KAIST Image Computing System (KICS): A Parallel Architecture for Real-Time Multimedia Platform using Multiple MVPs (TMS320C80)

JaeHo Jeon, GeonYoung Choi*, HeeYong Kim,
JunSeon Kim, HyungSun Kim, and HyunWook Park

Department of Information & Communication Engineering,
Korea Advanced Institute of Science and Technology (KAIST),
207-43 Cheongryangri, Dongdaemun, Seoul 130-012, Korea

* Also at Samsung Electronics Co., Ltd., Korea

ABSTRACT

This paper presents an efficient parallel architecture for the real-time multimedia platform using multiple multimedia video processors (MVP; TMS320C80), which are fully programmable general-purpose digital signal processors (DSP). We have implemented an efficient parallel system, called the KAIST Image Computing System (KICS) for a multimedia platform and an image processing system. The main architecture of the KICS is a message passing model with hierarchically segmented buses. There are two parallel clusters in which two PE's (Processing Elements) are pipelined and the master PE of each cluster can access common global memory at high speed. The applications of the KICS to the MPEG-2 encoder and the volume rendering are introduced. The implemented algorithms are functionally or spatially partitioned and assigned to each PE in consideration with the load balancing and the required data traffic between PE's. The performance analysis for the applications and the general image processing functions are performed. The programmability and the high-speed data-access capability of the KICS are its most important features as a high-performance system for real-time multimedia data processing.

Key words: *Multimedia system, Parallel processing, Digital signal processor (DSP), Multimedia video processor (MVP), MPEG-2 encoder, Volume rendering*

1. INTRODUCTION

Image and multimedia data processing systems usually require high-performance processing powers because the amount of data is very large and they need the real-time operation. Multimedia data processing, image processing, and medical imaging applications require the basic image processing algorithms such as filtering, convolution, Fourier transform, edge detection, boundary detection, image segmentation, and so on [1]. Although some powerful processors or dedicated multimedia image processing chips are reachable according to the rapid development of the semiconductor industry, it is difficult to implement flexible general purpose image processing system based on the specific hardware functionality. The concept that only one specific algorithm is working with the system is not applicable to multimedia data processing system. Because the computation and the required input/output strategies to PE's vary according to the applications and the implementations of the algorithms, the algorithm-specific hardware solution is not adequate. Also, this hardware-based solution cannot be used for new and various applications because of its limited programmability.

The rapid development of VLSI technology has allowed single-chip implementation for video processors which are extensions of general-purpose digital signal processors (DSP) [2]. In addition, programmable processors offer the highest flexibility in the architecture and in the implementation of the algorithms. From these points of view, the parallel architecture using programmable processors is very flexible because various applications can be implemented without changing system hardware and degrading the system performance. For the implementation of this programmable parallel architecture, the

high performance bus architecture is one of the most important consideration, because the data transfer between parallel processors is usually bottleneck for the performance of parallel image processing.

For a long time, parallel processing have been an area of much interest for both the academic community and industry. A large number of proposed architectures have been focused on image processing and computer vision applications because of heavy computational burden in these fields and possibility of the localized data processing in each processing element. Dantu et al. [3] proposed a segmented local bus between adjacent PE's in addition to a high-speed global network. NETRA [4] is a hierarchical architecture of processor clusters. Jeschke et al. [5] described a generic multiprocessor architecture for the MPEG-2 encoder; in this architecture, several processing elements (PE's) are grouped together to form a cluster, and the clusters are connected to each other via a global bus. The RHINE architecture [6] uses a video signal processor (VDSP), and the data path can be reconfigured in this architecture. Many reconfigurable solutions for programmable architectures have been proposed. However, there are still limitations in the sustained performance [7] and the flexibility of these architectures when they are used for various applications.

The KICS is designed as a programmable image processing system which can be cope with versatile applications in general image processing, image coding, medical imaging, and other multimedia applications. It is difficult to meet all the requirements which are needed in various fields. The KICS was architected and implemented to support the following features and the specifications. Among them, its programmable feature enables the system to adapt various applications. Using the high-speed I/O bandwidth of the KICS, the various mapping strategies can be selected without the burden in the required I/O bandwidth.

- *Programmable feature:* It should be possible to be applied to many kinds of new or existing algorithms without changing the system hardware architecture and without degrading the system performance. The programmable video processor is used for each PE and the executable program is downloaded from host computer to each PE.
- *High-speed I/O bandwidth:* The I/O bandwidth of the system almost determines the overall performance for image computing. Since a large amount of data should be manipulated in parallel or pipelined systems, the data transfer bandwidths to each PE and the global memory are the most important factor for system performance.
- *Flexible topology:* Using the simple and flexible topology of the architecture in which each multiple instruction and multiple data (MIMD) PE can operate independently, the various image computing processes can be performed in a given time.
- *Input image format:* The input analog video is digitized and stored in the RGB or the YCbCr 4:2:2 format.
- *Output display format:* The system can display a sequence of image frames which are stored in the global memory at a maximum image size of 1024 x 768 and which has full color with 24 bits. Also, the input video can be displayed in real-time.
- *Storage:* A digitized sequence of image data is stored in and retrieved from the 64Mbyte global memory.
- *Image frame grabbing:* Image is grabbed in resolution of 720 x 480 or 640 x 480 with the YCbCr 4:2:2 or the RGB format. Each component has an 8-bit depth.
- *Host interface:* The KICS is configured as an off-shelf-type system. It can be interconnected to the host system by using a SCSI-2 Wide interface.
- *Application-specific hardware interface:* For a specific algorithm, cost-effective hardware or VLSI-based hardware can be interfaced to the system

In this paper, the parallel architecture and the applications of the KAIST Image Computing System (KICS) are presented. The hardware and the software architectures of the KICS are presented in sections 2. The applications of the KICS to the MPEG-2 real-time encoder and the volume rendering are introduced in section 3. The performance of the KICS is evaluated in section 4, followed by the conclusions in section 5.

2. KICS ARCHITECTURE

2.1. Hardware Architecture of the KICS

The overall hardware block diagram of the KICS is shown in Fig. 1. In the KICS, parallelism and pipelined architecture are mixed. Two clusters, each consisting of two PE's, are parallelized and linked through the global memory. Two PE's inside a cluster are pipelined. The TMS320C80 of Texas Instruments [8], a well-known MVP (Multimedia Video Processor), is used as the processing unit of a PE in the KICS. Inside a MVP, one master processor (MP) and four advanced DSP's (ADSP's) operate in parallel or pipelined fashion. There exist hierarchically segmented global buses for two clusters to access the global memory independently. Also, all PE's are interconnected through data queues. Using the global memory and the data queues, several data and task partitioning methods can be selected. The I/O bandwidth of the cluster is 320 Mbyte/sec for the global memory and 80 Mbyte/sec for the data queues. Each PE has its own local memory and local bus, and the peak performance of each PE is 2 billion operations per second. The key features in the system architecture are programmability of PE, high-speed I/O bandwidth, simple topology, and a flexible structure. An image processing algorithm can be mapped to the parallel architecture by partitioning the data structure or tasks for parallel and pipelined architecture.

The KICS is configured as three functional units: the Image I/O Unit (IOU), the Image Processing Unit (IPU), and the Image Memory Unit (IMU). The master MVP, which is a processing unit in the IOU, manages the overall operation of the KICS. It has the interfaces to the host system, the PE's, the video acquisition, and the video presentation. The host computer is interconnected using the SCSI-2 Wide interface. The programs and the commands for all PE's including the master MVP are transferred from the host computer. All PE's in clusters are connected through the command queues which are used for the program downloading and the command/status passing from the host system. The IOU has the highest priority for the global-memory access among all PE's so that the video module can perform real-time acquisition and presentation. The video block has the features for the real-time display and capture from/to the global memory. Figure 2 shows the block diagram of the video interface. For the real-time capture of the input video, the analog input video is digitized to RGB or YCbCr format. It is reordered to 64bit in the DRAM interface block to be stored in the global memory. For the real-time display of image sequences, an image which is fetched from the global memory is converted to RGB color format and stored in the FIFO. The stored image in the FIFO is displayed through the frame memory and the digital-to-analog converter (DAC). The resolution for the DAC and the window size for the image data in the FIFO can be controlled independently. The captured image data is also displayed through the same path of the real-time display. Some graphics applications using the master MVP are also possible because the master MVP can access the global memory and the frame memory directly through the global memory multiplexer and the frame memory multiplexer.

The IPU consists of two clusters. A cluster includes two pipelined PE's. The first of two PE's, called master PE, can access global memory using a segmented hierarchical bus to the global memory. Each PE has a local memory, data queues, and a command queue. In addition, the master PE also has a global-memory interface, and an application-specific hardware interface. For the pipelined one cycle per column access [9] to the local memory which is implemented with dynamic random access memory (DRAM), a two-way memory-interleaving scheme is adopted. The data queues are inter-PE communication buffers between all the PE's. The command queue is also a communication queue between the master MVP of the IOU and PE's of the IPU. The data queue and command queue are implemented with dual port RAM (DPRAM). For some applications, the master PE has a special interface for a plug-on mezzanine card.

The IMU consists of four modules. A module of the IMU is composed of a memory device and a status arbiter and each module is operating independently. All the access to the memory device is performed by the fast page mode and the pipelined one cycle per column mode. The separated 64-bit buses from processors are multiplexed to access each memory module. The status arbiter of each module controls the arbitration and the bus multiplexer for the module itself. If a processor wants to access a module, it has to occupy the arbiter before accessing the memory. Then, the processor can access the memory device without any contention until it releases the module occupancy or until the IOU tries to access the module. The highest priority is given to a request from the IOU so that any access from the IOU can override the current access of the module from any master PE's at any states. Requests from the master PE's are served in the order received.

Fig. 1. The overall block diagram of the KICS (KAIST Image Computing System).
 (a) The KICS model: The message passing model with hierarchical segmented buses
 (b) The KICS configuration

Fig. 2. The block diagram of video module.

2.2. Software Architecture of the KICS [10]

The overall software structure of the KICS is shown in Fig. 3. The fully programmable feature and flexible algorithmic-mapping are most important considerations in the software architecture of the KICS. Only the boot handler is a built-in software in the master software, whereas the application softwares are implemented by downloading the linked libraries which are stored in the host computer. The task scheduling philosophy between PE's and the required parameters for the applications are also transferred to each PE.

The KICS software consists of the host software and the MVP target software. The host software consists of the host application processes on the Windows API [11] and a communication driver using the protocol of the SCSI-2 Wide interface. The diagnostic API commands the master MVP to monitor and to report the low-level status of hardware devices. The library API is the utility for the general image processing, the computer graphics, and the system library. Required API's are transferred and linked to the master software of the MVP target software using the communication API.

The MVP target software consists of the master software and the PE software. The master software manages the operation of all PE's in the KICS and includes device drivers for video, audio, and operation mode control. Two kinds of communication software are implemented using the double-buffering technique, which are the master communication driver for communication with the host computer through SCSI-2 Wide and the PE communication driver for communication between the PE's of the clusters and the master MVP. The master software is operating on the MP of the master MVP and supervises the operation of its ADSP's and all the PE's. The PE software is running on the MP of the PE. The system manager, cluster task manager, I/O drivers for master software and the command handler, I/O manager, ADSP task manager for the PE software are reside in the MP of the MVP. The ADSP applications, the video application, and the audio application are the core algorithms written in assembly language for the optimization of parallelism because C compiler of the MVP is not yet optimized for parallel operation of the ADSP's.

Parallelism of two clusters in the KICS and four ADSP's in each PE can be performed by spatial or functional partitioning according to the nature of the algorithm. The sustained performance of MVP is highly dependent on software programming, and it is not easy to utilize multiple internal processors such as MP and four ADSP's maximally. Therefore, tight-loop coding and resource management are required to maximize the performance of PEs for real-time requirement.

Fig. 3. The overall architecture of the KICS software.

3. APPLICATIONS OF THE KICS

3.1. An Application for the Real-Time MPEG-2 Encoder [10]

The MPEG-2 standard is more suited to high-end professional applications such as video production, editing, and broadcasting, which require high spatial resolution. MPEG-2 covers a wide range of applications with different bandwidth and presentation requirements, such as multimedia computing, direct satellite broadcast, and high-definition television (HDTV). In our implementation, two processing clusters perform the parallel operation with spatial partitioning of input video sequences. The input image frame is partitioned with some overlapped region, and stored in the global memory modules from the IOU. Each cluster fetches and encodes a half of frame. Inside a cluster, the MPEG-2 encoding algorithm is partitioned according to functional entities such as format conversion, motion estimation, predictive coding (motion compensation, DCT, and quantization), reconstruction, and entropy coding.

For the real-time requirement of the motion estimation in the PE1 pipeline, the dedicated hardware which is called the motion estimation subunit (MESU) is designed as an application-specific hardware of the IPU. The MESU contains motion-estimation-dedicated (block matching) chips for 8x8 block processing. Also, the KICS adopts a new three-level hierarchical search algorithm (TLHS). At the first level, the current macro block (MB) and the search window are decimated with a half sampling rate in the horizontal and the vertical directions. Then, a position with the minimum mean absolute error (MAE) can be obtained from the MESU. The second level compares the MAE's at eight adjacent points around the position. The third level computes the final motion vector with the half pixel accuracy using linear approximations of MAE's [12][13]. The second and third level are processed on the ADSP's of the PE1 pipeline. The ADSP's perform the pipelined coding loop using parallelized multiple operations and the MP manages the data I/O for the MESU and the ADSP's using intelligent packet transfer.

For the PE2 pipeline, the DCT and quantization are performed in two ADSP's and the inverse quantization and inverse DCT (IDCT) are performed in the other two ADSP's. The MP manages the data transfer from/to the data queue and the local memory for the ADSP's operations. The DCT and IDCT require many computation cycles in the PE2 pipeline.

There are many algorithms for reducing the number of operations in 8x8 block DCT [14][15]. However, there are several considerations for the implementation of DCT in the MVP as follows: A balance between the number of multiplications and the number of additions in the algorithm is useful because the ADSP has a separate multiplier and adder, and the ADSP can perform the multiplication and the addition simultaneously in one cycle. Another consideration is adaptability of the algorithm to the split arithmetic operations. The ADSP supports split operation for two 16-bit words of data in a register at a time. Also, the address offset to be accessed must not be remote from the current address. With the above considerations, Chen's and Lee's algorithms for the pipelined DCT and IDCT of eight 8x8 blocks were tested in the KICS. The internal double-buffering technique in the on-chip memory is used for the I/O of the processing data in ADSP's. The ADSP's utilized rounded multiplication and swapping of the data for effective parallelization in data flow graphs. Effective code alignment is important to remove cache miss of instruction, and hardware loops are used to reduce the setup and intermediate steps [16].

Fig. 4. The algorithmic mapping of shear-warp factorization in the KICS

3.2. An Application for the Volume Rendering

The volume rendering is a 2-D representation of a 3-D data set with several slices of tomography such as CT, MRI, and ultrasound. More general definition of the volume rendering is the direct mapping of the essential content of volumetric scalar data fields onto the intensity field that can be displayed on the screen. In earlier efforts, the direct surface visualization was proposed by M. Levoy [17]. It is based on the hybrid physical visual model of calculation of both the reflection and the transmission of light, called shading and classification, respectively. There exist some difficult points in that the algorithm handles a large amount of data set. Therefore, the volume rendering system must be equipped with powerful processing capability and large system memory. Lacroute et al. [18] presented a new algorithm for a fast classification from the parallel and the perspective projection. They used the shear-warp factorization and it includes three conceptual steps; shear and

resample the volumetric slices, project resampled voxel scanlines onto the intermediate image scanlines, and warp the intermediate image into a final image.

In the KICS, the pipeline for the shear-warp algorithm with a parallel projection is considered for implementation. The volume data are spatially partitioned between the clusters from the intermediate image plane. Also, the algorithmic mapping for the volume rendering is functionally partitioned between the PE1 and PE2; the master PE's of each cluster fetch the required scans of slices from the global memory, and perform the shearing and the resampling. The PE2 receives the resampled data through a data queue from the PE1 and performs the projection and the composition to generate the intermediate image of voxel scanlines. The result for the intermediate image from two PE's of each cluster are collected in the master MVP through the command queue and a final image is generated from the warping of the intermediate image. The 8-bit input samples are translated to the 16-bit width data from the input packet transfer and interpolated to get the resampled voxels in a two-dimensional slice, because the resampling weight is same in a slice. Two voxels are processed at the same time because of the 32-bit multiple arithmetic capability in the ADSP. The resampled voxels are transferred to the PE2's through the data queues, and the opacity calculation and the over operation of the voxels are performed in the PE2 pipeline. After the pipelined operation, the scanline of intermediate image are transferred to the master MVP through the command queues, and the master MVP do the warping transform to the intermediate image. The warping process in the master MVP is also pipelined with two clusters. Figure 4 shows the mapping of the algorithm to the KICS.

4. PERFORMANCE ANALYSIS

4.1. Performance for the Image Processing Functions

Table 1 shows the performance of the KICS in performing some image processing functions on a 512x512 image in terms of execution time. According to the nature of algorithms, the partitioning strategies should be selected to reduce the time for required I/O and processing of the algorithmic loop. Notes that the window and level are processed on 16-bit images for medical applications.

Function	Processing time
Thresholding	0.82 msec
Mean Square Error of Two Images	1.23 msec
Histogram Generation	1.9 msec
Interpolation	2.7 msec
Window & Level (16 bits)	3.7 msec
8x8 Block DCT	4.7 msec
3x3 Morphology (Erosion or Dilation)	5.6 msec
JPEG Encoding/Decoding	7.2 msec
Fast Fourier Transform (Complex 32 bits)	63.5 msec

Table 1. The KICS performance of some image processing functions for a 512x512 image.

4.2. Performance for the MPEG-2 Encoder [10]

The performances as a MPEG-2 encoder are estimated in each PE of a cluster, where the spatially partitioned image sequences were processed. The total execution time of the PE1 for a frame of image is smaller than 33 msec as shown in Table 2. The data I/O time and the processing time for the reconstruction are very expensive because it includes the linear interpolation for half pixel accuracy. The computational burden for the motion estimation is compensated using the extra hardware, the MESU. The required time for the DCT and IDCT are also estimated in the PE2. We analyzed Lee's and Chen's algorithms for optimal implementation in the KICS. The required numbers of cycles for eight 8x8 block IDCT's are 4,847 and 4,300 cycles for Chen's and Lee's algorithms, respectively, so that four parallel ADSP's can process IDCT, DCT, quantization, and inverse quantization in real-time.

The PE1 is I/O bounded and the PE2 is computation bounded. That is, the processing time of ADSP's in the PE1 is relatively smaller than the required I/O times, whereas the processing time of ADSP's for PE2 is larger than the required I/O time. The real-time application of the KICS to a MPEG-2 encoder is possible as shown in this section.

Function	I-picture	P-picture	B-picture
Retrieving Frame Sequence	1.62 msec	1.62 msec	1.62 msec
Motion Estimation	-	7.29 msec	10.75 msec
Prediction	7.29 msec	7.29 msec	7.29 msec
Reconstruction	12.96 msec	12.96 msec	-
Total	21.87 msec	29.16 msec	19.66 msec

Table 2. Total execution time for each function in the PE1 pipeline.

4.3. Performance for the Three Dimensional Volume Rendering

Table 3 shows the estimated execution time for each step of the algorithm as shown in Fig. 4. In this estimation, the octree or the run-length coding [18] for the new viewpoint is not implemented. Only three steps for the shear-warp algorithm is directly estimated. The required cycles of ADSP are 3 cycles/voxel and 10 cycles/voxel for step(1) and step(2), respectively. Four ADSP process the spatially partitioned voxel scanlines.

Step	Processing time
(1) Shear and Resample in PE1	158.3 msec
(2) Project and Composite in PE2	518.6 msec
(3) Warp in master MVP	2.5 msec

Table 3. The KICS performance of the shear-warp factorization for a 256x265x256 voxels

5. CONCLUSIONS

The KAIST image computing system (KICS) is a programmable parallel architecture for a real-time multimedia platform. The programmable feature, the high-speed I/O bandwidth, and the flexible topology are the important features of the system design. It uses multiple MVP's, which are one of the most powerful programmable digital signal processors. Its hardware architecture is based on the message passing model with hierarchically segmented buses. There are two processing clusters, each of which consists of two MVP's as PE. Each cluster can simultaneously access the global memory with high-speed I/O bandwidth. The parallelism or the pipelined processing are flexibly configured in the KICS. All the software are downloadable from the host computer for the adaptability to any kinds of applications, such as general image computing, image coding, medical imaging, and other multimedia applications. The performance of the KICS was presented in terms of the execution time for some image processing functions, a MPEG-2 encoder, and the volume rendering. The KICS is suitable as a platform for the general image processing system which needs a high-speed processing capability and a programmable parallel architecture.

6. REFERENCES

- [1] C. B. Basoglu, J. Reeve, Y. Kim, and S. Marquis, "UWGSP8: A Programmable Ultrasound Subsystem for Native Image Processing," SPIE Medical Imaging 1996, vol. 2707, pp. 378-388, 1996.
- [2] V. Bhaskaran and K. Konstantinides, "Image and Video Compression Standards: Algorithms and Architectures," Kluwer Academic Publishers, 1995.
- [3] R. V. Dantu, N. J. Dimopoulos, K. F. Li, R. V. Patel, and A. J. Al-Khalili, "Parallel Algorithms for Low-Level Vision on the Homogeneous Multiprocessor," Computers & Electronic Engineering, vol.20, pp.51-60, 1994.
- [4] A. N. Choudhary, J. H. Patel, and N. Ahuja, "NETRA: A Hierarchical and Partitionable Architecture for Computer Vision Systems," IEEE Trans. on Parallel and Distributed Systems, vol.4, pp.1092-1103, 1993.
- [5] H. Jeschkes, K. Gaedke, and P. Pirsch, "Multiprocessor Performance for Real-Time Processing of Video Coding Applications," IEEE Trans. on Circuits and Systems for Video Technology, vol.2, pp.221-230, 1992.
- [6] Y. Takeuchi, Z. Huang, M. Saeki, and H. Kunieda, "RHINE: Reconfigurable Multiprocessor System for Video CODEC," IEICE Trans. Fundamentals, vol.E76-A, pp.947-955, 1993.
- [7] J. E. Bush, Y. Kim, S. D. Pennington, and A. P. Allenman, "UWGSP7: A Real-Time Optical Imaging Workstation," SPIE Medical Imaging 1995, vol.2431, pp.429-441, 1995.
- [8] Texas Instrument, Inc., "TMS320C80 (MVP) Technical Brief," Texas Instrument Inc., Houston, Texas, 1995.
- [9] Texas Instrument, Inc., "TMS320C80 (MVP) Transfer Controller," Texas Instrument Inc., Houston, Texas, 1995.
- [10] H. S. Kim, K. Boo, S. W. Chung, G. Y. Choi, Y. J. Lee, J. H. Jeon, and H. W. Park, "An Implementation of MPEG-2 Encoder to Multiprocessor System using Multiple MVPs," SPIE Medical Imaging 1997, 1997.
- [11] S. Stanfield and R. Arveson, "Visual C++4 How-To," Waite Group Press, Corte Madera, CA, 1996.
- [12] J. Jeong and W. Ahn, "Subpixel-accuracy Motion Estimation Using a Model for Motion-Compensated Errors," Proceedings of Picture Coding Symposium 1993, Lausanne, Switzerland, Swiss Federal Institute of Technology, pp.13.4-6, 1993.
- [13] J. R. Jain and A. K. Jain, "Displacement Measurement and Its Application in Interframe Image Coding," IEEE Trans. on Communications, vol.COM-29, pp.1799-1808, 1981.
- [14] B. G. Lee, "A New Algorithm to Compute the Discrete Cosine Transform," IEEE Trans. on ASSP, vol.ASSP-6, pp.1243-1245, 1984.
- [15] E. Linzer and E. Feig, "New Scaled DCT Algorithms for Fused Multiply/Add Architectures," Proceeding IEEE International Conference on Acoustics, Speech, and Signal Processing, pp.2201-2204, 1991.
- [16] Texas Instrument, Inc., "TMS320C80 (MVP) Parallel Processor," Texas Instrument Inc., Houston, Texas, 1995.
- [17] M. Levoy, "Direct Visualization of Surface from Computed Tomography Data," SPIE Medical Imaging, vol.914, pp.828-841, 1988.
- [18] P. Lacroute, "Analysis of a Parallel Volume Rendering System Based on the Shear-Warp Factorization," IEEE Trans. on Visualization and Computer Graphics, vol.2, pp.218-231, 1996.