PM/II—Multiprocessor oriented byte-sliced LSI processor modules

by MARIO TOKORO
Keio University
Yokohama, Japan
and
TAISUKE WATANABE, KATSURA KAWAKAMI, JUN SUGANO and KATSUHIKO NODA
Matsushita Research Institute Tokyo, Inc.
Kawasaki, Japan

ABSTRACT

This paper is concerned with the design and implementation of LSI processor modules named PM/II. The basic concepts involved in designing PM/II modules are (1) to provide maximum flexibilities with the smallest kinds of modules, (2) to construct a wide variety of computers from micro to midi and/or maxi where the total number of components are at a minimum, and (3) to realize PM/II in "the process free design" which avails itself of the idea that basic architecture is invariant even when implemented in any semiconductor process. PM/II are now actually being produced in CMOS.

At present the PM/II modules provide an Executor (EX), a Sequence Controller (SC), and an I/O Controller (IOC). They realize three dimensional extensibilities, i.e., (1) for functions through functional decomposition, (2) for varying bit-width of operation by byte-slicing, and (3) for multiprocessor configurations by using a sophisticated inter-processor communication function. A multiprocessor system composed of PM/II’s is already running, proving the validity of the design concept.

INTRODUCTION

Recently rapid progress in semiconductor technology has put high density and low power-delay LSI products to practical use. In the case of density, it has even been said that for semiconductor processes there exist no problems even when the demand for large amounts of gates per chip are taken into account by computer architects. While various kinds of microprocessors have been consumed in great quantity, to a large extent the LSI-zation of mini and office computers has prevailed. LSI hardware has the potential to change the architecture of middle and/or large scale computers in the near future.

Several approaches to the development of LSIs for middle and/or large scale computers are immediately apparent. One approach is to make a chip which contains the total functions of an individual CPU. Yet this has the disadvantages of lack of flexibility; (1) when design errors occur, (2) when design changes in technological development occur, (3) in system configurations and applications, (4) in testing chips, and (5) in the trade off between pins, yield and power dissipation. So far only 8 to 16 bit simple microprocessors have been realized due to these disadvantages.

If the principle of mass-production of a few LSIs is hereafter held to, it seems profitable for manufacturers to develop as few kinds of chips as possible from which a variety of computers ranging from micro to midi and/or maxi, can be composed. The straightforward approach to this problem seems to be the use of the concept of processor modules, which are sometimes called computing modules or computer modules. This approach attempts to develop minimum kinds of LSI processor modules with maximum capabilities in each, so that any digital system can be composed of the smallest number of them. Some research on processor modules has previously been accomplished, with the one realization being bit-sliced microprocessors such as MM16701, Intel13000, AMD2900, Macrologic, SBP0400, and M10800. The performance and the flexibility of bit-sliced microprocessors in applications has been proved. However, a number of MSI modules and/or SSI modules are indispensable when composing an applications system. This is caused by the following: (1) bit-sliced microprocessors have been developed to extend the capability of ALU’s which still require the design and manufacture of many support modules, (2) they do not get rid of the family series of MSI’s, and (3) they have been developed with excessive restrictions on the number of gates per chip and on the number of pins. Yet processor modules should be essentially designed through analyzing various computers and their applications and decomposing into functions for those applications, and then synthesizing the
functions into combinations of modules in order not only to gain maximum flexibility in system configuration, but also to be able to compose systems with a minimum number of components.

PM/II is a series of processor modules designed with the above considerations taken into account. The authors have classified the applicability, (i.e., the extensibilities of the capability of processor modules) into the following three criteria: (1) extensibility of functions, (2) extensibility of bit-width of operations, and (3) extensibility of total system performance. The first means the availability for the addition or change of functions such as a multiplier/divider to a system. In order to satisfy this requirement, PM/II is functionally decomposed. For the second requirement, the execution module of PM/II is designed byte-sliced. And for the last one, a powerful function available for multiprocessor configuration is attached, since multiprocessor systems have been becoming important in realizing high performance systems.

In addition, the architects have proposed “the processor-free design” against various restrictions from semiconductor technology. That is the methodology of design in which the basic structure is invariant in spite of the change of semiconductor processes, and in which maximum performance can be obtained with a single process. CMOS process has now been taken for the actual development. However, PM/II modules will have the same capabilities in other technologies. Moreover, they will give better performance in other technologies such as Shottky TTL, and/or in improved technologies expected in the future.

The three basic modules in PM/II have been designed and are being processed. They are a Byte-Sliced Executor (EX), a Sequence Controller (SC), and an I/O Controller (IOC). At the present, a multiprocessor system with three processors including breadboards of PM/II is running, satisfying the design purpose and proving the validity of the design concept. The rest of this paper is devoted to the description of the design concept, architecture, application, and evaluation of PM/II.

**DESIGN CONCEPT**

A CPU can be functionally decomposed into a number of functional blocks such as several data processing blocks, a sequence control block, an I/O interface block, a control memory, and a main memory. Such a method of decomposition, e.g., functional decomposition, decreases the number of connections among blocks, and this leads to a decrease of pin numbers of a chip when producing an LSI module. It is also preferable from the point of semiconductor technology to have the work load, processing speed, and chip size of blocks balanced among the blocks, unless the flexibility of each block is lost.

On designing processor modules, the most principally used blocks have been newly designed, e.g., ALU, sequence control block, and I/O interface block, while control and main memory have not needed to be redesigned.

The ALU may vary its operations and bit width as a function of its performance objectives. Bit-slicing is applied to the module with the powerful instruction set for composing the ALU, so that functional flexibility and semiconductor technological balance is satisfied.

A sequence control block calculates the next address of its control memory. The address space is generally limited to a fairly small size, and dynamic microprogramming techniques are feasibly employed for large microprograms. This results in a sequence control module with a powerful addressing calculation capability, which is not bit-sliced.

Data processing blocks are connected to buses. Since bus configuration and the control method varies according to the application, it is profitable to separate I/O control functions from data processing blocks to form an I/O control module. An I/O control module has been designed for an asynchronous bus. The module is composed of relatively few gates, so it is not difficult to redesign to meet other buses.

Based on the above considerations, designs of these three modules have been carried out from the following view points, which are proposed to construct a figure of merit of processor modules, e.g., (1) processing speed, (2) configurational flexibility, (3) performance/cost of control memory, and (4) availability for multiprocessor systems.

**Processing speed**

**Pipelining**

In order to obtain high speed processing capabilities, instruction fetching and the execution of modules are overlapped to form pipelining. This technique is very effective when the cycle time of the control memory and the execution time of modules are balanced. Therefore it agrees with the process free design method.

Conditional jump instructions take extra cycles in pipelining, which may cause inefficiencies in execution time and control memory utilization. To minimize these inefficiencies, a signal named “Wait” has been introduced which suspends the execution until a designated condition is satisfied. This signal is available for quick responses in process synchronization which are frequently encountered by I/O instructions. On the other hand conditional branching routines are essentially left conditional. Microinterruption is provided to recover from a dead lock.

Inside the module EX, a pipeline technique is also adopted for data transmissions into and out of the ALU. Three-address instructions for EX are employed for this reason.

**Two data ports**

The data transmission delay between LSI chips is several times longer than that inside a chip, and this may limit the rate of data throughput. In order to gain a high rate of data throughput, two independent parallel data ports are provided, which largely affect the bit width of operations in an EX.
**Configurational flexibility**

*Byte sliced*

The width of parallel operations in an EX is designed to be eight bits, or a byte, which is generally the smallest unit of data. While this decision has been made to balance with the number of pins and the size of the chip area of an SC, this has satisfied the requirement of extensibility for bit-width of operations and consequently has brought about configurational flexibility.

**Two data ports**

An EX, provided with its two independent data ports, increases configurational flexibility of the PM/II. For example, one port is assigned for a system bus and the other for the processor bus which connects local memory, multiplier/divider, and other devices. Another scheme is where one port operates asynchronously and the other synchronously.

**External modification**

SC is designed to have an 8-bit External Address Modifier (XAM) input port, which enables it to modify the next address of the control memory dynamically. By connecting a PLA decoder or such a device to XAM, quick response for parallel branching is easily attained which is significantly effective for real time control and language emulation.

Addresses of general registers in EX can be designated by data. This enables indirect register designation which is essential to emulation.

**Performance/cost of control memory**

*Module instructions*

Each module has its own instruction set independent from the others, which is partially encoded. The microinstruction format becomes wide when modules are combined to compose a system, so that the control memory cost becomes quite expensive. However, when constructing a low price processor, one is not required to prepare a full parallel control of resources in the system. In order to meet the requirement of parallel control with reasonable cost, a "Chip Enable" bit is attached to the instruction of each module. If a chip is not selected, a "No Operation" instruction is generated inside. This bit very easily and directly enables the overlap of fields among modules.

**Availability for multiprocessor system**

*Communication synchronization*

Usually, a "Test and Set" instruction for a flag is used to synchronize communications between processors. However, it is more efficient to provide a register with a flag, which is set simultaneously when a message is transferred to the register if and only if the flag is off. This facility is adopted and implemented in an EX. A P-Register (PR), which is an 8-bit register in an EX, receives information from other processors. The content of the PR is not affected until a flag named P-Register Full (PRF) in the IOC is reset by an instruction.

**Bus system**

The IOC is designed to interface the EX with an asynchronous bus system. The asynchronous bus system is very general and extensible in multiprocessor construction. In this bus system, processors, memories, and the other devices are uniquely addressed, and the communications between them are carried out by handshaking. A distinctive feature of this bus system is its special signal (RSYNC) which supports the communications between processors.

**SPECIFICATIONS**

PM/II is being implemented by the CMOS process. The delay of logic elements has been confirmed on test chips, showing for example that the delays of an inverter, a 2-input NAND/NOR, and an I/O pin are at most 5, 10, and 30 nsec, respectively. The specifications of each module are shown in Table I. The maximum number of the pins of the LSI package are 60, and more than 4500 transistors are included in SC and EX chips. Cycle times of 240 and 280 nsec are expected in 16 and 32-bit system operations, respectively.

**Executor**

The block diagram and instruction set of the EX are shown in Figure 1. Four types of instructions are designed to obtain smooth pipeline processing. The first type of instructions, which are the most typical ones, consists of four fields, ALU Operation (AO), A, B and C. The contents of General Registers or I/O Registers designated by A and B are transferred to the ALU through an A and B-bus. Concurrently the contents of Accumulator (ACC), which holds the results of the previous operation, are stored into the register designated by the C field through the C-bus. Six kinds of operations, ADD, SUB, AND, OR, XOR, and LAJ, are performed in ALU. The LAJ (Load Adjust)
Figure 1—Block diagram and instruction formats of EX
The fourth type, which specifies the data transfer from carry and shift signals are propagated through the EX. According to the port control signals, while ER is connected PR are transferred into and out of the chip through D-port ACC or the ALU Status Register (ASR) to a destination used when combined with the SC instructions.

These instructions are activated when a Chip Select (CS) bit is on. If it is off, the operation is not performed, but carry and shift signals are propagated through the EX.

Two bi-directional transmission ports and four 8-bit I/O Registers are provided. Information stored in AR, DR, and PR are transferred into and out of the chip through D-port according to the port control signals, while ER is connected to the E-port. Using AR and DR for the address and data registers, respectively, D-port permits simple connections to an external bus. PR provides the function of receiving messages from other processors.

Four port control signals AROUT, DROUT, DRIN, and PRIN, are generated by the IOC. These signals are used for bus driver control so that the complicated bus interface circuitry is eliminated.

The low order 5 bits of ER can be used as an address pointer of the general registers. The ASR contains the status of the result of ALU operations, i.e., Carry, Overflow, Zero, and Sign. The ASR is connected not only to the corresponding pins but also to the C-bus, so that the content of the ASR can be stored into general registers. This enables the quick restoration of status which is indispensable to interrupt processing and emulation.

In order to connect several EX’s in cascade to make a powerful ALU, four pins CARRY IN, CARRY OUT, SRIN, and SROUT are provided. The status of the operations are represented by the ASR of the most significant EX and the Zero bits of all the ASR’s. A mode control input is used to specify the most significant EX which manipulates the carry and shift control. The two signals, Carry Generation (G) and Carry Propagation (P), are prepared for external look ahead carry generations.

Sequence controller

The block diagram and instruction set of the SC module are illustrated in Figure 2. The length of the Control Memory Address Register (CMAR) has been determined to be 12-bits wide, so that the next address calculation time is almost equal to that of the parallel data operation in two or four cascaded EX’s. Maximum size of control memory is 4K words which may be enough for ordinary applications, however, extension of size or dynamic microprogramming are easily attained with a few external IC’s.

The SC has five types of instructions. The first is a jump absolute/jump relative instruction with a 12-bit displacement field. The large displacement field not only releases programmers from the anxiety of overbound addresses but also achieves high speed processing. The second type of instruction sets the immediate data into the Stack or Counter. The third one is the conditional jump relative. Branching occurs if the contents of the T/F field are equal to the contents of a bit of the Control Status Register (CSR) or XAM as designated by the BP. The fourth one is provided for interrruptions, subroutine calls, and other uses. The contents of the register designated by the RS-field are transferred through the SC-bus to the register designated by the RD-field. If RS and RD are the Stack and CMAR, respectively, this instruction effects a so-called “return” from subroutine or interrupt process. Control memory addresses can also be dynamically modified by this instruction, if the XAM and CMAR are pointed to by RS and RD, respectively. Then each bit of the CSR is set or reset by the fifth type of instruction.

These instructions are activated by the two CE’s which are wired-ORed inside. If they both are off, the SC just increments its CMAR by one.

As shown in Figure 3, the instruction access and the calculation of the next address, which is done by adding the contents of the Pipeline Address Register (PAR) and the D-field of the microinstruction register, are performed simultaneously. Therefore, the instruction stored in the next address of a branch instruction is always executed. A sixteen-word Stack is prepared so that sixteen-level subroutine nesting or eight-level interruptions are available. A 12-bit Counter performs in two modes as specified by the Counter Mode. In the first mode it is decremented by the pulse from the external pin. If a bit of control memory output is connected to the pin, it acts as a loop counter. In the second mode, the contents of it are decremented at each machine cycle. Therefore, it plays an interval timer, an underflow of which generates an internal interruption. The CSR is sixteen bit wide and consists of a 4-bit ALU status, a one-bit IOC, 4-bit interrupt flags, a 3-bit interrupt mask, and 4-bit general purpose flags. Three pins are prepared to handle external interrupt requests. Two of them are usually assigned to the signals from the IOC, which represent unsuccessful transmission. The other is left for miscellaneous use. When an interrupt is accepted, the control memory address is automatically changed to the fixed address. The contents of CMAR and PAR must be stored in the Stack at the beginning of the interrupt process.

Every bit of XAM is used as a jump condition, and also the contents of XAM can be added to the contents of PAR, so that the next address of the control memory is modified dynamically.

I/O controller

The IOC is designed to interface the EX with the asynchronous bus system. The bus transmission scheme is illustrated in Figure 4. A remarkable feature of this scheme is that as a response to DTSD, three kinds of signals, Data Acknowledge (DTAK), Reject Sync (RSYNC) and Quit (QUIT), are provided for the simple construction of multiprocessor systems. DTAK is returned when there is a
Figure 2—Block diagram and instruction formats of SC
successful transmission. RSYNC means that the data transmission is rejected by the receiver. QUIT is generated by bus control circuits in the case of unsuccessful transmission caused by hardware trouble.

Figure 5 shows a block diagram and its instruction set. The IOC performs two kinds of operations. One is the master mode transmission, which is caused by Read/Write instructions. In this mode, AROUT and either DROUT or DRIN are controlled by the IOC. If the IOC receives RSYNC or QUIT instead of DTAK, it generates RR and QR, respectively. The other is the slave mode activated by reception of a DTSD when Device Select (DS) is on. DS must be connected to its address decoding circuitry which recognizes when the device is being addressed. In this transmission mode, PRIN controls the reception of the data from the "master" device into PR. Once the data are stored in the PR, the flag is set so that the SC recognizes that the message from an external device has been received. PRIN cannot be activated until the PRF is cleared. RSYNC is returned as the response to DTSD, when PRF is "on."

The first type of instruction to the IOC consists of a 2-bit Read/Write field and a one-bit W-field as shown in Figure 5. When the W-field is on, WAIT is activated during the transmission if it is being processed. WAIT suspends the next instruction until the end of the transmission. The second and third type of instructions provide functions to clear PRF and retry transmission, respectively.

**SYSTEM COMPOSITION WITH PM/II**

Examples of a processor system and a multiprocessor system are described below, which are mainly composed of EX's, an SC, and an IOC, as mentioned above.

**Processor composition**

Figure 6 exemplifies a standard processor system implemented with PM/II's. The system consists of four PM/II modules, a mapping array, a dynamic microprogram control module, a control memory, and extra circuitry. The number of IC's composing the system are listed in Table II, which shows that PM/II's decrease the need for extra circuitry.

The system contains two data buses. One is an asynchronous data bus which connects main memory, I/O devices, and the other processors. The other is a synchronous data bus which can connect local memory, multiplier/
divider, and so forth. This bus obviously can connect some I/O devices directly, therefore PM/II can easily construct intelligent I/O’s. The external mapping array is connected to XAM in the SC module which enables a sophisticated emulation for machine languages and higher level languages. These circuits depend on the target machine instruction, which can easily be constructed with a PLA. Two interrupt input pins of the SC are allocated to RR and OR of the IOC. The other interrupt input pin is left for devices which will be connected to the processor bus.

Microinstructions, which are stored in control memory, can be changed by the processor or other processors; therefore dynamic microprogramming is realized in this system. Byte calculations, such as byte-data addition, subtraction, shift, and data-emit, can easily be performed by controlling CE bits.

Systems composed of PM/II can have one of three basic types of microinstructions. Figure 7 shows them for a system with two EX’s. These are Horizontal, Joint, and Vertical types. A Horizontal type microinstruction is 41-bits long and has independent fields for SC, EX, and IOC modules. A two-bit CE field activates the EX modules. The CE bit of the SC is set permanently “on,” which does not appear in the instruction format as shown. A Joint type microinstruction, 33-bits long, is where a field of EX and SC instructions are partially overlapped. In this type 3 CE bits are required to control two EX’s and an SC independently. SC instructions with short formats, such as bit manipulation instructions, can be executed concurrently with EX operations, and EX instructions with short formats with SC operations. A Vertical type microinstruction, 24-bits long, is where EX and SC are fully overlapped. Two CE bits control the operations of EX and SC. When both EX’s are not selected, an SC function can be performed.

An emulator for a typical 16-bit minicomputer was implemented on this processor with the PM/II breadboard. The target minicomputer is MACC 7/F manufactured by Matsushita Communication Industry. About 3k bytes of control memory have been required when using vertical microinstructions. It is confirmed that the execution time is less than twice that of the target when executing FORTRAN programs, even without particular external circuits.

Multiprocessor

Figure 8 depicts a memory shared multiprocessor system currently in operation. P1 is built up by the PM/I components which are prototypes of PM/II. P2 consists of the PM/II modules implemented by the breadboard, and P3 is a MACC 7/F with the bus interface for the PM/II. The element processors, connected with a single bus, communicate through the shared memory and the P-Register. The microprograms of the element processor P1 can be loaded dynamically by P3, and P2 loads its own microprograms dynamically, while P3 has a fixed instruction set. Using the multiprocessor system, many fruitful experiments on the flexibility of hardware configuration, process efficiency, and control program description are made. As a result, the following advantages have been derived to display the applicability of PM/II in the multiprocessor environment; (1) hardware can be connected in a unique and simple way, (2) software for each processor can be developed and debugged almost independently, and (3) the conciseness and ease in description particularly of distributed function type and/or resource shared type control programs can be achieved.

A high level language interpreter is now working on the multiprocessor system, where P1, P2, and P3 process garbage collection, language interpretation, and I/O control, respectively.

Through monitoring the system, it has become clear that the bus contention begins to occur when more than three processors emulate machine instructions simultaneously. Based on the results of these experiments, a new multiprocessor system employing PM/II, in which every element

<table>
<thead>
<tr>
<th>CPU</th>
<th>PM/II modules</th>
<th>4 packages</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAPPING ARRAY</td>
<td>MSI</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>SSI</td>
<td>5</td>
</tr>
<tr>
<td>DYNAMIC MICROPROGRAM CONTROL</td>
<td>MSI</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>SSI</td>
<td>9</td>
</tr>
<tr>
<td>CONTROL MEMORY</td>
<td>LSI</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>SSI</td>
<td>4</td>
</tr>
<tr>
<td>EXTRA CIRCUITRY</td>
<td>SSI</td>
<td>18</td>
</tr>
</tbody>
</table>

TOTAL  | 77

Table 2 - Components of the system shown in figure 6
CONCLUSION

The design concept, specifications, and system composition of PM/II's are described above. Several instruction mixes are estimated, which are illustrated in Figure 9 and compare PM/II with three microprocessors on the market. Intel 3000 and PM/II represent their performance when they compose 16-bit computers, while PFL-16A is a 16-bit microprocessor itself. The performance of PM/II is measured at its vertical instruction set, which shows very satisfactory results.

The breadboards of PM/II are in operation, and have been used for experiments on machine language and high level language emulation, and so forth. A memory shared multiprocessor system with three processors including the PM/II is running, which has made clear the validity of the design concept.

Based on the results of these experiments, a new multiprocessor system employing PM/II is currently under development. Design of the operating system and language processors is nearly finished, and the hardware implementation will soon be started.

Along with the many results which have shown the applicability of PM/II, one disadvantage has appeared, i.e., the slight difficulty in microprogramming. This may be caused by the adoption of pipelining. In order to reduce this difficulty, and to support hardware and microprogram debugging even before hardware implementation, an integrated development support system named IMPULSE has been designed and implemented. IMPULSE consists of a debugging part and a microprogram generating part. The former debugs hardware and microprograms with a description of module connections and microprograms, driving a multi-level logic simulator. The latter is a hierarchical system composed of a high-level microprogramming language compiler, an optimizing assembler in which users can write programs without considering pipelining, and a general purpose micro-assembler. The precise description of IMPULSE will be available in a separate paper.

The three modules of PM/II are now being implemented in the CMOS process. This process is selected because of its high noise immunity and low stand-by power dissipation. PM/II will augment the consistency and completeness to develop other required modules in the future.

ACKNOWLEDGMENT

The authors wish to thank Dr. Y. Koike, the President of Matsushita Research Institute Tokyo, Inc.; Dr. T. Yoshida, the Managing Director, who powerfully promoted the research; Dr. M. Ohnuki, the Director and the colleagues of his group for their ardent semiconductor technological examination; Dr. K. Ueda for valuable discussion; Mr. S. Shimazaki for his assistance in the implementation and experimental stages.

The authors are also grateful to Professor H. Aiso of Keio University for his numerous suggestions from an architectural viewpoint.

REFERENCES
