PAPER *Special Section on VLSI Design and CAD Algorithms*

System-MSPA Design of H.263+ Video Encoder/Decoder LSI for Videotelephony Applications

Chawalit HONSAWEK[†a)], *Nonmember*, **Kazuhito ITO**[††], **Tomohiko OHTSUKA**[†††], *Regular Members*, **Trio ADIONO**[†], *Nonmember*, **Dongju LI**[†], **Tsuyoshi ISSHIKI**[†], *and* **Hiroaki KUNIEDA**[†], *Regular Members*

SUMMARY In this paper, a LSI design for video encoder and decoder for H.263+ video compression is presented. LSI operates under clock frequency of 27 MHz to compress QCIF (176×144 pixels) at the frame rate of 30 frame per second. The core size is $4.6 \times 4.6\,\text{mm}^2$ in a $0.35\,\mu\text{m}$ process. The architecture is based on bus connected heterogeneous dedicated modules, named as System-MSPA architecture. It employs the fast and small-chip-area dedicated modules in lower level and controls them by employing the slow and flexible programmable device and an external DRAM. Design results in success to achieve real time encoder in quite compact size without losing flexibility and expand ability. Real time emulation and easy test capability with external PC is also implemented.

key words: *system-MSPA, LSI design, videotelephony applications, H.263+ ITU standard*

1. Introduction

VLSI architecture for video signal processing may be classified and specified as homogeneous and heterogeneous processor architectures. Heterogeneous processors outperform homogeneous processors because of adaptation to the requirements of special subtasks by dedicated modules for high performance. The heterogeneous processor architecture is suitable in high performance aspect for specific applications such as low bit rate video coding. Many chips [1]–[5] have been presented based on these architectures. However, they suffer from complicated and inflexible design.

In this paper, excellent design of both video encoder and decoder for H.263+ videotelephony applications is presented. Each subtask of algorithm is designed as function specific dedicated module, which leads to high performances in terms of speed and silicon area. Especially, extremely fast motion estimator based on parallel array processors is employed, whose task usually takes the longest execution time. Each module is restricted to communicate with each other only via an external memory. The operation and I/O of dedicated modules are controlled by programmable Address Generation Unit (AGU). In this way, LSI becomes quite flexible to add and delete module for development and it becomes suitable for test and independent module design. In addition, the chips can operate at low clock frequency due to their function specific dedicated modules. Consequently, it achieves low power consumption and also easily design under low clock frequency.

We have developed MSPA [6], [7] and Window-MSPA [8]–[10] array architecture. MSPA architecture is used to deal with regular algorithm, such as matrix multiplication. While, Window-MSPA is used to deal with window operations in image processing, such as motion vector searching. In comparison with conventional methods, both MSPA and Window-MSPA achieved higher performance in term of operation-level parallel processing. Here, we name our architecture as System-MSPA architecture, which stands for the system level architecture of MSPA series of our processing arrays. The architecture is extremely optimized at the expense of memory bandwidth, which is much wider than the required one. This is proven to be the most effective architecture for H.263+ video codec LSI.

In Sect. 2, the System-MSPA architecture and dedicated modules are described. The chips are also implemented with test function which is described in Sect. 3. The LSI results and conclusion are described in Sects. 4 and 5, respectively.

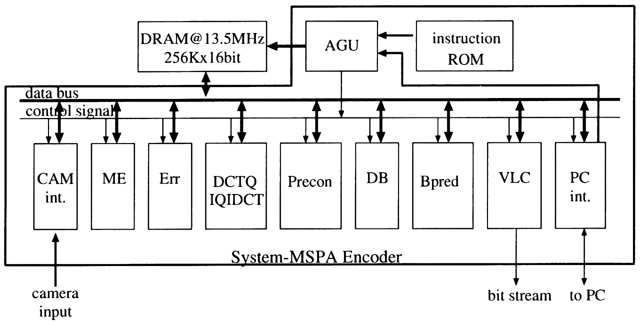
2. System-MSPA Architecture

The architecture, named as System-MSPA (Memory Sharing Processor Array) consists of dedicated modules, data bus, Address Generation Unit (AGU) and external DRAM memory. A block diagram of video encoder is shown in Fig. 1.

Each dedicated module works independently only with data exchanges with external DRAM memory. Its operation is controlled by program instructions of AGUs. In this way, System-MSPA achieves both fast speed and flexibility. Pipeline operations of modules can be managed by this architecture.

Manuscript received February 26, 2001.
Manuscript revised June 1, 2001.

[†]The authors are with the Department of Communications and Integrated Systems, Tokyo Institute of Technology, Tokyo, 152-8552 Japan.
[††]The author is with the Department of Electrical and Electronic Systems Engineering, Saitama University, Urawa-shi, 338-8570 Japan.
[†††]The author is with the Department of Electronic Engineering, Tokyo National College of Technology, Hachioji-shi, 193-0997 Japan.
a) E-mail: chawalit@vlsi.ss.tiech.ac.jp



CAM int.: Camera interface
ME: Motion estimation
Err: Error coding
DCTQ: Discrete cosine transform & quantization
IQIDCT: Inverse quantization & inverse discrete cosine transform
Precon: P macroblock reconstruction
Bpred: Prediction mode decision
DB: Deblocking filter
VLC: Variable length coding
PC int.: PC host interface

Fig. 1 System-MSPA encoder.

Table 1 Operation clock requirement and memory access for each module in encoder.

Module	operation clock		memory access
	1st stage	2nd stage	(clock)
Camera int.			1500
ME	12000		1100
Err		2300	2300
DCT-Q		1920	1920
IQ-IDCT		1920	1920
Precon		2430	2430
DB		1680	1680
Bpred		1870	1870
VLC			390
Total	12000	12120	15110
Budget	**15625**		

In this section, scheduling and pipelining for real time processing and circuit design of each module are illustrated.

2.1 Memory Bandwidth

For videotelephony applications with image size of 176×144 (QCIF) at 30 frames/s, the maximum budget of one PB-macro block processing is 18,180 clock cycles at operating frequency of 27 MHz. Since the image size is small, slow external DRAM with small bandwidth of 216 Mbit/sec (13.5 MHz 256 K × 16 bit) is sufficient in our system. This fact enables power dissipation lower. Table 1 shows the memory access within a budget of 15,625 clock cycles for each module in video encoder.

2.2 Scheduling and Pipelining

Operation scheduling of modules are controlled by instruction set of AGU, by setting specific register values. Each module accepts data of a 16 pixel × 16 pixel macro block from external DRAM, executes processing and produces output data to external DRAM. Therefore, AGU issues starting signals for each block, one by one and generates addresses of the external DRAM for input and output of each module at appropriate timing. The memory access timing of each module is decided by instruction program within a budget of 15,625 clocks to process each macro block.

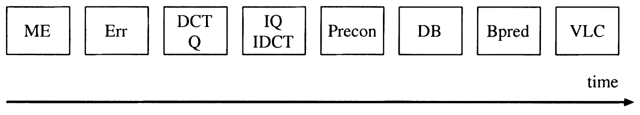


Fig. 2 Memory access for video encoder.

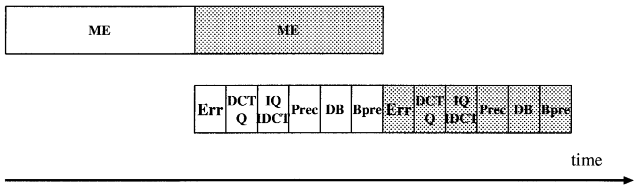


Fig. 3 Operation scheduling for video encoder.

The order of sequential memory access is shown in Fig. 2. Since camera interface is a line (of picture) level processing, the time slot for memory access of camera interface does not match to macro block level processing. The camera interface accesses memory in between memory access of any two modules only when video input data are available.

The operation scheme is restricted by the sequential access of modules to DRAM. The pipelining of MSPA allows the slower operation time of motion estimation (ME) with the reduced number of processor elements in the module. We adopt two stage pipeline operation as shown in Fig. 3. The ME module inputs and outputs data in small number of clocks, but its execution takes 12,000 clocks. The other modules operate in sequential within 12,120 total clocks. Table 1 shows the operation clock requirement for each module in video encoder.

2.3 Dedicated Modules

All modules are designed dedicatedly for a specific task, which leads to high speed operation and small area occupancy in chip. They are designed by VHDL language and are synthesized with Synopsys Design Compiler. In such a sense, modules are quite flexible against different applications or are able to be replaced with commercially available IP (Intellectual Property) modules.

2.3.1 Address Generation Unit (AGU)

For video encoder application, dedicated memory access is required, because of its macro block base processing. Special repetitive instructions with special memory access in AGU allows normal DRAM to effectively store several frame data. For example, one instruction can manage to transfer even frame data from DRAM to modules.

According to the block processing, external memory spaces are hierarchically divided into 4 frame (12 × 11 macro blocks), macro block (4 Y-blocks and Cr, Cb block), and block (8 × 8 pixels) data, each of which includes 8 bit data.

AGU generates pixel addresses in a macro block of external DRAM and send timing signals to each module for the module to read and write data from/to DRAM. This address generation is executed according to AGU instructions written in AGU ROM. Instructions are classified into three kinds, which handle

1. block read/write addressing instructions
2. subroutine, branch and register control instructions
3. PC test instructions.

The first kind of instructions activate loop instruction for memory access with a few instructions. In special case, one instruction can address read or write of one frame data.

This AGU functions can be replaced with normal CPU cores such as RISC processors. However, we do not require any arithmetic/logic functions of data to AGU. Instead, AGU instructions are specialized so as to read/write hierarchical video data smoothly from/to DRAM.

2.3.2 Motion Estimation (ME) with Window-MSPA Architecture

The function of the motion estimation is to find the most similar macro block in the previous picture to the currently coded one. To achieve this, the motion estimator compares the current macro block with all possible candidate macro blocks of the previous picture according to a cost function, called the distortion. This distortion used here is the sum of the absolute value of the pixel to pixel differences.

The motion estimation can be performed in two stages. At the first stage, it computes a motion vector with integer pixel resolution. At the second stage, it refines the resolution to a half pixel level. The second stage needs to perform 8 additional searches, corresponding to all possible horizontal and vertical displacements by −0.5, 0, and 0.5 pixels. Figure 4 shows a block diagram of a motion estimator. The distortion is calculated by half pixel motion estimation according to half pixel data from interpolator and then compare to the minimum distortion from the first stage. Half pixel resolution motion estimation can be designed by using array processors. However, in our case, one processor is enough to satisfy execution time budget. Half pixel processor also calculates the sum of current macro block which is used to compare the distortion to select macro block type MBTYPE (INTRA/INTER mode)

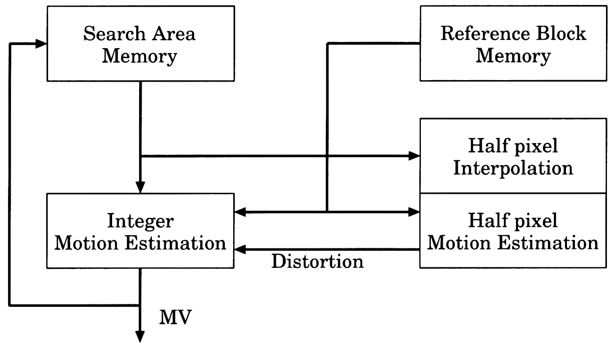


Fig. 4 Block diagram of motion estimation.

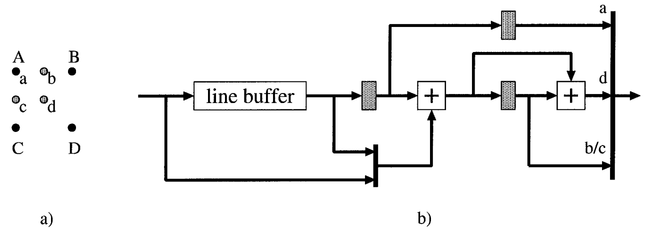


Fig. 5 Circuit of half pixel interpolation.

for coding. For half pixel interpolation, Fig. 5 shows a block diagram of a circuit that can calculate all types of pixel position with two adders.

The full search motion estimation of 16×16 macro block with search range of [−16,15] and integer pixel resolution is designed by using Window-MSPA array architecture. Window-MSPA (Memory Sharing Processor Array for window operation) is a processor array architecture developed for general window operations in image processing [8]–[10]. It takes the I/O relationship into account at scheduling, which makes it efficiently convey data into processor array by using data converters. Original Window-MSPA is developed for general repetitive window operations over the area in image data space. We call a part of image area for processing as window. It has several excellent features such as the minimum number of I/O ports, high parallel efficiency, small temporal data storages, free to select the number of processing elements (PEs). The architecture is flexible enough to satisfy our current requirement for both area and speed.

Usually, Window-MSPA architecture can be achieved by following normal, area-small, and speed-fast, three kind of design flows. The first two design formulas are given in Table 2 [10]. In block matching search, window size $K \times K$ becomes block size. The search area $N \times N$, and search range P where $P = N - K$. In the table, N_I indicates the number of input ports to array, N_R and N_C indicate the number of columns and rows of processor elements in Window-MSPA array respectively, and T_P denotes the computation time of window operations. The n is any divider

Table 2 Design formula.

	Normal Design	Area-Small Design
N_I	$\lceil \frac{N}{K} \rceil$	$\min\left\{K, \left\lceil \frac{\frac{P+1}{n}+K-1}{K} \right\rceil\right\}$
N_R	$P+1$	$\frac{P+1}{n}$
N_C	$\min\{P+1, max\{K, \lceil \frac{N}{K} \rceil\}\}$	1
T_P	$K \times N$	$\{K^2(P+1)\}n$
E_p	$\frac{K^2(P+1)^2}{T_P N_R N_C}$	$\frac{K^2(P+1)^2}{T_P N_R}$

Table 3 Parameter selection for $N = 47$, $K = 16$, $P = 31$.

	Normal Design	Area-Small Design
N_I	3	3
N_R	32	32
N_C	16	1
T_P	752	8192
E_P	68%	100%

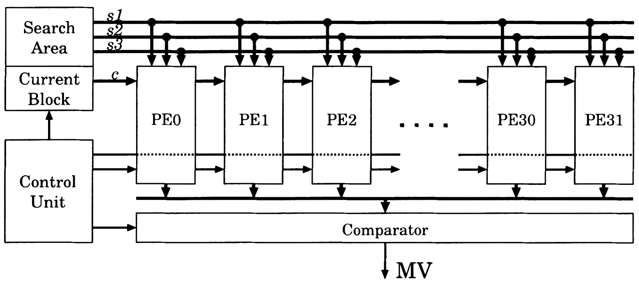


Fig. 6 Window-MSPA architecture motion estimation.

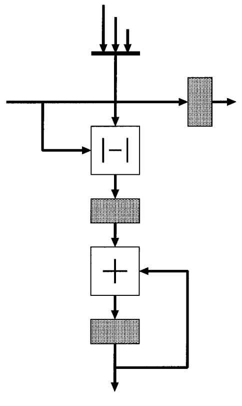


Fig. 7 Circuit inside processing element (PE).

of $P + 1$. E_P denotes parallel efficiency.

Table 3 gives the parameter selection for $N = 47, K = 16$, and $P = 31$. In this case, we adopt 32 PEs and 3 input ports for search area data inputs with execution time of 8192 clock cycles and 100% PE utilization.

Figures 6 and 7 show 32 PEs array processor motion estimation based on Window-MSPA architecture and logic inside PE. Three ports ($s1, s2, s3$) of search area data are distributed to all PEs that select the appropriate input to compute a sum of absolute difference (SAD). SAD values are generated sequentially to a comparator every 256 clock cycles to find a motion vector. Table 4 shows pixel flow for computing SAD values.

2.3.3 Discrete Cosine Transform (DCT)

Since conventional DCT circuits require many multipliers, they are very large and difficult to be implemented. Here, it is implemented by using distributed arithmetic architecture [11]. Distributed arithmetic architecture has a regular structure, simple control and interconnect, and achieve a good balance between performance and complexity of implementation.

Figure 8 shows a block diagram of DCT based on distributed arithmetic architecture. One line of block (eight data vector) is firstly loaded into the input registers. Then data are processed with bit serial data format by bit serial adders or substractors to generate two buses which distribute the bits into eight ROM accumulator units (RAC). Each RAC consists of two (for DCT and IDCT) 16 words of ROM and accumulator. Figure 9 shows circuit inside RAC. We employ 16 bits of precision for input and output to comply with the accuracy requirement specified by the standard. Therefore, one line of block can be calculated 1D-DCT in parallel within 16 clock cycles. The output data from RAC units are then loaded into the output register and then transferred to memory.

2.3.4 Variable Length Coding/Decoding (VLC/VLD)

The tasks of the VLC module are encoding picture headers and macro blocks, and the rate control. The VLC analyzes the received DCT coefficients and determine macro block type & code block pattern for chrominance (MCBPC) and code block pattern for luminance (CBPY). Based on the previously used and the received quantization (Q) steps, the previous and the received motion vectors (MVs), quantizer information (DQUANT) and motion vector data (MVD) are computed. Then, DCT coefficients in addition to MCBPC, CBPY, DQUANT, MVD, and so on, are encoded with variable length by using look-up tables.

The encoded bits are stored in a bit stream buffer. The size of the buffer is decided by the rate control strategy. The rate control consists of two layers; picture layer and macro block layer. At picture layer, based on the number of bits remaining in the bit stream buffer, the target number of bits (TARGET) for the next PB frame is determined so that half a PICBITS number of bits always remains in the bit stream buffer to avoid buffer underflow. In addition, the initial Q step is determined by the rate control in picture layer. If TARGET is much larger than the actual number of bits of the encoded previous PB frame (ACTBITS), Q step is set smaller than the average of Q steps used to encode the previous PB frame (Q_{AVE}). If TARGET is

Table 4 Pixel flow for computing sum of absolute difference.

Cycle	Input Data				Processor Input				
Time	c	$s1$	$s2$	$s3$	PE0	PE1	PE15	PE16	PE31
0	$c_{0,0}$	$s_{0,0}$			$c_{0,0}, s_{0,0}$				
1	$c_{0,1}$	$s_{0,1}$			$c_{0,1}, s_{0,1}$	$c_{0,0}, s_{0,1}$			
2	$c_{0,2}$	$s_{0,2}$			$c_{0,2}, s_{0,2}$	$c_{0,1}, s_{0,2}$			
15	$c_{0,15}$	$s_{0,15}$			$c_{0,15}, s_{0,15}$	$c_{0,14}, s_{0,15}$	$c_{0,0}, s_{0,15}$		
16	$c_{1,0}$	$s_{0,16}$	$s_{1,0}$		$c_{1,0}, s_{1,0}$	$c_{0,15}, s_{0,16}$	$c_{0,1}, s_{0,16}$	$c_{0,0}, s_{0,16}$	
17	$c_{1,1}$	$s_{0,17}$	$s_{1,1}$		$c_{1,1}, s_{1,1}$	$c_{1,0}, s_{1,1}$	$c_{0,2}, s_{0,17}$	$c_{0,1}, s_{0,17}$	
31	$c_{1,15}$	$s_{0,31}$	$s_{1,15}$		$c_{1,15}, s_{1,15}$	$c_{1,14}, s_{1,15}$	$c_{1,0}, s_{1,15}$	$c_{0,15}, s_{0,31}$	$c_{0,0}, s_{0,31}$
32	$c_{2,0}$	$s_{0,32}$	$s_{1,16}$	$s_{2,0}$	$c_{2,0}, s_{2,0}$	$c_{1,15}, s_{1,16}$	$c_{1,1}, s_{1,16}$	$c_{1,0}, s_{1,16}$	$c_{0,1}, s_{0,32}$
33	$c_{2,1}$	$s_{0,33}$	$s_{1,17}$	$s_{2,1}$	$c_{2,1}, s_{2,1}$	$c_{2,0}, s_{2,1}$	$c_{1,2}, s_{1,17}$	$c_{1,1}, s_{1,17}$	$c_{0,2}, s_{0,33}$
47	$c_{2,15}$		$s_{1,31}$	$s_{2,15}$	$c_{2,15}, s_{2,15}$	$c_{2,14}, s_{2,15}$	$c_{2,0}, s_{2,15}$	$c_{1,15}, s_{1,31}$	$c_{1,0}, s_{1,31}$
48	$c_{3,0}$	$s_{3,0}$	$s_{1,32}$	$s_{2,16}$	$c_{3,0}, s_{3,0}$	$c_{2,15}, s_{2,16}$	$c_{2,1}, s_{2,16}$	$c_{2,0}, s_{2,16}$	$c_{1,1}, s_{1,32}$
49	$c_{3,1}$	$s_{3,1}$	$s_{1,33}$	$s_{2,17}$	$c_{3,1}, s_{3,1}$	$c_{3,0}, s_{3,1}$	$c_{2,2}, s_{2,17}$	$c_{2,1}, s_{2,17}$	$c_{1,2}, s_{1,33}$
255	$c_{15,15}$	$s_{15,15}$		$s_{14,31}$	$c_{15,15}, s_{15,15}$	$c_{15,14}, s_{15,15}$	$c_{15,0}, s_{15,15}$	$c_{14,15}, s_{14,31}$	$c_{14,0}, s_{14,31}$
256	$c_{0,0}$	$s_{15,16}$	$s_{1,0}$	$s_{14,32}$	$c_{0,0}, s_{1,0}$	$c_{15,15}, s_{15,16}$	$c_{15,1}, s_{15,16}$	$c_{15,0}, s_{15,16}$	$c_{14,1}, s_{14,32}$
257	$c_{0,1}$	$s_{15,17}$	$s_{1,1}$	$s_{14,33}$	$c_{0,1}, s_{1,1}$	$c_{0,0}, s_{1,1}$	$c_{15,2}, s_{15,17}$	$c_{15,1}, s_{15,17}$	$c_{14,2}, s_{14,33}$
286	$c_{1,14}$	$s_{15,46}$	$s_{1,30}$	$s_{2,14}$	$c_{1,14}, s_{2,14}$	$c_{1,13}, s_{2,14}$	$c_{0,15}, s_{1,30}$	$c_{0,14}, s_{1,30}$	$c_{15,15}, s_{15,46}$
287	$c_{1,15}$		$s_{1,31}$	$s_{2,15}$	$c_{1,15}, s_{2,15}$	$c_{1,14}, s_{2,15}$	$c_{1,0}, s_{2,15}$	$c_{0,15}, s_{1,31}$	$c_{0,0}, s_{1,31}$

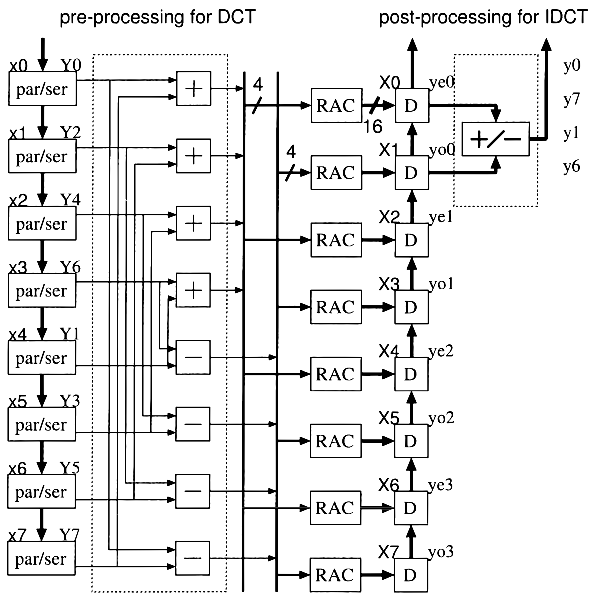


Fig. 8 Block diagram of DCT (also perform as IDCT).

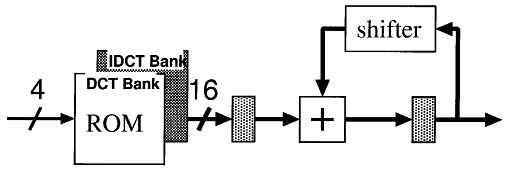


Fig. 9 Circuit inside ROM accumulator.

much smaller than ACTBITS, Q step is set larger than Q_{AVE}. The above mentioned rate control method is very simple and works in real time.

In H.263 standard [12], variable length codes (VLCs) are used for MCBPC, CBPY, MVs, transform coefficient (TCOEF), and so on. Although an efficient method to decode VLC by a processor is proposed, we adopt to decode these VLC by combinatorial circuits because each of VLCs consists of rather a small number of codes. TCOEF consists of the largest number of codes but it is only 102 and the gate count of the TCOEF decoding circuits is about 950. The VLC decoding circuit inputs the head of the bit stream data. If the matching code exists, the circuit outputs the decoded value (e.g. MV data in the case of motion vector decoding) and the decoded code length. In other words, the VLC decoding circuit is a read-only variable length content addressable memory (CAM) where the contents are the H.263 VLCs. We use a CAM for each VLC sets of MCBPC, CBPY, MVs, TCOEF, and so on. In order to efficiently communicate with the CAMs and maximize their usage, all the VLD circuit is designed as a finite state machine made of random logic. By using CAM and the finite state machine, each VLC is decoded in two clocks, one for VLC matching by CAMs and other for retrieving the decoded code from the bit stream buffer. This results in less number of clocks required to decode the bit stream data if compared with a processor style decoder. The small number of clocks enables us to use the clock of low frequency. It will lead to low power consumption, which would be desirable in mobile applications.

The output of the VLD is the picture type, macro block type, MV, quantization (Q) step to re-quantize DCT coefficients, and all the DCT coefficient for 6 blocks (4 luminance blocks and 2 chrominance blocks) in the case of Intra or P picture, or 12 blocks (8 luminance blocks and 4 chrominance blocks) in the case of PB or Improved PB frames mode. These data are saved in the shared memory at the locations specified by the master controller and used by other video decoding processes.

2.3.5 P Frame Reconstruction (Precon)

P frame reconstruction is to reconstruct the image from

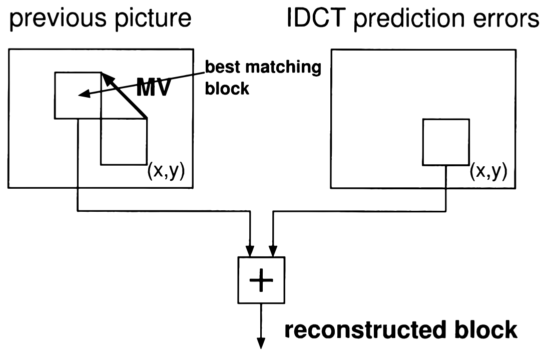


Fig. 10 Image reconstruction.

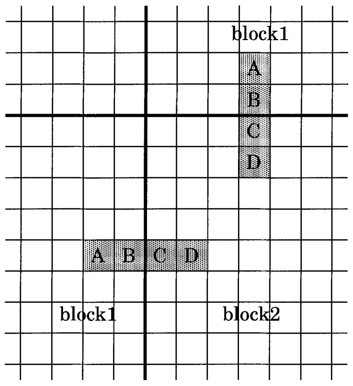


Fig. 11 Positions of filtered pixels.

the previous picture and IDCT prediction error signals as shown in Fig. 10. Since motion vector is half pixel resolution, motion vector with truncation to integer pixel resolution is used to indicate the best matching block in the previous picture. This block is extended to 9×9 pixels in order to deal with half pixel interpolation. Each of these block in the previous picture and prediction errors fetches the image data from main memory line by line. After it finishes computation, one line data (8 pixels) of reconstructed image block are then written back to main memory. By this way, it does not need local memory to store a block of image data. Although it decreases memory bandwidth efficiency due to data processing overhead, memory bandwidth is not major problem in our system.

2.3.6 Deblocking Filter (DB)

Deblocking filter is one option provided by H.263+ which reduces block noise significantly. Deblocking filter is applied on boundaries of 8×8 reconstructed block as shown in Fig. 11. It is applied at first on horizontal edges, then on vertical edges between *block*1 and *block*2. Two filters perform on 16 bit of two neighbor pixel data from main memory. Therefore, two positions of filters pixels are calculated in parallel. The results, which are adjusted by appropriate value in a function of quantization (Q) step and their value (A, B, C, D), are then

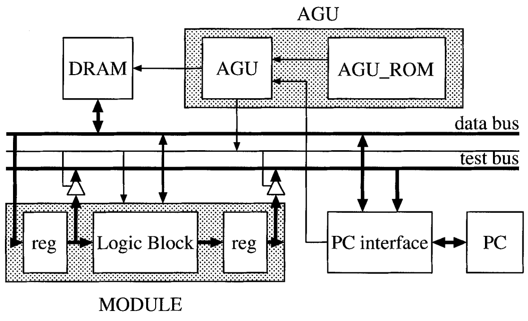


Fig. 12 System-MSPA test system.

written back to main memory.

3. Test

Since the input and output data of each module are stored in DRAM, testing of operations for each module is easier. PC interface can make DRAM access possible directly from outside PC through a data bus. Testing code is sent from PC to DRAM. Then, PC makes each module run in normal mode by issuing instruction set to AGU. After several clocks, PC gets testing output from DRAM. In this sense, PC interface can make H.263+ encoder and decoder work as an emulator. In similar way, the internal status are also observed for the testing purpose by probing registers in each module through a test bus. PC testing scheme is shown in Fig. 12.

Based on above mechanism, we developed PC emulation software, where two modes are introduced under testing mode. One is interactive mode and the other is emulation mode. In interactive mode, PC issues AGU instructions, reads/writes data from/to DRAM, and observe internal status of each module. In emulation mode, PC makes codec LSI run in several steps at normal speed. Either running instruction steps or end address of AGU ROM is specified in advance. By combining these two modes, testing can take place by PC instructions. This scheme makes testing simpler both in prototyping and in bare chip.

4. LSI Results

Tables 5 and 6 summarize the performance and characteristics of encoder and decoder chip. Both encoder and decoder are implemented on 4.6 × 4.6 and 4.0 × 4.0 mm^2 core size area in a 0.35 μm CMOS process. They work at low clock frequency of 27 MHz with small core area and require cheap external 13.5 MHz 256 K × 16 bit DRAM. Consequently, it achieves low power consumption and also easily design under low clock frequency. The maximum clock rate from simulation is 50 MHz. To reduce power consumption, the registers inside modules are disabled through feed back loop by using multiplexer when they do not work on their operations.

Table 5 Performance of system-MSPA chip.

Video standard	H.263+
Frame rate	30 frames/s
Image size	QCIF(176×144)
Target bit rate	less than 27.0 kbps
Search range	[−16,15.5] full search
Max. frame delay	3 frames

Table 6 Video encoder/decoder chip characteristics.

	Encoder	Decoder
Die size	5.6 mm × 5.6 mm	5.3 mm × 5.3 mm
Core size	4.6 mm × 4.6 mm	4.0 mm × 4.0 mm
Technology	0.35 μm, 3 metal layers	
Package	84 pins (56 signal pins)	108 pins (80 signal pins)
# of transistor	760 K	460 K
Clock Frequency	27 MHz (50 MHz max. from simulation)	
Power supply	3.3 V	
Power dissipation	310 mW	110 mW

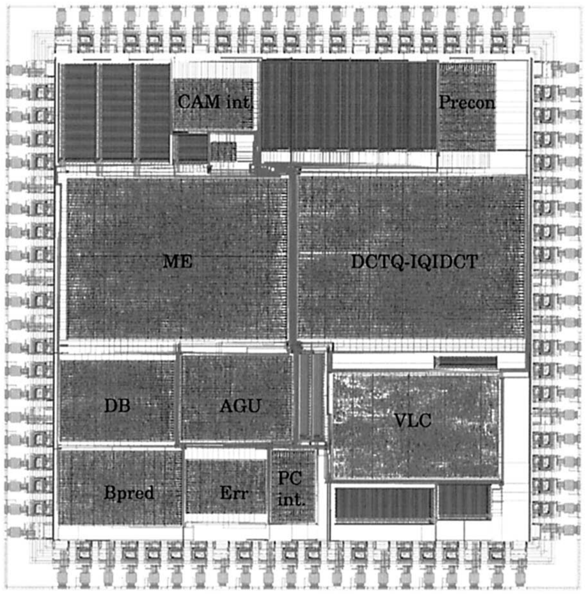


Fig. 13 Video encoder die micro-graph.

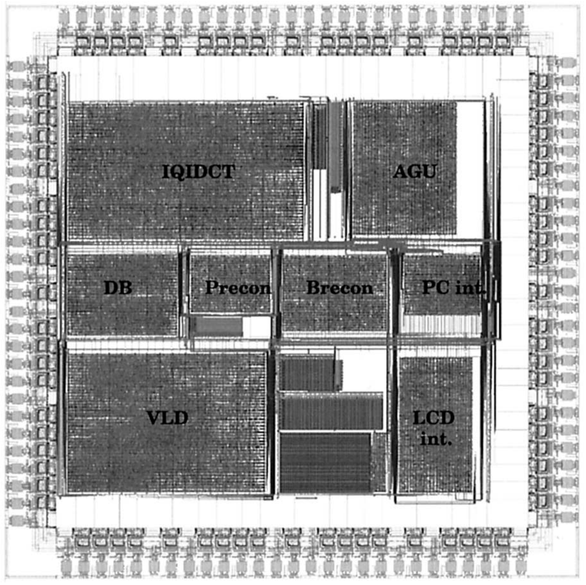


Fig. 14 Video decoder die micro-graph.

Table 7 Number of device, area and size ratio to LSI, power dissipation.

Module	**# of gate & mem trans.**	**Area (mm^2),(%)**	**power (mW)**
AGU	6180 27 K tr.	1.36 (6.4%)	12.8
CAM int.	2670 158 K tr.	2.72 (12.8%)	5.0
ME	20600 160 K tr.	5.45(25.7%)	204.5
Err	3100	0.76 (3.6%)	0.9
DCTQ-IQIDCT	20900 6 K tr.	3.35(15.8%)	63.4
Precon	3280	1.08 (5.1%)	1.0
DEBLOCK	6230	1.24 (5.8%)	0.5
Bpred	6530	1.08 (5.1%)	1.4
VLC	12270 74 K tr.	3.72(17.5%)	19.7
PC int.	1980	0.43 (2.0%)	-
Total	**83740 425 K tr.**	**21.20(100%)**	**310**

Figures 13 and 14 show a die micro-graph of video encoder and decoder. Dedicated modules are designed by VHDL codes and are synthesized by standard cell design. The memory is implemented with full-custom design using a physical layout editor. Dedicated modules also perform as IP (Intellectual Property) modules for both video encoder and decoder. For large enough memory bandwidth, all the tasks of both encoder and decoder can be grouped together and operate in sequential under supervisor of programable AGU. Therefore, both encoder and decoder can be combined to be one chip.

Table 7 shows the number of logic gates, area and the size ratio to the LSI and power dissipation of each unit in video encoder. The main data path units: ME, DCTQ-IQIDCT and VLC, occupy 25.7, 15.8 and 17.5% of core area. The average density of core area is 36,790 transistors/mm^2.

5. Conclusion

Video encoder and decoder based on H.263+ ITU standard for videotelephony applications have been developed with System-MSPA architecture. We succeeded to optimize design by employing the fast and small area dedicated modules in lower level and the slow and flexible system in upper level. Actually, the core area becomes quite small in comparison with conventional design. Furthermore, it is flexible for further development and easy testable. Cheap external DRAM is sufficient for our system.

Acknowledgements

Authors would like to thank all the members of CAD21 Research Body of Tokyo Institute of Technology for their suggestions and cooperations.

References

[1] D. Brinthaupt, J. Knobloch, J. Othmer, B. Petryna, and M. Uyttendaele, "A programmable audio/video processor for H.320, H.324, and MPEG," Proc. ISSCC96, pp.244–245, 1996.

[2] S.K. Rao, M. Hatamian, M.T. Uyttendaele, S. Narayan, J.H. O'Neill, and G.A. Uvieghara, "A real-time P*64/MPEG video encoder chip," Proc. ISSCC93, pp.32–33, 1993.

[3] D. Brinthaupt, L. Letham, V. Maheshwari, J. Othmer, R. Spiwak, B. Edwards, C. Terman, and N. Weste, "A video decoder for H.261 video teleconferencing and MPEG stored interactive video applications," Proc. ISSCC93, pp.34–35, 1993.

[4] M. Harrand, J. Sanches, A. Bellone, A. Tournier, O. Deygas, J.-C. Herluison, D. Doise, and E. Berrebi, "A single-chip CIF 30 Hz H261, H263, and H263+ video encoder/decoder with embeded display controller," Proc. ISSCC99, pp.268–269, 1999.

[5] M. Harrand, M. Henry, P. Chaisemartin, P. Mougeat, Y. Durand, A. Tournier, R. Wilson, J.-C. Herluison, J.-C. Longchambon, J.-L. Bauer, M. Runtz, and J. Bulone, "A single-chip videophone video encoder/decoder," Proc. ISSCC95, pp.292–293, 1995.

[6] D. Li and H. Kunieda, "Memory sharing processor array (MSPA) architecture," IEICE Trans. Fundamentals, vol.E79-A, no.12, pp.2086–2096, Dec. 1996.

[7] D. Li and H. Kunieda, "Automatic synthesis of a serial input multiprocessor array," IEICE Trans. Fundamentals, vol.E79-A, no.12, pp.2097–2105, Dec. 1996.

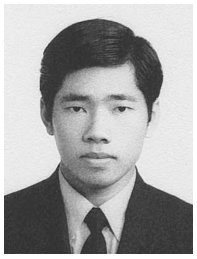
[8] D. Li, L. Jiang, T. Isshiki, and H. Kunieda, "Array architecture and design for image window operation processing ASICs," Proc. IEEE 1998 International Symposium on Circuit and Systems, Monterey, CA, U.S.A., June 1–3, 1998.

[9] D. Li, L. Jiang, and H. Kunieda, "Design optimization of VLSI array processor architecture for window image processing," IEICE Trans. Fundamentals, vol.E82-A, no.8, pp.1475–1484, Aug. 1999.

[10] D. Li, L. Jiang, T. Isshiki, and H. Kunieda, "New VLSI array processor design for image window operations," IEEE Trans. CASII, Analog and Digital Signal Processing, vol.46, no.5, pp.635–639, May 1999.

[11] M.T. Sun, T.C. Chen, and A.M. Gottlieb, "VLSI implementation of a 16 × 16 discrete cosine transform," IEEE Trans. Circuits & Syst., vol.36, no.4, pp.610–617, April 1989.

[12] ITU-T Recommendation H263, "Video coding for low bit-rate communication," International Telecommunication Union Standard, Feb. 1998.



Chawalit Honsawek was born in Bangkok, Thailand in 1975. He received the B.E. degree from Chulalongkorn University, Bangkok, Thailand and M.E. degree from Tokyo Institute of Technology, Japan, in 1996 and 1999, respectively. He is now working toward Dr.Eng. degree at Tokyo Institute of Technology. His research interests include LSI design for digital signal processing.

Kazuhito Ito received the B.S., the M.S., and the Ph.D. degrees in Electrical Engineering from Tokyo Institute of Technology, Tokyo (Japan), in 1987, 1989, and 1992, respectively. He was with the Tokyo Institute of Technology from 1992 to 1995 and visited the University of Minnesota from April 1993 to July 1994. Currently, he is an associate professor of the Department of Electrical and Electronic Systems, Saitama University, Urawa (Japan). His research interests include high-level synthesis in digital signal processing, VLSI design automation, and system LSI design methodology.

Tomohiko Ohtsuka was born in 1966. He recieved B.E., M.E. and Ph.D. from Tokyo Institute of Technology in 1989, 1991 and 1994, respectively. He joined the Department of Electric and Electronic Engineering, Tokyo Institute of Technology as a Research Assistant in 1994. He is presently an Associated Professor of Department of Electronic Engineering, Tokyo National College of Technology. His current research interensts include analog LSI CAD and VLSI system designs.

Trio Adiono received the B.S. degree in electrical engineering, the M.S. degree in Microelectronics from Bandung Institute of Technology, Indonesia, in 1994 and 1996, respectively. He currently working toward his Ph.D. degree in Very Large Scale Integrated Circuit (VLSI) Design at Kunieda Laboratory, Tokyo Institute of Technology. His current research interests include very low bit-rate video coding, motion estimation techniques, H.263 video standard, Mpeg-4 Video Codec, VLSI Architecture for Video Applications, and Machine Vision System.

Dongju Li was born in Shen-yang, Liao-ning Province, China. She received her B.S. degree from Liao-ning University and M.E. degree from Harbin Institute of Technology, China, in 1984 and 1987, respectively. She had been engaged in VLSI design as an engineer in VLSI Design Laboratory of Northeast Microelectronics Institute, Electronics Industry Ministry, China from 1987 to 1994. She received her Ph.D. degree in Electrical and Electronics Engineering from Tokyo Institute of Technology in 1998. She is presently research associate on Department of Communications and Integrated Systems, Tokyo Institute of Technology. Her current research interests include VLSI array processor architecture and its design methodology, system on chip, DSP, and VLSI design.

Tsuyoshi Isshiki has received his Bachelor's and Master's degree in electrical and electronics engineering from Tokyo Institute of Technology in '90 and '92, respectively. He has received his Ph.D. degree in computer engineering from University of California at Santa Cruz in '96. He is currently Associate Professor at Department of Communications and Integrated Systems, Tokyo Institute of Technology. His research interests include high-level design methodology for configurable systems, bit-serial synthesis, FPGA architecture, image processing, computer graphics, and speech synthesis.

Hiroaki Kunieda was born in Yokohama in 1951. He received B.E., M.E. and Dr.Eng. degrees from Tokyo Institute of Technology in 1973, 1975 and 1978, respectively. Since 1978, he has engaged in education and research on Distributed Circuits, Switched Capacitor Circuits, VLSI Digital Signal Processing, and LSI design and CAD at Tokyo Institute of Technology. He is currently Professor at Tokyo Institute of Technology. His current research interests focus on Multimedia LSI design, VLSI Signal Processing, Parallel Processing, VLSI CAD and Parallel Image Processing. Recent projects include fingerprint authentication system and video phone system. Dr. Kunieda is a member of IEEE CAS and SP society and IPSJ.