IEICE TRANS. FUNDAMENTALS, VOL.E83–A, NO.8 AUGUST 2000

PAPER *Special Section on Digital Signal Processing*

Motion Estimation with Power Scalability and Its VHDL Model

Ayuko TAKAGI[†], ***Student Member***, **Shogo MURAMATSU**[††], ***and*** **Hitoshi KIYA**[†], ***Regular Members***

SUMMARY In MPEG standard, motion estimation (ME) is used to eliminate the temporal redundancy of video frames. This ME is the most time-consuming task in the encoding of video sequences and is also the one using the most power. Using low-bit images can save power of ME and a conventional architecture fixed to a certain bit width is used for low-bit motion estimator. It is known that there is a trade-off between power and image quality. ME may be used in various situations, and the relation between demands for power or image quality will depend on those circumstances. We therefore developed an architecture for a low-bit motion estimator with adjustable power consumption. In this architecture, we can select the bit width for the input image and adjust the amount of power for ME. To evaluate its effectiveness, we designed the motion estimator by VHDL and used the synthesis results to estimate the performance.

key words: *low power, motion estimation, MPEG, less gray level image*

1. Introduction

In many multimedia applications, video data is often compressed according to the MPEG standard. There is considerable temporal redundancy in image sequences, and the MPEG codec uses motion estimation (ME) to eliminate the temporal redundancy of video frames. This ME is the most time-consuming task in the encoding of video sequences and is also the one using the most power. Many algorithms have been studied in efforts to reduce the complexity of ME and save power. In those algorithms, the quality of the encoded image is deteriorated in exchange to reduce the complexity and power. It is known that there is a trade-off between power and image quality. ME may be used in various situations, and the relation between demands for power or image quality will depend on those circumstances. For example, when running ME with a commercial power source there are no constraints on power availability and we will give more weight to image quality. But when running ME with a battery, power considerations can be more important than considerations of image quality. For instance, PCs with video cameras are used widely and the power considerations will change whether using indoors or outdoors. Balancing the needs for saving power and the needs for image quality, we must be able to select a suitable behavior for the motion estimator. We therefore developed an architecture for a low-bit motion estimator with adjustable power consumption.

Many algorithms have been studied in efforts to reduce the complexity of ME and there are two approaches. One is to reduce the number of search locations in ME and the other is to reduce the amount of computation per location. The latter is done by using images with far fewer bits/pixels. In this paper, we concentrate on the latter case, although we may implement the scalability in the former case. For an example of the former case, by controlling the search window size, we have to control the data flow of input image. Compared to the former case, the latter case does not need to control the data flow and the scalability can be implemented by controlling the bit width of input image. Also, these two approaches are independent from each other and these methods may be combined[1], [6]. This scalability could also be done using the combining method.

Many algorithms for making low-bit images have been proposed, and they differ on how to decide thresholds for deciding whether or not to transform 8-bit/pixel to data to data using fewer bits per pixel. [2]–[5]. A typical approach uses a linear quantization that truncates the least significant bits (LSB). Another approach is median-cut quantization which uses median values as thresholds for quantization [6], [7]. Here, we use a linear quantization to make low-bit images from their simplicity. Reducing the bit width of input image reduces power consumption, and the power required for ME is proportional to the bit width of the input image. To save power, it is necessary to use low-bit images. But in choosing the bit width, we must of course consider the trade-off between power and image quality.

In this paper, we developed an architecture for a low-bit motion estimator with adjustable power consumption. By selecting an appropriate bit width for the input image we can adjust the amount of power required for ME. To evaluate its effectiveness, we designed the motion estimator by VHDL and used the synthesis results to estimate the performance.

The organization of this report is as follows. In

Manuscript received December 2, 1999.
Manuscript revised March 1, 2000.

[†]The authors are with the Department of Electrical Engineering, Graduate School of Engineering, Tokyo Metropolitan University, Hachioji-shi, 192-0397 Japan.
[††]The author is with the Department of Electrical and Electronic Engineering, Faculty of Engineering, Niigata University, Niigata-shi, 950-2181 Japan.

Sect. 2, we briefly review the block-matching motion estimation with the method of using low-bit images and the conventional linear array. In Sect. 3, we discuss the architecture for low-bit ME and then we describe the architecture we developed. To show the significance, in Sect. 4, we estimate the performance from the synthesis result of the VHDL model, followed by the conclusion in Sect. 5.

2. Review

This section briefly reviews first the block-matching motion estimation and the method of using low-bit images. Then we briefly review the conventional linear array.

2.1 Block-Matching Motion Estimation

The MPEG codec is based on motion estimation, the process that takes most of the time needed for video encoding. There are several kinds of algorithms used for motion estimation, and block-matching algorithms are widely used in many kinds of video coding.

To see how these algorithms work, assume that a current frame is divided into blocks whose size is 16×16 called macro blocks. The process of block-matching is to find in a search window of previous frames the macro blocks most similar to the macro blocks in the current frame.

The accuracy of ME depends on the matching criteria, and one of the most popular criteria is the sum of absolute difference, or SAD, given by

$$SAD(k,l) = \sum_{i=1}^{16} \sum_{j=1}^{16} |P_t(i,j) - P_{t-1}(i+k, j+l)|, \quad (1)$$

where (k,l) is the location in the search window, $P_t(i,j)$ is a pixel at (i,j) in the current frame, and $P_{t-1}(i,j)$ is a pixel in the previous frame. When the value SAD(k,l) is minimum, (k,l) is the motion vector of the macro block.

A full-search block-matching algorithm exhaustively examines all the locations in a search window. This algorithm enables us to find the error-minimum block, but it requires a large amount of computation. Moreover, the algorithm is generally executed by using 8-bit images.

There are other algorithms reducing the complexity of ME, which reduce the number of search location. The architecture we developed is independent from these algorithms and it may be combined with them. For clarity here, however, describe its use with a full-search block-matching algorithm.

2.2 Motion Estimation Using Low-Bit Images

The procedure for motion estimation using low-bit images is shown in Fig. 1. The pixel data is quantized with the thresholds calculated for each macro block in the current frame. The images generated this way are used to calculate the motion vector implicit in the minimum solution Eq. (1). Using low-bit images reduces the calculation time required for motion estimation. To demonstrate our proposed architecture, we use linear quantization. Compared to other quantization methods, linear quantization has more image degradation but it is easy to implement on hardware architecture and makes it easy to evaluate the influence of using low-bit images.

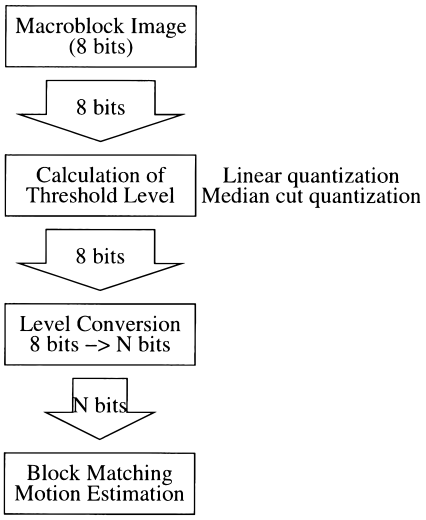


Fig. 1 Motion estimation using low-bit image.

2.3 Conventional Linear Array [8]

In this paper, we modify the conventional architecture partially. To illustrate the difference in our architecture, we briefly review the conventional linear array for motion estimator.

The block diagram of a motion estimator is shown in Fig. 2, where a is the data from a macro block, and $b0$, $b1$ are the data in two subregions of search window as shown in Fig. 3. The SAD is calculated in each of 16 processing elements (PE). Figure 4 shows the conventional architecture of a PE. Each data a from a macro block is send to the next PE by delayed flip-flop (DFF). The data flow for a PE is shown in Table 1. The first SADs in each PE (from PE0 to PE15) are output sequentially from 256 clock cycles to 271 clock cycles. After that, the SADs in each PE (from PE0 to PE15) are output sequentially every 256 clock cycles. The minimum SAD in a 16×16-block region is selected in the comparator module and the motion vector is given.

3. Proposed Architecture

In this section, we discuss the architecture for low-bit ME and describe the architecture we developed. To

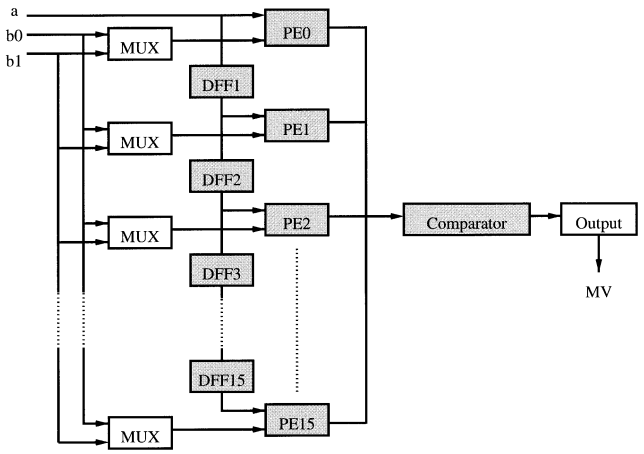


Fig. 2 Block diagram of the motion estimator for full-search block matching algorithm. DFF, PE, MUX stands for delayed flip-flop, processing element, and multiplexer.

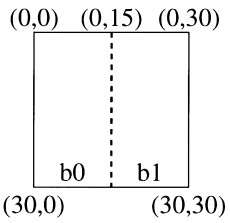


Fig. 3 Search window.

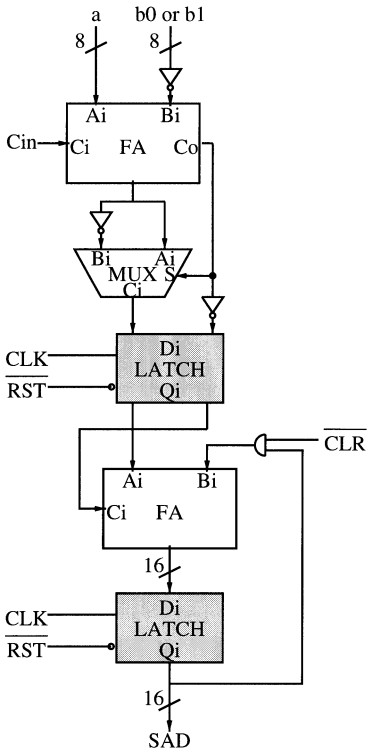


Fig. 4 The conventional internal architecture of PE.

Table 1 A data flow diagram for a full-search block matching motion estimation.

Clock cycle	Data sequences			PE0	PE1	PE2	.....	PE15
0	a(0,0)	b(0,0)		a(0,0)-b(0,0)				
1	a(0,1)	b(0,1)		a(0,1)-b(0,1)	a(0,0)-b(0,1)			
2	a(0,2)	b(0,2)		a(0,2)-b(0,2)	a(0,1)-b(0,2)	a(0,0)-b(0,2)		
⋮	⋮	⋮		⋮	⋮	⋮		
14	a(0,14)	b(0,14)		a(0,14)-b(0,14)	a(0,13)-b(0,14)	a(0,12)-b(0,14)		
15	a(0,15)	b(0,15)		a(0,15)-b(0,15)	a(0,14)-b(0,15)	a(0,13)-b(0,15)		a(0,0)-b(0,15)
16 + 0	a(1,0)	b(1,0)	b(0,16)	a(1,0)-b(1,0)	a(0,15)-b(0,16)	a(0,14)-b(0,16)		a(0,1)-b(0,16)
16 + 1	a(1,1)	b(1,1)	b(0,17)	a(1,1)-b(1,1)	a(1,0)-b(1,1)	a(0,15)-b(0,17)		a(0,2)-b(0,17)
⋮	⋮	⋮	⋮	⋮	⋮	⋮		⋮
16 +15	a(1,15)	b(1,15)	b(0,31)	a(1,15)-b(1,15)	a(1,14)-b(1,15)	a(1,13)-b(1,15)		a(1,0)-b(1,15)
2 x 16 + 0	a(2,0)	b(2,0)	b(1,16)	a(2,0)-b(2,0)	a(1,15)-b(1,16)	a(1,14)-b(1,16)		a(1,1)-b(1,16)
2 x 16 +1	a(2,1)	b(2,1)	b(1,17)	a(2,1)-b(2,1)	a(2,0)-b(2,1)	a(1,15)-b(1,17)		a(1,2)-b(1,17)
⋮	⋮	⋮	⋮	⋮	⋮	⋮		⋮
2 x 16 +15	a(2,15)	b(2,15)	b(1,31)	a(2,15)-b(2,15)	a(2,14)-b(2,15)	a(2,13)-b(2,15)		a(2,0)-b(2,15)
⋮	⋮	⋮	⋮	⋮	⋮	⋮		⋮
15 x 16 + 0	a(15,0)	b(15,0)	b(14,16)	a(15,0)-b(15,0)	a(14,15)-b(14,16)	a(14,14)-b(14,16)		a(14,1)-b(14,16)
15 x 16 + 1	a(15,1)	b(15,1)	b(14,17)	a(15,1)-b(15,1)	a(15,0)-b(15,1)	a(14,15)-b(14,17)		a(14,2)-b(14,17)
⋮	⋮	⋮	⋮	⋮	⋮	⋮		⋮
15 x 16 + 15	a(15,15)	b(15,15)	b(14,31)	a(15,15)-b(15,15)	a(15,14)-b(15,15)	a(15,13)-b(15,15)		a(15,0)-b(15,15)
16 x 16 + 0			b(15,16)		a(15,15)-b(15,16)	a(15,14)-b(15,16)		a(15,1)-b(15,16)
16 x 16 + 1			b(15,17)			a(15,15)-b(15,17)		a(15,2)-b(15,17)
⋮			⋮					⋮
16 x 16 + 15			a(15,30) b(15,31)					a(15,15)-b(15,30)

perform low-bit motion estimation, there are three different ways. One is to use the conventional motion estimator for 8 bit/pixel data and input the quantized low-bit data. The major component of the power dissipation is the switching power which is caused by the toggle of the signal. Using the low-bit data means to set the toggle rate of the non-used data to "0." From this reason, we can save power, but there are circuits not in use and it is redundant by means of power dissipation. The second one is to use the conventional motion estimator fixed for a certain bit width. This way, we can save power effectively but can not correspond to various bit width. The last is to use the motion estimator which corresponds with various bit width and

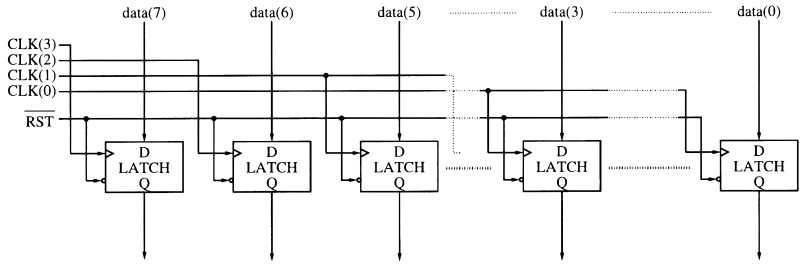


Fig. 5 Latch Module for variable bit width.

also deletes the redundancy of the power dissipation. The architecture we propose belongs to the last type. In this architecture, we can select an appropriate bit width for input image and adjust the amount of power required for ME.

3.1 Low-bit Motion Estimator

To perform low-bit ME, we generally need the input image to be quantized by a certain threshold in advance. Then the quantized low-bit data is processed by the conventional motion estimator for the specified bit width. This way we can save power but cannot run the ME on a different bit width. We assume here, however, that the appropriate bit width for the input image is different in different situations, and we describe a method fit for various bit widths.

In logic VLSI chips, most of the power is consumed by clock supplies for latches. In the motion estimator, PEs and Comparator have latches for which the bit width of the input data vary according to the bit width of the image. The DFFs also have input data of variable bit width. They are shown as gray boxes in Fig. 2 and Fig. 4. When a low-bit image is the input signal, the least significant bits (which corresponds to the truncated bits) are unnecessary and their processing wastes power. Here we describe a latch module dealing with variable bit width. With this module we can save power by cutting the specific clock supply to the latch according to the bit width of the input image.

3.2 Latch Module

We assume here that the input image can have 1-bit, 2-bit, 4-bit, and 8-bit bit widths. Instead of supplying the same clock signal, we use DFFs with different clock supplies. The input data is divided and stored in each of the latches, and they are synchronized with different clock signals: CLK(3), CLK(2), CLK(1), and CLK(0). Figure 5 shows an example of a latch module for 8-bit data. The 7th data bit is synchronized with CLK(3), the 6th data bit with CLK(2), the 5th and 4th data bits with CLK(1), and the 3rd, 2nd, 1st, and 0th data bits with CLK(0). For 16-bit data used in the accumulator inside a PE or in the comparator, the 15th through the 7th data bits are synchronized with CLK(3), the 6th data bit with CLK(2), the 5th and 4th data bits with CLK(1), and the 3rd through 0th data bits with CLK(0). According to the input bit width, we cut the clock supply for the corresponding latch and stop their action in order to reduce power.

To explain our method, we assumed that the input image can have four bit widths (8,4,2,1-bit). Please note that we may design this module with other bit widths, according to the desired environment. This example is shown in the simulation result.

3.3 General Architecture

In the motion estimator, latches at which there bit width change are used in each of the PEs and in the comparator. The DFFs also have input data with variable bit widths. They are shown in the gray boxes in Fig. 2 and Fig. 4. To make use of our proposed method, we replace those latches in each PE and Comparator with the latch module described in Sect. 3.2. We also replace the DFFs in Fig. 2 with the proposed architecture.

This proposed architecture is independent from the search algorithms. Therefore, it may be combined with any search algorithms that control search window size or reduce the number of search locations.

4. Performance Estimation

This section shows the significance of this new architecture by giving the estimated switching power. We estimated the performance of the following models: a standard motion estimator which the input bit width is 8-bit, a bit fixed motion estimator for which a certain bit width (4-bit, 2-bit, 1-bit) can be specified as the input bit width, and a motion estimator using the proposed architecture. We also designed the motion estimator corresponding to two bit width, 4-bit and 2-bit, using the proposed method.

4.1 PSNR Comparison

In our simulation, we used grayscale video sequences of a flower garden (QCIF size: 176 × 144 pixels). The codec we used is MPEG-2. The search window size is 31 × 31 and the GOP structure is IPPI [9]. The

Table 2 Estimation from the synthesis results of the conventional motion estimator and the proposed.

Architecture	Total cell area [μm^2]	Switching Power [mW]			
		1 bit/pixel	2 bit/pixel	4 bit/pixel	8 bit/pixel
Standard(8-bit)	2012989	72.6704	88.4672	123.9617	187.0301
Bit fixed(4-bit)	1327471	-	-	94.5026	-
Bit fixed(2-bit)	1020379	-	57.6781	-	-
Bit fixed(1-bit)	833291	39.4292	-	-	-
Proposed(8,4,2,1-bit)	2131717	45.9987	69.0458	120.8540	218.4220
Proposed(4,2-bit)	1339017	-	60.4503	109.4654	-

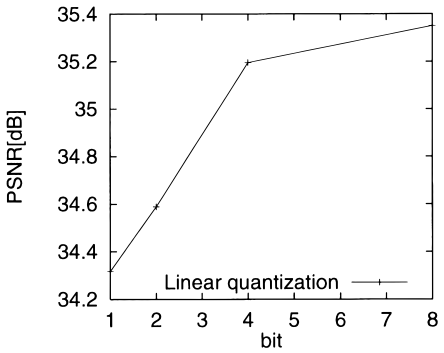


Fig. 6 PSNR comparison for low-bit motion estimation.

video data is 8-bit/pixel with no signs. Here we reduced the 8-bit/pixel data to 1-, 2-, or 4-bit/pixel data by linear quantization and simulated its processing at 1 Mbps. For motion estimation we used a full-search block-matching algorithm. Figure 6 shows the PSNR comparison of encoded image. The image quality deteriorates when data with fewer bits per pixel is used.

4.2 Estimation from Synthesis Results

Let $F_h \times F_v$ be the number of pixels per frame, F_t be the number of frames per second (fps), N be the number of pixels in a macro block, and T be the number of clock cycles for the ME of every macro block. Then the following condition for clock period t_{clk} is derived:

$$t_{clk} = \frac{N}{F_h \times F_v \times F_t \times T}[\text{sec}]. \quad (2)$$

The Quarter CIF (QCIF) is specified by $F_h \times F_v = 176 \times 144$ pixels, $F_t = 30$ fps, $N = 256$ pixels. The linear array ME requires 4096 cycles to complete the operations for a macro block so $T = 4096$. We estimate the performance in units of QCIF when CLK was constrained to $t_{clk} = 80$ ns.

To estimate the power consumption, we modeled the proposed architecture by VHDL as an edge-triggered synchronous system. The design has been synthesized by using the Synopsys design tools ver. 1998.02 [10] with the linear model of the standard cell library EXDLIB provided by VDEC for 0.5-μm CMOS technology [11]. The environments are as follows:

- No driving cell is set to CLK. Inverters are set to other inputs as driving cells.
- An inverter is set to each output as load.
- The operating condition is set to "WCCOM."
- The wire load is set to "20 × 20."

"WCCOM" stands for Worst Case COMmercial. This is the worst operating condition for commercial use which is defined in EXDLIB.

The constraints are as follows:

- CLK is constrained to $t_{clk} = 80$ [nsec]
- Area is not constrained.

Switching power is the major component of overall power dissipation, and in the work reported in this paper we estimates only the switching power. We annotate the switching information from the input image (1-bit/pixel, 2-bit/pixel, 4-bit/pixel, 8-bit/pixel), which was calculated from software model of ME. The synthesis results are listed in Table 2.

The standard architecture(8-bit) is the conventional architecture for motion estimation. Table 2 shows the switching power when using the standard 8 bit/pixel image and three kinds of low-bit images (4-bit/pixel, 2-bit/pixel, 1-bit/pixel). From the result of standard architecture (8-bit), we can see that it is effective on power to use low-bit images as an input. This is the most simple way to implement the power scalability, but there are circuits not in use which consume power. From this reason, the power reduction is small. The bit fixed architecture (4-bit, 2-bit, 1-bit) has no redundancies in the circuit and the reduction of power is the most. But these architectures can not implement the scalability because the input bit widths are fixed. The proposed architecture removes the redundancies of power dissipation from the standard architecture (8-bit) and we can save more switching power compared to the standard motion estimator. The power of the proposed architecture is increased compared to the bit fixed architecture (4-bit, 2-bit, 1-bit). It is because there are circuits for truncating the low-bit data and the signals are increased. We can see this effect in the increase of the total cell area. Moreover, we showed the simulation of the proposed architecture corresponding to 4-bit and 2-bit. Compared to the proposed architecture with 8,4,2,1-bit, the switching power is more reduced and the total cell area and its power is becoming closer to the bit fixed architecture (4-bit) By selecting

other bit widths like 4-bit and 1-bit, the total cell area and its switching power will be more close. The designer can select which bit widths to use according to the environment of the practical use.

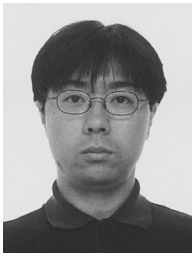
5. Conclusion

In this paper we described an architecture for low-bit motion estimation with power scalability. By cutting the clock supply to latches, we prevent the latch from expending power needlessly when using low-bit images as the input signal source. By selecting which clock supply to cut, we can control the power consumption of the motion estimator. This way we can select the ME behaviors appropriate to various situations. We estimated switching power from synthesis results and showed the effectiveness of our architecture. For the future work, we would like to research on implementing power scalability using other algorithms such as controlling the search window size and combining them with our proposed method.

References

[1] Y. Sasajima and T. Enomoto, "Snapping-off motion vector estimation algorithm using reduced pixel data," Proc. Society Conf. IEICE, vol.D-11-2, p.117, Sept. 1998.
[2] S. Lee, J.M. Kim, and S.I. Chae, "New motion estimation algorithm using adaptively quantized low bit-resolution image and its VLSI architecture for MPEG2 video encoding," IEEE Trans. Circuits & Syst. for Video Technol., vol.8, pp.734–744, Oct. 1998.
[3] J. Feng, K.-T. Lo, H. Mehrpour, and A.E. Karbowiak, "Adaptive block matching motion estimation algorithm using bit-plane matching," IEEE Int. Conf. Image Proc., pp.496–499, Washington, D.C., 1995.
[4] B. Natarajan, B. Vasudev, and K. Konstantinides, "Low-complexity algorithm for motion estimation and architecture for block-based motion estimation via one-bit transform," IEEE ICASSP'96, pp.3244–3247, 1996.
[5] L. Jiang, K. Ito, and H. Kunieda, "Bits truncation adaptive algorithm for motion estimation of MPEG2," IEICE Trans. Fundamentals, vol.E80-A, no.8, pp.1438–1445, Aug. 1996.
[6] H. Kiya, J. Furukawa, and Y. Noguchi, "Block matching motion estimation based on median cut quantization for MPEG video," IEICE Trans. Fundamentals, vol.E82-A, no.6, pp.899–904, June 1999.
[7] S. Muramatsu, H. Kiya, and A. Yamada, "A bit-operation algorithm of the median-cut quantization and its hardware architecture," IEICE Trans. Fundamentals, vol.E83-A, no.2, pp.320–328, Feb. 2000.
[8] H. Fujiwara, Y. Okada, T. Kobayashi, H. Uwabu, and M. Maruyama, "Research on specific architecture of video coding for multi-media information," IPSJ Trans., vol.35 no.7, pp.1422–1437, July 1994.
[9] MPEG Software Simulation Group, "MPEG-2 encoder/ decoder version 1.2." http://www.mpeg.org/MPEG/MSSG
[10] Synopsys, Inc., "Synopsys Online Documentation version 1998.02," CA, USA, 1998.
[11] K. Shiomi, K. Okino, T. Kawasaki, T. Ishihara, and H. Yasuura, "Development of a standard cell library for VDEC," IEICE Technical Report, CAS97-31, VLD97-31, DSP97-46, 1997.

Ayuko Takagi was born in Tokyo, Japan, on March 15, 1976. She received the B.E. and M.E. degrees in electronics and information engineering from Tokyo Metropolitan University in 1998 and 2000. She is currently a candidate for the D.E. degree at Tokyo Metropolitan University. Her research interest is in image processing.



Shogo Muramatsu was born in Tokyo, Japan, on May 2, 1970. He received the B.E., M.E. and D.E. degrees in electrical engineering from Tokyo Metropolitan University in 1993, 1995 and 1998, respectively. In 1997, he joined Tokyo Metropolitan University. Then, in 1999, he joined Niigata University, where he is currently a research associate of Electrical and Electronic Engineering, Faculty of Engineering. His research interests are in digital signal processing, multirate systems, image processing and VLSI architecture. Dr. Muramatsu is a Member of the Institute of Electrical and Electronics Engineers (IEEE) of USA.

Hitoshi Kiya was born in Yamagata, Japan, on November 16, 1957. He received the B.E. and M.E. degrees in electrical engineering from Nagaoka University of Technology, Niigata, Japan, and the D.E. degree in electrical engineering from Tokyo Metropolitan University, Tokyo, Japan, in 1980, 1982, and 1987, respectively. In 1982, he joined Tokyo Metropolitan University, where he is currently a Professor of Electrical Engineering, Graduate School of Engineering. He was a visiting researcher of the University of Sydney in Australia from Oct. 1995 to March 1996. His research interests are in digital signal processing, multirate systems, adaptive filtering, image processing, and efficient algorithms for VLSI implementation. He is an Associate Editor of Trans. on Signal Processing of IEEE and Trans. of IEICE. He is a Member of the IEEE, the Image Electronics Engineers of Japan and the Institute of Television Engineers of Japan.