UNIVAC®

9200

SYSTEM

SYSTEM DESCRIPTION

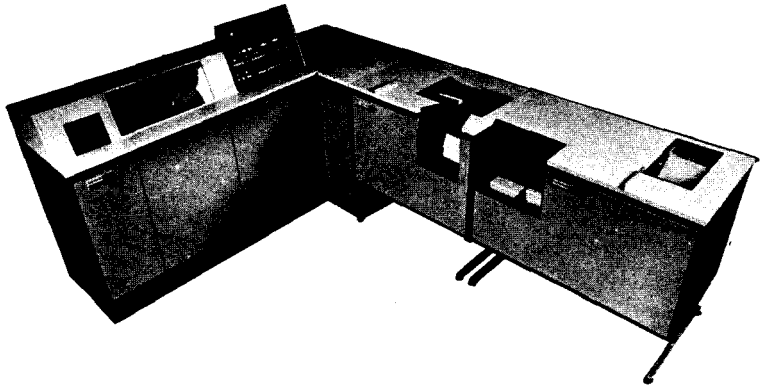
UP-4086 Rev. 2

CONTENTS

FIGURES

TABLE

1. INTRODUCTION



1.1. GENERAL

The extremely rapid development of electronic data processing in the last 15 years has been a natural response to the pressures generated by expanding business and scientific activities. Our economy has exhibited a vibrant expansion into new markets, new products, and new ideas; and scientific investigations have literally exploded in new and challenging areas. A concurrent explosion has occurred in the recordkeeping and other paperwork needed to support this activity. Business records and scientific data have multiplied to the point where they are manageable only through mechanized or electronic data processing.

1.2. ELECTRONIC DATA PROCESSING

Mechanized data processing systems were in use long before the advent of electronic computers. These systems were based primarily on the manipulation of punched-card records by electromechanical methods. The introduction of electronic computers in the early 1950's completely revolutionized and greatly extended the range of data processing in two important ways: increased processing speed and accuracy, and fully automated processing through internally stored programs.

The speed and flexibility of electronic data processing permits vast quantities of basic information to be synthesized into summary reports, trend analyses, or any other kind of data rendering that helps a business operate more effectively and efficiently. Moreover, these data syntheses can be obtained by electronic means within the same time frame as the data from which they are derived. For instance, the lag between an inventory depletion notice and a stock reorder issue can be reduced to a point where a business can operate constantly at optimum stock levels. The resulting efficiency produces many tangible results such as minimized inventory costs, better customer relations, increased business, and tighter fiscal control.

A company that introduces any form of automated data processing must evaluate carefully the cost of the data processing system versus the savings it will realize. There is a break-even point that cannot be exceeded without increasing overall operating costs. Many companies find this break-even point too low to support a computer system, but they must use automated data-processing techniques to remain competitive. The usual alternative is some form of mechanized, punched-card equipment which is only effective for a limited range of use.

1.3. THE 9200 SYSTEM AND THE NEW UNIVAC 9000 SERIES

The Univac Division of the Sperry Rand Corporation has long recognized the need for a computer system within the price range of punched-card tabulating equipment. To meet this need, the Univac Division is proud to announce the all-new 9200 System. . . a direct product of the careful, cost-conscious engineering which has characterized all Univac computers to give the user the maximum possible data-processing capability for the least possible cost.

The 9200 System costs no more than existing tabulating equipment and offers all the advantages formerly found only in the larger computer systems. Moreover, the 9000 Series offers programming compatibility among the systems that make up the series, and it covers a spectrum of data-processing needs from simple, card-oriented installations to large-scale, mass-storage systems. The 9200 System is completely modular; thus, it can be expanded easily to the 9300 System configuration, and compatible growth to the largest system is assured by progressive expansion rather than by a sudden changeover.

Although the 9200 System is designed for and priced within the range of modest-size, punched-card installations, it contains such state-of-the-art technical innovations as a plated-wire memory, monolithic integrated circuits, and the more familiar large-computer benefits such as internal programming and preprogrammed software packages. These features result in a system of exceptional performance with a program capacity that compares favorably with systems two or three times larger. The basic operational characteristics of the system are as follows:

card reading	– 400 cpm
printing	– 250 lpm
card punching	– 75 to 200 cpm
memory	– 8,192 bytes
	1.2 microsecond cycle time

Several of the technical innovations offered with the 9200 System are not even found in many of the larger and more expensive systems; they represent a true technological breakthrough that permits a very large data-processing capability to be offered at a fraction of former costs.

1.4. PROGRAMMING GROWTH AND COMPATIBILITY

The Univac approach eases many of the problems normally associated with the transition from a mechanized tabulating installation to a computer system. The changeover to large-scale computer methods takes place at a level which is wholly compatible with existing techniques, skills, and costs. Although the UNIVAC 9200 System contains the latest technical innovations, it is still a card-oriented system that can easily be utilized by current operating personnel. As part of the programming package for the system, Univac will provide a complete Report Program Generator that will fit the needs of the installation without introducing new and unfamiliar approaches.

As the user becomes more familiar with the capabilities and use of his system, Assembler is provided to extend its programming power. Utility routines and other easy-to-use programs are provided according to system requirements. The advantage to the user of this modular programming package is that operating personnel make increasingly efficient use of the system. The system is not only productive as soon as it is installed but it also provides both the direction and the means for future growth.

1.5. SUMMARY

The 9200 System is the first in the UNIVAC 9000 Series of computer systems. Each system is successively larger for handling the users' increasing data-processing capacity, and most important, programs and routines written for the smallest system will be compatible with the larger systems. This program-compatible feature will radically simplify the problems associated with expansion and the demands of more complex processing.

The design goal of the 9200 System has been to provide a computer that will meet the basic needs of the punched-card equipment user. It can be modified on site into the higher performance UNIVAC 9300 System with no dislocation in programming continuity but with a substantial increase in processing power.

2. SYSTEM OPERATION

2.1. GENERAL

The basic task of recordkeeping is to keep track of business transactions. This definition is probably an oversimplification in light of today's modern accounting systems because of the many different kinds of information that can be extracted from an accounting record, but the fundamental purpose of recordkeeping has really not changed. Only the methods or techniques of accomplishing this basic objective have changed. Recordkeeping has come a long way from cryptic marks on a clay tablet or from a roomful of Bob Cratchits making ledger entries with feather quills. Accounting methods have undergone the same revolutionary evolvement that has characterized other aspects of modern business and science. This evolvement was necessary and essential to the continued growth of business. The complexity and diversity of modern businesses have yielded many benefits such as mass production, uniform quality, and improved distribution to the consumer, but they also ignited a paper explosion that was confined only by the methods and equipment of automated data processing.

An important aspect of any recordkeeping function is the synthesis of data into a summary, or report, from which business decisions can be made. Business in the 19th and earlier 20th centuries frequently had to do without the nicety of many summary figures or reports because the logistics of information retrieval posed an impossible manpower problem. Thus, the combined pressures of increasing volume, diversification, and the ever-present need for comprehensive data analyses forced the development of mechanized, punched-card data processing.

The introduction of mechanized punched-card accounting machines and automated methods of data processing provided a partial solution to the problem of how to handle large, unwieldy masses of data. The rest of the solution has been provided by computers and the modern techniques of electronic data processing.

2.2. THE 9200 SYSTEM

The UNIVAC 9200 System is essentially a card-oriented system that performs all of the familiar punched-card functions such as card reading, calculating, printing, reproducing, summary punching, and interspersed gang punching. The power of the 9200 System resides in the speed and versatility of an electronic, stored-program computer applied to these basic tabulating functions. The result of this application is a tremendous increase in efficiency and processing potential but without an accompanying increase in costs. The cost of the 9200 System is within the same range as that of much slower punched-card equipment.

The speed and versatility of the 9200 System results in faster and more comprehensive reports, and it radically reduces the critical time between data availability and report generation. In one pass the 9200 System can perform the same functions that require four or five separate machine operations with punched-card equipment. Figure 2–1 illustrates the relative time advantage in operating speed of the 9200 System versus standard punched-card equipment for a given series of operations.

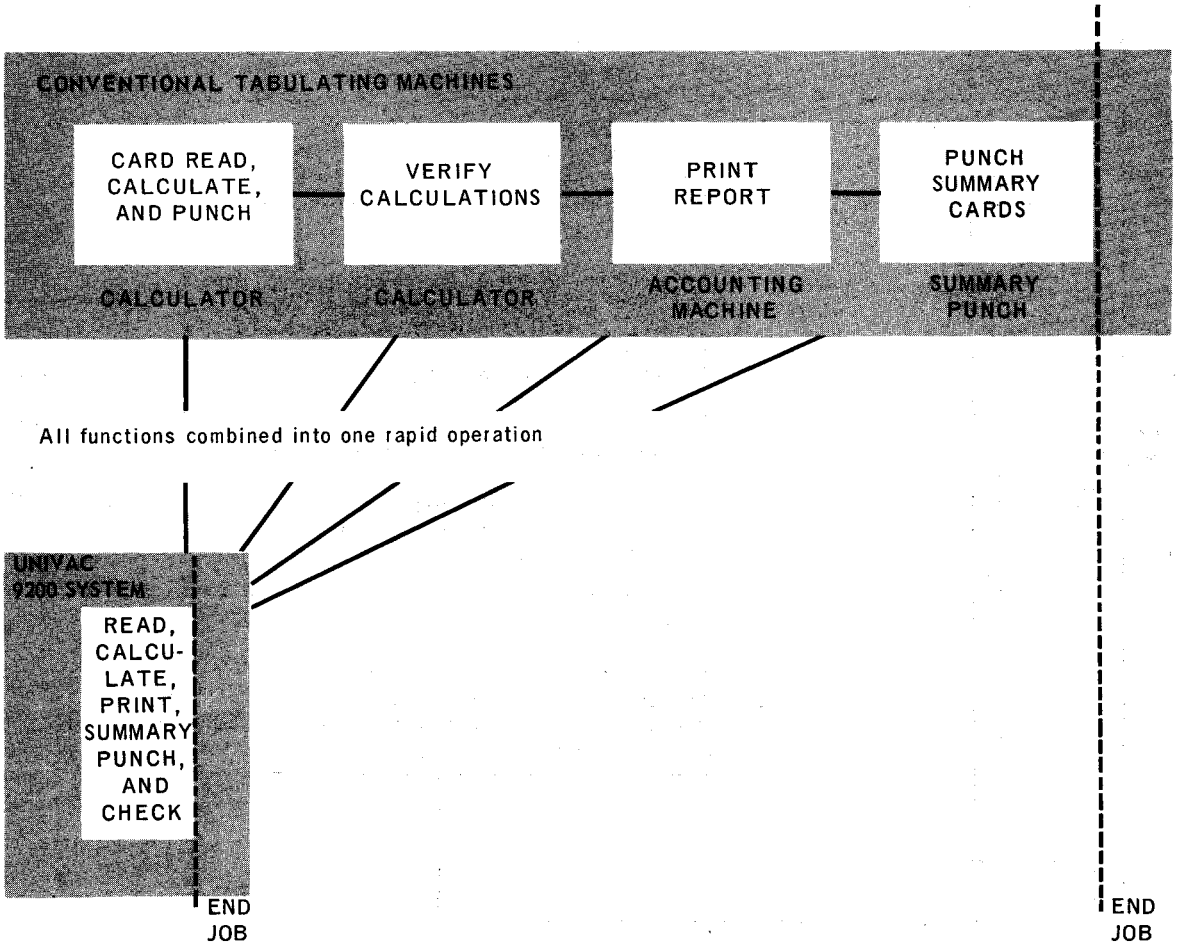


Figure 2–1. Relative Time Advantage of UNIVAC 9200 System Versus Conventional Tabulating Machines

The net advantages of the improved data processing efficiency offered to the user by the 9200 System are significantly better control, an increase in the volume of work that can be handled, development of more precise "management by exception" information, and an enlarged potential for exploring new application possibilities. Thus, the 9200 System actually represents a reduction in costs because far more data can be processed for a given equipment investment. Figure 2–2 illustrates how the 9200 System could be used for a typical inventory updating application. All computing and control functions are centralized in the computer, and information in punched-card or printout form is derived simultaneously with processing.

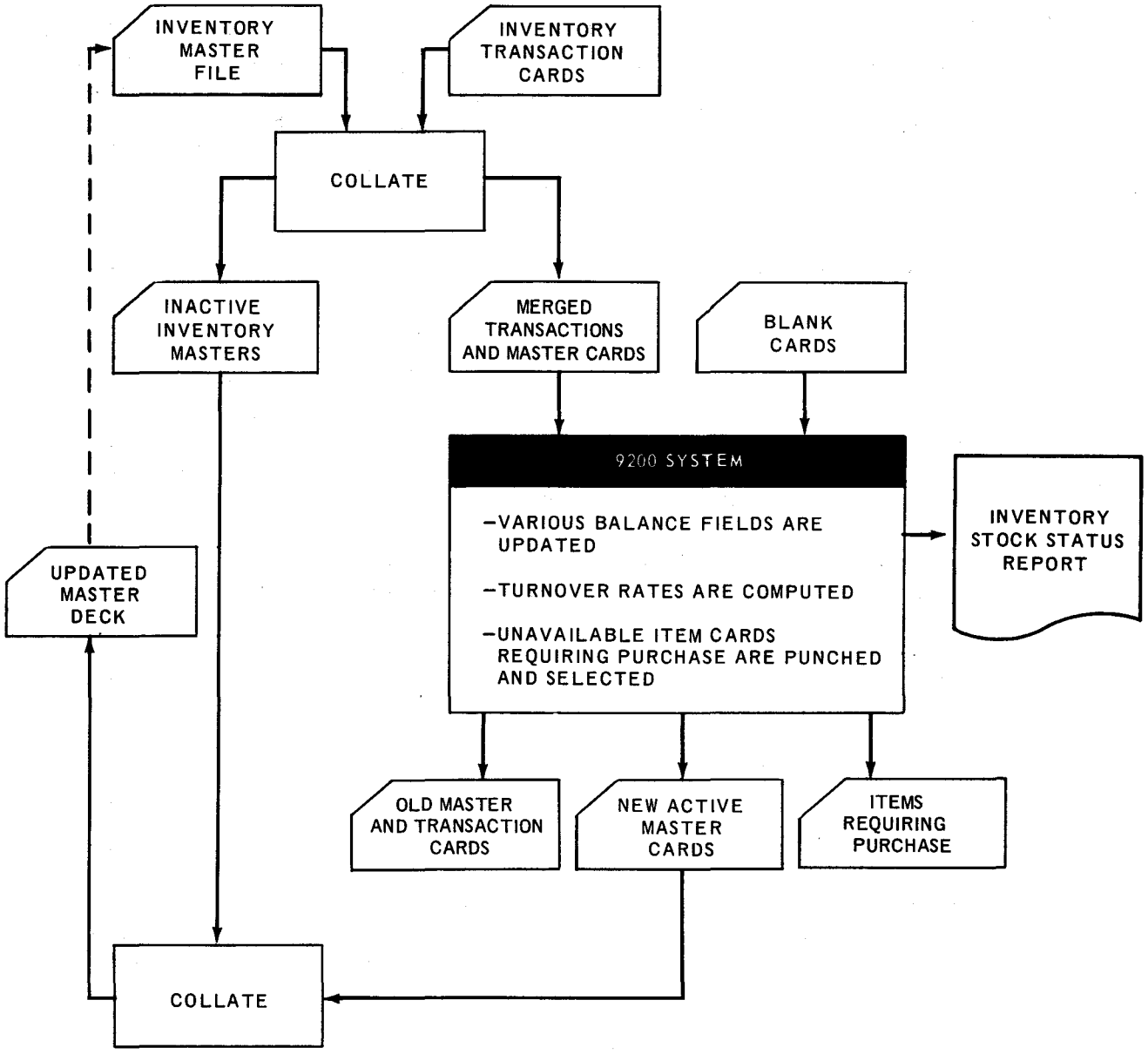


Figure 2–2. 9200 System Use for Inventory Updating Application

The 9200 System is not simply a scaled-down version of a larger computer; it is a complete, high-speed system designed for a low-cost market. Physically, the processor and the printer are constructed in an integrated unit to which all the peripheral equipment can be easily attached. This mode of construction permits the 9200 System to be expanded very easily to more versatile configurations or converted to the larger 9300 System. Thus, for the first time, a modern, high-speed computer system is available to users of punched-card equipment whose requirements formerly precluded the use of electronic computers. . .a system that makes no compromise with technology at the expense of performance.

2.2.1. Functional Configuration

The functional configuration of the 9200 System consists of input, processing, and output. In the basic system, input is from a card reader, processing is done in the central processor, and the outputs are to a card punch or a printer.

Inputs to the system are data to be processed and the instructions to process the data.

The central processor consists of three functional parts: storage, control, and arithmetic and logic. The storage portion of the 9200 System is a plated-wire memory with 8192 positions of storage. Data and the program instructions are stored in the memory at addressable locations.

The control portion of the processor acts upon and is acted upon by instructions stored in the memory to pace the computer, step-by-step, through the operations called for by the instructions. The control portion is thus the direct link between, for instance, a request to print a line and the actual printing of the line.

The arithmetic and logical functions of the central processor constitute the computational and decision-making area of the computer. The arithmetic operations are done under control of instructions in the memory. Logical operations are initiated by instructions, and the course of the program is influenced according to the outcome of the operation. For example, the result of a computation could be compared with a stored quantity to determine whether it is lower than, higher than, or equal to the stored quantity and the direction of the program would be controlled by the result of the logical comparison.

2.2.2. Software Packages

A program is a set of instructions which causes the computer to perform a controlled series of operations that transform given input into desired output. The program embodies the data processing operation the user desires to have done – prepare invoices, checks, registers, and so on. The "content" of a program is dictated by the requirements of the specific problem or accounting operation at hand and can be defined only by the user. The user's definition, when rendered in terms of the instructions that make up a program, involves many common computer-oriented operations which are not part of his definition. These common operations (sometimes called housekeeping chores) can be standardized for all programs and incorporated into ready-to-use software packages which allow the user to concentrate his programming effort on defining the program content while the repetitious housekeeping chores are taken care of automatically.

Software package aids are supplied with the 9200 System to satisfy most processing contingencies and to help the user make maximum use of his system. Section 5 of this manual contains a detailed account of the instructions and the internal structure of the central processor and describes the software packages that are available with the 9200 System.

2.3. 9200 CARD CONTROLLER SYSTEM

An especially powerful version of the 9200 System is one which includes the UNIVAC 1001 Card Controller in an online configuration. This configuration, called the 9200 Card Controller System, offers several advantages to the punched-card user. The most outstanding of these is the elimination of the preprocessing operation of card collating because of the dual file input capability of the 1001. An example of a dual file input feed in an inventory application is illustrated in Figure 2–3. Not only is the collating operation completely eliminated, but both files are read at a combined speed of up to 2000 cpm – considerably faster than conventional card reader speeds.

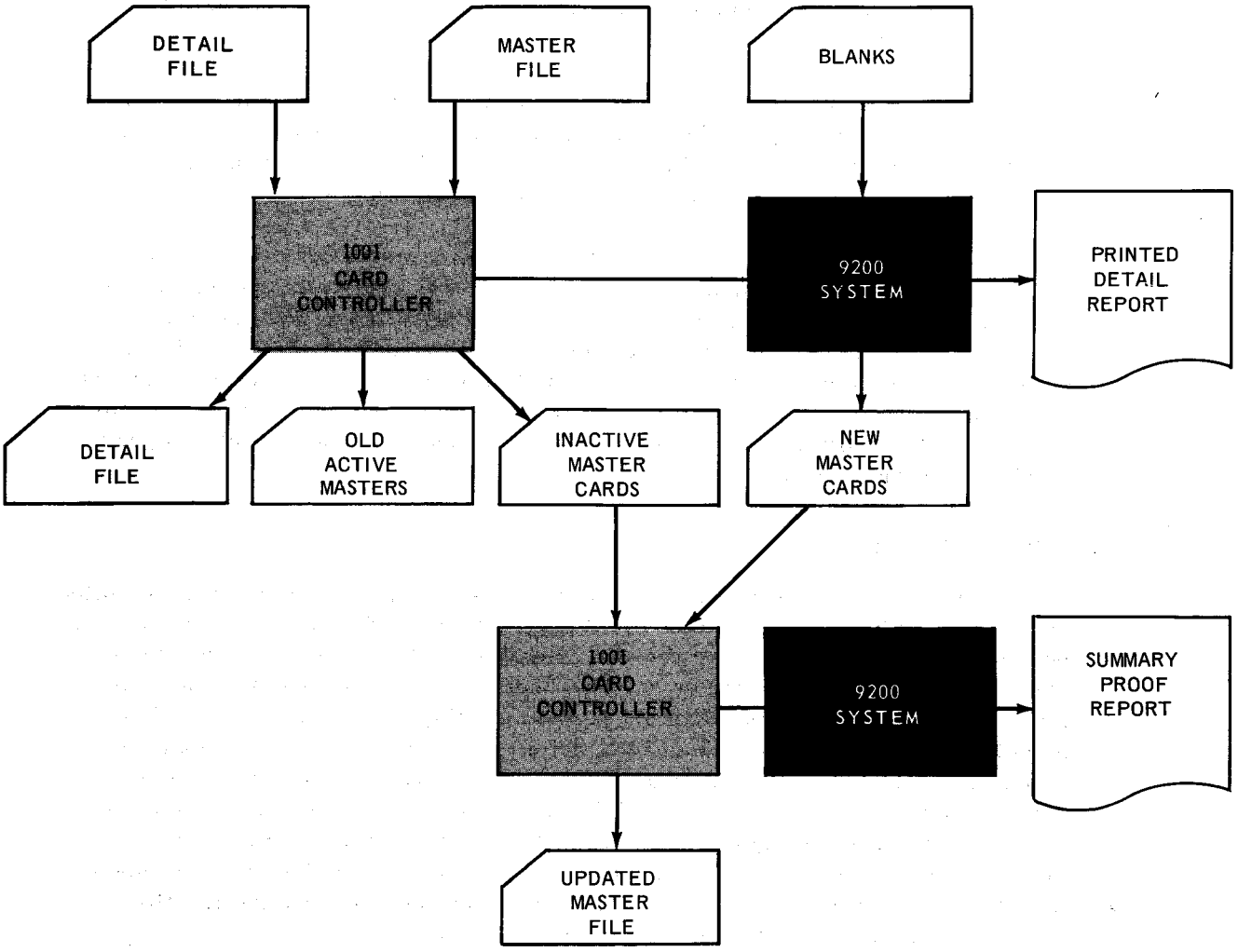


Figure 2–3. Dual File Input using 1001 Card Controller

Depending on the application, the user may take advantage of other unique features of the system such as advance file search, merging and selection while processing, and multifile processing. A direct application of the high-speed, flexible 9200 Card Controller System is shown by a comparison between it and a typical punched-card approach to a manufacturing application in Figure 2–4. The separate operations required by each approach are numbered on the figure. In the conventional system, nine distinct tabulating or processing functions are required to accomplish the same job that the UNIVAC system does in three operations. The Univac approach is not only faster, it is also simpler to use.

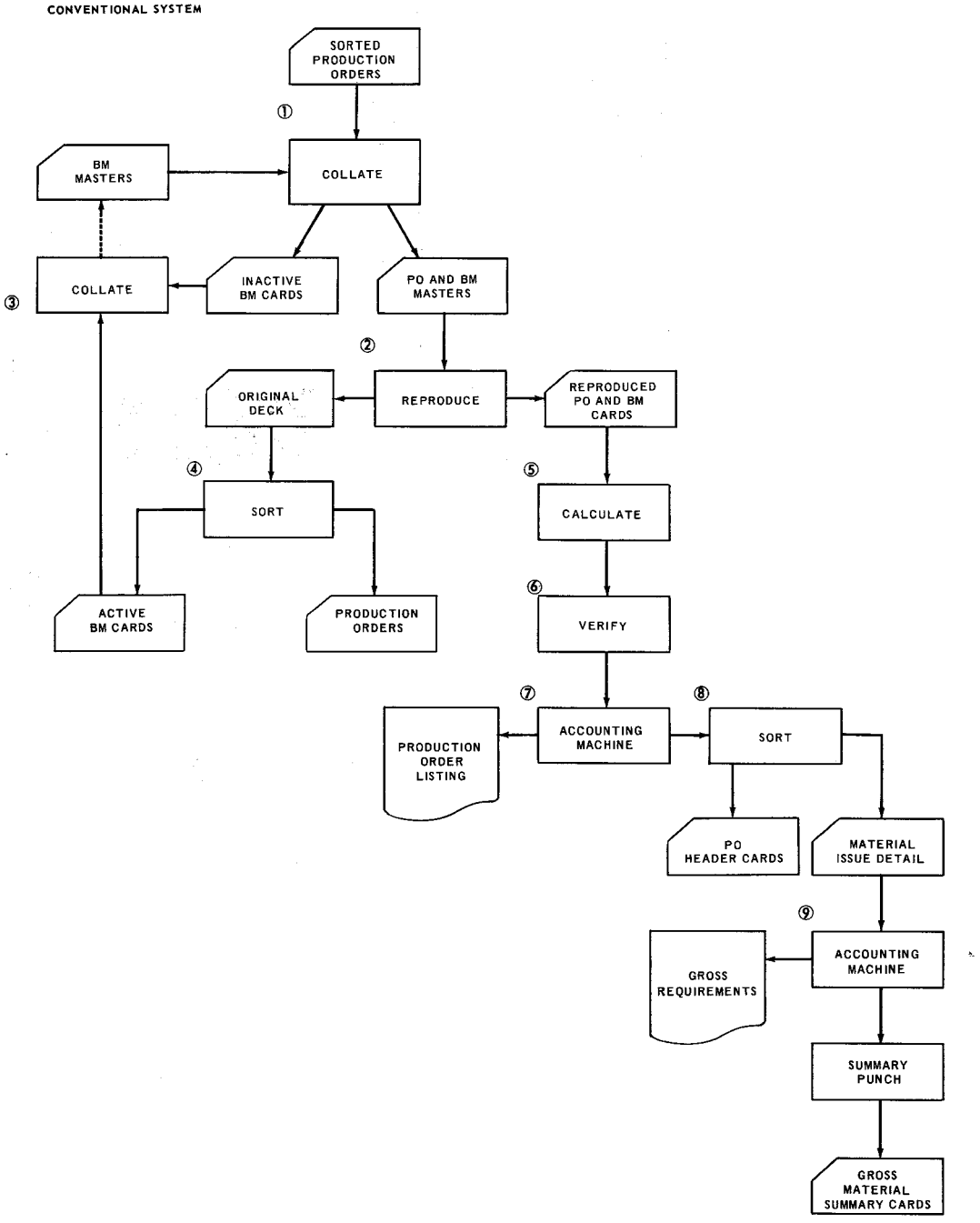


Figure 2–4. Comparison between 9200 System and Typical Punched Card Approach, Sheet 1 of 2.

UNIVAC 9200 SYSTEM WITH 1001 CARD CONTROLLER

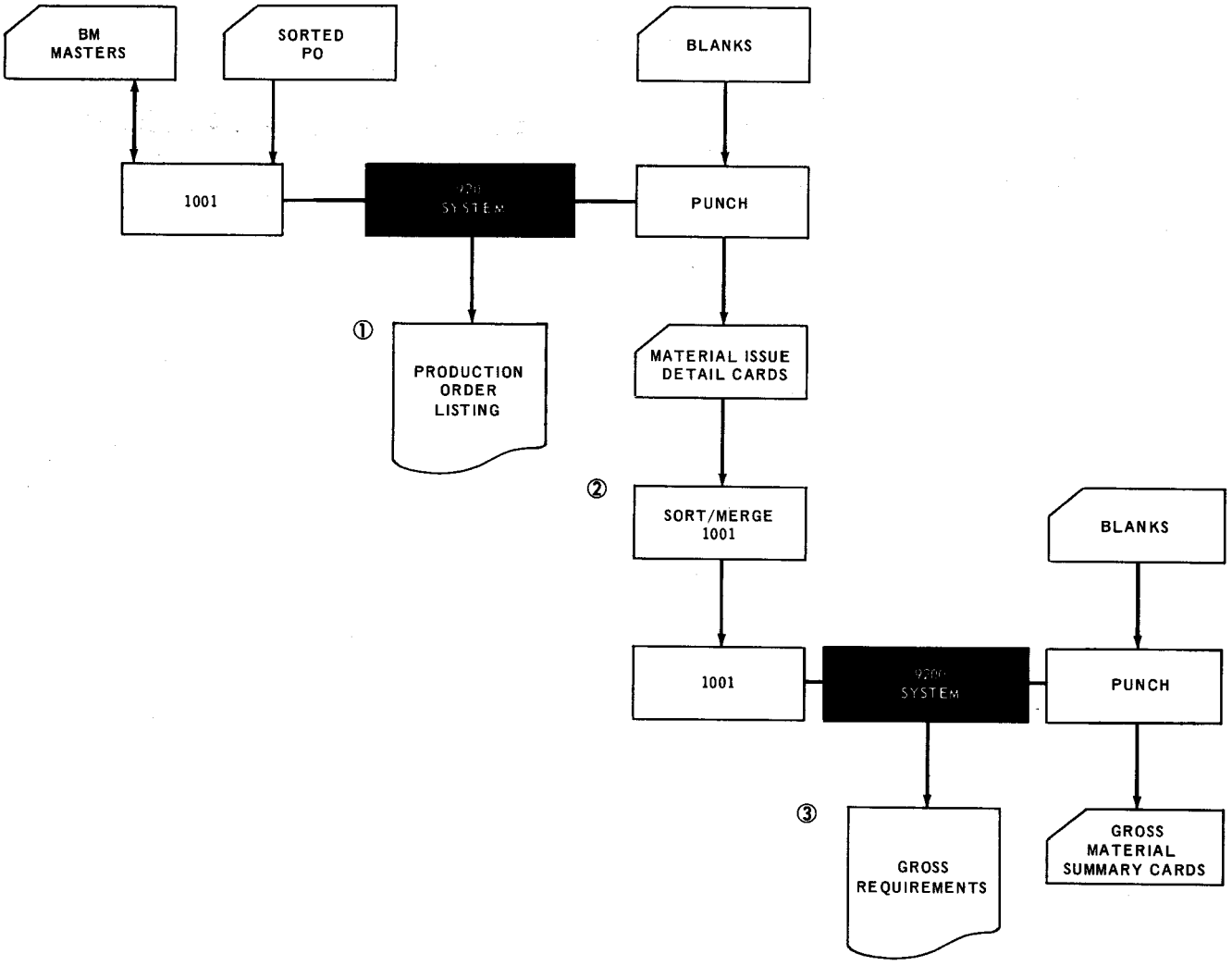


Figure 2–4. Comparison between 9200 System and Typical Punched Card Approach, Sheet 2 of 2.

3. INNOVATIONS

3.1. GENERAL

Before the components of the UNIVAC 9200 System are described in detail, some of the more outstanding technical innovations offered with the system will be explained in this section.

In the introduction to this manual it was stated that the new UNIVAC 9200 System uses a plated-wire memory and monolithic integrated circuits. The purpose of this section is to explain, briefly and simply, what these innovations are and how they enhance the qualities of the total system.

3.2. PLATED-WIRE MEMORY

The memory element used in a plated-wire storage system is basically a thin film, but the film is electroplated around a thin (0.005 inch) wire instead of being deposited as a tiny spot on a glass substrate. A plated-wire memory retains all the advantages of speed, small size, and low-power operation that characterize planar thin-film memories and offers, in addition, unique advantages available at present only in plated-wire memories.

3.2.1. NDRO Operation

Most core storage and thin-film memories operate in a destructive readout (DRO) mode, which means that each time information is read from a storage location, the information stored in that location is erased. If the information is to be retained, it must be rewritten into the memory. DRO operation consumes extra power and lengthens the memory cycle time because a write cycle must follow each read cycle.

The UNIVAC plated-wire memory operates in a nondestructive readout (NDRO) mode, which means that information does not have to be rewritten into memory after each read operation. Power consumption is thus reduced, and the memory cycle time is not lengthened by the necessity for a write operation after each read operation.

3.2.2. Ease of Fabrication

When special-purpose memories have offered NDRO operation in the past, intricate techniques involving multi-aperture cores or vacuum deposition to very exacting specifications were generally implied. These techniques are made obsolete by the methods of plated-wire memory fabrication. Plated-wire memory construction is much less complicated, thereby making it inherently more reliable than a planar, thin-film memory. Its relatively lower fabrication costs are also the basic reason why such an advanced type of memory can be offered in a system that can compete favorably with the cost of tabulating equipment.

In the extremely simple approach developed and conceived of by Univac, the wire on which the thin film is plated is also an integral part of the electronic circuitry for the memory. This feature of plated-wire memories reduces the number of separate elements required in the memory structure and simplifies the electronic operation of the memory.

Figure 3–1 is a photograph of the UNIVAC thin-film plated-wire memory used in the 9200 System. A general summary of plated-wire memory advantages is as follows:

- Nondestructive Readout
- Very Simple Construction
- Simple Read/Write Operation
- Small Switching Energy – about 1/100 of that required to switch conventional ferrite cores.

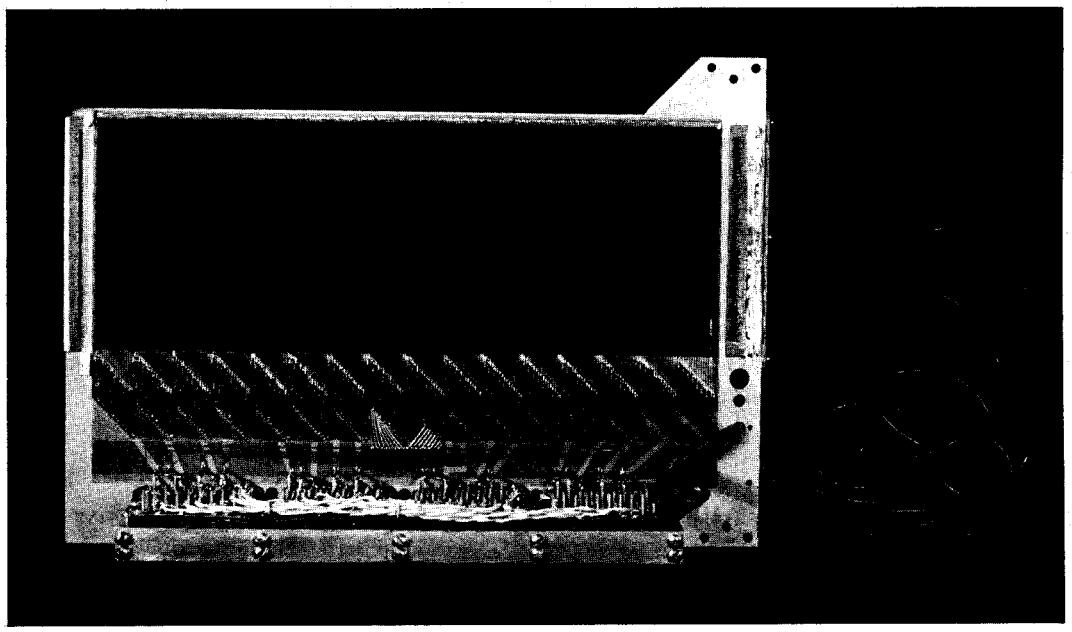


Figure 3–1. UNIVAC Thin-Film Plated-Wire Memory

3.3. MONOLITHIC INTEGRATED CIRCUITS

An integrated circuit can be thought of as an electronic component which contains a network of *passive* elements, such as resistors and capacitors, *active* elements, such as transistors, and interconnecting paths among all the passive and active elements. An integrated circuit is referred to as *monolithic* when all its elements are part of a single semiconductor crystal. It is referred to as *hybrid* if the passive elements are formed on a glass or ceramic substrate and the active elements are subsequently bonded to the substrate.

When circuit designs require tight tolerances or dissipate large amounts of power, the hybrid approach is usually necessary. By designing for a wider component tolerance range and by minimizing dissipation, Univac Division has been able to offer its basic logical NOR circuit in a fully monolithic package. Not only does this approach eliminate a large number of bonding operations, but it also assures more reliable performance in the typical integrated circuit because of its low dissipation and generous allowance for component variation.

The chief limitation to the size of a network or the number of independent networks that can be packaged in an integrated circuit is the number of terminals on the circuit from which connections can be made to the rest of the system. In the monolithic integrated circuits used in the 9200 System, each circuit has 12 distinct logic connections – which is 33% more than offered by competitive hybrid circuits.

The number of logical connections, and thus the utility of each circuit, is effectively multiplied by the extensive use of *diodes* (tiny electrical "valves" which open or close depending on how voltage is applied to them) to couple signals into and out of each circuit. The use of diodes increases the utility of each monolithic circuit by a factor of 2. This increase in the effectiveness of the basic circuitry by means of extensive diode coupling is a unique Univac development which enhances the basic 9200 System approach of maximum capability with no increase in costs.

A typical example of the structure of a 9200 System integrated circuit is shown in Figure 3–2.

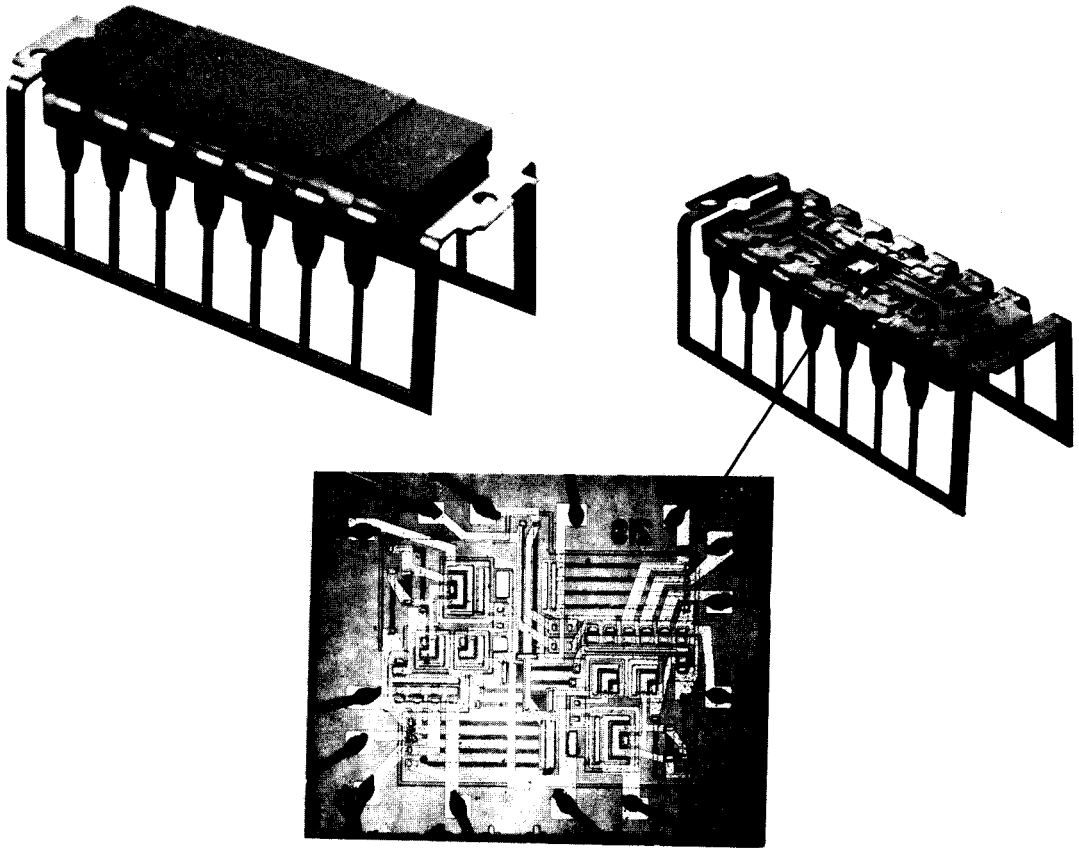


Figure 3–2. Monolithic Integrated Circuit

4. SYSTEM DESCRIPTION

In the 9200 System the features of larger and more powerful computers have been successfully incorporated into an inexpensive package that can be readily afforded by small punched-card installations and even by potential users who at present do not yet have automated data handling facilities. The UNIVAC 9200 System can be put into operation immediately by anyone who is familiar with standard data handling techniques, or in a very short time after a brief training period. Yet the 9200 System represents an open-ended invitation for expansion and growth because it is the first of a program-compatible computer family (the UNIVAC 9000 Series) which will encompass an exceptionally wide range of data processing demands. The operational capacity of the UNIVAC 9200 System is not limited by the speed of its processor because it contains an extremely rapid memory and very advanced and equally rapid circuitry. The exceptional processing and memory speed of the 9200 System yields the following partial list of benefits to the user:

- Eliminates "queueing" of I/O Units

  Because of the speed of the processor the programmer seldom has to allow time for the processor to "catch up" with the input/output units. Processing is done in parallel with card reading, punching, printing, or any other input/output function.

- Better operating balance

  The goal of maximum system utilization at all times is more closely approached.

- Larger data-handling capacity for a given amount of equipment

  Faster operating speed is directly equivalent to larger data-handling capacity because I/O equipment is used much more efficiently and because fixed quantities of data can be handled in shorter times by the processor.

4.1. THE UNIVAC 9200 SYSTEM

Figure 4–1 is a photograph of the basic UNIVAC 9200 System. A summary of overall System characteristics is included in the figure. The I/O units are combined with the processor to minimize the requirements for space, power, and cabling and to simplify the installation. Yet the system is completely modular and easily expandable to include any of the options provided with the system.

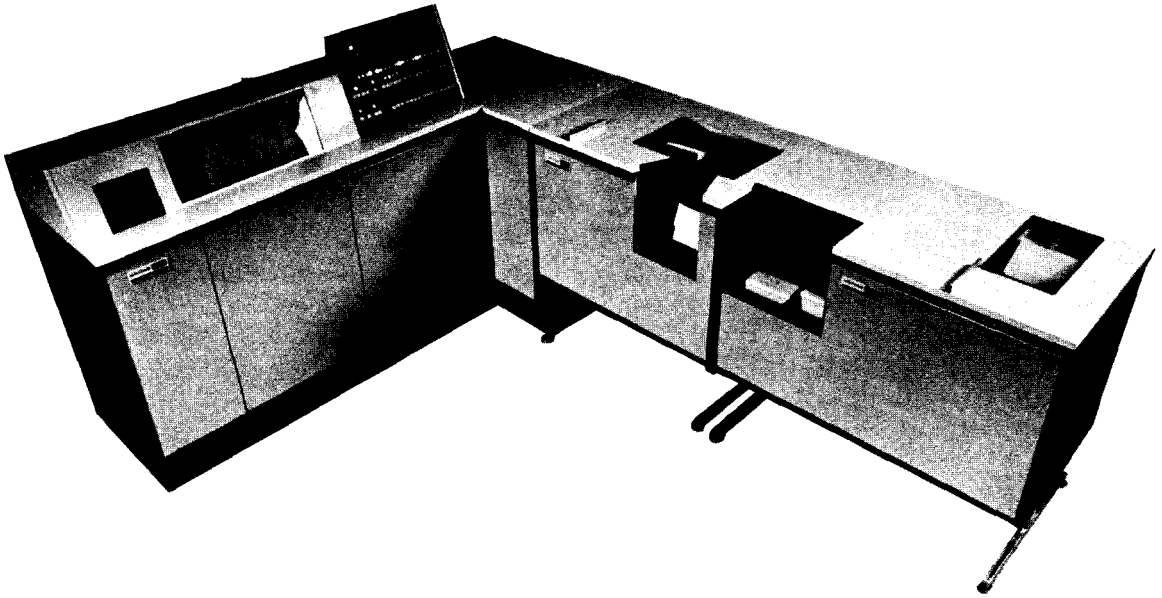


Figure 4–1. UNIVAC 9200 System

CHARACTERISTICS	
SYSTEM ORIENTATION	Card
BASIC MEMORY	8192 bytes
MEMORY CYCLE TIME	1.2μsec
ADD (DECIMAL) INSTRUCTION TIME (TWO 5-DIGIT FIELDS)	104μsec
CARD READ – BASIC READER – 1001 CARD CONTROLLER	400 cpm 1000/2000 cpm
CARD PUNCH	75–200 cpm
PRINT SPEED	250 lpm
VARIABLE-SPEED PRINTING (OPTIONAL)	250/500 lpm alphanumeric 500 lpm numeric
OVERLAPPED I/O UNITS	Standard
MULTIPLEXER CHANNEL RATE	85000 bytes/sec

The general layout of the equipment that comprises the basic system is shown in Figure 4–2. An overall block diagram of the system and its complete range of options is given in Figure 4–3.

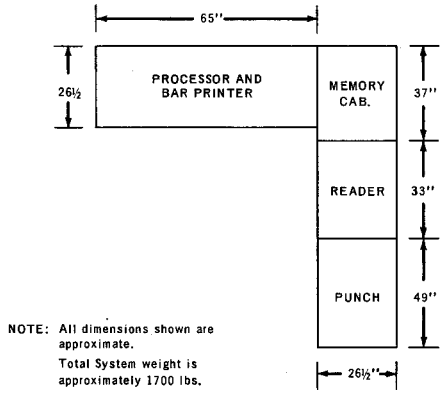


Figure 4–2. Basic System Dimensions

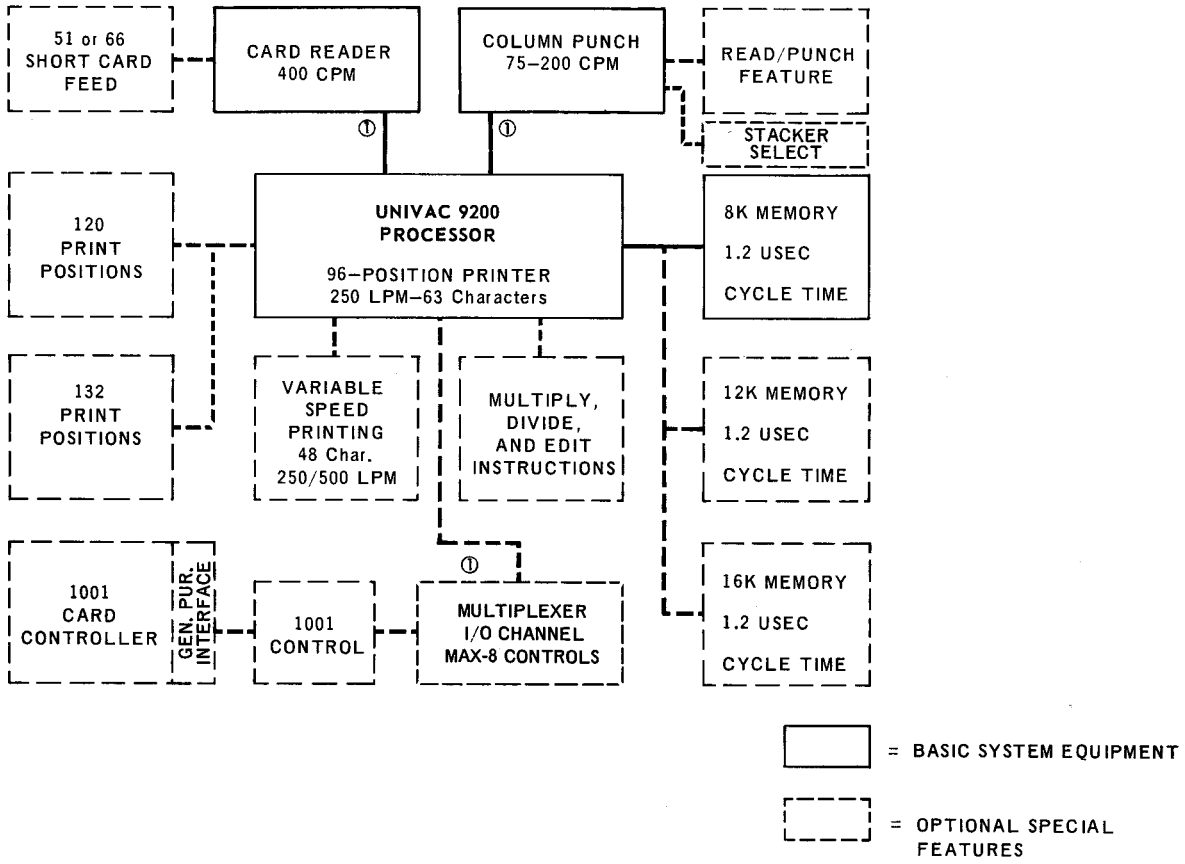


Figure 4–3. Block Diagram of 9200 System with Special Features

4.1.1. Processor

The computational power of the 9200 System is concentrated in a compact but extremely versatile processing unit. Work done by a calculator, reproducer, accounting machine, and summary punch can all be combined under the control of programmed instructions which are stored in the memory portion of the processor. The memory in the basic 9200 System has 8192 positions of storage, but for more comprehensive processing, memories are also available with 12,288 and 16,384 positions of storage. A position of storage in the 9200 System consists of an 8-bit byte. A general synopsis of processing unit characteristics is as follows:

MEMORY TYPE	Plated wire
MEMORY SIZE	8192 basic, 12,288 or 16,384 optional
MEMORY CYCLE TIME	1.2 microsecond
BASIC STORAGE UNIT	1 byte = 8 bits + parity bit = one alphabetic character or two packed numeric digits
CIRCUITRY TYPE	Monolithic integrated NOR circuits, diode-coupled
OTHER FEATURES:	All memory transfers are parity checked Card reading, card punching, and printing can be done simultaneously with processing A versatile two-address instruction controls the processor functions

4.1.2. Control Console

The entire 9200 System is controlled from a simple, highly functional control console (Figure 4–4) located on the right side of the printer chassis.

Power for the 9200 System (including I/O units) is switched on and off at the control console. Input/output functions are controlled by the upper row of switches on the console, and I/O error indications are given by the light above each group of I/O control switches.

The error lights merely indicate that an error has occurred and which I/O unit is at fault. The nature of the error is displayed in the centrally located row of display lights (Figure 4–4) by way of a program-controlled instruction. The display lights identify the device and give a coded display of the error.

Other lights and switches on the console are provided primarily for programmer, operator, or maintenance debugging and monitoring. For example, the contents of any address in memory can be displayed and altered by means of the MEMORY ADDRESS, DATA ENTRY, and DISPLAY and ALTER switches, and the program can be advanced step-by-step through the computer by means of the INST switch on the left side of the console. A detailed explanation of the use and interpretation of the console switches and indicators is beyond the scope of this manual. This information will be provided in forthcoming manuals dealing with operation and maintenance.

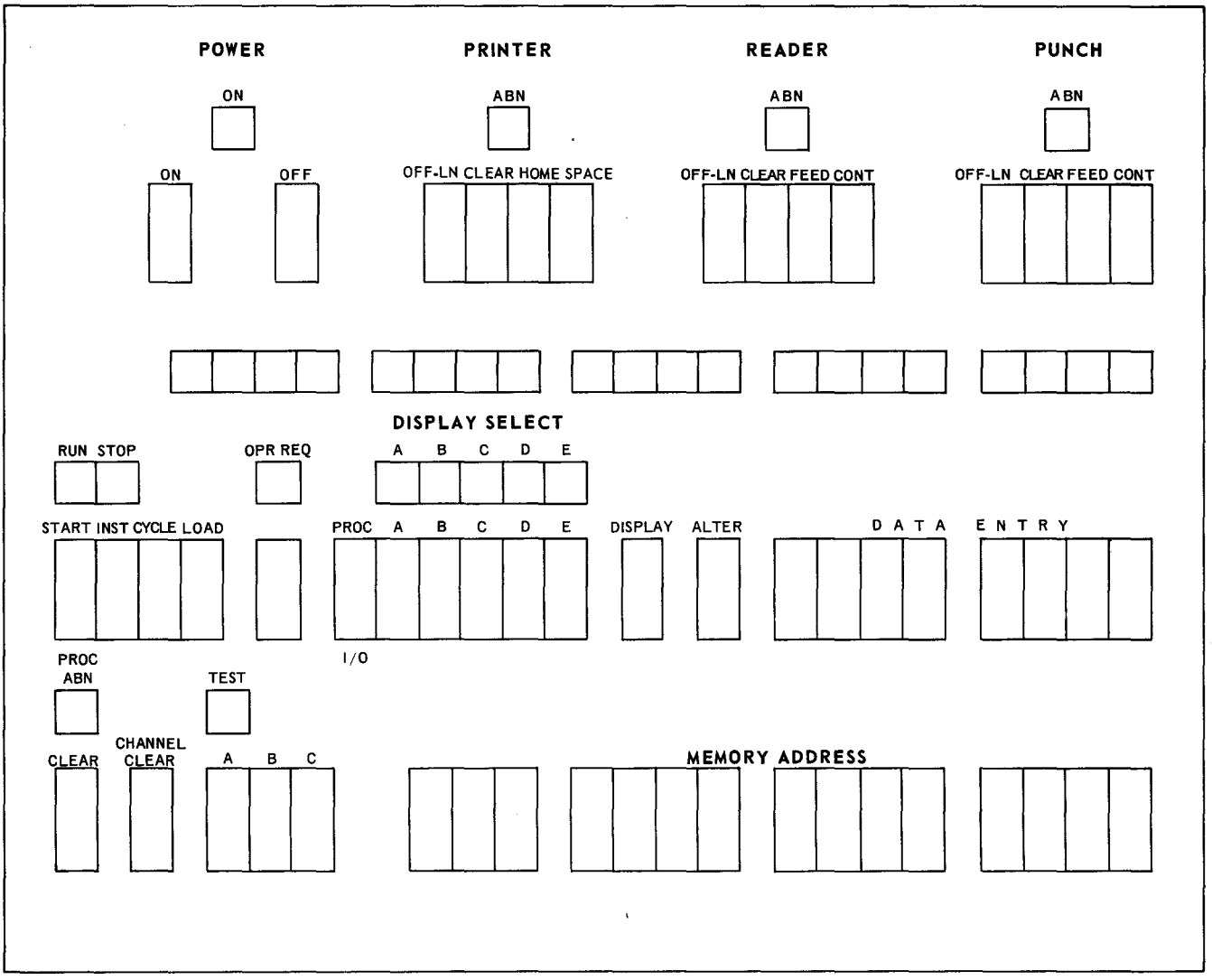


Figure 4–4. UNIVAC 9200 Control Console

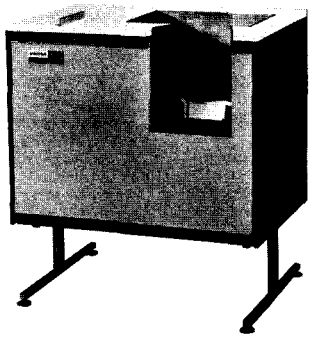
4.1.3. Multiplexer Channel

The operation of the reader, printer, and punch is integrated with the operation of the processor in such a way that the entire system operates as a unit. In addition, an optional multiplexer channel is provided to connect other I/O devices, such as the optional 1001 Card Controller, directly into the system. The multiplexer channel permits up to 8 control units and 64 I/O devices to be connected into the channel.

4.2. INPUT/OUTPUT UNITS

Input/output functions in the 9200 System consist of printing, card reading, and card punching. These functions are performed in three units under direct control of the processor.

4.2.1. Card Reader



CHARACTERISTICS	
CARD TYPE	80 column
CARD READING RATE	400 cpm
INPUT HOPPER CAPACITY	1200 cards
OUTPUT STACKER CAPACITY	1500 cards
SPECIAL FEATURES	51- or 66-column short card feeds

4.2.1.1. Card Reader Timing

General Timing for the Card Reader is presented in Figure 4–5.

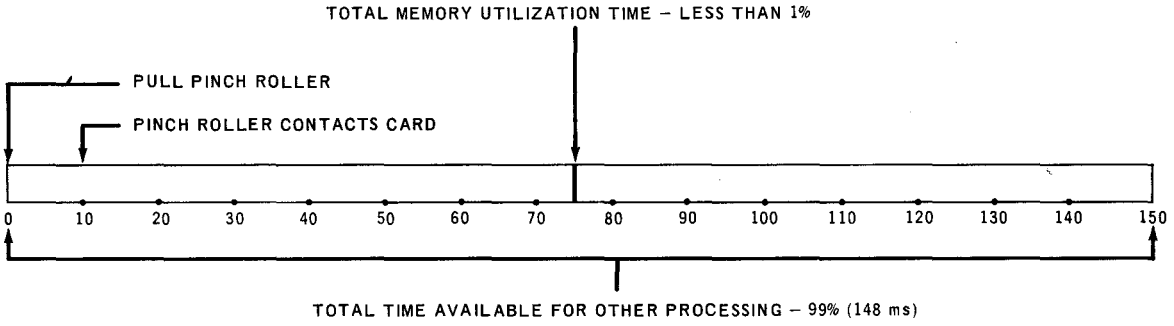


Figure 4–5. Card Read Cycle Timing

4.2.1.2. Card Reader Mechanism

The basic card path through the reader is presented in Figure 4–6.

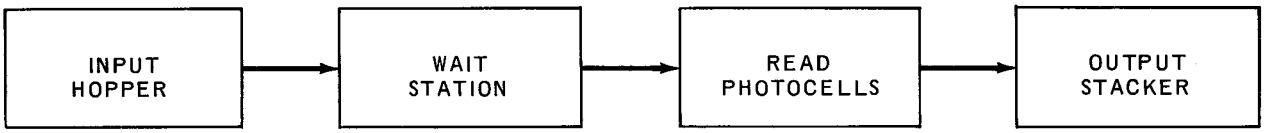


Figure 4–6. Card Reader Feed Path

4.2.2. Card Punch

CHARACTERISTICS	
CARD TYPE	80 column
CARD PUNCHING RATE	75 to 200 cpm 75 cpm when entire card is punched 200 cpm when col. 14 or lower is last column punched
INPUT HOPPER CAPACITY	1200 cards
OUTPUT STACKER CAPACITY	700 cards
REJECT STACKER CAPACITY	700 cards
SPECIAL FEATURES	Pre-punch read station Program stacker select

4.2.2.1. Card Punch Timing

Figure 4–7 shows the variation in card punching rate according to the last column punched. The memory utilization rates for the range of punch speeds is shown in the following table.

LAST COLUMN PUNCHED	PUNCH SPEED (in cpm)	CARD CYCLE TIME (in milliseconds)	TIME AVAILABLE FOR PROCESSING (in milliseconds)
14	200	300	299
20	173	347	346
30	147	408	407
40	120	500	499
50	104	577	576
60	92	652	651
70	83	722	721
80	75	800	799

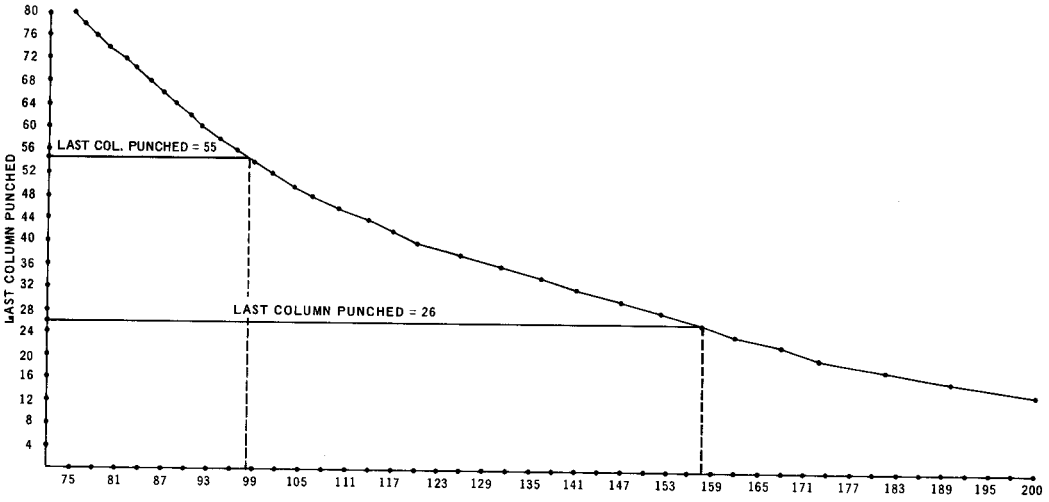


Figure 4–7. Card Punch Rate Versus Last Column Punched

4.2.2.2. Card Punch Mechanism

The basic punch mechanism is shown in Figure 4–8.

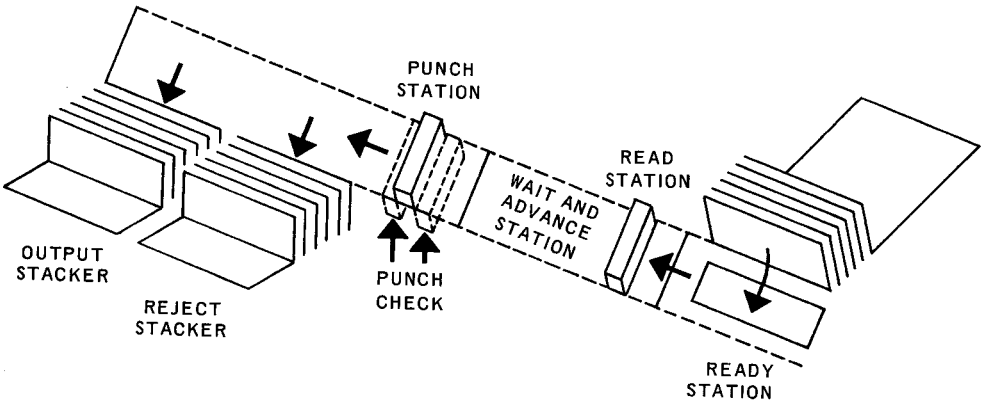
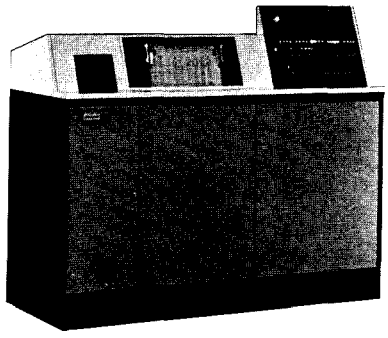


Figure 4–8. Basic Punch Mechanism with Special Features

4.2.3. Bar Printer



CHARACTERISTICS	
PRINTING METHOD	removable type bar
PRINT POSITIONS	96
PRINT FONT	63 characters
PRINTING SPEED	250 lpm with 63-character font
PAPER SPEED	25 inches per second (form skip speed)
PAPER SPACING	6 lines per inch
SPECIAL FEATURES	variable speed printing with 48-character font 250 lpm for alphanumeric lines 500 lpm for numeric lines 120 print positions 132 print positions 8 lines per inch

4.2.3.1. Bar Printer Timing

Figure 4–9 presents a general summary of the printer timing.

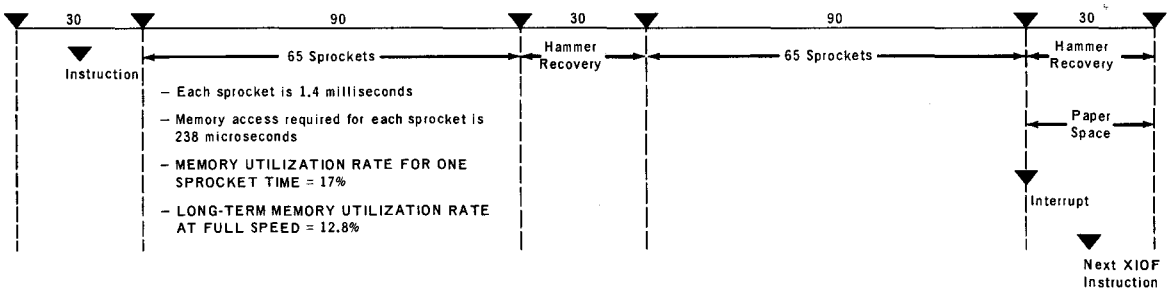


Figure 4–9. Basic Printer Timing

4.2.3.2. Bar Printer Mechanism

In designing a printer for the UNIVAC 9200 System, the same basic problem discussed in other parts of this manual was confronted and solved: How to retain the sophisticated system performance of a high-priced computer in an inexpensive system that can be afforded by a very small data-processing installation. The tangible result of the solution to this problem is the UNIVAC Bar Printer.

The bar printer is functionally similar to most commercial printers except that an oscillating type bar is used in place of the rotating type drum common to other printers. A simplified sketch of its operation is shown in Figure 4–10. Figure 4–11 is a photograph of the type bar removed from the chassis. The type bar oscillates horizontally in front of the paper. The character font is on the side of the bar facing the paper. Characters are printed when the print hammers strike the paper and push it against the selected characters. The motion of the bar, and thus its position relative to any given print hammer, is synchronized electronically. In effect, the printer knows which character is in front of a given hammer at any point in the print cycle. Directions to the printer from the computer control the hammers to be actuated.

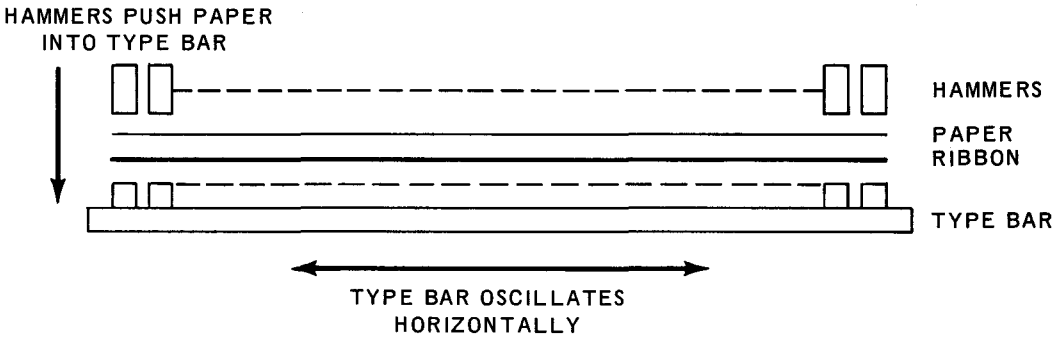


Figure 4–10. Basic Printing Configuration of Bar Printer

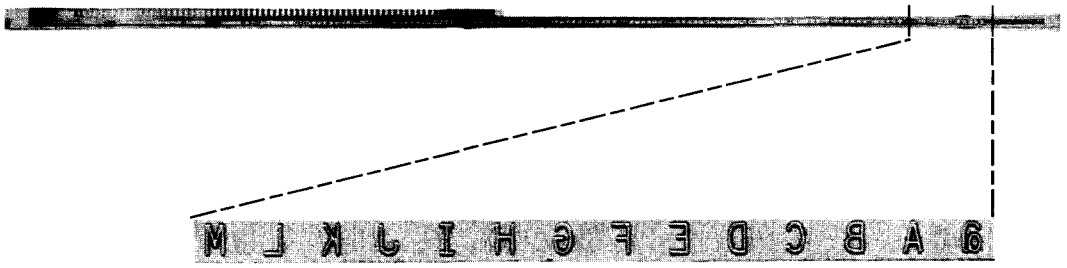
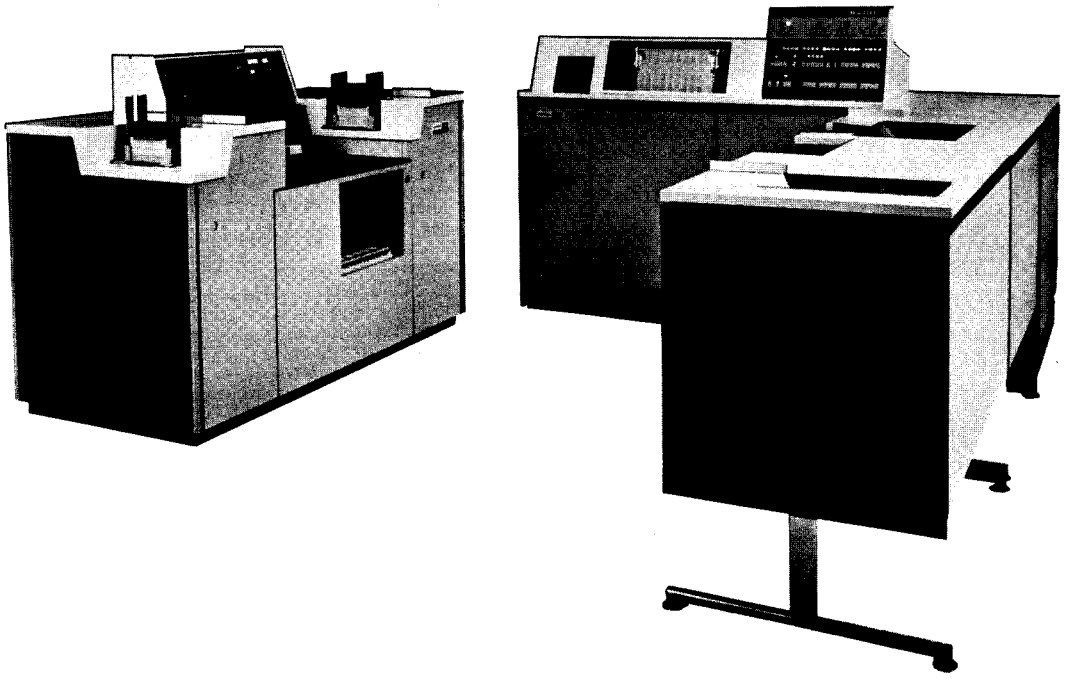


Figure 4–11. Type Bar Removed from Printer

The basic advantage of this very efficient printing technique is its simplicity and lower cost. Another advantage is the elimination of vertical smear or misalignment because the bar only moves in the horizontal plane. Also, the print bar can be changed in less than 60 seconds for quickly inserting special-character or foreign language fonts, and the easily changed type bar permits the printer to be optionally adapted for higher operating speeds when using a reduced type font.

4.2.4. The 9200 System with 1001 Card Controller



When the UNIVAC 9200 System is combined with a UNIVAC 1001 Card Controller, the result is an exceptionally powerful card processing installation. Above is a photograph of this combined system. The 1001 Card Controller can be used on-line or offline. Its chief offline function is collating. It has two card feeds, each of which can operate at 1000 cpm, for a combined card feeding rate of 2000 cpm. Card collating operations are thus performed very rapidly, but the 1001 Card Controller has multipurpose capabilities which also permit it to do card proving and editing, sorting, and statistical sorting. Computer processing functions such as comparison, addition, subtraction, and programmed multiplication are built into the 1001 Card Controller along with a core memory that has 256 positions of storage. Seven large-capacity stacker outputs provide considerable selection versatility. In short, the UNIVAC 1001 Card Controller is an extremely powerful card-handling device in its own right, but when it is incorporated into the UNIVAC 9200 System the resulting combination will eliminate many conventional punch card operations and result in a significant savings in machine time. Some of the multiple functions that can be performed with the combined system are as follows:

- *Multifile input* with merging and selection while processing.
- *Advance file search of master file(s)* concurrent with detail card reading and previous-record processing.
- *Offline use of 1001 Card Controller* for collating, editing, sorting, and proving at the same time that the 9200 System is computing and performing accounting machine operations.

The UNIVAC 9200 Card Controller System is offered in two configurations. In the basic configuration, card inputs are from the 1001 Card Controller; the card reader is not used. In the expanded configuration, the 400 cpm card reader is used with the 1001 Card Controller to provide a third source of card input. The 1001 Card Controller is quickly and easily incorporated into the 9200 System by way of the optional multiplexer channel in the computer. The 1001 also has a general-purpose plugboard for online use.

4.3. ERROR CHECKING

A comprehensive system of error checking within the processor and I/O units enables the status of the system during a fault condition to be determined rapidly and easily.

4.3.1. Processor

If any of the following processor or memory checks fail, the program is immediately halted, and an error indication is given by way of the console displays. The I/O units will complete any operation previously initiated.

- Memory Data Parity Check

  Every byte from memory is checked for odd parity. This check includes data and control information for the I/O units as well as processor data.

- Memory Address Check

  All memory addresses are checked to determine if they are within the memory address range of a given memory size. This also applies to I/O control and data.

- Restricted Memory Address Check

  The addresses of all memory accesses made under processor program state control or by an I/O data write into memory (except during initial load) are checked to make certain that they do not fall within the restricted memory area (addresses 000 through 063).

- Print Buffer Address Check

  All memory writes are checked to make certain that no data is written into the print buffer area of memory when the printer is busy.

- Divide Check

  Each quotient digit is checked for a value greater than 9 if the multiply, divide, and edit instructions or subroutines are used.

4.3.2. Card Reader

If any of the following abnormal or error indications are detected in the reader, the reader abnormal indication lights on the console, and details of the fault are displayed on the console.

- Offline

  Occurs when the unit is removed from program control by way of the console OFF-LINE switch and prevents any instructions from being accepted. No indication is given to the program unless an operation is attempted. This indication must be cleared manually by the operator.

- Hopper Empty

  Occurs when the input magazine is out of cards. This condition prevents instructions from being accepted but gives no program indication until an operation is attempted. This indication must be cleared manually by the operator.

- Stacker Full

  Occurs when the output magazine is full. This condition prevents instructions from being accepted but gives no program indication until an operation is attempted. This indication must be cleared manually by the operator.

- Misfeed

  Occurs when the input card has not properly entered the ready station. This condition prevents instructions from being accepted but gives no program indication until an operation is attempted. This indication must be cleared manually by the operator.

- Door Interlocks

  Automatically occurs when the casework is opened. The interlocks open to turn off power. This condition prevents instructions from being accepted but gives no program indication until an operation is attempted. This indication must be cleared manually by the operator.

- Card Jam

  Occurs when a card has not progressed properly through the output mechanism. An interrupt is given when a jam is detected. This indication must be cleared manually by the operator.

- Photocell Check

  Occurs when a card has not progressed through the read station properly. An interrupt is given when this error occurs. This indication must be cleared manually by the operator.

- Memory Parity

  Occurs when a data byte or a control byte has improper parity. An interrupt is given when this error occurs, the error indication must be cleared manually by the operator.

4.3.3. Printer

If any of the following abnormal or error indications are detected in the printer, the printer abnormal indicator lights on the console and details of the fault are displayed on the console.

- Offline

  Occurs when the printer is removed from program control by way of the console OFF-LINE switch. This condition prevents instructions from being accepted but no program indication is given until an operation is initiated. This indication must be cleared manually by the operator.

- Paper Low

  Indicates that the paper supply is exhausted. This condition is detected 15-1/3 inches beneath the print line. This detection occurs after a completion interrupt and will result in the rejection of the next order, or alternatively an interrupt if the next order has already been accepted. This condition is handled by the software to bring the system to a halt at the optimum point for replenishing forms.

- Fuse Detector

  Indicates that a condition has been detected which could eventually result in a blown actuator fuse. An interrupt is given to the program. This indication must be cleared manually by the operator.

- Door Interlocks

  Automatically occurs when the casework is opened. The interlocks open to turn off power to the printer. This condition prevents instructions from being accepted but gives no program indication until an operation is attempted. This indication must be cleared manually by the operator.

- Paper Run-Away

  Indicates that control of paper position has been lost. This detection occurs after a completion interrupt and will result in the rejection of the next order, or alternatively an interrupt if the next instruction has already been accepted. This indication must be cleared manually by the operator.

- Form Overflow

  Indicates that the code 001 has been detected by the Paper Loop Control during a line space operation. This detection occurs after a completion interrupt and will result in the rejection of the next order or, alternatively, an interrupt if the next instruction has already been accepted. This condition is handled by the software to give the programmers full control over vertical alignment.

- Parity

  Indicates that a data byte or a control byte has improper parity. An interrupt is given when this error occurs.

4.3.4. Punch

If any of the following abnormal or error indications are detected in the punch, the punch abnormal indicator lights on the console and details of the fault are displayed on the console.

- Offline

  Occurs when the punch has been removed from program control by way of the console OFF-LINE switch. This condition prevents instructions from being accepted but gives no program indication until an operation is initiated. This indication must be cleared manually by the operator.

- Hopper Empty

  Indicates that the input magazine is out of cards. This condition prevents instructions from being accepted but gives no program indication until an operation is attempted. This indication must be cleared manually by the operator.

- Stacker Full

  Indicates that the output magazine is full. This condition prevents instructions from being accepted but gives no program indication until an operation is attempted. This indication must be cleared manually by the operator.

- Stacker Jam

  Indicates that a card has not progressed properly through the output mechanism. An interrupt is given when a jam is detected. This indication must be cleared manually by the operator.

- Door Interlocks

  Indicates that the casework has been opened, and since hazardous parts of the mechanism may be exposed, the punch has been turned off. This condition prevents instructions from being accepted but gives no program indication until an operation is attempted. This indication must be cleared manually by the operator.

- Entry Checks

  Indicates that a card has not progressed properly through the entry portion of the mechanism. An interrupt is given to the program. This indication must be cleared manually by the operator.

- Photocell Check

  Active only if the read option has been installed. Indicates improper recognition of the card at the read station. An interrupt is given when this occurs. This indication must be cleared manually by the operator.

- Punch Check

  Indicates that improper punch die motion has occurred. An interrupt is given to the program.

- Parity

  Indicates that a data byte or a control byte has improper parity. An interrupt is given when this error occurs.

4.3.5. Multiplexer Channel

If either of the following two conditions occurs, a processor abnormal stop will result after the current device number and a channel status byte are stored. The processor will not be stopped for any peripheral condition occurring on the channel.

- Device Bus or Control Parity

  Indicates that the channel has received improper parity associated with a device number presented on the device bus or on the access of control data.

- Time Check

  Indicates that the proper signal sequences have not occurred on the interface and apparently some kind of stall condition exists. Most of these checks utilize a 70-microsecond delay which is used to monitor all interface signal sequences.

5. PROGRAMMING CONSIDERATIONS

5.1. GENERAL

This section presents a general introduction to programming and a description of programming aids which are applicable to the 9200 System. A description is also given of instructions, data formats, I/O interrupts, error checking, and other features that govern the operation of the 9200 System.

5.2. PROGRAMMING PRINCIPLES

Programming a computer means to direct it through a series of rigidly controlled operations towards a specific objective. The objective may be a complex arithmetical calculation, a payroll computation, or any one of hundreds of other applications. Ideally, the problem to be solved or the process requested of the computer would be expressed directly in the language of the user. Unfortunately, this ideal situation does not exist. The internal operation of a computer is controlled by electrical impulses which must be coded in some way to bridge the gap between the user's language and the electronic functions that comprise the computer.

There are certain economical and physical considerations in the structure of a computer which limit the nature of the coding and therefore restrict the nature of the language that can be used to communicate with a computer. For example, to represent the decimal number system directly in a computer without resorting to some form of coding, each electrical pulse would have to be capable of ten variations. A computer constructed to handle such pulses would be uneconomical because of the amount of circuitry required to keep the pulse variations separate and stable.

A reliable and effective way to represent data within a computer was suggested by the fundamental electronic operation of switching: on/off, current/no-current, voltage/no-voltage, and so forth. This approach reduces the number of required variations from ten to two, and it is less expensive to construct a circuit that will respond to the presence or absence of a pulse than one that responds to ten discrete variations of a pulse. A system in which only two values are permitted can handle any kind of decimal or alphabetic data, but to do so, a coded system of representation is required. Thus, if the two values the computer can handle are represented by 0 and 1, codes for the decimal numbers zero through nine might be: 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, and 1001. This system of representation is called binary, and it is the internal language of most digital computers.

Apart from the language difference, which erects a fundamental barrier to easy communication between man and computers, a computer must also be carefully directed through every step of its operation. For example, a problem such as 2 + 2 = ? cannot be expressed to a computer in just that way. The request must be more detailed; it must be a series of explicit directions that contain an implicit request. Thus, a computer cannot respond to the question "How much is 2 plus 2?" unless it is expressed in the following way:

1. Go to storage location X and transfer the contents found there to register A.
2. Go to storage location Y and add the contents found there to register A.
3. Transfer the contents of register A to storage location Z.
4. Print out the contents of storage location Z.

A complete statement of the directions to solve 2 + 2 = ? would also include instructions for storing the values to be added in locations X and Y. Once a problem is stated explicitly, it must then be converted into the binary language of the computer before it is meaningful to the computer. For example, instruction 2 in the preceding paragraph, "Go to storage location Y and add the contents found there to register A," may look like this in the computer:

01010101100000000000100111100111

Communication with a computer is thus a process of carefully describing in complete detail all the steps which the computer must follow to solve a problem and then translating these steps into a language in which all words are spelled with only two letters. This procedure must be performed faultlessly or the result of the computer operation will be incorrect.

In summary, the computer user would like to express the operations he wants performed in the rich, generalized language with which he is familiar, but economic considerations in the manufacture of computers rule out this form of expression and require instead that instructions be spelled out in considerable detail in a coded form.

5.3. PROGRAMMING PACKAGES

Fortunately, the computer itself is especially qualified to handle the problem of conversion between languages. A body of programs, called *language processors*, has been developed whose purpose is to accept as input an operation described in the users' language and to produce as output that same operation described in the language of the computer. The input to the processor is called *source code*, the output *object code*. The object code can then be used to direct the operation of the computer.

5.3.1. Report Program Generator

The basic language processor in the 9200 System is the Report Program Generator (RPG). The source language of the RPG is geared to the user who is interested in producing reports. The source language is divided into three parts: input, calculation, and output. There is a form for each part. On the input form the user describes the cards on which the input to the report is to be introduced. The input

cards are described in terms of the fields of data represented on the card. The user assigns each field a name and describes it in terms of its location on the card, its length, whether it is numeric or alphanumeric data, and, if it is numeric, where its decimal point is located. On the output form the user describes the format of the printed report. This description is also in terms of fields, assigned names, and field characteristics. Summary punched information can also be described on the output form. On the calculation form the user describes any calculations that are to be done on the input fields before printing and punching. The RPG produces from this source language an object program that will read the input described, do the specified calculations, and produce the printed reports and summary punched cards. The RPG thus provides a source language which is ideal from the point of view of tabulating personnel because it allows operations to be described in familiar tabulating terms, and it produces object code in which those operations are coordinated into one unified program.

5.3.2. Reproducer

In addition to the RPG, the UNIVAC 9200 software package contains a more specialized language processor called the Reproducer. The source language is designed to allow card gangpunch-reproduce operations to be described in familiar tabulating terms. Some of the functions that can be described in this language are as follows:

- Straight reproducing
- Selective reproducing by control punch
- Simple master card gangpunching
- Interspersed master card gangpunching
- Offset gangpunching
- Combined gangpunching and reproducing
- Sequence checking
- Selective reproducing by control field
- Punching counter-controlled consecutive numbers into detail cards
- Counter-controlled gangpunching
- Major-minor gangpunching

5.3.3. Assembler

Language processors such as the RPG and the Reproducer are extremely valuable to the user because they are expressed in problem-oriented source code. Certain operations not easily described in these problem-oriented languages may still be feasible for computer application. These operations are more conveniently described by a computer-oriented language processor, called the Assembler. The source language of the Assembler has the same syntax as the computer language. Consequently, any operation that can be described in computer language can also be described in Assembler language. However, the Assembler relieves the user from representing the program in the form of absolutely addressed binary instructions. Instead, the components of an instruction are represented in the form of symbolic operation codes and more familiar numeric systems. Alphanumeric labels are assigned to memory locations, and data and instructions are addressed by way of these labels.

5.4. SUBROUTINES

The Assembler source code language, although computer oriented, does not require that the user write his programs in the two-state binary representation required by the computer, thus greatly simplifying program writing while achieving maximum computer utility. However, if only the Assembler were used to write computer-oriented programs, the user would have to spend a significant portion of his time writing instructions which are common to most programs. These common functions are collected into subroutines which are provided in the 9200 System software package and which allow the user to devote more time to defining his specific processing needs.

5.4.1. Multiply, Divide, and Edit

The mathematical functions of multiplication and division and the logical function of editing are optional as single instructions in the 9200 System repertoire. In the absence of these optional instructions, the functions can be simulated by means of subroutines which are available for use with the 9200 System in object code form.

5.4.2. Preassembly Macro Pass

The form of some common subroutines varies according to their use. These variable subroutines are written in a general form, and parameters are established for them for each particular function. In the UNIVAC 9200 System parameters are established for general routines by a Preassembly Macro Pass.

Associated with the Preassembly Macro Pass is a *macro library*, which is a collection of routines written in a special *macro language*. The macro language is designed to allow the routines to be expressed in general form. The Preassembly Macro Pass reads the macro library from punched cards into the memory. *Macro instructions*, written by the user, are then read and used to select routines from the library and establish parameters for them. The particularized subroutines are then punched out in source code for subsequent assembly. Although used by the UNIVAC 9200 software system for its own purposes, this macro facility is also available to the user, who can establish his own library of routines.

5.4.3. Input/Output Control System (IOCS)

Subroutines to handle the operations concerned with reading input into the computer and producing output from the computer are available in a variety of forms. These subroutines are written in macro language. They are selected and parameterized by the user in a macro pass, and the resulting source code is assembled.

5.4.4. Loader

A Loader subroutine is provided to load programs and to initiate program execution.

5.4.5. Linker

The object code that goes into the composition of one program can be the result of more than one assembly. The output of each assembly is called an *element*. It is the function of the Linker to combine the elements making up a program into one unit.

5.5. PROGRAM TESTING

An important part of the programmer's job is to run his program on the computer, detect errors in the program, and modify his program to correct the errors. This run-and-correction procedure is called program testing, or debugging. Two debugging aids are provided with the 9200 System to help the programmer to examine and alter his program.

5.5.1. Memory Dump

The data and instructions pertinent to a program are stored in the computer memory. The Memory Dump subroutine permits the programmer to examine any portion or all of memory by printing out the contents of a specified memory area on the printer. The programmer, who knows what the memory should contain at each point in the program, can make a comparison between the data that should be stored and the data that is actually stored to determine where discrepancies have arisen. The Memory Dump subroutine can be incorporated into the program when it is assembled or linked, or it can be a self-loading deck.

5.5.2. Squeeze

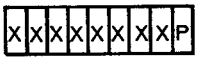
When a program error is detected, the programmer must modify his program. Generally, the program is modified by altering the source code of the element to be modified, by reassembling the element, and by linking the result of the assembly with the other elements of the program. However, if the change is minor, the programmer can modify his object code program deck directly by means of the Squeeze subroutine. To change the program with Squeeze, the programmer punches cards which specify the memory location to be altered and the information which is to go into the location. The output cards produced by Squeeze are placed at the end of the absolute program deck, and the program is modified when the deck is loaded.

5.6. DATA FORMATS

The basic unit of data in the 9200 System is an 8-bit byte to which a parity bit is added when stored in memory. A byte may represent a character or a number. When it represents a decimal number it may be expressed in two forms: unpacked and packed. In unpacked form, the byte is divided equally into a zone portion and a digit portion. The zone portion consists of the four most significant bits, and the digit portion consists of the four least significant bits. The sign of a decimal field in unpacked form is contained in the zone portion of the least significant digit.

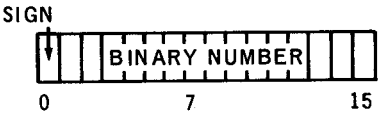
In packed form, each byte represents two decimal digits – one digit in the four most significant bits of the byte and one digit in the four least significant bits. The sign of a decimal field in packed form is contained in the four least significant bits of the least significant byte. Numerics which are to be compared or processed arithmetically are represented in packed format.

Binary numbers are represented as a signed halfword (refer to heading 5.7). Binary arithmetic is usually performed within the bounds of the general registers. Figure 5–1 shows the various data configurations in the 9200 System.

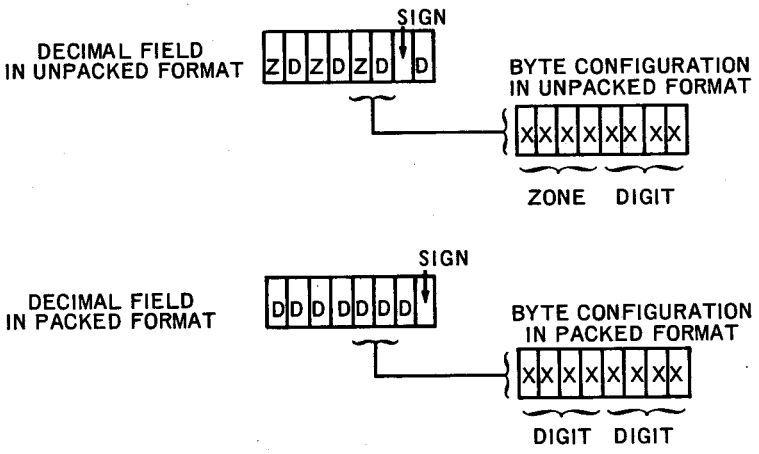


a. 1 byte = 8 data bits + parity bit

b. A character is represented by one 8-bit byte



c. A binary number is represented by 16 bits (15 bits + sign bit). It is two memory bytes or the contents of one general register.



d. Decimal Numbers

Figure 5–1. 9200 System Data Configurations

5.7. MEMORY ORGANIZATION

The basic capacity of main storage is 8192 positions. Each position is one byte. It may be expanded to 12,288 or 16,384 positions. The following types of information are stored in memory:

- data from input units
- processing results
- data for output units
- program instructions
- control information
- constants

Stored data may be alphabetic, decimal numeric, binary numeric, or special characters The memory cycle time is 1.2 microseconds. Memory location addresses are numbered consecutively from 0 through 8191 (12,287 or 16,383 if expanded). Each address specifies one byte. Bytes may be addressed singly or in groups. If a byte group is addressed, the starting address is always that of the most significant byte position regardless of how the data is processed. If the starting address of two consecutive bytes is even (2, 4, 6. . .8190), they are referred to as a half word. Instructions in the 9200 System repertoire are either four or six bytes in length. The instructions may specify either fixed-or variable-length fields. Field length is determined by a binary number in the instruction which is numerically one less that the actual field length required by the instruction; for example, a field length of five bytes is called for by a binary number of 0100 in the instruction.

The first 260 bytes of memory are assigned fixed functions in memory. Addresses 0 through 63 are restricted locations. These locations may be addressed only under special conditions. If these conditions are not met, an error will result.

5.8. REGISTERS

Certain portions of memory are assigned permanently as 16 general registers. Each register has a capacity of two bytes. The registers are divided into two equal groups. One group is used solely for processing functions, and the second group is used exclusively for input/output functions. The processing group is used whenever the computer is in a mode referred to as Processor Program State Control (PPSC), and the input/output group is used when the system is in I/O Program State Control (I/OPSC). The system is normally in PPSC but is switched to I/OPSC whenever an interrupt occurs. The double set of registers permits the information in the processing registers to remain unchanged until control is returned to the program. This approach is considerably more efficient and faster than using the same set of registers for both functions because the processing data does not have to be transferred to another part of memory for preservation and then returned to the registers after the input/output function is completed.

5.9. OVERLAPPED INPUT/OUTPUT

The speed of the processor memory permits the various system functions to be overlapped for maximum overall efficiency. This means that the ratio between the speed of the processor memory and the I/O units is high enough so that there is no delay caused by an I/O unit waiting for access to memory. This permits each I/O unit to operate at its optimum rated speed.

5.10. ERROR CHECKING

Error checking is described in detail in Section 4 under heading 4.3.

5.11. INTERRUPTS

Interrupts are an integral part of all phases of computer operation that use the I/O devices. An interrupt request is generated whenever an I/O unit is ready for servicing (read cycle completed, punch ready, printer ready, and so forth). It allows a break in the program to attend to the I/O unit until its requirements are satisfied. The point of this program break is recorded, and control is returned to that point after the I/O device has been serviced immediately. Thus, the speed limitation imposed on each unit is the inherent limitation of its normal electro-mechanical operation.

5.12. PROGRAM INSTRUCTIONS

The instructions that direct the operation of the system have the following main functions:

- control I/O units
- process fixed-length binary numbers
- process variable-length decimal numbers
- process fixed- and variable-length alphabetic data and special characters

The following three types of instruction formats are used to perform these functions:

RX – register to storage and storage to register
SI – instruction to storage
SS – storage to storage

5.12.1. RX Instruction Format

The RX instruction format is four bytes in length (see Figure 5–2). Instructions in RX format refer to one of the general registers by way of the register operand, R1. The registers are used for operations such as storing, loading, comparing, adding, subtracting, branching, and as index registers for all instructions. The operand 2 portion of the instructions is an address in memory containing information required by the instruction. Thus, an instruction in RX format may specify that the contents of a memory location be stored in a general register (specified by R1). The exact function (storing, loading, and so on) is determined by the OP code.

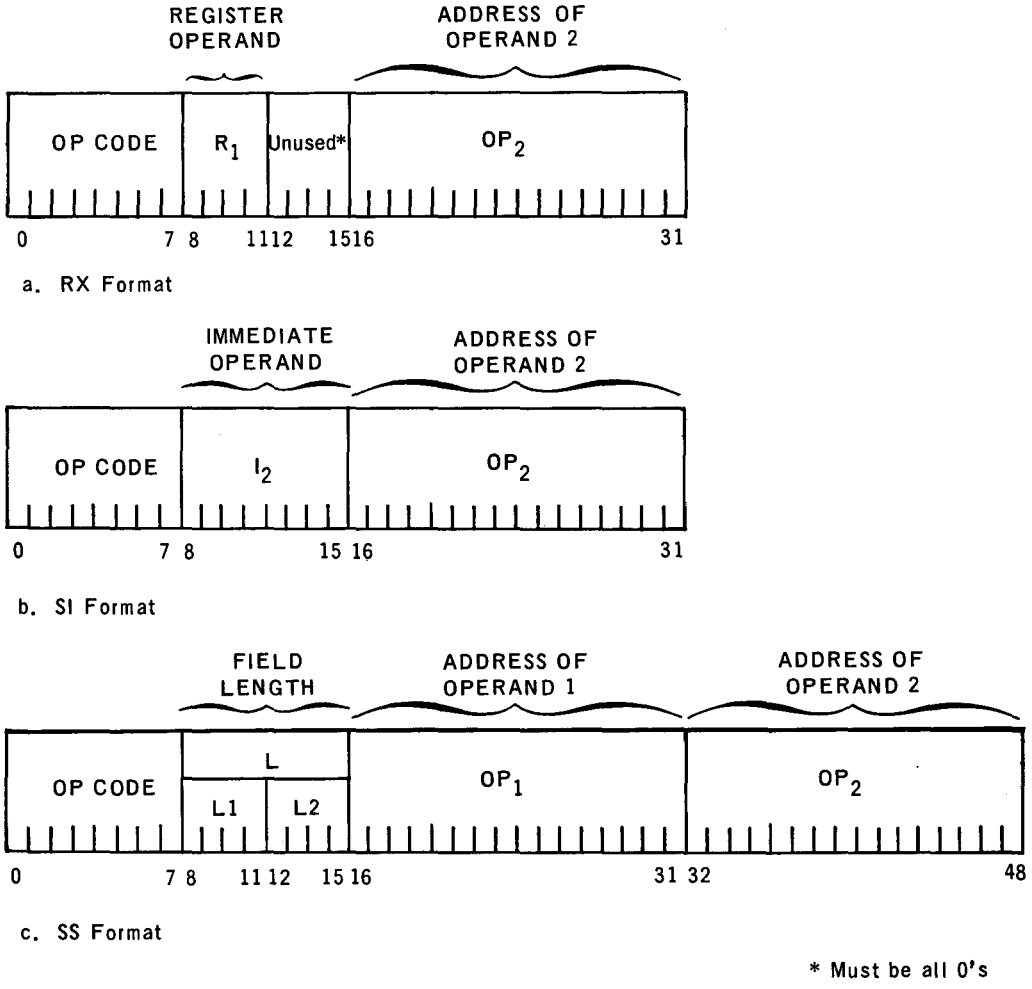


Figure 5–2. 9200 System Instruction Formats

5.12.2. SI Instruction Format

The SI instruction format is four bytes in length (see Figure 5–2). Instructions of this type contain an 8-bit value (I2) and the address (OP_2) of an operand in memory. Instructions in the SI format are normally used to store or test the value in the instruction (called immediate data) in or against the specified memory location.

5.12.3. SS Instruction Format

The SS instruction format is six bytes in length (see Figure 5–2). It contains two operand addresses and a length field (L) that determines the length, or the number of bytes, in the operands. In decimal operations, bits 8 through 11 (L1) control the length of operand 1, and bits 12 through 15 (L2) control the length of operand 2. In logical operations, bits 8 through 15 (L) control the length of both operands; operand length is specified by a binary number of bytes in the operand. Instructions in the SS format are normally used for the following:

Decimal Operations

- decimal arithmetic
- pack/unpack

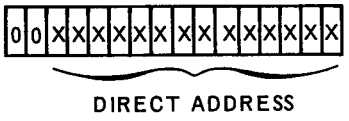
Logical Operations

- translate
- data transfer

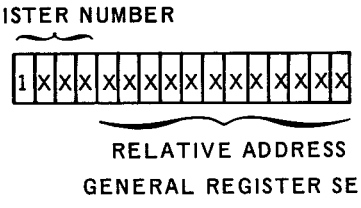
5.13. DIRECT ADDRESSING AND INDEXING

Operands in memory can be addressed directly by an instruction or indirectly by means of indexing. Direct addressing is specified by a 0 in the most significant bit position of the operands.

Indexing is specified by means of a 1 in the most significant bit position of the operand. In this case, the most significant bits contain the address of a general register that holds a base address. The remaining 12 bits are then added to the contents of the specified general register to yield the final, indexed memory address.



a. Operand Field Format for Direct Addressing



b. Operand Field Format for Indexing

Figure 5–3. Operand Addressing

5.14. OPERATION CODES

The operation code, which is an integral part of every instruction, determines the precise operation that will be performed by a given instruction. The effect of each operation code on the system is uniquely determined by internal circuitry. Thus, a given OP code bit configuration will always initiate the same operational pattern. OP code bit configurations are generally represented in hexadecimal. The following table shows the hexadecimal symbol for each 4-bit binary combination from 0000 through 1111.

4-bit binary group	hexadecimal notation
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

5.15. INSTRUCTION REPERTOIRE

The instruction repertoire for the 9200 System contains 32 instructions which control all of the processing and I/O functions of the system. A summary of the instruction repertoire is presented in Table 5–1. The instructions are grouped according to the operations they initiate in the system.

OPERATION	OP CODE	INSTRUCTION	MNEMONIC	FORMAT	INSTRUCTION TIMES (in microseconds)
Binary	40	Store Halfword	STH	RX	40.8
	48	Load Halfword	LH		40.8
	49	Compare Halfword	CH		40.8
	A6	Add Immediate	AI	SI	38.4
	AA	Add Halfword	AH	RX	40.8
	AB	Subtract Halfword	SH		40.8
Logical	91	Test Under Mask	TM	SI	No match or mask is zero 33.6 Partial or full mask 38.4
	92	Move Immediate	MVI		33.6
	94	AND	NI		33.6
	95	Compare Immediate	CLI		33.6
	96	OR	OI		33.6
	A9	Halt and Proceed	HPR		28.8
	D1	Move Numeric	MVN	SS	33.6+16.8(N)
	D2	Move Character	MVC		33.6+16.8(N)
	D4	AND	NC		33.6+16.8(N)
	D5	Compare Logical	CLC		50.4+16.8(N_E)
	D6	OR	OC		33.6+16.8(N)
	DC	Translate	TR		33.6+28.8(N)
	DE	Edit*	ED		See note 4
Decimal	F1	Move with Offset	MVO	SS	50.4+7.2(N_2)+12(N_1)
	F2	Pack	PACK		50.4+7.2(N_2)+9.6(N_1)
	F3	Unpack	UNPK		43.2+14.4(N_2)+9.6(N_1)
	F8	Zero and Add	ZAP		52.8+7.2(N_2)+9.6(N_1)
	F9	Compare Decimal	CP		52.8+7.2(N_2)+9.6(N_1)
	FA	Add Decimal	AP		52.8+7.2(N_2)+9.6(N_1)
	FB	Subtract Decimal	SP		52.8+7.2(N_2)+9.6(N_1)
	FC	Multiply Decimal*	MP		See note 4
	FD	Divide Decimal*	DP		See note 4
Branch	45	Branch and Link	BAL	RX	36
	47	Branch on Condition	BC		No branch 31.2 Branch 36
State Control	A0	Store State	SPSC	SI	48
	A8	Load State	LPSC		Load entire state word 48 Other than load entire state word 36
Special	A1	Supervisor Call	SRC	SI	24
I/O	A4	Execute I/O	XIOF	SI	Integrated I/O units 36 General-purpose channel 45.6
	A5	Test I/O	TIO		

*Available as options

NOTES:
1. Timing for all instructions assumes no indexing. Add 7.2 microseconds for each indexing operation.
2. N, N_1, N_2 = the number of bytes specified in the respective fields (L + 1, L_1 + 1, or L_2 + 1).
3. N_E = the number of most significant bytes that will compare identically between OP1 and OP2 in the CLC instruction.
4. Detailed timing formulae for the optional ED, MP, and DP instructions will be provided in "UNIVAC 9200/9300 Central Processor and Peripherals Programmers Reference", UP-7546 current version.

Table 5–1. Instruction Summary

UNIVAC

SPERRY RAND

UP-4086 Rev.2