WHAT CHOICES MAKE A KILLER VIDEO PROCESSOR ARCHITECTURE?

Jonah Probell
Ultra Data Corp.
14 Chestnut Street
Waltham, MA 02453, USA
781-209-0886
jonah@ultradatacorp.com

ABSTRACT

Many trade-offs exist in the design of digital signal processor architectures. Certain choices for those trade-offs lend themselves particularly well to the characteristics of video processing.

This paper discusses costs and benefits of design decisions for video processor architectures. The Ultra Data UD3000 is used as an example. The UD3000 is a video processor optimized for decoding advanced video standard bitstreams, such as H.264, for video sequences at up to HDTV display resolution and frame rate.

Trade-offs of software programmability versus fixed-functionality, different types of parallelism, on-chip memory versus off-chip memory bandwidth, processor performance versus area, and data access determinism versus software overhead are discussed.

1. INTRODUCTION

The UD3000 is a multiprocessor optimized for video decode applications and comprising five processor cores, a DMA controller, five dual-port local data memories, and other supporting devices. Of the five processor cores two are identical 32-bit RISC outer loop processors (OLP) executing an industry standard instruction set with configurable instruction and data caches. The other three processor cores are identical VLIW inner loop processors (ILP) comprising two execution units, a 32-bit RISC control unit (CU) and a 64-bit SIMD DSP vector unit (VU) with a five-port register file. The CU and VU each execute a proprietary instruction set and the ILPs have no caches, but a local instruction memory and single cycle access to the five local memories in the UD3000.

The UD3000 is fully software programmable. The ILP instruction set is optimized to include the operations common to decoding widely used video standards. No aspects of the hardware are specific to a particular video coding standard.

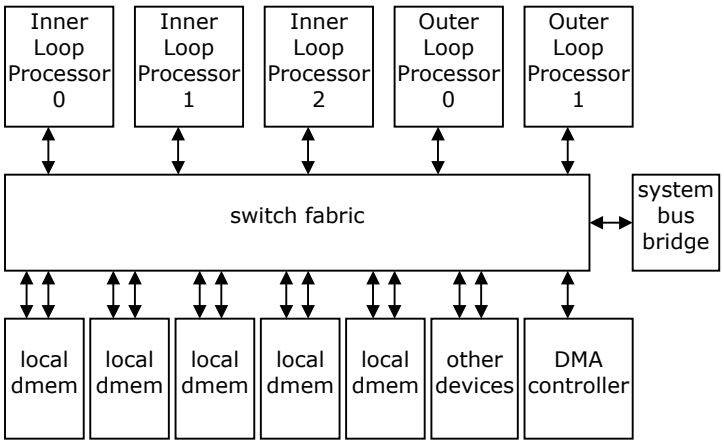

Figure 1: UD3000 block diagram.

The steps carried out to convert a video bitstream into a sequence of video frames are performed with a pipelined flow on the five processors. Each processor performs a unique function on each block of video data.

The UD3000 is capable of decoding H.264 video sequences at HDTV resolution and frame rate when the processor runs at 400 MHz. It is ideal for low cost HD

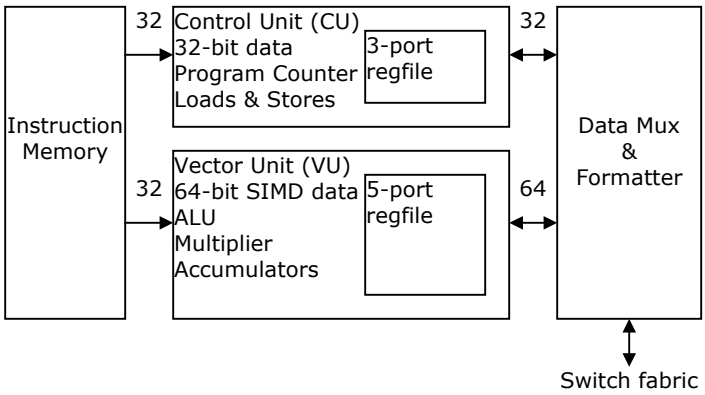

Figure 2: Inner Loop Processor block diagram.

DVD players, HD transcoding personal video recorders, video on demand cable and satellite set-top boxes, internet-enabled televisions, and 3G videophones.

The following sections discuss the tradeoffs made in the design of the UD3000. Section 2 is about the programmable versus the fixed-function approach. Section 3 is about the amount of instruction and thread level parallelism. Section 4 is about local memory and DMA controller complexity versus off-chip memory bandwidth. Section 5 is about techniques to avoid performance-degrading stalls.

2. PROGRAMMABILITY

Most video decoder chips today support MPEG-2 bitstreams with a highly optimized fixed-function hardware decoder core. Such cores are also available for MPEG-4, H.264, and Windows Media Video 9 bitstream decode. An ASIC for an application that uses a single mature video standard will benefit from a hardware decoder core because such a core can be optimized to process bitstreams with minimal on-chip logic and memory hardware.

Video decoding can also be performed by a software decoder program running on a programmable processor. A programmable processor will consume more silicon area than a hardware decoder core because it must have on-chip memory for a local instruction store, a general-purpose register file, and a general-purpose multiplier and ALU. An Athlon 800 is capable of H.264 decode at CIF (352x288) resolution and an Athlon 2 200 XP is capable of H.264 decode at D1 (704x480) resolution [1]. For embedded applications, digital signal processors, such as the TI TMS320DM342DSP are capable of H.264, but only at a fraction of D1 resolution [2]. TI's high end 600 MHz DM642 is only capable of H.264 at D1 resolution [3].

The UD3000 is a fully software programmable multiprocessor video DSP core capable of decoding H.264 at HDTV (1920x1080) resolution. A UD3000 video processor core is capable of executing any industry standard video codec. The processors in the UD3000 are optimized for video tasks in order to reduce silicon overhead. The UD3000 requires less than twice the silicon of two fixed-function video decoder cores with the same level of performance.

3. PARALLELISM

Processing video requires a high amount of arithmetic operations relative to data movement operations when compared to other types of processing. Matrix multiplication, in particular, is fundamental to video processing. It is used for two-dimensional spatial-frequency domain block transforms such as the discrete cosine transform. Many video algorithms also specify multi-tap filtering algorithms for sub-pixel interpolation. These routines, as well as deblocking filtering, deinterlacing, and color space conversion require performing the same operation on multiple pieces of data. Processing time can be saved by exploiting single instruction multiple data (SIMD) techniques.

The video sequence with the worst-case processing requirements has all small blocks. The smallest blocks in video coding are 4x4. 16-bit resolution is needed for most intermediate data values. To accommodate this, the VU executes instructions that operate on eight 8-bit or four 16-bit data elements in parallel within a 64-bit datapath. A wider datapath with greater SIMD parallelism would improve average performance but would not improve the worst-case. Therefore, a datapath wider than 64 bits would be a less efficient use of silicon.

SIMD processing decreases the number of processing instructions relative to data movement (load/store) and address calculation instructions in a program. As a result, many applications leave a SIMD datapath underutilized while performing the data movement operations needed to feed the datapath. That inefficiency is mitigated by a VLIW architecture. A VLIW machine with two execution units is nearly optimal for video DSP algorithms. In the UD3000, the CU manages address calculations and data movement while the SIMD VU remains fully utilized for its data processing tasks. The CU consumes a small amount of die area compared to the VU, but greatly improves performance over an architecture in which a single execution unit performs both data movement and computational instructions.

SIMD and VLIW instruction level parallelism in the inner loop processor yield excellent processing performance for the silicon area consumed, but not enough to achieve HDTV pixel rates in a single processor for the most advanced video algorithms. The UD3000 exploits thread level parallelism in the form of multiprocessing to achieve the required performance.

The decoding for any particular video block can have data dependencies on any part of the frame. As a result, the frame cannot be symmetrically partitioned across a multiprocessing system such as the symmetric

partitioning that is common for network packet processing applications. Instead, the video decoding tasks are each performed as separate threads running on separate processors. The data for each macroblock moves between processors in a pipelined flow as it is transformed from a compressed bitstream to a decoded pixel array.

OLP 0 extracts the prediction mode, motion vectors, and inverse transform coefficients from the bitstream and writes them to a local on-chip data memory (dmem). ILP 0 reads the prediction mode and motion vectors, performs the prediction, and writes the predicted block to a dmem. ILP 1 reads the inverse transform coefficients and the predicted block, performs the inverse transform, applies the differences to the prediction, and writes the reconstructed block to a dmem. OLP 1 reads the reconstructed block and neighboring blocks, computes deblocking filter parameters, and writes those to a dmem. ILP 2 reads the deblocking filter parameters and the reconstructed block edges, applies the filter, and writes the result to another dmem. The DMA controller reads the filtered block from dmem and writes it to the frame buffer in off-chip DRAM memory.

4. MEMORY BANDWIDTH

The process of converting a compressed bitstream into a series of decoded frames requires several data transformations and the temporary storage of intermediate data structures. These include prediction modes, motion vectors, reference frame buffers, predicted blocks, inverse transform coefficient blocks, inverse transformed prediction delta blocks, reconstructed blocks, neighboring rows and columns, and filter parameters.

To decode H.264 at Main profile, the only intermediate data structures too large to fit in on-chip memory are the reference frame buffers and reference motion vector arrays. At HDTV resolution, the reference buffers required by the H.264 standard consume over 12 megabytes.

If all other data structures were stored in off-chip memory, the bandwidth to access them would exceed the bandwidth available in the typical DRAM memory interfaces of consumer video products. Reducing off-chip memory bandwidth requirements allows systems to run with fewer memory chips and simpler interfaces, reducing costs. In the UD3000, on-chip local data memories are available to store all intermediate data structures. This includes a data memory for more than one full line of macroblocks. For Main profile, arbitrary slice ordering is not allowed. Macroblocks are coded in raster scan order. Neighboring rows and columns, needed for the deblocking filter, can be retrieved from a 64 kilobyte on-chip memory. The silicon cost for the on-chip memories in the UD3000 is justified by the overall system cost savings due to memory bandwidth reduction.

For H.264 at Main profile, the only memory accesses required by the UD3000 are reading the coded bitstream, writing the final filtered frame result, and reading inter prediction blocks from the reference buffers. Due to the filtering algorithm for sub-pixel interpolation, the block of data read is much larger than the size of the block of pixels predicted. As a result, a worst-case bitstream will require reading data equivalent to 2.3 frames for every frame predicted.

If frame buffers are organized in SDRAM in C style two-dimensional arrays, each row of a block access occurs within a different row of the SDRAM. Changing rows in SDRAM incurs row access (RAS) latency that cannot be masked by interleaving. If frame buffer data is grouped by squares with a number of pixels close to the size of an SDRAM row, the probability of block accesses falling entirely within a singles SDRAM row is maximized. If the square groups are larger than the largest block access then, in the worst case, a block access will require accesses to four groups in four different SDRAM rows.

In the UD3000, the choice was made to store data in groups of 32x32 pixels. Each group requires 1024 bytes, which is the smallest size of SDRAM row used in SDRAM chip sizes today. Pixels within groups are stored in raster order and groups within a frame are stored in raster order. This requires two levels of stride calculation logic and a complex state machine within the DMA controller that moves data between off-chip DRAM and local on-chip data memories.

5. STALL AVOIDANCE

Avoiding stalls allows a guarantee of the required performance with minimal processing power. In traditional cache-based processors, time spent waiting for the service of cache misses means that processing power is wasted. Even media processors with the best stride prediction algorithms sometimes miss [4]. As a result, many processors are needed in order to provide the required effective processing power. Contention

between multiple processors waiting for shared resources exacerbates the problem.

An architecture that eliminates sources of stalls due to contention and remote data accesses will better utilize available processing power. This results in a higher effective processing power and a smaller amount of logic to meet requirements.

Avoiding stalls has the added benefit of the fact that performance specifications are not based on statistical assumptions about miss rates and average latencies. Without stalls, hard real-time performance can be guaranteed.

The UD3000 eliminates stalls through the use of dual-port local data memories, a 5-port register file, and a DMA controller instead of caches for the main memory interface.

5.1 Dual-port DMEM

As video blocks move through the processor intermediate data structures are read from local memory, transformed, and written to a different local memory for the next processor in the data flow pipeline. For typical video codec software, local data memories are typically accessed by at most two processors: one upstream and one downstream. Single port memories would occasionally incur contention between accesses by the upstream and downstream processor. Two-port memories avoid the possibility of contention at the cost of more silicon area required for each SRAM cell. Memories with three or more ports would not improve performance enough to justify the significant increase in silicon area.

5.2 5-port regfile

A classic microprocessor register file has three ports, two for reading instruction operands and one for writing instruction results. In the UD3000 ILP loads and stores for the VU are performed by the CU and may be performed in the same cycle as a VU instruction that uses the two read and one write ports. The VU implements a five-port register file, with an extra write port for loads and an extra read port for stores. This allows data movements to occur simultaneously without performance degrading stalls in the ILP due to contention for register file ports.

The extra write port costs a two-to-one mux for each register. The extra read port costs higher fanout from each register and another read mux.

5.3 DMA Controller

A DMA controller typically requires a smaller silicon area than cache control logic and tags. However, a DMA controller comes at the cost of software overhead in the form of instructions to manage data movements [5].

Video processing lends itself well to a DMA controller implementation because the data accesses are relatively large and predictable and reads can be performed well in advance of when the data is needed by the downstream processor in the data flow.

Further complexity exists in the UD3000 DMA controller as it is aware of the data grouping in off-chip memory and it handles the extrapolation of data for block reads that fall partially outside of the frame. These features significantly reduce otherwise complex operations in software.

6. CONCLUSION

A killer video processor architecture is fully software programmable to accommodate different standards and standards changes. It employs an appropriate degree of instruction and thread level parallelisms. It organizes and locates data in order to minimize expensive off-chip memory bandwidth. A killer video processor architecture also effectively utilizes processing resources by avoiding sources of performance degrading stalls.

7. REFERENCES

[1] http://www.moonlight.co.il/cons_h264player.php

[2] http://www.planetanalog.com/printableArticle.jhtml?articleID=18902400

[3] http://www.wwcoms.com/products/codec/md264.htm

[4] Zucker, Daniel F., Lee, Ruby B., and Flynn, Michael J. An Automated Method for Software Controllerd Cache Prefetching.

[5] http://www.embedded.com/showArticle.jhtml?articleID=16700107