Multi-thread VLIW processor architecture for HDTV decoding

Hansoo Kim, Woo-Seung Yang, Myoung-Cheol Shin, Seung-Jai Min*, Seong-Ok Bae* and In-Cheol Park

Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Korea

*DSP Group, LG Corporate Institute of Technology, Korea

Abstract

This paper describes a single-chip high definition television (HDTV) decoder which performs system parsing, video decoding, audio decoding and resolution conversion. To process a huge amount of data and deal with various standards in the decoder, a multi-thread processor architecture is proposed to minimize the overhead cycles of task-switching. The features of parallelism and conditional branches in MPEG2 video decoding algorithm are considered to enhance the performance of the embedded processor and to reduce the size of code memory. Experimental results show that the proposed processor architecture is 5.3 times faster than a scalar processor at the cost of negligible increase of code memory.

I. Introduction

As the high definition television (HDTV) broadcasting begins, it is expected that the HDTV standards make a new television market. To expand the new market fast, the television receiver should be low-cost. A single chip implementation of complex functions of HDTV decoding is indispensable for the cost reduction. The HDTV decoding consists of system parsing, video decoding, and audio decoding. The system parsing deciphers the system information and demultiplexes video streams and audio streams. The video part of most HDTV standards use MPEG2 video MP@HL for video compression, while the audio part adopts various audio coding algorithms such as MPEG layer II audio (MUSICAM) or Dolby AC-3. As the low-cost implementation of the HDTV decoder is needed, many researches have been focused on developing HDTV decoder chip-sets [1,2] or single-chip MPEG2 MP@ML decoders [3]. However no one has integrated the full functions of HDTV into a single-chip.

The requirements for a single-chip HDTV decoder are as follows. First, since the video format can be large up to 1920x1080, a sufficient processing power is necessary to deal with a huge amount of video data in a tight time budget. Second, the decoder must support many video formats, because, for example, the HDTV standard generated by the Advanced Television Standard Committee (ATSC) includes 18 video formats of various video sizes and scanning types. Third, the system parsing part has to be programmable in order to support various HDTV standards, such as ATSC and Digital Video Broadcasting (DVB). Lastly, the decoder must include auxiliary functions such as on-screen display (OSD), video format conversion and sub-picture display. Hence HDTV decoders must have a great degree of programmability as well as high-performance.

In this paper, we present a single-chip HDTV decoder that satisfies the above requirements. Besides several dedicated hardwares for high performance video decoding, the decoder includes two processors: an application-specific DSP for audio decoding and a multi-thread processor for the system parsing and the calculation of virtual addresses in video decoding.

The remainder of this paper is organized as follows. Section 2 shows an overall architecture of the HDTV decoder chip. In Section 3, we present a multi-thread processor that performs the system parsing and the video decoding control simultaneously without overhead cycles. The VLSI implementation is described in Section 4.

II. Overall Architecture

The architecture of the single-chip HDTV decoder is illustrated in Fig. 1, which is composed of 5 blocks; video processing unit, audio processing unit, main processor, interface unit and display unit.

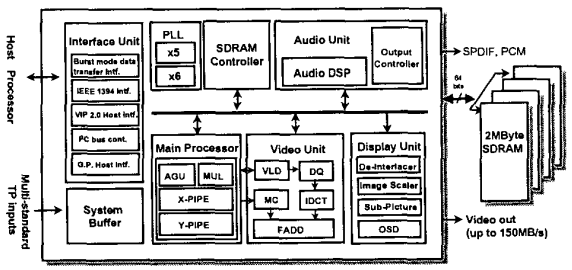

Figure 1: Block diagram of overall architecture

The video processing unit performs the MPEG2 video decoding algorithm. Since it requires very high processing power, the unit is composed of several dedicated hardwares including VLD (variable length decoding), DQ (dequantization), IDCT (inverse discrete cosine transform),

0-7803-5809-0/00/$10.00 © 2000 IEEE IEEE 2000 CUSTOM INTEGRATED CIRCUITS CONFERENCE

MC (motion compensation), and FADD (final addition). To improve the speed of VLD, multiple short codes used frequently are decoded in parallel. And to reduce the execution time of IDCT, we exploit the characteristics of little non-zero coefficients in IDCT input. Since a lot of IDCT input coefficients of most MPEG stream are zero, significant reduction in IDCT computation time can be obtained by performing IDCT on input coefficient-by-coefficient base and computing only non-zero coefficients. Detail explanations can be found in [4,5].

The audio processing unit is implemented with an application-specific DSP processor, since it has to support a number of audio decoding algorithms. To increase the performance, the audio processing unit has an architecture and an instruction set specialized for audio processing and bit-stream parsing.

The main processor is a multi-thread processor that has two execution-pipes. It performs the system parsing and controls the dedicated blocks of the video processing unit by activating control-signals. The processor is equipped with two execution-pipes to provide high performance required in calculating the virtual addresses of reference macroblocks and issuing them to video processing unit within a given timing budget. Though the system parsing does not require high performance, it should be serviced with low latency to prevent buffer overflow and under flow. Therefore a multi-thread technique is employed to eliminate overhead cycles of task-switching. We will explain detail features of this processor in Section 3.

The interface unit provides various interface standards such as PCI, IEEE 1394, VIP 2.0, I^2C, and internal host interface. Using these interfaces, the decoder communicates with external devices such as Tuner, VSB demodulator, Digital VCR and network devices and can download the on-screen display data and bit streams.

The display processing unit converts video format into a proper format for a specific monitor. It also performs de-interlacing, i.e., conversion of interlaced scanning into progressive scanning. For low-end applications, this unit can display the 16:9 video data on the 4:3 TV by performing either letter-box or pan-scan conversion. This unit also supports diverse OSD regions with any position and any size and various OSD functions such as hardware cursor, OSD scrolling and panning.

III. Features of Main Processor

The main processor performs two independent tasks concurrently; one is the video decoding and the other is the system parsing. Their requirements are very different from each other, i.e., the video decoding requires high computation power, while the system parsing has to be performed with low latency. As the system parsing itself is not computation-intensive and the processor has sufficient computation power to support the system parsing, it is not hard to perform the system parsing concurrently in the processor. However, to prevent overflow of system buffers and to prevent underflow of the video or audio stream buffers in the system parsing stage, the processor must decode system data as soon as they arrive at the system buffers. If two separate pipes process two tasks respectively, the pipe that processes the video decoding may be lack of computation power and the other pipe is apt to be idle. So executing two tasks in a shared pipe architecture are better than executing them in separate pipes and fast task-switching becomes very important in the shared pipe architecture..

One way to handle the task-switching is to use a periodical interrupt during the video processing task. To preserve the status of the video decoding after task-switching, saving and restoring of flags and register files are needed. So the conventional interrupt mechanism may cause too many task-switching overhead cycles, if the interrupt occurs frequently. For example, ATSC standard specifies a maximum 38.78 Mbps data transfer rate in a 6MHz cable television channel. In such a case, 1-byte system data are arrived at the system buffers at every 14 clock cycles, assuming that the system is clocked at 67.5 MHz. If we assume the task-switching overhead is four cycles, the task-switching causes 29% performance degradation.

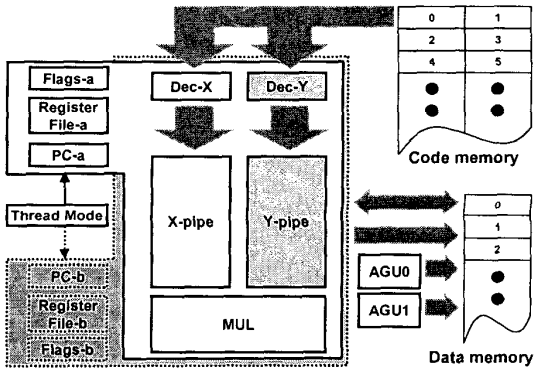

Figure 2: Architecture of main processor

To remove the task-switching overhead, we propose a multi-thread architecture that provides duplicated register files, but shares the execution unit. Fig. 2 shows the proposed architecture. The processor has two sets of register files and flags to preserve processing status of two-threads simultaneously and a bit called *thread mode* flag controls which set is active. Though it executes only one task at a time, the processor can always keep the processing status of both tasks. Hence, the video decoding and the system parsing can be switched with minimal overhead, that is, zero cycle. Thread switching is triggered by invoking a special instruction in software or by activating a request pin in hardware. When the processor has to wait for a time period

specified by DTS(decoding time stamp) or cannot progress a program any more due to waiting for external data, the program can trigger the task-switching using a special instruction that changes the *thread mode* flag. The hardware interrupt request signal is asserted when the number of system buffer entries exceeds a predefined number.

To achieve high-performance, the processor must exploit parallelism inherent in the video decoding algorithm such as parallelism between luminance and chrominance, parallelism among multiple blocks in a macroblock, and parallelism among horizontal, vertical, forward, and backward motion vectors, etc. To exploit the parallelism, the processor has two execution-pipes (X-/Y- pipe), as shown in Fig. 2. In every cycle, two instructions can be fetched and executed in parallel, and one execution-pipe can communicate with the other through a shared register.

Multiple-issue processors are likely to increase the size of code memory by inserting NOP instructions when two instructions are dependent to each other. To eliminate the unnecessary NOP instructions, the processor has two operating modes that control the issue rate. One is that two pipes are controlled separately by two 24-bit instructions and the other is that single X-pipe is controlled by one 24-bit instruction. As the parallelism is usually determined according to the decoding layers of MPEG2 syntax, the issue rate can be statically determined. When programmers fail to find sufficient parallelism, the issue rate control is more efficient than inserting NOP instructions into Y-pipe. Furthermore, additional power saving is expected by removing useless code fetch.

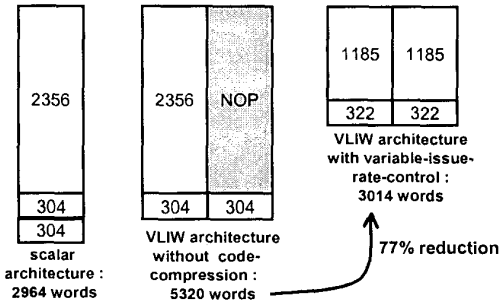

Figure 3: Reduction of code memory by variable-issue-rate-control scheme (a) code memory of scalar architecture (b) code memory of two-execution-pipe architecture without any code-compression (c) code memory of two-execution-pipe architecture with variable-issue-rate-control

Fig. 3 shows how much code size is reduced by the proposed scheme, that is, the effect of the issue rate control scheme on code reduction. In the assembly program of the video decoding and the system parsing, 2356 instructions must be executed sequentially and 304 pairs of instructions can be executed in parallel. Fig. 3(a) shows the code memory of 2964 words, assuming a scalar architecture. In Fig. 3(b), a two-execution-pipe architecture is assumed and the width of code memory becomes double for issuing two instructions at a cycle. However, because of the NOP instructions inserted in the sequential program segment, the code memory size becomes 5320 words. Fig. 3(c) shows the code memory of the proposed scheme. By controlling the issue rate, there is no need to insert NOP instructions in the sequential program segment and thus only 3014 words are needed. Hence the variable issue rate control scheme achieves about 77% code size reduction compared to two-issue processors.

To efficiently utilize the parallelism in video streams, the frequency of conditional branches should be minimized as much as possible, because the next instructions cannot be executed until the branch condition is resolved. To alleviate the performance degradation by the branch delay, many works such as delayed-branch and branch prediction have been studied. As branch prediction methods require a lot of hardware such as branch history buffers, prediction logic and so on, the main processor supports effective conditional instructions and condition generation hardwares to eliminate the conditional branches without additional hardware overhead. Fig. 4 shows two types of conditional branches which are frequently used in the video decoding and the system parsing. First, one instruction can be executed or not, according to a condition, as shown in Fig. 4(a). By using conditional execution instructions, the statements are simply implemented without any conditional branch. Second, many complex branch conditions often appearing in MPEG2 video decoding algorithms are associated with hardware logics that generate the complex conditions in a cycle, as shown in Fig. 4(b). With condition generation hardwares, the statements that contain the complex conditions can be executed with one instruction.

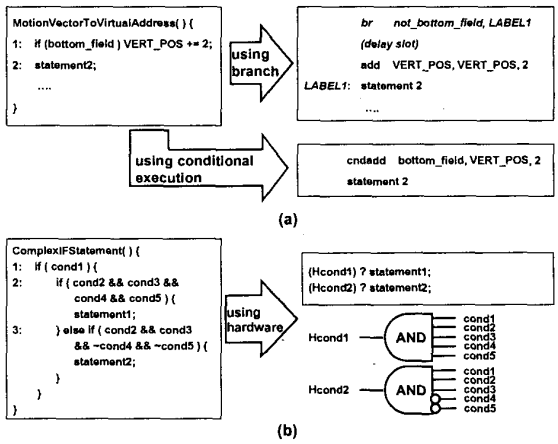

Figure 4: Conditional execution instructions for branch-free coding (a) simple conditional execution instruction (b) hardware condition generation

To visualize the performance improvement achieved by the proposed techniques, we simulated the assembly code of a video decoding program using our instruction set simulator. Three (I-, P-, B-) pictures of 1920x1080 resolution are used as a test stream. The number of cycle required to decode the stream is measured by applying one technique at a time. Fig. 5 shows the cycle count reduction and speedup achieved by each technique. Fig. 5(a) shows the cycle count taken in a base scalar architecture. Three times speedup is achieved by using the special condition generation hardware and further reduction by the use of some special instructions. Additional Y-pipe gives about one half speedup and totally five times speed-up is achieved by using all the proposed techniques.

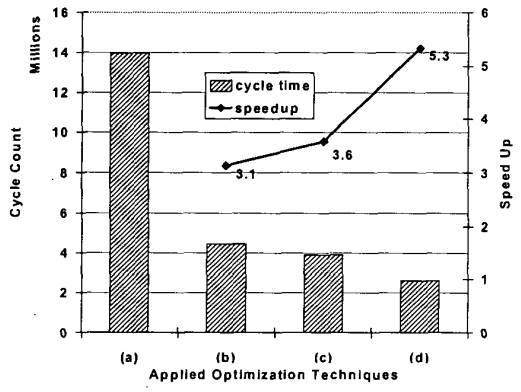

Figure 5: Cycle count and speedup of each optimization technique; (a) instruction count in a base architecture (b) instruction count when special conditions are used (c) when special instructions and special conditions are used at the same time (d) when two execution-pipes are used.

IV. VLSI Implementation

The chip is implemented using 0.25 μm 5-metal CMOS technology. Fig. 6 shows the layout of the single-chip HDTV decoder containing about 2.6 million gates in a die size of 9.1x9.1 mm^2. The characteristics of the decoder is summarized in Table 3.

Table 3: Specification of HDTV decoder

Technology	TSMC 0.25 um CMOS 5LM
Clock frequency	135 MHz, 67.5 MHz
Power supply	2.5V
Gate count	2,7 Million
Power consumption	2.8 W
Die size	9.1 x 9.1 mm^2
Number of I/O pin	388 pins
Package	PBGA

V. Conclusion

We have developed a single-chip HDTV decoder that processes system parsing, video decoding, audio decoding and display processing. To deal with various television standards, it is equipped with a programmable system parsing unit and an input/output format conversion unit. The programmability and concurrent processing of system parsing are achieved by incorporating a high-performance processor that has multi-threading and issue rate control capabilities and by attaching condition generation hardwares for complex conditions. Experimental results show that the proposed processor architecture is 5.3 times faster than a scalar processor in performing the video decoding algorithm.

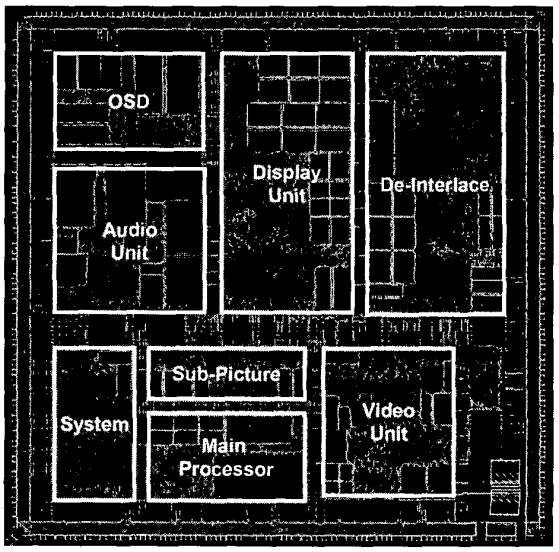

Figure 6: Layout of single-chip HDTV decoder

References

[1] K. Tsunashima, K. Muneishi, H. Morikawa, H. Kayashima and V. Sinyanskiy, "An integrated DTV receiver for ATSC digital television standard," IEEE transaction on consumer electronics, 1998, pp. 667-671.

[2] R. Sita, E. Brosz, R. Meyer, L. Philips and R. T. Ryan, "A single-chip HDTV video decoder design," IEEE transaction on consumer electronics, 1998, pp. 519-526.

[3] Y. Okada, et al., "An 80mm^2 MPEG2 Audio/Video decode LSI," ISSCC Digest of Technical Papers, pp.264-265, Feb., 1997.

[4] S.-O. Bae and K.-S. Kim, "Symbol parallel VLC decoding architecture for HDTV application," 1998 Digest of technical papers of ICCE, pp. 52-53, 1998.

[5] S.-O. Bae and S.-J. Min, "A computationally efficient IDCT algorithm," Proceeding of MWCAS'97, vol. 2, pp. 973-976, 1998.