FFT, ...etc

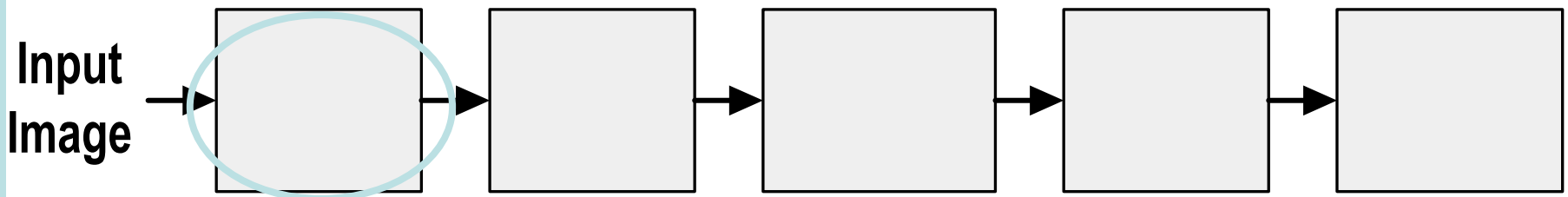
AP
Platform

Matrix decomposition:
SVD, QR, LU

Part 3

Implementation Strategies for FMAT Cores

- ❑ Processing an image in the RGB color space is not the most efficient method
- ❑ The calculation of YCrCb color components from RGB components consumes up to 40% of the processing power in a highly optimised decoder
- ❑ For more efficient compression, many broadcast, video and imaging standards use luminance and color difference video signals such as YCrCb



Baseline JPEG encoder

A color in the RGB color space can be converted to the YCrCb color space using this equation

$$\begin{bmatrix} Y \\ C_r \\ C_b \end{bmatrix} = \begin{bmatrix} 0.257 & 0.504 & 0.098 & 16 \\ 0.439 & -0.368 & -0.071 & 128 \\ -0.148 & -0.291 & 0.439 & 128 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \\ 1 \end{bmatrix}$$

The inverse conversion can be carried out using this equation

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.164 & 1.596 & 0.0 & -222.912 \\ 1.164 & -0.813 & -0.392 & 135.616 \\ 1.164 & 0.0 & 2.017 & -276.8 \end{bmatrix} \times \begin{bmatrix} Y \\ C_r \\ C_b \\ 1 \end{bmatrix}$$

Algorithm

$$c_{ijk} = \sum_{m=0}^2 a_{km} b_{ijm} \quad (0 \leq i \leq N-1, 0 \leq j \leq M-1, 0 \leq k \leq 2)$$

unsigned binary representation

Word length ←

$$b_{ijm} = \sum_{l=0}^{W-1} b_{ijm,l} 2^l \quad (0 \leq i \leq N-1, 0 \leq j \leq M-1, 0 \leq m \leq 2)$$

→ W=8

$$c_{ijk} = \sum_{m=0}^2 a_{km} \left(\sum_{l=0}^{W-1} b_{ijm,l} 2^l \right) = \sum_{l=0}^{W-1} \left(\sum_{m=0}^2 a_{km} b_{ijm,l} 2^l \right)$$

$$c_{ijk} = \sum_{l=0}^7 Z_l 2^l$$

$$Z_l = \sum_{m=0}^2 a_{km} b_{ijm,l}$$

- Depends on the $b_{ijk,l}$ values
- Has 2^4 possible values
- An input set of 4 bits ($b_{ij0,l}, b_{ij1,l}, b_{ij2,l}, b_{ij3,l}$) is used to retrieve the corresponding Z_l values

Algorithm

Since the element $b_{ij3} = 1$:

$$b_{ij3,l} = \begin{cases} 1 & \text{for } l = 0 \\ 0 & \text{for } l \neq 0 \end{cases}$$

$$c_{ijk} = \sum_{l=0}^7 Z_l^* 2^l + a_{k3}$$

$$Z_l^* = \sum_{m=0}^2 a_{km} b_{ijm,l}$$

↓

- Has 2^3 possible values
- An input set of 3 bits ($b_{ij0,l}, b_{ij1,l}, b_{ij2,l}$) is used to retrieve the corresponding Z_l^* values

Algorithm

$$c_{ijk} = (a_{k0}b_{ij0,0} + a_{k1}b_{ij1,0} + a_{k2}b_{ij2,0}) 2^0 +$$

$$(a_{k0}b_{ij0,1} + a_{k1}b_{ij1,1} + a_{k2}b_{ij2,1}) 2^1 +$$

$$(a_{k0}b_{ij0,2} + a_{k1}b_{ij1,2} + a_{k2}b_{ij2,2}) 2^2 +$$

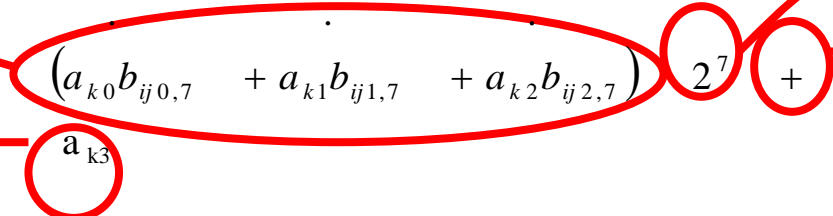
PE

Left shifter

Adder

ROM

Constant input
for first adder

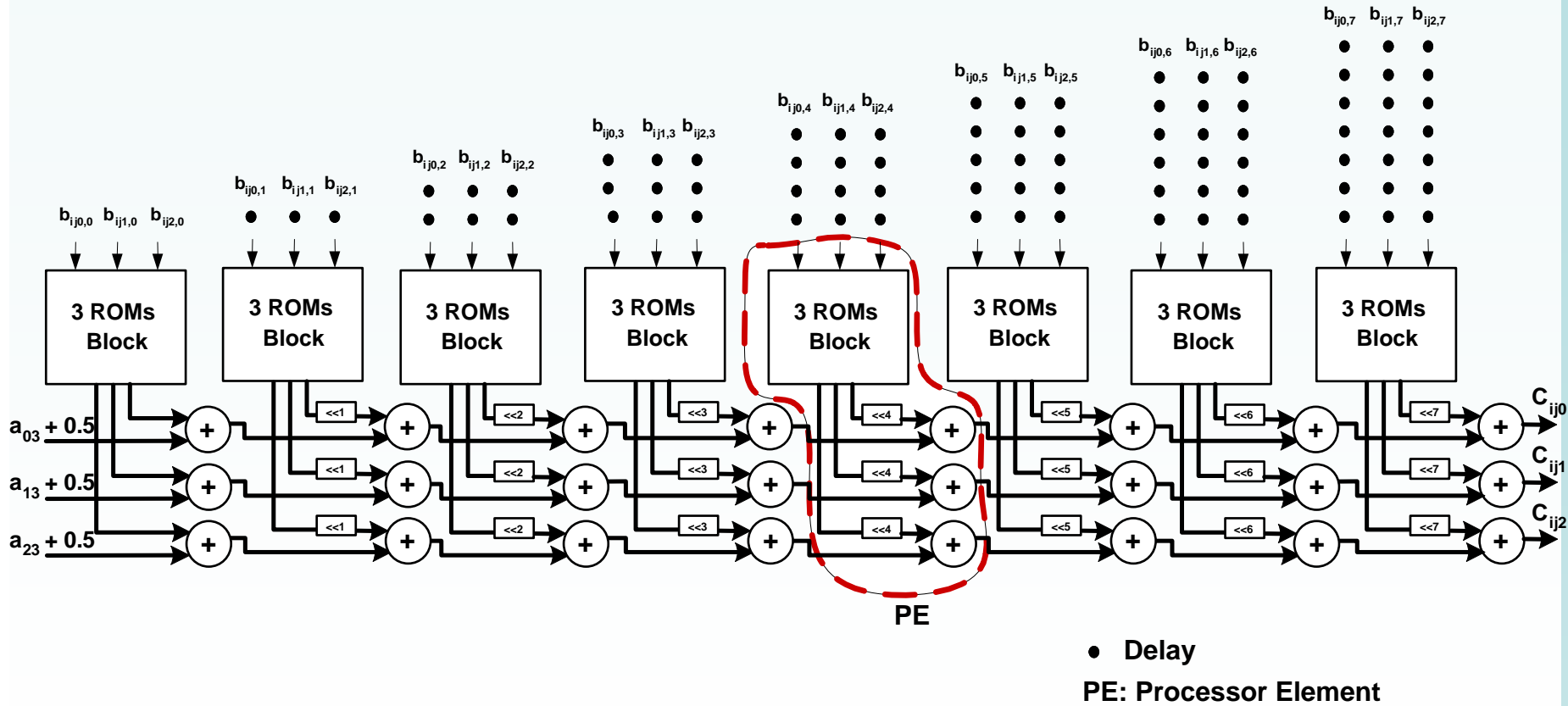


$b_{ij0,l}$	$b_{ij1,l}$	$b_{ij2,l}$
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

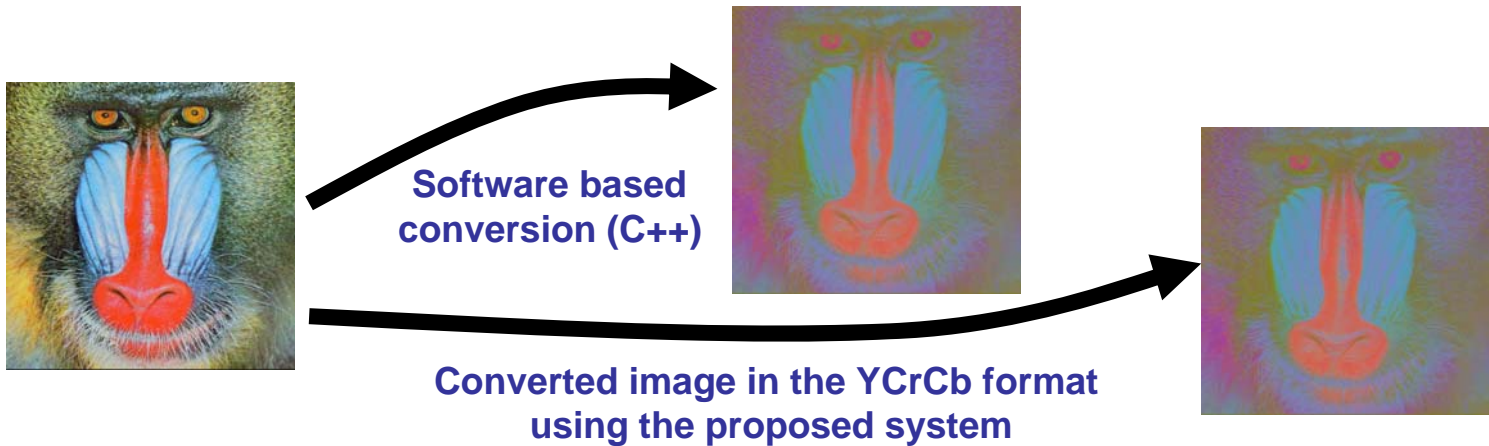
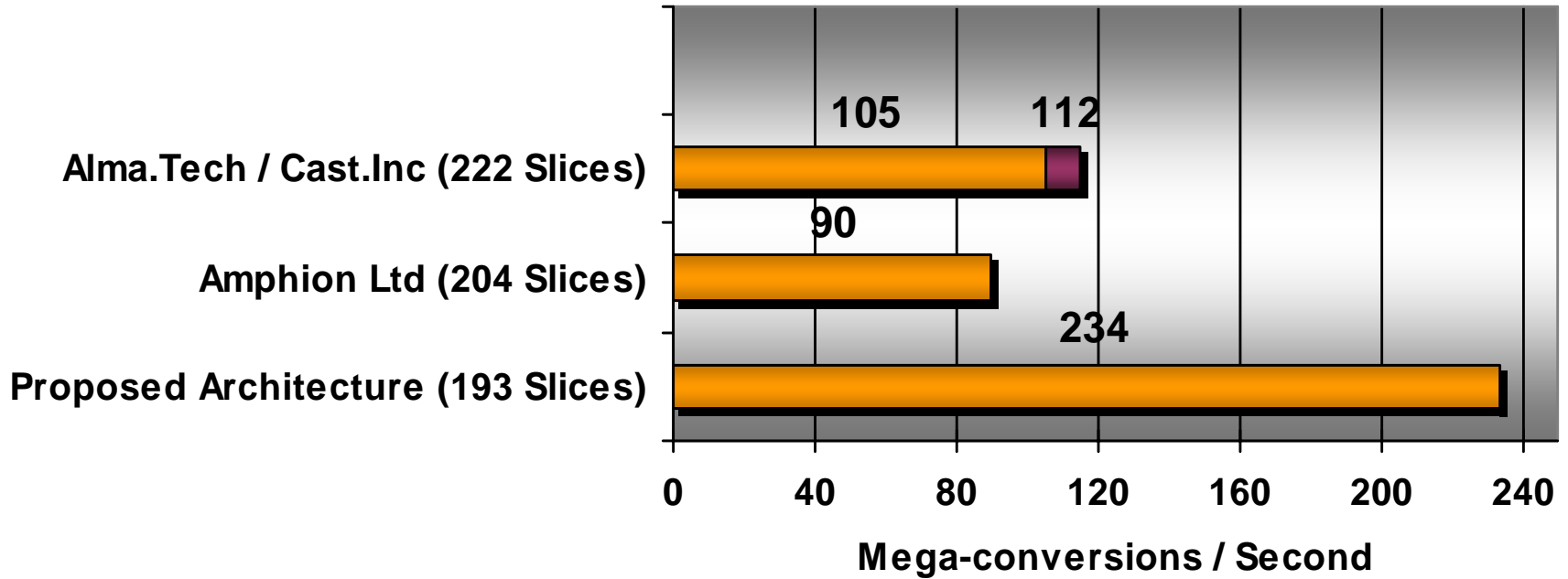
The content of the ROM i

- 0
- a_{i2}
- a_{i1}
- $a_{i1} + a_{i2}$
- a_{i0}
- $a_{i0} + a_{i2}$
- $a_{i0} + a_{i1}$
- $a_{i0} + a_{i1} + a_{i2}$

Architecture



Proposed architecture based DA principles

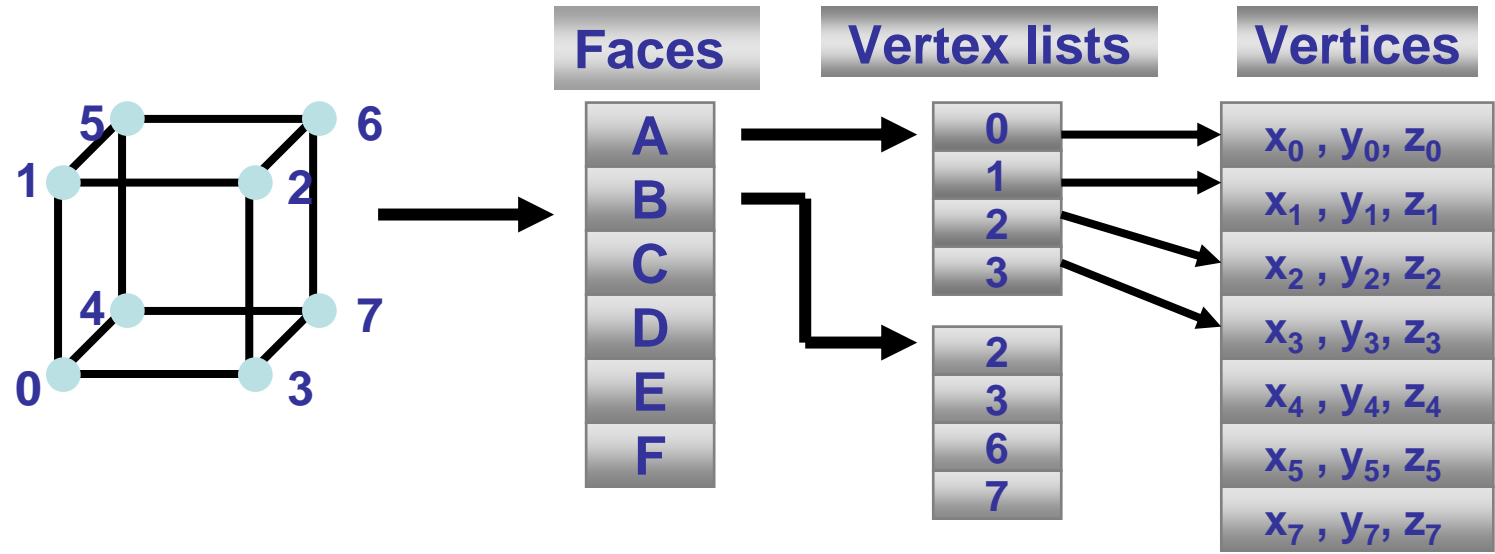


- ❑ Scientific and engineering applications
 - ❑ Computer Aided Design and Manufacturing
 - Computer simulation
 - ❑ Robotics
 - ❑ Space science

- ❑ Art and Entertainment
 - ❑ Games
 - ❑ Cartoon and animation
 - ❑ Multimedia and animation

3D Affine Transformations

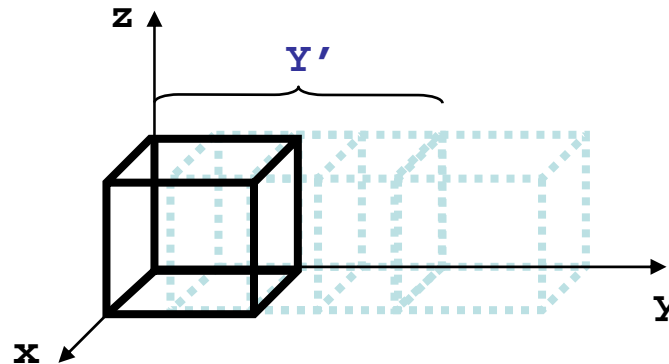
- In computer graphics the most popular method for representing an object is the polygon mesh model
- A polygon mesh model is a structure that consists of polygons represented by a list of (x, y, z) coordinates that are the polygon vertices



- 3D affine transformations are the transformations that involve rotation, scaling and translation
- An affine transformation can be presented using matrix notation

Translation

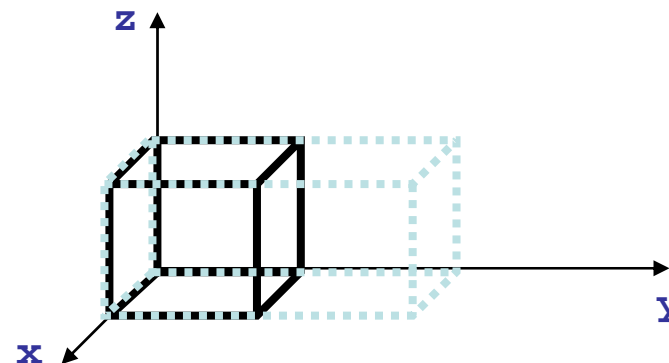
$$T = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$(T_x, T_y, T_z) = (0, Y', 0)$$

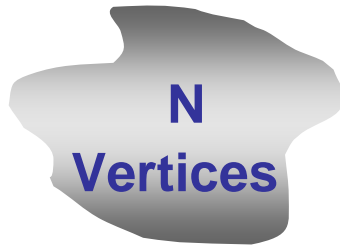
Scaling

$$S = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$(S_x, S_y, S_z) = (0, 1/2, 0)$$

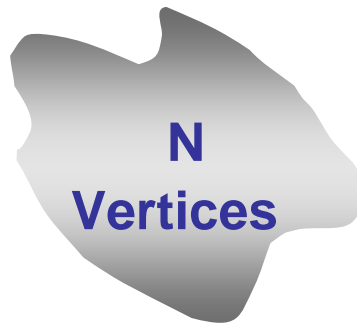
Algorithm



$$OP = \begin{bmatrix} x_0 & x_1 & \dots & x_N \\ y_0 & y_1 & \dots & y_N \\ z_0 & z_1 & \dots & z_N \\ 1 & 1 & \dots & 1 \end{bmatrix}$$

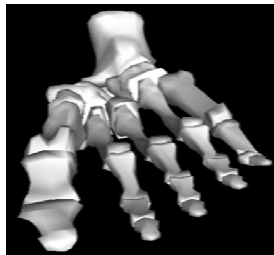


**Applying
a transformation**



$$\begin{bmatrix} x_0^* & x_1^* & \dots & x_N^* \\ y_0^* & y_1^* & \dots & y_N^* \\ z_0^* & z_1^* & \dots & z_N^* \\ 1 & 1 & \dots & 1 \end{bmatrix} = T \times \begin{bmatrix} x_0 & x_1 & \dots & x_N \\ y_0 & y_1 & \dots & y_N \\ z_0 & z_1 & \dots & z_N \\ 1 & 1 & \dots & 1 \end{bmatrix}$$

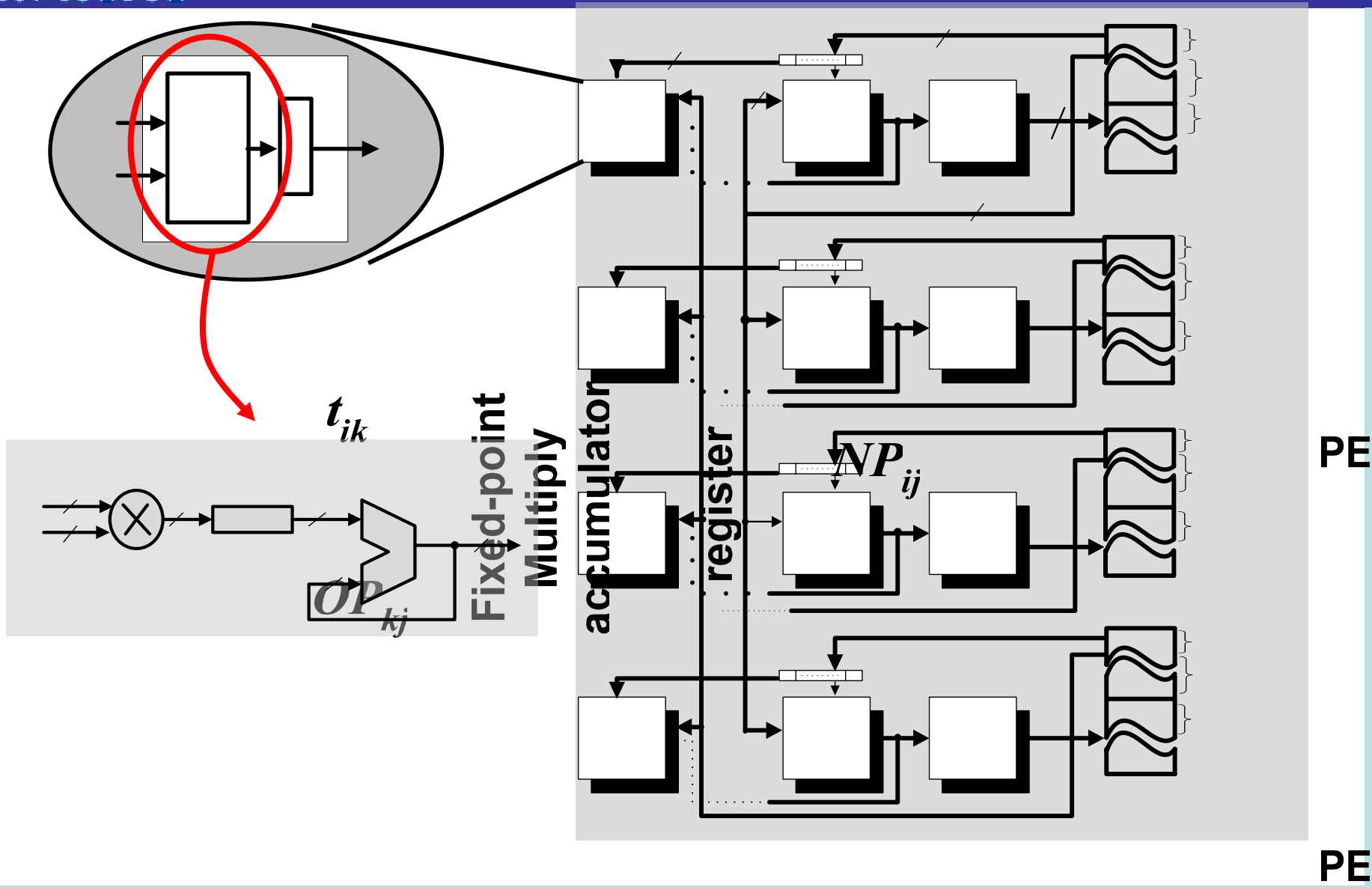
2162 vertices



5597 vertices



Architecture

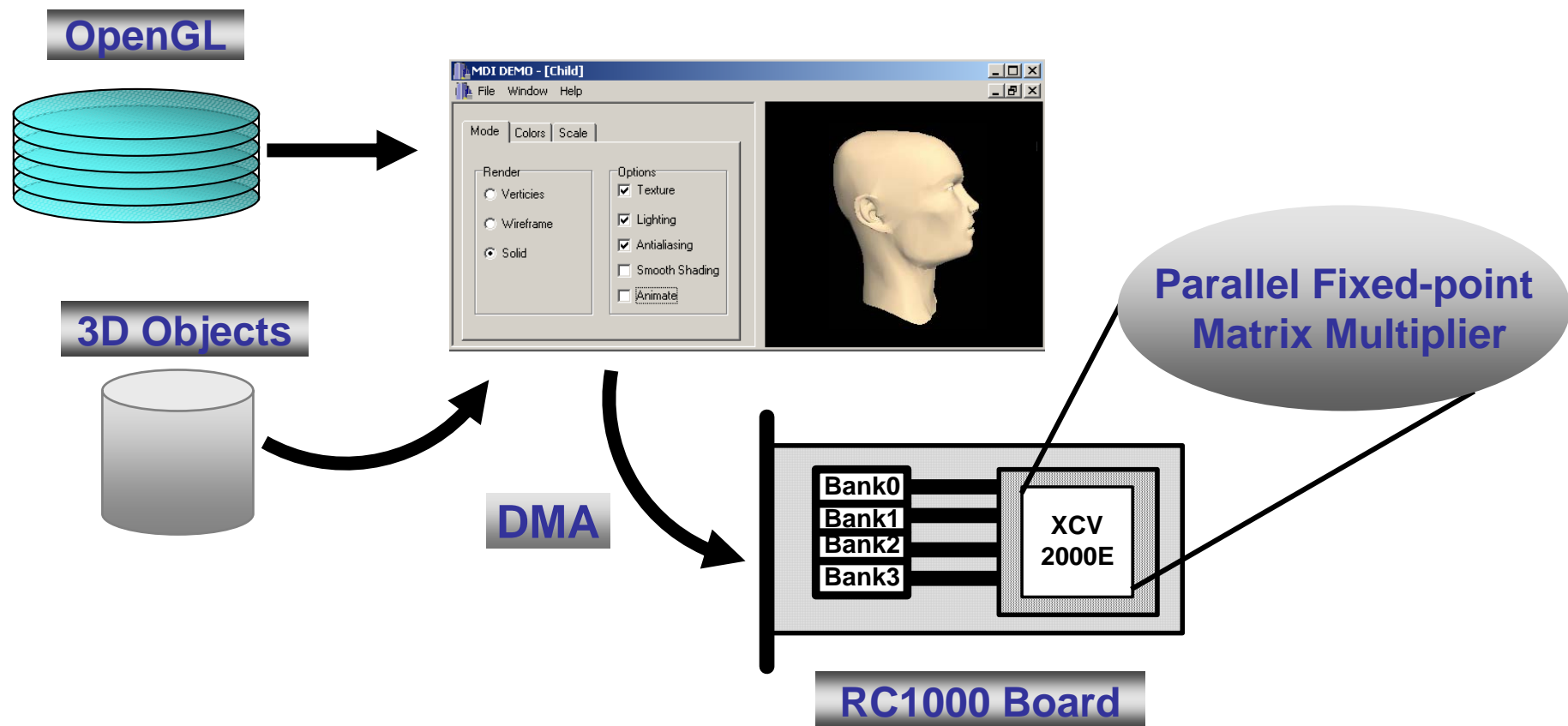


Implementation Report

MAC used	Number of PEs	Area (%)	Speed (Mhz)
Celoxica fixed-point library	4	17	22
	8	35	20
	16	75	17
Xilinx's CoreGen	4	10	35
	8	20	30
	16	42	24

MAC used	Number of PEs	Area (slices)		18 × 18 Mults used		Speed (MHz)	
		V-II	V-II Pro	V-II	V-II Pro	V-II	V-II Pro
Celoxica fixed-point library	4	2795	2780	-	-	36	52
	8	6057	5997	-	-	30	45
	16	14443	14360	-	-	26	38
Xilinx's CoreGen	4	931	925	16	16	83	114
	8	2795	2624	32	32	78	110
	16	5591	5494	56	56	71	102

	MAC used	Number of PEs	Speed (MHz)		Computation time (μs)	
			V-II	V-II Pro	V-II	V-II Pro
FPGA	Celoxica fixed point library	4	36	52	134.74	93.28
		8	30	45	89.88	59.92
		16	26	38	36.33	24.85
	Xilinx's CoreGen	4	83	114	58.43	42.55
		8	78	110	34.57	24.51
		16	71	102	13.30	9.26
RADEON FSC 32 MB graphics card	-	-	-	-	14.81	



Proposed system for 3D affine transformations on FPGA

□ The SVD of a Matrix A is a decomposition of the form

$$\square A = U D V^T$$

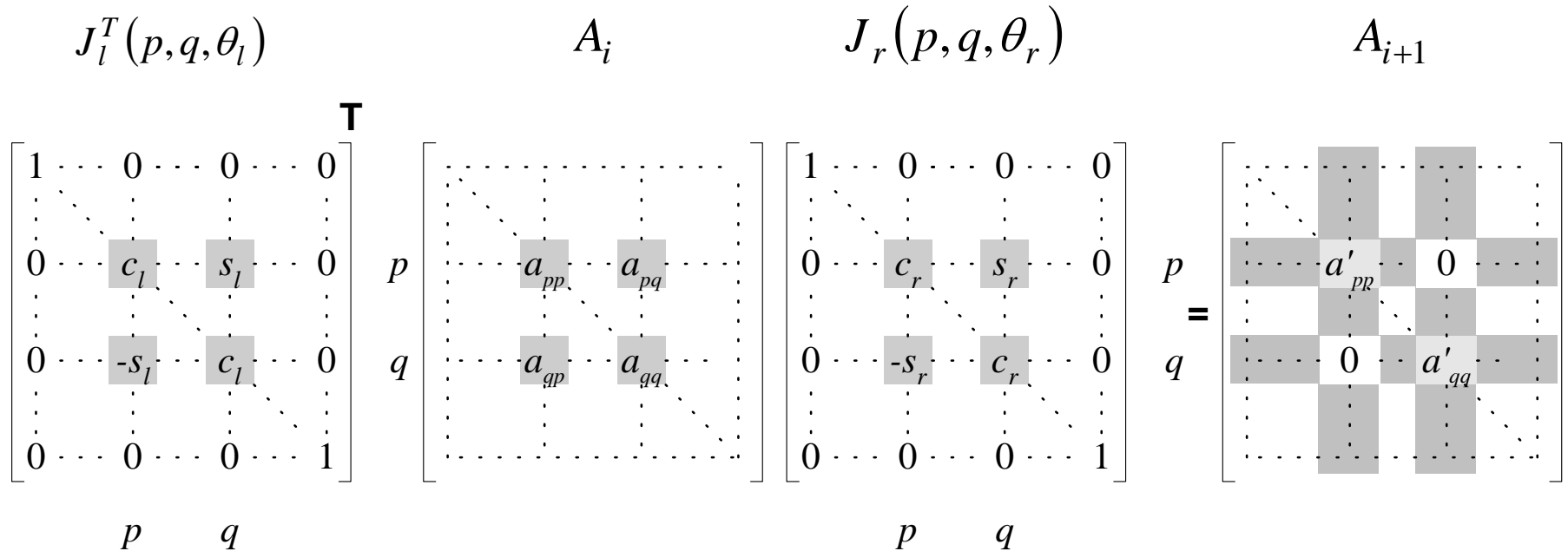
□ The EVD of a *symmetric* Matrix A is a decomposition of the form:

$$\square A = V D V^T$$

□ Where U and V are orthogonal matrices and D a diagonal matrix

□ The Jacobi algorithm generates the matrices U and V by performing a sequence of orthogonal two sided plan rotations (2D rotations) to the input matrix.

The Jacobi SVD/EVD Algorithm



$C = \cos \theta$, $S = \sin \theta$
 $l = \text{left}$, $r = \text{right}$

$$D = A_n \ ; \ V = \prod J_i^l \ ; \ U = \prod J_i^r$$

Two sided plan rotation

For the EVD $J_l^T = J_r$

- Each two sided plan rotation affects only two columns and two rows
- $N/2$ rotations can be performed in parallel (if N is the matrix size)
- A parallel two sided Jacobi algorithm can be applied

- ❑ Square Systolic array
- ❑ Parallel two sided Jacobi Algorithm
- ❑ $(N/2)^2$ processors
- ❑ $O(N \log N)$ time complexity

Synchronize the processors according to availability of new data instead of an iteration step synchronization strategy:

New algorithm processor

If {diagonal processor}

Solve SVD (EVD)

Output rotation parameters

Apply rotations to sub-matrix

Output data (sub-matrix)

Wait for new data

Else {off-diagonal processor}

Wait for new rotation parameters

Output rotation parameters

Apply rotations to sub-matrix

Output data (sub-matrix)

Wait for new data

Influence of the matrix size

Design statistics for matrix of 6 x 6

Matrix size		4	6	8
Word length				
14	Max Freq (MHz)	87.73	96.15	85.98
	Area (slices)	4827	9911	16681
	Area (%)	25	51	86
16	Max Freq (MHz)	96.38	88.54	84.44
	Area (slices)	5507	11296	19013
	Area (%)	28	58	99

Word Length	Maximum Frequency (MHz)	Area (slices)	Area (%)
14	96.15	9911	51
16	88.54	11297	58
18	84.77	12933	65
20	94.88	13923	72
22	85.40	15141	78
24	87.24	16374	85
26	83.98	17633	91

$$Y = D \cdot S + B$$

$$R_y = D(\theta)R_sD(\theta)^T + \sigma^2I_N$$

The matrix R_y is estimated as:

$$R_y = \frac{1}{\alpha} Y \cdot Y^T$$

$$g(\theta) = \frac{\sum_{n=1}^p \left| d(\theta)^T \hat{v}_n \right|^2}{\sum_{n=p+1}^N \left| d(\theta)^T \hat{v}_n \right|}$$

\hat{v}_n **Singular vectors of R_y**

Anb SVD Should be Calculated.

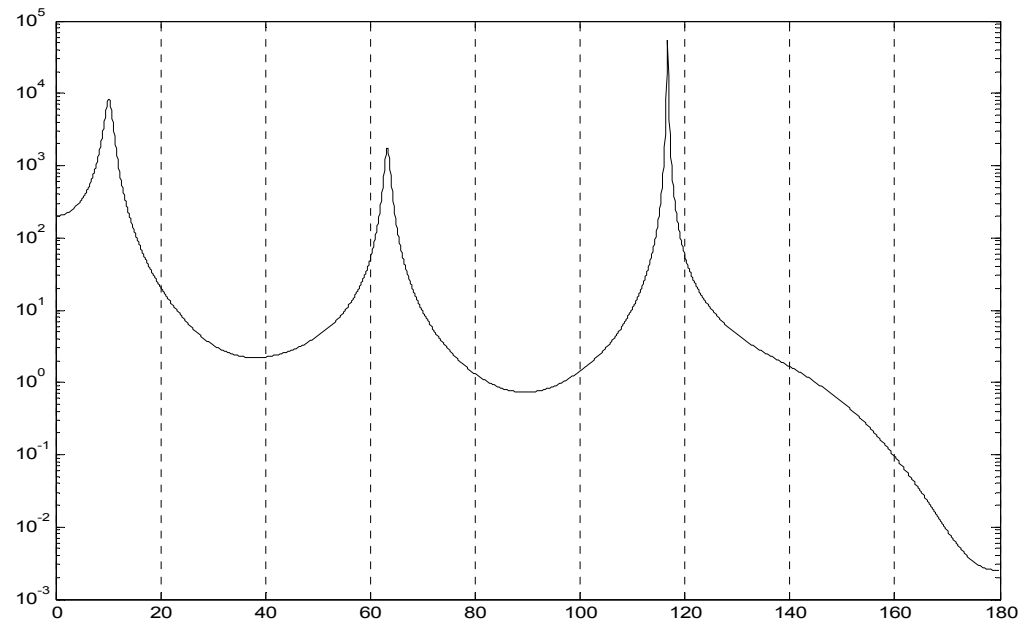
The implementation of MUSIC can be divided into two parts:

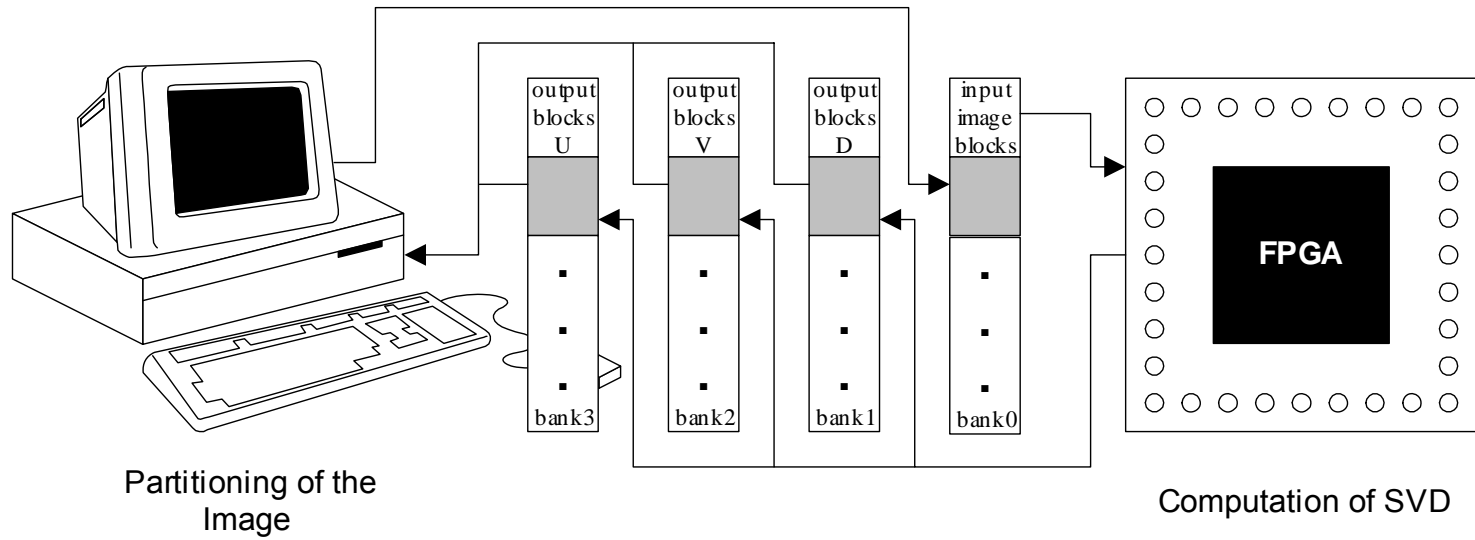
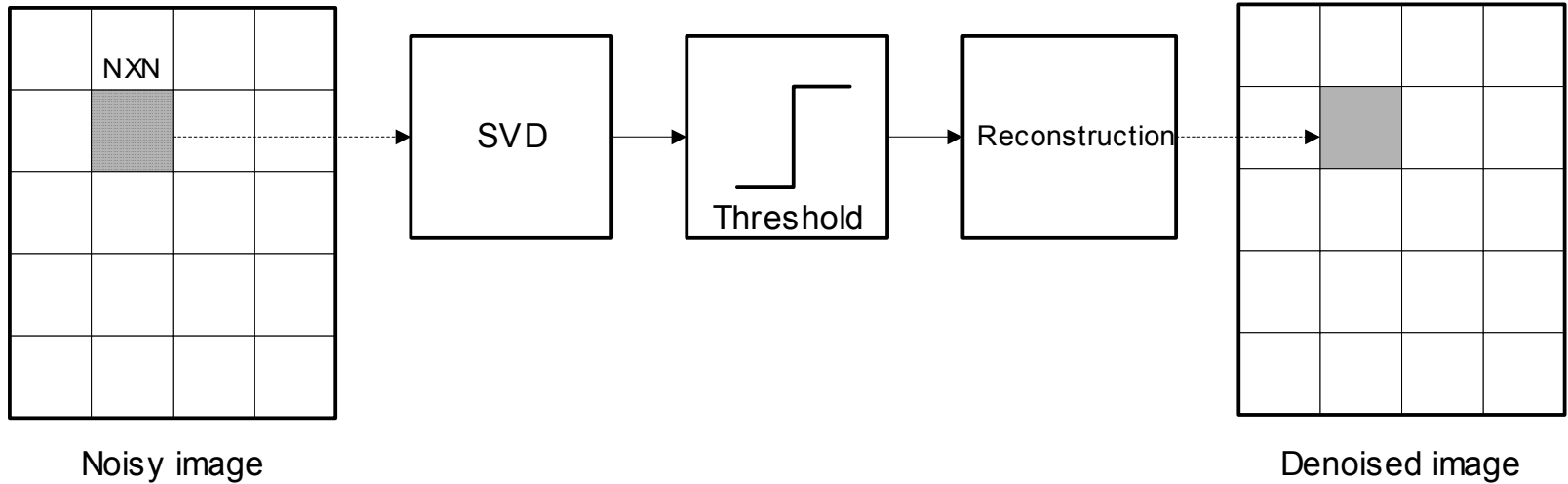
1- Computation of the SVD of Y (Implementation on FPGA)

2- Computation of the localisation function and the arrival angles (host PC, BC++ has been used for the development)

Source localisation
($P=3, W=16, N=8, SNR=10db$)

The computation time is 0.123ms, which means more than 8000 SVD could be computed per second





The computation time of the SVD of one block (8x8) is 0.123ms (fmax=84.44 MHz) which means 8130 blocks per second can be processed.



A. Noised Image



B. Low pass filter



C. SVD filter

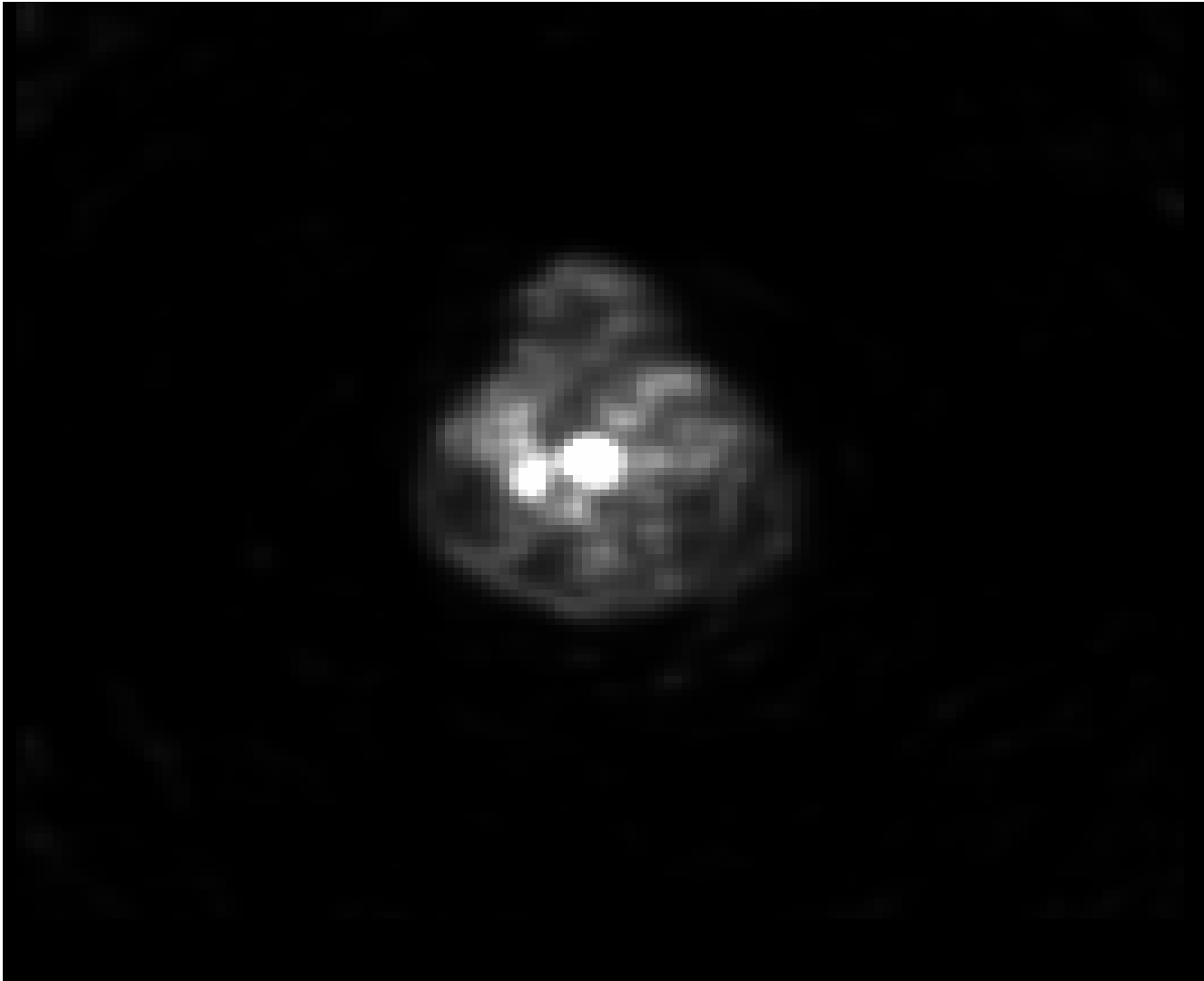


D. Median filter

- Discrete Wavelet Transforms have become powerful tools in many real world applications
 - Image/Video Compression (JPEG2000, MPEG-4)
 - Image/Video Enhancement, Segmentation
 - Telecommunication
 - Biomedicine

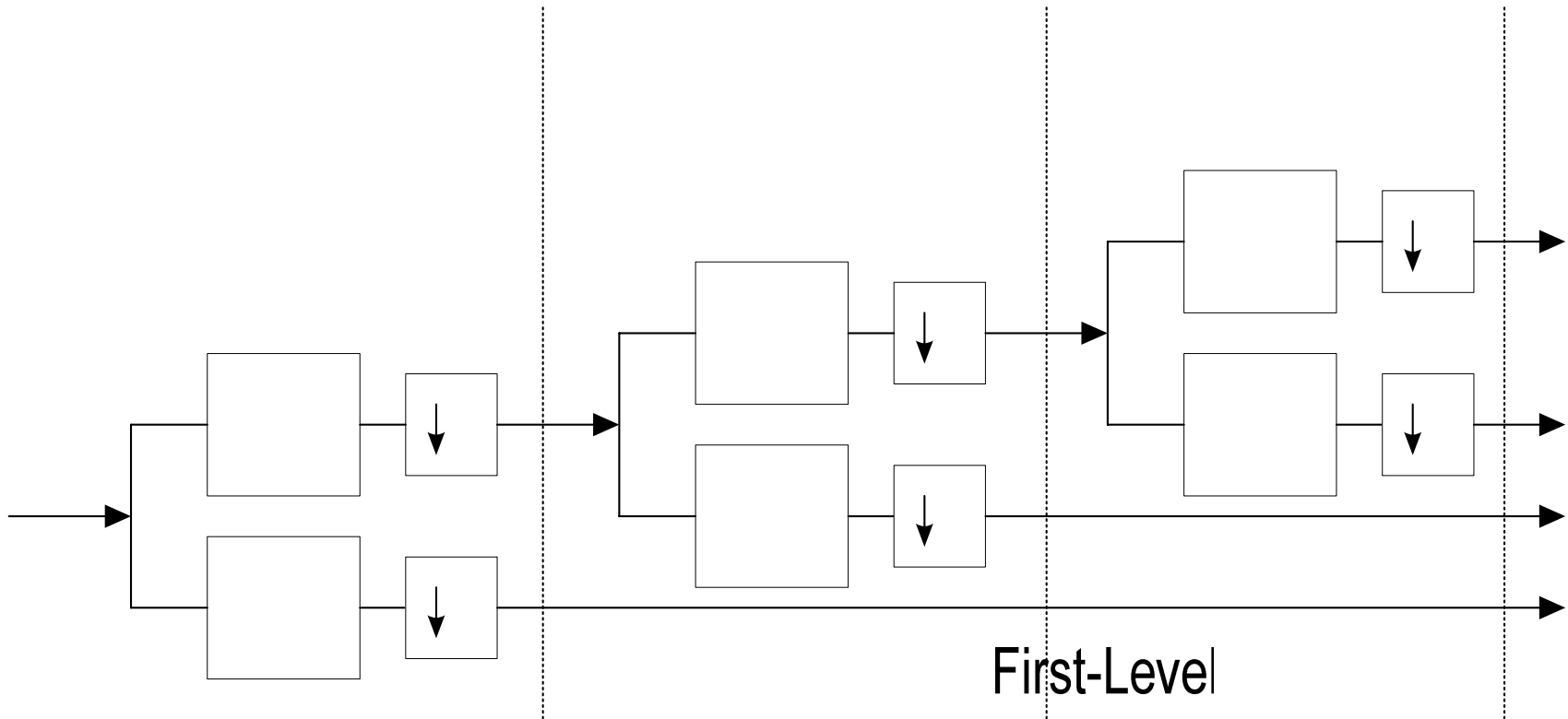
- Requires intensive computation power.

- Therefore, VLSI/FPGA implementations of the DWT are needed for real-time applications.



Discrete Wavelet Transform

- Wavelets can be implemented as a set of filter banks comprising a high-pass and a low-pass filter, each followed by downsampling by two.



□ In 1992, Cohen, Daubechies and Feauveau established the theory of biorthogonal wavelet systems.

□ Biorthogonal wavelet filters possess a linear-phase property and they have a symmetric (or anti-symmetric) impulse response.

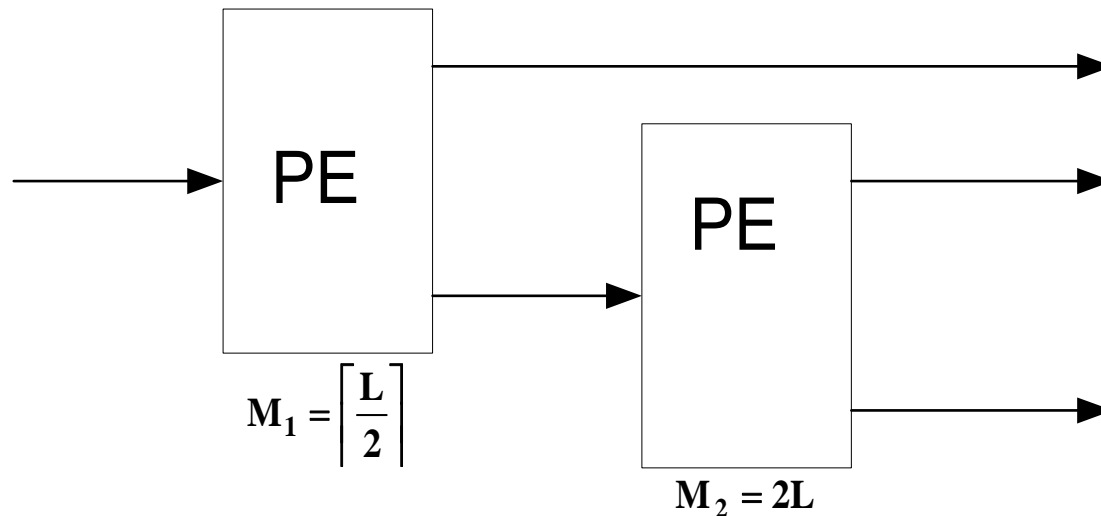
$$a(n) = \pm a(L-1-n)$$

□ Biorthogonal wavelets have been found to offer improved coding gain and an efficient treatment of boundaries in image coding applications.

(1-D) DBWT Architecture

The proposed architecture consists of 2 PEs.

- PE1 computes the first level of decomposition.
- PE2 computes the higher levels (k) of the decomposition, where $2 \leq k \leq K$.



Property-I: EVEN-numbered and ODD-numbered input samples can be processed separately in order to obtain decimated output.

$$DWT_{x(n)} = \begin{cases} d_{j,k} = \sum x(n)h_j^*(n-2^j k) \\ a_{j,k} = \sum x(n)g_j^*(n-2^j k) \end{cases} \quad \Rightarrow \quad \begin{aligned} a^{k+1}(n) &= \sum_{i=0}^{[L/2]-1} h^E(i) \cdot x^{k-E}(n-i) + \sum_{i=0}^{L-[L/2]-1} h^O(i) \cdot x^{k-O}(n-i) \\ d^{k+1}(n) &= \sum_{i=0}^{[L/2]-1} g^E(i) \cdot x^{k-E}(n-i) + \sum_{i=0}^{L-[L/2]-1} g^O(i) \cdot x^{k-O}(n-i) \end{aligned}$$

Property-II: Biorhogonal filters has symmetric filter coefficients:

$$a(n) = \pm a(L-1-n)$$

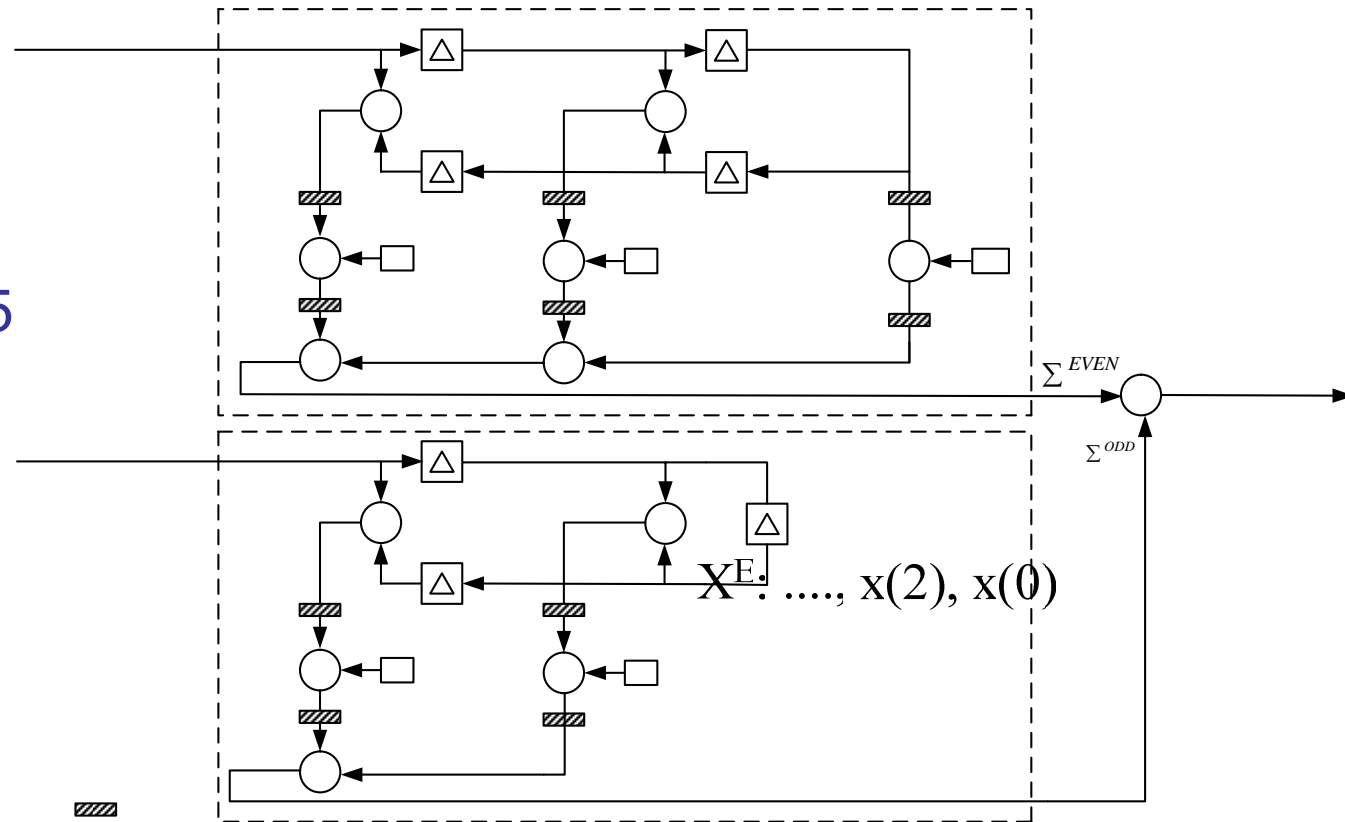
(1-D) DBWT Architecture (cont')

Processing Element- I

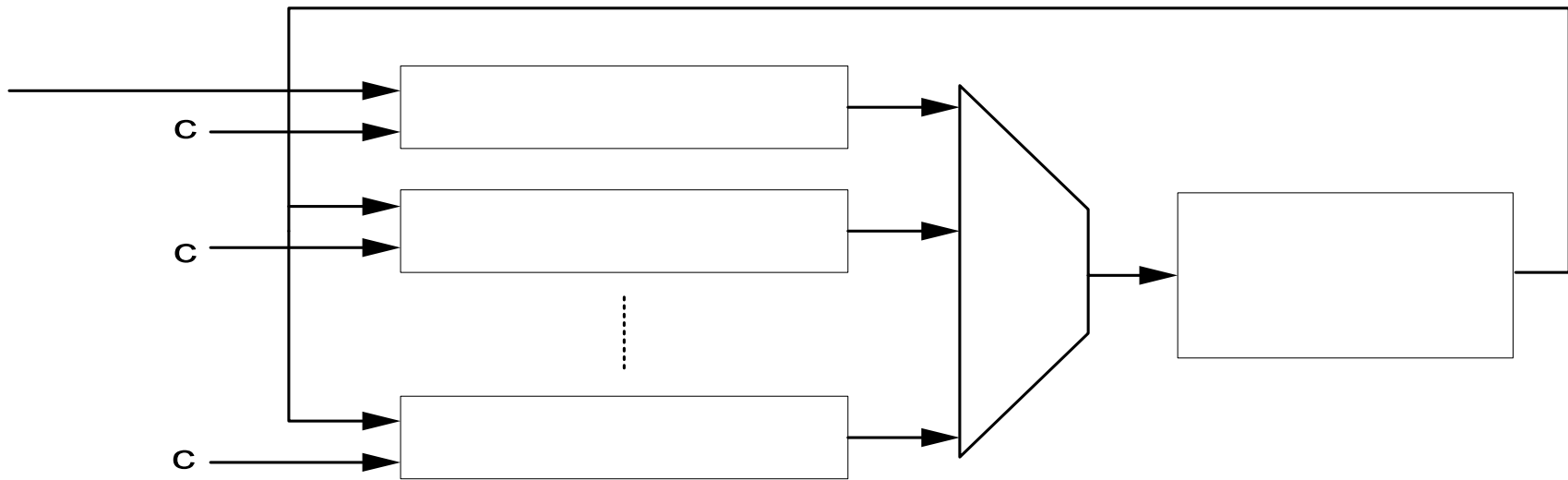
- Properties I and II have been used in the design of PE-I.
- Even and odd-numbered input samples are processed in parallel.

9-tap
Biorthogonal
Wavelet Filter has 5
MACs.

- C.T. = $N_0/2$ ccs
- Throughput: $2f_s$
- 100% Efficiency



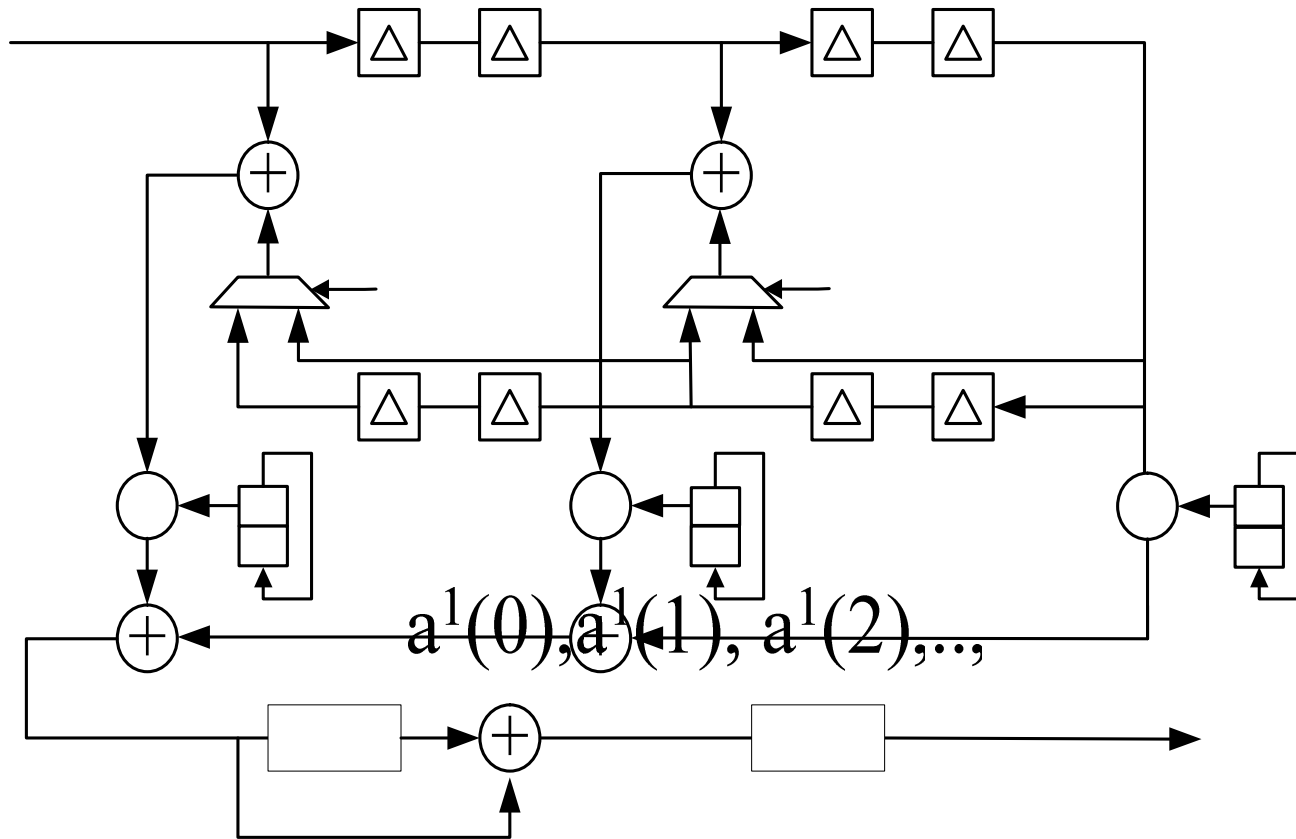
- ❑ PE₂ is responsible for decomposition levels of $k \geq 2$.
- ❑ It is based on Recursive Pyramid Algorithm.
- ❑ C.T. is $N_0/2$ for an input sequence of $N_0/2$ -samples.
- ❑ Latency is $\log(L)$.
- ❑ It consists of a parallel filter and a register bank



(1-D) DBWT Architecture

Processing Element- II (cont')

- PE₂ has $\text{floor}(L/4) = 3$ MACs to constitute a balanced team with PE₁.
- Computation time: $N_0/2$



- ❑ Biorthogonal (9,7) and (9,3) wavelet filters have been ported to a Virtex-2000E FPGA chip.
 - ❑ Arithmetic part was developed using Schematic
 - ❑ Control circuit was developed using Handel-C.

- ❑ Filter coefficient and input data wordlength of 9 bits have been used for a fair comparison.

- ❑ Output data wordlength of 16 bits is enough to represent wavelet coefficients without causing any overflow.

Wavelet Design	FPGA Device	Levels of Decomposition (K)	Input data Wordlength (bits)	Coefficient Wordlength (bits)	Area (CLBs)	Throughput (MSamples/Sec)
Proposed <u>Bior(9,7)</u>	V2000E	1	9	9	462	334
Proposed <u>Bior(9,7)</u>	V2000E	≥ 2	9	9	1402	320
Proposed <u>Bior(5,3)</u>	V2000E	1	9	9	340	341
Proposed <u>Bior(5,3)</u>	V2000E	≥ 2	9	9	1044	324

□ A single level of 9,7 filter bank implementation requires 462 slices (less than %3 of slices available on the FPGA device) for a input data throughput of 320 MegaSamples/sec.

□ Fmax slightly decreases as K increases, because of the growth in multiplier wordlength.

Design	Computation Time (ccs)	Number of MACs	Fmax (MHz)	Implementation
Proposed	$N_0/2$	$\lceil L/2 \rceil + L$	141	FPGA
<u>Uzun's</u> pipelined arch. [4]	$N_0/2$	$\sum_{k=1}^K L / 2^k$	131	FPGA
<u>Masud's</u> Pipelined arch. [1]	N_0	$K \cdot \lceil L/4 \rceil$	85	FPGA
<u>Masud's</u> PA arch. [1]	$O(KN_0)$	$\lceil L/4 \rceil$	-	-
<u>Jou</u> [2] ⁽¹⁾	$N_0/2$	$\lceil L/2 \rceil$	50	VLSI
<u>Mokthar</u> [3] ⁽²⁾	$O(KN_0l_i)$	$K \cdot \lceil L/2 \rceil$	70.2	FPGA

⁽¹⁾ This architecture can perform just 1st-level decomposition.

⁽²⁾ Bit-level architecture for DBWT.

❑ Pipelined Architectures:

❑ - Computation time: N_0 / Area: $O(KL)$

❑ PA (Folded) Architectures:

❑ - Computation time: $O(KN_0)$ / Area: $O(L)$

N_0 : Input data length

L: Filter Length

K: Decomposition Level

- ❑ The clock rate has been found around 131MHz, where $K \geq 3$.
- ❑ This is sufficient for carrying out a three level wavelet transform of a 1024 x 1024 pixel image at 50 frames/sec.
- ❑ Possible applications include:
 - ❑ Video processing in HDTV such as compression, motion estimation and compensation.
 - ❑ 3D Medical/Satellite Image Compression.

Part 4

Power Modeling

- ❑ Explore strategies for power and energy minimisation for FPGA based designs (Parallelism, Algorithm Optimisation, Arithmetic Techniques)
- ❑ Model the power and energy consumption using Functional Level Power Analysis and Modelling - a high level macromodelling methodology for FPGAs
- ❑ Propose a power aware design flow based on design space exploration, optimised IP core design and FLPAM

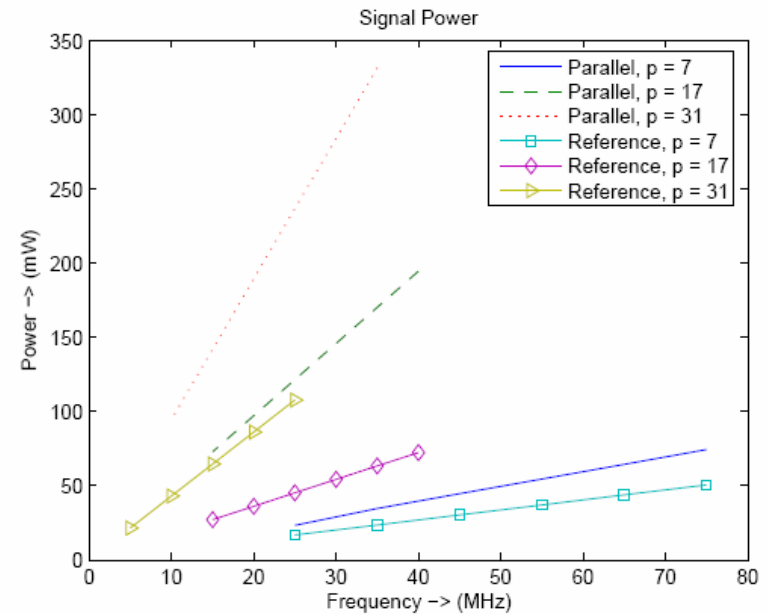
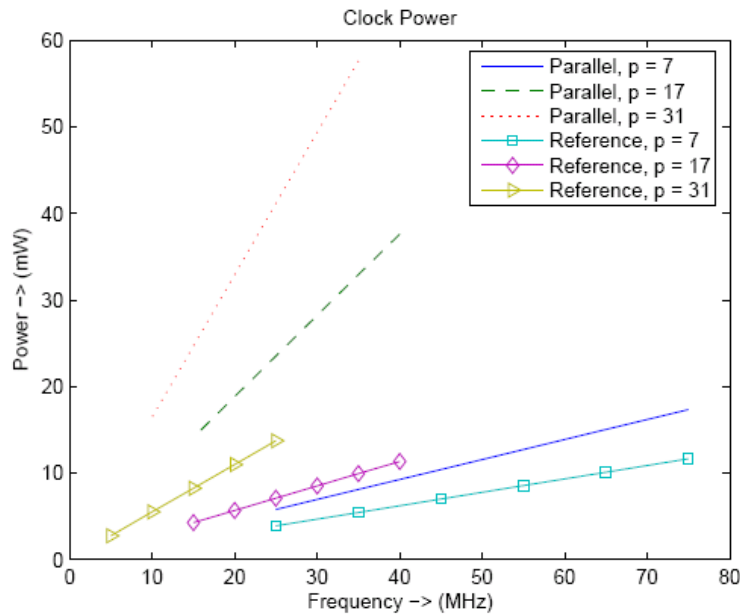
FLPAM Methodology :

- ❑ The order and number of terms in each equation is deduced from the logic resources being used on the FPGA, and the various parameters involved in the design
- ❑ The order of the equation, and the constant coefficients associated with each variable are unknown parameters
- ❑ The choice of a suitable model is a scientific decision, not a statistical one; but the model itself can be refined in later stages using statistical inputs

FLPAM Methodology :

- ❑ Non linear regression analysis is performed until convergence to determine the values of the constant coefficients in the model
- ❑ If the sum of square deviation is suitably low, the model is assumed to be correct. Else, other variables and parameters are introduced in the model in order to obtain a better fit
- ❑ Redundant coefficients (that equate to 1) can also be eliminated from the model

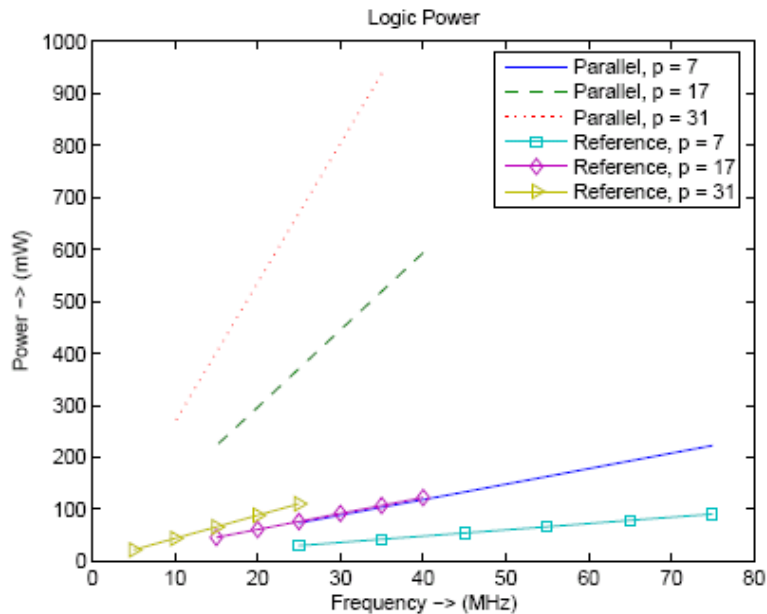
Using the FLPAM : FRAT - an example



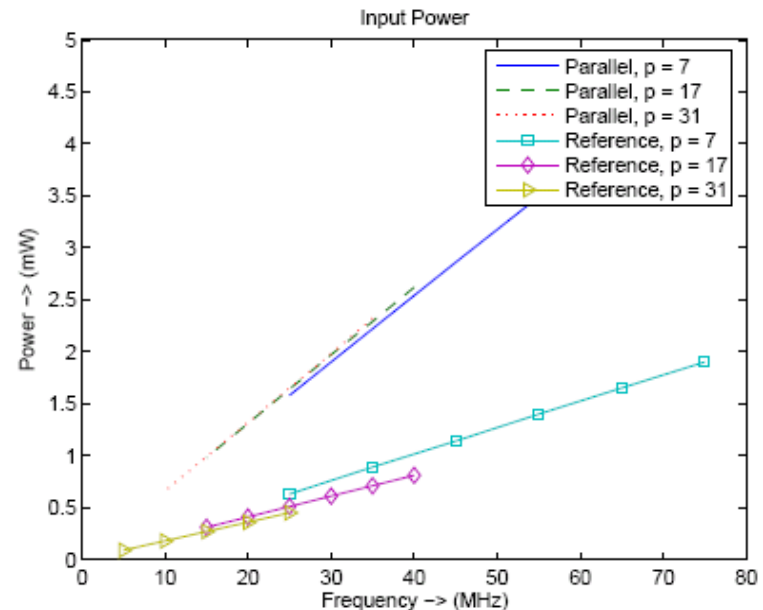
$$CP = c1 \cdot v^2 \cdot TA_{R/P} \cdot f + c2 \cdot f + c3$$

$$SP = s1 \cdot v^2 \cdot TA_{R/P}^2 \cdot f + s3 \cdot f + s4$$

Using the FLPAM : FRAT - an example

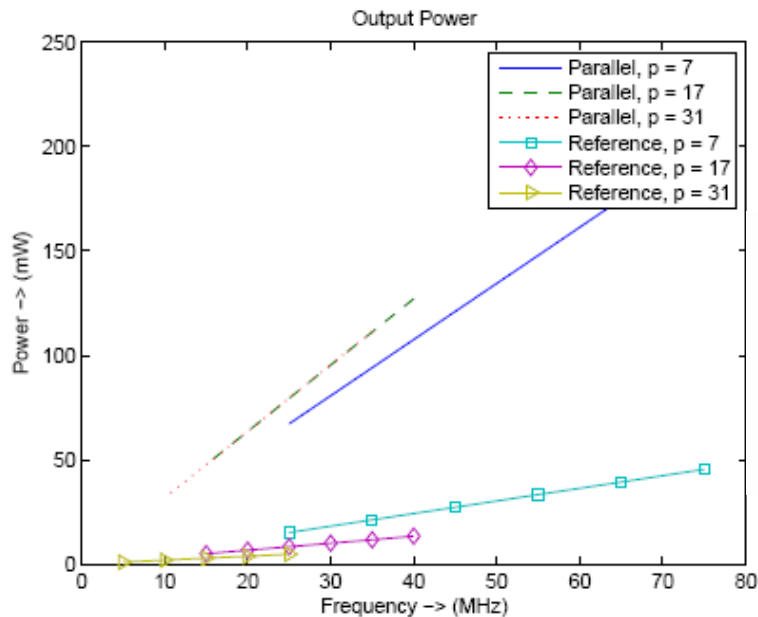


$$LP_P = l1 \cdot v^2 \cdot TA_P \cdot f + l2 \cdot f + l3$$



$$IP = i1 \cdot v^2 \cdot f \cdot (p^2)^{i2} + i3$$

Using the FLPAM : FRAT - an example

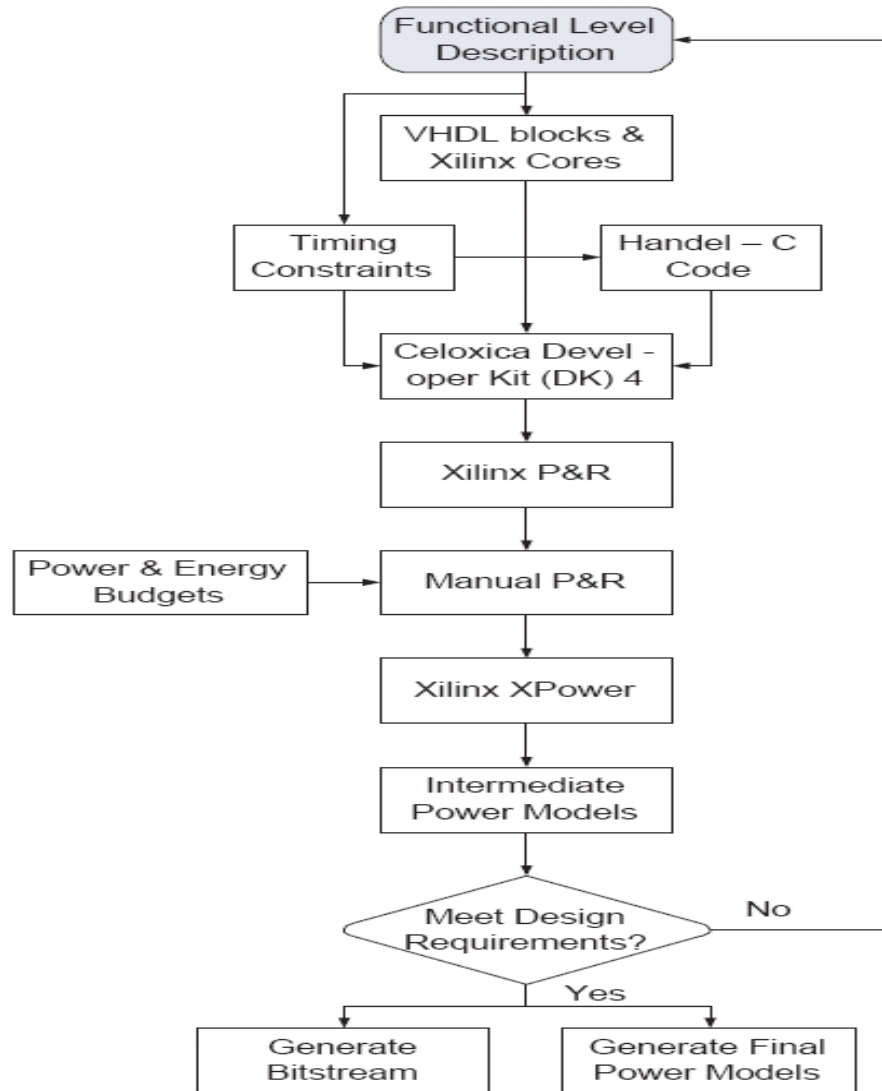


Area Models :

$$TA_R = (\alpha_1 \cdot p(p + 1) + \alpha_2) \cdot (\text{ceil}(\log_2(255 * p))) + \alpha_3$$

$$TA_P = \alpha_1 \cdot p^2 + \alpha_2 \cdot p \cdot \text{ceil}(\log_2(255 * p)) + \alpha_3$$

$$OP = o_1 \cdot (v^2 \cdot N) \cdot f^2 \cdot \text{ceil}(\log_2(255 \cdot p)) \cdot p \cdot (p + 1)^{o_2} + o_3$$



Using the FLPAM : Sample model coefficients obtained after regression analysis

Coeff.	Virtex-E	Virtex-II	Virtex-IV
$\alpha 1$	1.97E-01	2.22E-02	2.02E-02
$\alpha 2$	6.86E-01	2.87E+01	4.16E+01
$\alpha 3$	3.25E+01	-2.32E+02	-3.74E+02
$c 1$	2.95E-04	1.85E-04	5.79E-05
$c 2$	6.34E-02	6.45E-02	8.28E-03
$c 3$	-3.97E-02	1.93E+00	8.17E-03
$s 1$	4.45E-04	5.44E-05	3.90E-05
$s 2$	1.28E+00	1.59E+00	1.58E+00
$s 3$	1.80E-01	2.53E-01	6.73E-02
$s 4$	-5.27E-03	1.13E+01	3.39E-01

Reference Architecture

Coeff.	Virtex-E	Virtex-II	Virtex-IV
i1	1.61E+01	1.23E-03	1.13E-04
i2	8.13E-03	1.11E+00	1.11E+00
i3	-5.46E+01	1.85E-01	3.61E-02
i4	6.83E+01	8.26E-01	9.11E-02
o1	-8.80E-07	-9.40E-08	0.00E+00
o2	1.66E+00	2.37E+00	0.00E+00
o3	9.73E-01	1.96E+00	0.00E+00
o1	-4.58E-06	-1.66E-09	2.35E-02
o2	1.06E+00	3.49E+00	-1.08E+00
o3	2.45E+01	1.42E+02	-6.41E-02

Parallel Architecture

- ❑ Key characteristics of FLPAM :
- ❑ High level - system variable based model
- ❑ Platform independent - same model fits Virtex-E and Virtex 4
- ❑ Provides quick and reasonably accurate pre-silicon estimates of power dissipation
- ❑ Enables design space optimisation by core developer, and design space exploration by the core user
- ❑ Number of cores: FRAT, FRIT, FHT/FWT, GMM, Inner Product and Color space conversion have been successfully modeled

Part 5

Conclusions

- ❑ The work presented is a general-purpose *Custom Coprocessor for matrix Algorithms using Reconfigurable Hardware*
- ❑ Due to the importance of matrix algorithms in *image and signal processing*, novel architectures based on CA and DA principles have been presented in this talk
- ❑ A high level macromodelling methodology: FLPAM has been introduced
- ❑ Future Work: Take advantage of the most recent FPGAs, Low Power, RC for Mobile Computing, Information Retrieval and Face Recognition