An Efficient MPEG-4 Video Codec using Low-power Architectural Engines

Bontae Koo, Juhyun Park, Seongmo Park, Seongmin Kim, Nakwoong Eum
Advanced Micro-Electronics Research Lab., ETRI
161 Kajong-dong, Yusong-Gu, Taejeon, 305-350
Tel 042-860-1329, Fax 042-860-6836, e-mail koobt@etri.re.kr

Abstract

We present a low-power MPEG-4 video codec chip capable of delivering high-quality video data in wireless multimedia applications. The discussion will focus on the architectural design techniques for implementing a high-performance video compression/decompression chip at low power architectures. The proposed MPEG-4 video codec can perform 30 frames/s of QCIF or 7.5 frame/s of CIF at 27MHz for 128k~144kbps. By introducing the efficiently optimized Frame Memory Interface architecture, low power motion estimation and embedded ARM microprocessor and AMBA interface, the proposed MPEG-4 video codec has low power consumption for wireless multimedia applications such as IMT-2000.

1. Introduction

The market for wireless multimedia applications, such as MPEG video camera, wireless videophone, and portable wireless multimedia terminal, has been on the rise. Video and audio processors are essential to make this multimedia system high performance, low cost and low power consumption. MPEG-4[1] is a promising standard for coding audiovisual objects on the mobile and portable multimedia applications because it has high coding efficiency and high error resilience. These applications require low power consumption and high cost-effectiveness to success in the mobile and portable wireless multimedia systems. In order to achieve these requirements, both the algorithm and chip implementation have to be improved.

In this paper, we have developed a low-power MPEG-4 video codec which has a decoding and encoding function and is suitable for wireless multimedia applications. The architectural design techniques are introduced to reduce cost and power consumption. This MPEG-4 video codec covers a wide range of bit rates and various image formats. Specifically, it is suitable for low-bit-rate communication. And it supports both H.263 ITU-T recommendation and MPEG-4 SP@L2, and performs 30-framts/s encoding and decoding in QCIF, 7.5 frames/s codec in CIF at 27MHz operation.[2] Low power is an essential factor for battery-driven wireless applications. And software programmability with an RISC processor is preferable in order to cope with MPEG-4 standardization. A combination of RISC and dedicated hardware modules was chosen in order to satisfy requirements for both low power and programmability. Furthermore, various low-power design architecture techniques are employed.[3] This paper presents the MPEG-4 video codec, specifically, its architecture and low-power techniques.

2. MPEG4 Video Codec Architecture

2.1 overview

The block diagram of MPEG-4 video codec is shown in Figure 1. It is composed of a ARM7TDMI RISC processor, modified AMBA(Advanced Microprocessor Bus Architecture) [4] interface and dedicated hardware modules, such as discrete cosine transform(DCT), inverse DCT(IDCT), motion estimation and motion compensation(MEMC), video interface modules (VIM and VOM) and stream interface module(ISC and SP), and frame memory interface (FMI) module.

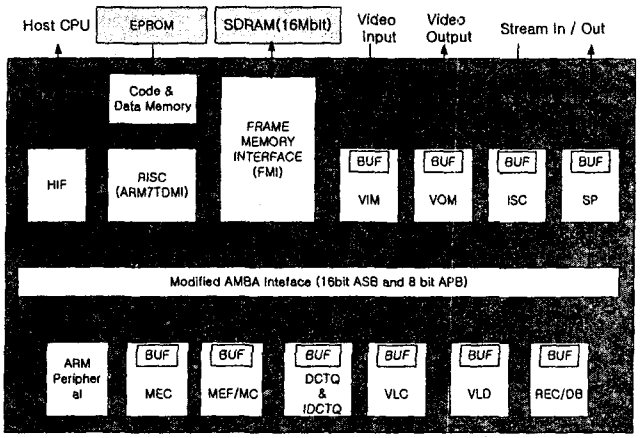

Figure 1. The Block diagram of MPEG-4 video codec

These hardware modules and the RISC processor have sufficiently minimum-size local memories for parallel operation. The local memories also reduce access to the external memory through I/O pads, which saves on power dissipation. The total amount of the internal local memory is 9.53kB. Further, a 16Mbit external frame memories is connected to the frame memory interface (FMI) module via 16-bit memory bus. Only the FMI module can transfer data among internal local memories and can access external memory. Using this DMA controller of FMI is more power efficiency and high performance than general DMA controller because general DMA controller is composed of temporal buffer in itself to transfer data among local memories and external memory. ARM7TDMI RISC is 32-bit simple processor. It access 32-kB instruction and data Memory in which the whole firmware program is stored. And we adopted main system bus interface with modified AMBA interface – 16-bit advanced system bus(ASB) and 8-bit advanced peripheral bus (APB). RISC processor executes complicated processing, such as bit stream syntax processing, rate control, parameter setting of the registers in internal modules on pipeline. The RISC

processor also controls all hardware modules, including FMI modules. Host interface module (HIF) transfer data between the processor and an external host processor. And it supports an I2C serial bus interface and an Intel and Motorola parallel bus interface. Video input module (VIM) converts video get from external CMOS image sensor to 4:2:0 Y,Cr,Cb format, which is MPEG-4 video input standard. Video output module (VOM) can display RGB and several Y,Cr,Cb formats for LCD. A motion estimation in coarse resolution (MEC) and a motion estimation in fine resolution (MEF) execute a three-level hierarchical motion estimation to reduce power dissipation. And motion compensation (MC) not only executes motion compensation processing but also handles creation data for motion estimation. MEC executes first-step motion estimation using the half-resolution pictures. Second MEF executes the second-step motion estimation using integer-pel pictures, followed by the third-stop motion estimation using half-pel pictures. The search area of the motion estimation is -8/+7, which is fixed. In order to reduce spatial redundancy, DCT/IDCT and Q/IQ are used. To achieve more compression ratio, AC/DC prediction algorithm was adopted. Variable length coding (VLC) and VLD translate from DCT coefficients to variable length code and vice versa. Post-processing is used to reduce the visual artifacts and improve the quality of compressed video. De-blocking filter operations are performed along the 8x8 block edges at the decoder as a post-processing operation.[5] This MPEG-4 video codec performance is 30 frames/s of encoding and decoding with QCIF, and 7.5 frames/s codec with CIF at 27MHz. And it also supports H.263 ITU-T recommendation.

.2 Pipeline and scheduling

or simplifying the design of timing control of the codec, e used vector pipeline architecture and adopted parallel rocessing at MB level. We calculated memory operation nd scheduling operation of control task. RISC and ardware modules in this MPEG-4 codec have their own cal memories and can operate concurrently. Figure 2. ows the timing chart of the codec for QCIF 30 frames/s. d Figure 3. shows the MB-based pipeline parallel eration in MPEG-4 codec.

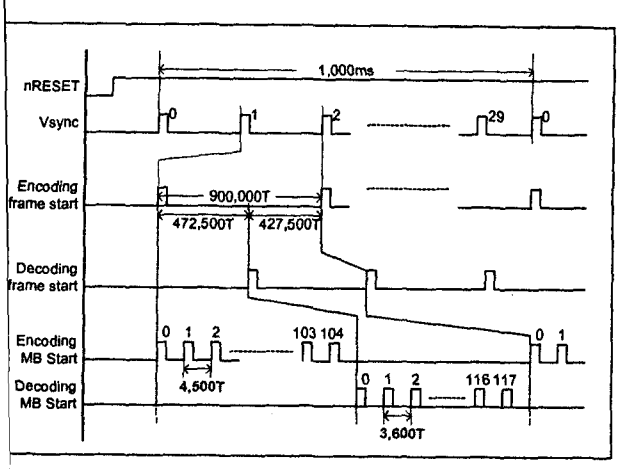

Figure 2. Timing chart (e.g. QCIF, 30frame/s)

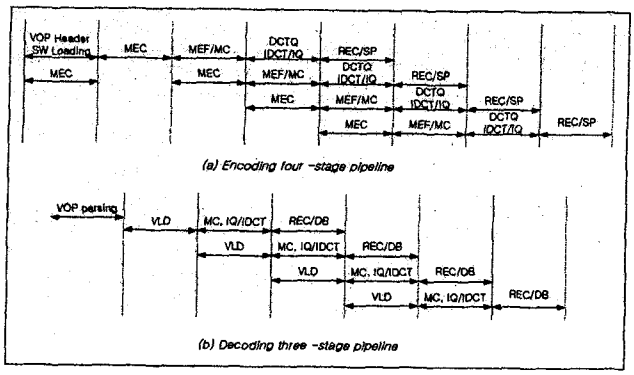

Figure 3. MB-based pipeline

The pipeline stages consist of four stages in encoding operation and three stages in decoding operation. Each MB encoded in a period of 4,500 clock cycle and decoded in a 3,600 clock cycles. As in the Figure 3., the pipeline architecture can reduce power consumption because parallel operation with these hardware modules reduces the required clock rate. The low clock rate results in low power dissipation. Figure 4. shows the MB-based pipeline schedule. This fixed time slot architecture has the advantage of simplifying SDRAM interface because SDRAM access arbitration is not necessary. The height of each block presents the relative size of memory or bus bandwidth. 16-bit SDRAM data bus width is used for saving the memory bandwidth and achieving efficient memory access, thus all of memory loading for each MB coding are performed in one MB time slot.

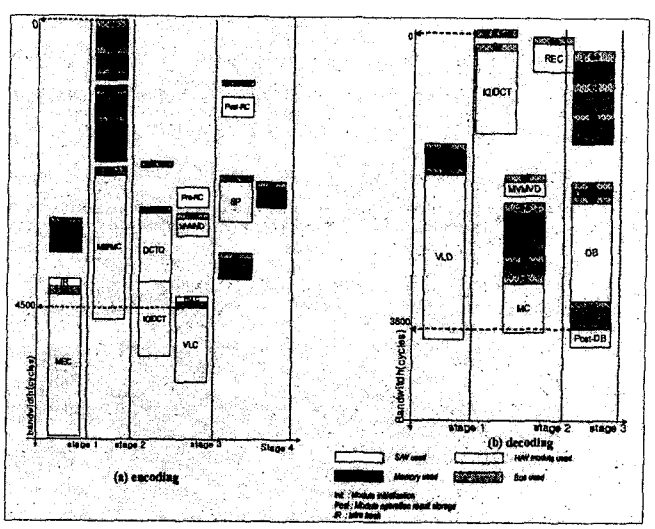

Figure 4. MB-based pipeline schedules.

3. Low power architectural design techniques

3.1 Efficient Frame Memory Interface (FMI)

Frame memory interface module(FMI) can transfer data among internal local memory and external memory, SDRAM. To achieve high efficiency for DMA transfers and to reduce the required capacity of the external memory, we devised an efficient frame memory interface modules. FMI consists of a DMA controller and an SDRAM controller respectively in Figure 5.. FMI performs both

SDRAM read/write control, DMA scheduling, and peripheral functions. The function of SDRAM controller generates SDRAM interface signals, nCAS, nRAS and nWE etc.

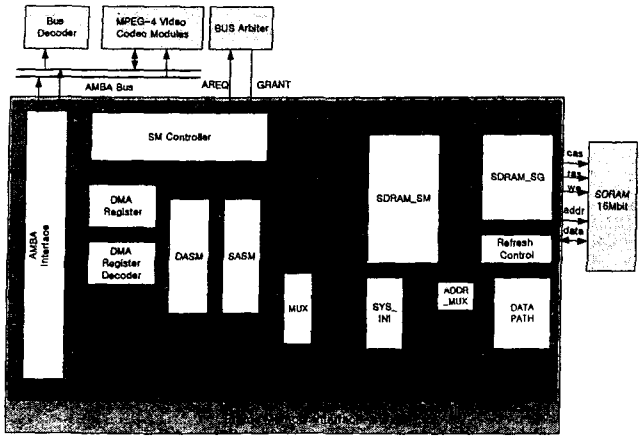

Figure 5. The Block diagram of FMI

DMA operations are divided into slave mode and master mode. In the slave mode, RISC processor initializes register of DMA controller with transfer source and destination address, transfer length, transfer direction and etc. After register initialization, on receipt of DMA request, bus arbiter grants DMA controller to be master. And then DMA controller can control system address and data buses and proceeds to transfer data until a stop conditions is met. After then, DMA controller returns to the slave mode. Figure 6. shows the data transfer operation flow of FMI in write and read mode.

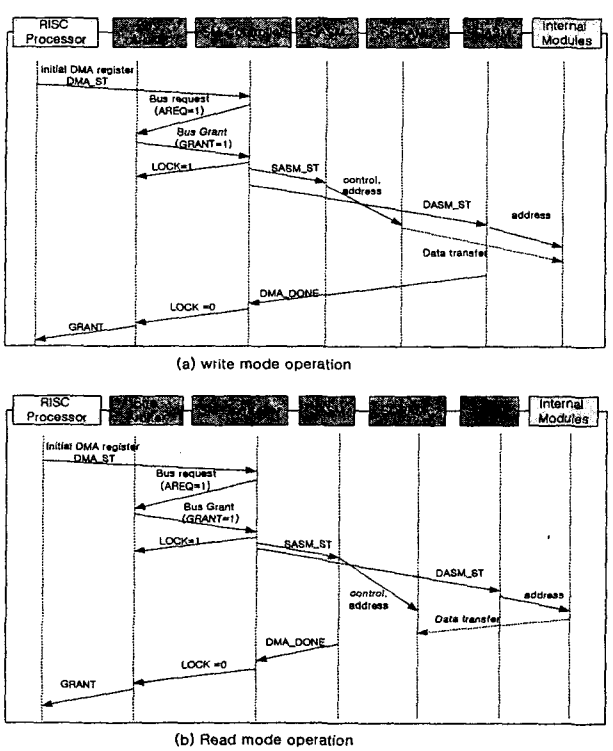

Figure 6. The data transfer operation flow of FMI in write and read mode.

DMA controller has special features. It can interface with all of internal modules with only one channel (AMBA bus), is made up programmable DMA and supports both burst block mode and packet mode data transfer. On the other hand, it has architecture of dual addressed DMA without buffered memory. In dual addressed DMA transfer, an explicit address is required for both source and destination. Each DMA transfer then requires two bus transactions; the first to read the data from the source, and the second to write the data to the destination. An explicit address is generated for each transaction, and the data must be buffered internally by the DMA controller between read and write.[6] But the proposed DMA controller need not buffered memory. So it can transfer data between internal local memories and external memory at two times speed than conventional DMA controller. It can reduce required clock rate and data bandwidth.

For real time coding of MPEG-4 video codec, CIF (352x288) format video process requires 4,500 clock cycles in encoding per each MB and 3,600 clock cycles in decoding. This operation frequency is 27MHz and 16Mbit (512x16bitx2banks) external memory, SDRAM, has same system clock operation. Figure 7. shows all of the memory access time and SDRAM bandwidth allocation. Total memory access cycles are about 2400 cycles (54% SDRAM allocation in pipeline) in encoding and 2030 cycle (64%) in decoding. The rest of the pipeline cycles are used initialization of modules and firmware such as scheduling and sequencing.

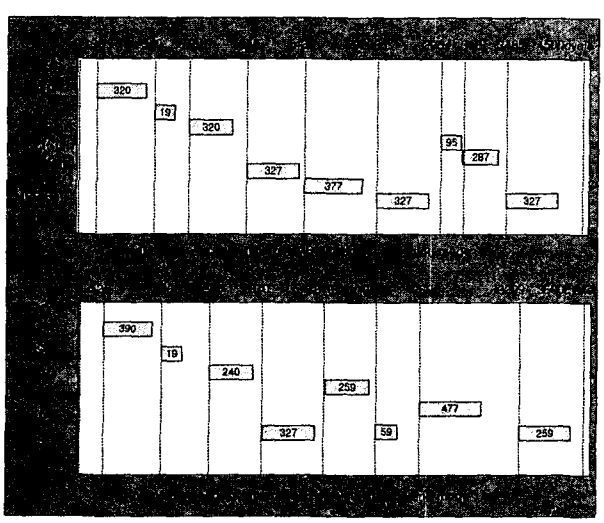

Figure 7. Memory access time per MB-based pipeline

3.2 Low power mixed mode motion estimation

A low power motion estimator is applied for the MPEG- video codec. To reduce the computational complexity an the power consumption of motion estimation, not onl three-step hierarchical block matching algorithm is use rather than full-search block matching algorithm, but als mixed mode motion estimation which is based o unnecessary computation using motion estimation skij Figure 8. shows the block diagram of the proposed mixe motion estimator. The motion estimation architecture ha

a normal mode operation and skip mode operation according to skip decision.

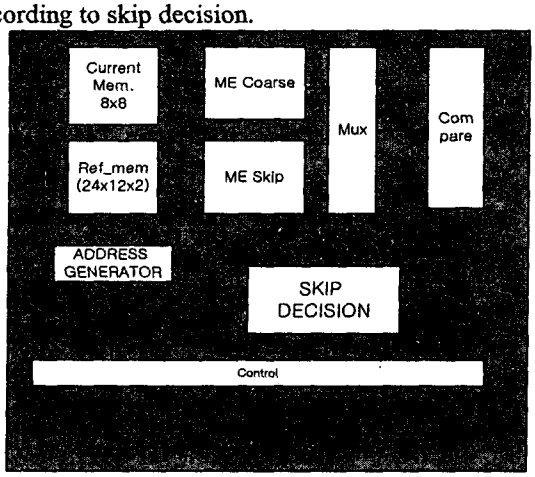

Figure 8. Block diagram of the mixed motion estimator

This block consists of current memory, reference memory, motion estimation coarse, motion estimation skip, and motion estimation skip decision. The search area of the motion estimation is -8/+7 and processing elements are eight. Eight candidates for motion vector are examined concurrently using the block matching technique. Before motion estimation operation, the first search is conducted with the motion estimation skip, which is motion vector prediction and motion compensation with previous parameters. These architecture and skip algorithm enables a high speed, low power operation. Table1. shows percentage of disabling calculating the actual operation. In all test sequences, this reduced computation load to 32% that of conventional three-step methods.

Table 1. Percentage of disabling motion estimation skip

	Carphone	Foreman	Coast	Stefan
CIF 76x144)	30 %	24.6%	36.7%	37.4%
SNR	32.59 dB	31.67 dB	29.00 dB	23.94 dB

4. Implementaion

All modules are designed using mixed HDL languages, Verilog-HDL and VHDL, and these modules are synthesized with a 0.35um HYUNDAI standard cell library. Table 2. shows the specification of MPEG-4 video codec. And Figure 9. shows a layout pattern of the MPEG-4 video codec LSI.

Table 2. LSI specifications

Process Technology	0.35um CMOS (4-layer metal)
Chip Size	9.1 x 9.1mm^2
Clock Frequency	27MHz
Supply Voltage	3.3V
Power Consumption	~500mW(estimated)
Package	240 pin MQFP

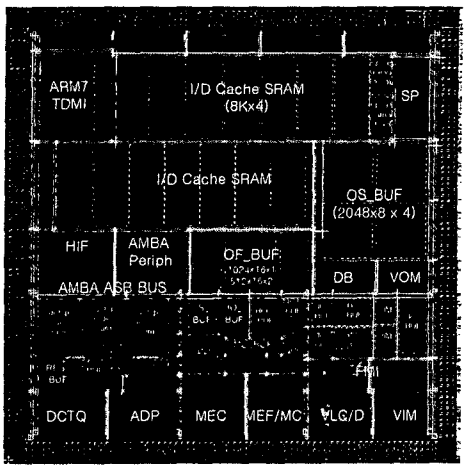

Figure 9. Layout pattern of MPEG-4 video codec.

5. Conclusion

As you see in previous sections, the performance of the image processor such as MPEG-4 video codec, depends on the efficiency of the chip with a capacity of frame memory interface, which can transfer data between internal local memories and external memory. And Motion estimation operation has a lot of computation. It is necessary that the computation of motion estimation should be reduced. Because of the efficient frame memory interface (FMI), which has a real-time data transfer DMA architecture, and the low computational and low power motion estimator, the propose MPEG-4 video codec can perform 30 frame/s of QCIF or 7.5frames/s of CIF at 27MHz operation. The proposed architecture brings about high cost-effectiveness and low power consumption for wireless multimedia applications such as IMT-2000.

References

[1] ISO/IEC JTC1/SC29/WG11 N3312, MPEG-4 Video Verification Model version 16.0

[2] Juhyun park, Bontae Koo et al"MPEG-4 Video Codec for Mobile Multimedia Applications", IEEE ICCE pp.156-157, Jun. 2001.

[3] M. Takahashi, et al.," A 60 mw MPEG-4 Video Codec using clustered voltage scaling with variable Supply-Voltage Scheme", IEEE Journal of Solid-State-Circuits,vol.33 no.11 p1772-1780, Nov. 1998.

[4] Advanced Microcontroller Bus Architecture (AMBA) specification, ARM, Doc.Num.: ARM IHI 0001D, 1997

[5] Sun Deuk Kim et al "A Deblocking Filter with Two Separate Modes in Block-Based Video Coding" IEEE transactions on circuits and systems for video technology, vol 9, No.1 Feb. 1999.

[6] Butterfly Microcontroller Handbook, MITEL, Technical Doc. P10.6, 1999