File Number 1440-00
Form F20-0340

IBM Systems Reference Library

Planning for an IBM 1440 Data Processing System

This publication discusses some of the major considerations in planning for an IBM 1440 Data Processing System. Chapters on both systems planning (file organization, file techniques, timing, etc.) and installation planning (education, physical planning, conversion, etc.) are included.

Contents

Introduction

The availability of the IBM 1440 Data Processing System places a powerful computer at the disposal of a much wider segment of business than ever before. Many companies installing this system will be experiencing the planning, installing and operating aspects of a stored-program machine for the first time. For many other organizations already using data processing systems this will be the first experience with advanced storage devices such as the 1311 Disk Storage Drive. The purpose of this manual is to assist these companies in installing the 1440 by providing detailed information on systems and installation planning. The intent, however, is not to provide standard formulas but rather to provide guidelines which can be modified to meet individual needs.

The sections on systems planning – systems design, file techniques, timing, programming, etc. – contain numerous technical details; those on installation planning – organizing the data processing department, education, physical planning, conversion, etc. – are more general in nature. Similarly, since installing a 1440 requires the active participation of both management and data processing personnel, portions of the manual are of interest to management, others to systems analysts and programmers. Each reader should select the portions appropriate to his requirements.

Throughout the manual there are illustrative forms which can be used for planning, reporting and operating functions. Most of them have been developed and are being employed by present users of data processing systems. They have been included as examples to assist users in the development of forms adapted to their own special requirements.

This manual is intended to supplement other IBM services such as schools, test centers and previously published IBM manuals that contain detailed information on equipment, programming, systems and applications. The publications relevant to a particular user are supplied to him by his local IBM branch office.

1440 Data Processing System

Organizing the Data Processing Department

Because of the differences among companies in size, type of business and philosophy of organization, it is impossible to prescribe a standard data processing organization. However, experience has taught at least two general lessons. The first is that a successful data processing organization requires active support from top management. The second is that within the data processing organization certain specific functions must be assigned as the definite responsibility of particular individuals.

Management Participation

The extensive changes in systems and methods, individual responsibilities, and the potential for savings that can result from installing a 1440 Data Processing System make it important that top management actively direct and coordinate the undertaking. Accordingly, the manager of the data processing department normally reports directly to a vice-president, controller or some other equally responsible officer. The executive responsible for the installation of the system should be sufficiently familiar with the details of the installation program to be able to select the proper courses of action required for the success of the program.

The data processing manager is usually assigned the direct responsibility for the data processing system. He has the authority and responsibility for planning the applications, selecting the equipment and installing the system, as well as for operating and maintaining the system once it is installed. He receives assistance from management in overall planning, in ironing out any interdepartmental differences, in implementing required changes in policy and procedures, and in budgetary control.

Functions to Be Performed

Setting up an efficient data processing organization requires a clear understanding of each of the major functions to be carried out. These will be discussed later in the manual. For the present, however, they will be introduced to the reader and explained very briefly.

The major functions may be divided into two groups: planning and development functions, which are primarily performed before the physical installation of the data processing system, and operational functions, which start after the system is installed. This division should not, however, obscure the fact that planning for all functions takes place before installation.

The following represent planning and development functions:

Problem definition. – This involves the study of existing company operations to determine the applicability and profitability of data processing equipment. It requires defining applications in sufficient detail to estimate systems timing, cost and personnel requirements.

Systems design. – In systems design the detail input, output and processing requirements are established, and a system flowchart is prepared showing the sources of data, the operations to be performed and the results produced. The flowchart is usually supported by descriptions of the operations to be performed and by layouts of the input and output documents. One or more passes of data through the computer will be shown, as required by the application. Each pass of the data represents a program to be written for the computer.

Programming. – From information documented during systems design, the programmer prepares a block diagram showing the sequence of events necessary to accomplish the objective specified for each run. The program is written in a symbolic programming language which is later translated into machine language by the computer. The machine language, or object program, is then tested on sample data. Any corrections to the program are then made. The programmer also prepares operating instructions for use of the personnel who will run the computer.

Liaison. – A close working relationship must be maintained between those doing the systems design and programming and those within the company who are most familiar with the various requirements and objectives of a particular application.

Procedures and documentation. – The installation of a data processing system will cause changes in the operations of both the departments supplying data and those receiving the system's reports and results. The establishment, clarification and documentation of these changes is a function which must be performed. Moreover, procedures and methods within the data processing department itself must be established. Programming and documentation techniques, standard program routines and operating procedures must also be set up.

Conversion. – The function of planning and effecting an orderly conversion from existing methods to those of the data processing system must be performed. Several matters which are discussed later can be included here in the broad category of conversion. These are: conversion of data files, physical site preparation, and education of personnel both in and out of the data processing department.

Turning now to postinstallation problems, the following represent functions which are essential to the successful operation of a data processing system.

Supervision/management. — All work must be carefully scheduled in advance if conflicts are to be avoided and deadlines met. Any offline auxiliary equipment must also be scheduled.

Controls .— The receipt and disposition of input and output material will have to be controlled. System and accounting controls, such as batch totals, record counts, daily cash and billing amounts, etc., will have to be maintained.

Console Operation. — A console operator is usually responsible for machine operation. His duties consist of mounting disk packs, putting cards and forms into feeding mechanisms, operating the system from the console, etc.

Auxiliary operation. — Many data processing systems require supporting punched card equipment. In this event an auxiliary machine operator will be required. He may also serve as the console operator.

Library control and maintenance. — The responsibility for control (issue, receipt and storage) of disk packs, program decks and operating records must be assigned. This is often the duty of the librarian. In smaller installations this person can assume other duties such as secretarial, reception, clerical, etc.

Program maintenance. — Existing applications must be maintained. This includes correcting program deficiencies (usually errors of omission in handling unusual situations), making requested changes to existing programs, improving programs, writing small one-time routines, etc.

Conversion. — During the conversion of an application to the data processing system, files must be created, pilot operations run, etc.

Documentation. — Operating records must be kept up to date and any changes to programs or procedures must be documented.

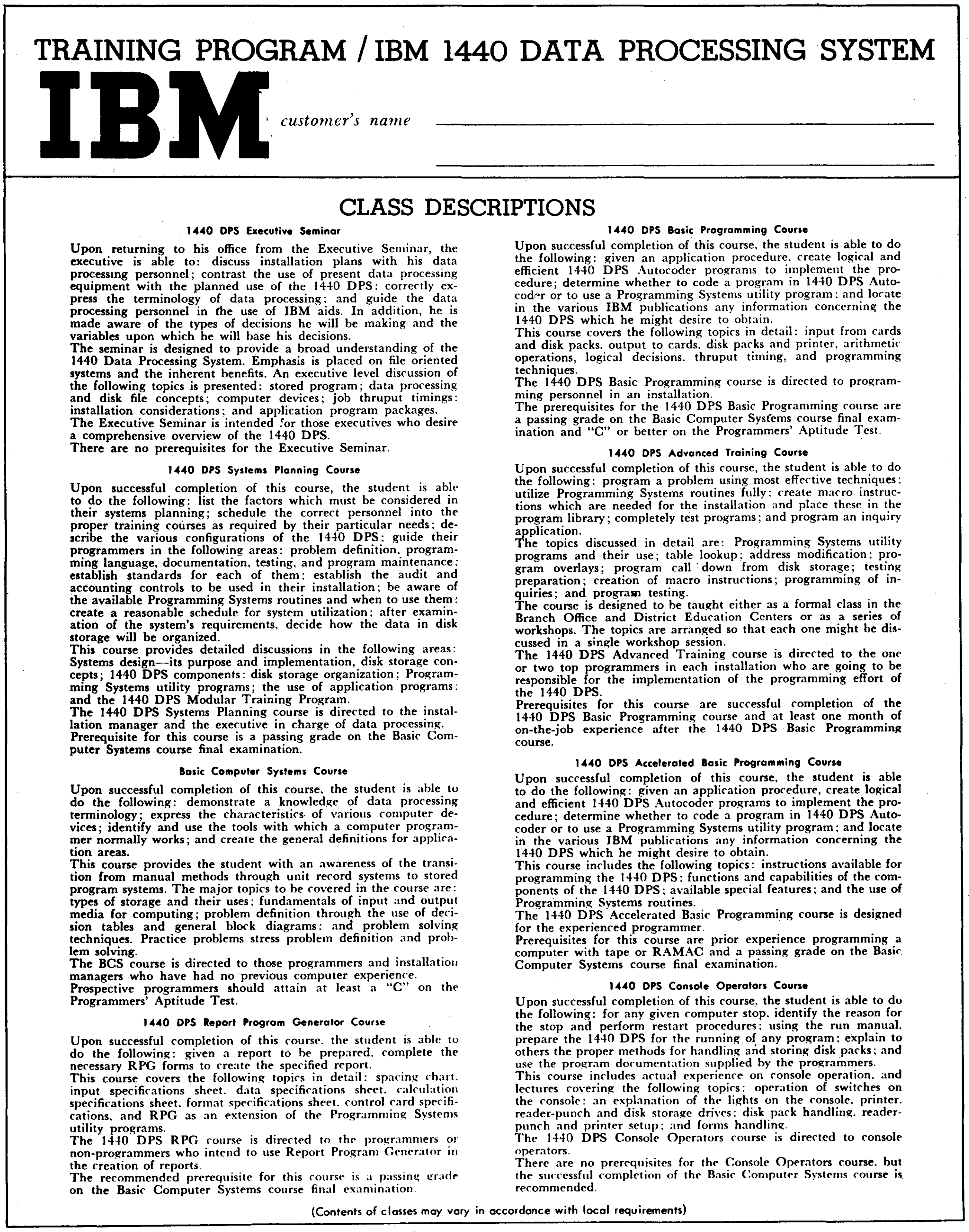
Demonstrations and training. — Both public relations and educational needs must be met.

Selection of Personnel

The selection of personnel to perform the functions listed above is of primary importance and has as much influence on the success of an installation as any other factor.

While data processing personnel should, of course, have considerable general intelligence, they should also possess certain specific aptitudes. These may be measured to some extent with the aid of an aptitude test which IBM will supply on request. Grades attained by those who have taken this test and subsequent success in the data processing field have a definite relationship.

Other important factors also require consideration. Among these are the experience and past performance of the individual being considered, his educational qualifications, his interest in data processing, his knowledge of those areas of the business being considered for data processing applications, etc.



TRAINING PROGRAM / IBM 1440 DATA PROCESSING SYSTEM

IBM *customer's name* ______________________

____

CLASS DESCRIPTIONS

1440 DPS Executive Seminar

Upon returning to his office from the Executive Seminar, the executive is able to: discuss installation plans with his data processing personnel; contrast the use of present data processing equipment with the planned use of the 1440 DPS; correctly express the terminology of data processing; and guide the data processing personnel in the use of IBM aids. In addition, he is made aware of the types of decisions he will be making and the variables upon which he will base his decisions.
The seminar is designed to provide a broad understanding of the 1440 Data Processing System. Emphasis is placed on file oriented systems and the inherent benefits. An executive level discussion of the following topics is presented: stored program; data processing and disk file concepts; computer devices; job thruput timings: installation considerations; and application program packages.
The Executive Seminar is intended for those executives who desire a comprehensive overview of the 1440 DPS.
There are no prerequisites for the Executive Seminar.

1440 DPS Systems Planning Course

Upon successful completion of this course, the student is able to do the following: list the factors which must be considered in their systems planning; schedule the correct personnel into the proper training courses as required by their particular needs; describe the various configurations of the 1440 DPS; guide their programmers in the following areas: problem definition, programming language, documentation, testing, and program maintenance; establish standards for each of them: establish the audit and accounting controls to be used in their installation; be aware of the available Programming Systems routines and when to use them: create a reasonable schedule for system utilization: after examination of the system's requirements, decide how the data in disk storage will be organized.
This course provides detailed discussions in the following areas: Systems design—its purpose and implementation, disk storage concepts; 1440 DPS components: disk storage organization; Programming Systems utility programs; the use of application programs: and the 1440 DPS Modular Training Program.
The 1440 DPS Systems Planning course is directed to the installation manager and the executive in charge of data processing.
Prerequisite for this course is a passing grade on the Basic Computer Systems course final examination.

Basic Computer Systems Course

Upon successful completion of this course, the student is able to do the following: demonstrate a knowledge of data processing terminology; express the characteristics of various computer devices; identify and use the tools with which a computer programmer normally works; and create the general definitions for application areas.
This course provides the student with an awareness of the transition from manual methods through unit record systems to stored program systems. The major topics to be covered in the course are: types of storage and their uses; fundamentals of input and output media for computing; problem definition through the use of decision tables and general block diagrams: and problem solving techniques. Practice problems stress problem definition and problem solving.
The BCS course is directed to those programmers and installation managers who have had no previous computer experience.
Prospective programmers should attain at least a "C" on the Programmers' Aptitude Test.

1440 DPS Report Program Generator Course

Upon successful completion of this course, the student is able to do the following: given a report to be prepared, complete the necessary RPG forms to create the specified report.
This course covers the following topics in detail: spacing chart, input specifications sheet, data specifications sheet, calculation specifications sheet, format specifications sheet, control card specifications, and RPG as an extension of the Programming Systems utility programs.
The 1440 DPS RPG course is directed to the programmers or non-programmers who intend to use Report Program Generator in the creation of reports.
The recommended prerequisite for this course is a passing grade on the Basic Computer Systems course final examination.

1440 DPS Basic Programming Course

Upon successful completion of this course, the student is able to do the following: given an application procedure, create logical and efficient 1440 DPS Autocoder programs to implement the procedure; determine whether to code a program in 1440 DPS Autocoder or to use a Programming Systems utility program; and locate in the various IBM publications any information concerning the 1440 DPS which he might desire to obtain.
This course covers the following topics in detail: input from cards and disk packs, output to cards, disk packs and printer, arithmetic operations, logical decisions, thruput timing, and programming techniques.
The 1440 DPS Basic Programming course is directed to programming personnel in an installation.
The prerequisites for the 1440 DPS Basic Programming course are a passing grade on the Basic Computer Systems course final examination and "C" or better on the Programmers' Aptitude Test.

1440 DPS Advanced Training Course

Upon successful completion of this course, the student is able to do the following: program a problem using most effective techniques; utilize Programming Systems routines fully: create macro instructions which are needed for the installation and place these in the program library; completely test programs; and program an inquiry application.
The topics discussed in detail are: Programming Systems utility programs and their use; table lookup; address modification; program overlays; program call down from disk storage; testing preparation; creation of macro instructions; programming of inquiries; and program testing.
The course is designed to be taught either as a formal class in the Branch Office and District Education Centers or as a series of workshops. The topics are arranged so that each one might be discussed in a single workshop session.
The 1440 DPS Advanced Training course is directed to the one or two top programmers in each installation who are going to be responsible for the implementation of the programming effort of the 1440 DPS.
Prerequisites for this course are successful completion of the 1440 DPS Basic Programming course and at least one month of on-the-job experience after the 1440 DPS Basic Programming course.

1440 DPS Accelerated Basic Programming Course

Upon successful completion of this course, the student is able to do the following: given an application procedure, create logical and efficient 1440 DPS Autocoder programs to implement the procedure; determine whether to code a program in 1440 DPS Autocoder or to use a Programming Systems utility program: and locate in the various IBM publications any information concerning the 1440 DPS which he might desire to obtain.
This course includes the following topics: instructions available for programming the 1440 DPS: functions and capabilities of the components of the 1440 DPS: available special features: and the use of Programming Systems routines.
The 1440 DPS Accelerated Basic Programming course is designed for the experienced programmer.
Prerequisites for this course are prior experience programming a computer with tape or RAMAC and a passing grade on the Basic Computer Systems course final examination.

1440 DPS Console Operators Course

Upon successful completion of this course, the student is able to do the following: for any given computer stop, identify the reason for the stop and perform restart procedures: using the run manual, prepare the 1440 DPS for the running of any program: explain to others the proper methods for handling and storing disk packs: and use the program documentation supplied by the programmers.
This course includes actual experience on console operation, and lectures covering the following topics: operation of switches on the console: an explanation of the lights on the console, printer, reader-punch and disk storage drives: disk pack handling, reader-punch and printer setup: and forms handling.
The 1440 DPS Console Operators course is directed to console operators.
There are no prerequisites for the Console Operators course, but the successful completion of the Basic Computer Systems course is recommended.

(Contents of classes may vary in accordance with local requirements)

Figure 1. 1440 Class Descriptions

Education

Available Tools

An effective education program is designed to meet the needs of the individuals at each company and depends on such variables as the experience of the personnel, the functions the various individuals are to perform, the time schedule to be followed, and the location and availability of training classes. Most effective education programs combine a number of tools – classroom training, installation workshops, reading, on-the-job training, seminars, and demonstrations. Each of these is discussed below.

Classroom Training

IBM education centers – or branch offices, depending on location – provide a series of 1440 modular training classes to allow customer personnel to attend the specific courses required to meet their needs. Figure 1 is an example of the courses presently available. In addition to these courses, others are given at Endicott, Poughkeepsie and San Jose. These include computer seminars for executives, methods classes for data processing managers, and special industry classes.

Installation Workshops

Installation workshops, held in branch offices when demand and circumstances warrant, are divided into two parts: (1) teaching sessions which supplement the classes described above and provide for an interchange of ideas and techniques among users, and (2) working sessions for personnel engaged in preinstallation systems design and programming.

The teaching sessions cover a number of topics including installation planning, disk storage organization, utility programs, programming techniques, testing procedures, etc. Usually a topic can be presented in about three hours. All of the topics may be incorporated into one continuous course or, more likely, scheduled as a series of half-day lectures.

The workshop sessions consist of a group of systems analysts and programmers working on systems design and programming on a regularly scheduled basis. IBM systems engineers are available at the workshop for guidance and to answer technical questions.

Figure 2 illustrates a common sequence of classes and installation workshops. Naturally, the program to be followed by a particular installation is dependent on its needs and must be worked out individually. The IBM sales representative or systems engineer will provide information on the available classes and advice on establishing an education program. Form R25-1629, containing the class descriptions and two planning charts, is also available to assist in education planning. (See page 89 for more details on this form.)

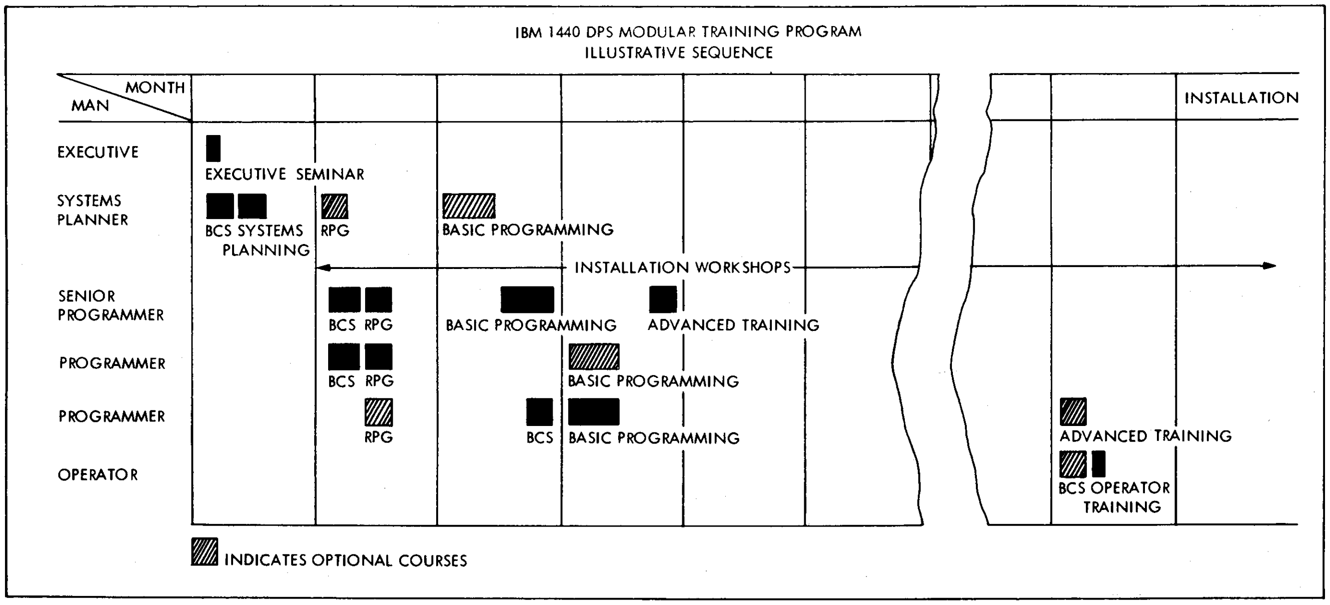


Figure 2. Illustrative Education Schedule

Publications

There are over 100 IBM publications on the 1440 and related subjects. Most of these are contained in the Systems Reference Library which each customer receives. This library consists of a series of manuals in a loose-leaf binder, describing the 1440 components and programming systems, plus a copy of all 1440 installation supplies. It is organized in five main sections:

Systems information (condensed) — introductory and summary publications on system units, features, and programming.

Machine system — basic reference publications on each unit, the special features of the system and physical installation.

Programming systems — general and detailed publications on each 1440 programming system.

Installation supplies — forms and reference cards to facilitate installation, programming and operation.

Supplementary information — education material and special-use publications.

A bibliography of the above material is also provided. The library should be kept intact for reference purposes, in a location accessible to all members of the department. Any additional manuals required may be ordered from the local IBM branch office.

In addition to the Systems Reference Library, other IBM promotional, application and general systems publications are available. Promotional brochures briefly describe the highlights of a data processing system, programming system, or application. Application manuals describe the jobs or procedures performed on data processing systems. General systems manuals describe systems considerations applicable to more than one data processing system. The IBM sales representative or systems engineer will supply the appropriate publications in these categories.

In addition to IBM publications, other material can be found in periodicals, and in books and manuals available from bookstores or business libraries.

Seminars

Soon after the order for the system is placed, an introductory seminar is usually conducted by the user for the personnel concerned with the installation of the 1440. The seminar's format will vary but the general purpose is normally the same: to explain the functions of the various 1440 units, to review the major applications, and to draw the personnel into active participation in the installation plans. A suggested general outline for the seminar is:

Introduction — purpose of seminar
Role of the 1440 in company operations
1440 machine description (film optional)
Organization
- a. Management's participation and function
- b. Staffing for planning and operation
  - Selection
  - Education
- c. Progress reporting
- d. Coordinating responsibility

Systems design
- a. Applications
- b. Scheduling
- c. Exceptions
- d. Controls

Discussion period

An introductory seminar is an excellent method of establishing a means of coordinating effort between executives, middle management and the data processing personnel responsible for the installation of the 1440. A well organized seminar is very helpful in creating the interest and enthusiasm essential to a successful installation.

Demonstrations

Whenever possible, visits should be made to other 1440 installations, or to installations performing similar functions on other data processing systems. Many insights can be gained into procedural, operational, programming and personnel considerations.

On-the-Job Training

While much of the basic information can be obtained from classes, manuals, etc., on-the-job training is very important in the education of data processing personnel. On-the-job training is particularly important for new programmers. When a programmer returns from school he should be given a few relatively easy programs to write under the guidance of an experienced programmer. Similar on-the-job training should be given in areas whenever the need becomes apparent.

Considerations in the Training of Personnel

The previous section outlined education from the viewpoint of the various tools available; this section discusses education programs from the viewpoint of the types of personnel to be trained.

Management

If at all possible, management and executive personnel should attend the executive classes conducted by IBM. These classes will acquaint them with systems concepts, applications and installation procedures. The sharing of ideas and concepts with other executives attending the class should be a valuable experience.

Another excellent method of learning the concepts of the system is to see the 1440 in use. Demonstrations of applications being performed at other customers' installations are particularly helpful.

Programming Personnel

Programming personnel should attend programming classes at an IBM education center or branch office. Once the programmer has put into practice the theory learned in the programming class, he should attend an advanced training course which will further develop sound programming practices. Returning to the programming effort, the programmer can then refine and complete his programs.

Generally the first test session is as much an education process as it is a test of programs. The cycle of programming, desk checking and testing will continue until all applications are completed and ready for operation. As programs are written and tested, the programmer will experience on-the-job training, an invaluable part of his education.

Programming personnel should be aware of the functions and operation of the IBM 1440 utility programs since they will reduce the overall programming effort. Programs pertinent to the system being installed are taught in special classes or in workshops.

Operating Personnel

Ideally, operating personnel should attend a two-week basic computer systems concepts course before the arrival of the 1440. While this training is not an absolute necessity, it will greatly aid the understanding of the internal functions of the machine and improve operating efficiency. After completion of the concepts course the operating personnel should attend a console operations class.

In some installations, program assembly and testing are a function of the console operators, rather than the programming personnel. In these cases standard procedures for program assembly, testing and reporting the results should be taught to the operators. It is most important that good communication exist between the programming and operating personnel if these two vital facets of the installation are to function smoothly.

Other Than Data Processing Personnel

Brochures, company publications, seminars and departmental meetings are the usual methods for advising employees of management plans and acquainting general personnel with the 1440 Data Processing System and its function in the company.

Personnel supplying data to be processed and those receiving data processed by the system should be informed of the systems requirements and results. Changes in methods, controls, format and data handling should be fully explained. Before conversion, a class is normally conducted to educate personnel in the new procedures and give them a full understanding of their own functions in the new system. In addition to the class, a dry run through the new procedures is suggested.

After the 1440 has been installed, an open house may be arranged to show the 1440 Data Processing System in actual operation. The open house will complete the initial education cycle. However, a continual education program will help to promote a sound installation.

Systems Design

Summary of Systems and Programming Effort

The systems development effort may be divided into three phases: problem definition, detailed systems design and programming. While these phases overlap and vary somewhat from one installation to another, the following outline typifies what is done.

Problem Definition

The problem definition or general study phase is concerned with (1) the broad investigation of a company's operations and procedures, (2) the selection of a number of application areas for further analysis, (3) the study and documentation of the present procedures, including volumes, costs and schedules, (4) the consideration of alternate approaches and data processing systems, including their timing and costing, and usually culminates with (5) the documentation of the best approach and the submission of recommendations to management.

Detailed Systems Design

Detailed systems design is concerned with providing the precise information required for programming and operating the procedure. Generally, some preliminary systems analysis is performed during problem definition and then refined in the detailed system design.

After the application has been sufficiently defined it is broken down into a series of computer runs and, where necessary, auxiliary machine runs. To accomplish this, the systems analyst goes through an iterative process which ordinarily involves (1) segmenting the application into runs on the basis of core storage capacity, number of available disk packs, natural job breaks, etc., (2) laying out the runs in detail, including the design of file and input/output records, (3) preparing a flowchart of the systems design, (4) estimating running time, and (5) evaluating and optimizing the systems design. After the analyst arrives at the "best" design, he assembles and completes the necessary documentation and, upon obtaining appropriate approval, either starts programming himself or turns over the material to a programmer. (Before programming starts, the data processing manager decides the sequence in which the various programs should be written and what programming language or system should be used for each program.)

Programming and Program Testing

Guided by the information provided by the detailed system design, the programmer prepares an overall logic block diagram of the run, checks it with the systems analyst and then does the detailed programming, entering the required symbolic language program instructions on the coding sheets.*

Upon completion of the programming and preliminary desk checking, the coding sheets are used as the basis for the punching and verification of the source-language program cards. After the cards are interpreted, listed, edited and checked, the program is ready to be assembled.

The next step is to prepare the data that will be used in testing the program after it is assembled. Where possible, the data and results are created by the old manual or machine methods.

When the time for the test session arrives, the programs that are ready are forwarded to the test center and assembled. In this process a program listing, showing both the symbolic and actual instructions and addresses, is prepared as well as a deck of object program cards. The program listing and cards are returned to the programmer, who then desk-checks the assembled program. Any errors are corrected and, if necessary, the program is reassembled.

After the program has been successfully assembled, the object program cards and the test data cards are forwarded to the test center for testing of the object program. Program testing involves processing the test data through the object program until an error or end-of-job condition is reached and then printing out the test results. The programmer is given the various printouts to assist him in checking out and modifying the program. Some programs may require several cycles of testing, program modification, reassembly and desk checking.

After satisfactory testing has been accomplished the final documentation stages are completed, including the preparation of a console manual and a run book. The program and the documentation are then tested with live data. After any required corrections are made, the program is ready for operational use.

Further details of the systems and programming effort briefly described above are provided in subsequent chapters.

* The basic idea of symbolic programming is that symbols are written in place of actual machine operation codes and machine addresses. After the entire program has been written in the symbolic language, the symbols are translated into actual machine instructions and addresses by an assembly program. The program written in symbolic language is called a source program; the assembly program translates the source program into an object program.

Relationship of Problem Definition, Detailed Systems Design and Programming

The three phases of preinstallation effort are closely tied together and each phase can be properly accomplished only with an understanding of the others. Thus effective systems design requires a knowledge of programming, and effective problem definition requires a knowledge of equipment capabilities and systems design. Furthermore, one portion of the installation effort usually cannot be fully completed before proceeding to the next. For example, missing information in the systems design often becomes apparent when the programming effort gets under way. The systems man must keep these considerations in mind and come up with the proper amount of detail required at each phase of the system effort. He must strike a balance between too little detail on one hand and overelaborate or extraneous detail on the other. Similarly a balance must be maintained between too sophisticated an approach, which may lead to delays or a system retrenchment, and too limited an approach, which may not take enough advantage of the potential of the 1440.

Problem Definition

In most instances problem definition will have been substantially completed before the installation planning period. We shall, however, briefly cover some of the considerations involved; a fuller discussion is given in *The Study Organization Plan — Basic Systems Study Guide* (F20-8150).

There are a number of ways in which problem definition studies are conducted, but most include the following:

Source documents. — How does the data enter the system, who enters it, how is it recorded, and for what purpose? In addition, such information must be gathered as the length of the data, method of coding, type of codes, estimated volumes (normal and peak load), and peak times. At the same time, samples of source documents are collected and all items of data clearly identified. The movement of the documents is charted to show what data is added by whom, and by what path the documents travel. The percentage of an individual's time spent in performing each task is reported. Specially handled items are noted as to their differences from normal items, and their volumes.

Control over the accuracy of source recording is also studied carefully. In many cases it is possible to institute additional routines to improve the quality of data entering the system. Such control is easier to maintain than are procedures for correcting erroneous data after it has been entered. A control over the number of documents should be planned to insure that all data enters the system.

Procedures and processing. — What data is processed, by whom, how often and why? In this phase of the systems study the analyst follows the path of the source documents through their processing to determine how the data is manipulated to achieve final results. He notes exceptions and the way they are processed, time necessary for processing, the number of people and the percentage of their time involved, as well as volumes. The system of controls is recorded, and time schedules are observed.

One of the easiest ways to document procedures and processing is through the use of flowcharts. Such charts show each processing step, what is accomplished and by whom; they present the flow of work from the receipt of the source document to final reporting. With such a step-by-step analysis, the task of designing the new system is facilitated. Flowcharts help determine organizational changes dictated by changes in procedures. The flow of data from one application area into another should also be indicated on the flowchart.

Report requirements. — Investigation of report requirements is one of the most important aspects of the systems study. This part of the systems study begins with an investigation of current reports by the analyst, who discusses the report requirements with appropriate individuals. It is important that requirements be thoroughly studied to avoid the possibility of having to go back later and introduce new data into the system.

Existing reports are used as the basis for the discussions. The first step is determining whether any information on current reports is unnecessary. The next step is to determine what additional information would help each recipient do his job better. Another aspect of report study is the time schedule. When is the information most effective in helping management perform its responsibilities? By determining this factor, priority schedules for processing can be established.

For best results the analyst discusses report requirements with management periodically during the study. At the outset of the study it is necessary for the analyst to have an idea of what is required in the way of output from the system. This helps to determine whether the raw data can be made available for reports. In addition, during these discussions management may develop other ideas of what information will help them in directing the business.

File requirements. — Provision is made to determine what document, card or disk files need to be maintained. Some data may be necessary for current inquiry, some for year-end reports and some for audit purposes. Still other data may not have to be retained at all. In each case, what data will be retained, how, and by whom, must be determined.

Systems flowchart. — After the systems study, the next step is to define how the system will operate from start to finish. This takes into account all processing, whether by manual means, unit record equipment or the 1440. A flowchart may be used for this representation. This flowchart will serve as a basis for organizational changes, personnel requirements in the various areas, unit record machine requirements and subsequent block diagramming.

The better the planning at this point, the more marked the results. Since the flowchart may be thought of as the master plan for the organization's data processing, it deserves ingenuity and scrutiny and should be reviewed by the data processing manager for completeness. Subsequent reviews should be made with heads of affected departments for their assurance of completeness and accuracy.

Management Review

At the end of this phase, management should review all that has occurred to date. The data processing manager discusses the results of the procedure survey. Included is the definition of each application area as found in the systems study. The solutions that have been worked out in general form are discussed in order to assure management that their objectives will be met. Advantages and benefits of the solutions are included. Any questions as to interpretation of management objectives should be clarified at this meeting.

Organizational changes necessitated by altered work flow should also be discussed. The data processing manager should previously have prepared recommendations as to personnel and departmental changes necessary to carry out the master plan.

Another aspect of this meeting is the assignment of conversion priorities to the application areas. Two criteria generally used are financial return to the company and the most logical sequence of events leading toward completion of the master plan.

This management review has at least three objectives — approval of the master plan, either as presented or as amended; a preliminary determination of departmental and personnel changes; the establishment of priorities determining the course of action.

Problem Definition Pointers

Acquire a background of different system approaches. If possible, study the system approach of similar companies in the industry. IBM sales representatives and industry representatives are available to supply experience and know-how. Application writeups and other published material are also good sources of ideas. This information should be used as a guide, not copied. Procedures borrowed from other sources, without an understanding of their objectives, may lead to a system approach that has little chance of holding up under detailed study.

Make documentation thorough and concise. Define the major programs and document their main functions. While it is too early to determine the exact composition of input/output and file records, strive to make them representative of the final records both in length and content. This is important for proper system selection and utilization studies. A general systems design is more than an outline. It should contain most of the elements which, when sophisticated by detailed study and design, result in an effective solution.

Be realistic in the systems approach. Consider the programming, conversion and procedural changes required on the part of your organization and weigh the complexity of the systems design against the number and experience of the personnel to be assigned to the project. A workable approach can be sophisticated in time; an overambitious approach is subject to failure.

Develop standards early in the procedure. The early standardization of flowcharting and documentation techniques is important. This is especially true when the major programming effort is being accomplished by two or more people. Often standardization is initially overlooked, requiring extensive redocumentation in later stages when creating job description and program run manuals.

Carefully consider the need for controlled and checked input data. Well designed procedures for error detection and control are a vital factor in the success of any data processing installation.

Detailed Systems Design

Segmenting the Runs

Once an application has been sufficiently defined and the necessary organizational and systems changes approved, the systems analyst can proceed with the detailed systems planning. This requires breaking the job down into a series of processing runs and detailing and documenting these runs. The number of runs and the functions performed on each will depend on such factors as:

The capacity of the particular 1440 system (amount of core storage, number of disk drives available, number of card read punch units, etc.).

Natural functional segments. Most applications have natural segmentation covering such areas as input conversion and editing, sorting, merging, file updating, file maintenance, etc.

Volume of transactions.

Time schedules.

File organization.

Audit trails and control provisions.

Since the above are only some of the many factors to be considered, the sequence and functions of the various runs often have to be modified and refined until the final systems design is completed. The systems analyst thus often starts with a tentative system segmentation, then proceeds to work out further details, as described below, until he optimizes the systems design.

General Systems Flowchart

Each evolving application should be represented by a general systems flowchart (Figure 3) showing all the runs and their interrelationship. Some of the functions of this flowchart are:

- To represent all runs for an application, indicating all input and output.
- To identify each file and report.
- To serve as a guide for overall systems design.
- To aid in determining the complexity of the application.
- To indicate the number of machine runs to be programmed.
- To serve as a basis for estimating 1440 run time.
- To aid in determining the amount of programming effort required.
- To provide the data processing manager and other management with a means of reviewing overall plans for the program.

The general systems flowchart can thus become a skeleton upon which the system is built.

File Organization and Design of File Records

Usually the file organization and file records are partially defined during the general systems study and then further clarified during the detailed systems design. This is a very important part of run definition, since file organization, record length, record arrangement, etc., greatly influence the programming task and the system throughput. These considerations are discussed in detail in the file section and should be carefully examined.

The file record format decided upon should be entered on the 1311 disk storage layout form (X24-1711) or the 1440 storage layout form (X24-6438), or on a form similar to that shown in Figure 4. Note that the illustrative form provides for entering such details as the actual and symbolic field names, their relative location and the size and number of decimals in each field. It can be easily reproduced to provide multiple copies of the records used in a number of runs.

When designing records used on multiple runs, particular care should be taken to insure the presence of all the fields needed for the various uses. The record lengths and characters should also be checked for conformity to the requirements of the pertinent sorting, file organization and utility programs.

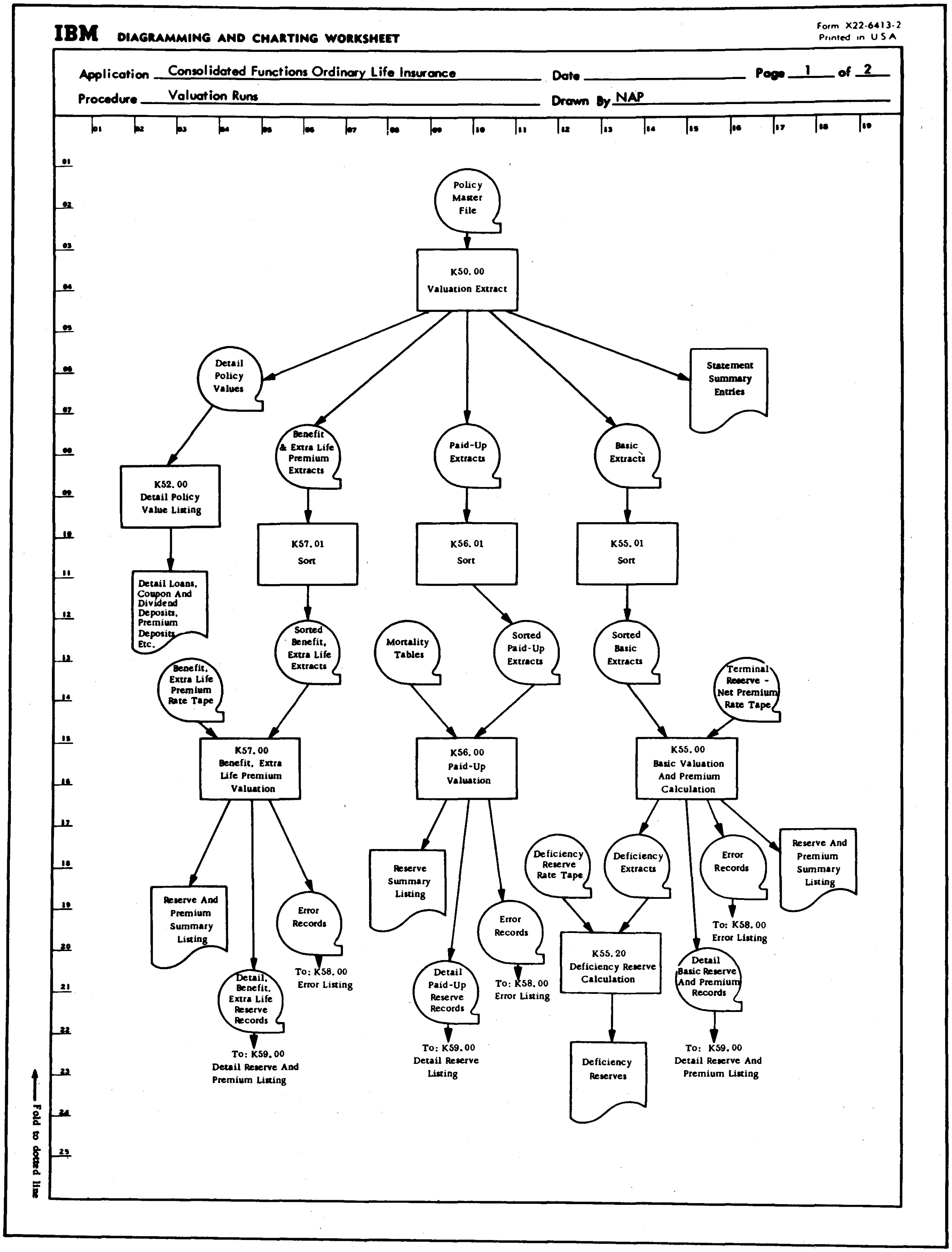
Card Design

In addition to the normal considerations involved in card design (the sequence of information in source documents, the ease of punching, the location of data in other cards, etc.) it is necessary to consider the effect of card layout on throughput.* This is important since the 1442 reads and punches cards one column at a time, and the number and location of the fields therefore affects reading and punching speeds. Some points to note are as given below. Other considerations concerning the effect of card design on processing speed are discussed in the timing section.

- Information should be oriented to the left end of the card to increase the time available for processing. For example, reading columns 60-70 on a Model 1 card read punch requires 123 ms, leaving 87 ms for processing; reading columns 30-40 requires 84 ms, leaving 126 ms for processing.
- To increase throughput, when reading and punching in the same card, the information to be punched should ordinarily be placed at the left end of the card, immediately followed by the fields to be read.
- If punching only, the information can be punched most rapidly if it is positioned at the left end of the card. For example, punching columns 60-70 on a Model 1 card read punch requires 1,085 ms: punching columns 30-40 requires 710 ms.
- When processing several cards and punching the results into one card, the results may be either stored in the disk pack for punching in a subsequent pass, or punched in a trailer card.
- Accumulated to-date information can be stored on disk packs rather than punched into cards. Similarly, punching cards in intermediate stages of a run for later use in report printing can be eliminated since the data can also be stored on the disk pack.

The input and output card layouts decided upon should be entered on the card-punching or verifying instructions form (X24-6299), the multiple layout form (Figure 5), a 1440 storage layout form (X24-6438) or on a form similar to the one in Figure 4.

* This refers to systems using the 1442 Card Read Punch. Card layout does not affect throughput on systems using the 1444 Card Punch. This unit punches cards in parallel (that is, it punches the corresponding 80 punching positions simultaneously) and the number and location of the punching columns therefore does not influence punching speed.


IBM DIAGRAMMING AND CHARTING WORKSHEET

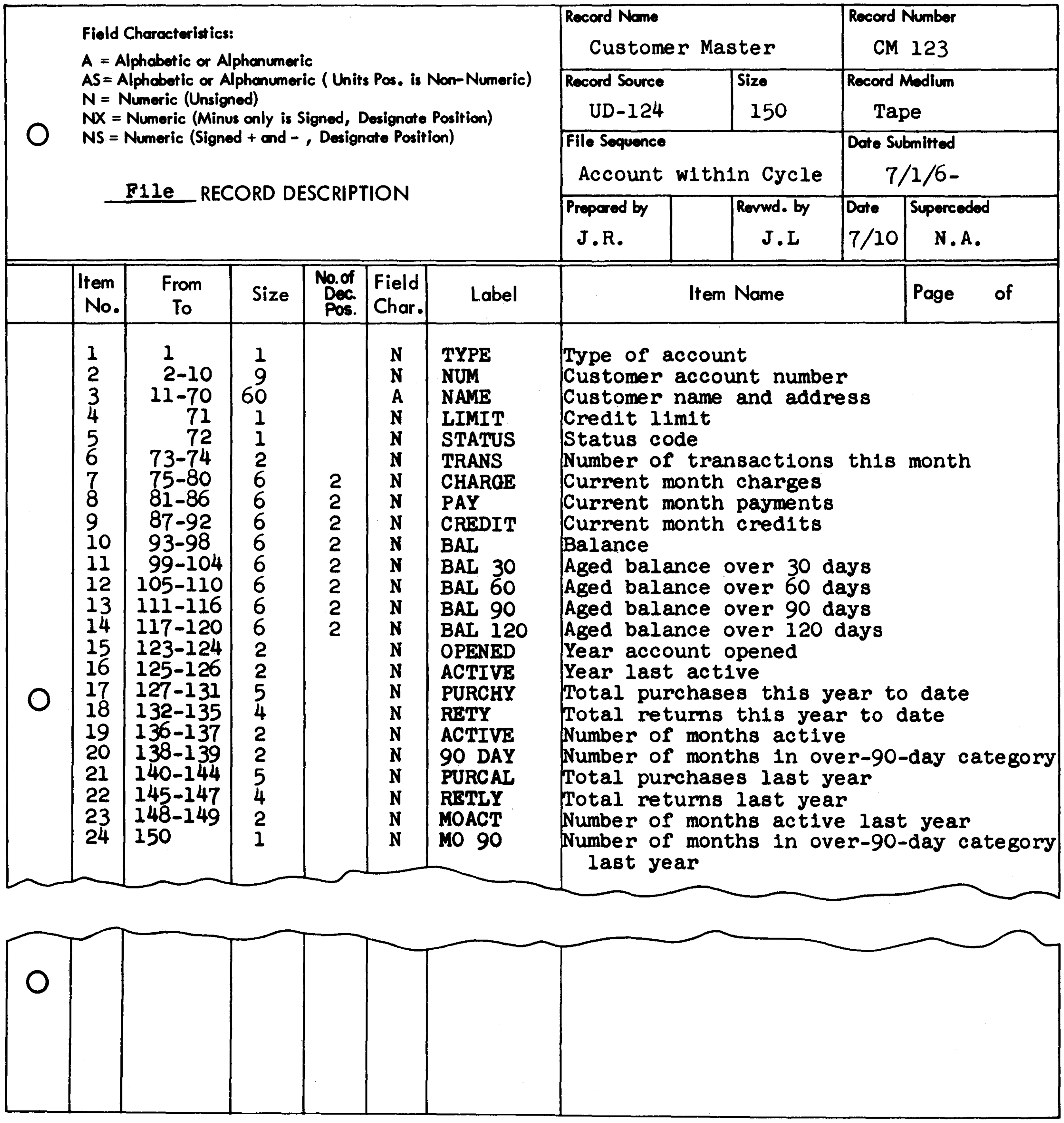
Form X22-6413-2
Printed in U.S.A.

Application Consolidated Functions Ordinary Life Insurance Date ______ Page 1 of 2

Procedure Valuation Runs Drawn By NAP

Policy Master File
K50.00 Valuation Extract
Detail Policy Values
Benefit & Extra Life Premium Extracts
Paid-Up Extracts
Basic Extracts
Statement Summary Entries
K52.00 Detail Policy Value Listing
K57.01 Sort
K56.01 Sort
K55.01 Sort
Detail Loans, Coupon And Dividend Deposits, Premium Deposits Etc.
Sorted Benefit, Extra Life Extracts
Mortality Tables
Sorted Paid-Up Extracts
Sorted Basic Extracts
Benefit, Extra Life Premium Rate Tape
Terminal Reserve - Net Premium Rate Tape
K57.00 Benefit, Extra Life Premium Valuation
K56.00 Paid-Up Valuation
K55.00 Basic Valuation And Premium Calculation
Reserve Summary Listing
Deficiency Reserve Rate Tape
Deficiency Extracts
Error Records
Reserve And Premium Summary Listing
Reserve And Premium Summary Listing
Error Records
Error Records
To: K58.00 Error Listing
K55.20 Deficiency Reserve Calculation
Detail Basic Reserve And Premium Records
To: K58.00 Error Listing
Detail, Benefit, Extra Life Reserve Records
Detail Paid-Up Reserve Records
To: K58.00 Error Listing
To: K59.00 Detail Reserve And Premium Listing
To: K59.00 Detail Reserve Listing
Deficiency Reserves
To: K59.00 Detail Reserve And Premium Listing
Fold to dotted line

Figure 3. General Systems Flowchart



Field Characteristics:

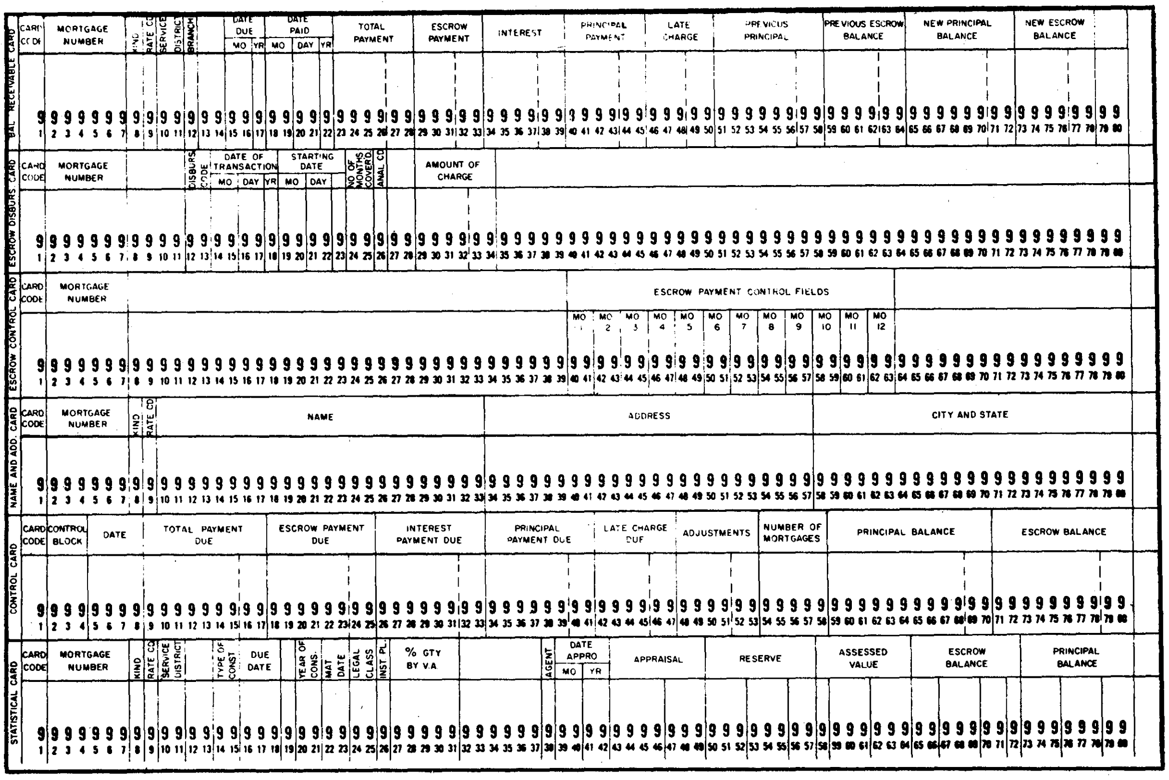
A = Alphabetic or Alphanumeric
AS = Alphabetic or Alphanumeric (Units Pos. is Non-Numeric)
N = Numeric (Unsigned)
NX = Numeric (Minus only is Signed, Designate Position)
NS = Numeric (Signed + and - , Designate Position)

File RECORD DESCRIPTION

Record Name: Customer Master		Record Number: CM 123	
Record Source: UD-124	Size: 150	Record Medium: Tape	
File Sequence: Account within Cycle		Date Submitted: 7/1/6-	
Prepared by: J.R.	Revwd. by: J.L	Date: 7/10	Superceded: N.A.

Item No.	From To	Size	No. of Dec. Pos.	Field Char.	Label	Item Name	Page of
1	1	1		N	TYPE	Type of account	
2	2-10	9		N	NUM	Customer account number	
3	11-70	60		A	NAME	Customer name and address	
4	71	1		N	LIMIT	Credit limit	
5	72	1		N	STATUS	Status code	
6	73-74	2		N	TRANS	Number of transactions this month	
7	75-80	6	2	N	CHARGE	Current month charges	
8	81-86	6	2	N	PAY	Current month payments	
9	87-92	6	2	N	CREDIT	Current month credits	
10	93-98	6	2	N	BAL	Balance	
11	99-104	6	2	N	BAL 30	Aged balance over 30 days	
12	105-110	6	2	N	BAL 60	Aged balance over 60 days	
13	111-116	6	2	N	BAL 90	Aged balance over 90 days	
14	117-120	6	2	N	BAL 120	Aged balance over 120 days	
15	123-124	2		N	OPENED	Year account opened	
16	125-126	2		N	ACTIVE	Year last active	
17	127-131	5		N	PURCHY	Total purchases this year to date	
18	132-135	4		N	RETY	Total returns this year to date	
19	136-137	2		N	ACTIVE	Number of months active	
20	138-139	2		N	90 DAY	Number of months in over-90-day category	
21	140-144	5		N	PURCAL	Total purchases last year	
22	145-147	4		N	RETLY	Total returns last year	
23	148-149	2		N	MOACT	Number of months active last year	
24	150	1		N	MO 90	Number of months in over-90-day category last year	

Figure 4. Illustrative Record Layout Form



BAL. RECEIVABLE CARD: CARD CODE | MORTGAGE NUMBER | KIND | RATE CD | SERVICE | DISTRICT | BRANCH | DATE DUE (MO, YR) | DATE PAID (MO, DAY, YR) | TOTAL PAYMENT | ESCROW PAYMENT | INTEREST | PRINCIPAL PAYMENT | LATE CHARGE | PREVIOUS PRINCIPAL | PREVIOUS ESCROW BALANCE | NEW PRINCIPAL BALANCE | NEW ESCROW BALANCE

ESCROW DISBURS. CARD: CARD CODE | MORTGAGE NUMBER | DISBURS. CODE | DATE OF TRANSACTION (MO, DAY, YR) | STARTING DATE (MO, DAY) | NO OF MONTHS COVER'D | ANAL CD | AMOUNT OF CHARGE

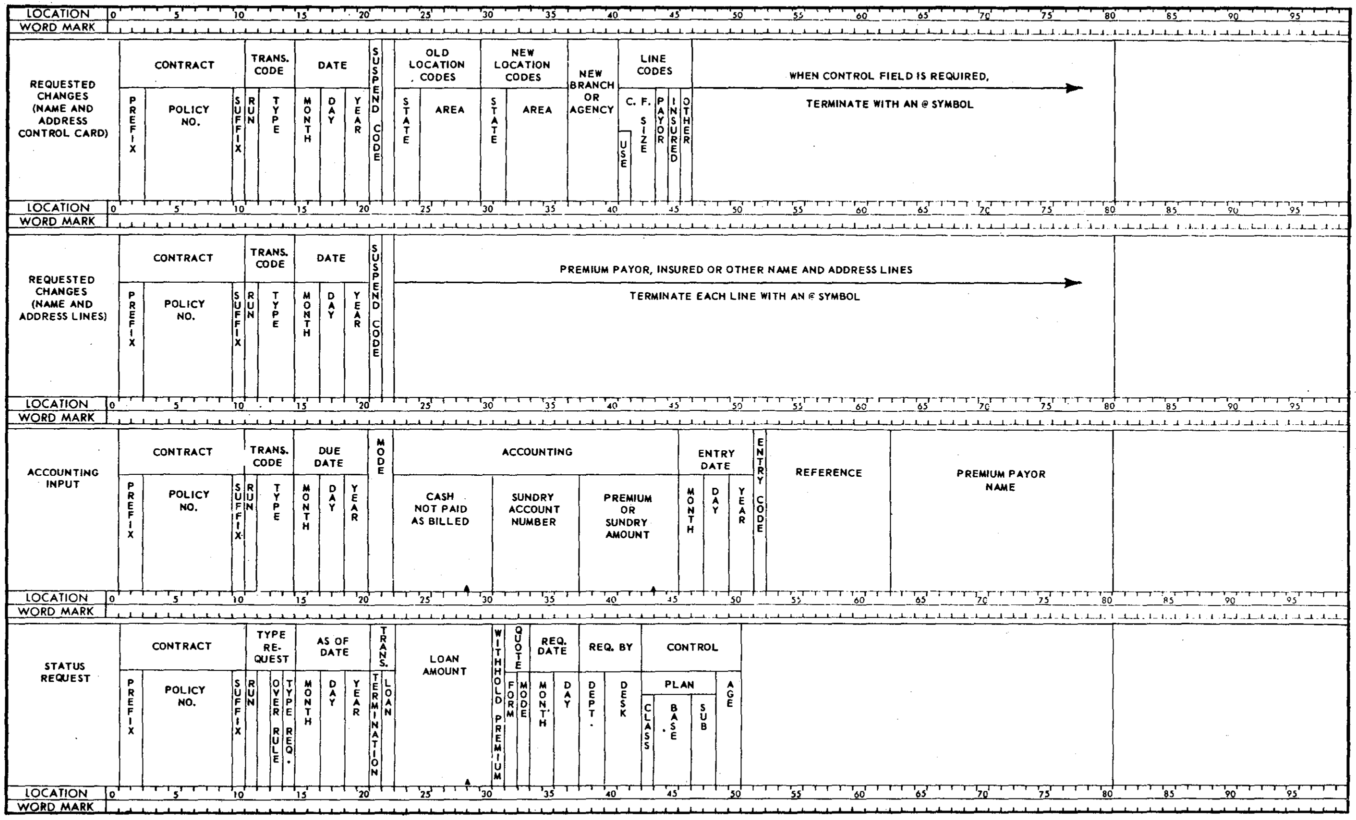
ESCROW CONTROL CARD: CARD CODE | MORTGAGE NUMBER | ESCROW PAYMENT CONTROL FIELDS (MO 1, MO 2, MO 3, MO 4, MO 5, MO 6, MO 7, MO 8, MO 9, MO 10, MO 11, MO 12)

NAME AND ADD. CARD: CARD CODE | MORTGAGE NUMBER | KIND | RATE CD | NAME | ADDRESS | CITY AND STATE

CONTROL CARD: CARD CODE | CONTROL BLOCK | DATE | TOTAL PAYMENT DUE | ESCROW PAYMENT DUE | INTEREST PAYMENT DUE | PRINCIPAL PAYMENT DUE | LATE CHARGE DUE | ADJUSTMENTS | NUMBER OF MORTGAGES | PRINCIPAL BALANCE | ESCROW BALANCE

STATISTICAL CARD: CARD CODE | MORTGAGE NUMBER | KIND | RATE CD | SERVICE DISTRICT | TYPE OF CONST | DUE DATE | YEAR OF CONS. | MAT DATE | LEGAL CLASS | INST. PL. | % GTY BY V.A. | AGENT | DATE APPRO (MO, YR) | APPRAISAL | RESERVE | ASSESSED VALUE | ESCROW BALANCE | PRINCIPAL BALANCE

Figure 5. Multiple Layout Form



LOCATION 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95
WORD MARK

REQUESTED CHANGES (NAME AND ADDRESS CONTROL CARD): CONTRACT (PREFIX, POLICY NO., SUFFIX) | TRANS. CODE (RUN, TYPE) | DATE (MONTH, DAY, YEAR) | SUSPEND CODE | OLD LOCATION CODES (STATE, AREA) | NEW LOCATION CODES (STATE, AREA) | NEW BRANCH OR AGENCY | LINE CODES (C. F. USE, SIZE, PAYOR, INSURED, OTHER) | WHEN CONTROL FIELD IS REQUIRED, TERMINATE WITH AN @ SYMBOL

REQUESTED CHANGES (NAME AND ADDRESS LINES): CONTRACT (PREFIX, POLICY NO., SUFFIX) | TRANS. CODE (RUN, TYPE) | DATE (MONTH, DAY, YEAR) | SUSPEND CODE | PREMIUM PAYOR, INSURED OR OTHER NAME AND ADDRESS LINES — TERMINATE EACH LINE WITH AN @ SYMBOL

ACCOUNTING INPUT: CONTRACT (PREFIX, POLICY NO., SUFFIX) | TRANS. CODE (RUN, TYPE) | DUE DATE (MONTH, DAY, YEAR) | MODE | ACCOUNTING (CASH NOT PAID AS BILLED, SUNDRY ACCOUNT NUMBER, PREMIUM OR SUNDRY AMOUNT) | ENTRY DATE (MONTH, DAY, YEAR) | ENTRY CODE | REFERENCE | PREMIUM PAYOR NAME

STATUS REQUEST: CONTRACT (PREFIX, POLICY NO., SUFFIX) | TYPE REQUEST (RUN, OVER RULE, TYPE REQ.) | AS OF DATE (MONTH, DAY, YEAR) | TRANS. (TERMINATION, LOAN) | LOAN AMOUNT | WITHHOLD PREMIUM | QUOTE (FORM, MODE) | REQ. DATE (MONTH, DAY) | REQ. BY (DEPT., DESK) | CONTROL (PLAN: CLASS, BASE, SUB; AGE)

Figure 6. Storage Layout Form

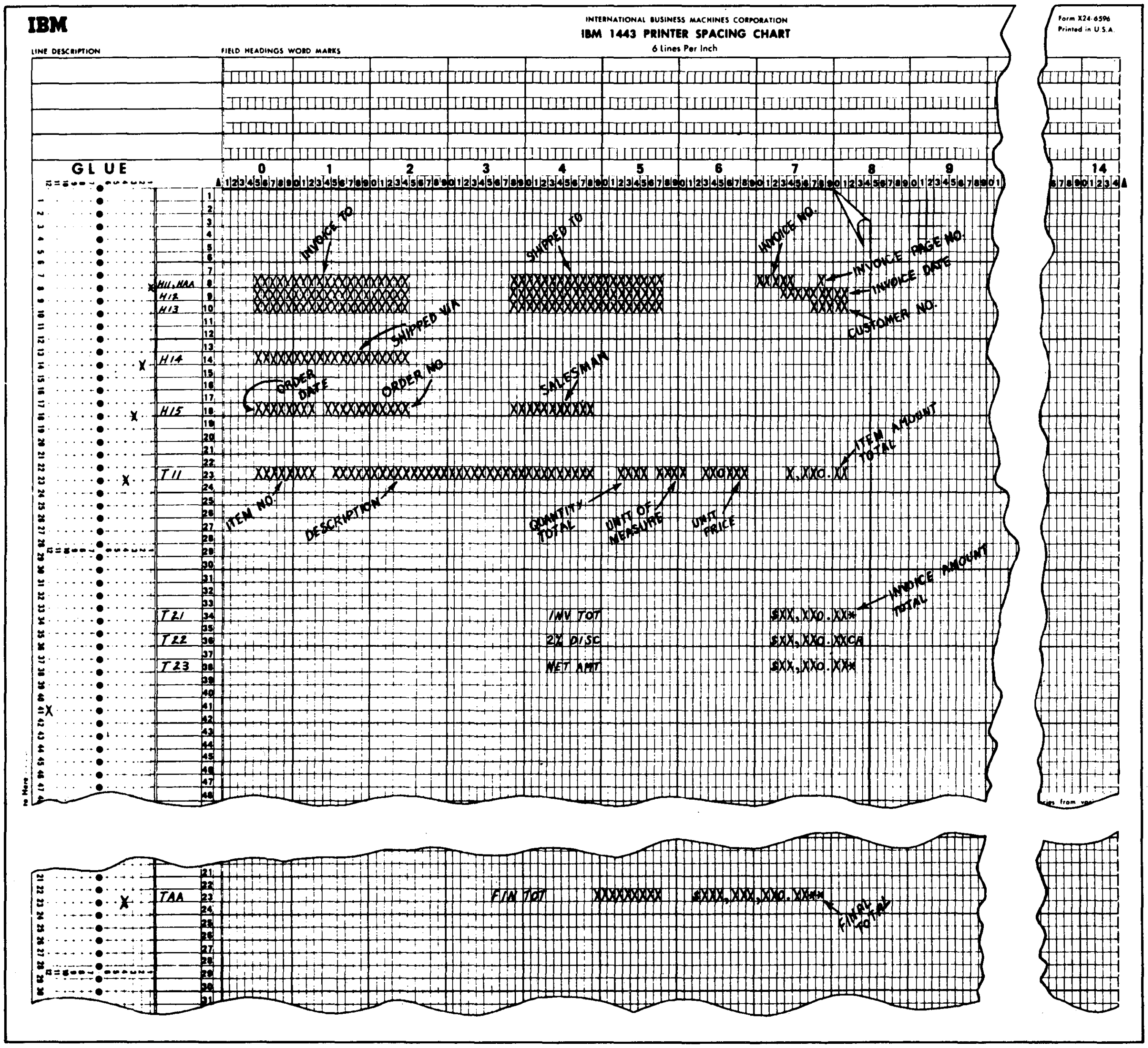


Figure 7. Illustrative Spacing Chart

Form Design

The publication IBM *1443 Forms Specifications* (A24-3041) contains information on form design and the considerations for handling forms used with the 1443. The following general considerations are in addition to the ones covered in the above manual:

- For many reports it may be possible to eliminate preprinted forms, since the form heading can be printed by the computer. The heading information can be stored in the disk pack or can be entered by cards at the start of the run.
- Consider converting as many jobs as possible to a standard utility paper. By standardizing on form widths, length, and number of copies, printer setup time will be minimized since fewer form changes will be needed.
- Consider the possibility of printing side-by-side reports to save printing time and paper costs.

Once the form design is set, the 1443 Spacing Chart (X24-6596) or any other standard spacing chart should be used to indicate the name of each field and its location on the form (Figure 7). Each type of line to be printed should be shown. Where a previously available preprinted form is used, it can be pasted to a spacing chart or annotated so that the programmer can easily determine the print positions for each field.

TYPE OF DOCUMENT	Transaction Code		"X" Control Punch	Daily Stock Balance							Requirements vs. Issues						Purchase Order Status	
	Card Column 34	Card Column 35	Card Column	Transaction	Pending	Stock Balance	On Order	Miscellaneous Requirement	Contract Requirement	Available Excess	Contract Requirement	Contract Issue	Contract Balance	Miscellaneous Requirement	Miscellaneous Issues	Miscellaneous Balance	Purchase Order Quantity	Accumulated Quantity
Contract Requirements - Increase	1	A	4 1	+					+	-	+		+					
Contract Requirements - Decrease	2	A	4 0	-					-	+	-		-					
Miscellaneous Requirements - Increase	1	B	4 3	+				+		-				+		+		
Miscellaneous Requirements - Decrease	2	B	4 2	-				-		+				-		-		
Purchase Order	1	C	6 2	+			+			+							+	-
Purchase Order - Reversing Code	2	C	6 3	-			-			-							-	+
Change Order - Increase	1	D	6 2	+			+			+							+	-
Change Order - Decrease	2	D	6 3	-			-			-							-	+
Goods Received - Traveler	1	E	6 6	+		+	-											
Goods Received - Traveler - Reversing Code	2	E	6 7	-		-	+											
Goods Received - Advance	1	F	6 6															+
Goods Received - Advance - Reversing Code	2	F	6 7														+	
Goods Received - Rejection	2	G	1 8	-	+	-												
Goods Received - Rejection - Reversing Code	1	G	1 5	+	-	+												
Return Ticket	2	H	4 4	-	-					-								
Return Ticket - Reversing Code	1	H	4 5	+	+					+								
Stocking Memorandum	1	J	1 5	+	-	+												
Stocking Memorandum - Reversing Code	2	J	1 8	-	+	-												
Store Return - Planner	1	K	1 9	+		+			+			+	+					
Store Return - Planner - Reversing Code	2	K	3 9	-		-			-			-	-					
Store Return - Shop	1	L	1 6	+		+				+								
Stock Substitution - Contract	[illegible]	R	[illegible]	-					-	+	-		-					
Stock Substitution - Contract - Reversing Code	1	R	4 1	+					+	-	+		+					
Stock Substitution - Miscellaneous	2	S	4 2	-				-		+					-	-		
Stock Substitution Misc. - Reversing Code	1	S	4 3	+				+		-					+	+		
Scrap Parts	2	T	4 4	-	-					-								
Scrap Parts - Reversing Code	1	T	4 5	+	+					+								
Filled from Surplus - Contract	2	U	4 0	-					-	+	-		-					
Filled from Surplus - Contract - Reversing Code	1	U	3 5	+					+	-	+		+					
Filled from Surplus - Miscellaneous	2	V	4 2	-				-		+				-		-		
Filled from Surplus - Misc. - Reversing Code	1	V	4 3	+				+		-				+		+		

Figure 8. Illustrative Decision Table

Processing Description

While the forms described previously (input, output and file layouts, flowcharts, etc.) indicate much of the processing required, other tools are often necessary to present and clarify the processing elements. These include narrative descriptions, decision tables and block diagrams. Each is briefly described below:

Narrative description. — Narratives are frequently used to give an overall description of the system and its purpose and scope. They are also used to describe internal computer processing as well as external clerical and auxiliary machine operations.

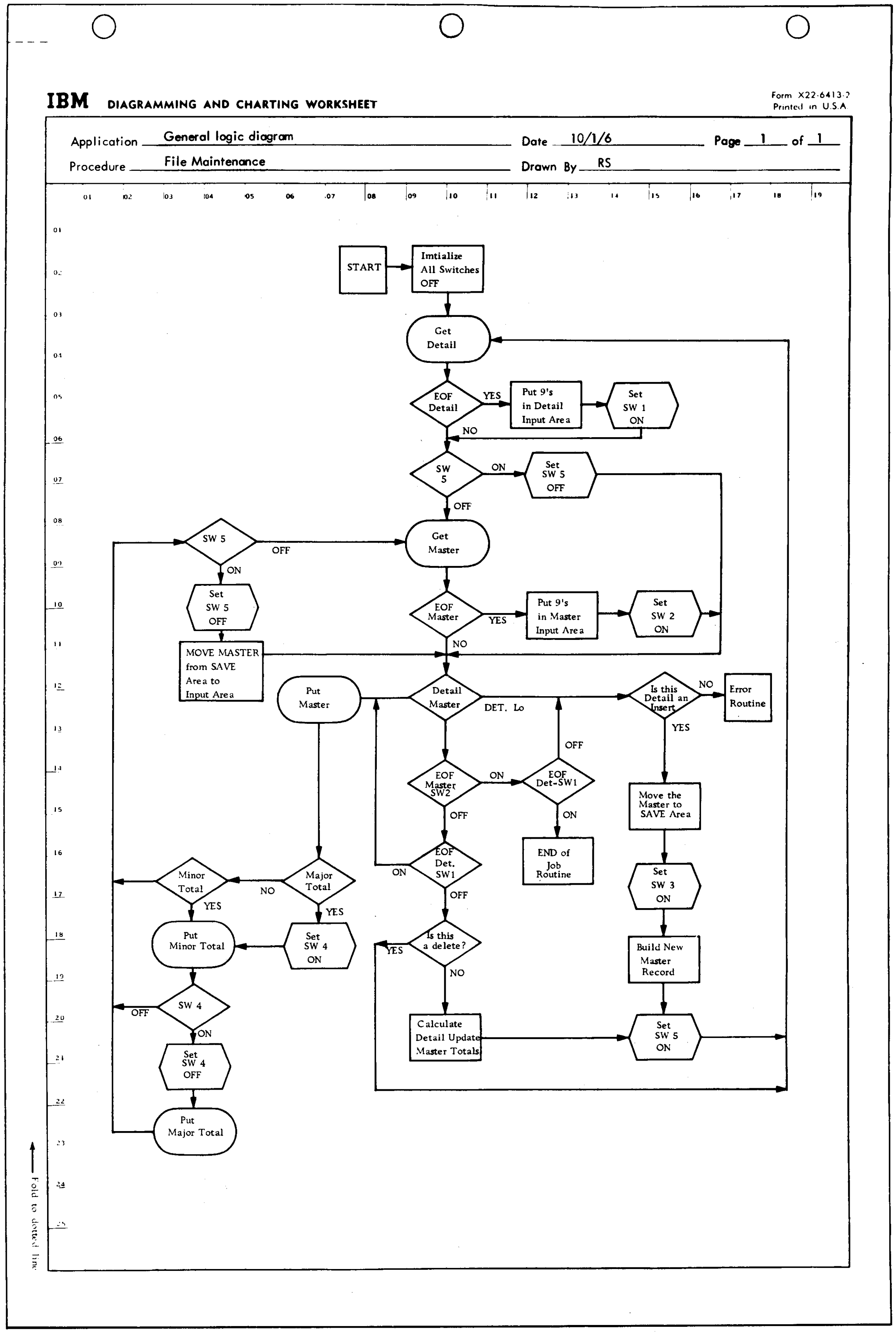
Decision tables. — Decision tables are a tabular method of representing various conditions and alternatives and are a means of bringing together and presenting the information needed to express decision logic in a way that is easy to visualize and follow. By showing alternate courses of action under various combinations of conditions, a decision table helps the analyst to think through a problem and its solution. He is also encouraged to reduce the documentation to its simplest form.

Decision tables can be used effectively for systems analysis, procedure design and documentation. Their use expedites and simplifies the time-consuming functions of problem definition and systems analysis. For example, in an inventory application many types of transactions must be processed against the inventory balance record. A decision table, such as the one shown in Figure 8, listing all the transactions with their effect on each of the fields of the inventory record, is a good method of analyzing and presenting these variables.

The use of decision tables is described in detail in *Decision Tables — A Systems Analysis and Documentation Technique* (F20-8102).

General logic block diagrams. — A general logic block diagram can be used by the systems analyst to provide the programmer with a base on which to develop the program. With this type of diagram (Figure 9) the analyst may present such information as:

- The basic logic for deletions and/or insertions.
- The collating logic for input cards, input disk files and disk master records.
- The points of disk, card and printer output.
- The logic for correct end-of-job procedure.


IBM DIAGRAMMING AND CHARTING WORKSHEET

Form X22-6413-? Printed in U.S.A.

Application	General logic diagram	Date 10/1/6	Page 1 of 1
Procedure	File Maintenance	Drawn By RS	

Figure 9. Illustrative General Logic Block Diagram

Controls

Proper input, output and processing controls are of the greatest importance to any data processing installation and must be thoroughly provided for in systems design. This subject is covered in detail in the following manuals:

Document and Accounting Controls (C20-8060)

The Auditor Encounters Electronic Data Processing (F20-8057)

In-Line Electronic Accounting, Internal Control and Audit Trail (F20-2019)

Input Data Validation and Systems Controls in an Automatic Data Processing System, IBM Systems Research Institute, 787 United Nations Plaza, New York 17, New York.

Detailed Systems Design Pointers

Precisely define an application before programming begins. Unless all pertinent details are covered, applications which have theoretically been programmed adequately may face many unforeseen difficulties when subject to full production.

Break each application down to its smallest elements. How many exceptions are there? What are they? What causes them? What is the frequency of exceptions? How are they handled? Questions like these can be resolved only by detailed study.

Identify all transactions and their cyclic volume fluctuations.

Keep operating personnel aware of development progress. It is also important to reach an understanding with the supervisory heads of the various departments as to how procedural changes dictated by systems design will affect their particular areas of responsibility.

Before programming starts, firm up all elements of the program. Following are some of the areas requiring finalization before major programming commences:

1. Master file records. – Define as to content and length. All coding requirements within these records should be established and a final table of codes developed.
2. Card formats. – Input and output card formats should be fixed.
3. Printed reports. – While the actual artwork and final design of printed reports need not be accomplished, the makeup of all heading, detail and total lines should be decided upon.
4. Audit control. – Effective audit and control techniques should be developed to meet the company's and the auditor's requirements.
5. Exceptions. – The handling of all exceptions should be formalized. Decide at what points in what programs the exceptions have to be handled, and the method of treating such exceptions.
6. Common labels. – A table of common symbolic labels should be developed, defining all important elements of data. This is most helpful in file applications where different programs are to be written using common master files.
7. Documentation. – Document all input, output and processing objectives. This point cannot be overstressed. Too often important considerations are forgotten if not properly documented.

Documentation for Programmers

The systems analyst should turn over the proper amount of documentation and systems detail to enable the programmer to proceed with a minimum of questions. Even when one person does the programming and systems work, effective documentation at this point is important for control, checking and clarification purposes. The following documentation or its equivalent is suggested at the junction between systems design and programming. Actually most of the items listed are necessary programming tools. Standard card, record and storage layouts, planning charts, coding sheets, etc., should be used whenever possible. A list of these forms can be found under *Installation Supplies* in the 1440 Systems Reference Library bibliography.

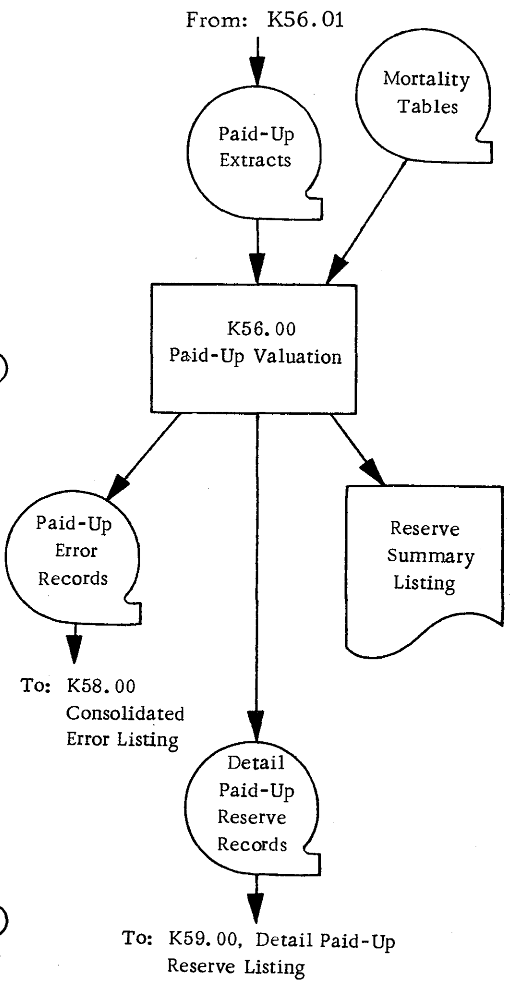
For each application:

- *General system flowchart* showing all the programs required for a specific application.
- *A list of the computer runs* showing estimated timing and core storage requirements.
- *A brief overall narrative* describing the system.

For each run:

- *Input/output flow diagram* (Figure 10) showing in detail all the inputs and outputs to a particular run and a brief description of the internal processing.
- *Detailed file layouts* showing the information to be maintained on the disk files.
- *Card input/output layouts* showing the information in all input and output cards.
- *Report layouts* showing the exact location of all printed data.
- *Run processing description* including any details concerning the run processing not covered above.

RUN INPUT/OUTPUT FLOW DIAGRAM	Run Name Paid-Up Valuation				Run Number K 56.00
	Duration 30-60 Min				
	Prepared By R T				Date Submitted 7/1/6
	Appvd By LS	Date 7/6	Revwd By	Date	Supercedes N.A.



Input

Mortality Tables.
Policies Paid-Up by Conversion to Reduced Paid-Up or Extended Insurance.
Paid-Up Additions.

Processing

Compute Mean Reserves by Extract from Cash Values for Policies on Reduced Paid-Up or Extended Insurance.
Compute Mean Reserves from Mortality Tables for Paid-Up Additions.
Accumulate Mean Reserves by Cell and Standard.

Tape Output

Paid-Up Extract Errors.
Detail Records (for Each Extract) of Reserves.

Print

Reserve Summary Listing by Valuation Cell.
Reserve Totals by Valuation Standard

Figure 10. Illustrative Input/Output Flow Diagram

File Organization and File Techniques

File Concepts

File organization (the way the various records are stored in disk storage) and the manner in which disk records are retrieved for processing greatly affect systems throughput and programming effort. Choosing the best file organization and retrieval technique for a particular application is thus a key systems design function and requires consideration of many factors. Among these are:

- The number of files stored on the disk packs and the size and number of records in each file.
- The number of additions and deletions to the disk files.
- The ratio of the number of transactions processed to the number of records in the master files (the activity ratio).
- The size of the control fields of the records.
- The core storage available for the record retrieval portion of the processing program.
- The functions of the IBM disk file organization and input/output control system programming packages.
- The number of file drives on the system.
- The number and type of input/output units on the system.

Discussion of the major file organization and retrieval techniques will be preceded by a brief section on the physical organization and operation of the 1311 Disk Storage Drive and the type of information that can be stored in disk storage; readers familiar with this material may prefer to skip to "Basic Methods of File Organization and Processing", page 25.

IBM 1311 Disk Storage Drive

As many as five disk storage drives are available to the 1440 Data Processing System. On these drives can be mounted individual disk packs with a storage capacity of 2,000,000 alphameric characters per pack. In this way a total of 10,000,000 characters can be made available to the system at any time.* The disk packs can be removed by the operator and replaced with others, in effect providing the system with virtually unlimited disk storage.

The disk pack weighs approximately ten pounds and consists of six 14-inch disks. Each disk surface, except the upper surface of the top disk and the lower

* In the sector mode. In track record mode (special feature), the track is not divided into sectors and the maximum capacity is 2,980,000 on line characters per disk pack and 14,900,000 per system.

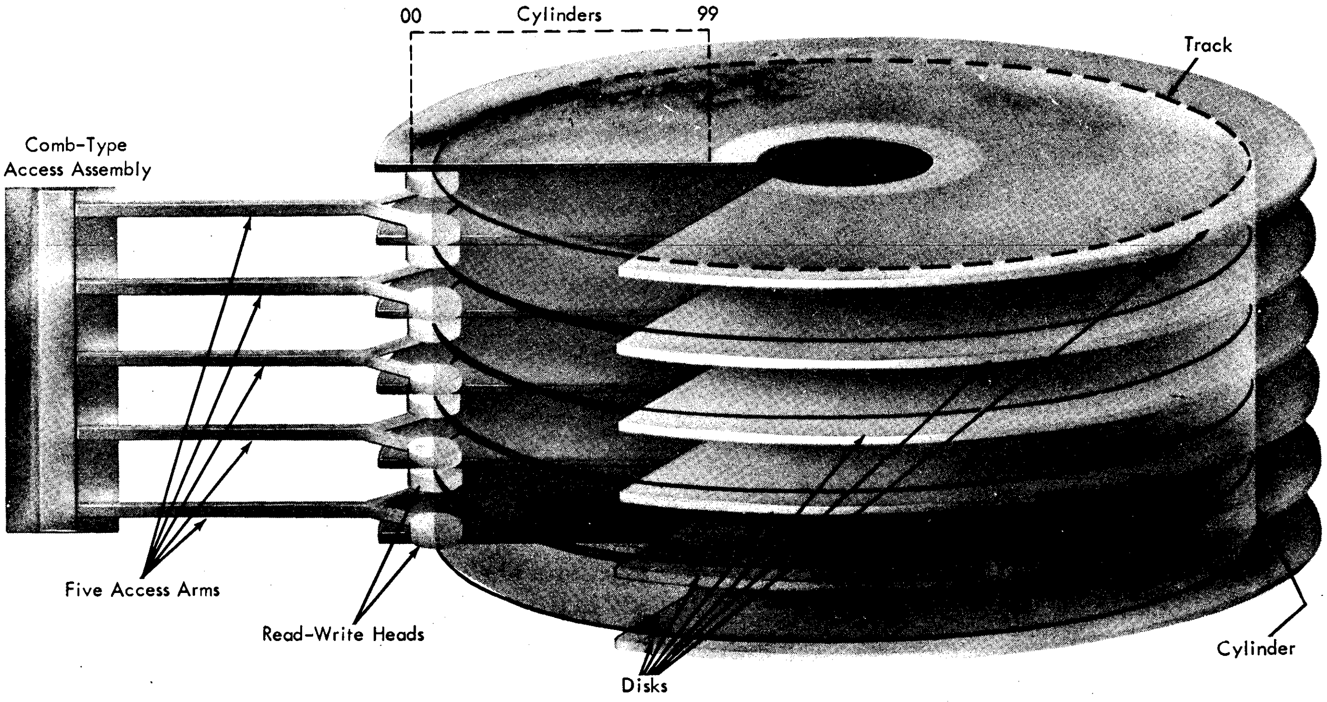


Figure 11. Schematic of 1311 Disk Storage Drive and Disk Pack

surface of the bottom disk, is used as a recording surface. Ten storage surfaces are therefore provided by a disk pack.

Mounted vertically on a disk drive is a comb-type access assembly with five access arms, each containing two read-write heads. The access assembly is used to position the read-write heads relative to the disk surfaces (Figure 11). Each of the access arms, with its two read-write heads, moves between the bottom surface of a disk and the top surface of the next lower disk, and is capable of reading or writing data on both of these surfaces.

As the disk revolves, the read-write heads cover a circular path on the surface of the disk – this is called a track. The ten tracks that are exposed simultaneously to the ten read-write heads as the disk pack revolves are referred to as a cylinder. A cylinder, in this sense, is a quantitative concept rather than a physical component. It may be considered to be a three-dimensional entity comprising a particular track on each of the ten disk surfaces. There are 100 cylinders, numbered 00 to 99, in a disk pack. The read-write heads can be positioned at any one of these 100 cylinders.

Each track is divided into 20 equal-size sections known as sectors. Every sector contains a preassigned six-digit address and provides storage for either 90 characters of data with word marks or 100 characters of data without word marks (Figure 12).

All sectors are addressable. Therefore a disk pack provides 20,000 addresses that may be accessed for reading or writing of data. This figure can be arrived at in the following way:

20 addresses per track	×	10 tracks per cylinder	=	200 addresses per cylinder
200 addresses per cylinder	×	100 cylinders	=	20,000 addresses per disk pack

Perhaps the cylinder concept can best be understood by visualizing 100 cylinders arranged concentrically, with each of the cylinders successively smaller in diameter to permit placement of one inside the other, as shown in Figure 13. Imagine the cylinders as turning and standing on end. Around the circumference of each cylinder are ten tracks for the magnetic recording of data (Figure 14). To gain access to the data on any particular cylinder, the access arms must be first directed by the program to move "through" the cylinders until the desired cylinder is located, at which time the access arms stop. Therefore, all data recorded on the ten tracks on any cylinder is available at one setting of the access mechanism.

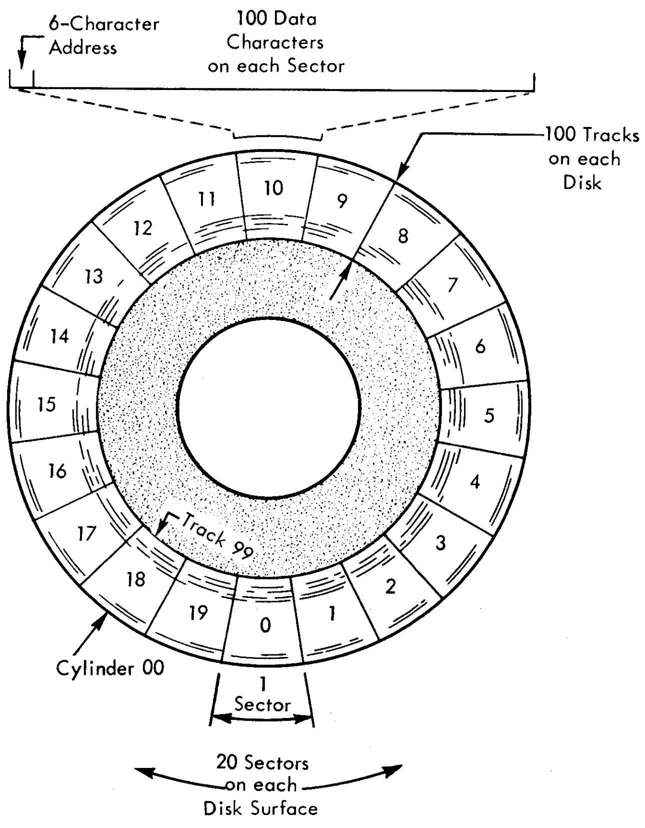


Figure 12. Arrangement of Data on Disk Surface

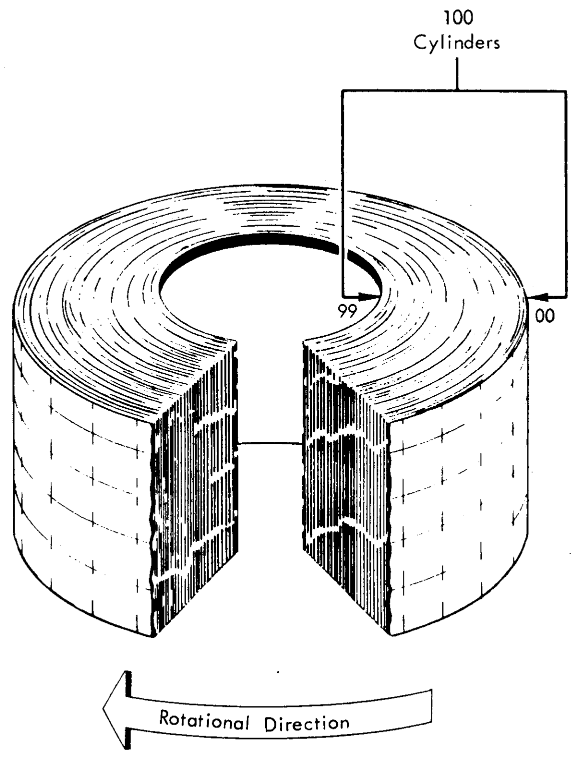


Figure 13. Visualization of Cylinder Concept

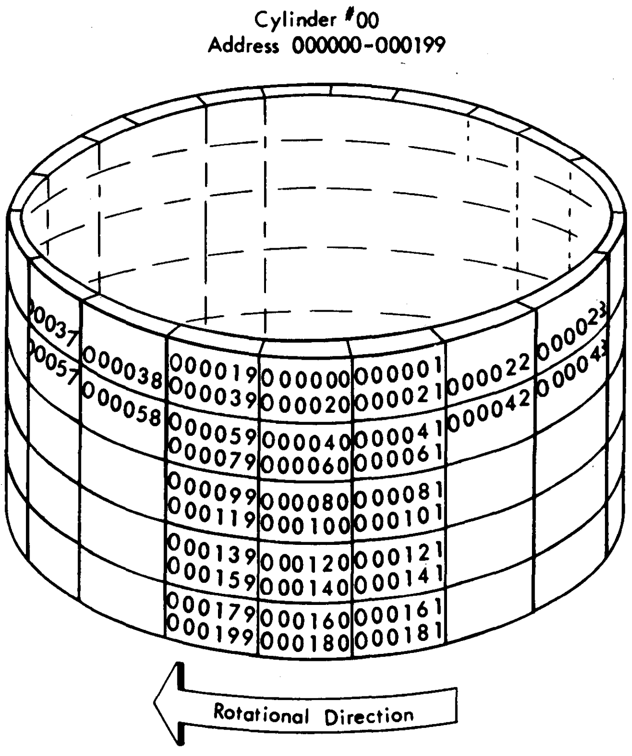


Figure 14. Sector Addresses on Cylinder 00

1311 Disk Storage Operations

Disk storage instructions consist of five basic operations:

Seek Disk
Read Disk
Write Disk
Write Disk Check
Scan Disk (special feature)

A general description of the various disk operations follows.

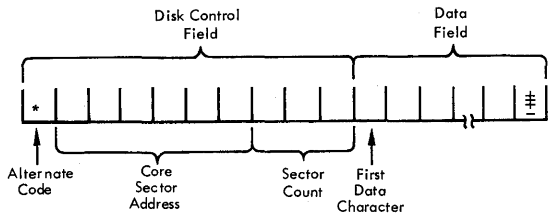
SEEK DISK OPERATION

The Seek Disk operation consists of an instruction that directs the access mechanism and the associated read-write heads to the proper cylinder on the disk pack. Data on the disk records is not acted upon by this instruction. The seek instruction merely positions the access arms at the proper cylinder.

The operand of the Seek Disk instruction, as well as the operand of the four other disk instructions, specifies the core storage address of the disk control field (described under "Read Disk Operation"). The disk control field, in turn, indicates the address of the record to be processed.

READ DISK OPERATION

The Read Disk operation transfers data from disk storage to the core storage area of the processing unit. The data from disk storage is read into core in an area immediately following a 10-digit disk control field. The disk control field and the data field are set up as shown below:



The *core sector address* portion of the disk control field contains the disk address of the data to be read or written. If more than one sector is read or written, the address of the first sector is specified.

The *sector count* portion of the disk control field specifies the number of disk sectors that are to be read or written by the instruction. The number of positions in core storage reserved for the *data field* must be large enough to contain all the data read from the disk. The group mark with a work mark at the end of the data field defines the total length of the area reserved for data.

In the normal operation of read and write instructions, the sector count in the disk control field is decremented by one just before a sector is read from or written into disk storage. After a sector is transferred, the sector address in the disk control field is incremented by one, except for the last sector transferred. The operation is stopped when, after transferring a sector, the sector count is detected as having reached zero. (Incrementing and decrementing are automatic functions of the processing unit.)

The disk address of the data normally indicates which storage drive is to be used by the system as shown in the examples below.

Address Block	*Selects Disk Drive*
000000-019999	0
020000-039999	1
040000-059999	2
060000-079999	3
080000-099999	4

If the disk addresses within a disk pack do not correspond to the range of numbers for the disk storage drive on which it is located, a disk *alternate code* is used.

This code provides the flexibility to operate upon any block of disk addresses, regardless of which disk drive the pack is located on, and, when the occasion requires (for example, in a disk copying run) to have more than one disk pack containing the same disk addresses online at the same time. The selection of a particular disk storage drive is made by using the following alternate codes:

Alternate Code	*Selects Disk Drive*
0	0
2	1
4	2
6	3
8	4

For example, if address block 060000-079999 were located on disk drive 4 (instead of disk drive 3), the alternate code in the disk control field would be 8.

When this flexibility is not required, the drive is selected directly from the core sector address and an asterisk (*) is placed in the alternate code portion of the disk control field.

WRITE DISK OPERATION

The Write Disk operation transfers data from core storage to disk storage. An area in core storage is set aside for the disk control field, which specifies the disk storage drive to be used, the disk address where the data is to be written and the number of sectors to be written.

The data to be transferred to disk storage is located immediately to the right of the disk control field. If, for example, 100-character records are to be written into disk storage, 111 core storage positions will be reserved: ten positions for the disk control field, 100 for the data, and one for a group mark – word mark to indicate the end of the data.

WRITE DISK CHECK OPERATION

A Write Disk Check instruction must be the next file instruction following a Write Disk instruction. The Write Disk Check operation causes the data just written in disk storage to be read and compared with the original source data in core storage. When the disk data does not compare, bit-by-bit and character-by-character, with the core storage data from which it was written, a Disk Error indicator is set.

SCAN DISK OPERATION (SPECIAL FEATURE)

This feature provides the system with the ability to make a rapid search of a disk pack for a record with a specific code or condition without reading the entire record into code storage. (The code or condition is referred to as the search argument.) Only one seek and one scan disk instruction are required to cause the program to search through an entire cylinder (200 sectors). The scan can be made to compare the search argument (located in core) with the disk data, on the basis of the argument being low-or-equal to the data, equal to the data, or high-or-equal to the data. The program can be directed to scan all records in a pack, or all records in a cylinder, or any specific number of records.

The disk control field initially is established to show the sector address where the scan is to begin and the maximum number of sectors to be scanned. The search argument area adjacent to the disk control field is established in the format of the sectors being searched. The search argument located in this area must correspond to the positions in which the information to be compared (the identifier) is located in the *disk record*. Fields in the search argument area corresponding to those in the disk record that are *not* to be compared in the scan, must be filled with Scan Skip signals ($). A group mark-word mark must be placed after the last position of the argument. The last character of a sector cannot be included in the scan argument.

Each sector is examined in sequence until the search argument is satisfied or the sector count reaches zero. The core sector address segment in the disk control field used with the scan instruction will then contain the address of the sector satisfying the scan condition. The address is then used in the disk control field of the Read Disk instruction in order to read the sector into core storage.

The scan operation can be performed only in the sector mode of operation.

FOUR MODES OF READ OR WRITE DISK OPERATIONS

Read and write instructions have four modes:

Sector mode
Track record mode
Sector count overlay mode
Address mode

SECTOR MODE

The Sector mode is the normal mode of operation. Read and Write operations in the Sector mode transfer data to or from the disk, but do not transfer disk sector addresses. The number of sectors to be handled within one operation is designated by the sector count part of the disk control field.

Each sector is transferred only upon absolute comparison of its address with the address in the disk control field. The core sector address in the disk control field is automatically increased by one for each sector transferred except the last. Similarly, the sector count is reduced by one and indicates by a 000 setting that the required operation has been completed.

If a group mark with a word mark is sensed in the data field before the sector count reaches zero, the operation is terminated. Likewise, if the sector count reaches zero before the group mark with a word mark is sensed, the operation is terminated. Either of these events will result in the Wrong Length Record indicator being turned on.

The Wrong Length Record indicator is tested by a Branch If Indicator On instruction. It is reset automatically by the next disk operation.

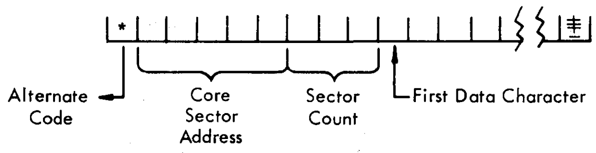
TRACK RECORD MODE (SPECIAL FEATURE)

Normally, one 100-character record is written in or read from each of the 20 sectors of a track. The Track Record mode makes it possible to read or write one record in place of the 20 sectors of a track. The record is referenced by one six-position address and contains 2,980 positions of data.

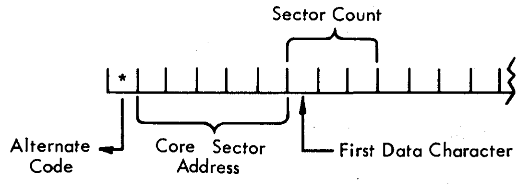
SECTOR COUNT OVERLAY MODE

The sector count overlay mode is designed to facilitate the processing of variable length records. In this mode, the first data character transferred from disk storage is stored in the leftmost position of the sector count part of the disk control field *instead* of the position immediately to the right of the disk control field. When a record is stored in disk storage from core storage, the transmission of data starts with the leftmost position of the sector count field. Disk Control Fields for normal read/write operations and for read/write operations that use the Sector Count Overlay mode are shown below.

Normal Read/Write Operation



Sector Count Overlay Mode



Initially, the sector count must be set to a number greater than 001 before a read operation in the Sector Count Overlay mode is executed.

ADDRESS MODE

This mode of operation allows sector addresses recorded in disk storage to be changed. It provides for the transfer of both data and disk sector addresses to and from the file, one complete track at a time. The sector count must be set to 020 (sectors are transferred in blocks of 20).

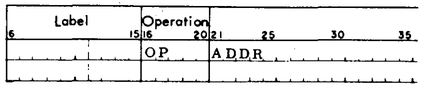
The operation requires that the disk control field contain an address of one of the sectors within the track. This address must be satisfactorily compared with its counterpart in disk storage before the transfer can take place.

READING AND WRITING WITH WORD MARKS

In all reading and writing operations, the data, together with word marks, may be read from and written onto disk storage. When word marks are written on the disk, the data is written in eight-bit BCD coding. The use of eight-bit coding reduces the number of characters that can be stored on a sector from 100 to 90, and reduces the total number of characters on a disk pack from 2,000,000 to 1,800,000. In the Track Record mode, the maximum number of characters on one track is reduced from 2,980 to 2,682.

FORMAT OF DISK STORAGE STATEMENT

The disk storage statement takes the following form:



where OP is one of the disk mnemonic operation codes and ADDR is the address of the leftmost position of the disk control field used in the operation.

The mnemonic operation code to be used for all seek operations is SD (Seek Disk).

Read, Write, and Write Disk Check mnemonic operation codes are shown in Figure 15 and the Scan Disk (special feature) mnemonic operation codes in Figure 16.

	Modes				
	SECTOR	ADDRESS	SECTOR OVERLAY	TRACK RECORD (Special feature)	
Operation (*WM=Word Mark)	(data only)	(data and addresses)	(data and sector count)	data only	data and addresses
READ DISK without WM* with WM	RD RDW	RDT RDTW	RDCO RDCOW	RDTR RDTRW	RDTA RDTAW
WRITE DISK without WM with WM	WD WDW	WDT WDTW	WDCO WDCOW	WDTR WDTRW	WDTA WDTAW
WRITE DISK CHECK without WM with WM	WDC WDCW	WDC WDCW	WDC WDCW	WDC WDCW	WDC WDCW

Figure 15. Mnemonic Operation Codes Used for Read, Write and Write Disk Check Operation by Modes

Operation	Condition		
	Low or Equal	Equal	High or Equal
SCAN DISK (Special Feature) without WM with WM	SDL SDLW	SDE SDEW	SDH SDHW

Figure 16. Mnemonic Operation Codes for the Scan Disk Operation

Types of Data That Can Be Stored on the Disk Packs

The disk packs can be used to store many kinds of files, records, tables and other data, including:

- *Master files* containing semipermanent information some of which is updated periodically.
- *Transaction files* containing information used to update master files.
- *Report files* containing information extracted from master files or information accumulated as transactions are processed.
- *Programs.* The programs and subroutines required for a number of 1440 processing runs can be stored on disks and read into core storage when required.
- *Tables* such as insurance rating tables, shipping rate tables, etc., or tables used to assist in locating disk records, can be maintained in disk storage and read into core storage when required.
- *Temporary storage.* In many applications it will be advisable to store results in the disk pack for subsequent processing, punching or printing.

Since processing master files is a primary function of the 1440 and since these files normally use a major portion of the disk packs, the subsequent sections are concerned primarily with the organization and processing of master files.

Basic Methods of File Organization and Processing

There are two basic approaches to organizing disk files: sequential and random. In *sequential file organization* the disk records are stored on the disks in control field sequence. In *random file organization,* the disk records are stored on the disks in a random sequence resulting from a method of control field conversion. (In control field conversion a programmed routine takes the control fields of the records and develops a series of disk addresses in a different order from that of the original control fields.)

There are also two basic methods of processing transactions against disk files to update the appropriate disk records: sequential and random. In *sequential processing,* transactions are first arranged in control field sequence and then entered into the data processing system. In *random processing,* transactions are entered into the system without any prearranging and are processed in the sequence entered.

Either method of processing can be used with either method of file organization. Thus transactions can be processed sequentially or randomly, against either sequential or random files. The two types of file organization and processing, and the distinctions between them, will be discussed in the subsequent sections. Sequential file organization will be covered first.

Sequential File Organization

Some of the characteristics of sequential file organization are:

- It can be used effectively for both random and sequential processing and for applications with high, average or low activity ratios.
- Since the disk packs can be removed from the disk drives, it is possible to start a data file on one disk pack, process to the end of it, remove the disk pack and replace it with another continuing the file. The disk storage capacity of the system is thus expandable when processing against a sequential file.
- Numerical, alphabetic or alphameric control fields of any normal size can be used without modification.
- Since the disk master records are stored sequentially, reports can be prepared with a minimum of sorting.

The following sections describe ways of retrieving records (that is locating and reading them) in sequential files, and setting up and maintaining this type of file.

Methods of Record Retrieval in Sequentially Organized Disk Files

There are at least four common methods of retrieving records in sequential files: direct addressing, disk scanning, table lookup and consecutive processing. In reading the following sections, remember that they deal with how disk records are located for updating purposes, assuming that the records have been previously entered in the disk file.

DIRECT ADDRESSING

In direct addressing, the control field of a disk record, or a transaction to update the record, is the same as the actual disk address, or can be simply modified to equal the disk address. A company with employee numbers ranging from 10,001 to 15,000 (4,000 employees, 20% expansion factor) could use the actual employee numbers as the disk address of the employee disk records (assuming a one-sector, 100-character employee master record). When processing the payroll, the employee's detail card(s) could then be read and the appropriate employee disk record retrieved by issuing a seek to the address specified by the employee number and then reading the record. If it were desired to locate the employee disk records in another part of the file – say, address 00001 to 05000 – this could be done by subtracting 10,000 from each employee number before issuing the seek instruction.

If the employee disk records were 200 characters long, each one would require two disk sectors and an appropriate disk address could be assigned to the employee disk records by multiplying the employee number by 2 and adding a constant to arrive at an address within the desired portion of the disk pack. Thus, with employee numbers ranging from 10,001 to 15,000 and 200-character disk records, multiplying by 2 and subtracting 20,001 would result in storing the employee disk records in sectors 00001 to 10000 with one record starting every two sectors.

When processing transactions using direct addressing, disk records can thus be retrieved by transferring the actual control field (or a slightly modified control field) to the disk control field and issuing a seek instruction followed by a read instruction. Figure 17 is an overall block diagram of a direct addressing application where issue cards, in item-number sequence, are used to update an inventory master file in the same sequence. Note the use of the IOCS macro instruction, Get Disk Record, which generates the required seek and read instructions.

With direct addressing, records can be processed randomly or sequentially. When processing transactions randomly, one seek and one read are normally required to retrieve a record. When processing sequentially, one seek is normally needed per cylinder, plus one read per record, except when the access arm is used alternately to update a second type of record in the same disk pack. In this case, one seek and one read are normally required for each file record updated. Direct addressing is thus an efficient method of record retrieval, since access time is minimized and little or no processing time is required to convert the control data of the record to a file address.

However, the control fields used in most applications do not contain the correct number of digits, or the correct distribution within the range of numbers, for use as direct addresses. When this is the case, either the control fields can be changed to coincide with file addresses, or provision can be made to associate a file address with the existing control fields. This association can be made in or out of the computer. Each of these possibilities is discussed below.

Renumbering. – Of these alternatives, renumbering requires the most external systems modifications – for example, changing catalogs, part listings, order forms, etc. Careful consideration should therefore be given to the approaches described below, unless the file is small and the change can be easily accomplished.

Externally assigning direct addresses. – Some methods that have been used for externally assigning direct addresses and entering them into the input cards are:

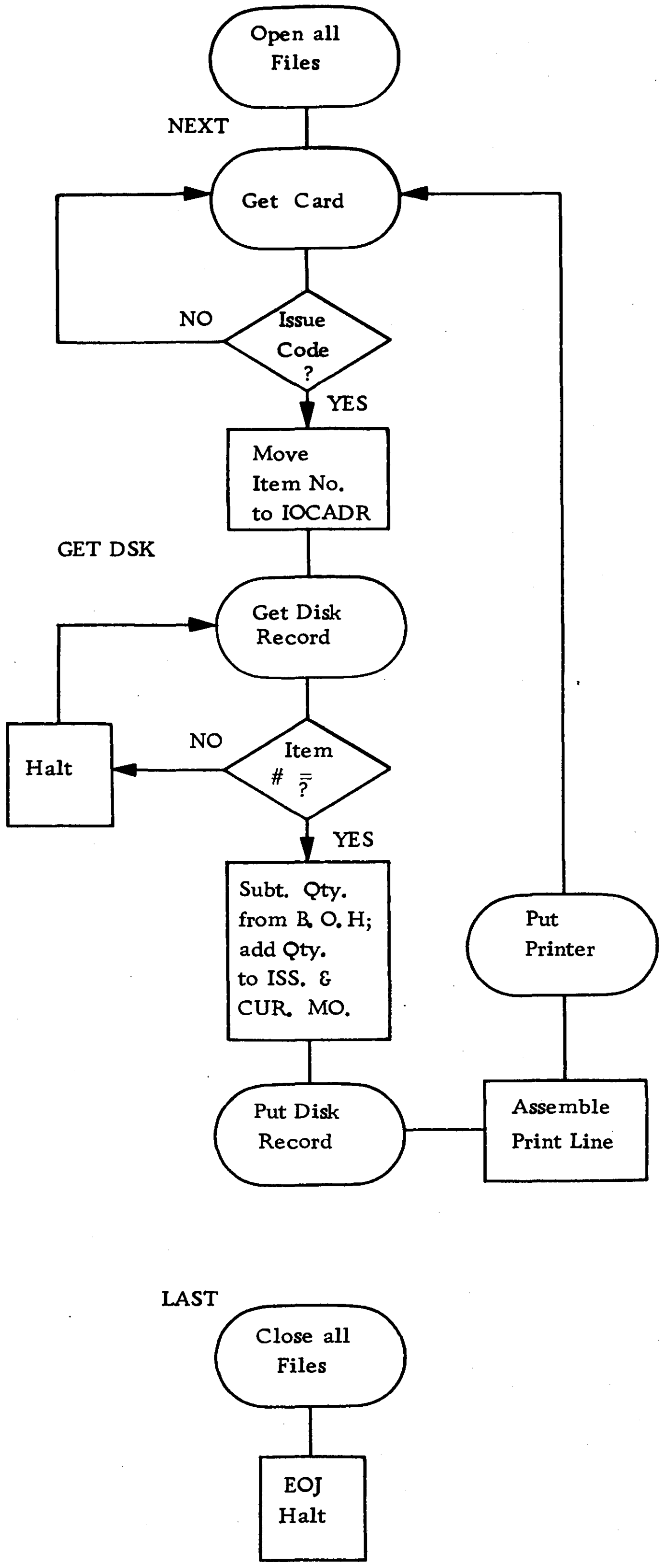


Figure 17. Illustrative Application: Record Retrieval and Processing Using Direct Addressing

- A manual coding step. The disk address is written on the input transaction by a manual coding step. Both fields are then keypunched in the input card.
- Tub files or master decks. The actual control field and the disk address are entered by input cards pulled from a tub file or reproduced from a master deck.
- Input generated from a prior run. Cards or records generated on previous 1440 runs contain the required direct addresses to provide an automatic re-entry to the system.
- Preprinted forms. The corresponding disk address is printed on the preprinted form as well as the actual item number. Both fields are then keypunched into the input card.

As can be seen, the effort required to associate control fields with direct disk addresses may be substantial. For this reason, if the actual control fields do not conform to the requirements necessary for direct addressing, other techniques should be examined carefully before planning either a major change of existing control fields or the external assignment of direct addresses.

DISK SCANNING

This method uses the optional Scan Disk feature, which enables the 1440 to make a rapid search of the records in disk storage for a specific control field. Only one seek and one scan disk instruction are needed to cause the program to search an entire cylinder.

In general terms this method of retrieval operates as follows:

A transaction is read, a seek is made to the proper cylinder (if required), and the file is rapidly scanned to locate the disk record containing the same control data as the transaction record. A read instruction is then used to read the disk record into core storage for updating.

In more technical terms the method can be outlined as follows:

The transaction control field to be matched with the disk record is placed in core storage in the search argument area next to the disk control field. (The transaction control field is entered in the search argument area corresponding to the position in which the control field is located in the disk record.) The disk control field is used to indicate where the scan is to begin and the maximum number of sectors to be scanned. A seek is then made to the desired cylinder (if required).

Once the access arms are on the appropriate cylinder, the scan is initiated and a search is made from sector to sector until the matching disk record is found. After the match is found, the disk control field contains the sector address of the record fulfilling the scan condition. A read instruction is then given (after a slight modification of the disk control field) and the record read into core storage.

With the disk scanning techniques, records can be processed randomly or sequentially. When processing randomly, disk scanning is effective in files of one cylinder or less. As files become larger, scanning may become too time-consuming and it is suggested that a table lookup method (described below) be first utilized to determine the cylinder in which the record is located and that the Scan Disk feature then be used to locate the records. Figure 18 illustrates the use of this feature.

EXAMPLE OF SCAN DISK TECHNIQUE

<u>GIVEN</u>:

1. Inventory summary file located in Cylinder 10 of a disk pack on File 0.
2. Part number contains eight alphameric characters.
3. Each summary record is 30 characters long.

Part # 8	Description 12	Unit Cost 5	On Hand 5

Summary Record on File

4. Each sector contains three summary records. File is organized sequentially.

<u>REQUIRED</u>:

Retrieve the summary record for part number #12AB9634 using Scan Disk features.

<u>SOLUTION</u>:

1. Establish disk control field in core to scan cylinder 10.

(Core Layout)	Core Sector Address	Sector Count
	*002000	200

Disk Control Field

2. Place the part number (i.e., search argument) adjacent to DCF followed by a group mark - word mark. In this case the part number is located in positions 1-8 of the summary record.

(Core Layout)	Core Sector Address	Sector Count	Search Argument
	*002000	200	12AB9634 ǂ

Disk Control Field

3. Execute seek to cylinder 10.
4. Execute scan instruction for low/equal.
5. A compare is automatically made sector by sector for low or equal condition. When this is found the core sector address contains the proper sector.
6. The word mark under the group mark search argument is removed by a Clear Word Mark instruction.
7. A read instruction is given to bring the sector containing the record into core.

Figure 18. Example of Scan Disk Technique

TABLE LOOKUP

In table lookup methods of record retrieval, tables are used to aid in locating the records to be updated. The tables can be kept entirely in core storage if space permits, in the disk file, or in both areas, and are used to find the cylinder and track on which a desired record is located. A scan disk or other technique is then used to find the needed record and the record is then read into core storage for processing.

Table lookup approaches are generally used when:

- Direct addressing is not feasible.
- Transactions are processed randomly against a sequentially organized file.
- Transactions are processed sequentially against a sequential file and the ratio of transactions processed to the total number of records in the disk file is not high.

There are two primary types of tables: (1) *cylinder tables,* used to determine the cylinder containing the desired record, and (2) *track tables,* used to determine the track within the cylinder on which the record is stored. There is usually one track table per cylinder. The following sections describe three illustrative table lookup techniques which can be modified to meet specific requirements.

Cylinder Table in Core.—A cylinder table (Figure 19), containing the control field of the highest record stored on each cylinder, is maintained in core storage. There is thus one table entry the length of the control field for each cylinder of the master file. The cylinder to which each entry refers is indicated by the relative location of the entry. The processing to retrieve a record with a cylinder table in core and the optional Scan Disk feature is outlined below:

1. Read the transaction card (or a transaction record from disk storage) into core storage.
2. Compare the control field from the transaction card with the control fields in the cylinder table. When an equal or low condition results, the proper cylinder has been located.
3. Place the cylinder address (the address of the 00 sector of the cylinder) in the disk control field and initiate a seek. The access arm will then move to the desired cylinder.
4. Initiate a disk scan of the cylinder to determine the address of the desired record (other ways of performing this function are discussed on page 30).
5. Read the desired record into core storage for processing.

This technique can be used for processing transactions randomly or sequentially.

Cylinder Table in the Disk File.—When it is not practical to store the cylinder table in core, it can be maintained in disk storage and read into core when required. While the table can be kept on any cylinder, it is usually desirable to store it on cylinder 00. (In most cases this location will minimize seek time.)

The processing to retrieve a record with a cylinder table on cylinder 00 and the optional Scan Disk feature is outlined below:

1. Initiate a seek to cylinder 00.
2. Read the transaction card or a transaction record from disk storage.
3. Read the first block of the cylinder table into core storage.
4. Compare the control field from the transaction card with the control fields in the first block of the cylinder table. When an equal or low condition results, the proper cylinder has been found. If this condition is not met in the first block of the cylinder table, read in the next block and continue the comparison until the desired cylinder is identified. (See Figure 20 for a technique to minimize the time for this function.)
5. Place the cylinder address (the address of the 00 sector of the cylinder) into the disk control field and initiate a seek. The access arm will then move to the desired cylinder.
6. Initiate a disk scan of the cylinder to locate the address of the desired record.
7. Read the desired record into core storage for processing.

This technique can be used for processing transactions randomly or sequentially and is shown in block diagram form in Figure 66, page 93.

Cylinder and Track Table.—In the previous techniques a cylinder table was used to determine on which cylinder the desired record was located. Since the record could be the first or the last one in the cylinder, it might require anywhere from 24 ms to 422 ms (422 ms are required to scan all the records in a cylinder) to locate the record. Track tables are used to speed this operation. (A track table contains the control field of the highest record stored on each track in a cylinder and is generally located on the first sector of each cylinder for ease of programming.)

After the cylinder table has been used to determine the desired cylinder, a seek to that cylinder is executed and the track table read into core. The track containing the record is determined by comparing the transaction control field against the control fields in the track table for an equal or low condition. A disk scan (or similar technique) can then be used to find the desired record.

In outline, the processing to retrieve a record using a cylinder table on cylinder 00 and a track table on the first sector of each cylinder is as follows:

1. Initiate a seek to cylinder 00.
2. Read the transaction card (or a transaction record from disk storage) into core storage.
3. Read the first block of the cylinder table into core storage.
4. Compare the control field from the transaction with the control fields in the first block of the cylinder table. When an equal or low condition results, the proper cylinder has been located. If this condition is not met in the first block of the cylinder table, read in the next block and continue the comparison until the desired cylinder is identified. (See Figure 20 for technique to minimize the time for this function.)
5. Place the cylinder address (the address of the 00 sector of the cylinder) into the disk control field and issue a seek. The access arm will then move to the desired cylinder.
6. Read the track index from sector 00 of the cylinder into core storage.
7. Compare the transaction control field against the control field in the track table. When an equal or low condition results, the appropriate track has been found.
8. Initiate a disk scan of the track to determine the address of the desired record. (Other ways of performing this function are described below.)
9. Read the desired record into core for processing.

This technique can be used for processing transactions randomly or sequentially and is shown in schematic form in Figure 21 and in block diagram form in Figure 67, page 94.

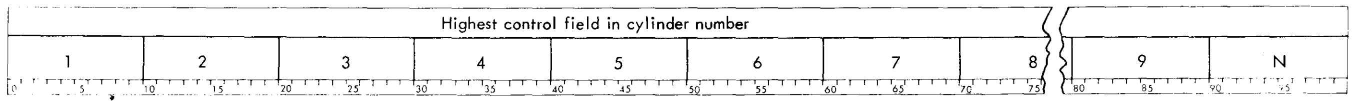


Figure 19. Cylinder Table in Core: Ten-Position Control Fields—File N Cylinders Long

Assume that the file is located on cylinders 1-99 and that each record has a ten-position control field.

Each sector of the cylinder table contains the control fields of the highest records on the corresponding ten cylinders. The cylinder table is located on sectors 0-19 of cylinder 00 and is arranged as shown below. The stagger arrangement is used to minimize the time required to read and compare the entries in the table.

Highest Control Field on Cylinders																			
Label	1-9	70-79		10-19	80-89		20-29	90-99		30-39			40-49			50-59			60-69
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

Cylinder Table (Sectors 0-19, Cylinder 00)

1. Read sector 1.
2. Compare the transaction control field against the ten entries in sector 1 for an equal or low condition. If the condition is not met, read sector 4 and repeat the process. When the condition is met, the correct cylinder has been located.

Staggering the entries in the cylinder table in this manner provides 4 ms in which to perform the ten compares for each sector of the table eliminating the necessity for a full 40 ms rotational delay before reading the next sector. The entire cylinder table can thus be searched in a maximum of 1-1/2 disk revolutions (60 ms). Cylinder tables for cylinders 1-9, 10-19, 20-29, 30-39, 40-49, 50-59 and 60-69 are checked on the first rotation. The tables for cylinders 70-79, 80-89 and 90-99 are checked during the first half of the second rotation.

Figure 20. Stagger Technique for Minimizing Time to Check Cylinder Table

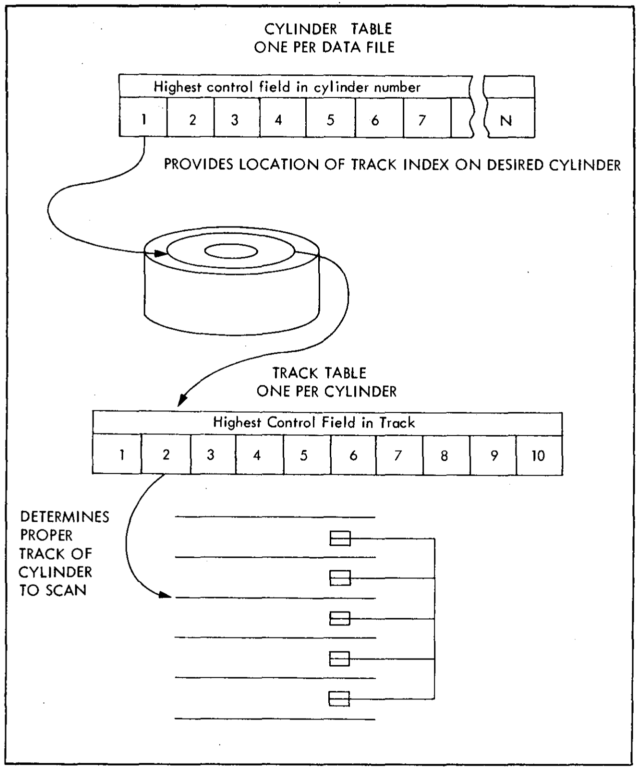

CYLINDER TABLE
ONE PER DATA FILE
Highest control field in cylinder number
1
2
3
4
5
6
7
N
PROVIDES LOCATION OF TRACK INDEX ON DESIRED CYLINDER
TRACK TABLE
ONE PER CYLINDER
Highest Control Field in Track
1
2
3
4
5
6
7
8
9
10
DETERMINES PROPER TRACK OF CYLINDER TO SCAN

Figure 21. Schematic of Use of Cylinder and Track Tables

Locating Records on a Track.—The previous indexing techniques all used the optional Scan Disk feature to finally locate the desired record. However, there are several methods of finding a record once the track on which it is located has been determined. One of these is to start at sector zero of the track, read the disk record into core, and compare its control field with the transaction control field. If it matches, the disk record is then updated; if not, the process is continued until a match is found. This approach is easy to program but has limitations from a timing standpoint because each subsequent read requires a complete 40-ms disk rotation. (However, the number of reads to locate a record can be minimized by reading a block of records into core.)

A quicker method is to stagger the reading of the sectors. Instead of reading sector zero and then sector one, a number of sectors are skipped so that if no match occurs, sufficient time is available to read another record without the loss of a full 40 ms for rotational delay. (Figure 22 illustrates this technique.)

The stagger method results in locating a record in an average of one and a half revolutions. A maximum of three revolutions will be required to examine every record on a surface. Thus, with this technique, the time required to retrieve a 100-character record after its track location is determined varies from 24 to 142 ms.

EXAMPLE OF TECHNIQUE FOR MINIMIZING TIME TO LOCATE A GIVEN RECORD ON A TRACK

GIVEN:

1. There are 40 summary records per track (50 characters each), two per sector.
2. The disk control field is set up to read in sector 0 of the track containing the desired record (0200 in this example).
3. A maximum of 40 compares (two summary records per sector; 20 sectors per track) might be required before locating the summary record to be updated.

REQUIRED:

Read and compare all records on the track against a transition control file in the minimum amount of time.

SOLUTION:

The first sector on the track is read and the two records in it are checked. If neither is the one desired, the fourth sector on the track is read and checked. (Since the time needed to check the two records is less than 4 ms, there is practically no rotational delay before reading the fourth sector.) The search continues in this manner, reading every third sector until either the record is found or 21 reads have been made. The latter would indicate a not-in-file condition. An average of 1 1/2 revolutions (60 ms) is required to locate a record. The maximum is three revolutions.

After each read the DCF is increased by 00002 to read the next record. After seven reads, the DCF is decreased by 00018 to allow the next revolution to continue the search. If the record is not found on the third revolution, the first record on the next track is read. This situation or a cylinder overflow will cause a branch to a not-in-file routine.

The records read and checked on each of the three searches are shown below.

1st Search
02000 - 2003 - 2006 - 2009 - 2012 - 2015 - 2018

2nd Search
02001 - 02003 - 02007 - 02010 - 2013 - 2016 - 2019

3rd Search
2000 - 2005 - 2008 - 2011 - 2014 - 2015

Figure 22. Example of Technique for Minimizing Time to Locate a Given Record on a Track

Another method of record retrieval, once the record's track location is determined, is to initiate a disk scan of the track. Every record on the track could then be scanned in one disk rotation. A read instruction would then be used to read the record into core. With this technique (used in step 8 of the previous example) the time required to retrieve a 100-character record after its track location is determined varies from 64 to 102 ms.

CONSECUTIVE PROCESSING

In consecutive processing, the records are stored on the disks in control field sequence and the transactions are entered in the same sequence. The disk records are read starting with a beginning disk address and continuing from sector to sector (or track to track) to the end of the data file. When a disk record is matched by a transaction, the record is updated and subsequently written back on the disk.

In outline, the processing is as follows:

The first transaction and a series of disk records* are read into core storage. The transaction control field and the control fields of the disk records are then compared until an equal condition is found. The corresponding record is then updated and is subsequently written back into disk storage.

Note that all disk records are read into core (similar to tape processing) but only the active records are modified and re-entered on the disk. Record retrieval is thus based on sequentially reading all the disk records into core to see whether they are to be updated. Only one seek instruction is normally required for each cylinder, and processing is continued from the point of the last match.

This technique is effective when the ratio of transactions processed to the number of disk records is high. Many records can then be updated with one setting of the access arm and a minimum of time lost in reading inactive records.

Figure 23 is an abbreviated block diagram of a consecutive processing approach. While in this example only one disk record is read at a time, a block of records is frequently read (depending on available core storage) to increase throughput by minimizing the time required for rotational delay.

Control Sequential Processing.—This is essentially the same as consecutive processing but applies only to the processing of files organized according to the control-sequential method of file organization provided by the IBM file-organization programs. In this type of organization, records are arranged in sequential order by control data, such as customer number. The sequential records are stored successively on disk (from sector to sector) wherever possible. When a sequential record is not stored in successive order, a linkage is provided to locate that record for processing. This out-of-order condition may have occurred, for example, when additions were made to a file after it was originally set up. The last field of each record is a linkage field that gives the address of the next sequential record if that record is filed out of order. The linkage field in the out-of-order record provides the address of the next record in sequence.

* One disk record or a block of disk records can be read into core at a time, depending upon available core storage.

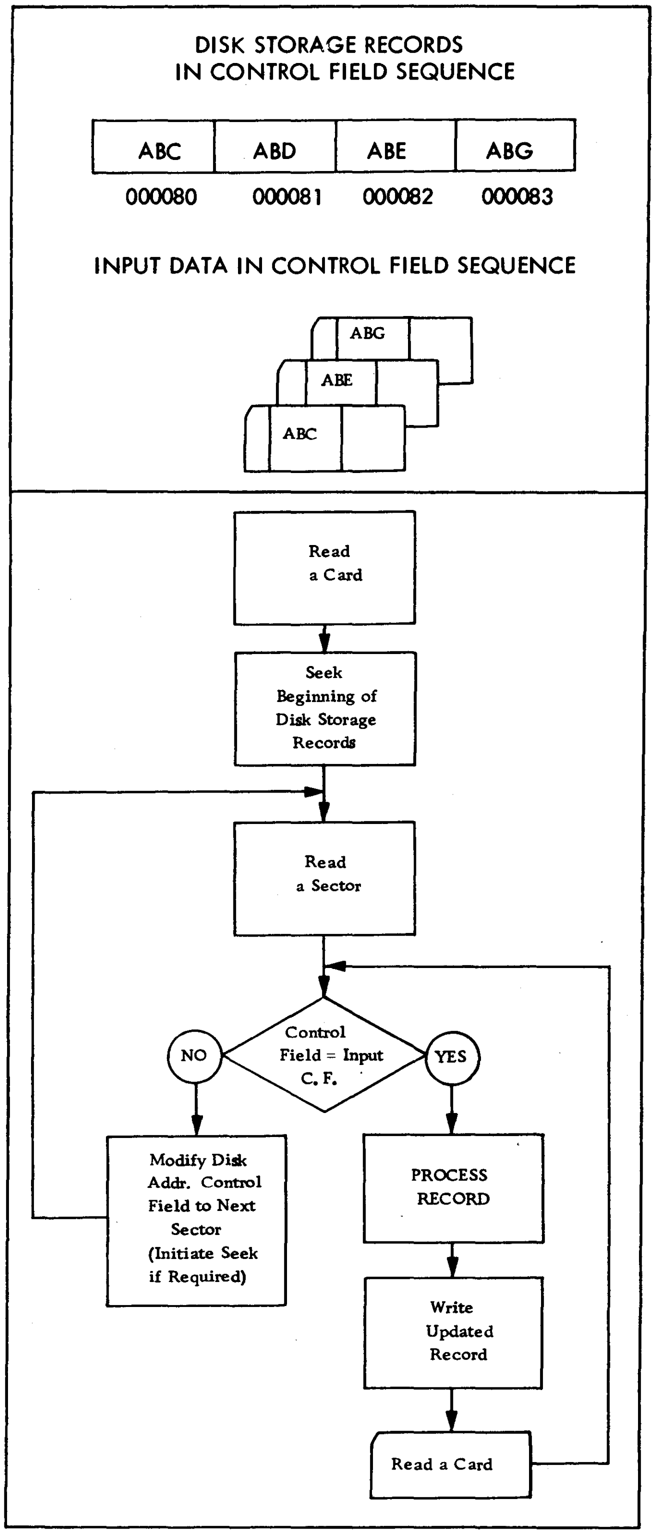

DISK STORAGE RECORDS IN CONTROL FIELD SEQUENCE
ABC
ABD
ABE
ABG
000080
000081
000082
000083
INPUT DATA IN CONTROL FIELD SEQUENCE
ABG
ABE
ABC
Read a Card
Seek Beginning of Disk Storage Records
Read a Sector
Control Field = Input C. F.
NO
YES
Modify Disk Addr. Control Field to Next Sector (Initiate Seek if Required)
PROCESS RECORD
Write Updated Record
Read a Card

Figure 23. Abbreviated Block Diagram of Consecutive Processing Approach

The above examples of the four common methods of retrieving records illustrate some of the many possible variations that can be used. Each application should be thoroughly analyzed to select the variation best suited to it.

Setting Up and Maintaining Sequential Files

The foregoing section outlined some common methods for retrieving disk records for updating purposes and assumed that the records were already entered in disk storage. Setting up and maintaining the records requires that provision be made for initially loading the files and constructing any needed tables, and for adding, deleting and modifying records once the files are set up.

The file maintenance function and the updating function are basic to data processing procedures and in actual practice the two functions often overlap. However, file maintenance usually denotes adding, and deleting records and modifying the reference and qualitative portions of records previously entered on the file; updating usually denotes changing the quantitative portions of records stored on a file. In tape processing, a new master tape is created during the normal updating routines; additions are inserted in place and deletions removed as the tape is written. In disk processing, master records usually retain their physical location on the original pack during normal updating routines; only active records are rewritten and a complete new file is not created. Thus disk file maintenance techniques generally provide for periodic recopying runs to rewrite the entire file so that additions can be inserted into their proper place between records previously entered on the file.

LOADING THE FILES

In most cases a separate load routine is used for initial loading of the files. For example, the IBM 1440 Control Sequential Disk File Organization package provides one routine (CSLOAD) for the initial loading of a sequenced file and another (CSADD) for additions to the file. The load routine (CSLOAD) is also used after a file has been reorganized and it is necessary to load the entire file back into the proper disk storage locations.

In addition to entering the disk records, load routines usually provide for preparing any necessary tables for locating records. An example is the distribution index created in the CSLOAD routine (Figure 24).

FILE MAINTENANCE

Once the files are loaded, a number of techniques are used for file maintenance. Among them are the following:

- File maintenance and file copy are performed in one run. On this combined run the new master records are inserted in correct sequence (interspersed between existing records where necessary), old records are dropped and a new file is written. The new file is then processed on the next updating run. The old file is held for audit trail purposes.
- File maintenance and file copying are performed on separate runs. On the file maintenance run added records are entered in a special additions area, and an address link is provided to locate the new records. Then, either on a periodic basis or when additions exceed the space allotted, a copying run is provided to rewrite and reorganize the file.
- File maintenance and updating are combined in one run, and file copying is performed on a separate run. Provision is made to add new records in an additions area during the updating run. When necessary, the file is reorganized and written on a new file during a separate copying run.

All three techniques also provide for deleting master records when required. Figure 25 shows the relationship of the various runs in the techniques just described.

FILE REORGANIZATION

Multi-File Systems.—All the techniques mentioned above provide for periodically copying and reorganizing the master file. On a two-file system, this is a rapid and simple operation, since it is possible to read records from an "old" disk pack and write them out on a "new" pack while incorporating additions and deletions. The additions and deletions can be entered from change cards on the file copying and reorganization run or they can be entered into an additions portion of the file on previous runs. Where table lookup methods of record retrieval are used, the required tables are generally reorganized during this run in addition to the reorganization of the records themselves. The new disk file is used for subsequent processing and the old file is held for audit trail purposes.

Single-File Systems.—Since recopying a complete pack cannot be done on a one-file system, other techniques are used for file reorganization. Among them are the following:

Flip-Flop Technique.—This technique simulates two-file disk copying by dividing the disk pack into parts and copying from one part to another. The portion of the disk pack containing the master records to be reorganized is written on a blank portion of the pack and at the same time insertions, deletions and changes are incorporated. (The changes can come from disk storage or from cards.) After this has been done, the reorganized file is written back into the area of the file from which it originally came.

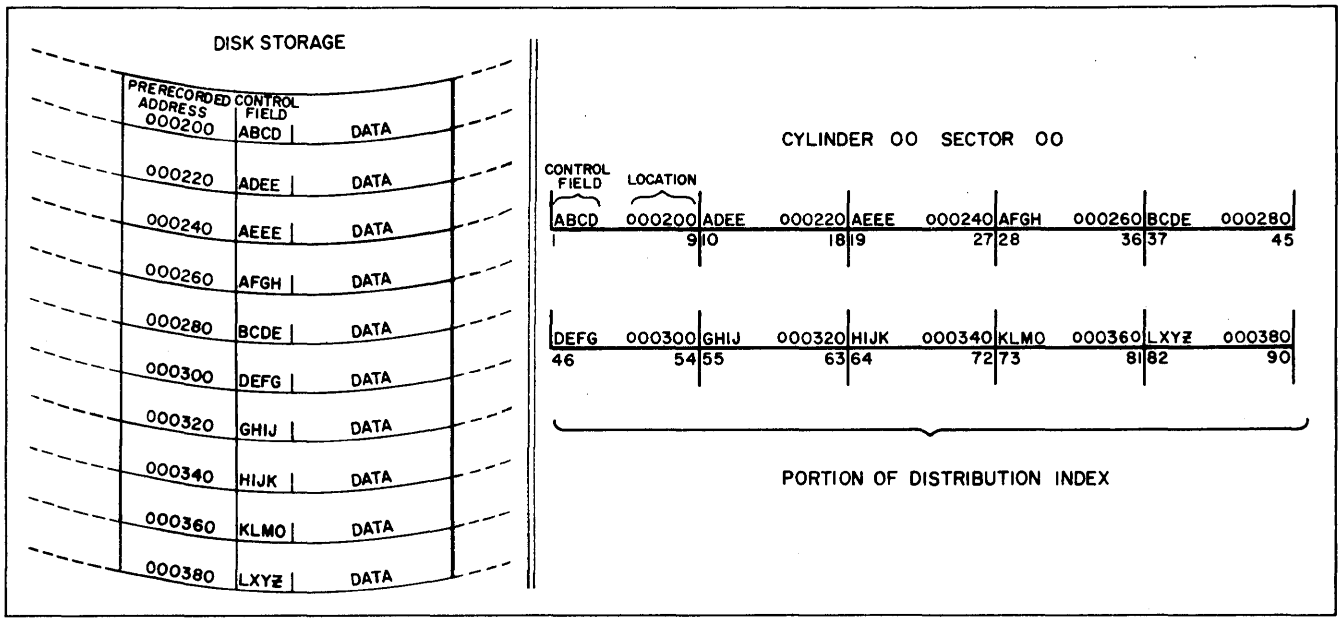


Figure 24. Example of Distribution Index of the Type Set up by IBM Control Sequential File Routines

Overlay Technique.—This technique provides for using a work area in the disk pack to store a portion of the master file while it is being reorganized and then rewriting the reorganized portion of the file over the area it previously came from. It differs from the above-mentioned flip-flop technique, which reorganizes a complete file and then writes it back, in that one portion of the file is reorganized and written back at a time until the entire file has been processed. Since one block of records is processed at a time, the overlay technique requires a minimum of file storage to reorganize a file.

Cylinder Shift Technique.—This technique is similar to the cylinder overlay technique in that a work area is used to store a portion of the file while it is being reorganized. However, rather than writing the information back into the same area from which it came, the reorganized section of the file is written back, shifted to the left over a portion of the file previously reorganized. After the whole file is reorganized and shifted, it is reshifted back to its original location.

All three techniques described above provide for reorganizing and rewriting a disk file on a single pack system. With any of them, changes can either be entered into the 1440 at the time of the reorganization or be entered at a prior time into an addition area in core (address linkage fields are used to connect the additions to the preceding and succeeding records). These file maintenance techniques can thus be used for daily file maintenance or can be used periodically to reorganize a file containing linked additions and items for deletion. In reorganizing files that call for table lookup retrieval techniques, both the records and the tables are usually reorganized on the same run.

File Punchout Technique.—This technique calls for punching out the master records, including any additions previously entered into the additions area of the file. The punched file is resequenced and modified offline and then rewritten back onto the disk pack. With the 1440 Card Read Punch, this method is effective for small files, but the punching time required for larger files is usually prohibitive. With the 1444 Card Punch, however, punching time is greatly reduced and the technique can be readily used for larger files.

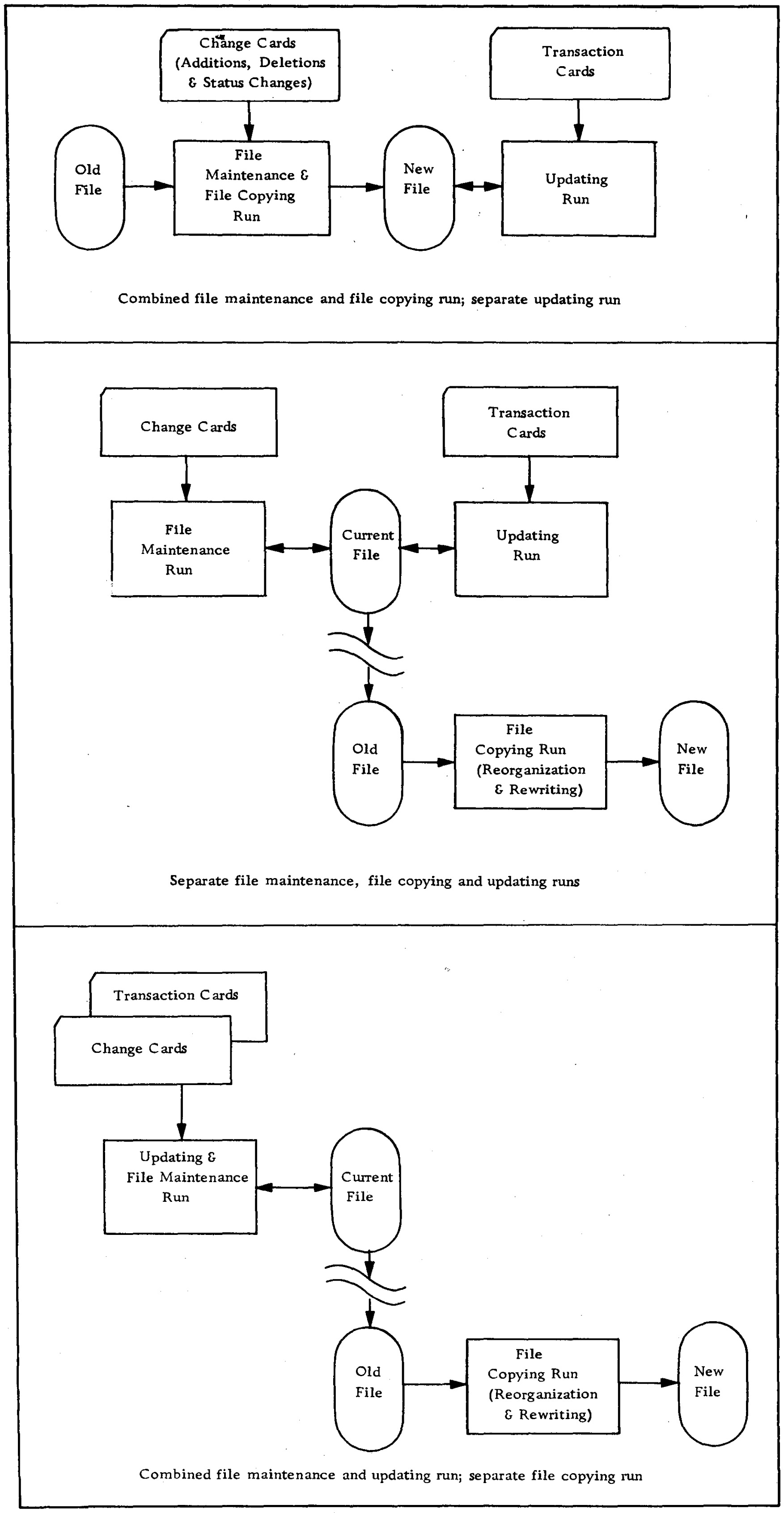

Change Cards (Additions, Deletions & Status Changes)
Transaction Cards
Old File
File Maintenance & File Copying Run
New File
Updating Run
Combined file maintenance and file copying run; separate updating run
Change Cards
Transaction Cards
File Maintenance Run
Current File
Updating Run
Old File
File Copying Run (Reorganization & Rewriting)
New File
Separate file maintenance, file copying and updating runs
Transaction Cards
Change Cards
Updating & File Maintenance Run
Current File
Old File
File Copying Run (Reorganization & Rewriting)
New File
Combined file maintenance and updating run; separate file copying run

Figure 25. Relationship of File Maintenance, File Copying, and Updating Runs in Three File Maintenance Techniques

AUDIT TRAILS

While the possible loss of data on the disk files is minimized by system reliability and by such techniques as read and write checks, a means of reconstructing data should be provided. Basically, this requires that reserve files be established containing the data as of a cutoff point. The reserve files are held, along with the transactions for the period, so that if it becomes necessary to reconstruct a file the transactions that occurred since the last cutoff date can be processed against the reserve file to bring it up to date. The reserve file may be in disk pack or card form or in a combination of these forms.

Multi-Drive Systems.—The ability to rapidly copy packs (a complete pack can be copied in about six minutes) on a multi-drive system permits very direct reconstruction procedures. Daily, or periodically, the updated file is written on a reserve file. One file is then processed on the succeeding updating run and the other is retained as a reserve file.

If it ever becomes necessary to reconstruct the updated file, the reserve file is mounted on the disk storage drive and is updated by the transactions and changes that have occurred since it was created. (The transactions can be maintained in either cards or disk storage.) Next the updated file is copied; one file is then used for subsequent updating, the other for reserve purposes.

Quite often, copying the file for audit trail purposes is combined with copying the file for file maintenance purposes. The reorganized file is then used as input on the next processing cycle and the old file temporarily retained as the reserve file.

Single-Drive Systems.—Since it is impossible to copy from pack to pack in a single-drive system, a number of techniques are used to provide the required audit trail information. Among them are:

Periodic punchout of all master records.—In this approach all master records are periodically punched into cards and the cards held as a reserve file. With the 1442 Card Read Punch this approach is time-consuming and is effective primarily with short files. With the 1444 Card Punch, however, punching time is greatly reduced, and the technique can be readily used for larger files.

Periodic punchout of quantitative portion of the disk records.—Instead of punching out the entire disk record, only the quantitative portion of the record (that is, the balance fields and other changeable fields such as date of last transaction) are punched out.

In one example of this approach a card file containing the reference and qualitative portions of the disk records is maintained offline. For every new record added to the disk file a card is added to the offline file. The card file thus duplicates a portion of the records on the disk file.

Periodically the quantitative portion of the disk records is punched into spread cards. The spread card file and offline file thus constitute a complete reserve file.

Periodic punchout of summary records coupled with use of a reserve (shelf) disk pack.—This approach provides for the maintenance of a reserve disk pack and for the periodic updating of the reserve pack by use of summary records punched from the regular disk pack. The reserve pack is updated at the end of the period and held until the next period for backup purposes.

Figure 28 shows one example of the use of this technique. Note that the file maintenance processing is duplicated for each pack, except that the reserve pack is maintained only on a monthly basis. However, no daily transactions are processed against the reserve pack. At the end of the period, summary records containing the updated quantitative portions of the records are punched from the regular pack and used to update the reserve pack.

Punching out of new balance records, coupled with the use of a reserve disk pack.—When a record is updated it is tagged with a special code. At the end of the day (or period) the tagged records are punched out. The cards are then processed against the reserve disk pack. This method is effective when a large number of transactions are processed against individual master records, since the reserve pack can then be updated with relatively few entries. Figure 26 shows an example of this technique.

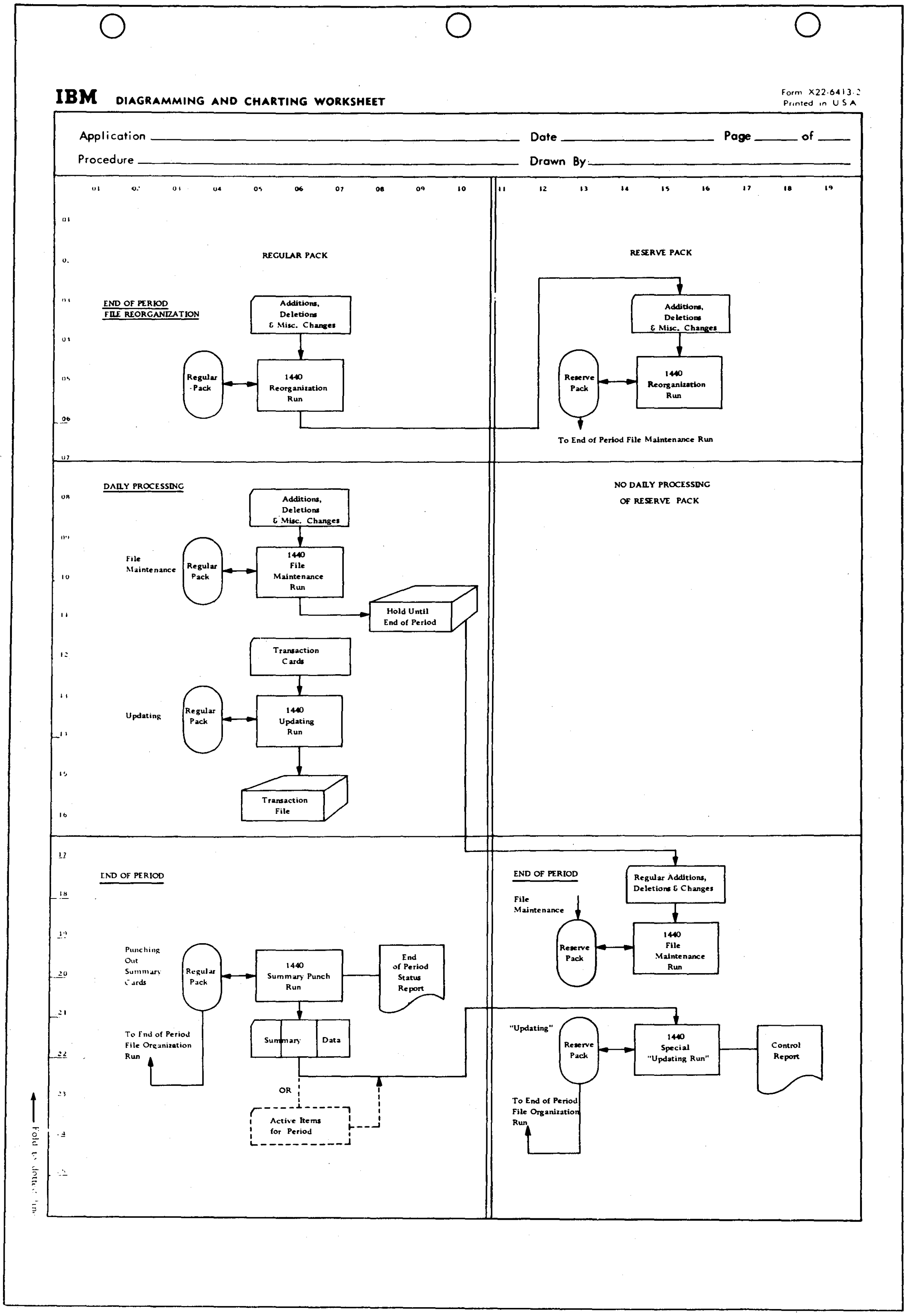


Figure 26. Overall Flowchart: Reserve Pack and Summary Punchout Technique (Also Active Record Punchout Technique)

Random File Organization

In random file organization a conversion routine is used to operate on the existing control fields to produce disk addresses. In most cases the use of a conversion routine results in assigning the same disk addresses to more than one record. Such duplicate disk addresses are called synonyms.

Conversion Routines

The objective of a conversion routine is to convert the control data of the records in a file to a randomly distributed series of disk addresses within a desired range and with a minimum number of synonyms. It is not possible to specify any one method of making this conversion. Each situation must be studied individually to determine the best method.

Figure 27 shows that address conversion is like a funnel directing data into the desired portion of a disk pack. The example illustrates the use of 11,000 disk sectors to store 10,000 one-hundred-character records. The control data of the records consists of eight-position fields. Thus, though these fields are used to identify only 10,000 records, there are 100,000,000 numbers available in the range that eight positions provide.

Step 1 is the application of some arithmetic operation to the existing control fields to create new numbers that fall between 00000 and 99999.

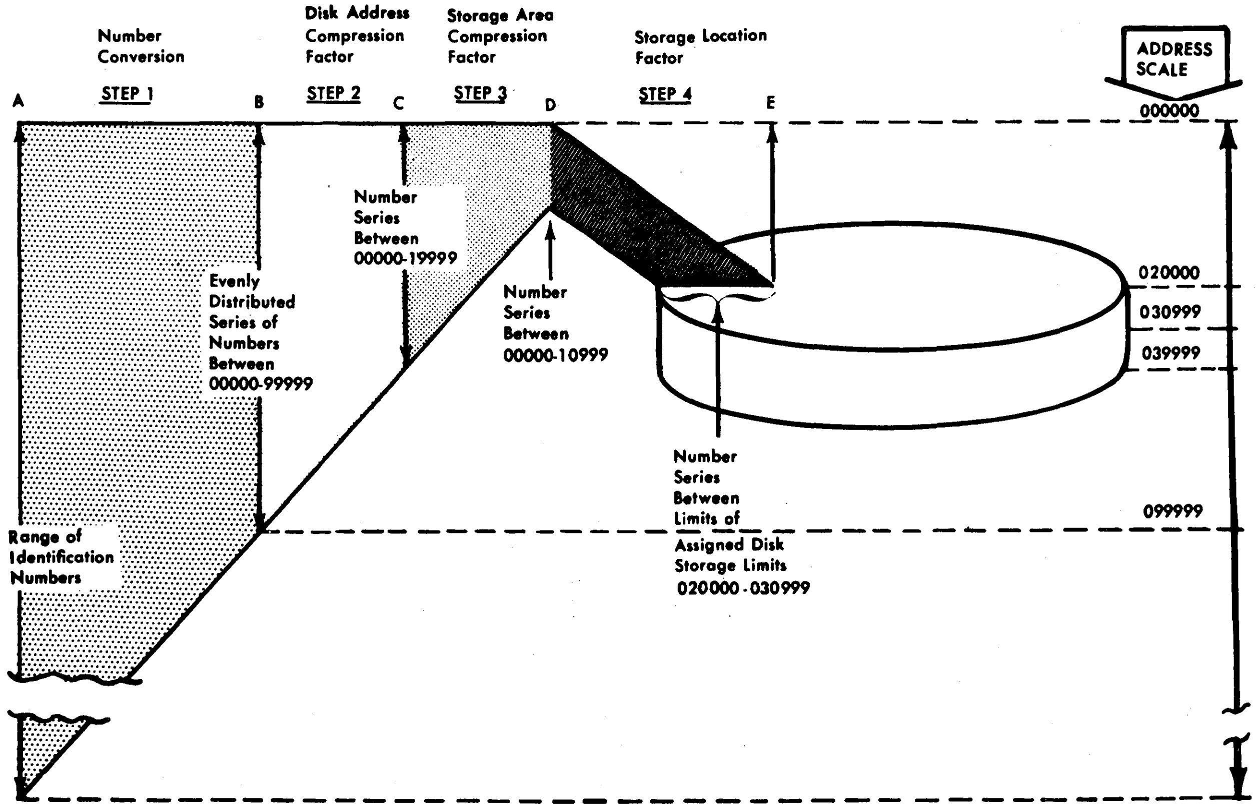
Step 2 reduces the derived number to bring it within the acceptable address range of the IBM 1311 by applying a compression factor. This is accomplished by multiplying the number by .2 to create a number range between 00000 and 19999.

Step 3 applies a second compression factor to bring the addressing range into a limited portion of the disk pack. To reduce the range to 11,000 sectors, the number is again multiplied by a factor. In this case, the factor is .55, and the addresses fall into the range 00000-10999.

In actual practice, the compression factors in steps 2 and 3 would be combined into one factor of .11.

Step 4 is to add a constant when necessary to place the addresses within a specific set of cylinders or tracks.

Several methods of number conversion have been found to be practical. Two of these are outlined on the following page.



Number Conversion
Disk Address Compression Factor
Storage Area Compression Factor
Storage Location Factor
ADDRESS SCALE
A
STEP 1
B
STEP 2
C
STEP 3
D
STEP 4
E
000000
Number Series Between 00000-19999
Evenly Distributed Series of Numbers Between 00000-99999
Number Series Between 00000-10999
020000
030999
039999
Number Series Between Limits of Assigned Disk Storage Limits 020000-030999
099999
Range of Identification Numbers

Figure 27. The Steps in Address Conversion

DIVIDING BY RANGE

A simple, but often effective, method of arriving at a random distribution of addresses is accomplished by dividing the control data by the range into which the addresses are to fall. The remainder of the division is used as the disk address. The results of the operation tend to a random distribution of numbers within the desired range. Thus, no additional compression factor is required. A further refinement is to use the prime number closest to the range.

Using the item number 53TP4197 (zone bits ignored) and assuming the records are to be fitted into 11,000 sectors:

```
                         4852.
                 ↗ 10,999/53374197. ← control data
     range − 1 ↗         43996
                         93781
                         87992
                          57899
                          54995
                           29047
                           21998
                          007049 ← disk address
```

SQUARING

Another method of arriving at a random distribution is to square the control number and select a number from the center portion of the product. Again using the item number 53TP4197.

```
53TP4197² =          53374197
                     53374197
                    373619379
                   480367773
                   53374179
                 213496788
                373619397
               160122591
              160122591
             266870985
             2848804905394809
```

The center six digits, 049053, are each obtained by adding down from five to eight other digits. These six digits can be expected to be the most random in the product, and when additional item numbers are treated in this manner, these digits usually have an even distribution over the numbers between 000000 and 999999.

In the event that the original number is too large to be handled in this way, the number can be overlapped and added and the shortened number can then be squared.

Chaining

The file organization technique used with a file employing indirect addressing must be able to accommodate the synonyms (or duplicate addresses) produced. The term chaining is applied to a technique that has been found to be efficient in most cases. Chaining is particularly suited for use in 1311 disk storage, since successive links of a chain of synonyms can often be read without reseeking.

As each record is read in to be loaded in disk storage, its control data is converted to a disk address. These converted addresses are called home addresses. Because only one record can be stored in each location, the first synonym (home record) is placed in the original address developed (home address). The additional record or records (non-homes) are stored in overflow locations. The address of the first overflow location is stored in the home-address location. The address of the second overflow location is stored in the first overflow location, etc. Chaining requires that, in the home address and all overflow locations, space be reserved for the address of the next location or link in the chain (Figure 28).

A chained file is normally loaded in two passes. During the first pass, only those records that can be placed in home locations are loaded. The second pass places all of the remaining records in overflow locations. Each overflow record is placed in an available location as close as possible to the previous link in the chain.

Retrieval of a record is accomplished by converting the control data to the home address. The record in the home-address location is read into core storage, and its control data is compared with that of the record being sought. If the control fields are not equal, the address of the first overflow record is extracted from the home record and another read command is issued using this address. The process is repeated until the desired record is found.

The time required to retrieve a particular record depends upon its position in a chain. All home records can be retrieved with a single seek and read. Subsequent links of a chain may require additional reads and, possibly, additional seeks.

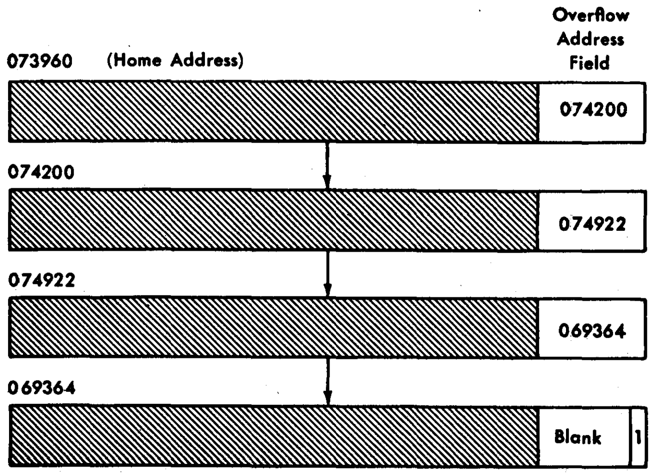


Figure 28. Example of Disk Storage Chaining

FILE PACKING

The average number of reads required to retrieve a record from a chained file depends primarily upon the number of synonyms developed by the conversion routine. The number of synonyms produced by a randomizing conversion routine can be reduced by assigning more disk storage space than actually required by the file. The percentage of the file area actually used for records is called the packing factor. For example, 11,000 disk locations might be used to store 10,000 records. Thus, the file would be said to be 91% packed.

With a chained file, the average time required to retrieve records diminishes as the packing factor decreases. However, it is impossible to state any given packing factor as the best for all chained files. The packing of an efficiently organized file can vary from 65% to 95%.

The time required to retrieve a record is not the only factor by which the efficiency of a file is measured. For example, a file might be packed close to 100% if by so doing the entire file can be kept within one disk pack. Another point to consider is anticipated growth. Room must be left in a file area to accommodate records added at a later date.

BLOCKING RECORDS IN A CHAINED FILE

Records can be easily blocked in 1311 disk storage. Because a single 100-character sector is the smallest unit of storage that can be read or written, the block length should be some multiple of 100 characters. The block length is limited primarily by the amount of core storage available for reading and writing the disk.

One reason for blocking records in a chained file is to save storage space. For example, four 125-character disk records can be stored in one five-sector block instead of the eight sectors required without blocking. Another reason is that the average number of reads required to locate a record can usually be reduced by increasing the blocking factor (number of records per block). The greater the blocking factor, the greater the chance that overflow records will be in the same block as the home record. When they are, no additional read is required. It should be remembered, however, that the longer the block length, the longer the time required to accomplish a single read operation.

FREQUENCY LOADING

The total time required to process a chained file can often be reduced by loading the most active records first, thus assuring their being placed in a home location or a prime position in a chain. It has been found that in many applications about 80% of the transactions apply to only 20% of the file.

If no activity count is available when initially loading a file, it may be worthwhile to set up a field in each record to accumulate such a count. At a later date, the file can be sorted on this count and reloaded.

ADDITIONS AND DELETIONS

Continual additions to and deletions from a file affect the efficiency of any organization scheme. The effect on a chained file, however, is comparatively slight. Thus, chaining is particularly suited to files with a very large turnover.

Timing

Systems timing is necessary for choosing alternate systems approaches and for estimating the time to accomplish a given job. For simple runs, throughput time may be estimated by brief calculations or by referring to the tables shown in this manual and then adding setup and card-handling times. Many cases, however, require more detailed timing analysis. Briefly, one effective way of doing this is as follows:

A general block diagram of the program is drawn, showing the card-reading, disk, card-punching, printing and processing operations in their proper sequence. The amount of time for each of these operations is then calculated and laid out on the timing layout chart (X24-3097). After completion of the chart, the duration of the cycle is determined by measuring the time across the chart. The cycle time is converted to rated throughput (cycles per minute) by dividing it into 60,000 or by using the tables in the timing reference card (Figure 47). Job time is then arrived at by dividing the cycles per minute into the volume and adding handling and setup times.

Where different transactions result in different processing cycles, each cycle should be laid out, converted to cycles per minute and the resulting throughput rates for the various transactions divided into their volumes. The individual results can then be totaled and added to handling and setup time to arrive at job time.

To yield correct results, timing should be based on accurate volumes and on detailed knowledge of the functions to be performed, the effect of the input/output commands, and the way the functions will actually be programmed. Bear in mind that most of the speeds given in the following pages are maximum rated speeds; setup and handling time must be added to arrive at actual throughput.

Using the Timing Layout Chart

The timing layout is designed to aid in graphically timing 1440 processing runs. It serves as a tool for both estimating cycle times and determining the optimum sequence of programmed operations.

The layout chart is scaled across the top in milliseconds (0-1,500) and down the left side with lines representing the basic 1440 functions. To use the chart, the time required to perform each function is first calculated and lines then drawn to indicate the number of milliseconds required for each one. The sequence of the various functions is indicated by the starting and ending points of the function lines – if the functions overlap either partially or fully, their lines overlap accordingly. The total time for the processing cycle is determined by measuring the distance necessary to include all the function lines on the milliseconds scale. The timing reference card then provides the rated throughput per minute.

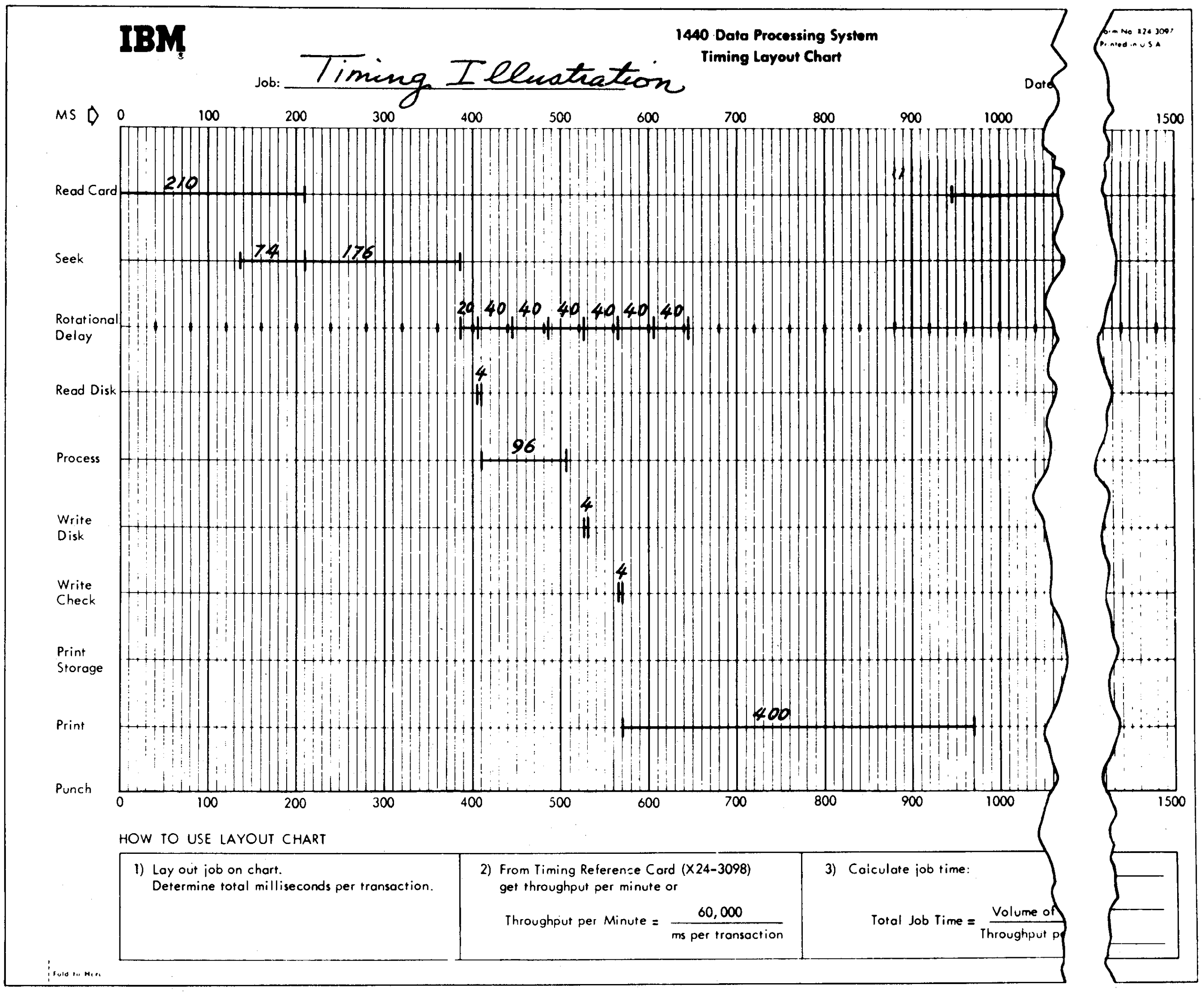
Figure 29 illustrates the use of the timing layout chart by showing a processing cycle which includes reading a card, seeking the corresponding master record in the disk file, updating the master record and printing a line on a transaction report. Columns 1 through 80 are read and a Model 1 Card Read Punch and Model 1 Printer are used. The disk record contains 200 characters.

To start the timing layout, a line for the 210 ms required for card reading was first drawn (0 to 210 ms). It was then determined, using the formula shown in the reference card, that 74 ms of the card read cycle would be available for processing and for seeking the master record. Since seek time is assumed to be an average of 250 ms in this example, the seek time line was drawn from 136 to 386 ms. After the proper cylinder has been located by the seek, it is necessary to account for the disk rotation time until the proper record is positioned for reading under the read-write heads. Since the time for one complete disk rotation is 40 ms, an average of 20 ms will elapse before the record can be read. Lines for the rotational delay (386-406 ms) and the read disk data transfer (406-410 ms) were thus drawn on the timing layout.

From the point where the data is first read (406 ms), the same sector location will again be available under the read-write head for writing back the updated master record every 40 ms. Thus, processing time between disk operations should be considered in increments of 40 ms, the disk rotation time which must elapse before the updated master record can be rewritten. This is indicated on the chart by laying out successive 40-ms increments to show the times available to execute a sector write operation. (No additional seek is necessary to rewrite the master record in this example, since the access mechanism remains at the last used cylinder position until a new seek command is issued.)

In the example, process time was assumed to be 96 ms. The timing line sequence after the record is read is thus process time (410-506 ms), rotational delay (506-526 ms), write time (526-530), rotational delay (530-566) and write check time (566-570). At the conclusion of these functions the printing operation can be executed. A line of 400 ms (546-946) was thus drawn for printing time. Since the last 24 ms of the print cycle can be overlapped by the reading of the next card the total processing cycle is 946 ms, equivalent to 62 transactions per minute.

For purposes of clarity, head select time was not discussed in the above. See page 46, and Figure 41, page 47, for further details of this timing element.


IBM

1440 Data Processing System
Timing Layout Chart

Job: Timing Illustration

HOW TO USE LAYOUT CHART

1) Lay out job on chart. Determine total milliseconds per transaction.

2) From Timing Reference Card (X24-3098) get throughput per minute or

Throughput per Minute = $\frac{60,000}{\text{ms per transaction}}$

3) Calculate job time:

Total Job Time = $\frac{\text{Volume of}}{\text{Throughput p}}$

Figure 29. Illustrative Timing Layout

Individual Function Times

Card Reading

READ CYCLES AND SPEEDS: 1442 CARD READ PUNCH

The 1442 Card Read Punch, Model 1, has a basic read cycle of 210 ms as shown in Figure 33. This provides for card reading at a rate of 285 cards per minute. However, if the read instruction for the next card to be read is given during the processing time available prior to the clutch latch point, the clutch will remain engaged and the 10 ms clutch pickup time will be eliminated, resulting in a cycle of 200 ms. Thus the reading rate is increased from 285 to 300 cards per minute.

In the illustrative diagram of the read cycle the fact that 126 ms has been allotted for the card read time implies that all 80 columns of the card are read, but this need not be the case. A group mark with a word mark in core storage terminates the actual reading, so that if, for example, 30 columns of the card were to be read, only 71 ms would be required for reading and clutch pickup time. The remaining portion of the read cycle (139 ms) is available for processing or overlapping with other input/output functions.

A similar situation exists for the 1442 Card Read Punch, Model 2.* The basic read cycle is 160 ms (375 cards per minute) as shown in Figure 30, but if the next read command precedes the clutch latch point, 10 ms can be saved as with the Model 1. The resulting cycle is then 150 ms or 400 cards per minute.

The card-reading time shown in the illustrative diagram is the 96 ms required to read an 80-column card; when fewer columns are read, a corresponding portion of this 96 ms may be overlapped or used for internal processing time.

When the shorter read cycles (200 ms or 150 ms) are programmed, the processor is interlocked for the 20 ms for Model 1 (or 15 ms for Model 2) after the clutch latch point, reducing the available process time by that amount. Figure 31 illustrates the relationship between columns read and the process time available for both models at the basic cycle times.

If the total read cycle exceeds 210 ms – that is, if the card read time and the computation or other time is greater than 210 ms – the cycle is increased by the extra milliseconds over 210 and the number of cards is reduced correspondingly. (For Model 2, substitute 160 ms for 210 ms in the above.)

* The 1442, Model 4, has the same read timing as the 1442, Model 2. The Model 4, however, has no punching capability.

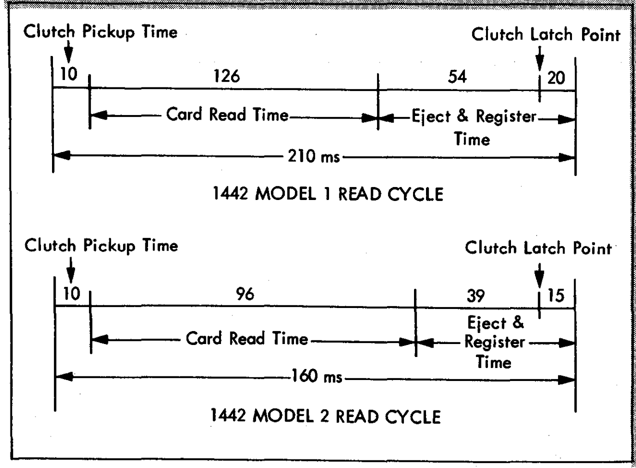


Figure 30. Read Cycles

	MODEL 1 at 285 cpm			MODEL 2 at 375 cpm*	
Cols. Read	Process or Overlap* Time (ms) Available	Card Read* Time (ms)	Cols. Read	Process or Overlap Time (ms) Available	Card Read*** Time (ms)
80	74	136	80	54	106
75	81	129	75	59	101
70	87	123	70	64	96
65	94	116	65	68	91
60	100	110	60	74	86
55	107	103	55	79	81
50	113	97	50	84	76
45	120	90	45	89	71
40	126	84	40	94	66
35	133	77	35	99	61
30	139	71	30	104	56
25	146	64	25	109	51
20	152	58	20	114	46
15	159	51	15	119	41
10	165	45	10	124	36
5	172	38	5	129	31
1	177	33	1	133	27

* Card Read Time (T) = 32.4 + 1.3 (no. of columns read) (including clutch pickup time)
** Process or Overlap Time = 210 - T
*** Card Read Time (T) = 26.1 + 1.01 (no. of columns read) (including clutch pickup time)
**** Process or Overlap Time = 160 - T
(When 300 CPM or 400 CPM cycles are used subtract 20 ms for process time available.) See Figure 48 for a more detailed table of read time.

Figure 31. Card Read Times

Card Punching

PUNCHING CYCLES AND SPEEDS: 1442 CARD READ PUNCH

The 1442 Card Read Punch, Model 1, is capable of punching at the rate of 80 columns per second, or 12.5 ms per column. The Model 2 speed is 160 columns per second (6.25 ms per column) – twice that of the Model 1. The time required for one complete punch cycle is the actual punching time plus the card movement time (210 ms on Model 1, 160 ms on Model 2) needed to move the next card to the punch station after one card is punched. When a punch-and-feed command is given, the card being punched is released immediately after punching is completed and the next card then begins moving to the punch station. Consequently, the fewer columns punched, the shorter the punch cycle. Blank columns and fields between those to be punched require the same time as punching, since it can be considered that the 1442 "punches" blanks. Thus the punching speed of the 1442 is a function of the last column punched plus card movement time. This is expressed by the formulas for the calculation of punching time shown below and in the timing layouts shown in Figure 32.

Punch Time (Mod. 1) = (Last Col. Punched $\times$ 12.5 ms) + 210
Punch Time (Mod. 2) = (Last Col. Punched $\times$ 6.25 ms) + 160

With the optional Punch Column Skip Feature the time required for spacing over blank columns can be used for processing.

Figure 33 shows the punching time and the cards per minute for various numbers of columns punched. Note that the card movement time can be used for processing. Since the movement time is the same as a card read cycle, card reading can frequently be overlapped with this portion of the punch cycle. Also note the effect of card design on throughput: punching columns 1-5 (no other columns punched) requires 272.5 ms on a Model 1 Read Punch; punching columns 76-80 requires 1,000 ms.

The times shown in Figure 33 also hold for processing runs that involve punching into the card just read, as long as read and process time is kept within the available processing time at normal cycle speed (1,210 ms or 160 ms). If the time required for reading and processing the card prior to punching it exceeds the available time, the additional milliseconds must be added to the cycle time.

PUNCHING CYCLE AND SPEED: 1444 CARD PUNCH

The 1444 Card Punch operates in parallel fashion at a rate of 250 cycles per minute, or one cycle every 240 ms. Of the 240 ms, the first 37 are for punch start time, the next 181 are for punching, and the remaining 22 can be used for processing.

For maximum punching speed in a continuous punching application, the next punch instruction must be given within the 22 ms of process time; if not, 60 ms will occur before the start of the next punching cycle.

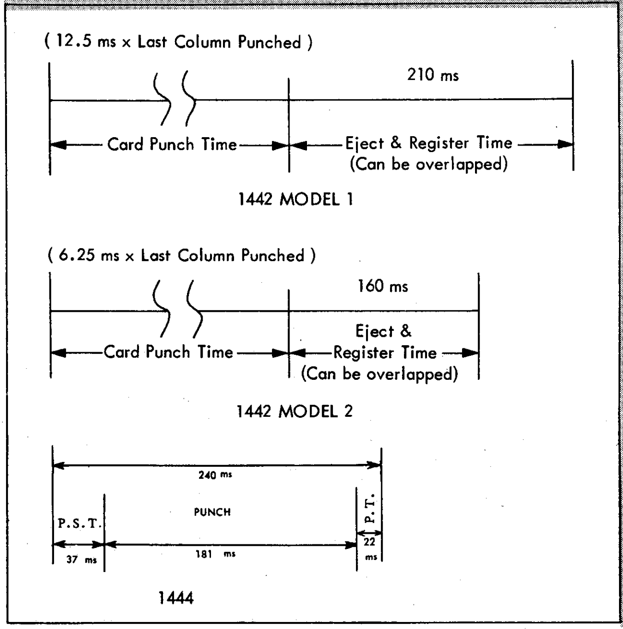


Figure 32. Punch Cycles (Punch and Feed Instruction Used)

	MODEL 1 *			MODEL 2 **		
Last Column Punched	Punch Time**** (ms)	Total Time (ms)	**** cpm	Punch Time*** (ms)	Total Time (ms)	**** cpm
80	1000	1210	50	500	660	91
75	937.5	1147.5	52	468.75	628.75	96
70	875	1085	55	437.5	597.5	100
65	812.5	1022.5	59	406.25	566.25	106
60	750	960	63	375	535	112
55	687.5	897.5	67	343.75	503.75	119
50	625	835	72	312.5	472.5	127
45	562.5	772.5	78	281.5	441.25	136
40	500	710	85	250	410	146
35	437.5	647.5	93	218.75	378.75	159
30	375	585	103	187.5	347.5	173
25	312.5	522.5	115	156.25	316.25	189
20	250	460	130	125	285	210
15	187.5	397.5	151	93 75	253.75	237
10	125	335	180	62.5	222.5	270
5	62.5	272.5	221	31.25	191.25	314
1	12.5	222.5	270	6.25	166.25	361

* 210 ms of process time available during card movement time
** 160 ms of process time available during card movement time
*** Punch Time = 12.5 (last column punched) (Model 1)
" = 6.25 (last column punched) (Model 2)

**** CPM = $\frac{60,000}{\text{Punch Time}}$

Figure 33. Card Punch Times

Printing

CHARACTER SETS AND PRINT CYCLES

The 1443 Printer is operator-adjustable for optimum performance on each report being printed. This is made possible by changeable typebars containing different sets of type and the optional Selective Character Set feature. Figure 34 shows the print cycles for the four character sets and the two printer models.

The 24 ms available at the end of the cycle permits the spacing or skipping of two lines. Each additional line skipped or spaced requires an additional 10 ms. The time required for the additional lines must be allowed for in the overall cycle time.

The optional Print Storage feature permits an overlap during most of the print cycle. When the feature is installed, the execution time for a print instruction is 2.4 ms. This time includes the instruction time and the time required by the processing unit to enter the data to be printed into the special print storage positions. The remainder of the print storage cycle involves sending data to the printer from the special print storage positions in core storage. This time can be overlapped with other processing or input/output operations. Note that 2.4 ms is the total time required, regardless of whether 120 or 144 positions are printed.

The process overlap time available with print storage is the difference between 2.4 ms and the corresponding cycle times for the different character sets. For example, for the 52-character set the overlap equals 400 — 2.4 ms. Figure 35 shows the overlap available with the four character sets and the two models of the printer.

CONSOLE PRINTER TIME

The optional 1440 Console Printer (available as a buffered or unbuffered unit) serves as both input and output for the 1440 system. The timing of the unbuffered console printer is:

Input
$T = .0111(L_I + 1) + \text{Operator Keying Time}$
Output
$T = .0111(L_I + 1) + 68\,(L_B) + 800\,(CR-1)$
Where LI = Length of the instructions.
LB = No. of characters printed and spaces skipped.
CR = Number of carriage returns.

The only possibility of overlap time with the unbuffered console in either of these operations comes on the last carriage return, which signals the end of the respective operations. Thus, the time needed to print out two lines of 50 characters each is:

$$\begin{aligned} .0111 \times 8 &= .088 \text{ ms} \\ 68 \times 100 &= 6800.000 \text{ ms} \\ 1 \times 800 &= \underline{800.000 \text{ ms}^*} \\ & 7600.088 \text{ ms} \end{aligned}$$

* Can be overlapped by processing.

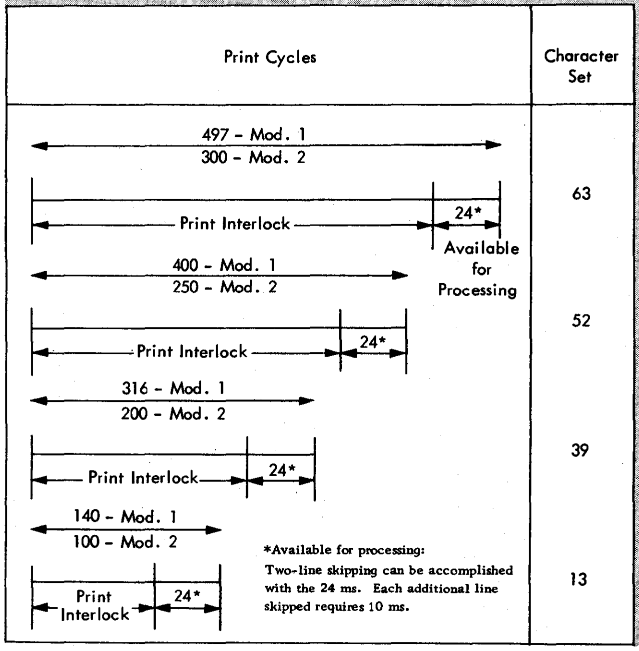


Figure 34. Print Cycles

Char. Set	Model 1		Model 2		Available Computing Time Without Print Storage Models 1 & 2	Available Overlap with Print Storage	
	Lpm	Print Cycle	Lpm	Print Cycle		Model 1	Model 2
13	430	140	600	100	24	137.6	97.6
39	190	316	300	200	24	313.6	197.6
52	150	400	240	250	24	397.6	247.6
63	120	497	200	300	24	494.6	297.6

Figure 35. Printing Speeds

LISTING

The speeds for single- or double-spaced card listings where all 80 columns of the card are read are shown in Figure 36 for both model printers, with and without Print Storage. Where less than 80 columns are read the speed would increase up to the maximum rated printing speeds. Figure 37 shows the timing layout charts used to arrive at the listing speed for the Model 1 printer using the 52-character typebar.

Note, in the timing layout shown, that the print cycle with the 52-character typebar is 400 ms and the read cycle of a 1442, Model 1, is 210 ms. Since the read cycle is considerably shorter than the print cycle, the optional Print Storage feature permits overlapping the entire reading operation with the printing. This allows the 1443 to run continuously at its maximum printing rate – in this example 150 lines per minute.

The Print Storage feature can also be used with another approach to listing data from cards. Suppose that two or three cards are read, and the information from these cards is printed on one line. This would make it possible, with some of the character sets, to overlap multiple card-reading cycles with one print cycle. With a 63-character typebar, for example, two cards can be read on a Model 1 card read punch within the 497-ms print cycle. This allows a listing of data from these cards on a two-up basis at a speed of 240 cards per minute. While it is not always possible or desirable to list two or three cards on one line, where applicable the increase in listing speeds makes this an approach worthy of consideration.

GROUP PRINTING

Figure 38 shows the speed for three typical group-printing jobs in conjunction with the Model 1 and 2 card read punches.

SUMMARY

From the above discussion it can be seen that printing speeds are affected by numerous considerations: the number of characters required, the model read-punch installed, the use of the optional print storage feature, the possibility of listing the data from two cards on one line, etc. Other factors affecting printing throughput are whether forms can be printed side by side, and whether the information is printed from the disk file or from cards. It is therefore important to analyze the various considerations when estimating printing times and designing 1440 systems approaches.

Type of Listing	Card Read Punch Model	Print Storage (X)	Lines-Per-Minute			
			63	52	39	13
Reading 80 columns:printing one single or double spaced line	1		98	117	140	239
"	2		104	124	152	273
"	1	X	120	150	190	300
"	2	X	120	150	190	400

Figure 36. Card Listing Speeds—1443 Printer, Model 1 (Reading 80 Columns and Printing a Line)

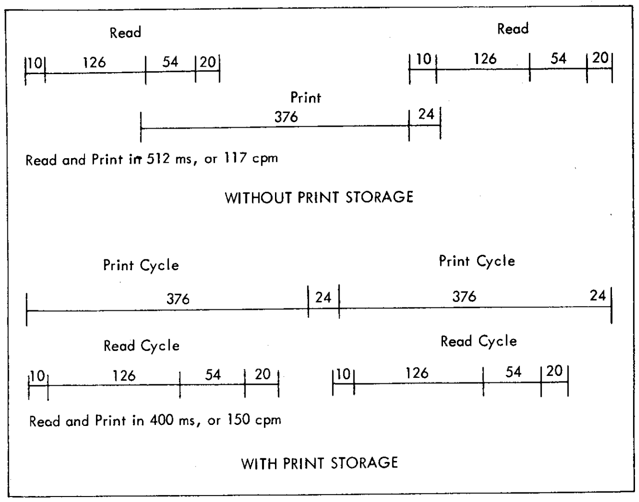


Figure 37. Listing with 1443 Printer, Model 1, 52 Character Set and 1442 Card Read Punch, Model 1

Type of Control	Card Read Punch Model	Cards-Per-Minute			
		63	52	39	13
Minor after 10 cards	1	249	260	269	293
Mi, Int after 20 cards	1	245	254	264	286
Mi, Int Ma after 40 cards	1	255	263	271	288
Mi after 10 cards	2	311	328	343	375
Mi, Int after 20 cards	2	307	323	337	370
Mi, Int, Ma after 40 cards	2	324	336	349	377

Figure 38. Group Printing Speeds with 1443 Printer, Model 1

Disk Storage Timing

The following factors are involved in timing disk operations:

Seek time — the time required for the movement of the access arms from one cylinder to another.

Rotational delay — the time required for the desired record to rotate to the read/write head after the previous function has been completed (usually seeking or processing).

Read, write and write-checking time — the time required for the reading, writing and write-checking of the disk record. (In certain file techniques disk read time is also required to locate a record; see page 30.)

Process time — the time required to process the record.

Head select time — a head select time of 2 ms is involved whenever a disk read, write or write-check instruction is accessed. This is the time required for the head circuitry to prepare for transferring data bits.

Scan disk time — the time required to scan a portion of a file when the optional Scan Disk feature is used to locate a disk record.

These factors will be clarified below and are also discussed in the file organization section of the manual.

SEEKING DISK STORAGE RECORDS

Two modes of operation for seek instructions are "Return to Home" and "Direct Seek". The first is the standard mode of operation. In the return-to-home mode all seeks are achieved by moving the access arms to a home position outside cylinder 00 and then counting into the desired cylinder. This function is automatically performed by the system. Direct Seek is an optional special feature which, with appropriate programming, permits the access arms to move from one track to another without first requiring the arms to return to the home position.*

After a seek instruction in either mode has been issued, processing may continue until another disk storage instruction is issued. The length of the seek will depend on the total number of cylinders that must be passed during the seek operation.

Figure 39 shows the seek time from cylinder to cylinder without direct seek. Note that the minimum seek time is 75 ms and the maximum 392 ms. Figure 40 shows the seek time, with the optional Direct Seek feature, based on the number of cylinders traveled. The minimum seek time in this case is 54 ms, the maximum 248.

Seek time varies within these ranges as a function of the file organization, the number of disk records in the file, the type of processing (random or sequential) and the distribution of transactions. Random processing of a randomly organized file will probably involve the greatest amount of seek time; sequential processing of a sequential file, probably the least. In estimating seek time it is therefore necessary to consider the specifics of a particular approach.

TO ↓	FROM										
	00	09	19	29	39	49	59	69	79	89	99
00	75	88	101	114	127	140	153	167	179	192	204
09	175	188	201	214	227	240	253	267	279	292	304
19	143	156	169	182	195	208	221	235	247	260	272
29	153	166	179	192	205	218	231	245	257	270	282
39	168	181	194	207	220	233	246	260	272	285	297
49	184	197	210	223	236	249	262	276	288	301	313
59	200	213	226	239	252	265	278	292	304	317	329
69	215	228	241	254	267	280	293	307	319	312	344
79	232	245	258	271	284	297	310	324	336	349	361
89	248	261	274	287	300	313	326	340	352	365	377
99	263	276	289	302	315	328	345	355	367	380	392

Figure 39. Cylinder Seek Time—without Direct Seek

Cylinders Traveled	Time in Milliseconds
1	54
2	67
3	80
4	90
5	105
6	115
7	130
8	140
9	155
10	165
20	130
30	137
40	154
50	170
60	185
70	202
80	217
90	235
99	248

Figure 40. Cylinder Seek Time—with Direct Seek

* This feature reduces access time by allowing an access mechanism to be repositioned directly to a new setting without first having returned to the home position. A program-generated number that is equal to twice the cylinder diffeıence between the new address and the previous address controls the feature.

ROTATIONAL DELAY AND READING AND WRITING TIME

After the access mechanism is positioned on the cylinder, the read or write instruction given will select one of the ten read-write heads and begin comparing addresses until the desired record is found. It is unlikely that the record will be under the head immediately; a short delay is normally necessary while the head waits for the record to rotate to it. If the address is just coming under the head, the wait time is 0. If the first character of the address just passed under the head, the wait time is 40 ms. This time is referred to as rotational delay and averages 20 ms, the equivalent of half a revolution of the disk. After a record is found, reading or writing or write-checking can take place at 2 ms per sector. After a write operation, a rotational delay of up to 38 ms (depending on number of sectors read) is required until the record is available for the write check. The rotational time can be used for processing.

For an example of timing for rotational delay and disk reading and writing, see Figure 41 showing the reading, updating and writing of a two-sector record (200 characters). If possible, processing should be kept within the available rotation time; otherwise the total cycle time will be increased by increments of 40-ms rotation time. Rotation time between the write and write-check operations can be used for such functions as updating control totals and arranging fields for printing.

SCAN DISK TIME

Through the use of the optional Scan Disk feature it is possible to search an entire cylinder in 422 ms (22 ms for rotational delay and head select time + 400 for scanning). After the desired record is located a read instruction is given to read it into core. If less than an entire cylinder is searched the time is reduced proportionally (for example, a 100-sector scan would require 222 ms).

DUMMY SEEK TECHNIQUE

Access motion time is composed of two operations: "Return to Home" and "Advance from Home". The first can normally be overlapped if a seek to cylinder 00 is issued before a card read or punch instruction or a print instruction as shown in Figure 42.

Figure 43 illustrates the reduction of total time by use of the dummy seek technique under the following conditions:

1. Processing of the present transaction is complete.
2. The access mechanism is located at cylinder 59.
3. The disk record for the next transaction is located in cylinder 79.
4. Eighty columns are read from the input card.

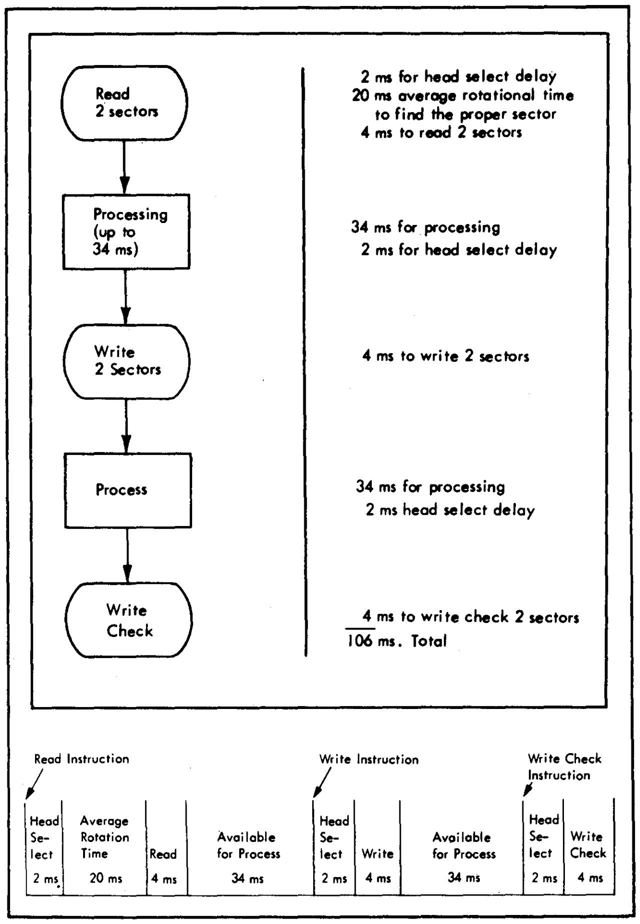


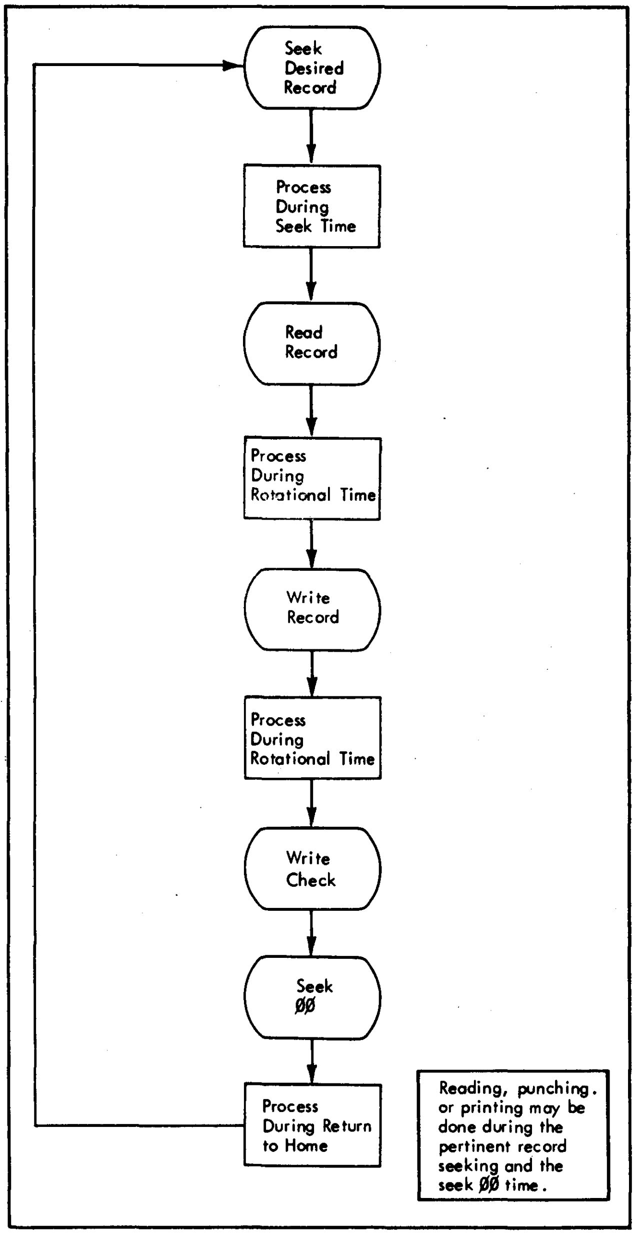
Figure 41. Disk Storage Timing for a Two-Sector Record

In this example the return-to-home time from cylinder 59 is virtually eliminated, and cycle time has been reduced from 451 to 368 ms. It should be noted that the access time for the next transaction will be from cylinder 00 to the home position and then to the required cylinder location. Figure 44 can be used to determine seek times when using the dummy seek technique.

SEEK OVERLAP

The seek overlap special feature allows a disk read, write, or scan operation to be overlapped with seek operations on other disk drives. Without this feature, two disk operations cannot be performed concurrently; it should be noted, however, that seek time and rotation time can be utilized for other operations that do not pertain to a disk drive.

* In Figure 44, note that 4 ms of the disk rotation time is used to read two sectors. This leaves 36 ms of rotation time before the record can again be accessed, 2 ms of which will be needed for head select time. The remaining 34 ms is available process time.



Seek Desired Record
Process During Seek Time
Read Record
Process During Rotational Time
Write Record
Process During Rotational Time
Write Check
Seek ØØ
Process During Return to Home
Reading, punching, or printing may be done during the pertinent record seeking and the seek ØØ time.

Figure 42. Block Diagram for Dummy Seek Technique

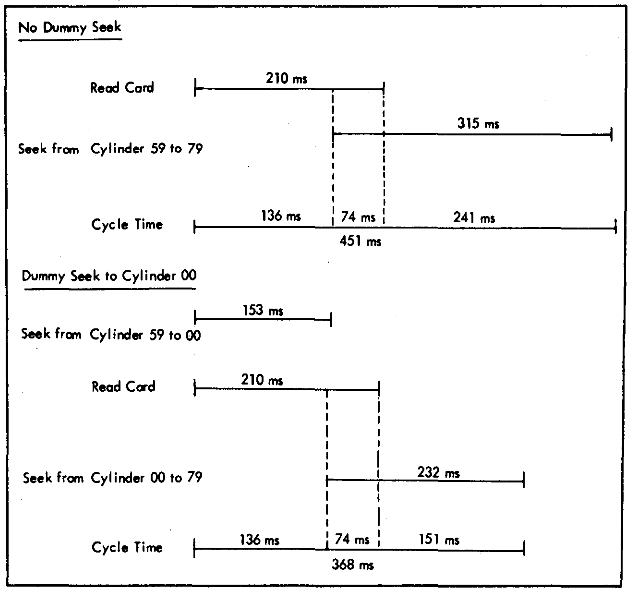


Figure 43. Throughput Time for Dummy Seek Technique

Cylinder	Return to 00 from: (in ms)	Advance from 00 to: (in ms)
00	75	75
09	88	175
19	101	143
29	114	153
39	127	168
49	140	184
59	153	200
69	167	215
79	179	232
89	192	248
99	204	263

Figure 44. Cylinder Seek Time for Dummy Seek Technique

Processing

A portion of the cycle time for any job is taken up by processing. In some cases all of the processing can be overlapped by input/output operations; in others the processing requires additional time. This should be considered in determining how precisely processing time must be calculated for timing estimates.

A figure of four instructions per ms is often used to estimate processing time. This assumes an average instruction length, data field and mixture of instructions. To make closer processing timing estimates, it is necessary to consider the individual instructions used and the number of data characters involved in each operation. One approach that can be used is:

- Develop a general block diagram for the mainline of the run.
- Define the operations performed in each block.
- Determine the number and type of instructions required to accomplish the operations in the block and the length of the data fields.
- Using the formulas listed in Figure 45, calculate the time required to perform the operations.

Instruction	Timing (ms)	Notes
Add (one address)	$.0111X(L1+1+2L_A)$	
Add	$.0111X(L1+1+L_A+L_B)$	No recomplement
Add	$.0111X(L1+1+L_A+3L_B)$	Recomplement (without multiply divide special feature)
Add	$.0111X(L1+1+L_A+2L_B)$	Recomplement (with multiply divide special feature)
Branch Unconditional	.0111X(L1+1)or**2	
*Branch If Bit Equal	.0111X(L1+2)or**3)	
Branch If Character Equal	.0111X(L1+2)or**3)	
Branch If Indicator On	.0111X(L1+1)or**2)	
Branch If WM and/or Zone	.0111X(L1+2)or**3)	
Clear Storage	$.0111X(L1+1+L_X)$	
Clear Storage and Branch	$.0111X(L1+1+L_X)or^{**}2+L_X)$	
Clear Word Mark	.0111X(L1+3)	
Clear Word Mark (one address)	.0111X(L1+3)	
Compare	$.0111X(L1+1+2L_W)$	
Control Carriage	.0111X(L1+1)	Plus remaining form-movement time, if carriage is moving when instruction is given.
*Divide	$.0111X(L1+2+7L_R L_Q+8L_Q)$	
Halt	.0111X(L1+1)	
Halt and Branch	.0111X(L1+1)or**2	
Load Characters	$.0111X(L1+1+2L_A)$	
Load Characters (one address)	$.0111X(L1+1+2L_A)$	
*Modify Address	.0111X(L1+9)or***7)	
*Modify Address (one address)	.0111X(L1+9)or***7)	
Move Characters	$.0111X(L1+1+2L_W)$	
Move Characters (one address)	$.0111X(L1+1+2L_A)$	
Move Characters and Edit	$.0111X(L1+1+L_A+L_B+L_Y)$	Without zero suppression
Move Characters and Edit	$.0111X(L1+1+L_A+L_B+L_Y)$	With zero suppression
Move Characters and Suppress Zeros	$.0111X(L1+1+3L_A)$	
Move Numerical	.0111X(L1+3)	
Move Record	$.0111X(L1+1+2L_A)$	L_A (in A Register) includes GM/WM or RM
Move Zone	.0111X(L1+3)	
*Multiply	$.0111X(L1+3+2L_C+5L_C L_M+7L_M)$	Average time
No Operation	.0111X(L1+1)+"Index Cycles" if necessary.	
#Print	.0111X(L1+1)	

Figure 45. Instruction Times

(continued on next page)

Instruction	Timing (ms)	Notes
#Punch Card	.0111X(L1+1)	
#Punch and Feed	.0111X(L1+1)+12.5(L_B)ms	
#Read Card	.0111X(L1+1)	
#Read Console Printer	.0111X(L1+1)	Plus operator keying time.
Read Track Sectors w-Addresses	.0111X(L1+1)+62ms	Assumes sector before index pulse is used for address comparison
Read Sector Mode	.0111X(L1+1)+2N_S+22ms	Assumes no cylinder overflow.
Read Sector Count Overlay	.0111X(L1+1)+2N_S+22ms	Assumes no cylinder overflow.
Set Word Mark	.0111X(L1+3)	
Set Word Mark (one Address)	.0111X(L1+3)	
*Stacker Select	.0111X(L1+1)	
*Store A Address Register	.0111X(L1+5)	
Store B Address Register	.0111X(L1+4)	
Subtract No recomplement	.0111X(L1+1+L_A+L_B)	
Subtract (without multiply - divide special feature)	.0111X(L1+1+L_A+3L_B)	Sign of result changed and stored in true form.
Subtract (with multiply - divide special feature)	.0111X(L1+1+L_A+2L_B)	
Subtract (one Address)	.0111X(L1+1+2L_A)	
Write Console Printer	.0111X(L1+1)+68(L_B)+800x (CR-1)	CR=Number of the returns of the Print element.
Write Track Sectors w/Addresses	.0111X(L1+1)+62ms	Assumes sector before index pulse is used for address comparison
Write Sector Mode	.0111X(L1+1)+2n_S+22ms	Assumes no cylinder overflow.
Write Sector Count Overlay	.0111X(L1+1)+2N_S+22ms	Assumes no cylinder overflow.
Zero and Add	.0111X(L1+1+L_A+L_B)	
Zero and Subtract	.0111X(L1+1+L_A+L_B)	
Zero and Subtract (one Address)	.0111X(L1+1+2L_A)	

* Special Feature
** If Store Address Register Feature is installed.
*** Applies when a carry from the hundred's zone to the unit's zone occurs
# See text for timing of input/output cycle.

Key to abbreviations used in formulas:

L_A = Length of the A field	L_W = Length of A or B field, whichever is shorter	A = A address of instruction
L_B = Length of the B field	L_X = Number of characters to be cleared	B = B address of instruction
L_C = Length of Multiplicand field	L_Y = Number of characters back to rightmost zero in control field	Ap = Previous setting of A-address register
L_I = Length of Instruction	L_Z = Number of zeros inserted in a field	Bp = Previous setting of B-address register
L_M = Length of Multiplier field	I/O = Timing for Input or Output cycles	X = Thousands and tens of starting address
L_Q = Length of Quotient field	Fm = Forms movement times	NS = Number of sectors
L_R = Length of Divisor field	BI = Next Instruction if branch occurs	SS = Size of sections (100 or 90,
L_S = Number of significant digits in Divisor (excludes high-orde· zeros and blanks)		ms = Milliseconds

Instruction Times (Continued)

Timing of Utility Programs

Figure 46 gives the approximate times required to process one full disk pack using the disk utility programs (see *Disk Utility Programs for* IBM *1440/1311—Preliminary Specifications*, C24-3002, for details of these programs):

Clear Disk Storage			
Same Address	5.5 minutes		
New Address	3.1 minutes		
	1442	1442	
Disk-to-Card	Model 1	Model 2	(Note 1)
Move Mode	622 minutes	342 minutes	
Load Mode	782 minutes	432 minutes	
	1442	1442	
Card-to-Disk	Model 1	Model 2	(Note 1)
	111 minutes	85 minutes	
	139 minutes	106 minutes	
Copy Disk			(Note 2)
Minimum	4.3 minutes		
Maximum	7.0 minutes		
	1443	1443	
Print Disk	Model 1	Model 2	(Note 3)
	142 minutes	88 minutes	
	293 minutes	184 minutes	

Note 1: The time required to load or unload a disk pack in the load mode using the card programs depends on the frequency of word marks in the file. The times given here are for the best case, where each card contains 50 data characters.

Note 2: The time required for the copy-disk program depends on the relative positions of the index points on the two disk drives.

Note 3: The special character substitution option requires an additional 25 minutes if the print storage feature is not available. No additional time is required if the feature is available.

Figure 46. Timing of Utility Programs

Maximizing Throughput

To insure that throughput is maximized for the particular functions performed on a run, several points must be considered. Some of the major ones are listed below and are discussed in the following paragraphs:

- Is the program taking advantage of all overlapping possibilities?
- Would the redesign of input or output forms affect throughput?
- Can the file organization be changed to increase throughput?
- Should system components be changed to effect an increase in throughput?
- Is the program designed to minimize setup time?

Overlapping.—Overlap occurs when two or more operations take place during the same period of time. For instance, a seek instruction, once initiated, does not interlock the processing unit, and, while the access mechanism is moving to the addressed cylinder, the processor can be doing arithmetic, data moving, or other than disk I/O operations. The card read, punch and feed, and write a line operations also have portions of their cycles that may be overlapped.

After the operations are laid out on the timing layout chart, the chart should be used as an aid in evaluating overlapping possibilities. Each operation can be examined for available process time. The chart should show another operation initiated and overlapping any available time.

If all available time is not being used, more efficiency may be gained by changing the sequence of operations. An example of this would be in a random disk operation where there is a card read, a disk seek and a line printed after some processing. Instead of a program that reads, seeks, processes and prints one record at a time, a program could be written so that the next card could be read during the seek for the previous one. This would result in the overlap of all or practically all of the card read time.

Input/output redesign.—Can redesigning the input cards help the timing? If the data to be read can be put in the first half of the card, the remainder of the read time can be used for processing. Perhaps changing the location of the punch fields will shorten the overall cycle. If the printed output can be changed from one-up to two-up, throughput can be increased. Some applications may be run three-up at substantial gains in throughput. Inherent in these changes of output format are other programming changes which may affect the logic flow of the entire system. They must be fully examined before deciding on the change.

File organization.—File organization can greatly influence throughput and should be carefully analyzed on the basis of the considerations discussed in the previous sections on file organization and timing.

Component changes.—An increase in throughput may warrant component changes such as adding the optional Print Storage feature to allow almost complete overlap of the print operation; or providing additional print positions to allow side-by-side printing of certain forms; or, on multiple-drive systems, adding the Seek Overlap feature to allow seeking on one drive while a disk read, write, or write check operation is taking place on another drive.

Minimizing setup time.—Setup time can be divided into five phases:

a. Program loading
b. Resetting machine indicators and sense switches
c. Setting up the printer
d. Loading disk drives
e. Preparing the card read punch

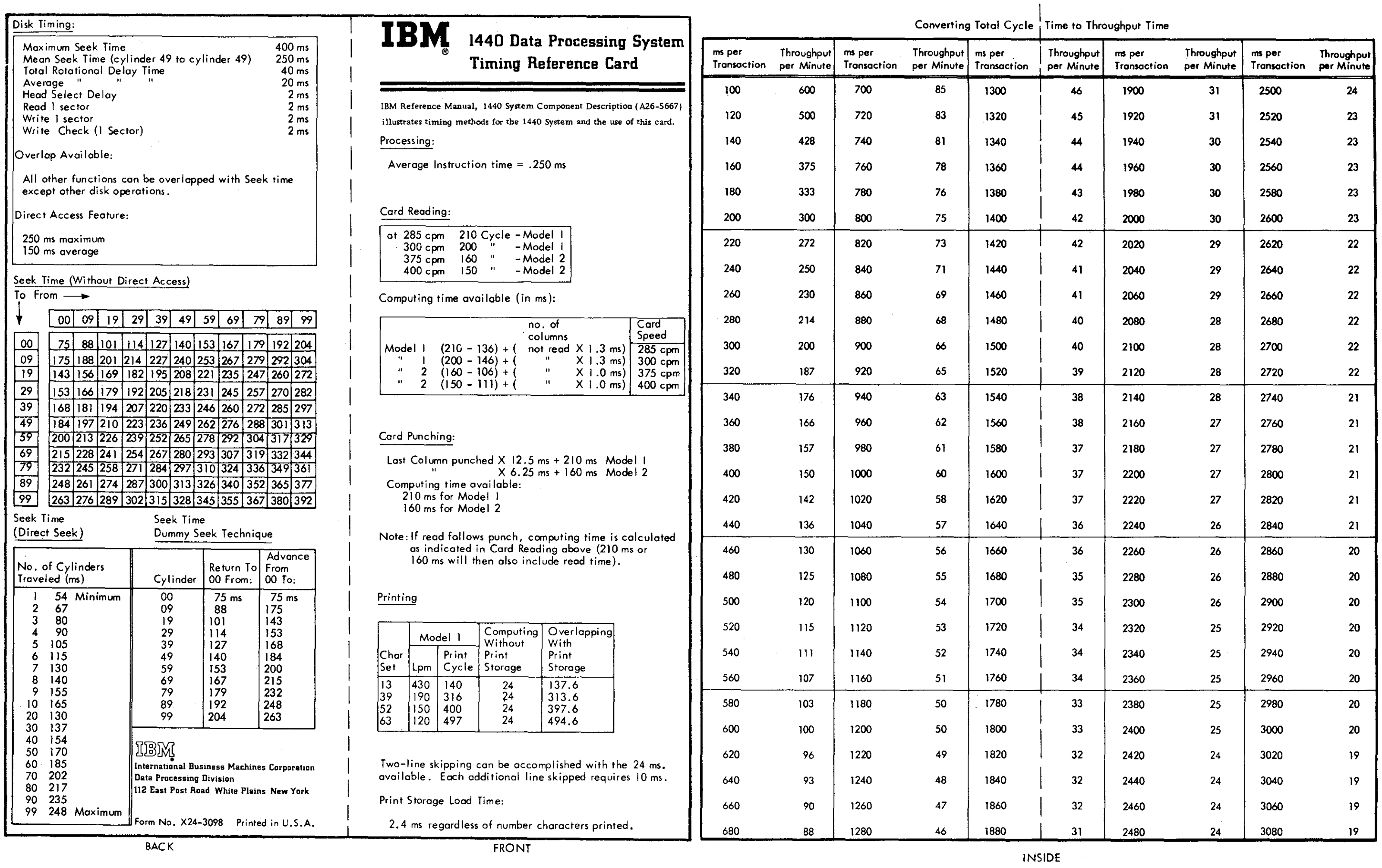
Here are some suggestions to effect reductions in total setup time:

1. Use blank paper and produce page headings under program control. This may be accomplished in the housekeeping section of the program.
2. Use a small supervisory routine to control the loading of programs and to simulate the last-card switch. (This routine would be used in conjunction with the first suggestion.) Machine indicators can be set or reset in the run-to-run control program. Sense switches can be simulated by reserving an area of perhaps ten positions in storage. Each program would load a constant card to set the switch positions at 1 to signify ON, or blank to signify OFF. These positions can then be tested in the program in the same way as the sense switches. (This is also a good systems approach, because it reduces the chance of operator error.) The alteration switches on the console should be used for control during the operation of the program. The functions that they control should be common from one program to another (for example, to control the operation of a disk-error procedure).
3. Use multi-format cards or stock 5081 cards whenever possible to reduce punch setup.
4. Load the card reader for the next program during running of the previous program. The supervisory routine can control last-card and program-loading functions.
5. In a multi-drive 1311 system, time can be saved if the next sequential pack is placed on an unused drive. Flip-flopping of packs can then eliminate pack change time. This approach is particularly valuable where sequential file organization is used and the file exceeds three packs.
6. Store the programs on the file. The correct program can be loaded under control of a supervisory routine, and processing can automatically start. The remainder of the setup can be handled as in the numbered recommendations above.

The above techniques can improve the daily output of a 1440 system by decreasing setup time. Also, by reducing operator intervention, the chances of setup error are reduced.

Timing Summary

Figure 47 presents a timing summary containing the cycle times for the various functions and a table for converting cycle time to throughput. Figure 48 is a detailed table of card-reading times. The basic reading, punching and printing cycles are shown together in Figure 49 for ease of reference. Figure 50 is a summary of seek times.



Disk Timing:

Maximum Seek Time	400 ms
Mean Seek Time (cylinder 49 to cylinder 49)	250 ms
Total Rotational Delay Time	40 ms
Average " " "	20 ms
Head Select Delay	2 ms
Read 1 sector	2 ms
Write 1 sector	2 ms
Write Check (1 Sector)	2 ms

Overlap Available:

All other functions can be overlapped with Seek time except other disk operations.

Direct Access Feature:

250 ms maximum
150 ms average

Seek Time (Without Direct Access)

To ↓ From ⟶

	00	09	19	29	39	49	59	69	79	89	99
00	75	88	101	114	127	140	153	167	179	192	204
09	175	188	201	214	227	240	253	267	279	292	304
19	143	156	169	182	195	208	221	235	247	260	272
29	153	166	179	192	205	218	231	245	257	270	282
39	168	181	194	207	220	233	246	260	272	285	297
49	184	197	210	223	236	249	262	276	288	301	313
59	200	213	226	239	252	265	278	292	304	317	329
69	215	228	241	254	267	280	293	307	319	332	344
79	232	245	258	271	284	297	310	324	336	349	361
89	248	261	274	287	300	313	326	340	352	365	377
99	263	276	289	302	315	328	345	355	367	380	392

Seek Time (Direct Seek)

No. of Cylinders	Traveled (ms)	
1	54	Minimum
2	67	
3	80	
4	90	
5	105	
6	115	
7	130	
8	140	
9	155	
10	165	
20	130	
30	137	
40	154	
50	170	
60	185	
70	202	
80	217	
90	235	
99	248	Maximum

Seek Time Dummy Seek Technique

Cylinder	Return To 00 From:	Advance From 00 To:
00	75 ms	75 ms
09	88	175
19	101	143
29	114	153
39	127	168
49	140	184
59	153	200
69	167	215
79	179	232
89	192	248
99	204	263

IBM
International Business Machines Corporation
Data Processing Division
112 East Post Road White Plains New York

Form No. X24-3098 Printed in U.S.A.

BACK

IBM® 1440 Data Processing System Timing Reference Card

IBM Reference Manual, 1440 System Component Description (A26-5667) illustrates timing methods for the 1440 System and the use of this card.

Processing:

Average Instruction time = .250 ms

Card Reading:

at 285 cpm	210 Cycle	- Model 1
300 cpm	200 "	- Model 1
375 cpm	160 "	- Model 2
400 cpm	150 "	- Model 2

Computing time available (in ms):

		no. of columns		Card Speed
Model 1	(210 - 136) + (	not read	X 1.3 ms)	285 cpm
" 1	(200 - 146) + (	"	X 1.3 ms)	300 cpm
" 2	(160 - 106) + (	"	X 1.0 ms)	375 cpm
" 2	(150 - 111) + (	"	X 1.0 ms)	400 cpm

Card Punching:

Last Column punched X 12.5 ms + 210 ms Model 1
" X 6.25 ms + 160 ms Model 2
Computing time available:
210 ms for Model 1
160 ms for Model 2

Note: If read follows punch, computing time is calculated as indicated in Card Reading above (210 ms or 160 ms will then also include read time).

Printing

	Model 1		Computing Without Print Storage	Overlapping With Print Storage
Char Set	Lpm	Print Cycle		
13	430	140	24	137.6
39	190	316	24	313.6
52	150	400	24	397.6
63	120	497	24	494.6

Two-line skipping can be accomplished with the 24 ms. available. Each additional line skipped requires 10 ms.

Print Storage Load Time:

2.4 ms regardless of number characters printed.

FRONT

Converting Total Cycle Time to Throughput Time

ms per Transaction	Throughput per Minute	ms per Transaction	Throughput per Minute	ms per Transaction	Throughput per Minute	ms per Transaction	Throughput per Minute	ms per Transaction	Throughput per Minute
100	600	700	85	1300	46	1900	31	2500	24
120	500	720	83	1320	45	1920	31	2520	23
140	428	740	81	1340	44	1940	30	2540	23
160	375	760	78	1360	44	1960	30	2560	23
180	333	780	76	1380	43	1980	30	2580	23
200	300	800	75	1400	42	2000	30	2600	23
220	272	820	73	1420	42	2020	29	2620	22
240	250	840	71	1440	41	2040	29	2640	22
260	230	860	69	1460	41	2060	29	2660	22
280	214	880	68	1480	40	2080	28	2680	22
300	200	900	66	1500	40	2100	28	2700	22
320	187	920	65	1520	39	2120	28	2720	22
340	176	940	63	1540	38	2140	28	2740	21
360	166	960	62	1560	38	2160	27	2760	21
380	157	980	61	1580	37	2180	27	2780	21
400	150	1000	60	1600	37	2200	27	2800	21
420	142	1020	58	1620	37	2220	27	2820	21
440	136	1040	57	1640	36	2240	26	2840	21
460	130	1060	56	1660	36	2260	26	2860	20
480	125	1080	55	1680	35	2280	26	2880	20
500	120	1100	54	1700	35	2300	26	2900	20
520	115	1120	53	1720	34	2320	25	2920	20
540	111	1140	52	1740	34	2340	25	2940	20
560	107	1160	51	1760	34	2360	25	2960	20
580	103	1180	50	1780	33	2380	25	2980	20
600	100	1200	50	1800	33	2400	25	3000	20
620	96	1220	49	1820	32	2420	24	3020	19
640	93	1240	48	1840	32	2440	24	3040	19
660	90	1260	47	1860	32	2460	24	3060	19
680	88	1280	46	1880	31	2480	24	3080	19

INSIDE

Figure 47. Timing Reference Card

Model 1						Model 2				
Cols Read	Process Time Available[1]		Start and Read Time[2]		Read Time[3]	Process Time Avail.[4]		Start and Read Time[5]		Read Time[6]
	At 285 cpm	300 cpm	285 cpm	300 cpm		385 cpm	400 cpm	385 cpm	400 cpm	
80	74	54	136	146	126.3	54	39	106	111	96
75	81	61	130	140	119.8	59	44	101	106	91
70	87	67	123	133	113.3	64	49	96	101	86
65	94	74	116	126	106.8	69	54	91	96	81
60	100	80	110	120	100.3	74	59	86	91	76
55	107	87	103	113	93.8	79	64	81	86	71
50	113	93	97	107	87.3	84	69	76	81	66
45	120	100	90	100	80.8	89	74	71	76	61
40	126	106	84	94	74.3	94	79	66	71	56
35	133	113	77	87	67.8	99	84	61	66	51
30	139	119	71	81	61.3	104	89	56	61	46
25	146	126	64	74	54.8	109	94	51	56	41
20	152	132	58	68	48.3	114	99	46	51	36
15	159	139	51	61	41.8	119	104	41	46	31
10	165	145	45	55	35.3	124	109	36	41	26

1 Processing Time Available = 210 - Start and Read Time
200 - Start and Read Time

2 Start and Read Time = $.0111(L_I+1)+10+\left[21+1.3(L_B+1)\right]$

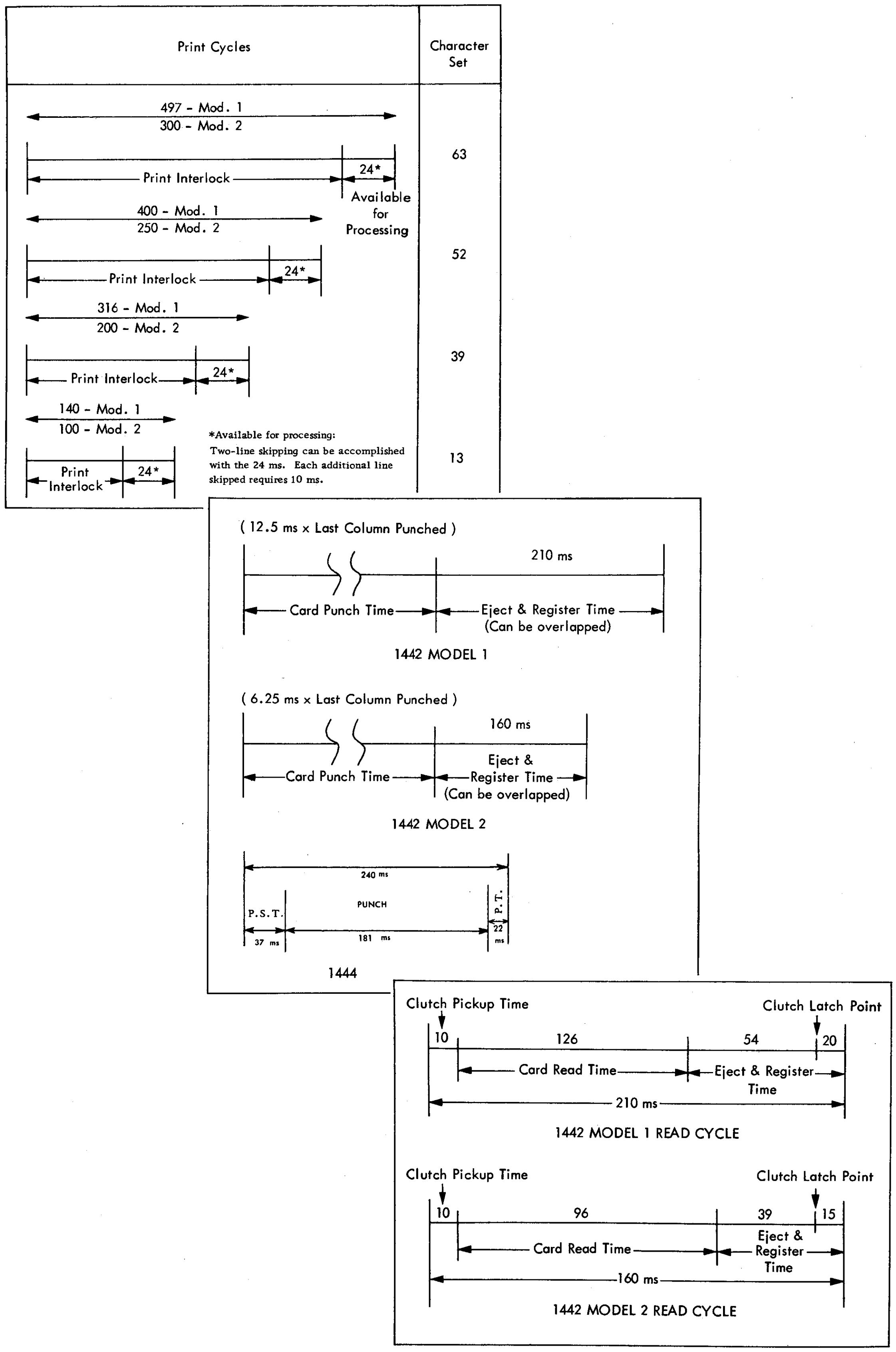
3 Read Time = $(21+1.3\left[L_B+1\right])$

4 Processing Time Available = 160 - Start and Read Time
150 - Start and Read Time

5 Start and Read Time = $.0111(L_I+1)+10+\left[15+1.0(L_B+1)\right]$

6 Read Time = $(15+1.0\left[L_B+1\right])$

Figure 48. Card Reading Times



*Available for processing:
Two-line skipping can be accomplished with the 24 ms. Each additional line skipped requires 10 ms.

Figure 49. Summary of 1440 Machine Cycles

		FROM CYLINDER										
		00	09	19	29	39	49	59	69	79	89	99
Return to Home Advance from Home Total	00	23 52 75	36 52 88	49 52 101	62 52 114	75 52 127	88 52 140	101 52 153	115 52 167	127 52 179	140 52 192	152 52 204
	09	23 152 175	36 152 188	49 152 201	62 152 214	75 152 227	88 152 240	101 152 253	115 152 267	127 152 279	140 152 292	152 152 304
	19	23 120 143	36 120 156	49 120 169	62 120 182	75 120 195	88 120 208	101 120 221	115 120 225	127 120 247	140 120 260	152 120 272
	29	23 130 153	36 130 166	49 130 179	62 130 192	75 130 205	88 130 218	101 130 231	115 130 245	127 130 257	140 130 270	152 130 282
	39	23 145 168	36 145 181	49 145 194	62 145 207	75 145 220	88 145 233	101 145 246	115 145 260	127 145 272	140 145 285	152 145 297
To Cylinder	49	23 161 184	36 161 197	49 161 210	62 161 223	75 161 236	88 161 249	101 161 262	115 161 276	127 161 288	140 161 301	152 161 313
	59	23 177 200	36 177 213	49 177 226	62 177 239	75 177 252	88 177 265	101 177 278	115 177 292	127 177 304	140 177 317	152 177 329
	69	23 192 215	36 192 228	49 192 241	62 192 254	75 192 267	88 192 280	101 192 293	115 192 307	127 192 319	140 192 312	152 192 344
	79	23 209 232	36 209 245	49 209 258	62 209 271	75 209 283	88 209 297	101 209 310	115 209 324	127 209 336	140 209 349	152 209 361
	89	23 225 248	36 225 261	49 225 274	62 225 287	75 225 300	88 225 313	101 225 326	115 225 340	127 225 352	140 225 365	152 225 377
	99	23 240 263	36 240 276	49 240 289	62 240 302	75 240 315	88 240 328	101 240 345	115 240 355	127 240 367	140 240 380	152 240 392

Figure 50. Summary of Seek Times

Documentation

A tremendous amount of information and knowledge goes into the planning and operation of a data processing installation. In the original systems study the pertinent information on an application is collected and analyzed; in subsequent procedure development a great deal of new material and information is generated and used. Ordinarily, many individuals are involved in this process and they cannot keep track of the myriad details unless these details are recorded in clear and orderly fashion. This means that the significant information must be written down and kept clear and concise. The creation and maintenance of these records — documentation — is an essential and vital part of a computer installation.

Purposes of Documentation

Effective documentation accomplishes a number of purposes:

- It clarifies the scope of the job, identifies the methods used, and indicates changes from previous operations.
- It aids in communication between data processing personnel and those in other departments.
- It serves as a necessary reference for the system and programming personnel who work on many different jobs over a long span of time.
- It provides flexibility in personnel assignment. Changes in personnel can be handled with a minimum of difficulty when the work has been well documented.
- It provides for communication between programmers and helps coordinate their efforts in related programs.
- It serves an important function in program testing and operation by providing the base information for other individuals working with the program.
- It facilitates program modification.
- It helps in recording and evaluating installation progress.

Thus adequate documentation is of prime importance in the planning and operation of an installation. Documentation standards should be defined early in the preinstallation period and firmly adhered to throughout the subsequent work.

Guides to Effective Documentation

In planning a system of documentation, it is important to determine just how much documentation is needed. Insufficient information may necessitate restudy of a job or complete reprogramming when modifications are required; too much documentation may become burdensome and inefficient. Some of the features of effective documentation are as follows:

Current working value. — The process of creating the documents and the documents themselves should be instrumental to and simultaneous with application and program development. If the documentation can be helpful to the data processing personnel in the working stages, it is far more likely to be used. Postponement of documentation often means that early considerations and decisions are forgotten; frequently, too, when a project has reached the operational stage, time is required for other tasks and the documentation is never completed.

Clarity and intelligibility. — To be of value, documentation should have maximum representation — that is, it should display the facts of a run in a clear manner. This would include showing the "big picture", emphasizing unique aspects, and providing a method by which all details of a run or program can be easily traced. Liberal use of diagrams, standardized symbols and programming techniques, consistent formats, and special forms are some of the devices which can assist in achieving this objective.

Ease of alteration.— Most programs are likely to change substantially during development and even after reaching the operational stage. An easy method of entering changes as they occur should be provided so that the documentation always reflects a true picture of the program. This helps not only in the making of revisions, but in the indoctrination of new personnel working with the program.

Organized filing. — A logical and orderly filing system should be used. This should include a clear system of arrangement and labeling and provide for easy reference and maintenance of the records.

Types of Documentation

The various phases of the preinstallation and postinstallation periods generate and require different types and degrees of information. The following represents a fairly standard approach and can be adapted to specific needs.

Application Manual

Records generated during the preinstallation period and applying to an entire application should be kept in this general classification. Considerable time and effort go into determining systems approaches, procedural changes, cost savings, and basic requirements. Many discussions are held and agreements reached between management, line departments, and data processing personnel. This information should be carefully recorded and retained to provide necessary reference material for the programmers, to state the authority for procedural changes, and to avoid later misunderstandings and questions.

A separate binder, with index tabs and designated sections, should be used to hold the material for each application. It should first contain a narrative which gives the objectives of the job, its scope, purpose, and other pertinent information necessary for a clear, concise description. This writeup can also describe the changes from prior procedures, the basis for the particular approach, and the sources for the information upon which conclusions were based. To support this and establish authority for the new procedures, a section should be provided for the correspondence, meeting reports, and agreements pertaining to the application.

Next, an overall system flowchart (Figure 3, page 12) should be included. This chart displays the flow of data through the system, the sources of the data, and the processing steps required. Input should be clearly labeled, disposition of output shown, and controls and audit trails clearly indicated. A computer program identification code should be shown on the flowchart for cross reference. This code can be used throughout the entire system, including the program cards. The system flowchart is often augmented by a table of computer runs listing programs for the application and specifying the machine time required, volumes, schedules, and other pertinent information.

Record layouts, sample forms, and any other material necessary for clear understanding of the application should also be included.

Any procedures for manual processing or unit record equipment pertinent to the application should be documented. This could be in the form of flowcharts along with the processing instructions. Forms showing keypunching requirements and their accompanying formats would be helpful here.

Figure 51 is an example of a check list which can be used to help assure that all required documents concerning an application are available. Note that in this example, the documentation for each program run is included in the application binder. More commonly, the runs are maintained in a separate run book as described below.

Master Run Book

A separate run manual should be prepared for each major computer program or group of programs. It should contain all the information relevant to the programs and should be organized in a fashion similar to that of the application manual – that is, in a binder divided into sections, with appropriate index tabs, etc.

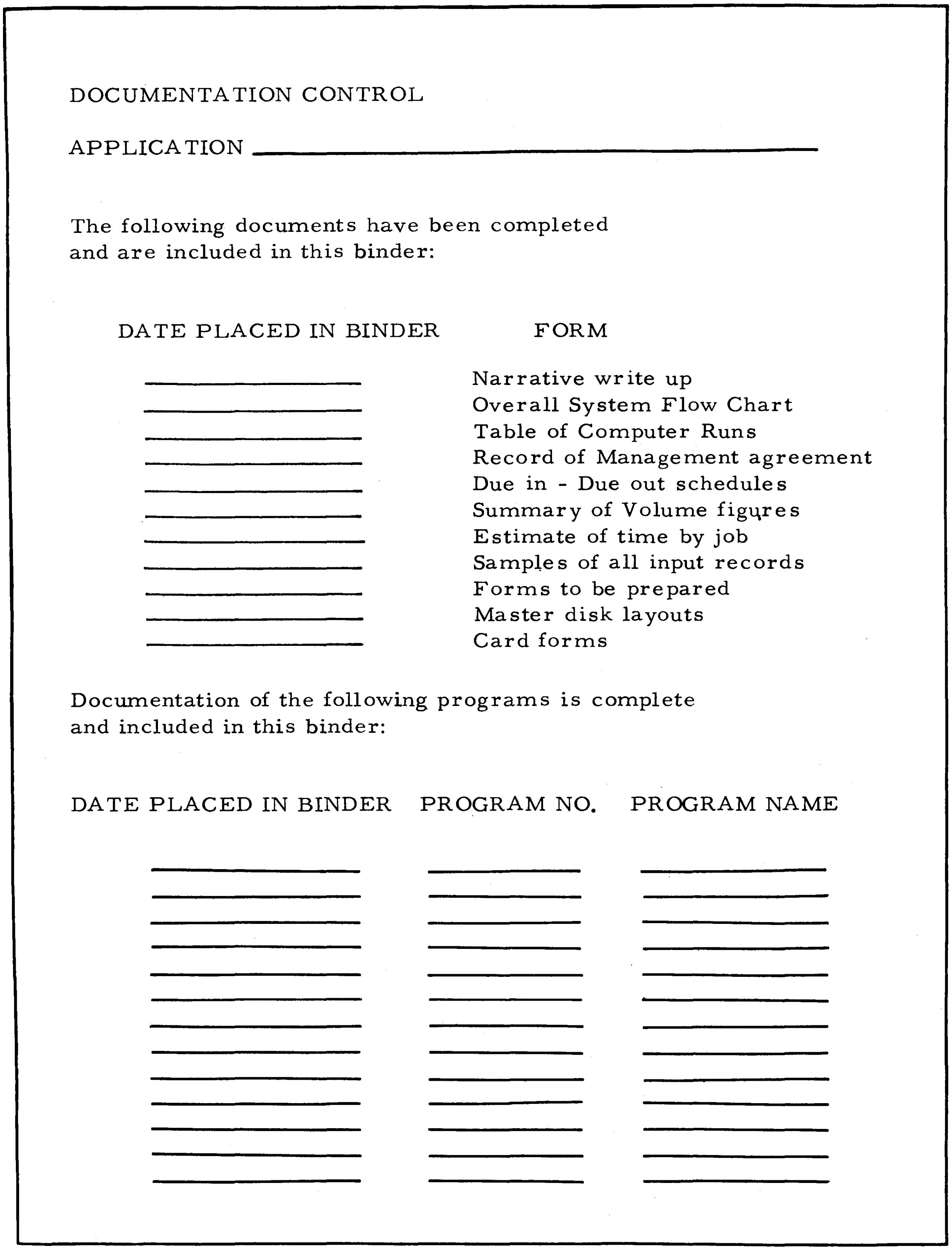
Most of the material required for the program documentation will be developed as a by-product of programming and testing.

Some of the questions to be answered by the run book are: What is the program? What does it do? How does it accomplish its function? Who wrote it? When was it written? Careful documentation concerning these and the other products of the programming effort will greatly assist anyone in finding what he needs to know about the program. The following paragraphs describe the materials which should be included in the run book:

Run description. – A section should be provided giving the program identification, its function, the name of the programmer, names of other people familiar with the program, input description and sequence, output description and sequence, controls, machine setup instructions, a list of programmed halts and restart procedures, disposition of input and output, and any other information pertinent to the overall operating picture of the program. Figure 52 and Figure 56, page 76, are examples of forms which can be used to record this information. The forms are used by the computer operator when the program goes into production.

General block diagram and description. – A general input/output diagram showing the input, main flow of the data, and the output of the program is needed. Along with this should be a narrative describing the run, giving its purpose, input and output requirements, and other pertinent information. Figure 10, page 19, shows the type of form that might be used for this information. In addition, the relationship of the particular program to the whole application should be indicated. This could be accomplished by including a portion of the overall flowchart of the application and outlining the specific run.

Block diagrams. – Current block diagrams incorporating all changes and revisions should be included. The degree of detail should be commensurate with the need for adequate understanding and with the re-



DOCUMENTATION CONTROL

APPLICATION ______________________________

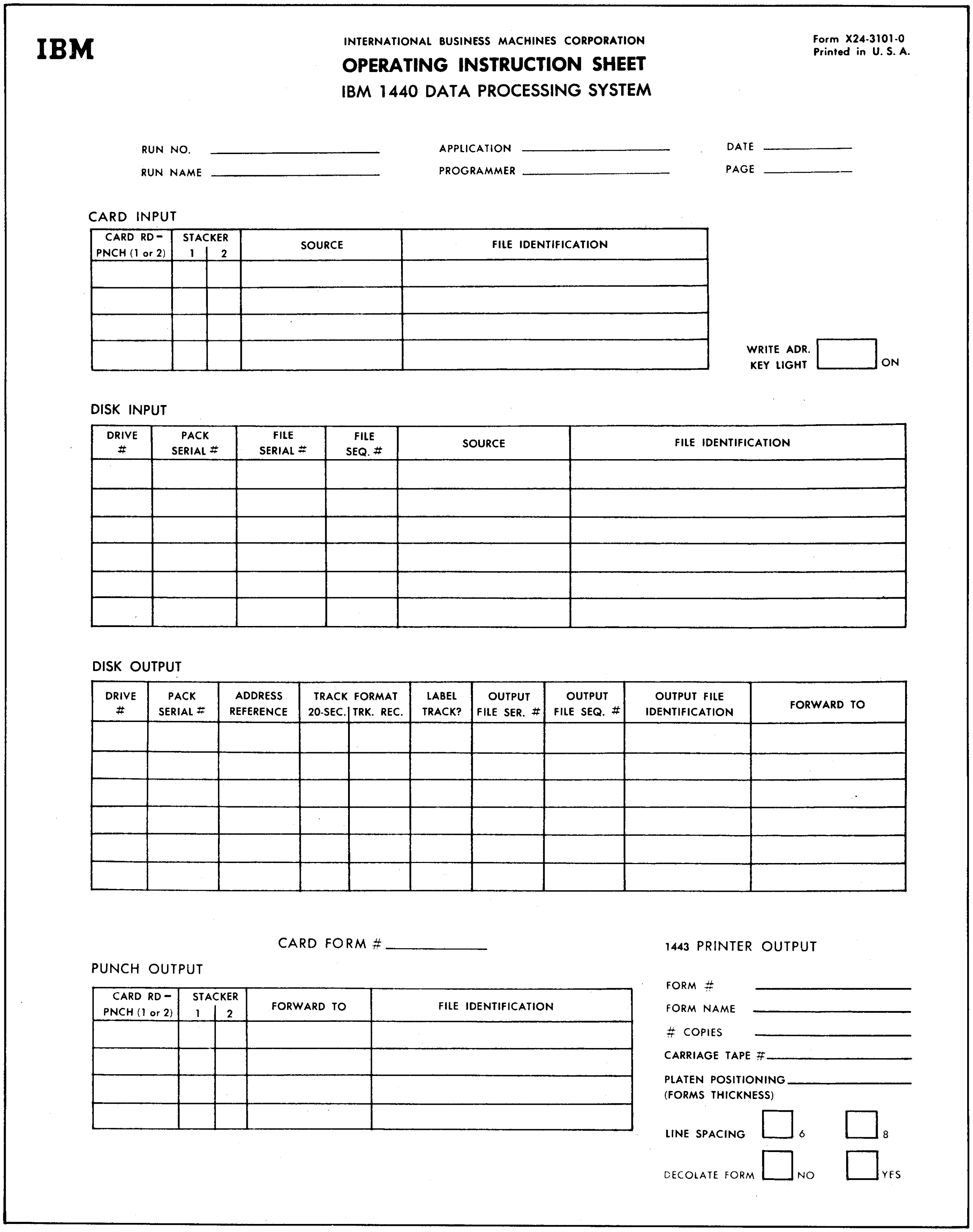
The following documents have been completed
and are included in this binder:

DATE PLACED IN BINDER	FORM
	Narrative write up
	Overall System Flow Chart
	Table of Computer Runs
	Record of Management agreement
	Due in - Due out schedules
	Summary of Volume figures
	Estimate of time by job
	Samples of all input records
	Forms to be prepared
	Master disk layouts
	Card forms

Documentation of the following programs is complete
and included in this binder:

DATE PLACED IN BINDER	PROGRAM NO.	PROGRAM NAME

Figure 51. Table of Contents—Illustrative Application Manual



IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

OPERATING INSTRUCTION SHEET

IBM 1440 DATA PROCESSING SYSTEM

Form X24-3101-0
Printed in U. S. A.

RUN NO. ______ APPLICATION ______ DATE ______

RUN NAME ______ PROGRAMMER ______ PAGE ______

CARD INPUT

CARD RD–PNCH (1 or 2)	STACKER 1	STACKER 2	SOURCE	FILE IDENTIFICATION

WRITE ADR. KEY LIGHT ☐ ON

DISK INPUT

DRIVE #	PACK SERIAL #	FILE SERIAL #	FILE SEQ. #	SOURCE	FILE IDENTIFICATION

DISK OUTPUT

DRIVE #	PACK SERIAL #	ADDRESS REFERENCE	TRACK FORMAT 20-SEC.	TRACK FORMAT TRK. REC.	LABEL TRACK?	OUTPUT FILE SER. #	OUTPUT FILE SEQ. #	OUTPUT FILE IDENTIFICATION	FORWARD TO

CARD FORM # ______

PUNCH OUTPUT

CARD RD–PNCH (1 or 2)	STACKER 1	STACKER 2	FORWARD TO	FILE IDENTIFICATION

1443 PRINTER OUTPUT

FORM # ______

FORM NAME ______

# COPIES ______

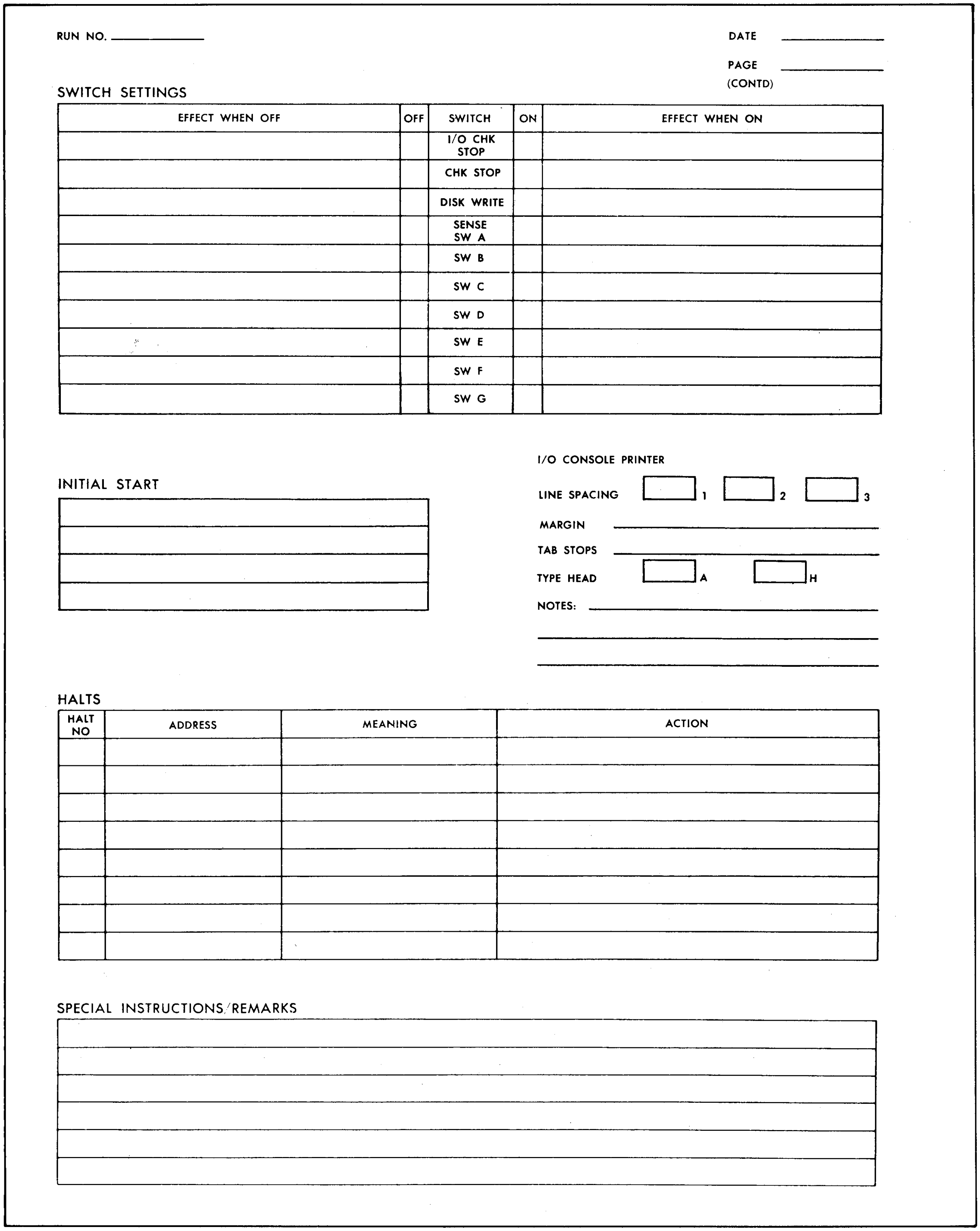
CARRIAGE TAPE # ______

PLATEN POSITIONING ______
(FORMS THICKNESS)

LINE SPACING ☐ 6 ☐ 8

DECOLATE FORM ☐ NO ☐ YES

Figure 52. 1440 Operating Instructions (Front)



RUN NO. ____________ DATE ____________

PAGE ____________
(CONTD)

SWITCH SETTINGS

EFFECT WHEN OFF	OFF	SWITCH	ON	EFFECT WHEN ON
		I/O CHK STOP		
		CHK STOP		
		DISK WRITE		
		SENSE SW A		
		SW B		
		SW C		
		SW D		
		SW E		
		SW F		
		SW G		

INITIAL START

I/O CONSOLE PRINTER

LINE SPACING ☐ 1 ☐ 2 ☐ 3

MARGIN ____________

TAB STOPS ____________

TYPE HEAD ☐ A ☐ H

NOTES: ____________

HALTS

HALT NO	ADDRESS	MEANING	ACTION

SPECIAL INSTRUCTIONS/REMARKS

Figure 52. 1440 Operating Instructions (Back)

quirements defined by the installation's programming standards.

Record layouts. — Detailed layouts of all records used in the program should be included. These would include input and output cards, master disk files, printed and typed reports, and a storage layout. (Figure 4, page 13, and Figure 5, page 14, show some sample layout forms.) The storage layout could be obtained from either preliminary planning records or from a machine print of the loaded program with the various areas of the program underlined and labeled.

Sample forms. — Card and report forms used in the program should be shown, along with form numbers, approximate quantities, numbers of carbons, etc. Sample reports can have actual information printed on them for the purpose of clarity. The carriage tape needed for the printer can also be included here.

Program listings. — A current machine listing of the source and object decks should be included. Any revisions as a result of desk-debugging or actual testing should be shown on the listings. (Since these listings do not fit easily into a 9½" x 11½" binder, they are often bound in a separate larger one.)

Program history. — Initially, this can show the programming progress, such as assemblies, tests, etc.; later it will contain the record of program maintenance and provide an audit trail by listing all modifications to the program. These should be documented by showing date, authorization, the addition or deletion, and the initials of the programmer making the change. Figure 53 shows a table of contents from a sample master run book.

Console Run Book

A run book should be set up for use at the console to assist in running the programs. It should contain complete operating instructions and other information required for normal operation of the runs. Many of the items included in the master run book described above can simply be duplicated and inserted in the console run book.

Some items which might be included are: a brief program narrative, checking of internal and external controls, halt listing and action to be taken, program setup information, disposition of files, sample carriage control tapes and printed forms, etc., and any information needed by the console operator to run the program.

The data in the console book should be organized in a form suited primarily for operator convenience and can be arranged either in a separate binder for each run, or in a binder for all runs in a particular application. For any occasions when more information than contained in the console run book is needed, the operator should have access to the master run book.

File Description Manual

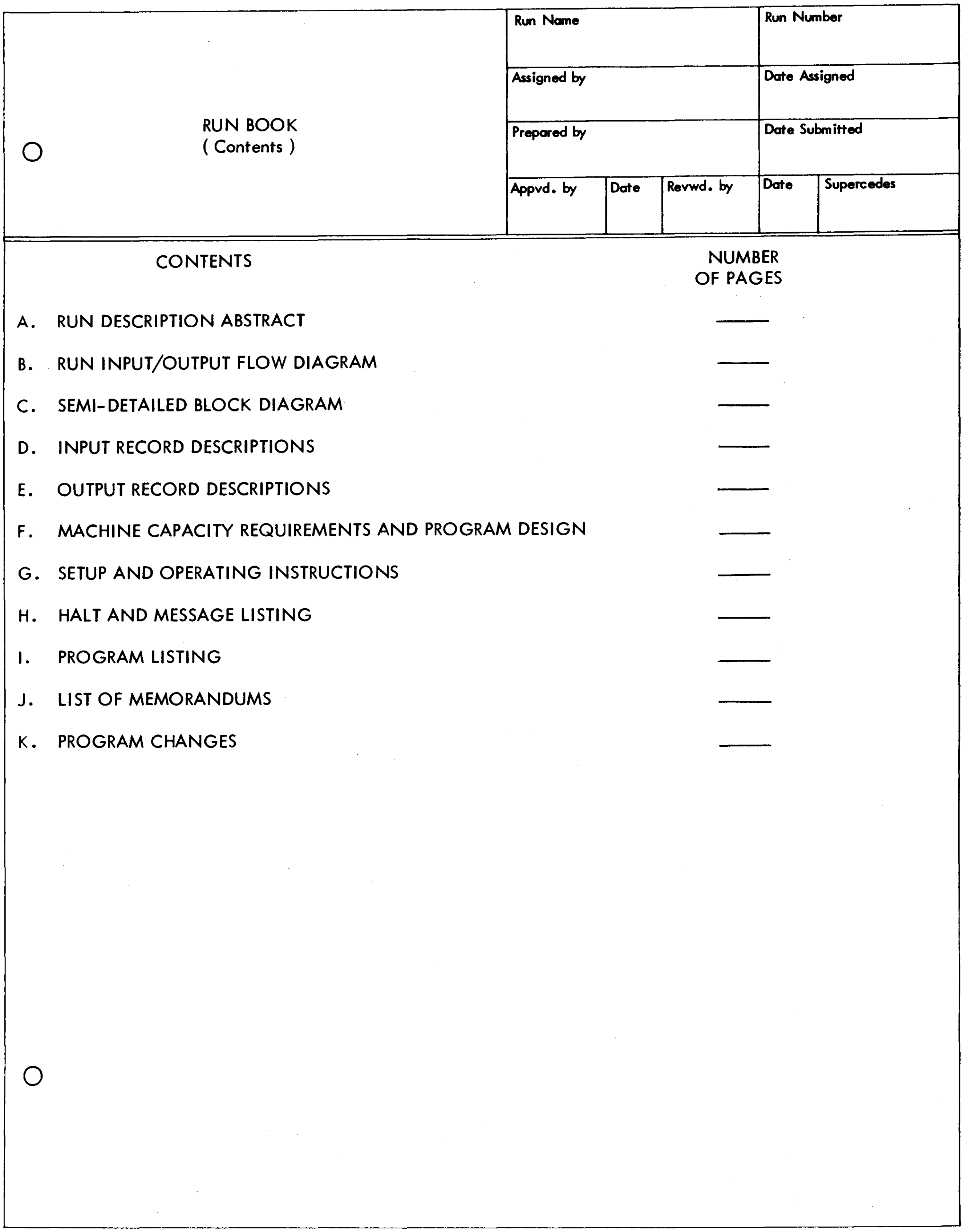
This manual contains a description of every file used in the system. It lists all record contents and formats, reasons for the formats, volumes, source, retention, uses, etc., and can be organized according to application, type, or whatever order would be most suitable. This type of manual has proved particularly helpful in installations using disk files.

Department Manuals

In some instances, it is desirable to prepare manuals for the departments providing data for and serviced by the data processing installation. The contents of these manuals vary according to the needs of the individual departments. Such items as pertinent correspondence, sample reports, control requirements, scheduling information, volumes, and retention periods may be included.

Summary

The information prepared for documentation should be reviewed and revised until it is sufficiently clear and complete to serve as the reference necessary during testing and conversion and after installation. Although considerable time must be devoted to preparing and maintaining complete, accurate and timely records, the amount of time can be justified and is usually small in comparison with the problems that can occur if proper attention is not given to documentation.



RUN BOOK
(Contents)

Run Name			Run Number	
Assigned by			Date Assigned	
Prepared by			Date Submitted	
Appvd. by	Date	Revwd. by	Date	Supercedes

CONTENTS	NUMBER OF PAGES
A. RUN DESCRIPTION ABSTRACT	———
B. RUN INPUT/OUTPUT FLOW DIAGRAM	———
C. SEMI-DETAILED BLOCK DIAGRAM	———
D. INPUT RECORD DESCRIPTIONS	———
E. OUTPUT RECORD DESCRIPTIONS	———
F. MACHINE CAPACITY REQUIREMENTS AND PROGRAM DESIGN	———
G. SETUP AND OPERATING INSTRUCTIONS	———
H. HALT AND MESSAGE LISTING	———
I. PROGRAM LISTING	———
J. LIST OF MEMORANDUMS	———
K. PROGRAM CHANGES	———

Figure 53. Table of Contents—Illustrative Master Run Book

Programming Standards

Establishing a system of programming standards* is an important part of installation planning. The early establishment and enforcement of standards governing program writing and implementation will significantly contribute to the success of an installation by providing a means of improving communication between programmers, of reducing duplication and ambiguity, and of minimizing difficulties caused by personnel turnover. It will also assist in program maintenance, in providing the information required by other personnel, and in numerous other activities.

Functions of Programming Standards

Some of the functions of standards and their benefits are as follows:

Role in communication. — The adoption of standards makes communication easier and more consistent. For example, a standard disk labeling procedure reduces the degree of communication required for the programmers in writing interrelated programs and aids the operating group in carrying them out.

Another result of standards is to make programs easier to operate and understand. Well defined terms, straightforward coding, standard labels and storage areas, and similar items greatly help people other than the original programmer in working with a program and, when necessary, in making any revisions and modifications.

Time savings. — Adhering to standards saves considerable programming effort. In the absence of standards, programmers normally take the time to develop consistent approaches for themselves. Standards can be applied in many programming areas without affecting a programmer's creativity. On the contrary, they relieve him of much detailed clerical work and allow him to concentrate on the main program logic.

Aid to management. — Establishment of standards provides management with criteria to measure programming progress. It helps supervisors in evaluating what has already been accomplished and in determining what remains to be done.

Development of general programming routines. — The objective here is the identification of common program requirements and the provision of routines to meet these requirements. Utility programs available from IBM should be evaluated for their role in filling these needs. In many cases they can be used without modification; in others it may be desirable to develop special utility programs. After the selection of utility programs has been made, the programmers can be instructed to use specific packages for certain functions and further advised as to what options are required or preferred.

Centralization of documentation. — Programming standards developed for use in the installation should be thoroughly documented. This will relieve programmers of the necessity of documenting a portion of their work and it will also provide information for new programmers and computer operators.

Areas of Standardization

Block Diagramming

Block diagrams show the sequence in which the functions of a run are to be performed and depict the logical elements of the program. They are one of the more important areas of program development and rules and regulations should be established early in the installation effort concerning them. Agreement should be reached on the symbols to be used, the degree of detail to appear on the diagrams, and whether the diagrams must be approved for completeness, logic and accuracy prior to coding.

There are several levels of block diagrams: the first is the general block diagram, a concise statement of the entire run. Normally drawn on one page, it outlines the major logical elements of the program, input and output functions, and the relationship of each function to the other major functions of the run.

Semidetailed diagrams, in which the major blocks of the general block diagram are expanded into smaller components, are the next stage of block diagramming. Most programmers use these block diagrams to aid them in expressing the major elements of a program and to show sequential and logi-

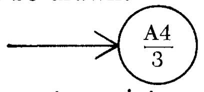
* 1440 installations fit into a broad range of size and applicability. Machine configurations can vary from 4K card systems to 16K systems with five disk drives. The system can be an integral unit or be part of a larger data transmission system. It may handle one application or perform the complete data processing function. Personnel available for preparatory work may consist of one or two programmers, or a large staff. All these factors must be considered when the approach to programming is decided and programming standards determined. Therefore this section does not attempt to specify the exact standards to be followed by an installation, but rather to outline some possibilities to assist installations in determining their own standards.

cal relationships. When working with this type of diagram, the various sections of the program can be rearranged until they fit properly and additional details can be added piece by piece once the basic structure is established. Extensive logic checking can take place at this point to detect any omissions or serious errors. Thorough checking at this level will result in fewer program errors and in easier testing.

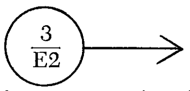
The next stage of the effort will be either programming or the drawing of a detailed block diagram, depending on the programmer's preference.

To attain maximum clarity and communication in block diagrams, standards should be set up for the symbols to be used (Figure 54), the technique of drawing, and the formats employed. Diagrams should be drawn on standard size sheets — normally on 8½" x 11" or 11" x 16½". Figure 3, page 12, shows form X22-6413 (diagramming worksheet) available for this purpose.

A consistent method of identifying each section and page of the diagram and indicating interrelationships of the documentation should be used. One way to do this is to use a letter of the alphabet for each broad block of the general diagram. Then as each block is exploded, the pages for that section can be numbered consecutively — for example, A1, A2, A3, B1, B2, etc. The system can also be used to identify entry and exit connectors with entry and exit points on other pages. To illustrate, an exit from a symbol or page could be drawn:



This references entry point number 3 on page A4. On page A4 the entry would be shown as:



This indicates that it is entry point 3 from an exit on page E2.

Adhering to methods and conventions such as these, a further explanation of block diagramming techniques can be gained by reading *Flow Charting Techniques* (C20-8152).

Programming Labels

A program labeling system should be devised, not only to standardize and clarify common usage, but to indicate as closely as possible the relationship between the block diagramming and the coding of a program. The problem here is twofold: not only should the program labels reference back to the block diagram, but the program listing also needs directional labeling so that branch points and subroutines can be easily located.

One method of doing this is to relate the page and line numbers on the coding sheets to the page and block numbers on the block diagrams. Alphameric program page numbers are permitted so that coding sheets can be tied directly to pages of a block diagram which uses a lettering system.

The same idea may be used for labeling branch points on the coding sheet. The programmer can number the blocks in a logical section from 01 to 99 (any later additions could be inserted with decimal notations so that sequence can still be maintained). The desired numbering need only appear on branches and may be constructed in the following manner:

E	06	090
Logic Diagram Section and High-Order Coding	Logic Diagram Block	Coding Line Number

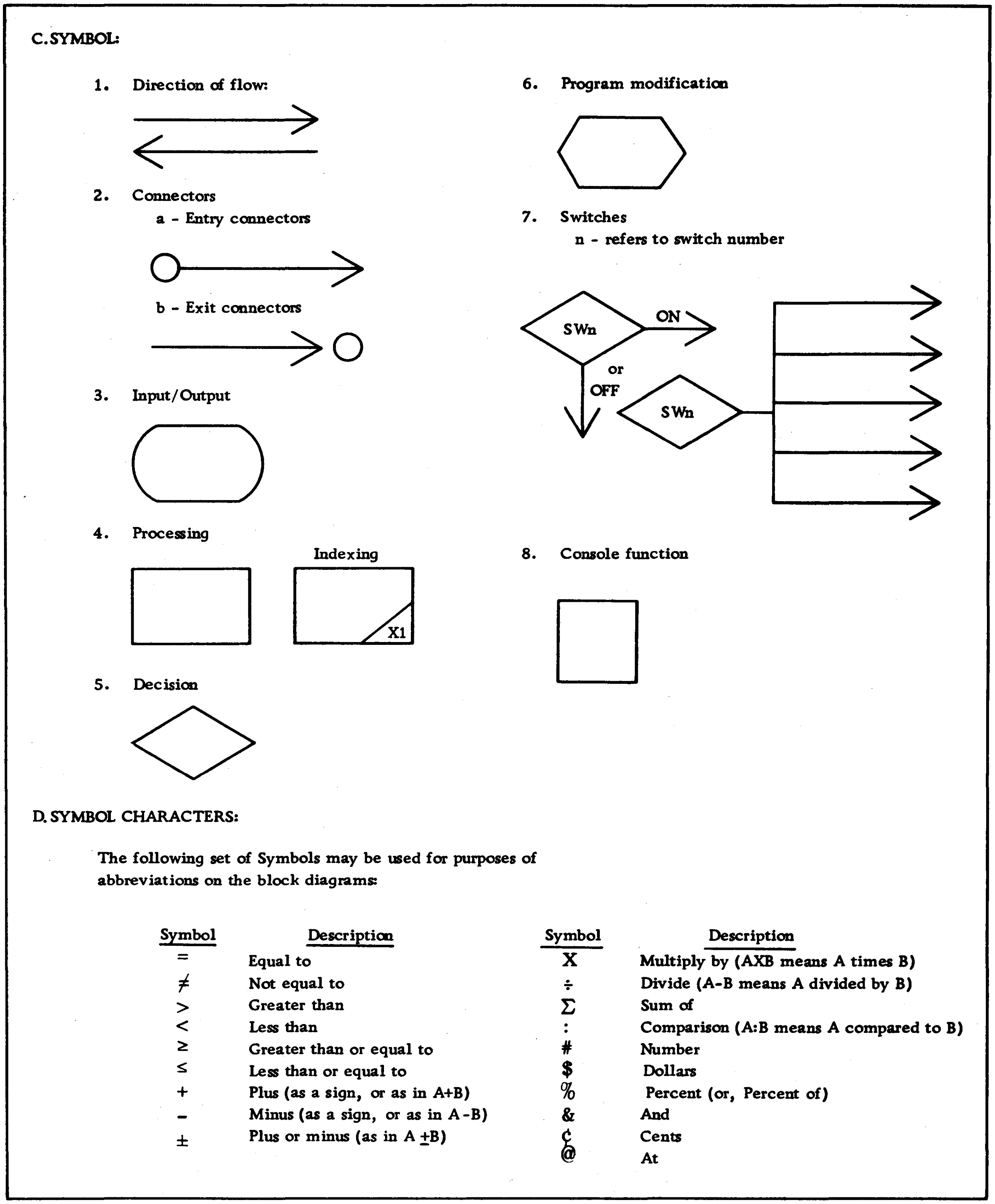
Immediate cross-reference between the block diagram and the coding is possible with these labels, and directional labeling between program sections is also accomplished.

Another area of program labeling which may be standardized is the labeling of areas, constants, and messages. Common letters (prefixes or suffixes) for various type of statements in the symbolic language will assist the understanding of a program. For example, all labels of constants could be preceded by the letter C, followed by a descriptive term for the constant; thus the stored alphameric constant PAGE might be labeled as CPAGE, the stored numerical constant 576 as C576. The same idea could be used for field definition. A ten-position storage area set aside to contain an AMOUNT field could be labeled DAMT10 where D represents field definition and 10 (ten) the size of the field. Input, output and work areas may be labeled in the same manner.

A standardized labeling system used consistently can provide continuity both within and between programs. It is a communication line between the block diagram and the coded program, a programmer and his work, and between two programmers.

Subroutines

Installations will often find that many subroutines are common to more than one program; these can include input and output routines, deblocking, counting, sequence checking and error routines, end-of-job routines and many others. Any routines which will be used repetitively should be standardized, documented and made available to the other programmers. They can be kept in prepunched form and, when required, reproduced and then inserted in the program deck. Standard methods for transferring control between



C. SYMBOL:

1. Direction of flow:

2. Connectors
   a - Entry connectors

   b - Exit connectors

3. Input/Output

4. Processing

   Indexing

   X1

5. Decision

6. Program modification

7. Switches
   n - refers to switch number

   SWn

   ON

   or

   OFF

   SWn

8. Console function

D. SYMBOL CHARACTERS:

The following set of Symbols may be used for purposes of abbreviations on the block diagrams:

Symbol	Description	Symbol	Description
=	Equal to	X	Multiply by (AXB means A times B)
≠	Not equal to	÷	Divide (A÷B means A divided by B)
>	Greater than	Σ	Sum of
<	Less than	:	Comparison (A:B means A compared to B)
≥	Greater than or equal to	#	Number
≤	Less than or equal to	$	Dollars
+	Plus (as a sign, or as in A+B)	%	Percent (or, Percent of)
-	Minus (as a sign, or as in A-B)	&	And
±	Plus or minus (as in A ±B)	¢	Cents
		@	At

Figure 54. Illustrative Page of Programmer's Handbook

the main routine and the subroutines should be devised to assist in utilizing the standard routines.

A group writing in full Autocoder can incorporate the routines into the macro library and use macro instructions to extract the required subroutine.

The establishment of tested subroutines is a substantial improvement over having each programmer develop his own. Such techniques allow the programmer to concentrate on the main logic of his program. As programming progresses, subroutines will be developed and proved; these can then be documented and put into a subroutine library for use by the entire group.

General Programming Areas

Core storage organization. – A standard should be established whereby the same storage areas are reserved in all programs for common functions such as disk input and output, card input and output, work areas, constants, etc. A storage layout is often used to identify areas of core reserved for these purposes.

Halts. – Halts are normally used for a number of purposes – for example, (1) to inform operators of the program status (that is, program loaded and ready, or program complete), (2) to advise of a condition requiring operator attention and decision, such as a disk label error, a choice to take a memory dump or not, etc., (3) to advise that the program has reached an uncorrectable error, such as an out-of-sequence record condition or an invalid card.

Common types of halts should be standardized – for example, all end-of-job halts can be coded 999. This will not only aid program standardization but will help avoid operating errors and increase operating speed and efficiency.

In cases where it is more desirable to bypass certain error conditions and continue processing than to halt the computer for operator intervention, error messages may be printed out on the 1443 or on the 1447. These should be of standard format in order to provide a consistent, easily understandable error log.

Wherever possible, errors should be handled by the program and operator intervention kept to a minimum. However, where necessary to have manual corrections, the instructions should be clear, concise, and standardized for common conditions.

Program controls. – Internal program controls should be spelled out in as much detail as possible to assure that the control procedures are fully understood and implemented by individual programmers. Some controls such as checking card codes, sequence checking, card counts, hash totals, and disk label checking are common to almost all programs and should be included in the standards. Provision should also be made for standard routines for determining that the correct program is loaded and the correct cards and files are used.

Device allocation. – It is a good practice to standardize the assignment of such devices as disk units, stackers, sense switches and index registers when such assignment can be common to a large number of programs within the installation. Examples of this principle would be the assignment of sense switch G to cause a printout of a disk record, or index register 1 for disk record blocking.

Input and output. – An important consideration in standardizing input and output is operator handling and convenience. In many cases, the jobs processed on the 1440 will be of a short-run nature and setup time will occupy a relatively high percentage of job time. Reduction of setup time can thus measurably improve machine utilization. Report programs can be written to include headings and descriptive information so that the use of special forms is minimized and a standard form can be left in the printer to be used for many jobs. Programs written with this in mind can incorporate skips to channel on the carriage tapes at the beginning and end of a job. This relieves the operator of special attention to the printer and positions the paper for immediate detachment at the end of the job. Jobs with punched card output can be programmed to begin and end with dummy punch cycles to allow the operator to remove cards without using the non-process runout key.

An additional technique for saving operator time is the use of standard start procedures and standardized error and exception messages. If the console typewriter is used for messages, the format should be the same for all jobs. Messages should be brief and standard, with the same information appearing in the same location for all jobs. Any end-of-job indication given on the typewriter should also be the same for all jobs.

Program Blocking

The programmer should make every attempt to develop his program as a series of logical subroutines or program blocks. This process can naturally develop from translation of block diagrams into coded routines. Each block should be documented by comments giving the title of the block, its function and an explanation of any unusual techniques employed. References to block diagrams may also be incorporated in the comments.

Program blocking when carried to its fullest extent is often referred to as modular programming. This is the building-block concept in which a complex program is broken into logical parts and simplified sections, each of which is programmed independently, and the whole then controlled by a single main-line routine. The main features of modular programs are:

- A main-line routine directs the flow of data to the required processing blocks – that is, no processing block can direct data to another processing block.
- Communication between the processing blocks and the main-line routine is accomplished by a switch or branch.
- All common areas are part of the main-line program.
- Input and output functions common to more than one routine are executed by the main-line program.

The advantages of this approach are that programming is easier, more than one person can work on the same program, and other programmers can understand the program more readily. Program modification is also facilitated since changes need be made only in the affected block. In the same way, additions can be made by expanding the main-line routine to allow the proper incorporation of the block.

Programmer's Handbook

The standards formulated can be most effectively communicated to the programmers by the creation of a programmer's handbook. This should be designed to consolidate the material that a programmer needs to systematize his work, to use programming packages and aids most effectively, and to adhere to installation standards.

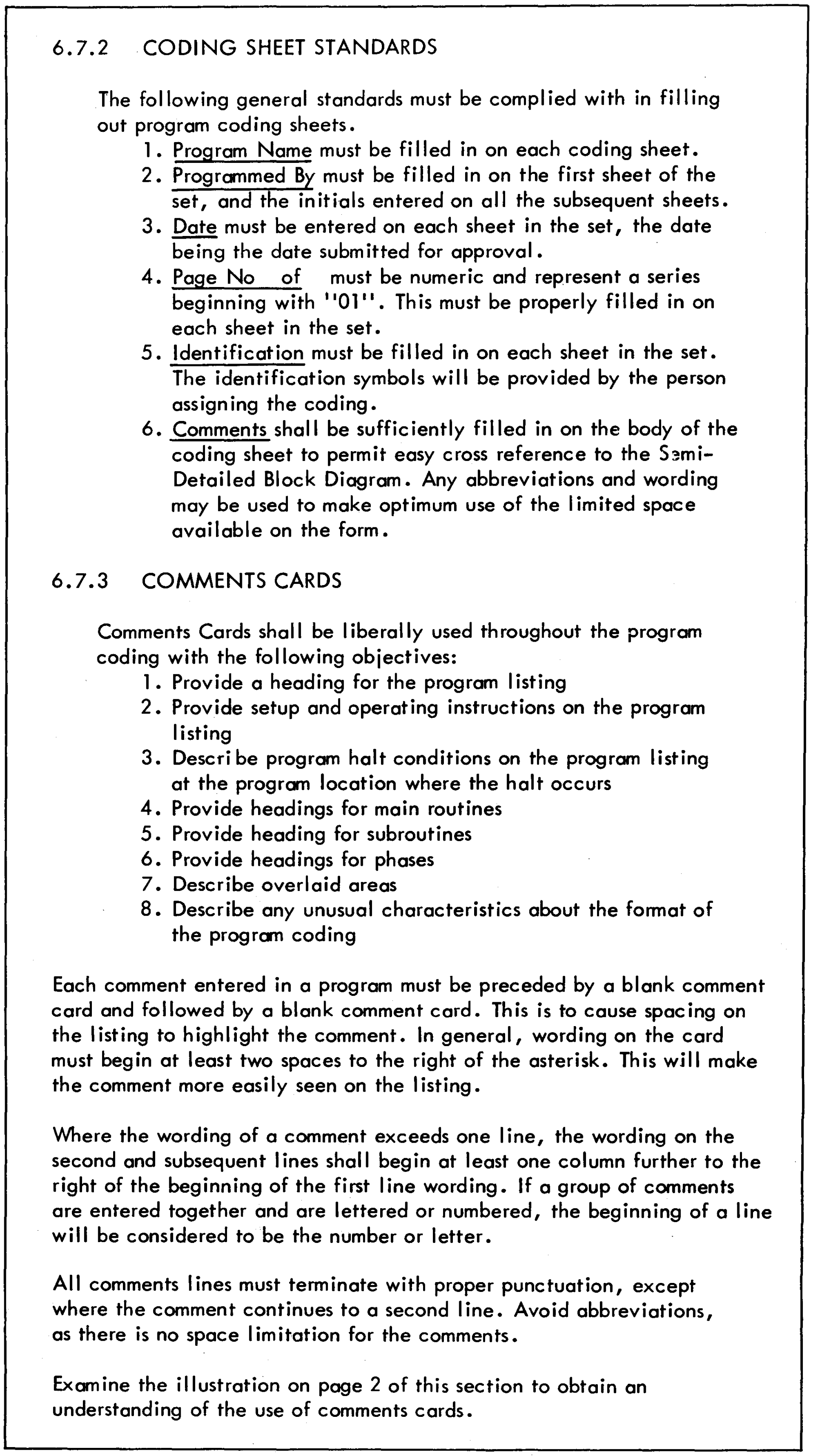
The handbook could contain the directives covering the detailed standards to which a programmer must conform; there could also be a section for memos or bulletins which disseminate information and pointers pertaining to programming, equipment and other pertinent topics. As different features develop or come up, they can be recorded and issued in this format.

A series of memos and directives provides a better means of clarifying policy and standards for programs than oral presentation in a meeting. Too often word passed orally is forgotten by the programmer or is not fully understood. A written document clarifies the issue and provides ready reference when needed.

The programmer's handbook should be started early in the programming installation effort and supplemented whenever the occasion warrants. Such a reference manual can prove very valuable in the improvement of program writing and testing.

Conclusion

The above are a few of the possibilities for programming standards. Since every installation is unique, standards must be developed by each installation in the light of its own requirements. An installation without standards is subject to unnecessary loss of time, redoing of work, and additional expense. A sound set of standards can avoid this and help provide a base for data processing development.



6.7.2 CODING SHEET STANDARDS

The following general standards must be complied with in filling out program coding sheets.

1. Program Name must be filled in on each coding sheet.
2. Programmed By must be filled in on the first sheet of the set, and the initials entered on all the subsequent sheets.
3. Date must be entered on each sheet in the set, the date being the date submitted for approval.
4. Page No of must be numeric and represent a series beginning with "01". This must be properly filled in on each sheet in the set.
5. Identification must be filled in on each sheet in the set. The identification symbols will be provided by the person assigning the coding.
6. Comments shall be sufficiently filled in on the body of the coding sheet to permit easy cross reference to the Semi-Detailed Block Diagram. Any abbreviations and wording may be used to make optimum use of the limited space available on the form.

6.7.3 COMMENTS CARDS

Comments Cards shall be liberally used throughout the program coding with the following objectives:

1. Provide a heading for the program listing
2. Provide setup and operating instructions on the program listing
3. Describe program halt conditions on the program listing at the program location where the halt occurs
4. Provide headings for main routines
5. Provide heading for subroutines
6. Provide headings for phases
7. Describe overlaid areas
8. Describe any unusual characteristics about the format of the program coding

Each comment entered in a program must be preceded by a blank comment card and followed by a blank comment card. This is to cause spacing on the listing to highlight the comment. In general, wording on the card must begin at least two spaces to the right of the asterisk. This will make the comment more easily seen on the listing.

Where the wording of a comment exceeds one line, the wording on the second and subsequent lines shall begin at least one column further to the right of the beginning of the first line wording. If a group of comments are entered together and are lettered or numbered, the beginning of a line will be considered to be the number or letter.

All comments lines must terminate with proper punctuation, except where the comment continues to a second line. Avoid abbreviations, as there is no space limitation for the comments.

Examine the illustration on page 2 of this section to obtain an understanding of the use of comments cards.

Figure 55. Illustrative Page of Programming Standards Manual

Programming Aids

A large number of programming aids are provided by IBM to assist users in programming the 1440. These are of two basic types. *Programming Systems* are generalized programs developed and maintained by IBM as an integral part of the data processing system. Some of the types of programs in this category are:

- Automatic test programs to facilitate program testing.
- Compiling and assembly programs that simplify program writing by allowing the use of symbolic names for data and instructions and through other features designed to save program writing time and minimize programming errors.
- File organization programs to assist users in establishing and maintaining disk files.
- Input/output programming systems to simplify program preparation by providing routines to eliminate the detailed programming necessary for input/output operations.
- Report generators to produce programs to write reports.
- Generalized sort programs to enable users to perform sort operations without the necessity of writing individual programs to do so.
- Utility programs to assist in frequently required operations such as loading disks from cards, unloading disks to cards or printer, etc.

Application programs are programs written to perform specific applications and are designed so that they may be modified by the user to fit his individual requirements. Most of the programs are for use in a particular industry.

Further information on the functions and use of both types of programs is available in *Programming Systems Concepts* (F20-8147), *Catalog of Programs for IBM Data Processing Systems* (C20-8090), IBM *1440 Systems Summary* (A24-3006), and in the bulletins and manuals written on the individual programs. The following is a partial listing of the 1440 programming systems.

1440 Programming Systems

1440 Basic Autocoder – a symbolic programming system designed to simplify the preparation of programs for 1440 card systems. Source programs written in 1440 Basic Autocoder language are keypunched into cards and processed by the Basic Autocoder program to produce a machine-language object program. The object program is punched, one instruction per card, into the original source cards. A program listing is also prepared.

1440 Autocoder – an advanced symbolic programming system that represents a significant extension of 1440 Basic Autocoder. The language provides continuity with the 1401 and 1410 Autocoder systems. Some of the advantages of Autocoder over Basic Autocoder are (1) more powerful language that includes macro instruction facilities, (2) more freedom with literals, and (3) a more automatic assembly process through use of 1311 disk storage.

1440 Autocoder-1401 Processor – a 1401 assembly program which will assemble programs written in 1440 Autocoder language. The output of the assembly program is the 1440 machine-language object program punched in cards.

1440 Input/Output Control System (IOCS) – designed to eliminate to a large extent the detailed programming required for input/output operations, IOCS provides a set of generalized routines for blocking and deblocking, together with the associated macro instructions, GET, PUT, OPEN and CLOSE, and standardized routines for input/output error correction, end of file and disk labeling. The routines are called for by Autocoder statements and are entered into the object program during the Autocoder assembly.

1440 Disk Utility Programs – designed to assist the user in the frequently required operations of a 1440 installation. Loading disks from cards, unloading disks to cards or printer, and disk-to-disk operations can be performed without a special programming effort on the part of the user. The programs included are Clear Disk Storage, Disk to Card, Card to Disk, Copy Disk, Disk to Printer, Disk Record Load.

1440 Disk File Organization Programs – help users establish and maintain their data files in 1311 disk storage. The programs are designed to support either a random file using the chaining method of organization or a sequential file using a method of organization referred to as control sequential. There are eight programs for random file organization methods and five for sequential file organization methods. The Random Disk File Organization Routines are:

1. PASS1	Loads home master records.
2. PASS2	Loads non-home master records.
3. RNADD	Adds master records to an organized file.

4.	TRADD	Loads or adds trailer records (single or multiple).
5.	RNDEL1	Deletes master records and associated trailers. (Also used to tag records for subsequent deletion.)
6.	RNDEL2	Deletes tagged records.
7.	TRDEL	Deletes single trailer records.
8.	RNUNLD	Unloads a random file for reorganization.

The Control Sequential Disk File Organization Routines are:

9.	CSLOAD	Loads a sequenced file.
10.	CSADD	Adds records to a sequential file.
11.	CSDEL1	Deletes or tags records.
12.	CSDEL2	Deletes tagged records.
13.	CSUNLD	Unloads and resequences an organized file.

1440 Basic Report Program Generator (BRPG) – used to produce report programs for the 1440 Card System with a minimum of programming time and effort. Instead of writing a specific program for a desired report, the user states his reporting problem and solution (the report specifications) in the BRPG language. This language is essentially problem-oriented rather than machine-oriented.

The BRPG will produce programs which will write reports in varying formats, using source data contained in card files. The reports can be prepared in printed and/or punched card form.

The reports produced by programs generated by the Basic Report Program Generator range from a simple listing of items from the input file to complex reports that incorporate editing and calculations upon the input data. The types of lines which may be printed or punched include heading lines, detail lines, and total lines (including offset totals).

The BRPG processor is a "load-and-go" system. That is, when the specification cards containing the job definition are read into the 1440 with the source data cards, the desired report is produced automatically.

1440 Report Program Generator (RPG) – facilitates the preparation of programs to write reports of variable format from data stored either in punched cards or in 1311 disk storage. The need for writing a specific program is reduced to the requirement of supplying a set of report specifications. The report can be in any format desired and can be printed and/or punched into cards and/or written on 1311 Disk Storage.

The input to the generator program comprises specifications of the data from which the report is to be made and specifications of the desired report. The output from the generator is at the option of the user. Under one option a 1440 machine-language program is produced which, when executed, will prepare the desired report. Under another option a symbolic-language report program is produced by the RPG processor. The program must be further processed by the IBM 1440 Autocoder program to produce the object program in machine language. The user may specify that the report includes various classes of headings and totals, format control, input data selection and forms control.

1440 Sort 5 – a set of generalized sorting routines that may be incorporated in the 1440 Autocoder Library. The user supplies the Autocoder processor with a description of the sort required and the 1440 system to be used, causing it to generate a sort program conforming to his specifications. This generated program, with control cards that describe the data file to be sorted, may then be loaded and executed. Sort 5 will sort fixed-length, blocked records contained in a punched card file or in disk storage (move mode).

1440-1448 Input/Output Control System – relieves the programmer of much of the programming effort required to control data transmission to and from remote terminals. It serves the user in three distinct ways:

- It provides concise, efficient, pretested routines to handle such programming functions as priority request, end-of-block detection, error detection, status alteration, control word address initialization, output scheduling, and coordination of the 1440/1448 system with other input/output devices.
- It schedules real-time routines and defines what programming functions can be performed within these routines.
- It provides all functions on an option basis so that the user does not require core storage for routines that are not needed.

1448 IOCS may be added to the 1440 Autocoder Library in the same way that 1440 IOCS is added.

1401/1440 DDC Input/Output Control System – an IOCS for direct data channel connection of two 1440s or of a 1440 and a 1401. The system provides concise, efficient, pretested routines to handle:

- Program detection of read request or write request by either system.
- Priority (interrupt) request (included with Direct Data Channel feature of 1440 systems when the 1448 is attached).
- Error detection.
- Output scheduling.
- System-to-system read-write.
- Coordination with other IOCS programs in either system.
- Scheduling of the user's Direct Data Channel routine for each system.

1440 Intersystem IOCS – provides all the functions of the 1440-1448 IOCS plus the following functions for servicing the communication needs between the 1440 and 1410-7010 systems:

- Reading and writing of data messages.
- Reading and interpreting of control messages.
- Writing of sense data.
- Writing of service messages.
- Priority (interrupt) request.
- Error detection.
- Output scheduling.
- Coordination with other IOCS programs.
- Scheduling of the user's routine for processing records received from the 1410/7010 system.

The program is compatible with the 7750 IOCS systems provided for the 1410/7010.

1440 Autotest – is designed to facilitate program testing. It provides the ability to test a number of object programs in a batch and produce, with minimum operator intervention, the necessary documentation to use the result of the test. The Autotest program:

- Clears core storage before loading each object program.
- Loads the object program.
- Executes the object program.

In addition, Autotest provides the following features, which may be selected by the user:

1. Clear Disk Storage. – Selected areas of disk storage can be cleared before or after execution of the object program.
2. Print Disk. – Data from selected areas of disk storage can be printed before or after execution of the object program.
3. Core Storage Printout. – The contents of core storage can be printed at an end-of-job or hang-up condition.
4. Disk Record Load. – This provides for the creation of disk records from card input prior to the test of each individual program.
5. Snapshot Program. – Data from a selected area of core storage can be printed at specified times during the execution of the object program.
6. Disk Trace. – Data that is read from, or written on, the disk during execution of the object program is traced and printed with the control field of the read or write instruction.
7. Autopatch. – This allows the programmer to correct the object program without reassembling or computing three-position machine addresses. Patch instructions may have indexed operands.

Any number of these optional features may be used when testing an object program.

Program Testing

Introduction

Program testing is one of the most important tasks in preparing for the installation of a 1440 system. The primary purpose of preinstallation testing is the detection and correction of program and logic errors before machine installation. Testing, however, confers many other valuable by-products which can benefit the preinstallation period. Some of these are:

- Early operator and programmer training through actual hands-on experience.
- Greater familiarization with the machine functions and program instructions.
- Appreciation for the importance of good machine operation, scheduling, and optimum machine utilization.
- A lightening of the workload during the conversion period.

For these reasons, then, the initial test sessions should be scheduled as early as practical in the programming effort and should be preceded by careful and thorough preparation.

Once decisions have been made as to the programs to be tested, and the testing schedules have been set up, a great deal has to be done before the programs can be taken to a test center. Complete documentation is required for each program, every detail of which should be carefully checked for accuracy. This includes not only clerical proofing but examination of the program logic and coding. Test data should be prepared and stepped manually through the program so that errors can be found and corrected before the test session. In addition, programmers should become familiar with Autotest, utility programs, and initializing routines during the pretest period.

Intermediate checkpoints and predetermined result data should be prepared beforehand, as well as a schedule of operations for the sessions. Those attending the tests should be assigned specific operations during the machine session.

When the above has been done the programs can be taken to the test center for testing on the 1440. There the programmer will get hands-on experience while running his programs; he will learn how to use Autotest and how to recognize error indications and to take steps to correct them. Periods of time will be allowed between machine sessions so that the programmer can correct errors and prepare for retesting.

The subsequent section describes, in detail, preparations for testing, the testing session and remote testing.

Pretest Preparation

The success of a test session is directly proportional to the amount and quality of the preparation which precedes it. Efficient and proper utilization of time and materials in the test preparation helps ensure that a minimum of difficulty will be experienced in actual testing and that desired results will be achieved as scheduled.

Preliminary Planning

Plan to take more programs than can likely be run in the time allowed. Sometimes programs are found to need major revisions which eliminate the possibility of any further immediate testing; others may be debugged and checked out more rapidly than anticipated. The presence of additional programs reduces the tendency to make hurried error-prone corrections; should a program not be ready for retesting it may be bypassed and another program substituted.

Elementary programs for input, output and conversion should be scheduled first, since they will provide experience for the more complex programs. When possible, in scheduling the work over the series of test sessions, no program should be dependent upon the test of another in the same session, for if trouble is encountered on the first, no input is available for the subsequent program. If a situation occurs where it is necessary to test interdependent programs, input data for the dependent programs should be prepared as if it were the actual output of a prior program.

Autotest and Utility Programs

Prior to testing, a review should be made of Autotest, a monitor program that greatly facilitates testing, and the 1440 utility programs. The utility programs are adaptable to specific needs through control cards prepared by the user. Many of them, such as storage printouts, clear disk, card to disk, disk to printer, etc., can fill programming and testing requirements and should be utilized, where possible, for the purposes of efficiency and standardization.

A determination of the utility programs to be used should be made early in the programming effort, and programming personnel should become familiar with those selected. Procedures can be set up for the use and maintenance of these decks and the availability of them to those concerned. Responsibility should be assigned for the preparation of necessary control cards, their proper insertion in the decks, and the placing of the decks in the proper order for testing. Operating instructions should be reviewed carefully and listings made of the programmed halts which can occur during the operation of the utility programs.

Another necessary area of familiarization is the Autopatch card format. The Autopatch feature of Autotest facilitates changes and corrections to assembled object programs, making it possible to minimize program reassemblies. Autopatch provides for deleting any object instruction, adding new instructions, replacing instructions, and adding subroutines.

Console Operation

Proper operation of the machine has an important influence on the effectiveness of a test session. Therefore all personnel who will be testing should be familiar with the 1440 console operations and techniques. The 1440 System Component Description Manual and the initial orientation session will be good aids in this area. Any questions not clarified should be listed and answers sought before the test session. Assignments to machine components should be rotated so that all programming personnel have the opportunity to gain experience on the entire system during the test sessions.

Desk Checking

Proper desk checking has great influence on the success of a test session since many programming and clerical errors can be discovered and corrected before the visit to the test center. Every "bug" uncovered in the desk-checking phase saves a portion of a machine session and the accompanying costs.

Desk checking is most beneficial when done either by someone other than the programmer or by the programmer and another person. Besides picking up errors that the original programmer might overlook, it familiarizes a second person with the program.

Throughout the process of desk checking, two areas should be kept in mind: clerical accuracy and programming logic. The following is a description of a method of desk checking and a few of the checkpoints to be considered at each step of the way.

PROOFING THE PROGRAM

- A read-through of the block diagram. In addition to its checking value, this phase will familiarize the desk checker with the objectives of the program before he becomes immersed in the detailed coding. Checkpoints include errors in logic, closed loops, errors in program switching, and completeness of problem definition.
- A read-through of the handwritten coded program prior to the keypunching of the program cards. This will eliminate unnecessary repunching. Checkpoints include:
    1. Consistency of the program with the block diagram.
    2. Field lengths and field definition—that is, all constants and fields are defined by declarative statements and have the correct length and word marks.
    3. Word marks properly set in the input and output areas as necessary and their presence or absence due to the effect of previous operations.
    4. Tracing of all branches.
    5. Proper operation codes and modifiers.
    6. Sign control in arithmetic and compare operations.
    7. Proper length of edit words.
    8. Field lengths and work marks for comparing operations.
    9. Proper setting and clearing of group marks for input and output operations.
    10. Adherence to programming standards.
- A key verification of program cards.
- Interpretation of the program cards not punched on an IBM 26 Printing Card Punch.
- Machine listings. Listings vary from a simple listing of the information punched, to specially prepared ones which include editing features.

  The cards should first be listed in page and line number sequence and proofread against the original program. Once this listing has been verified the original handwritten program sheets can be destroyed to avoid duplicate maintenance of documentation. Checkpoints here include differentiation between 2 and Z, 5 and S, 1 and I, 0 and O, U and V; proper alignment of card columns; identification in cc. 76-80; proper control and end cards; and proper sequence of cards.

Some other program listings that may be helpful are:

1. Listing the program deck by operation code to insure that only valid operation codes are used and punched in the proper card columns.
2. Listing the decks by label to detect duplicate, misspelled or unreferenced labels.

STEPPING THE DATA THROUGH THE PROGRAM

In this phase, sample data is stepped through the program just as if the 1440 were processing it. The data should include precalculated results and contain the transactions necessary to check all program branches. The checkings would proceed from the simplest case to the most complex. Notations can be made during this process on the data and block diagram and a storage layout can be used to note the exact result of each machine operation. Care should be exercised to precisely follow the rules governing the function of each operation code. Checkpoints here include:

1. Direction of branch instructions, particularly those resulting from an internal computation or condition.
2. Assurance that last-card and end-of-file procedures have allowed the last records to be processed before the end-of-job halt.
3. Index register usage and address modification.
4. Correct character adjustment, particularly after program revisions.
5. Accuracy of results from the program as compared with the results obtained by the system presently being used. If possible, someone other than the programmer should obtain the present-system results. This will help prevent repetition of the same error.

FURTHER DESK CHECKING CONSIDERATIONS

It has often been found valuable to do a thorough job of desk checking immediately after writing the program, while its concepts are still fresh, and then put it away until a later date when it can again be completely desk-checked.

A careful, complete job of desk checking can eliminate many program errors before machine testing and will result in better and more efficient testing. In addition, it can eliminate costs incurred for travel and living expenses during extra test sessions required to find errors, and can make the time available for more constructive purposes.

Preparation of Test Data

The emphasis here is on the quality and completeness of the test data, not on its quantity. Volume should be held to the minimum that will permit thorough testing, since repetition of data wastes machine time and increases preparation time. Artificial test samples should be used until successfully processed, then actual information can be used to check that all conditions were taken into consideration.

The test data should be carefully checked for accuracy, for the presence of invalid double-punched columns, and for proper codes to designate the type of record. A printed documentation of the data is an invaluable aid in testing and can be coordinated with the routines each piece of data is designed to check out.

TYPES OF DATA REQUIRED

Program checkout data.—Input cards should be prepared to test each condition provided for in the program. These cards should be carefully prepared to cover all conditions and combinations of conditions.

The first test sample that is prepared should go through the main line of the program. Each succeeding test sample should be an extension of the first one and should introduce a minimum of new conditions.

Test samples should also be prepared to check for error conditions—for example, improper codes, cards out of sequence, etc. These cases should be tested after the valid input data has been checked. In this way, main routines receive priority testing and are not complicated by exceptions.

Finally, combinations of data which test all parts of the program should be run to test program continually.

Master file data.—Data needed for disk pack master records should be prepared and punched in cards. Only master records referred to by the input test data need be loaded. Loading of unused records increases file-loading time and consequently decreases available test time. Autotest can be used to load the records on the disk pack.

Result data.—The next step following data preparation is the calculation of results for the test data. The format of the precalculated results should be such that it can be easily compared with that which the 1440 is expected to produce. Wherever possible, the results should be obtained by the present accounting and/or machine methods and preferably by someone other than the programmer. If the programmer prepares the results, he is subject to the same errors or omissions in logic that he might have made in the program. Types of result data are:

- Output cards. Sample decks of output cards may be prepared to match those punched by the 1442. This provides a direct comparison for quickly checking the accuracy of results.
- Printed output. Preprinted forms may be prepared for visual comparison with those printed on the 1443.
- Typewritten output. Anticipated output should be prepared in the same fashion as other printed output.
- Disk pack records. The data for updated master files should be prepared in the same format as the expected results. This can then be used for comparison with the disk pack dump to make sure any updating has been handled correctly.

Program checkpoint data.—Checkpoints incorporating partial results should be established at strategic points in the program. The snapshot feature of Autotest can then be used to print out the pertinent areas of core. This will assist in testing various branches, multiple calculations, long, complex programs, and will help locate problem areas quickly.

Test Materials and Supplies

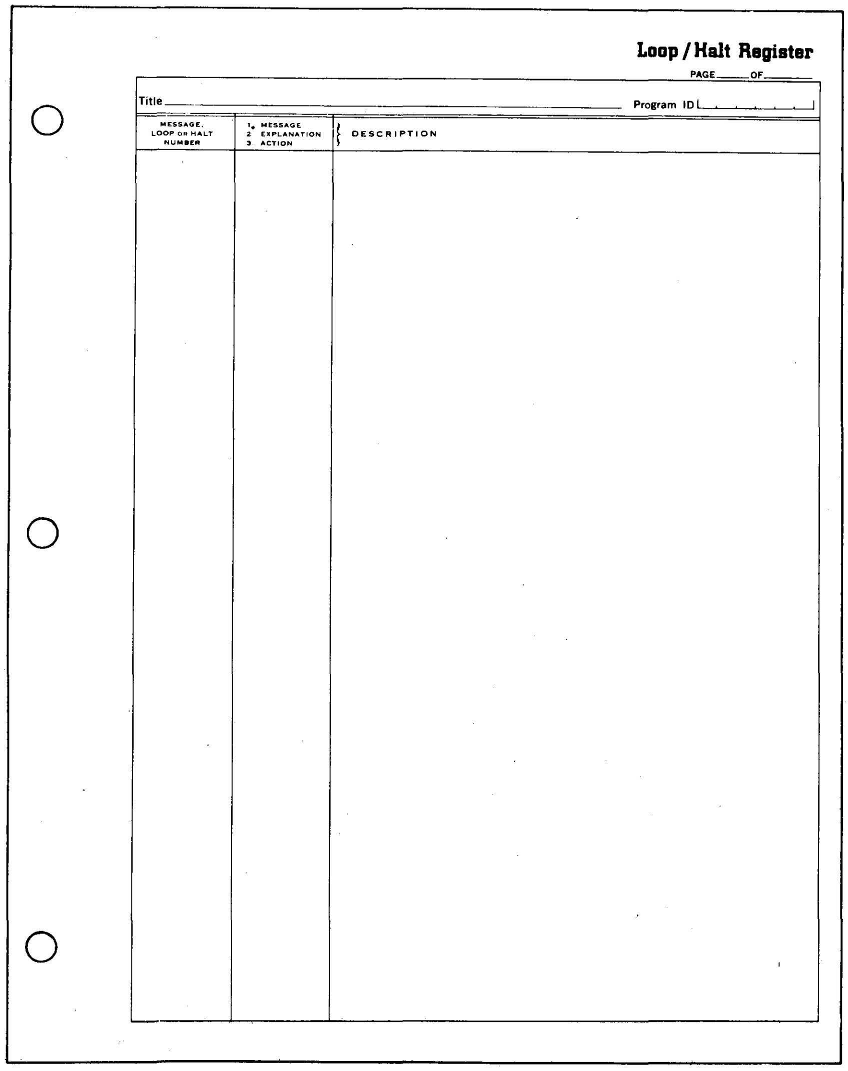
The greatest benefit from a test session can be realized only if complete documentation is brought to the test center. A consistent system of notation should be adopted and all material marked so that it can be easily associated with the corresponding program. A check list of materials follows:

1. All pertinent block diagrams.
2. Punched and verified source program decks.
3. Assembled program listing and object deck.
4. Verified listings of master file data and test input.
5. Test samples, intermediate checkpoint results, and predetermined final results in output format.
6. Layouts of card input and output, disk pack records, and core storage.
7. List of halt definitions and restart procedures.
8. Operator's check list.
9. Carriage tapes and preprinted forms.
10. New source decks to be assembled.

Other useful supplies would be blank layout and programming forms, reference manuals, paper, pencils, erasers, marking pens, rubber bands, rulers and flowcharting templates.

Most of the materials listed thus far will be used chiefly for the debugging periods between machine sessions. During the actual testing, only the necessary minimum amount of materials should be taken into the 1440 room.

Particular attention should be given to preparing detailed instructions for programmed halts (Figure 56). Each halt should be identified by either its I-Address Register or A-Address Register setting, or



Loop / Halt Register

PAGE____OF____

Title________________ Program ID

MESSAGE, LOOP OR HALT NUMBER	1. MESSAGE 2 EXPLANATION 3. ACTION	DESCRIPTION

Figure 56. Loop Halt Register

both, and the specific reason for the halt should be given. This documentation should include information for restarting the program and the action to be taken specifically regarding this halt. Testing is greatly facilitated when the people involved in the test procedure are thoroughly familiar with the programmed halts and necessary action. Often a great deal of time is wasted because a legitimate halt is not recognized or the proper corrective action is not known.

Decks to be used more than once, such as utility routines or those containing input cards that will have output information punched into them, should be prepared in duplicate to prevent loss of time at the test center in case the cards become mutilated or out of order.

Emphasis should be placed on the necessity for labeling all material clearly and placing it in the same sequence as the order of programs to be run. This is essential for an efficient, well coordinated test session.

Schedule of Operations

A test schedule of operations should be developed. This schedule will help assure that all tests are performed in their logical order, use the proper data, and are checked to the proper result in the most efficient manner. Provision should be made to log any significant result or departure from the expected operation. In addition, there should be a method for recording any program changes, and there should be a procedure for handling any unexpected errors or halts.

Sufficient time should be allowed between machine tests for careful debugging, checking the effect of corrections and additions to the program, and for reorganizing the next session in the light of the one just completed.

Final Review

The last step to be accomplished before the test session is a review of all preparations. At this time it is important to redefine the objectives of this session and see that they are completely understood by all concerned.

The test center operation is a verification of past effort, and the most significant test result is one that proves this effort has been nearly perfect. If any of the preparations are incomplete or inadequate, the tests should be canceled and the trip rescheduled for a later date.

Test Scheduling

Early in the installation period it should be decided what programs will be tested and how much time will be needed. An experienced systems person can help estimate the time required to write the programs, taking into consideration the programmers' experience, aptitude, and job complexity. On the basis of estimated completion dates, test time can then be set up on a long-range basis. The results of the first test session will determine when further testing should begin and how much time should elapse between test sessions. This may vary according to the number of programmers, the magnitude of the job, and the distance to be traveled. Initial programs will need more testing per instruction than those written after the programmers gain more experience.

Some groups find that about two weeks are required between sessions for corrections and additional preparations and that scheduling two or three sessions every second or third week answers their needs best. Other groups, with a test center easily accessible, may find that one session two or three times a week will better suit their requirements. Remote testing is advisable as soon as the programmers complete initial familiarization with testing procedures.

The amount of time required for testing should be requested well in advance. Information on test allowances and options, testing locations, and necessary procedures will be supplied by the IBM representative. General information about test center system specifications, materials and supplies, transportation, and accommodations should be reviewed in detail several weeks before the scheduled date of the first test sessions. Special attention should be given to the specifications of the testing machine to make sure any particular programming considerations will be accommodated. Testing dates at the data processing center can then be arranged through the IBM representative.

If desired, materials may be shipped to the test center before the scheduled testing date. Ample time should be provided for their arrival, and the test center manager notified as to the method of shipment and/or the carrier. Materials should be clearly labeled and itemized and arrangements made for their return, since most test centers have limited storage facilities and cannot retain the material between trips. Source and object decks should be duplicated, particularly when being shipped, to prevent the necessity of repunching, should losses occur.

Testing Sessions

First Test Session

The first visit to the test center should be scheduled as early in the programming effort as practical. A prime objective of the initial test should be to familiarize the programming group with Autotest and system operation and to confirm their programming concepts.

Plans should be made for the testing of only a few simple programs. Emphasis should be placed on proper machine operation and testing procedure at that time rather than complete testing of the programs. The main highlights for the group are the learning of machine handling and techniques, the observation of utility program usage, and instruction in Autotest and the procedures for testing programs.

Two or three machine sessions of a half-hour each can be scheduled for the first day, about three hours apart. The first session is usually devoted to an orientation on the 1440 system and Autotest, and is generally conducted by one of the test center representatives. He explains each unit of the system in detail and demonstrates its operation. Particular attention is given console operation to prepare the group for its later use in testing. The programs brought by the group can then be assembled, and will give the group an opportunity to learn this aspect of testing. After the initial machine session, the group can use the time before the next session to check the postassembly listings for errors and make any necessary corrections.

During the next machine session, an IBM representative will again be present to assist in the actual running of the programs. The group should be as interested in learning testing techniques and program debugging as in seeing the results of their programs. Machine setup preparatory to running the program, quick recognition of the reasons for machine halts, necessary corrective procedures, program restarts, and the retrieving of all required information for later analysis should be highlighted during this session.

After these first sessions, the group will be better prepared to begin work on the more complicated programming efforts and will be able to handle the testing of larger, more complex programs with greater facility.

Arrival at the Test Center

Early arrival should be planned, particularly on the first visit to a test center. Time is needed to become familiar with the surroundings and test center procedures. In addition, test center representatives are available to give the latest information on programming systems, suggest helpful methods for testing and debugging, and answer any questions.

A conference room or cubicle is assigned to the group for use while at the test center. This room can be used for the work to be done between machine sessions and for storing all testing materials.

Before going to the 1440 room, materials should be checked and everyone given an outline with the sequence of programs to be tested. The group should be in the computer room with the materials, ready to go on the system promptly at the appointed time.

Only the people actively involved in the testing should be present during the machine session. The presence of too many people around the system tends to slow down the testing and inhibits a smooth and efficient operation.

Autotest

Autotest is used at the test centers to facilitate testing. It is a monitor program which allows application programs to be batched sequentially for testing; provides automatic file generation, automatic storage and file dumps; allows individual program operating instructions and program patching; and permits areas of core storage to be printed at particular points in the program (a "snapshot"). (The features of Autotest are outlined in the "Programming Aids" section of the manual and are described in detail in *Autotest for the* IBM *1440. Preliminary Specifications,* C24-3058.)

Program decks to be tested in this manner are set up with appropriate control cards, master file data, operating instruction cards, object program and test data. The programs can be batched in order of importance and in any quantity. The Autotest package will print the operating instructions and allow the operator time to implement them, create the disk files, load the object deck and begin processing of the test data. When the program test is completed, automatic storage and disk printouts will be given and the next program called in for the same procedures. Complete information concerning each program appears on the printer and in the punch pockets.

If a processing error occurs as a program is being tested, the operator can record the information displayed on the console (Figure 57) and manually return the computer to control of the Autotest package. The monitor will give the storage and file printouts, skip the rest of the test data for that program, and then go on to the next program.

The greatly increased throughput speed made possible by the use of Autotest enables many more programs to be tested. In addition, reduced operator intervention and automatic card handling eliminate many operating errors and help increase the productiveness of the session.

Completion of Machine Session

The above Autotest procedures are used for each program to be tested until either the testing is completed or the allotted time has passed. The group should then vacate the area promptly, being careful to run remaining cards out of the 1442, remove printouts from the 1443 and 1447, and collect all other materials. Before leaving the machine room, the group should check with the test center representative to see how much of the test allowance was used. He

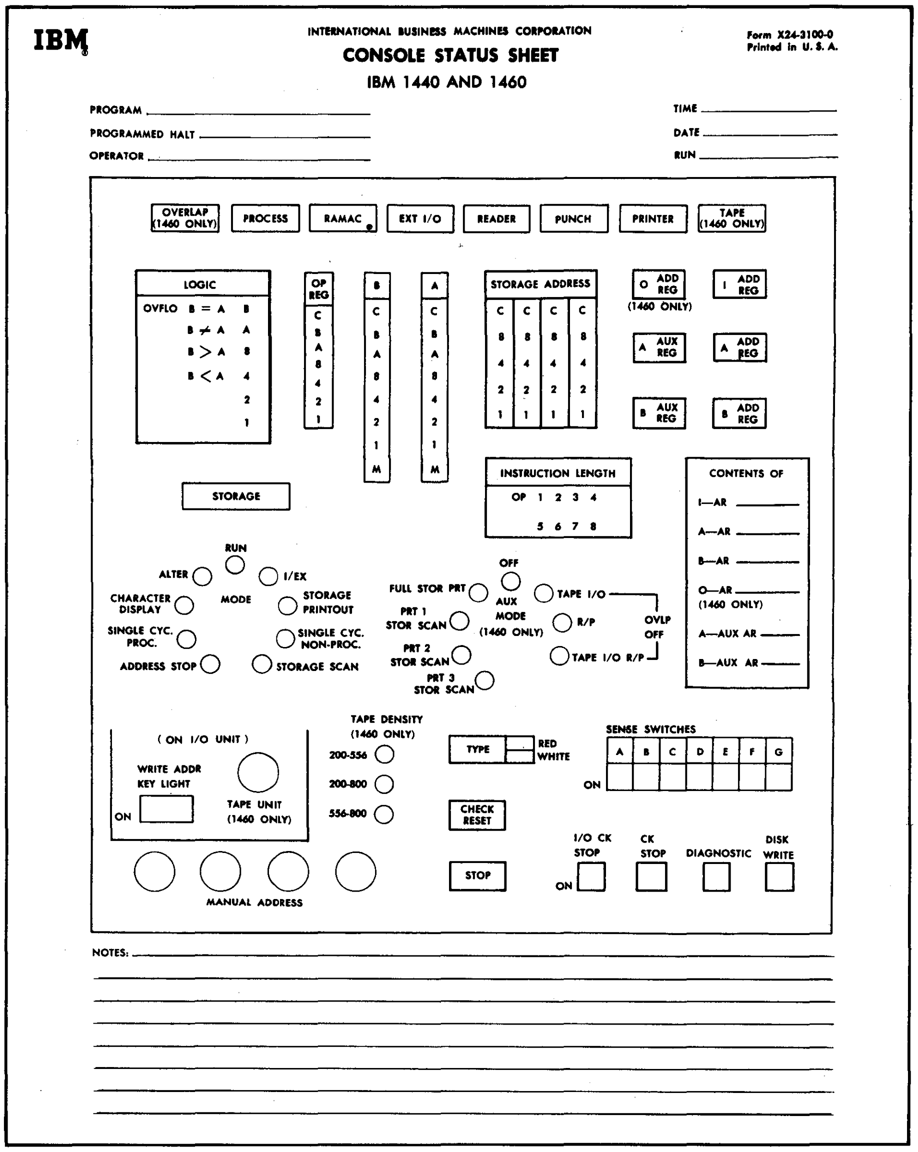


Figure 57. Console Status Sheet

should be informed at that time of any time lost during the session which was not chargeable to the group, so that it can be immediately deducted from the elapsed time.

Desk Debugging

Upon completion of each machine session, the group can return to the assigned room or cubicle to debug any programs which failed to reach the end of job or did not produce correct results. To locate and correct program defects, the program listings, record layouts, block diagrams, disk pack dumps, storage prints, and console recording sheets are used.

The storage print or memory dump is a primary tool in the program checking. Programmers are often tempted to ignore the fact that the storage print and not the program listing is the last word on what storage contained. The storage print and console sheet should first be checked to see what caused a program error. Then comparison can be made with the program listing. The storage print is especially useful in determining the contents of input, output and work areas at the time of the error.

In addition, the storage dump allows the programmer to inspect the results of any program modification and provides a complete listing of all intermediate and final results computed up to the point of the dump; it shows exactly what test data was used, and allows the programmer to determine whether anything in storage has been modified which should not have been.

To locate an error, the programmer should start at the point where the program stopped and work backward, analyzing each instruction until he finds the one which caused the problem. Frequently this is an instruction some distance away which set up a condition causing a stop later in the program. Logic instructions, such as Compare or Branch If Zone, can be the cause of this type of problem. Also, instructions which erroneously alter storage areas will cause subsequent operations to be performed incorrectly.

If the programmer has already surmised that the error halt was caused by a condition occurring some time previously in the program, he may start his checking from a point at which the program was known to be functioning correctly and then work forward to the error halt.

A good rule while checking is to assume that each step is wrong until examination of storage contents proves otherwise. Reference to block diagrams, record layouts, listings, and input listings simplifies the task. If the programmer has difficulty in locating an error, it may help to have someone assist him since errors can frequently be located more easily and faster by someone who has a different perspective on the job.

The programmer should not be satisfied when the error which caused the stop is located. He should step further through the program to make sure that the next instruction or routine will not cause another stop. He should strive for perfect output from each test.

Programs which did not stop during the test run have to be checked against the precalculated results to determine whether they were operating correctly. Arithmetic and editing errors are prevalent here. Programmers should be careful to analyze disk printouts as well as the card and printer output.

Another important checkout is the verification of the intermediate or partial results. This allows the source of errors to be narrowed down successively. It also assists in proving the correctness of a program and in checking that no undetected errors remain. The intermediate results may be obtained by using the snapshot feature of Autotest.

If a program seems to be completely correct, final checking should be done back at the home installation. A program which seems to have satisfactorily completed its tests should be reviewed in complete detail to verify its accuracy, and then should be checked with the persons responsible for the data in the report and for the use of the report to be sure it incorporates all of their requirements.

Corrections of Errors

All errors encountered in the test session should be thoroughly analyzed with two questions in mind: How does the correction of the error affect the rest of the program? And has the same error been made anywhere else in the program? A poor job of desk debugging has usually been done if the same type of error occurs again in the next test of the program.

With this in mind, correction of the errors can take place. In the case of a simple error such an an incorrect address, a correction can be made by repunching the object program card. In other cases, additional instructions have to be inserted in the program by patching. To do this, patch cards are punched in the Autopatch format and are inserted at the end of the object program deck. When a number of patches have accumulated in a program or when extensive program revision is required, the necessary changes should be made to the source deck and the program reassembled.

After corrections have been made and materials updated and reorganized, the group can then go back to the 1440 to test the programs again. The same procedure is followed until the program is free of errors.

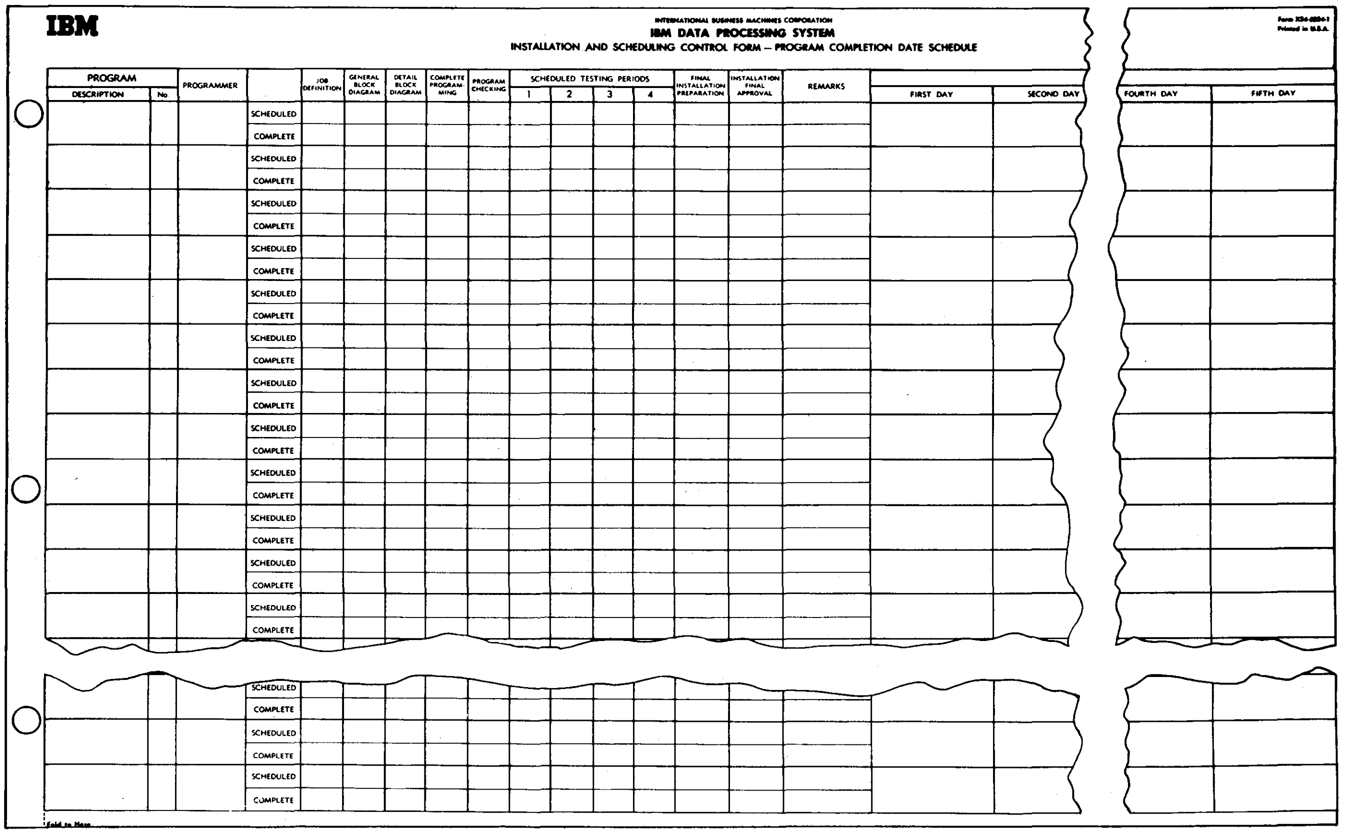
Post-Test Evaluation

Each visit to the test center should be followed by an evaluation of the total test session. Pretest preparation should be reviewed to decide whether it was adequate or whether improvement is needed. Errors should be analyzed to see whether any area of weakness in the coding was revealed and, if so, what steps can be taken to improve it. Performance during machine sessions should be examined for effectiveness and corrections made to improve techniques. The status of each program should be reviewed to determine what additional work is needed.

Consideration should be given to recording the status of programs on a test log sheet, such as the one shown in Figure 58, to provide a current report of testing progress. The log should show which programs have been assembled and the number of times each has been tested; it will thus be a record of what has been accomplished to date and what further work needs to be done.

Objectives set for the test session should be reviewed to see whether they were accomplished. Future testing schedules should be considered to see whether any changes or revisions are needed.

When it is felt that a program is completely tested and can be filed, all related materials should be carefully checked for completeness and accuracy. The established system of documentation will help keep all materials properly organized. No program should ever be set aside unless all its documentation is up to date.


IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
IBM DATA PROCESSING SYSTEM
INSTALLATION AND SCHEDULING CONTROL FORM — PROGRAM COMPLETION DATE SCHEDULE

Form [illegible]
Printed in U.S.A.

PROGRAM		PROGRAMMER		JOB DEFINITION	GENERAL BLOCK DIAGRAM	DETAIL BLOCK DIAGRAM	COMPLETE PROGRAMMING	PROGRAM CHECKING	SCHEDULED TESTING PERIODS				FINAL INSTALLATION PREPARATION	INSTALLATION FINAL APPROVAL	REMARKS				
DESCRIPTION	No								1	2	3	4				FIRST DAY	SECOND DAY	FOURTH DAY	FIFTH DAY
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																
			SCHEDULED																
			COMPLETE																

Figure 58. Program Completion Date Schedule

Assembly Procedures

Careful planning is needed to get maximum use of the machine time required for program assembly. In addition to the machine and desk-checking procedures stated earlier, programs should be thoroughly checked for inclusion of the required control cards, proper sequence, labeling, etc., before being taken to the machine for assembly. At the computer, careful attention should be given to the setup so that time is not lost in restarting the assembly process.

A minimum of two or three hours should ordinarily be allowed between assembly of a program and its first test. If possible, it is more desirable to allow at least a week between assembly and test of a program to enable the programmer to make all patches needed, become familiar with the assignment of core by the assembly program, etc., and to continue to do some desk checking with the assembled program listing. Many users assemble new programs during a test session of other already assembled programs, and take the cards and the assembly listings to the home locations for checking prior to attending a subsequent test session.

After programmers have gained experience in running assemblies on the computer, they can begin to take advantage of the remote assembly facilities offered by test centers. This is a service available to testing groups which enables them to send source programs to the test center to be assembled by the IBM personnel there. Post-listings are returned in time to permit checking and patching before the actual test session.

Decks sent to the test center for assembly prior to the test session should always be duplicated and should be clearly labeled. Explicit instructions should be enclosed for the assembly operation, and clear directions given for disposal of decks and post-assembly listings. Sufficient time should be allowed for their processing and return before the scheduled test sessions. Machine time used at the test center for running remote assemblies is charged against the test allowance and provision should be made for recording this time.

Use of remote assembly facilities will save time and effort for the programming group and increase efficiency both in pretest preparation and testing operations.

Remote Testing

After sufficient experience has been gained in the test center to understand the problems involved in testing, the group can begin to use the remote testing service offered by IBM. They need no longer go to the test center in person but can send their programs to be run by test center personnel.

Complete data should be sent — Autotest control cards, decks, test data, special instructions and directions for return of materials. The test center personnel will run the program, using Autotest. They will not debug or alter the program in any way. All decks and output material will then be returned promptly, with a notation as to how much test time was used.

Autotest is especially useful for remote testing. It incorporates all the features necessary for having programs run on a remote basis and helps insure that the programmer will not forget any details necessary for the proper running of his program. It also makes retesting of programs much easier and simpler. Once the decks are set up in the proper sequence and with appropriate control cards, they need not be changed; only the patch cards making necessary corrections will be inserted in preparation for the next test.

Considerable savings in time and expense can result from use of this remote testing service. Since the transportation to the test center, the living expenses, and the time and effort involved in the conducting of a test session can be costly, efforts should be made to convert to remote testing as soon as feasible. The preinstallation period can be facilitated and simplified by use of this method.

Postinstallation Testing

The final check that should be made on a data processing program is a volume test, or what is known as pilot operation. A large number of actual transactions from a previous period are used, so that any possibilities which may not have been considered in making up original test cases are brought to light. The results produced by the computer system can be checked against the results produced by the previous manual or unit record methods. Since the transactions have already been processed, there is no pressure for results and time is available for careful analysis of the program operation.

One of the benefits of pilot operation is that it enables the department for which the work is being done to see whether the program actually does what was intended. It is easy for people to give an incorrect impression when stating a problem, or to overlook some special conditions and situations which may arise. Therefore, the program cannot be actually finished until it has been proved that it produces precisely correct results using actual data in volume.

This means that testing will be a continuing process after the 1440 is installed. Programs need completion, modifications will be made, new applications developed, and more requirements added. The fact that testing procedures used in the preparation period are carried into this stage constitutes another reason for developing good habits when first going into the testing phase.

Physical Planning

General Considerations

A 1440 installation requires that certain physical requirements be considered early in the preinstallation planning stage and that these requirements be completed before the system arrives. While the 1440 system does not impose as extensive environmental preparation as earlier nontransistorized systems, certain tolerances must be maintained.

Electrical Power Supply

The amperage requirements vary with the model of the system and the number of components and can be determined by referring to page 6 of IBM *1440 Data Processing System Installation Manual — Physical Planning* (A24-3007). Power sources should be avoided that supply, in addition to the 1440 system, large cycling loads such as elevators, air conditioning units, arc welders, etc. The electrical requirements of the 1440 system are 208 or 230 volts, 60 cycles, 3 phase, 4 wires. The fourth wire is a noncurrent-carrying ground wire, and should be an insulated grounding wire carried back to the electrical service ground or other suitable building ground.

Power to the 1440 system is furnished through a power cord attached to the 1441 unit. This cord is rated at 60 amperes and is terminated in a Russell and Stoll plug #7328. IBM furnishes the plug on the power cord; the receptacle which will accept this plug should be furnished and installed by the user. Sufficient lead time must be allowed on ordering and installing electrical equipment to assure that the receptacle is installed and power is available when the IBM customer engineers are ready to apply power to the system. There should be several 115-volt convenience outlets located adjacent to the 1440 system to facilitate servicing.

The power feeder serving the machine room should be equipped with a remote-operated main-line circuit breaker. This will permit all power to be shut off from a single location in the event of an emergency.

If sufficient power for the 1440 system is dependent on other equipment being released, it should be remembered that for the installation and conversion period, it will be necessary to have power on both the 1440 system and the equipment it is replacing.

Environmental Conditions

The 1440 system with power on requires room temperature between 60° and 90°F. and relative humidity between 10 and 80%. (If other equipment, such as the 1412, is installed in the same room, closer limits prevail.) Air conditioning requirements can be approximated by determining the total BTU output (see page 6 of the 1440 physical planning manual) and allowing 12,000 BTU/hr. per ton of refrigeration. Other factors to be taken into consideration are the normal room load (lights, solar load, etc.), the number of people, volume of traffic, and the amount of offline equipment that will be in the 1440 room. Design conditions of 75°F. and 50% relative humidity are often recommended since it is not desirable to operate the system at or near any of its limits. A slightly lower relative humidity of 35-40% may be desirable in the winter to minimize condensation on windows.

It should be remembered when evaluating the existing building air conditioning system that this system is often used to heat the building in winter and might be incapable of sufficient cooling on a year-round basis. Also, if second or third shift operation is planned, the use of an independent air conditioning system for the 1440 room may be necessary to eliminate running central air conditioning systems at night.

An instrument with a seven-day chart recording both temperature and humidity should be provided in the machine room. This can be a valuable aid in sensing impending temperature or humidity problems caused by air conditioning malfunction or other factors. A pilot-light indicating system to warn when the room conditions are reaching specified limits is helpful and may prevent interruption of machine operations.

Flooring

The 1440 system should be installed on a sound building floor capable of supporting the weight of the various units (see page 1 of the 1440 physical planning manual). If the soundness of the floor is in doubt, the services of a structural engineer should be secured. If a raised floor is installed to conceal interconnecting cables and power receptacles, it should be constructed of noncombustible, nonconductive material. For safety reasons, there should be no exposed metal on the floor surface. A floor covering which does not crack or flake, such as pure vinyl, is recommended.

If a raised floor is not installed, a machine arrangement should be devised which will allow the interconnecting cables to be placed out of the traffic area around the machines. The use of ramps to protect the exposed cables should be considered.

Work Space

The need for a proper amount of work storage space in the 1440 machine room cannot be overemphasized. Work tables, card cabinets and disk pack storage cabinets should be strategically placed so as to fulfill the requirements of the operating and programming personnel present in the machine room. An area should be allotted within the machine room, or directly adjacent, for customer engineering use.

Room Preparation

The installation of the 1440 may require adding a room or renovating existing space. One effective way of handling this problem is to engage the services of an architect and advise him of all the requirements which must be met. Another approach is to engage individual contractors to handle the specific items necessary in the site preparation. Regardless of the method, a scale room layout (form X24-3083 contains scale templates of the 1440 components) specifying all the requirements must be drawn and agreed upon to allow the interconnecting cables to be ordered on schedule. The time allowed for room construction prior to the time the system arrives will depend on the extent to which the room is being modified. Room preparation should be completed at least one week before the system is shipped.

The movement of supplies and machine units should be considered in the selection and preparation of a computer site. The path the equipment will follow from the delivery platform to the 1440 room must be surveyed to determine the adequacy of the doorway dimensions, elevator capacity and clearance space in halls, especially at corners.

Paint, draperies and carpets all add to the attractiveness of an installation. However, the material in the draperies and carpets should be lint-free and easily cleaned so as not to create a dust factor which might be detrimental to good machine operation. Paint and soundproofing material should be of a type that does not flake or dust.

All intended construction should meet with insurance requirements as well as local building and electrical codes. As a safety factor, the room should have portable carbon dioxide fire extinguishers located in easy-to-reach places. A fire alarm system incorporated in the room provides an additional safety device. If the construction of the building makes the use of sprinklers necessary, the sprinkler heads in the 1440 room should have a minimum rating of 175°F. in order to minimize the possibility of accidental discharge.

Cleanliness and Appearance

A neat and well organized machine room will promote the operating efficiency of the system. Dirt, dust and other foreign particles can be a definite hazard to good machine operation. Arrangements should be made to have the room cleaned daily by cleaning the floor with a damp or treated mop and by vacuuming any rugs in the area. Sweeping, cleaning with dust cloths, and other cleaning methods which create dust in the air should not be used. Any floor wax used should be applied very lightly and buffed until it has a hard surface; waxes which have a tendency to flake after application should not be used. No steel wool or other abrasives should be used in buffing the floor.

Further details on the above subjects are available in the following manuals:

IBM *1440 Data Processing System Installation Manual – Physical Planning*	A24-3007
Physical Planning	F24-1052

Conversion

Basic Considerations

The conversion from the old procedure and equipment to the new one should be as thoroughly planned as the permament operating procedure. Some of the key decisions to be made in conversion planning involve the scope of the intial data processing functions, in what order the functions will be converted, and how the cutover will be tested. The major considerations in these decisions are briefly discussed below.

Dividing the Conversion Job

It is unlikely that anyone has ever converted a complete data processing installation all at once; provision is usually made to divide the functions and install them sequentially. One method of doing this is to completely prepare one application (or a limited number of applications) for operation at installation time, leaving the other applications for subsequent dates. For example, if payroll were to be converted, it would include all plants, branch offices, departments, etc., from the outset. Under this method conversion plans for one or a few applications extend throughout the organization.

Another approach is to plan initial conversion of a greater number of applications, but to simplify them by handling them on a partial basis – that is, as they affect only a few departments or a portion of the business. For example, payroll could be cut over one department at a time, or billing, one cycle at a time.

Another method is to follow very closely, at first, the approach used previously and, as experience is gained, change to a more integrated one. For example, a unit record procedure could be closely paralleled at first and later modified to take greater advantage of 1440 capabilities.

These are only some of the possibilities; in choosing the approach bear in mind that there is a tendency to underestimate what is required and that it is easier to speed up an underambitious approach than to effectively modify an overambitious one.

Sequence of Applications to Be Converted

Another basic decision to be made is the sequence in which the various applications will be converted. One consideration is the saving to be realized when a given job is on a production basis. Since each application will have its own justification, an analysis will normally indicate which jobs should be started first.

A second consideration is the complexity of the conversion job. Some of the applications will not be changed as much as others, and thus will probably present an easier conversion task. For example, applications already being processed on unit record equipment will usually not be as difficult to convert as jobs presently performed on a manual basis.

Release of installed equipment will be another factor. Present machine utilization for each application should be considered. Availability of time on the 1440 during the conversion is another consideration. Working with the estimates of machine usage a determination can be made of the workload on the system at each point during the conversion operation. Time estimates for one-time jobs that will be run during conversion should be included. System utilization for a given job will be somewhat greater during conversion than it will be during production. Reruns may also be necessary in those cases where verification of output indicates incorrect processing of one or more data records. Revisions to programs or procedures to correct such discrepancies may increase the running time.

Pilot and Parallel Operation

The two basic methods for proving the accuracy of a new procedure are parallel and pilot operations. Some installations will use a combination of these methods because of varying application requirements.

Parallel operation is the simultaneous processing of source data through both the old and new procedures. The results of the two procedures are then compared. This operation is usually continued for at least one accounting cycle.

Pilot operation consists of processing data from a previous period, usually on a full-volume basis. The output is checked for accuracy by comparison with the results from the previous period. Under pilot operation it is not necessary for the two systems to have access to the same records concurrently, and for this and other reasons, pilot operation is usually a better approach to select.

In some cases it will be difficult to compare results of the old and new systems because reports will have been added or eliminated and other changes made in the functions of the system. A further problem is that discrepancies found may not all represent errors in the new procedure. In these instances it will be necessary to examine the results of the new procedure in considerable detail, manually duplicating some portions to verify accuracy.

When agreement is reached that the new procedure is accurate and functions properly, the old system can be discontinued. The amount of parallel or pilot processing to be performed will not be the same in all instances, and extremes should be avoided.

Conversion Time Schedules

A realistic time schedule for conversion should be developed, based on the amount and type of work to be done as well as on the personnel and equipment available. Additional personnel and/or equipment may be needed on a temporary basis to complete preparations for conversion.

The job of converting each application should be assigned as a specific responsibility. Some of the tasks involved in a conversion operation are listed below and are discussed in subsequent paragraphs.

- Orientation of personnel.
- Gathering the data required for the master files.
- Editing the files for completeness, accuracy and correct formats.
- Creating new files and procedures to maintain them until production starts.
- Providing for training personnel in the departments which supply source data or receive processed data.
- Establishing schedules for cutover to the data processing system.
- Planning for pilot or parallel operation.
- Coordinating actual conversion.
- Cross-checking the results of processing by the two systems.

Orientation of Personnel

The conversion will be more easily implemented if all personnel involved have a clear understanding of what is to be done.

Preparation of the source data for the system is one of the essential areas. The persons responsible for this preparation must be thoroughly familiar with what will be required. It is necessary that they know the reasons for changes to previous procedures and the effect of various types of data on the overall system. It is important to issue clear, detailed instructions in written form. These instructions should cover all phases of coding, card punching, and creation of information.

In some instances it will be necessary to provide the same input data in different formats during the conversion period. This means an extra workload for the department supplying the data, and adequate facilities must be provided to assure that all schedules can be met.

Education of the people responsible for the source records will prevent many problems that could otherwise prolong the conversion period. Cooperation from these people will be of great assistance toward a smooth transition.

Persons who will receive the output from the system should also be educated as to the results of the system. Some reports will be produced in revised form, and all users of these reports should understand the revisions. Similarly, everyone should be advised of any changes in the schedules for the preparation of various reports. Misunderstandings in this area can be a source of considerable controversy but can be minimized by keeping everyone informed.

It will normally be necessary for departments receiving documents from the new system to verify the completeness and accuracy of the reports. A procedure should be established for reporting discrepancies so that the data processing department can make the necessary system revisions. Close cooperation between the departments will facilitate the job of locating and correcting problem areas.

Conversion of Master Files

Conversion of master files should begin before actual machine installation. Often the data to be included is gathered from several sources and its compilation and editing may require extensive clerical effort. Emphasis should be placed on the accuracy of the information. Control totals should be established where applicable and the new card or disk pack files balanced to these totals.

After the master files have been established, they must be maintained until they are used in production procedures. Controls and routines for additions and deletions should be set up and all personnel should be briefed in these procedures. One person should be assigned the responsibility of maintaining the files and the accuracy of the control figures.

In generating the new files, it is often desirable to use installed unit record equipment. Gang punching or reproducing information can reduce the card punching load. Unit record equipment can be used for listings as well as for assigning numbers, sequence checking, and editing for valid codes. Usually several special 1440 programs are required for converting the data to disk pack records or for editing and verification of card files.

Initial Loading of Disk Packs

Programming aids such as the card-to-disk utility program or the file organization programs can often be used to load the disk packs with the information contained in card files. This will result in savings in programming time and effort since the programming required can be limited to the preparation of control information, stated in terms of the desired results. The possibility of errors is reduced by the limited quantity and the simplified nature of the programming.

During this stage of conversion, it is important to make sure that audit trails are provided. It may be advisable to create duplicate disk packs containing the newly created master information. Another point to note is that the interchangeability of disk packs may allow for initial file loading on another system (1440, 1401, 1460, 1410) prior to installation. Use of the data center or a service bureau for this purpose may facilitate conversion.

Conversion of other disk files, such as the 305, involves considerations similar to the above, in addition to the necessity of punching out the information from the RAMAC® file.

Conversion Pointers

Conversion procedures often represent a major portion of effort in installing a data processing system. Application design should be compatible with the ability to realistically gather and incorporate the required data.

When faced with extensive conversions, thought should be given to an interim system approach short of final objectives. While such a two-step approach can be criticized for increasing the programming effort, the advantages of smoother conversion and faster system production are powerful incentives to adopting it. An added advantage lies in the fact that a two-step programming effort will allow the programming group more time for the development of techniques and skills. The second-step programs should be of a higher degree of quality and sophistication than the first-step programs.

Combining punched card files in consolidated disk records often exposes many hidden errors. Edit runs should be performed on each file, testing for invalid punching, coding errors and completeness of data. Reconciliation procedures need to be worked out beforehand in the event of unmatched conditions arising. A suggestion is to maintain card files separately until such time as all errors are resolved, rather than combining incorrect data and then attempting to clean up the errors and missing elements.

Develop a detailed conversion schedule well in advance. Define starting and completion dates for each segment of the conversion. Programs necessary for conversion should be accorded ample time and consideration. If volume keypunching is required, perform a pilot study to establish time requirements.

Depending on programming and systems people to carry the full load of conversion is usually not advisable. A better suggestion is to employ the personnel who will eventually become the system operators to aid in conversion. This will give these people a better understanding of the application, plus valuable experience in systems operation.

Remember that many departments within an organization will affect and be affected by the new system and that the establishment of good communications and controls between departments prior to conversion is of the utmost importance.

Introduction of proper data control techniques is neccessary during the conversion period. This is the best time to instill good habits in the area of controls and audit trails, as the pattern that is set during the conversion will become permanent practice. It is important that all changes made to the programs or the procedures during the conversion be properly documented.

One common error in preparing for the installation period is the lack of a realistic conversion time schedule. Conversion transitions take time, much of it hidden. Following are some of the areas requiring careful consideration:

Final program testing. – Time must be allowed for extensive program testing. Ideally, production data should be used in final tests. If the change in application design dictates otherwise, it is suggested that the test data be compiled by other than the programmers. This is one good method for eliminating many of the errors that too often do not show up until production runs are attempted.

Clerical retraining. – It takes time for clerical and keypunching personnel to develop speed and efficiency in handling and preparing new source data. Coding errors are to be expected, especially during this initial period. Editing is an important function of an EDP system. Normally, editing functions should be accomplished in short, preproduction programs, rather than be incorporated into main programs where cumbersome and difficult restart procedures can be encountered.

Program run times. – Allow for realistic initial program run times, remembering that optimum run times cannot be expected during initial installation periods. Job setup times are bound to be out of proportion until operator proficiency improves through experience. Program reruns are to be expected.

Planning and Progress Control

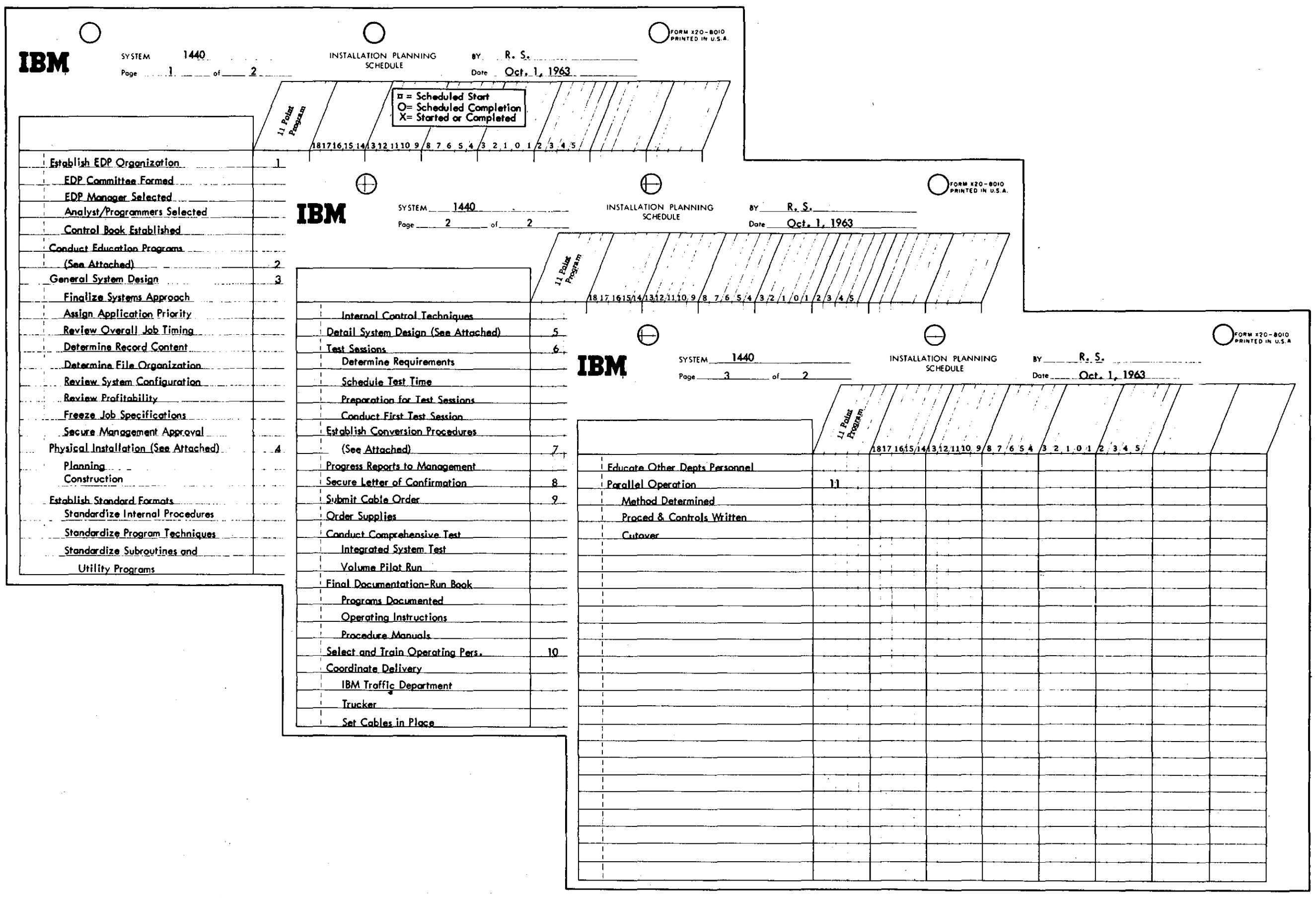
Each of the areas described previously — education, systems design, programming, physical planning and conversion — requires that a great many related activities be carried out. To assist in the planning, scheduling and control of these activities, it is advisable to establish a series of installation schedules. These schedules provide both a check list of the jobs to be done and a means of scheduling and following up work. Some typical schedules and the reasons for their use are described below.

General Preinstallation Schedules

One of the first tasks of the data processing manager is to prepare a general overall preinstallation schedule. The schedule should be established as realistically as possible and modified, if necessary, as the detailed study and programming get under way. Frequent reviews should be made and action taken when needed to assure that the program is progressing satisfactorily.

Figure 59 is a typical overall planning schedule setting up target dates for all major activities. Note that where required — for example, under "Conduct Education Programs" — reference is made to other schedules providing more detailed information. The eleven points shown on the form refer to the IBM eleven-point program for installation planning.

While it is impossible early in the installation cycle to pinpoint the exact dates for detailed systems design, program testing, etc., target dates should be set to represent either estimates as to when these steps can be accomplished or the dates for establishing the detailed plans for the required areas.


IBM SYSTEM 1440 Page 1 of 2 INSTALLATION PLANNING SCHEDULE BY R. S. Date Oct. 1, 1963 FORM X20-8010 PRINTED IN U.S.A.

¤ = Scheduled Start
O= Scheduled Completion
X= Started or Completed

11 Point Program

18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5

Establish EDP Organization	1
EDP Committee Formed	
EDP Manager Selected	
Analyst/Programmers Selected	
Control Book Established	
Conduct Education Programs	
(See Attached)	2
General System Design	3
Finalize Systems Approach	
Assign Application Priority	
Review Overall Job Timing	
Determine Record Content	
Determine File Organization	
Review System Configuration	
Review Profitability	
Freeze Job Specifications	
Secure Management Approval	
Physical Installation (See Attached)	4
Planning	
Construction	
Establish Standard Formats	
Standardize Internal Procedures	
Standardize Program Techniques	
Standardize Subroutines and	
Utility Programs	

IBM SYSTEM 1440 Page 2 of 2 INSTALLATION PLANNING SCHEDULE BY R. S. Date Oct. 1, 1963 FORM X20-8010 PRINTED IN U.S.A.

11 Point Program

18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5

Internal Control Techniques	
Detail System Design (See Attached)	5
Test Sessions	6
Determine Requirements	
Schedule Test Time	
Preparation for Test Sessions	
Conduct First Test Session	
Establish Conversion Procedures	
(See Attached)	7
Progress Reports to Management	
Secure Letter of Confirmation	8
Submit Cable Order	9
Order Supplies	
Conduct Comprehensive Test	
Integrated System Test	
Volume Pilot Run	
Final Documentation-Run Book	
Programs Documented	
Operating Instructions	
Procedure Manuals	
Select and Train Operating Pers.	10
Coordinate Delivery	
IBM Traffic Department	
Trucker	
Set Cables in Place	

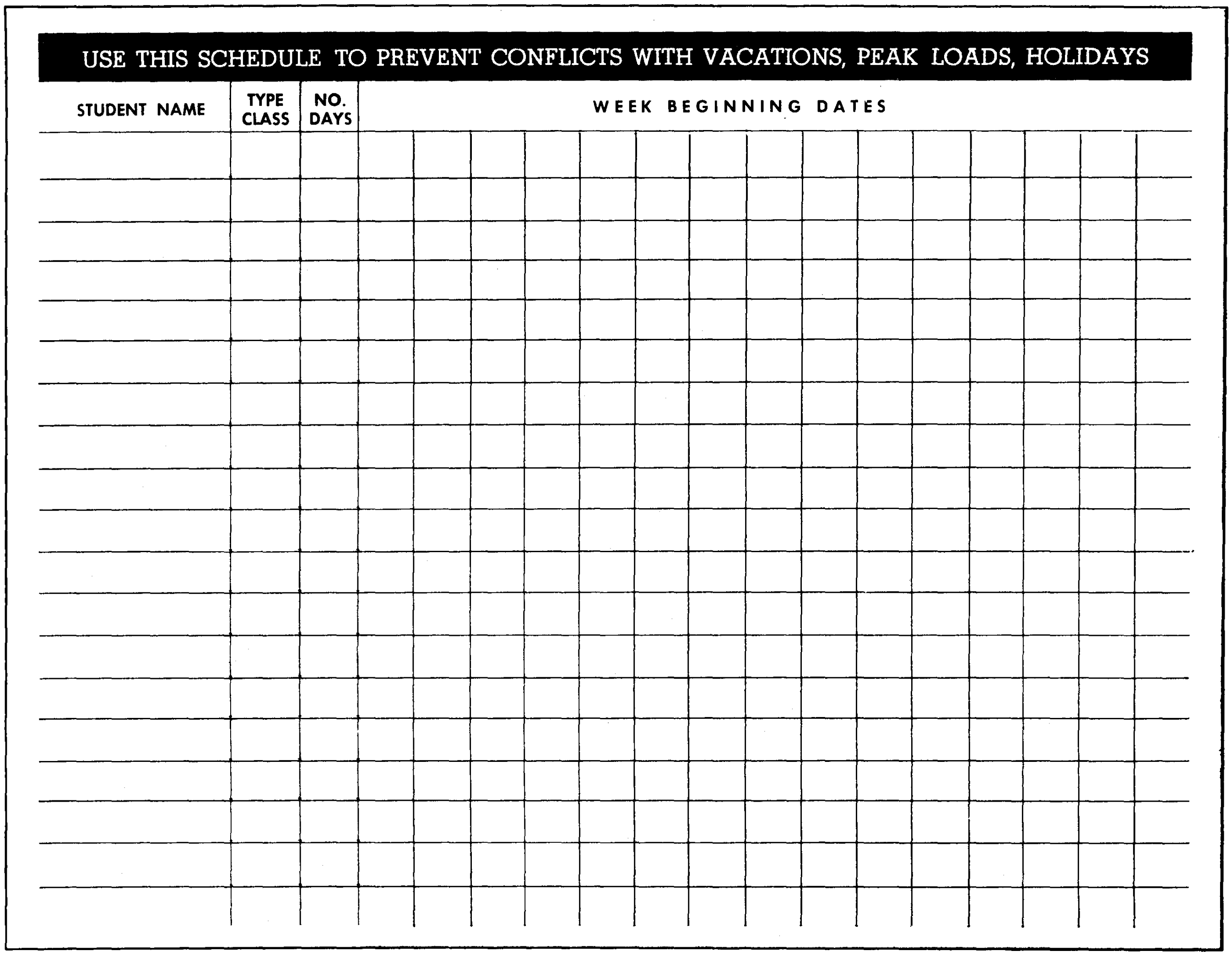
IBM SYSTEM 1440 Page 3 of 2 INSTALLATION PLANNING SCHEDULE BY R. S. Date Oct. 1, 1963 FORM X20-8010 PRINTED IN U.S.A.

11 Point Program

18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5

Educate Other Depts Personnel	
Parallel Operation	11
Method Determined	
Proced & Controls Written	
Cutover	

Figure 59. General Installation Planning Schedule



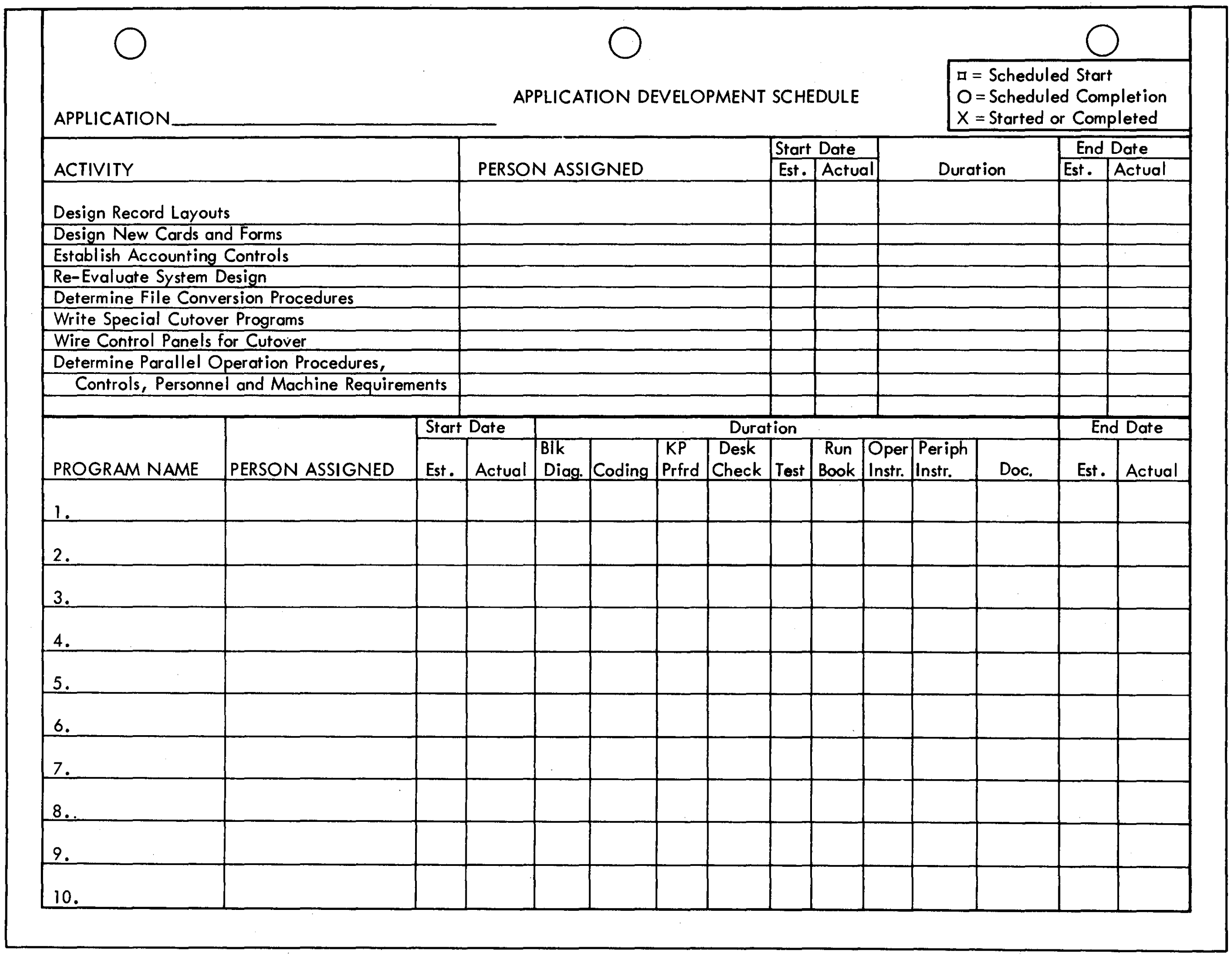
USE THIS SCHEDULE TO PREVENT CONFLICTS WITH VACATIONS, PEAK LOADS, HOLIDAYS

STUDENT NAME	TYPE CLASS	NO. DAYS	WEEK BEGINNING DATES														

Figure 60. Education Schedule

Education Schedules

Setting up an education schedule requires a knowledge of the available classes, their starting dates and prerequisites, any possible schedule conflicts with peak load variations, etc., as well as a knowledge of the individuals' training needs. The IBM training program brochure (R25-1629) is an excellent aid in these considerations. The first page of the brochure lists the class descriptions, the middle pages provide an aid in assigning classes on the basis of the various prerequisites, and the last page (Figure 60) can be used for scheduling the classes to prevent conflicts with vacations, peak loads and holidays.



APPLICATION DEVELOPMENT SCHEDULE

APPLICATION________________________________

⌑ = Scheduled Start
O = Scheduled Completion
X = Started or Completed

ACTIVITY	PERSON ASSIGNED	Start Date		Duration	End Date	
		Est.	Actual		Est.	Actual
Design Record Layouts						
Design New Cards and Forms						
Establish Accounting Controls						
Re-Evaluate System Design						
Determine File Conversion Procedures						
Write Special Cutover Programs						
Wire Control Panels for Cutover						
Determine Parallel Operation Procedures,						
Controls, Personnel and Machine Requirements						

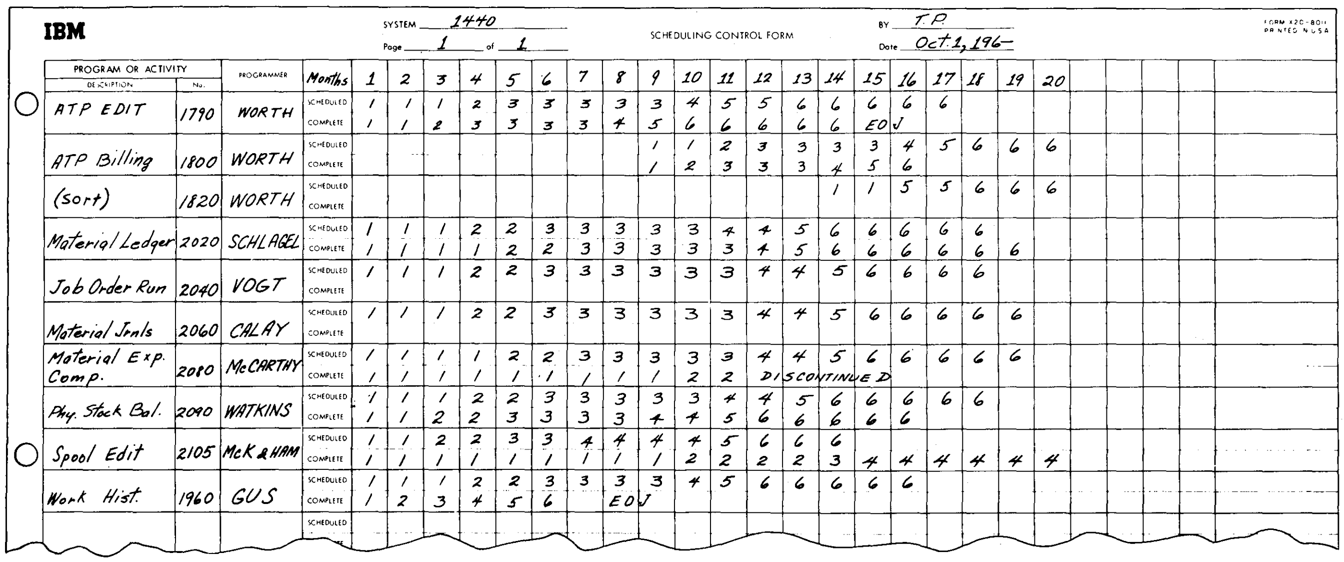
PROGRAM NAME	PERSON ASSIGNED	Start Date		Duration									End Date	
		Est.	Actual	Blk Diag.	Coding	KP Prfrd	Desk Check	Test	Run Book	Oper Instr.	Periph Instr.	Doc.	Est.	Actual
1.														
2.														
3.														
4.														
5.														
6.														
7.														
8.														
9.														
10.														

Figure 61. Application Development Schedule

Application Development Schedules

As soon as the data processing manager or the systems analyst gains sufficient insight into the general system design, he should determine the amount of time and effort required in the areas of systems work, programming, testing and debugging, conversion, offline and computer procedure writeups, etc., to complete the total installation plan. Every application area, and as many programs as possible within each area, should be defined, and the total number of man-days in each of the systems and programming areas should be estimated. This will aid in determining whether there will be enough time among the people concerned to meet the installation date. It can also serve as a basis against which actual progress can be measured to determine whether a proper timetable of events is being kept.

When determining whether there is enough manpower available to meet deadlines, it is important to think of miscellaneous occurrences that could consume the time of the people directly involved with the installation of the computer. Such items as schooling, vacations, holidays, test sessions, meetings and other business responsibilities often account for a surprisingly large portion of available time.



IBM SCHEDULING CONTROL FORM

SYSTEM 1440 Page 1 of 1 BY T.P. Date Oct. 1, 196–

FORM X20-8011 PRINTED IN U.S.A.

PROGRAM OR ACTIVITY		PROGRAMMER	Months	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
DESCRIPTION	No.																						
ATP EDIT	1790	WORTH	SCHEDULED	1	1	1	2	3	3	3	3	3	4	5	5	6	6	6	6	6			
			COMPLETE	1	1	2	3	3	3	3	4	5	6	6	6	6	6	EOJ					
ATP Billing	1800	WORTH	SCHEDULED									1	1	2	3	3	3	3	4	5	6	6	6
			COMPLETE									1	2	3	3	3	4	5	6				
(sort)	1820	WORTH	SCHEDULED														1	1	5	5	6	6	6
			COMPLETE																				
Material Ledger	2020	SCHLAGEL	SCHEDULED	1	1	1	2	2	3	3	3	3	3	4	4	5	6	6	6	6	6		
			COMPLETE	1	1	1	1	2	2	3	3	3	3	3	4	5	6	6	6	6	6	6	
Job Order Run	2040	VOGT	SCHEDULED	1	1	1	2	2	3	3	3	3	3	3	4	4	5	6	6	6	6		
			COMPLETE																				
Material Jrnls	2060	CALAY	SCHEDULED	1	1	1	2	2	3	3	3	3	3	3	4	4	5	6	6	6	6	6	
			COMPLETE																				
Material Exp. Comp.	2080	McCARTHY	SCHEDULED	1	1	1	1	2	2	3	3	3	3	3	4	4	5	6	6	6	6	6	
			COMPLETE	1	1	1	1	1	1	1	1	1	2	2	DISCONTINUED								
Phy. Stock Bal.	2090	WATKINS	SCHEDULED	1	1	1	2	2	3	3	3	3	3	4	4	5	6	6	6	6	6		
			COMPLETE	1	1	2	2	3	3	3	3	4	4	5	6	6	6	6	6				
Spool Edit	2105	McK & HAM	SCHEDULED	1	1	2	2	3	3	4	4	4	4	5	6	6	6						
			COMPLETE	1	1	1	1	1	1	1	1	1	2	2	2	2	3	4	4	4	4	4	4
Work Hist.	1960	GUS	SCHEDULED	1	1	1	2	2	3	3	3	3	4	5	6	6	6	6	6				
			COMPLETE	1	2	3	4	5	6		EOJ												
			SCHEDULED																				

Figure 62. Illustrative Programming Progress Chart

Program Progress Charts

This type of chart (Figure 62) is helpful in controlling and showing an overall picture of programming progress. The programs are listed vertically on the left side of the chart, grouped by application. Across the top of the chart, calendar periods are shown starting with a date some time prior to the delivery date and ranging up to or after the delivery date. The length of the periods and the total amount of time involved will vary depending on the nature of the programs involved.

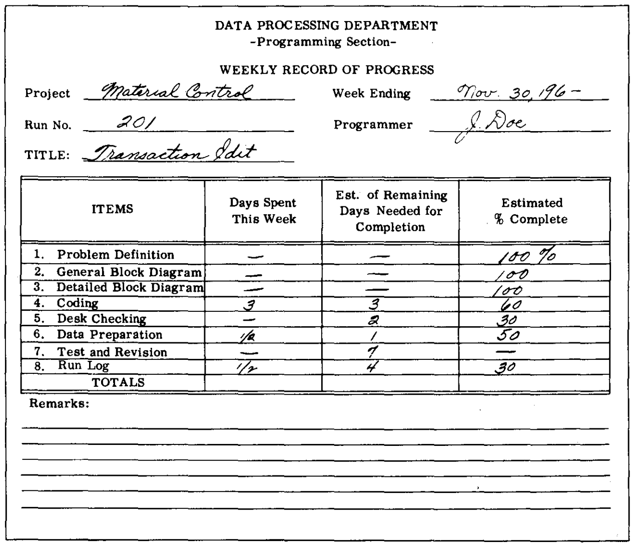
For each program two horizontal rows are provided, the top one representing planned dates and the bottom one indicating dates of actual completion. In this way a comparison between the plan and its implementation is always available. A number code or a horizontal bar consisting of several colors may be used to represent the different phases of program development.

As programming proceeds, planned dates will sometimes turn out to have been unrealistic and the estimates will have to be revised. Whenever such revisions are made, a new chart should be made up. The latest chart will then reflect the new expected completion dates. The old ones should be retained for purposes of historical record.

The chart provides a compact, overall view of planning and progress. It is well, however, to maintain in addition more detailed records of progress. These serve two purposes: they facilitate frequent evaluations of progress and they make up a detailed record of experience which is valuable in future planning.

A convenient form for recording progress on a program is illustrated in Figure 63. A form such as this may be submitted each week by each person working on the program.

The information contained in these records may be posted to a record-of-progress chart (see Figure 64), which spells out information contained in both the program development chart and the weekly progress record. The programs of an application are listed down the left side of the form and the activities associated with each are listed horizontally across the top. The percentage of completion for each activity is shown. In addition, the number of days each activity was expected to take, as well as the number of days actually invested in it, is given.



DATA PROCESSING DEPARTMENT
-Programming Section-

WEEKLY RECORD OF PROGRESS

Project Material Control Week Ending Nov. 30, 196–

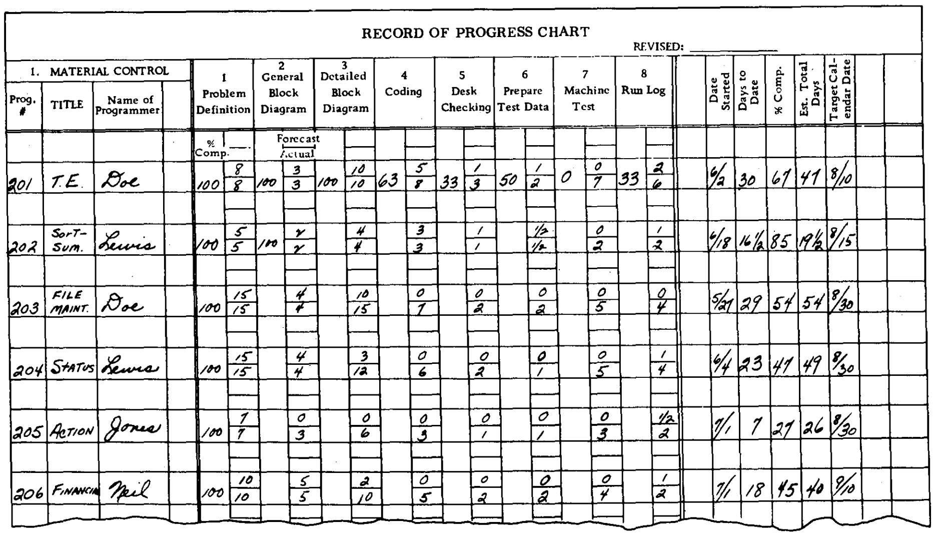
Run No. 201 Programmer J. Doe

TITLE: Transaction Edit

ITEMS	Days Spent This Week	Est. of Remaining Days Needed for Completion	Estimated % Complete
1. Problem Definition	—	—	100 %
2. General Block Diagram	—	—	100
3. Detailed Block Diagram	—	—	100
4. Coding	3	3	60
5. Desk Checking	—	2	30
6. Data Preparation	1/2	1	50
7. Test and Revision	—	7	—
8. Run Log	1/2	4	30
TOTALS			

Remarks:

Figure 63. Weekly Record of Progress



RECORD OF PROGRESS CHART

REVISED: ______

1. MATERIAL CONTROL			1 Problem Definition		2 General Block Diagram		3 Detailed Block Diagram		4 Coding		5 Desk Checking		6 Prepare Test Data		7 Machine Test		8 Run Log		Date Started	Days to Date	% Comp.	Est. Total Days	Target Calendar Date
Prog. #	TITLE	Name of Programmer	% Comp.	Forecast / Actual																			
201	T.E.	Doe	100	8/8	100	3/3	100	10/10	63	5/8	33	1/3	50	1/2	0	0/7	33	2/6	6/2	30	67	41	8/10
202	SORT-SUM.	Lewis	100	5/5	100	7/7		4/4		3/3		1/1		½/½		0/2		1/2	6/18	16½	85	19½	8/15
203	FILE MAINT.	Doe	100	15/15		4/4		10/15		0/7		0/2		0/2		0/5		0/4	5/21	29	54	54	8/30
204	STATUS	Lewis	100	15/15		4/4		3/12		0/6		0/2		0/1		0/5		1/4	6/4	23	47	49	8/30
205	ACTION	Jones	100	7/7		0/3		0/6		0/3		0/1		0/1		0/3		½/2	7/1	7	27	26	8/30
206	FINANCIAL	Neil	100	10/10		5/5		2/10		0/5		0/2		0/2		0/4		1/2	7/1	18	45	40	9/10

Figure 64. Record of Progress Chart

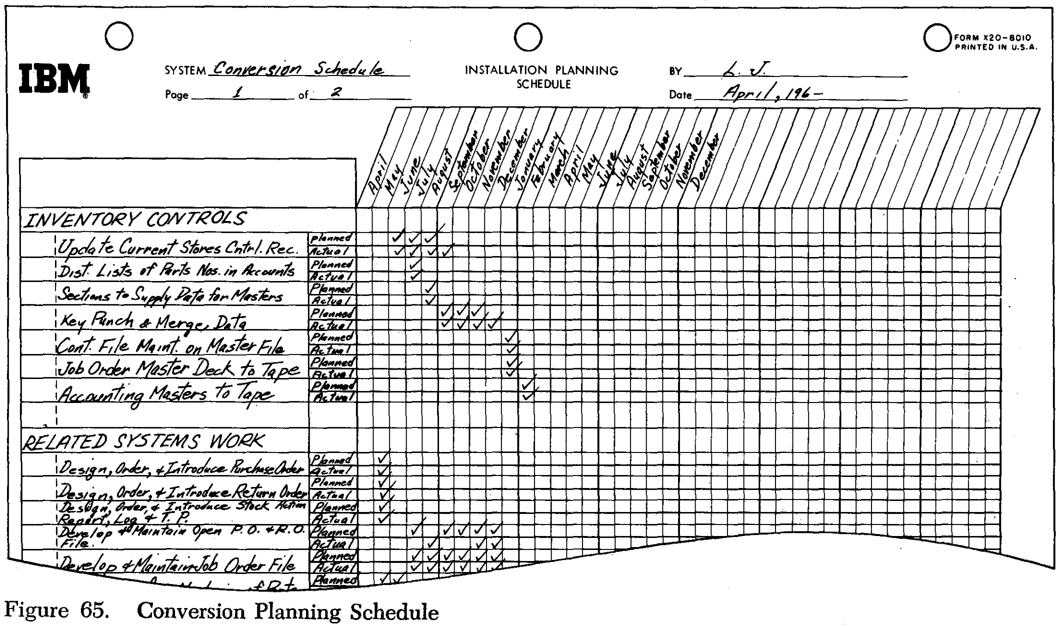
Conversion Planning Charts

A conversion planning chart (Figure 65) should be established spelling out the actions required to accomplish the conversion. Items to be included, by dates of performance, are the conversion of present card formats to the new formats, conversion of present card files to disk files, additional keypunching, a listing of the one-time machine runs to create the master files, etc.

In setting up the schedule, remember that regardless of the planning done, unforeseen circumstances frequently arise to slow the schedule down. If proper allowances have been built into the schedule, these circumstances can be handled without delay to the operation as a whole.

Progress Meetings

The personnel responsible for the installation program should meet periodically to evaluate the progress of the installation effort. These meetings are frequently scheduled on a biweekly basis and run from one to three hourse in duration. They normally include an examination of the performance during the period since the last meeting and the development of plans for the coming period. When any applications or programs are added or deleted, they should be accounted for. If the schedule shows that the effort is falling behind, steps must be made to speed it up. This might require additional manpower, overtime, or possibly outside programming aid.



IBM

SYSTEM Conversion Schedule

Page 1 of 2

INSTALLATION PLANNING SCHEDULE

BY L. J.

Date April, 196-

FORM X20-8010 PRINTED IN U.S.A.

		April	May	June	July	August	September	October	November	December	January	February	March	April	May	June	July	August	September	October	November	December
INVENTORY CONTROLS																						
Update Current Stores Cntrl. Rec.	planned			✓	✓	✓																
	Actual			✓	✓	✓	✓															
Dist. Lists of Parts Nos. in Accounts	Planned				✓																	
	Actual				✓																	
Sections to Supply Data for Masters	Planned					✓																
	Actual					✓																
Key Punch & Merge, Data	Planned						✓	✓	✓													
	Actual						✓	✓	✓	✓												
Cont. File Maint. on Master File	Planned										✓											
	Actual										✓											
Job Order Master Deck to Tape	Planned										✓											
	Actual										✓											
Accounting Masters to Tape	Planned											✓										
	Actual											✓										
RELATED SYSTEMS WORK																						
Design, Order, & Introduce Purchase Order	Planned		✓																			
	Actual		✓																			
Design, Order, & Introduce Return Order	Planned		✓																			
	Actual		✓																			
Design, Order, & Introduce Stock Action Report, Log & T. P.	Planned		✓																			
	Actual		✓																			
Develop & Maintain Open P.O. & R.O. File.	Planned				✓		✓	✓	✓	✓												
	Actual					✓			✓	✓												
Develop & Maintain Job Order File	Planned				✓	✓	✓	✓	✓	✓												
	Actual				✓	✓	✓	✓	✓	✓												
[illegible]	Planned		✓	✓																		

Figure 65. Conversion Planning Schedule

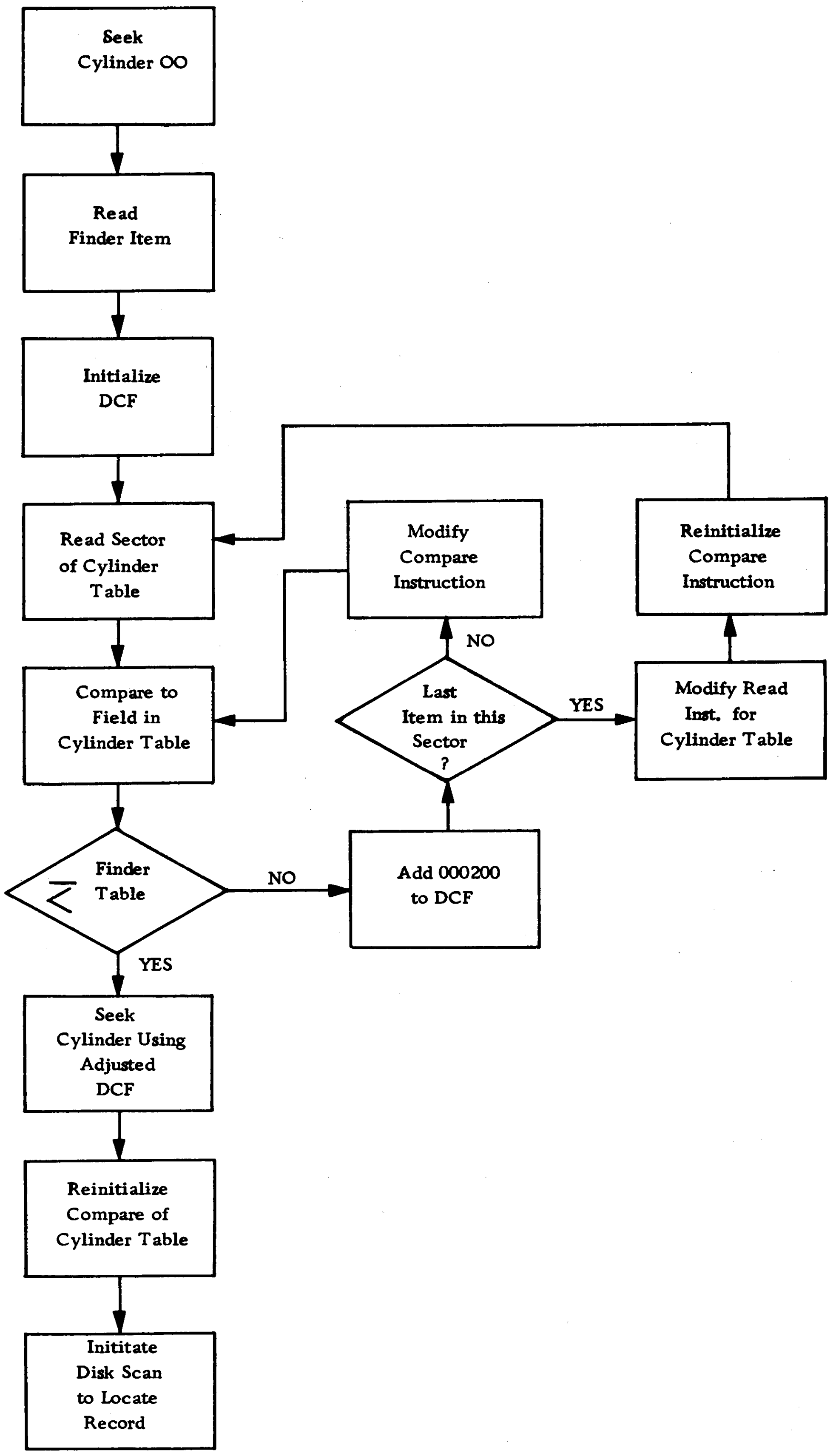
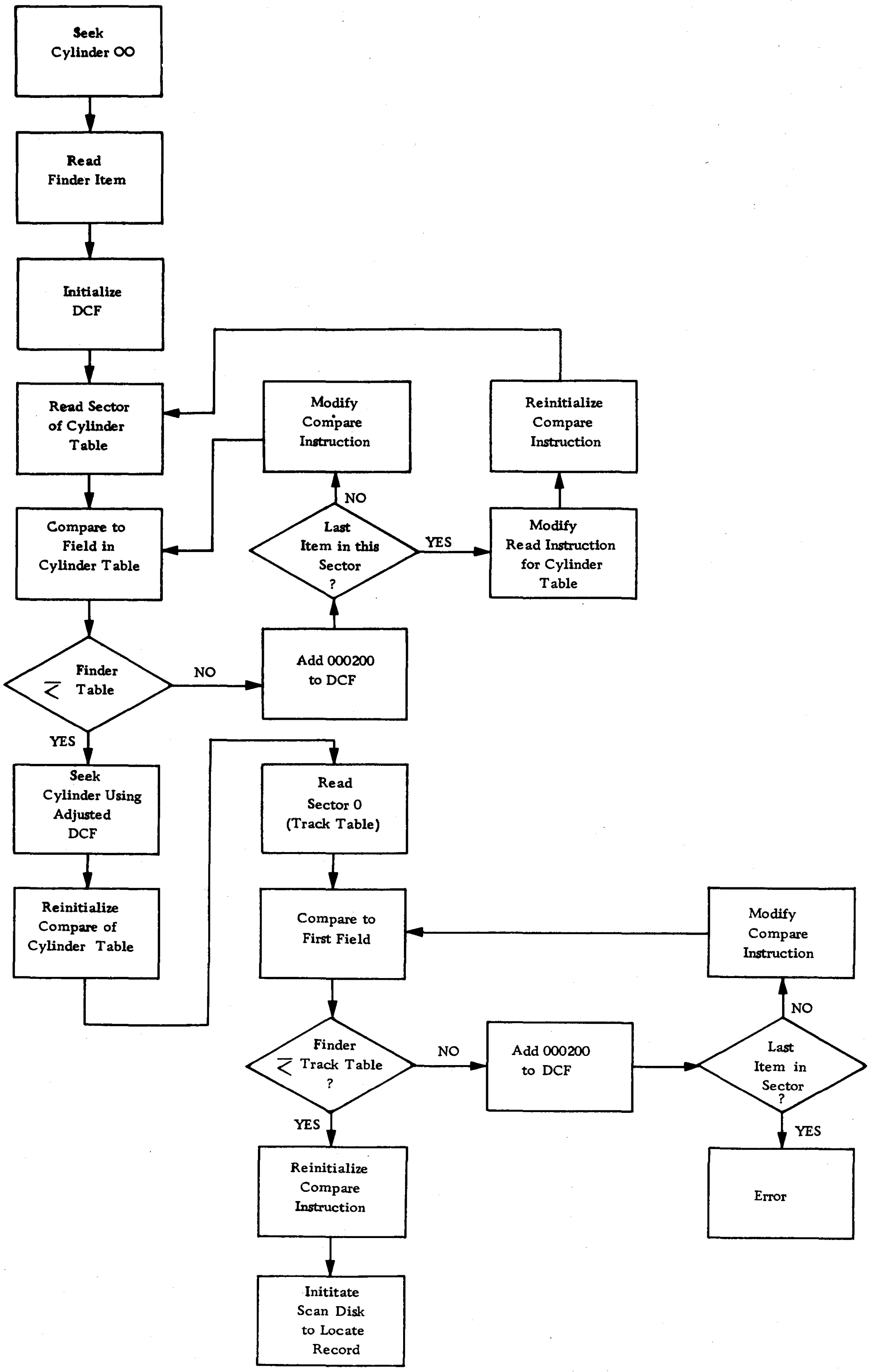
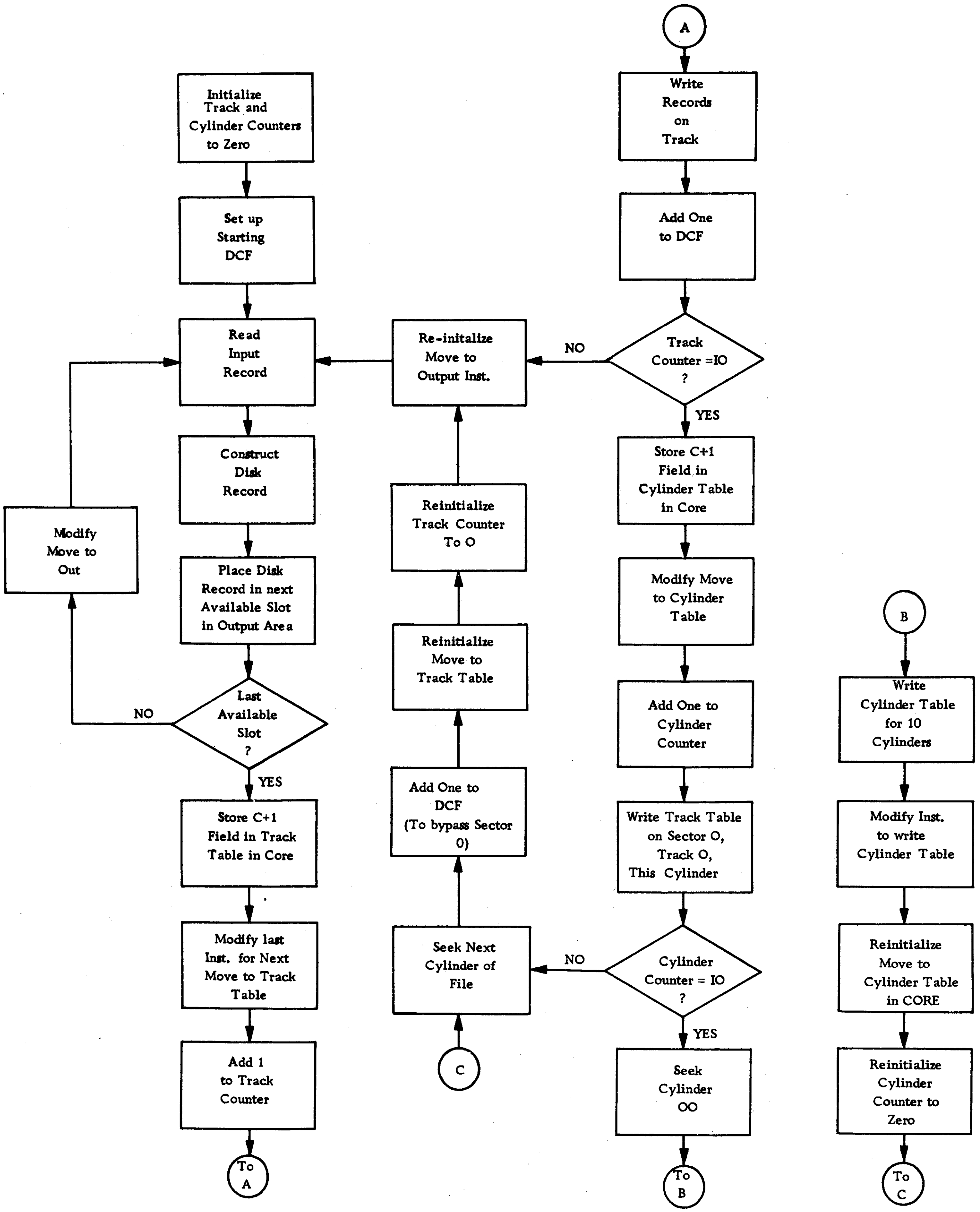

Seek Cylinder 00
Read Finder Item
Initialize DCF
Read Sector of Cylinder Table
Compare to Field in Cylinder Table
Finder ≧ Table
NO
YES
Add 000200 to DCF
Last Item in this Sector ?
NO
YES
Modify Compare Instruction
Modify Read Inst. for Cylinder Table
Reinitialize Compare Instruction
Seek Cylinder Using Adjusted DCF
Reinitialize Compare of Cylinder Table
Inititate Disk Scan to Locate Record

Figure 66. Record Retrieval—Cylinder Table on Cylinder 00 and Scan Disk Feature



Seek Cylinder 00
Read Finder Item
Initialize DCF
Read Sector of Cylinder Table
Modify Compare Instruction
Reinitialize Compare Instruction
Compare to Field in Cylinder Table
NO
Last Item in this Sector ?
YES
Modify Read Instruction for Cylinder Table
Finder ≷ Table
NO
Add 000200 to DCF
YES
Seek Cylinder Using Adjusted DCF
Read Sector 0 (Track Table)
Reinitialize Compare of Cylinder Table
Compare to First Field
Modify Compare Instruction
NO
Finder ≷ Track Table ?
NO
Add 000200 to DCF
Last Item in Sector ?
YES
YES
Reinitialize Compare Instruction
Error
Inititate Scan Disk to Locate Record

Figure 67. Record Retrieval—Cylinder Table on Cylinder 00, Track Table on First Sector of Each Cylinder



A
Initialize Track and Cylinder Counters to Zero
Set up Starting DCF
Read Input Record
Construct Disk Record
Place Disk Record in next Available Slot in Output Area
Last Available Slot ?
NO
YES
Modify Move to Out
Store C+1 Field in Track Table in Core
Modify last Inst. for Next Move to Track Table
Add 1 to Track Counter
To A
Re-initalize Move to Output Inst.
Reinitialize Track Counter To O
Reinitialize Move to Track Table
Add One to DCF (To bypass Sector 0)
Seek Next Cylinder of File
C
Write Records on Track
Add One to DCF
Track Counter =IO ?
NO
YES
Store C+1 Field in Cylinder Table in Core
Modify Move to Cylinder Table
Add One to Cylinder Counter
Write Track Table on Sector O, Track O, This Cylinder
Cylinder Counter = IO ?
NO
YES
Seek Cylinder OO
To B
B
Write Cylinder Table for 10 Cylinders
Modify Inst. to write Cylinder Table
Reinitialize Move to Cylinder Table in CORE
Reinitialize Cylinder Counter to Zero
To C

Figure 68. Illustrative Block Diagram—Creation of Cylinder and Track Tables

F20-0340

Printed in U.S.A. F20-0340

IBM®

International Business Machines Corporation

Data Processing Division

112 East Post Road, White Plains, N. Y. 10601

IBM Technical Newsletter

File Number 1440-00

Re: Form No. F20-0340

This Newsletter No. N20-0008

Date February 7, 1964

Previous Newsletter Nos. None

Changes to "Planning for an IBM 1440 Data Processing System" (F20-0340)

The material in this newsletter should be inserted in *Planning for an IBM 1440 Data Processing System* to incorporate new material on the 1301 Disk Storage and the 7335 Magnetic Tape Unit, which were announced after the initial edition of the manual was printed. The changes to be made are as follows:

Add additional copy to pages 21, 24, 33 and 35 of the original manual.

Add the following pages to the appendix:

	New Page
FILE ORGANIZATION AND FILE TECHNIQUES	96 and 97
These pages cover the same information for the 1301 as previously given for the 1311 on pages 20 and 21 of the initial edition.	
DISK STORAGE TIMING—1301 DISK STORAGE	98 and 99
These pages cover the same information for the 1301 as previously given for the 1311 on pages 46 and 47 of the initial edition.	
SORT TIMING—SORT 5 OBJECT PROGRAM TIMING ESTIMATES	100
This page shows approximate sort time for the Sort 5 object program when used on the 1440.	
IBM 7335 TAPE TIMING	101
This page contains timing formulas for the IBM 7335 Magnetic Tape Unit.	

Changes to be entered in F20-0340

The first two changes below have been made in some of the first-edition copies and may therefore not have to be made in your copy.

On Figure 12, page 21, replace the wording "Track 99" with the wording "Cylinder 99".

Insert the following in the right-hand column of page 24:

The disk storage statement takes the following form:

Label 6–15	Operation 16–20	21 25 30 35
	OP	ADDR

Insert the following at the bottom of page 33:

Disk Storage/Tape Systems.—Disk file reorganization can ordinarily be performed very quickly on systems containing both disk and tape storage. Where two or more 1311 Disk Storage Drives are installed, the disk files can be reorganized either as described above under *Multi-File Systems* or as described in the paragraph below.

On disk storage/tape systems containing one or more disk storage units, the file to be reorganized is read from disk storage and written on tape. Additions or deletions can be incorporated while the records are written on tape or on a subsequent processing run. The reorganized records are then written back into disk storage.

IBM Corporation, Technical Publications Dept., 112 E. Post Road, White Plains, N.Y. 10601

Insert the following on the bottom of the right-hand column on page 35:

Disk Storage/Tape Systems.—Since the information in disk storage can quickly be entered on magnetic tape, these systems permit rapid reconstruction procedures. Daily, or periodically, the updated disk file is written on tape. If it ever becomes necessary to reconstruct the disk file, the information on tape is first written back into disk storage. The disk file is then updated by the transactions (if any) that have occurred since the tape was created. Next the updated disk file is used to create a new magnetic tape; the disk file is then available for subsequent updating.

Staple Here

Staple Here

Staple this page to page 96 of "Planning for an IBM 1440 Data Processing System"(F20-0340).

Staple Here

Staple Here

File Organization and File Techniques

File Concepts

File organization (the way the various records are stored in disk storage) and the manner in which disk records are retrieved for processing greatly affect systems throughput and programming effort. Choosing the best file organization and retrieval technique for a particular application is thus a key systems design function and requires consideration of many factors. Among these are:

- The number of files in disk storage and the size and number of records in each file.
- The number of additions and deletions to the disk files.
- The ratio of the number of transactions processed to the number of records in the master files (the activity ratio).
- The size of the control fields of the transactions and the disk records.
- The core storage available for the record retrieval portion of the processing program.
- The functions of the IBM disk file organization and input/output control system programming packages.
- The number and type of disk storage units and input/output units on the system.

Discussion of the major file organization and retrieval techniques will be preceded by a brief section on the physical organization and operation of the 1301 Disk Storage unit and the type of information that can be stored in disk storage; readers familiar with this material may prefer to skip to "Basic Methods on File Organization and Processing", page 25.

IBM 1301 Disk Storage (Models 11, 12, 21, 22)

As many as five 1301 Disk Storage modules can be attached to the 1440 system. Each module has a capacity of 20,000,000 alphameric characters, divided into 200,000 individually addressable 100-character sectors.* Both the 1301 and the 1311 disk storage units can be simultaneously connected to the 1440 system, so that the maximum online disk storage is 110,000,000 characters* (100,000,000 characters of 1301 storage; 10,000,000 of 1311 storage).

Each 1301 module consists of 21 magnetic disks for recording data. Each surface of the 21 disks, except the upper surface of the top disk and the lower surface of the bottom disk, is used as a recording surface—a total of 40 recording surfaces per module.

A comb-type access assembly with 20 access arms, each containing two read-write heads, is mounted vertically on each module. The access assembly is used to position the read-write heads relative to the disk surfaces (Figure 70). Each of the access arms, with its two read-write heads, is located between the bottom surface of a disk and the top surface of the next lower disk, and is capable of reading or writing data on both of these surfaces.

* In the sector mode. In track record mode (special feature) the track is not divided into sectors, and the maximum capacity of each module is 25,430,000 alphameric characters; maximum online disk storage (1301 and 1311 together) is 142,050,000 alphameric characters.

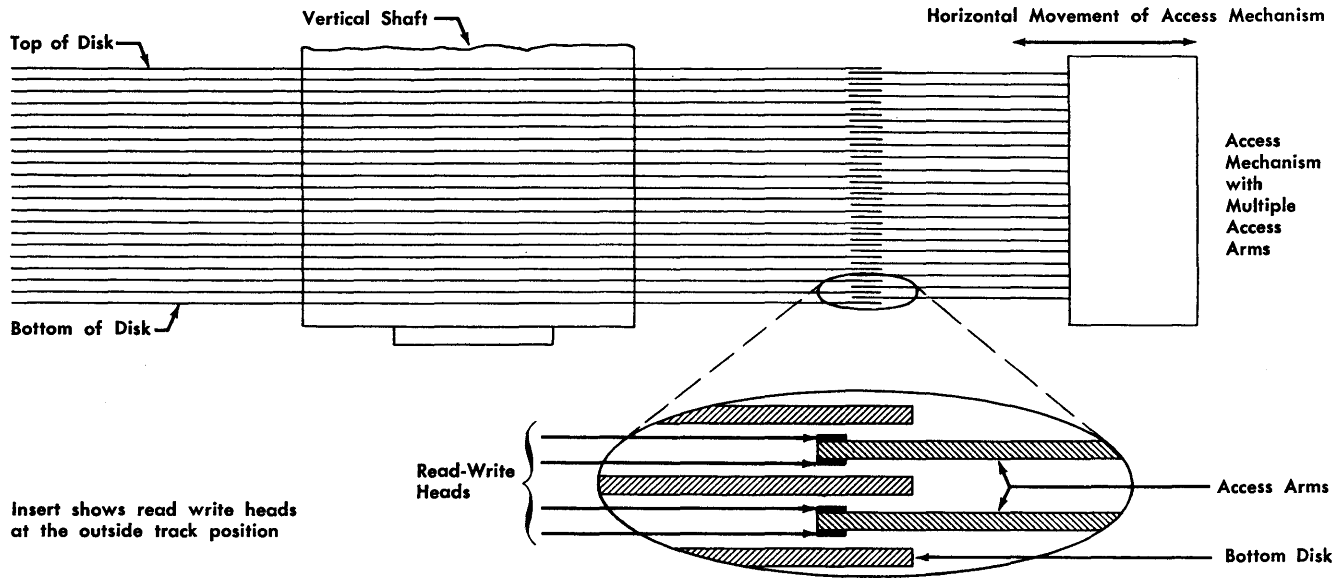

Vertical Shaft
Top of Disk
Horizontal Movement of Access Mechanism
Access Mechanism with Multiple Access Arms
Bottom of Disk
Read-Write Heads
Access Arms
Bottom Disk
Insert shows read write heads at the outside track position

Figure 70. Disk Array and Access Mechanism for One Module of 1301 Disk Storage

As the disks revolve, the read-write heads cover circular paths (called tracks) on the disk surfaces. The 40 tracks that are exposed simultaneously to the 40 read-write heads are referred to as a cylinder. A cylinder, in this sense, is a quantitative concept describing the data that is available at one setting of the access mechanism. One 1301 module may be thought of as 250 cylinders of data, each cylinder consisting of a particular track on each of the 40 disk surfaces. The 40 read-write heads can be positioned at any one of the 250 cylinders.

Each of the 250 tracks on a disk surface is divided into 20 equal-size parts known as sectors. The sectors contain a preassigned six-digit address and provide storage for either 100 characters of data without wordmarks or 90 characters of data with wordmarks (Figure 71). All sectors are addressable, with each module providing 200,000 sectors that may be accessed for reading or writing of data. This figure can be arrived at in the following way:

20 sectors per track	× 40 tracks per cylinder	= 800 sectors per cylinder
800 sectors per cylinder	× 250 cylinders	= 200,000 sectors per module

The sector addresses run sequentially within track, cylinder and module starting from the outermost cylinder in a module, as indicated in the table below:

Sector Address	Cylinder	Track	Sector	Module
000000	First	First	First	First
000019	First	First	Last	First
000799	First	Last	Last	First
199999	Last	Last	Last	First
200000	First	First	First	Second

Perhaps the cylinder concept can best be understood by visualizing 250 cylinders arranged concentrically, with each of the cylinders successively smaller in diameter to permit placement of one inside the other, as shown in Figure 72. Imagine the cylinders as turning and standing on end. Around the circumference of each cylinder are 40 tracks for the magnetic recording of data. To gain access to the data on any particular cylinder, the access arms must be first directed by the program to move "through" the cylinders until the desired cylinder is located. All data recorded on the 40 tracks in the cylinder is then available at the one setting of the access mechanism (a total of 80,000 alphameric characters).

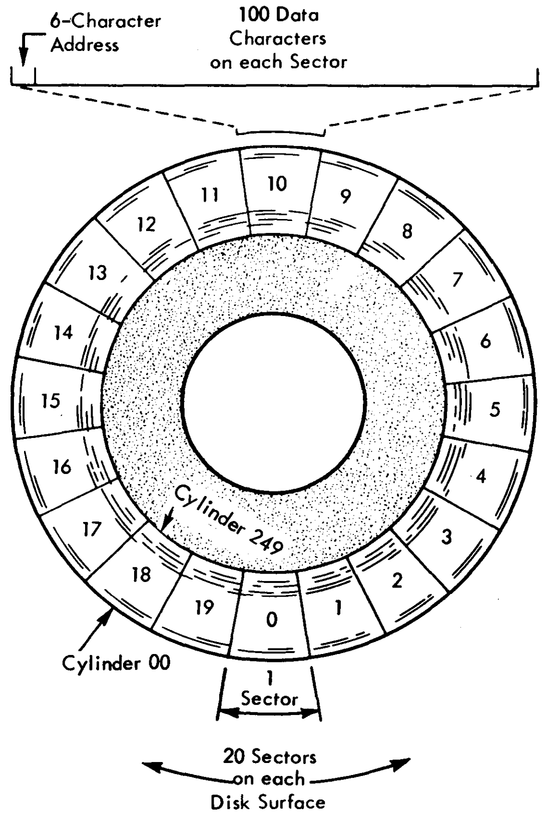


Figure 71. Arrangement of Data on Disk Surface (in Sector Mode)

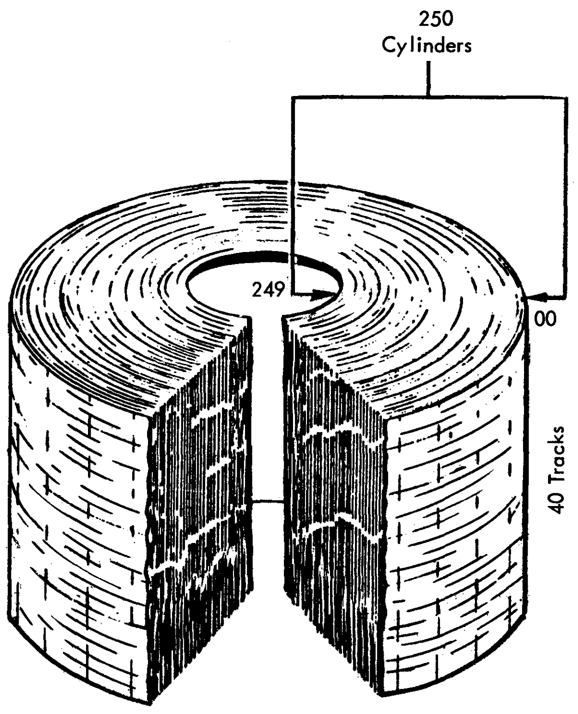


Figure 72. Visualization of Cylinder Concept

Disk Storage Timing (IBM 1301 Disk Storage)

The following factors are involved in timing disk operations:

Seek time – time required for the movement of the access mechanism from one cylinder to another.

Rotational delay – the time required for the desired record to rotate to the read/write head after the previous function has been completed.

Read, write and write-checking time—the time required for the reading, writing and write-checking of the disk record.

Head select time – the time required for the head circuitry to prepare for transferring data bits. A head select time of 1.66 ms is involved whenever a disk read, write or write-check instruction is called for.

Process time – the time required to process the record.

Scan disk time – the time required to scan a portion of a file when the optional Scan Disk feature is used to locate a disk record.

These factors will be clarified below and are also discussed in the file organization section of the manual.

SEEKING DISK STORAGE RECORDS (ACCESS TIME)

The time required for the access mechanism to move from one cylinder to another depends on how far the mechanism moves, within certain machine-defined limits. For purposes of calculating access time, the 250 cylinders in a module are divided into five areas of 50 cylinders with each area subdivided into six sections as shown below.

Access motion time from a record on one cylinder to a record on another cylinder is 50 ms within a section, 120 ms from section to section within an area, and 180 ms from one area to another. For example, an access movement from cylinder 0 to cylinder 9 (within a section) requires 50 ms; from cylinder 0 to cylinder 10 (section to section in one area) requires 120 ms; from cylinder 0 to cylinder 50 (area to area) requires 180 ms.

Figure 73 can be used to determine seek time from one cylinder to another, or from one sector address to another. The access time in milliseconds is indicated by the point of intersection of two lines on a coded area of the figure, one line drawn horizontally from a FROM sector address or cylinder and the other drawn vertically from a TO sector address or cylinder.

For example, to move the access mechanism from track 000000 to 039999 requires 120 ms of access motion time. To move the access mechanism from track 039999 to 040000 requires 180 ms.

After a seek instruction is issued, processing can continue until another disk storage instruction using the same disk drive or another input/output instruction is given.

SEEK OVERLAP

Seek overlap is a standard feature on the 1301, making it possible to overlap seeking on the different 1301 modules installed on a system. It also allows a disk read or write operation on one disk drive to be overlapped with seek operations on other disk drives. When both the 1301 and 1311 are installed on one system, the 1301 seek instruction is normally given first to allow overlap with 1311 file operations without requiring the optional Seek Overlap feature on the 1311.

ROTATIONAL DELAY AND READING AND WRITING TIME

After the access mechanism is positioned on the cylinder, the read or write instruction given will select one of the 40 read-write heads (this requires a head select time of 1.66 ms) and begin comparing addresses until the desired record is found. It is unlikely that the record will be under the head immediately; a short delay is normally necessary while the head waits for the record to rotate to it. If the address is just coming under the head, the wait time is 0. If the first character of the address just passed under the head, the wait time is 33.3 ms. This time is referred to as rotational delay and averages 16.7 ms, the equivalent of half a revolution of the disk. After a record is found, reading or writing or write-checking can take place at 1.66 ms per sector. After a write operation, a rotational delay of up to 31.7 ms

Area	Cylinders	Cylinders in each section					
		1	2	3	4	5	6
A	0-49	0-9	10-19	20-24	25-34	35-44	45-49
B	50-99	50-59	60-69	70-74	75-84	85-94	95-99
C	100-149	100-109	110-119	120-124	125-134	135-144	145-149
D	150-199	150-159	160-169	170-174	175-184	185-194	195-199
E	200-249	200-209	210-219	220-224	225-234	235-244	245-249

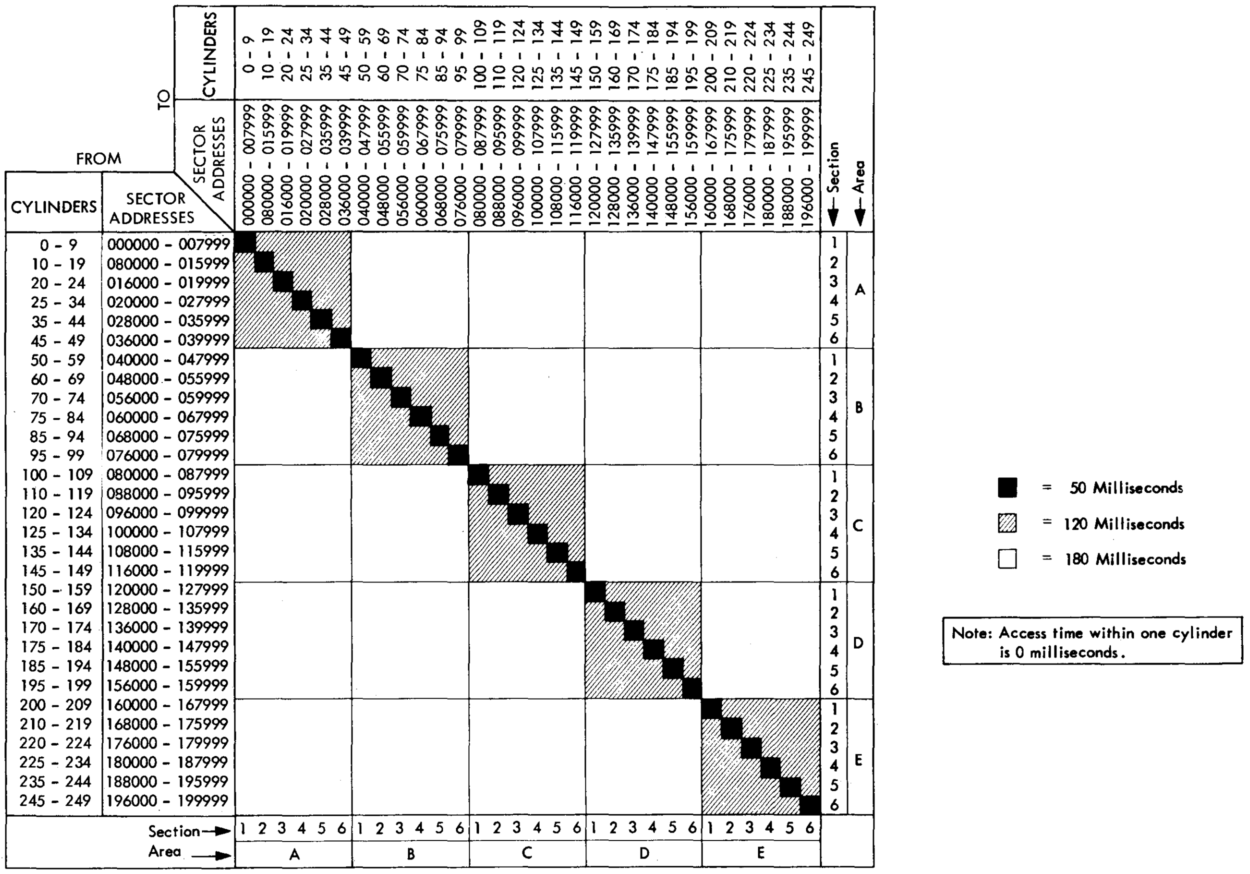


Figure 73. Seek time—1301 Models 11, 12, 21, 22

(depending on number of sectors read) is required until the record is physically available for the write check. The rotational time can be used for processing.

For an example of timing for rotational delay and disk reading and writing, see Figure 74 showing the reading, updating and writing of a one-sector record (100 characters). If possible, processing should be kept within the available rotation time; otherwise the total cycle time will be increased by increments of 33.3 ms rotation time. Rotation time between the write and write-check operations can be used for such functions as updating control totals and arranging fields for printing.

SCAN DISK TIME

Through the use of the optional Scan Disk feature it is possible to search an entire cylinder in 1,350.3 ms (18.3 ms for rotational delay and head select time + 1,332 ms for scanning the 800 sectors). After the desired record is located, a read instruction is given to read it into core. If less than an entire cylinder is searched the time is reduced proportionally (for example, a 100-sector scan would require 184.9 ms).

In Figure 74, note that 1.66 ms of the disk rotation time is used to read one sector. This leaves 31.66 ms of rotation time before the record can again be accessed, 1.66 ms of which will be needed for head select time. The remaining 30 ms is available process time.

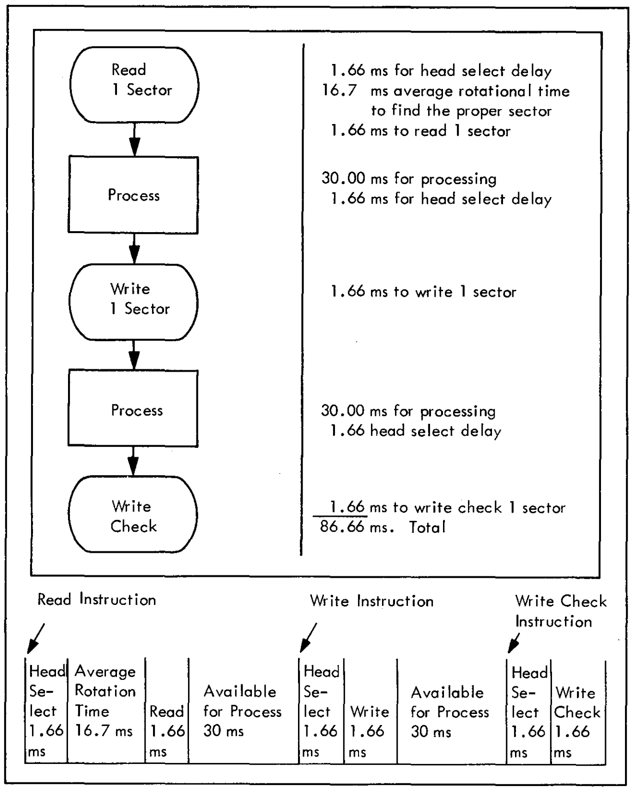


Figure 74. Disk Storage Timing for a One-Sector Record

Sort 5 Object Program Timing Estimates

Figures 75 and 76 show approximate sort times for the Sort 5 object program when used on IBM 1440 Data Processing Systems equipped with 1311 Disk Storage and 4,000, 8,000, 12,000 or 16,000 positions of core storage. These times should be used for planning purposes only.

Figure 75 shows the time required when the sort run is to terminate at the end of phase 4; Figure 76 shows the time required when the run is to terminate at the end of phase 6. Times are shown for object program runs when the direct seek (DS) feature is used and for runs when a normal seek (NS) operation is used.

The times are based on the following assumptions: disk input and output without any additional options; optimum input/output blocking; and the work area, input area, output area, and work cylinder optimally placed on the disk pack(s). The times do not include the time required to load the program.

These times are the result of object-program runs with control field data (CF) of ten characters. With an increase of control field data, there is a corresponding increase in processing time.

Machine Size	Record Length	Input File Size	Estimated Times	
			DS	NS
4k	20	2,000	1.50	1.70
	80	5,000	7.20	8.10
	100	5,000	7.77	7.78
	200	10,000	16.30	18.00
		25,000	46.90	*
8k	20	25,000	27.45	30.97
	40	5,000	3.10	3.30
	100	2,000	1.40	1.48
	500	10,000	11.50	12.70
12k	40	25,000	20.50	22.40
	100	10,000	8.40	8.90
	500	5,000	4.40	4.80
16k	20	25,000	20.20	20.60
	80	10,000	8.08	8.14
	200	5,000	3.60	3.80
	500	10,000	9.40	9.94

* Timing runs not made.

Figure 75. Sort 5 (Phases 1-4) Run Time in Minutes

Machine Size	Record Length	Input File Size	Estimated Times	
			DS	NS
4k	20	2,000	3.94	4.55
		25,000	84.29	134.47
	40	5,000	15.75	21.69
		10,000	35.64	52.15
	80	2,000	7.02	8.99
		5,000	20.80	26.93
	100	5,000	20.26	29.73
	200	10,000	54.30	73.14
		25,000	164.70	*
8k	20	25,000	55.29	68.74
		50,000	113.34	179.45
	40	2,000	3.47	3.86
		5,000	9.92	11.52
	80	10,000	25.69	41.52
	100	2,000	4.14	5.17
		5,000	12.42	18.87
	500	2,000	9.22	11.95
		10,000	67.57	79.15
12k	20	2,000	3.44	3.65
		50,000	102.14	122.92
	40	25,000	50.60	76.70
	80	25,000	46.30	68.94
	100	10,000	26.14	*
	200	2,000	5.29	9.10
		10,000	35.07	55.14
	500	5,000	18.24	*
16k	20	25,000	47.62	54.52
		50,000	99.77	113.89
	40	2,000	3.66	3.69
		5,000	8.85	9.52
	80	5,000	9.55	11.40
		10,000	19.93	*
		25,000	61.49	*
	100	2,000	4.02	4.35
	200	5,000	12.10	*
	500	10,000	45.36	60.74

* Timing runs not made.

Figure 76. Sort 5 (Phase 1-6) Run Time in Minutes

Tape Timing—IBM 7335 Magnetic Tape Unit

The 7335 Magnetic Tape Unit, Model 1 (one tape drive with control unit) or Model 2 (two tape drives, packaged together, with control unit) provides the 1440 with tape processing capability. A maximum of two tape drives (7335 Model 2) can be attached to a single 1440 system. The units are governed by the tape control unit, which permits one tape unit to operate at a time. If one tape drive is busy, the other drive cannot be used until all operations on the busy one have been completed.

TAPE OPERATIONS

The 7335 records data on tape in BCD or binary form. During reading or writing, tape is moved from the file tape reel through a horizontal vacuum column to the machine reel. Tape speed is 36 inches a second. Tape may be backspaced over a record or may be rewound to the beginning of the reel.

While tape is moving backward (machine reel to file reel), no reading or writing takes place. There are two speeds of rewind: high and low. High-speed rewind is approximately 2.2 minutes per 2,400 feet of tape per reel. Low-speed rewind is 36 inches per second. Both rewinds will position tape to reflective spots. After a high-speed rewind, tape must be manually loaded into the vacuum columns for further reading or writing.

TIMING FORMULAS

The factors used in timing the 7335 Magnetic Tape Unit are given below:

$C = .050$ ms, the character rate of the 7335.

N is the number of characters in the record block.

Start time is the time necessary for the tape unit to accelerate to operating speed.

Stop time is the time necessary for the tape unit to decelerate and stop.

Record-check time is the time it takes to read or write the check character. This time is based on the read-write head gap (the distance that separates the read and write heads) and the time it takes a single character written on tape to travel from the write head to the read head.

Load point time is the time required to pass the tape from the load point to first record. When reading or writing from the load point, start time is increased by about 27 ms.

The read and write operation timing formulas are as follows:

Read Operation Timing.—During a 7335 tape-read operation, the tape control unit is interlocked $20.5 + .050N$ ms (Figure 77). This includes:

10.3 ms	– start time
9.8 ms	– stop time
.4 ms	– record-check time
.050N ms	– record time

During the same read operation, the processing unit is interlocked for $10.4 + .050N$ ms (Figure 77). This includes:

10.3 ms	– start time
.1 ms	– part of .4 ms record-check time
.050N ms	– record time

Therefore, in a tape-read operation, processing can take place during 10.1 ms of stop time and record-check time. A tape transmission error condition can be recognized .3 ms after the processing interlock is released.

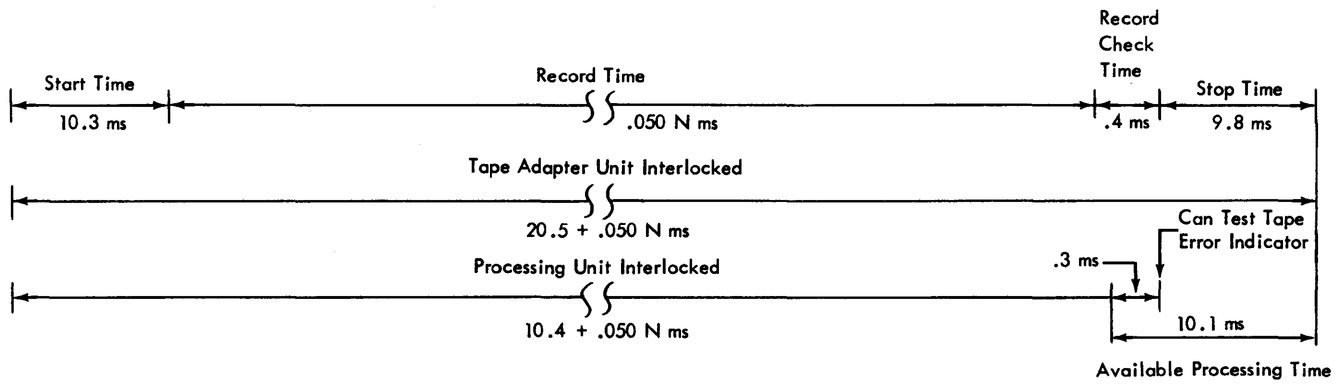


Figure 77. Read Operation Timing

Write Operation Timing.—During a 7335 tape-write operation, the tape control unit is interlocked 20.3 + .050N ms (Figure 78). This includes:

7.2 ms — start time
4.4 ms — stop time
8.7 ms — record-check time
.050N ms — record time

During the same write operation, the processing unit is interlocked for 7.2 + .050N ms (Figure 78). This includes:

7.2 ms — start time
.050N ms — record time

Therefore, in a tape-write operation, processing can take place during the 13.1 ms record check and stop time. A tape transmission error condition can be recognized 8.7 ms after the processing interlocked is released.

If the tape transmission error test is given during the 8.7-ms record-check time, the processing unit is interlocked until the error indicator is interrogated. The difference between the reading record-check time of .4 ms and the writing record-check time of 8.7 ms is due to the read-write head gap time (8.3 ms).

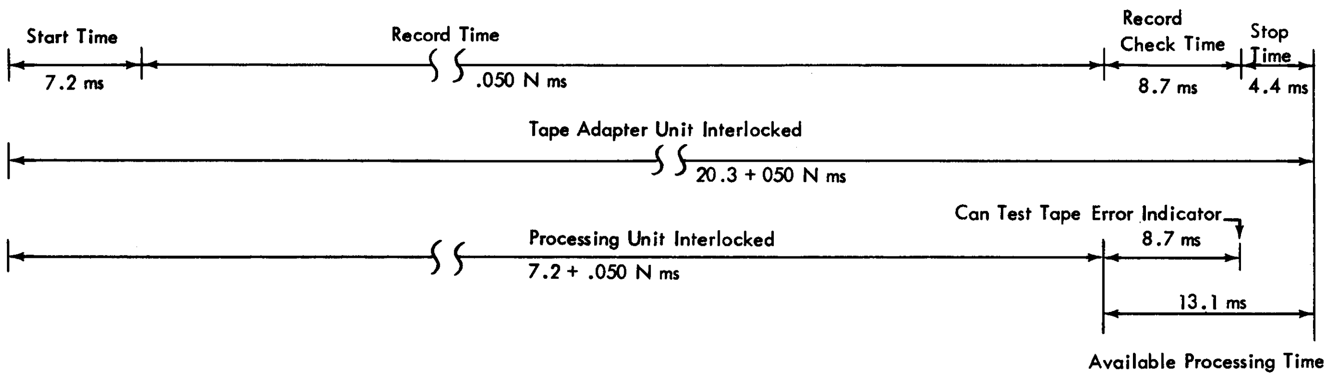


Figure 78. Write Operation Timing

F20-0340 (N20-0008)

Printed in U.S.A.

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601