EFFICIENT HARDWARE ARCHITECTURES FOR MPEG-4 CORE PROFILE

Daniel Larkin[*]*, Andrew Kinane, Valentin Muresan, Noel O'Connor*

Centre for Digital Video Processing,
Dublin City University,
{larkind,kinanea,muresanv,oconnorn}@eeng.dcu.ie

ABSTRACT

Efficient hardware acceleration architectures are proposed for the most demanding MPEG-4 core profile algorithms, namely; texture motion estimation (TME), binary motion estimation (BME) and the shape adaptive discrete cosine transform (SA-DCT). The proposed ME designs may also be used for H.264, since both architectures can handle variable block sizes. Both ME architectures employ early termination techniques that reduce latency and save needless memory accesses and power consumption. They also use a pixel subsampling technique to facilitate parallelism, while balancing the computational load. The BME datapath also saves operations by using Run Length Coded (RLC) pixel addressing. The SA-DCT module has a re-configuring multiplier-less serial datapath using adders and multiplexers only to improve area and power. The SA-DCT packing steps are done using a minimal switching addressing scheme with guarded evaluation. All three modules have been synthesised targeting the WildCard-II FPGA benchmarking platform adopted by the MPEG-4 Part9 reference hardware group.

1. INTRODUCTION

The ongoing global trend to shift multimedia applications from desktop platforms (e.g. set-top boxes or PCs), to mobile platforms (e.g. PDAs, smart-phones, or game boxes) has encountered several technical hurdles: very demanding real-time applications, low bandwidth mobile networks, and mobile devices hardware limitations. The latter include low computational power, low memory capacity, short battery life and strict miniaturisation requirements. The solution to the conflict between short battery life and power-hungry deep sub-micron technologies does not necessarily lie in high-throughput fully-parallel architectures (e.g. systolic arrays). Rather, power efficient hardware acceleration is being investigated either within the behavioural domain by exploiting the nature of the media processing operations to be accelerated (e.g. regularity, redundancy) or in the the structural domain by employing dynamic power management (DPM) techniques. While the latter is a more straightforward technique, the former implies innovation at the algorithmic level and has the potential for much greater (10X to 20X) power savings [1].

We propose three video processing architectures that offer an appropriate trade off between area, speed and power consumption. The target tools are texture motion estimation (TME), binary motion estimation (BME) and the shape adaptive discrete cosine transform (SA-DCT). The modules have been synthesised targeting the Annapolis WildCard-II FPGA prototyping platform (PCM-CIA card with a Xilinx Virtex-II XCV3000 FPGA) adopted by the MPEG-4 Part9 reference hardware group. Benchmarking figures are presented for this platform and full MPEG conformance testing for each of the modules is currently being undertaken.

The rest of this paper is organized as follows. A brief overview of ME & BME architectures is given in section 2. The general ME architecture is presented in 2.1. The optimized PE for TME & BME are presented in section 2.2 and 2.3. Section 3 discusses a Shape Adaptive DCT architecture. Section 4 discusses MPEG-4 Part9 Integration, while section 5 draws conclusions on the work presented.

2. MOTION ESTIMATION (ME)

The block-matching (BM) based approach of the ME algorithm has been found to be the most amenable to hardware implementation offering the best performance/complexity tradeoff. Systolic arrays (SA) are amongst the first architectural solutions proposed for BM-based ME. They were designed to maximally exploit BM operations' regularity usually in a full search (FS) strategy. This eliminated the significant control circuitry overhead [2]. However, they sacrifice area and power to achieve high throughput. SA implementations can be classified as 1-D or 2-D approaches, with global or local accumulation [3]. Clock rate, frame size, search range, and block size are the parameters used to decide on the number of processing elements (PEs) in the systolic structure [2]. Recently, some ME optimization approaches have been proposed to tackle memory efficiency. This is achieved by a high degree of on-chip memory content re-use, parallel pel information access and memory access interleaving [4].

2.1. PROPOSED GENERAL ME ARCHITECTURE

The proposed architecture comprises a low-cost power-efficient fast exhaustive BM architecture for ME. ME's high computational requirements are addressed here by implementing in hardware an early exit mechanism based on SAD cancellation. This achieves redundant computation elimination by adapting the datapath resources to the video input characteristics (i.e. motion level, object size, object texture, etc.). Due to the fact that this approach is based on re-mapping and partitioning the video content by means of pixel subsampling (see fig. 1), only architectures with a 2^{2*n} number of PEs can be implemented. For conciseness, only a 4xPE architectural solution is discussed here. As it can be seen in fig. 2, the general 4xPE ME architecture proposed here consists of 4 parallel PEs and an update stage. While the PEs for texture ME (TME) are based on de-accumulation operations (see subsection 2.2), the PEs for binary ME (BME) are based on XOR-based bit

[*]The support of Enterprise Ireland is gratefully acknowledged

counting operations between run length coded blocks (see subsection 2.3).

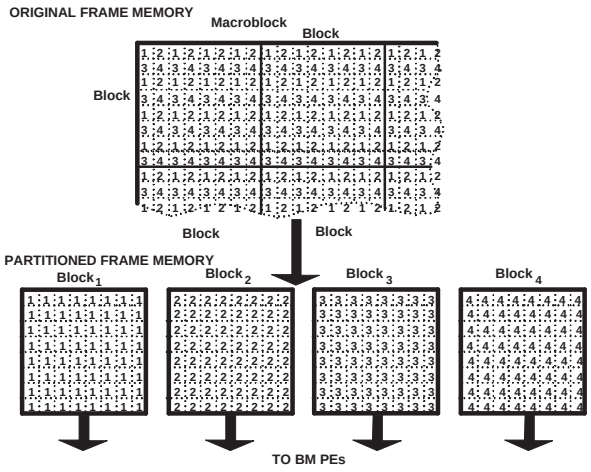

Fig. 1. Video Data Re-mapping and Partitioning

2.1.1. Update Stage

The early exit mechanism embedded in all PEs of the proposed general ME architecture is based on SAD cancellation. This approach has been proposed in software, but has never been leveraged in a hardware implementation so far. It is basically an AND function between the early exit condition to be met in all PEs during the match at any time. For the undecided cases, when the early exit condition was not met in all PEs by the end of the PE-level SAD calculations, the update stage (US) is turned on to execute in parallel with the next match's operations (executed in PEs) the verification of the current total (macroblock) SAD. The US takes at most 11 cycles while the PE level BM takes 64 cycles (8x8). Therefore, a pure sequential scheduling of the US operations is implemented in the US hardware (see fig. 2) to run in parallel with the PE operations. There are three possible US operation scenarios in a block match. The most frequent one is when the US is idle all the time during matches that are early exited. The second and third one happen when the US is launched because the early exit condition has not been met in all PEs by the end of the match. Thus in the second scenario the US is cancelled after 5 steps because the total SAD value turns out to be worse than the current minimum SAD and no PE-level SAD update is deemed necessary. The third scenario happens after the aforementioned 5 steps when the total calculated SAD turns out to be a new minimum. In this case the total min SAD and PE-level min SAD values are updated. Hence, corrections are also carried out within a single cycle inside the PEs' de-accumulators that have started the new match less than 11 cycle ago from previous PE-level min SAD values. Note that if a new SAD-cancellation occurs during an US run and if a new match has to be skipped, this early exit does not affect the US's operations. This is due to the fact that a match skip means that the resulting total SAD value was getting larger than the current minimum SAD (that can only be altered by a smaller total SAD value).

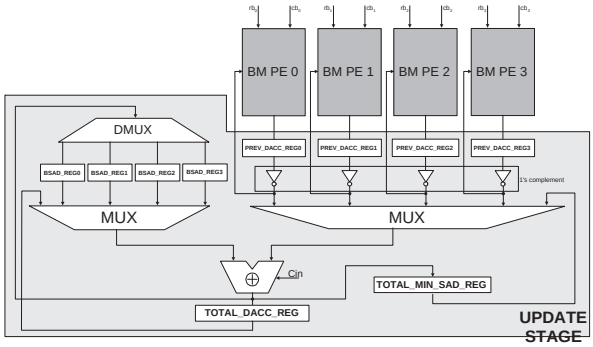

Fig. 2. 4xPE Architecture = 4BM PEs + Update Stage

2.2. Texture Motion Estimation

To date, SAD-cancellation has been employed in software implementations only where the current block match was cancelled when it was obvious that the current SAD was greater than the minimum SAD found so far. In order to implement this approach in parallel hardware, a SAD-cancellation mechanism is needed to encompass both the block (PE) and macroblock (US) levels. The solution proposed is to employ block-level parallelism in the SAD formula and then transform the equation from calculating an absolute-value to calculating a relative-value (by de-accumulation) to the current minimum SAD. That is, a 16x16 MB SAD calculation is carried out in four 8x8 BMs relatively to the current minimum SAD and then updated at PE&US level if better. A detailed description of the texture PE may be found in [5].

2.3. Binary Motion Estimation

BME finds numerous applications within video coding. It is an integral element of MPEG-4 Binary Shape Coding, where it typically consumes over 90% of the overall resources required for shape encoding. BME has also been proposed [6] as a way to reduce complexity for regular 8-bit Texture Motion estimation when coupled with a suitable binarization filter. There are a number of hardware implementations of BME. In the context of BME for shape coding Chang et al [7] proposes a 1D systolic array hardware architecture implementation coupled with a full search strategy. It uses 16 SAD PE's, where each PE calculates the SAD for one candidate BAB search position. The PE architecture consists of an XOR and Adder tree structure.

2.3.1. Optimized BME Architecture

In general, with binary valued pixels the SAD calculation reduces to the following Bit Counting (BC) operation:

$$BC\left(B_{curr}, B_{ref}\right) = \sum_{i=1}^{i=N} \sum_{j=1}^{j=M} B_{curr}\left(i,j\right) \otimes B_{ref}\left(i,j\right) \quad (1)$$

Where B_{curr} is the block under consideration in the current frame and B_{ref} is the block at the current search location in the search window. In previous BME research no attempts have been made to optimize the SAD PE datapath, despite the fact there is a number of inherent redundancies which can be exploited. Similar to the approach adopted in section 2.2, BC early termination can be employed to cease processing when the minimum BC has been

exceeded. Furthermore equation (1) gives a zero intermediate result when both $B_{curr(i,j)}$ and $B_{ref(i,j)}$ have the same value. This results in unnecessary memory accesses and operations. To minimise this, we propose using run length encoding (RLE), thereby accessing only relevant data. In order to use RLE the BC calculation must be reformulated to equation (2) (a full discussion of this is presented in [8]).

$$BC = TOT_{ref} - TOT_{curr} + 2 \times \sum DIFF_{curr_{rle}} \quad (2)$$

Equation (2) is beneficial from a low power hardware perspective since

- TOT_{curr} is calculated only once per search.
- TOT_{ref} can be updated in 1 clock cycle.
- Incremental addition of DIFFcurr allows early termination.
- Irrelevant data is not accessed.

The run length codes are generated in parallel during the first match when early termination is not possible. After the encoding the logic can be powered down until the next current block is processed. When there are fewer black pixels than white pixels in the current MB it is preferable to use the inverse run length codes as this further exploits redundancies. The BC operation is reformulated [8] to that of equation (3). This is also amendable for hardware implementation as it requires minimal extra logic.

$$SAD = TOT_{curr} - TOT_{ref} + 2 \times \sum DIFF_{curr_{invRLE}} \quad (3)$$

A detailed view of the BC Processing Element is shown in Fig. 3 and discussed in [8]. It can been seen that there are two early termination mechanisms and associated control logic.

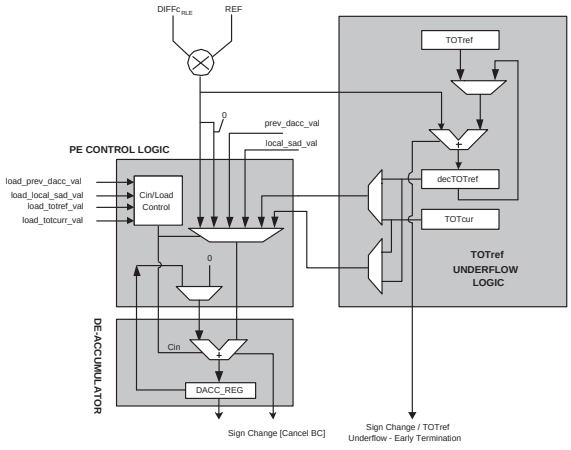

Fig. 3. RL SAD PE

3. SHAPE ADAPTIVE DISCRETE COSINE TRANSFORM

An efficient architecture (Fig. 4) has been designed to compute the SA-DCT, which offers a good trade-off between speed, area and power [9]. All k coefficients $k = 0, 1, \ldots, N-1$ are computed serially for each N-point 1D DCT by reconfiguring the datapath based on $\{k, N\}$. This is achieved using even/odd decomposition (EOD), followed by adder-based distributed arithmetic using a multiplexed weight generation module (MWGM) and a partial

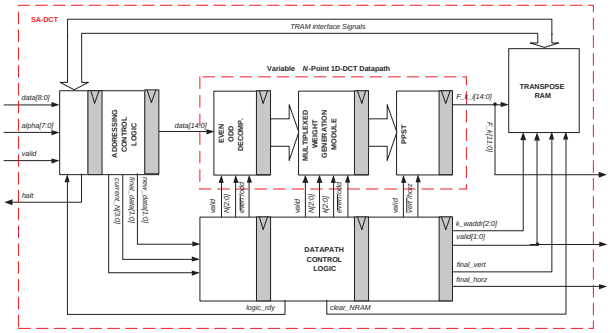

Fig. 4. SA-DCT Architecture.

product summation tree (PPST). The input buffer and TRAM addressing logic perform the SA-DCT packing stages without the need for shift registers. The properties of the architecture include:

- Reconfiguring adder-only based distributed arithmetic datapath to eliminate multipliers and exploit adder re-use.
- Efficient data addressing using k and N to perform SA-DCT packing with minimal switching.
- Local module clock gating based on the k and N to avoid wasted power.

The dynamic nature of the SA-DCT processing steps pose significant VLSI implementation challenges, but this architecture exploits this by computing the coefficients serially and re-allocating resources. The total adder count is 27, which compares favourably with other SA-DCT specific approaches [10].

4. SYNTHESIS AND MPEG-4 PART9 INTEGRATION

4.1. Synthesis Results

The ME, BME and SA-DCT cores have been synthesised in isolation targeting the Xilinx Virtex-II FPGA on the Annapolis WildCard-II platform, with the results shown in Table 1. These results may be used to accurately benchmark against other implementations submitted to the MPEG-4 Part9 group since the target platform must be adhered to when submitting modules. It must be noted that the results presented in Table 1 are not the absolute optimal achievable by design since the target technology is an FPGA. For example with 90nm ASIC technology, the SA-DCT achieves an operating frequency at 250MHz consuming 0.468mW [10].

4.2. Hardware / Software Integration

The SA-DCT module has been integrated with an adapted version of the multiple IP-core hardware accelerated software system framework developed by the University of Calgary [11]. The entire system along with host software calls has been implemented on a Windows 2000 laptop with the Annapolis WildCard-II PCMCIA FPGA prototyping platform installed.

An efficient hardware module controller (HWMC) for the SA-DCT core has been developed to interface between the core and the rest of the system. Its processing steps are summarised as follows:

1. Waits until activated by host software via interrupt controller strobe. The host software also writes to 4 master socket hardware registers to configure the SA-DCT core with the following parameters:

Module	Area (Logic) [Gates]	Area (Local Memory) [Gates]	Maximium Frequency [MHz]	Power [mW]	Throughput [kB/s]	Multipliers	CLB Slices	Block RAMs
SA-DCT	26642	13330	62.9	70-80	256684	0	2630 (18%)	0
ME	10800	0	99	31.40	25-140	0	395 (1%)	0
BME	6913	16854	76.8	12.95	100-200	0	433 (3%)	0

Table 1. Xilinx Virtex-II XC2V3000 Synthesis Results

- Frame dimensions.
- Number of burst frames to be processed.
- SRAM read start address (texture and shape).
- SRAM write start address (SA-DCT coefficients).

2. Issues SRAM access request to memory arbiter.
3. When request granted, reads a set of 8x8 texture and alpha blocks from the SRAM and transmits data to SA-DCT core.
4. Buffers resultant data from core and writes to SRAM.
5. Checks to see if all burst frames processed. If no more data, go to step 1 and relinquish SRAM access request. Otherwise repeat steps 3-5 until all data processed.

The SA-DCT has a variable load depending on the shape information and the SA-DCT core has been designed such that it finishes processing smaller video object plane (VOP) blocks earlier. The SA-DCT HWMC has been designed with this in mind so that the SA-DCT relinquishes control of the SRAM at the earliest possible moment. At present HWMCs are being designed for the ME and BME modules similar to that implemented for the SA-DCT. They will have interleaved fast local memories that will store the search windows as addressed by the ME and BME cores. This is done to avoid power-consumptive off-chip memory accesses and facilitate date re-use, which is particularly important for the ME/BME modules since they address a lot of data. When complete all three modules will be integrated together with appropriate SRAM arbitration logic. It is envisaged that a shared local memory could be used for all three modules to avoid redundant off-chip accesses.

A software API has been developed that configures the system and simulates the SA-DCT, quantisation, inverse quantisation and ISA-DCT steps of an MPEG-4 codec. The hardware module computes the SA-DCT and the rest is computed in software on the host computer to verify the core functionality. Full MPEG-4 part 9 conformance testing requires the compilation of the entire reference software replacing the reference SA-DCT function with an API call to the SA-DCT hardware accelerator. Future steps involve developing a similar API for the ME and BME modules and full conformance testing for all three modules with the MPEG group.

5. CONCLUSIONS

Efficient hardware acceleration architectures have been presented for TME, BME and the SA-DCT video processing algorithms. All three have been synthesised targeting the MPEG-4 Part9 reference hardware integration framework with results outlined in section 4. MPEG-4 core profile implementations are hampered by the lack of robust video object segmentation systems. However, there is significant ongoing research in the field of video feature extraction (MPEG-7), which may be leveraged in the future for crude semantic object segmentation.

6. REFERENCES

[1] "Minimize IC power without sacrificing performance," July 15 2004, [Online] http://www.eedesign.com/.

[2] S.C. Cheng et al., "Algorithms mapped to systolic-array implementation," *IEEE Trans. Consumer Electron.*, vol. 39, no. 3, pp. 292 – 297, Aug. 1993.

[3] E. Chan et al., "Motion estimation architecture for video compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 5, pp. 741 – 757, Oct. 1997.

[4] Y.K. Lai et al., "A data-interlacing architecture with two-dimensional data-reuse for full-search block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 2, pp. 124 – 127, Apr. 1998.

[5] Valentin Muresan and Noel O'Connor, "Updated Status and documentation of the 4xPE Hardware Acceleration Module for Motion Estimation, ISO/IEC JTC1/SC29/WG11 N6757," in *70th MPEG-4 Meeting Document Register*, Palma de Mallorca, Spain, Oct. 18-22, 2004.

[6] B.Natarajan, V. Bhaskaran, and K. Konstantinides, "Low complexity block based motion estimation via one bit transform," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 4, pp. 702–706, August 1997.

[7] Hao-Chieh Chang, Yung-Chi Chang, Yi-Chu Wang, Wei-Ming Chao, and Liang-Gee Chen, "VLSI Architecture Design of MPEG-4 Shape Coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 9, Sept. 2002.

[8] Daniel Larkin, Valentin Muresan, and Noel O'Connor, "Updated Status and documentation of the Shape Coding Binary Motion Estimation Hardware Acceleration Module," in *N6759 Contribution to AHG on MPEG-4 Part-9: Reference Hardware ISO/IEC JTC1/SC29/WG11*, Palma de Mallorca, Spain, Oct. 18-22, 2004.

[9] Andrew Kinane, Valentin Muresan, and Noel O'Connor, "Updated Status and Documentation on Hardware Acceleration Module for SA-DCT for MPEG-4 Part 2, ISO/IEC JTC1/SC29/WG11 N6756," in *70th MPEG-4 Meeting Document Register*, Palma de Mallorca, Spain, Oct. 18-22, 2004.

[10] Andrew Kinane, Valentin Muresan, and Noel O'Connor, "An Optimal Adder-Based Hardware Architecture for the DCT/SA-DCT," in *Proc. SPIE Video Communications and Image Processing (VCIP)*, July12–15.

[11] Tamer S. Mohamed and Wael Bedawy, "Multiple IP-Core Hardware Accelerated Software System Framework for MPEG-4 Part 9," in *ISO/IEC JTC1/SC29/WG11 M10954 Contribution to AHG on MPEG-4 Part 9: Reference Hardware*, Redmond, USA, July 2004.