J. L. DIVILBISS
Associate Professor
Graduate School of Library Science
University of Illinois
Urbana-Champaign

A Hardware Tutorial

A BRIEF HISTORY OF THE MINICOMPUTER

In a practical sense, the minicomputer age began in 1964 with the Digital Equipment Corporation's introduction of the PDP 8. Potential computer users who had been unable to afford a $500,000 machine found that for the then remarkably low price of $27,000 they could purchase a general purpose computer, limited in power, to be sure, but nonetheless a real computer. The price was achieved by a combination of several factors: a simple, classical, no-frills, logical design; a superior packaging technique for the electronic circuits; and, most important, use of a short word length. Most of the large computers of that era were using word lengths ranging from about 30 to 50 bits, the length being influenced by considerations of accuracy and instruction format. By cutting the word length to 12 bits, DEC was able to greatly reduce the hardware needed in the arithmetic unit. A short word length limits neither accuracy nor type of operations performed, but it often means that computation proceeds more slowly. For example, numerical computations may require cumbersome multiple precision routines in order to secure adequate accuracy. In simple terms, short word machines achieve low hardware costs at the expense of execution efficiency. Since there are many applications in which the slowest computer is still much faster than the application requires (e.g., a computer controlling a lathe) the loss of execution efficiency may not be important.

By 1966 DEC had sold 500 PDP 8s, and computer users were fiercely debating the issue of small dedicated computers versus timesharing systems. The controversy is by no means resolved, but there are many applications in which the two approaches are essentially noncompetitive. And, as Corey will make clear in the next paper, many of the configurations in use today combine features of the two approaches.

In 1969 the trade press began referring to small computers as "minicomputers"–terminology that may well have been influenced by the appearance of miniskirts on the fashion scene.

By 1970 DEC had sold 8,500 PDP 8s and the company's growth from three employees to over 6,000 provided a model for numerous competitors to emulate. Very often, a bright engineer with ambition would leave his computer-manufacturer employer, hire a few friends, borrow some capital and launch himself into the computer business. As evidence of this activity, thirty-six new brands and models of minicomputers were introduced in 1970. Most of these machines were perfectly respectable computers but with relatively little to differentiate brand X from brand Y.

The year 1971 was a rather cheerless one for the electronics industry as the abrupt reduction in government support of the space program brought widespread unemployment and business slowdowns. The one bright light on the economic horizon was the burgeoning minicomputer industry. The small size and low cost of the minis opened up a host of new applications ranging from the mundane (controlling traffic lights) to the exotic (voice recognition systems). In this phase of the development we see an interesting regenerative cause and effect relationship. New applications increase sales; increased sales permit production economics and lower prices; lower prices stimulate new applications. In mid-1974 we appear still to be in the regenerative development phase.

In 1972 Computer Automation, Inc. introduced a full-scale computer for less than $1,000. Of course, to achieve that price the NAKED MINI was marketed without cabinet and power supply (hence, the name), but that was entirely appropriate since these machines were intended for incorporation into larger pieces of electronic equipment. Once the $1,000 barrier was broken, it was easier to see the computer as a *component* (albeit a very sophisticated one) in a larger system. The NAKED MINI was also sold as a complete, free-standing computer (i.e., with cabinet and power supplies) for about $2,000.

Minicomputers are usually defined as being physically small, short word machines selling for less than a specified amount, i.e., $20,000. This seems to be a reasonable enough definition, but recent trends in hardware design–e.g., toward longer words–have confused the issue. The 12-bit word of the PDP 8 *is* restrictive in terms of instruction format and addressing, and thus, 16 bits has become a more popular word length. About 50 manufacturers make 16-bit machines, a handful make 18- and 24-bit machines. In 1973 Interdata announced a 32-bit computer with up to one million bytes of directly addressable core memory. This machine looks like a mini and with 32K bytes of core is priced like a mini at $9,950. On the other hand, it can be purchased with a million bytes of core for $171,650, a price well outside the range usually associated with minicomputers. Note also that the 32-bit word length is the same as that used in the IBM 360/370 series. This blurring of the distinction between minicomputers and large computers has resulted in the bizarre expression "mega-mini" to describe the larger minis.

The year 1973 was also one in which the POS (point of sale) market started having a substantial impact on the computer industry. Retailers began to see that by using computer terminals in place of cash registers they could gather more complete and timely data for better control. As an example, a typical grocery store computer terminal costs about the same as a mechanical cash register (about $2,500) but has a much more extensive and flexible set of functions, such as check verification. Typically, all the terminals in a store are connected to a minicomputer on the premises.

Many packaged food items now carry machine-readable uniform product codes in the form of a bar code printed on the package. Eventually most of the data entered at checkout stations will probably be from machine-readable labels rather than from hand keying. Libraries clearly should profit from this trend since input devices (such as light pens) developed for POS will be adaptable for library activities such as circulation. Although light pens are already in use in libraries, the widespread use of POS terminals should result in cheaper, more reliable equipment and some desirable standardization of data formats.

In the brief history given here I have tried to include examples illustrating significant trends in minicomputer development: minis are getting smaller, they are getting larger, they are becoming more versatile, and they are becoming more specialized. Thus the minicomputer manufacturers are attempting to fill every economic niche, much as Darwin's finches diversified to fill every ecological niche. One might call this adaptive radiation in the marketplace.

WHAT'S NEW IN HARDWARE

Addressing Schemes

One of the more conspicuous trends in computer hardware is that memories have gotten cheaper, larger and more complex in their organization.

For minicomputers, the increase in memory size has exacerbated a problem alluded to earlier—that a short instruction word just does not provide enough bits to specify a large number of addresses. Minicomputer designers have gotten around this problem with a variety of ingenious techniques. The address portion of the instruction may give an address directly, or relative to the present address, or within a "page" or specifying a new page, or where the real address can be found, etc. The last method is called indirect addressing and should be especially meaningful to librarians. When a librarian encounters a tracing card in the catalog with the rubber-stamped message "FOR LOCATION OF COPIES SEE AUTHOR OR MAIN ENTRY CARD," he or she is experiencing indirect addressing. That is, the tracing does not give the location of the book, it gives the location of the card bearing the location information.

A nonexhaustive search of manufacturers' literature reveals the Lockheed SUE as the champion in terms of addressing schemes. The SUE provides *seventeen* different addressing modes. This wealth of addressing schemes makes possible very clever and powerful programs, but it does nothing to simplify the work of the programmer. Corey will amplify this point in the following paper.

IBM's recent introduction of virtual storage in their machines might lead the casual observer to believe that VS had just been invented. Actually, VS goes back to a 1960 University of Manchester development and has been incorporated in large computers such as the Burroughs machines for several years. VS is just now beginning to make an appearance in minicomputers but there is every reason to feel that it will be a significant factor in the near future. Basically, VS is a system of automatically swapping information back and forth between disk and core so that the application program can access the entire disc memory space as if it were core. A simple example may help to clarify this. Imagine a small computer with 8 "pages" of core where a page is defined as 1,024 words. A program written for this machine might occupy 32 pages of memory with 8 pages in core and the other 24 on disk. During the execution of the program, memory access instructions covering the entire 32-page region are executed. If an address refers to one of the pages in core the program proceeds normally. If the program calls for an address from one of the disk pages, a "page fault" condition is raised. The VS system suspends execution of the program just long enough to transfer the needed page from disk to core. Since one of the core pages is overwritten by this operation, it is necessary to save that page by transferring it to disk before making the disk to core transfer. It is clear which page needs to be brought in from disk, but how does the VS system decide which core page is to be overwritten? The simplest and most common algorithm is to overwrite the *least recently used* page.

In principle, all this could be done on almost any machine by just writing a sufficiently clever operating system. Actually, VS is not feasible unless a good part of the work is done by hardware designed for the various VS functions. As an example, the *least recently used* function is more efficiently done with a hardware push-down stack than with a program.

The advantages associated with VS are substantial. For one thing, programmers can be freed from the grubby details of managing external files. Programmers generally would like their machine to have more core; VS does not give them more core, but it makes it possible for them to write their programs as if they had more core. A second advantage is that VS can minimize problems associated with machine expansion. Consider the previously cited machine having 8 pages of core and running a 32-page program. If an additional 8 pages of core is purchased it will be unnecessary to make any changes in the program. Independently of the amount of actual core, the program will run in 32 pages of virtual storage. The only effect on the program will be faster execution since fewer disc swaps will be needed.

Input/Output

In early DEC literature the block diagram of the PDP 8 shows a single Teletype hanging on the accumulator. The Teletype was and is a modestly priced device suitable for limited I/O service. For PDP 8s used in classical scientific programming applications the Teletype was an adequate if not elegant solution to the problem of holding the cost down. Since that time minis have been used in a wide range of applications that require more sophisticated I/O facilities. The typical minicomputer sold today has an I/O bus that can accommodate up to 256 external devices. The bus can be thought of as a kind of party line (i.e., all the external devices are tied to the same set of wires) with a set of address lines to select a particular device and another set of lines for the data. Thus, when the CPU causes a message to be printed it merely specifies the printer on the address lines and the data on the data lines. But how does the input device get the attention of the CPU? The primitive way to handle this is to write a program that constantly interrogates the external device to see if it has anything to say. Thus, at some point in a circulation transaction the CPU would repeatedly ask the badge reader "Do you have another digit to give me?" Of course, the badge reader is pretty slow and the CPU is not getting much else done during this process.

The solution to this problem is provision of a hardware interrupt feature so that external devices can interrupt the program by causing a transfer of control to a special I/O routine. Following the interrupt, the interrupted program resumes. Very elaborate interrupt priority schemes are now available that permit device A to interrupt a background program, device B to interrupt device A, device C to interrupt device B, etc. An optional feature on some machines is an interrupt generated by a power failure. This interrupt is at the top of the priority list and transfers control to a special "tidying up" section of code that gets the machine ready for the return of power. Because of this, a lightning bolt in the neighborhood will not necessitate reloading the program. (The CPU has about 100 microseconds warning on a power failure, plenty of time to straighten out its affairs.)

Programmed data transfer, where the CPU controls I/O on a byte-by-byte basis, is fine for mechanical devices such as typewriters but is too slow for communication with high-speed devices such as magnetic tape, disc drives and other computers. With such high-speed devices the objective is usually to transfer fairly large blocks of information. To facilitate high-speed block transfers, many minicomputers now offer direct memory access as an optional extra. The DMA hardware is essentially a microcontroller interposed between the memory and the I/O bus. If a program requires that a block of data be transferred from core to disc (page swapping for example), the CPU transfers to the DMA the starting address in memory and the block size. Transfer of information can then proceed without executing an instruction for each word. Instead, each time the disc is able to accept a word, the DMA suspends execution of the main program for one memory cycle and uses that cycle to fetch the appropriate word. That way, transfer can proceed at a rate determined by the external device and at the same time interfere minimally with normal program execution. The trick of using a single memory cycle (which might come in the middle of an instruction) for this purpose is generally called "cycle stealing." Typical DMA transfer rates range up to about 1 million words/second.

Instructions

One of the most effective ways for minicomputer manufacturers to compete is through the introduction of extensive and versatile instruction sets. As an example, the NAKED MINI can read punched paper tape with a two-instruction routine whereas the comparable NOVA requires a 24-instruction routine. Of course, it is not a question of how big the instruction set is, but rather how well it fits the application.

There are basically two ways of incorporating a set of instructions into a machine. The traditional method is to *wire* into the control unit the exact sequence of gating signals needed for each instruction. In other words, the instruction set is inextricably tied to the hardware design. In the *microprogramming* approach, a special read only memory *control store* is used to define the gating sequences. The execution of a particular instruction calls up a small section of the very high-speed ROM, and the control code for that instruction is simply a sequence of gating signals.

The most conspicuous advantage to microprogramming is that the instruction set of a particular machine can be tailored to the application. This may mean inclusion of fancy instructions (table search, error code generation and checking, etc.), and it can also mean adapting the instruction set to fit a particular compiler. At least one minicomputer is marketed in which a high-level language is available because it "fits" the instruction set.

At this writing the ultimate in microprogrammable machines is one in which a writable control store permits the end user of the computer to define the instruction set. Again, as Corey's paper will make clear, all this power and versatility does not necessarily make the computer easier to program.

Peripherals

While it is true that the price of minicomputers has dropped sharply in recent years, the quoted low price of a computer may be misleading if "computer" refers to just the CPU. It is not uncommon for I/O and external storage devices to bring the total cost of a system up to triple the cost of the CPU. The high cost of peripherals has tended to limit applications of minicomputers, especially when peripherals designed primarily for large computers were used. The problem with standard, large-machine peripherals is that capacity, complexity and cost are all inappropriate for small, low-cost machines. Not too surprisingly, a wide range of peripherals developed expressly for minicomputers has appeared in recent months. Some of these devices are just simpler, slower versions of standard hardware–line printers capable of a few hundred lines/minute are a good example. Other peripherals developed for the minicomputer market do not have parallels in the big machine market and deserve brief mention.

As little as a few hundred dollars will now purchase a "floppy disk" system with a storage capacity in the few million bit range. The disk itself is a mylar disk housed in either an envelope or a small carton; cost of the disk is as little as $4.

Cassette tape transports use Phillips-type cassettes that superficially resemble the cassettes used in home tape recorders. Actually, the only real difference between audio cassettes and computer cassettes is that the latter are made to much higher quality standards. Cassette transports range from a few hundred to a few thousand dollars; storage on a cassette can range up to seven million bits. (In more graphic terms, a single shirtpocket-sized cassette might hold as much information as five boxes of tab cards.)

Several manufacturers offer tape cartridge systems that are rather diverse in terms of speed, capacity and cost. Comparisons between the various systems are made on the basis of these factors and many other details of construction such as physical size, adaptability to automatic changing, protection from dust, etc.