

Chapter 39

Parallel operation in the Control Data 6600¹

James E. Thornton

History

In the summer of 1960, Control Data began a project which culminated October, 1964 in the delivery of the first 6600 Computer. In 1960 it was apparent that brute force circuit performance and parallel operation were the two main approaches to any advanced computer.

This paper presents some of the considerations having to do with the parallel operations in the 6600. A most important and fortunate event coincided with the beginning of the 6600 project. This was the appearance of the high-speed silicon transistor, which survived early difficulties to become the basis for a nice jump in circuit performance.

System organization

The computing system envisioned in that project, and now called the 6600, paid special attention to two kinds of use, the very large scientific problem and the time sharing of smaller problems. For the large problem, a high-speed floating point central processor with access to a large central memory was obvious. Not so obvious, but important to the 6600 system idea, was the isolation of this central arithmetic from any peripheral activity.

It was from this general line of reasoning that the idea of a multiplicity of peripheral processors was formed (Fig. 1). Ten such peripheral processors have access to the central memory on one side and the peripheral channels on the other. The executive control of the system is always in one of these peripheral processors, with the others operating on assigned peripheral or control tasks. All ten processors have access to twelve input-output channels and may "change hands," monitor channel activity, and perform other related jobs. These processors have access to central memory, and may pursue independent transfers to and from this memory.

Each of the ten peripheral processors contains its own memory for program and buffer areas, thereby isolating and protecting the

more critical system control operations in the separate processors. The central processor operates from the central memory with relocating register and file protection for each program in central memory.

Peripheral and control processors

The peripheral and control processors are housed in one chassis of the main frame. Each processor contains 4096 memory words of 12 bits length. There are 12- and 24-bit instruction formats to provide for direct, indirect, and relative addressing. Instructions provide logical, addition, subtraction, shift, and conditional branching. Instructions also provide single word or block transfers to and from any of twelve peripheral channels, and single word or block transfers to and from central memory. Central memory words of 60 bits length are assembled from five consecutive peripheral words. Each processor has instructions to interrupt the central processor and to monitor the central program address.

To get this much processing power with reasonable economy and space, a time-sharing design was adopted (Fig. 2). This design contains a register "barrel" around which is moving the dynamic information for all ten processors. Such things as program address, accumulator contents, and other pieces of information totalling 52 bits are shifted around the barrel. Each complete trip around requires one major cycle or one thousand nanoseconds. A "slot" in the barrel contains adders, assembly networks, distribution network, and interconnections to perform one step of any peripheral instruction. The time to perform this step or, in other words, the time through the slot, is one minor cycle or one hundred nanoseconds. Each of the ten processors, therefore, is allowed one minor cycle of every ten to perform one of its steps. A peripheral instruction may require one or more of these steps, depending on the kind of instruction.

In effect, the single arithmetic and the single distribution and assembly network are made to appear as ten. Only the memories are kept truly independent. Incidentally, the memory read-write cycle time is equal to one complete trip around the barrel, or one thousand nanoseconds.

¹AFIPS Proc. FJCC, pt. 2 vol. 26, pp. 33-40, 1964.

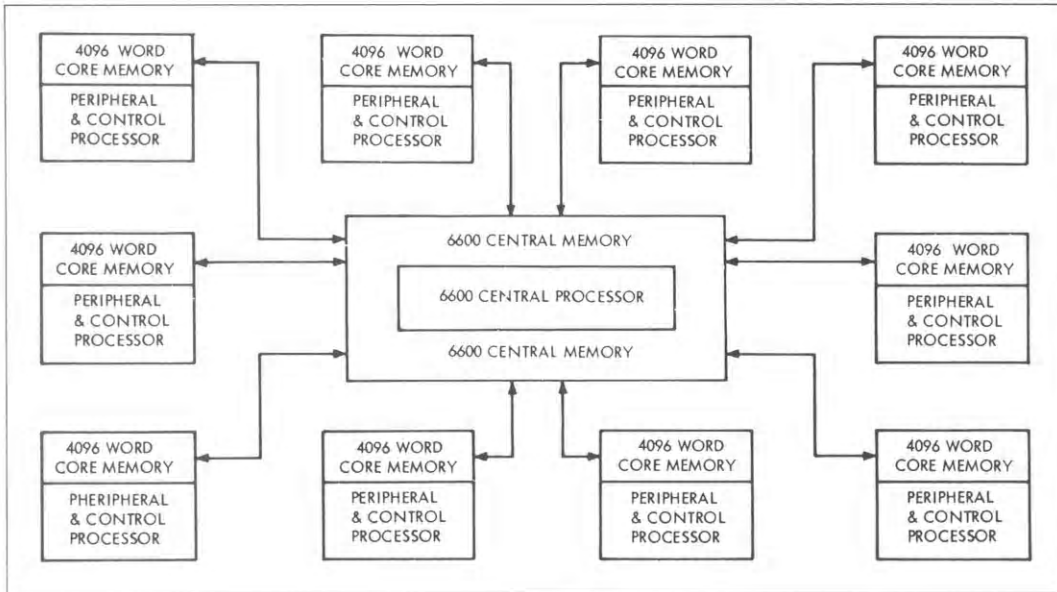


Fig. 1. Control Data 6600.

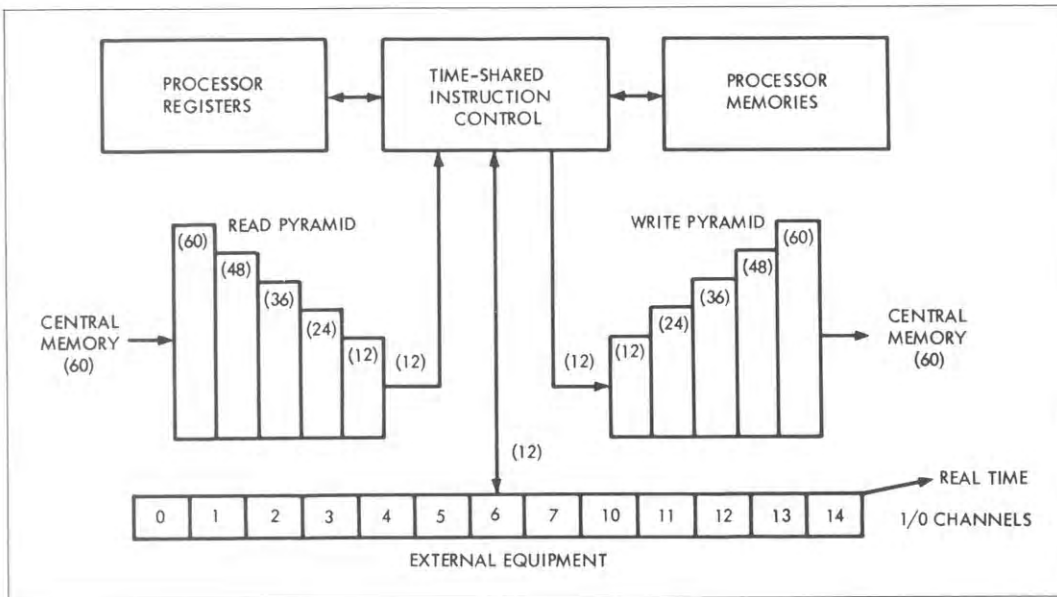


Fig. 2. 6600 peripheral and control processors.

Input-output channels are bi-directional, 12-bit paths. One 12-bit word may move in one direction every major cycle, or 1000 nanoseconds, on each channel. Therefore, a maximum burst rate of 120 million bits per second is possible using all ten peripheral processors. A sustained rate of about 50 million bits per second can be maintained in a practical operating system. Each channel may service several peripheral devices and may interface to other systems, such as satellite computers.

Peripheral and control processors access central memory through an assembly network and a dis-assembly network. Since five peripheral memory references are required to make up one central memory word, a natural assembly network of five levels is used. This allows five references to be "nested" in each network during any major cycle. The central memory is organized in independent banks with the ability to transfer central words every minor cycle. The peripheral processors, therefore, introduce at most about 2% interference at the central memory address control.

A single real time clock, continuously running, is available to all peripheral processors.

Central processor

The 6600 central processor may be considered the high-speed arithmetic unit of the system (Fig. 3). Its program, operands, and results are held in the central memory. It has no connection to the peripheral processors except through memory and except for two single controls. These are the exchange jump, which starts or interrupts the central processor from a peripheral processor, and the central program address which can be monitored by a peripheral processor.

A key description of the 6600 central processor, as you will see in later discussion, is "parallel by function." This means that a number of arithmetic functions may be performed concurrently. To this end, there are ten functional units within the central

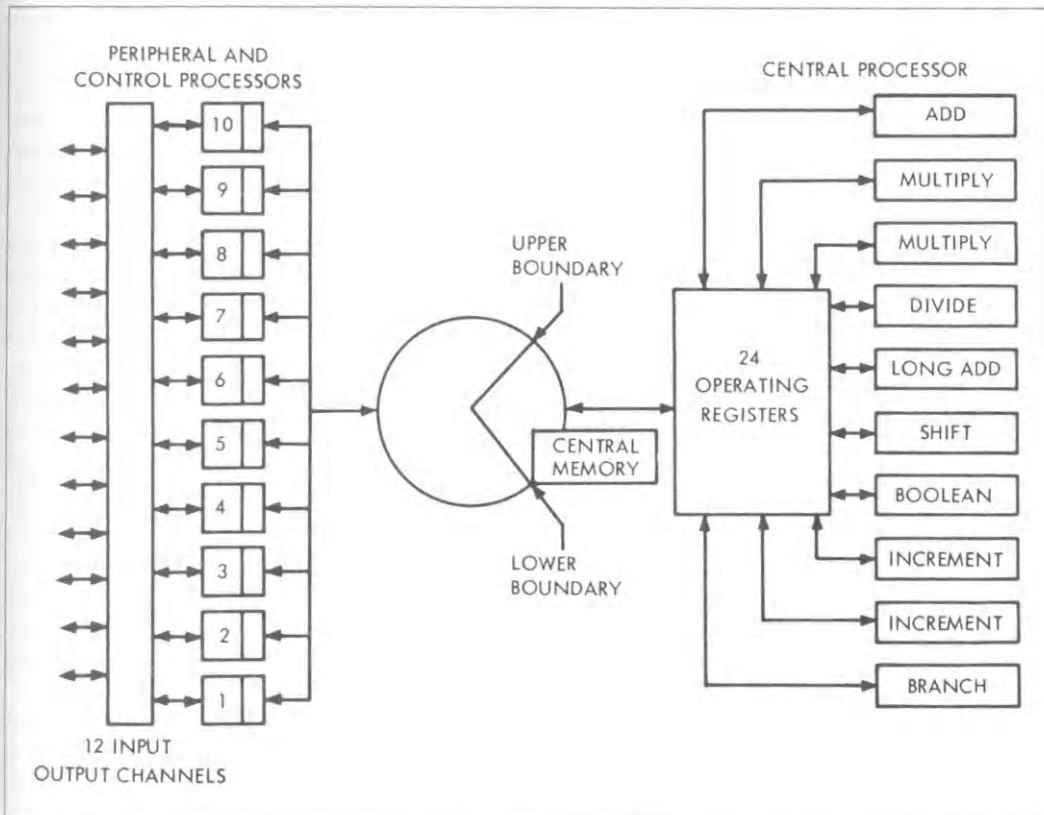


Fig. 3. Block diagram of 6600.

processor. These are the two increment units, floating add unit, fixed add unit, shift unit, two multiply units, divide unit, boolean unit, and branch unit. In a general way, each of these units is a three address unit. As an example, the floating add unit obtains two 60-bit operands from the central registers and produces a 60-bit result which is returned to a register. Information to and from these units is held in the central registers, of which there are twenty-four. Eight of these are considered index registers, are of 18 bits length, and one of which always contains zero. Eight are considered address registers, are of 18 bits length, and serve to address the five read central memory trunks and the two store central memory trunks. Eight are considered floating point registers, are of 60 bits length, and are the only central registers to access central memory during a central program.

In a sense, just as the whole central processor is hidden behind central memory from the peripheral processors, so, too, the ten functional units are hidden behind the central registers from central memory. As a consequence, a considerable instruction efficiency is obtained and an interesting form of concurrency is feasible and practical. The fact that a small number of bits can give meaningful definition to any function makes it possible to develop forms of operand and unit reservations needed for a general scheme of concurrent arithmetic.

Instructions are organized in two formats, a 15-bit format and a 30-bit format, and may be mixed in an instruction word (Fig. 4). As an example, a 15-bit instruction may call for an ADD,

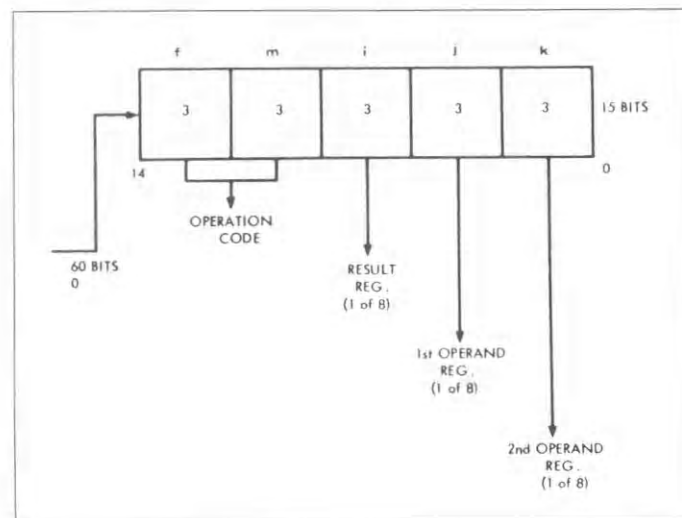


Fig. 4. Fifteen-bit instruction format.

designated by the f and m octal digits, from registers designated by the j and k octal digits, the result going to the register designated by the i octal digit. In this example, the addresses of the three-address, floating add unit are only three bits in length, each address referring to one of the eight floating point registers. The 30-bit format follows this same form but substitutes for the k octal digit an 18-bit constant K which serves as one of the input operands. These two formats provide a highly efficient control of concurrent operations.

As a background, consider the essential difference between a general purpose device and a special device in which high speeds are required. The designer of the special device can generally improve on the traditional general purpose device by introducing some form of concurrency. For example, some activities of a housekeeping nature may be performed separate from the main sequence of operations in separate hardware. The total time to complete a job is then optimized to the main sequence and excludes the housekeeping. The two categories operate concurrently.

It would be, of course, most attractive to provide in a general purpose device some generalized scheme to do the same kind of thing. The organization of the 6600 central processor provides just this kind of scheme. With a multiplicity of functional units, and of operand registers and with a simple and highly efficient addressing system, a generalized queue and reservation scheme is practical. This is called the *scoreboard*.

The scoreboard maintains a running file of each central register, of each functional unit, and of each of the three operand trunks to and from each unit. Typically, the scoreboard file is made up of two-, three-, and four-bit quantities identifying the nature of register and unit usage. As each new instruction is brought up, the conditions at the instant of issuance are set into the scoreboard. A snapshot is taken, so to speak, of the pertinent conditions. If no waiting is required, the execution of the instruction is begun immediately under control of the unit itself. If waiting is required (for example, an input operand may not yet be available in the central registers), the scoreboard controls the delay, and when released, allows the unit to begin its execution. Most important, this activity is accomplished in the scoreboard and the functional unit, and does not necessarily limit later instructions from being brought up and issued.

In this manner, it is possible to issue a series of instructions, some related, some not, until no functional units are left free or until a specific register is to be assigned more than one result. With just those two restrictions on issuing (unit free and no double result), several independent chains of instructions may proceed concurrently. Instructions may issue every minor cycle in the

absence of the two restraints. The instruction executions, in comparison, range from three minor cycles for fixed add, 10 minor cycles for floating multiply, to 29 minor cycles for floating divide.

To provide a relatively continuous source of instructions, one buffer register of 60 bits is located at the bottom of an instruction stack capable of holding 32 instructions (Fig. 5). Instruction words from memory enter the bottom register of the stack pushing up the old instruction words. In straight line programs, only the bottom two registers are in use, the bottom being refilled as quickly as memory conflicts allow. In programs which branch back to an instruction in the upper stack registers, no refills are allowed after the branch, thereby holding the program loop completely in the stack. As a result, memory access or memory conflicts are no longer involved, and a considerable speed increase can be had.

Five memory trunks are provided from memory into the central processor to five of the floating point registers (Fig. 6). One address register is assigned to each trunk (and therefore to the floating point register). Any instruction calling for address register result implicitly initiates a memory reference on that trunk. These instructions are handled through the scoreboard and therefore tend to overlap memory access with arithmetic. For example, a new memory word to be loaded in a floating point register can be brought in from memory but may not enter the register until all

previous uses of that register are completed. The central registers, therefore, provide all of the data to the ten functional units, and receive all of the unit results. No storage is maintained in any unit.

Central memory is organized in 32 banks of 4096 words. Consecutive addresses call for a different bank; therefore, adjacent addresses in one bank are in reality separated by 32. Addresses may be issued every 100 nanoseconds. A typical central memory information transfer rate is about 250 million bits per second.

As mentioned before, the functional units are hidden behind the registers. Although the units might appear to increase hardware duplication, a pleasant fact emerges from this design. Each unit may be trimmed to perform its function without regard to others. Speed increases are had from this simplified design.

As an example of special functional unit design, the floating multiply accomplishes the coefficient multiplication in nine minor cycles plus one minor cycle to put away the result for a total of 10 minor cycles, or 1000 nanoseconds. The multiply uses layers of carry save adders grouped in two halves. Each half concurrently forms a partial product, and the two partial products finally merge while the long carries propagate. Although this is a fairly large complex of circuits, the resulting device was sufficiently smaller than originally planned to allow two multiply units to be included in the final design.

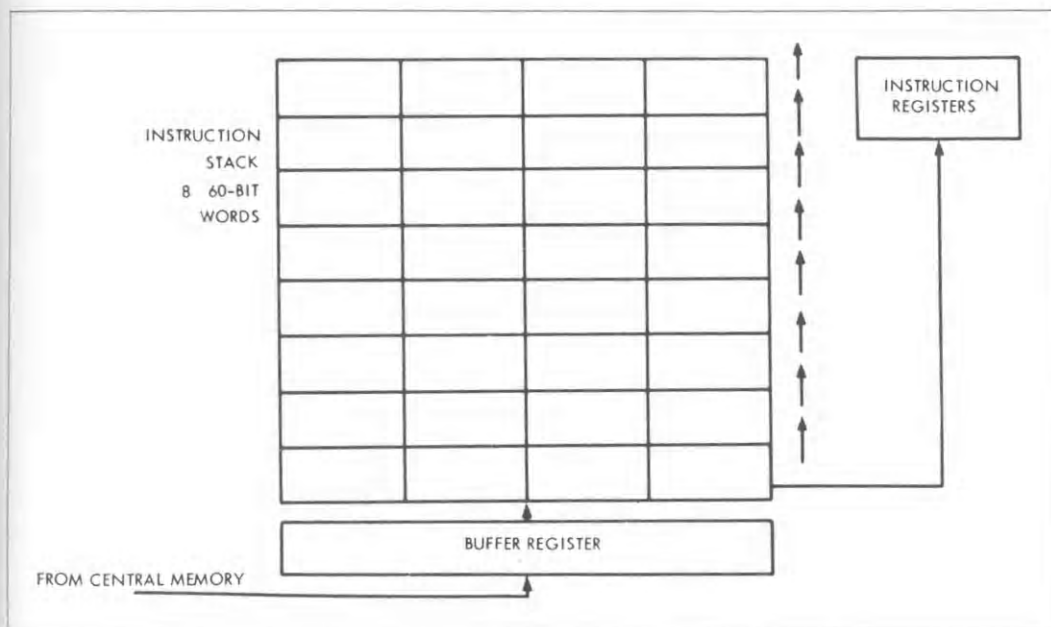


Fig. 5. 6600 instruction stack operation.

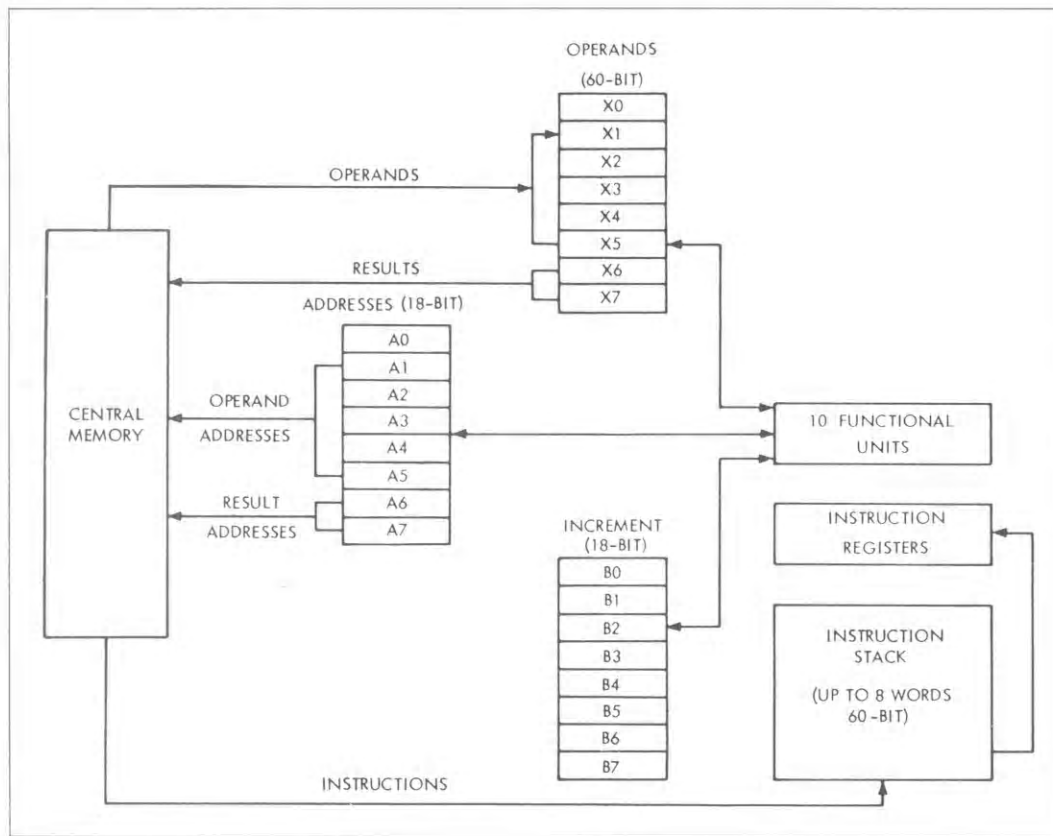


Fig. 6. Central processor operating registers.

To sum up the characteristics of the central processor, remember that the broadbrush description is "concurrent operation." In other words, any program operating within the central processor utilizes some of the available concurrency. The program need not be written in a particular way, although certainly some optimization can be done. The specific method of accomplishing this concurrency involves *issuing* as many instructions as possible while handling most of the conflicts during *execution*. Some of the essential requirements for such a scheme include:

- 1 Many functional units
- 2 Units with three address properties
- 3 Many transient registers with many trunks to and from the units
- 4 A simple and efficient instruction set

Construction

Circuits in the 6600 computing system use all-transistor logic (Fig. 7). The silicon transistor operates in saturation when switched "on" and averages about five nanoseconds of stage delay. Logic circuits are constructed in a cordwood plug-in module of about $2\frac{1}{2}$ inches by $2\frac{1}{2}$ inches by 0.8 inch. An average of about 50 transistors are contained in these modules.

Memory circuits are constructed in a plug-in module of about six inches by six inches by $2\frac{1}{2}$ inches (Fig. 8). Each memory module contains a coincident current memory of 4096 12-bit words. All read-write drive circuits and bit drive circuits plus address translation are contained in the module. One such module is used for each peripheral processor, and five modules make up one bank of central memory.

Logic modules and memory modules are held in upright hinged chassis in an X shaped cabinet (Fig. 9). Interconnections between modules on the chassis are made with twisted pair transmission

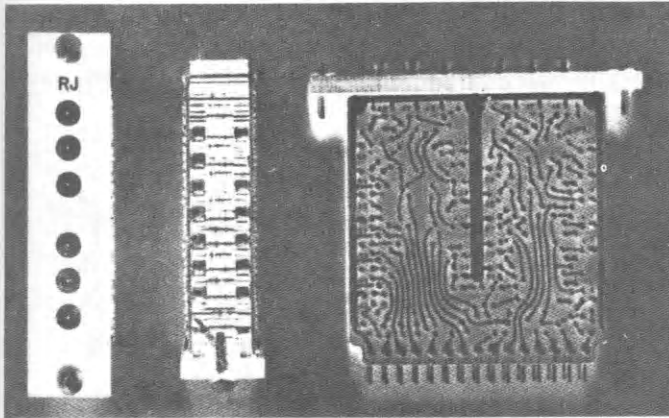


Fig. 7. 6600 printed circuit module.

lines. Interconnections between chassis are made with coaxial cables.

Both maintenance and operation are accomplished at a programmed display console (Fig. 10). More than one of these consoles may be included in a system if desired. Dead start facilities bring



Fig. 9. 6600 main frame section.

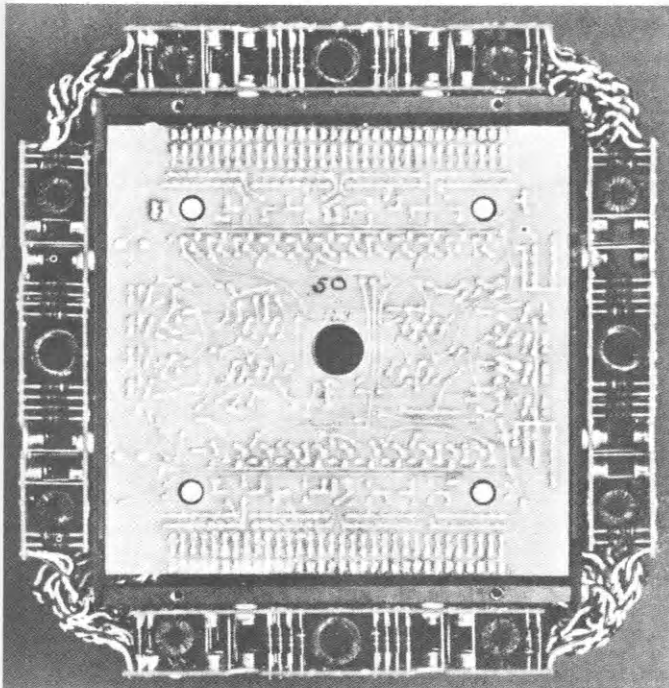


Fig. 8. 6600 memory module.

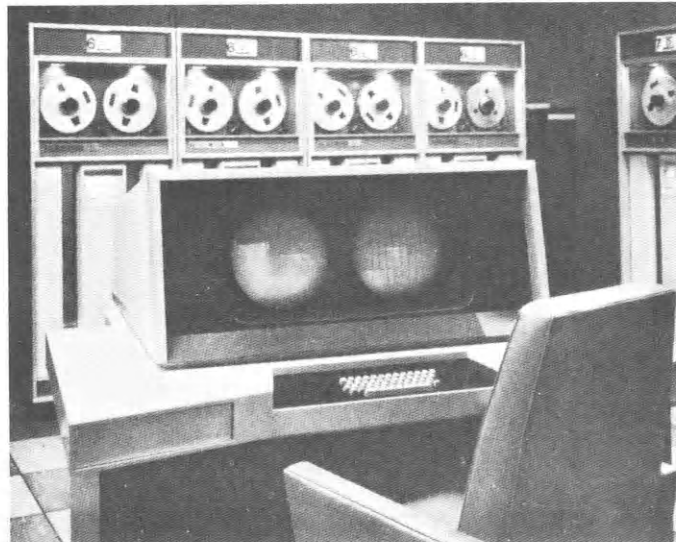


Fig. 10. 6600 display console.

the ten peripheral processors to a condition which allows information to enter from any chosen peripheral device. Such loads normally bring in an operating system which provides a highly sophisticated capability for multiple users, maintenance, and so on.

The 6600 Computer has taken advantage of certain technology advances, but more particularly, logic organization advances

which now appear to be quite successful. Control Data is exploring advances in technology upward within the same compatible structure, and identical technology downward, also within the same compatible structure.

References

AllaR64; ClayB64