MOS/LSI launches the low-cost processor

Small computers for dedicated applications can be assembled from a handful of standard ICs

Gerald Lapidus Associate Editor

Large-scale integration of MOS circuits has enabled the placement of a lot more processing power on a single semiconductor chip, and hence in one IC package. The present impact of the development is perhaps greatest for the designer of equipment such as programmable calculators, machine and process controllers, business machines, and peripheral gear. Until now, manufacturers of this type of equipment, who often sell finished gear in "onesies and twosies" to end users such as laboratories, universities, or for industrial applications, have been inclined to build in minicomputers because nothing else was available to fulfill their needs. The coming of the low-cost processor permits many of these manufacturers to build this computing capability into their equipment at a fraction of the cost of minicomputers. Still another type of user is replacing hard-wired solid-state logic or even relay logic with these programmable devices to lower manufacturing costs and simplify program changes.

One drawback of these MOS processors is slower instruction execution times than with minicomputers built with bipolar circuits. Another relates to the still generally "immature" applications support available from the semiconductor makers of these devices. (Minicomputer makers, in contrast, usually provide packages that are complete with substantial software libraries, interfacing, compatible peripherals, and field service.) A third relates to the need to buy large quantities of a given IC in order to offset masking charges for customized instruction sets and related features.

In spite of these obstacles the ability to purchase basic processors for as little as $50 (practical computers built from these devices may cost several hundred dollars) suggests that the new ICs will achieve wide popularity. Furthermore, it is conceivable that some minicomputer makers may incorporate the MOS ICs into the low ends of their lines.

The present low level of support by semiconductor manufacturers is due mainly to the early state of market

I. Typical applications of low-cost processors

Color analyzer for textile industry	College courses in computer design
Machine-tool control	Burglar alarm system
Telephone switching	Automotive testing
Module tester	Magnetic credit-card reader
CRT and video display systems and controllers	Photon-counting equipment
Computer terminal	Hobby kits
Process control	Traffic control
Optical character reader	Parking garage fee computer
I/O control and microprocessor	Instruments (e.g., automatic capacitance bridge)
Digital cassette system	Signal analysis
Data collection	Batching system
Data communication terminal	Keyboard logic
Point-of-sale system	Supervisory system
Typesetting	Fast Fourier processor
Peripheral control	Pattern-recognition system
Airborne navigation computer	Controller for security system
Inventory-control system	Dedicated processor for speech processing
Smog-detection devices	Radar signal processor
Medical electronics	Digital nuerosimulator
Telemetry ground station	Music synthesizer

Tradeoffs in designing MOS/LSI processors

Driving a wedge into the low-cost computer market puts severe constraints on the processor design. The primary problems are organizing the system to get the most speed out of MOS/LSI, while keeping the cost down to remain competitive with standard minicomputers. An important element in achieving these goals is the use of microprogramming for instruction execution instead of the more common random logic. A portion of the design philosophy used in American Micro-systems' 7200 is presented below, based on the experiences of Gaymond W. Schultz and Ray M. Holt of American Micro systems. Additional details will be included in a paper presented at the 1972 Fall Joint Computer Conference in Anaheim, Calif., on December 7. The complete paper will be published in the AFIPS conference proceedings, vol. 41.

The following factors were considered in evolving an optimum LSI design:

1. The processor should be able to function as an 8- or a 16-bit machine and its performance should be equivalent to TTL-based minicomputers.
2. Speed is a major factor in achieving performance approaching that of TTL minicomputer designs.
3. TTL requirements external to the LSI should be minimized for cost effectiveness.
4. Only those LSI processing technologies that are presently being used in low-cost mass production should be considered.
5. The machine must be capable of addressing 65K words or bytes of memory directly. The 16-bit address to memory must not be sent as a single byte transfer, even in 8-bit versions.

The tradeoffs in implementing the control section for the LSI machine are bounded primarily by two criteria: system partitioning and related pin limitations, and efficiency of chip-area utilization.

If the entire machine could be placed on a single chip, only the second restriction would present a problem. The alternative is to divide the system into several chips and partition the registers and arithmetic logic unit (ALU) as byte "slices." However, this approach introduces speed problems because of chip-to-chip transition delays of approximately 150 to 200 ns. The transition problem can be solved by using pipelining techniques that mask the time spent in communicating control between chips so that the registers and ALU section can be operated autonomously.

In byte-slice partitioning, pin limitation is a major factor in the design of the control section. Present packaging technology and costs constrain the designer to 16-, 24-, and 40-pin packages. The 16- or 24-pin configurations are undesirable because the registers and ALU could only be partitioned into 4-bit slices and the resultant chip-to-chip transitions between the various sections of the ALU would greatly reduce machine speed. Therefore, a 40-lead package was selected. The leads were then allocated as follows:

- At least six pins for power sources and clocks on all the chips.
- Fifteen pins on the register chip for 16-bit memory bus interface.
- At least three ALU lines (carry in, carry out, and zero detection) needed.

This left a maximum of 15 pins for controlling the registers and ALU. This pin limitation problem, combined with the fact that, for a given speed, read-only memories are approximately six times more efficient in chip-space usage than sequencers and associated random logic, lead the designers to consider microprogramming techniques for instruction execution.

Aside from the basic LSI layout problems, a number of other system tradeoffs heavily favored the microprogrammed approach, including:

- The need to emulate instruction sets for existing TTL designs to minimize software development.
- The need to apply the LSI machine to areas where the microlevel of control is either mandatory or desirable for meeting speed requirements. A good example is a CRT display terminal in which the microcontrol could eliminate the need for core storage of macroinstructions.
- The desire for flexibility at the lowest logic level.
- The short development spans, which do not allow for complete redesign of complex chips when minor errors are discovered during the design and debug phase. Here, all that is required is the modification of the appropriate microinstructions.

Instruction decoding

The instruction fetch and decode process poses a difficult speed problem. For the first phase of multiphase instruction execution, it is desirable to load the instruction register (IR) with a new instruction and immediately relate the IR contents to a microinstruction ROM location. However, this ROM address-mapping process requires typically a minimum of 300 ns as well as 300 to 400 ns to access the first microinstruction used in executing the macroinstructions. This is approximately equal to the time required to execute two microinstructions. This problem can be solved by designing the control section to initiate the execution of one or more microinstructions after the instruction register is loaded. Thus, as soon as the mapping array output is valid, it causes microinstruction completion to be inhibited or skipped. The inhibited microinstructions are used to fetch additional bytes in the process of forming a 16-bit memory address. For single-byte instructions, such as "clear accumulator," the fetch address phase is skipped. For all other instructions the address is fetched and the instruction is decoded concurrently.

This same technique may be applied during the execution of memory reference instructions to differentiate between executions that store or fetch operands. In this case it is assumed that the execution phase will fetch an operand; if it is to store, the fetch operand microinstructions are inhibited or skipped.

Speed is not a problem for the second and third phases because the instruction already is in the IR.

The speed of the LSI machine is determined primarily by the speed at which arithmetic operations can be performed—specifically, the speed at which the operands can be accessed from two registers, added, and then returned to one of the original sources or a new register. ROM access speed and the microprogrammed branching process also become important speed factors.

development. All suppliers are busily at work improving product support, but since their present aims are toward computer system OEMs, it is not likely they will carry their support far enough to compete with traditional minicomputer suppliers for end-user business.

However, this is not the prime area they are trying to capture. Long-term goals are a new class of "subminicomputer" applications involving simple control and data processing applications for which minicomputers are prohibitively costly. A related area is replacement of underutilized minis. (One industry wag characterizes this segment of the market as applications in which "thumbtacks are being driven by sledgehammers.")

Among the uses visualized are as intelligent terminals that perform some simple processing steps on data prior to entering them into a central computer; in programmable control of discrete or continuous processes; as replacements for hard-wired relay and solid-state logic; in automatic testing; and in sophisticated calculators. A list of potential applications is given in Table I. This list was generated by applications engineers at Intel, but is applicable to devices produced by the other processor IC makers as well.

The devices

Processor chips evolved from devices developed for special customers for use as peripheral controllers and calculators, which a number of semiconductor manufacturers have been developing for several years. The first product line to enter the general marketplace was Intel Corporation's MCS-4 microcomputer set, introduced in 1971. Other IC makers who have taken the plunge since then or are just now introducing the devices are American Micro-systems, Inc., Fairchild Semiconductor, and National Semiconductor.

A bird's-eye view shows two basic types of organization—parallel as in conventional minicomputers and binary-coded decimal BCD (4-bit) serial as in calculators. The parallel machines have 8- or 16-bit word lengths, some of which are expandable to up to 32 bits by paralleling additional circuits. The BCD serial processors, on the other hand, offer extremely long word lengths (up to 100 bits or 25 BCD digits), which are operated on one digit at a time. This tends to slow down instruction execution time even further, but serial processors are less expensive to produce than parallel processors. They also offer variable word lengths, which provide some advantages in flexibility.

Standard instruction sets typically consist of the 40 or 50 found in low-cost minis, including addition and subtraction, but not multiplication and division. These functions must be accomplished through time-consuming software routines or else implemented in instruction-execution circuits built from read-only memory (ROM) hardware, as is also the case with low-cost minis. An important feature of these IC sets not found in a number of minicomputers is the ability to develop a fully customized, ROM-based instruction set to tailor the processor completely to the needs of the application. (See box on page 34.)

Each manufacturer offers a compatible family of semiconductor random-access memories (RAMs) and ROMs for program and data storage. For customers requiring a nonvolatile form of read–write storage such as core, the manufacturers say that interfacing is

What the minicomputer men say

A sampling of leading minicomputer manufacturers indicates that they do not expect processor IC families to make a significant dent in their market, at least not in the near future. Their arguments are based on slower performance on the part of processor chip sets due to MOS technology and meager support services. However, all admit to watching this field carefully and some are considering incorporating these processors into their peripherals at least. Selected comments follow:

CPU cost incidental

We recognize the importance of chip technology for central processing units—it will help lower the overall price of a computer and extend computer applications into new markets. But we don't foresee the minicomputer market being significantly affected by chip technology at this time because of the incidental cost of the CPU with regard to the overall computer system. There are enough extraneous costs that are not directly benefited by chip technology, such as software, memory storage, peripherals, warranty, and sales-marketing support, to keep the minicomputer a viable force in existing markets.

—Roger Cady, PDP-11 Engineering Manager
Digital Equipment Corp.

No cost impact

It is possible for a minicomputer company to go into the semiconductor business or a semiconductor company to go into the minicomputer business, but in each case the economics of the particular business takes over and prices tend to become cost-competitive. The cost of minicomputers will not be greatly affected by integrating the processor circuits any further in the next few years.

—Lawrence Seligman, Staff Engineer
Data General Corp.

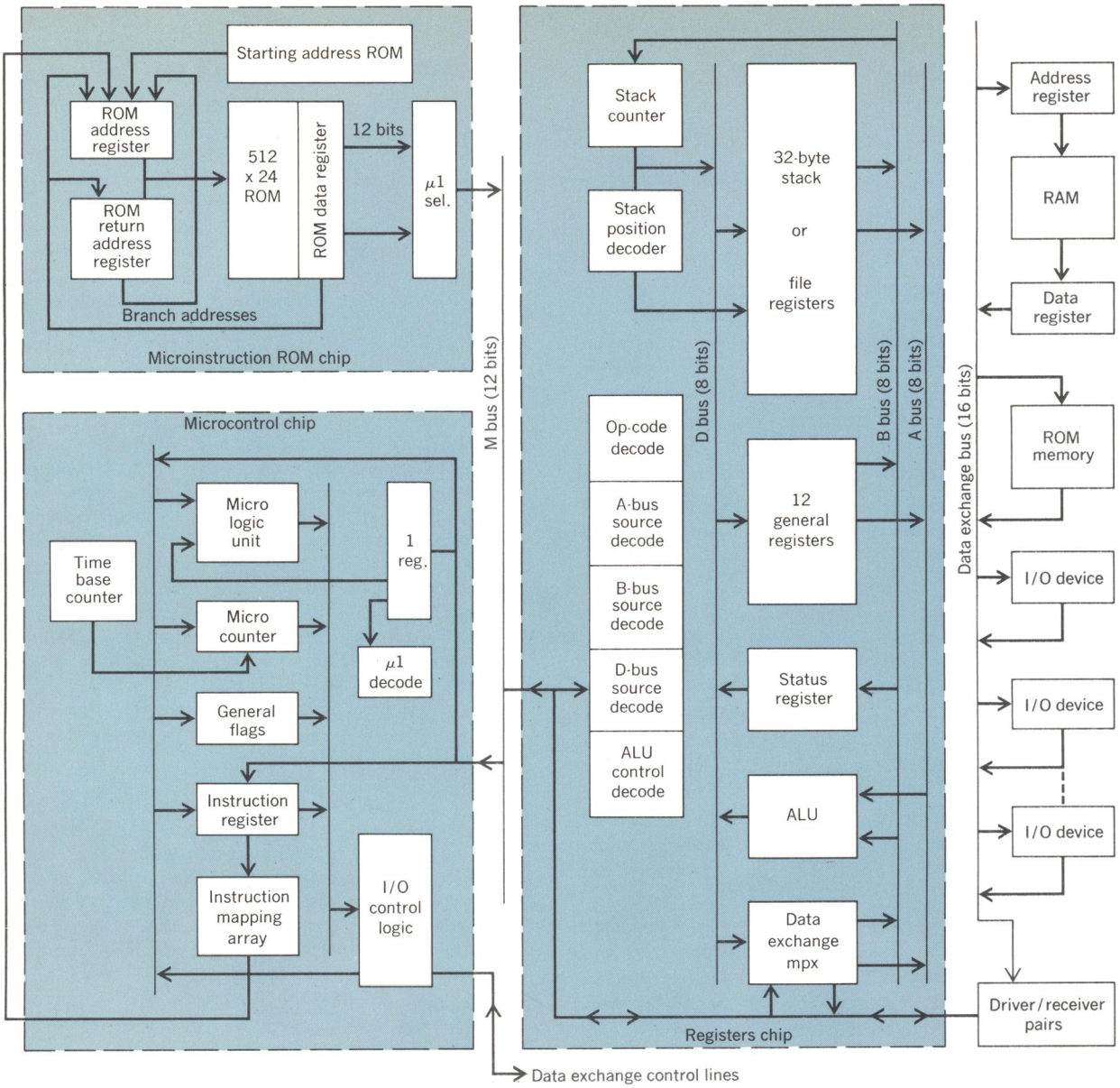
Peripheral role

These subprocessors could be put to good use by minicomputer suppliers as peripheral controllers. The devices could greatly simplify controller design because their use would standardize the interface.

Thus, we could have a standard piece of hardware that might be called an "open-ended controller." It would serve to interface processors to any peripheral simply by adding the proper program and a few front-end circuits. Similarly, peripheral driver software development would be simplified, so that instead of writing separate drivers for each type of I/O device, we might instead have several general classes of drivers, each of which could handle a number of related peripherals.

Such a controller could handle control and data check functions as well as perform code conversion and formatting. At present many of these functions are performed by the minicomputer. Thus, another advantage would be to remove some of the minicomputer overhead and free it for other jobs.

—Arthur Furman, Vice President of Development
Interdata, Inc.



[1] American Microsystems' 7200 consists of three chips. Basic word length is 8 bits, expandable to 16 bits by an additional register and ALU chip. Its most significant features are a universal bussing system and a 32-register pushdown stack.

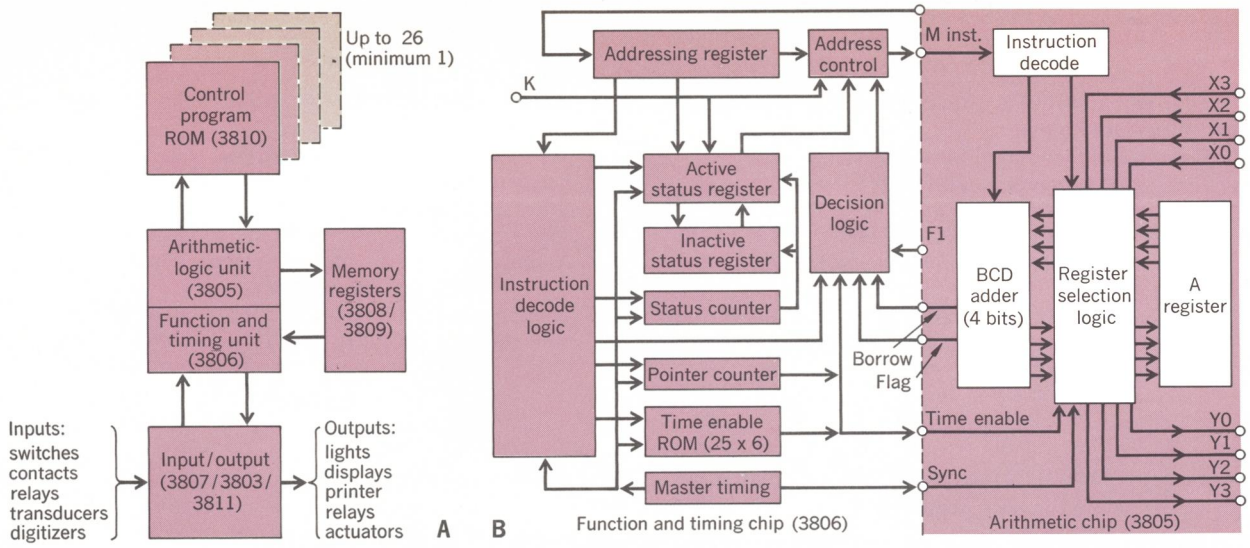
not difficult and standard interfaces may soon be offered if there is sufficient demand.

In the software area, all suppliers presently offer, or will soon announce, basic programs such as assemblers, loaders, and text editors. Some even offer Fortran capabilities using host computers for assembly. Interfacing I-Ds is at present largely a do-it-yourself effort, with advice available for interfacing common peripherals such as teletypewriters. One exception is keyboards and displays, which can be interfaced with standard ICs, thanks to calculator ancestry.

Although little immediate impact is expected on the sales of minicomputer makers, there is a long-term potential for strong competition. (See box on page 35 for what the minicomputer men say.) As the costs and speed of MOS continue to improve, or if some day it becomes feasible to build these processors with all-bipolar techniques, the performance gap between these processor IC families and standard minis may shrink. A by-product of this reduction might be to tempt some IC manufacturers to make a full-scale effort to enter the minicomputer marketplace and offer software and service comparable to that of traditional mini suppliers. If this should come about, the first to feel the pressure could be minicomputer suppliers offering stripped-down mainframes to OEMs at bargain prices.

Some computer suppliers may also incorporate these processor families into the low ends of their lines. Already one or two small companies have been formed to market such processors as the heart of general-purpose computer packages.

The balance of this article describes the significant features of processor IC families offered by American Micro-systems, Fairchild Semiconductor, Intel, and National Semiconductor. Each has a unique form of architecture and different capabilities.



[2] Fairchild's PPS-25 is aimed at filling the gap between calculators and minicomputers. A—Basic circuit complement. B—Functional block diagram of ALU. The similarity to calculators is borne out by the use of long registers, modest temporary storage capacity, contact-oriented inputs, and display or printer outputs. Data register access is mask programmable to permit various fields to be defined instead of the system being constrained to the standard 100-bit word length.

American Micro-systems: universal bus

The basic version of the AMI 7200 consists of three chips (excluding program storage), as shown in Fig. 1. The three-chip processor forms an 8-bit machine, which can be expanded to 16-bit parallel operation by adding another register and ALU (arithmetic and logic unit). Its most significant architectural features are a universal bussing system and a 32-register pushdown stack, both of which will be covered shortly.

The basic operating sequence begins with an instruction being read out of memory into the microcontrol chip, where it is decoded and the appropriate starting address in the microinstruction ROM is accessed to begin the sequence of steps required to execute the specified instruction. Thus, each instruction is decoded by relating the instruction to a starting point in the ROM. The microinstruction ROM can handle 512 words, 24 bits each.

Pipeline microinstruction execution increases processing speed by utilizing the hardware more efficiently (on a time basis). Although the microinstruction format is 24 bits long, microinstructions are processed in two 12-bit segments. The first 12 bits specify two registers from which operands must be accessed; the last 12 bits specify the destination of the result and the ALU operation that is to be performed. In actual operation, however, the last 12 bits of a given instruction are processed while the first 12 bits of the next instruction are being decoded. In this way, the machine can be set up to perform the next required instruction while the current instruction is being completed. This is advantageous because register accessing and ALU operations tend to take a relatively long time to accomplish.

By using 24-bit instruction words, the 7200 is a three-address machine (three locations can be specified in a single microinstruction), yet the pipelining technique reduces processing time to one third or one half of TTL machines not using pipelining, according to AMI.

The data-exchange bus permits parallel data transfers between memories, processor, the control panel, and I/O devices. Bus usage is controlled on a handshake basis, whereby any device on the line can request use of the bus to transfer data between it and any other device. If more than one processor is placed on the line, a separate controller is used to organize priorities between processors, thus permitting multiprocessing operation. All other devices on the line are assigned priority by polling them on a daisy-chain basis or else determining priority with a priority chaining network.

Interrupts and subroutines, whether real-time or program-generated, are handled by a 32-register pushdown stack. Here, the sensing of an interrupt causes the status of the program counter and status register to be stored in three of the registers on a last-in, first-out basis. Subroutine calls cause the contents of the program counter to be stored in two additional registers. The pushdown stack alternatively may be used as a file of 32 general registers in addition to the 12 registers ordinarily used for this purpose.

The price of a set of the three chips (8-bit version) with the standard ROM instruction set is $300 in OEM quantities.

PPS-25: long words

The Fairchild processor bears a close resemblance to a calculator in architecture but offers many programmable, general-purpose features. Indeed, it is billed by Fairchild as filling the gap between intermediate to upper end calculators and minicomputers. The similarity to calculators is borne out by the use of long registers of 25 BCD digits (100 bits), modest temporary storage capacity, contact-oriented inputs (e.g., keyboards and digitizers), and display or printer outputs.

The basic circuit complement (Fig. 2) contains a 3805 arithmetic unit and a 3806 function and timing unit, which together perform all timing and arithmetic functions. Microprograms and lookup tables are stored on one or

II. Comparison of MCS-4 and MCS-8 processors

MCS-4 CPU	MCS-8 CPU
Four-bit parallel CPU with 45 instructions. Decimal and binary arithmetic	Eight-bit parallel CPU with 45 instructions. Binary arithmetic
Sixteen 4-bit general-purpose registers	Six 8-bit general-purpose registers
Nesting of subroutines up to three levels	Nesting of subroutines up to seven levels
Test on external signal	Interrupt capability
Synchronous operation with memories	Asynchronous operation with memories
No interface circuitry to memory and I/O required	Approximately 20 standard TTL packages required for memory and I/O interface
Memory capacity expandable through bank switching	Memory capacity expandable through bank switching
Minimum system: CPU and one ROM	Minimum system: CPU, 20 TTLs, one standard ROM

more (up to 26) 3810 ROMs, each a 256 × 12 matrix.

Data are stored in shift registers, if required, with the system able to handle up to seven registers (including ALU register).

The ROM is mask programmable and data register access is also mask programmable to permit various fields to be defined as needed, instead of being constrained to the standard word length. Variable word lengths are tracked with a pointer counter that supersedes the standard time-enable signals that otherwise would define word length. In the standard system each 4-bit BCD digit is treated as a separate character and these characters are shifted out and processed one at a time. The arithmetic unit can operate on one group of 25 digits in a single ALU cycle (62.5 μs), and no instruction requires more than a single cycle.

The PPS-25 therefore is a series-parallel machine. Data can also be stored in hexadecimal form for better packing density, but when used in this manner the system cannot treat the data as numeric.

The ALU calls up the program steps or data from the appropriate control ROM. Each control ROM contains 256 instruction addresses. Therefore, a full complement of 26 ROMs can store up to 6656 instructions.

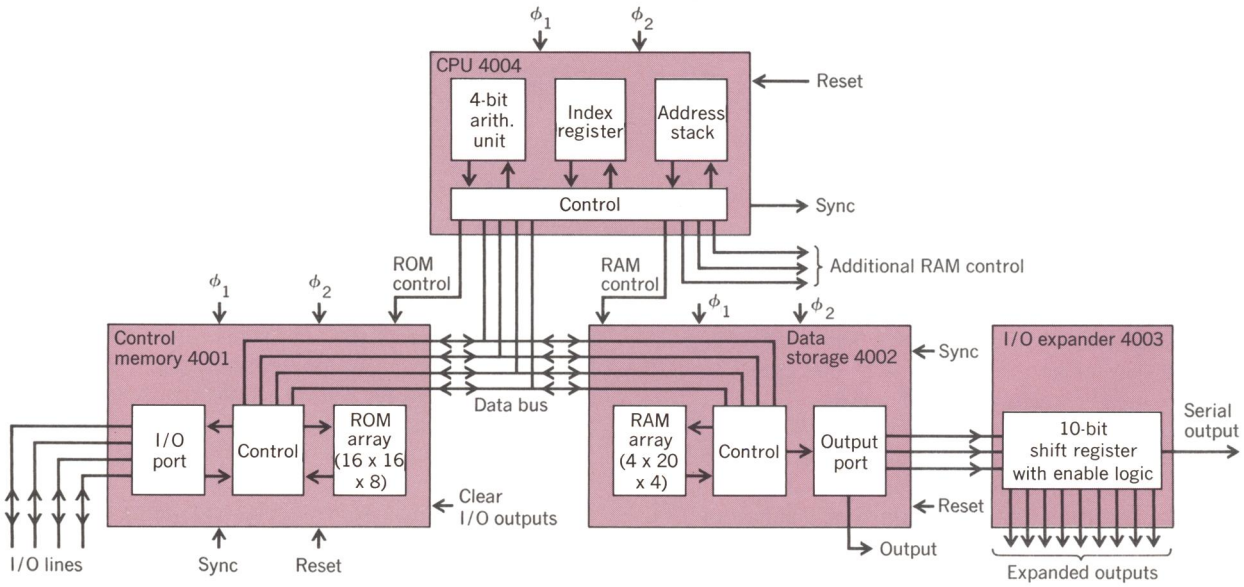
Because of its resemblance to calculators, the PPS-25 is best suited to applications requiring arithmetic operations, such as point-of-sale terminals, cash registers, banking terminals, and small business machines. Other areas in which this capability may be required are process control systems, vending machines, medical instrumentation, and numerical machine control.

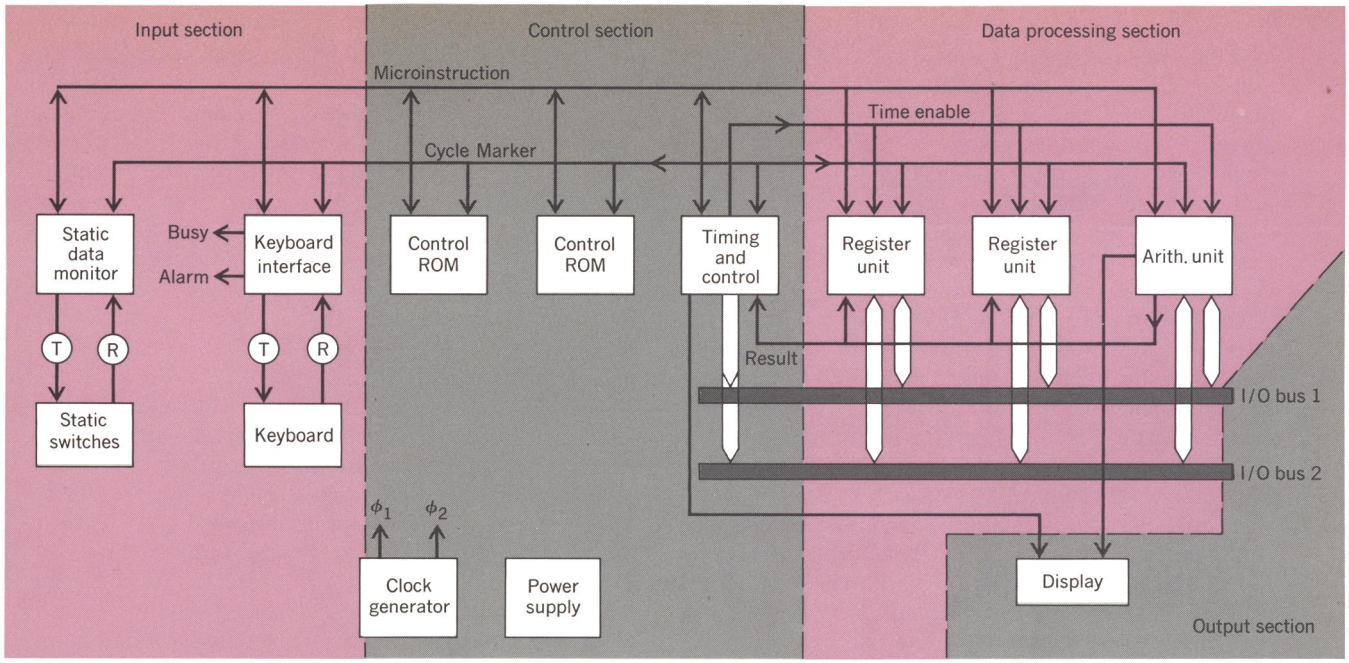
For keyboard entry, the 3803 and 3807 devices provide a direct interface with up to 62 keys and 32 mode switches. On the output side the 3811 permits up to 16 digits to be displayed, or else multiple 3811s can be used to provide a general communication interface.

Intel MCS-4 and MCS-8: heavy backup

The MCS-4 and 8 are both 8-bit systems. The MCS-4 operates on 4 bits at a time and MCS-8 processes all 8 bits in parallel. They were conceived for different types of applications. The MCS-4 is an outgrowth of the Busicom printing calculator chip, whereas the MCS-8 evolved from the Datapoint 2200, a "smart" terminal. The MCS-4 is better suited to decimal arithmetic than its brother because its instruction set is specifically designed

[3] Intel's MCS-4 can be bought in the minimum configuration (one CPU and one ROM chip) for less than $50 in quantities of 100. Data are handled in 4-bit segments and instructions are processed 4 bits at a time, with either eight or 16 timing intervals required for complete execution of an instruction.





[4] National's MAPS is a serial system. Processors with this type of architecture are particularly suited to applications requiring operator interaction and long data words. Examples are credit verifiers, data terminals, and scientific calculators.

to handle BCD numbers without conversion from basic binary. BCD numbers can be handled on the MCS-8 only if a lengthy subroutine is employed for binary to decimal conversion. The MCS-8, because of its larger word size, is more suited to performing simple operations on large quantities of data, especially where the more powerful addressing capability of an 8-bit machine is needed. Other instruction capabilities found in the MCS-8 and not in the 4 are AND, OR, and other logic instructions to facilitate data handling. An additional MCS-8 feature that is especially useful in real-time operation or in data communications is priority interrupt. Comparative specifications for the two families are listed in Table II.

These families are aimed more closely at the end user than are the other processors covered here. Their standardized instruction sets substitute for custom ROM masking, which could make low-volume applications costly. However, the drawback of a standard instruction set may be less flexibility and lower efficiency in performing specialized tasks.

Other aids offered by Intel include assembled prototyping boards consisting of processor chips, clock generators, RAMs, electrically programmable and erasable PROMs, teletypewriter interface, and a hardware bootstrap loader. The purpose of the kits is to permit the system designer to develop and check out his programs before committing himself to a finalized system. The software library is quite extensive compared with other processor families and presently includes a text editor, printer driver, RAM test program, logic subroutines for the MCS-4, arithmetic routines, teletypewriter keyboard input, A/D conversion, assemblers, loaders, and Fortran IV assembler and simulator.

The MCS-4 family consists of a central processor chip (4004), ROM chips (4001), RAM chips (4002), and shift-register chips (4003) for I/O expansion, as shown in Fig. 3. The minimum configuration, one central processor unit and one ROM chip, costs less than $50 in quantities.

Instruction execution in the MCS-4 requires either eight or 16 cycles of a 750-kHz clock. It begins with generation of a synchronizing signal that is sent to the RAMs and ROMs to indicate the start of an operation (first timing interval). In a typical instruction sequence, the CPU first sends 12 bits of address data (in three 4-bit segments) to the program storage devices in the first three clock intervals. One word in each ROM is selected, but only the ROM chip holding the desired instruction is permitted to read out. This instruction is sent over the four-line data bus in two 4-bit segments (two intervals). Execution is completed in the next three intervals for single-word instructions. An additional set of eight intervals is required for two-word instructions.

The MCS-8 instruction execution follows a similar sequence, but requires five timing intervals for one-word instructions, ten for two-word instructions, and 15 for three-word instructions. An MCS-8 system is built up in a manner similar to the MCS-4, but is 8 bits wide. Among the circuits it is designed to work with are 2048-bit ROMs (types 1301, 1601, and 1701); 256-bit and 1024-bit RAMs (1101, 1103, and 2102); and single and dual 1024-bit shift registers (1402, 1403, 1404, 2401, and 2405). A complete functioning system can be built with one CPU, one ROM, and 20 standard TTL devices. The CPU alone costs $60 in OEM quantities.

MAPS and GPC/P: serial and parallel

National offers two virtually unrelated processors: MAPS (microprogrammable arithmetic processor system), a 76-bit serial, low-cost processor; and GPC/P, a parallel processor that can be configured for a standard word length of 4 to 32 bits, in increments of 4 bits.

MAPS is designed to operate in systems that require

long data word lengths and do not demand rapid instruction execution, with mixed BCD and binary arithmetic functions. Data can be processed in any of six mask programmable word lengths up to 76 bits long. MAPS is particularly suited to applications-based operator interaction because of compatible keyboard interface and input data-monitoring devices offered as part of the family. Examples of such applications are calculators, credit-verification systems, and data terminals.

The basic members of the MAPS family are an arithmetic unit (MM5700), register unit (MM5701) containing two 76-bit shift registers, timing and control unit (MM5702), keyboard interface unit (MM5704), control ROM (MM5703 and MM5705), and a static data monitor (MM5706). A typical MAPS configuration is shown in Fig. 4. Price of a typical system such as a 16-digit, 10-function, 2-memory calculator is less than $50 in quantity.

Microinstructions governing basic processor operations are distributed around the basic functional blocks rather than in separate ROM circuits, as is the case with other processors. Data are operated on by 32 microinstructions contained in the arithmetic and register chips. For basic control operations an additional 23 microinstructions are available in the timing and control circuit. Program instructions are stored in the control ROM; each contains 256 words (10 bits/word) and page-selection logic for expansion.

The basic system is expandable by adding register units, control ROMs, keyboard interface elements, and input data monitors. Up to 32 control ROMs, or else the equivalent of 8192 instruction words stored on RAMs or PROMs (programmable ROMs), can be directly addressed by the system. Up to ten register chips can be accommodated, providing up to twenty 19-digit data registers. Likewise, up to four keyboard input devices can be connected, each able to handle 32 dynamic keys, plus eight static switches.

The instruction set (distributed microinstructions) either may be selected from one of the standard sets provided by National, or customized according to application requirements. Program instruction words are 10 bits long and are either single- or double-word units. Typical instruction execution time is 150 μs (2-μs/bit data rate).

The GPC/P is a fully parallel processor, in which the basic data word can be 4 to 32 bits long, with the desired word length obtained by connecting the appropriate chips in tandem. The two basic devices are the register and arithmetic-logic unit (RALU) and the control ROM. The remaining elements are standard bipolar circuits.

The RALU is a 4-bit slice, shown at right in Fig. 5. For longer word sizes, additional RALUs are paralleled. The RALU consists of a 16-word pushdown stack, program counter, status register, two memory registers, four accumulators, an arithmetic-logic unit, and an I/O multiplexer. The control ROM holds the microinstruction ROM, control circuits, and timing. Additional control ROMs may be used to expand microprograms.

The program memory may be a RAM, a ROM, a PROM, or a combination of these. In some applications, the same memory may be shared by a number of processors. An example of this sort of application is a case in which one processor is used as the controller for a peripheral device that inputs data and the other processor is used as a conventional central processor. The memory is composed of standard devices. Maximum memory size (address limitation) is 65 536 words.

The control function of the processor is provided by a microprogram contained in a ROM. Associated with the ROM are circuits that provide ROM address control (that is, branching within the microprogram) and distribution of the control signals to the system. Conditional branching is accomplished by selecting one of the 16 possible inputs to the conditional jump multiplexer and gating its output to the ROM address control logic. Inputs to the multiplexer are signals generated within the processor or signals generated in the I/Os, such as an interrupt request.

A group of 16 flag flip-flops are available to be set or reset under control of the microprogram. Some flags are dedicated to requirements of the processor (e.g., read memory flag and write memory flag), whereas others may be used as required by the system application (e.g., as status flags).

[5] National's GPC/P is a fully parallel processor built up from two basic chips—the RALU (register and arithmetic unit) and CROM (control ROM). Each is 4 bits long and word lengths are expandable to 32 bits by adding chips in parallel. The maximum memory size is 65 536 words.