Power and Memory Bandwidth Reduction of an H.264/AVC HDTV Decoder LSI with Elastic Pipeline Architecture

Kentaro Kawakami
Department of Informatics and Electronics
Graduate School of Science and Technology, Kobe University
Kobe, Hyogo 657-8501
Tel : +81- 78-803-6234
Fax : +81-78-803-6317
e-mail : kawakami@cs28.cs.kobe-u.ac.jp

Mitsuhiko Kuroda
Department of Informatics and Electronics
Graduate School of Science and Technology, Kobe University
Kobe, Hyogo 657-8501
Tel : +81- 78-803-6234
Fax : +81-78-803-6317
e-mail : mitsuhiko@cs28.cs.kobe-u.ac.jp

Hiroshi Kawaguchi
Department of Computer and Systems Engineering
Kobe University
Kobe, Hyogo 657-8501
Tel : +81- 78-803- 6317
Fax : +81-78-803-6317
e-mail : kawapy@cs.kobe-u.ac.jp

Masahiko Yoshimoto
Department of Computer and Systems Engineering
Kobe University
Kobe, Hyogo 657-8501
Tel : +81- 78-803- 6214
Fax : +81-78-803-6394
e-mail : yosimoto@cs.kobe-u.ac.jp

Abstract - We propose an elastic pipeline that can apply dynamic voltage scaling (DVS) to hardwired logic circuits. The proposed pipeline can also reduce a required local bus bandwidth. In order to demonstrate its feasibility, a hardwired H.264/AVC HDTV decoder is designed as a real-time application. The proposed architecture reduces a power to 56% in a 90-nm process technology, compared to the conventional clock-gating scheme or a local bus bandwidth to 37.2%.

I. INTRODUCTION

Dynamic voltage scaling (DVS) is an effective technique for general-purpose processors to achieve both high peak performance and low average power [1,2]. Because an operating frequency in a CMOS digital circuit, f, is formulated as follows [3], an LSI performs faster as a supply voltage, V_{dd}, becomes higher:

$$f = k\frac{(V_{dd} - V_{th})^{\alpha}}{V_{dd}}, \quad (1)$$

where k is a constant. α represents a velocity saturation index in a short-channel MOSFET, and is about 1.6 in a 90-nm process technology used in this study. V_{th} is a threshold voltage of the MOSFET. In addition, since a power in an LSI, P, is expressed as follows, the supply voltage drastically increases the power:

$$P = a \cdot C \cdot f \cdot {V_{dd}}^2 + V_{dd} I_0 10^{\frac{V_{gs} - V_{th}}{S}}, \quad (2)$$

where a is an activation ratio, C is a total capacitance, I_0 is a leakage current when $V_{gs} = V_{th}$ and S is a subthreshold swing.

Figure 1 illustrates the relations between an operating frequency and power. In DVS, since a supply voltage can be optimized if an operating frequency does not need to be the maximum, the power can be decreased by the low operating frequency and low supply voltage. If an operating frequency can be set low, in other words, if a required performance is low, DVS adaptively lowers both the operating frequency and supply voltage in order to reduce the power. If the maximum performance is instantaneously needed, the highest supply voltage and highest operating frequency are utilized so that DVS can accommodate the peak performance.

Because (1) and (2) are approved for every CMOS digital circuit, applications of DVS are not limited to general-purpose processors, and thus it is theoretically applicable to hardwired logic circuits. However, there are few examples applying DVS to hardwired logic circuits for real-time processing. This reason is explained in this way. A dedicated hardware is often built with pipeline architecture for high throughput, which requires a certain number of clock cycles in the worst case. Assuming the worst-case workload, each pipeline stage has to be configured, and every pipeline operation simultaneously starts at the beginning of an allocated period. Consequently, a required operating frequency is uniquely fixed. There is no room to change the operating frequency. Hence, DVS is not applicable to a hardwired circuit.

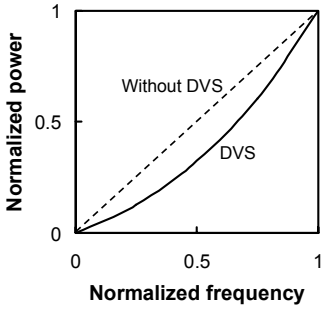

Figure 1. Relationship between power and frequency in DVS.

In order to apply DVS to hardwired logic circuits, we propose an elastic pipeline architecture. Since this architecture can save the number of cycles in the pipeline operation depending on characteristics of input data, it enables hardwired logic circuits to employ DVS, and thus achieves lower power. As a design example, we propose a novel H.264/AVC high-definition television (hereafter, H.264 HDTV) decoder. The power can lower to 50% of the conventional pipeline architecture using clock gating.

The following sections are as follows. The elastic pipeline architecture is proposed in Section II. The reason why the number of execution cycles in H.264 decoding is varied in pipelined stages, is mentioned in Section III. We discuss memory bandwidth required for the elastic pipeline in this section. In Section IV, the novel H.264 HDTV decoder architecture is described, and the effectiveness of the proposed architecture is verified by designing a dedicated

1-4244-0630-7/07/$20.00 ©2007 IEEE.

hardwired decoder. Finally, the findings of this paper are summarized in Section V.

II. ELASTIC PIPELINE ARCHITECTURE

A. Conventional Pipeline Architecture

Figure 2 shows a timing diagram of the conventional pipeline architecture. The worst-case execution cycles (WCEC) signify the maximum number of execution cycles required for one pipeline process. A hatching area in the figure represents processing cycles during which a stage is running with a datum. Since all pipelined processes in the conventional architecture start at the beginning of the allocated WCEC supposing the worst-case workload, all the stages have to idle until the next WCEC even if they finish earlier. The clock-gating technique may be utilized to cut off the unnecessary power caused by the idle cycles, but the power saving is limited.

Extending Fig. 2, let us consider a case that M pipeline stages process N data. The number of execution cycles to complete the N-th datum in the M-th pipeline stage, EC_{conv}, is expressed in the conventional pipeline architecture as follows:

$$EC_{conv} = (M + N - 2) \times WCEC + W_{M,N}, \quad (3)$$

where $W_{M,N}$ ($\leq WCEC$) indicates the number of processing cycles for the N-th datum in the M-th stage. As N is larger, EC_{conv} becomes closer to $N \times WCEC$, which means that there is not much room to apply DVS. Namely in the conventional pipeline architecture, only the execution cycle variation of $W_{M,N}$ can be exploited.

When the conventional pipeline accesses an external RAM through a local bus to read input data and/or write output data, the bandwidth of the local bus, BW, must satisfy the following:

$$BW \geq WCDA \times f / WCEC, \quad (4)$$

where $WCDA$ represents the worst-case data amount transferred from/to the external RAM within one pipeline process, and f represents a operating frequency of the pipeline.

B. Elastic Pipeline Architecture

Figures 3 (a) and (b) illustrate a concept and timing diagram of the proposed elastic pipeline architecture. A stage in the elastic pipeline sends a process completion signal to a pipeline controller once its process is completed. If all the completion signals arrive at the pipeline controller, it sends back start signal to the pipeline stages. After receiving it, each stage turns off the process completion signal, and then starts a next pipeline process. In this manner, the elastic pipeline architecture shortens the number of execution cycles as many as possible, and saves cycles. As illustrated in Fig. 3 (b), the elastic pipeline architecture requires the less number of cycles than the conventional one since common idle cycles are eliminated (compare with Fig. 2).

Eventually, N data processed by the M stages in the elastic pipeline require the following execution cycles, EC_{prop}:

$$EC_{prop} = \sum_{i=1}^{M+N-1} \max(W_{1,i}, W_{2,i-1}, \cdots, W_{M,i-M+1}), \quad (5)$$

where $W_{p,q}$ represents the number of execution cycles for the q-th datum processed by the p-th pipeline stage. If either $q<1$ or $q>N$, then $W_{p,q}=0$.

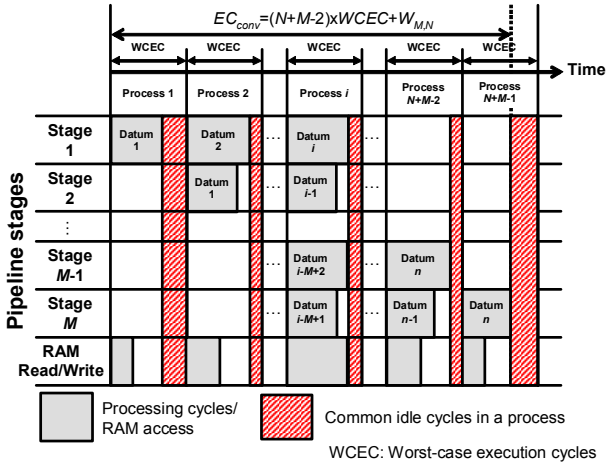

Figure 2. Timing diagram of the conventional pipeline.

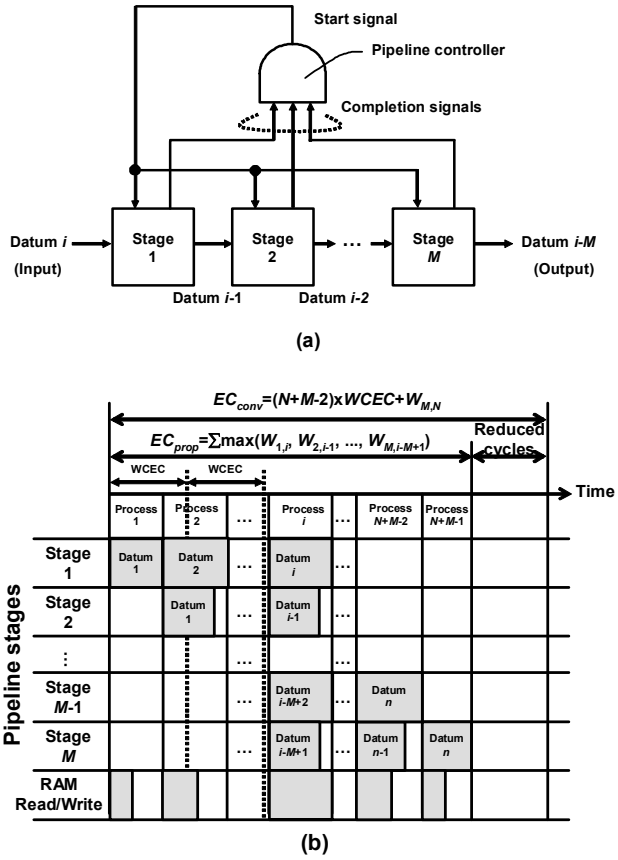

Figure 3. Proposed elastic pipeline architecture, (a) concept, and (b) timing diagram.

III. EXECUTION CYCLE VARIATIONS IN H.264 DECODING

Figure 4 shows a block diagram of a typical H.264 decoder. In the following six functional blocks, the numbers of execution cycles are varied, which can be exploited with

the proposed elastic pipeline architecture.

A. Entropy Decoding

An entropy decoder transforms an input bit stream to syntax elements, which include information on macro block (MB) types, intra prediction modes, motion vectors, coded block patterns, coefficients, and so on. If context adaptive binary arithmetic coding (CABAC) is selected for an entropy coding mode, firstly a bit stream is expanded by a CABAC decoder and then syntax elements are decoded from the expanded bit stream. Some MBs have more than a hundred of syntax elements, but other MBs have only one syntax element if they have been encoded as skip MBs. Consequently, the number of execution cycles required for the entropy decoder depends on how many syntax elements an MB has.

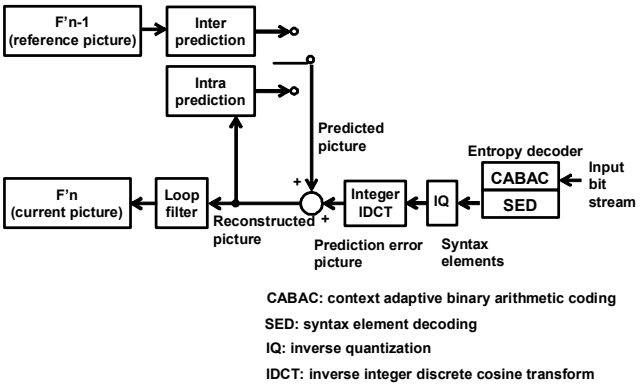

Figure 4. Block diagram of typical H.264 decoder.

B. IQ and IDCT

According to coefficients decoded by the entropy decoder, an inverse quantization (IQ) and inverse integer discrete cosine transform (IDCT) generate a prediction error picture. The coefficients are structured in a 4 × 4 matrix form, and thus the IQ and IDCT basically carry out 4 × 4 matrix operations. An MB has 24 matrixes comprised of 4 × 4 coefficients. A 4 × 4 matrix sometimes has a zero matrix, in which case the matrix operations in the IQ and IDCT can be cancelled. Hence, the number of execution cycles for the IQ and IDCT is varied depending on how many non-zero matrices an MB includes.

C. Intra Prediction

An intra prediction in Fig.4 generates a predicted picture with neighbor MBs that were beforehand processed, according to an intra prediction mode. H.264 provides 13 types of intra prediction modes, which require different calculations. For example, the Intra_4x4_Horizontal prediction mode and Intra_4x4_Vertical prediction mode just copy pixel data. These prediction modes do not make any calculation. On the other hand, the Intra_4x4_DC prediction mode requires a 8-tap filter operation. Therefore, the number of execution cycles for the intra prediction depends on an intra prediction modes chosen in an MB.

D. Inter Prediction

An inter prediction generates a predicted picture from motion vectors and a reference picture which were beforehand constructed. H.264 provides an integer-pixel, half-pixel, and quarter-pixel precision motion vectors. If a motion vector is an integer-pixel precision, its motion compensation is not carried out, and a portion of the reference picture pointed by the integer-pixel precision motion vector is simply utilized for the predicted picture. Else if a motion vector is a half-pixel precision, a 6-tap filter is required in order to generate a half-pixel precision picture from the integer-precision reference picture. In the other case that a motion vector is a quarter-pixel precision, another 2-tap filter is needed in order to generate a quarter-pixel precision picture from the half-pixel precision picture. In this way, the number of execution cycles for the inter prediction depends on the pixel precision.

E. Loop Filter

A reconstructed picture is generated by adding a prediction error picture to a predicted picture. The reconstructed picture needs to pass through a loop filter before it is output to reconstructed picture memory. The loop filter smoothes pixels on block boundaries. Although an MB has 48 block edges to be filtered, all the edges do not need to be filtered. Depending on an MB type or pixel values, each edge is adaptively decided to be filtered or not. This influences the number of execution cycles required for the loop filter.

F. Data Transfer from/to External DRAM

H.264 Main Profile Level 4 requires a 96-Mbit coded picture buffer [4], where reference pictures for the inter prediction are stored. This buffer is usually configured as an off-chip DRAM. The off-chip DRAM is also utilized as a bit stream buffer. An H.264 decoder LSI accesses this DRAM to read in bit streams and predicted pictures, plus write out reconstructed pictures. The reconstructed pictures are outputs of the decoder, and are reused as reference pictures for next decoded pictures.

The majority of the total amount of data transferred from/to the off-chip DRAM is read as reference pictures for inter prediction. The pixel rate required for the inter prediction depends on how many, how precise and what type of motion vectors an MB has.

IV. LSI IMPLEMENTATION

This section proposes a novel H.264 decoder LSI on which the elastic pipeline architecture is implemented. DVS properly works since the proposed architecture saves the number of execution cycles thanks to the elastic pipeline. Power reduction is also discussed.

A. Proposed H.264 Decoder Architecture

Figure 5 illustrates the block diagram of the propose H.264 HDTV decoder, which supports Main Profile Level 4 at a 108-MHz operation. Since this specification requires a 96-Mb memory as a buffer for decoded pictures, the proposed architecture essentially uses an external DRAM.

Some buffers are located between functional blocks. All the buffers are two-bank SRAMs for double buffering, except the RAM for intra prediction that is not frequently

accessed. For instance, while the IQ/IDCT block is writing processing result into Bank 0, the prediction error adder is reading the previous result which was written by the IQ/IDCT block. In a next pipeline process, Bank 0 and Bank 1 change places. Since the IQ/IDCT and inter prediction have no data dependency, they can be parallelized. Then, a predicted and prediction error pictures are added at the prediction error adder. Finally, the loop filter smoothes the reconstruction picture.

As entropy decoding, H.264 Main Profile provides two methods, CABAC and context adaptive variable length coding (CAVLC). Because CABAC achieves 15% higher coding efficiency [5], only CABAC is considered in this paper.

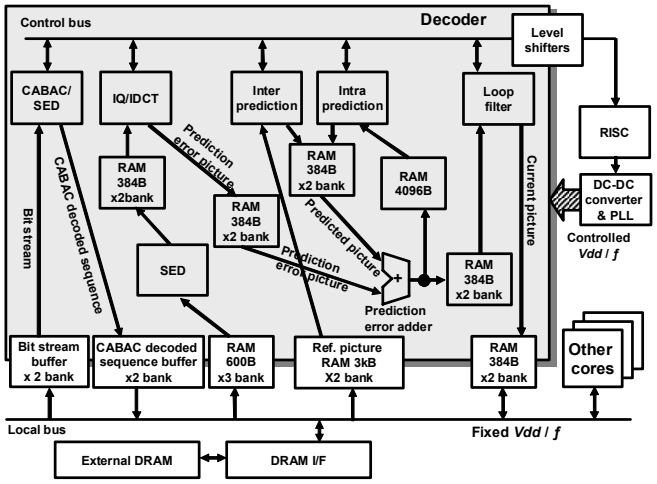

Figure 5. Block diagram of the proposed H.264 decoder.

Reference [6] proposes a cycle-saving CABAC architecture that can decode three bits per cycle. However, the simulation result indicates that the most cycle consuming MB requires more than 1,000 bits when a 20-Mbps bit stream is assumed, which makes it difficult to implement CABAC into an MB pipeline. Actually, the H.264 standard allows an MB to contain 3,200 bits. As a result, this paper uses a buffering scheme, where a decoded bit stream is preliminarily buffered. This buffer averages the workload of CABAC decoder. Since H.264 Level 4 limits a bit stream to 20 Mbps, CABAC can utilize 5.4 (= 108 MHz / 20 Mbits) cycles for 1 bit decode in the buffering scheme.

As mentioned above, CABAC is independent from the MB pipelining. The decoder accesses an external DRAM to read bit stream for CABAC, a CABAC-decoded stream for SED, reference picture for inter prediction, boundary pixels of MB for the loop filter, and to write the CABAC-decoded stream and a current picture. The required traffic for decoding one MB is 41 kbits in the worst case, and thereby the bandwidth between the decoder and DRAM is estimated at 10 Gbps (=41 kbits/MB×8,160 MBs/frame×30frames/s). In reality, wider bandwidth is necessary since other cores as a audio codec and post video processing request certain traffic.

If all the pipeline stages finish by the WCEC, the elastic pipeline can reduce processing cycles. Assuming that HDTV video sequences contain 244,800 MBs/s (= 8,160 MBs/frame×30 frames/s), 440 cycles are available for an MB process at an operating frequency of 108MHz. Since the function of the prediction error adder is 384-time addition of simple 8-bit data and clipping, it would always need 384 cycles and destruct elastic pipelining. However, this value is reduced to a quarter (96 cycles) by a four-degree parallelism. The area overhead is small because the 8-bit addition and clipping operation are fortunately not much.

A supply voltage and operating frequency outside the decoder should not be controlled since it is preferable that the DRAM interface operates at a fixed supply voltage and operating frequency for compatibility with other hardware cores. In order to solve the interface problem between the DVS domain and outside, we allocate two-bank SRAMs there. Figure 6 demonstrates how to control the two-bank SRAMs. Now, the supply voltage and operating frequency in Bank 0 are controlled in synchronization with those of the decoder. While the decoder reads/writes data from/to Bank 0, the supply voltage and operating frequency in Bank 1 are as same as those of the local bus and the external DRAM. The external DRAM can read/write data from/to Bank 1. In a next pipeline process, the roles of these are alternated. Therefore, the dynamic controls of the supply voltage and operating frequency in the decoder do not give any influence to the outside. The SRAM which sends back the CABAC-decoded sequence to the SED is organized as a three-bank SRAM. Since it is difficult to know how much CABAC-decoded sequence is required to decode one MB, the CABAC-decoded sequence for the MB often straddles two banks. In this case, the third bank is demanded to store the CABAC-decoded sequence from the external DRAM for the next MB. Since each bank in the three-bank SRAMs also cyclically alternate its role in every pipeline process, the proposed architecture never disturbs the outside of the decoder.

B. Cycle Reduction by Elastic Pipeline

To verify the execution cycle reduction by the elastic pipeline, we designed every block with the H.264 reference software JM 9.6 [7], and carried out cycle simulation with Celoxica Handel-C [8]. Operating and idle times of each functional block are obtained from the cycle simulation. Gate level netlists generated from register transfer level netlists are acquired with a logic synthesis tool (Design Compiler). Power of each functional block is estimated by SPICE with a 90 nm-process model file.

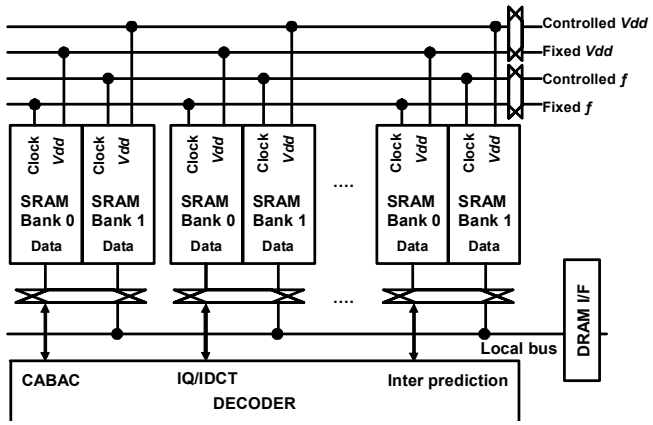

Figure 6. Two-bank SRAM for interface with fixed-voltage local bus.

Figure 7 shows a fluctuation of the execution cycles in the pipeline process. The normalized local bus traffic is also considered as well, when a bandwidth of the external DRAM is varied. The vertical axis is an execution cycle ratio normalized by the WCEC, and normalized traffic on a local bus. This figure indicates the following points.

- The local bus traffic is varied in synchronization with the execution cycle of the decoder.
- If a low-frequency DRAM is used, the workload of the local bus becomes larger than that of the decoder. This hinders an execution cycle reduction of the decoder since the DRAM interface occupies the multi-bank SRAMs in the decoder for a longer time than its execution time.
- If the bus frequency is 6.4 Gbps (= 16 bits×400 MHz), the workload of the local bus becomes the same as that of the decoder.
- The average execution cycles in the decoder are approximately 50% of the WCEC.
- Thus, if a high clock frequency DRAM is utilized, an operating frequency may be halved, and power reduction by DVS can be expected.

Since a B picture requires a larger volume of data for inter prediction, the bus traffic becomes busier when the DRAM bandwidth is 4.3 Gbps (= 16 bits × 266 MHz). In this case, the elastic pipeline wastes common idle cycles until data arrival from the DRAM. It degrades the performance of the elastic pipeline.

C. Power Estimation

For DVS, a supply voltage and operating frequency are changed by a feedback mechanism as shown in Fig. 8 [9]. A frame is divided into slots that have a same amount of data to be processed. A set of MBs are assigned to a slot.

If there is a margin to make an operating frequency lower, the feedback mechanism selects a lower frequency on a slot-by-slot basis. The first slot is always processed at the maximum frequency. Since the elastic pipeline reduces the number of execution cycles in the pipeline processes, the first slot is potentially completed earlier. Then, assume that the second slot also reduced its execution cycles. Now, the third slot acquires the time margin, ΔH. Even considering a voltage/frequency transition time, the third slot has twice as long time as T_{slot} (processing time for a slot), which allows the third slot to be processed at an half of f_{max}. Note that a real-time operation is guaranteed in this feedback mechanism.

The power reduction factor depends on the number of slots prepared. If there are few slots prepared, there are few chances to lower an operating frequency and supply voltage, which in turn increase power. Alternatively, if there are many slots, there are many chances to lower them. However, it consumes much transition times since it takes a certain transition time to change the operating frequency and supply voltage. This makes substantial processing time shorter, and increase the power. Namely, there is the optimum number of slots per frame.

Figure 9 shows that the power reduction depends on the number of slots per frame. Note that, in this study, the transition time is assumed to be 50 μs [1] and the numbers of prepared frequency/voltage sets are two (f_{max} and f_{max} /2).

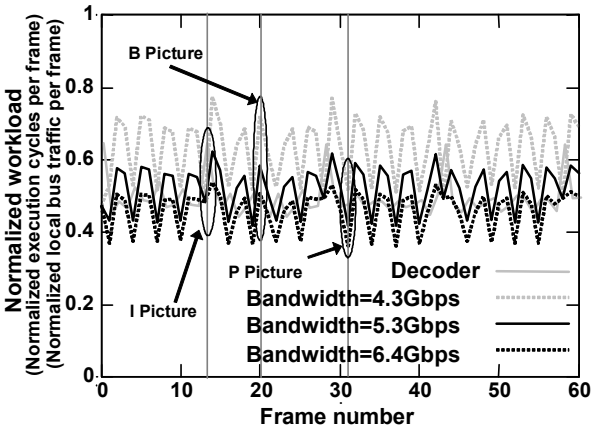

Figure 7. Cycle reduction of each frame in the proposed H.264/AVC decoder.

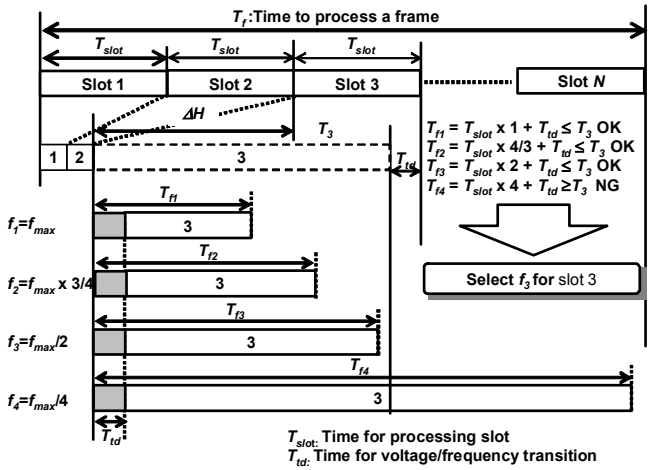

Figure 8. Feedback control mechanism.

The optimum number of slots per frame was obtained as 60 from the result of six video sequences. If the number of slots becomes smaller than the optimum number, the power reduction becomes drastically worse. Alternatively, if the number of slots becomes larger than the optimum number, the power reduction becomes gradually worsen.

Figures 10 (a) and (b) concretely show the relation of the number of slots and power reduction. The number of frequency transitions increases according to the number of slots per frame. In Fig. 10 (b), a terrible situation happens. Because many frequency transitions are occurred, the period of the maximum frequency occupies a half of the total processing time, in which case the power is not reduced very much.

Figure 11 demonstrates the power reduction in six video sequences, when the memory bandwidth is changed. Each sequence has 148 frames. The simulation result indicates the elastic pipeline can reduce the required bandwidth to 3.72 Gbps (while the conventional scheme needs 9.76 Gbps). If the decoder always operates at 108 MHz and a bandwidth of 3.72 Gbps or less, some frames can not be completed the decoding process within 1/30 second, especially in the case of the sequence "Soccer" that requires the largest amount of DRAM access among the six sequences.

The figure also shows the power reduction is saturated at 6

Gbps. If the bandwidth is set to 6 Gbps or more, the elastic pipeline often takes longer time (more execution cycles) than DRAM access cycles in most of the pipeline processes, and the local bus becomes busy since it can not catch up with the DRAM access. In this case, the power reduction is restricted by the elastic pipeline itself but the memory bandwidth. On the contrary, the bandwidth is set to less than 6 Gbps, DRAM access is frequently busy, and the pipeline idles. The proposed architecture in the worst case bandwidth of 10 Gbps reduces 44% power on average of the six sequences.

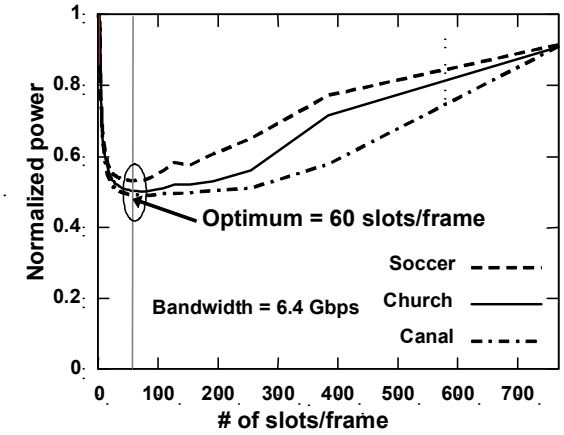

Figure 9. The Number of slots vs. power.

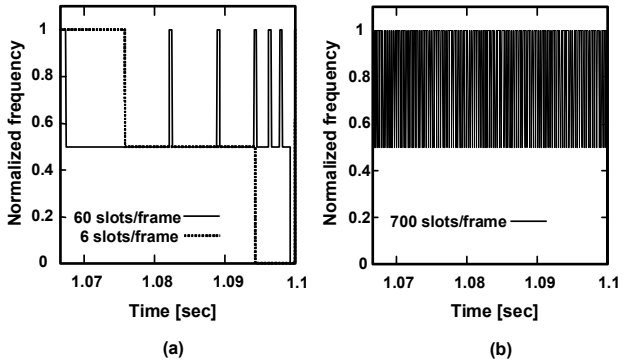

Figure 10. Operating frequency transitions, (a) adequate number of slots achieves low power, and (b) excessive number of slots degrades power reduction.

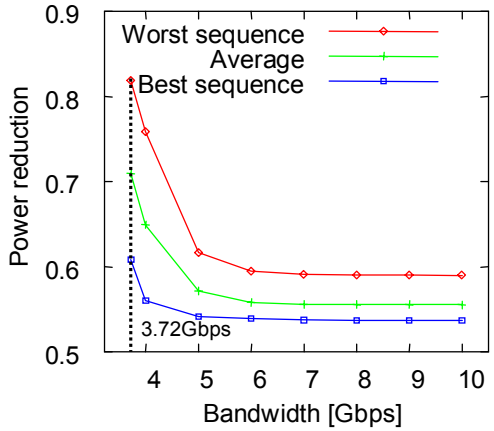

Figure 11. Power reduction ratio when memory bandwidth is varied. The polygonal lines indicate the best, average and worst power reductions in six video sequences (Canal, Church, Intersection, Japan room, Soccer and Whale show).

V. SUMMARY

We proposed the elastic pipeline architecture with which DVS can be applied to hardwired logic circuits, and designed a hardwired H.264/AVC HDTV decoder to verify its feasibility. The elastic pipeline architecture can reduce local bus bandwidth or execution cycles of pipeline process. The proposed H.264 HDTV decoder architecture reduces the bandwidth to 3.72 Gbps and the power to 74% in that case. If bandwidth is 6Gbps or more, the proposed architecture reduces the power to 56%.

References

[1] K. J. Nowka, G. D. Carpenter, E. W. MacDonald, H. C. Ngo, B. C. Brock, K. I. Ishii, Tuyet Y. Nguyen, and Jeffrey L. Burns, "A 32-bit PowerPC system-on-a-chip with support for dynamic voltage scaling and dynamic frequency scaling," IEEE J. Solid-State Circuits, vol.37, no.11, pp.1441-1447, Nov. 2002.

[2] K. Kawakami, M. Kanamori, Y. Morita, J. Takemura, M. Miyama, and M. Yoshimoto, "Power-minimum frequency/voltage cooperative management method for VLSI processor in leakage-dominant technology era," IEICE Trans. Fundamentals, vol.E88-A, no.12, pp.3290-3297 Dec. 2005.

[3] T. Sakurai, and A. R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," IEEE J. Solid-State Circuits, vol.25, no.2, pp.584-594, Apr. 1990.

[4] H.264/AVC reference software, http://iphome.hhi.de/suehring/tml/.

[5] S. Saponara, C. Blanch, K. Denolf, and J. Bormans, "The JVT advanced video coding standard: Complexity and performance analysis on a tool by tool basis", IEEE Packet Video 2003.

[6] W. Yu, Y. He, "A high performance CABAC decoding architecture," IEEE Trans. Consumer Electronics, vol. 51, no. 4, pp.1352- 1359, Nov. 2005.

[7] Joint Video Team (JVT) of ISO/IEC MPEG&ITU-T VCEG, "ISO/IEC 14496-10," May, 2003.

[8] Celoxica Handel-C, http://www.celoxica.com/.

[9] H. Kawaguchi, Y. Shin, and T. Sakurai, "μITRON-LP: power-conscious real-time OS based on cooperative voltage scaling for multimedia applications," IEEE Trans. Multimedia, vol.7, no.1, pp.67-74, Feb. 2005.