IEICE TRANS. ELECTRON., VOL.E92–C, NO.10 OCTOBER 2009

INVITED PAPER *Special Section on Hardware and Software Technologies on Advanced Microprocessors*

An Integrated Platform for Digital Consumer Electronics

Junji MICHIYAMA[†a)], ***Member***

SUMMARY This paper describes the architecture of an integrated platform developed for improving the development efficiency of system LSIs built into digital consumer electronics equipment such as flat-panel TVs and optical disc recorders. The reason for developing an integrated platform is to improve the development efficiency of system LSIs that serve the principal functions of the said equipment. The key is to build a common interface between each software layer, with the system LSI located at the lowest layer. To make this possible, the hardware architecture of the system LSI is divided into five blocks according to its main functionality. In addition, a middleware layer is placed over the operating system to improve the ease of porting old applications and developing new applications in the higher layer. Based on this platform, a system LSI called UniPhier™ has been developed and used in 156 product families of digital consumer electronics equipment (as of December 2008).

key words: *platform, system LSI, consumer electronics, video codec*

1. Introduction

Numerous types of digital consumer electronics, for example digital TVs, optical disc recorders and mobile phones, are developed every year. Most digital equipment uses a system LSI as the key device to perform the most important functions. However, it is becoming increasingly resource-intensive to develop and/or port software between system LSIs. This prompted us to develop an integrated platform for digital equipment, designed to both improve development efficiency and allow the rapid implementation of new features.

The present paper is organized as follows. Section 2 describes the concepts of an integrated platform. Sections 3 and 4 explain the hardware architecture and the software architecture, respectively. Section 5 covers low power consumption technologies, especially for mobile applications. In Sect. 6, we introduce two system LSIs that exploit this platform. Section 7 concludes the paper.

2. Concept of a Platform for CE Equipment

This chapter describes the development progress and the concept of the platform.

2.1 Trends in Applications

Starting with flat-panel TVs that are adopting larger displays with full high-definition pictures, most digital consumer electronics, including storage products such as optical disc recorders and HDD recorders, and battery-operated products such as compact video cameras and mobile phones, are rapidly advancing to the point where they can support high-definition content. Higher picture and sound quality can be achieved by digital processing of video, voice and audio signals. As a typical example of digital signal processing, the evolution of video codec algorithms is shown in Fig. 1. This chart indicates how a variety of methods have been made into standards and adopted in digital consumer electronics. In this chart, arrows are used to indicate the beginning of deployment of products using the standardized video codec algorithm. The video codec algorithms adopted by digital TVs, compact video cameras and optical disc recorders primarily include MPEG-2, MPEG-4, AVC/H.264, and VC-1. With the increase of PCs and mobile phones in addition to the digital consumer electronics mentioned above which is compatible with high definition (HD) content, distribution of HD content is expected to accelerate in the future. With the enhancement of the HD value chain, the use of digital equipment by connecting to a network and exchanging HD video content will become essential throughout the world.

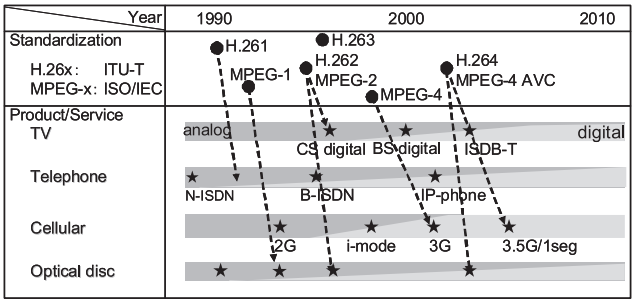

Fig. 1 History of video codec algorithms.

2.2 Problems Caused by Legacy Solutions

The high functionality and network connection ability of digital consumer electronics is leading to a rapid increase in software development volume in the system LSI (Refer to Fig. 2). In 2005, the number of lines of code in software for an optical disc recorder exceeded 10 million.

For video codec processing in optical disc recorders, DVD recorders are equipped with MPEG-2, but Blu-ray recorders need to be equipped with MPEG-4 AVC/H.264, and to connect to a broadband IP network and download

Manuscript received March 13, 2009.

[†]The author is with Panasonic, Kadoma-shi, 571-8501 Japan.
a) E-mail: mitiyama.junji@jp.panasonic.com

DOI: 10.1587/transele.E92.C.1240

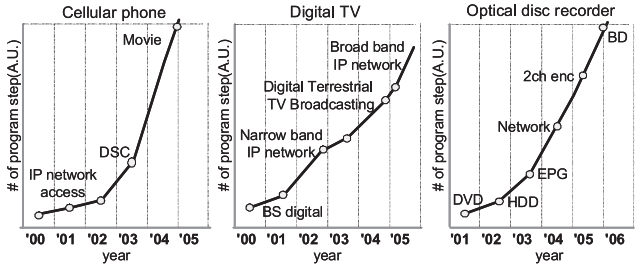

Fig. 2 Number of program steps. (lines of code)

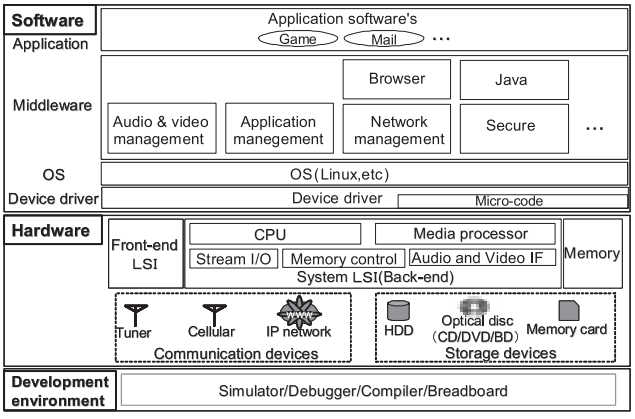

Fig. 3 Overall architecture of integrated platform.

moving pictures for viewing, compatibility with VC-1 is also becoming necessary. As mentioned in the section above, video codec algorithms are converging on MPEG-2, MPEG-4 AVC/H.264, and VC-1, regardless of digital AV equipment type. Similarly, audio codec is converging on AAC and AC3. These trends indicate the need for improving development workload by building a mechanism for sharing software among different digital consumer electronics or among different types of system LSIs, instead of developing software for each model of digital consumer electronics or for different LSI type adopted by digital consumer electronics.

2.3 Concept of A Platform Architecture [1]

The mechanism for sharing software among digital consumer electronics or among different types of system LSIs is called here an 'integrated platform.' To create this integrated platform, the key is to standardize as much as possible the software architecture and hardware architecture that form the system, as well as the interface between the software and hardware, among digital equipment and among system LSI types.

The hardware architecture of the integrated platform for digital equipment needs to have high performance in video, audio, and voice signal processing. As shown at the bottom of Fig. 3, the five blocks common to all system LSIs were created. An original bus optimized for transferring media data is implemented. Each block is designed to extract higher transferring performance from the original bus.

The advantage of having these five blocks is to enable independent development of each block by defining the interface between the blocks, and to clear the way for technological evolution of each block.

An outline of each block is given below.

- CPU block:
  Consists of a micro-controller and its peripheral hardware for controlling the entire item of digital consumer electronics.
- Media processor block:
  Consists of a DSP and hardware accelerators. Performs a variety of digital signal processing tasks, such as encoding and decoding of video, audio and voice and processing for the improvement of their quality.
- Audio and video IF block:
  Functions as an interface to various output devices such as image display devices and speakers, and various input devices such as cameras and microphones. This block contains a filtering function and graphics accelerator.
- Stream I/O block:
  Dedicated to input/output of bit streams, this block performs as an interface to the TV tuner, optical drive and baseband of cellular phone and others.
- Memory block:
  This block provides an arbitration function for bus masters and interface circuits to the SDRAM. This arbitration function enables access to the SDRAM without sacrificing throughput or response to contention for access to the SDRAM by multiple bus masters.

Software for the integrated platform is shown in the top part of Fig. 3. It consists of the microcode for processing the AV media executed by the DSP, and the device drivers, operating system, and middleware executed by the CPU. The operating system can be either Linux or an RTOS, depending on the type of equipment. The feature adopted here is the mechanism, which has been newly defined as a framework, of separating the mediation processing of competing tasks between applications; and the processing of each application required by the middleware to be embedded in the product. This concept improves the independence of application development, allowing the reuse and portability of applications.

Development environment also plays an important role for the integrated platform. A system LSI simulator which simulates major functions in the system LSI is developed before fabrication of the system LSI. The system LSI simulator is available for early development of software and verification of system LSI.

3. Hardware Architecture

This chapter describes the media processor block and graphics accelerator within the audio and video IF block.

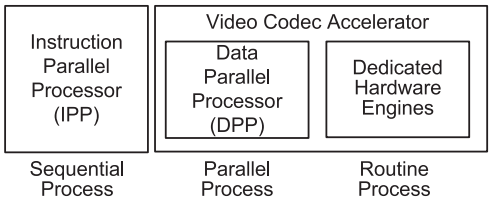

Fig. 4 Media processor platform architecture.

3.1 Media Processor

The media processor consists of three components (Fig. 4). We categorize the AV codec processes into sequential processes such as audio codecs, and parallel processes such as pixel-level processes. From another viewpoint, we categorize them as routine processes that are defined by standards, and as non-routine processes such as the process of enhancing image quality.

3.1.1 Instruction Parallel Processor (IPP) [2]

The instruction parallel processor (IPP) is a media processor that performs sequential processing. Its internal structure was determined according to the analysis result of presumed applications.

The architectural features of the IPP are

1) the ability to issue three instructions per cycle and a very efficient pipeline structure to improve IPC;
2) multi-threaded architecture with a thread management unit to support concurrent operation of real-time applications; and
3) a memory sub-system with software-controllable cache to minimize data transfer overhead.

The IPP can issue up to 3 instructions per one cycle to 3 execution units. The IPP is amenable to the attachment of user-customized functionality. Our analysis shows that the average instruction parallelism in multimedia applications is around 2 to 3 instructions. Therefore, the pipeline structure of the IPP is ideal for maintaining good sustained performance using a C++ compiler.

For the purpose of securing real-time performance during the parallel processing of multiple applications, a new mechanism, called a virtual multi-processor (VMP), has been adopted.

Each software thread is mapped to a LP (logical processor), and the context of each thread is kept in the context memory. The thread management unit (TMU) performs thread scheduling according to the priority, and supports hardware context switching. The TMU also supports resource management of processor performance by using time-based scheduling. Each LP can maintain its scheduled processor performance with no impact from other LPs. By using these mechanisms, VMP allows the IPP to maintain sustained performance during concurrent operations of multimedia applications without an RTOS.

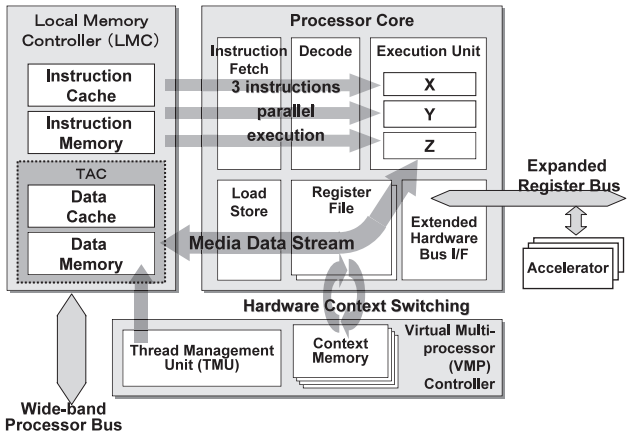

Fig. 5 Outline structure of IPP.

The memory sub-system is also the key to processor performance and cost (chip area) trade-off. For real-time applications, using local data-memory architecture is an easy way to guarantee memory access latency, but this will take a larger area. To avoid area impact, the memory sub-system of the IPP is cache-based and has several unique features. Use of a cache makes program development easier, but on the other hand, it tends to impede the assurance of real-time performance. The IPP, therefore, is equipped with a cache support function tailored to the media processing characteristics, along with an interlock mechanism with the VMP, thereby meeting the performance assurance required for media processing.

A simplified block diagram of the IPP is shown in Fig. 5. It consists of a processor core, virtual multi-processor controller, expanded register bus, instruction cache/instruction memory, and data cache/data memory. The processor core executes the parallel processing of 3 instructions. The virtual multi-processor controller is equipped with a hardware-based context switch function to reduce the overhead caused by switching the context.

The IPP can be enhanced with an expanded computing unit such as a bit-processing accelerator for accelerating a specific function. The DPP and hardware engine are connected with the IPP via an expanded register bus, and are mapped as a part of the IPP memory space. This arrangement enables easy control of the DPP and hardware engine from the software on the IPP.

3.1.2 Video Codec Accelerator

The video codec accelerator consists of a data parallel processor [3] and a dedicated hardware engine. To achieve high performance, low power consumption and low cost at the same time as a platform, the data parallel processor is used for processing that requires flexibility. On the other hand, a dedicated hardware engine is used for processing that is specified by standards and does not need flexibility.

The data parallel processor is designed to use software to process signals at the pixel level (hereinafter called pixel

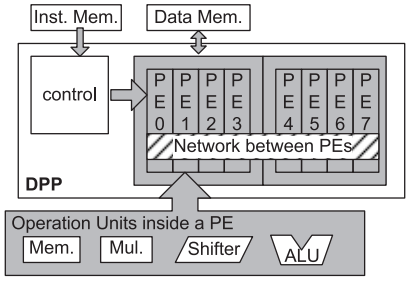

Fig. 6 Data parallel processor architecture.

Table 1 Decoding process of each standard.

	MPEG2	H.264	VC-1
Variable Length Coding	Huffman	Huffman Arithmetic	Huffman
Inverse Transform	DCT	integer DCT	integer DCT
Motion Compensation	16x16, 16x8	16x16 – 4x4	16x16 - 8x8
Filter	None	de-blocking	de-blocking

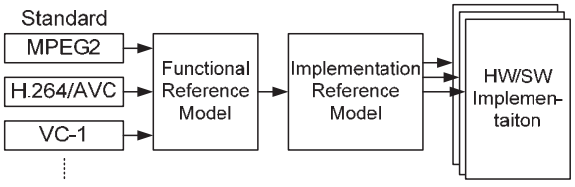

Fig. 7 Development flow of video codec accelerator.

processing) flexibly and at high speed. Pixel processing involves two types of parallelism: (1) applying the same operation to multiple pixels, and (2) combining multiple operations e.g. product and sum operations. The data parallel processor adopts an SIMD (single instruction multiple data) style processor configuration, which has a high affinity with type (1) parallelism, meaning that all the processor elements (PE) perform the same operations under a single instruction memory and controller. To achieve type (2) parallelism, the data parallel processor is equipped with multiple operation units in a single processing element, as shown in Fig. 6. This configuration enables simultaneous operations, such as multiplication and addition., to be included in the processing.

The current video codec standards applicable to high definition television mostly include MPEG-2, H.264/AVC, and VC-1, so the video codec accelerator needs to accommodate all of these standards. The accelerator also needs to be compatible with traditional standards for low-resolution images such as MPEG-4.

As shown in Table 1, MPEG-2, H.264/AVC, and VC-1 each adopt a variety of tools for increasing the compression rate, but the basic processing flow and processes are based on similar structures. A single accelerator compatible with multiple standards provides area efficiency that is approximately 30% higher than if multiple accelerators, each designed for a single standard, are adopted.

The development approach shown in Fig. 7 was used to design a multi-platform video codec accelerator that would ensure compatibility with a variety of standards and applicability to a wide range of digital AV products. At first, a functional-level reference model was developed according to the definition of the target standards. Then, a reference model with a processing hierarchy structure and hardware functional blocks matching those of a system LSI implementation was developed. Using this reference model, issues of functionality and performance of the video codec accelerator can be found earlier.

Developing and building the reference model into a system LSI simulator allows rapid verification of the functions and performance of the entire system LSI. Therefore, the reference model can also be used to verify implemented hardware and software, making it readily applicable to other products as a platform.

3.2 Graphics Accelerator

Graphics accelerators for digital consumer electronics are used in a wide variety of ways, from execution of gaming applications to rendering of user interfaces used in widescreen flat-panel TVs. To meet these varied requirements, the architecture of graphics accelerators is designed primarily for high portability, with enhancement in performance and support of multi-task processing also being a high priority.

Achievement of portability is described first. Graphics accelerators used in mobile phones require a support for 3D rendering for applications such as games. On the other hand, digital TVs and optical disc recorders do not necessarily require such support, since their principal use is as a user interface. In the sense to make the graphics accelerator as a platform, the principle of supporting 3D graphics rendering was selected in preference to a 2D graphics rendering. As mentioned earlier, the problem was that the small displays used in mobile phones and the widescreen used in digital TVs present a substantial difference in the performance requirements of a graphics accelerator. Graphics processing can be briefly classified into vertex processing and pixel processing. Vertex processing performance is expressed as the number of polygons which can be rendered per unit time, while pixel-processing performance is expressed as the number of pixels that can be rendered per unit time. For example, rendering capability of a digital TV with full HD resolution currently requires a high pixel processing capability of 200M to 400M pixels/sec, but the requirement for vertex processing capability is not as high. In contrast, the pixel-processing capability required by a mobile phone with a small display is relatively low, but it needs to render 3D-CG games smoothly, which requires an effective vertex processing performance of 1M to 2M polygons/sec. To meet these requirements, the vertex processing block and the pixel processing block are modularized to enable parallel processing and allow scalability in the performance, as modules can be easily added. The vertex-processing blocks adopts SIMD architecture, while the pixel processing blocks are operated by a sin-

gle controller for suppressing areas otherwise required for multiple control circuits. Moreover, Blu-ray recorders have an additional requirement: the rapid decompression of PNG images. However, this requirement can also be met just by adding an optional module. Furthermore, the microcode and the graphics libraries for the graphics accelerators have been unified as much as possible to improve the portability of higher-level software. The porting of externally-supplied applications, such as browsers and Flash players, will be needed in the future, but it will be met by supporting industry standard APIs.

This paragraph discusses methods of improving the performance of the graphics accelerators. The performance of graphics processors is determined by their overall design: not just the hardware, but also the software such as the device driver and the graphics library. In games, for example, the application layer has a heavy processing workload. However, the performance of CPUs designed to be embedded in CE products is generally weak, and CPU performance compared with the performance of a graphics accelerator is far lower than that of PCs or game machines. For this reason, a graphics accelerator cannot deliver the performance needed by an entire system unless it can execute graphics processing without placing an excessive workload on the CPU. To implement this function, the issuing of commands from the CPU to the graphics accelerator is executed by adding the list of commands to a ring buffer on the external memory of the graphics accelerator. In this configuration, when the CPU passes the initial address of the data array to the graphics accelerator and instructs execution of the commands, the graphics accelerator can access the required data and render without intervention from the CPU. This architecture enables the CPU to add new commands at any time, independently of the graphics accelerator's processing activity, without causing wasteful waiting times on the part of the CPU. Access to virtual memory of the Linux OS from the graphics accelerator is also possible. Measurements made on actual applications indicate that the graphics accelerator and CPU perform separate processing for more than 90% of the time.

This paragraph describes multi-tasking capabilities of the graphics accelerator. Graphics accelerators used in digital CE products are required to cope with different tasks such as games and the windowing system asynchronously in parallel. In addition, image blitting and scaling needs to be executed in the background to improve the performance of the entire system. Graphics accelerators have therefore been designed to process the multiple tasks required by applications, libraries and kernels, etc., using time-division processing.

4. Software Architecture

4.1 Application Framework

In the traditional application development of embedded software, the following items are required in addition to the typical functions of an application.

1) Contention arbitration between applications (Requirement for changing software behavior to match the applications and system conditions) to meet the needs of each product, such as mobile phones or digital TVs (for example, hardware restrictions and transmission/broadcasting specifications).
2) User interface (customization according to the evolution of display size or input method, sales region, target users, etc.).

A mechanism for separating the contention arbitration and user interface from the original functions of each application has been newly defined as a framework. This enables independence of application development.

In addition, under this framework, elements such as various AV formats usable from each application, stream processing for achieving networking, copyright management, and network protocol interface were defined as a common element and made into a component, thus making the software assets shareable among a range of products from mobile phones to Blu-ray recorders.

As an example, in the software development of single-segment broadcast viewer on a mobile phone, use of the shared framework and components has enabled reduction of development lead-time by approximately 40%.

4.2 Development Environment

To support the rapidly increasing software development workload, in addition to the application framework, an environment for advanced development of software, called the system LSI Simulator, has been implemented. LSIs for embedded use are increasing their scale with the required higher performance, and the development lead-time of software is getting longer. Starting software development after producing LSIs also makes the total product development lead-time longer. For this reason, there is an increasing need for a method of developing and verifying software before producing an LSI. For this purpose, active attempts have been made to accelerate software development and verification by describing hardware behavior in C language [4]–[6]. Figure 8 shows a configuration example of the system LSI Simulator developed for mobile phones. This system LSI Simulator models not only the hardware behavior of the LSI in C language, but also the construction that includes the behavior of devices outside the LSI, such as handover scenarios, memory, power IC, LCD control and others. As a result, almost all of the software in a mobile phone can be operated using system LSI Simulator.

As shown in Fig. 8, software integration and verification on the system LSI Simulator enables the extraction of a great part of the bugs in the logic. Although re-verification is needed after releasing the LSI, the total development lead-time of a mobile phone can be reduced by more than 3 months.

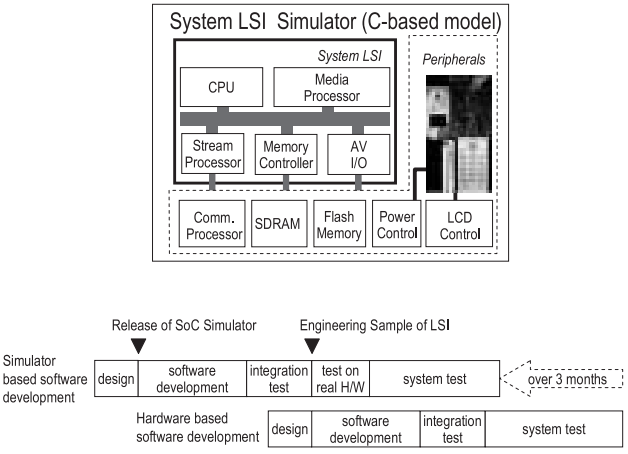

Fig. 8 System LSI simulator and software development process.

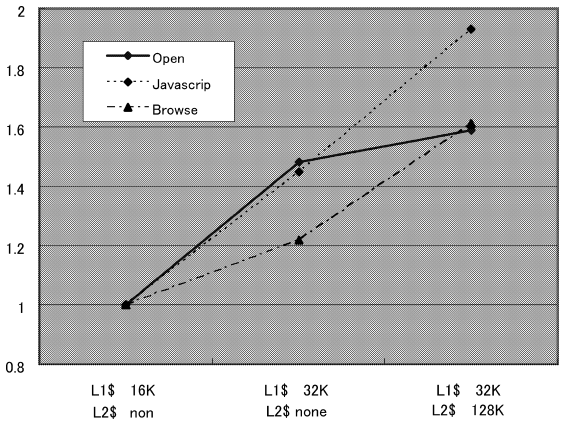

Fig. 9 Performance improvement of applications derived from hardware modification.

4.3 Performance Tuning Using a Simulator

Generally speaking, accuracy of execution time cannot be obtained from a C model simulator. However, as a platform for embedded systems, it is necessary for assuring not only "the logical results of the computations," but also "the physical instant at which these results are produced" [7]. Achieving only the platform function through software development with the framework and system LSI Simulator does not necessarily satisfy the latter condition.

For this reason, we operate the actual software used by our products on the system LSI Simulator not just for an advance development of software, but also to perform advanced software analysis for finding performance bottlenecks. Performance improvement of an embedded system can be achieved by making the hardware higher in performance and faster in speed or by improving the software itself. The features of the system LSI level simulator include the simulation of the entire system LSI with a PC (personal computers), and the capability of tracing all instructions and memory accesses during the operation. This enables performance tuning at the software level as well as an analysis of hardware selection suited for the software. An example of checking the memory architecture according to software type is shown in Fig. 9. By using these results, it is possible to make a judgment on the method of feedback to the hardware or software. For example, Fig. 9 shows that utilizing the level 2 cache has a greater effect on a full browser; this happens because of its large working set. Execution of Javascript and Open GL, due to the limited functions with a smaller working set can be improved by using an level 1 cache of increased size, other than the adoption of an instruction parallel processor.

4.4 Future of the Software Architecture

With the increasing volume of programs, separation of functional achievement and performance tuning is expected to be a time-consuming problem. Therefore, adopting a concept of timing constraints is necessary for the application framework [8]. As a method of achieving this purpose, we have to consider such programming techniques for describing timing constraints separately from the framework such as aspect-oriented programming [9].

5. Low Power Consumption Techniques

In this section, some low power consumption techniques applied to mobile phones are described [9].

The mobile phones demand high-performance of the application processors for the digital television viewer application, 3D games and other heavy-load applications. On the other hand, mobile phones also demand low power consumption of the application processors for voice telephony, music player and other light-load applications.

Consequently, the chip for the mobile phones must achieve appropriate power consumption to each application with wide range of performance.

Figure 10 is a timing diagram of an Intermittent Operation.

It introduces an intermittent operation technique to reduce the power consumption for light-load applications. The general way of power management, processors work as slow as possible to save power. In audio playback, the Application Processor operates at their top speed for as short as possible, and for the remaining time, which will be a lot longer, the processor will be in a sleep state to cut the clock and leakage power.

Figure 11 is a block diagram of system LSI for mobile phones. This chip's architecture is based on the Panasonic's multimedia platform, and it is separated into Non-real-time processing section centering on the general purpose CPU, and real-time processing section centering on the dual-core media DSP. This is in order to release the CPU from the workload of real-time processing. The Audio IO(AIO) domain is separated real time section more finely, and it is cut out the minimum circuit that must always be working. In audio playback, the other application processors decode and

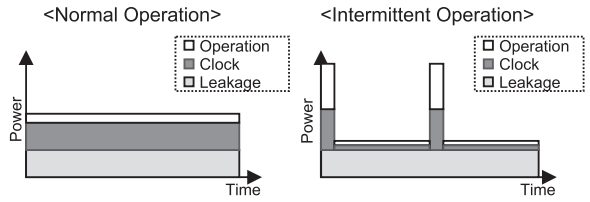

Fig. 10 Timing diagram of an Intermittent Operation.

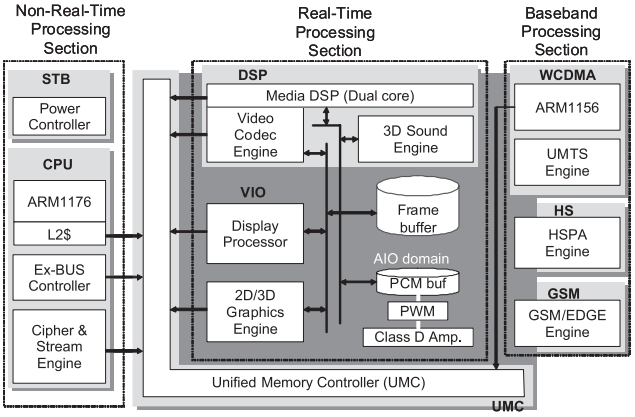

Fig. 11 Block diagram of system LSI for mobile phones.

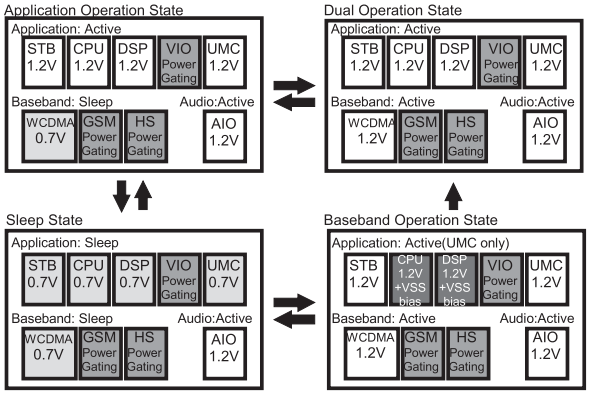

Fig. 12 Power state.

transfer the audio data intermittently, while the Audio IO is always processing sound.

Figure 12 shows the power state diagram. There are 4 power states according to the operating condition of the 2 processors. Vertical arrows are used to indicate transitions of the Application Processor, and horizontal ones are used to indicate transitions of the Baseband processor. During audio playback, the Application Processor repeats transition between the Application Operation State and Sleep State, and the VDD level is raised and lowered.

Figure 13 shows the timing diagram during audio playback. During the operation state, the Application Processor consumes up to 140 mW, by processing at top speed. This operation period is only 6% of the total time, and the power consumption is averaged. During the remaining 94% of the time, the Application processor is in the sleep state

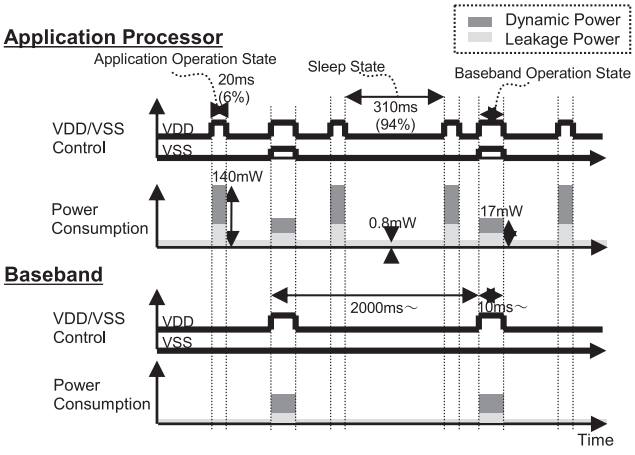

Fig. 13 Timing chart.

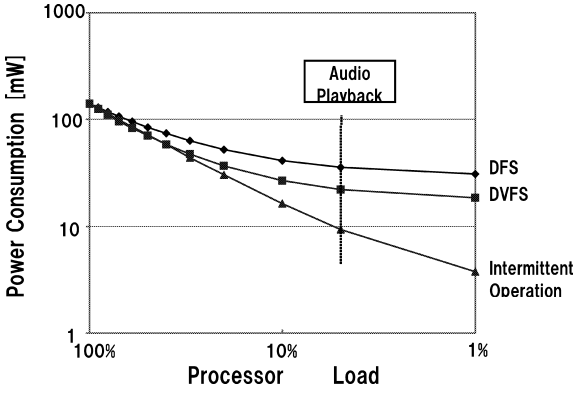

Fig. 14 Evaluation results.

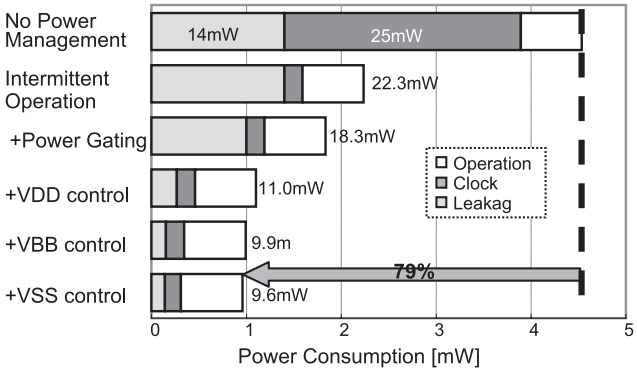

Fig. 15 Measurement results.

with VDD control. The power consumption is only 0.8 mW during the sleep state.

Figure 14 shows a relation between the power consumption and the processor's work load among DFS, DVFS and the intermittent operation with leakage control. The power consumption of DFS and DVFS on this diagram is a simulated value. When the work load is below 10% of the maximum performance, the intermittent operation is more effective than DFS and DVFS in low power consumption.

Figure 15 shows the measured power consumption dur-

ing audio playback at normal conditions. The intermittent operation can reduce the average of the clock power from 25 mW to 2 mW. The leakage control mechanisms also can reduce the average leakage power from 14 mW to 1.4 mW. As a result, the power consumption of the Application Processor is reduced by 79%, which will be 9.6 mW during audio playback.

6. Development Results of the Integrated Platform

The integrated platform is currently adopted by products in five categories: 1) Home-use AV products such as digital TVs and optical disc recorders, 2) Mobile phones, 3) Personal AV products such as camcorders and audio players, 4) Car AV products such as car navigation systems and 5) Safety & security systems. Some extra functions are added or eliminated on demand, when the platform is implemented for each system LSI. As shown in Fig. 16, products adopting the platform have been increasing every year and have reached a total of 156 product families currently.

Shown below are the overviews of two system LSIs developed based on this integrated platform.

• System LSI for Blu-ray recorders

The system LSI for Blu-ray recorders is fabricated using 45-nm CMOS technology and contains 250 million transistors. A microphotograph and the specification of the LSI are shown in Fig. 17 and Table 2 respectively.

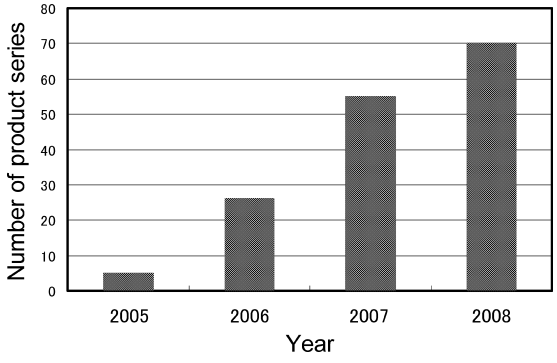

Fig. 16 Number of products families using the integrated platform.

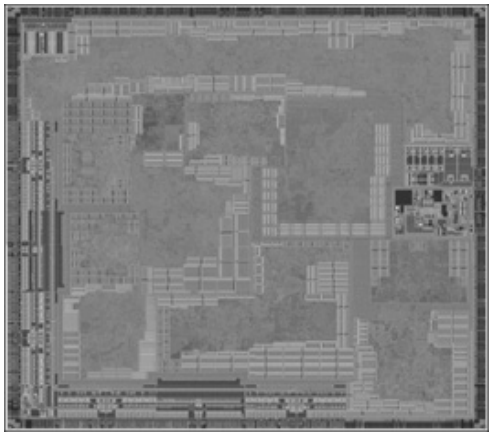

Fig. 17 Chip diagram of system LSI for Blu-ray recorders.

• System LSI for mobile phones

The system LSI for mobile phones [10] is shown in Fig. 18. This system LSI can be divided into two large units: the Application processor and baseband processor. The former has been developed based on the integrated platform.

This system LSI is produced by using a 45-nm process and integrating 230 million transistors. The specification of the system LSI is shown in Table 3.

7. Conclusion

This paper describes the motive for developing an integrated platform and its architecture as well as its actual implementation in various products in the form of a system LSI. Reusing the platform and the intellectual property in actual operations allows delivering of high quality products. Furthermore, a major benefit is achieved by converting the technology developed for featuring a specific product into intellectual property and reusing the intellectual property for other products within the mechanism of an integrated platform. These advantages can be obtained by the following two actions. The first is to define the interface between hardware blocks in order to implement software modules easily. The second is to keep the integrated platform fresh contin-

Table 2 Chip features of system LSI for Blu-ray recorders.

Process	45nm
Number of Transistors	250M
Supply Voltage	Core 1.2V IO 3.3V

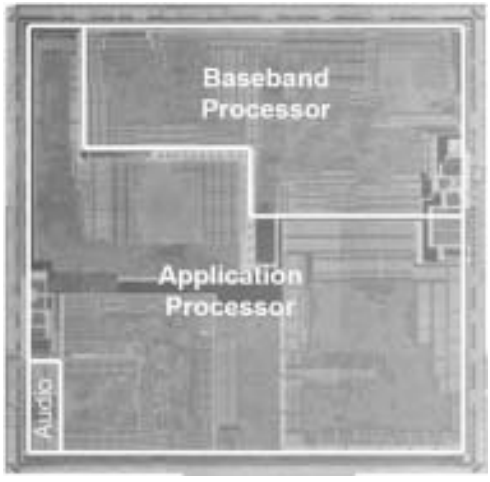

Fig. 18 Chip diagram of system LSI for mobile phones.

Table 3 Chip features of system LSI for mobile phones.

Process	45nm Triple-Vth
Number of Transistors Amount of SRAM	280M 25Mb
Die Size	8.05nmx8.19mm
Supply Voltage	Core 1.2V/0.7V IO 1.85V/2.5V/2.85V

uously. To keep it fresh is performed by selecting technology that is adaptable to multiple products in the early stage and building it into an integrated platform and by making the technology featuring a specific product into easily attachable intellectual property. These continuous activities are important for improving the development efficiency of an integrated platform with well-balanced functionality, performance and cost, and for system LSIs that implement the integrated platform.

Acknowledgments

This integrated platform is developed together with the Strategic Semiconductor Development Center and in-house divisions of Panasonic Corporation.

References

[1] T. Kiyohara, "Multimedia processor-based platform for a wide range of digital consumer electronics," Cool Chips VIII, April 2005. http://www.coolchips.org/index.html
[2] M. Nakajima and T. Yamamoto, "Instruction parallel processor (IPP) architecture on panasonic integrated platform for digital CE," Spring Processor Forum, May 2005.
[3] T. Tanaka, T. Furuta, H. Nishida, K. Yoshioka, and T. Kiyohara, "A pixel level parallel processing architecture for multi-standard video codec," International Conference on Consumer Electronics, Jan. 2006.
[4] A. Mizuno, "H.264 decoder LSI development utilizing C-based high level design flow," TOSHIBA Review, vol.63, no.7, pp.35–38, 2008.
[5] Virtio, Corp., System Verification with Virtio Virtual Prototyping Technology, Virtio White Paper, 2004.
[6] CoWare, Inc., Virtual platforms for software development — Adapting to the changing face of software development, CoWare White Paper, 2005.
[7] H. Kopetz, Real-Time Systems: Design Principles for Distributed Embedded Applications, Kluwer Academic Publishers, 1997.
[8] S. Ren, G. Agha, and M. Saito, "A modular approach to programming distributed real-time systems," J. Parallel Distrib. Comput., vol.36, no.1, pp.4–12, 1996.
[9] T. Yokoyama, "An aspect-oriented development method for embedded control systems with time-triggered and event-triggered processing," IEEE Real-Time and Embedded Technology and Applications Symposium 2005, pp.302–311, 2005.
[10] M. Shirasaki, Y. Miyazaki, M. Hoshaku, H. Yamamoto, S. Ogawa, T. Arimura, H. Hirai, Y. Iizuka, T. Sekibe, Y. Nishida, T. Ishioka, and J. Michiyama, "A 45 nm single-chip application-and-baseband processor using an intermittent operation technique," ISSCC Dig. Tech. Papers, pp.156–157, Feb. 2009.

Junji Michiyama received the B.S. and M.S. degrees in Applied Physics from Miyazaki University in 1983 and 1985, respectively. He has belong to Panasonic Corporation since 1985. He engaged mainly in development of flash memory and MPEG-4 system LSI. Now he is a director of platform development center in corporate R&D division. He is a member of JSAP and IEEE.