ISE-TR-86-56

A LOOSELY COUPLED MULTIPROCESSOR SYSTEM : ADMS

—— BASIC DESIGN ——

by

Sanae Amada

Masashi Tsuchida

Yutaka Sato

June 6, 1986

INSTITUTE
OF
INFORMATION SCIENCES AND ELECTRONICS

UNIVERSITY OF TSUKUBA

A LOOSELY COUPLED MULTIPROCESSOR SYSTEM : ADMS

---BASIC DESIGN---

by

Sanae AMADA*, Masashi TSUCHIDA**, and Yutaka SATO***

* Institute of Information Sciences & Electronics,
University of Tsukuba****.

** The System Development Laboratory,
Hitachi Manufacturing Co. Ltd..

*** Doctoral Program in Engineering, University of Tsukuba.

****Sakura-mura, Niihari, Ibaraki, 305 Japan.

Abstract

In the recent status of increasing of the distributed processing system, the computer architecture must be considered more seriously for the improvement of the system throughput. We have an idea to apply a loosely coupled multiprocessor configuration as a computer for the system. In this paper, we describe on the basic design of above mentioned configuration containing the introduction of a high-level architecture.

Contents

1. INTRODUCTION

In these ten years, distributed processing systems have taken the place of centralized processing systems. Reasons of the replacement are;

- conveniences of the distributed system on the user,
- the remarkable cost down of the hardware introduced by many innovations of semi-conductor processes, and
- the progress of software technologies, etc..

However we found that computers applied to the distributed system were not enough tuned for the system. Because, super-minis and medium or small size general purpose computers were used as processing elements for the system, and they had the same architecture to ones developed for the centralized system. Thus, we started our investigation in 1981.

The final target of our study is to develop a computer most fittable to the distributed environment. In the way of our study, we found that a loosely coupled multiprocessor architecture might have the enough fitness for the system. The architecture would have many not enough understood problems to use as a processing element in the distributed environment. However, the introduction of so-called high-level architecture might have many advantages to realize a general purpose processing element with the loosely coupled architecture.

We named our system as ADMS (advanced distributed multiprocessor system).

2. REQUIREMENTS FOR THE SPECIFICATION

A computer to use with a distributed environment ought to have following features.[1]

* Extremely high reliability.
* Enough flexibility of the system.
* High level security.
* Good interface for the user.
* High performance for general purpose uses.
* Multi-language applicability.

Ordinarily, in a distributed systems, plural computers are placed at plural independent sections of the organization. The fact that the computer is set up in not a centralized computer room but a work-shop has to be considered cautiously. Specifications of the computer ought to be influenced by the evaluation of the following items.

(1) The computer will process the information of the work-shop in more direct manner than the case of centralized system. That is the highest cause of conveniences of the distributed system. However, if the computer goes out of order, the tasks of the work-shop may be stopped and the influence will be fatal. And, computers may placed at a long distance from some service point of the system manufacturer.

So that, the computer has to have extremely high reliability.

(2) In the centralized system, the fluctuation of the volume of tasks in many work-shop in the organization is summed up and averaged. This averaging will have an effect to reduce the fluctuation of the load of the computer. In the distributed system, the fluctuation of the tasks in a work-shop

will have a direct influence to the computer both in quantity and in quality. So, the computer for the distributed system must have the enough flexibility for the variety in quality and the enough expandability for the fluctuation in quantity. It is preferable that the computer will be able to correspond to these transitions without the replacement of the whole computer system.

(3) In the distributed system, the computer will be operated by a large number of person in the work-shop. Still more, the majority of them are not so skilful as the person in the computer room on the operation and programming. As a result, they may violate the computer with some misoperations or malicious attempts. So, the computer must have the enough ability for the security. Still more, it must have a good interface for the user.

(4) Various kinds of tasks will be processed with the computer. For instance, tasks concerning with a personal management, a transaction process, CAD, CAM, and a management control of the section may be processed with a computer at a time. So, the computer must have the high performance for general purpose uses, and have the enough multi-language applicability.

3. LOOSELY COUPLED MULTIPROCESSOR

By intuition, we understand that a multiprocessor architecture must be effective for the clearance of requirements mentioned in the previous chapter. Still, we must prove the effectiveness of it in the engineering level, of course. Certainly many multiprocessor systems are under practical

uses at present, but we find the fact as follows.

Multiprocessor systems to use for the general purpose use have relatively few processors in the system. As well known, they have a tightly coupled architecture and the collision of the access to their storages or internal busses may limit the progress of the performance expected by the increase of the number of processors. The augmentation of their operating system is another problem. Though we can obtain a few systems having many more processors recently[2], the responsibility for the performance of the system rests with the user. After all, a tightly coupled architecture can not satisfy our requirements for the performance.

Another type of the multiprocessor system is a loosely coupled one, and it can have so many processors, for example 128 processors in a system[3]. However, their application is limited in some special areas, for instance, as the treatment of graphic data, calculation of the differential equation, etc.. To realize our computer, we must study and understand the reason why the loosely coupled architecture cannot be applied to general purpose uses. Our computer must have the high performance in not the SIMD (single instruction multi-data) mode but the MIMD (multi-instructions multi-data)mode.

There will be essential questions as follows.

(1) How many processors can be connected in a system?

(2) How ought to be the information path between processors?

(3) How ought to be the algorithm and the practical size of the partition of tasks into processors?

(4) How ought to be the logical communication, including synchronising between processors?

(5) How ought to be the basic configuration of the operating system?

(6) How ought to be the characteristic of the system implementation language?

(7) How ought to be the physical configuration of each processor?

4. BASIC CONFIGURATION

We assume the number of processors in a system as up to 16. The number is relatively small, but it will be sufficient for the purpose. For instance, we suppose the throughput of each processor as 2 MIPS, so the total throughput will excel 20 MIPS. (A few number of processors will engaged in not the problem solving use but the special function use in the system).

Then we use busses as the information path between processors. The reason of the selection is the high throughput of the communication and the easier expandability of the system. A feature of our system is the use of plural busses, and yet, they are allotted for the type of the information.

One message bus communicates short messages (for example, 64 bytes length, and used for the control of the system) in a short response time, and one or more memory busses communicate long data blocks (for example, 1 Kbytes length) with the high transfer efficiency. This configuration of busses may be effective to have the high performance on the communication[4]. The information are transmitted in a form of bit parallel and word serial, and the transfer rate is set up to 20 Mbytes per second. The value is confirmed to be enough

for the intra system communication by a preliminary simulation. Fig.1 shows the system configuration of our system. A bus-arbitor is prepared to have the quick response in the use of the message bus. And the message bus has a broadcast function to use for the scheduling and the search of the specific object. The control bus in Fig.1 has a role of the exclusive control between processors and the transmission of the clock signal.

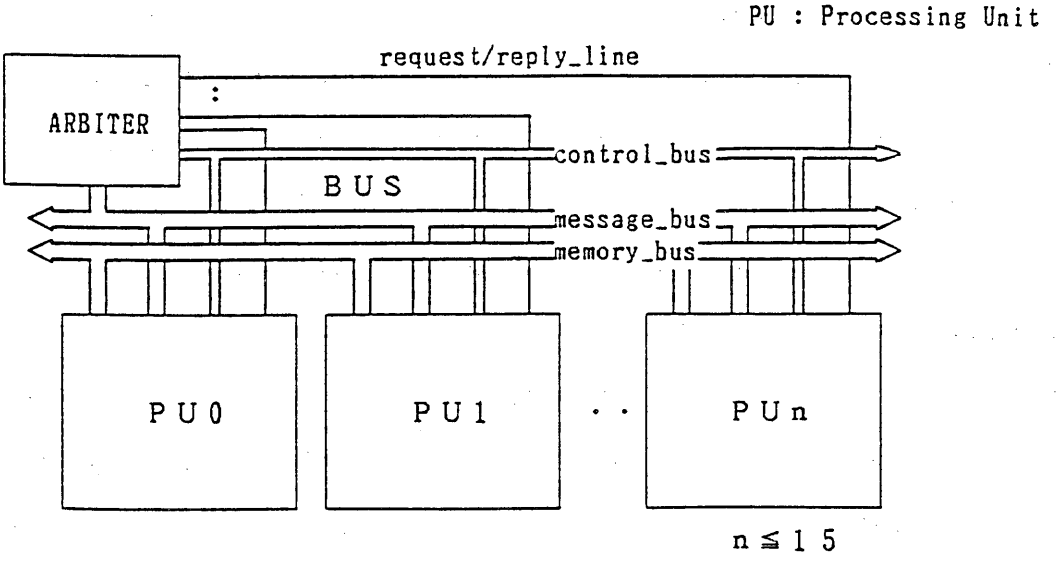

Fig.1 System Configuration of ADMS.

In generally, the most important and difficult problem is the partition of a process into processors in case of multiprocessor systems. However, we assume that our system will have so many independent processes and each process has not so large program size, when the total volume of the task is very large. This hypothesis will lead us to set up the following conclusion. There is no inevitability to divide a process into plural processors. Each processor will have plural processes at a time in generally. Problems are; how ought to allocate many processes into plural processors, what

is the most efficient method to communicate between processes each other, and what is the most efficient processing way in a processor. On the first problem, we consider that a specific processor may have a specially tuned feature and a specific process will be allocated into the said processor by the operating system. Namely, our system has a functionally distributed feature in company with a load-balanced distributed feature. The third problem is quite same to that of single processor system.

5. INTRODUCTION OF THE HIGH-LEVEL ARCHITECTURE

In a viewpoint of the loosely coupled configuration, we introduce a so-called high-level architecture in our system. The concept of the architecture is proposed by G.J.Myers[5] et. al., and contains the object oriented architecture, the capability-based addressing, and the tagged data.

In our system, all distinguishable resources in the system are recognized as objects. Each of them is an abstraction and identified with an UID (unique identifier), a type, and an expression. The type is a operational procedure permitted to operate on the object, and defines the character of the object. The object is an unit for accessing, and it is accessed using a capability which is a sort of pointer to the object. This logical configuration lead us to conclude that the physical multiprocessor environment is transparent for the user, because the subject can access to any object using a capability in a same procedure when the target object exists on a same processor with the subject or on a different processor. And, the system has the higher security by the

reference and the check of access grant using the type in case of the access. The fact that the relationship between a subject and a called object is relatively loose, gives the decrease of amount of the communication. That is, the architecture is suitable for the multiprocessor environment, because of the simple access procedure and less amount of the communication.

The procedure of the operation on an object is shown in Fig.2. Here, a subject send a message to a type manager which exist on the same processor with the called object,

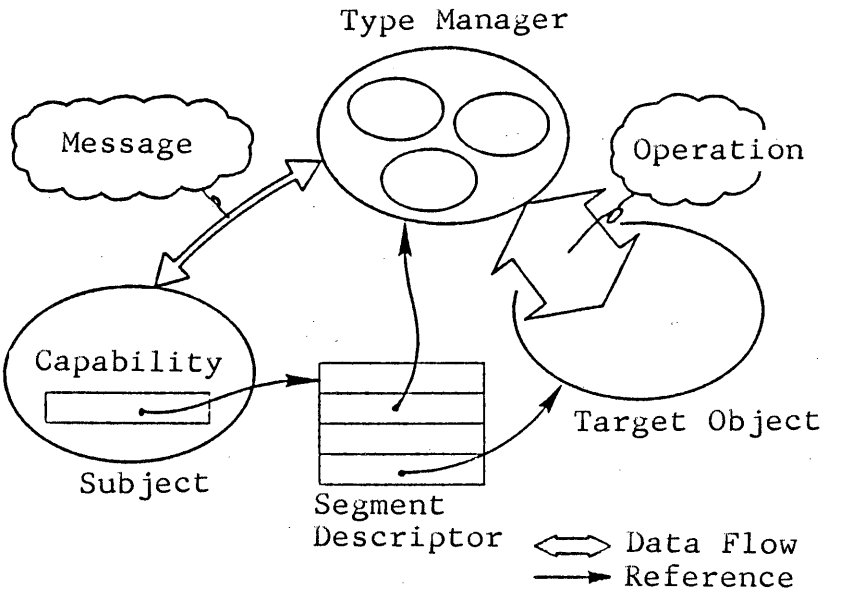

Fig.2 Basic Access Mechanism of the Object.

and asks to it to operate the object. The subject does not operate the object in direct manner, but call the procedure. Here, the type manager is a collection of operations, and it is an identity which identifies the operation on various objects.

The capability is a logical unique address and an access grant for an object. And, the possession of a capability means an addressability. The fact that the capability is an

address and that is not depend on the physical location of the called object is advantageous for the multiprocessor environment. Primarily, the concept of the capability is introduced for the protection of resources, and the function is kept in our system.

We don't adopt the tagged data, because of the unfitness to some languages and the complexity of the management of tags. The introduction of the high-level architecture gives many conveniences on the loosely coupled multiprocessor system as mentioned above. In addition, the so-called semantic gap is reduced in its width by the introduction of the architecture, and so, the good interface is prepared for the user.

6. LOGICAL CONFIGURATION AND OPERATING SYSTEM

The systems architecture of ADMS is shown in Fig.3.

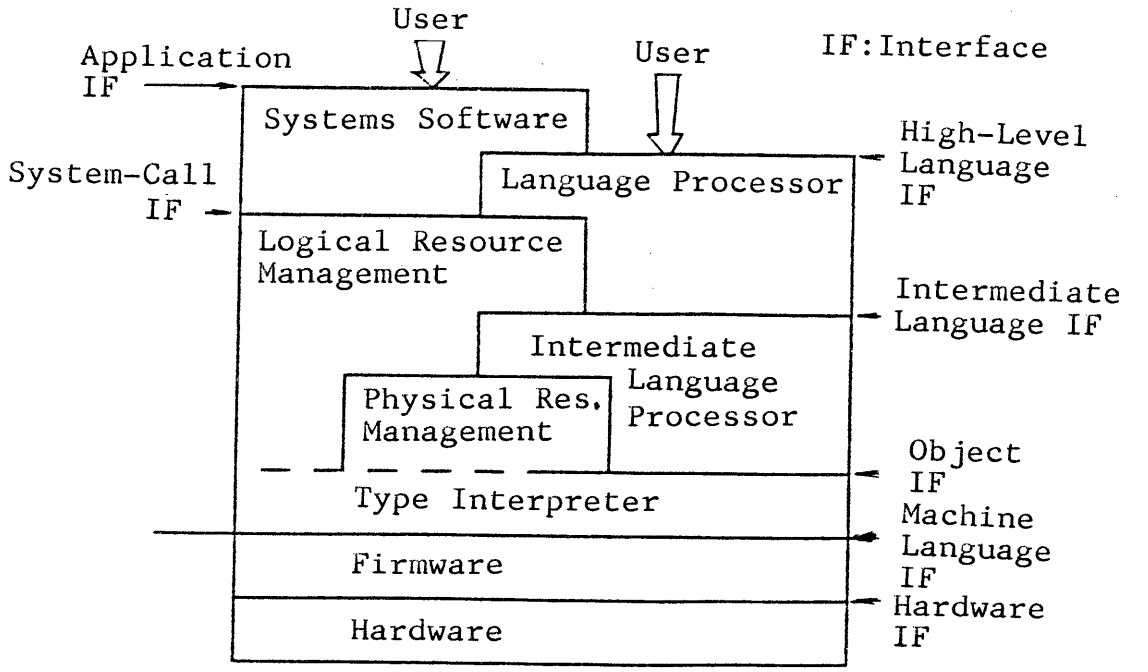

Fig.3 Systems Architecture of ADMS.

It seems to be relatively complexed, but the reader will be able to understand its inevitability. The portability of the software, the use of VLSI as the hardware, and the efficient interpretation of the object oriented architecture are reflected in the systems architecture.

The accurate address coversion mechanism is shown in Fig.4, which is shown conceptually in Fig.2. The explanation

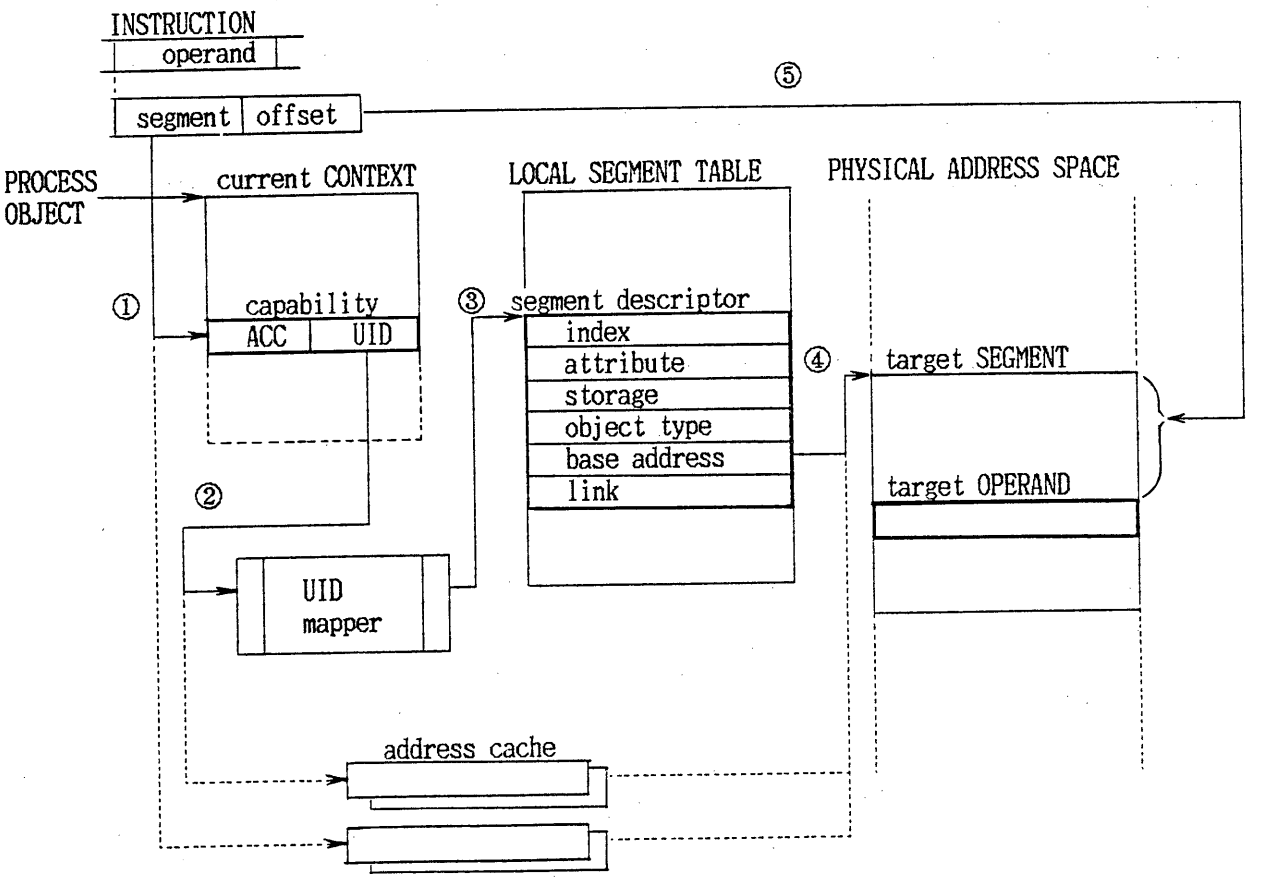

Fig.4 Address Conversion Mechanism.

of the figure will be mentioned in the following paper, and by the mechanism, the capability is converted to the physical address. A hash mechanism is applied as the UID mapper, and the use of the mechanism is effective to search objects and

to avoid the collision of the UID.

Features of the operating system (OS) are as follows.

(1) It has a moduled construction to match with the object oriented configuration and to have high executive efficiency by the concurrent operation on plural processors.

(2) The size of each module is comparatively large to avoid the increase of communications. Concretely, each module have the function of a type manager.

(3) Each module is executed as a process, and the inter-module communication is quite same to the inter-process communication.

(4) The communication between user and OS is the inter-process communication, too.

(5) Especially, it must have the high efficiency in the communication between processors.

The executive environment of programs on ADMS is shown in Fig.5. The context and domain object have a function to link objects with the capability, and by linking we can set up the executive environment. The link can be effective over processors.

For the inter-process communication, we prepare "message" and "port", and they have a feature as a sort of object. The port object is a communication media, and in case of inter-process communication, the sending process and the receiving process send out message objects equally to the port object. By the mechanism, we can prepare various types of communication.

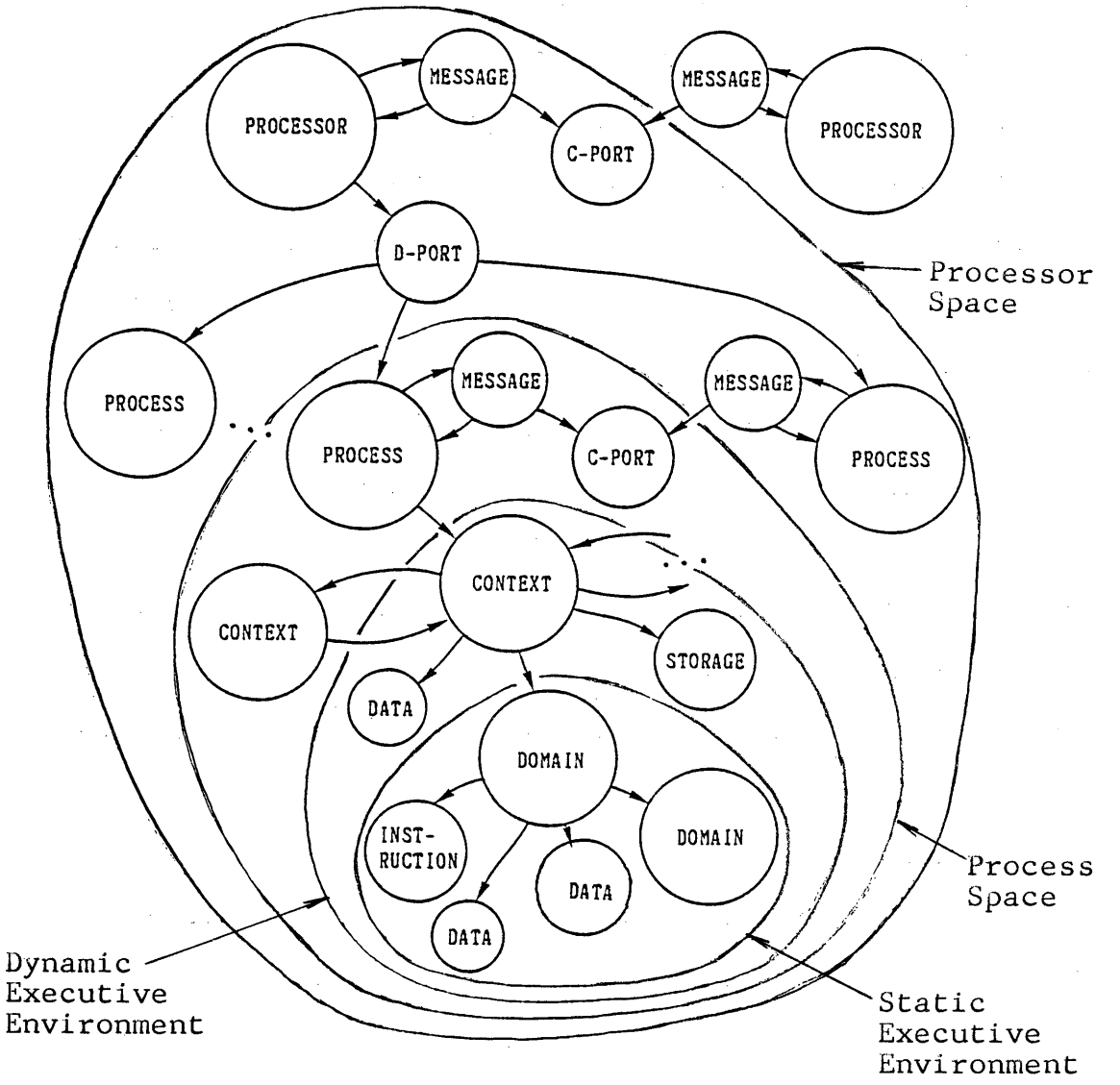

Fig.5 Executive Environment on ADMS.

7. HARDWARE

The inner configuration of the processor is shown in Fig.6.

Here, DPU (data processing unit) is the main processing device and has a role of system control, storage management and protection, input and output management, compiling of the source program, etc.. DPUs in a system have an equal architecture, and so, ADMS is a homogeneous system in the system

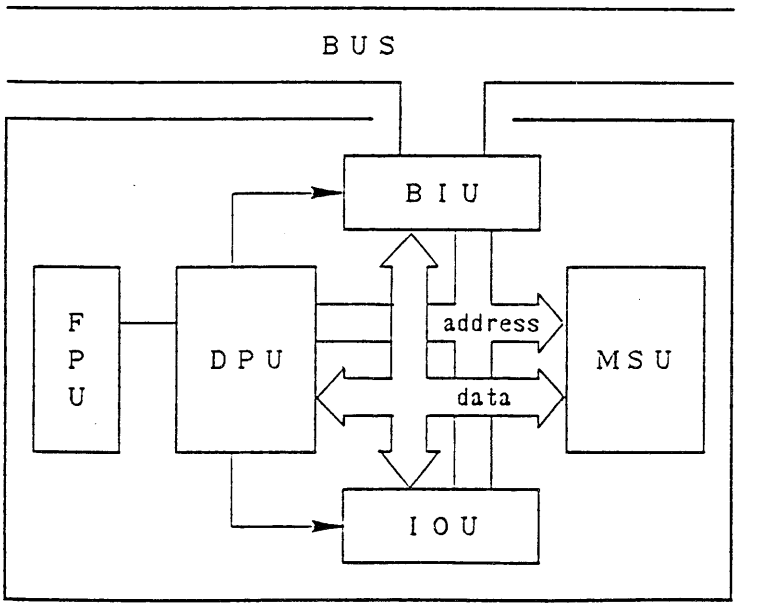

Fig.6 Inner Cinfiguration of a Processor.

level.

FPU (functional processing unit) has a role to execute the object program in high efficiency, and the architecture can be tuned for the role of the processor. For instance, the front-end processor and back-end processor will have special inner architecture. Thus, ADMS is a heterogeneous system in the application level.

The use of two processors means tuning up of processors just mentioned above and the realization of the high performance by the parallel operation with two processors.

BIU is the bus interface unit and shapes an interface for busses. It has buffer storages to have the high throughput on the communication between processors.

IOU is the I/O interface unit and shapes an interface concerning with input and output between DPU and peripherals.

MSU is the main storage unit prepared to each processor.

DPU and FPU have an inner configuration of 32 bit word

architecture, and it will be realized with VLSIs and the micro-code technology. Details of the hardware will be mentioned in the following paper.

8. LANGUAGES

The system implementation language of ADMS (SIL-ADMS) is based on concurrent Pascal and expanded in its specification to apply to the multi-user environment. In addition, some concepts are imported from Ada. We prepare objects on which the system can recognize, as follows.

* System management objects (processor, process, storage).
* Execution management object (instruction, data, domain, context).
* Inter-process communication object (message, port).
* Access management object (type, template).

An outline of the function is:

* All sorts of object ought to be declared with an identification name.
* The process is an object which can operate concurrently, and it is defined and declared explicitly in SIL-ADMS.
* The non-determinacy is designated by a revised select statement.
* The concurrency is designated explicitly by a cobegin statement.
* By the inter-process communication, all entries are declared on its name and type, and they must be linked each other.

The intermediate language of ADMS is based on the Actor model [6][7], and implemented using expanded Pascal-P. How-

ever, we use it under following conditions.

* We use the Actor as its attribute is a sort of data.
* We put an acquaintance on a sort of the message which can be sent to an Actor.
* By the communication, we predicate the existence of the reply.
* We don't allow the dynamic generation and destroy of the Actor.

Several instructions are added for the communication and logical operation, and instructions concerning with the file are struck off.

9. CONCLUSION

In our study to develop a computer which is most fittable for the distributed processing system, we have arrived to an idea to apply a loosely coupled multiprocessor configuration. In addition, it seems the introduction of a so-called high-level architecture have many advantages for the computer.

We have decided a basic design and rough specifications of the computer system as mentioned in previous chapters, and added some reasons on the decision. The concrete and detailed design is a future problem, but we are confident of our plan.

ACKNOWLEDGEMENT

We thank to Mr. Katsumi GOTO, Mr. Masamitsu BABA, Mr. Syuichi SUZUKI, Mr. Atsushi FUJIOKA, and Mr. Norio OHASHI, they were good colleagues of our investigation.

References

1. S.Amada, M.Tsuchida, Y.Sato, et al. : The System Architecture of a Computer to use for the Distributed Processing System, IPS Japan, SG Computer Archi., 48-3, pp.1-10,(1983), (Japanese).
2. Shiva Product Line Technical Description, Shiva Multisystems Corp.,(1985).
3. T.Hoshino, Y.Oyanagi, et al. : Monte Carlo Simulation of a Spin Model on the Parallel Computer, Comp. Physics Commun., Vol.34, No.1/2, pp.31-38, (1984).
4. G.Ricart : An Optimal Algorithm for Mutual Exclusion in Computer Networks, Comm. ACM, 24,1 pp.9-17, (Jan. 1981).
5. G.J.Myers : Advances in Computer Architecture, John Wiley & Sons. Inc. (1978).
6. C.Hewitt : Viewing Control Structures as Pattern of Passing Messages, J. of Artificial Intelligence, Vol.8, pp.323-364, (1977).
7. A.Yonezawa : A Tutorial on ACTOR Theory, J. IPS Japan, Vol. 20, No.7, pp.580-589, (1979), (Japanese).

INSTITUTE OF INFORMATION SCIENCES AND ELECTRONICS
UNIVERSITY OF TSUKUBA
SAKURA-MURA, NIIHARI-GUN, IBARAKI 305 JAPAN

REPORT DOCUMENTATION PAGE	REPORT NUMBER ISE-TR-86-56

TITLE

A Loosely Coupled Multiprocessor System : ADMS
---Basic Design---

AUTHOR(S)

Sanae Amada
Masashi Tsuchida
Yutaka Sato

REPORT DATE	NUMBER OF PAGES
June 6, 1986	16
MAIN CATEGORY Multiprocessor System	CR CATEGORIES C.1.2, C.1.3, D.1.3 D.3.2, D.4.1

KEY WORDS
Distributed Processing, Multiprocessor, High-Level Architecture, Operating System, System Implementation Language.

ABSTRACT

In the recent status of increasing of the distributed processing system, the computer architecture must be considered more seriously for the improvement of the system throughput. We have an idea to apply a loosely coupled multiprocessor configuration as a computer for the system. In this paper, we describe on the basic design of above mentioned configuration containing the introduction of a high-level architecture.

SUPPLEMENTARY NOTES