

ECE 734 FALL 2000

VLSI ARRAY STRUCTURES FOR DIGITAL SIGNAL PROCESSING

FINAL PROJECT REPORT

ONE & TWO DIMENSIONAL DISCRETE COSINE TRANSFORM

IMPLEMENTATIONS

PART I: 2-DIMENSIONAL DCT PERFORMANCE SURVEY

PART II: 1-DIMENSIONAL DCT / IDCT HARDWARE DESIGN

**VIJAY S. SRINIVASAN
ELECTRICAL AND COMPUTER ENGINEERING
UNIVERSITY OF WISCONSIN-MADISON**

Introduction : Motivation and Background

The Discrete Cosine Transform has long been the basic transform coding method for the JPEG and MPEG standards. It helps separate the image into parts (or spectral sub-bands) of differing importance - with respect to the spatial quality of the image. In that respect it is similar to the Discrete Fourier Transform since it transforms a signal or image from the spatial domain to the frequency domain. However one primary advantage of the DCT over the DFT is that the former involves only real multiplications, which reduces the total number of required multiplications, unlike the latter. Another advantage lies in the fact that for most images much of the signal energy lies at low frequencies, and are often small - small enough to be neglected with little visible distortion. The DCT does a better job of concentrating energy into lower order coefficients than does the DFT for image data. This characteristic of the DCT, referred to as energy compaction efficiency, along with other advantages resulted in the JPEG and MPEG standards adopting the DCT as a standard for image compression.

The 2D DCT formula can be implemented in hardware or software, and numerous fast implementations exist today based on fast algorithms developed for computing the 1D DCT.

Two-Dimensional Discrete Cosine Transform Implementation Survey : Background

The first part of this project, the Survey, compiles together a list of recent DCT implementations in hardware and software. Implementations have been divided into 4 categories : ASIC based, FPGA based, DSP based and General Purpose Microprocessor based implementations. The first two are hardware and the latter two are software implementations. Performance figures are given for each implementation on a maximum of four parameters : speed, area, power and precision. Rankings based on the figures are estimated for each implementation inside its own category, specifically for example ASIC based designs are not ranked against FPGA based designs. The objective is to gain an insight into 2D DCT implementation platforms and styles, and primarily to get an idea of state-of-art performance figures available today.

One-Dimensional Discrete Cosine Transform Design & Implementation : Background

The second part of the project, the Implementation, involves the schematic based design of a 1D DCT straight from the formula (no fast algorithm) using Altera's MaxplusII FPGA design tools. The speed of computation and area usage is compared with two other IEEE documented 1D DCT implementations on Altera devices. Design files - schematics and simulation traces - have been included at the end of the report. The objective is to come up with a working 1D DCT implementation, not fast or optimal in any way (in fact slow and big), but correct and able to demonstrate why faster and smaller 1D DCT implementation is a big issue.

PART I

2-DIMENSIONAL DCT IMPLEMENTATION PERFORMANCE : A SURVEY

The survey compiled focuses primarily on the performance figures of the various implementations studied, and a brief description of the architecture of each implementation indicates methods followed for the implementation. An effort has been made to explain architectures without focusing on the details, and some frequently occurring terms have been explained below.

Brief explanations of frequently occurring terms

1D DCT usage in computing the 2D Discrete Cosine Transform : The formula for the 2D DCT is separable, which means that it can be broken into two sequential 1D DCT operations, one along the row vector and the second along the column vector of the preceding row vector results. Called the Row-Column decomposition method, this is the most common method deployed for computing the 2D DCT, and implementations usually focus on optimizing the 1D DCT so that the Row-Column 2D DCT implementation performs better when using the optimized 1D DCT block along rows and columns. Roughly ninety percent of the survey implementations below follow this method, and in the rest the implementations are either a direct 2D DCT computation using the 1D DCT in other ways or a recursive computation where a 1D DCT is used for 2x2 2D DCT, then this is used for a 4x4 2D DCT and so on.

Distributed Arithmetic : This is a technique very commonly used where Multiply-Accumulate plays a predominant role in the operation, especially true with signal processing applications. Typically it serves to eliminate multiplications and replace them with adds, which is useful since a multiplication consumes much more time than an add. An example of the result is a case where N multiplies followed by an N-input add has been replaced by a series of N-input adds followed by a single multiply.

Bit serial architecture : Primarily used in the context of multipliers, these are architectures where a single bit of each input word is transmitted during each processing cycle. This reduces I/O, however an n-bit word requires n-processing cycles for transmission. There is typically a large area overhead in latches associated with serial architectures.

Asynchronous designs : These do not employ global clocking, and instead rely on local handshaking protocols to activate only those components that are necessary to perform a given computation. This reduces switching capacitance and activity in general and presents power saving potential.

Reporting / Estimation of Performance Figures

Throughput : In all the cases below I have given a value of this parameter for a measurement of speed, and I have estimated its value whenever the concerned publication did not give a specific value, but instead gave a clock frequency. My assumption in such cases is that for the initial 64 samples there is a latency of 64 cycles during which data values are read in and 1D DCT is performed along rows (row-column method), and in the next 64 cycles the DCT values are output, and after this initial phase the rest of the outputs occur in a pipelined fashion - that is, while the 64 DCT values are output, the next 64 values are read in parallel at the input, and so on. This works out to a throughput of 0.5 pixels / cycle for the initial 64 values, and 1 pixel / cycle for the rest of the data. There might be cases where my assumption may not be entirely correct, and the design may in fact continue to function at 0.5 pixel / cycle rate or less (no parallel input of next set of data while output of present data), or may even function at pixel/cycle values greater than 1 (example - more than 1 pixel input / DCT output per cycle). But the assumption I make above can be taken as an average case between these extremes, and serves to provide some measure of speed when otherwise there is no other data available for speed apart from the clock frequency. In cases where data for Latency is available, I take this data into account for throughput and the pixel rate in such cases will be less than 1 per clock cycle.

Comparisons/Ranks : Regarding throughput, I have already explained my basis for valuing this parameter. For area, in cases where only the number of gates is available, I make an estimate using an average of 4 transistors per CMOS gate (assumed 2-input nand equivalent) for ASIC designs, and for FPGAs where data for Logic Elements / Logic Blocks is given, I make an estimate of the average number of usable gates per LE / LB from the corresponding datasheet of the device from the manufacturer's website. Power figures are reported as they are, and are not estimated at all when the number is not available, except in one case where energy data was available. Precision data is inferred from the publication when it is not specifically given, and usually involves studying the block diagram, signal flow graph (ASICs and FPGAs) and CPU register lengths (DSPs and Microprocessors). All rank figures are given as (rank position) / (total comparisons for that parameter in that implementation category)

References: IEEE/IEE Publications, company websites, and other independent publications available on the web have been used extensively for collecting data, and in order for the data to be reasonably relevant to available technology today, my basis for data collection is as follows : For ASIC and FPGA based designs I have restricted my reporting data for IEEE / IEE transactions/conferences/journals since 1998. For Microprocessor and DSP based designs the data reported has been restricted to implementations documented since 1995, except in a couple of cases where figures after 1995 are not available, and post 1990 figures are used. For each case I have given the People involved, Year of Implementation, and indicated the reference Publication, below the title of each implementation. Additional references have been listed at the end.

(A) HDL / Schematic based Hardware Implementations – ASIC Based

1. Configurable Image Transform Processor Unit.

Stephen Molloy, Rajeev Jain : January 1998, IEEE Journal of Solid State Circuits

Description

This implementation executes a number of image transform operations including an 8x8 Discrete Cosine Transform, 1D or 2D FIR Filter and NxN vector quantization. The configurable nature of the design allows the user to select the type of transform operation to execute, and the appropriate parameters (for example whether to execute a 4x4 or 8x8 DCT). It follows the Row-Column method using an N-point 1D DCT. Booth Multipliers and a Carry-Save Adder Tree are used for computing 1D DCT along rows and columns. The main execution unit is fabricated as an ASIC, while the reconfigurable / programmable control logic has been implemented on a Xilinx FPGA

Performance

Speed : 64 results complete in 128 cycles (no output for first 64 cycles, 64 outputs in next 64 cycles) ; Maximum Clock Frequency : 50Mhz (20ns clock). Thus 64 pixels (8x8 block) complete in 2560 ns, or 25 million pixels /second

Rank : #5 / 9

Area : 1.2um CMOS process ; 79 sq. mm. die size ; 180-pin PGA ; 110K transistors

Rank : #5 / 9

Power : 1.39W @ 50Mhz, 5V

Rank : #4 / 4

Precision / Accuracy : Pixel Input : 14-bit ; Coefficient Input : 12-bit ; Internal Precision : 26-bit (maintained in full throughout internally) ; Rounded Output : 14-bit

2. Low Complexity 2D DCT Processor for multimedia applications

L. Fanucci, R. Saletti, F. Vavada : 1999, 6th IEEE International Conference on Electronics, Circuits and Systems

Description

The implementation makes use of the separability technique as a calculation method (1D DCT on rows and columns) and distributed arithmetic for the architecture. The implementation has been made completely H.263 compliant, including precision and mean square error levels. As the main objective was reduced complexity, various architectures for the computation of the 1D DCT were analyzed and it was found that the distributed arithmetic architecture offered the least complexity (5K gates for an 8 point 1D DCT) as compared to others (Parallel Multiplier type : 43 Kgates, Carry Save Adder and Multiplier type : 68 Kgates, and Booth Multiplier type : 20 Kgates) The

implementation has been described in VHDL and synthesized using the Synopsys logic synthesis tool. The processor supports both Forward and Inverse DCT.

Performance

Speed : Maximum Frequency : 36Mhz ; Estimating with the steady state 1 pixel / cycle standard, this translates to a throughput of 36 million pixels / cycle

Rank : #2 / 9

Area : 0.8um CMOS semi-custom library from AMS foundry; 15 Kgates; 33 sq. mm.

Estimating 4 transistors / gate for CMOS, this translates to ~60K transistors

Rank : #3 / 9

Precision / Accuracy : Pixel Input : 9-bit ; Coefficient Input : 12-bit ; Internal Precision, Output: H.263 compliant

3. 2D DCT / IDCT for real-time video applications

Ig-Kyun Kim, Jin-Jong Cha, Han-Jin Cho : 1999, 6th International Conference on VLSI and CAD

Description

The implementation satisfies the accuracy specification of the ITU-T. The separability technique is used to apply the 1D DCT along rows and columns to compute the 2D DCT, and uses distributed arithmetic combined with bit-serial and bit-parallel structures to implement the inner vector product concurrently. No multipliers are required as a consequence (memory look up tables instead) and the accuracy is higher while maintaining internal precision compared to earlier implementations, because the accumulated results undergo fewer rounding and truncation stages.

Performance

Speed : Maximum Clock Frequency / Input Sampling Rate: 33Mhz

Throughput = 33 million pixels / second, assuming 1pixel / cycle

Rank : #2 / 9

Area : 0.5um CMOS technology library ; 10000 gates ; estimating 4 transistors / CMOS gate, number of transistors = 40K transistors

Rank : #3 / 9

Precision / Accuracy : Pixel Input : 9-bit ; DCT Output : 12-bit

4. Simple processor core design for DCT/IDCT

Tian-Sheun Chang, Chin-Sheng Kung, Chein-Wei Jen : April 2000, IEEE Transactions on Circuits and Systems for Video Technology

Description

The implementation is a cost-effective processor core design that is H.263 compliant and suitable for incorporation into a digital camera. The implementation uses techniques like distributed arithmetic, a fast direct 2D DCT algorithm and sub-expression sharing to reduce the hardware cost and enhance computing speed. It uses the direct 2D approach rather than the row-column decomposition approach, since the former requires N 1D DCT computations as opposed to $2N$ computations and memory transposing hardware in the latter, for a $N \times N$ 2D DCT.

Performance

Speed : Operating Frequency : 54.9Mhz ; Throughput : 23.6 million pixels / second
Rank : #7 / 9

Area : 0.6um SPDM CMOS library; 1493 gates; core area : 0.78 sq.mm. ; 24K transistors
Rank : #1 / 9

Precision / Accuracy : Precision : Output 16-bit

5. Design and Implementation of DCT / IDCT chip with a novel architecture

Kuo-Hsing Cheng, Chih-Sheng Huang, Chun-Pin Lin : May 2000, IEEE International Symposium on Circuits and Systems

Description

The implementation makes use of the row-column decomposition method, and uses a modified version of an existing fast algorithm by which the multiplier designs are simplified by making all the coefficients positive cosine forms and having less round off error. After final simplification of the original DCT signal flow graph corresponding to the original algorithm, the design is found to require 8 multipliers versus 13 in the original, and 21 adders versus 29 in the original. .

Performance

Speed : Operating Frequency : 100Mhz ; Estimated throughput - Using the 1 pixel / cycle steady state norm, this works out 100 million pixels / second

Rank : #1 / 9

Area : 0.6um Compass CMOS library; 39K gates; core: 15.33 sq.mm.; 156K transistors
Rank : #9 / 9

Precision / Accuracy : Input Data : 9-bit ; Output Data : 12-bit

6. Design and Synthesis of Built-in Self Testable 2D Discrete Cosine Transform circuits

Han Bin Kim, Dong Sam Ha : 2000, 13th Annual IEEE International ASIC / SOC Conference

Description

This is an implementation of a 2D DCT with built-in self test capability (BIST). The design achieves high fault coverage with a small area overhead and system performance degradation. The 2D itself is implemented using the row-column decomposition method with a transposing memory, and the algorithm employed is the one proposed by Chen et al. Due to the area overhead the number of multipliers used increases to 24 as compared to 16 for Chen's algorithm. The testing process gets 100% stuck-at fault coverage. The circuit is described in VHDL and a high-level BIST synthesis tool is used to generate a datapath and controller in the register transfer level with BIST capability by reconfiguring existing registers. The area overhead of the BIST circuitry is found to be 9.95%.

Performance

Speed : Operating Frequency (with BIST): 30.8Mhz ; Throughput : Estimating 1 pixel / cycle in the steady state after latency period, this works out 30.8 million pixels / second

Rank : #4 / 9

Area (with BIST) : 0.5um HP CMOS library ; 27K gates; core: 11.92 sq.mm. Calculating with 4 transistors / 2-input Cmos nand gate, this works out to 108K transistors

Rank : #4 / 9

Precision / Accuracy : Output : 16-bit

7. Micropipelined Asynchronous Discrete Cosine Transform Processor

David Johnson, Venkatesh Akella, Brett Scott : December 1998, IEEE Transactions on VLSI Systems

Description

The implementation is an asynchronous one to demonstrate the usefulness of such designs when clock generation (skew and distribution overhead) and the power consumed in clock circuitry become limiting factors. The asynchronous pipeline (also known as a micropipeline) is different from the synchronous one in that it can contain a variable number of data items, and is in fact a FIFO when no logic is present. The row-column decomposition method is used to compute the 2D DCT, and a distributed arithmetic architecture is used for the 1D DCT block. 2 separate 1D DCT blocks along with a transposing RAM are used in the 2D DCT architecture. The first such implementation by the authors was then optimized with 2 improvements - using double-edge triggered flip-

flops instead of level-sensitive latches to save more power, and using completion detection circuitry to exploit data-dependent processing delay.

Performance

Speed : Latency(best optimized design) : 1.414us ; Throughput (best optimized design) : 1.373us/block ; Estimating total time for 1 8x8 block = 1.414+1.373 = 2.787us, this translates to 64 pixels / 2.787us, or 22.964 million pixels / second

Rank : #8 / 9

Area (best optimized design) : 153K transistors

Rank : #8 / 9

Power : Energy= 0.932uJ/block; Estimating power consumption as amount consumed during actual conversion, 0.932 uJ is consumed in 1.373 us. This translates to ~0.678W

Rank : #3 / 4

Precision / Accuracy : Input : 9-bit ; Internal precision : 15-bit ; Output : 12-bit

8. Low Power 2D DCT chip design for wireless multimedia terminals

Liang-Gee Chen, Juing-Ying Jiu, Hao-Chich Chang, Yung-Pin Lee, Chung-Wei Ku : 1998, IEEE International Symposium on Circuits and Systems

Description

This implementation is based on a direct 2D algorithm which employs N 1D DCTs with some additions, and a distributed arithmetic architecture. Compared to the row-column method, which requires 2N 1D DCTs, this significantly contributes to power reduction. The adder employed in accumulation for the 1D DCT is a carry-select, Manchester type (which improves on the carry-look ahead by using a single gate) for speed and low power. The ROM used for the DA-based architecture is improved upon for low-power, and a low-voltage 2-port SRAM is used.

Performance

Speed : Latency : 198 cycles ; Clock Frequency : 100Mhz ; Estimating that 198 cycles pass without any output (computation occurs in this time) and in the next 64 cycles the DCT coefficient results are output, this means 198+64=262 cycles for 64 pixels @100Mhz. This translates to 24.43 million pixels / second

Rank : #6 / 9

Area : 0.6um SPDM CMOS library; 152K transistors ; core area : 50.63 sq.mm.

Rank : #7 / 9

Power : Supply Voltage : 2V ; Consumption : 138mW

Rank : #2 / 4

Precision / Accuracy : Input : 9-bit ; Internal Precision : 16-bit ; Output : 16-bit

9. Low Power DCT core using adaptive bitwidth and arithmetic activity exploiting signal correlations and quantization

Thucydides Xanthopoulos, Anantha Chandrakasan : May 2000, IEEE Journal of Solid-State Circuits

Description

This implementation uses the row-column decomposition method and a distributed arithmetic method for computing the 2D DCT. The Chen fast DCT algorithm is used, enhanced with activity reduction methods. The 2 techniques used for activity reduction are the MSB rejection method (rejects common MSBs of AC coefficients thus reducing internal bit widths for processing) and a row-column classification method (maximizes image peak SNR and minimizes number of arithmetic operations), both of which together result in a 50% power reduction.

Performance

Speed : Max Clock Frequency : 43 Mhz ; Assuming it takes 64 cycles for data read in and 64 cycles for DCT coefficients to complete reading out (the common case) this translates to 128 cycles for 64 pixels @ 43 Mhz, or 21.505 million pixels / second

Rank : #9 / 9

Area : 0.6um 3ML CMOS technology library ; 120K transistors ; area : 14.5 sq.mm.

Rank : #6 / 9

Power : Min Supply Voltage : 1.1V ; Consumption : 4.38mW @ 1.6V, 14Mhz

Rank : #1 / 4

Precision / Accuracy : Input : 8-bit ; Internal Precision (inferred) : 8-bit ; Output (inferred) : 8-bit

(B) HDL / Schematic based Hardware Implementations – FPGA Based

1. Epld based architecure of Real Time 2D Discrete Cosine Transform

S. Ramachandran, S.Srinivasan, R.Chen : 1999, IEEE International Symposium on Circuits and Systems

Description

This implementation fits a 2D DCT onto one piece of a commercially available Embedded Programmable Logic Device (EPLD). In addition the output is a quantized DCT stream. The architecture is regular and highly pipelined, and has been realized using VHDL and schematic packages. The design has been implemented using an Altera Flex10K Epld device.

Performance

Speed : 79 clock cycles for an 8x8 block ; Frequency of operation : 25Mhz ; This translates to 3160ns for 64 pixels, or 20.253 million pixels / second

Rank : #1 / 6

Area : 160K gates ; 80% Flex10K device utilization ;

Rank : #6 / 6

Precision / Accuracy : Input : 8-bit ; Output (quantized) : 10-bit

2. Parallel Implementation of 2D-Discrete Cosine Transform using Epld

D.V.R. Murthy, S. Ramachandran, S.Srinivasan : 1999, 12th International Conference on VLSI Design

Description

The implementation is a regular, linear, pipelined 2D DCT which fits into 4 pieces of commercially available Eplds. The design has been realized using VHDL and fitted into Altera's Flex10K Epld devices.

Performance

Speed : Operating Frequency : ~7.4 Mhz or a 135ns clock ; 8x8 block computes in 72 clock cycles = 9720ns, this translates to 6.584 million pixels / second

Rank : #6 / 6

Area : 30K gates in each of 4 devices = 120K gates total ; 65% to 75% chip utilization in each device

Rank : #5 / 6

Precision / Accuracy : Input : 8-bit ; Output : 15-bit

3. Applying an XC6200 to Real Time Image Processing

Roger Woods, David Trainor, Jean-Paul Heron : 1998, IEEE Design and Test of Computers

Description

The implementation exploits an algorithm's inherent parallelism, and uses pipelining and distributed arithmetic organization. The 1D DCT is applied in the row-column decomposition method, and the 2D architecture itself is made of 2 1D DCT blocks and a matrix transposing circuit, along with parallel-serial-parallel converters. Implementation is on a Xilinx device.

Performance

Speed : Operating Frequency : 23.04Mhz or a 43.4ns clock ; Processing capacity = 25 frames of VGA resolution / second = 640x480x25 pixels/second or 7.68 million pixels / second

Rank : #5 / 6

Area : The design occupies a 78 x 64 block within the Xilinx FPGA, or 4992 cells. A cell is seen to be equivalent to a flip-flop and a fixed number of gates and-or logic) from the paper, and therefore the design is estimated to take up area equivalent to roughly 5000 x (flip-flops and the fixed gate number) Estimating a maximum of roughly 10 nand equivalent gates for a edge triggered D-flop and an equal number of used gates for and-or logic per cell, this translates to 5000 x 20 gates or roughly 100K used gates for the design

Rank : #4 / 6

Precision / Accuracy : Input: 8-bit; Internal Precision: 12-bit; Output : 12-bit

4. Efficient Implementation for high accuracy DCT processor based on FPGA

L.Naviner, J-L.Danger, C.Laurent : 1998, Ecole Nationale Supérieure des Telecommunications[see additional references : #7]

Description

The implementation is based on distributed arithmetic architecture, and complete use of the logic cells of the programmable device has been made with various architectural optimizations which include pseudo multiplexing, special encoding and resource sharing for multiplications, additions and accumulations of partial inner products.

Performance

Speed : Operating Frequency : 36Mhz ; Throughput : 18 million pixels / second

Rank : #2 / 6

Area : Logic Elements : 2040 ; Estimating 20 gates per logic element (10 for the register and 10 and-or logic gates) this translates to roughly 40K used gates for the design

Rank : #1 / 6

Precision / Accuracy : Input : 11-bit; 17-bit ; Internal Precision : 19-bit ; Output : 14-bit

5. Discrete Cosine Transform : IP available for purchase from Altera Corporation

Amphion Semiconductor Ltd : available now

Description

This is a commercially available IP designed by Amphion Semiconductor Ltd and available for purchase/licensing from Altera Corporation. It is a fully parameterized

function (to comply with H.261, H.263, JPEG, MPEG-1, MPEG-2, MPEG-3), and also incorporates the IDCT. The function can fit into a single Flex10K100 device.

Performance

Speed : Operating Frequency : upto 50Mhz ; Throughput : 17.45Mhz or 17.45 million pixels / second

Rank : #4 / 6

Area : Logic Cells used : 4386 ; Estimating roughly 20 used gates per cell (10 for the register and 10 for and-or logic, this translates to roughly 88K used gates for the design

Rank : #2 / 6

Precision / Accuracy : Parameterizable, typical settings are given : Input : 8-bit ; Internal Coefficients : 8-bit ; Internal Precision : 12-bit ; Output : 8-bit

6. X DCT IDCT Forward and Inverse Discrete Cosine Transform : IP available for purchase from Xilinx Corporation

Xentec Inc : Available for purchase Feb 2000

Description

The implementation incorporates the forward and inverse DCT and has been tested with a variety of Xilinx Devices including the Virtex, Virtex-E and Spartan-II devices. The design is mentioned as fully synchronous, allowing fast operation and having a low gate count. Both the synthesized EDIF version of the core and RTL synthesizable source code are available from Xilinx.

Performance

Speed : (best figures, IDCT-DCT combined, Virtex-E device) : Operating Frequency : 36Mhz (27.77ns clock) ; Latency 64 cycles ; Estimating 128 cycles for DCT results to complete output from the start of Input, this works out to 3555.5ns for 64 pixels which translates to roughly 18 million pixels / second

Rank : #2 / 6

Area :

(Virtex-E Device, combined DCT-IDCT)

1027 Configurable Logic Blocks ; Looking up the Virtex-E Data Sheet from the Xilinx web pages, one CLB is equivalent to 4.5 Logic Cells, each of which consist of a register, carry logic and function generator - together estimating 20 gates usage per logic cell, and therefore 90 gates usage per CLB, this translates to roughly the equivalent of 92K gates for the design

Rank : #3 / 6

Precision / Accuracy : Input : 8-bit ; Output : 12-bit

(C) Digital Signal Processor based Software Implementations

1. Image Processing on the TMS320C6X VLIW DSP

Brian L. Evans, Niranjan Damera-Venkata, Magesh Valliappan : Presentation at the University of Texas at Austin

Description

The implementation is an exercise to benchmark a JPEG codec on a high performance commercial DSP. The row-column decomposition method is used to break the operation into 1D DCT along rows and columns. The device used was a Texas Instruments TMS320C6201 as part of a TI C6x Evaluation Module.

Performance

Speed : The TMS320C6201 on the evaluation board runs at 133 Mhz (7.5ns clock) ; The computation of the forward DCT takes 226 cycles ; this works out to 1695ns for 64 pixels which translates to roughly 38 million pixels / second. There are a couple of caveats here : (1) The input data is assumed already present in the memory before the 226 computation cycles start (2) Output data is stored back into memory and not serially output like the hardware implementations described previously.

Rank : #1 / 5

Precision / Accuracy : Input : 16-bit ; Output (inferred) : 16-bit

2. An 8 x 8 Discrete Cosine Transform Implementation on the TMS320C30

Application Report : SPRA115, available from Texas Instruments web site

Description

The implementation is an application note available from Texas Instruments' web site, and uses the row-column Decomposition method to compute the 2D DCT. The TMS320C30 is a floating point DSP which has the ability to compute parallel multiplications/additions, and extended precision registers, which results in a higher speed and precision as compared to an implementation on the TMS320C25, described in the same application note.

Performance

Speed : Operating frequency : 40Mhz ; Computation time for forward DCT : 33.6us ; This works out to 33.6us for 64 pixels or roughly 2 million pixels / second

Rank : #3 / 5

Precision / Accuracy : Input (inferred) : 32-bit ; Output (inferred) : 32-bit

3. Feig's 8 x 8 DCT algorithm on the Motorola DSP56300

Dror Halahmi : September 1998, Application Note, Motorola Semiconductor Israel

Description

This implementation incorporates both the forward and inverse DCT. A reference code is written in MATLAB to keep a benchmark for 2D DCT results, and DSP results are compared with these. A comparison is also made between a straightforward implementation and the fast implementation with Feig's algorithm.

Performance

Speed : Direct computation from the formula : 21661 cycles ; Fast implementation using Feig's algorithm : 1245 cycles ; From Motorola's datasheets it is seen that the fastest DSP56300 family device operates at 150Mhz (6.66ns clock); this works out to 8300ns for 64 pixels (fast implementation), or a throughput of 7.7 million pixels / second ;

Rank : #2 / 5

Precision / Accuracy : Input : 9-bit ; Internal precision : 24-bit ; Output : 12-bit

4. Digital Signal Processing on the Motorola ColdFire Architecture

William Hohl, Joe Circello : Last updated 1996, Motorola ColdFire Application Note

Description

The row-column decomposition method is used to compute the 2D DCT. Though fast implementation can be achieved by using a fast algorithm, this implementation uses a direct matrix formulation to multiply input data by coefficients - and therefore is a little slow. An MCF5200 device is used for testing.

Performance

Speed : Operating frequency : 33Mhz ; 6300 cycles ; This works out to 189us for 64 pixels, or a throughput of 0.34 million pixels / second

Rank : #5 / 5

Precision / Accuracy : Input (inferred) : 16-bit; Output (inferred) : 32-bit

5. Discrete Cosine Transform on an Analog Devices DSP

Analog Devices Application Note, last updated 1993

Description

The implementation follows the row-column decomposition method for the 2D DCT, and is done on an ADSP-2100 family device. The routine is an in-place computation, and uses

Hou's Fast Discrete Cosine Algorithm to perform the operation - this is a numerically stable, fast and recursive algorithm, where the next larger DCT is generated from 2 identical, smaller DCTs. Performance figures from the implementation on the ADSP-2171 have been given below.

Performance

Speed : Operating Frequency : 33Mhz ; 2492 cycles ; This works out to 75.5us for 64 pixels, or a throughput of 0.85 million pixels / second

Rank : #4 / 5

Precision / Accuracy : Input : 8-bit ; Output : 12-bit

(D) General-Purpose Microprocessor based Software Implementations

1. A Fast Precise Implementation of 8 x 8 Discrete Cosine Transform using SIMD extensions and MMX Instructions

Intel Application Note : April 1999

Description

The implementation is an exercise in demonstrating the capabilities of the Single Instruction Multiple Data Extension (SIMD) and MMX Instructions of Pentium III processors. No application note is currently available to demonstrate the SSE2 capabilities of the Pentium 4 processor for an 8x8 DCT (though Pentium 4 8x8 IDCT performance results are available), therefore results given below do not necessarily indicate state-of-art processing capabilities of 8x8 DCTs on Intel Processors. The implementation in this application note incorporates the Forward and Inverse DCTs, and the 2D DCT is computed by means of the Row-Column method. However a transposition overhead, almost always the case with other Row-Column implementations, is avoided by transforming groups of 4 columns (after 1D DCT computation along rows) in parallel. In addition the forward 2D DCT processes columns first, then rows. The 1D DCT itself is computed directly from the formula, involves 32 multiplications and 24 additions, and the entire 2D DCT operation takes 256 multiplications and 256 additions (excluding auxiliary operations)

Performance

Speed : Clock Cycles : 250 ; Calculating with the fastest Pentium III processor available today (933Mhz, 1.1ns clock), this works out to 275ns for 64 pixels which translates to roughly 233 million pixels / second. There are a couple of caveats here - this high figure cannot be realistically compared with figures for hardware implementations, which take

into account the entire process from loading in 64 pixels and streaming out 64 DCT results. The figures here indicate the time it takes for the processor to take values already present in memory, process them, and put them back to memory.

Rank : #2 / 2

Precision / Accuracy :_(satisfies IEEE standard 1180-1990) ; Input : 9-bit ; Internal Precision : 16-bit ; Output : 16-bit

2. 2D Discrete Cosine Transform on the PowerPC (AltiVec extended) architecture

Motorola Application Note : Last updated May 1999

Description

This C/C++ implementation demonstrates the SIMD capable AltiVec extension to the existing PowerPC architecture. It uses the row-column decomposition method for computing the 2D DCT, and the 8 point 1D DCT used here is a scaled version of Chen's algorithm. A matrix transposition operation is used, unlike the Intel implementation discussed previously. The authors indicate that performance can vary depending on the C compiler, and most importantly, it is assumed that all required instructions and data are present in the L1 cache - and therefore the same restrictions on comparisons discussed previously for GP microprocessors apply.

Performance

Speed : Clock cycles : 102, includes function call overhead ; The fastest AltiVec extended PowerPC microprocessor available in the market today runs at 733Mhz (1.36ns clock, source : motorola website), this works out to 139ns for 64 pixels, which translates to a throughput of 460 million pixels / second. The L1 cache condition mentioned above must be re-emphasized here along with this amazingly high figure

Rank : #1 / 2

Precision / Accuracy : Input (inferred) : signed 9-bit 2's complement ; Output (inferred) : signed 12-bit 2's complement

PART II

1-DIMENSIONAL DCT / IDCT : DESIGN & IMPLEMENTATION

I have designed and implemented an 8-point 1-Dimensional Forward and Inverse Discrete Cosine Transform's datapath using Altera's MaxplusII FPGA design tools and a schematic based approach, and fitted it into an Altera Flex10K device. The primary objective of the implementation was to develop a basic 1D DCT datapath, simulate it, and thereby get an estimate of how fast a straightforward formula based 1D DCT would run. To this end, the control logic has not been incorporated inside the design. A complete 1D DCT design can be implemented by incorporating the control logic in the design, but for the purposes of the project the control logic signals are provided externally through the simulator. This serves the purpose well enough since only the DCT datapath speed and area are relevant here.

Formulae

1-Dimensional Forward Discrete Cosine Transform :

$$C(u) = a(u) \sum_{x=0}^{N-1} f(x) \cos [(2x+1).u.\pi / 2N]$$

for $u = 0,1,2,\dots,N-1$

1-Dimensional Inverse Discrete Cosine Transform :

$$f(x) = \sum_{u=0}^{N-1} a(u) C(u) \cos [(2x+1).u.\pi / 2N]$$

for $x = 0,1,2,\dots,N-1$

Where,

$$a(u) = \begin{cases} \text{Sqrt} (1 / N) & \text{for } u = 0 \\ \text{Sqrt} (2 / N) & \text{for } u = 1,2,\dots,N-1 \end{cases}$$

Approach

For an 8-point Forward and Inverse DCT, $N = 8$, and plugging this value with $x, u = (0,1,2,\dots,7)$ into the above formula, a relatively straight forward series of Multiply-Accumulate (MAC) operations are obtained which are more intuitive to implement using digital logic.

Proceeding with the above operations, the following example equations are obtained for the forward and inverse DCTs (3-decimal place precision) :

Forward :

$$c(0) = 0.353 [f(0) + f(1) + f(2) + f(3) + f(4) + f(5) + f(6) + f(7)]$$

$$c(1) = 0.491f(0) - 0.491f(7) + 0.416f(1) - 0.416f(6) + 0.278f(2) - 0.278f(5) + 0.098f(3) - 0.098f(4)$$

....continues till $c(7)$

Inverse :

$$f(0) = 0.353c(0) + 0.491c(1) + 0.462c(2) + 0.416c(3) + 0.353c(4) + 0.278c(5) + 0.192c(6) + 0.098c(7)$$

$$f(1) = 0.353c(0) + 0.416c(1) + 0.192c(2) - 0.098c(3) - 0.353c(4) - 0.491c(5) - 0.462c(6) - 0.278c(7)$$

...continues till $f(7)$

To put it simply, this series of MAC operations requires a multiplier for generating the various product terms using the 7 coefficients (0.098, 0.192, 0.278, 0.353, 0.416, 0.462, 0.491) and an accumulator for summing up the product terms in different ways. At first glance it seems as if the entire operation reduces to designing a large combinational logic unit which uses a large number of multipliers and accumulators. However this is not possible because :

(1) Such a combinational unit uses an extremely large number of gates, which makes the design impossible to fit in any device available from Altera's device library as part of the MaxPlusII ver10.0 Baseline software that is available free to students.

(2) Even when the design is optimized a little so that the number of gates used will be within range of an available device, the MaxPlusII design compiler is not able to compile a design with a large number of multipliers, possibly due to insufficient resources that are required for computationally intensive blocks like multipliers.

Having encountered these problems, a totally different implementation approach had to be thought out. The solution was a serial, shift - register based architecture.

Implementation

The final solution (reference : top level design schematic) uses the following components:

(1) *Multiplier Unit* (quantity : 1) : Accepts 10-bit input data, multiplies this data with the 7 coefficients mentioned previously, outputs 7 11-bit product terms for that input data. Repeats this operation for the 8 points of input data.

Inside Architecture : made with parameterized multipliers (Altera MaxPlus component) : Inputs 10 bits (input data) and 13 bits (coefficients) wide, output 23 bits wide, truncated to 11 bits.

(2) Register File Unit (quantity : 8) : Set of registers 11-bits wide, registers product term data on positive clock edge from either (1) Immediately preceding multiplier/register file unit in the shift-register architecture(2) another register file unit in the shift register architecture for Forward DCT specific-rearrangement (3) another register file unit in the shift register architecture for Inverse DCT specific-rearrangement. The input is made selectable by a select logic input incorporated in the design, driven by the control logic (provided externally here via the simulator)

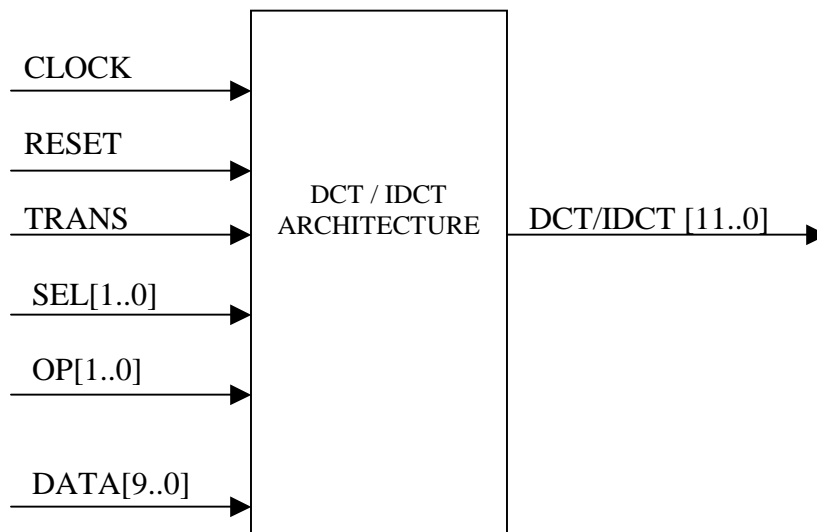
Inside Architecture : made of Multiplexers, 8-bit Registers available from MaxPlus library

(3) Accumulator Unit (quantity : 1) : Accepts 8 11-bit product term data values and one 2-bit Op-code that specifies either (1) Addition of all 8 product terms (2) Subtraction of sum of last 4 product terms from the sum of first 4 product terms (3) Subtraction of sum of last 5 product terms from the sum of first 3 product terms (4) Subtraction of sum of last 3 product terms from the sum of first 5 product terms. These 4 types of accumulations are the only operations to be performed in the accumulation stage of the FDCT and IDCT MAC operations, and can be seen in the MAC equation examples given previously. Outputs 12-bit results.

Inside Architecture : made of parameterized 12-bit adders/subtractors and a parameterized multiplexor available from the MaxPlus library.

About the Multiplier and Accumulator Units

The Multiplier uses signed, fixed-point 13-bit representation for the coefficients (accurate to 3 decimal places), signed 2's complement 10-bit representation for input data (accommodates 2 decimal place accuracy for IDCT input data), and a truncated, fixed-point, signed 2's complement 11-bit representation for the result product terms. The Accumulator uses 1 guard bit to accommodate extreme transform DCT/IDCT values, and outputs 12-bit DCT/IDCT data which is read as follows : The 2 LSBs of the result indicate 2 binary places of accuracy ; The 10 MSBs represent the integer magnitude of the result in 2's complement format. A Block Diagram of the final implementation has been drawn below.



1D DCT / IDCT System Operation : Walk Through

#0 : A reset pulse (active high) is provided at the RESET input of the system, and a clock input is maintained to the system on the CLK input pin after the reset.

#1 : The start of 10-bit input data is signalled by a rising edge on the TRANS input.

#2 : Parallel to TRANS rising edge, the first input data is input at DATA[9..0] and routed through the multiplier, which generates the 7 product terms corresponding to that data, and these values are registered into the shift register architecture ; this process is repeated for 8 clock cycles, at the end of which all the product terms corresponding to the 8-point data are present in the 8 registers of the shift register architecture. The SEL[1..0] input to the shift register architecture is maintained at 0 during these 8 cycles to enable normal shift operation, as each product-term set is generated, shifted into the next available register, and stored.

#3 : Before the 9th clock cycle, the 2-bit SEL[1..0] input to the shift is changed to either 1 or 2 depending on whether it is a Forward or Inverse DCT. Once this signal has been changed to 1 or 2, the ninth clock cycle rising edge re-arranges the current values in the 8 registers of the shift registers so that the last register contains exactly those product terms making up $c(0)/f(0)$ (depending on FDCT/IDCT), the one before that contains exactly those product terms making up $c(1)/f(1)$ and so on. Before the 10th clock cycle rising edge , the SEL[1..0] signal is changed back to 0 to re-enable normal shift operation.

#4 : Also in parallel with the 9th clock cycle, after the re-arrangement, the 8 product terms making up $c(0)/f(0)$ become available to the Accumulator for the final accumulate operation. The OP[1..0] signal is maintained at 0 for a 8-term add (since the first results $c(0)$, $f(0)$ of both the FDCT and IDCT are additions of 8 product terms. The Accumulator sums up these terms combinatorially and the first results $c(0)/f(0)$ are available after the 9th clock cycle rising edge.

#5 : The OP[1..0] signal is changed appropriately before each subsequent cycle for computing (accumulating) each result in the next 7 cycles, since some results are subtraction of sum of last 4 product terms from sum of first 4 product terms, some are that of the last 3 from the first 5, and so on. The remaining FDCT / IDCT results are available after each of the next 7 clock rising edges. The values for OP[1..0] correspond to the 4 types of accumulating operations (mentioned in the description of the Accumulator Unit) that are to be performed for each set of 8 product terms corresponding to each result.

#6 : The TRANS signal is lowered back to 0 after the 16th cycle (end of 8-point DCT).

Operation Summary : Serial input of 8-point data on the rising edge of each of the first 8 cycles; Serial output of 8 results after the rising edge of each of the next 8 cycles.

Results

2 sets of 8-point data were used to simulate the system : 1 set for FDCT and the other for IDCT. The FDCT and IDCT values were first benchmarked by using Matlab to calculate the reference values accurate to 3 decimal places. Then the simulation results were compared against these values. The results have been given in the following table.

Test vectors

FDCT input data : [75 63 22 111 88 17 66 05]

IDCT input data : [-20.25 66.75 -22.00 115.25 11.50 72.25 -100.00 15.25]

Result Table

Reference DCT Result	Implementation DCT Result	Absolute Error	Reference IDCT Result	Implementation IDCT Result	Absolute Error
158.038	156.75	1.288	69.814	69.25	0.564
36.713	37.00	0.287	7.600	8.25	0.650
-37.750	-38.75	1.000	-77.798	-77.75	0.048
20.553	20.25	0.303	23.258	23.75	0.492
39.244	39.25	0.006	29.149	29.25	0.101
30.966	31.00	0.034	-28.622	-28.25	0.372
-64.344	-65.50	1.156	53.918	53.75	0.168
-1.539	-2.50	0.961	-134.595	-134.25	0.345

The figures indicate an acceptable level of accuracy.

System Performance

It is easily seen from the design schematic and main simulation trace that the critical paths affecting the maximum frequency of operation are

Path #1 : The maximum input data - output product term path time through the Multiplier Unit during data input

Path #2 : The maximum clock rising edge - dct / idct output path time. This is composed of (max clock-output path time of the Shift Register) + (max input-output path time through the Accumulator Unit)

The greater of these values will decide maximum possible clock frequency, and therefore data throughput.

Accordingly 3 test points were traced out of the multiplier outputs to the top level to get an estimate of maximum input-output path time through the multiplier, the clock - dct / idct output being already available as part of the main simulation traces, simulations were

run, and the maximum path times for each of these two paths was recorded. However, the obtained path times may not be the maximum values, and may be slightly higher for particular bit combinations. Still, the figures give a rough idea of the path times for each of the above critical paths. The following data was obtained for the paths :

Path #1 (max) : 31.2 ns

Path #2 (max) : 34.2 ns

Taking into account the fact that other particular bit combinations may result in slightly higher values, we can estimate that *Path #2* is the critical path for the dct/idct system, and will assign this a maximum possible value of 35 ns. This translates to the maximum clock frequency being 28.5 Mhz, or a 35ns clock.

Final Speed and Area Performance Figures :

The number of cycles from start of input data to end of output results is 16 cycles, and at maximum frequency this works out to $16 \times 35 = 560$ ns, for one 8-point DCT. This translates to a throughput of 1.785 MDCTPS, or Million DCTs Per Second, a standard method to measure 1D DCT throughput. The design uses 2638 Logic Cells (source : generated design report), and at roughly 20 used gates per cell, this works out to roughly 53K used gates.

Comparisons with two IEEE documented 1D DCT implementations

Two IEEE documented 1D DCT implementations are compared with these figures.

(1) Implementation of the 1D Discrete Cosine Transform in the Residue Number System

P.G. Fernandez, A Garcia, J. Ramirez, L. Parrilla, A. Lloris : 1999, Conference Record of the 33rd Asilomar Conference on Signals, Systems and Computers

The Residue Number System is a powerful solution for maintaining constant performance while increasing precision, due to its limited carry propagation and parallel nature. This implementation is an 8-point 1D DCT processor on an Altera Flex10K device, and achieves a throughput improvement of over 21% with respect to a distributed arithmetic based processor. Gate usage has been estimated using used 20 gates / LE.

Speed : Throughput = 5.58 MDCTPS

Area : Logic Element usage = 1701, or ~34K used gates

Precision : Input 8-bit ; Output precision not available

(2) A new architecture to compute the DCT using the Quadratic Residue Number System
P.G. Fernandez, A Garcia, J. Ramirez, L. Parrilla, A. Lloris : May 2000, IEEE International Symposium on Circuits and Systems

This implementation is by the same people who implemented the 1D DCT processor mentioned previously, this time using the Quadratic Residue Number System. The reduction in total number of adders and multipliers is in the range 21-26% over an RNS based system, for a 1D DCT processor. Figures have been given for the fastest speed grade Flex10KE device. Gate usage has been estimated using used 20 gates / LE.

Speed : Throughput = 92.59 MDCTPS

Area : Logic Element Usage = 2308, or ~46K used gates.

Precision : Input 10-bit ; Internal 32-bit ; Output 24-bit

Final Analysis

ECE734 Fall2000 1-D DCT Implementation Performance Comparison Table

Implementation	Speed (MDCTPS)	Area (Gates)
1D DCT Quadratic RNS Architecture	92.59	46K
1D DCT RNS Architecture	5.58	34K
1D DCT ECE734 Fall 2000 Implementation	1.785	53K (datapath only)

*MDCTPS : Million DCTs Per Second

The above figures indicate that although the DCT / IDCT Implementation designed and implemented in this project is functionally correct, there is scope for a great deal of optimization.

Conclusions and Achievements

(1) A comprehensive survey of 2-Dimensional Discrete Cosine Transform Implementations has been done, and this has given me an idea of how 1D DCT blocks are used in 2D DCT implementations and what kind of performance I should expect in a high performing 2D DCT processor today. Second, I have become aware of various architectural techniques used to improve the performance of 2D DCT processors. And third, I have gained an insight into estimating performance data for digital designs in general by doing this survey, and I hope to be able to improve upon these skills further.

A note on estimation : Although every effort has been made to reasonably estimate data when it is not present, it is possible that some of the estimates made for the survey in this document may not be accurate because of possibly inaccurate assumptions. The estimated data should therefore not be used when correct data is required.

(2) A 1-Dimensional DCT / IDCT computing system has been implemented in hardware on an FPGA and simulated successfully. It is apparent that the straight-forward implementation that has been demonstrated in this project, although functionally correct, is no where near the kind of speeds that 1D DCT processors are expected to achieve today. Specifically comparing my ECE734 project 1D DCT/IDCT implementation against the last IEEE documented 1D DCT implementation, the latter is roughly 52 times faster than the former, while using roughly 13% fewer gates, with the entire control logic in the design ! This sums up why research into 1D DCT / IDCT implementations is an active field today - in order to achieve much better speeds at lesser cost.

Vijay S Srinivasan
01/31/2001

1-D DCT Design Schematics and Simulation Traces

Altera's MaxPlusII Baseline Version10.0 was used to implement and simulate the DCT design. The following schematics and simulation traces are available for download at <http://www.cae.wisc.edu/~vsriniva/ece734>.

1. Top Level DCT / IDCT schematic : dct.gdf
2. Multiplier Unit schematic : mult1x7.gdf
3. Accumulator Unit schematic : accumulator.gdf
4. Register File Unit schematic : register_11.gdf
5. DCT Simulation Trace (with the test vector given earlier in this document) : fdct.scf
6. IDCT Simulation Trace (with the test vector given earlier in this document) : idct.scf

Note :

1. Altera's MaxPlusII will be required to view the above files
2. The 8 DCT/IDCT results in the Simulation Traces are after the rising edge of each of the 8 cycles from 9th through 16th cycles, counted from rising Trans signal. Data format (2's complement): Integer Magnitude in 10MSBs; Fractional Magnitude in 2 LSBs.

Additional References and IEEE / IEE publication access resources

Company websites and documented implementations on the Internet :

1. <http://www.ti.com> : Texas Instruments
2. <http://e-www.motorola.com> : Motorola Semiconductor
3. <http://www.analog.com> : Analog Devices
4. <http://www.altera.com> :Altera Corporation
5. <http://www.xilinx.com> : Xilinx Inc
6. <http://www.intel.com> : Intel Corporation
7. <http://www.com.enst.fr/recherche/num/navi0165.html> : FPGA Implementation Survey
8. http://www.ece.utexas.edu/~bevans/hp-dsp-seminar/04_C6xImage/sld001.htm : DSP Implementation Survey

IEEE / IEE documents on the internet :

(referred documents have been detailed earlier under the title of each implementation)

1. <http://www.ieee.org> : IEEE Xplore, available through UW-Madison Library System
2. <http://www.iee.org> : IEE Online Journals Database, available through UW-Madison Library System

Books :

1. Discrete Cosine Transform – Algorithms, Advantages, Applications : K.R.Rao and P.Yip