A Novel Reconfigurable Low Power Distributed Arithmetic Architecture for Multimedia Applications

Zhenyu Liu, Tughrul Arslan and Ahmet T. Erdogan

University of Edinburgh

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Outline

- Introduction *Go*
- ROM-based and adder-based DA *Go*
- Architecture of Reconfigurable DA *Go*
- DCT and its implementation *Go*
- Evaluation *Go*
- Summary *Go*

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Introduction

- Motivation
  - Distributed Arithmetic (DA) has been wildly adopted in many applications such as DCT (Discrete Cosine Transform), DFT (Discrete Fourier Transform), FIR (Finite Impulse Response), and DHT (Discrete Hartley Transform).
  - From literature, there is not existing domain specific architecture for reconfigurable DA.
- The proposed architecture could
  - Implement DA with the lower power consumption, area occupation and less delay time.

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Outline

- Introduction *Go*
- ROM-based and adder-based DA *Go*
- Architecture of Reconfigurable DA *Go*
- DCT and its implementation *Go*
- Evaluation *Go*
- Summary *Go*

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

ROM-based and adder-based DA

DA computes the inner product of two vectors.

$$Z = A \cdot X = \sum_{i=0}^{L-1} C_i X_i \qquad (1)$$

A = $[C_0, C_1, \cdots, C_{L-1}]$ is an M bits fixed coefficient vector
X = $[X_0, X_1, \cdots, X_{L-1}]$ is an N bits input vector.

C_i and X_i can be express in two's complement binary as follows:

$$C_i = -C_{i,(M-1)} \cdot 2^{M-1} + \sum_{j=0}^{M-2} C_{i,j} \cdot 2^j \qquad (2)$$

$$X_i = -X_{i,(N-1)} \cdot 2^{N-1} + \sum_{k=0}^{N-2} X_{i,k} \cdot 2^k \qquad (3)$$

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

ROM-based DA

By substituting (3) in (1), the output Z is given by:

$$Z = A \cdot X = \sum_{i=0}^{L-1} C_i \left(-X_{i,(N-1)} \cdot 2^{N-1} + \sum_{k=0}^{N-2} X_{i,k} \cdot 2^k \right)$$

$$= -\sum_{i=0}^{L-1} C_i X_{i,(N-1)} \cdot 2^{N-1} + \sum_{k=0}^{N-2} \left[\sum_{i=0}^{L-1} C_i X_{i,k} \right] \cdot 2^k \qquad (4)$$

By defining the term R_{N-1} as

$$R_{N-1} = \sum_{i=0}^{L-1} C_i X_{i,N-1} \qquad (5)$$

Substituting (5) in (4), (4) can be written as

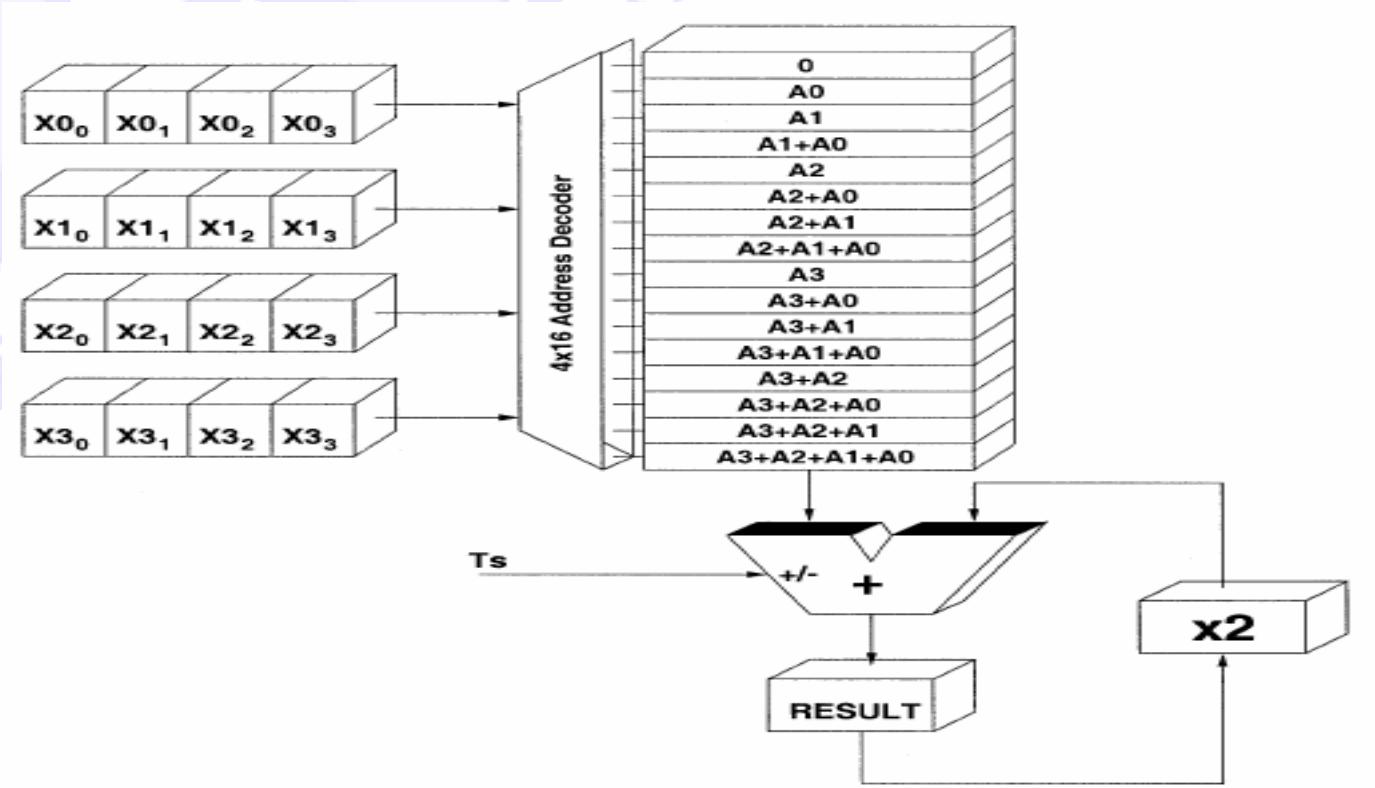
$$Z = -R_{N-1} \cdot 2^{N-1} + \sum_{k=0}^{N-2} R_k \cdot 2^k$$

$$= \sum_{k=0}^{N-1} S_k \cdot R_k \cdot 2^k$$

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

ROM-based DA

- R_k has 2^L possible values for $k = 0,1,\cdots,L-2$, these values can be stored in a ROM of size 2^L



Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

ROM-based and adder-based DA

DA is computes the inner product of two vectors.

$$Z = A \cdot X = \sum_{i=0}^{L-1} C_i X_i \qquad (1)$$

A = $[C_0, C_1, \cdots, C_{L-1}]$ is an M bits fixed coefficient vector

X = $[X_0, X_1, \cdots, X_{L-1}]$ is an N bits input vector.

C_i and X_i can be express in two's complement binary as follows:

$$C_i = -C_{i,(M-1)} \cdot 2^{M-1} + \sum_{j=0}^{M-2} C_{i,j} \cdot 2^j \qquad (2)$$

$$X_i = -X_{i,(N-1)} \cdot 2^{N-1} + \sum_{k=0}^{N-2} X_{i,k} \cdot 2^k \qquad (3)$$

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

adder-based DA

By substituting (2) in (1), the output Z is given by:

$$Z = A \cdot X = \sum_{i=0}^{L-1} X_i \left(-C_{i,(M-1)} \cdot 2^{M-1} + \sum_{j=0}^{M-2} C_{i,j} \cdot 2^{j} \right)$$

$$= -\sum_{i=0}^{L-1} X_i C_{i,(M-1)} \cdot 2^{M-1} + \sum_{j=0}^{M-2} \left[\sum_{i=0}^{L-1} X_i C_{i,j} \right] \cdot 2^{j} \qquad (6)$$

By defining the term T_j as

$$T_j = \sum_{i=0}^{L-1} X_i C_{i,j} \qquad (7)$$

Substituting (7) in (6), (6) can be written as

$$Z = -T_{M-1} \cdot 2^{M-1} + \sum_{j=0}^{M-2} T_j \cdot 2^{j}$$

$$= \sum_{j=0}^{M-1} S_j \cdot T_j \cdot 2^{j}$$

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

adder-based DA

Common Terms Sharing

Suppose the input vector and fixed coefficient vector are

- $X = [X_0, X_1, X_2, X_3]$
- C_{00}= 1101b, C_{10}= 1011b, C_{20}= 1110b, C_{30}= 0011b;

$$T_j = \sum_{i=0}^{L-1} X_i C_{i,j} \; ; \qquad T_0 = \sum_{i=0}^{L-1} X_i C_{i,0} =$$

$$\begin{array}{r} X_0 \cdot (1\ 1\ 0\ 1) \\ X_1 \cdot (1\ 0\ 1\ 1) \\ X_2 \cdot (1\ 1\ 1\ 0) \\ +) \; X_3 \cdot (0\ 0\ 1\ 1) \\ \hline \end{array}$$

$$= \begin{array}{cccc} X_0 & X_0 & 0 & X_0 \\ X_1 & 0 & X_1 & X_1 \\ X_2 & X_2 & X_2 & 0 \\ +) \; 0 & 0 & X_3 & X_3 \\ \hline \end{array}$$

$$= (X_0 + X_1 + X_2)2^3 + (X_0 + X_2)2^2$$
$$+ (X_1 + X_2 + X_3)2^1 + (X_0 + X_1 + X_3)2^0$$

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

adder-based DA

Common Terms Sharing

Directly implement:

- 2^3: *X0+X1+X2*
- 2^2: *X0+X2*
- 2^1: *X1+X2+X3*
- 2^0: *X0+X1+X3*

Schemes:

- *Scheme* I: *X0+X1*
- *Scheme* II: *X1+X2*
- *Scheme* III: *X0+X2 and X1+X3*

Conclusion:

- *Different common terms sharing scheme affect the overall hardware efficiency and power consumption significantly.*
- *The implementation of Adder-based DA will not limited by the applications. The common terms sharing does not depend on the specific application.*

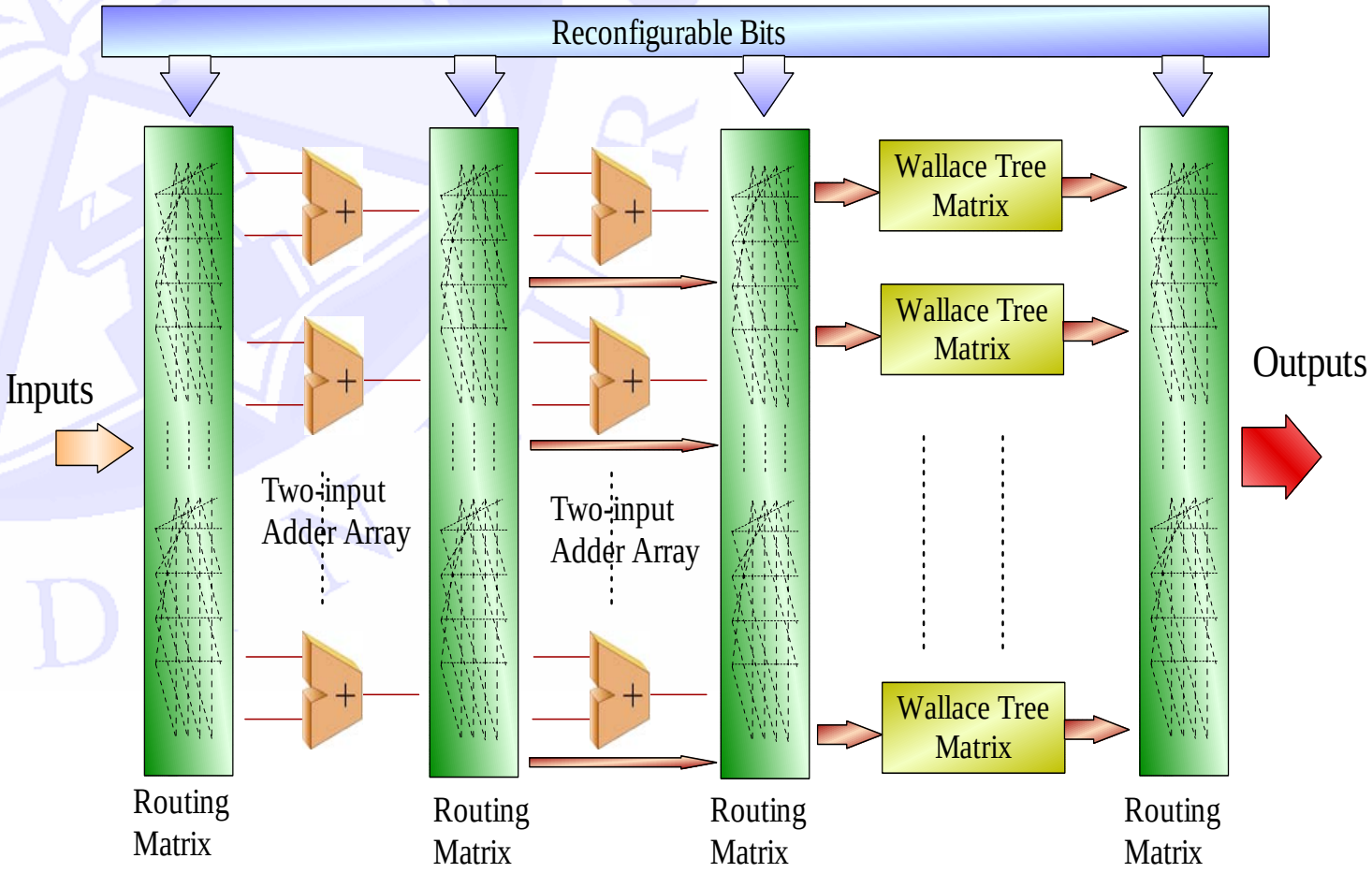
Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Outline

- Introduction *Go*
- ROM-based and adder-based DA *Go*
- Architecture of Reconfigurable DA *Go*
- DCT and its implementation *Go*
- Evaluation *Go*
- Summary *Go*

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Architecture of Reconfigurable DA



Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Outline

- Introduction *Go*
- ROM-based and adder-based DA *Go*
- Architecture of Reconfigurable DA *Go*
- DCT and its implementation *Go*
- Evaluation *Go*
- Summary *Go*

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

DCT and its implementation

For eight points 1D DCT, coefficient matrix F_k is given as following:

$$F_k = [F_k(i)] = C_k \cdot \cos\frac{\pi k(2i+1)}{16}$$

Where

$$C_k = \begin{cases} 1/\sqrt{N} & k = 0 \\ 1/\sqrt{2/N} & 1 \le k \le N-1 \end{cases}$$

For an input vector X_i, the DCT output vector $\{Y_0, Y_1 \ldots Y_7\}$ is given as:

$$Y_k = \sum_{i=0}^{7} [F_k(i)] \cdot X_i = \sum_{i=0}^{7} F_k \cdot X_i$$

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

DCT and its implementation

$$F_0(i) = \begin{vmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{vmatrix} \qquad F_1(i) = \begin{vmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{vmatrix}$$

$F_0(i)$ and $F_1(i)$ in 2's complement format

- For 8-point 1D DCT, 96 (=12*8) terms are needed for eight inputs with 12-bit coefficients. It means that 672 two-input adders are required when implemented directly.

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

DCT and its implementation

- For our adder-based architecture
  - Deducting the zero and duplicate terms which need no further calculation, there are one term of 8 inputs and 22 terms of 4 inputs,

Common Terms Sharing scheme for DCT

inputs	Terms
8	T(01234567)
4	T(0123),T(4567),T(0124),T(0145),T(0356),T(0135), T(0246),T(1247),T(2435),T(1357),T(0257),T(0167), T(1346),T(1237),T(3567),T(1457),T(0236),T(1256), T(0347),T(2467),T(2367),T(0456),

Table 1: Unique terms of DCT

- The scheme
  - Take T(0123) as the example, three possible schemes, there are [T(01),T(23)], [T(02),T(13)], [T(03),T(12)]
  - Our scheme: T(01), T(23), T(45), T(67), T(06), T(35), T(24), T(17), T(07), T(25), T(16), T(34)

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

DCT and its implementation

- Standards fully supported
  - ISO/IEC 14496-2:2004
  - IEEE Std 1180–1990
    - ☆ Image pixel representation: 8 bits for 8X8 DCT
    - ☆ Input bits for the forward transform: 9 bits
    - ☆ Coefficients representation: 12 bits
    - ☆ 1D DCT outputs: 14 bits
- Results
  - 600,929 um^2 with UMC 0.18um CMOS technology library
  - 15.2mW at 20MHz
  - Run up to 144MHz (6.93ns) with 112-bits (=14bits×8) outputs

The architecture can reach up to 16.128Gbps for the 1D DCT.

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Outline

- Introduction *Go*
- ROM-based and adder-based DA *Go*
- Architecture of Reconfigurable DA *Go*
- DCT and its implementation *Go*
- Evaluation *Go*
- Summary *Go*

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Evaluation

- Compare with the common subexpression elimination with CSD code

  ☆ The paper of Macleod, M.D. and Dempster, A.G. ," Common subexpression elimination algorithm for low-cost multiplierless implementation of matrix multipliers", Electronics Letters, May 2004

  ☆ The paper of Tian-Sheuan Chang, Jiun-In Guo, etc. " Hardware-efficient DFT designs with cyclic convolution and subexpression sharing", IEEE Transactions on Circuits and Systems, Sept. 2000

  – All the implementations are targeted on 8X8 DCT with bitwidth at 8.
  – The number of required adders is 65 and 130 respectively in the above papers.
  – For our architecture, a total of 35 adders are needed.

Our method achieves 46% and 73% reduction respectively.

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Experiment platform

- Tool software in experiment;
  - Ambit BuildGates V4.0-s002
    Cadence Design Systems, Inc.
  - ISE V7.1i Xilinx, Inc.
  - Cadence Silicon Ensemble and Synopsys PrimePower
- Target cell and PFGA device;
  - UMC 0.18um three-metal CMOS technology library
  - Virtex-E xcv1600e

Evaluation

- Power consumption comparison with ASIC design

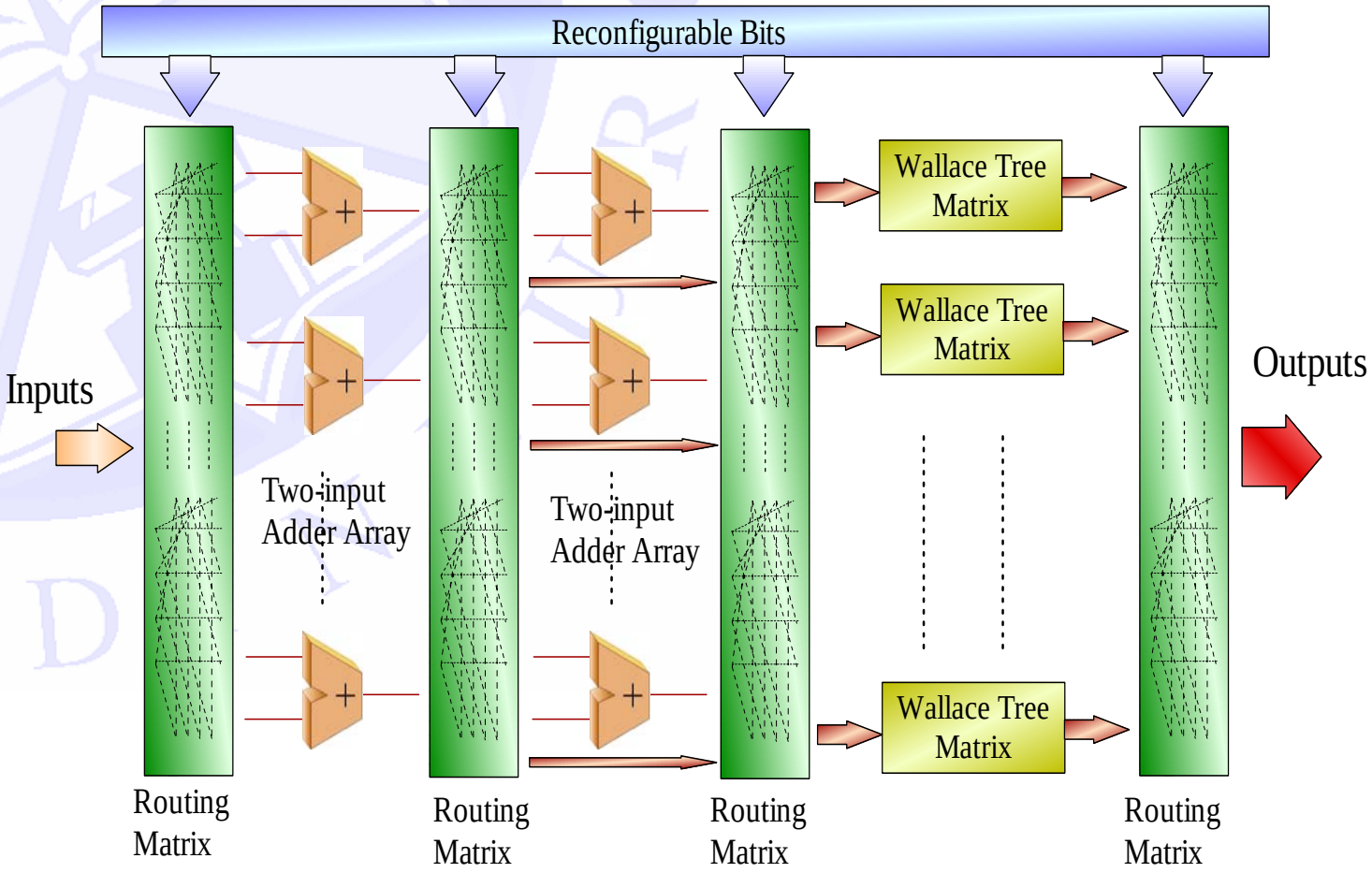
  ☆ The paper of Ghosh, S., Venigalla, S.and Bayoumi, M. “Design and implementaion of a 2D-DCT architecture using coefficient distributed arithmetic”, VLSI 2005

  – 12.45mW for 1D DCT with ST Microelectronics, hcmos9, 0.12um technology at 1.5V, with 50MHz. Considering dissipated power is approximatively proportional to the square of supply voltage, it can be scaled to 7.97mW for 1.2V.

  – Our architecture consumes 7.13mW with UMC 0.13um CMOS technology library at 1.2V, 50MHz.

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Architecture of Reconfigurable DA



Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Evaluation

- Area & Delay Comparison with ASIC designs
  - Normalized delay-area product was adopted to evaluate performance.
  - Six existing designs are selected for comparison, which are 12-bit word length of data path.
  - Reconfiguable rouing network in our design is removed for fair comparison.

	Design 1	Design 2	Design 3	Design 4	Design 5	Design 6	Average	Proposed (Scaled)
Normalized delay-area product	853112.48	1033565.85	1073023.72	1330855.61	2005223	2223371.27	1419858.66	1241578.80

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Outline

- Introduction *Go*
- ROM-based and adder-based DA *Go*
- Architecture of Reconfigurable DA *Go*
- DCT and its implementation *Go*
- Evaluation *Go*
- Summary *Go*

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

Summary

- A novel reconfigurable architecture for DA and its DCT implementation was introduced in this paper.

- The proposed reconfigurable architecture could provide an efficient hardware platform for implementing DA. The DCT implementation shows it can fully satisfy the requirements of real-time image processing.

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group

A Novel Reconfigurable Low Power Distributed Arithmetic Architecture for Multimedia Applications

Thank you!

Email: zhenyu.liu@ed.ac.uk
System-Level Integration Group