Implementation of Real-Time Parallel Processing in A Motion Control System

C. C. Chan, Fellow, IEEE, Zhu meiling

Department of Electrical and Electronic Engineering.
The University of Hong Kong, Hong Kong
E-mail: ccchan@hkueee.hku.hk

Abstract

The paper proposes an architecture and implementation of a motion control system with a master-slave multiprocessor mode. Some major problems which must be considered in multiprocessor systems design, including multiprocessor system architecture, interconnection network, hardware circuit design and software design are studied.

1. Introduction

The research and development of real-time parallel processing in which multiprocessor cooperates closely together has been a very active area. Industrial automation applications may involve control of processes, data acquisition, motion control system, etc.[1-3]. Multiprocessor systems make high-throughput, real-time multi-task parallel processing feasible. Multiprocessor systems are very complex, they include processors, memory modules, I/O interface units and an interconnection network. The interconnection network structure is a very important part of such systems, because it allows communication among processors and memory modules, and therefore data exchange among processors. In general, no standard mode for the interconnection network exists. It is always designed according to requirement of each application. In this paper, the multiprocessor system architecture and the interconnection network for a motion control system with the master-slave mode for parallel processing is described.

2. Multiprocessor Parallel Processing Architecture

Generally, a motion control system has the following major control functions:

1) I/O interface units:
Including CRT display, keyboard, operation panel.

2) Servo analogy output, position sensing and feed.

3) System management and data processing:
Including arrangement and scheduling of control tasks, motion locus computation and error compensation, etc.

According to real-time requirements, a multiprocessor system architecture is used in the motion control system to partition the control tasks into subtasks. These subtasks are executed closely but independently by three processors. Each processor is required to complete a part of tasks. Fig. 1 shows the architecture of motion control system with three processors. The advantages of the multiprocessor system architecture with the master-slave mode are obvious:

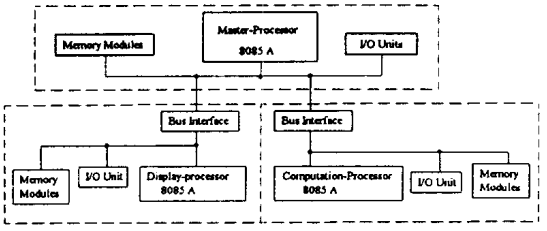

Fig. 1 Multiprocessor System Architecture with the Master-Slave Mode

- Partition control tasks into subtasks, then distribute these subtasks among subsystems so as to allow each subsystem to complete stipulated control functions separately.
- Processors execute in parallel at the software instruction level, so that the motion control system abilities in data processing and real-time response increase.
- All being managed by one operating system that provides management of system hardware and software sources, cooperation and scheduling of control tasks.
- Each subsystem has its own CPU, local memory modules and interface units to form a microprocessor. Only one microprocessor called master-processor is at the master level. The master-

0-7803-2018-2/95/$4.00

processor controls the management, cooperation and scheduling of overall tasks. Other processors of the system are called slave-processor. These slave processors execute in parallel their own programs to complete their own subtasks.

- Each slave-processor is at the same level, however, controlled by the master-processor.
- Each processor is connected in tightly coupled mode [4]. Communication between the two slave processors is realized by the master-processor.

The motion control system consists of three Intel 8085A microprocessors, each one with its own memory modules, interface circuits, encoders, etc. According to different tasks they execute, the master-processor is responsible for system management, cooperation and scheduling of tasks; the display-processor is responsible for CRT display, keyboard and operation panel; the computation-processor is responsible for motion locus computation, position axes servo and I/O, respectively.

2.1 Interconnection network

The interconnection network is a very important part in a multiprocessor system architecture. It may be connected in I/O data path loosely coupled mode or in common address space tightly coupled mode. The former mode is simple in system structure. However, its communication rate is slow. It requires intermediate processes, and thus overhead. The latter mode provides a very fast data transfer to adapt to real-time response requirement. Here the tightly coupled mode is applied in the motion control system. The objective of interconnection network is to connect processor with relative memory modules at the correct time to exchange information. When more than one processor requests access to the same memory modules simultaneously, one of the main problems is that the contention problem may arise for the use of the same communication path. The contention problem of communication path should be eliminated in multiprocessor system design.

From the structure point of view, an efficient mode is the partition of the set of memory modules into two groups. The first group for local-memory modules consists of a set of independent memory modules, each one associated with a processor and accessible only by the processor. These memory modules save the programs and data of each processor. The second group for common memory modules consists of a set of memory modules that the master-processor and the slave- processor may access, as shown in Fig. 2. Each of subsystems has its own independent special memory modules. There are also common memory modules for the master-processor and the slave-processor. The master processor may obtain the access to common memory modules through control the bus interface. In design, the common memory modules are set a space belonging to memory modules of the slave-processor, and accessed by the master-processor when necessary. The common memory modules are the physical medium of exchange information between the master-processor and slave-processor.

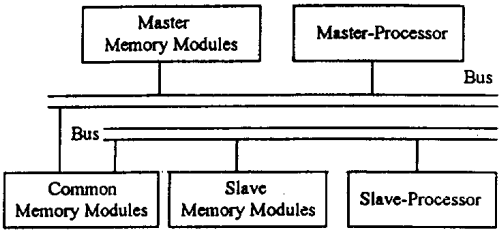

Fig. 2 Interconnection network with the Master-Slave Mode

2.2 Communication Strategies

Communication protocol is another critical problem to design the multiprocessor system. Communication means communication protocol among multiprocessor system. It indicates the queue hierarchy of each processor, information exchange mode, information sequence, and code form of basic data element.

Two operating modes exist in the multiprocessor system structure. They are multiprocessor mode and partitioned mode. The multiprocessor mode provides processor-processor communication via the common memory modules. The partitioned mode, each CPU is treated as a separated system having its own control program and data. The master-processor is in charge of operating the system functions of scheduling, resource allocation, etc. Slave-slave processors communicate via the master-processor. Hence, the queue hierarchy of three processors is determinate. The communication priority between two slave processors and the master processor is determined by the master-processor. The priority arrangement eliminates the contention problem of communication path.

The communication strategies indicate the control decisions when information exchange. In the structure of common memory modules interconnection, the processes of information exchange are classified as the following three steps:

1) Create communication path between the master-processor and the slave-processor:
   The master-processor sends a bus request software instruction to the slave-processor. Then, the slave-processor submits its bus control power to the

master-processor. The master-processor obtains the control power, and creates the communication path accessing to common memory modules.

2) Information exchange between the master-processor and the slave-processor:
   According to transmission form of information, the master-processor reads information from the slave-processor or writes information to the slave-processor via the common memory modules.
3) Delete communication path:
   The master-processor sends a bus release software instruction upon information exchange completion. Then, the slave-processor recovers its bus control power and the common memory modules are accessed by the slave-processor once again.

3. Hardware Design

Fig. 3 shows the function block of the motion control system. This section will describe two major problems in some detail in design: communication path circuit design and common memory modules interface circuit design.

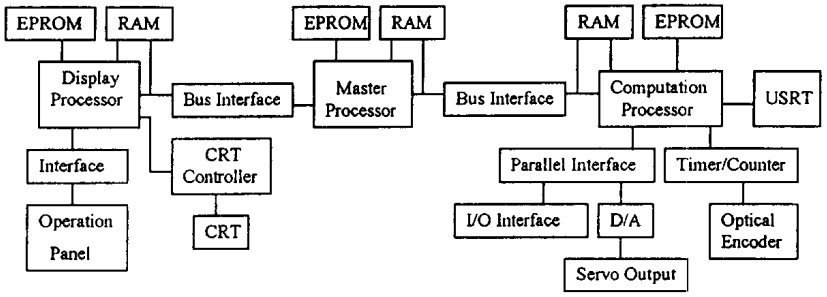

Fig. 3 System function Block

3.1 Communication Path Circuit Design

As stated above, when the master-processor requests communication with any slave-processor, the communication path must be created firstly, as shown in Fig. 4.

The master-processor makes encoder input control signal pin $\overline{G}$ low level enable via reading a special address instruction. Then the encoder outputs a low level signal at pin A, or the input S pin of R-S latch enable. The output pin Q of R-S latch is high level, or the master-processor sends request HOLD signal to the slave-processor. HOLD signal indicates that the master-processor will use buses of the slave-processor. Once the slave-processor receives HOLD request signal, it will stop using buses, or buses of the slave-processor is in the high-impedance state. Then, the slave-processor sends HLDA response signal to the master-processor. HLDA signal indicates that the slave-processor has submitted its bus control power, or common memory modules are allocated to the master-processor. Then the master-processor obtains the access control to common memory modules.

When communication completes, the master-processor sends a software instruction to make encoder output pin B low level enable, or the input R pin of R-S latch enable. Then, the output pin Q of R-S latch is low level, or HOLD pin of the slave-processor is low level. The slave-processor recovers its bus control power, or the common memory modules are reallocated to the slave-processor. And at the same time the communication path is deleted.

The communication process is completed in an interruption program of the master-processor.

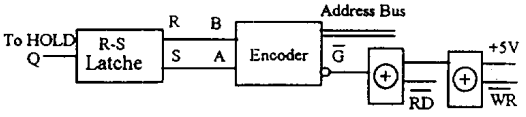

Fig. 4 Creation and Deletion of the Communication Path

3.2 Common Memory Modules Interface Circuit (CRAM)

CRAM is the physical medium of exchange information between the master -processor and the slave-processor.

In order to connect related processor to CRAM and eliminate the case in which the buses of two subsystems are connected directly or indirectly. It is necessary to design bus isolation switch. When any a subsystem requests access to CRAM, the bus isolation switch must turn on to connect related subsystem to CRAM, and vice versa, the switch must turn off. Each subsystem must access to CRAM in share-time mode. Hence, the CRAM logical interface circuit must be designed. Its task is to control bus isolation switch on or off to connect the requesting subsystem to CRAM and prohibit other subsystems access to CRAM at the same time. Fig. 5 shows the CRAM interface circuit.

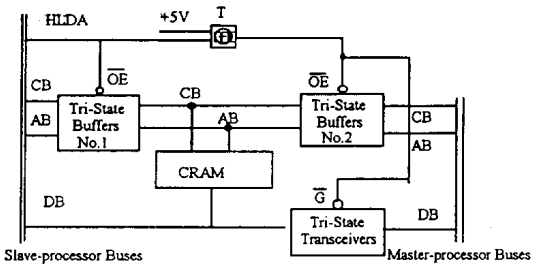

Fig. 5 CRAM Interface Circuit

Data bus (DB) uses tri-state transceivers with bi-directional transmitting function as bus switch. The data bus switch on or off and direction of data transmission may be controlled by the input $\overline{G}$ pin and DIR pin separately.

Address bus (AB) and control bus (CB) use tri-state buffers as buses switch. Their on or off may be controlled by output enable pin $\overline{OE}$. The CRAM logical interface circuit consists of HLDA signal of the slave-processor and gate circuit T. Its control process is described as below:

Normally, HLDA signal of the slave-processor is low level, the buses between the master-processor and the two slave processors are isolated by the tri-state buffers and tri-state transceivers. When the master-processor requests exchange information, the slave-processor outputs HLDA high level, and buses of the slave-processor are in the high impedance state. The No. 1 tri-state buffers turn off to isolate the slave-processor access to the CRAM, and at the same time, the No. 2 tri-state buffers turn on to connect master-processor to CRAM. CRAM becomes a part of the master-processor memory modules. In this way, the multiprocessor system completes the communication interconnection between the master-processor and the slave-processor, and realizes data exchange between the master-processor and the slave-processor.

4. Software Design

To complete control tasks, the motion control system composed of multiprocessor must rely on each processor to execute tasks cooperatively. In this section, the allocation of tasks and software control will be described.

4.1 The Master-Processor

The major tasks of the master-processor include system management, tasks allocation and process arrangement of system software and hardware sources; program management of different kinds of work mode; motion control program management (including edit, interpretation, and modification of numerical control program); diagnosis analysis of system running state; information exchange between the display-processor and the computation-processor.

The control software of the master-processor includes one main program and two interruption programs. Main program is a loop program which manages system initialization, diagnosis, work mode selection, etc. RST 7.5 and RST6.5 interruptions are used as communication between the master-processor and two slave processors. The display-processor sends RST 7.5 interruption to the master-processor at 20ms interval; the computation-processor sends RST 6.5 interruption to the master-processor at 10ms interval. In the two interruption programs, the master-processor, on one hand, sends HOLD request to make slave-processor enter HOLD mode, and on the other hand, the master-processor exchanges information with CRAM. The master-processor deletes the HOLD mode to recover normal operation of slave-processor upon communication completion.

4.2 The Display-Processor

The display-processor controls the operation panel, information display and exchange. Its control software includes one main program and two interruption programs. The main program is a loop program. It controls panel operation and saves related display characters to display buffers. RST 6.5 interruption program displays the content of the display buffers and produces RST 5.5 interruption. The RST 5.5 interruption program acquires the state of operation panel.

4.3 The Computation-Processor

The major tasks of the computation-processor are motion locus computation, position servo control, position feed and I/O control.

The control software includes one main program and three interruption programs. RST 6.5 interruption produced by timer at certain time interval controls locus computation and position servo. RST 5.5 interruption program controls reference pulse orientation and correction. INTR interruption is used as serial process.

5. Conclusion

The proposed motion control system has been applied in laser cutting machine. The practical application has shown that the multiprocessor system architecture with the master-slave mode enhances system performance and throughput.

References

[1] H. W. Lawson, *Parallel Processing in Industrial Real-Time Application*. 1992

[2] W. Li and R. Venkatesan, "A highly reliable parallel processing controller for vector control of AC induction motor," *Proeedings IECON' 92 Conference*, PP. 43-48, Nov. 1992.

[3] H. Le-Huy, "Microprocessors and Digital IC' s for Motion Control," *Proceedings of The IEEE*, vol. 82, no. 8, pp. 1140-1164, August 1994

[4] Philip H. En slow Jr., *Multiprocessor and Parallel Processing*. 1974

[5] M. H. Miller, K. P. Garrard, T.A. Dow and L. W. aylor, "Controller design for a modern diamond turning machine," *Proceedings APSE Annual Conference*, pp. 62-65, 1991.