GA34-0002-1

Systems

IBM System/7 System Summary

IBM

GA34-0002-1

Systems

IBM System/7 System Summary

IBM

Second Edition (September, 1971)

This is a major revision of GA34-0002-0 and merges the information formerly contained in the publication *IBM System/7 Modular System Programs Concepts and Facilities,* GC34-0012-0, thus obsoleting these two publications. New and updated information has also been included in this revision; therefore, users of the previous editions are urged to review this edition in its entirety.

Changes are periodically made to the information herein. Before using this publication in connection with the operation of IBM systems, refer to the System/7 Bibliography, GC20-1737, for the editions that are applicable and current.

The illustrations in this manual have a code number in the lower corner. This is a publishing control number and is not related to the subject matter.

Copies of this and other IBM publications can be obtained through IBM Branch Offices.

A form for readers' comments is provided at the back of this publication. If the form has been removed, send your comments to IBM Corporation, Genéral Systems Division, Systems Publications, Department 707, Boca Raton, Florida 33432. Comments become the property of IBM.

Contents

Preface

This publication is an introduction and reference summary to the IBM System/7; a high-speed, real-time computing system designed for applications requiring sensor-based input/output operations. Customer executives and programmers who need a summarized description of what System/7 is and how it can be used will find this manual helpful.

The subject matter is divided into 3 major categories:

- System design concepts and applications
- Machine units and operating features
- Program preparation and programming system facilities

No prior knowledge is necessary before using this manual as an introduction to the System/7 applications, although a familiarity with basic data processing system and sensor-based concepts will be helpful. For a comprehensive understanding of System/7 program preparation, the reader should also be familiar with the techniques dealing with program assembly by means of the macro assembler.

Related Publications

- *IBM System/7 Functional Characteristics,* GA34-0003, is a reference manual for programming the applications of System/7 in machine language.
- *IBM System/7 Modular System Programs (MSP/7) Programming Guide,* GC34-0013, contains detailed information for coding application programs with a macro assembler or in an assembler language.
- *IBM System/7 Installation Manual–Physical Planning,* GA34-0004, is a reference manual for installing the machine units and describes customer interfaces to the I/O modules (analog, digital, and 2790 Control).
- *IBM 2790 Data Communication System–Component Description,* GA27-3015, describes the 2790 machine units mentioned in this publication.

Chapter 1. Introduction

In this publication, "sensor-based system" is a term used to define a computer (IBM System/7), the equipment that can be attached to the computer, and the methods the computer uses to monitor and control the attached equipment. Electromechanical sensors installed at the equipment being monitored and digital data sources (such as manual entry devices) send analog or digital input signals to the computer. The analog or digital signals represent the status of the activity being monitored and are translated by the computer into meaningful data. The computer can be programmed to accept the input signals on a priority basis, measure and/or record the data, check the data against predetermined standards, and return output signals to the attached equipment. The output signals from the computer control the physical process during the actual time it is being monitored; this is what is meant by real-time sensor-based input/output operations.

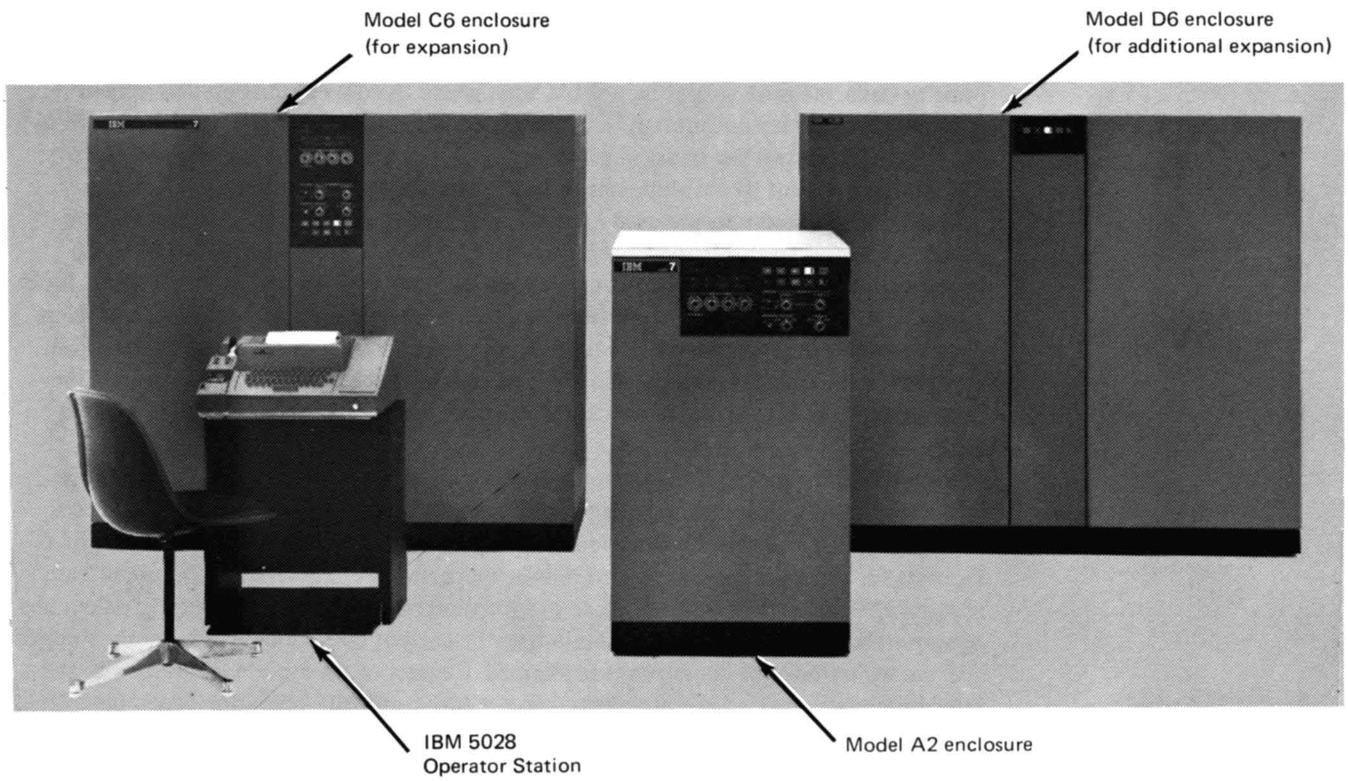
Sensor-based computers are used in many ways; for example, in monitoring of large numbers of manufacturing machines, in controlling one or more continuous or batch processes, or monitoring of one or more sensor-based inputs from a test instrument. Input signals to the computer may come from analog transducers of different types, and from digital sources, such as contact closures. Computer output (both analog and digital) can be used for control of displays, recorders and control mechanisms of many types. Computer output can be sent to magnetic disk storage, punched in paper tape, and printed on paper.

Because some operations being monitored and controlled by the computer are more important than others, they are assigned an order of preference—a *priority level*—based on urgency. The computer is designed and programmed to recognize the priority level of requests for processing in the various jobs being performed. These requests are presented as *interruptions* to the computer. If an interruption request has a higher priority than the operation being performed by the computer, the current operation is temporarily stopped, and the higher-priority operation is performed. Control of the system is returned to the interrupted operation in the program when the higher-priority operation is completed.

SYSTEM DESIGN AND APPLICATIONS

System/7 (Figure 1-1) is a high-speed, real-time computing system designed for applications requiring sensor-based input/output operations; it can be used as (Figure 1-2):

1. a single stand-alone computer
2. a computer interconnected with an *IBM host computer* (System/360, System/370, or 1800 Data Acquisition and Control System) via telecommunications
3. a computer interconnected with an *IBM host computer* (1130 Computing System) via the Storage Access Channel (SAC) attachment.



Note. The smallest configuration consists of a 5028 Operator Station with a Model A2 enclosure.

BR0779A

Figure 1-1. Units of the IBM System/7

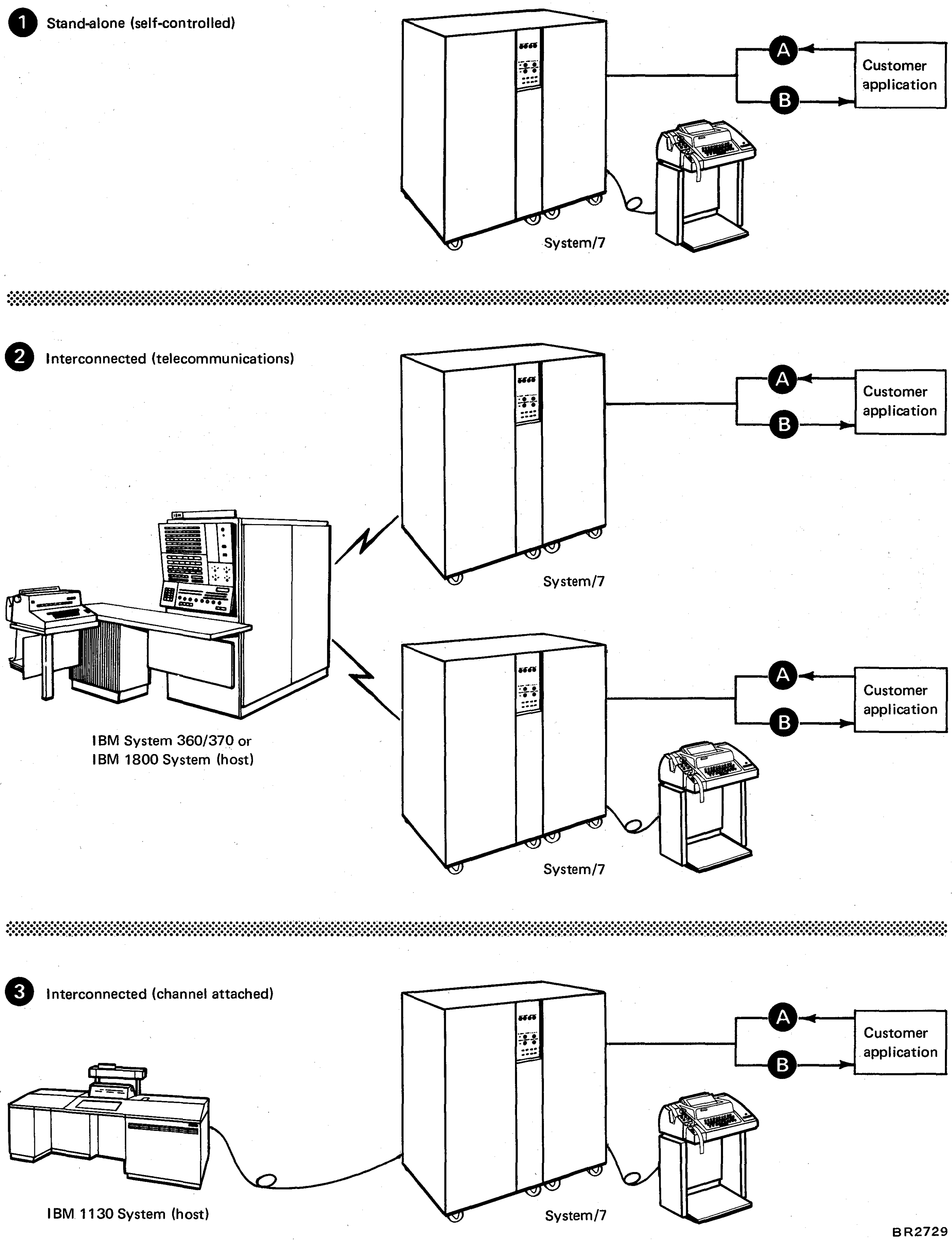


Figure 1-2. Stand-alone and interconnected systems

Designed for flexibility, System/7 can serve a wide variety of applications in:

- data acquisition
- process control
- plant automation
- laboratory automation

These applications share (Figure 1-2):

(A) the requirement of collecting data from instruments or sensors associated with a physical operation, and (except for data acquisition),

(B) the need for generating signals which in turn can be used to control some aspect of the associated physical operation.

So that control can be efficient and timely, many applications often require that data be acquired on a real-time basis. Applications differ in the balance sought between data acquisition, control, and the required response times.

In the design of a data acquisition system for either a data logging or a real-time operating environment, certain aspects of the user's equipment specifications must be considered. These include:

- Output signal levels
- Signal conditioning requirements
- Blocking of output signals
- Required sampling rates
- Resolution and accuracy
- System interruptions
- Operating environment

Knowledge of these specifications guides the systems designer in selecting the right components for a configuration.

Figure 1-3 shows an example of how System/7 can be used in a sensor-based manufacturing environment. Attached to a host and located near the physical process, System/7 acquires real-time data and controls (sensor-based) tasks while the host system prepares programs, stores programs and data, and delivers more computing power when required.

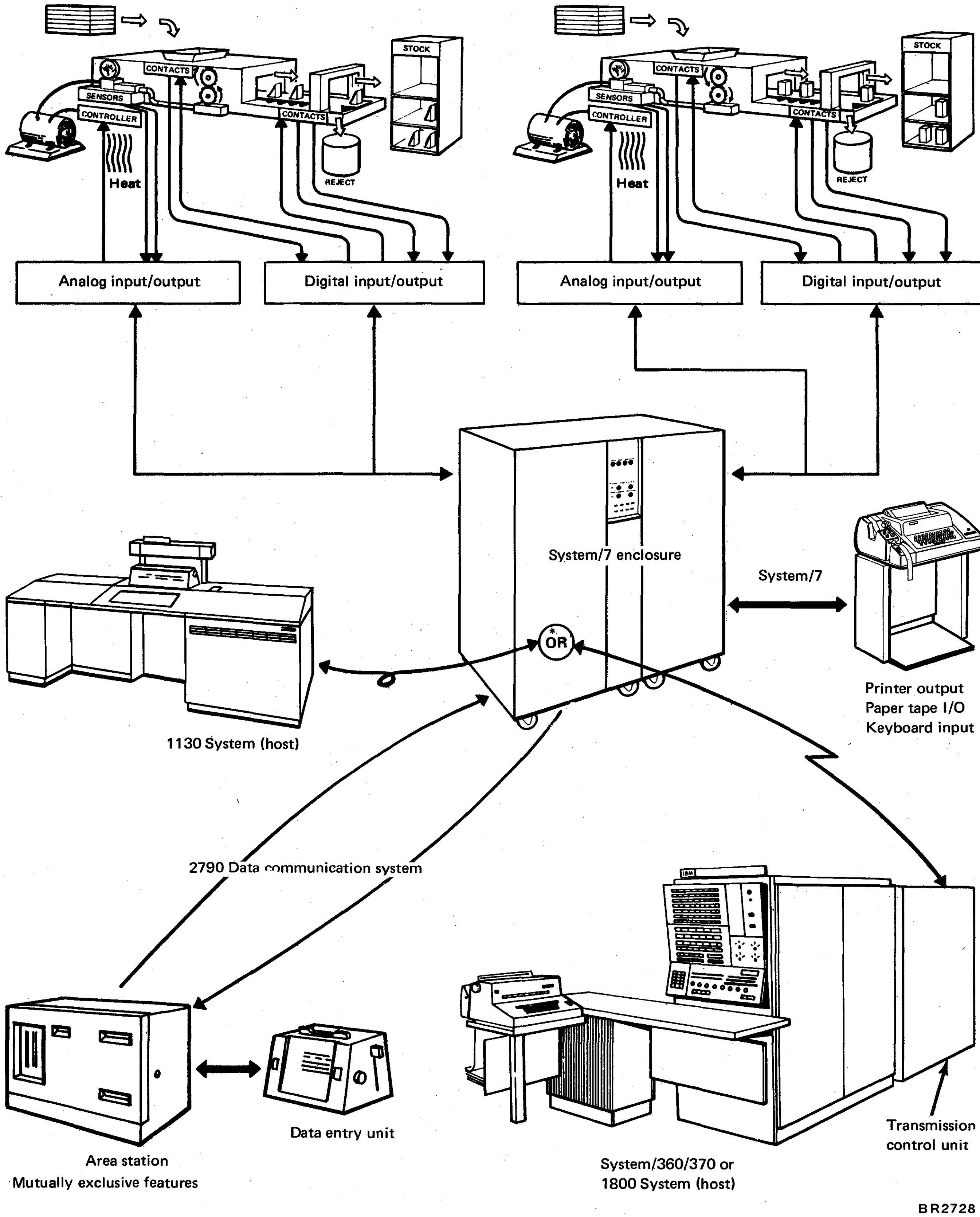


Figure 1-3. Sensor-based computing system in a manufacturing environment

Data Acquisition

Data acquisition applications are primarily concerned with the collection of data from instruments or sensors. Often this data is only recorded or partially processed as it is acquired, with the total processing being completed at some later time. The application may also require that some control be applied by the computer, such as opening a valve to admit a sample for analysis. Generally, however, these control functions are secondary in importance to data collection. Examples of applications in which real-time data acquisition frequently predominates include:

- Oceanographic research
- Water and air pollution monitoring
- Hospital patient monitoring
- Vibration or wind tunnel studies
- Testing

An example of the application of System/7 in testing is its use during the development of jet engines, when testing is conducted to validate the original design and to collect data for design improvements.

As Figure 1-4 shows, test signals–such as those from pressure transducers, thermocouples, flowmeters, and tachometers–can be wired directly to the System/7. The test stand operator starts and controls the test from the operator terminal, and specifies whether it is a calibration run or a static or a dynamic test. System/7 monitors the data directly at the test stand. The automatic range selection feature of the analog input system automatically adjusts for high and low level analog signals. Key performance parameters, previously chosen by the test engineer, are printed at the operator terminal to keep test personnel informed throughout the test. Crucial vibration instrument readings are checked to ensure that they are within prestated limits; if not, the operator is signaled, or the test stand is shut down. Open channels and abnormal readings are monitored, and the operator can instruct the computer not to accept these data points. At the conclusion of the test, the necessary calculations are performed and the test reports printed. Benefits derived from this application of System/7 are:

- Efficient utilization of test stands
- Automatic collection of data and repeatable calculation of test results
- Immediate availability of test results at the conclusion of test procedures
- Reduced total turnaround test time
- Ease of system calibration
- Continuous feedback of test conditions to the operator during the testing cycle; this ensures that a valid test has been completed
- Detection of crucial events that occur in short time intervals
- Better maintenance scheduling

Process Control

In process control applications, data acquisition is important, but there is typically more need for the control the computer makes possible. In general, computer process control involves the acquisition of analog and digital data from an industrial process, the calculation of control corrections to ensure the proper functioning of the process, and the application of appropriate control signals to the controlling elements in the process (see Figure 1-3).

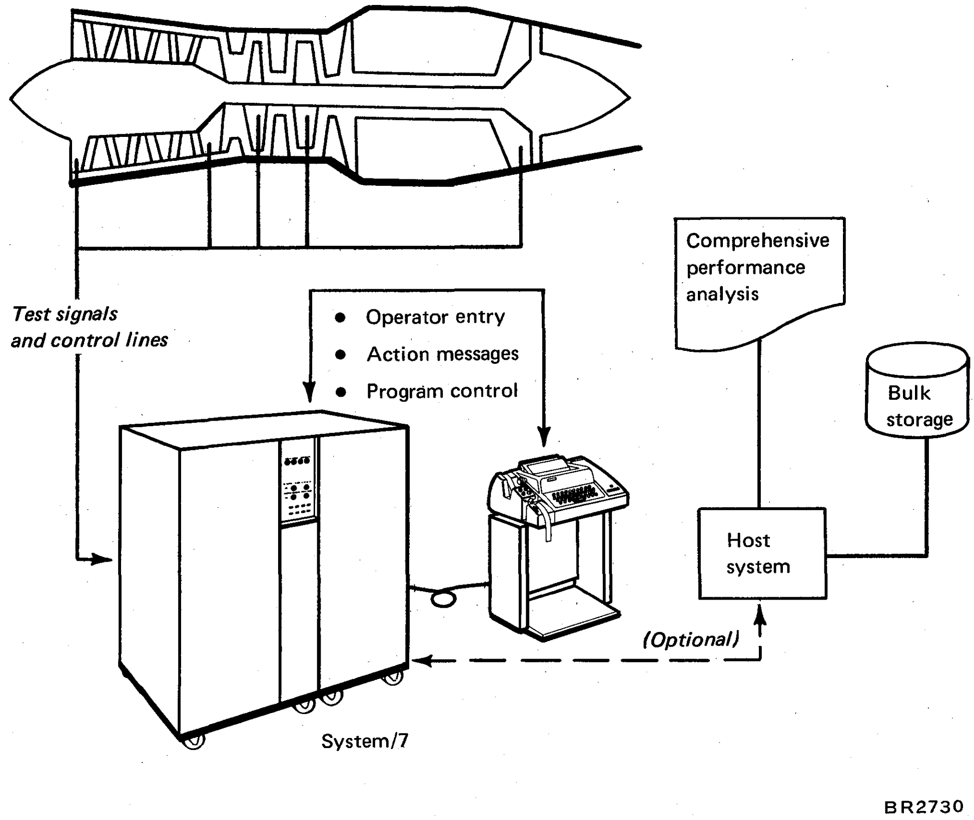


Figure 1-4. Data acquisition (testing a jet engine)

In order to be effective, the acquisition, calculation, and control must be done in real time; that is, the delay between acquiring the data and controlling the process must be such that the effectiveness of the control is not lost. In some processes, this may mean a delay of only a fraction of a second; whereas delays of several hours may be tolerable in other processes. Typical process control applications are:

- Vehicular traffic
- Paper making
- Petroleum refining
- Glass making
- Cement kiln operating
- Ore refining
- Electric generation control (load frequency control)
- Electric utility substation monitoring and control

An example of the use of System/7 in process control is the digital monitoring and control of high voltage (HV) substations and extra-high-voltage (EHV) substations from a centralized dispatching office. Supervisory control of distribution substations can be done through a discrete system or it can be included with the bulk transmission digital monitoring and control system.

Present-day objectives are (1) to maintain tighter control and (2) to continuously analyze system operating conditions in order to reduce outages and optimize the use of substation and line equipment. In addition, engineers need more operating data to plan, design, and improve equipment performance. Automatic logging of many variables yields the needed operating data.

In order to accomplish the objectives of substation monitoring and control, a basic stand-alone (see ❶, in Figure 1-2) or an interconnected (see ❷ and ❸, in Figure 1-2) System/7 capable of expanding may serve as few as 1 or 2 substations or as many as 50 or 60.

Separate communication lines between a System/7 in each substation and a larger IBM computer (host) in the central dispatcher's office, with backup lines to essential substations, link the network. Consequently, the reliability of the entire digital supervisory system is increased. Benefits derived from this application of System/7 are:

- Reduced manpower
- Placing of more substations under control of one dispatcher
- Quicker diagnosis of disturbances
- Rapid restoration of the system
- Transmittal of system-wide information to a central dispatcher
- Clear and concise presentation of system status
- Assistance in reduction of outages
- Increased safety through checking, classifying, recording, and displaying of clearance tags
- Improved maintenance
- Optimized investment in substation and line equipment
- Collection of more information for planners

Plant Automation

In contrast to the continuous nature of process control, plant automation is concerned with separate items—how they are manufactured, moved from location to location (as, for example, on a conveyor belt), and tested for proper operation. Here the need for data acquisition exceeds the need for process control. Typical data collection functions include counting items on a conveyor belt, measuring the performance of a device in response to computer-generated signals, or recording the movement of objects or material from one location to another.

In addition to data acquisition and process control, many plant automation applications entail more communication between man and machine than other applications. For example, operators must enter digital data such as their identification, inventory information, or a request for a particular series of tests. Thus, a system designed for plant automation must be equipped with suitable input/output devices for such communication. Examples of plant automation applications:

- Controlling warehouse stacker cranes
- Testing digital computers
- Testing automotive carburetors
- Quality control inspecting and sorting
- Production monitoring and control

An example of the use of System/7 in plant automation is the monitoring of production machines during their operation. This results in reduced unit costs, closer adherence to production schedules, and improved control of performance through the operation of production machinery at a sustained higher rate of efficiency.

The computer can monitor the actual piece-count or production rate and the machine status. Electrical connections between the installed automatic machinery and the System/7 enable it to automatically monitor and collect the necessary production information. Other related information is entered or requested through data communication terminals in the manufacturing area or through the computer's operator station, as shown in Figure 1-5.

The system then compares the collected machine performance data with stored performance standards. If the computer-collected data does not measure up to the standards, the system prints out a message defining the problem and its location, so that corrective action can be taken immediately. Because of such preventive action, nonproductive machine time is reduced and machine efficiency is maintained at a high level.

Besides timely corrective messages, periodic exception reports and end-of-shift summary reports can be produced to continually inform management of the performance of individual machines and departments. Some specific applications of production monitoring and control occur in loom production, glass and rubber production, transfer line and tablet line control, and shop floor control systems.

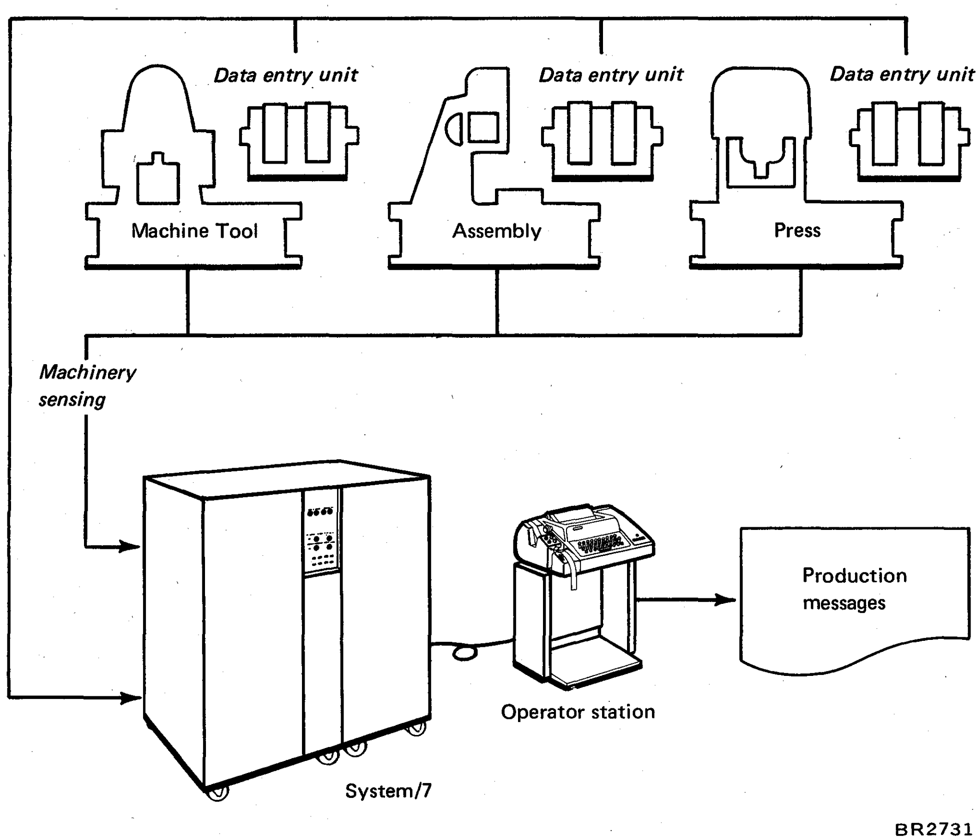


Figure 1-5. Plant automation (production monitoring and control)

An example of the application of System/7 in shop floor control is shown in Figure 1-6. The shop environment—which includes such things as the shop floor layout, the number and variety of machine tools, the manpower available, the number and variety of parts manufactured, and the number of orders released weekly—dictates that information be routed to and from all areas of the shop floor on a timely basis.

System/7, in conjunction with 2790 Data Communication System devices, captures the information at the source, as it occurs. This information, together with a production control data base (which might be stored on a System/7 disk storage module or at a host computer), goes to management in the form of exception reports about the shop floor and prompts meaningful management decisions. Significant benefits may be obtained in the following applications:

- Automatic validation of shipments according to plan, monitoring and testing of goods to specification, and the staging of goods for production according to plan
- Monitoring of plant facilities to identify machine utilization and availability, and tool utilization and maintenance requirements
- Monitoring of the attendance, productivity, and efficiency of production personnel
- Monitoring and reporting of quality defects
- Direct dispatching of jobs in a predetermined priority sequence
- Staging of finished goods for shipment, and direction of packaging and loading operations according to plan
- Counting of items as they pass each reporting station in the production operation

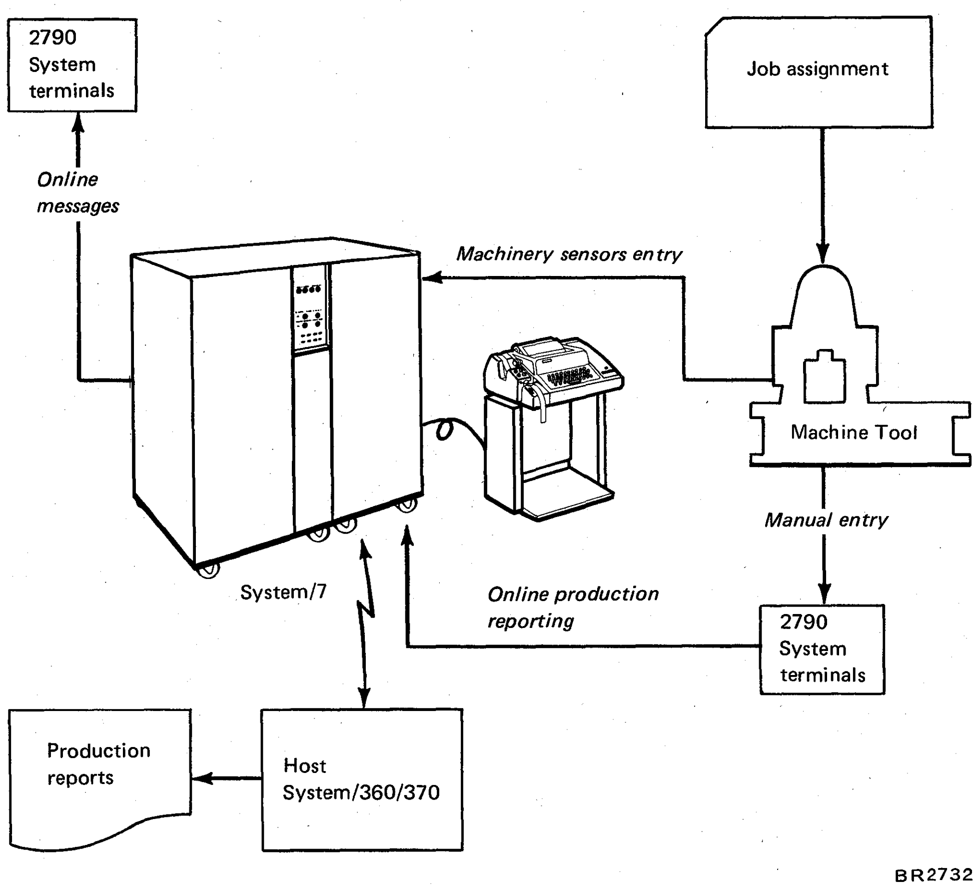


Figure 1-6. Plant automation (shop floor control)

Laboratory Automation

Laboratory automation applications cannot be easily categorized. They differ in their emphasis on data acquisition, process control, and man-machine communication. In industrial situations, laboratory automation may involve the collection, analysis, and summarizing of data from an analytical instrument such as a mass spectrometer or a chromatograph. A physical research laboratory, on the other hand, may be concerned with the control, collection of data, and data analysis in an X-ray diffractometer experiment. Or, a physical research laboratory may be concerned with measuring the physiological reaction of an experimental animal to externally applied stimuli. In a nuclear laboratory, the computer may record nuclear events in real-time for later analysis on a more powerful computer.

System/7 features—such as fast response to interrupts, powerful arithmetic capabilities, auto-ranging analog input, easy attachment to larger IBM computers, and modular expandability—make it ideally suited for laboratory automation applications.

An example of laboratory automation is the monitoring and controlling of laboratory chromatographs. In research, quality control, and medical laboratories, chromatographs represent a significant financial investment in equipment and operating personnel. Efficient operation to ensure accuracy, reproducibility, and maximum output per instrument are key objectives of any laboratory.

To help ensure that these objectives are met, System/7 monitors and controls laboratory chromatographs and analyzes and formats the collected data into meaningful reports. Many IBM 1800 users have proven the techniques associated with digitizing, data smoothing, averaging, base line correction, peak location, and allocating areas in the chromatogram to known constituents. With System/7 these techniques can now be employed to automate laboratories which could not afford the larger 1800 system.

When operating as a stand-alone unit, the System/7 permits a considerable degree of chromatograph automation. Full laboratory automation, which might encompass a variety of instruments and a number of analytical procedures, can be achieved by connecting one or more System/7s to an IBM host computer and analyzing the data portion of the application in the host computer. Some benefits derived from this application of System/7 are:

- Standardized operating procedures
- More productive use of the analyst's time
- Reduction in time taken by technicians, which allows them to initiate runs on more instruments
- Simplified calibration procedures
- Multiple analysis procedures
- Better reproducibility
- Greater accuracy of results
- Fewer test runs required
- Greater output from instruments
- Reduced turnaround time

SYSTEM COMPONENTS

Because of the versatility demanded by sensor-based data applications, modular design principles are used throughout System/7. This modularity means that a user can (1) choose the units of the system that are needed to perform functions required by the application and (2) expand an initial configuration easily as the application grows. With the modular design of System/7 and the ease of communication between System/7 and other data processing systems, many configurations are possible.

The four basic units of System/7 are (see Figure 1-7):

Processor Module

This unit computes at high speed and controls attached input/output (I/O) devices. It contains facilities for communicating with other computing systems.

Input/Output Modules

The two basic types of I/O modules, as shown in Figure 1-7, are sensor-based and disk storage.

Sensor-Based Modules

Sensor-based modules are of two types for maximum flexibility in meeting the application requirements. They are (a) an analog input module that receives analog sensor input from the instruments being monitored and sends this data to the processor module, and (b) a multifunction module that accepts analog and digital inputs, and transmits control information, in the form of analog and digital signals, to attached control devices. The multifunction module also has an input path for data entered manually from IBM 2790 data communication devices.

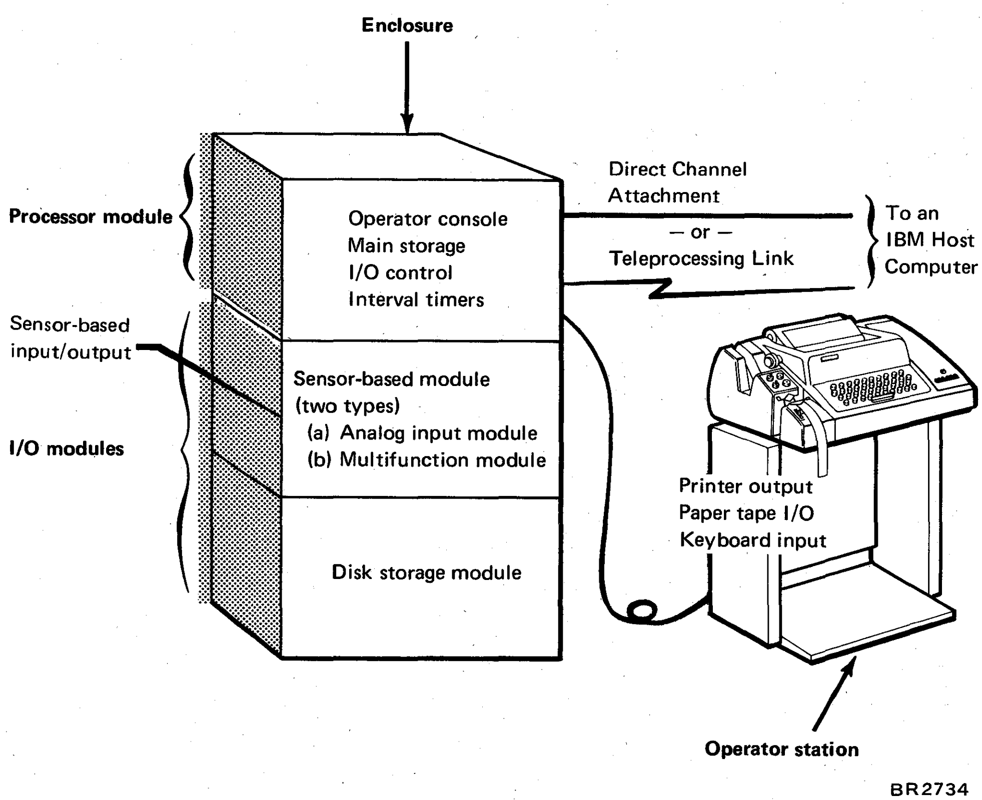


Figure 1-7. Modular design principles

Disk Storage Module

This high-capacity, random access data storage device is available in four models. Models 1 and 2 have two magnetic disks, one above the other. The upper disk can be removed from the drive and replaced with another interchangeable disk, allowing unlimited amounts of offline storage. Models 3 and 4 have a single disk that cannot be removed.

Enclosures

These units supply power, mounting space, and environmental protection for the processor and I/O modules. Bigger models accommodate more I/O modules.

Operator Station

Through this input/output unit, the system operator communicates with the system and enters programs into the processor module.

SYSTEM CONFIGURATIONS

System/7, because of its modular and independent design, offers a wide variety of system configurations. These range in size from systems with only a processor module and one I/O module to those with a processor module and eleven I/O modules, and in complexity from systems that are completely self-controlled to those that communicate directly with IBM 1800 Systems, 1130 Computing Systems or System/360 or System/370 Systems.

Figure 1-8 shows the self-controlled, stand-alone configurations. Any of these configurations can satisfy complete sensor-based applications. The smallest configuration is the two-position enclosure with a processor module and one I/O module. The I/O modules indicated can be any of those described in detail in Chapter 2. An operator station is included with each configuration.

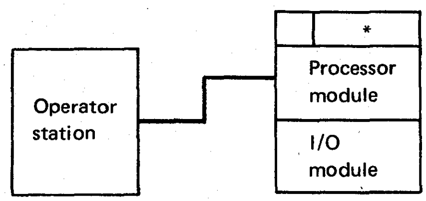
A basic stand-alone configuration, the lowest-cost configuration of System/7, can be expanded for more computing power by adding more storage capacity.

The configurations in Figure 1-9 show how System/7 can be used as a satellite processor (interconnected) in a distributed sensor-based application controlled by an IBM host system (System/360, System/370, or 1800 System). In this configuration the central data processing system controls many sensor-based applications at the same time by maintaining a supervisory role and directing the sensor-based systems to perform the actual data acquisition and control. The central data processing system can transfer both data and programs to and from the System/7. In the satellite processor configuration, the System/7 is also fully capable of independent operation as a complete sensor-based system.

Figure 1-10 shows how the System/7 can expand the capabilities of the 1130 Computing System. Without System/7, sensor-based data is collected *manually* for entry into the 1130. With System/7, it can be collected directly, in real-time, preprocessed, and presented to the host system (1130) in the most useful form for any further processing required. The 1130 host system transfers data and programs to System/7 storage and receives data from System/7. System/7 can also operate independently of the 1130 and fully process sensor-based data.

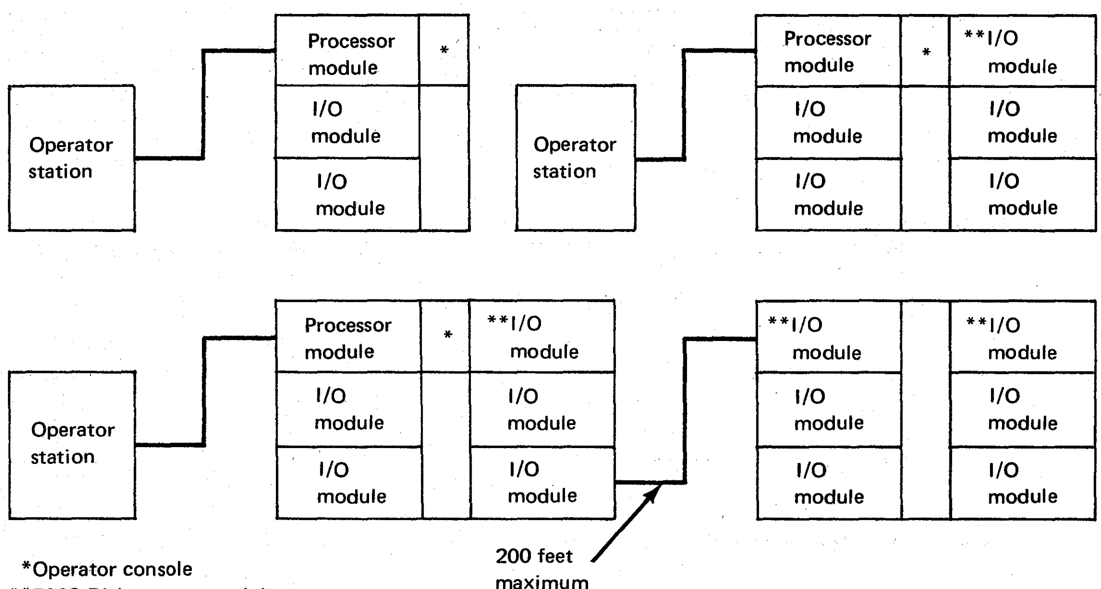
Stand-alone system (basic)

- Model AXX processor module
- One I/O module (any one of the three available models)
- Operator station
- Enclosure



Stand-alone system (expansions)

- Model AXX processor module
- Two to eleven I/O modules (any combination of the three available models)
- Operator station
- Enclosure



*Operator console
**5022 Disk storage module not recommended in this location

BR0780B

Figure 1-8. System/7 stand-alone configurations

Features of satellite processor configurations

- One or more of the stand-alone configurations with a model AXX processor module (contains an asynchronous communications control feature)
- Can be linked to System/360, System/370, or an 1800 System

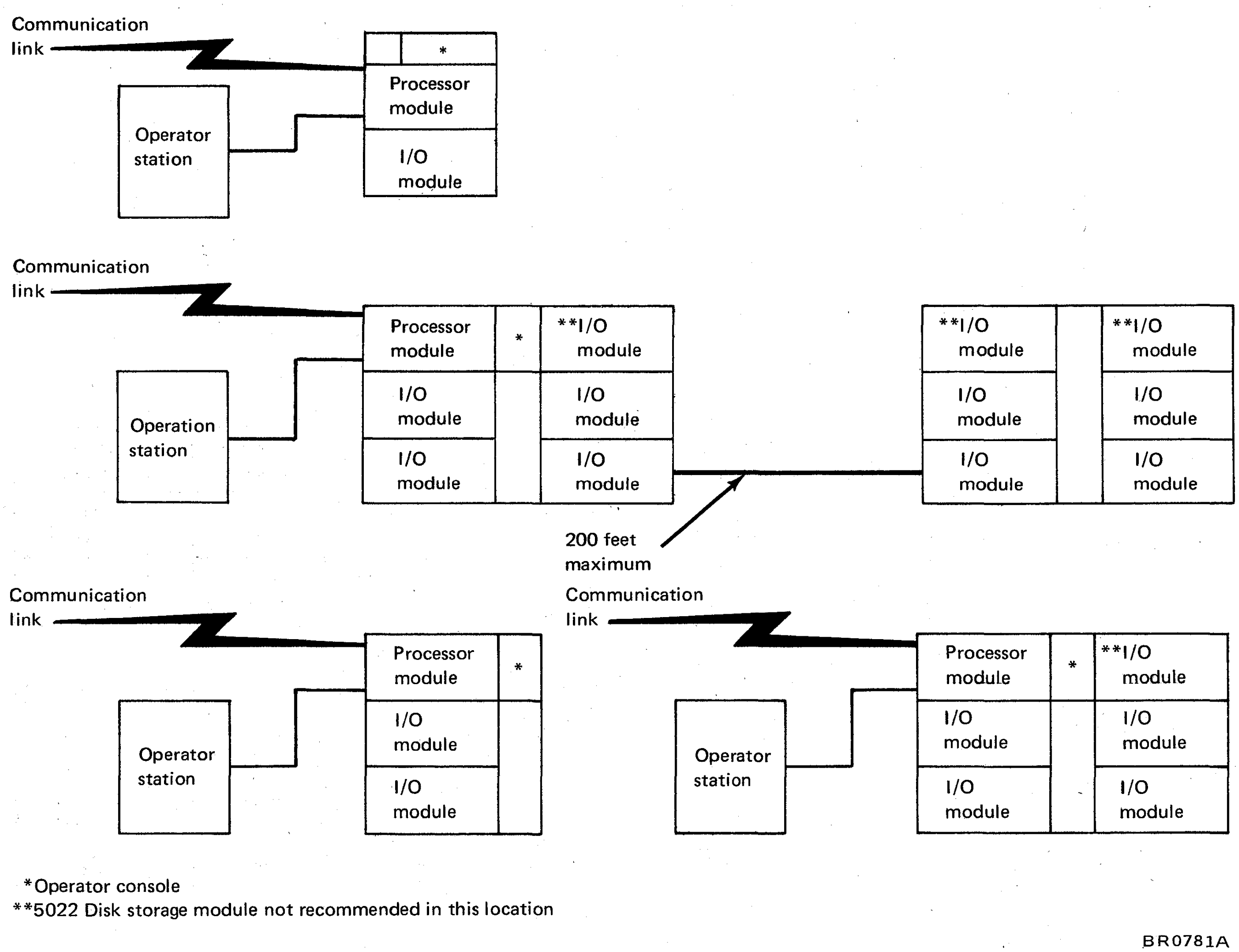
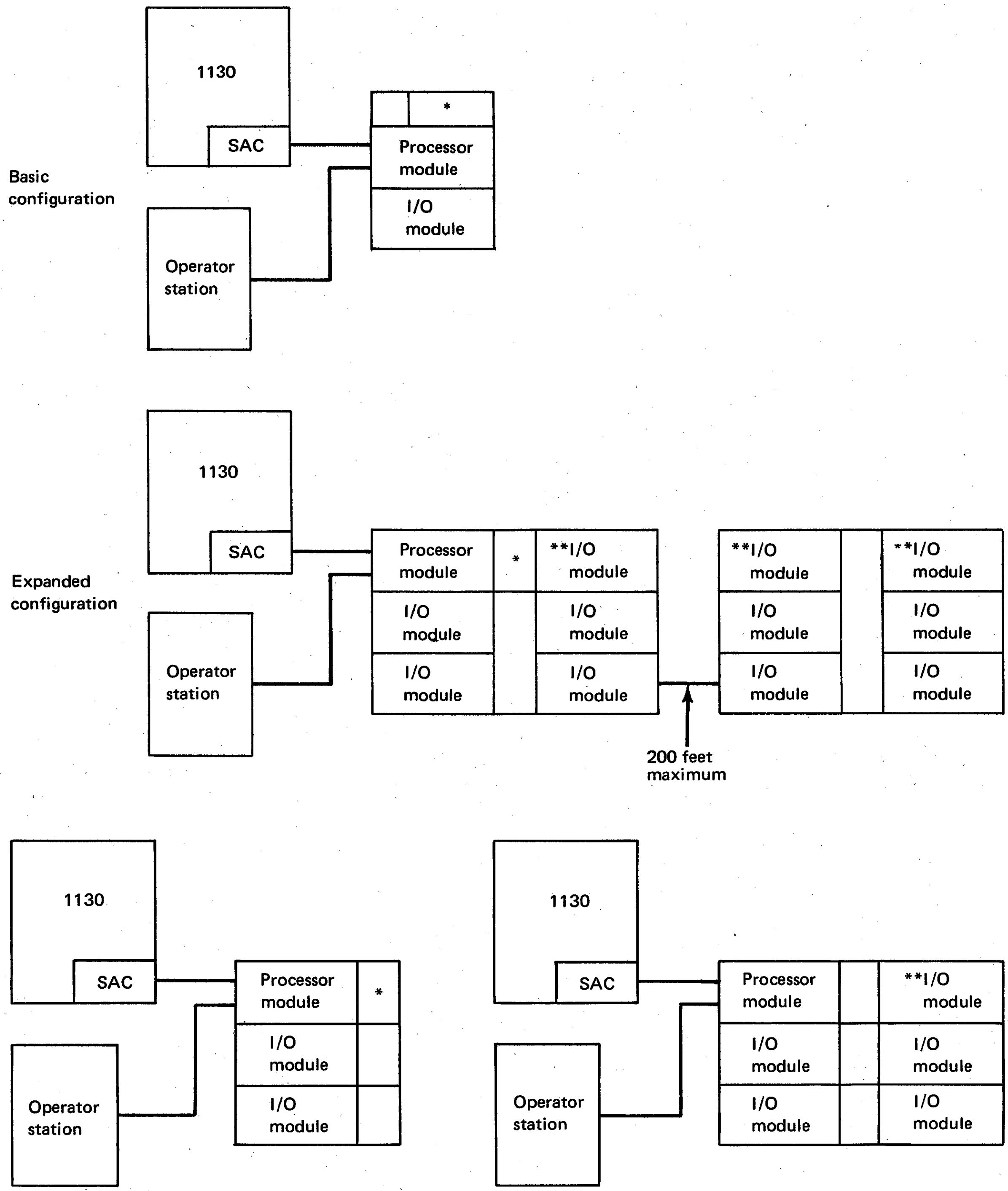


Figure 1-9. Sample configurations of satellite processors

Features of 1130-S/7 configurations

- System uses one of the stand-alone configurations with a model BXX processor module (contains an 1130 attachment)
- Data is transferred via the 1130 storage access channel (SAC)



*Operator console
**5022 Disk storage module not recommended in this location

BR0782A

Figure 1-10. IBM 1130–System/7 configurations

SYSTEM PROGRAMMING SUPPORT

IBM programming support (Figure 1-11) for the System/7 is designed for the user with a stand-alone system as well as for the user who has access to one of several larger IBM computing systems. This section of the "Introduction", covers the *highlights* of system programming, starting with a preliminary discussion of object program preparation, macro assemblers, and multisystem programming. For details of the following System/7 programming features refer to Chapter 3, "Program Preparation" and Chapter 4, "System Programming Facilities and Functions."

Modular System Programs for System/7 (MSP/7)

The purpose of "modular system programs" is to supply the System/7 customer with a programming package containing options he selects to customize routines needed to fit his unique operating environment. MSP/7 includes two options for preparing (coding and assembling) System/7 programs. The System/7 user without access to one of several larger IBM computing systems can code his programs and then assemble them on his System/7. The System/7 user with access to any one of four larger IBM computing systems (host systems) can code his programs and assemble them on the host system, using the host preparation facility furnished by IBM for that particular host system. There are two methods of preparing System/7 programs (Figure 1-11):

1. *Host preparation facility,* for coding source programs and assembling them on an IBM host computer (System/360, System/370, 1130, or 1800). This is a programming facility consisting of:
   - a macro language for powerful and simplified program coding; it extends the macro assembler library of a host system, thus making the host system capable of preparing System/7 machine language programs (object programs).
   - control programs for system supervision, sensor-based I/O, disk storage I/O, start/stop telecommunications, and a 2790 Data Communications System.
2. *System/7 Assembler,* for coding source programs and assembling them on a System/7 having 4,096 words of storage or more. The programming support consists of:
   - mnemonic operation codes, extended mnemonics, system utilities, assembler instructions, and arithmetic subroutines with a number of important advantages over coding source programs in the actual language of the computer.
   - a System/7 Assembler Program that converts a source program written by a programmer into a System/7 machine language program (object program) on paper tape.
   - system utilities: zero, dump, and patch routines.

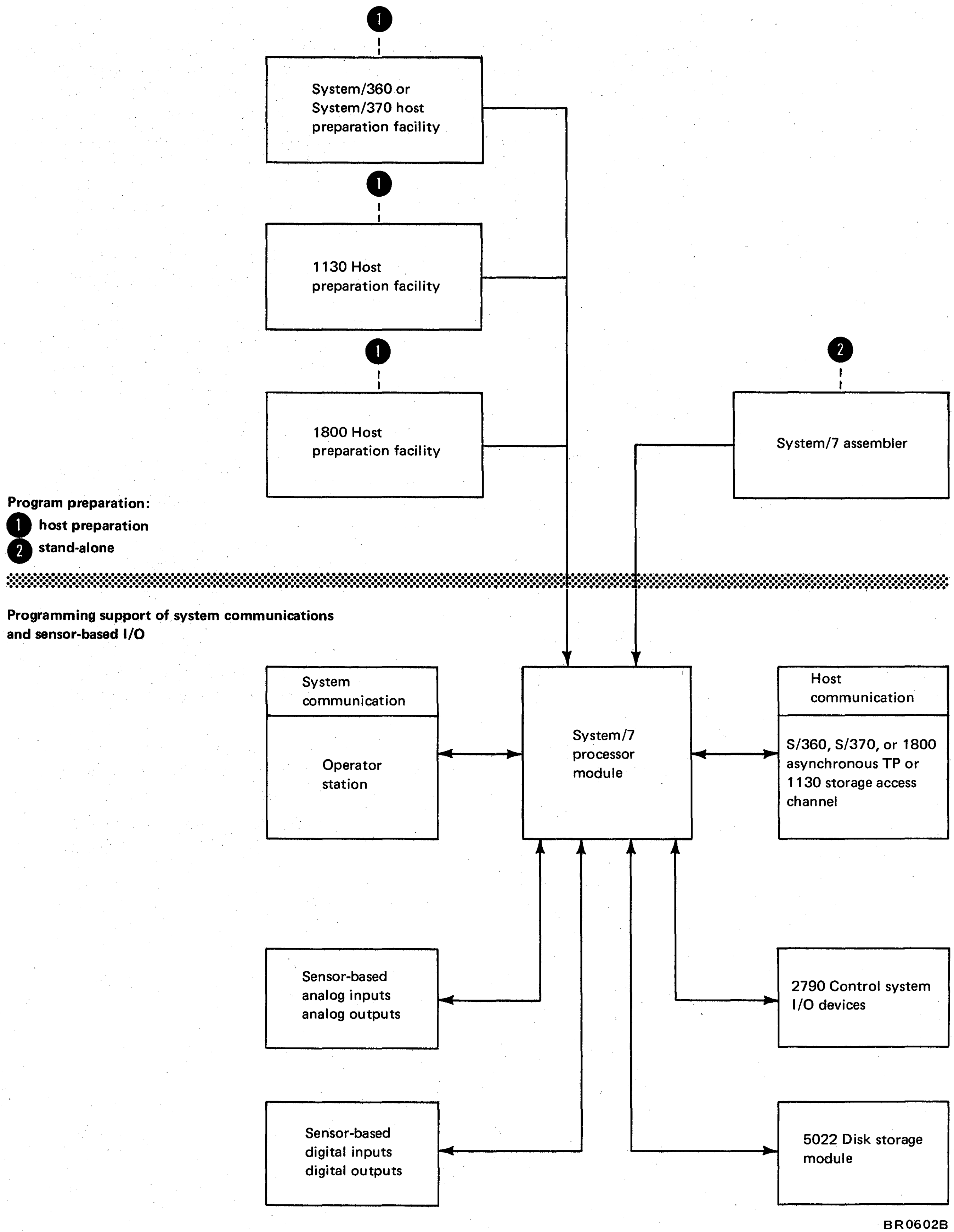


Figure 1-11. MSP/7 support of the System/7

System/7 Object Program Preparation

Figure 1-12 shows two basic methods of preparing object programs for System/7. As used in this publication, the term "assembler" denotes a *program* that is written, supplied, and supported by IBM to operate *on* a System/7 or on an IBM host-preparation computer. These assemblers convert the programmer's input statements (source program defining the instructions to be executed) into machine instructions (object code) that can be executed on the System/7.

Without the support of another computing system (host preparation facility), the System/7 Assembler (A in Figure 1-12) can assemble source programs and produce executable instructions (object program). This assembler is a single-pass, one-for-one assembler; that is, it requires only one reading of the source program, and it converts one source language instruction at a time into one System/7 machine instruction. A single loading of the System/7 Assembler can assemble several source programs.

System/7 object programs can be prepared on a host computer with the host preparation facility (B in Figure 1-12). The *macro assembler* (program) of the host computer assembles the source program and an *output handler* (program) converts the host system machine language instructions into a storage load module for the System/7. The *storage load module* is a logical group of machine instructions in a format suitable for loading into the System/7 main storage for execution. Depending upon the System/7 application, the object program in this case can be one or more storage load modules.

A macro assembler interprets and assembles macro statements coded by the programmer in his source program. Specifically defined for System/7, macros (macro instructions/statements) are similar to subroutines. They control System/7 input/output devices (analog input, digital input, 2790 control), perform commonly used system functions (task scheduling), and perform arithmetic and code conversion operations.

The MSP/7 host preparation facility has a library of macros that can be included (stored) in one of the host system's macro libraries. These macros serve a variety of functions. By selecting various options, the programmer can customize routines to fit a particular application. He combines IBM-written macro subroutines, user-written instructions, and user-written macro routines into source programs for assembly on the host system.

When the macro assembler of the host system encounters a macro, it generates a sequence of assembler language statements previously defined under that macro name, or it inserts the macro subroutine into the object program at that point (depending on how the macro instruction is coded). The output of this assembly is translated (by an MSP/7 output handler routine in the host system) into a System/7 storage load module comprising a supervisory framework of system functions and customer-specified and application-oriented routines.

As Figure 1-12 shows, storage load modules may be converted to 8-track paper tape and loaded into System/7 through the operator's paper-tape reader, or they may be transferred directly to System/7 via a channel attachment or teleprocessing link.

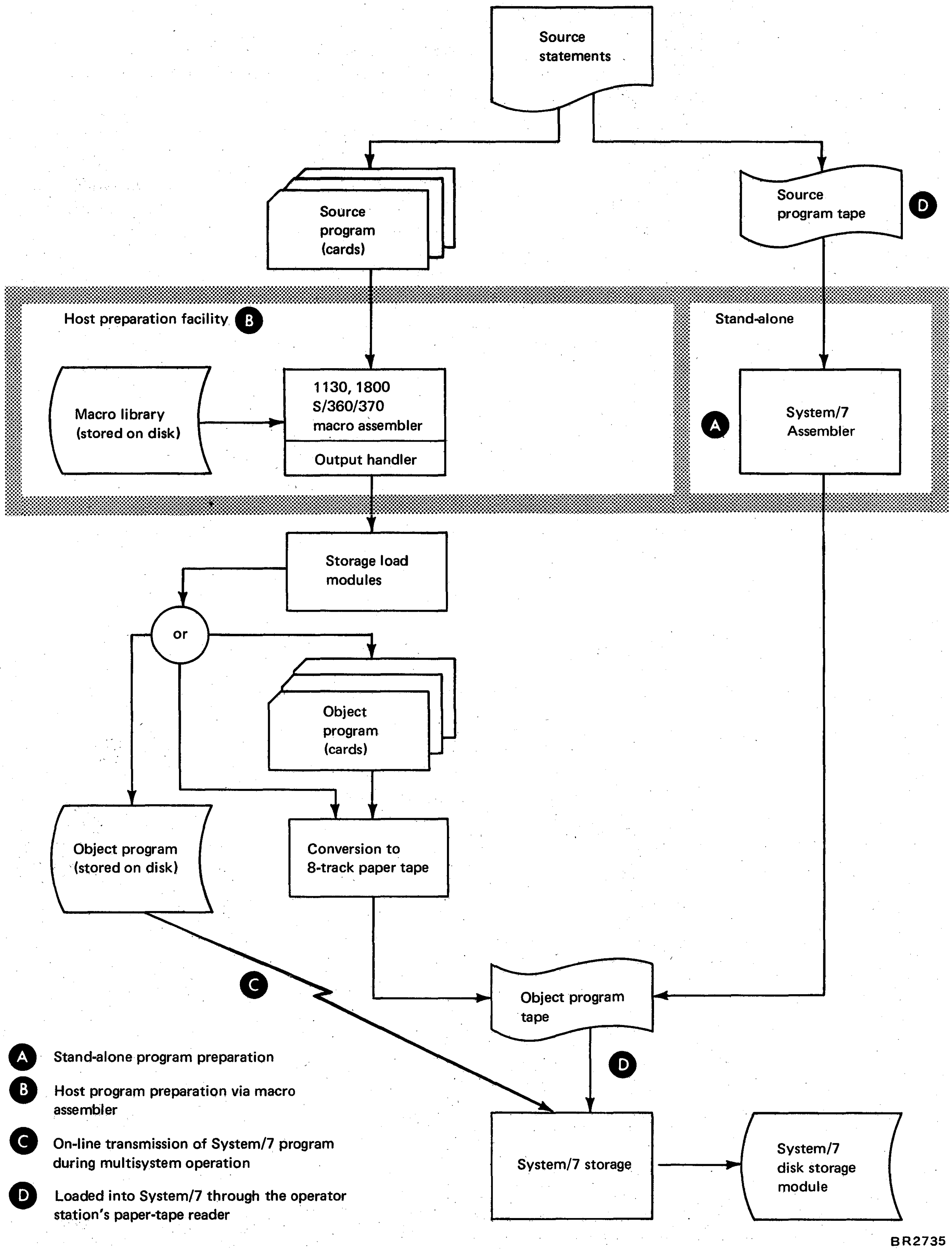


Figure 1-12. System/7 object program preparation

IBM Host Preparation Computers

As used in this publication, the term "host preparation facility" refers to the *program support* that IBM supplies System/7 customers for preparing (assembling) their programs on a larger IBM computing system (host system). These IBM host preparation computers are an 1130 Computing System operating under 1130 Disk Monitor System Version II (DM2); an 1800 Data Acquisition and Control System operating under 1800 Multiprogramming Executive Operating System Version III (MPX); or a System 360 or System/370 operating under the Disk Operating System (DOS) or the Operating System (OS). As a multisystem, the IBM 1130 can be directly connected to the System/7, by the addition of the 1130 attachment feature to the System/7 processor module (BXX). The operation of a System/7 and an 1130 for multisystem operation is enhanced by the use of the 1130 Distributed System Program (DSP), a program facility with support routines as extensions to the 1130 Disk Monitor System, Version II.

An IBM 1800 or System/360 or System/370 communicates with the System/7 through asynchronous communications control feature connected to the System/7 processor module (AXX). An extension to the 1800 Multiprogramming Executive Operating System, the 1800 Distributed System Program (1800 DSP), controls the operation of System/7 or several System/7s coupled to an IBM 1800. System/7 communicates with System/360 and System/370 by means of the currently supported telecommunication access methods for these systems.

Multisystem Program Support

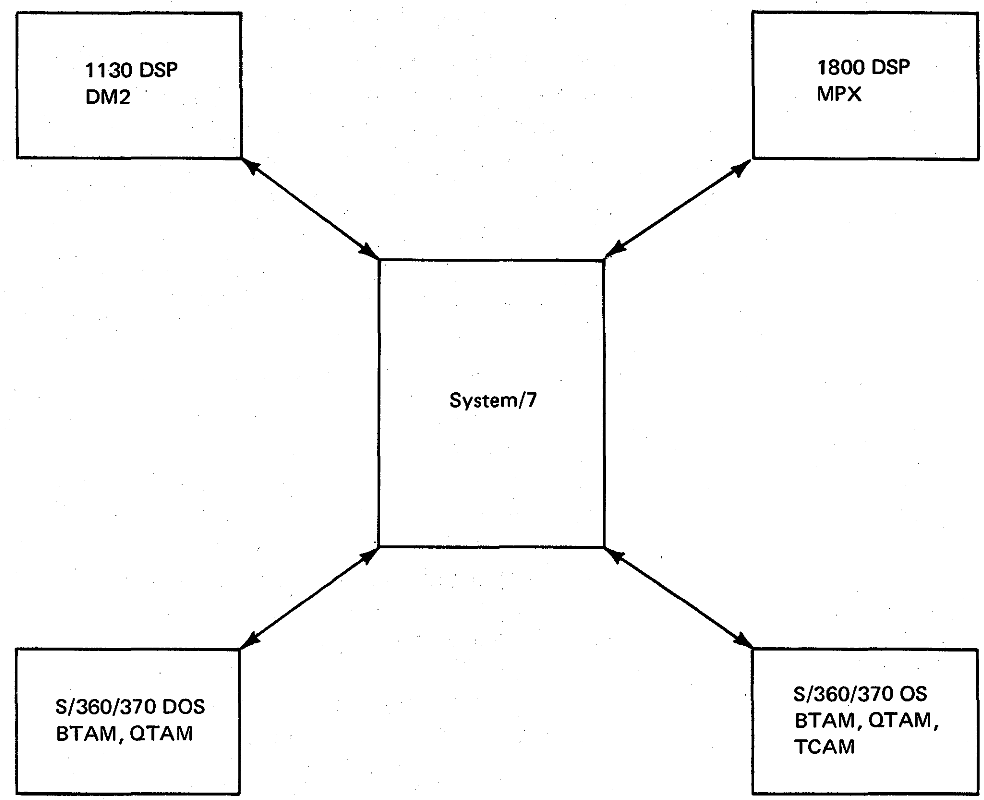
Under this heading, there are two broad subjects for consideration (Figure 1-13):

- The Distributed System Programs (1130 DSP, 1800 DSP) for an 1130 or an 1800 interconnected to a System/7.
- System/7 communication with the System/360 and System/370 by means of the current telecommunication access methods for these systems.

Sensor-based applications sometimes require systems that can furnish large storage facilities or data reduction capabilities. A configuration appropriate to many of these applications is a multisystem configuration, in which:

- A small computer, designed for sensor-based applications, is connected directly to the sensors.
- The small computer is connected to a larger host computer that supplies the large storage facilities or data reduction capabilities.
- The larger computer can communicate with one or more smaller computers.
- The larger computer can be used for local processing, as well as for preparation of programs for execution on the smaller computer.

This multisystem concept links computers together so that each can communicate with and draw upon the resources of the others.



1130 DSP — 1130 Distributed System Programs
1800 DSP — 1800 Distributed System Programs
BTAM — Basic Telecommunications Access Method
QTAM — Queued Telecommunications Access Method
TCAM — Telecommunications Access Method
OS — System/360/370 Operating System
DOS — System/360/370 Disk Operating System
DM2 — 1130 Disk Monitor System Version 2
MPX — 1800 Multiprogramming Executive Operating System

BR2736

Figure 1-13. Host communication and operating facilities

Distributed System Operation

The term "distributed system" is used in this publication to denote a system where a small sensor-based computer and a large host computer operate as a single entity sharing computing facilities. Each user and program has access to those computer facilities that are required to solve an application or problem. Several of the advantages of a distributed system in sensor-based applications are:

1. Computers can be located where they are most convenient for the users.
2. Each computing system can be assigned to operate on that portion of the application for which it is best suited.
3. Reliability and efficiency is increased through separate processors.

There are several levels of complexity in distributed system operations:

1. At the lowest level the System/7 acts independently of the larger computer to which it is linked. A program can be prepared on the larger computer under host preparation facilities and loaded into the System/7 via multisystem communication. System/7 can occasionally transmit data to the larger computer.
2. A second level of operation is the procedure by which the System/7 acts in consort with the larger computer for the solution of a problem. The problem may be the transferring of data to, or the initiation of programs in, the larger computer. Here programs for the solution of an application reside in both computers.
3. At the third level of operation, the System/7 and the larger computer operate not only in concert (in terms of second level) but also in full multisystem fashion where data, programs, and tasks are exchanged between the two systems. In this mode of operation, the facilities of full distributed system support are present. A System/7 can communicate with:
   - an IBM 1130 via the 1130 Storage Access Channel Attachment.
   - an IBM 1800, System/360, and System/370 via the Asynchronous (start/stop) Communications Control.

For multisystem communication, the following programs are available:

1. Between System/7 and 1130: 1130 Distributed System Program (1130 DSP).
2. Between System/7 and 1800: 1800 Distributed System Program (1800 DSP).
3. Between System/7 and System/360 or System/370: DOS BTAM/QTAM or OS BTAM/QTAM/TCAM.

System/360 and System/370 Telecommunications Access Methods

IBM programming support for teleprocessing systems takes the form of access methods under OS and DOS, which operate on the System/360 and System/370.

The principal function of a telecommunications access method is to control the transmission of information between a computer and remote teleprocessing equipment in much the same manner as other access methods support other types of input/output equipment. Thus, the programmer designs, writes, and tests his application routines in the usual manner; and he performs input/output operations by means of macro instructions supplied by the access method. He may also develop his own macro instructions to replace or augment those supplied by the access method. *For purposes of explanation, the MSP/7 macros for sensor input/output might be considered as a sensor-based access method.*

Three telecommunications access methods operate on the System/360 and System/370: (1) Basic (BTAM); (2) Queued (QTAM); and (3) the Telecommunications Access Method (TCAM).

BTAM is designed to furnish the basic modules for constructing a teleprocessing program, including routines for controlling a variety of terminal units, communication lines, and transmission control units. With a minimum of system overhead, it not only provides the basic tools to build a sophisticated system but also adapts easily to support special configurations. BTAM has the basic power to perform the following System/7 tasks:

- Poll and receive messages
- Address and send messages
- Dynamically chain input buffers
- Detect and correct errors
- Write output buffer chains
- Translate code

QTAM includes the BTAM capabilities and, in addition, extensive queuing facilities. QTAM is directly useable without modification to a number of common teleprocessing applications; for example, in data collection and message switching. QTAM provides the basic means for communicating to a System/7, such as:

- Controlled and automatic terminal polling and message receipt
- Controlled and automatic terminal addressing and message transmission
- Input/output buffering
- Error detection and checking
- Message queuing, logging, and routing
- Code translation

Programs written with the System/7 communications macros enable the System/7 to simulate the 2740 Communications Terminal Model 1 with its record-checking feature and—depending upon the communication method selected and the hardware configuration needed—its addressing feature. This feature accommodates several System/7s and 2740 terminals of the same type on one line.

Because TCAM has a high degree of compatibility with QTAM, converting to TCAM is easy. Furthermore, QTAM and BTAM message control programs servicing other lines and networks can co-reside with TCAM in the same System/360 or System/370. Other advantages:

- TCAM operates under control of the System/360 Operating System (OS).
- TCAM combines the broad range of device support found in BTAM with the broad range of applications of QTAM.

To construct teleprocessing applications with TCAM, the user combines a message control program and one or more TCAM application programs. Defined and generated by the user with TCAM macro instructions, the message control program describes the teleprocessing network and specifies the device-dependent handling needed to insulate the application programs from device-dependent considerations. Different applications may have different handling requirements, and alternate paths and procedures must be developed to meet these needs.

Application programs can be developed separately—either as a single application which services several terminals concurrently, or as a single terminal with several independently developed application programs. TCAM is especially recommended in situations where several terminal types (start-stop terminals, binary synchronous terminals, display stations) are present on a system or where the same terminal is needed for several applications.

Chapter 2. System Units and Features

This chapter describes the machine units and operating features of System/7. Figure 2-1 shows the relationship between the input/output signals, the input/output modules, the internal interface, the processor module, and input/output devices interconnected to the processor module.

PROCESSOR MODULES

The processor module is available in two basic models, AXX and BXX. The letter A indicates that the model can operate as a stand-alone system or employ the asynchronous communication control feature; the letter B indicates that the 1130 Computing System attachment feature is included with the module. The next two characters (XX) specify the storage size in increments of 2k (2,048 words) beginning with 02, which represents 2,048 words of storage, and ending with 16, which represents 16,384 words of storage. For example, B02 specifies a processor module with the 1130 attachment feature and 2,048 words of storage. B08, as a second example, indicates a storage capacity of 8,192 words.

Both the AXX and the BXX have the following characteristics:

High-Speed Processing. The storage cycle time for the processor module is 400 nanoseconds.

Sixteen Bit Storage Word. Each word of storage consists of sixteen data bits with a parity bit for each eight bits.

High-Speed Priority Interruption System. The processor module has four levels of priority interruption, each level with 16 sublevels (total of 64). Switching from one level of interruption to the next is performed by hardware. The hardware switching time for a new interruption is two processor cycles (800 nanoseconds). The hardware switching time for return to a lower level which is "in process" is one processor cycle (400 nanoseconds).

Duplicate Registers for Each Level of Interruption. The processor module contains seven index registers, an instruction address register, logical registers, and an accumulator for each level of interruption.

Extremely Rapid Switching for Interruptions. The duplication of the registers and accumulator for each interruption level means that the program does not have to save the contents of the registers from the previous level of processing. Because there is no housekeeping overhead, the interruption routine can immediately begin processing to service the I/O module that caused the interruption.

Dynamic Assignment of Interruption Level and Sublevel. In System/7, the programmer assigns each individual device (with an interruption capability) to the priority interruption level and sublevel that best suits the application requirements.

Class Interruption System. A class interruption system causes entry into error recovery routines required by abnormal system conditions. Class interruptions cannot be disabled (masked off so that they do not interrupt the system), but class interruptions automatically disable all four priority interruption levels. The only register whose contents are automatically saved by a class interruption is the instruction address register in use at the time of the interruption. The error recovery procedure must save the contents of any other registers that must be retained.

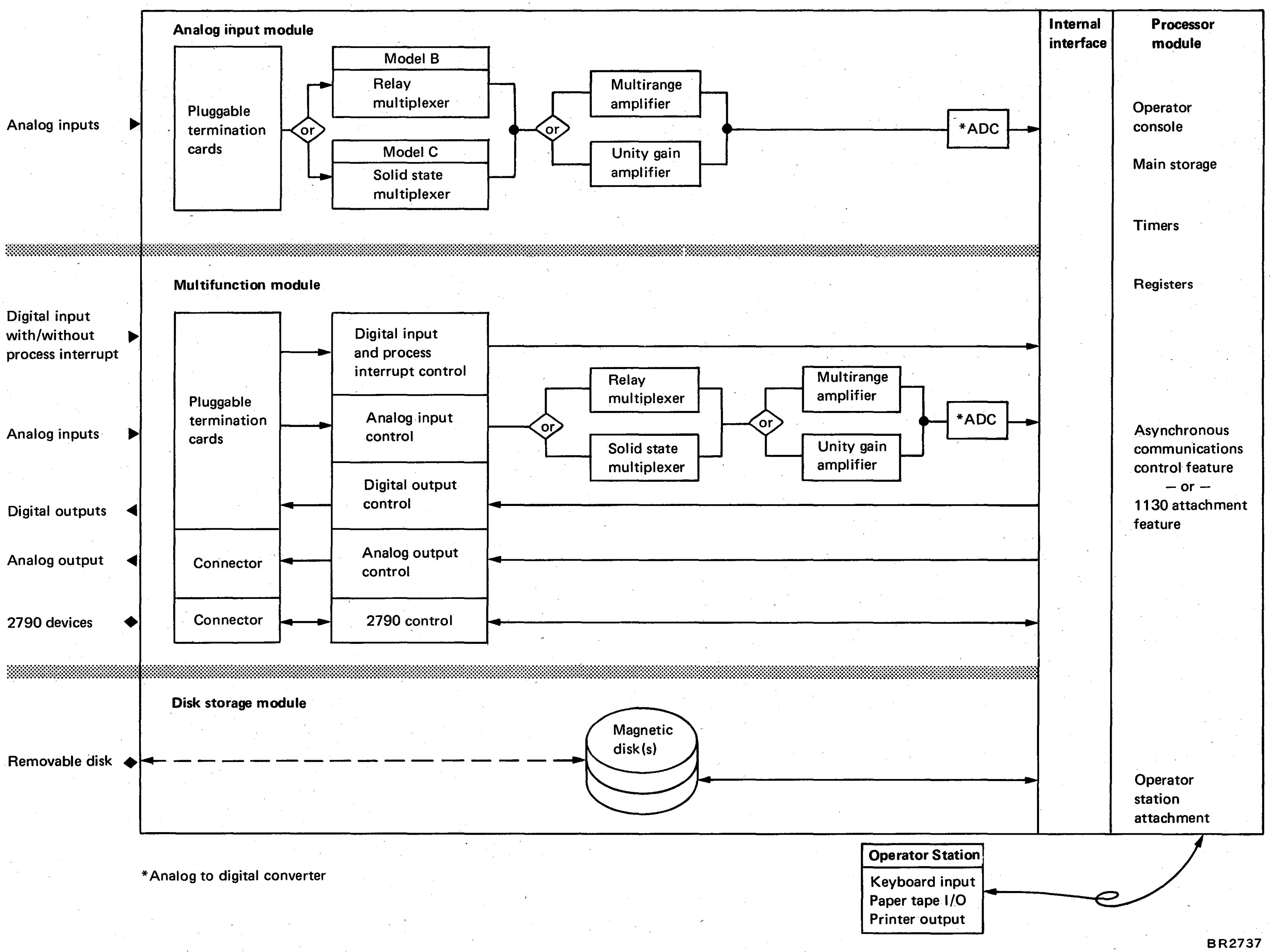


Figure 2-1. Units and features

Interval Timers. The processor module contains two interval timers. These timers are separately controlled 16-bit (one word) binary counters that count at 50-microsecond intervals. Under program control, the timers can be set to any value, started, read, and stopped. Once a value has been set into the timer and the timer has been started, the value in the timer at any instant can be read without stopping it. The timer causes an interruption request to the processor after it has counted the specified number of 50-microsecond intervals.

A Versatile Instruction Set. The following instructions can be executed by the processor module.

- Arithmetic instructions (storage to accumulator and register-to-register)

  Add
  Subtract
  Add immediate
  Complement register
  Add register
  Subtract register

- Logical instructions (storage-to-accumulator or storage-to-register and register-to-register)

  Logical AND
  Logical OR
  Logical exclusive OR
  AND register
  OR register
  Exclusive OR register

- Shift instructions

  Shift left logical
  Shift right logical
  Shift right arithmetic
  Shift left circular

- Branch and skip instructions

  Add to storage and skip
  Skip on condition
  Branch and link
  Branch on condition
  Branch
  Branch and link long

- Register-to-register instructions

  Load from register
  Store to register
  Sense level and mask
  AND to mask
  OR to mask
  Interchange register

- Input/output instruction

  Execute input/output

- State control instruction

  Level exit

Asynchronous Communications Control Feature

This feature, available on the AXX processor modules, is required when System/7 operates in a distributed sensor-based application having any one of the following host configurations:

- System/7 to System/360 Model 25
  (via System/360 Model 25 integrated communications adapter)
- System/7 to System/360 Model 30 or larger, or System/370
  (via IBM 2701, 2702, 2703)
- System/7 to System/370 Model 135
  (via System/370 Model 135 integrated communications adapter)
- System/7 to IBM 1800
  (via IBM 1800 RPQ C08763)

The asynchronous communication control transfers programs and data between the host computer and System/7, under control of System/7 and host programs. It can also transmit an initial program load (IPL) to System/7, under control of the host computer.

Four line-adapter features are available for selecting different communication modes.

1. Two-wire common-carrier facilities in point-to-point communication mode.
2. Two-wire limited-distance facilities operating in either point-to-point or multi-point mode.
3. Four-wire leased-line facilities operating in either point-to-point or multi-point mode.
4. Two-wire leased line facilities operating in point-to-point mode.

Transmission speed is 134.5 or 600 bits per second.

IBM 1130 Computing System Attachment Feature

Processor modules containing this feature are designated by BXX model numbers. Occupying the space in the BXX models that is reserved for the asynchronous communication control feature in the AXX models, this feature connects the System/7 to the storage access channel (SAC) in the IBM 1130 Computing System. It makes possible the initial program load (IPL) of System/7 under control of the 1130.

Between the 1130 and System/7, data transfers directly, storage-to-storage, under control of the 1130. Data transfers as fast as the 1130 can take storage cycles. The processor module interrupts the 1130 to request data or programs. The 1130 interrupts the processor module to indicate that the 1130 must transfer data to the processor module or obtain data from the processor module. It also interrupts the processor module when any data transfer is completed.

INPUT/OUTPUT MODULES

System/7 is designed to collect sensor-based data and directly control complex operations. To serve a wide variety of applications, an important objective in its design, System/7 employs two basic types of input/output (I/O) modules:

1. sensor-based modules
2. disk storage module

For sensor-based applications on System/7, the I/O module is the basic building block. Each module is self-contained; that is, it houses all the electrical/mechanical components necessary to serve the I/O operations—analog input, analog output, digital input, digital output, 2790 I/O control, and magnetic disk storage—and to connect with the System/7 internal interface (see Figure 2-1). In addition, an I/O module may be physically placed in any module position in the enclosures (except that allocated to the processor module and those positions—shown in Figure 1-8—not recommended for the disk storage module). The system can be expanded in convenient field-installable sections by adding more enclosures and I/O modules.

Sensor-based modules are the system's interface between the sensors that detect the data and the processor module that acts on the data. Through them, the processor module directs the issuing of control signals. They also serve as the man/machine interface that is required by many plant or laboratory automation systems. Two types of sensor-based modules are available:

1. analog input module
2. multifunction module

Besides large on-line storage capacity (and virtually unlimited offline storage with the removable, interchangeable, disk option), the disk storage module gives the system high-speed data access time, and the ability to load programs into the processor module storage for execution.

Analog Input Module

Two models of the analog input module are available, Model B01 and Model C01. Both models offer the following:

Fourteen-bit, Analog-to-Digital Converter. This unit converts analog data received from the process into a 14-bit binary number plus a sign bit.

Multi-range Programmable or Auto-ranging Amplifier. This amplifier has 7 full-scale ranges: ±10 mV, ±20 mV, ±40 mV, ±80 mV, ±160 mV, ±640 mV, and ±5.12V. Any one of these ranges can be selected by the program, or the analog input module can select the best amplifier range for the voltage being sensed. An optional feature is an amplifier with only the ±5.12V range.

The accuracy of the various ranges varies from 0.35% of full-scale value on the lowest range to 0.05% of full-scale value on the highest range. Repeatability ranges from 0.20% of full-scale value on the lowest range to 0.03% of full-scale value on the highest range.

One Hundred Twenty-eight Input Points. The analog input module accommodates up to 128 two-wire differential analog input points. These points are available in groups of 16.

External Synchronization Control. By means of this control, attached devices signal System/7 when data to be sensed is available. The processor module directs the analog input module to request data from the process sensor. When the sensor indicates that the data is available, the processor module directs the analog input module to perform the analog-to-digital conversion and transmit the data to the processor module.

Plug-in Customer Connections. Customer connections to the analog input modules are made with plug-in cards at the rear of the module.

Temperature Reference Attachment. This attachment converts to digital values the analog data presented by a cold junction thermocouple temperature compensation unit.

Model B01 employs a mercury-wetted contact relay multiplexer capable of up to 200 samples per second. This model operates with common mode potentials of up to 250V with a common mode rejection ratio in excess of 120 decibels (db). Three termination cards are available for use with this multiplexer: an analog signal filter card, a filter card with a cold junction thermocouple temperature compensation unit, and a custom card the customer uses to design his own input filter network. Customer connection to these termination cards, which plug into the back of the module, is by two screw connections for each input point.

The model C01 has a solid-state differential multiplexer. The multiplexer is capable of 7,000 samples per second when the amplifier is programmed for automatic selection of the range; 14,000 samples per second when the program specifies an amplifier range less than 5.12V; and 20,000 samples per second when the program specifies the 5.12V full-scale amplifier range. For this multiplexer, the highest voltage for common mode plus normal mode is 10. The common mode rejection ratio ranges from 80 db to 114 db, depending on the resistive source unbalance and the range. Five termination cards are available for use with this multiplexer: a connector card, a polarized filter card, a non-polarized filter card, a nonpolarized filter card with a cold junction thermocouple temperature compensation unit, and a custom card the customer uses to design his own input filter network.

Multifunction Module Model A01

This I/O module incorporates all of the sensor-based input/output capabilities in a single module. With an A02 processor module, it gives the smallest System/7 the capability of performing any of the sensor-based functions that can be performed by larger configurations of the system.

In one package, the multifunction module combines analog input, analog output, digital input, digital output, and 2790 data collection facilities. It is designed to accommodate combinations of the input/output functions.

Analog Input

Although the analog input component of the multifunction module offers the same speed, accuracy, and common mode options as are offered by the analog input module, it has these differences:

1. Either the mercury-wetted contact relay multiplexer or the solid state multiplexer may be chosen. The two multiplexers are mutually exclusive.
2. Up to 32 points of analog input are available in 8 groups of 4 points each.

Analog Output

This feature of the multifunction module presents a control signal to an external device in the form of a program-variable voltage. It has the following characteristics.

1. Isolated output points. No more than 2 isolated output points can be installed in each multifunction module.
2. High accuracy output range. The unipolar analog output voltage ranges from 0 volts to 10.23 volts with 10-bit resolution.
3. Read-back before read-out. The binary register that determines the analog output voltage can be read by the processor module before the actual output operation is performed. This permits the processor module to check that the correct value is in the register before the corresponding analog output is produced.

Digital Input

Through the digital input feature, the multifunction module can accept a binary 1 or 0 condition as an input. Typically such inputs indicate the opening or closing of a switch or a binary voltage state from attached devices. Stored as binary ones or zeros in input registers, such inputs are read into the processor module storage on demand. Characteristics of the digital input feature are:

1. Up to 128 digital input points per multifunction module are available in groups of 16 points.
2. Voltage range of −52.8 volts to +52.8 volts.
3. Direct read of one 16-bit group within one execution of input/output instruction time.
4. Process interrupt. The process interrupt feature is available on one or both of the first two digital input groups. With this feature, a 16-point group is compared with a 16-bit reference register (which can be set to any value by the program). Interruption requests are then initiated on the basis of either an equal or unequal comparison with the contents of the reference register. The choice of comparisons is under program control and can be changed at any time.
5. A variety of input connections. The terminations available for the digital input feature are for contact sensing, voltage sensing, or customer designed input systems. Connection to the external process is by screw terminals on plug-in termination cards.
6. Differential input voltage sensing with each point isolated.

Digital Output

The multifunction module provides the following digital output capabilities.

1. Up to 64 digital output points are available in 4 groups of 16 points each.
2. Contact output group, which has the capability of making or breaking circuits up to 2A and 125V (100 VA maximum). The mercury-wetted contact relay isolated outputs operate up to 250 Hz.
3. Medium power group. This type has an isolated electronic digital output capable of switching 52.8 volts at 450 mA.
4. Low power group. This output option supplies digital outputs of 6V at 4 mA.
5. Direct write of one 16-bit group within one execute input/output instruction time.
6. Read-back before read-out. Data can be transferred to a holding register in the multifunction module and read back before activating the output point.

2790 Control

Figures 2-2 through 2-9 show the units associated with the 2790 Control. The 2790 controls data collection, operator guidance, data output, and other communication between man and machine. One 2790 Control can be attached to a System/7. This feature has the following capabilities.

1. Attachment of 2791 and 2793 Area Stations. These stations transfer data by means of a 2-wire high-speed communication line. They read badges and cards and accept data from keyboards. Visual displays on the area stations and printers attached to them record output information. The 2791 Area Station has a 12-key manual entry keyboard, an operator panel with pushbutton switches and program-controlled display panel, an 80-column card reader, and a 10-column badge reader. Additionally, the 2791 Model 1 has attachments for three 1035 Badge Readers, a 1053 Printer, and up to 16 data entry units. Although the 2793 Area Station has the same attachment facilities as the 2791 Area Station, it does not have the keyboard or operator panel.
2. Attachment of IBM 2795, 2796, 2797 Data Entry Units and 2798 Guidance Display Unit. Attached to the 2790 Control through the area stations, these units read cards and badges and data from manual switches.

The 2795 Data Entry Unit has a 10-column badge-card reader and two 10-position rotary switches. The 2796 Data Entry Unit has all the 2795 Data Entry Unit features plus 2 rotary switches and 2 rocker-thumbwheel switches. The 2797 Data Entry Unit has all of the 2795 features plus a 10-key manual keyboard that accepts up to 6 digits.

A data entry and output unit, the 2798 Guidance Display Unit handles multistep, interactive transactions via the System/7. For input, it has a 56-character alphameric keyboard and eight control keys, with a 16-position display the operator verifies before he transmits any 16-character combination of the 56-characters entered from the keyboard. For output, the same 16-position display and a guidance panel directs the operator step-by-step through transactions or gives him status information under program control. The operator's instructions are recorded on two removable overlays written by the user to suit his specific applications. Up to 16 customer-defined instructions can be actuated at a time, and through the use of 3 manual settings on each of 2 panels, a total of 48 lines of information are displayed.

3. Attachment of IBM 1053 Printer Model 1 and 1035 Badge Reader. These units attach to the 2790 Control through the area stations. The 1053 is an output printer (it has no keyboard for data entry) that prints a line up to 13 inches long at 14.8 characters per second. An input unit, the 1035, transmits numeric data from 10-column badges.

The number of area stations and data entry units that can be attached to the system depends upon the application. Up to 16 area stations, with up to 16 data entry units attached to each area station, can be accommodated by the 2790 control feature. Up to 12 2798 Guidance Display Units can be attached to a 2791 Area Station and/or 2798s per 2793, or a total of 128 2798s per System/7. Each 2795, 2796, 2797, and 2798 is attached to a 2791 Model 1 or 2793 Area Station by a 4-wire cable up to 1,000 wire-feet long and supplied by the user.

5022 Disk Storage Module

This large capacity, on-line storage module is available in four models (model 1, model 2, model 3, and model 4). Containing 2 magnetic disks each, models 1 and 2 differ only in data access speed. The upper disk (in a disk cartridge) is removable, but the lower disk is fixed. Models 3 and 4, which each contain a single fixed disk, differ only in data access speed.

Large On-line Storage Capacity. Models 1 and 2 store 2,457,600 words of data online. Models 3 and 4 store 1,228,800 words of data online. Each word contains 16 binary digits (bits).

Virtually Unlimited Off-line Storage. Each removable disk can store 1,228,800 sixteen-bit words. The removable disks on models 1 and 2 are fully interchangeable so that any number of disks necessary to store your data can be available.

Data Storage and Access Time. Each disk contains 200 cylinders and stores data in 128 word sectors. Four additional cylinders are available for assignment as alternate cylinders should irrecoverable failures be found on any of the 200 primary cylinders. For all four models, the following chart shows:

1. The minimum seek time (adjacent cylinders)
2. The maximum seek time (204 cylinders)
3. The average time to access the data stored in any sector (when repositioning the read/write element is required)

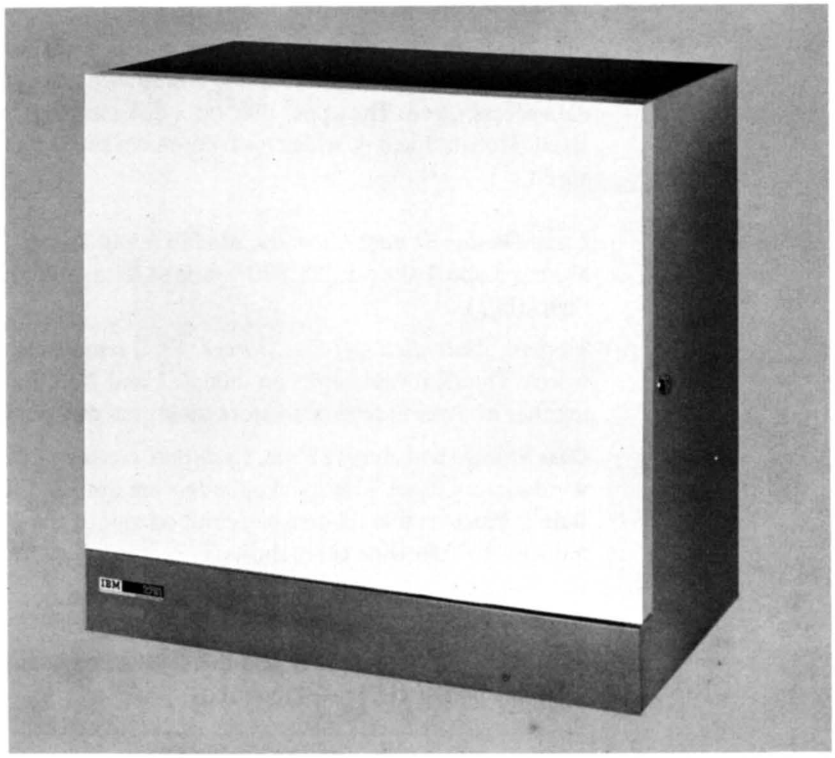
Disk module	Cylinder seek time		Average data access *
	Minimum	Maximum	
Model 1	32 ms	727 ms	275 ms
Model 2	28 ms	255 ms	135 ms
Model 3	32 ms	727 ms	275 ms
Model 4	28 ms	255 ms	135 ms

* Includes average rotational delay of 20 milliseconds

BR2475

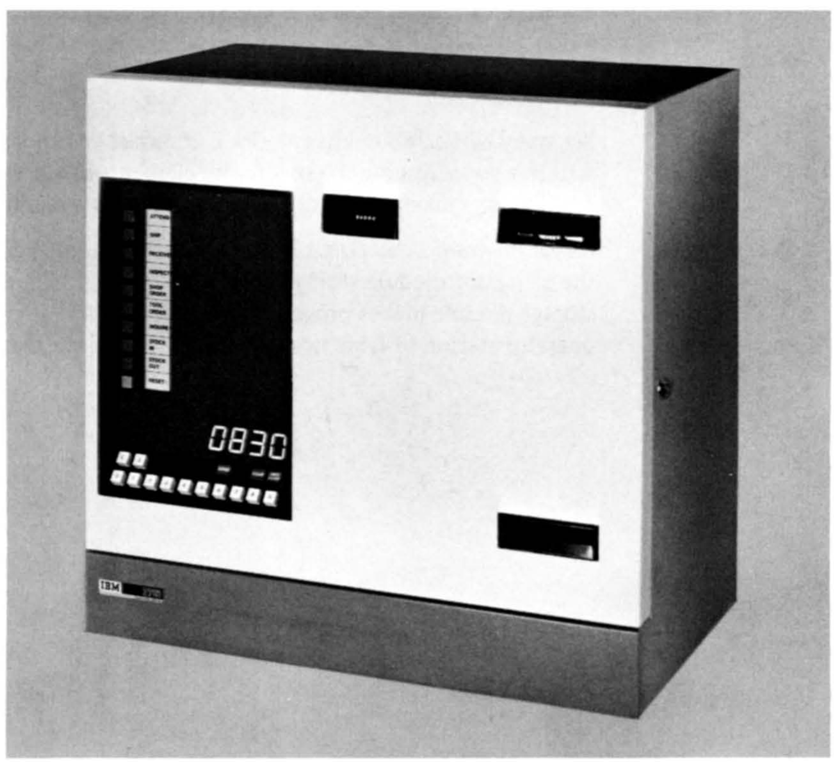
As many as 12,288 words (model 1 or model 2) can be transferred to or from the disk without repositioning the read/write elements of the module. Data transfers to and from the disk at a maximum rate of 99,500 words per second.

Initial Program Load (IPL). The disk storage module can be used to load programs into the processor module storage for execution. The high data transfer rate of the disk storage module makes program loading faster than it can be done either through the operator station or from host processor storage via the communication link.



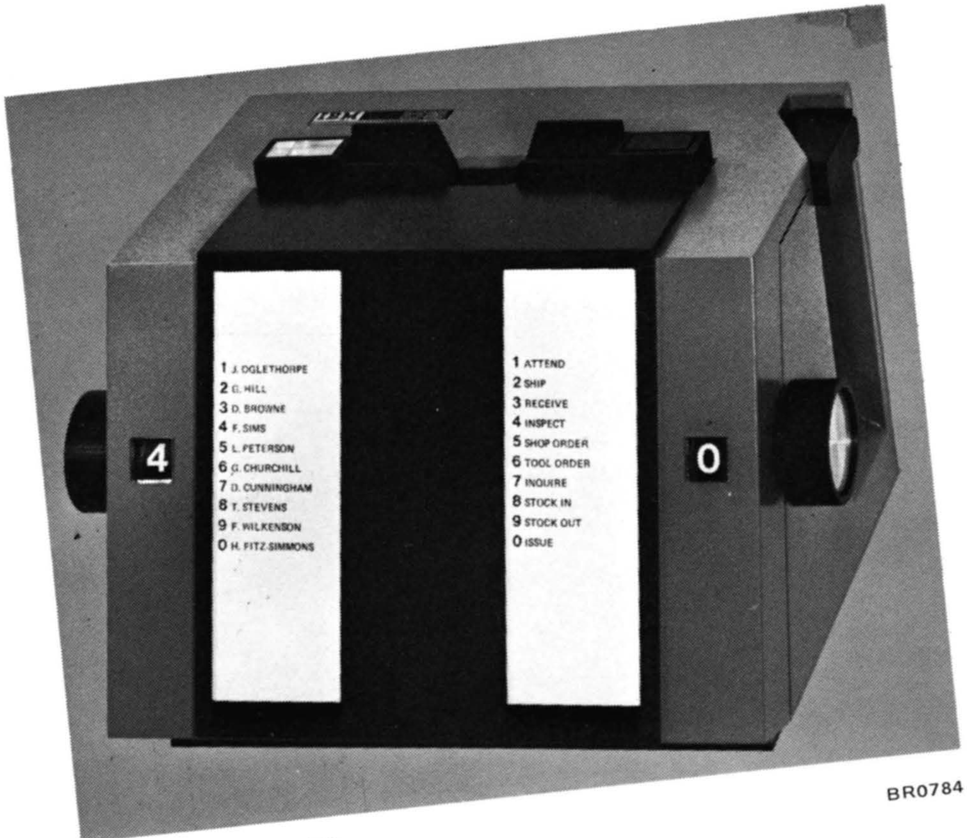
BR0783

Figure 2-2. IBM 2791 Area Station



BR2655

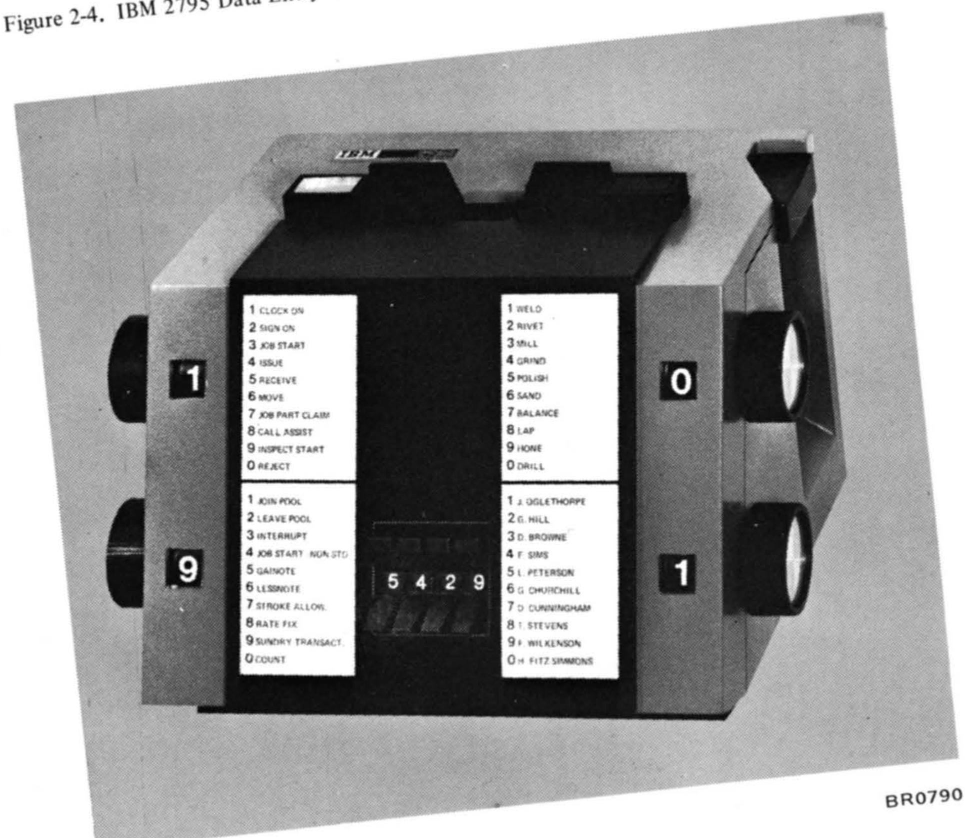
Figure 2-3. IBM 2793 Area Station



BR0784

IBM 2795 Data Entry Unit

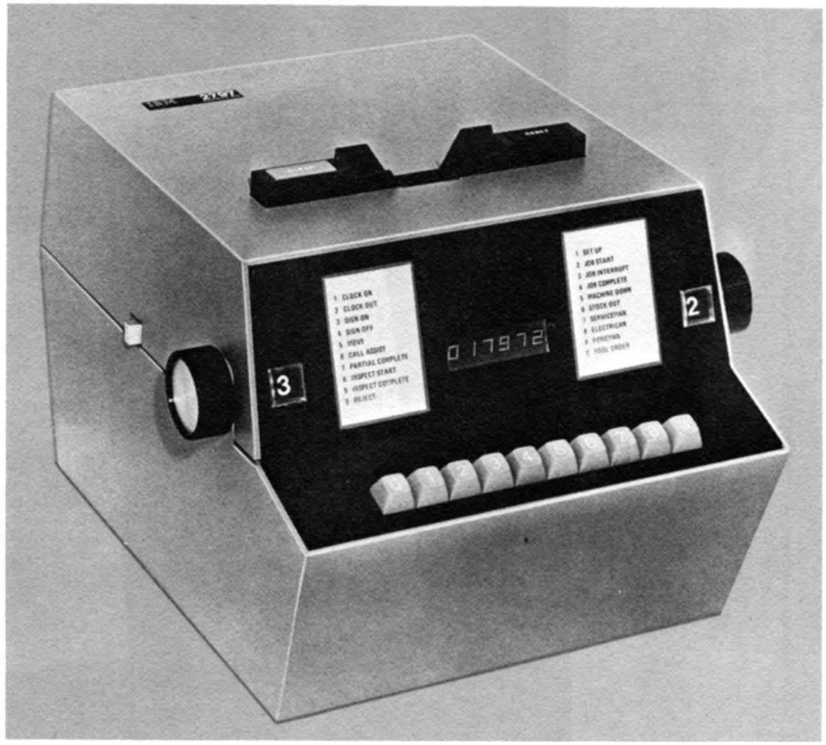
Figure 2-4. IBM 2795 Data Entry Unit



BR0790

IBM 2796 Data Entry Unit

Figure 2-5. IBM 2796 Data Entry Unit

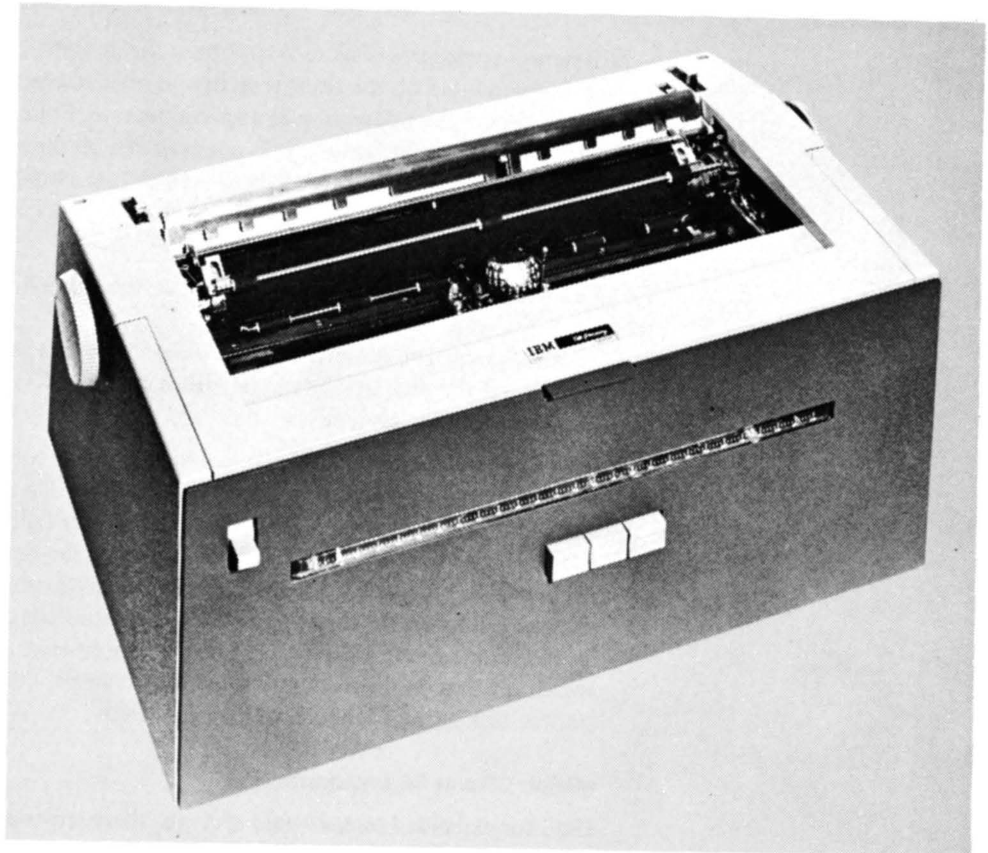


BR2738

Figure 2-6. IBM 2797 Data Entry Unit

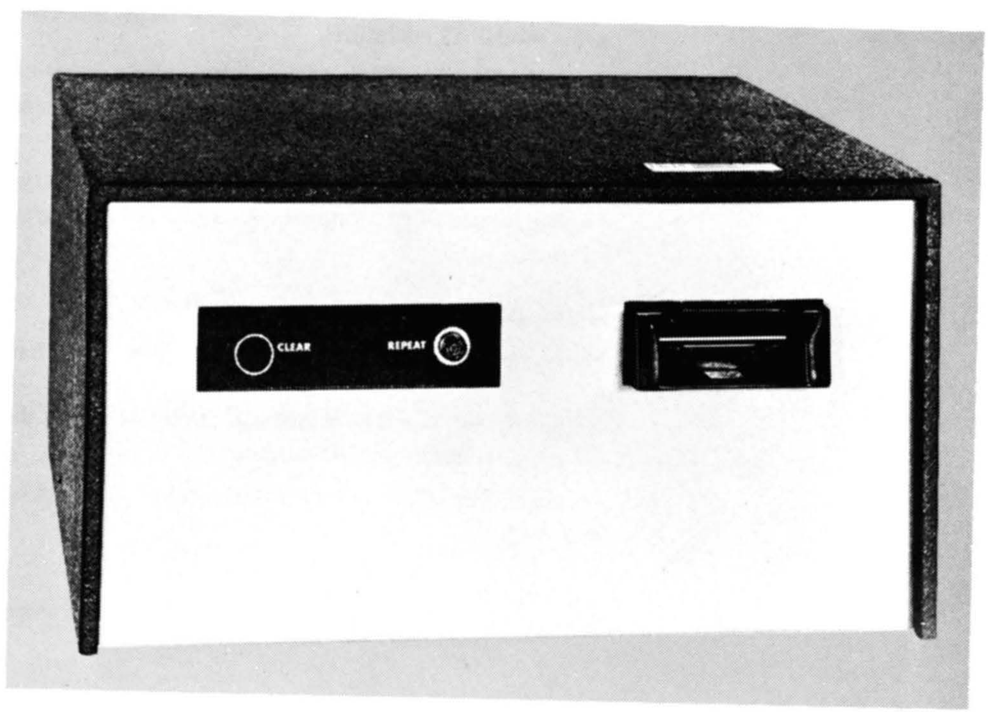
BR2739

Figure 2-7. IBM 2798 Guidance Display Unit



BR2740

Figure 2-8. IBM 1053 Printer



BR2741

Figure 2-9. IBM 1035 Badge Reader

ENCLOSURES

System/7 enclosures (Figure 2-10) house the modules and provide power and signal distribution; mounted on the front is an operator's console. All five models of the enclosure offer power failure detection as a special feature. Four models are equipped with an internal air isolation feature. It is not available on the model A2 enclosure. Figure 2-11 shows combinations of module positions and attachments for each enclosure model.

Model A2 Enclosure

This model, the smallest of the enclosures, has the following attributes:

Module Space. The A2 enclosure accommodates one processor module and one I/O module. (If the disk storage module is used as the I/O module, no sensor-based I/O can be attached to the system.)

Power Failure Detection. This feature signals the processor module when input voltage to the enclosure falls below a safe operating level. The processor module then has a minimum of 8 milliseconds to institute any power failure procedures before the system becomes inoperative. Controlled by a switch on the operator's console, an auto-restart routine of the power failure detection feature automatically turns on power to the system and resets it when the input power to the enclosure is restored. An IPL is then performed by the selected unit. Only one IPL source can be selected at a time. Selection is made by switch settings on the System/7 operator's console. Possible IPL sources are the operator station, disk storage module, or host processor.

Models C3 and C6 Enclosures

Used for expanded sensor-based systems, these larger enclosures have the following attributes:

Module Space. The C3 enclosure accommodates a processor module and any two I/O modules. The C6 has space for a processor module and any five I/O modules.*

Power Failure Detection. This optional feature is the same as the power failure detection in the model A2 enclosure.

Expandability. The C3 and C6 enclosures have (as an optional feature) the capacity to attach another enclosure (a model D3 or D6) containing I/O modules up to 200 feet from the C3 or C6.

Internal Air Isolation. This optional feature protects systems that are to be operated in a severe industrial environment. Air for cooling the system is filtered and circulated internally.

Models D3 and D6 Enclosures

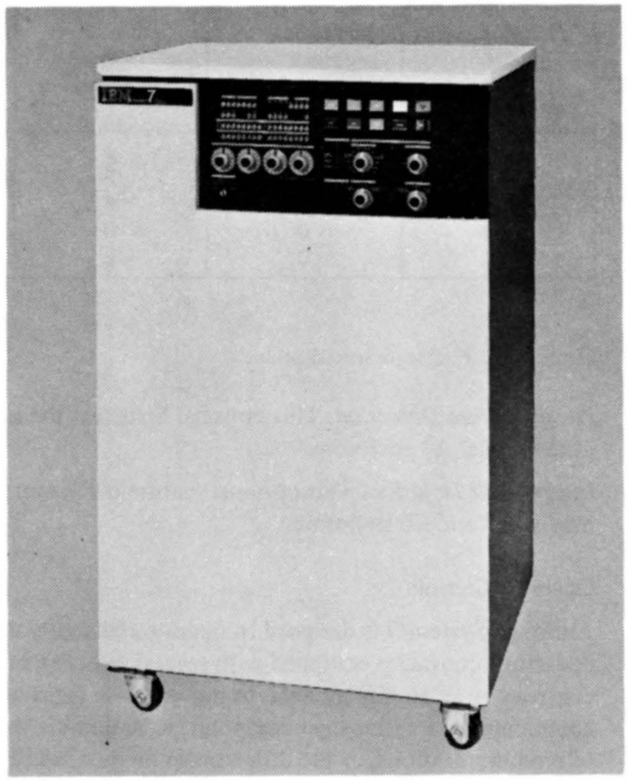
These enclosures are used to expand a system. They have the following specifics:

Module Space. The D3 enclosure accommodates any three I/O modules. The D6 accommodates any six I/O modules.*

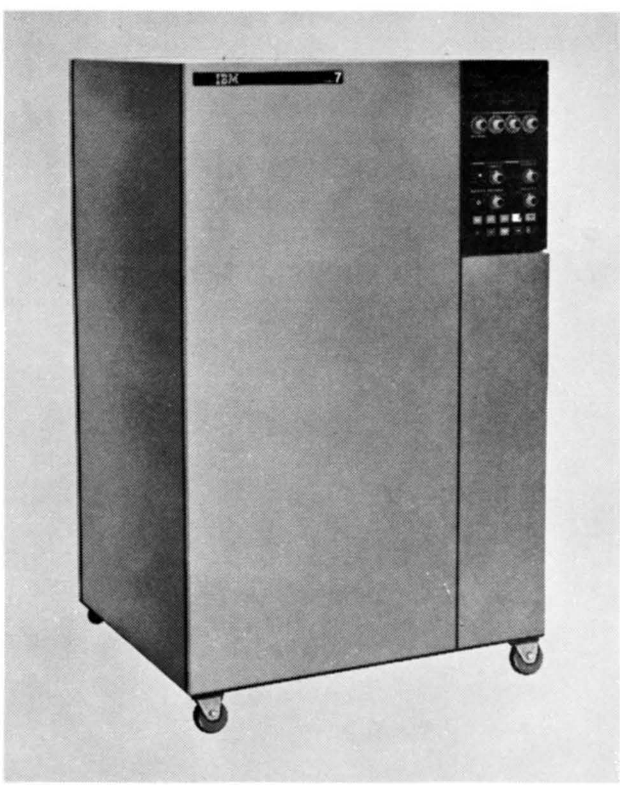
Location. The D3 and D6 enclosures can be attached to C3 or C6 enclosures up to 200 feet away.

---

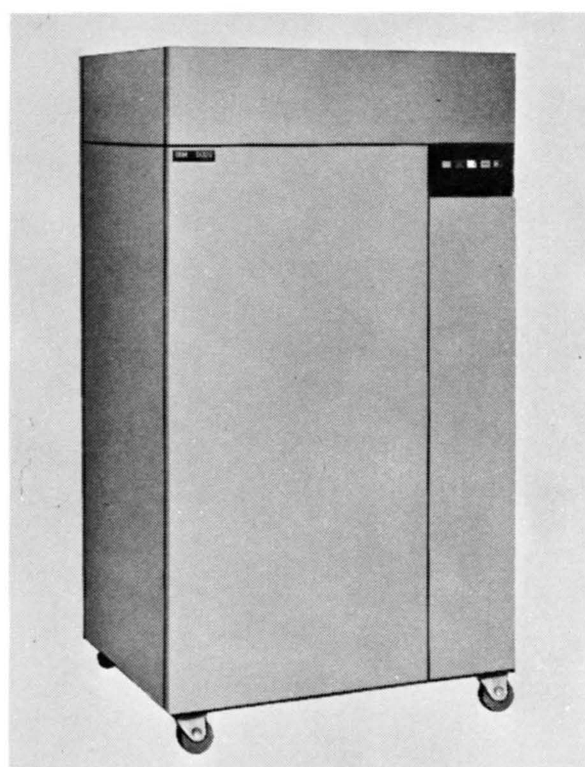
*Except in the positions not recommended for the disk storage module, as shown in Figure 1-8.



Model A2 Enclosure BR0785A



Model C3 Enclosure BR0786A



Model D3 Enclosure BR0787A

Figure 2-10. System/7 Enclosures

Enclosure model	Number of processor modules accommodated	Number of I/O modules accommodated	Attachable to C3/C6	Operator station attached
A2	1	1	No	Yes
C3	1	2	No	Yes
C6	1	5	No	Yes
D3	0	3	Yes	No
D6	0	6	Yes	No

BR2645

Figure 2-11. Enclosure specifications

Power Failure Detection. This optional feature is the same as the power failure detection in the model A2 enclosure.

Internal Air Isolation. This optional feature is the same as the internal air isolation in the models C3 and C6 enclosure.

Operator Console

Although System/7 is designed to operate essentially without operator intervention, its operator's console is equipped with several switches and indicators. These are used to control power, to display data, to move it into registers and main storage, to control communication with a host computer, to determine the source of IPL from paper tape, telecommunications, or the disk storage module, and to diagnose problems. A key-operated, enable-disable switch prevents unauthorized use. On the D3 and D6 enclosures, the operator's console has only power controls and indicators.

IBM 5028 OPERATOR STATION

The IBM 5028 Operator Station (Figure 2-12) is the input/output unit for the system operator. He uses it for both program preparation and initial program load (IPL).

Keyboard Entry to the System. Data can be entered directly into the system through the keyboard on the operator station at a rate of 10 characters per second. The data code used by the operator station is American National Standard Code for Information Interchange (ASCII).

Printer Output From the System. Messages from the system to the operator print out through the operator station printer at 72 characters per line, 6 lines per inch, on 8-1/2-inch-wide paper rolls. Paper is fed by a friction feed mechanism. To prevent overprinting of characters and damage to the platen, the printer automatically carriage returns and line feeds when the carriage reaches the end of the printed line. Single-part paper is used for printing.

Paper Tape Reader Input. Data enters the system through a paper tape reader at a rate of 10 characters per second. The paper or lubricated, nonmetallic plastic tape is 8-track, one inch wide. All 256 binary bit combinations of 8 bits can be entered into the system from the paper tape reader. Initial program loading from the operator station can be performed through the paper tape reader.

Paper Tape Punch Output. Output data from the system can be produced as punched paper or lubricated, nonmetallic plastic tape. All 256 binary bit combinations can be punched in the 8-track, one-inch-wide tape, at a rate of ten characters per second.

Local/Remote Operation Control. Controls for these operations are located on the keyboard and are labeled LOCAL and REMOTE. By setting the switch to LOCAL, the operator in effect disconnects the station from the system so that he can use the station to prepare program tapes (in assembler language) or to prepare data for entry into the system. Thus, he can punch data into tape, insert the tape in the paper tape reader, and go on to other operations he must perform. When the system requires the data punched into the tape, it can perform a paper tape read operation to obtain the data.

With the switch at LOCAL, information entered at the keyboard is printed by the printer and, optionally, punched by the paper tape punch. Data read from the paper tape reader is printed and, optionally, punched by the paper tape punch. With the switch at REMOTE, the processor controls all operator station operations.

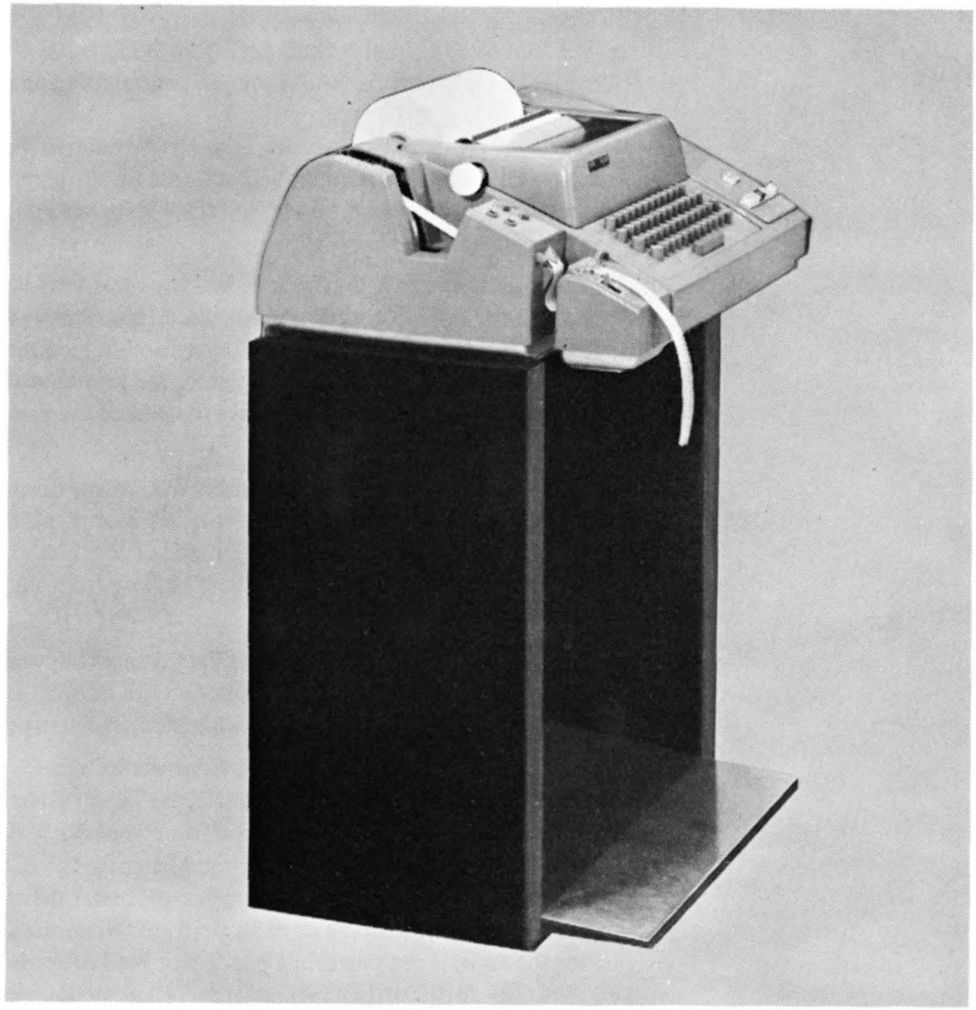


Figure 2-12. IBM 5028 Operator Station

Chapter 3. Program Preparation

System/7 programs can be prepared either on a host preparation facility or on System/7. This chapter describes both methods, and Chapter 4 describes the programming facilities and functions of MSP/7.

HOST PREPARATION FACILITIES

Modular System Programs for System/7 are written in assembler language macro instructions specifically defined for the System/7 and added to the macro library of the supporting, or host, system. Host preparation facilities provide a convenient means of coding programs in a high-level language and assembling them on a computing system with more power than a System/7. To call a sequence of instructions or a function to accomplish a desired result, the programmer writes macros and instructions. Then an MSP/7 output handler translates the macro assembler output of the host system into a System/7 storage load module. This storage load module can be entered into System/7 over a teleprocessing link or channel link, or it may be punched on 8-track paper tape for the operator station's paper-tape reader.

In addition to the macros furnished by IBM, other macros designed to meet specific needs may be defined and created by the user and eventually catalogued on the macro library of the host computer. Using macros to write source programs simplifies the programmer's job and standardizes the sequences of instructions applied to accomplish a desired result. For example, the logic of a program may require the same instruction sequence to be executed again and again. Rather than code this entire sequence each time it is needed, the programmer writes a macro defined for that sequence.

A macro assembly produces the routines required to operate the System/7. When coding, the programmer specifies *which* routines to include in the assembly by selecting options and variable parameters.

Macro Support Requirements

The requirements for preparing programs on a host computer are as follows:

- A System/360 or System/370 with DOS, and a macro assembler with the MSP/7 macro extension.
- A System/360 or System/370 with OS, and a macro assembler with the MSP/7 macro extension.
- An 1130 Computing System with the 1130 Disk Monitor System, Version 2 macro assembler and the MSP/7 macro extension, and a minimum of 8,192 words of storage.
- An 1800 Data Acquisition and Control System with the Multiprogramming Executive Operating System, macro assembler and the MSP/7 macro extension, and a minimum of 16,384 words of storage.

The output of the macro assembler is reformatted by the MSP/7 output handler program into a System/7 storage load module, which can be entered into the System/7 by:

- A teleprocessing link, if the System/7 is attached to a System/360, a System/370, or an 1800 system.
- A channel link, if the System/7 is attached to an 1130.
- Paper tape directly into the System/7.

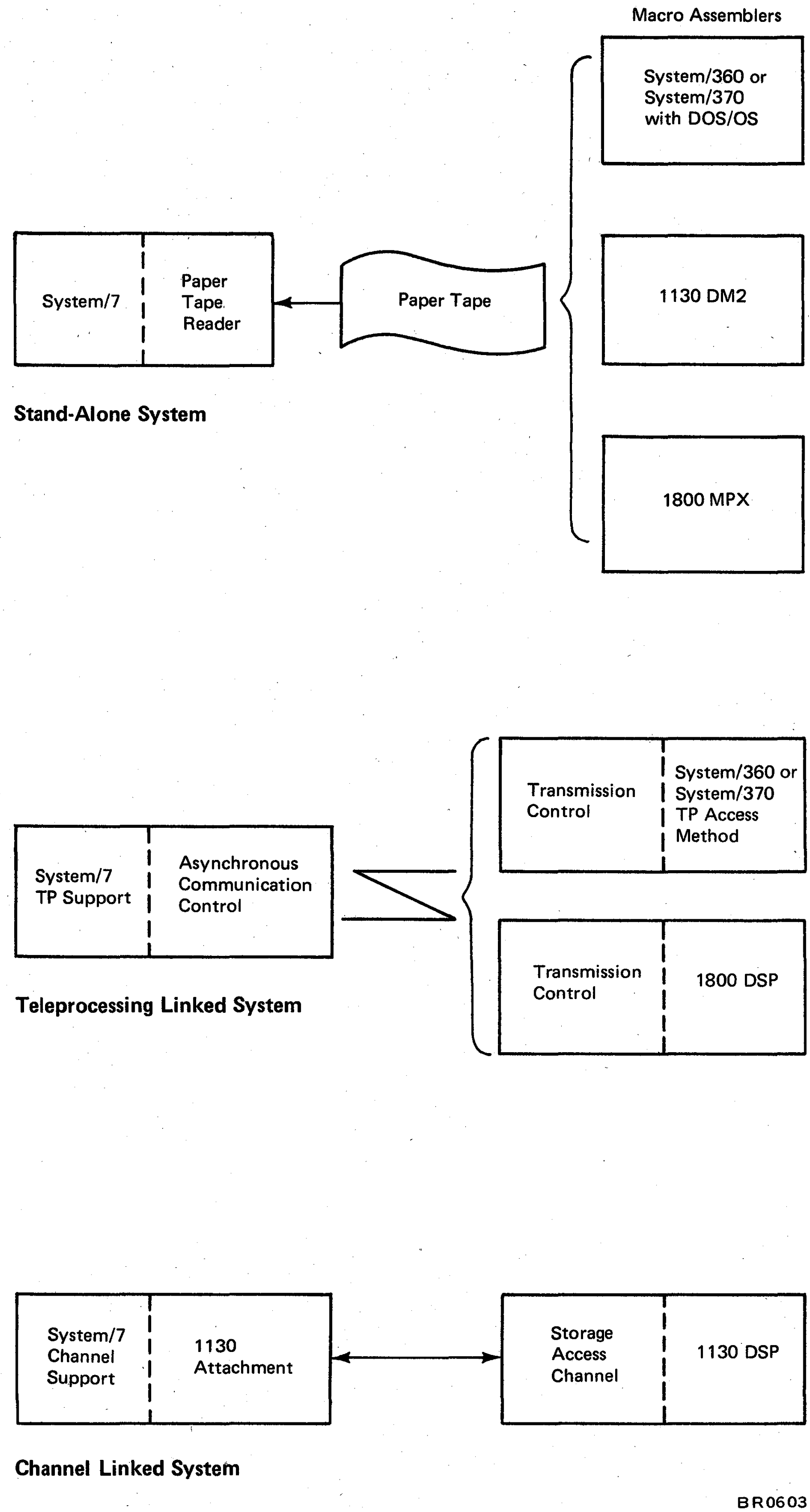


Figure 3-1. Loading the System/7 object program via host computer

When assembling System/7 programs, the programmer must observe the standard programming conventions for each host assembler. Figure 3-1 shows the three methods of loading a System/7 program that was assembled on a host computer.

Output Handler

The output handler routine (see Figure 1-12) translates the host assembler output into an MSP/7 storage load module. The routine is stored in the core image library (or equivalent) of the host processor and is executed as another job step or procedure at the conclusion of the macro assembly. The output of the handler is an MSP/7 storage load in the form of a paper tape, a card deck, or a disk file, depending on the host system being used. The output handler also prints an MSP/7 formatted program listing.

The IBM 1130 or 1800 systems can reformat a source program prepared in 1130 or 1800 format and translate and punch it into cards for the System/360 or System/370 assembler. This makes it possible for the 1130 user to conveniently transfer his program preparation to a System/360 or System/370.

Modular System Programs Macro Instructions

MSP/7 macro instructions expedite the preparation of System/7 programs. Specific MSP/7 macros support the functions peculiar to a sensor-based environment. To illustrate the inherent flexibility of the MSP/7 host preparation facilities, the following examples are cited:

By calling just one macro, the programmer can have a significant portion of his program prepared. Thus, this single macro could build a program of any number of individual instructions with basic routines to operate the System/7. After adding some application coding, he has a program to run the System/7 in a specific sensor-based environment.

Another example of the flexibility of MSP/7 macros is their usage in the preparation of a routine to service an I/O interrupt. Although the programmer has several ways to write the interrupt service routine, the simplest is to use an access macro that performs a data transfer on an input. The access macro provides all the coding required to read in data from a device, and transfer data from the device to storage.

The program routines serve as a base on which the programmer can add other macros to expand and modify certain MSP/7-provided functions. These built-in options make it possible for the programmer to readily customize the program to suit the requirements of his installation. This customizing saves storage space and ensures that only the routines needed to support the particular installation are included in the System/7 storage load module.

At times macro operands control the amount of coding generated by the macro statement, as illustrated in Figure 3-2.

MSP/7 provides a tailored operating system for each storage load module where:

- Each storage load can contain different system routines
- Each storage load can selectively use the input/output modules required by the application
- Each user can define unique macros for specific functions
- Macros can be permanently stored in the host computer or set up temporarily

Macro Organizations

Macros are organized into four categories:

- *Instruction macros (P-type)* supply one machine instruction each time they are used. Included with the *P-type* macros are the nonexecutable assembler instructions, and the extended instructions. (Refer to "Appendix A.")
- *System macros ($-type)* produce subroutines to operate the system, such as most of the control functions.
- *Access macros (@-type)* act as a simplified interface with the system macros.
- *Specification macros (#-type)* specify to the assembler the configuration and system requirements for the object program.

The MSP/7 macros and subroutines are described in "Appendix B."

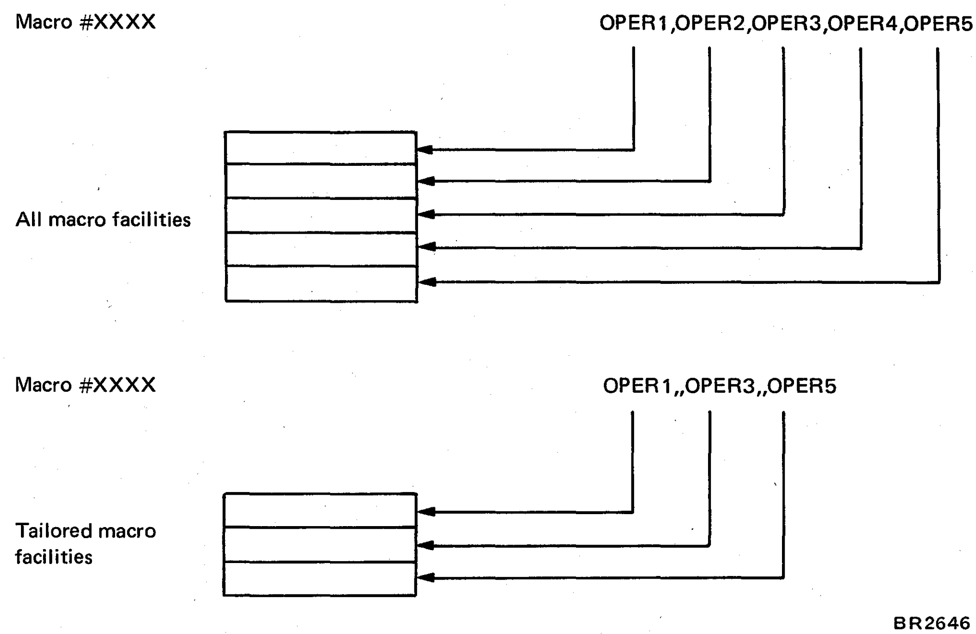


Figure 3-2. Macro instructions

SYSTEM/7 ASSEMBLER

This assembler prepares object programs entered from the 5028 Operator Station of the System/7. The assembler uses the paper-tape reader and the keyboard as input devices, and the paper-tape punch for the preparation of the object tape. The printer produces the assembly listing.

For program preparation, the System/7 must have a minimum of 4,096 words of storage.

The System/7 Assembler is supplied in the form of a self-loading paper tape, which contains:

- A bootstrap loader
- The assembler
- Multiply and divide subroutines
- Utilities (zero, dump, and patch)

The following introduction is for those readers who may not be familiar with basic assemblers.

Introduction to the Assembler

Programming in an assembler language offers a number of important advantages over programming in the actual language of the computer.

First, machine instructions can be referenced through mnemonic operation codes. The assembler translates these mnemonics into the desired machine codes and performs extensive error checking.

Second, addresses of data and instructions can be written in symbolic form. In practice, almost all addresses are so written. Furthermore, the use of symbolic addresses reduces the clerical aspects of programming, eliminates many programming errors, and makes the program easier to modify. If the symbols are chosen to be meaningful, the program also is easier to read and understand.

The advantages of using an assembler are:

Renaming Symbols. A symbol can be equated or substituted for another symbol so that both refer to the same item. This makes it possible for the same program or a storage area to be referred to by different names in different parts of the program.

Automatic Storage Assignment. The assembler assigns consecutive addresses to program elements as it encounters them. After processing each element, the assembler increments a counter by the number of words assigned to that element. This counter indicates the storage location available to the next instruction.

Convenient Data Representation. Constants can be specified as decimal digits, alphabetic characters, hexadecimal digits, and storage addresses.

Program Listing. For every assembly, the user can obtain a program listing.

Error Checking. The source program is examined by the assembler for errors arising from incorrect usage of the instructions. When an error is detected, a warning message appears in the program listing.

System/7 Assembler Program

The assembler program converts a source program into a machine language program. The conversion is one-for-one; that is, the assembler produces one machine language instruction for each source program input instruction.

The capabilities of the paper-tape assembler are as follows:

One-Pass Assembler. After the assembler is read into the System/7, the user's source program need be read only once. Forward references are resolved by overlay patches located at the end of the object paper tape.

Mnemonic Operation Codes. The assembler accepts all the System/7 machine instructions (*P-type*). In addition, it covers many extended mnemonics, assembler instructions, and the multiply and divide subroutines. (Refer to "Appendix A" for a list of System/7 Assembler instructions.)

Multiple Assemblies. The assembler is designed to produce multiple assemblies without having to be reloaded. Partial or complete program assemblies may be made without regard for the assembly sequence or the number of programs to be assembled. Assembly options may be specified for each assembly independently of other assemblies.

Subroutine Call. During the assembly, a reference can be made to a subroutine. The referenced subroutine is added at the end of the assembly as an additional source paper tape.

Assembly Process

Figure 3-3 shows the assembly process of the System/7 Assembler.

After the assembler tape is read into storage, the operator is queried on whether the program is to generate:

- The full assembly list containing the source and object program with error notations, a symbol table, and the resolution patch table.
- Only the error list.
- An object paper tape.
- A partial assembly which may be appended to the original object paper tape.

Then the source program (which has been previously punched on paper tape) is entered. The assembler does all the necessary processing of instructions into machine codes and the assigning of storage addresses. Unresolved addresses and other conditions resulting from forward addressing or references to subroutines are resolved at the end of the assembly. Unresolved operands are prepared as patches and are added to the object paper tape. Then the object tape punched by the assembler is ready for loading into the System/7.

The source program is written and punched into paper tape (source tape). The assembly process follows:

1. Read assembler tape into System/7.
2. Select assembly output options from the operator station.
3. Read source tape into System/7.
4. Output options are performed.
5. Additional programs may be assembled; assembly output options are selected again, and the new assembly process begins by loading the source tape. The assembler tape is loaded only *once* per assembly process.

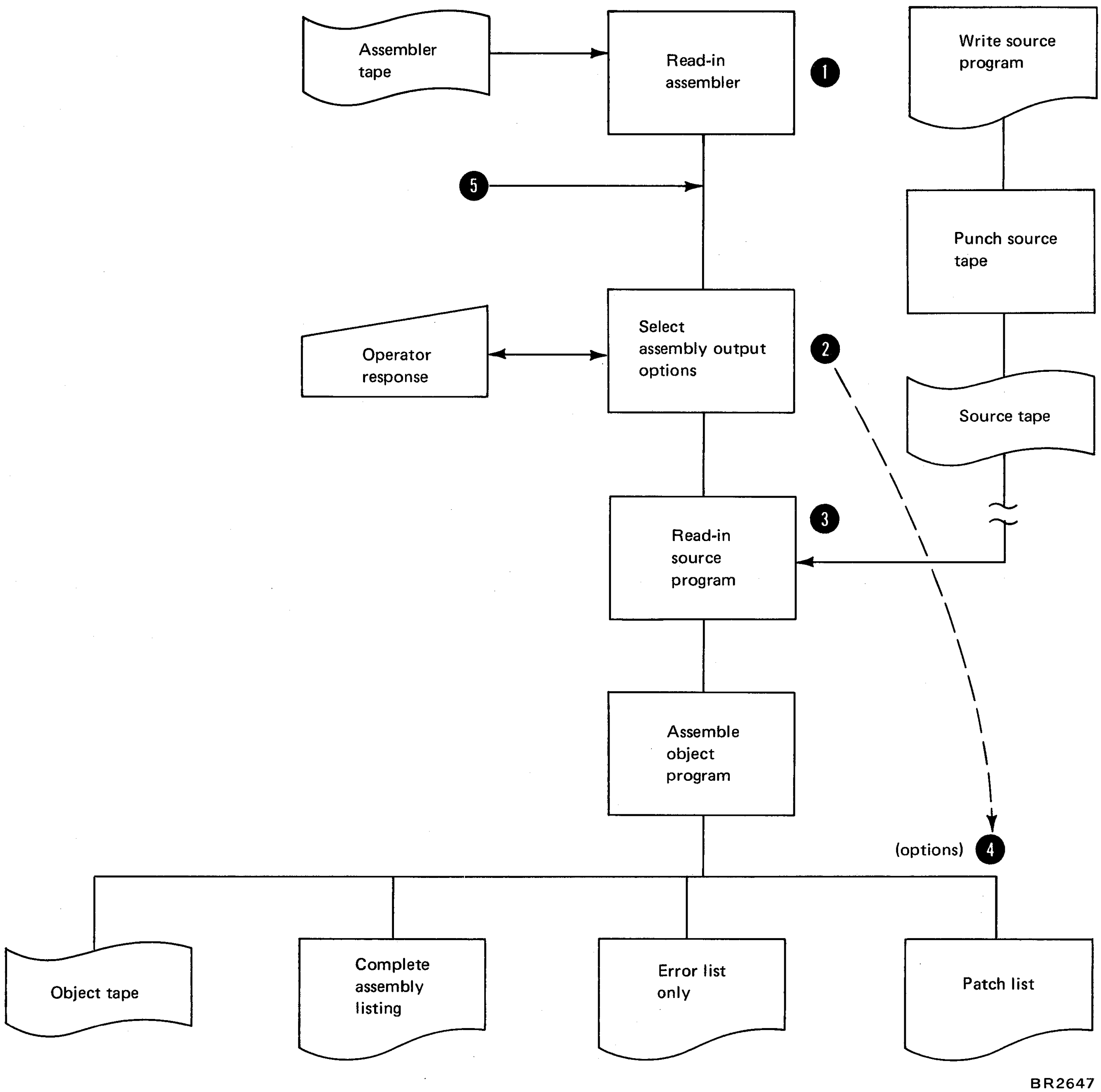


Figure 3-3. System/7 Assembler processing

Chapter 4. System Programming Facilities and Functions

The programmer who is preparing System/7 programs on an IBM host computer, writes his source program in assembler language macro instructions, they are assembled by a macro assembler, and the output handler routine translates the host assembler output into an MSP/7 storage load module. The programmer who does not have access to one of the four IBM host computers writes his source program using mnemonic operation codes, assembler instructions, and subroutines; they are assembled on the System/7 by the System/7 Assembler.

Although this chapter emphasizes the System/7 program extension to the macro assembler libraries of the host computers, compatible object programs can be prepared by the System/7 Assembler. Program segments can also be prepared by the System/7 Assembler and incorporated into a program previously assembled by a macro assembler.

An effective sensor-based application program can be prepared to operate the System/7 and perform the following tasks:

- Service interruptions
- Service sensor-based I/O devices
- Task scheduling through timer control
- Initialize the system
- Error recovery
- Operator/system communication
- Aid in debugging programs
- Support teleprocessing
- Operate under control of 1130 and 1800 Distributed System Programs
- Service 2790 I/O devices
- Service 5022 Disk Storage Module
- Arithmetic subroutines

INTERRUPTION LEVEL PROCESSING

Two types of interruptions can occur on the System/7:

1. *Priority interruptions* are issued when I/O modules request service, programmed clocks signify the completion of a timed interval, or as a result of a Set Programmed Interrupt instruction.
2. *Class interruptions* are issued as a result of a program check, machine check, or power/thermal warning (caused by a power or thermal condition).

Priority Interruptions

- Four levels of priority interruption
- 16 sublevels per interruption level
- Automatic hardware branching to the interruption servicing routine provided by MSP/7
- Instruction address register back-up, accumulator, and seven index registers provided for each level
- Interrupting devices can be assigned to interruption levels and sublevels under dynamic program control

System/7 is an interrupt-driven system that provides direct hardware branching for processing on four interruption levels with sixteen sublevels per interruption level. This permits automatic (rather than program) access to as many as 64 interrupt servicing routines.

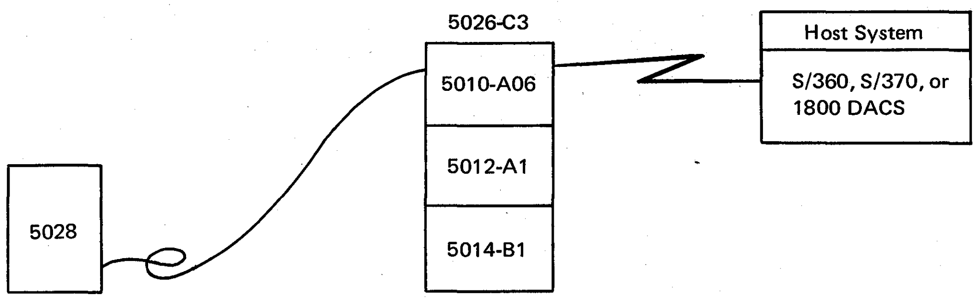
During program preparation, the programmer assigns his various interrupting sources to one of the four interruption levels (numbered 0 through 3). Each device or interrupting source is assigned an interruption level and sublevel (or displacement) by means of the 'prepare I/O' command. This prepares the device to present the processor its sublevel and module identification when an interruption occurs.

As an *example* for the System/7 programmer, Figure 4-1 illustrates one *initial approach* to the coding required for defining a System/7 configuration. Figure 4-2 shows how the priority interruptions are processed in the system shown in Figure 4-1. In the example shown, the current program (level-3 assigned to the operator station) being processed is interrupted by a level-2 priority interruption (2790 Control); the level-2 program is being processed when a level-0 (process interrupt) is branched to; the level-0 program is completely processed; during the level-0 processing, a level-1 priority interruption took place, so it is next to be serviced; the level-2 program is resumed and completed; the level-3 is then resumed and completed. When all processing is completed, the processor enters a wait state, waiting for the next interrupt request and further processing.

Program switching can occur from one level to another with minimum overhead because the machine status of the interrupted program does *not* have to be saved by programming. Instead, each of the four priority levels in the processor has its own set of registers. Thus, machine status switching following an interruption in System/7 is performed in less than 1 microsecond.

A 'set programmed interrupt' instruction to activate a specific interruption level under program control can be used by the programmer. MSP/7 provides a subroutine to permit the user to queue multiple program interruption requests on each level. The queued program interruptions are serviced interspersed with the I/O interruptions so that the 'set programmed interrupt handler' subroutine does not monopolize an interrupt level. MSP/7 also makes provisions for masking out one or all of the interruption levels; while the level is masked, no interruptions on that level are serviced. When the level is unmasked, any pending interruptions will be serviced.

When a change is made from one level to another, the operational registers are automatically saved. An accumulator, an address register, and seven index registers are provided for each level. MSP/7 also has provisions to allocate a level save and work area to each level.



5010-A06	Processor module with asynchronous communications control and 6144 words of storage
5028	Operator station
5026-C3	Enclosure
5012-A1	Multifunction module with analog input (AI), digital input (DI), process interrupt and 2790 Control
5014-B1	Analog input module

Priority level	Sublevel			
	0	1	2	3
Level 0	Program interrupt	Timers	Process interrupt 1	*
1	Program interrupt	Process interrupt 2	5014 AI	*
2	Program interrupt	2790 Control	5012 AI	*
3	Program interrupt	Operator interrupt	Asynch. comm. control (ACCA)	*

*A system macro (null interrupt handler) is automatically included at assembly time for interruptions that the user does not specify. The macro prints an error message containing the level and sublevel along with the device subaddress and module address of the interrupting source.

BR2648

Figure 4-1. Defining a system

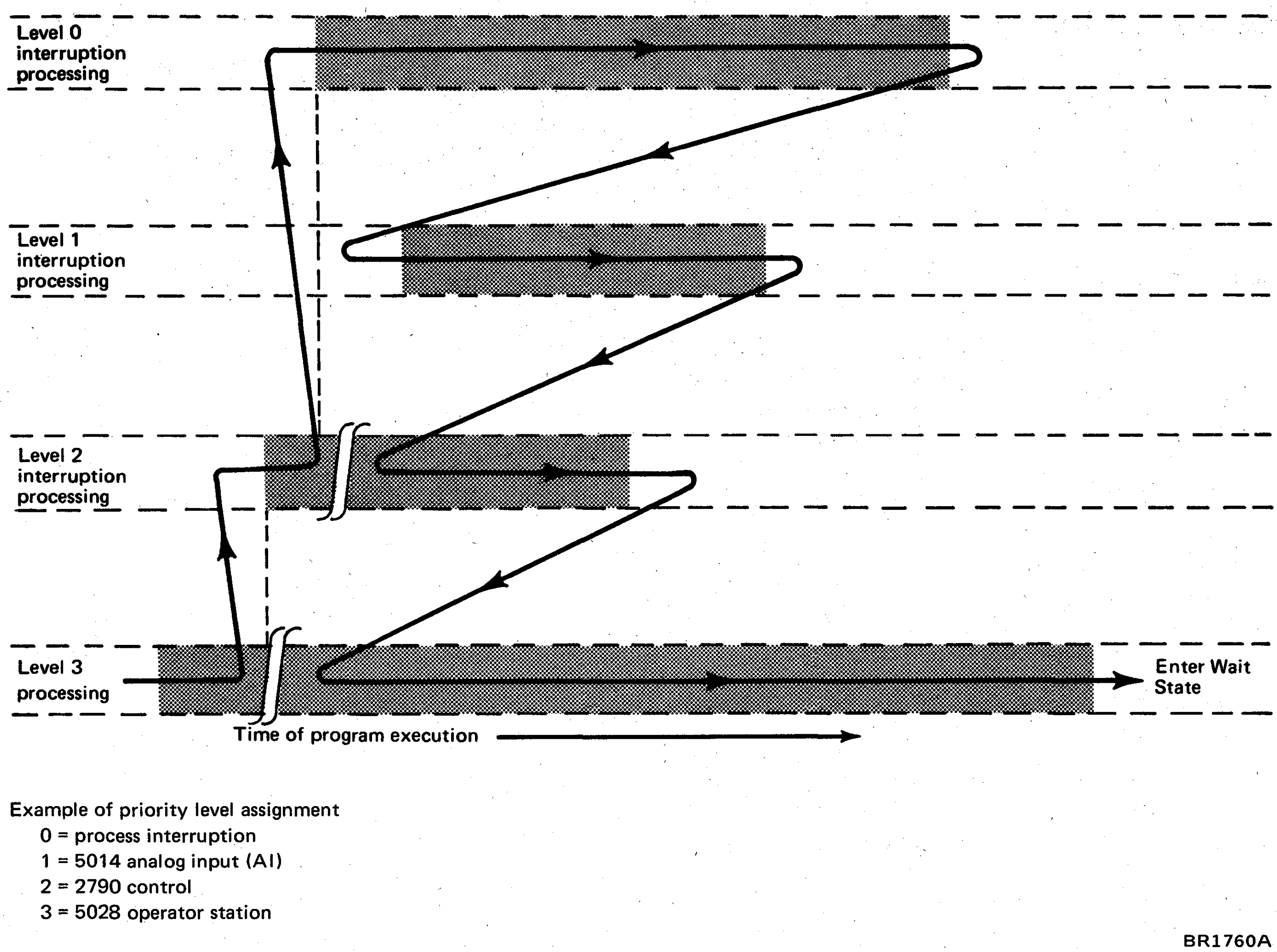


Figure 4-2. Priority interruption level processing

Class Interruptions

- Class interruptions take precedence over priority interruptions.
- Internal machine error conditions can cause one of three types of class interruptions:
  1. Program check, caused by an invalid machine instruction.
  2. Machine check, caused by a machine error or by program execution.
  3. Power/thermal warning, caused by a power or thermal error condition.

Class interruptions cannot be prevented from interrupting the system because the conditions that cause such interruptions are usually serious, and could impair further system operations. Therefore, class interruptions immediately disable all four priority interruption levels and cause a branch via one of the main storage locations reserved for class interruption start addresses. A class interruption does *not* cause a change in priority level; the interruption is serviced on the level that is active when the error condition occurs. If no level is active when a power/thermal warning interruption occurs, the interruption is serviced on priority level 3. When a class interruption is serviced, the servicing program should save any register contents and/or status information needed to restart the program. When the processor is in the stop state, lights on the operator console indicate the cause of a class interruption. In addition, a unique machine instruction (load processor status) checks the processor status to determine the specific cause of the interruption.

SENSOR-BASED INPUT/OUTPUT DEVICE SERVICING

To service his I/O devices, the programmer writes MSP/7 macros with all the required coding to transfer analog or digital data. With additional coding, he can use these same macros to operate the I/O sensor-based devices.

By selecting options and adding any application coding, the user can customize the analog and digital I/O subroutines obtained through the assembler process.

TASK SCHEDULING

Another service provided by MSP/7 is the scheduling of programs to be executed periodically. Macro facilities are available to build a task scheduler table that designates the programs to be run and the frequency of their execution. The task scheduler routine also determines *when* the programs in the table are to be executed.

Options permit the user to dynamically add or delete programs from the task scheduler table.

Two interval timers in the processor module control the scheduling function and time the tasks. The user can optionally select a basic timing routine that provides:

- A 48-bit (3-word) counter representing the number of timer zero interrupts
- Time-of-day clocks (year, day, hour, minute, second, hour/minute)

SYSTEM INITIALIZATION

Before the System/7 processor can execute a program, the operator station and other interrupting sources must be assigned to the desired interrupt level/sublevel, and areas must be reserved (set up) in main storage to hold addresses, data tables, data constants, and the like. An MSP/7 macro ($INIT) prepares the interrupting devices and sets up the areas that are necessary for System/7 to operate under program control (system initialization). Areas reserved in main storage are usually set to a known value (initialized) before program execution begins. Specifically, the macro routine performs the following functions:

- Stores the addresses of the 4 interruption level branch table vectors in storage locations 16 through 19.
- Stores the addresses of the 3 class interrupt routines in storage locations 9 through 11.
- Verifies that the I/O modules defined as interrupting sources are correctly identified to fit the user's configuration.
- Prepares the operator station and the asynchronous communications control adapter, if present.
- Loads, prepares, and starts timer zero, if a timing function is specified in the user's program.

If the user requires additional non-I/O initialization procedures (for user-defined areas, pointers, tables, digital input latch control instruction, relocation of address constants, and so on), he must execute macro instructions defining these functions *before* the MSP/7 initialization is performed. In this way, he will not define areas reserved by the MSP/7 initialization.

TELEPROCESSING SUPPORT

The asynchronous communication control and the MSP/7 communication support program handle data transmission to a remote System/360 or System/370. The communication program and the control have the characteristics of the IBM 2740 Model 1 Communication Terminal with record checking feature (horizontal and vertical checking of each character) and optional station control feature (multipoint). The teleprocessing link can be established over a private, leased, or dial-up line.

Before data transmission can be established, the System/360 or System/370 must have a suitable telecommunication access program resident in its operating system. The communication support program must also be in System/7 storage before data or an object program can be received by the System/7. (A minimal teleprocessing routine must be loaded into the System/7 through paper tape, disk, or using IPL capability of the communications adapter.)

The communication support program allows for both transmission and reception of data (or programs) over switched or nonswitched telephone lines. It initializes and controls the communication link, formats the data, and performs error recovery functions.

With teleprocessing, a host system can IPL a System/7.

1130 AND 1800 DISTRIBUTED SYSTEM PROGRAMS

The 1130 and 1800 Distributed System Programs have subroutines for controlling the exchange of data between an 1130 or 1800 system and a System/7. The packages contain the following:

1. General control subroutines
2. Request control subroutines
3. Transmission control subroutines
4. Interface control subroutines
5. Utility programs

The subroutines may be called from programs written in 1130/1800 assembler or in the 1130/1800 FORTRAN language.

The 1130 connection is through the 1130 Storage Access Channel. The 1800 connection utilizes a start/stop teleprocessing linkage.

2790 CONTROL SUPPORT

The 2790 Control allows the System/7 to be the system controller for I/O devices of the 2790 Data Communications System. MSP/7 controls the 2790 real-time data collection and plant communication system by providing all the required coding to satisfy the operating requirements of 2790 I/O devices. One 2790 Control may be attached to the System/7. The 2790 Control supports a maximum of 16 area stations connected serially, starting and ending at the 2790 Control in a loop configuration. Data entry is through individual I/O devices at work stations which transmit numeric data from card and badge readers, dial selection, and key entry. Alphameric data can be entered from a 2798 Guidance Display Unit. Outputs are provided to printers and guidance display units.

The support routine of MSP/7 performs the following functions:

1. Services interrupts from the 2790 Control.
2. Deciphers requests and activates the necessary functions to accomplish the required tasks.
3. Performs all control sequences and required data checking.
4. Controls the variable data rates for transmission to and reception from the various I/O devices.
5. Controls the routing of messages to the desired device (operator station, transmit to host computer, transmit to user program for additional processing, etc.).

ERROR RECOVERY

Error recovery is an important routine of MSP/7. Whenever possible, the error recovery routine attempts to continue the operation under progress. Unless the user has specified his own error-handling routines, the MSP/7 error recovery routine follows one of these procedures:

1. For recovery from program errors at any of the four interrupt levels, the operator takes action.
2. For recovery from a machine check, MSP/7 exits from the level and prints an error message.
3. For recovery from a power failure, the operator intervenes through an available feature that provides an automatic restart. With this feature, an IPL is performed by the selected unit when power is restored.
4. For extraneous interrupts, an error routine that prints an error message and exits from the level.
5. Alarm messages are printed for I/O device errors only after the specified retry count is exceeded.

Error Logging

Error logging is done by counters assigned to various I/O interrupting sources. Each occurrence of an error is counted by module address and I/O device experiencing the error (if applicable).

OPERATOR/SYSTEM COMMUNICATION SUPPORT

Subroutines service two-way communication between the operator and the system through the operator station. By typing in a coded message, the operator can request System/7 to perform a selected function. The system interprets the message, then provides linkage to the selected functional program. Such messages may request that a program be executed, the time of day read out, or a mask changed.

Messages from the system may request operator intervention, that some data be manually entered, or just that some status information be given.

Character code translation subroutines convert codes for binary, paper tape, and teleprocessing transmission.

DEBUGGING SUPPORT

Several utilities assist in program correction and checkout. They are incorporated into the object program as specified by the user. Usually, they are used only during program checkout and then are deleted.

The support includes a routine to list on the printer or to punch on paper tape the contents of storage (dump).

Another routine prints out the contents of selected registers (snapshot dump) which allows the programmer to trace the execution of a program.

Insertion of data into storage (patch) is supported either from the paper-tape reader or from the keyboard.

5022 DISK SUPPORT

MSP/7 supports one 5022 Disk Storage Module by providing the following functions:

1. Installation and maintenance programs distributed with the System/7 Assembler
   - A disk initialization program controls disk formatting, surface analysis, alternate track assignment, and loading of the IBM supplied IPL load record to disk.
   - A disk patch program provides a method of altering records residing on the disk.
2. Input/output control for
   - Seek, read, write and write verify functions
   - Multiple sector operations
   - Error handling routines

ARITHMETIC SUBROUTINES (INTEGER)

A multiply subroutine allows the user to multiply two 16-bit numbers and generate a 32-bit product.

A divide subroutine enables the user to divide a 32-bit number—possibly the product of a multiplication—by a 16-bit number.

A square root subroutine allows the user to calculate the square root of a 32-bit number.

Assembler Instructions

Mnemonic	*Instruction*
PORG	Program Control
PEQU	Symbol Definition
PDC	Define Word Constant
PDS	Define Storage Area
PEBC	Define EBCDIC Character
PLIST	Assembler List
PEND	Assembly End

Extended Mnemonics

*Indicates extended mnemonics not supported by the System/7 Assembler.

Mnemonic	*Instruction*
*PWRI	Write Immediate
*PRDI	Read Immediate
*PREP	Prepare I/O
*PHIO	Halt I/O
*PSPI	Set Program Interrupt
PBCR	Branch on Condition Register
PBZ	Branch on Zero
PBNZ	Branch on Not Zero
PBP	Branch on Positive
PBNP	Branch on Not Positive
PBN	Branch on Negative
PBNN	Branch on Not Negative
PBNE	Branch on Not Even
PBCY	Branch on Not Carry
*PBA	Branch on Accumulator Contents
PBO	Branch on Overflow
PBER	Branch on I/O Error
PBL	Branch Long Unconditionally
*PBR	Branch to Index Register
*PCLR	Clear Register
*PNOP	No Operation
PSZ	Skip on Zero
PSNZ	Skip on Not Zero
PSP	Skip on Positive
PSNP	Skip on Not Positive
PSN	Skip on Negative
PSNN	Skip on Not Negative
PSE	Skip on Even
PSNC	Skip on No Carry
PSNO	Skip on No Overflow
PSNER	Skip on No I/O Error

Note. For a description of the assembler instructions and all machine mnemonic instructions, refer to the publication *IBM System/7 Modular System Programs Programmer's Guide,* GC34-0013.

Refer to *System/7 Functional Characteristics*, GA34-0003, for the description of the machine instructions.

Instruction	Mnemonic	Operation code	Execution time in nanoseconds
Load and store			
Load accumulator	PL	11000	800
Load and zero	PLZ	11001	1200
Load immediate	PLI	01100	400
Load index long	PLXL	10001	1200
Store accumulator	PST	11010	800
Store index	PSTX	01101	800
Arithmetic			
Add	PA	10000	800
Subtract	PS	10010	800
Add register	PAR	11111	400
Subtract register	PSR	11111	400
Complement register	PCR	11111	400
Add immediate	PAI	01110	400
Logical			
AND	PN	11100	800
OR	PO	11101	800
Exclusive OR	PX	11110	800
AND register	PNR	11111	400
OR register	POR	11111	400
Exclusive OR register	PXR	11111	400
Shifting			
Shift left logical	PSLL	00010	400 + 50N + 50 if odd no. shifts
Shift left circular	PSLC	00010	400 + 50N + 50 if odd no. shifts
Shift right logical	PSRL	00010	400 + 50N + 50 if odd no. shifts
Shift right arithmetic	PSRA	00010	400 + 50N + 50 if odd no. shifts
(N is the number of bits shifted)			
Branching			
Branch	PB	00111	400
Branch and link	PBAL	01011	400
Branch and link long	PBALL	01010	800
Branch on condition	PBC	01000	400–no branch 800–branch taken or if R field = 0
Skip on condition	PSKC	01001	400
Add to storage and skip	PAS	01111	1200
Register-to-register			
Store to register	PSTR	11111	400
Load from register	PLR	11111	400
Interchange register	PIR	11111	400
Load processor status	PLPS	11111	400
Inspect IAR backup	PIIB	11111	800
AND to mask	PNM	10110	400
OR to mask	POM	10110	400
Sense level and mask	PSLM	11111	400
State control			
Level exit	PLEX	00110	400
Stop	PSTP	00100	400
Input/output			
Execute I/O	PIO	00001	1800 + D

(D, the delay inherent in a system configuration, varies from 100 to 2100 ns depending on the physical location of the I/O module concerned. D varies from 100 to 800 ns on systems without a 5026 Enclosure Model D3 or D6.)

Appendix B. System/7 Modular System Programs Macros

Appendix B, shows System/7 macros grouped within their logical functions.

Nucleus macros (Required system or execution and frequently used macros)

Macro	Type	Function
#CONF	Specification	Defines your system configuration; sets flags to notify PEND to include various system routines
#ERP	Specification	Defines entry points to your error recovery routines for class interrupts
$ERP	System	Performs IBM-defined error recovery procedures for class interrupts, unless a #ERP overrides the procedure Performs error logging functions
@INHB	Access	Inhibits interrupt levels
@NABL	Access	Enables interrupt levels
$SPI	System	Executes a set programmed interrupt instruction to activate a specific interrupt level
@SPI	Access	Defines the interrupt level to activate with a PSPI instruction
$QUE	System	Adds and deletes entries from a queue on a first-in/first-out basis
@QIN	Access	Defines the entry to add to a queue
@QOUT	Access	Defines the entry to delete from a queue
$INIT	System	Initializes interrupt level branch tables and class interrupts, prepares natively attached devices, verifies and prepares I/O devices defined by #ISRC, starts timer zero
$NINT	System	Prints an error message indicating that a particular interrupt level branch table has no servicing routine associated with it or an MSP/7 interrupt routine received an interrupt it could not service
#IOLT	Specification	Provides a parameter list of information required to use I/O devices
$LOAD	System	Allows you to load online a 5028 storage load overlay module
@LOAD	Access	Permits access to $LOAD

Operator station macros

Macro	Type	Function
#OPTR	Specification	Generates a table of all program entry points that the operator may access from the operator station keyboard
$OPTR	System	Processes operator requests from the keyboard. These requests may call routines that you have defined or IBM-defined routines. $OPTR also allows the operator to correct or cancel a request
$LVLC	System	Services the operator requests, NBL and INH, if they are defined by #OPTR
$LOCK	System	Services operator request LOK, if it is defined by #OPTR
$OPR	System	Generates the code to transfer ASCII data between the operator station and the storage buffers you define. Performs the I/O function you specify with the three access macros: @OPR, @OPRI, and @OPRO
@OPR	Access	Generates a call to $OPR to queue a standard parameter list that represents a request for service from the operator station. This parameter list is defined with the #IOLT macro
@OPRI	Access	Generates the required parameter list and calling sequence for $OPR to read a message from the keyboard, print the message, and issue a CR and a line feed
@OPRO	Access	Generates the required parameter list and calling sequence for $OPR to print a message, followed by CR and line feed

Timing and scheduling macros

	Macro	Type	Function
Timing	#DBTC	Specification	Defines the basic timing cycle and specifies how $BTIM will be used
	$BTIM	System	Services interrupts from interval timers zero and one and maintains the basic period clock and time of day clocks, specified by #DBTC
	$TIMR	System	Loads interval timer one with a count to establish a delayed execution of a program
	@TIMR	Access	Access to $TIMR to load interval timer one
	@TDAY	Access	Access to $BTIM to set basic period and time of day clocks
Scheduling	#SCHD	Specification	Defines entries in the task scheduler table during assembly
	$SCHD	System	Initiates the execution of programs listed in the task scheduler table
	@SCHD	Access	Adds, deletes, and modifies programs in the task scheduler table during execution

Sensor-based input/output macros

Macro	Type	Function
#ISRC	Specification	Builds an interrupt source table that is used by $INIT to prepare devices that generate interrupts and by interrupt service routines to identify interrupt sources
#AIDT	Specification	Builds a device table for an I/O module that contains analog input functions
#DAIP	Specification	Symbolically describes an analog input point
$AI	System	Processes analog input data
@AIRS	Access	Specifies how analog input points are to be processed
#DAOP	Specification	Symbolically describes an anlog output point
$AO	System	Processes analog output data
@AOSP	Access	Writes data into an anlog output holding register or output register
@AORD	Access	Reads data from either an analog output holding register or output register into a specified storage location, and provides a read-back check for analog output data
#DDIG	Specification	Symbolically describes a digital input group (16 points)
$DI	System	Processes digital input data
@DISG	Access	Reads a digital input group and stores it in a specified storage location
@DICN	Access	Sets or changes any of the control groups (interrupt/noninterrupt, compare equal/unequal and latch/unlatch)
#DDOG	Specification	Symbolically describes a digital output group (16 points)
$DO	System	Processes digital output data
@DOSG	Access	Writes data into either the digital output register or output holding register
@DORD	Access	Reads data from either the digital output holding register or the output register into a specified storage location

Teleprocessing macros

Macro	Type	Function
#COMM	Specification	Defines your communication needs
$COMM	System	Processes your communication messages
@XMIT	Access	Allows message transmission from System/7 to a host computer

Teleprocessing utlities

Utility	Function
UZERO	Clears memory and registers
UTIPL	IPL loader of subsequent TP-loaded storage load

2790 Control macros

Macro	Type	Function
#LOBE	Specification	Defines the 2790 configuration
$LOBE	System	Processes the interrupts from the 2790 Control
#AS	Specification	Builds a table of all area stations and data entry units and associates each to a transaction group
#TGRP	Specification	Builds a transaction group table and associates each entry with a transaction list or another transaction group
#TLST	Specification	Together with #ASTP or #DSTP, defines a transaction
#ASTP	Specification	Defines one step of a transaction for an area station
#DSTP	Specification	Defines one step of a transaction for a data entry unit
@LBGO	Access	Starts the 2790 Control
@LBST	Access	Stops the 2790 Control
@LBWR	Access	Generates the calling sequence for $LOBE to print a message on a 1053 printer

Conversion, arithmetic, and debug macros

	Macro		
	System	**Access**	**Function**
Conversion	$EBAS	@EBAS	Converts EBCDIC to ASCII and vice versa
	$PTAS	@PTAS	Converts PTTC/EBCD to ASCII code and vice versa
	$EBPT	@EBPT	Converts PTTC/EBCD to EBCDIC code and vice versa
	$ASCB	@ASCB	Converts ASCII to binary and vice versa. Printable decimal and hexadecimal ASCII characters are produced or converted
Arithmetic	$MULT	@MULT	Multiplies two 16-bit numbers
	$DIVD	@DIVD	Divides a 32-bit number by a 16-bit number
	$SQRT	@SQRT	Extracts the square root of a 32-bit number
Debug	$DUMP	@DUMP*	Prints or punches the contents of specified storage locations
	$SNAP	@SNAP	Prints contents of IAR, accumulator and index registers
	$PACH	*	Allows corrections to be made to a program through the operator station keyboard without stopping the program in process

*DUMP and $PACH can also be accessed when the operator types DMP and PAT, if the related #OPTR is properly coded

Disk support macros

Macro	Type	Function
#DIT	Specification	Creates required disk information table
#DBB	Specification	Defines drive, volume, and disk address sector boundaries within which disk operations must occur
$DISK	System	Generates the I/O control routine required for disk operation
@DISK	Access	Provides linkage to I/O control routine ($DISK)

Absolute Address. Specific location in storage.
Access. Retrieval of data from an input/output device.
Access Method. Any data management facility available to the user for transferring data between main storage and an input/output device; commonly a general term for telecommunications program support.
Accumulator. Sixteen-bit register that performs both arithmetic and logical operations.
ADC. See *Analog-to-Digital Converter.*
Address. Identification of a register or storage location.
Alphabetic Characters. Any of the characters A through Z (plus the symbols #, $, and @ in MSP/7).
Alphameric. Characters that include letters of the alphabet, numerals, and other symbols, such as punctuation or mathematical symbols.
Analog. Pertaining to representation by means of continuously variable physical quantities; in this publication, the term analog input or analog output implies a continuous voltage applied at the appropriate source.
Analog-to-Digital Converter (ADC). Electromechanical device that senses an analog signal and converts it to a proportional representation in digital form.
ASCII. American Standard Code for Information Interchange.
Assembler. A computer program that converts the programmer's input statements (source program) into an object program.
Assembly. Output of an assembler.

Bit. Binary digit, either 0 or 1; abbreviation of *binary digit.*
Buffer (Buffer Storage). Computer storage used as a temporary holding area during a transfer of information; for example, input or output data and status words.

Call. Program branching or transfer of control to a specified subroutine; invocation of a subroutine.
Channel. Path along which signals can be sent.
Class Interruption. Control signal generated due to an abnormal internal machine condition which causes entry into an error recovery routine.
Clock. Register or storage location whose contents change at regular intervals so as to measure time.
Console. Consists of switches and indicators for communication between the computer operator and the computer.

Data. Any representation, such as a digital or analog quantity, to which meaning has been assigned.
Data Acquisition. Gathering, evaluating, and/or recording of data.
Data Entry. Single block of data entered into the computer by an operator from a single data device, such as a card reader, badge reader, keyboard, or switch.
Data Link. Communication lines and modems used in transmitting information between two or more stations.
Debug. To isolate and correct any errors in a computer program.
Device. Mechanical or electrical contrivance with a specific purpose.
Digital. Pertaining to the use of discrete integral numbers in a given base to represent all the quantities that occur in a problem or a calculation.

Digital Input/Output. Input or output quantity made up of a set of discrete magnitudes which represent the present condition (input) or status to be set (output) in the system. In this publication, *digital input/output* is also used to refer to the hardware systems which accept or provide digital input and output functions.
Disk Operating System. Disk-resident programming system that provides operating system capabilities.
Distributed System. Used in this publication to denote the operation of a small sensor-based computer and a larger host computer, where each computer performs a portion of the application, yet each can operate as a single entity.
Distributed System Programs (DSP). IBM-supplied programs that support IBM 1130 and 1800 distributed system operation.
DOS. See *Disk Operating System.*
DSP. See *Distributed System Programs.*
Dump. Writing of data from a storage device onto a recording medium; for example, on a printer or console typewriter.

Enable. State in which the occurrence of a condition results in an interrupt.

Facility. Anything used or available for use in furnishing communication service; commonly, a general term for communication programming support and equipment.

Initial Program Load (IPL). Procedure that causes the computer to start processing. In System/7, IPL loads a program beginning at storage location 0 and continues until loading is completed; control is given to the instruction at location 0 for further system initialization.
Initialize (Initialization). To set counters, switches, and addresses to 0 or other starting values at the start of, or at prescribed points in a program.
Input/Output Device. Commonly called I/O; a general term for the equipment used to communicate with a computer.
Interruption (Interrupt). Control signal that requests attention of the computer from the current operation; for example, System/7 can recognize a request (interruption) for servicing a sensor-based I/O unit, stop the current operation, service the I/O request, and return to the original operation at the point it was stopped.
Interval Timer. Electronic counter that counts intervals of time under program control.
IPL. See *Initial Program Load.*

Load. Fetch and store; to read into storage.

Macro. Program source statement written in assembler language, interpreted by a macro assembler to perform a function or call a subroutine.
Macro Assembler. Program that converts macros (defined in a macro library) into machine instructions.
Macro Library. Collection of macro definitions that can be stored for use by a macro assembler.
Magnetic Disk Storage. Storage device that uses magnetic recording of data on flat rotating disks.
Mnemonic. Code or abbreviation that represents an instruction to the computer.
Mnemonic Operation Code. Computer instruction written in symbolic notation that is converted into an operation code by the computer.
Module. (1) A physical component of the System/7; an I/O module. (2) The same as a routine; a separate program unit that can be combined with other similar units.
Monolithic Integrated Circuit. Type of integrated circuit wherein the substrate is an active material, such as the semiconductor silicon.

Monolithic Storage. Storage made up of monolithic integrated circuits.
Multiplexer. Hardware device that allows handling of multiple signals over a single channel; for example, many analog signals are channeled to one analog-to-digital converter (ADC).
Multisystem. Computers that are interconnected, either by direct connection between channels or teleprocessing linkage, so that each can communicate with and draw upon the resources of the others.

Object Program. Fully assembled program; output of an assembler (executable machine-language instructions); can be one or more storage load modules.
One-Pass Assembler. Assembler requiring only one reading of the source program.
Operating System. Software that controls the execution of computer programs and which may provide scheduling, debugging, input/output control, accounting, compilation, storage assignment, data management, and related services.
OS. See *Operating System.*
Output Handler. Program that converts assembler output into a storage load module.

Priority Interruption Assignment. Priority interruption levels must be assigned using the interaction of functions with each other as a primary basis. Before program and hardware checkout, the on-line systems designer must ensure that all possible interruptions are operating compatibly when worst-case conditions occur. It may be necessary to reassign the priority levels of key interruptions dynamically under program control.
Priority Interruption Level. Operations are assigned an order of preference based on urgency.
Processor. Device (unit or module) capable of receiving data, manipulating it, and supplying results, all under program control; System/7 digital computer.
Program. Series of logical steps designed to do a specific job.

Queue. List of items in a computer system waiting their turn in line for service.

Real Time. Pertaining to the performance of a computation during the actual time that the related physical process transpires in order that the results of the computation are useful in guiding the physical process. For example, the automatic control of manufacturing processes by a computer in an oil refinery.
Register. Circuitry used for temporary storage of data to be used in arithmetical, logical, or transferral operations.
Routine. Ordered set of computer instructions that may have some general or frequent use.

Satellite Computer (Real-Time). As a satellite computer, the real-time system relieves the larger system of time-consuming input and output functions, such as monitoring the input from sensors at a physical process, and sending output signals to control a physical process.
Satellite Processor. Subordinate processor to a higher-priority processor in a distributed system.
Sensor. A device that converts measurable elements of a physical process into data meaningful to a computer.
Sensor-based. Pertaining to an environment that uses sensing devices, such as transducers or temperature sensors, to monitor a physical process.
Sensor-based Computer. A computer designed and programmed to receive real-time data (analog or digital) from transducers (sensors) and other data sources that interface with a physical process. The computer may also generate signals to elements that control the process. For example, the computer might receive data from a gauge or flowmeter, compare the data with a predetermined standard, and then produce a signal that operates a relay, valve, or other control mechanism.

Sensor-based System. An organization of components, including a computer whose primary source of input is data from sensors located at a physical process, and whose output can be used to control the related physical process.
Source Language. Original form (format) in which a program is prepared prior to processing by the computer; source programs are written using a source language format.
Source Program. Program that serves as input to an assembler; programmer's input code/statements; program written in a source language.
Start-stop System Communication. Data transmission in which each group of code elements corresponding to an alphabetical signal is preceded by a start signal which serves to prepare the receiving mechanism for the reception and registration of a character, and is followed by a stop signal, which serves to bring the receiving mechanism to rest in preparation for the reception of the next character.
Storage. Pertaining to a device (such as a magnetic disk or monolithic integrated circuit) where data can be entered, held, and retrieved.
Storage Cycle. Periodic sequence of events occurring when information is transferred to or from main storage.
Storage Load Module. Logical group of machine instructions in a format suitable for loading into main storage for execution; the output of an output handler.
Subroutine. Portion of a program that defines desired operations and which may be included in another program to produce the desired operations.
System. The organization of hardware, software, and people for cooperative operation to complete a set of tasks for desired purposes.

Telecommunications (Teleprocessing). Method of machine-to-machine communication over a telephone or equivalent type line.
Teleprocessing. See *Telecommunications.*
Timer. Device that signals the end of a set interval of time.
Transaction. Series of data entries; may consist of 1 to 13 data entries from an IBM 2796 Data Entry Unit, or 1 to 16 data entries from a 2791 Area Station or 2795 Data Entry Unit.
Transaction Group. Set of 9 transactions associated with an input device such as an IBM 2791 Area Station, 2795 Data Entry Unit, or a 2796 Data Entry Unit.

Word. Unit of storage; 16 bits in System/7.

Index

When more than one page reference is given, the major reference is first

READER'S COMMENT FORM

IBM System/7 System Summary GA34-0002-1

- Your comments, accompanied by answers to the following questions, help us produce better publications for your use. If your answer to a question is "No" or requires qualification, please explain in the space provided below. Comments and suggestions become the property of IBM.

	Yes	No
• Does this publication meet your needs?	☐	☐
• Did you find the material:		
Easy to read and understand?	☐	☐
Organized for convenient use?	☐	☐
Complete?	☐	☐
Well illustrated?	☐	☐
Written for your technical level?	☐	☐

- What is your occupation? ______________________________.
- How do you use this publication?

As an introduction to the subject?	☐	As an instructor in a class?	☐
For advanced knowledge of the subject?	☐	As a student in a class?	☐
For information about operating procedures?	☐	As a reference manual?	☐

Other ______________________________.

- Please give specific page and line references with your comments when appropriate. If you wish a reply, be sure to include your name and address.

COMMENTS:

- **Thank you for your cooperation. No postage necessary if mailed in the U.S.A.**

GA34-0002-1

YOUR COMMENTS, PLEASE...

Your answers to the questions on the back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Note: Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

Cut Along Line

Fold Fold

FIRST CLASS
PERMIT NO. 110
BOCA RATON, FLA
33432

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
General Systems Division
Boca Raton, Florida 33432

Attention: Systems Publications, Department 707

Fold Fold

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

GA34-0002-1

IBM System/7 System Summary Printed in U.S.A. GA34-0002-1

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)