Multi-format Media Player/Recorder Software Design Methodology for Programmable Processors with Hardware Accelerators

Sriram Sethuraman, Sankaranarayanan Parameswaran, Dileep Tamia, Aditya Kulkarni, Manish Singhal
Ittiam Systems Pvt. Ltd, 1 Richmond Road, Bangalore-560025, India.
Email: sriram.sethuraman@ittiam.com

***Abstract*-- Programmable DSP with hardware accelerators is becoming prevalent for multi-format portable media player implementations owing to factors such as lower cost, higher quality expectations, and the flexibility demands. A unified multimedia software design methodology is presented for multithreading on such architectures.**

I. INTRODUCTION

Consumer electronics devices such as portable media players/recorders have a need for supporting a wide spectrum of audio/video compression standards, file formats, and interfaces. The low price point target, quality expectations, and flexibility demands of such a multi-use appliance have resulted in processor architectures that combine a programmable DSP and multiple limitedly configurable co-processors/accelerators. In this correspondence, we present the various requirements for such appliances, highlight the software design challenges of programming for such architectures, and propose a unified multimedia software design methodology for achieving the best possible resolution or quality on the given target platform.

II. PROCESSOR ARCHITECTURE AND CHALLENGES

Video coding standards such as MPEG-2, MPEG-4, and H.264 share a lot of commonality as they all fall under the motion compensated transform coding paradigm. For instance, motion compensation (MC) primarily requires the ability to fetch reference regions and perform suitable interpolation. Combination of a DMA engine with a vector-processing unit (VPU) can be used to accelerate MC. Similarly, forward/inverse transform coding and quantization can be accelerated using a VPU supporting multiply-and-accumulate (MAC) operations. Audio coding standards such as MP3, AAC, AC-3 come under the class of transform-based subband coding algorithms which can all largely benefit from a vector MAC unit. Variable length decoding, used by both video and audio, follow a common Huffman decoding process that can be accelerated using a dedicated hardware block.

Given a DSP subsystem with a central DSP, DMA engine, a heterogeneous set of co-processors and/or hardware accelerators as mentioned above along with some shared memory, the challenge lies in designing a multi-threaded processing pipeline that maximizes the processing throughput. This requires ensuring that the co-processors/accelerators that are on the critical path are idle for as short a time as possible, while taking care of constraints due to dependencies in the processing chain, internal memory available for code and buffering, shared memory design (e.g. number of read/write ports), and cycle overheads involved in setting up each co-processor/accelerator. Some prior work for specific decoders or encoders on specific processors can be seen in [1], [2], [3], [4].

III. UNIFIED DESIGN METHODOLOGY

In this paper, we propose the following unified methodology for arriving at an optimal multithreaded pipeline design that is applicable to a broad class of video and audio encoders or decoders that need to be mapped on to an architecture as described above in II.

1. The various processing stages that need to happen on a basic processing unit are identified along with their inter-dependencies. Multiple mutually exclusive modes are identified at this stage and are statistically characterized in terms of how often they occur. Data movements from external memory to the shared buffers are modeled as processing stages that will happen on the DMA unit.
2. Each processing stage is assigned to a particular processing unit (DSP/co-processor /hardware accelerator) based on the suitability and flexibility. Multiple candidates, if found suitable, are identified for later flexibility in scheduling. For example, a multi-MAC co-processor will be ideally suited for computations such as filtering, transforms, mean squared error computation, etc. Similarly from the flexibility point of view, normative parts of a compression standard can be implemented in a hardware accelerator and non-normative parts (where newer algorithms can be invented) can be kept on a programmable co-processor.
3. Given the inter-dependency between the processing stages, stagger the processing such that all algorithmic dependencies are handled. This means that stages down in the dependency chain are made to work on past basic units while the stages up in the dependency chain are working on newer basic units.
4. Each processing stage on each processing unit is assigned a suitable buffer based on their access constraints.
5. For some stages, the complexity is fixed and hence the time taken will be the same. For other stages, the complexity can vary and hence the time taken will also vary. The time taken for the fixed complexity stages and the average time and range of time taken for the varying complexity stages are measured on the respective mapped processing units.
6. Explore the granularity of processing in multiples of the basic unit and select the one that gives the best

performance while meeting the platform constraints and dependency constraints. The factors that would influence this are, the savings stemming from going to that granularity, the ability to exploit statistical properties in a reasonably straight-forward fashion, and the amount of memory required in the shared buffers.

7. Identify the possible variations in the staggering that are possible without breaking the dependencies and platform constraints. Some changes in the buffer assignments from step 4 above and multiple processing unit to processing stage mappings are attempted at this stage. Select a subset of these possible pipelines that offer the best timing (based on the average processing time) for processing at the chosen granularity.
8. Of these selected pipelines, a single pipeline is chosen based on its ability to offer the best overall timing in the presence of statistical variations. For instance, let us consider a case where a particular processing stage has to perform a task *N* times in the worst case and *m* times on the average. In a particular pipeline, there may be certain other processing stages scheduled on the other processing units in parallel with the above such that it would not matter whether this processing unit performs the task *m* times or *N* times. On the other hand, there could be a pipeline where the processing on the other units is only in parallel with the first *m* tasks and hence can translate to an overall timing reduction statistically.
9. On the chosen pipeline, identify the processing unit that is on the critical path, and, if possible, move some tasks that are currently scheduled on this unit and that could be performed on other units without breaking the dependencies or buffer constraints to those other units.

IV. Video Encoder Design Example

Figure 1 illustrates a multithreaded pipeline for performing video encoding on a DSP subsystem with a vector MAC unit, a hardware accelerator capable of performing quantization, inverse quantization, and variable length encoding, a DMA engine, and 3 shared memory buffers with the constraint that only one processing unit can access a buffer at a time. The pipeline spans over 4 macroblocks (N-2 to N+1). It can be seen that processing tasks are assigned according to suitability and flexibility discussed earlier, while meeting the buffer access and data dependency restrictions. For example, buffer B access determines the total time taken. The schedule has also minimized the idle time on the critical path (which is the vector MAC unit in this case). The motion estimation step has a variable complexity and can go on till the end of variable length encoding without affecting the cycles per MB.

V. Results And Conclusions

Using the proposed design methodology, on a commercially available sub-$20 processor with multiple co-processors and hardware accelerators, multiple video codecs at up to D-1 resolution and multiple high quality audio codecs have been realized to achieve close to home-theatre quality. A complete multimedia system shall be demonstrated during the presentation.

References

[1] J-H Kuo, J-L Wu, Jim Shiu, K-L Huang, A low-cost media-processor based real-time MPEG-4 video decoder**,** ICCE 2002, pp. 272 – 273.

[2] Lulin Chen, Zhihai He, Sriram Sethuraman, Chang Wen Chen, MPEG-4 encoder implementation on MAP-CA, ICCE 2002, pp. 276 – 277.

[3] Wittenburg, J.P., Pirsch, P., Meyer, G., A multithreaded architecture approach to parallel DSPs for high performance image processing applications, Proc. Of Workshop on Signal Processing Systems 1999, pp. 241 – 250.

[4] Jeong-Min Kim, Seok-Kyun Hong, Eel-Wan Lee, Soo-Ik Chae, Multithread video coding processor for the videophone, Proc. Of Workshop on VLSI Signal Processing VIII, 1995, pp. 470-480.

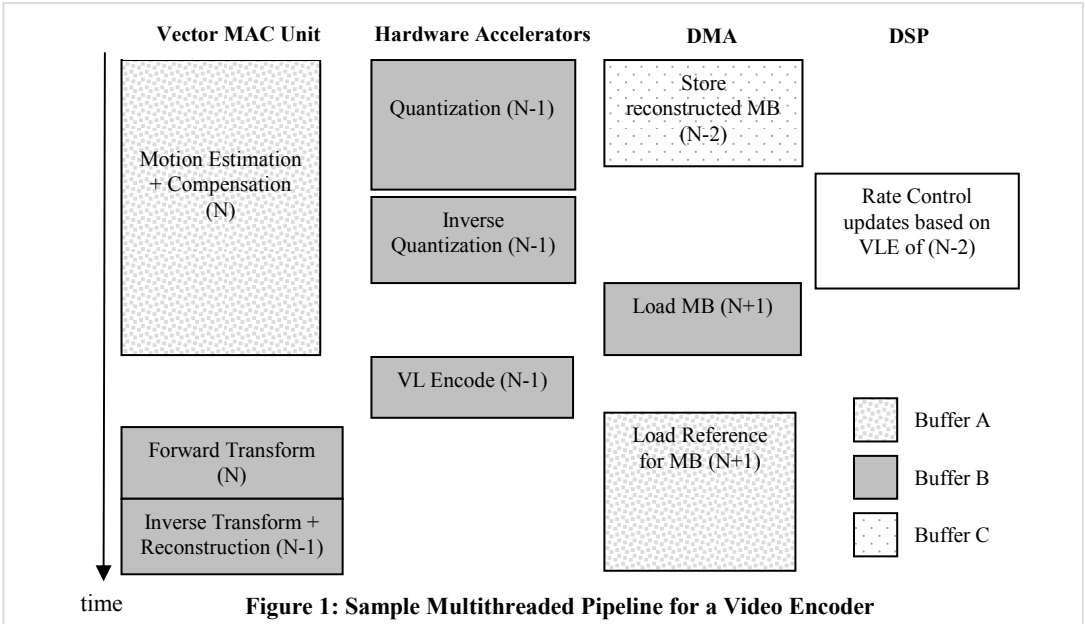

Figure 1: Sample Multithreaded Pipeline for a Video Encoder