# MICROPROCESSOR SCORECARD®

Jerry Ogdin°°

Microcomputer Technique

When the history of digital electronics is written, 1974 will be remembered as the year of the microprocessor explosion. From humble beginnings with two products in 1971, an industry that had been predicted for at least a decade came to pass. Now, a mere four years later, there are at least twenty-five different microprocessors available, announced, promised or under development.

This embarrassment of riches has not been lost on forward-thinking digital designers and system architects. The microprocessor has been designed as a sub-system into a startling variety of products and applications. The dollar volume growth from one year to the next appears to be about five-to-one in microprocessor and related component sales. Clearly, it has become a popular tool. Well over 100,000 will be shipped this year.

For the digital designer unfamiliar with computers, all this has come as a great shock. Safely able to ignore the vagaries of software in the past, many engineers have begun to learn that some of their hard-won skills are becoming obsolete overnight. For years, for example, the objective was logic minimization; later, engineers successfully retrained themselves into minimizing system cost (usually synonomous with minimum package count). Now, after all that, the package count for incredibly complex systems is on the order of ten off-the-shelf chips, and all the important logic is in software.

Once the prospective user acclimates to the unique suite of microprocessor characteristics like word size and speed, one of the hard tasks begins: How, in a new and burgeoning field, with new major announcements actually crowding each other out of space in the trade press, does the system architect with limited microprocessor experience select the one best processor for the job? For that matter, how does an old-hand at micros know when there are significant new advances worthy of attention? This edition of "New Logic Notebook" will help.

In addition to a single large reference chart and a brief outline of the features of important products, this issue covers the process of selection. Here we suggest some criteria and what to look for, and some helpful hints to ease you over the learning curve.

°° This survey is part of a monthly publication, the "New Logic Notebook". For information, write : Microcomputer Technique Inc., 11227 Handlebar Road, Reston, VA 22091, U.S.A.

## PROCESSOR SELECTION

Selection of a microprocessor is not a task that can be performed in abstract; the requirement of the application must always be kept clearly in view. In general, almost any processor can satisfy almost any application's requirement, so long as speed is not a factor. However, most applications have real-time constraints, and so processor selection is an important activity in successful system design.

Some processors are naturally easier to use for certain applications than others. Some of the available processors, like the Intel MCS-4, Fairchild F-8 and Electronic Arrays' device, make the handling of small arrays very easy. Others have, for example, features that allow simple input/output interfaces. Each processor tends to have some strong points and some weak when viewed in the light of typical applications of microprocessors. However, a processor's particular strong point may prove worthless in some specific applications; and a weakness may disappear in other uses.

Viewed from the vantage point of the system as a whole, the experienced designer can see three major areas for investigation:

- Software Design, which covers that features of the microprocessor as seen by the programmer,
- Hardware Design, including the requirements of the environment in which the processor must operate, and
- System Design, which affects the interface conventions between hardware and software design.

In the detailed discussions of microprocessor feature analysis that follow, no implications of relative importance should be attached to the order of presentation. The order of discussion is intended to cover the broadest issues first, and the most detailed issues last. Nor is this treatment covered in the same order as the checklist that appears on this page.

An optimal microprocessor selection can be made using either of two strategies. In one scheme, the designer re-arranges this list into an order of decreasing importance; the top of the list should be the most important parameter about the potential microprocessors. The system designer can then check the Scorecard and other available information to select those processors that deserve further examination for less important criteria.

Software Design
- Word Size (Data/Instruction)
- Address Capacity (Program words)
- Register-Register Add Time
- Number of Registers
  - Arithmetic
  - Index
  - General Purpose
- Return Stack Size
- Addressing Modes

Hardware Design
- Clock (kHz/phases)
- Voltages Required
- Power Dissipation
- Compatibility
- Package Sizes
- Completeness of Parts Family

System Design
- Features
  - Interrupts
  - One-chip CPU
  - Microprogrammable
  - DMA Ability
  - BCD Arithmetic
- Vendor Commitment
  - Software Support
  - Documentation
  - Application Notes
  - Design Aids
- Product Longevity
  - Second Sources
  - Standardization
- Price & Availability

Alternatively, the system designer can assign weighting factors to each attribute in the list, and rate each microprocessor of interest on a scale from, say, 1-10. The sum of the products of rating times weighting should pinpoint the best processor and neighboring alternatives.

A general note of caution is important here. Always check the final findings in a processor selection exercise with the vendor, the vendor's literature and other knowledgable users of the device. You will probably uncover some interesting tidbits of information that may change your selection. Be wary, too, of many of the manufacturers' literature; often, quite ordinary features of the computer

are described in such glowing terms as to imply more capability than really exists. The use of inappropriate terms in describing features is very common; consultation with a knowledgable user can point out these potential sources of embarrassment.

## SYSTEM DESIGN CONSIDERATIONS

In the design phase of microprocessor based applications, the choice of a processor should--ideally--be an open question. In those cases where the selection of some arbitrary processor is not over-ridden by prior organizational commitment to a particular technology, the first decision point comes during design. In order to select from the myriad of processors, some well-placed questions can probably eliminate several contenders. In general, if various processors can be dropped from further consideration, the designer's job is simplified because a smaller number of processors need be examined in detail.

### System Features

Some major features of microprocessors are:

Interrupt Structure,
One-chip CPU Packaging,
Microprogrammability,
DMA Ability, and
Arithmetic Modes.

These factors, while not the only important features of microprocessors, are important enough to be considered among the first items to examine. If, for instance, the application demands the ability to accept or emit large blocks of data quickly, then processors without a DMA ability can probably be rejected from further consideration.

Interrupts may or may not be important, depending upon two factors: 1) does the application require real-time quick response to external events, and 2) does the software design strategy encourage the use of interrupts? Many applications won't require interrupts--even real-time applications. Instead of allowing main-line processing programs to be interrupted, software organization may assure the testing of some external status signal frequently enough to guarantee service. There are some software design philosophies that consider interrupts as nuisances to be avoided except for nearly catastrophic fault conditions; the software designer should make a choice before the processor is selected.

One-chip CPU packaging has a substantial effect on assembly and repair costs as well as affecting the size of the finished product. Even among one-chip CPUs there are tremendous varieties in the number of ICs which must be added before achieving a working computer. A CPU that multiplexes addresses through the same pins as data requires several external packages to catch and hold a memory address. Most of the larger packages avoid this multiplexing. Other areas which frequently require substantial amounts of supporting logic are state decoding and I/O control. The signals required to enable interrupts, enable and address input/output ports, write into storage and interrupt or reset the system may be available at CPU pins or they may require as many as ten or twenty packages of decoding, storage and timing logic. While the parts cost of these extra chips is relatively small, they can double or triple the manufacturing costs of the computer.

Microprogrammability allows the fine structure of the microprocessor to be changed, while the gross structure remains. In particular, the number of registers the programmer sees and the suite of instructions can be modified within limited realms, but gross changes are not possible. Microprograms are usually stored in read-only memories, either on the computer's control chip or ex-

ternally in standard ROM's connected to the control logic. Some of these ROM's are mask-programmed, while others can be field-programmed or replaced with read-write storage to ease the development of customized instruction sets. Microprogramming is important when a microcomputer is being designed that will emulate some other (more popular) computer, or when some specialized applications-oriented instructions need to be implemented to augment an existing instruction set. Common but time-consuming software routines (such as multiply) can be implemented as microprogrammed instructions to speed up application execution. Virtually any program can be reduced to a microprogram. Relatively complicated processing tasks (like Fast Fourier Transforms) will execute much faster if implemented as a microprogram.

DMA is an abbreviation for Direct Memory Access; this has been a popular option on minicomputers for a long time. DMA lets high-speed peripheral devices gain direct access to main storage without bothering the CPU; the alternative requires the software to read/write each and every word between main storage and the peripheral equipment. In order for DMA to work, the CPU must be prevented from interfering with main storage at the same time the data transfer is going on. Some microprocessors have special controls that suspend the CPU cycles (and, in fact, remove the device from the data/address bus by disabling three-state output drivers) whenever DMA is in progress. Processors that don't have an inherent DMA ability can be used in DMA applications, but the additional hardware is often complicated.

Arithmetic is almost always performed in two's-complement form on microprocessors. In addition, some processors have special instructions designed for handling BCD numbers. Whether these instructions are important or not depends upon whether the input/output data must be in BCD form. If BCD data is presented to the computer, or required from it, the BCD arithmetic instructions may be required. Remember that if BCD quantities are converted to binary for arithmetic operations and then reconverted to BCD for output, there may be some slight inaccuracies; the quantity 0.1, for example, cannot be represented in binary, so that 0.1 + 0.1 will not yield exactly 0.2.

Vendor Commitment

Whether or not the vendor will continue making the microprocessor long enough to be of use during the applications life is an important decision factor. Furthermore, the manufacturer's commitment to the support of the product line can make life easy or difficult for the system designer. Because of the complexity of these products, there are nearly always idiosyncrasies that are uncovered only after several applications have been completed using the product. Those manufacturers who release and then abandon their products are not likely to follow-up with literature that aids the system designer in avoiding these known pitfalls.

Software support by the vendor is an important measure of the corporation's commitment to servicing the real needs of the user. A primitive assembler, issued on a take-it or leave-it policy, is a definite indication of a lack of corporate commitment to the product line; support could be withdrawn at any time. On the other hand, those vendors that have invested time and money in the development of elaborate programming aids are, in essence, publicly stating their determination.

Minimal software support for a microprocessor should include:

An assembler that allows symbolic operation codes and
symbolic statement labels,
A monitor (often called a debug monitor) that executes on
the microprocessor,
An editor that allows the programmer an easy way to change
source programs for re-assembly, and
A simulator that executes machine code for the microprocessor
on a larger computer or minicomputer.

Without these essential tools, programming costs become virtually uncontrollable. Regardless of the claims of some companies, *no* significant quantities of software are programmed without an assembler or compiler of some kind. There is no efficiency to be gained by not using an assembler.

Additional software that the manufacturer should provide may include:

A compiler for a higher-level language (such as Intel's PL/M, or
Transitron's proposed FORTRAN), and
Libraries of commonly-used subroutines.

Higher-level languages may or may not be useful, depending upon the application environment. In nearly all cases, object code produced by compilers will be slightly less efficient than object code produced by a good programmer using assembly language. In applications with few copies of the program (low production volumes), higher-level languages are most attractive because the cost of writing the original code is a significant cost of the system. In large-volume applications, however, the additional program space required by the object code produced by a compiler may be costly enough to warrant the additional effort of assembly language programming.

Software libraries will become popular, as they have in other computer user communities. Unfortunately, the range of applications for microprocessors may be so broad as to make the majority of packages in the software library relatively useless. Since there is little quality control imposed on actual program correctness by library maintaining agencies such programs must be carefully scrutinized. Unless an entire application can be adopted with little modification, software libraries will probably remain relatively valueless for the next few years.

Not all software needs to come from the microprocessor maker's offices. There are numerous hardware and software design and implementation aids that are being offered by small "secondary" firms. If the quality and cost of these tools are significantly noteworthy, these should be considered as part of the available software support.

Documentation is the most apparent indicator of the manufacturer's commitment to a product. Skimpy, vague and ambiguous "specifications" usually connote a management disinterest in the product line. The quality of documentation can not be measured by volume alone, either. The discerning factor is *completeness*. There should, preferably, be separate documents (or at least separate sections of a common document) for system overview, hardware interface design and programming.

Generally, the degree of acceptance of a microprocessor is proportional to the quality of the documentation.

Application Notes are one particular form of documentation endemic to the semiconductor business; they are not common in the community of computer vendors. However, the engineering problems of designing microprocessors into final systems virtually require the use of application notes. Some microprocessor vendors publish one circuit diagram ("take-it or leave-it") which is inevitably a model of cost-ineffective design. Others publish alternative ways of accomplishing the same objectives, complete with hardware and software considerations (there are very few of these). Finally, some of the vendors seem to think that a brief paper on "How I did X on the Y computer" qualifies as an application note: These are useless.

Application notes will become more popular as second sources begin to appear. These companies, in order to avoid a "me-too" appearance, will probably invest in the development of unique application notes (most likely incorporating some unique components from their general product line). It will be one of the few ways these companies can penetrate a prime-source market.

Design Aids include start-up conveniences like prototyping cards and development systems. In order to avoid a large amount of primitive circuit design and fabrication during the early stages of product development, some kind of prototyping card set must be available. A prototyping card usually contains the microprocessor, master clock and storage/peripheral interface circuitry on a single printed circuit card. A rich set of cards might include RAM and ROM module cards and specialized input/output cards for different classes of peripheral devices.

When complete families of cards are available, the vendor commonly offers at least one version with all the cards packaged into a cabinet, complete with necessary systems software. It is not unreasonable for the user to expect an assembler, editor and debug monitor to be executable on this development configuration.

## Product Longevity

Designing in a microprocessor that won't be on the market when production needs it is not considered to be good form in electrical engineering. Products that are second-sourced or otherwise entrenched are far more likely to be around when needed.

Second sources are beginning to appear; MIL supplies Intel's 8008; AMI is committed to supply Motorola's 6800. Because of the complexity of LSI, second sources will have to be developed with the cooperation of the prime source. The second source no longer has the luxury of simply "steaming" open a competitor's package to see how to duplicate it.

Standardization as an industry-wide event, is not likely to happen in the microprocessor field. To formally standardize on microprocessor capabilities is to standardize the class of favored applications. On the other hand, de facto standards already exist . Certainly, when a competitor refers to his own product as "Intel 8080-like," a standard exists. The 8080 is clearly today's standard in microprocessors. However, the dam of pent-up microprocessor releases from other vendors is about to break.

## Price and Availability

Prices for microprocessors are deceptive. First, and most important, main storage RAM and ROM costs are typically five to ten times the CPU's costs, so that the microprocessor cost is insignificant when compared to the whole system cost. Secondly, a practice known as "kit pricing" is popular in the microprocessor marketplace (as a direct consequence of the storage costs).

Most microprocessor vendors offer a kit price encompassing all the parts required in a particular application. The individual component prices, in these volume quotes, are not made available.

In essence, the vendor is reducing the microprocessor costs in order to acquire the memory business. This is sometimes called "giving the microprocessor away."

Availability is an important measure of the suitability of a microprocessor. More important than simple dates, however, is the history of performance of the vendor. Some microprocessors make it to the sample stage, but production proves more diffiuclt. A processor can be "available" or available. By "available" we mean that samples can be obtained and used, and production is scheduled for the future. By the time the product becomes available, on the other hand, many users have been through the initial learning phase, all the anomolies are known and evidence of production capability exists.

## HARDWARE DESIGN CONSIDERATIONS

Microprocessors require other ancillary parts to make entire application systems work. As an absolute minimum, a microprocessor needs a program store (although the Burroughs Mini-D has it on the CPU chip). Some microprocessors are easy to interface to storage--and others are hard. To the extent that the vendor has put all of the necessary interfacing circuitry on the CPU, memory and input/output parts, a family exists.

### Completeness

Completeness of the parts family is an important selection parameter. Some vendors seem to feel that if the processor can be connected to a RAM, regardless of the number of additional chips involved, then their RAM is a member of the parts family. Other semiconductor manufacturers have taken special pains to be sure that storage and processor and input/output parts can all fit together without intervening logic; the latter systems are considered to have complete parts sets.

### Master Clocks

Clocks are required for most microprocessors, although some of the newer processors only require a frequency-controlling two-terminal device (crystal or R-C net). Some of the vendors offer clock or clock-driver chips. In many designs, the DIP-socket sized crystal oscillators such as those from Motorola or Vectron are used. In others, crystal-controlled stability is not important.

Clock frequency has little to do with relative data manipulation speed, and shouldn't be used as a selection criterion. The number of phases, however, is important; four phases are harder to generate than one or two. In clock schemes with multiple phases (particulary common in MOS processors), the requirements for overlapped or closely-controlled relative rise and fall periods should be investigated. Sometimes four phases are easier to make than overlapped, synchronized two-phase clock signals.

### Power

Voltages and power dissipation are considerations in many systems. Many of the MOS processors require two supplies (usually +5 and -9 or -10) to be TTL compatible; if that compatibility is unnecessary, these can be operated from a single supply. Power dissipation is most often a function of the intended operating speed range.

Compatibility

Electrical compatibility with other logic circuitry is required in most applications. Most of the microprocessors offer some degree of TTL compatibility but read the fine print carefully. Numerous variations in input/output logic levels are common--even on one chip. MOS-to-TTL conversion is "delicate" at best and the manufacturer's recommendations should be followed religiously. TTL's comparitively high input current requirements frequently demand that buffers be added between MOS and TTL portions of mixed-logic systems. The speed of MOS circuitry frequently degrades as more loads are paralleled, requiring the addition of buffers even within portions of MOS-only systems.

Packaging

Package size is important to many system designers, particulary those with cramped layouts. In general, package sizes with small numbers of pins are easier to physically install into a system, while microprocessors in packages with large numbers of pins are easier to interface to.

SOFTWARE DESIGN CONSIDERATIONS

After the system is designed, and the hardware has taken shape, someone has to program the microprocessor. What the programmer sees is largely determined by the architecture of the computer and the environment in which that native architecture operates.

Word Size

This is nearly always the first parameter specified during microprocessor selection, even though it should be the last. Word size affects the microscopic efficiency of some common kinds of software operations but it seldom affects the overall throughput of the design. In many applications, four-bit processors out-perform similar designs on eight-bit processors, although the opposite is also true.

In general, ease of programming is inversely proportional to:

1) Word size, because smaller data word sizes require multi-word operations on practical data quantities, and
2) The number of different register sizes in a computer, because simple register-to-register data transfers are limited.

Computers with small word sizes (four-bit; and to a lesser extent eight-bit) always require a different register size for storing some addresses. For example, the program location counter is often made an integer number of native words in size. When the word size is small, the manipulation of storage addresses and program addresses is complicated. Remember, the ease of programming, not the overall application efficiency, is what is influenced by word size.

Throughout the history of the development of computers there have been no four-bit word computers, and only a few eight-bit words, until the advent of the microprocessor. Until the limitations of restricted semiconductor real estate were imposed, there was little economic incentive to develop small-word computers. Difficulty of programming may, in the future, make the smaller word sizes appropriate only for special high-volume and extremely cost-sensitive applications.

Addressing Capacity

How large a program can be written without resorting to special external hardware and internal software techniques is defined by the program addressing range. The ideal processor for an application has neither too small nor too large a capacity. A too small range means that extra hardware will be required to extend the addressing. On the other hand, excessive capacity means that extraneous address bits will be carried in every instruction that refers to storage; those bits cost money.

Addition Time

Register-to-register addition time is a popular estimate of the computing speed of a computer. This instruction is chosen as a selection factor because nearly every computer has an add instruction. Microprocessors with more than one programmer-accessible register for data manipulation on the CPU chip can usually perform a fast register-addition in a minimum instruction execution time. Some processors, however, are organized as one-accumulator computers so that register-to-register additions are not provided; in this case, the addition of the contents of an arbitrary storage location to the accumulator is often scored as the minimum addition time. The incrementing of a register by one is not considered a good test of addition time.

Addition time should not be the only criterion used in timing estimation. Computer makers have been known to treat that one instruction uniquely so that the machine appears to the casual observer to be faster than it really is.

Register Complement

The number of registers in the microprocessor is probably the most important feature of its architecture. Different registers in a computer have different uses; the ways that they may be used are embodied in the instruction set.

The most precious resource that the programmer can allocate in software is the set of CPU registers. The more registers there are, the less likelihood there is of main storage references. Generally, references to main storage are more expensive in time than the references to on-chip registers.

Arithmetic (or ALU) registers are those on which ALU functions can be performed; the register can be a source or destination of operands for the operation. Registers that can supply but not receive operands for the ALU are not considered arithmetic registers.

An index register is a programmer-accessible register that is implicitly included in certain references to main storage. Unless the contents of the register can be added to another value (from, say, the instruction itself) during the storage addressing cycle, it is not an index register.

All other programmer-accessible data registers in a microprocessor (excluding the ALU and index registers) are called general purpose or scratch-pad registers. (In some large-scale computers a general-purpose register can be used in any of these three modes; that practice is not popular in microprocessors.)

Return Stack

Return addresses are usually handled through the medium of a push-down stack in microprocessors. Except for the Intersil 6100, which emulates the DEC PDP-8, microprocessors put subroutine return addresses into a push-down stack in a read-write memory; that memory may be on the CPU chip

or in external main storage. Return addresses are not saved within the program storage area as in many minicomputers because most programs for microprocessors are stored in read-only memory.

When the return addresses are stored in an on-chip push-down stack, there is some natural limit to the number of dynamic subroutine calls. If there are eight stack positions, then generally only seven subroutine calls may be active at one time; if interrupts are anticipated, the real used stack size must be kept smaller to allow some stack depth for the interrupt service routine.

When return addresses are stored in RAM, an on-chip stack pointer is maintained in the CPU. When a subroutine is called, the return address is pushed into the RAM stack, and the pointer is updated.

The on-chip stack allows faster subroutine calls because RAM doesn't have to be accessed. Stacks in RAM are of potentially huge depth, and this allows certain kinds of algorithms to be easily programmed. If the on-chip stack is accessible to the programmer, the depth of that stack can be extended by software. Most on-chip stacks are not accessible, and this imposes a rigid limit on the allowed depth of subroutine calling.

Addressing Modes

The addressing of data and program segments in storage is an area that presents enormous variations from computer to computer. The problem is to allow references to be made to any arbitrary word in the addressing space, yet eliminate the need for a full set of address bits in each and every instruction. If the full address is required in each instruction, program sizes tend to grow. Most microprocessors take advantage of the fact that most data and program references are local in scope; that is, references are most often close to the address of the current instruction being executed or close to the last datum referenced in main storage.

Each microprocessor vendor seems to adopt a unique vocabulary for describing machines. Sometimes, the intent is obviously to allude to features by name that don't really exist. In other cases, the reason is ignorance. In the descriptions of addressing modes that follow, most of the popular techniques will be covered using industry-standard terminology. In comparing computers, compare the addressing modes by what they do, not by what the documentation calls them.

Direct addresses are the most common kind, and are represented by instructions that contain a storage address wide enough to refer to any point in main storage.

Abbreviated addresses are similar to direct addresses but only part of the address is carried in the instruction. The rest of the bits of the address are contained in a register that is usually part of the CPU. Most commonly, the address part that is supplied from the instruction represents the least-significant bits of the intended address.

Immediate addressing means that the instruction doesn't have an address part at all; the program location counter is issued to storage as the datum address. The datum fetched, then, is part of the instruction itself. Certain instructions like jump and call nearly always have immediate operands. Many microprocessors have other data processing instructions that admit immediate operands.

Relative addressing shortens the address part of the instruction by permitting references within some narrow range relative to a CPU register. Most often that register is the program location counter. The address field of the instruction is added to the program location counter's value (in some cases with sign-extension) to arrive at a datum address. In larger computers, the base-displacement form of addressing is used; this relative form uses a special (base) register plus the displacement carried as an abbreviated address in the instruction to compute a storage address.

Indexed addressing is similar to relative addressing but a special register is used. To compute a datum address, the address part of the instruction is added to the the contents of the index register by the CPU.

Indirect addresses allow a named location to contain, instead of data, the address of data. Therefore, the instruction refers to a word or words that, in turn, refers to the datum. This is particularly important in generalized software. In some computers, one of the bits of the address that has been fetched indirectly may further specify indirection. This multi-level indirect addressing is intellectually stimulating, but seldom useful.

Register addressing is used when the CPU has several possible registers that can be referred to. Since the number of register is relatively small, only a few bits are necessary to select a register.

Register-indirect addressing combines the last two addressing schemes. The instruction specifies a register address and indirection. The selected register contents are used to address main storage; this keeps the number of address bits in each instruction very small.

The register-indirect form of data addressing is so common in microprocessors that it deserves special mention. Many of the four- and eight-bit processors offer only this addressing mode; therefore, all the data addresses must be in CPU registers. This frequently makes programming difficult, particularly when making references to isolated words in storage. When accessing contiguous elements of an array the register-indirect form is efficient. Unfortunately, most actual programs don't spend much time cycling through array elements.

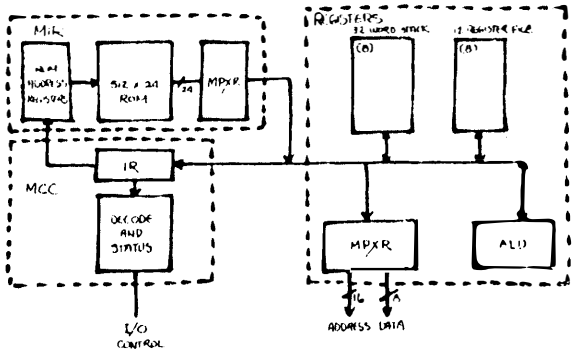
As this last example has illustrated, it is possible to combine two or more of these addressing schemes together into more elaborate combinations. It is quite common, for example, for microprocessors to provide for abbreviated but indexed addresses.

AND NOW...

The processors:

## AMI 7200

The American Micro-Systems 7200 microprocessor represents the most sophisticated and comprehensive design introduced until this year. Unfortunately, AMI apparently couldn't build it. As an object lesson in architectural ambition, however, it serves as a good model. It also serves as a warning to those who design around "paper tigers."




The 7200 was designed to be implemented on three chips with a microprogrammable eight- or sixteen-bit data word structure. Microinstructions were to be held in a 512-word, 24-bit ROM with most control logic on a separate chip. An eight-bit register and ALU (RALU) chip was to be cascaded into a sixteen-bit version. The RALU's 12-word register file was to have held two accumulators, four index registers, two program counters and four temporary registers. Because pairs of important registers were designed in, interrupt servicing would have been simplified.

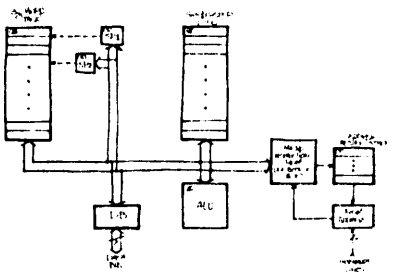
The control logic called for time slots of 90 nanoseconds, and a 540 nanosecond microinstruction execution time. This was apparently the Achille's heel of the design, because it stretched the capabilities of PMOS technology.

The MIR (Microinstruction ROM) was to have held the microprogram for the particular eight- or sixteen-bit instruction/data configuration. The instruction register was designed to be part of the Microcontrol chip (MCC), and nine bits it were to be used as an address into the 24-bit microcode ROM. In addition to the instruction register, the MCC held all master control functions and instruction decoding that was dependent upon external status conditions, and an interval timer.

The Register and Arithmetic Logic Unit was designed with a 32-word, eight-bit push-down stack for data and subroutine return addresses, and a file of twelve registers. The register file was probably designed as a fifteen or sixteen word eight-bit array, with several of the register pairs treated as single sixteen-bit registers by the programmer. ALU operations were on eight-bit quantities, and addresses were designed to be sixteen bits wide.

## AMI 7300

The AMI 7300 will, at first glance, appear to be similar in architecture to other "slice" architectures. Such is not the case. The 7300 was (was, because like its predecessor, the 7200, the 7300 has been withdrawn) composed of a control chip and a register chip, and it had a 22-bit wide microinstruction. There, however, the similarity with other architectures ends. One chip held the microinstruction ROM and basic controls, while the other contained the registers and the ALU. The data bus was arranged for eight bits, and the inter-chip bus was eleven bits wide. (The micro-instruction was passed to the ALU in two phases.)




The design of the RALU chip included a thirty-two byte array that, with two pointers, could be used as one or two push-down stacks. There were sixteen general purpose, eight-bit wide registers, and the ALU was to operate on eight-bit quantities from registers, the stack, or the data bus to the rest of the system (including main storage and peripherals).

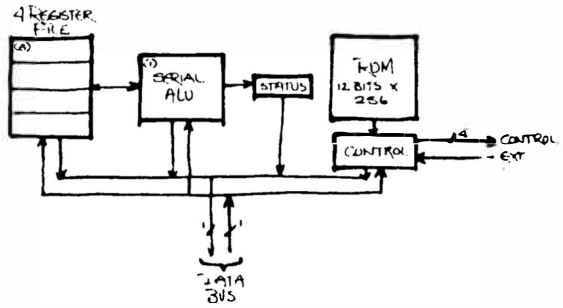
The MIR (Microinstruction ROM) was designed to be a 512-word, 22-bit mask-programmable ROM. The ROM address was held in a nine-bit register, and attached to a seven-level push-down stack so that micro-subroutines could be easily created. External control and interrupt lines fed a starting-address register for interrupt recognition and input-output control lines came from the MIR.

In the 7300 design, each microinstruction required four microseconds for execution, and execution of one microinstruction was to be overlapped with the fetch of the next. There were no intrinsic features for peripheral device or main storage control.

## BURROUGHS MINI-D

The Burroughs Mini-D is a unique departure from contemporary design. It was obviously conceived quite some time ago, and was designed to minimize on-chip real estate. All data flow in and out of the chip goes over a one-bit bus; internally, however, the Mini-D is organized as an eight-bit processor. All instructions are contained on a ROM located on the CPU chip itself; the ROM may contain 256 twelve-bit instructions. Reference to data storage is, like access to peripherals, through the sole one-bit wide data bus. While the Mini-D is microprogrammable, most applications are probably programmed directly into the ROM. Since the ROM is part of the processor chip and must be custom mask-programmed, it is practical only for very large volumes such as those found in consumer-credit terminal applications.

The internal organization of the processor provides four general-purpose eight-bit registers and a serial arithmetic logic unit. The program location counter need be only eight bits long to address the entire program ROM. The instruction set is quite obscure, having been designed for maximum flexibility in only twelve bits. It will be quite familiar to anyone who has programmed a narrow-word microprogrammable computer before.

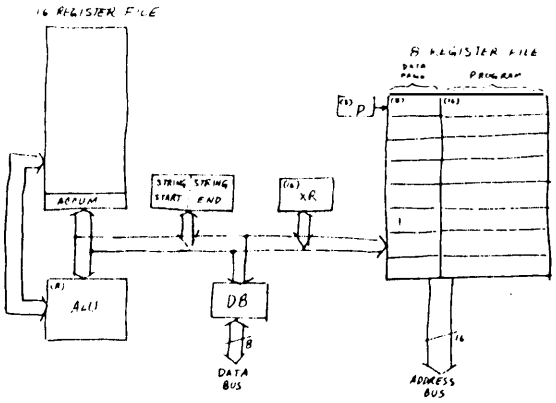



There have been persistent rumors that the semiconductor house that makes the Mini-D for Burroughs is negotiating for sales rights. However, it appears that Burroughs will not permit any marketing of the product by any third party. The Mini-D would probably be especially attractive to the consumer and entertainment markets.

The Mini-D is a good model to study because it is likely that custom processors for extremely high volume applications will be, in some ways, similar. Intel, for instance, has designed and built a custom special-purpose CPU chip with an electrically re-programmable ROM on-board.

## ELECTRONIC ARRAYS

The Electronic Arrays' microprocessor (as yet unnamed) is an eight-bit computer with sixteen data registers and a unique kind of stack. Only a small amount of data has appeared in print about the processor; since it is a year until samples will be seen, many things could change.




The programmer has use of sixteen general-purpose eight-bit registers, one sixteen-bit index register, and an eight-place stack that is unique among microprocessors. One of the sixteen data registers is the accumulator, and it alone may be shifted; it is probably treated as a special register by the user.

The uniquely organized push-down stack has eight levels; a three-bit pointer (P) selects the current entry. Subroutine calls are made by incrementing the P-register, and returns decrement it. The particular stack item that is indexed by the P-register holds the current program location counter and a data page pointer.

The EA microprocessor's storage addressing scheme arbitrarily divides a sixteen-bit address into two eight-bit halves. The most-significant bits select one 256-byte page, and the least-significant eight bits select one byte of that page. In the currently-selected push-down stack item, sixteen bits hold the program location counter (page and byte). The extra eight-bit entry in each stack item, called *data page*, is used to select a default page at which data is stored; it typically points to locations in RAM.

Instructions that refer to data in main storage may have eight- or sixteen-bit addresses; that is, these instructions are either two or three bytes long. The three-byte instructions have a one-byte operation code and a two byte, sixteen-bit address. The shorter instructions use the currently selected data page register contents to select a page, and the instruction supplies the byte address within that page. In other words, the eight bits from the stack are "laminated" with the eight bits from the instruction to form a full sixteen-bit datum address.

This is one of the neater solutions to the addressing problem, allowing short addresses but with a variable range. It is fairly easy to arrange the bulk of data for a suite of subroutines in a 256-byte area; contrary to popular opinion among some designers, data is *not* generally accessed from contiguous locations.

The EA microprocessor is the first attempt to provide an array processor. A pair of four-bit registers (string-start and string-end) are used to select a sequence of one to sixteen registers to be treated as an array. From the sparse information available now, it appears that an array of up to sixteen bytes in memory and an array in the on-chip registers may be added, subtracted, AND-ed, OR-ed and XOR-ed, with results sent to the register array. The registers can be loaded and stored as arrays, too, which should simplify interrupt servicing.

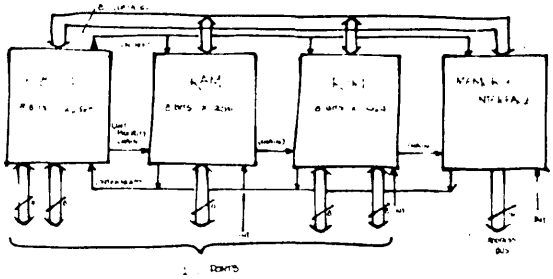
The pointer to the stack can be program or hardware controlled, so vectored interrupts via the stack are possible. The stack is said to be accessible, so software can be used to extend its depth.

Caution should be exercised because the EA micro is not yet available. When it is, however, it promises to permit extremely small programs for medium-size applications.

## FAIRCHILD F-8

Fairchild Semiconductor's new F-8 microprocessor is a unique architecture; the CPU has no program location counter. Much more logic is packed on the CPU chip itself than in other micros and some other important registers are duplicated on each and every memory chip in the set.

The F-8 Family consists of a CPU and a RAM chip, a mask-programmed ROM chip, and a general memory interface chip. The RAM and ROM chips also provide input-output paths at chip's edge. The 1024x8 ROM provides two bi-directional eight-bit channels and the 256x8 RAM has one. The addresses are programmed into ROM I/O ports when the mask is cut.

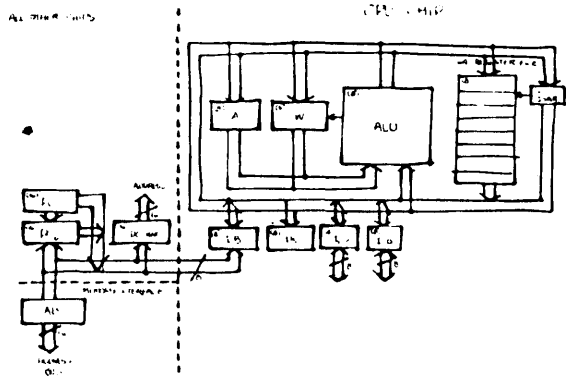


Each memory chip accepts one interrupt signal (therefore, there may be as many different interrupts as there are memory chips) and each chip connects to two neighbors in order to accomplish chaining of interrupt priorities. Each chip also has a timer that can be programmed to interrupt; timers count in 15.5 microsecond increments.

The most important feature of the memory chips is that they each have two program location counters and a data address register; like registers in all chips all contain the same value. There are two program location counters on each chip so that interrupts and single-level subroutine calls can be performed quickly. All reference to data in RAM and ROM is made through the sixteen-bit data counter (memory address register) on the memory chips. Each of the three registers may be read into predetermined registers in the CPU's register file, modified and sent back out to the storage chips.

One major disadvantage of the F-8 storage addressing scheme is the required use of the data counter. There are no instructions that explicitly refer to storage addresses, so that the global data counter must be loaded before referring to any data in RAM or ROM. This has been a serious programming disadvantage with the Intel 8008 and was rectified in the 8080 by adding instructions that have explicit storage addresses. Without these instructions, random references to main storage are complicated. For small applications, and those possessing highly regularized and structured arrays, the F-8 CPU 64-word register file may alleviate the storage addressing problem.

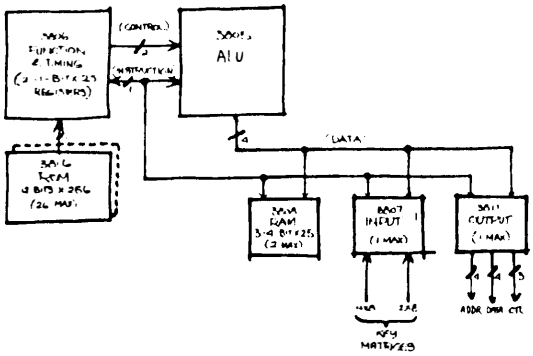
The CPU also has two bi-directional input-output ports and a master clock on the single chip. Duplicates of the program location counter(s) and storage address registers are *not* on the CPU chip. The master clock is controlled by a crystal or R-C network connected to two pins; the maximum frequency is 2 MHz. All this logic in one package means that some small applications can be designed with only one more chip (a ROM holding the program). The large register file should allow many applications to get along without any external RAM.




Of the CPU registers, W holds ALU and interrupt status, A is the accumulator, and ISAR is an indirect scratch-pad address register. The ISAR points to one of the 64 registers in the file; that register, and the lowest-numbered twelve registers, are accessible from instructions. Registers 9 through 15 have access to the W-register status bits and the external program counters.

## FAIRCHILD PPS-25

The most obscure microprocessor on the market is undoubtedly the Fairchild PPS-25. Actually, the device is not nearly as complicated as the available documentation appears to make it.

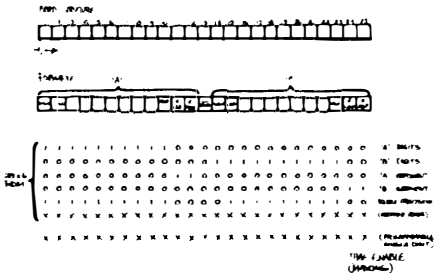



The PPS-25 chip set includes a two-chip CPU and ROM, RAM and I/O chips. However, internally the PPS-25 is different from other processors (except the ill-fated MAPS from National). To begin with, the RAM module is organized as three twenty-five digit registers. Now, this seems to be an odd way to organize storage for a microprocessor, but it is most appropriate for numeric processors such as are found in navigation instruments and calculators. However, few applications really require twenty-five digits of precision. In fact, the PPS-25 design allows the different parts of a twenty-five digit register to be used in different ways.

The use of the parts of a twenty-five digit register is the most confusing part of the system, and the literature is less than lucid. The so-called "time enable" notion is the culprit; it is so poorly explained that most potential customers have simply dropped any consideration of the product. Perhaps a clearer explanation will help:

Data travels between RAM and CPU on a four-bit bus. During a basic 62.5 microsecond machine cycle, there are 25 mini-cycles of 2.5 microseconds each. During each mini-cycle, a new digit is presented to the CPU from the selected register in the RAM, and an old digit is replaced. Therefore, the twenty-five digits are presented in twenty-five time-sequential "nibbles."

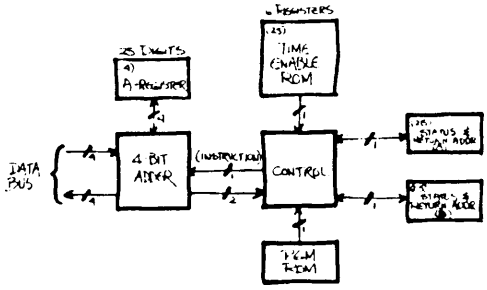
Inside the PPS-25's CPU, there is a 6x25 bit ROM that represents six masks. Each mask is 25-bits long--the same as the number of digits in a RAM register. Each bit in the mask specifies whether the corresponding digit from the RAM register is to be operated upon by the current instruction. In most cases, the mask is all zeros except for a "burst" of one bits defining that part of the data to be used. The example here shows how the 6x25 ROM might be specified for an application that uses nine-digit floating point quantities. Two of the masks select the actual digits of significance (left and right half, so two numbers can be packed into a single 25-digit register), and two masks select the exponent digits. For double precision operations, another mask is specified that allows eighteen decimal digits of precision. The sixth mask is not specified in this example. For manipulation of all data not explicitly covered by a ROM pattern, the PPS-25 provides a single register that can be programmed to hold a number in the range 0 to 24, so that any single digit can be selected and operated upon; this would be used to access and operate on the sign words, for example.



Each instruction has a three-bit code that specifies one of the six masks that defines the data format. The seventh code causes the selection of one specific digit out of the 25 (based on a register that is under the programmer's control). (The eighth code--zero--is used to mark those instructions that do not refer to data in RAM registers.)

Summarizing, then, an instruction that manipulates data specifies one of six masks, the bits of which "enable" certain digits to be processed by the instruction. "Time enable" is just a complicated way of saying "AND-mask."

Once the serial four-bit wide data path and how it is controlled by the "time enable" is understood, the only remaining idiosyncrasy is the instruction fetching sequence. Once that is explained, the rest of the architecture is seen to be conventional.




The instruction fetch uses the same twenty-five sub-cycles of 2.5 microseconds each. During the execution of one instruction, the next instruction is being fetched. The twenty-five time slots used during data operations on RAM are simultaneously used to send an address to ROM and then fetch an instruction. This transmission takes place on a one-bit wide path. During the fourth through twelfth time slots, a ROM instruction address is sent out; the least-significant bit is sent first, and the last bit is always zero (this ninth bit was included in hopes that there might be a 512-word instruction ROM someday). During the fourteenth through twenty-fifth time slots, the instruction is serially transmitted from the ROM to the CPU.

Fairchild is de-emphasizing the marketing of the PPS-25 in favor of the newer F-8.

## GI CP-1600

Except to a few privileged customers, not much has been published about General Instruments' CP-1600. From information that is beginning to "leak," we have learned that it is a sixteen-bit processor built into a 40-pin package with a fast MOS technology. The architecture includes an eight register file, but it is probable that some of those registers are used for program counter and storage control. All data and addresses share a single sixteen-bit data bus, and there are controls that provide for DMA ability and interrupts.

The processor has apparently been delivered to the customer for whom it was originally designed. GI isn't telling whether it will be sold publicly.

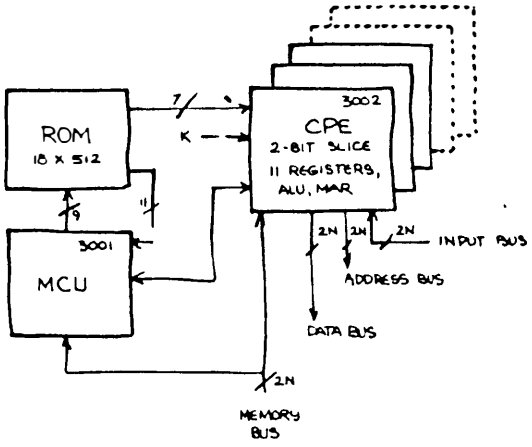
## INSELEK

Inselek is rumored to be working on a CMOS-On-Sapphire eight-bit slice machine with a cycle time of around 300 nanoseconds.

## INTEL 3001/2

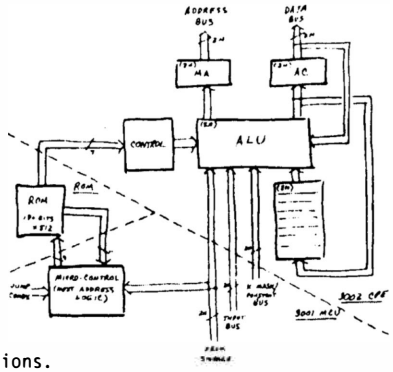
Intel's new chip set ushers in a new era of performance in microprocessors. The 3000 family is a microprogrammable processor implemented in two-bit slices. The Schottky TTL circuitry provides a "typical" microinstruction cycle time of 160 nanoseconds. The architecture and microinstruction set has provisions for adding optional hardware to do almost anything, including going even faster. The product is aimed at replacing TTL controllers and processors and Intel professes no plans to introduce a general purpose macroinstruction set, even one like the immensely popular 8080.

The microprogram is stored in conventional bipolar ROM's, and RAM's can be easily substituted during development. The 40-pin 3001 Microprogram Control Unit (MCU) generates the ROM address sequences by analyzing the current microinstruction, status flags and up to eight bits of the macroinstruction fetched from main storage. Extensive control logic includes four- and sixteen-way jumps that are steered by macroinstruction bits. The MCU also stores the ALU flag bits and selectively re-inserts them on succeeding cycles to implement some functions.



The MCU has no provisions for saving or restoring the current microinstruction address. This weakness eliminates micro-level subroutines from the microprogrammer's arsenal; that's a serious inconvenience. Attempts to save the address externally will be hampered by the fact that only eight of the nine address bits can be loaded back into the chip. There are ways to implement micro-subroutines with external hardware, but they all appear to be awkward.

The business end of the computer is built up with 28-pin 3002 Central Processing Elements (CPE). Each CPE contains two bits of ALU, memory address register, an accumulator and eleven general data registers. Carry propagate pins are provided for both ripple and look-ahead carry strategies; a compatible eight-bit look-ahead generator has been announced as part of the series. All paralleled inputs are specially buffered to permit a vast number of these chips to be stacked together (anybody need a 320-bit word?).




The CPE architecture is conceptually simple and is based on an ALU with three-way multiplexers on each of its data inputs. All chip inputs must pass through the ALU enroute to the accumulator (AC) or register file. A Memory Address (MA) register is provided; memory and output data come directly from the AC. The eleven scratchpad registers are organized as one temporary holding register and ten general purpose registers. The majority of the microinstructions perform functions on the accumulator and the holding register in the scratchpad file; this allows more function-control codes in these instructions.

An unusual feature of the CPE is the K input bus. On the way to the ALU, the accumulator and input bus are AND-ed with the K bus which supplies a bit mask. The K bus can also feed micro-programmed constants directly into the ALU. A basic eighteen-bit wide microprogram word must be extended, of course, to provide the desired number of K bits for generalized masking. Functions such as byte masking or swapping can be implemented with only a couple of K bits tied to the appropriate CPE pins.

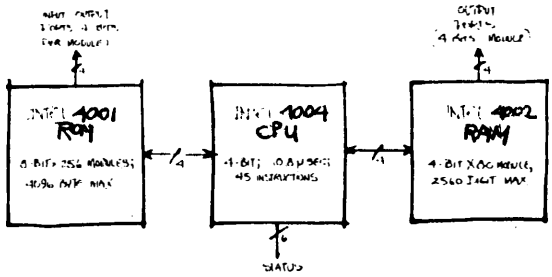
Each microinstruction moves data from a specified source bus or register to a selected destination through the ALU. The microinstruction includes ALU function code, status flag control bits and conditional jump fields and is executed in a single clock cycle.

The announced chip set includes a priority interrupt control unit which can jam an address into ROM. The interrupt does not affect the microprogram address register and the microprogram is written so that interrupts are recognized only at the time a macroinstruction fetch cycle is about to be initiated.

Microinstruction fetch and execute delays can be overlapped by providing latches between the ROM and CPE's. Details on this technique are not yet available but Intel claims that it improves the "typical" cycle time to 120 nanoseconds. Worst-case times were not established as we went to press. Microprocessors are no longer slow.

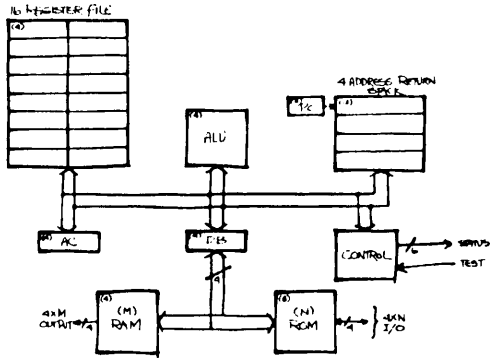
## INTEL 4004

Intel's 4004 is the original microprocessor. It is a member of the MCS-4 family of parts based on a four-bit word. The addressing structure is that of a "Harvard" class computer; data and program are not in the same storage medium or addressing system. Because of the unique RAM, the addressing arrangement and the input/output port locations, the MCS-4 is sometimes difficult for




neophytes to understand. However, once the Rubicon has been crossed, the MCS-4 will be seen to be a powerful and quite complete chip set.

The MCS-4 set consists of chips that perform CPU functions, handle RAM and ROM storage (and input/output, coincidentally, on the same chips), and can be interfaced to a variety of main storage media. Recently added chips can handle generalized input and generate master clock signals. All inter-chip communication is through a single four-bit bus. While this allows all parts to be designed into 16-pin DIP's, it does slow the system down. Instruction execution takes 10.8 or 21.6 microseconds.




The microprocessor has a sixteen-register file of general purpose four-bit digits, in addition to a four-bit accumulator. Return addresses are stored into a four-place push-down stack, the selected item of which is the current program location counter. With a twelve bit location counter, the 4004 may address up to 4096 eight-bit words of program.

Since instructions cannot refer to data with explicit direct addresses, only register-indirect addressing is provided. Generally, an entire sequence of instructions is required to select a stored datum; careful allocation of data storage can make addressing less difficult.

Interrupts are not provided in the 4004. One unique pin at the edge of the chip (called TEST) is available to the conditional jump instructions to sense important external events. All input/output is performed through ports located in storage chips or accessed through storage interfacing chips. When the software is mask-programmed into type 4001 ROM chips, the four input/output bits can each be programmed for either input or output. The RAM modules provide only output ports.

The MCS-4's head start on the industry and its 16 pin packages will make it an economical choice for some time to come.

## INTEL 4040

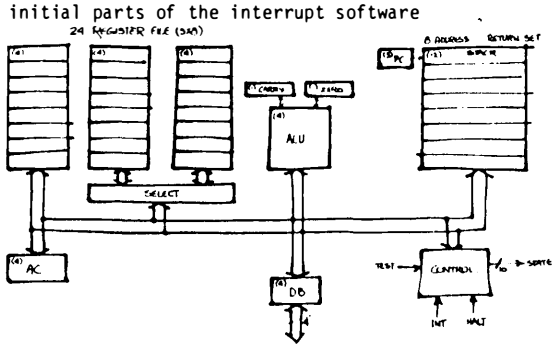
Just as Intel's experience with the 8008 led them to introduce an improved model (the 8080), their experience with the original microprocessor, the 4004, has produced specifications for the 4040. Most of the 4004's ills have been cured by the 4040. In a departure from the MCS-4 16-pin scheme, the 4040 has been packaged into a 24-pin DIP.

The 4040 can be best examined in relation to the 4004:

| Feature | 4004 | 4040 |
|---|---|---|
| Program storage space (bytes) | 4096 | 8192 |
| Register file (four-bit digits) | 16 | 24 |
| Logic instructions (AND, OR) | no | yes |
| Interrupt | no | yes |
| Halt instruction | no | yes |
| Return address stack | 4 x 12 | 8 x 12 |

Communication between the two program banks is enhanced by the eight additional registers added to the 4040. However, use of subroutines in one bank by programs in the other is a software problem, because the longer push-down stack is still only twelve bits wide. Special inter-bank jump and call routines must be used to avoid losing the thirteenth bit of the return address.

Interrupts are handled by creating a subroutine call to a fixed twelve-bit address, but the bank selection is not changed. Intel suggests duplicating the initial parts of the interrupt software in each bank of program storage.



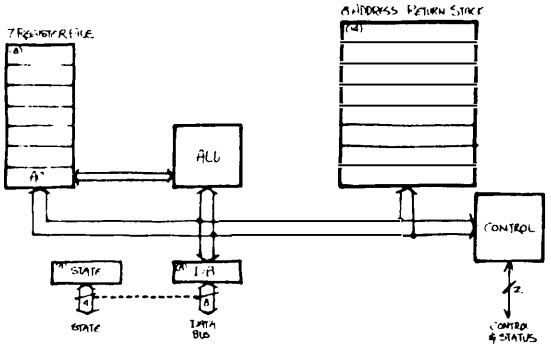

The larger register files are necessary to support the kind of programs likely to be seen by the 4040. The return address stack has been lengthened to eight places, allowing up to seven subroutine levels. Usually, the programmer should leave one unused level so that interrupts can be serviced.

The data register file has been divided into two parts. Registers 0 through 7 are now duplicated; which register of the pair is selected is dependent upon the most recent register-bank select instruction issued. Registers 8 through 15 are not shared. Usually, the programmer will preserve one set of the 0-7 registers for use with subroutines in the first half of ROM, and the other set will be selected just prior to selecting the second half of ROM.

A new part will appear with the 4040. It is an integrated replacement for the 4008/4009 memory interface pair that are used for interfacing to conventional storage chips. Other new parts include a ROM with four 4-bit input/output ports and a clock generator chip. A general purpose I/O chip for the system is being developed.

## INTEL 8008

The 8008 is an eight-bit microprocessor in a single eighteen-pin package. Two eight-bit numbers can be added in 20 microseconds; there is a grade-out, known as the 8008-1, that can add in 12.5 microseconds.




Internally, the CPU is organized as a seven-register computer. One of those registers is the accumulator; nearly all arithmetic operation use it, as well as all input/output. Because the 8008 does not have instructions that have direct addresses, two of the CPU registers are used for all references to main storage. In order to fetch an operand from storage, the program must load an address into two (specific) registers of the file. This means, then, that at least three instructions must be used to refer to arbitrarily placed data. Furthermore, some common operations (such as moving data from one place in storage to another) are uncommonly complex.

Because of the small number of pin-outs, the 8008 requires that all data and addresses go through a single bi-directional eight-bit bus. Addresses, up to fourteen bits long, require two sequential clock periods, and off-chip latches are required to hold the result.

The 8008 has an interrupt mechanism that is a model of simplicity. When the interrupt signal is raised, the processor inhibits incrementing of the program counter, issues an acknowledgement, and proceeds through a normal instruction fetch. The external interrupt hardware is expected to "jam" in a

substitute instruction, usually a subroutine call that simultaneously saves the program counter and invokes the interrupt service routine.

When an interrupt strikes, it is always necessary to record something. In most computers, interrupts cause the CPU registers to be saved in main storage so they can be used for recording the fact of the interrupt, and any essential processing can be done immediately. In the 8008, the register pair used for storage addressing may already contain values established by the interrupted program, so they cannot be disturbed. The system designer can choose to reserve two of the general purpose registers as interrupt registers (into which the storage addressing registers can be placed during interrupt processing), but this steals two of the seven registers available. In most cases, users avoid the interrupt mechanism except to start the processor at power-up.

A major disadvantage of the 8008 is the addressing scheme that requires programming addresses into a specific register pair. The interrupt handling problem is but one example of the side-effects of that choice. Fortunately, the problem has been cured in the 8080.

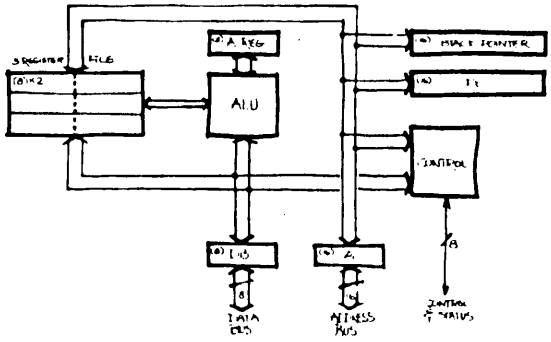
A two-phase clock generator chip is being added to the 8008 family of parts. Many of the new parts for the newer 8080 will work with the 8008. Another component is rumored to be in design; it will replace the approximately twenty IC's necessary to interface the 8008 to storage and input/output.

## INTEL 8080

Intel's 8080 is an out-growth of their experience with the 8008. Because of some of the idiosyncrasies mentioned in the discussion of that older product, the 8080 was developed. Architecturally, the 8080 is significantly different from its predecessor. However, by design, there is major software compatibility; 8008 users have had little trouble upgrading to the 8080.

The most significant change between the 8008 and 8080 is the change from 18- to 40-pin packaging. This allows many formerly multiplexed data paths to be separated out and handled in parallel. This, as well as the change from PMOS to NMOS, has allowed a ten-fold increase in processor speed. Furthermore, most of the external logic required to support the older processor is now on the 8080 CPU chip, and all the important interfacing signals appear on designated pins.

Architecturally, the 8080 has a three-register sixteen-bit file and an accumulator. For compatibility with the 8008, many instructions treat these as seven separate eight-bit registers. The sixteen-bit stack pointer is used to place all return addresses in RAM. Of course, this means the program counter has to be assigned a unique on-chip register. Note also that the storage addresses now extend over sixteen bits, and appear on a separate bus from the data and instruction bytes.




The stack, in RAM, can also be used to store data. It is of unlimited depth (to the limits of storage), although execution times for stack-referencing call instructions are slightly longer. Generality always exacts a toll in efficiency.

New instructions have been added to the set. The 8080 has instructions that permit explicit addressing of storage locations. Furthermore, where the 8008 only had a single pair of eight-bit registers allowed to address storage, the 8080 permits any of the three main registers to hold and output an address when using register-indirect addressing. This simplifies interrupt handling, too.

The push-down stack can be used to hold the status bits and important register contents for interrupt servicing, avoiding another problem that was well-known on the 8008.

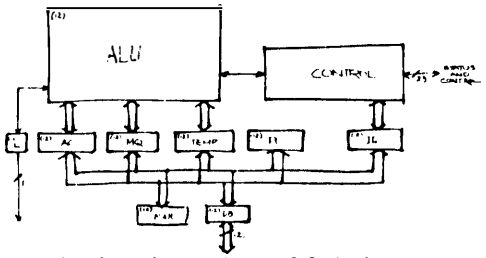
The 8080 has two potential disadvantages: From a software standpoint, the lack of indexed addressing is serious in some applications; from a hardware view, the need for a third power supply must be considered.

In one recent study, the 8080 was a mere $20 more expensive than the 8008 when overall system costs were calculated. As industry marketing pressure mounts, that gap will probably be reduced until the 8080 is cheaper to use than the 8008.

With the recent introduction of a whole family of support parts for the 8080, Intel has opened the way toward the design of small systems with minimal outboard logic. Standard ROM and RAM chips from Intel's line graded out for appropriate speed and some unique memory parts are assigned unique 8000-series part numbers. Other parts in the series include a three-state latch, a three-to-eight decoder, and drivers for clocks and busses. A universal synchronous/asynchronous interface part has also been announced.

## INTERSIL 6100

Intersil's approach to the market is based on two ideas: CMOS for low power drain, and compatibility with DEC's PDP-8 to ease customer education. The 6100's instruction set is identical to that of the PDP-8; the speed is lower, but so is the power required. The wide operating temperature range and relaxed power supply tolerances that are characteristic of CMOS apply to the 6100.




The usual PDP-8 architecture applies. There is a single accumulator, twelve bits long, extended at one end by a one-bit link register. An accumulator extension, the MQ register, is included. All of the PDP-8 instructions that use the MQ are provided, but not all of the Extended Arithmetic Element (EAE) instruction set is included. Because the chip is entirely static, the clock frequency can be reduced down to DC; while this is unimportant for production, it can be useful during software testing.

The PDP-8 is followed exactly, and this may present at least one problem. In the PDP-8, a subroutine call modifies the first word of the subroutine; the return address is stored there. This scheme won't work if subroutines are stored in ROM, as might be done in a microprocessor application. Dummy subroutines must be loaded in RAM (where they are subject to possible corruption) in order to link to and from subroutines in ROM.

Two package sizes for the 6100 are being implemented. The standard 40 pin package is intended for general use; a 28 pin version with reduced capabilities will also be available for dedicated minimum parts applications.

## MIL

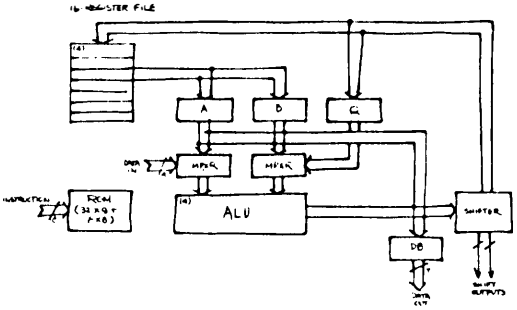
Microsystems International has made two abortive stabs at the microprocessor sweepstakes, first with a four-bit CPS/1, and later with an eight-bit model 7114. Now, MIL is successfully second-sourcing the Intel 8008 in a plastic package. They are expected to second-source another microprocessor before long.

## MONOLITHIC 6701

While the Monolithic Memories 6701 is not a complete microprocessor, it has all the essential features except instruction fetching and decoding. It might be considered as a Register/ALU four-bit slice, as found in other architectures (Intel, National, and Raytheon, for instance). The 6701 makes no assumptions about sources of instructions, and hence has no intrinsic provisions for referring to storage.

Historically, the 6701 has grown out of experience with the 74181 Arithmetic Logic Unit. To the ALU Monolithic has added a sixteen-register file and shift capabilities. There are two four-bit address lines into the register file so that inputs to the ALU can be selected. The Q register is always available and is used as an accumulator outside the register file.

Control of data flow within the 6701 requires sixteen bits. Two four-bit groups are used to select the A- and B-register contents, and eight bits are used to select the functional sequence inside the chip. The eight function bits are fed into two read-only memories to derive seventeen control signals. There are also four-bit input and output ports for communicating with a bus system.

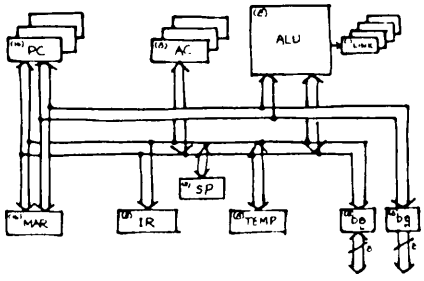



The 6701 can perform a register-to-register add in about 200 nanoseconds. When combined with instruction fetch and decode logic, a computer with a 400 nanosecond cycle (plus main storage cycle time) could be created.

The bipolar processor is also available in the 5701 model; it is specified over the military temperature range.

## MOSTEK 5065

The Mostek 5065 is the most elaborate among microprocessors in interrupt servicing. The eight-bit machine is a classical one-accumulator design. However, the programmer's registers (Program Counter, Accumulator and Link bit) are all provided in triplicate; the set of registers operated upon depends upon external pins, usually connected to interrupt circuitry, or on instructions executed.




The triplicated registers avoid any need to save the state of the machine when an interrupt occurs; shorter and faster interrupt routines result. Two interrupt input pins that may be selectively enabled under program control select one of the three register sets. Software also has the capability of changing to a new set of registers. The worst-case interrupt response time of the 5065 is about 23 microseconds.

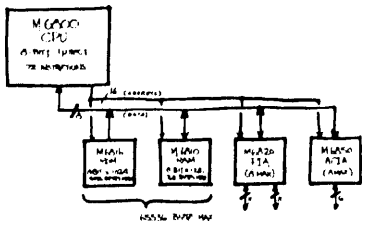
All data is operated on in eight-bit byte increments. The shared address/data bus has an eight-bit bidirectional half, and an eight-bit outbound-only half used for the higher-order bits of storage addresses. Addressing is accomplished via current-page/page-zero selection scheme involving one bit in the instruction. Another bit selects indirect or direct addressing. In indirect addressing, the most significant bit of the sixteen-bit address is interpreted as another indirect bit; this provides multiple-level indirect addressing (uncommon in microprocessors), but limits addressing to

32,768 bytes. An eight-bit stack pointer (SP) handles subroutine calls by storing the return address into page zero, the first 256 bytes of main storage.

Mostek developed the 5065 for a large European electronics manufacturer building special-purpose terminals; Mostek is just beginning to release production units to the U.S. market.

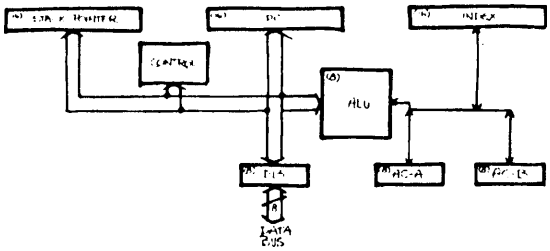
## MOTOROLA 6800

The Motorola 6800 is an integrated chip set with some special parts for input-output. Special ROM and RAM chips are included in the set so that small applications can be supported with no external circuitry. Specifically, the system provides a 1024x8-bit ROM, a 128x8-bit RAM, a bi-directional Peripheral Interface Adapter (PIA), and an Asynchronous Communications Interface Adapter (ACIA). The PIA offers bi-directional control on two eight-bit data busses, in addition to control and interrupt lines for communication to and from peripheral devices. Peripheral controllers still need to be constructed, but the interfacing of these controllers to the CPU promises to be easier than with any other microprocessor. The ACIA includes parallel/serial conversion on the chip.



All these parts connect directly to one another. An early application of the 6800 with 2Kx8 ROM, 256x8 RAM and five PIA's required *no* interface components between parts in the microprocessor family.

The CPU itself is organized as a conventional computer with an eight-bit data/instruction path. Instructions execute in from two to twelve microseconds. There are two accumulators and a sixteen-bit index register. Return addresses are stored in a stack in RAM, and that stack is referenced via a pointer in the CPU.

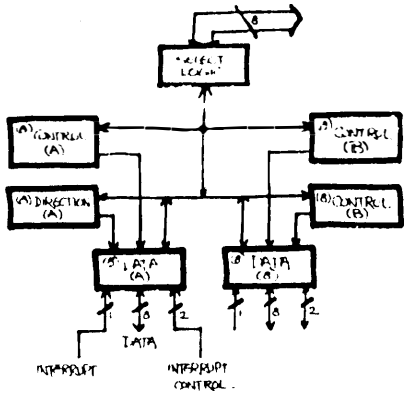



Abbreviated addresses of eight bits can be directed to refer to the first 256 bytes in main storage. Most programs are organized to refer to these locations for common work space. This abbreviated addressing is designed to compensate for the small number of CPU registers available.

Programming of the 6800 is relatively easy, but there are anomolies. There is, for example, no direct path from the two accumulators to the index register; to compute an index requires storing the sixteen-bit value into RAM, and then loading it into the index register. The choices of various addressing modes are somewhat arbitrary because only certain instructions admit certain modes; it is the programmer's responsibility to remember which addressing forms are valid for each instruction.

The 6800 provides for maskable interrupts at four levels. All the input/output devices appear on one interrupt level so that once an interrupt is detected polling must be used to isolate the device requiring attention. When an interrupt occurs, the processor places status and register contents into the push-down stack automatically. The interrupt return instruction refreshes the state of the machine from the stack. A software equivalent of the interrupt is provided, too; the computer's state is pushed onto the stack, and an interrupt service routine is invoked.

The PIA replaces the usual disorderly array of I/O circuitry with a single chip. Each PIA connects to the address and data busses and is treated in software just like storage locations. There are no special I/O instructions. The chip enable inputs are tied to memory address lines; this gives



each PIA a unique set of addresses. Inside the PIA there are two independent halves. Each half can handle interrupts, generate control output levels or pulses and transmit or receive up to eight bits. Each port has an associated control register, an output word register and a direction register, all accessible to the programmer.

To use the PIA, the direction register is loaded with eight bits to establish each corresponding I/O line as an input or output. The output word register is loaded with the bits to be sent out and input lines are read by simply reading from the storage address assigned to the PIA. In addition to the eight data lines, each port has an interrupt input and a multi-purpose control line. Control register bits define the multi-purpose line to be a second interrupt input, interrupt acknowledge output, control signal or a pulse generating output. Other control register bits provide masking for each interrupt line and the interrupt flags required by the interrupt polling routine.

The ACIA is derived from the PIA design. It incorporates the essential features of a UART on-chip, but it does not handle a pair of data paths like the PIA. The ACIA can be easily interfaced with one of the popular modems.

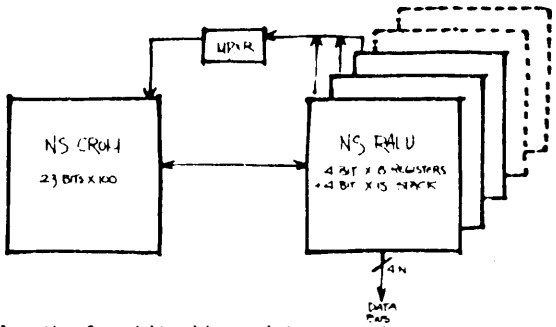
American Micro-systems has signed an agreement to second-source the entire 6800 parts set and expects to sample before 1975. Both Motorola and AMI intend to restrict sales of the PIA and ACIA to 6800-line customers.

## NATIONAL CMP-8

In addition to their well established four-bit slice microprogrammed GPC/P, National will soon be marketing their first single-chip processor. The CMP-8 will be a 40-pin NMOS device with features and add-ons for the data communications market. The eight-bit processor will be introduced early in 1975 along with several special purpose I/O chips. In addition to a UART, the line will include a programmable baud-rate generator.

## NATIONAL GCP/P

The GPC/P was the first microprogrammable microprocessor. Most customers have avoided microprogramming, preferring to use National's own microprograms in either the four-, eight- or sixteen-bit (IMP-4, -8 and -16) chip sets. This may change with the introduction of National's FACE chip.

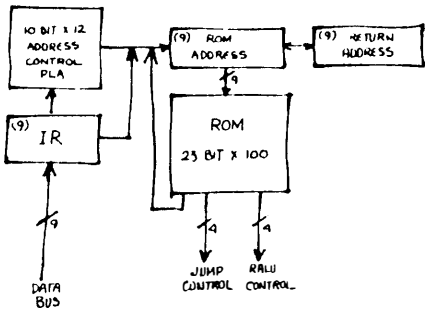



A GPC/P (General Purpose Controller/Processor) is composed of two major components: CROM (Control Read-Only Memory), and RALU (Register-ALU). The RALU is a four-bit slice of the registers and arithmetic-logic unit, reminiscent of the internal organization of many minicomputers. The CROM contains macroinstruction decoding control logic and a 100 by 23-bit ROM that contains the microprogram. A minimal system consists of one CROM and one RALU; that CROM must be suitably micro-programmed to operate on four-bit data quantities. In a minimum parts example, the four-bit wide registers are too

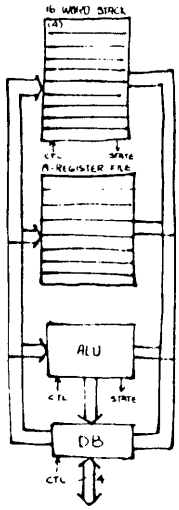
small for some required facilities, such as a program location counter, so that these must be added outboard of the GPC/P itself. This is, in fact, how the IMP-4 and IMP-8 operate.

In a larger configuration, four RALU's can be connected end-to-end to create a sixteen-bit word computer. One or two CROM's can be programmed to control this configuration. National supplies pre-programmed CROM's with IMP's, and a five- or six-chip set (depending upon instruction set) comprises the basis for the IMP-16. Similarly, two RALU's and a CROM form the basis for the IMP-8. A CROM, a RALU and a special interface chip (FILU) form the IMP-4.

It appears to be possible to create word widths of up to about twenty-four bits without being affected by propagation delays.




More than one CROM can be connected to a system, and the CROM's may be individually selected by a small amount of outboard logic. In some applications, then, available CROM chips may be used for most instructions and a custom CROM added to extend the instruction set. National is already doing this with an extended arithmetic set of instructions for the IMP-16. Each CROM holds 100, 23-bit microinstruction words. There is a four-bit path over which the CROM and RALU's communicate. The RALU contains multiplexing circuitry that allows external bus lines to appear like another register, and this permits input and output. To the microprogram, of course, main storage of the macromachine is just another peripheral device.




The RALU contains eight randomly-accessible registers, a sixteen-word push-down stack and the ALU. Each microinstruction can specify any two registers as sources for the ALU and a register as destination for the result. Sixteen bits of microcode are transmitted to the RALU on a four wire multiplexed control bus. A four-phase clock is required.

Usually, one of the RALU registers is allocated as the program location counter for the macromachine. The other registers are used as programmer-accessible registers, memory address registers, RAM-stack pointers, etc. The built-in push-down stack can be used for data and/or return addresses, depending upon the microprogram.

The microinstruction language is rich and apparently well designed.

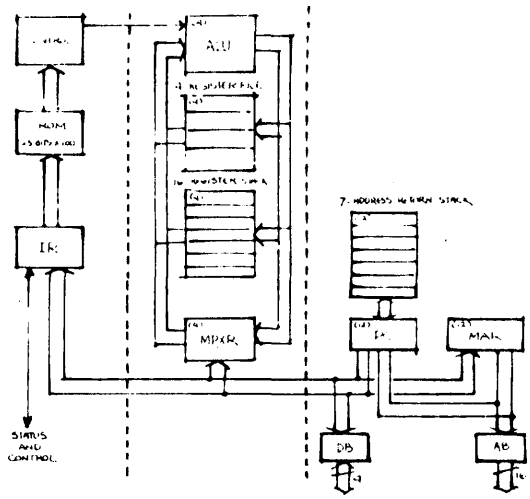
The newest member of the GPC/P parts family allows microprograms to be stored in user-programmed RAM or ROM. Until now, microprogramming of the GPC/P has been made costly by mask charges to create new CROM chips; several iterations are usually required to work out all the bugs.

The FACE (Field-Alterable Control Element) removes the ROM from CROM, and allows the customer to connect any available storage medium. Except for the outboard store, FACE appears to the GPC/P to be identical to CROM.

## NATIONAL IMP-4

The IMP-4 is National's latest computer built with their GPC/P parts set. In order to build a cost-effective computer, however, a new part had to be introduced with wide registers for such elements as the program location counter. A minimum system consists of three chips: one RALU holds

the four-bit data registers, one CROM holds the micro-program, and the FILU (Four-bit Interface Logic Unit) adds all that logic necessary to make a four-bit processor work with a twelve-bit address bus.




In the IMP-4, three of the RALU's seven registers are reserved for constants. The other four random-access CPU registers are used as four-bit accumulators in the user's program. The sixteen-place four-bit push-down stack is used for data in the IMP-4. The CROM is a standard GPC/P part and draws its instructions from the four-bit instruction register that is physically located on the FILU chip.

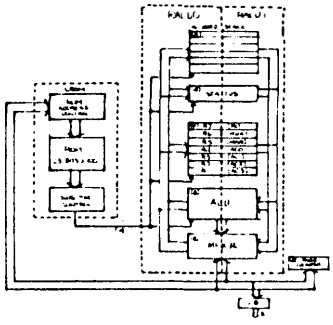
The FILU is the entire interface to the rest of the computer, both storage and peripherals. On that chip are found a Memory Address Register (MAR), instruction register, program location counter and a seven-place return address push-down stack. All of the address registers are twelve bits wide.

The IMP-4 is somewhat slower than other four-bit competitors, requiring twelve microseconds for a register-to-register binary addition. However, the added feature of microprogrammability could be exploited to compensate for that in many applications.

It is axiomatic that programming ease is inversely proportional to the number of different addressing schemes that have to be used with different register sizes. The IMP-4 suffers from this problem; it is difficult to program. The machine's language loosely follows Data General Nova style, but the inflexibility of the jump addressing and the complicated way of loading the storage address (MAR) make practical programs relatively long and difficult to follow.

## NATIONAL IMP-8

National's IMP-8 is the second-most popular configuration designed around the GPC/P architectures. The basic system consists of one CROM and two RALU chips. Because all registers are only eight bits wide, one additional external register is required to extend the program location counter; that external eight-bit register holds the most-significant bits of the current program address and is called a page-selection register (page counter).



The RALU's organization provides the user with four accumulators. The other three main registers are used for storage addressing (MAR and MDR), and the least significant bits of the program location counter. The eight-bit MAR specifies the least significant bits of the address; the eight most significant bits come from the page counter or are set to zero. Data, therefore, can be addressed directly or indirectly via the current page or page zero.

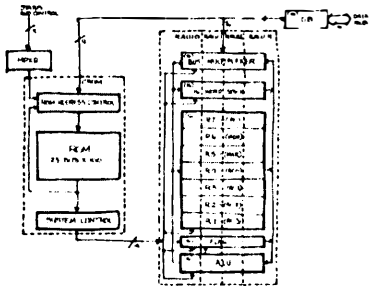
Since indirect addressing is provided, most of page zero is typically reserved for addresses to other pages in main storage; each indirect address occupies two bytes. Two of the accumulators can be used as index registers for addressing, and that alleviates many complexities in programming

Jumps and subroutine calls affect both the internal and external halves of the program location counter. Subroutines put two bytes onto the RALU push-down stack, so that eight call-levels can be used; the stack can also be used to store temporary data, one byte at a time.

## NATIONAL IMP-16

The IMP-16 is, by far, National's most popular computer. It is significantly different from other designs but its concepts are so fundamental that many of the microprocessors now being designed will look similar.

The IMP-16 is made up of one or two CROM chips (depending upon whether the user selects the standard or the augmented instruction set), and four RALU chips. The IMP-16 requires a significant amount of external hardware to make a complete computer; of course, it is this external hardware that allows flexibility. Instead of being "locked in" to a particular conditional jump structure, for example, the designer using the IMP-16 parts or boards can specify unique combinations; the block labelled MPXR is an example of that required circuitry.



Since each register of the user's computer is sixteen bits long, the architecture is quite easy to follow. The seven randomly-accessible registers of the RALU are allocated by the microprogram so that the programmer has four accumulators. Two of the accumulators can be used as index registers; the other two have some special implications for certain instructions. The other three registers of the RALU are used by the microprogram for program counter (PC), and storage addressing (MAR and MDR).

The sixteen-place sixteen-bit push-down stack is used for subroutine calls and interrupt handling; it can also be used for data storage under program control. The sixteen bit "flags" register provides storage for such entities as carry and overflow conditions from the ALU--that requires four bits; the remaining twelve bits can be used by the programmer as status latches or as a way to communicate status to external devices.

The IMP-16 instruction set is vaguely modelled after the popular Data General Nova series. The instruction set is disorderly, and it takes some experience with the computer to be able to easily remember subtle differences among nearly identical instructions. Programming high-low-equal tests is particularly difficult because of the unfortunate choice of jump conditions.

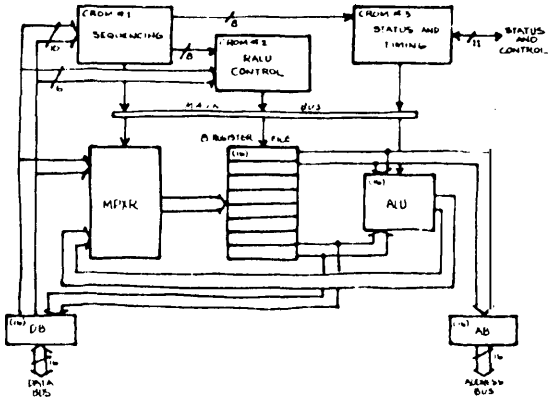
With the introduction of the FACE chip (see the GPC/P parts description), the IMP-16 will certainly become an even more popular computer. The IMP-16 card systems can accept FACE by plugging it into the second CROM socket provided on the card.

## RAYTHEON RP16

Raytheon's RP16 uses four-bit RALU slices built with bipolar technology to implement a sixteen-bit microprocessor. Three additional chips are included in the set and all three are required to control the RALU's. The design uses ECL circuitry for some internal functions but all interfaces are compatible with low-power Schottky TTL. The microcycle time is 200 nanoseconds and the basic macroinstruction takes one microsecond. The RP16 will appear in two slightly different models: Model A has arithmetic features (a divide instruction, for example), and Model B is intended

for byte-handling applications. Both models have multiply and double-precision add instructions and model B includes byte-wise comparisons.

One major bus is the primary control route from the three control ROM's to the RALU chips. The eight sixteen-bit registers in the RALU file are mostly occupied with overhead functions. The following assignments are made in the microprogram that Raytheon intends to supply:



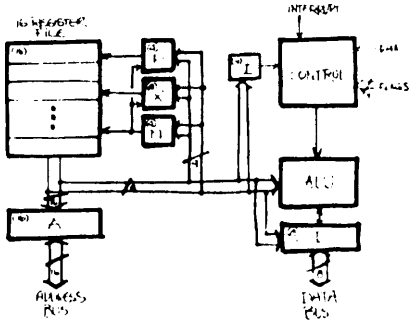

Stack Pointer to RAM
Program Location Counter
*Base Register (for base-displacement addressing)
*Index Register
*Accumulator
*Accumulator Extension (for multiplication)
Working Register (typically used as MDR)
Address Register (MAR)

The four registers marked with asterisks are those that are useful to the programmer. The architecture that is microprogrammed in, then, is that of a two-address, one-accumulator machine.

A two-level interrupt scheme is included with both external and program-initiated interrupts possible. This is also one of the few processors which offer a military operating temperature range.

## RCA COSMAC

RCA will soon provide an eight-bit microprocessor in the six-to-eight microsecond speed range in CMOS. The COSMAC CPU has sixteen general-purpose sixteen-bit registers. Each register may be used for data-holding, as an index register, or as a program location counter. Thus, the depth of an on-chip push-down stack is limited by other uses of the registers as determined by the programmer. At any one point in time, one register is the program location counter (as selected by a four-bit P-register), and one is the current operand (as selected by a four-bit N-register). Since a program location counter in one program can be treated as an operand in another, the push-down stack of return addresses can be extended at will by storing register contents (eight bits at a time) into main storage.

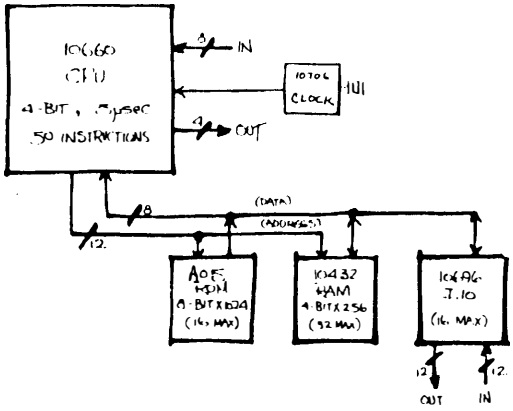



The register-indirect memory referencing scheme is similar to that used in other processors: Instructions do not contain addresses operands, so all operand addresses must first be loaded into registers. COSMAC's register file has been organized to largely ameliorate this problem by providing many wide registers and allowing the programmer to use any one of them to address memory. In many applications the net result is a shorter program store, and in many other applications the necessity to continually load addresses becomes a trememdous burden. It all depends on the data structures of the application.

COSMAC is being built in a two-chip form at present, although future plans call for a one-chip version. A higher-speed version on a sapphire substrate is being investigated. RCA has been actively publicizing applications they've developed, but it will be toward years' end before samples will be available at large.

## ROCKWELL PPS-4

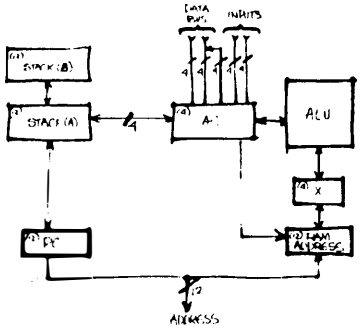
The PPS-4 was Rockwell's first microprocessor and its list of special memory and input/output parts keeps growing. The CPU can be supported by a crystal controlled clock chip, RAM and ROM chips (including one unique chip with both RAM and ROM), and a wide range of specialized input/output devices.




Several different kinds of storage chips are available in the set. Read-only memory is available in 1024 and 2048 byte increments, and read-write storage comes in 256-by-4 packages. One package has 70x8 ROM plus 76x4 RAM. The input/output facilities consist of:

General I/O, with three four-bit input and output groups per chip,
Keyboard/Display circuit, that handles 64 keys and a 16-digit seven-segment display,
Printer controller, for the Seiko drum printers,
1200 bps UART and modem, and
Bus Interface circuits for general interface needs.

The keyboard/display and printer controller chips suggest that Rockwell is aiming at the desk calculator market; the other devices suggest otherwise. The telecommunications interface is one of the most recent announcements.




The central processor of the PPS-4 is organized as a conventional one-accumulator computer. The ALU operates only on four-bit digits, and results generally are sent to the accumulator. All storage references are made through the twelve-bit RAM address register, the four-bit digits of which can be loaded from the accumulator. The X register provides an additional arithmetic temporary register that can be used to modify the RAM address register to simulate indexing.

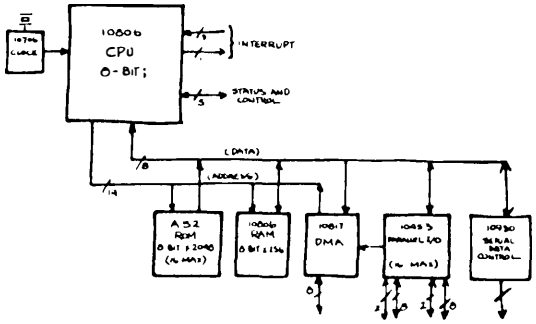
The PPS-4, like many other small microprocessors, has no instructions that refer to RAM directly; all datum references must be made through the address held in the RAM address register.

The program location counter has two stack positions available for subroutine return address saving. One of the stack positions can be copied (four bits at a time) into the accumulator, so that a common subroutine can be used to extend the apparent stack depth by using RAM.

Instructions come in from the data bus in eight-bit increments, but data uses only four bits of the bus. The CPU has two four-bit input status groups that can be read to simulate the effect of interrupts.

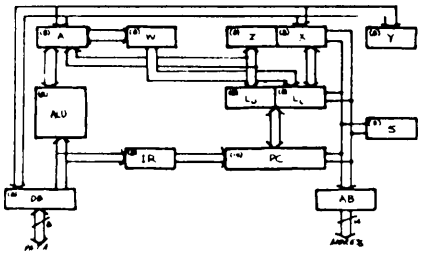
## ROCKWELL PPS-8

Rockwell's experience with the PPS-4 has led them to expand the word size and develop the PPS-8. The newer microprocessor uses many of the same basic design philosophies as the four-bit computer, but more facilities for handling high-speed input/output problems are part of the set.




The PPS-8 parts set has a CPU, clock, ROM, RAM, and generalized input/output interfaces with a special DMA chip. The DMA controller has the necessary storage address registers and CPU-inhibit lines to allow peripheral devices to have direct access to the RAM via the data bus. The clock and ROM chips are the same ones used with the PPS-4. The 1200 bps serial data control chip is also common to both microcomputer sets, and interfaces in accordance with the RS-232C specification.

The DMA controller has eight input lines that are connected to different peripheral device controllers. These eight DMA request lines are inherently priority encoded; there are eight separate storage address registers in the DMA chip, so that DMA requests can be interrupted. This may require some careful peripheral controller design to avoid timing problems. Also, the termination of a block transfer is controlled by the least significant eight bits of the address, and a record length counter that is limited to 256 bytes; longer transfers require software intervention.




The CPU is obfuscated by its genesis. In the translation of the PPS-4 design into an eight-bit computer, some unusual complications have been introduced. The A and W registers (accumulator and working, respectively) are used for most data manipulation. The X, Y and Z registers are used for temporary storage; some instructions treat pairs of them as a single sixteen-bit register. The Z-X pair, for instance, is used as a storage address register for register-indirect addressing. The L-register, on the other hand, is used to address bytes in program space (ROM). Each time L is used to fetch from ROM, it is automatically incremented by one. The L register is also used for subroutine return addresses, and so forms a one-level push-down stack for the program counter. Since L can be accessed from the program, the stack depth can be extended as necessary by software; for brief routines that call no other subroutines, the L register need not be perserved in RAM.

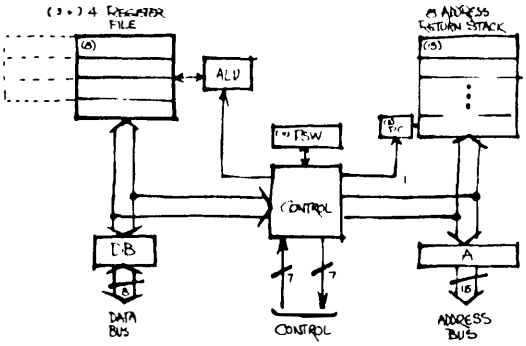
The five-bit stack register (S) allows reference to the first 32 bytes of RAM. The X, Y, Z, A and L registers can be pushed onto the data stack; this is particularly useful during interrupt servicing. Each time the state of the machine is saved, seven words are required in the stack.

Three levels of priority interrupts are handled by the CPU by creating subroutine calls in the hardware. Interrupts must be disabled in order to use the L-register as a data-reference register (as might expected when indexing across an array), so that an interrupt doesn't destroy the address in the L-register. Since the highest priority interrupt can't be disabled, the L-register must be used with caution.

In general, the PPS-8 is probably an efficient computer. It suffers from a lack of consistency in architecture, and many people will find it hard to program.

## SIGNETICS 2650

The 2650 is Signetics' first venture into the microprocessor marketplace. The computer is organized for eight-bit data, and a 32,768 word maximum program size. Available information is sketchy, but some essential features have been documented.




The central processor chip has a split register file and an eight-address return stack, the top-most item of which holds the current program location counter. The processor is controlled by the fourteen bit program status word (PSW).

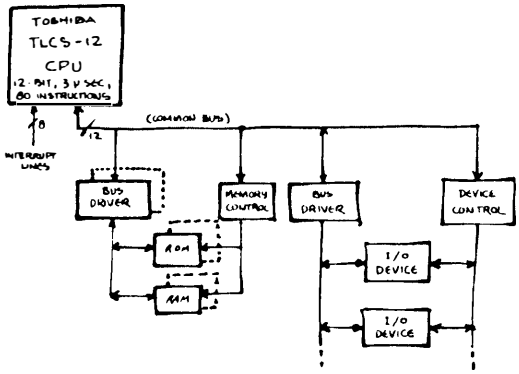
The register file is divided into three pairs of eight-bit registers, and a common register. The selection of which half of the register file is used by instructions is dependent upon a bit in the PSW. A program, then, will normally have four registers to operate on. When a subroutine is called, or when an interrupt strikes, the register selection bit in the PSW can be changed to provide three preserved and three working registers.

Other PSW bits control the significance of the carry bit in shifts and computations, whether the sign enters into comparisons or not, and (in general) extend the properties of classes of instructions.

## TEXAS INSTRUMENTS

Rumors abound about TI's entry into microprocessor field. Certainly, when the giant moves the industry will take notice. The most persistent rumor says that TI will begin to sample a small bipolar processor to selected customers in September, 1974. Other rumors mention two other microprocessor designs on the drawing boards.

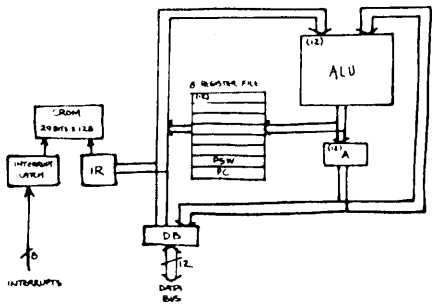
## TOSHIBA TLCS-12




Toshiba's TLCS-12 is a twelve-bit computer composed of a CPU chip, bus driver chips, storage and device control components. Most of the interfacing difficulties are handled by the driver and control components; all chips communicate on a single twelve-bit bus.

The memory control chip allows common ROM and RAM parts to be used. The device control is a general purpose part that translates inter-chip signals into TTL-compatible inputs and outputs. Interrupt signals are all fed to the CPU chip and subjected to an eight-bit parallel mask. The highest priority unmasked interrupt pending will be treated by the CPU.

The CPU is controlled by a microprogram that is stored in a 128-word ROM. The 29 microprogram bits control all data flow within the processor and all external bus signals. Data manipulation is performed on eight registers in the main file. Two of the registers are reserved for program location counter and program status word by the microprogram; the other six registers are available for use as storage addresses and data. The first eight locations of main storage do not normally exist in a TLCS-12 configuration; their addresses are reserved to refer to the eight file registers. The next eight words in storage hold program status words for each of the eight possible interrupts.

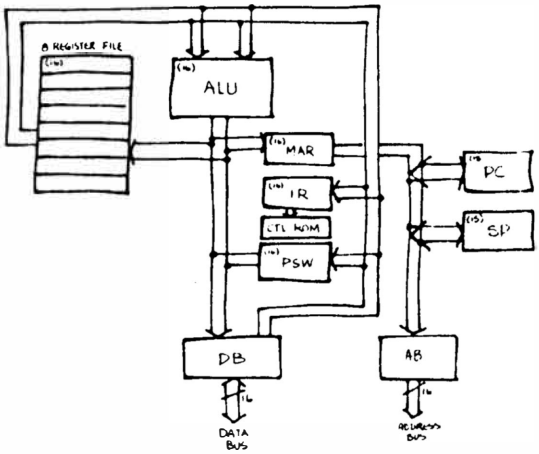


The TLCS-12 apparently requires a moderate amount of common TTL circuitry to assemble a working system. Because the clock frequency is temperature sensitive, software cannot depend upon counting instruction execution cycles for precise short-term timing.

## TRANSITRON TMC/1601

The 1601, from an unlikely source, is a newly-rumored bipolar sixteen-bit computer that is currently under design. The CPU is architecturally similar to the DEC PDP-11, but the structure is based on control ROM's and four-bit Register-ALU slices. Because of a highly generalized addressing scheme, the computer is likely to be easy to program. Some early samples show as much as 33% improvement in program size as compared to some popular sixteen-bit machines.

The programmer's view of the machine includes an eight-register file; each register can be used for any arithmetic or indexing operation. There are some special implications for certain registers in subroutine calling and interrupt handling, though; so the programmer may have to restrict the use of one or two registers.




Subroutine return stacking is done in RAM, and the stack is addressed by a pointer in the CPU. The upper and lower limits on the address of stack entries are stored in special protected locations in RAM so that stack underflow and overflow can be detected.

When an interrupt strikes the 1601, the program location counter, the current program status word and one of the registers in the programmer's file are all pushed onto the stack in RAM. The interrupting device's address is then added to the contents of RAM location zero to obtain the address of two words that contain the new program counter and PSW for that interrupt. This vectoring scheme allows multi-level interrupt handling, because the new PSW can selectively mask off lower-priority interrupts.

## WESTERN DIGITAL

Western Digital is said to be working on two microprocessors. One, a microprogrammed PDP-11/05 emulator, is being built expressly for DEC; first parts should be delivered to DEC before the end of this year.

Another device is being designed to compete with the Intel 8080. This one is probably an NMOS design to be announced early in 1975.

* * * * *

This month's issue written by:

J. Ogdin and

S. McPhillips

of Microcomputer Technique's staff.

## MICROPROCESSOR MANUFACTURERS

American Micro-Systems, Inc.
3800 Homestead Road
Santa Clara, CA 95051
(408) 246-0330

Burroughs Corporation
Defense, Space & Special Systems Group
Paoli, PA 19301
(215) 269-1100

Electronic Arrays, Inc.
550 Middlefield Road
Mountain View, CA 94043
(415) 964-4321

Fairchild Semiconductor
313 Fairchild Drive
Mountain View, CA 94040
(415) 962-3867

General Instrument Corporation
Microelectronics Division
600 W. John Street
Hicksville, NY 11802
(516) 733-3304

Inselek
743 Alexander Road
Princeton, NJ 08540
(609) 452-2222

Intel Corporation
3065 Bowers Avenue
Santa Clara, CA 95051
(408) 246-7501

Intersil
10900 Tantau Avenue
Cupertine, CA 95014
(408) 257-5450

Microsystems International Ltd.
75 Moodie Drive
Ottawa, Ontario
Canada K1Y 4J1
(613) 828-9191

Monolithic Memories, Inc.
1165 E. Arques Avenue
Sunnyvale, CA 94086
(408) 739-3535

Mostek
1215 W. Crosby Road
Carrollton, TX 75006
(214) 242-0444

Motorola Semiconductor
Box 20912
Phoenix, AZ 85036
(602) 244-3965

National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, CA 95051
(408) 732-5000

Raytheon Company
Semiconductor Division
350 Ellis Street
Mountain View, CA 94042
(415) 968-9211

RCA Corporation
Solid State Division
Somerville, NJ 08776
(201) 722-3200

Rockwell International
Microelectronics Device Division
3310 Miraloma Avenue
Anaheim, CA 92803
(714) 632-5803

Signetics Corporation
811 E. Arques Avenue
Sunnyvale, CA 94086
(408) 739-7700

Tokyo Shibaura Electric Company
1, Komukai Toshibacho
Kawasaki-City
Kanagawa, 210, Japan
Kawasaki 51-3111

Transitron Electronic Corporation
168 Albion Street
Wakefield, MA 01880
(617) 245-4500

Western Digital Corp.
19242 Red Hill Avenue
Newport Beach, CA 92663
(714) 557-3550

# MICROPROCESSOR SCORECARD®

| MICROPROCESSOR SCORECARD | CLASSIFICATION | TECHNOLOGY | PARTS FAMILY: Clock Driver | PARTS FAMILY: I/O Interface | PARTS FAMILY: UART/USRT | PARTS FAMILY: RAM | PARTS FAMILY: ROM PROM | PARTS FAMILY: Interface | FEATURES: Interrupts | FEATURES: One Chip CPU | FEATURES: Microprogrammed | FEATURES: Accessible Stack | FEATURES: DMA Ability | FEATURES: BCD Arithmetic | WORD SIZE (Data Instruction) | ADDRESS CAPACITY (Program Words) | CLOCK (MHz Phases) | REGISTER ADD TIME (usec per Data Word) | NUMBER OF CPU REGISTERS: ALU | NUMBER OF CPU REGISTERS: XR | NUMBER OF CPU REGISTERS: GP | RETURN STACK SIZE (Nr. x Bits) | VOLTAGES REQUIRED | POWER DISSIPATION | OPERATING TEMPERATURE RANGE (°C) | PACKAGE SIZES (Dip Pins) | PRICE RANGE (approx. 100 qty. CPU) | STATUS: First Samples | STATUS: First Deliveries | REMARKS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BURROUGHS MINI-D | 8 Bit CPU | PMOS | | | | | | | ✓ | | | | | | 8/12 | 256 | 1000/1 | 9 | 3 | | | 1 | 1 x 8 | -12 +5 | | 0 ~ 70 | 16 | $ 60 | 2073 | 3073 | ROM on CPU |
| ELECTRONIC ARRAYS | 8 Bit CPU | NMOS | | ✓ | | ✓ | ✓ | | ✓ | ✓ | | ✓ | ✓ | ✓ | 8 8 | 64K | 1000/2 | | 1 | 1 | 15 | 7 x 16 | | | | 40 | | 3075 | | 16 Byte String Operations |
| FAIRCHILD F-8 | 8 Bit CPU | NMOS | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | ✓ | ✓ | ✓ | 8 8 | 64K | 2000.0 | 2 | 65 | | | (RAM) | +5 +12 | 6 | 0 ~ 70 | 40 | $ 75 | 4074 | 1075 | Clock on Chip |
| FAIRCHILD PPS-25 | 4 Bit CPU | PMOS | | ✓ | | ✓ | ✓ | | | | | | | ✓ | 4 x 25 12 | 6656 | 400/2 | 62.5 | 1 | | | 4 x 12 | -9 +5 | 6 | 0 ~ 70 | 16 18 24 40 | $ 60 | | 2071 | |
| INTEL 4004 | 4 Bit CPU | PMOS | ✓ | ✓ | | ✓ | ✓ | ✓ | | ✓ | | | | ✓ | 4 8 | 4K | 740/2 | 10 8 | 1 | | 16 | 3 x 12 | 15 or (-10 +5) | 10 | 0 ~ 70 | 16 | $ 30 | 2071 | 4071 | |
| INTEL 4040 | 4 Bit CPU | PMOS | ✓ | ✓ | | ✓ | ✓ | ✓ | ✓ | ✓ | | | | ✓ | 4 8 | 4K | 740.2 | 10 8 | 1 | | 24 | 7 x 12 | 15 or (-10 +5) | 10 | 0 ~ 70 | 16 24 | $ 40 | 4074 | 4074 | 8K Program Space in Two Banks |
| INTEL 8008-1 | 8 Bit CPU | PMOS | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | | | | 8 8 | 16K | 800.2 | 12 5 | 1 | | 6 | 7 x 14 | -9 +5 | 10 | 0 ~ 70 | 18 | $ 60 | 4071 | 1072 | Second Source MIL |
| INTEL 8080 | 8 Bit CPU | NMOS | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | ✓ | ✓ | ✓ | 8 8 | 64K | 2083.2 | 2 | 1 | | 6 | (RAM) | -5 +5 +12 | 10 | 0 ~ 70 | 40 | $ 200 | 4073 | 2074 | |
| INTEL 3001 | 2 Bit Slice | Bipolar | | | | ✓ | ✓ | ✓ | ✓ | | ✓ | | ✓ | | 2N 18+ | 512 | 6061/1 | 165 | 2 | | 10 | (NONE) | 5 | 4 5 | 0 ~ 70 | 28 40 | $ 550 | 3074 | 3074 | |
| INTERSIL 6100 | 12 Bit CPU | CMOS | ✓ | | | ✓ | | | ✓ | ✓ | | | ✓ | | 12 12 | 4K | 4000.0 | 5 | 1 | | 1 | Modifies Program | 5 | 01 | -55 ~ +125 | 28 40 | $ 175 | 4074 | 1075 | PDP 8 Code Clock on Chip |
| MONOLITHIC MEMORIES 6701 | 4 Bit RALU | Bipolar | | | | | | | | | ✓ | | | | 4 17 | | 6666/1 | 2 | 3 | | 16 | (NONE) | +5 | 1 | 0 ~ 75 | 40 | $ 95 | 1074 | 2074 | |
| MOSTEK 5065 | 8 Bit CPU | PMOS | | | | | | | ✓ | ✓ | | ✓ | ✓ | | 8 8 | 32K | 1400/3 | 10 | 3 | | | (RAM) | -12 -5 +5 | 7 | 0 ~ 50 | 40 | $ 100 | 1074 | 3074 | 3 State CPU |
| MOTOROLA 6800 | 8 Bit CPU | NMOS | | ✓ | ✓ | ✓ | ✓ | | ✓ | ✓ | | ✓ | | ✓ | 8 8 | 64K | 1000.2 | 2 | 2 | 1 | | (RAM) | 5 | 25 | 0 ~ 70 | 24 40 | $ 150 | 2074 | 4074 | Second Source AMI |
| NATIONAL CMP-8 | 8 Bit CPU | NMOS | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | ✓ | ✓ | | 8 8 | 64K | '2 | 16 | 2 | 2 | | (RAM) | | | | 40 | | 1075 | 2075 | |
| NATIONAL GPC P | 4 Bit Slice | PMOS | | | | | | | | ✓ | ✓ | ✓ | ✓ | | 4N 23 | 100 | 715.4 | 14 | 8 | | | 16 x 4N | -12 +5 | 7 | 0 ~ 70 | 22 24 | $ 150 | 1073 | 3073 | 1 ≤ N ≤ 6 |
| NATIONAL IMP-4 | 4 Bit CPU | PMOS | | | | | | ✓ | ✓ | | ✓ | | | ✓ | 4 4 | 4096 | 500.4 | 12 | 4 | | | 7 x 12 | -12 +5 | 10 | 0 ~ 70 | 24 40 | $ 150 | 3074 | 4074 | 16 x 4 Data Stack |
| NATIONAL IMP-8 | 8 Bit CPU | PMOS | | | | | | | ✓ | | ✓ | ✓ | ✓ | | 8 8 | 64K | 715.4 | 4 6 | 3 | 1 | | 16 x 8 | -12 +5 | 10 | 0 ~ 70 | 22 24 | $ 230 | 4073 | 1074 | |
| NATIONAL IMP-16 | 16 Bit CPU | PMOS | | | | | | | ✓ | | ✓ | ✓ | ✓ | | 16 16 | 64K | 715.4 | 4 6 | 2 | 2 | | 16 x 16 | -12 +5 | 14 | 0 ~ 70 | 22 24 | $ 310 | 1073 | 3073 | |
| RAYTHEON RP-16 | 4 Bit Slice | Bipolar | | | | | | | ✓ | | ✓ | | | | 4N 48 | 64K | 5000/1 | 1 | 1 | 1 | 2 | (RAM) | 5 | | -55 ~ +125 | 48 | | 4074 | | |
| RCA COSMAC | 8 Bit CPU | CMOS | | | | | | | ✓ | ✓ | | ✓ | ✓ | ✓ | 8 8 | 64K | 2670/1 | 6 | 1 | | p | r x 16 | 5 ~ 12 | 01 | 55 ~ +125 | 28 40 | $ 300 | 4074 | 2075 | (p+2r) ≤ 30 |
| ROCKWELL PPS-4 | 4 Bit CPU | PMOS | ✓ | ✓ | ✓ | ✓ | ✓ | | | ✓ | | ✓ | | ✓ | 4 8 | 4K | 200/2 | 5 | 1 | | 1 | 2 x 12 | 17 | 225 | 0 ~ 70 | 42 | $ 45 | 1072 | 3072 | 8K Program Space in Two Banks |
| ROCKWELL PPS-8 | 8 Bit CPU | PMOS | ✓ | ✓ | ✓ | ✓ | ✓ | | ✓ | ✓ | | ✓ | ✓ | ✓ | 8 8 | 16K | 256.2 | 4 | 1 | 1 | 2 | (RAM) | 17 | 3 | 0 ~ 70 | 42 | $ 47 | 4074 | 1075 | |
| SIGNETICS 2650 | 8 Bit CPU | NMOS | | | | | | | ✓ | ✓ | | | | ✓ | 8 8 | 32K | 1200/1 | 4 8 | 7 | | | 8 x 15 | 5 | 5 | 0 ~ 70 | 40 | $ 120 | 4074 | 1075 | |
| TOSHIBA TLCS-12 | 12 Bit CPU | NMOS | | ✓ | | ✓ | ✓ | | ✓ | ✓ | ✓ | | | | 12 12 | 4K | 1000.3 | 13 | 4 | | 6 | (RAM) | -5 +5 | 8 | -20 ~ +80 | 16 24 26 42 | $ 215 | 2074 | 3074 | |
| TRANSITRON TMC 1601 | 4 Bit Slice | Bipolar | | | | | | | ✓ | | ✓ | ✓ | ✓ | | 16 16 | 32K | | 4 | u | u | | (RAM) | | | | 48 | | 2075 | 3075 | (u+v) ≤ 8 |

®Microcomputer Technique, Inc.