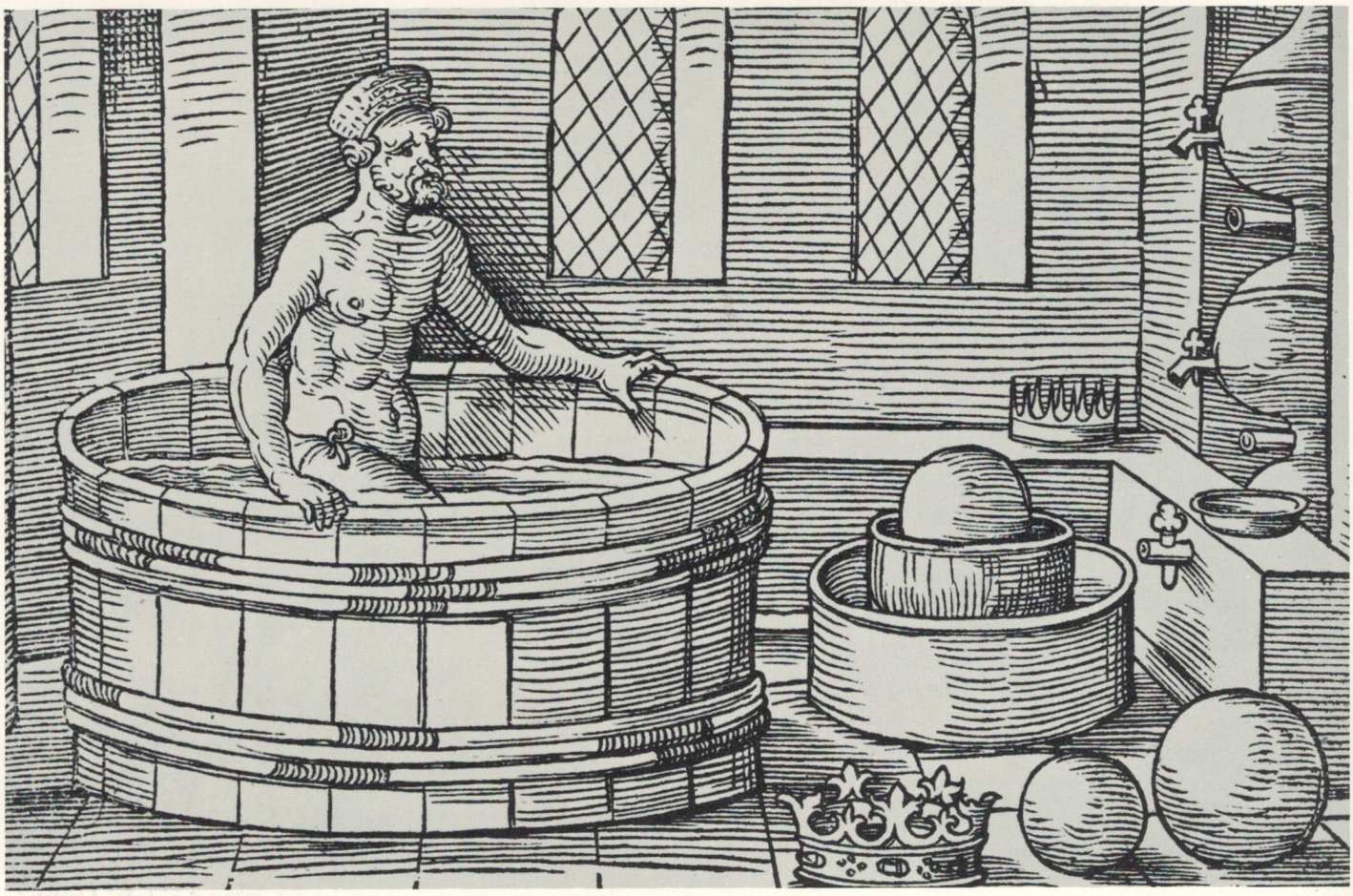
ROLM
CORPORATION

The world's toughest mini-computer goes anywhere



Medieval woodcut of Archimedes solving a problem of the relative density of silver and gold.

Innovation

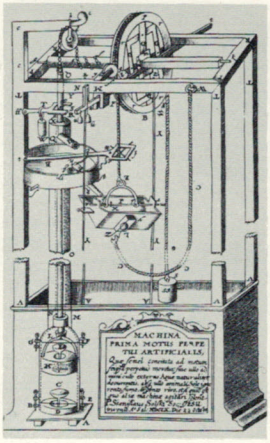
Throughout history, scientific innovators and problem solvers have searched for answers with an inquiring mind—frequently in unusual places. It is said that Archimedes, intent on solving a problem of the relative density of silver and gold, discovered the answer while in his bath. He was so excited about finding the solution, he ran through the streets naked, shouting "Eureka—I found it."

Today, "Eureka" has been overworked by movie and television versions of mad scientists. The word has become a laugh line. However, at Rolm Corporation we encourage its use. Because at Rolm, we value innovation highly.

Technical innovation is not just our goal, it is the only thing we can bank on. We're betting that our innovations will open up new and economical ways to use small computers. The Ruggednova severe environment minicomputer is just the first step.

Our dedication to innovation is matched only by our determination to produce high quality equipment and deliver it on time. In fact, it all fits together. Only by innovating can we produce quality for less money. It's the only way we know how to receive attention in a marketplace dominated by giants who are household words.

The Ruggednova Model 1601 is our first product. While it wasn't born in a bath tub, it could operate there, or in a host of other severe environments. Its construction, pricing and delivery are realistic evidence that we are doing what we say we can do—through innovation.



Thirteenth century diagram of the perpetuum mobile.

Necessity and the mini-computer

Whenever an idea captures the imagination of the scientific community it flourishes and pops up everywhere. For example, over the centuries, hundreds of perpetual motion machines have been designed. We'll probably see many more versions of them in our lifetime.

In the late 1960's, the mini-computer idea caught the attention of the technical community and the machines flourished. There's no doubt we're going to see a lot more of them in the 70's.

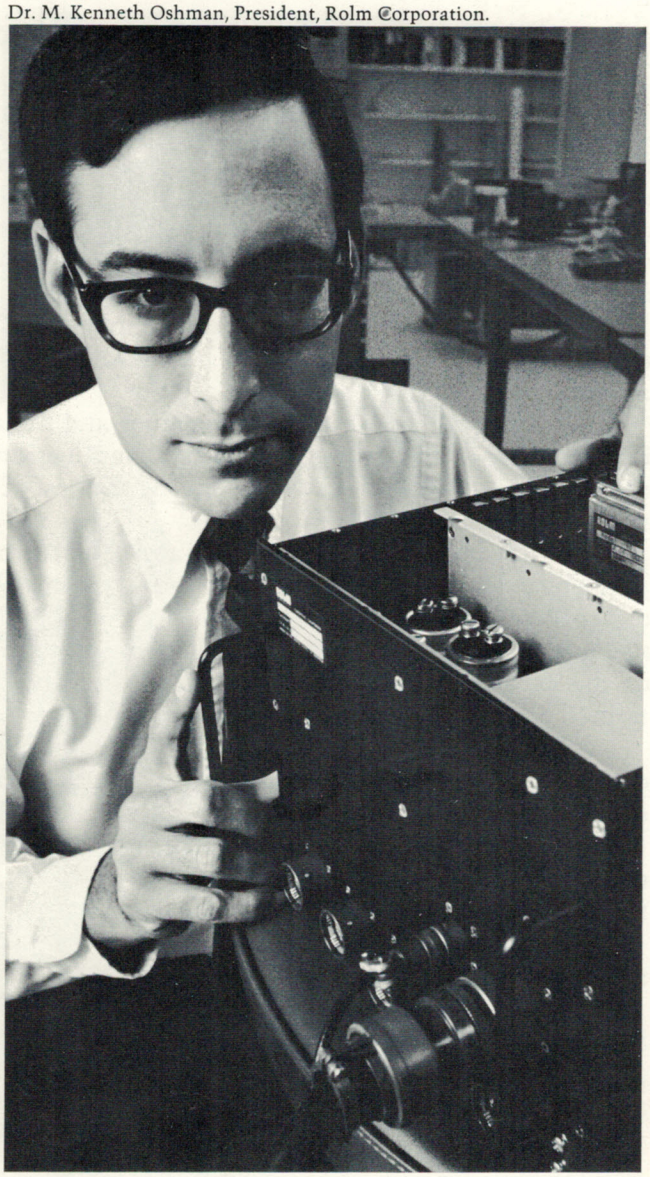
The small computer earned this surge of attention by virtue of answering a need through innovative technology. In fact the industry responded to necessity so rapidly that we are in the midst of a computer revolution. In their short life, small machines have become a common denominator in systems design. Systems built around them are, and will become, more complex. The mini-computer has achieved a final status—as a component.

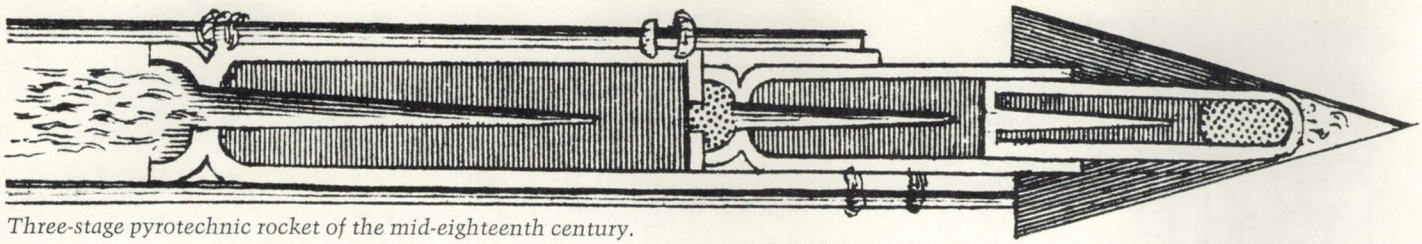
The Ruggednova was born in the thick of this trend and carries it one step further. For, while small machine utility is increasing and unit costs are coming down, today's machine is stuck in the lab. The Ruggednova severe environment computer will go anywhere. And, it is traveling at economical, off-the-shelf prices.

Built to meet severe environment Mil Specs, the Model 1601 is a rugged version of Data General Corporation's Nova. The architecture, software and I/O's are identical to the Nova. What we've done is to open up a world of new, economical applications by removing the environmental restraints which are generic in commercial machines.

We believe the Ruggednova is the next logical extension of mini-computer technology. Just look around and find an application that doesn't need to fly, travel the back roads or operate under extremes of temperature, RFI, shock, vibration and humidity.

Dr. M. Kenneth Oshman, President, Rolm Corporation.





Three-stage pyrotechnic rocket of the mid-eighteenth century.

Why militarize a commercial machine?

When we built the Ruggednova, we had one objective in mind: to build the world's most versatile general purpose mini-computer at a reasonable price.

By building the machine to Mil Specs we established the environmental operating range for ourselves and for our customers. However, the concept of versatility demanded more than a rugged package design that would stand up under severe environments. Our concept demanded the best machine design as well.

We looked the field over and decided that Data General Corporation's Nova had the best small computer architecture going.

By taking the Nova logic architecture and building it in a totally different, rugged package, the Ruggednova design answered our objective by offering the best of two worlds—the lab and the field.

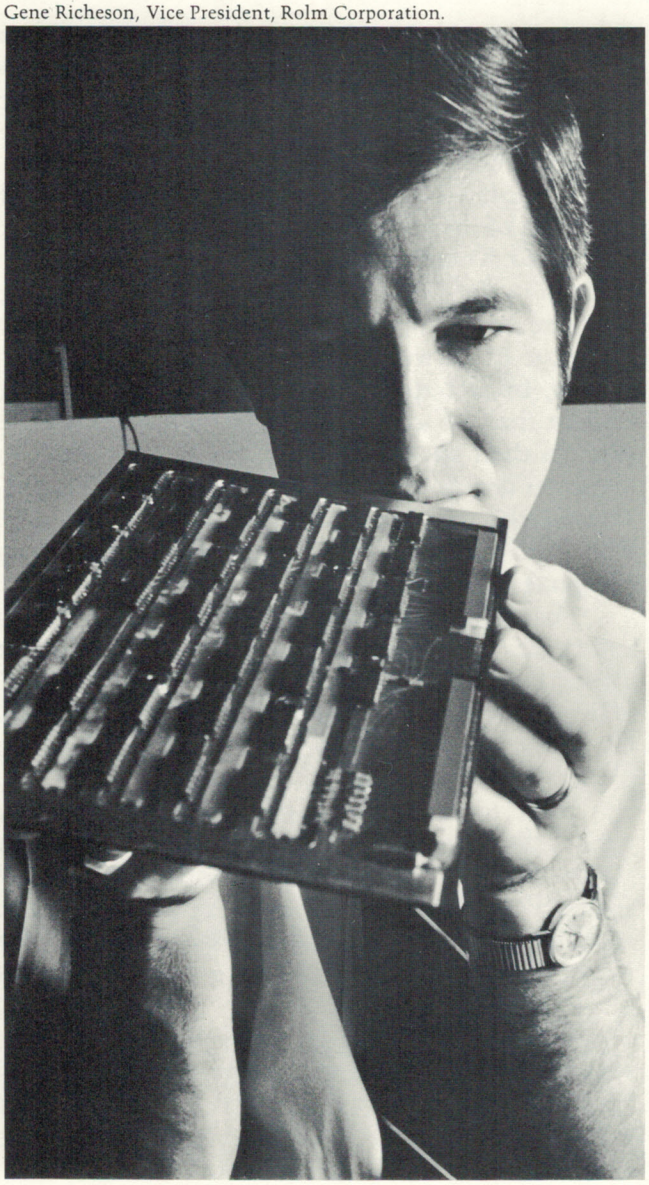
The advantages of our approach accrue to the computer user in lower costs and in usability. The Nova design is proven and field tested. Software exists today, not months from now, and it is field proven. By virtue of the Nova success, there is a growing multitude of trained programmers who know the machine.

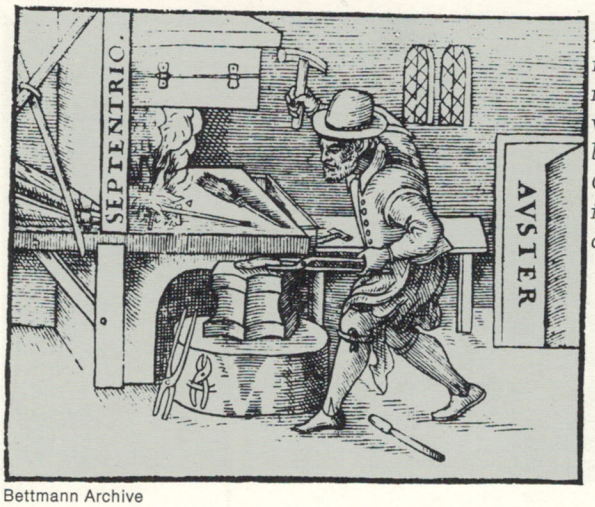
Additionally, I/O interfaces exist for a large selection of peripheral equipment. These are normally not available for military or rugged computers.

It is also attractive to have a low-cost counterpart commercial machine to use in the lab while the rugged machine is working in the field. You can buy a Model 1601 and a Nova for less money than any comparable severe environment machine on the market today.

These are the reasons we took the Nova and built it to meet—among others—the environmental requirements of Mil-E-5400 for airborne equipment, Mil-E-16400 for shipboard equipment and Mil-E-4158 for ground electronic equipment. The result is the world's most versatile general purpose mini-computer, the Ruggednova

Gene Richeson, Vice President, Rolm Corporation.





Bettmann Archive

Pace-setting research in magnetic force was undertaken by William Gilbert early in the 17th century.

Configure a complete system ruggedly

The physical and technical aspects of the Ruggednova insure that you can configure a complete system, ruggedly.

The machine is packaged in a standard ATR box 7⅝″ x 10⅛″ x 12½″ which contains a power supply and slots for 14 circuit modules.

The Central Processing Unit occupies five slots, leaving room for options or user designed circuits within the basic box. The CPU is an implementation of the Nova architecture. One of the key features of the system is the utilization of four hardware full word accumulators, two of which may be used as index registers.

The 1601 has a program interrupt structure which permits automatic identification of up to 16 interrupting devices. And, the machine's input/output system allows the program to address up to 62 external devices.

The basic machine includes a Direct Memory Access channel which permits external devices to have direct access to memory on a cycle steal basis. The machine can directly address 1024 memory locations and has a total capacity of 32K words which may be core memory or MOS Read Only Memory, or any combination of both.

The core memory is modular, available in 4K x 16-bit increments which are packaged in 7⅝″ x 10⅛″ x 3¼″ modules. These modules literally plug into the basic box or to another core memory module.

There is provision for substantial expansion capability in the 1601, including up to five I/O interface circuit modules, and extended arithmetic option using a hardware multiply/divide and up to 4096 words of MOS Read Only Memory.

Rolm also offers a significant repertoire of standard I/O devices which plug into the basic system package. Our objective is to minimize the effort and cost in integrating the Ruggednova into a user's system, thus making the machine a workable, reliable component or building block for any system.

Dr. Robert Maxfield, Vice President, Rolm Corporation.

Sixteenth century woodcut of a contemporary scholar.

Bettmann Archive

Software, tested and proven

The beauty of the Ruggednova software is that it has been developed for almost two years. All the existing Nova software is available for use on our machine. And the selection of programs is continually growing.

What this means is that when you order your Ruggednova, you immediately have a library of field tested and proven software. In fact, arrangements can be made for short term rental of a commercial Nova which can be used by your programmer while you await delivery of your Model 1601, so you can get to work right away.

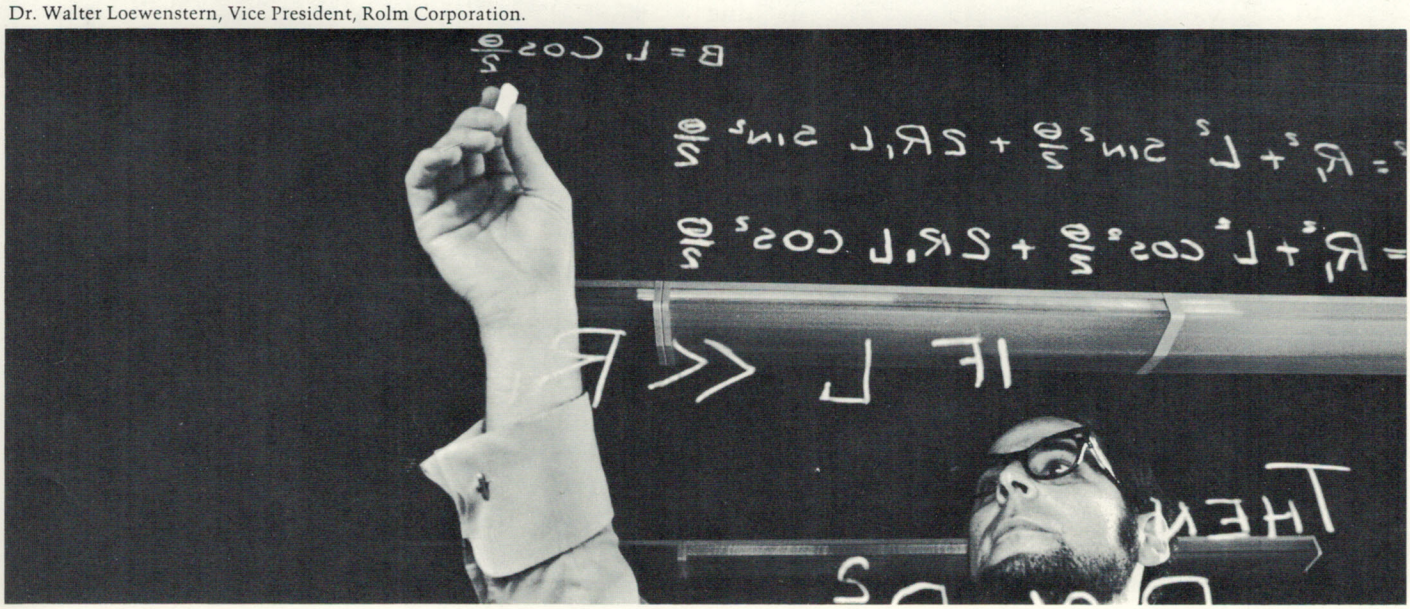
The software includes a powerful assembler, a context oriented text editor, a multiple breaking point debugger, complete hardware diagnostics, mathematical routines, including floating point arithmetic and utility routines. This list has been recently expanded to include Fortran IV, Algol 60 and operating systems for discs and mag tape.

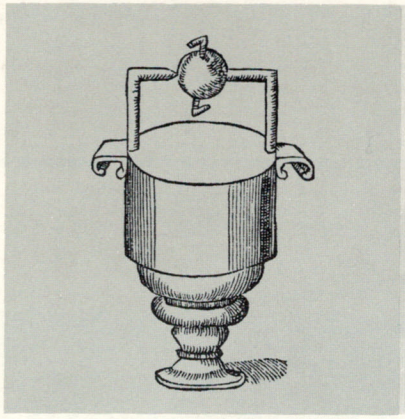
About pricing and delivery

The basic Ruggednova including the Central Processing Unit with four hardware accumulators, 4K memory, 16 level programmed priority interrupts, I/O capability for up to 62 devices, Direct Memory Access, Power Failure Protect and Auto Restart, Teletype interface and power supply is priced "off-the-shelf" at less than $20,000.

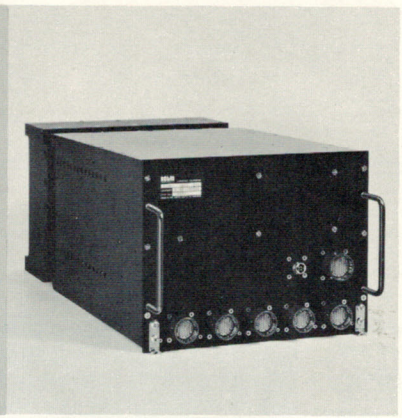
Production is underway in our new manufacturing plant located in Cupertino, California. We've been delivering Ruggednovas on time since March 1970. You'll have your machine 90 days after we receive your order—just try us.

Dr. Walter Loewenstern, Vice President, Rolm Corporation.





The aeolipile, the first working steam engine.



The Rolm Model 1601 Severe Environment Computer.

Rolm Model 1601 specifications and instructions

PRELIMINARY SPECIFICATIONS

The Rolm Model 1601 is a 16-bit word general purpose computer designed to meet severe environmental requirements of the Mil-Specs listed below. It is an implementation of the NOVA architecture under license from Data General Corporation. Its instruction set, I/0 interface and all software are identical and common with the NOVA.

The machine has four accumulators, two of which may be used as index registers. It offers a choice of core memory in 4K increments or MOS Read Only Memory in 256 word increments up to a total memory capacity of 32K 16-bit words (64K bytes). The Rolm Model 1601 comes in a standard ATR box 7⅝" x 10⅛" x 12½" but is also adaptable to rack mounting in a 19 inch RETMA rack with 60 Hz power. It weighs less than 40 lbs. including 4K core memory. There is ample room in the basic box for various standard or custom I/0 interface devices as well as the extended arithmetic option and MOS Read Only Memory. Core memory comes in plug-in modules of 4K words (8K bytes) each and literally plug into the basic box. The NOVA architecture has the most flexible I/0 facility ever built into a machine of its class. It includes a high speed Data Channel and automatic interrupt source identification, power monitor and auto restart and Mil-Spec power supply as standard equipment.

Electrical Specifications:

Power Requirements

Single phase, 400 cycle, 115 volt power capable of supplying approximately 2 amps in accordance with Mil-Std 704A.

Power Dissipation

70 watts with 4K memory and teletype interface.

150 watts with maximum I/0, ROM, Extended Arithmetic and 16K memory.

I/0 Bus Levels (TTL logic)

The Rolm Model 1601 is designed to meet the following environmental specifications:

Environmental Specifications:

Temperature: −55°C to +95°C Case temperature. Conductive cooling with no forced air or liquid coolant required.

Shock: 15g's, 11 milliseconds, 3 axes.

Vibration: Mil-E-5400 Curve IV, to 10g's.

Reliability: MTBF in excess of 11,000 hours. Calculated according to Mil Handbook 217A.

Humidity: 95% relative humidity.

RFI: Mil Standard 461A.

Altitude: 80,000 feet.

Explosive atmosphere, sand and dust, salt spray, and fungus resistance in accordance with Mil-E-5400K.

Quality Assurance: In accordance with Rolm quality assurance standards which are based upon the guidelines of Mil-Q-9858A.

INSTRUCTIONS

ARITHMETIC AND LOGICAL INSTRUCTIONS

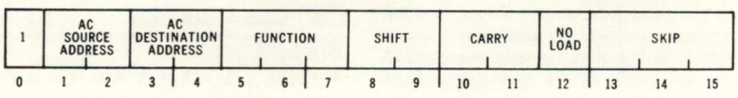
An instruction that has a 1 in bit 0 performs one of eight arithmetic and logical functions as specified by bits 5–7 of the instruction word. The function, which may be anything from a simple move to a subtraction, always uses the contents of the accumulator specified by bits 1 and 2; and if a second operand is required, it comes from the accumulator addressed by bits 3 and 4. The instruction also supplies a carry bit to the shifter with the result. Bits 10 and 11 specify a base value to be used in determining the carry bit. The instruction supplies either this value or its complement depending upon both the function being performed and the result it generates. The mnemonics and bit configurations and the base values they select are as follows.

Mnemonic	Bits 10–11	Base value for carry bit
	00	Current state of carry
Z	01	Zero
C	10	Complement of current state of carry
O	11	One

The three logical functions simply supply the listed value as the carry bit to the shifter. The five arithmetic functions supply the complement of the base value if the operation produces a carry out of bit 0; otherwise they supply the value given. The carry bit can be used in conjunction with the sign of the result to detect overflow in operations on signed numbers. But its primary use is as a carry out of the most significant bit in operations on unsigned numbers, such as the lower order parts in multiple precision arithmetic.

The 17-bit word consisting of the carry bit and the 16-bit result is operated on by the shifter as specified by bits 8 and 9.

Mnemonic	Bits 8–9	Shift operation
	00	None
L	01	Left rotate one place. Bit 0 is rotated into the carry position, the carry bit into bit 15
R	10	Right rotate one place. Bit 15 is rotated into the carry position, the carry bit into bit 0
S	11	Swap the halves of the 16-bit result. The carry bit is not affected



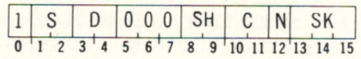
The shifter output is also tested for a skip according to the condition specified by bits 13–15. The processor skips the next instruction if the specified condition is satisfied.

Mnemonic	Bits 13–15	Skip function
	0	Never Skip
SKP	1	Always Skip
SZC	2	Skip on Zero Carry
SNC	3	Skip on Nonzero Carry
SZR	4	Skip on Zero Result
SNR	5	Skip on Nonzero Result
SEZ	6	Skip if Either Carry or Result is Zero
SBN	7	Skip if Both Carry and Result are Nonzero

Arithmetic and Logical Functions

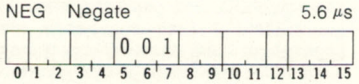
The eight functions are selected by bits 5–7 of the instruction word. For convenience the accumulators addressed by the S and D parts of the instruction are referred to as ACS and ACD.

COM Complement 5.6 μs



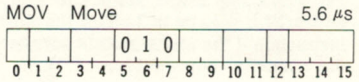
Place the (logical) complement of the word from ACS and the carry bit specified by C in the shifter. Perform the shift operation specified by SH. Load the shifter output in carry and ACD unless N is 1. Skip the next instruction if the shifter output satisfies the condition specified by SK.

NEG Negate 5.6 μs



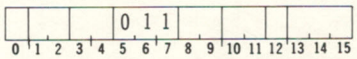
Place the twos complement of the number from ACS into the shifter. Perform the shift operation specified by SH. Load the shifter output in carry and ACD unless N is 1. Skip the next instruction if the shifter output satisfies the condition specified by SK.

MOV Move 5.6 μs



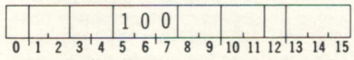
Place the contents of ACS and the carry bit specified by C in the shifter. Perform the shift operation specified by SH. Load the shifter output in carry and ACD unless N is 1. Skip the next instruction if the shifter output satisfies the condition specified by SK.

INC Increment 5.6 μs



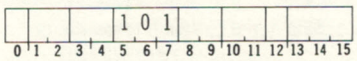
Add 1 to the number from ACS and place the result in the shifter. Perform the shift operation specified by SH. Load the shifter output in carry and ACD unless N is 1. Skip the next instruction if the shifter output satisfies the condition specified by SK.

ADC Add Complement 5.9 μs



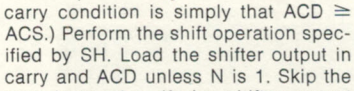
Add the (logical) complement of the number from ACS to the number from ACD, and place the result in the shifter. Perform the shift operation specified by SH. Load the shifter output in carry and ACD unless N is 1. Skip the next instruction if the shifter output satisfies the condition specified by SK.

SUB Subtract 5.9 μs



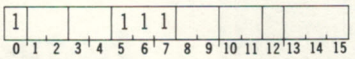
Subtract by adding the twos complement of the number from ACS to the number from ACD, and place the result in the shifter. If the signs of the operands are the same and ACD ≥ ACS, or the signs differ and ACD is negative, supply the complement of the value specified by C as the carry bit; otherwise supply the specified value. (For unsigned numbers the carry condition is simply that ACD ≥ ACS.) Perform the shift operation specified by SH. Load the shifter output in carry and ACD unless N is 1. Skip the next instruction if the shifter output satisfies the condition specified by SK.

ADD Add 5.9 μs



Add the number from ACS to the number from ACD, and place the result in the shifter. If both summands are negative, or their signs differ and their magnitudes are equal or the positive one is the greater in magnitude, supply the complement of the value specified by C as the carry bit; otherwise supply the specified value. (For unsigned numbers the carry condition is simply that the sum is ≥ 2^{16}.) Perform the shift operation specified by SH. Load the shifter output in carry and ACD unless N is 1. Skip the next instruction if the shifter output satisfies the condition specified by SK.

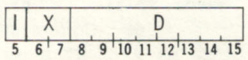
AND And 5.9 μs



Place the logical and function of the word from ACS and the word from ACD in the shifter. Supply the value specified by C as the carry bit. Perform the shift operation specified by SH. Load the shifter output in carry and ACD unless N is 1. Skip the next instruction if the shifter output satisfies the condition specified by SK.

MEMORY REFERENCE INSTRUCTIONS

Bits 5–15 have the same format in every memory reference instruction whether the effective address is used for storage or retrieval of an operand or to alter program flow. Bit 5 is the indirect bit, bits 6 and 7 are the index



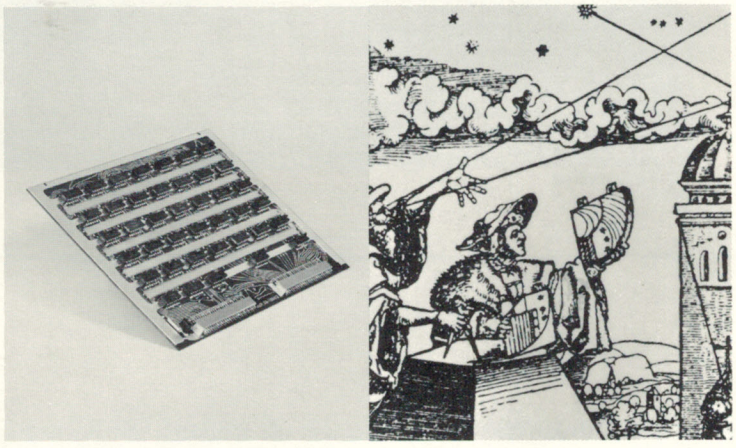
bits, and bits 8–15 are the displacement. The effective address E of the instruction depends on the values of I, X, and D. If X is 00, D addresses one of the first 256 memory locations, i.e., D is a memory address in the range 00000–00377. This group of locations is referred to as page zero.

If X is nonzero, D is a displacement that is used to produce a memory address by adding it to the contents of the register specified by X. The displacement is a signed binary integer in twos complement notation. Bit 8 is the sign (0 positive, 1 negative), and the integer is in the octal range −200 to +177 (decimal −128 to +127). If X is 01, the instruction addresses a location relative to its own position, i.e., D is added to the address in PC, which is the address of the instruction being executed. This is referred to as relative addressing. If X is 10 or 11 respectively, it selects AC2 or AC3 as a base register to which D is added.

X	Derivation of address
00	Page zero addressing. D is an address in the range 00000–00377.
01	Relative addressing. D is a signed displacement (−200 to +177) that is added to the address in PC.
10	Base register addressing. D is a signed displacement (−200 to +177) that is added to the address in AC2.
11	Base register addressing. D is a signed displacement (−200 to +177) that is added to the address in AC3.

If I is 0, addressing is direct, and the address already determined from X and D is the effective address used in the execution of the instruction. Thus

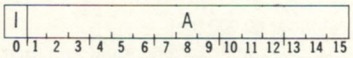
Bettmann Archive



Rolm printed circuit modules are conductively cooled.

16th century astronomer conducts an instruction class.

a memory reference instruction can directly address 1024 locations: 256 in page zero, and three sets of 256 in the octal range 200 less than to 177 greater than the addresses in PC, AC2 and AC3. If I is 1, addressing is indirect, and the processor retrieves another address from the location specified by the address already determined. In this new word bit 0 is the indirect bit: bits 1–15 are the effective address if bit 0 is 0; otherwise they specify a location



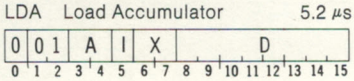
for yet another level of address retrieval. This process continues until some referenced location is found with a 0 in bit 0; bits 1–15 of this location are the effective address E.

If at any level in the effective address calculation an address word is fetched from locations 00020–00037, it is automatically incremented or decremented by one, and the new value is both written back in memory and used either as the effective address or for the next step in the calculation depending on whether bit 0 is 0 or 1. Addresses taken from locations 00020–00027 are incremented, those from locations 00030–00037 are decremented.

Move Data Instructions

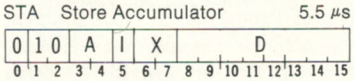
These two instructions move data between memory and the accumulators. In the descriptions of all memory reference instructions, *E* represents the effective address. The time given in the top line is for direct addressing, in page zero or relative to PC. Base register addressing requires an additional .3 μs; indirect addressing requires one extra memory cycle time per level; autoincrementing and autodecrementing require no extra time.

LDA Load Accumulator 5.2 μs



Load the contents of location E into accumulator A. The contents of E are unaffected, the original contents of A are lost.

STA Store Accumulator 5.5 μs

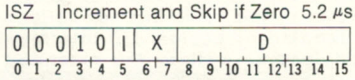


Store the contents of accumulator A in location E. The contents of A are unaffected, the original contents of E are lost.

Modify Memory Instructions

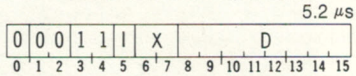
These two instructions alter a memory location and test the result for a skip. They are used to count loop iterations or successively modify a word for a series of operations.

ISZ Increment and Skip if Zero 5.2 μs



Add 1 to the contents of location E and place the result back in E. Skip the next instruction in sequence if the result is zero.

DSZ Decrement and Skip if Zero 5.2 μs

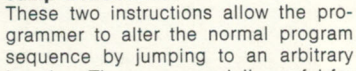


Subtract 1 from the contents of location E and place the result back in E. Skip the next instruction in sequence if the result is zero.

Jump Instructions

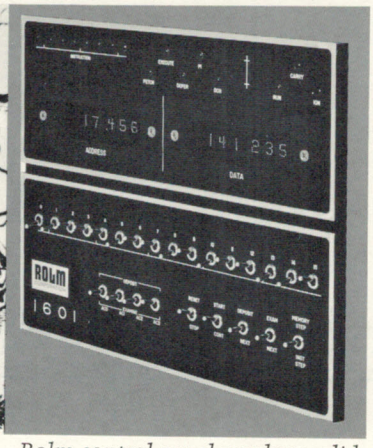
These two instructions allow the programmer to alter the normal program sequence by jumping to an arbitrary location. They are especially useful for calling and returning from subroutines.

JMP Jump 2.6 μs



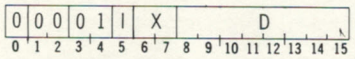
Load E into PC. Take the next instruction from location E and continue sequential operation from there.

16th century navigators practice azimuth plotting techniques.



Rolm control panel employs solid state light emitting diodes.

JSR Jump to Subroutine 3.5 μs



Load an address one greater than that in PC into AC3 (hence AC3 receives the address of the location following the JSR instruction). Load E into PC. Take the instruction from location E and continue sequential operation from there. The original contents of AC3 are lost.

INPUT-OUTPUT INSTRUCTIONS

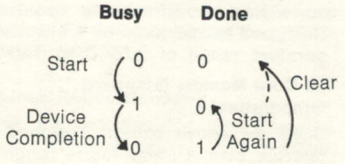
Instructions in the in-out class govern all transfers of data to and from the peripheral equipment, and also perform various operations within the processor. An instruction in this class is designated by 011 in bits 0–2. Bits 10–15 select the device that is to respond to the instruction. The format thus allows for 64 codes of which 62 can be used to address devices (octal 01–76). The code 00 is not used, and 77 is used for a number of special functions including reading the console data switches and controlling the program interrupt.

Every device has a 6-bit device selection network, an Interrupt Disable flag, and Busy and Done flags. The selection network decodes bits 10–15 of the instruction so that only the addressed device responds to signals sent by the processor over the in-out bus. The Busy and Done flags together denote the basic state of the device. When both are clear the device is idle. To place the device in operation, the program sets Busy. If the device will be used for output, the program must give a data-out instruction that sends the first unit of data — a word or character depending on how the device handles information. (The word "output" used without qualification always refers to the transfer of data from the processor to the peripheral equipment; "input" refers to the transfer in the opposite direction.) When the device has processed a unit of data, it clears Busy and sets Done to indicate that it is ready to receive new data for output, or that it has data ready for input. In the former case the program would respond with a data-out instruction to send more data; in the latter with a data-in instruction to bring in the data that is ready. If the Interrupt Disable flag is clear, the setting of Done signals the program by requesting an interrupt; if the program has set Interrupt Disable, then it must keep testing Done or Busy to determine when the device is ready.

In all in-out instructions bits 8 and 9 either control or sense Busy and Done. In those instructions in which bits 8 and 9 specify a control function, the mnemonics and bit configurations and the functions they select are as follows.

Mnemonic	Bits 8–9	Control function
	00	None
S	01	Start the device by clearing Done and setting Busy.
C	10	Clear both Busy and Done, idling the device.
P	11	Pulse the special in-out bus control line — the effect, if any, depends on the device.

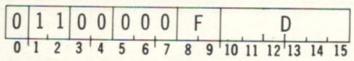
The overall sequence of Busy and Done states is determined by both the program and the internal operation of the device.



The data-in or data-out instruction that the program gives in response to the setting of Done can also restart the device. When all the data has been transferred the program generally clears Done so the device neither requests further interrupts nor appears to be in use, but this is not necessary. Busy and Done both set is a meaningless situation.

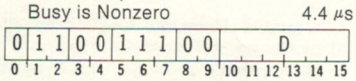
Bits 5–9 specify the complete function to be performed. If there is no transfer (bits 5–7 all alike), bits 3 and 4 are ignored and bits 8 and 9 may specify a control function or a skip condition.

NIO No I/O Transfer 4.4 μs



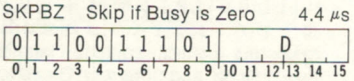
Perform the control function specified by *F* in device *D*.

SKPBN Skip if Busy is Nonzero 4.4 μs



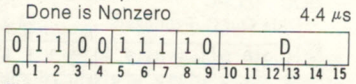
Skip the next instruction in sequence if the Busy flag in device *D* is 1.

SKPBZ Skip if Busy is Zero 4.4 μs



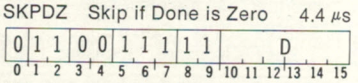
Skip the next instruction in sequence if the Busy flag in device *D* is 0.

SKPDN Skip if Done is Nonzero 4.4 μs



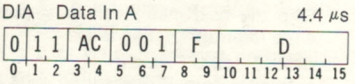
Skip the next instruction in sequence if the Done flag in device *D* is 1.

SKPDZ Skip if Done is Zero 4.4 μs



Skip the next instruction in sequence if the Done flag in device *D* is 0.

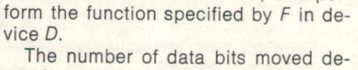
DIA Data In A 4.4 μs



Move the contents of the A buffer in device *D* to accumulator *AC*, and perform the function specified by *F* in device *D*.

The number of data bits moved depends on the size of the device buffer, its mode of operation, etc. Bits in *AC* that do not receive data are cleared.

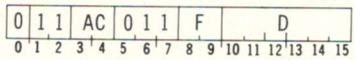
DOA Data Out A 4.7 μs



Send the contents of accumulator *AC* to the A buffer in device *D*, and perform the function specified by *F* in device *D*.

The amount of data actually accepted by the device depends on the size of its buffer, its mode of operation, etc. The original contents of *AC* are unaffected.

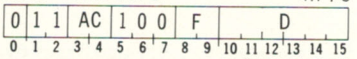
DIB Data in B 4.4 μs



Move the contents of the B buffer in device *D* to accumulator *AC*, and perform the function specified by *F* in device *D*.

The number of data bits moved depends on the size of the device buffer, its mode of operation, etc. Bits in *AC* that do not receive data are cleared.

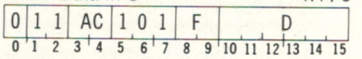
DOB Data Out B 4.7 μs



Send the contents of accumulator *AC* to the B buffer in device *D*, and perform the function specified by *F* in device *D*.

The amount of data actually accepted by the device depends on the size of its buffer, its mode of operation, etc. The original contents of *AC* are unaffected.

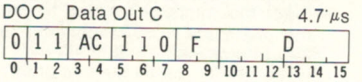
DIC Data in C 4.4 μs



Move the contents of the C buffer in device *D* to accumulator *AC*, and perform the function specified by *F* in device *D*.

The number of data bits moved depends on the size of the device buffer, its mode of operation, etc. Bits in *AC* that do not receive data are cleared.

DOC Data Out C 4.7 μs



Send the contents of accumulator *AC* to the C buffer in device *D*, and perform the function specified by *F* in device *D*.

The amount of data actually accepted by the device depends on the size of its buffer, its mode of operation, etc. The original contents of *AC* are unaffected.

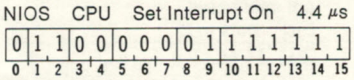
Special Code-77 Functions

In-out instructions with the code 77 in bits 10–15 perform a number of special functions rather than controlling a specific device. In all but the skip instructions bits 8 and 9 are used to turn the interrupt on and off. The mnemonics are the same as those for controlling Busy and Done in I/O devices, but with code 77 they select the following special functions.

Mnemonic	Function
S	Set the Interrupt On flag to enable the processor to respond to interrupt requests.
C	Clear the Interrupt On flag to prevent the processor from responding to interrupt requests.
P	None.

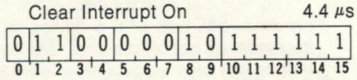
Rolm Model 1601 options

NIOS CPU Set Interrupt On 4.4 μs



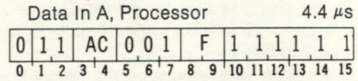
Set the Interrupt On flag to allow the processor to respond to interrupt requests.

NIOC CPU
Clear Interrupt On 4.4 μs



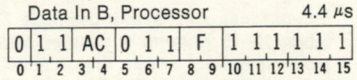
Clear the Interrupt On flag to prevent the processor from responding to interrupt requests.

DIA *AC*,CPU
Data In A, Processor 4.4 μs



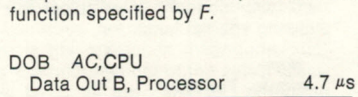
Read the contents of the console data switches into accumulator *AC*, and perform the function specified by *F*.

DIB *AC*,CPU
Data In B, Processor 4.4 μs



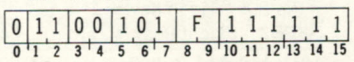
Place in accumulator *AC* the device code of the first device on the bus that is requesting an interrupt ("first" means the one that is physically closest to the processor on the bus.) Perform the function specified by *F*.

DOB *AC*,CPU
Data Out B, Processor 4.7 μs



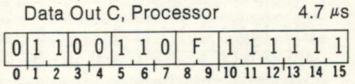
Set up the Interrupt Disable flags in the devices according to the mask in accumulator *AC*. For this purpose each device is connected to a given data line, and its flag is set or cleared as the corresponding bit in the mask is 1 or 0. Perform the function specified by *F*.

DIC 0,CPU
Data In C, Processor 4.4 μs



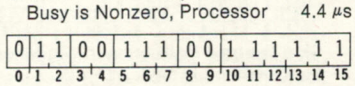
Clear the control flipflops, including Busy, Done and Interrupt Disable, in all devices connected to the bus. Perform the function specified by *F*.

DOC 0,CPU
Data Out C, Processor 4.7 μs



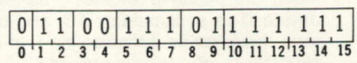
Perform the function specified by *F*. and then halt the processor.

SKPBN CPU Skip if
Busy is Nonzero, Processor 4.4 μs



Skip the next instruction in sequence if the Interrupt On flag is 1.

SKPBZ CPU Skip if
Busy is Zero, Processor 4.4 μs



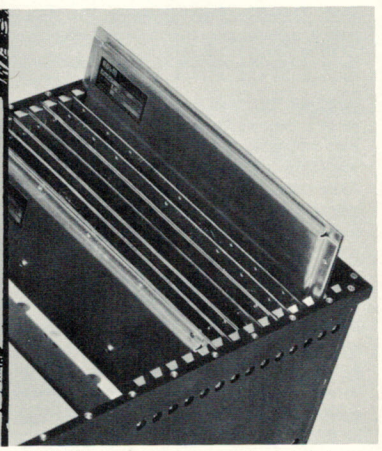
Skip the next instruction in sequence if Interrupt On is 0.

The assembler recognizes a number of convenient mnemonics for instructions with device code 77.

Mnemonic	Meaning	Mnemonic Equivalent		Octal Equivalent
READS	Read Switches	DIA	—,CPU	060477
IORST	IO Reset	DICC	0,CPU	062677
HALT	Halt	DOC	0,CPU	063077
INTEN	Interrupt Enable	NIOS	CPU	060177
INTDS	Interrupt Disable	NIOC	CPU	060277
INTA	Interrupt Acknowledge	DIB	—,CPU	061477
MSKO	Mask Out	DOB	—,CPU	062077

Bettmann Archive

Woodcut depicting an early 18th century thermometer.



1601 has space to accommodate 14 printed circuit modules.

Extended Arithmetic

The extended arithmetic unit provides hardware 16-bit signed multiply and divide in 9.7 μsec each and n-bit shift (n=1, . . . , 16) in 4.7 μsec. This option requires two circuit modules and plugs into the basic ATR box.

4K Core Memory (wide temperature) (4096 words of 16 bits each)

This option includes 4K words (8K bytes) of wide temperature magnetic core memory with all necessary electronics mounted within a modular subassembly. The memory can be plugged directly into the basic box or another identical memory module with no wiring modifications required. Designed to operate over a case temperature range of −55°C to +95°C.

4K Core Memory (standard temperature)

Same as above except designed to operate over a temperature range of 0°C to +50°C. Still meets severe shock and vibration requirements.

MOS-Read Only Memory

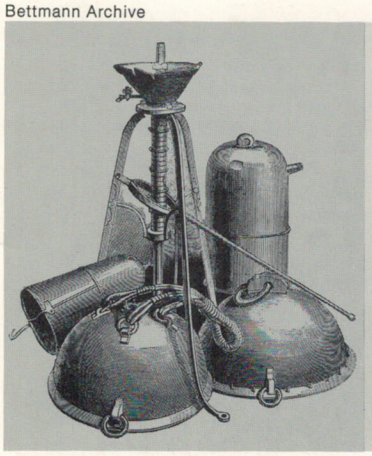
Metal Oxide Semiconductor (MOS) read only memory (ROM) is available in increments of 256 sixteen bit words with up to 8 increments or 2048 words on a circuit module. There is provision for 2 MOS ROM modules (4K words) in the basic package (4 modules without Extended Arithmetic).

Custom MOS-ROM. User provides tape or listing of each 256 word increment and pays non-recurring mask charge for first unit. Additional copies of some 256 word increments are then standard MOS-ROM units.

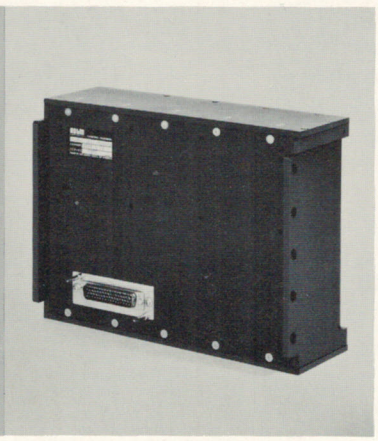
Standard MOS-ROM. Certain 256 word increments are available without non-recurring mask charge and the list will continue to grow. In order of availability standard ROM includes:

The following illustrations reprinted from "The Discovery of Nature" by Albert Bettex by permission of Droemer-Knaur Publishing Company, Munich-Zurich: Cover: Cosmic Catastrophe, Buffon's theory of the beginning of the universe; woodcut of Archimedes; perpetuum mobile; 18th century rocket; aeolipile.

Bettmann Archive



Equipment used by von Guericke for air pressure experiment.



Core memory modules are available in 4K increments.

1. Bootstrap and binary loader; BCD to binary and binary to BCD utility routines.

2.- 5. Diagnostics to perform built-in self test function.

6.- 9. Floating Point Interpreter (4 increments or 1024 words).

10.-17. Extended Floating Point Interpreter (8 increments).

Real Time Clock

This option provides a flag which can be enabled by the program to provide a program interrupt at a fixed frequency. A crystal clock may be used as the time source.

Control Panel (Commercial Type)

This control panel is functionally identical to the control panel for the commercial NOVA computer. It mounts in a 19-inch Retma rack and is attached to the Model 1601 by a cable and 85-pin connector. Its construction is suited for commercial or laboratory type environment.

Control Panel (Mil Spec)

This panel is also functionally identical to the above except that the 16-bit address and 16-bit data registers are presented in octal form. All indicators including the octal indicators are high reliability hermetically sealed light emitting diode (LED) type. This panel may be remotely located or used as a portable panel for trouble shooting and maintenance.

Power Converter

This device provides for 60 Hz to 400 Hz power conversion and enables the Model 1601 to be operated from 60 Hz power.

19-Inch Retma Mounting

This kit provides for mounting the basic ATR-type box or I/0 expansion box, and up to 12K core memory in a standard 19-inch Retma rack. It requires 8¾" of front panel space.

Expansion Chassis

This is an ATR type box similar to the basic box of the Model 1601. It contains a power supply and slots for up to 14 I/0 options. It is useful for system configurations which require more I/0 modules than the 5 provided for in the basic box.

I/0 Options

To minimize the design effort and cost required to interface the Model 1601 with a system, a large and continuously growing repertoire of I/0 options are available. These options are implemented on the standard circuit modules, and as such are designed to plug into the basic box. The present list includes:

1. 3 x 16 bit parallel output buffer, logic levels.
2. 3 x 16 bits unbuffered parallel input, logic levels.
3. 16 bit parallel output buffer, line driver.
4. 16 bit parallel input buffer, line receiver.
5. Serial I/0 double buffer, logic and line levels.
6. 16 bit parallel output buffer, contact closures.
7. 16 system interrupts, wired priority, individual arming.
8. D/A converter in 8, 10, or 12 bit accuracies.
9. A/D converter in 8, 10 or 12 bit accuracies.
10. Analog multiplexer, 16 channels random access or cyclic.
11. I/0 bus line drivers.
12. I/0 bus line receivers.

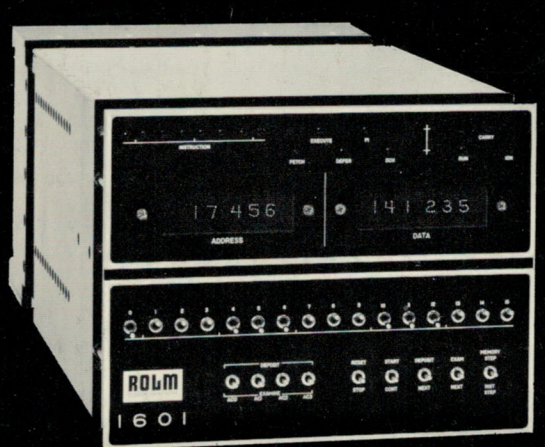
Peripheral Controllers

The following available controllers interface the Model 1601 with various standard or user supplied peripheral devices. Some of these peripheral controllers are for standard NOVA peripheral equipment, others are not. They are all implemented on circuit modules which plug into the basic box or expansion chassis.

1. Data channel controller—for interfacing high speed devices to the Model 1601 via the direct memory access (DMA).
2. Multiplexing controller—for operating several slower speed devices simultaneously from the direct memory access.
3. Teletype controller—for interfacing with the KSR-33, ASR-33, KSR-35, ASR-35.
4. Teletype controller—for interfacing with the KSR-37 or ASR-37.
5. Paper tape reader—300 characters per sec.
6. Paper tape punch—63.3 characters per sec.
7. Card reader—225 or 400 cards per minute.
8. Incremental plotter—300 points per sec.
9. Data phone controller—for Bell type 201-2400 pps data phone.
10. Line printer—300 lines per minute.
11. Disc controller—for data transfers through data channel for fixed head disc units of 32K, 64K or 128K words.
12. Magnetic tape controller—for interfacing and controlling up to 4 magnetic tape transports, in 9-track IBM—compatible format at 24, 75 or 150 ips.
13. IRIG—Time Code Translator provides for reading a standard IRIG serial TCG format.

ROLM
CORPORATION

10300 N. TANTAU AVENUE, CUPERTINO, CALIF. 95014 (408) 257-6440 TWX: 910-338-0247



The world's toughest mini-computer goes anywhere