

PROCEEDINGS OF SPIE

SPIDigitalLibrary.org/conference-proceedings-of-spie

DDGIPS: a general image processing system in robot vision

Yuan Tian, Jun Ying, Xiuqing Ye, Weikang Gu

Yuan Tian, Jun Ying, Xiuqing Ye, Weikang Gu, "DDGIPS: a general image processing system in robot vision," Proc. SPIE 4212, Reconfigurable Technology: FPGAs for Computing and Applications II, (6 October 2000); doi: 10.1117/12.402515

SPIE.

Event: Information Technologies 2000, 2000, Boston, MA, United States

DDGIPS—a General Image Processing System in Robot Vision

Yuan Tian, Jun Ying, XiuQing Ye, WeiKang Gu

Institute of Information and Communication Engineering
Zhejiang University, Hangzhou, P. R. China, 310027

Abstract

Real-Time Image Processing is the key work in robot vision. With the limitation of the hardware technique, many algorithm-oriented firmware systems were designed in the past. But their architectures were not flexible enough to achieve a multi-algorithm development system. Because of the rapid development of microelectronics technique, many high performance DSP chips and high density FPGA chips have come to life, and this makes it possible to construct a more flexible architecture in real-time image processing system. In this paper, a Double DSP General Image Processing System (DDGIPS) is concerned. We try to construct a two-DSP-based FPGA-computational system with two TMS320C6201s. The TMS320C6x devices are fixed-point processors based on the advanced VLIW CPU, which has eight functional units, including two multipliers and six arithmetic logic units. These features make C6x a good candidate for a general purpose system. In our system, the two TMS320C6201s each has a local memory space, and they also have a shared system memory space which enables them to intercommunicate and exchange data efficiently. At the same time, they can be directly interconnected in star-shaped architecture. All of these are under the control of a FPGA group. As the core of the system, FPGA plays a very important role: it takes charge of DPS control, DSP communication, memory space access arbitration and the communication between the system and the host machine. And taking advantage of reconfiguring FPGA, all of the interconnection between the two DSP or between DSP and FPGA can be changed. In this way, users can easily rebuild the real-time image processing system according to the data stream and the task of the application and gain great flexibility.

Keywords: Real time, Image processing, DSP, TMS320C6x, FPGA, Robot vision

1. Introduction

The object of image processing and video processing are digital image and digital video data flow. Such data sources have two important features: huge volume of consecutive data and strict limit of frame and field time¹. To overcome these limitations, a real-time (RT) or quasi-real-time (QRT) image processing system must be a high speed computational system. This puts forward high demands to all subsystems when we design a RT or QRT system.

To obtain the performance, many traditional system constructed some special algorithm modules and then linked them, such as the 2-dimension vision intellectual subsystem of our former generation AVL system. Since the task of RT image processing system is becoming more and more complicated, the size of the system is also becoming larger and making design harder. The morass forces us to find a more efficient way to develop an image processing system. Normally, when we design a system, we often partition the task into subtasks which can be realized in different modules and composite them. So, it is valuable to consider how to utilize the existent modules to construct a new RT system and how to develop a reconstructable system module. In this way, we can prolong the life of the modules and shorten the design period.

To achieve the universality and reusability is an efficient way to prolong the life of the module. Along with the development of the integrate circuit, many kinds of CPU (including general purpose CPU, DSP, SIMD and MIMD) and PLD come to life. This makes it possible to realize the reusability of module construction. So we start to develop a "course

(data) driving, reusability based" RT image processing system design.

1.1 The development of microprocessor

Intel manufactured the first microprocessor in 1971. In the following 20 years, microprocessor became the foundation stone of modern information industry. It plays an important role from computer system to the simplest toy. And in nineties, microprocessor gets much higher performance. The appearance of superscalar and superpipeline technique gives the second generation RISC an excellent computability. And in signal processing, DSP employs new architecture. Compared with the traditional DSP, they have such new features:

- (1) Higher data processing parallelism: It has more multiplier and employs SIMD (single instruction multiple data-flow) architecture;
- (2) Special purpose on-chip hardware , such as optimized data path.
- (3) Coprocessor which might be used in DCT or Viterbi decoder
- (4) VLIW architecture

At the same time, DSP has gained a much high computation speed. For example, the TMS320C62x family of TI (Texas Instruments) can run at 200MHz for peak execution of 1600MIPS or 1GFLOPS. And the family will soon get a new member with 5000MIPS peak execution or 3GFLOPS.

With these development, we can achieve most of the real-time task on microprocessor. And because of the programmability, it is viable to construct a general purpose real time platform which is based on microprocessor.

1.2 Programmable Logic Device (PLD) and FPGA computation system

PLD is a kind of user-define-function device. There are many kinds of PLD, such as PLA, GAL, EPLD, CPLD and FPGA. And the scale of PLD increased from original tens of logic gates to millions of logic gates. Attributing to the application of high density FPGA, we can design a system more flexibly.

In a system , microprocessor is variable in time dimension, while it is fixed in space dimension. We can program it with different function at different time. But the jointing hardware is normally fixed in both of space dimension and time dimension. This restricts the flexibility of our design.

A new type of PLD appeared in late eighties - FPGA (Field-Programmable Gate Array) which arises the third type of computational system can help us to resolve this problem. We call it FPGA computational system. It synthesizes the advantages of general purpose computational system and special computational system, that is to say, it is variable in both of space dimension and time dimension.

The basic architecture of FPGA computational system is shown in Fig. 1. It consists of a FPGA and a microprocessor (general CPU or DSP or SIMD or MIMD). As the computing engine, modern SRAM-based FPGA has a very high program speed, and there are some other techniques to accelerate program speed. So FPGA can be reconstructed according to the need of the task, providing task-oriented computing component. In this way, we can gain facilities of the software programmability and the hardware programmability and reduce the design period evidently.

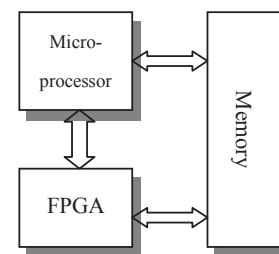


Fig 1. FPGA computational system

Because of the excellent architecture of FPGA computational system, people developed many kinds of FPGA computational system, such as SPLASH^{4,5}, PRISM-I⁶, DFPGA⁷, OneChip⁸ and Garp⁹. Early systems just use commercial FPGA to construct computing-accelerate-board and append it to the general computational system to obtain higher computation capacity. In these systems, the general microprocessor has a loose coupling with FPGA. The subsequent system integrates FPGA and microprocessor into a single chip to enhance the coupling. This expands the concept of FPGA computation distinctively.

The reusability is the standard of the module performance when we design a reusable module. FPGA computational system has great flexibility in both of time-dimension and space-dimension. This makes a great advantage in programmability and reusability. So we draw the conclusion that FPGA computational system is a very excellent typical real-time system. Our task is to utilize FPGA computational system architecture to structure some components of RT image processing system and to assemble a real RT image processing system.

2 The architecture of Double Processors System

Our information processing system is embedded in a PC, so the scalar of the system has an upper limit. To gain a high performance within the constrain, we develop a double C62 processing system which has two C62 as the processing core. In this system, FPGA (we choose FLEX family of ALTERA co.) acts as the joint and control center. The abstract architecture of the Double C62 Processing System are shown in Fig 2. The system consists of Microprocessor Module, FPGA Group Module and Bus Interface Module.

The Microprocessor Module includes C62, Memory Space and its control logic, and some firmware interface, such as JATG port and serial port. Microprocessor is the basic processing cell to perform the subtask .

We setup two processor modules in our system. They both work under the control of the FPGA. As the control center, FPGA is in charge of the communication and synchronization of the two CPU and the information exchange between the outer-system and our subsystem (throughout the master-slave bus and interconnect bus). It can also dispatch processing instructions and subtasks at system level. The flexible architecture of FPGA ensure the performance of our information processing system. According to the task of the system, we can reconfigure FPGA to make the two processors work in pipeline or in SIMD mode to cooperate in an image processing. We also realize the system bus control logic in FPGA. System bus is the interface to interconnect and expand the information processing components and to communicate with the outer system. This is prescribed in the prototype machine. System bus interface is one of the system exterior features.

2.1 Introduction of the microprocessor

We choose TMS320C6x as the microprocessor in our system. The TMS320C6x generation of DSP is the newest one of TMS320 family. The TMS320C62x (C62x) devices are fixed-point DSPs, and the TMS320C67x (C67x) device are floating-point DSPs in TMS320C6x generation. They are code compatible. C62x and C67x both use RISC-like VelociTI™ architecture, a high-performance, advanced VLIW (very long instruction word) architecture, making these DSPs excellent choices for multi-channel and multi-function applications.

The VelociTI is an advanced VLIW architecture to achieve high performance through increased instruction-level parallelism. A traditional VLIW architecture consists of multiple execution units running in parallel, performing multiple

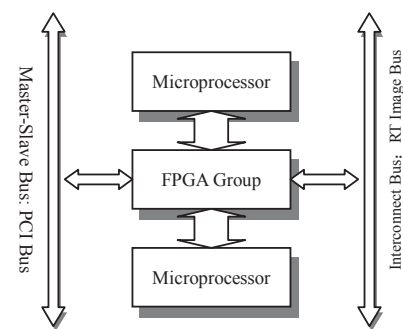


Fig 2. Double C62 system abstract architecture

instructions during a single clock cycle. Parallelism is the key to extremely high performance, taking these DSPs well beyond the performance capabilities of traditional superscalar designs. VelociTI is a highly deterministic architecture, having few restrictions on how or when instructions are fetched, executed, or stored. It is this architecture flexibility that is key to the breakthrough efficiency levels of the 'C6x compiler. VelociTI's advanced features include:

- (1) Instruction packing: reduced code size
- (2) All instructions can operate conditionally: flexibility of code
- (3) Viable-width instructions: flexibility of data types
- (4) Fully pipelined branches: zero-overhead branching

Employing the advanced VLIW architecture, TMS320C62x can run at 200MHz for peak execution of 1600MIPS. This performance comes from the following features:

(1) Instruction Parallelism

The C6x CPU has two data-path, each has 4 independent functional units to achieve different process simultaneously. And as an extension of RISC, the VelociTI is a load-store architecture, so the CPU can load and store data at the same cycle.

A VLIW instruction word includes execute packets to different units. A typical parallelism is eight functional units executing on different operands simultaneously. In this way, it can achieve 1600MIPS at 200MHz CPU clock.

(2) Pipeline

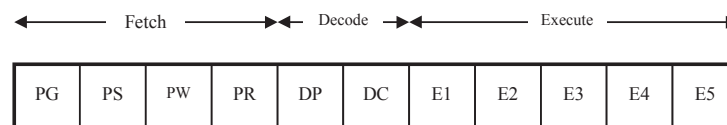


Fig 3 Pipeline stages and phases

All instructions in the C62x instruction set flow the fetch, decode, and execute stages of the pipeline, and each stage consumes several cycles (Fig.3). The fetch stage has four phases: Program address generate (PG), Program address send (PS), Program access ready wait (PW) and Program fetch packet receive (PR). The decode stage has two phases: Instruction dispatch (DP) and Instruction decode (DC). The execute stage of the pipeline requires a varying number of phases, depending on the type of instruction. It covers a maximum of 10 phases. But most of the C62x instructions use only five phases.

Even though execution of a single instruction is more than one cycle, the pipeline architecture execute different instruction fetch of different phase simultaneously. So in full pipeline, CPU can execute an instruction fetch every cycle.

(3) Component Parallelism

In C62x chip, DMA as well as CPU can access the internal and external resources of the chip. DMA is independent with CPU, And many on-chip components are parallel-accessible to accelerate the parallelism. A typical one is internal data memory architecture. So there are three requesters have access to data memory controller : the two CPU data paths and the DMA. Thus, they compete for the resource. They can access any resource in parallel unless resource access conflict occurs, that is, they do not access the same bank of memory or the same peripheral device.

2.2 Memory architecture

The structure of the memory map is very important for our system's performance, because the object of our RT system is huge image-data-processing. In parallel processing, memory space is often used as buffer of large scale of data communication. According to the coupling degree, the memory architecture has two typical types: SMA (Shared Memory Architecture) and DMA (Distributed Memory Architecture). SMA is a kind of tied-coupling architecture. There is a global shared memory in the system, by pooling which all of the processors can share information. On the contrary, DMA is a loose-coupling architecture. Each of the processors has its own local memory. If one of the processors wants to partake the information of the other, it can only get this through the interconnect network.

SMA and DMA both have advantages and disadvantages¹³. On designing the memory group of a RT image processing system, we must consider the features of the applications^{14,15,16,17} to synthesize the SMA and DMA:

- 1) An important factor to affect the efficiency of image processing is the bottleneck of data communication. This requires multiple communication modes: (1)point-to-point;(2)point-to-group;(3)group-to-point;(4)group-to-group. It is obvious that pure SMA and DMA is not a proper solution. For example, pure-shuffle-network is a typical point-to-point communication based net. SMA and VSMA architecture with global-variable is not suitable to this application. Point-to-point communication needs the support of Asynchronous Global Network. And other communication modes need the support of the system memory.
- 2) The system must support the data reform and geometrical transform efficiently. This is important many application, such as 2-dimension unitary transformation and sorting application and other middle-level or high-level processing.
- 3) The system must support global-variable transfer mode to propagate complicate data-structure efficiently. This is important in high-level image processing.
- 4) The node-machine should be a PC system to support task-level parallelism.

Because of these consideration and to utilize the on-chip parallel components of TMS320C6201 and to decrease the scale of the system, we choose the Shared-Distributed-Memory-Architecture (SDMA) to form our system. SDMA synthesizes the advantages of SMA and DMA . In this way, we can improve the parallelism, expandability and programmability of the system. But in the peripheral device of TMS320C6201, the External Memory Interface (EMIF) has no interconnect logic of shared memory just as C40 does. So we must design the share-control logic in our double C62x system. One choice is to use

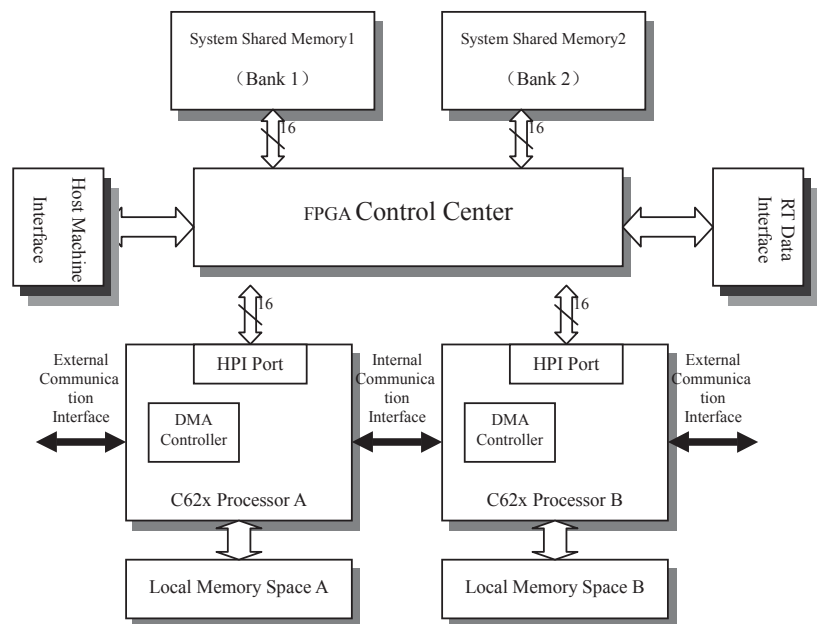


Fig 4 FPGA controlled Double C62x DSMA Diagram

the multiple C40 shared-memory-architecture. That is, use different CE space to construct local-memory space and global-shared space, and the user provides the shared-space access arbitration.

Because C62x has no on-chip shared-space control bus, the global shared memory must be connected to a shared-space access controller (SSAC). SSAC arbitrates the accesses to the shared space from the CPUs, data ports, bus controller and other requester. Compared with the multiple C40 Distributed-Memory-Architecture, we should add some data/address partition logic. And because C62x has many data bus and address bus, the partition logic will increase the complexity of the system and introduce additional access delay.

In Fig.4 shows the realization diagram of our double C6201 RT image processing system. It is a FPGA-computational-system-model based SDMA system. The two C62x processors have only local memory connected to the EMIF, and there is an independent global shared memory space in the system. And to be seasoned with the parallelism of the two CPU and other modules, the shared space has two independent 16-bit-wide banks to form a 32-bit-wide memory space. And the access to the shared space is under the arbitration of the FPGA control center. There are following requesters to the shared memory:

1. Two TMS320C6201 processors. The requests include getting the image data to process, outputting the result data to the shared memory, data exchange between two C62x. C62x access the shared memory through the HPI (Host-Port Interface) port. HPI port is a 16-bit-wide parallel port through which a host processor can directly access the CPU's memory space. The host device functions as a master to the interface, which increase the ease of access. The host and CPU can exchange information via internal or external memory. The host also has direct access to memory-mapped peripherals, such as the serial port. Connectivity to the CPU's memory space is provided through the DMA auxiliary channel.
2. RT data interface. RT data interface is the interconnect bus interface of the prototype machine. It is in charge of the information exchange and cooperation between the subsystem. In our real system, the RT data interface has two user-defined interface: input data port and output data port, and the system shared memory acts as the image frame memory. In this way, the data transfer and processing can be pipeline and parallel to improve the system performance.
3. Host machine load-store processing. The host machine can read/write the shared memory with large block of data through the HPI port. The host machine can access the two banks of system memory simultaneously. To ease our debug, we set the system resource lucent to host machine, that is the host machine can access all of the resource. So the host machine can get the final result of the system to exchange data with other processing system.

The control center arbitrates the accesses to the two system memory from these three kinds of five requesters. And theses accesses can be parallel under some condition. The advantage of the design is: (1) Simplify the design. Not need to realize the share logic in memory address, we can use the C62x's on-chip memory interface to construct the memory architecture, so the system design is very regular and the C62x has a simple local memory composition. (2) Fully utilize the HPI port. As the host machine interface, HPI is often used in information exchange between the processor and the host machine. Attribute to the powerful function of C62x's HPI port, using HPI port can achieve a large block of data exchange to gain higher performance. (3) To exert the usage of the FPGA. FPGA is the center of the system and the host device of the two C62x. It performs the arbitration of the requests from consumer device. Attribute to the programmability of FPGA, the system gains great agility and powerful function. (4) The interconnect communication port between the two processor makes the cooperation of them more convenient and more quick. And the port is very useful in expanding our system.

2.3 I/O interface design

Our information processing system is designed as an accelerating system plugged in PC. The advantages offered by local bus designs have motivated several versions of local bus implementations such as VESA, PCI. They are both open standard for system I/O buses and gain much support. VESA is widely used in 486, but in Pentium or more high performance system, PCI takes place of it. PCI2.1 has following main features: Processor independent, Supports multiple families of processors as well as future generations of processors (by bridges or by direct integration); low power cost; bursting-mode read and write, it can achieve 132 MB/s peak, 264 MB/s peak or 528 MB/s peak; Parallel bus operation; Support system bus, PCI bus, extended bus operation in parallel; Software transparent; Hidden bus arbitration; Reduced pin amount.

Because the mainstream of PC has PCI controller and PCI extend slots, and because of its high performance and the embed requirement of our subsystem, PCI is our best choice. PCI bus interface logic is one of the outer features of our information processing subsystem.

Interconnect bus is a real time bus to connect multiple information processing subsystems. It is in charge of high speed data exchange between subsystems and coordinate the operation of two processors. Most of time, it works at full load. So the main design principle is: simple logic, high stability. To achieve it, we designed the different functions separately. During designing our subsystem components, we setup two sets of real time parallel buses and serial buses to get more flexibility.

2.3.1 Parallel Bus

Parallel Bus acts as a transfer channel for a large quantity of data. In real image system, it is mainly responsible for the input of image data and the output of the result. So the efficiency and speed of transfer is the goal of our design. In our information processing subsystem, Parallel Bus is connected directly to the central controller FPGA. According to different system topology, its interconnection logic can be redefined or reconfigured. At the same time, Parallel Bus connect with FPGA as a client of system central control unit, which makes the bus able to access all of the system resource, including system memory, C62x, and host machine interface. In practice, as main channel for transferring image data, its task is exchanging data between two system's global memory, accessing or controlling the shared memory of the interconnected systems.

The Parallel Bus we designed is a kind of uni-master bus, that is to say, one side of transfer is configured as master by hardware and can't change the role since then. It consists of 32-bit-wide data/address bus and 4-bit-wide control bus. Its data/address bus is bidirectional, *Address* and *Data* are multiplexed. Analog to other kind of bus, four control pins are used for bus response, including CLK, Frame, R/nW and TRDY signal. CLK is the clock signal driven by master, usually we use 33MHz clock from C62x working environment. In order to improve transfer efficiency and synchronization, we use synchronous bus as prototype. Frame is control signal for one frame of data transfer. When it becomes valid from invalid state, there will be a transfer of a frame of data; otherwise, it indicates there is the last transfer. R/nW signal's level indicate whether read or write during transfer. TRDY is the slave device ready signal. During one frame transfer transaction, only when slave device is ready can transaction be complete. The bus usually work in burst mode to improve speed, burst length can be set to a large number. Because of the similarity of read/write phase with PCI, there is no necessary to introduce it here. During the address phase, address can be encoded of some special addresses, this makes it possible to access all resource.

In fact, with the help of serial bus, the master terminal can hand over authority to the slave in our uni-master bus. In

this way, we can realize multi-master bus architecture. But as soon as hand-over succeed, the bus still works in uni-master mode. To simplify bus logic and make high performance data transfer implemented easily, we didn't implement multi-master in our Parallel Bus. Since the data/address bus is up to 32 bits, we can access 4G space. Making use of large space, we can implement one-to-multiple parallel bus interconnect without difficulty(namely encode different devices with different address in Parallel Bus). Parallel Bus support more than one device, which makes it possible to implement multiple architecture mode when interconnect systems. This means in application, we can find a data-stream-oriented solution according to different data stream.

2.3.2 Serial Bus

Our double C62x system use the on-chip C62x Serial Bus Interface McBSP (Multi-channel Buffered Serial Port) to construct the Serial Bus. Here, the usage of the Serial Bus is to transfer message between systems or processors to synchronize the operations. The messages include System Processing Complete, Data Read, Parallel Bus Authority Hand-over and so on. The definitions of the messages are setup in the application system software.

McBSP is the Serial Bus standard of TI's TMS320 chip family. The McBSP provide:

- (1) Full-duplex communication
- (2) Double-buffered data registers, which allow a continuous data stream
- (3) Direct interface to industry-standard codecs, analog interface chips (AICs), and other serially connected A/D and D/A devices
- (4) External shift clock or an internal , programmable frequency shift clock for data transfer
- (5) Autobuffering capability through the 5-channel DMA controller
- (6) A wide selection of data sizes, including 8, 12, 16, 20, 24 and 32 bits
- (7) Programmable polarity for both frame synchronization and data clocks

A TMS320C6201 has two McBSP ports. In our double C6201 system, we use two of them to interconnect them and the other two as the communication ports to give the system 2-dimension freeness. Compared with C40's 6 communication ports, we have more limit in system interconnection. We often connect the information subsystems in a closed loop or linearly.

3 Multi-DDGIPS interconnection

In application, we often interconnect multiple processing modules to achieve a real-time image processing. The DDGIPS can be interconnected in different modes according to the task. The abstract model of the DDGIPS is shown in Fig 5. It shows the outer features of the system: one PCI-based master-slave bus, two FPGA-based parallel interconnect bus and two McBSP-based serial bus. The master-slave bus has poor predicability because of its multi-course features. So we realize the interconnection through the two parallel bus and the two serial bus.

3.1 Data-oriented pipeline-architecture interconnection

Normally, the low level image processing system has high correlation with the image data flow structure. And due to the data structure, pipeline architecture is

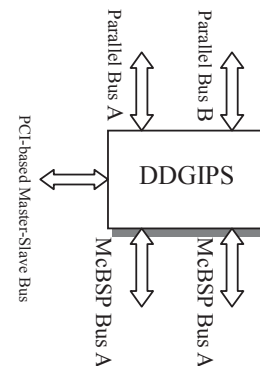


Fig 5 Interconnect Model

familiar in image processing. Our DDGIPS can achieve pipeline-architecture through simple interconnection.

A typical pipeline-architecture interconnect mode is shown in Fig 6. In a single DDGIPS, the parallel bus A is set as a unilateral input parallel bus, and bus B is set as a unilateral output parallel bus. So bus B can gluelessly connect with bus A of the following DDGIPS to construct a pipeline-architecture image processing system.

In some applications of RT image processing system, the volume of the data exchanged between the sequent stages is very huge. Our DDGIPS has two banks in the memory map, so the data exchanging can be parallel with the image processing to improve the efficiency of the pipeline.

3.2 Flexible interconnect architecture

In some general purpose image processing system, especially not data-flow oriented middle-level processing system, the goal of parallel architecture system design is to construct a tied-interconnect-coupling, low communication overhead processing array, such as double loop network and super-cube architecture network. Different topology has different requirement to the freeness of the interconnected subsystem. But in our DSGIPS, freeness is only 2° . This is a disadvantage to form a traditional interconnect, but we can design a flexible bus interface which can connect multiple devices.

Just as shown in Fig 7, we can construct a point-to-group interconnection by using the parallel interface. System A is configured as master device of the bus while all the other devices hanged on the bus must be slave devices. The accessible resources of all systems should have unitive addresses to distinct.

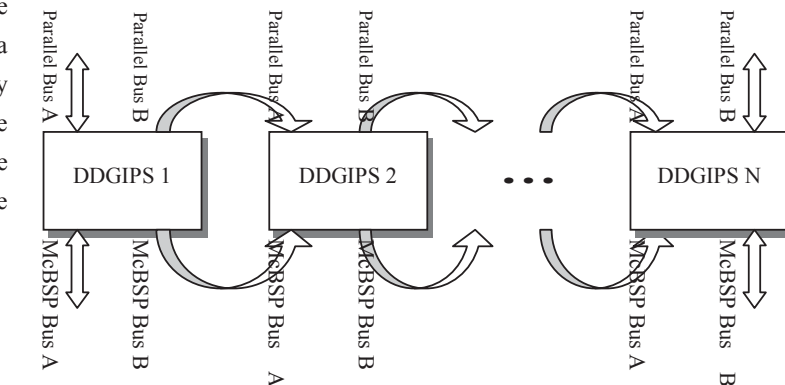


Fig 6 Pipeline Interconnect

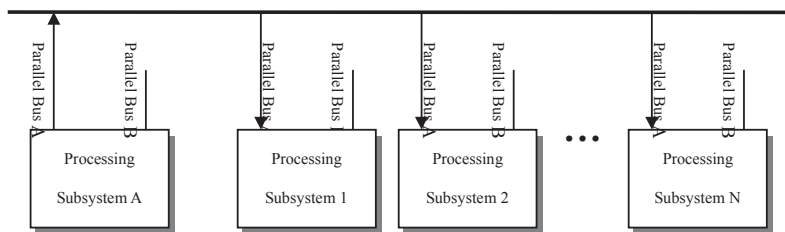


Fig 7 point-to-group diagram

To realize the model, we can use two parallel buses to construct two point-to-group interconnect buses to exchange data between different system. And we can use McBSP's ability of multi-channel communication to make the data exchange more efficient and the system interconnection more flexible. The diagram is shown in Fig 8.

4 Conclusion

Attribute to the powerful computational ability and the flexibility of DDGIPS, we can apply it directly in many RT image processing applications or expand it to a large scale system by interconnecting DDGIPSs at different processing level to achieve some complicated RT task.

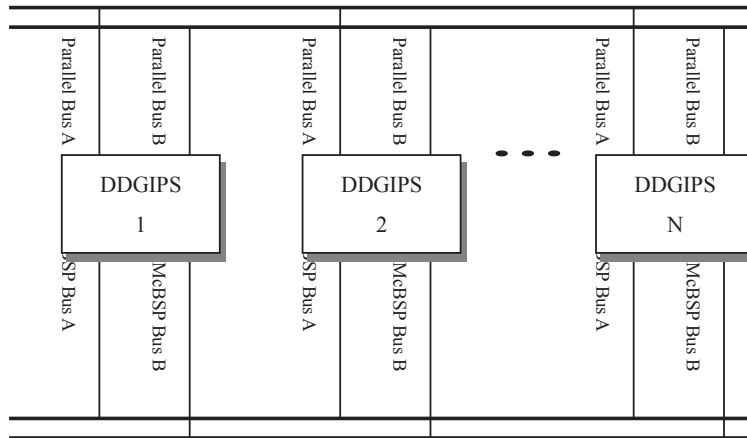


Fig 8 Flexible Interconnect diagram

REFERENCES

1. Li HongSong, *Digital Video Techniques and Applications*: Tsinghua University Publishing Company, May 1997
2. Fan Chengfa, *The Research of Intelligent Image Comprehension System and 2-Dimemsion Vision Comprehension System in ALV*, The Ph. D dissertation in Zhejiang University, Sep. 1998.
3. Wu Di, *The Architecture of 2-Dimemsion Vision Comprehension System in ALV*, The Master dissertation in Zhejiang University, Mar. 1994.
4. Gokhale M, et al., *SPLASH: A Reconfigurable Linear Logic Array*, Proc. Interl. Conf. on Parallel Processing, 1990.
5. Gokhale M, et al., *Building and Using a Highly Parallel Programmable Logic Array*, IEEE Computer, Vol. 24, No. 1, 1991.
6. Athanas P M, Silverman H F, *Processor Reconfiguration through Instruction-set Metamorphosis*, IEEE Computer, Vol. 26, No. 3, 1993.
7. Dehon A, *DPGA-coupled Microprocessors: Commodity for the Early 21st Century*, Proc. IEEE symposium on FPGAs for Custom Computing Machines, 1994.
8. Wittig R D, Chow P, *OneChip: A FPGA Processor with Reconfigurable Logic*, Proc. IEEE symposium on FPGAs for Custom Computing Machines, 1996.
9. Hauser J R, Wawrzynek J, *Garp: A MIPS Processor with a Reconfigurable Coprocessor*, Proc. IEEE symposium on FPGAs for Custom Computing Machines, 1997.
10. *TMS320C62x/C67x CPU and Instruction Set Reference Guide*, Texas Instruments, March 1998.
11. *TMS320C62x/C67x Programmer's Guide*, Texas Instruments, February 1998.
12. *TMS320C6201/C6701 Peripherals Reference Guide*, Texas Instruments, March 1998.

13. Yan XueQiang, *Research and Design of a High Performance DSP (TMS320C40) Based Multiprocessor System*, The Ph.D dissertation in Zhejiang University, Nov. 1997.
- 14 Ying BiShan, *The Research and Realization of Parallel Program Environment and Shared Memory Architecture*, The Ph.D dissertation in West-North Engineering University, Mar. 1994.
15. Ying BiShan, etc. *The Principle and Realization of MPP System with Distribute-Shared Memory*, Micro-Computer Development, No. 3 1993.
16. Michael and Songnian Zhou, *Algorithms Implementing Distributed Shared Memory*, IEEE Computer, May 1990.
17. Bill Nitzberg and Virginia Lo, *Distributed Shared Memory: A Survey of Issues and Algorithms*, IEEE Computer, August 1991.
18. PCI Special Interest Group, *PCI Local Bus Specification, Production Version*, Revision 2.0, April 30, 1993.
19. VESA, *VESA VL-Bus Proposal*, Version 2.0p, July 8, 1993.