A Computer Communication Control Technique for Virtualization of Network Memory Resources and Its Implementation

KOICHI GOTO*, KOSAKU INAGAKI*, MASARU TSUKAMOTO*, MASATOSHI YOSHIKAWA*, SHUZO YAJIMA* and KAZUO IWAMA**

This paper presents the attempt we have made in linking two minicomputers, the PDP 11/40 and MELCOM 70/35. The main purpose of the link is to enhance the power of the PDP 11/40 as a stand-alone machine by virtue of the utilization of memory resources of the MELCOM 70/35. The technique we employed was to virtualize memory resources of both minicomputers and to reconstruct new memory hierarchy on the link of them. We considered any kind of hardware level information transfers in a system to be "communication" and developed a processor to control such communication as extensively and uniformly as possible. This processor is called the communication control unit (CCU) in this sense. The CCU constructs virtualized memory resources on a network of heterogeneous computers, creating a new physical view of extended memory resources. The CCU is composed of microprogrammed bit-slice microprocessors with cycle time of 200 ns and high speed memory units. It is equipped with cache memory to increase the throughput of the PDP 11/40 as well as to control various communications. This technique has two potential impacts. One is concerning the software transplantation. The other is concerning the personal computing network in the future.

1. Introduction

Much progress has been made in computer technology in the past decades, and with the recent advance of LSI technology distributed processing has become a very important concept in this area. Computer networks occupy a principal position and a large number of investigations have been done by many researchers [1].

In our laboratory, too, a local computer network LABOLINK [2] was developed and has been used to support research and education activities of the staff. It is characterized by the star network which is linked by optical fiber cables. The research objective was to investigate an easy yet versatile way of accessing several different computers from a single minicomputer. LABOLINK has been extended, and now new LABOLINK II is under development [3, 4].

This paper presents the attempt we have made in linking the two minicomputers, PDP 11/40 and MELCOM 70/35. These computers have their own architectures and characteristics, respectively. In short, we wished to fully use the rich software accumulation of the PDP 11/40 and fully utilization of large file devices of the MELCOM 70/35. We do not wished to rewrite the large amount of software of the PDP 11/40 including DOS, RT11 and UNIX [5].

The technique we employed was to virtualize the memory resources of both minicomputers and to reconstruct new memory space on the link of them. We thought as follows. Any kinds of transfers of information in a system can be considered to be "communication". Message or file transfer among host computers in a computer network is, of course, communication. Data transfers can also be regarded as communication or data transfer between CPU, registers, main memory and peripheral devices. We decided to develop a processor to control such communication in some virtual mode as extensively and uniformly as possible. This processor is called communication control unit (CCU) in the meaning of "communication" in this paper. The CCU is composed of microprogrammed bit-slice microprocessors with cycle time of 200 ns and high speed memory units. It is equipped with a cache memory of 4 K bytes to increase the throughput of the PDP 11/40 as well as to control the communication between its CPU and the main memory.

This approach is different from that of conventional communication processors [6]. In conventional systems, intra- and inter-computer communications are clearly distinguished, and communication processors do not manage intra-computer communication. Here intra-computer communication means memory access, peripheral device access, etc. in a computer. Our CCU can also manage intra-computer communication, and it is possible for a computer to utilize network resources as if they were a port of its standard peripheral devices without modification of its system programs. It means that network resources can be virtually regarded as its peripherals as viewed by its system program. Therefore, if our CCU is employed, a computer system even with

*Department of Information Science, Faculty of Engineering, Kyoto University, Kyoto 606, Japan.
**Department of Computer Sciences, Faculty of Science, Kyoto Sangyo University, Kyoto 603, Japan.

few peripheral devices can utilize system programs which require many peripherals.

Such a type of resource sharing rather resembles the ones realized by some computer complexes, e.g., link of homogeneous computers like in [7, 8] and link of heterogeneous computers in [9]. However, the use of the devices of other computers are not transparent to the operating system in them. It is necessary to develop new system programs in such systems. The most notable feature of our approach exists in the transparency of the use of other computers' devices, which has been attained by the unified management of intra- and inter-computer communications by the CCU. Therefore the modification of system programs and user programs is not necessary in our system. A computer can behave as if it is a stand-alone computer. Communication facilities are managed and supplied by the microprogram in the CCU. Of course, it is also possible to communicate with other computers under the control of the system program. Such circuits can also be provided by the microprogram in the CCU.

In Chapter 2, basic concepts of our system are discussed. Chapter 3 is concerned with a local computer network LABOLINK II and the link between the PDP 11/40 and the MELCOM 70/35. In Chapter 4 the hardware organization of the CCU is described. Chapter 5 is concerned with microprogram of the CCU.

2. Basic Concepts

2.1 Concept of Virtuality

The term "virtual" has been used in various techniques for computer systems, such as virtual memory, virtual machine, virtual terminal, virtual network, etc. [10]. Virtualization is essentially a kind of mapping. In virtualization techniques logical resources are mapped to physical resources. There are many other techniques which do not use the term "virtual" explicitly but contain this mapping, e.g., high-level programming languages, cache memory. Methods to map logical resources to physical resources are various. For example, in virtual memory technology, the logical resource is large main memory, and the physical resources are small main memory and magnetic disks. The control mechanism of virtual memory executes mapping between them. In high-level language programming, logical data structures and operations for them are mapped to main memory addresses and sequences of machine instructions by a compiler.

Our virtualization technique aims at the realization of the mapping of hierarchical logical resources to hierarchical physical resources in a unified fashion. Hierarchy means that of memory devices, e.g., the memory hierarchy of cache, main memory and peripheral memory devices. Many virtualization techniques exemplified above are mapping only from one layer to another layer in hierarchical structures of computer systems. Our technique does the mapping from logical to physical resources in a hierarchical structure. Various hardware resources, such as main memory, peripheral devices, network resources, and such resources are reconstructed virtually on the network in a hierarchical structure. The concept of "communication" plays an important role in this technique. It is stated in the following sections.

Virtualization often causes a decrease in system throughput. In order not to decrease the system throughput, our system has employed a cache memory, which is at the highest level in the hierarchy of memory resources in our system. The details will be stated in the following chapters.

2.2 Communication in a Computer System

Communication, in its original meaning, is the transfer of information between distinct places. A lot of communications exist in a computer system. There is a hierarchy of different memory resources in the access time and capacity, starting with the fast, but limited in size, registers in the CPU, via the cache memory and the main memory, till the large, but slow, secondary memory resources such as magnetic disks or tapes. Data transfers between memory resources in a hierarchical structure stated above can all be uniformly considered as communication.

2.3 Communication Control Technique

We propose a new communication control technique for resource sharing in the meaning of "communication" explained above.

We introduce a new communication control unit (CCU) into the computer system and all the memory resources are connected with the CPU via the CCU [Fig. 1]. The role of the CCU is to make up a new hierarchy of memory resources virtually on the really existing memory hierarchy by controlling various "communications" uniformly including communication with other computers. That virtual hierarchy is shown in Fig. 2. For the CPU, it seems that the CCU does not exist. However the CPU can get more powerful memory resources by virtue of the CCU. First, the CCU has a cache memory, so access time of the main memory is improved. Second, the CCU offers memory resources of other computers by virtualizing them at hardware level,

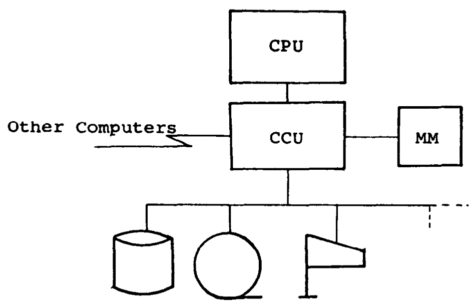


Fig. 1 Introduction of the CCU.

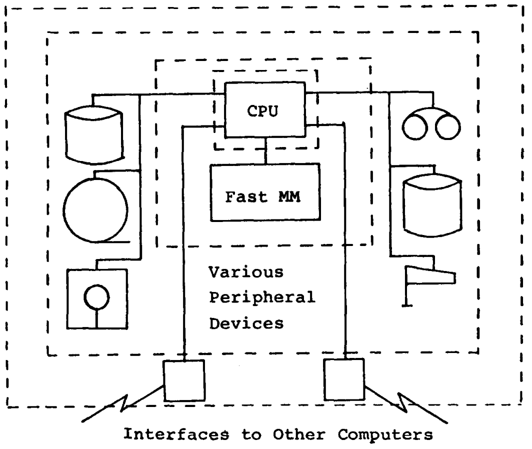


Fig. 2 Virtualized System.

so the CPU can use them without changing the software system. Those devices are called virtual devices in this paper. The devices of its own can be used as usual. These are called actual devices. Moreover it is possible to communicate with computers in the computer network.

This technique to construct a virtual memory hierarchy has a constraint in that the access time of virtual devices is generally more than that of an actual device. However, this constraint is not usually a crucial one, if the performance is not an essential problem.

The main reason why we have taken this approach is as follows. It is usually very difficult to even slightly modify a huge software system. It is almost impossible when its source list is not available. Our minicomputer PDP 11/40 has now three operating systems, DOS, RT-11 and UNIX. Rewriting such operating systems or future come ones is an excessively cumbersome tasks. This is why we employed a hardware approach to realize our virtualized system.

Moreover our attempt includes the implementatoin of cache memory in the virtual resource control mechanism. Many of the techniques for virtualization inevitably cause the decrease of the throughput of the system. We have made it possible to increase the throughput of the system by the introduction of a cache memory. The details are described in the following chapters.

2.4 Potential Impacts

Our attempts also finds its advantage in other situations. One is the transplantation of software.

It is very difficult to transplant a software system to other types of computers. Rewriting requires enormous time. Emulation generally causes a decrease of throughput and is often impossible.

Our approach can become effective in such transplantation of a software system. If the user has his own minicomputer system, then purchase only a CPU for that software and apply our virtualization technique. The software system can operate by utilizing that CPU and virtualized resources in effective manner. The illustration is shown in Fig. 3.

The other is the personal computing network [11] of the future. In the future, the TSS may turn over its position to the personal computing network. Although processors may become inexpensive, peripheral devices will still be price high. In such a situation each computer, probably heterogeneous, has to utilize some common virtual resources via a large-capacity high-speed network, e.g., an optical fiber network. Our technique has the possibility of enabling such a personal computing network. One reason is that the computers linked by our technique are mainly used as stand-alone machines. They can communicate between one another as network host computers, if necessary. So this technique might be used as a desirable form of a personal computing network

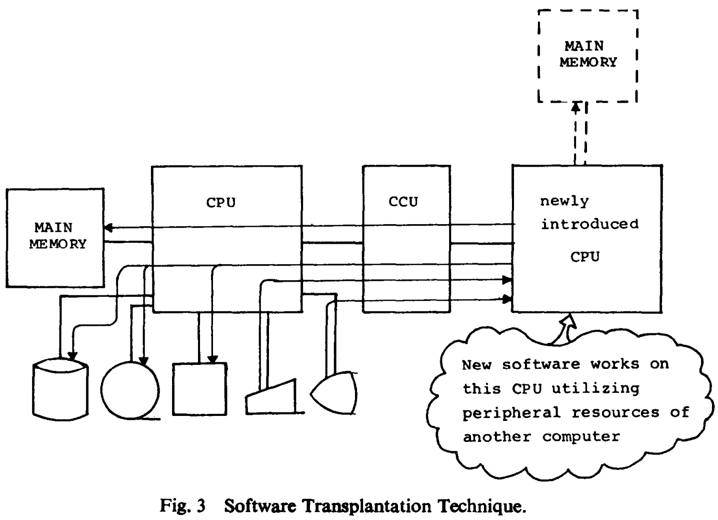


Fig. 3 Software Transplantation Technique.

that various heterogeneous computers work as stand-alone machines, though they utilize resources of others or some common resource bank.

3. Implementation of the Technique

The communication control technique described above is applied to the link between the PDP 11/40 and the MELCOM 70/35 in LABOLINK II.

3.1 LABOLINK II

LABOLINK II is a local computer network being developed in our laboratory for research and education. LABOLINK II is a successor of a computer network LABOLINK.

Computers in the LABOLINK II network are the minicomputers in our laboratory, the PDP 11/40 and the MELCOM 70/35, the large-scale FACOM M-200 in the Data Processing Center of Kyoto University, the medium-scale HITAC 8350 in the Department of Information Science and a microprocessor for terminal switching [Fig. 4]. The PDP 11/40 is connected with the HITAC 8350 via 1M bps optical fiber cables and the MELCOM 70/35 is connected with the FACOM M-200 via 2,400 bps TSS lines.

3.2 PDP 11/40 and MELCOM 70/35

(1) PDP 11/40

The PDP 11/40 is a 16-bit-word general purpose minicomputer. Fig. 5 shows the system configuration of the PDP 11/40. The PDP 11/40 has a special input/output bus called UNIBUS, whose maximum data transfer rate is 2.5M words/sec. All the devices of the PDP 11/40 system are connected with the UNIBUS. A device can become bus master to make a non-processor transfer of data directly to or from the main memory, or to interrupt program execution and force the processor to branch to a specific address.

(2) MELCOM 70/35

The MELCOM 70/35 is also a 16-bit-word general purpose minicomputer. Fig. 6 shows the system configuration of the MELCOM 70/35 system. The MELCOM 70/35 has four I/O channels, the program control channel (PCCH), the Direct Memory Access channel (DMA), the selector channel (SEL) and the multiplexer channel (MPX). All the peripheral devices are connected with the PCCH through which the CPU sends control data and gets status of devices. Though low-speed devices are connected with only the PCCH, high-speed devices are connected with one of the other channels for data transfers with the main memory.

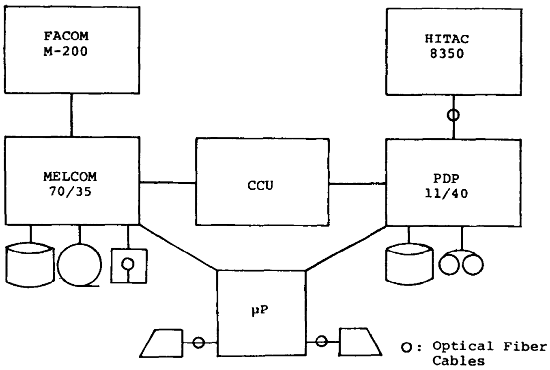


Fig. 4 LABOLINK II.

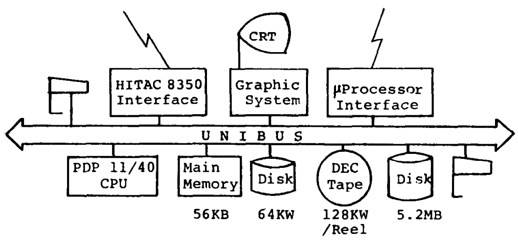


Fig. 5 System Configuration of the PDP 11/40.

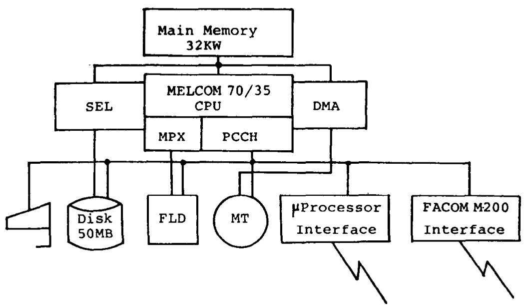


Fig. 6 System Configuration of the MELCOM 70/35.

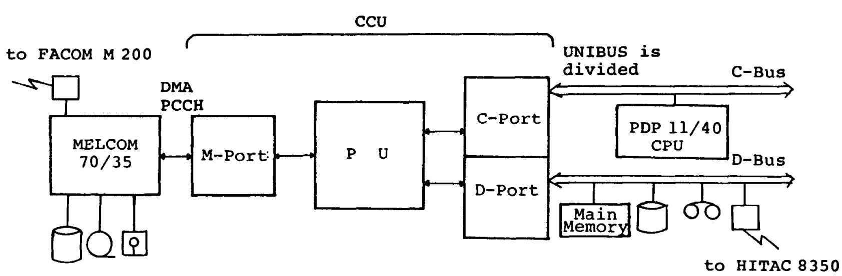


Fig. 7 Overview of the Link.

3.3 Virtualized System

Fig. 7 shows the overview of the link between the PDP 11/40 and the MELCOM 70/35 via the Communication Control Unit (CCU). Detailed description of the CCU will be given in Chapter 4. The CCU is connected with the PDP 11/40 via the UNIBUS and the MELCOM 70/35 via the PCCH and DMA, but to the PDP 11/40 the CCU is more than a mere device on the UNIBUS. The CCU divides the UNIBUS into two different buses, the CPU bus (C-BUS) and the device bus (D-BUS). There is only the CPU on the C-BUS and other devices are all on the D-BUS, but from the view of the CPU, all devices including those of the MELCOM 70/35 seem to be on its own bus. The CCU makes up new resource circumstances for the PDP 11/40, which are shown in Fig. 8 and described below.

(1) Several peripheral devices and interfaces which are on the UNIBUS from the first can be used the same as before.

(2) The main memory whose access time is faster than before by virtue of the employment of the cache memory in the CCU. As the PDP 11/40 is controlled asynchronously, improvement of the access time of the main memory causes improvement of the cycle time of the CPU.

(3) Several new peripheral devices are added on the bus virtually. They are a magnetic tape, flexible disks and a 50M-byte disk, which are virtualized by the CCU according to the specifications for the PDP 11/40. It is possible, for example, to virtualize the 50M-byte disk for the MELCOM 70/35 to two 20M-byte disks for the PDP 11/40.

(4) Facility for message and file transfers as a host computer in the LABOLINK II are also implemented in the CCU.

Moreover it is also possible, in principle, for the MELCOM 70/35 to use memory resources of the PDP 11/40 virtually, although it is not attempted in present experiment. In order to support the accesses from the PDP 11/40 to the MELCOM 70/35, we had to develop a user level program to manage the accesses from the PDP 11/40 at the MELCOM side. However, in the case of the accesses from the MELCOM 70/35 to the PDP 11/40, no user or system level program is necessary. All the accesses from the MELCOM 70/35 are controllable by the microprogram in the CCU in our configuration. The cache memory in the CCU is not available from the MELCOM 70/35.

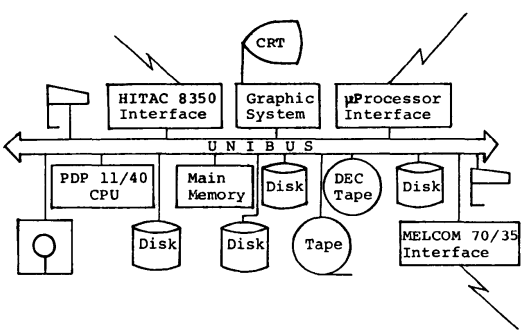


Fig. 8 Virtualized System of the PDP 11/40.

4. Communication Control Unit

4.1 Design Features

To form the system described in preceding chapters, we consider that the CCU should satisfy the following requirements.

(1) The CCU must control various communications in the system, such as access to the main memory, data transfers among various peripheral devices and the CPU or the Main memory, data transfers with the MELCOM 70/35 for virtualization, message exchanges on LABOLINK II and management of the cache memory. In a sense, the CCU can be regarded as a sort of special purpose computer.

(2) If the CCU malfunctions, the whole system could be damaged. So the CCU should have sufficient reliability.

(3) If any failure happens, it must be easy to detect it. Therefore it should be easy and efficient to test the CCU.

(4) The CCU should operate rapidly to manage the system and be able to transfer large amounts of data at high speed.

(5) Input/output control mechanisms of various computers differ very much from one another. We considered that it is important that the CCU has generality being able to be used in other applications, especially in an application executing a high-speed data transfer using optical fiber cables.

To satisfy the requirements described above, we have introduced microprogram control mechanism into the CCU and employed bit-slice bipolar microprocessors. Because the control mechanism is simpler than using hard wired circuits, reliability can be regarded as higher. And it is easy and effective to test the CCU by its own microprogram. The use of bipolar bit-slice microprocessors gives the CCU sufficient and high-speed operational ability.

We have divided the CCU into two parts. The first one is the processing unit (PU) which has overall control of the CCU and whose operation is performed sequentially by microprogram. The second part is called the input/output port (I/O port). By changing microprograms and I/O ports, the CCU can be applied to various purposes.

4.2 Hardware Organization

The block diagram of the CCU in this system is shown in Fig. 9. Modules surrounded by a broken line are the processing unit (PU) and others are the I/O ports. The

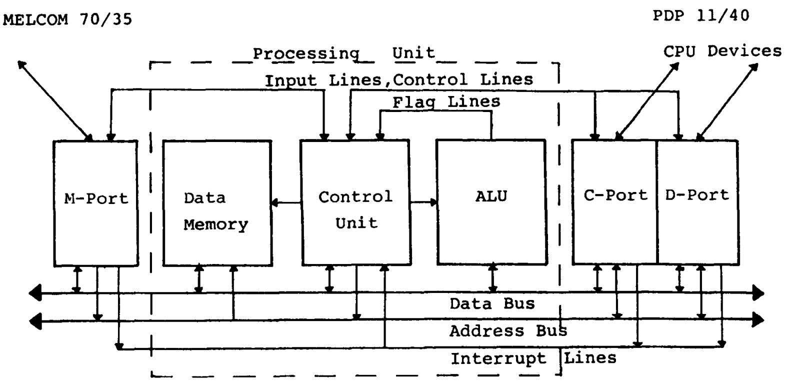


Fig. 9 Organization of the CCU.

CCU is implemented using about 500 IC and LSI packages.

(1) Processing Unit

The PU consists of three modules, the control unit (CU), the arithmetic logic unit (ALU) and the data memory (DM). A 16-bit data bus and a 12-bit address bus connect the modules above. The CU receives signals through flag lines from the ALU and input lines from the I/O ports, and it also controls other units, such as the I/O ports, the ALU and the DM, by microinstructions. The overall system is controlled by a single clock. Execution time of each microinstruction is divided into 4, 5 or 6 phases. The length of one phase is 50 ns, so execution time is 200 ns, 250 ns or 300 ns.

Currently the CCU has 1K words of RAM for the control memory (CM) in the CU. The CM can be extended up to 4K words if necessary. The devices used are Intel 2125 1024×1 bit static MOS RAM's (access time is 30 ns typically and 45 ns maximum). 64-bit width word microinstructions are used for the control. The DM acts as the following types of registers or memories; control registers of virtual memory resources for the PDP 11/40, device registers for virtual resources and temporary buffers for data transfers.

(2) Input/Output Port

Main role of the I/O ports is high-speed signal interfacing with computers. In addition to that, a special function module which is necessary in some application of the CCU is placed in the I/O port. In our case, the cache memory is such a special function module. The number of the I/O ports is not restricted, but the total numbers of control lines between the CU and the I/O ports are restricted.

There are three I/O ports in our system, which are the M-port, the C-port and the D-port.

(a) M-Port

This I/O port is connected with the MELCOM 70/35 via the PCCH and the DMA. The program controlled data transfers are executed via the PCCH, and high-speed data transfers with the main memory are executed via the DMA.

(b) C-Port

This I/O port is connected with the CPU of the PDP 11/40 via the C-BUS. The only devices connected with this C-BUS are the CPU and the C-port. The C-port is provided with the cache memory which has been newly introduced for the PDP 11/40.

(c) D-Port

This I/O port is connected with the D-BUS which links the main memory and various peripheral devices. Communication between the CPU and devices on this D-BUS are done as if the UNIBUS were not divided by the CCU.

(3) Cache Memory

The cache memory in the C-port has been introduced in order to increase the system performance of the PDP 11/40. The devices used are Intel 2125AL static MOS RAM's. This cache memory is decomposed into 16 blocks and the size of each block is 256 bytes. The blocks of the cache memory are organized to realize the full associative method.

The cache memory is controlled by both a hard-wired circuit and microprograms of the PU. The hard-wired circuit performs the data transfers between the CPU and the main memory via the cache memory. Microprograms manage the replacement algorithm, algorithm so-called pseudo-least-recently-used method (PLRU method) which will be discussed in Chapter 5.

If an accessed datum exists in the cache memory, it is obtained immediately by the CPU. At the same time the PU updates the cache control information when the PU has no other tasks. If an accessed datum does not exist in the cache memory, the CPU reads it from the main memory directly. If the replacement hardware is idle, the C-port interrupts the PU and then the PU decides which block will be replaced and informs the C-port. The C-port replaces the block by the hard-wired circuit. If the block to which the CPU wants to write exists in the cache memory, the datum is written not only into the cache memory but also into the main memory at the same time.

5. Control Microprogram

5.1 Organization of Microprogram

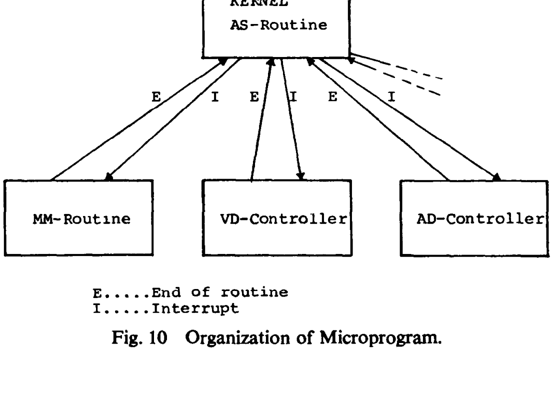
The organization of one of the microprograms is shown in Fig. 10. There are two types of microprograms, those of the kernel microprogram and other microprograms. The control is always in the kernel microprogram unless there are some other tasks that the CCU must do. The kernel microprogram is to manage the cache and control information by means of the PLRU algorithm. Other microprograms are started when interruptions from the I/O ports are accepted. There is a mismatch routine (MM-routine) which is started when a mismatch happens in the cache memory, actual device controller (AD-controller) and virtual device controller (VD-controller), which is started when the PDP 11/40 makes an access to an actual device or a virtual device.

The microprogram consists of about 300 microinstructions. It was developed by using a microprogram assembler developed by us [4]. We have various other microroutines for testing and maintaining the CCU and the system, which amount to about 500 steps.

5.2 Cache Memory Control

There are a couple of microprograms to control the cache memory. One is access scan routine (AS-routine) and the other is MM-routine. Their operations are closely related to the replacement algorithm.

We have employed a new replacement algorithm called pseudo-least-recently-used (PLRU) algorithm. This algorithm is a modified version of the least-recently-used (LRU) algorithm [12]. In the LRU algorithm, a block to be replaced is the block which has not been used for the longest time. In our algorithm, the term "pseudo" means that the microprogram to control replacement does not always run. Namely, the microprogram does not know all the access information. This microprogram, the AS-routine, is the kernel microprogram which runs any time there are no other demands to the CCU. The problem is how PLRU algorithm works effectively compared with LRU algorithm. By computer simulation,



E.....End of routine
I.....Interrupt

Fig. 10 Organization of Microprogram.

we have investigated the efficiency of PLRU algorithm. As a result, it has been verified that the hit ratio of PLRU is only 2 or 3 percent worse than that of LRU [13]. If the CPU makes an access to the main memory and the data exists in the cache memory, the AS-routine updates the priority information of the associated blocks of the cache memory.

The MM-routine is started when a mismatch in the cache memory occurs and the replacement hardware is idle. The MM-routine decides which block should be replaced according to the priority information prepared by the AS-routine. It makes the replacement by sending the memory address being accessed by the CPU and the block number to be replaced. The hardware replaces that block automatically. Replacement occurs from the address being accessed in sequential order. The amount of hardware has decreased by virtue of proper assignment of the cache management mechanism to hardware and firmware with almost no decrease in efficiency.

5.3 Virtual Device Control

In this system, the PDP 11/40 can make an access to peripheral devices of the MELCOM 70/35 as if they were its own. Such logically existing devices are virtual devices.

In the present system, the media or the areas accessed by the user programs of the MELCOM 70/35 and the PDP 11/40 are separated. The magnetic tapes or the flexible disks for the PDP 11/40 users must be different volumes from that for MELCOM 70/35 users. They are prevented from being accessed by MELCOM 70/35 users. If PDP or MELCOM users want to use the MELCOM devices, they will set their own volumes on the devices. As for the 50 MB disk of the MELCOM 70/35 system, a fixed area is allocated to PDP users, and is prevented from being accessed by MELCOM users. Though this way of realization may not be efficient, it is simple and the advantage of usability of various devices should be considerably large in small systems as ours. If the MELCOM (PDP) user wants to access a PDP (MELCOM) file, he utilizes the network access technique and obtains it via PDP to MELCOM (MELCOM to PDP) file transfer.

Virtual device controller (VD-controller) manages accesses of the PDP 11/40 to virtual devices. Peripheral devices on the UNIBUS have their own device registers. Such registers of virtual devices are virtually realized in the DM of the CCU. The VD-controller consists of two microprograms VDC-I and VDC-II. When the CPU or some device makes an access to one of those virtual registers, the I/O port detects it and interrupts the PU. The VDC-I is started. If a data transfer is requested by that access, the VDC-I interrupts the MELCOM 70/35 to start the support program and sends the contents of all the registers of the virtual device and ends.

The support program of the MELCOM 70/35 examines the data sent from the CCU, and translates various control parameters for the virtual devices to those for the peripheral devices of the MELCOM 70/35. If the data transfer is from the MELCOM 70/35 to the PDP 11/40, the program loads data from the device into the buffer area in the main memory of the MELCOM 70/35 and adds the control information. Then it starts VDC-II. VDC-II reads contents of the buffer and sends data to the PDP 11/40. When the data transfer cannot be performed, this operation is repeated. The process of data transfers is shown in Fig. 11.

If the data transfer is from the PDP 11/40 to the MELCOM 70/35, the support program stores only the control information into the buffer. VDC-II reads data from the main memory of the PDP 11/40 and sends them to the MELCOM 70/35. Then the MELCOM 70/35 stores them into the device. The access of the MELCOM 70/35 to a device of the PDP 11/40 as a virtual device is realized by supporting all the management operations by the firmware in the CCU. The CCU utilizes the DM as the buffer for this operation. The bus acquisition of the DMA device can compete with that of the replacement hardware. Since the transformation of error information is rather complex, presently only error/no error is reported from the MELCOM 70/35.

5.4 Actual Device Control

Actual devices mean original devices of the PDP 11/40. There are three kinds of jobs that the CCU performs to manage actual devices.

(1) Control of device register accesses from the CPU

When the CPU makes an access to one of the devices, the CCU detects it by decoding its access address and connects two buses and this access is performed.

(2) Control of non-processor requests from devices to perform direct memory access

This request is accepted without informing the CPU and DMA data transfers are executed only under the control of the CCU.

(3) Control of bus requests from devices to interrupt the CPU

After I/O operations the actual devices make bus requests to interrupt the CPU and inform their status. The CCU accepts such requests and gets status of devices, and then makes bus requests to the CPU as actual devices.

These three jobs are executed by the AD-controller.

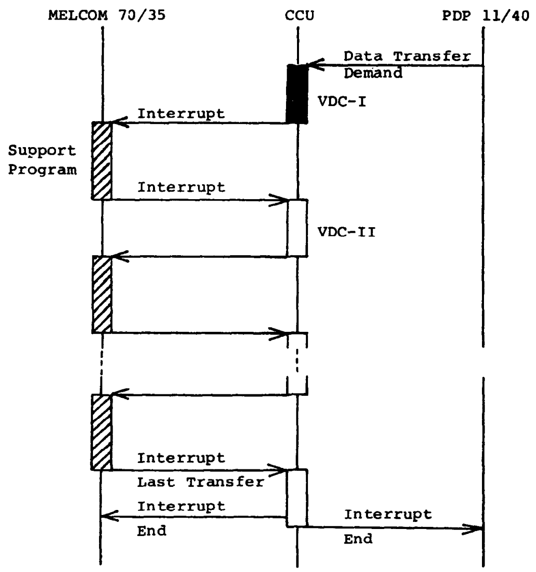


Fig. 11 Diagram of Data Transfer.

5.5 Network Communication Control

The PDP 11/40 and the MELCOM 70/35 are included in the local computer network LABOLINK II, so the CCU must support message and file transfers between the PDP 11/40 and the MELCOM 70/35. The CCU has those registers which act as the device registers for the communication between the PDP 11/40 and the MELCOM 70/35. The sender, one of the two computers, makes an access to those registers and sends control information requesting conventional data exchanges. Then the CCU understands the specified receiver and performs a data transfer according to the contents of the registers.

5.6 Test Microprogram

To increase the reliability of the CCU, we developed microprograms to test and maintain the hardware of the CCU and the system. By running test microprograms, we can easily know the existence of faults and their locations. Special microinstructions are prepared to test the CM itself, which automatically change the contents of the CM in a sequential test mode.

6. Conclusion

In pursuing this study, we paid attention to the concept of virtuality. It is because the link was the tight coupling of two heterogeneous minicomputers with quite different architectures. Some virtualization was necessary to make the best use of them. The main purpose of the link is to enhance the power of the PDP 11/40 as a stand-alone machine by virtue of the utilization of memory resources of the MELCOM 70/35. The technique we employed is to virtualize the memory resources of both minicomputers and to reconstruct new virtual memory space on their link.

Any kinds of transfers of information in a system, such as data transfers among a CPU, registers, main memory and peripheral devices as well as message and file transfers in computer networks, can be considered as "communication". We have developed a communication control unit (CCU) which controls such "communication" uniformly and reconstructs a new hierarchy of memory resources virtually on LABOLINK II. The devices of the MELCOM 70/35 are virtualized by the CCU as if they were the standard devices of the PDP 11/40. For the PDP 11/40, the CCU seems not to exist, however the PDP 11/40 obtains more devices than before and a faster main memory. In addition, message and file transfers on the LABOLINK II are also possible.

Although we have designed the CCU as universally as possible, the application of this CCU is still limited. It is because the uniform management of various data transfers can be easily realized only when the architecture of the computer is suited for such organization. The UNIBUS of the PDP 11/40 was suitable for our purpose, but the architecture of the MELCOM 70/35 was rather hard to deal with. Since many of mini- and micro-computer systems have unified buses similar to the UNIBUS, our technique would be applicable to many such systems, although sometimes restricted to small systems.

The virtualization of the main memory of the MELCOM 70/35 in the view from the PDP 11/40 is, in principle, possible, if the time-out mechanism in the CPU of the PDP 11/40 is removed. Since the memory bus of the MELCLM 70/35 is not open to the user, the access to the MELCOM's memory bus is possible only by software. Therefore it is now impossible to supply the hardware to support the cache control. If considerable decrease in the performance of the system is admitted, such virtualization is possible in our present system.

We think this technique might be utilized for software transplantation or personal computing networks in future, especially networks using optical fiber cables.

Acknowledgment

The authors wish to thank Mr. H. Fujita and Mr. S. Ishiguro for their contribution to this research. Thanks are also due to Associate Professor Y. Kambayashi, Research Associate H. Hiraishi and other members of our laboratory for their useful discussions. This work is supported in part by the science foundation grant of the Ministry of Education and Culture of Japan.

References

1. Thurber, K. J. and Freeman, H. A. A Bibliography of Local Computer Network Architectures. *ACM Computer Architecture News*, 7, 5, (Feb. 1979), 22–27.
2. Yajima, S., Kambayashi, Y., Yoshida, S. and Iwama, K. Labolink: An Optically Linked Laboratory Computer Network. *IEEE Computer*, **10**, 11, (Nov. 1977), 52–59.
3. Fujita, H. A Virtual Computer Connection Using Microprogrammed Intelligent Interface in LABOLINK. Master Thesis of Department of Information Science, Faculty of Engineering, Kyoto Univ., (1978).
4. Inagaki, K., Goto, K., Ishiguro, S., Yajima, S. and Iwama, K. A Microprogrammed Interface for Computer Connection and Its Design and Maintenance Aids. Technical Report on Information Processing, *IEEJ*, (Aug. 1979), (in Japanese).
5. Ritchie, D. M. and Thompson, K. The UNIX Time-Sharing System. *Communications of ACM*, **17**, 7, (July 1974), 365–375.
6. Hashimoto, A. and Yamashita, M. Communications Processor. *J. IECE Japan*, **62**, 11, (Nov. 1979), 1296–1303, (in Japanese).
7. Honda, N., Katsuyama, Y., Fujii, M. and Tokura, N. A Device-Sharing System for a PDP-11/20 dual system. *J. IPSJ*, **14**, 10, (Oct. 1973), 794–801, (in Japanese).
8. Matsuura, T., Sakemoto, T., Yano, S., Tokura, N., Fujii, M. and Okamoto, T. A Mini-Computer Complex and Its Operating System. Trans. *IPSJ*, **18**, 9, (Sep. 1979), 913–920, (in Japanese).
9. Kamibayashi, N., Takeyama, A., Nishigaki, H. and Aiso, H. The Distributed Bus Control Scheme and Intelligent Interface for a Mini-Computer Complex System, KOCOS. *Trans. IECE Japan*, J61-D, 11, (Nov. 1978), 842–849, (in Japanese).
10. Weegnaar, H. J. Virtuality and Other Things Like That. *Proc. of Compcon Fall*, (1978), 287–293.
11. Carnegie-Mellon Univ., Department of Computer Science. *Proposal for a Joint Effort in Personal Computing*. (Aug. 1979).
12. Gibson, D. H. Considerations in Block-Oriented Systems Design. *SJCC*, (1967).
13. Yoshikawa, M. Consideration on a Control Unit for Virtualization of Memory Resources in a Computer Network and the Implementation of Its Microprogram. Graduate Thesis of Department of Information Science, Faculty of Engineering, Kyoto Univ., (1980), (in Japanese).

(Received March 26, 1980: revised June 17, 1980)