Energy-Efficient Hardware Architecture for Variable N-point 1D DCT

Andrew Kinane, Valentin Muresan, Noel O'Connor,
Noel Murphy, and Sean Marlow

Centre for Digital Video Processing,
Dublin City University,
Glasnevin, Dublin 9,
IRELAND
kinanea@eeng.dcu.ie
http://www.cdvp.dcu.ie

Abstract. This paper proposes an energy-efficient hardware acceleration architecture for the variable N-point 1D Discrete Cosine Transform (DCT) that can be leveraged if implementing MPEG-4's Shape Adaptive DCT (SA-DCT) tool. The SA-DCT algorithm was originally formulated in response to the MPEG-4 requirement for object based texture coding, and is one of the most computationally demanding blocks in an MPEG-4 video codec. Therefore energy-efficient implementations are important - especially on battery powered wireless platforms. This N-point 1D DCT architecture employs a re-configurable distributed arithmetic data path and clock gating to reduce power consumption.

1 Introduction

Natural video scenes consist of a stationary background and moving foreground objects that are of arbitrary shape. When encoding texture, a video codec system divides each rectangular video frame into an array of non-overlapping 8x8 texture pixel blocks and processes these sequentially. In previously standardized video coding schemes (e.g. MPEG-1, MPEG-2) the 8x8 Discrete Cosine Transform (DCT) processes all blocks, regardless whether they belong to the background or to a foreground object. The DCT is used because it transforms video data into a format more amenable to efficient compression for transmission or storage. MPEG-4 uses the Shape Adaptive DCT (SA-DCT [1]) to support object-based texture encoding, which in turn allows object manipulation as well as giving improved compression efficiency.

In MPEG-4, the object shape description of a frame is termed the alpha-plane or video object plane (VOP). The alpha-plane of a video object can be provided by (semi-) automatic segmentation of the video sequence. This technique is not covered by the MPEG-4 standardization process and depends on the application. The 8x8 alpha block corresponding to a particular texture block defines which pixels in the texture block are part of the video object (VO). For blocks that are located entirely inside the VOP, the SA-DCT behaves identically

to the 8x8 DCT. Blocks that are located entirely outside the VOP are skipped to save needless processing. Blocks that lie on the VOP boundary are encoded depending on their shape and only the opaque pixels within the boundary blocks are actually coded.

This paper addresses the problem of accelerating the variable N-point 1D DCT function required for the SA-DCT with power efficient hardware. A survey of current state of the art implementations of the DCT and SA-DCT is given in [2]. The SA-DCT is less regular compared to the 8x8 block-based DCT since its processing decisions are entirely dependent on the shape information associated with each individual texture block. The 8x8 DCT requires 16 1D 8-point DCT computations if implemented using the column-row approach [1]. Each 1D transformation has a fixed length of 8, with fixed basis functions. This simplifies hardware implementations since the data path is fixed and all parameters are constant. Depending on the shape, the SA-DCT requires up to 16 1D N-point DCT computations where N = 2,3,. . . ,8 (N = 0,1 are trivial cases). Fig. 1 shows an example of how N can vary across a boundary block, where N is defined as the length of VOP pixels in a particular row or column.

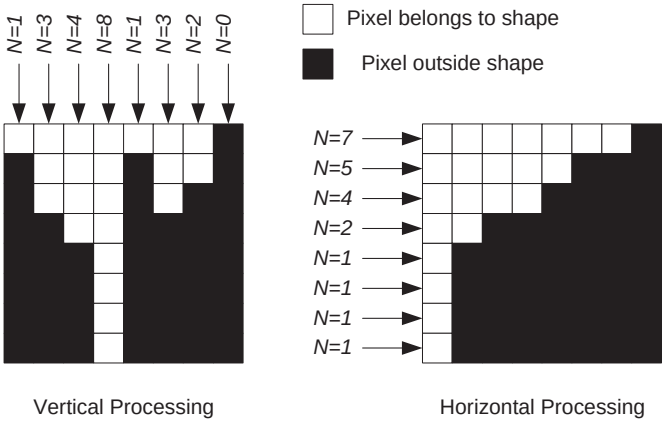

Fig. 1. Example of boundary block requiring 15 1D DCT processes with various data vector lengths N.

In this case the basis functions vary with N, complicating hardware implementation. However, the variable nature of the N-point DCT load affords the possibility of dynamically clock gating logic that is independent of the shape information.

[1] Each column of the input 8x8 block is processed with an 8-point DCT and results are stored in an intermediate memory. Then the 8 rows of the memory undergo an 8-point DCT giving the final output.

The outline of this paper is as follows. Section 2 discusses the general low power design philosophy adopted and how it relates to the particular properties of the N-point 1D DCT function. Section 3 describes in more detail the energy-efficient architecture that implements the N-point 1D DCT, and the final sections discuss future work and offer some concluding remarks.

2 Low Power Design Methodology

2.1 Top Down Approach

In general, power consumption in a CMOS circuit has two components - dynamic power and static power. Traditionally the dynamic component was by far the most dominant, but this is changing due to physical phenomena as process technologies shrink below 90nm. It is estimated [3] that when devices are scaled to 45nm (around the year 2007) the static component will be equal in proportion to the dynamic component. Static power is usually tackled using a variety of dynamic power management (DPM) techniques. If a sub-system is not required to process data at all, it may be shutdown in order to avoid unnecessary power consumption. If the sub-system is still required but has a reduced load, a DPM controller can scale the operating frequency and voltage to save power based on the dynamic processing load. Video processing is very non-uniform by nature so there is scope to apply DPM techniques to video processing architectures for energy conservation purposes. Dynamic power consumption is caused by circuit node switching activity. To reduce this component, a system designer primarily attempts to reduce the complexity of the required processing on the premise that if there are less operations to be carried out, there will be less switching and hence less energy dissipation. It is generally accepted that most power savings are achieved at the higher levels of abstraction (algorithmic and architectural levels) since there are wider degrees of design freedom [4, 5]. Applying low power techniques at the logic, circuit and physical levels are also important but in general depend on the technology available with which the optimized architecture is to be implemented. Therefore initial work on the N-point 1D DCT has focused on optimizing the number of basic operations involved at a generic architectural/algorithmic level (additions/subtractions, multiplications, shifts, processing stages etc.) while maintaining the required accuracy of the output generated.

2.2 N-point 1D DCT Properties

The N-point 1D DCT is described by equation 1, which is essentially N dot products between an N-point data vector **f** and an NxN matrix of N cosine basis functions **A**. Each dot product produces a single DCT coefficient *F(u)*. In this case, power savings are achieved by optimizing the number of multiplications and additions necessary while preserving the acceptable quality of the DCT coefficients produced as defined by the MPEG-4 standard [6]. Hardware

multipliers are inherently more complex and power consumptive compared to adders so greatest effort is focused on minimizing these.

$$\mathbf{F} = \mathbf{Af} \tag{1}$$

where

$$F(u) = \sqrt{\frac{2}{N}} C(u) \sum_{x=0}^{N-1} f(x) \cos\left[\frac{(2x+1)u\pi}{2N}\right]$$

$$= \sum_{x=0}^{N-1} A(x) * f(x)$$

$$C(u) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{for } u = 0 \\ 1, & \text{for } u \neq 0 \end{cases}$$

$$u = 0, \ldots, N-1$$

3 Energy Efficient Architecture

3.1 Re-configurable Distributed Arithmetic Data Path

One commonly employed technique for implementing a dot product of a variable data vector with a vector of constants is Distributed Arithmetic (DA), especially if energy efficiency is paramount to the implementation [7]. In short DA is an efficient architecture for computing a dot product by transforming it to a series of additions, look-ups and shift operations. DA distributes the bits of the input variable vector *f(x)* and uses an aggregation of these bits for each bit position to form weights which are linear additive combinations of a constant basis vector *A(x)*. These weights are then scale accumulated together to produce a final coefficient *F(u)* achieved without the need for any multiplications. Multiplications are inherently more complicated and power consumptive compared to additions (greater area, possibly more clock cycles required, more switching) so eliminating them will certainly result in a more energy-efficient architecture. Since the 1D N-point DCT is itself a series of dot products, DA seems a logical architectural choice.

A variation on DA is New Distributed Arithmetic (NEDA) [8] that distributes the bits of the constant basis vector *A(x)* (as opposed to the data vector) and forms weights that are linear additive combinations of input data vector *f(x)*. The implication of this alternative approach is that no ROM look-ups are required. This is important for a variable N-point 1D DCT implementation since the values for each NxN matrix where N = 2,3,...,8 would require storage with conventional DA.

The architecture proposed in this paper leverages the NEDA properties, however, the data path taken to form the weights depends on the value of N. Essentially this implies that the addends from the data vector *f(x)* used to form the weights are multiplexed by the value of N. The number of hardware adders required for the NEDA tree is optimized by using the greedy algorithm described in [9].

A conceptual architecture for the 1D N-point 1D DCT is illustrated in Fig. 2. Presented at the input ports are the data vector *f(x)* and the value of N for this vector, where N has been previously decoded from the corresponding alpha block. The first stage involves combining the input data vector using 21 two input additions and the results are stored in the first register stage regardless of the value of N. Next, for each of the N coefficients 13 weights are computed using a linear additive combination of a subset of the 23 primary addition values and the primary inputs themselves. The combination taken depends on the value of N. Each coefficient *F(u)* is computed differently depending on the value of N, and N is used to multiplex the addends in the weight generation addition stage. The number of weights for each coefficient was chosen to be 13. Experimentation has shown that this is the lowest number possible to comply with the standard in terms of performance [10]. The potential to vary the number of weights used to save power is discussed in the next sub-section. Once these 13 weights for each of the N coefficients have been computed and stored in the second register stage, these weights are then combined in a scaled manner according to the NEDA architecture using a carry save adder tree. The final outputs of each of the carry save adder trees represent the final N coefficient values that can be stored in output registers.

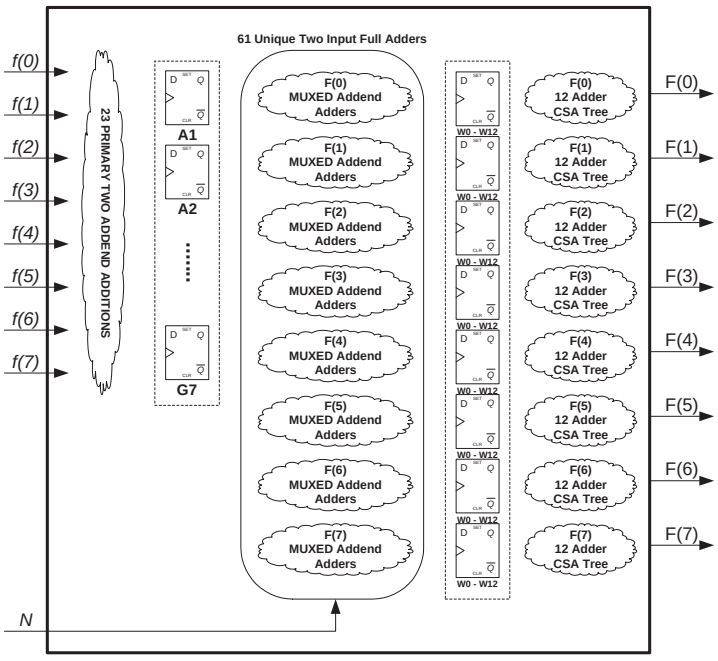

Fig. 2. A conceptual NEDA architecture implementing the 1D N-point 1D DCT, where the value of N is used to multiplex the data path.

3.2 Trading Power for Performance

As mentioned in the previous sub-section, 13 weights are required for each coefficient computation (N in total) for strict compliance with the MPEG-4 standard performance requirements [6]. The DCT coefficients are said to be compliant if, when reconstructed to pixels using an inverse transform (IDCT), they are within two grey levels of the same data that has undergone a double precision DCT/IDCT process. However, a user may tolerate some perceptual visual degradation if it means a longer battery life. By reducing the number of weights used, the numbers of adders used to compute the coefficients are reduced as a result thus saving power. Unfortunately, reducing the number of weights affects the accuracy of the coefficients, and it is clear that this represents a power versus performance trade-off. The number of adders required for all values of N using different amount of weights Q to generate DCT coefficients with the proposed architecture are listed in Table 1. For a particular Q value, the total number of unique adders required to implement a variable N-point 1D DCT is an intersection of the adder requirements for each individual N value. The optimal adder requirements are obtained using a combinatorial optimization technique [9].

Table 1. Number of adders necessary for each value of N for various numbers of NEDA weights Q

N→	8	7	6	5	4	3	2	1	Total	Saving
Q=13	47	53	28	22	18	14	13	0	**180**	n/a
Q=12	46	52	27	21	17	13	12	0	**172**	4.44%
Q=11	45	49	26	20	16	12	11	0	**163**	9.44%
Q=10	42	47	25	19	15	11	10	0	**151**	16.11%

Experimentation has been carried out using the MPEG-4 video test sequences using various values of Q for the proposed architecture and the results are presented in Table 2.

Table 2. Performance experimentation results for various values of Q

	Akiyo			**Container**			**Coastguard**		
Q	PSNR **[dB]**	Error [%]	Max Pel Error	PSNR **[dB]**	Error [%]	Max Pel Error	PSNR **[dB]**	Error [%]	Max Pel Error
13	55.844	0	0	56.095	0	0	55.8	0	0
12	53.519	0.002	3	55.694	0.000	3	53.575	0.005	3
11	49.684	0.872	5	52.953	0.083	4	50.047	0.681	5
10	44.562	10.081	8	44.529	6.586	9	45.109	8.376	8

These results illustrate that using 13 weights (Q = 13), there are no errors and the architecture is fully compliant with the MPEG-4 standard. However, as Q reduces, the pixel errors become more significant as expected but less power is consumed since there is less processing. However, the results show that the pixels do not deviate wildly from the required accuracy. For example, with the "container" test sequence with Q = 10, 6.586% of the VOP pixels reconstructed violate the standard but the maximum error is only a difference of 9 grey-levels which is imperceptible to the human eye. A user may tolerate such degradation on a mobile platform if it means a longer battery life.

3.3 Clock Gating Possibilities

As well as the potential to trade precision for power, this architecture has been designed such that it is possible to clock gate redundant logic based on the value of N. The adder logic for each coefficient has been partitioned to allow each one to be clock gated individually. To illustrate, consider the possibility that N = 5 for a particular computation. In this case the only valid coefficients are *F(0)* to *F(4)*. The logic producing coefficients *F(5)*, *F(6)* and *F(7)* is not needed and so can be clock gated for this particular computation. This can be generalized to say that if N is less than 8, coefficients *F(N)* to *F(7)* are irrelevant and can be clock gated. In this way the power consumed is not constant but depends on the shape (and hence the value of N).

3.4 Summary

The power efficient properties of this architecture may be summarized as follows:

- The 1D SA-DCT computation unit is configurable based on the value of the transform length N. This is much more efficient then having a separate computation unit for each value of N.
- The accuracy of the DCT coefficients produced by the architecture can be dynamically adjusted by decreasing the number of adders used in the data path. This can be done to save power if deemed acceptable by the user.
- Within the computation unit unnecessary logic is clock-gated based on the value of N, such that less power is consumed when the computational load is smaller (smaller N).
- The architecture has the general distributed arithmetic property of no multipliers. An m bit multiplication is much more power consumptive compared to an m bit addition, especially if the computation must take place in one clock cycle.
- The architecture has the NEDA property of requiring no power-consuming ROM lookups that are necessary with conventional distributed arithmetic. This is especially important since the SA-DCT has numerous cosine basis function possibilities and would require a relatively large ROM.
- Multiplications have been eliminated, but also the number of adders necessary to produce the weights has been optimized using a combinatorial optimization technique [9].

4 Conclusions and Future Work

This work has enhanced the NEDA architecture for implementing an 8-point 1D DCT [8] to compute a variable length N-point 1D DCT N = 0,1,...,8. This is achieved by implementing a multiplexed coefficient generation data path based on the value of N. In the future, it is intended to leverage this 1D N-point DCT processing element to implement a full energy efficient SA-DCT hardware core. Part 9 of the MPEG-4 standard defines hardware architectures for the most computationally demanding tools in the standard and currently there is no SA-DCT implementation, only a conventional 8x8 DCT [11]. The SA-DCT is required to realize the object-based processing capabilities of the MPEG-4 standard, and it intended that this work would fill the void. The intended design flow is as follows:

- SystemC RTL description of SA-DCT core using Microsoft Visual C++.
- Synthesis check and translation to Verilog using Synopsys SystemC Compiler.
- SystemC/Verilog co-simulation with Synopsys VCS to verify translation.

At this point the design flow diverges into two separate paths. The first path involves targeting an ARM processor prototyping platform (Integrator/CP with Xilinx FPGA) and integration with other DCU MPEG-4 hardware acceleration architectures. This process involves:

- Synthesis of Verilog/VHDL code using Synplicity Pro targeting Xilinx VirtexE XCV2000E FPGA technology.
- Place and route to ARM Integrator/CP platform using Xilinx ISE.
- Integration with other DCU hardware accelerators.

The second design flow involves targeting the Annapolis WildCard-II PCMCIA card to benchmark against other architectures proposed to the MPEG-4 reference hardware forum:

- Synthesis of Verilog/VHDL code using Synplicity Pro targeting Xilinx XC2V3000-4-FG676 FPGA technology (integrated on Annapolis WildCard-II architecture).
- Place and route to Annapolis WildCard-II PCMCIA card using Xilinx ISE.
- Performance evaluation using WildCard-II card features. The WildCard-II allows board level current, voltage and power to be measured dynamically.

It is envisaged that the eventual SA-DCT hardware core developed will be an energy-efficient hardware implementation of the SA-DCT function. This solution could then be integrated onto a wireless platform supporting MPEG-4 where the battery life is limited.

Acknowledgements

The support of the Research Innovation Fund of Enterprise Ireland is gratefully acknowledged.

References

1. Sikora, T., Makai B., Shape-Adaptive DCT for Generic Coding of Video, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 5, No. 1, February 1995, pp 59–62.
2. Muresan, V., Kinane, A., Larkin, D., Hardware Acceleration Architectures for MPEG-based Mobile Video Platforms: A Brief Overview, Proc. 4th European Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), London, April 2003, pp 456–461.
3. Butts, J.A., et al. A Static Power Model for Architects, 33rd International Symposium on Microarchitecture, December 2000.
4. Ruby, W., (Low) Power To The People, EDAVision Magazine, March 2002.
5. Poppen, F., Low Power Design Guide, Low Power Design and Services Group, OFFIS Research Institute, http://www.lowpower.de [Accessed 16th Sep. 2003].
6. MPEG-4: Information Technology - Coding of Audio Visual Objects - Part 2: Visual, ISO/IEC 14496-2, Amendment 1, July 15th 2000.
7. Xanthopoulos, T., Chandrakasan, A.P., A Low-Power DCT Core Using Adaptive Bitwidth and Arithmetic Activity Exploiting Signal Correlations and Quantization, IEEE Journal of Solid-State Circuits, Vol. 35, No. 5, May 2000, pp 740 - 750.
8. Shams, A., Pan, W., Chidanandan, A., Bayouni, M.A., A Low Power High Performance Distributed DCT Architecture, Proceedings of the IEEE Computer Society Annual Symposium on VLSI 2002 (ISVLSI'02).
9. Potkonjak, M., Srivastava, M.B., Chandrakasan, A.P., Multiple Constant Multiplications: Efficient and Versatile Framework and Algorithms for Exploring Common Subexpression Elimination, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 15, No. 2, February 1996, pp 151 - 165.
10. Kinane, A., Power-Efficient Hardware Accelerator for MPEG-4 Shape Adaptive Discrete Cosine Transform Tool Software Demo, Internal Technical Report, Visual Media Processing Group, DCU, February 2004.
11. Mattavelli, M., Turney, R., Text of ISO/IEC DTR 14496-9 Information technology - Coding of audio visual objects - Part 9: Reference hardware description, ISO/IEC 14496-9, Pattaya, March 2003.