

1-Mar-04 (1)

CEG5010: Design Space Exploration of Variable Radix and Wordlength Implementations of Discrete Cosine Transform

1-Mar-04 (2)

Introduction

- A design environment for variable radix and wordlength fixed-point implementations
- Addresses the problem of converting a floating-point description into a hardware-efficient implementation
- Floating-point expressions as input
- Automatic conversion from floating-point to fixed-point based on simulation and optimization
- Synthesizable VHDL descriptions as output

1-Mar-04 (3)

Floating to Fixed Point Conversion

- Algorithmic descriptions are in floating-point
- Fixed-point implementations are preferred
- Quantization effect – increasing wordlength reduces quantization error
- Tradeoffs between area and accuracy
- Large and complex design space
- Difficult to optimize wordlengths to absolute minimal values manually

1-Mar-04 (4)

Digit-Serial Computation

- Arithmetic operations are carried out in multiple cycles, each of which for one digit of a variable
- Less hardware because hardware for arithmetic operations are folded and reused
- More complicated control circuit
- Tradeoffs among area, latency and throughput
- Bit-parallel and bit-serial architectures are special cases of digit-serial architectures

1-Mar-04 (5)

Motivation

- Incorporate the two design methodologies into design space
- Wordlengths and radices of variables can be different, and the system handles the conversion between any two variables
- Design constraints
 - Output errors
 - Area
 - Latency
 - Throughput

1-Mar-04 (6)

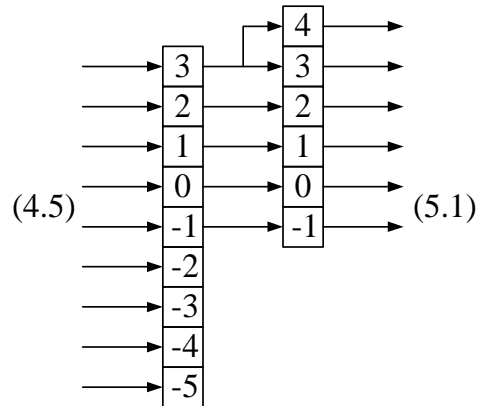
Motivation

- Optimization targets
 - Minimize area
 - Minimize output errors
- Optimization based on simulation
 - Worst-case analysis is too pessimistic
 - Runtime analysis provides a larger space for optimization
- Increase productivity, explore more of the design space in a given time

1-Mar-04 (7)

Two Design Methodologies

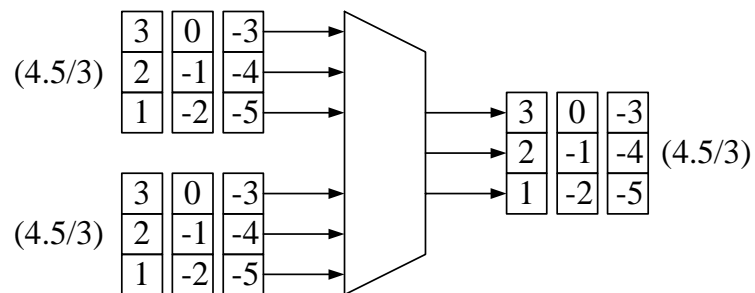
- For typical floating to fixed-point conversions, optimization is achieved by reducing wordlengths of variables



1-Mar-04 (8)

Two Design Methodologies

- In a traditional digit-serial system, all variables have the same digit size



1-Mar-04 (9)

Two Design Methodologies

- Contradiction
 - Floating to fixed-point conversion minimizes area based on variable wordlength
 - Digit-serial computation minimizes area by fixing wordlengths and using folded arithmetic operators
- Problem – traditional digit-serial architecture constrained wordlengths to be identical and wordlength reduction technique cannot be applied

1-Mar-04 (10)

Two Design Methodologies

- Intermediate results usually require higher precisions than input and output values
- The number of digits is proportional to the wordlength of a variable in constant radix digit-serial architectures
- The throughput of a system is limited by the variable that has the most digits
- For arithmetic operators associated with variables with less number of digits, resource is not fully-utilized

1-Mar-04 (11)

Integrating the Methodologies

- Fixing the number of digits, n
- Wordlength and fractional wordlength of variables, w and f , can be different
- Digit size of variables, $d = \lceil w/n \rceil$, can also be different
- Variable radix, variable wordlength architecture
- Format conversion modules inserted in-between operators, which convert variables from one format (w_1, f_1, d_1) to another (w_2, f_2, d_2)

1-Mar-04 (12)

Integrating the Methodologies

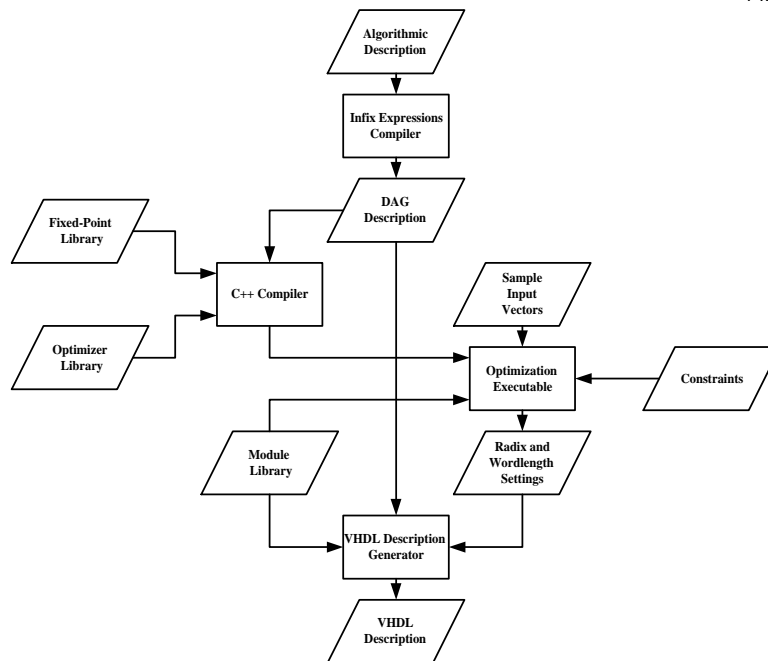
- More efficient utilization of resources, as intermediate computations do not bottleneck the system
- Conversion modules produce resource usage overhead
- Optimization must take the area used by conversion modules into account

1-Mar-04 (13)

Design Flow

- Set of infix expressions as input
- Intermediate Directed Acyclic Graph (DAG) representation
- Sample input vectors for simulation
- User-specified design constraints
- Output synthesizable VHDL descriptions

1-Mar-04 (14)



Optimization

- Infix expressions are translated into a DAG, where nodes represent arithmetic operations
- Each arithmetic operator has a vector of parameters that controls its precision
- Optimization in a multi-dimensional, discontinuous search space
- Downhill simplex method, which does not require computation of derivatives

Discrete Cosine Transform

- Systolic computation of DCT via computation of discrete moments (DM)
- Setting up the relationship between DCT and DM
- The N -point DCT is defined by the equations

$$X(k) = c_k \sum_{n=0}^{N-1} x(n) \cos \frac{\pi(2n+1)k}{2N}, \quad 0 \leq k \leq N-1,$$

$$c_k = \begin{cases} 1/\sqrt{N} & \text{if } k = 0, \\ \sqrt{2/N} & \text{otherwise.} \end{cases}$$

Discrete Cosine Transform

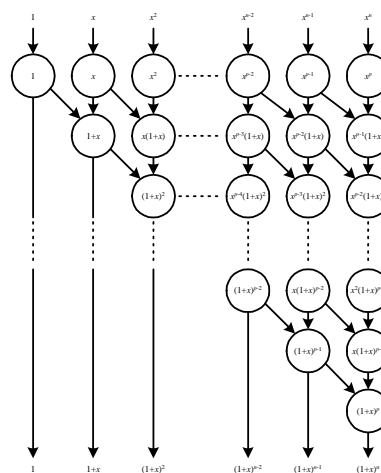
- The equation is transformable into

$$X(k) = c_k \left(x_{k,0} + \sum_{r=0}^p a_r m_{k,2r} \right) + R_p, \quad 0 \leq k \leq N-1$$

- $m_{k,2r}$ is the result of DM, a_r is a vector of constants, c_k is a constant, R_p is the Taylor's remainder term
- Ignoring R_p , DCT is transformed into a dot product of $m_{k,2r}$ and a_r plus $x_{k,0}$ and times c_k

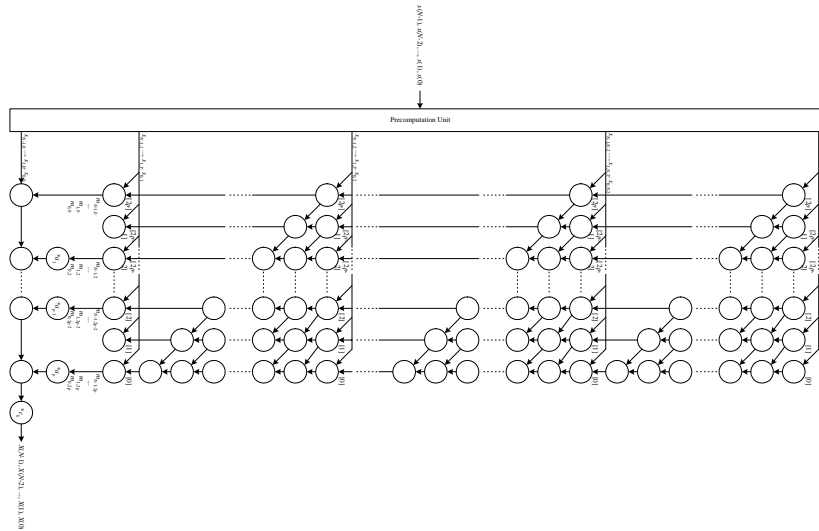
Discrete Cosine Transform

- DM is computed by cascaded p -networks
- A p -network resembles a Pascal triangle



1-Mar-04 (19)

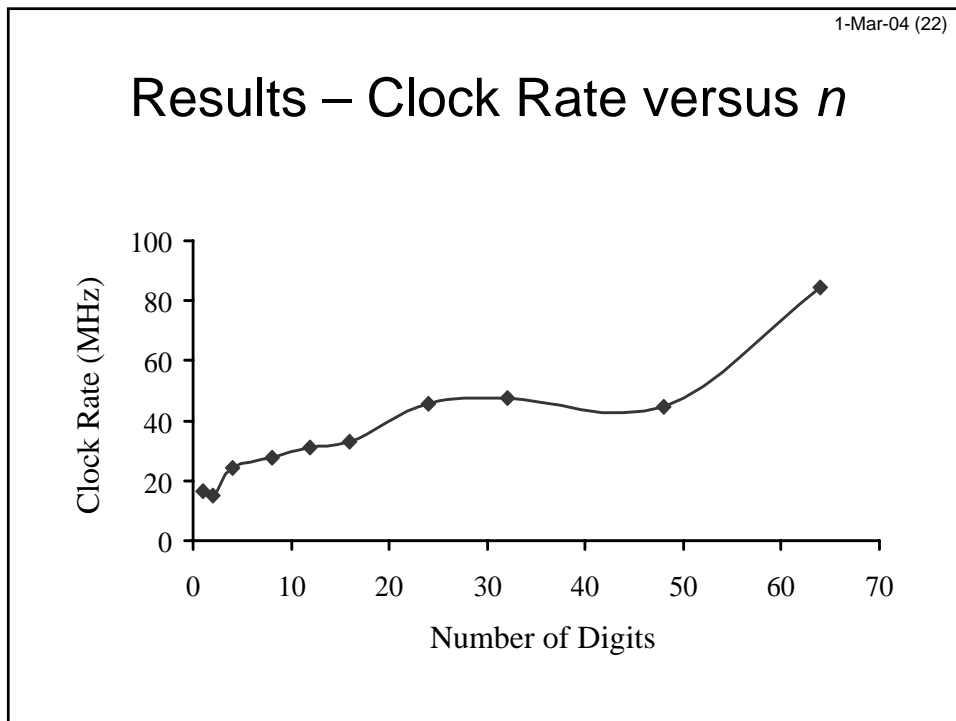
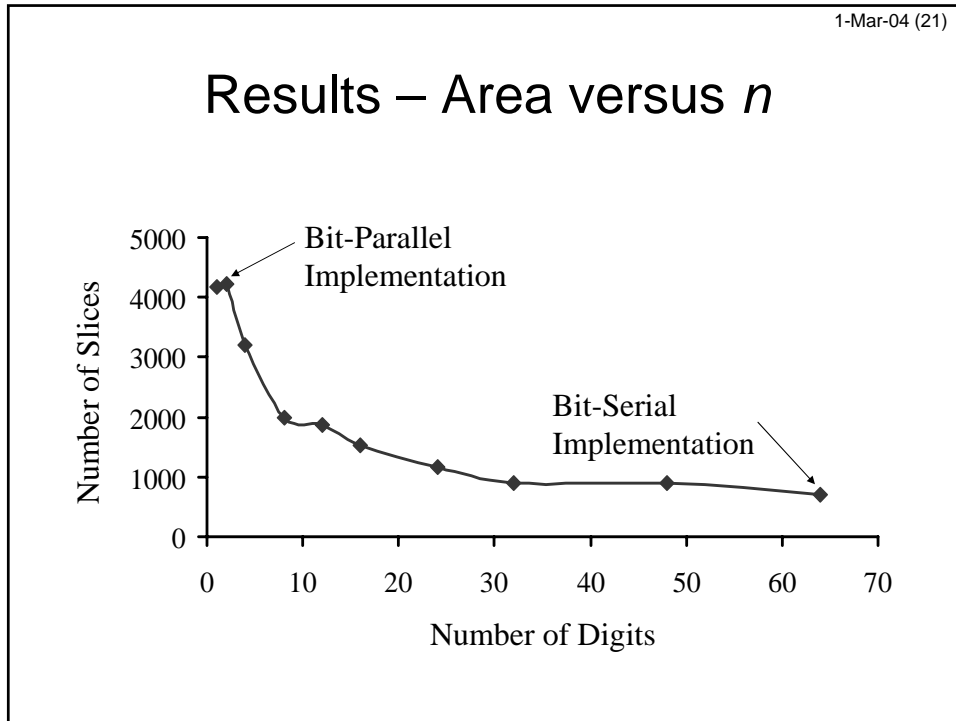
Discrete Cosine Transform

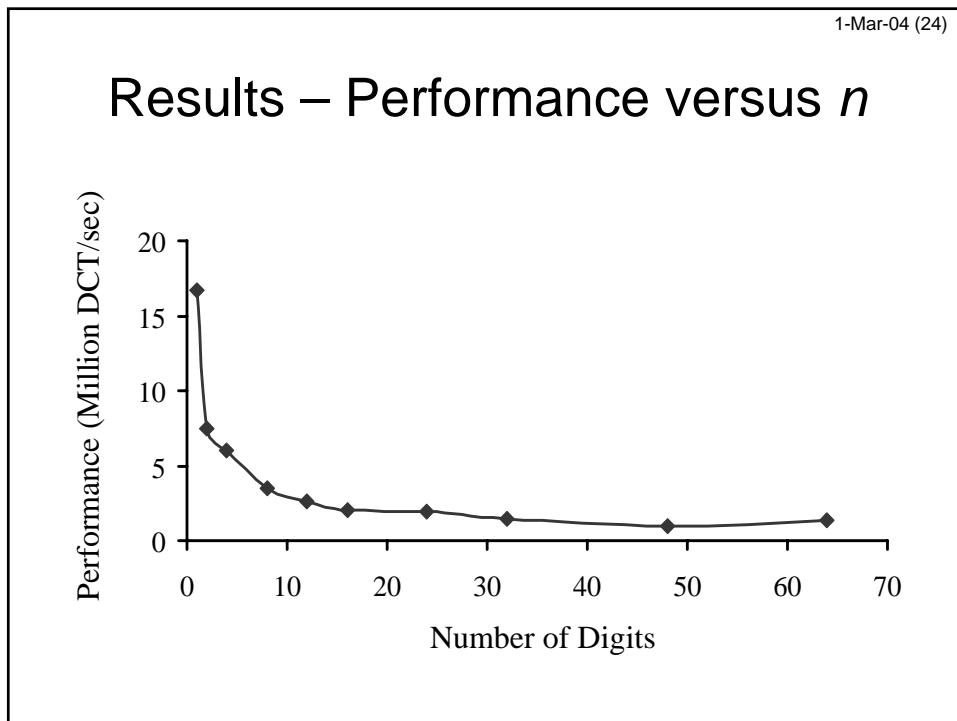
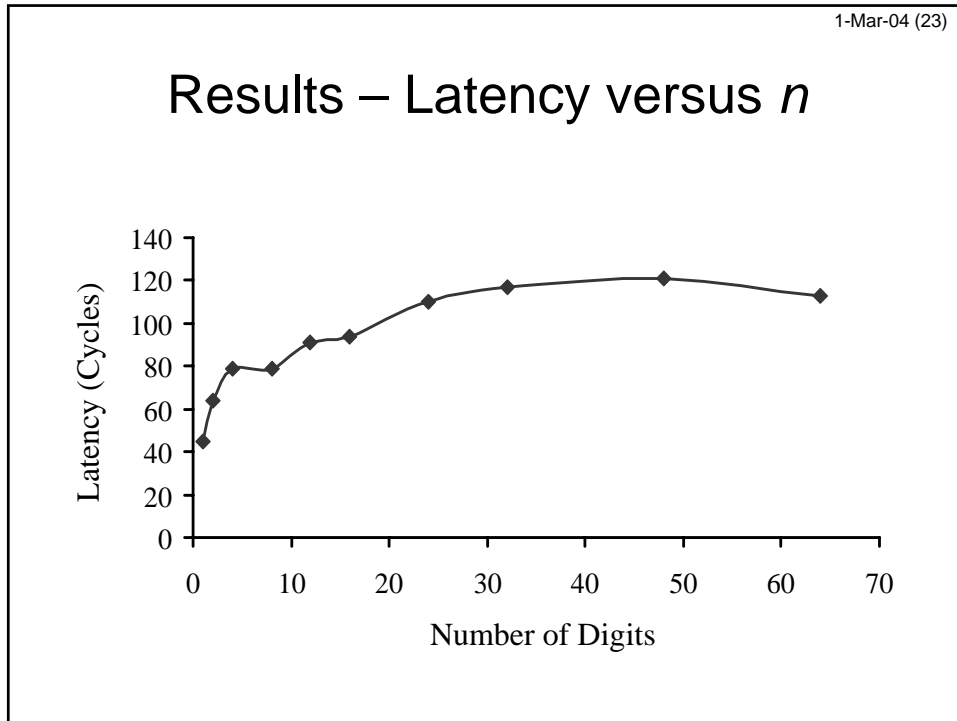


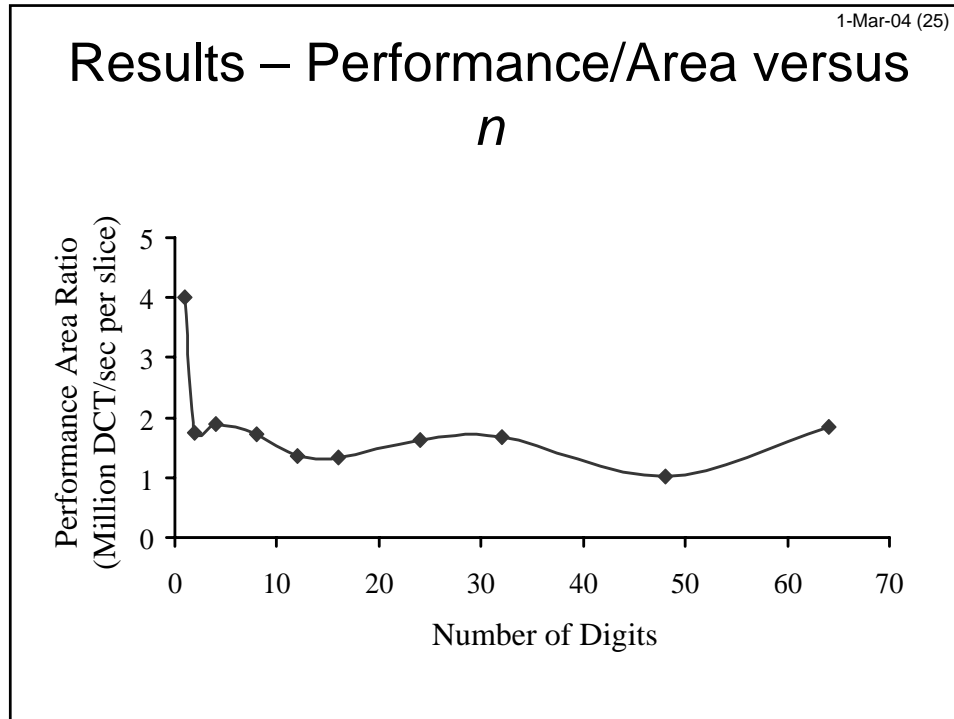
1-Mar-04 (20)

Results – Experiment Setup

- Choose $p=3$, $N=8$
- Systolic structure contains 157 adders and 5 multipliers, excluding those belong to precomputation unit
- Regularity of structure enables compact description – 80 lines of code
- Output error constraint 1.0/256.0
- 200 entries in sample input vector
- Different number of digits, n , were experimented







- 1-Mar-04 (26)
- ## Results – Other Observations
- Throughput is controlled by n
 - Intermediate values can have errors with magnitude up to 1.0×10^2
 - Signal to noise ratio (SNR) is maintained between 2.0×10^2 and 4.0×10^2 throughout the system

1-Mar-04 (27)

Conclusions

- A flexible design environment for variable radix and wordlength implementations
- The idea of variables having the same number of digits but different digit size is a design alternative that incorporates the advantages of the two traditional design methodologies

1-Mar-04 (28)

Conclusions

- DCT as application example
- Create multiple implementations from a single description
- Helps to explore tradeoffs among accuracy, area, latency and throughput
- Designers can concentrate on higher level algorithmic issues
- Reduces turnaround time, improves productivity