APPLICATIONS SOFTWARE DEVELOPMENT STANDARDS MANUAL

FEBRUARY 1977

BFC 5025

basic / four corporation
an MAI company
18552 MacArthur Boulevard, Irvine, California 92714

All specifications subject to change without notice.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

SECTION I – INTRODUCTION

PURPOSE

The Applications Software Development Standards Manual presents generally accepted methods and procedures for system design, programming and documentation for BASIC/FOUR system applications. Consistency is the key to quality workmanship in the end product. This manual is also intended for use as an educational tool for personnel who are unfamiliar with BASIC/FOUR system applications software standards. The standards set forth in this manual are used to develop all BFC applications software packages.

SOURCE

The knowledge and experience necessary to develop software standards can only be acquired through involvement in software development and field experience with BASIC/FOUR systems. The methods and procedures described in this manual have been obtained through the contributions of a variety of analysts and programmers with this background.

REVISIONS

Since the purpose of this manual is to present generally accepted practices, it will be improved by suggestions and additional contributions from the field.

CONTENTS

The use of the standards presented herein provides for the adaptation to BASIC/FOUR equipment capabilities and the Business BASIC language, and also presents terminology and application software definitions and requirements.

The four subjects of concentration are:

1. BASIC/FOUR Fundamentals
2. Systems Design
3. Programming
4. User Documentation

The material covering each subject is separated into logical sections to highlight the requirements of successful applications software development.

SECTION II – FUNDAMENTALS

SYSTEM CHARACTERISTICS

The BASIC/FOUR system is designed to fulfill business processing requirements in the interactive time-sharing environment. The system hardware consists of the CPU and peripheral devices including magnetic disc drives, line printers, video display terminals (VDT's), magnetic tape drives, paper tape punches, paper tape readers, card readers, and executive display terminals (EDT's).

Business BASIC is an expansion of the BASIC (Beginner's All-Purpose Symbolic Instruction Code) programming language. It provides the capability and flexibility necessary to support the BASIC/FOUR system hardware in an interactive business environment. Most of the original problem-solving capabilities of BASIC have been retained in Business BASIC. Also, all peripherals are supported by complete formatting and editing capabilities, as required by business applications. In most applications during execution, a program displays instructions on the VDT screen and the operator simply replies by keying in the requested information.

The Basic Operating System Software (BOSS), for Business BASIC provides the interface between the interactive terminal users, the hardware, and the software components of the BASIC/FOUR system. BOSS monitors the operations of each task and allocates the system resources on a first come, first serve basis. BOSS is composed of three major software elements:

- Business BASIC Interpreter – Interprets the Business BASIC language statements in order to select the required system routines for execution.
- Program Executive – Provides execution scheduling of CPU time, allocates peripheral devices, controls file organization and performs all disc space allocation and de-allocation.
- Configuration – Contains the specifications necessary to support the exact hardware configuration of a BASIC/FOUR system.

Magnetic disc storage is the primary storage medium used on the BASIC/FOUR system. Disc capacity may be expanded from through the use of off-line removable disc packs or additional disc drive(s).

Data file records stored on magnetic disc are organized in variable fixed length format. Once a file record size is defined, the disc space required is reserved. If a record length is less than the defined size, the unused disc space remains available only for that record. A record may not exceed the defined file record length.

The BASIC/FOUR system has the ability to communicate asynchronously over telephone lines. Remotely located VDT's and printers may use the CPU and disc storage located at a home office. Also, by using the synchronous communications capability, the BASIC/FOUR can communicate with another CPU.

Due to the high degree of system interaction between programmers and operating personnel, who are subject to error, the system recognizes errors made while keying in statements. Error codes are issued and immediately displayed on the terminal. Errors generated while executing a program are also detected by the system, however, such errors are usually controlled by the program or instructions are displayed.

Business BASIC is a simple to use, high level language. Although it is semi-documenting, complete system design, program and operator documentation is critical for any system. Recommended documentation forms and suggested formats are available which are designed especially for use with BASIC/FOUR systems.

The absence of forced program formatting in Business BASIC, makes it very powerful as well as flexible. However, since programs function logically, it is sensible that they should be organized logically. The programs which together form a system should have consistent structure. This simplifies debugging during testing and programming changes are easier later when modifications or additions are made. (Refer to Program Organization).

It is very important to become familiar with the terminology used in conjunction with the BASIC/FOUR system. Analysts, programmers, operators and management who communicate accurately, produce the best applications software system. The Glossary Of Terms includes most of the terms and definitions used with the BASIC/FOUR system.

PROCESSING TYPES

ON-LINE PROCESSING — The on-line interactive capabilities of the BASIC/FOUR system provide the ability to immediately update system files. Updating occurs only after each transaction is entered and validated through the VDT screen. The information necessary to provide hard copy audit trails is usually stored in a disc file which is updated by each transaction processed. This is the primary method of processing used on the BASIC/FOUR system.

BATCH PROCESSING — Processing groups of related transactions may be implemented using magnetic tape, disc, or paper tape batch files, prepared off-line. However, batch processing is more efficient on the BASIC/FOUR system when the interactive, on-line capabilities are utilized. This can be done by using the VDT for the entry and validation of transactions. Once validated, the transactions are grouped in a disc file and stored for updating at a later time. In most cases, when this method of processing is employed, batch files are printed, corrected and balanced before being used to update the system files.

FILE TYPES

PROGRAM files are one record files which are used to store programs on the disc.

INDEXED files contain data records which physically exist in record number sequence 0 - n, where n + 1 is the allocated number of records for the file. The records may be directly accessed using the IND= option or accessed in sequence. INDEXED files may be used as master files, report files, or batch files when the file record numbers, 0 - n, are compatible with the required processing sequence. Also, system control files which may contain small amounts of data such as dates, company name, and processing status flags, are often INDEXED files.

DIRECT files contain data records which are ordered by "keys" in a file directory. Although keys may be variable in length, they should be uniform within a file. The DIRECT file is the primary file type used on the BASIC/FOUR system and may be read randomly using the KEY= or IND= options. DIRECT files may be read logically according to the key sequence OR they may be read logically using the IND= option, when no particular record sequence is required.

SORT files contain only keys (no record data) and may be used in conjunction with DIRECT or INDEXED files to produce "sorted" reports. Also, SORT files may sometimes be used as master files, when only the key information is required.

APPLICATION FUNCTIONS

BASIC/FOUR system applications software requires the consistent use of the following function types:

File Maintenance Function — necessary to create and maintain critical data contained in master files.

Entry Function — provide for the entry and validation of transactions, such as: orders, invoices, etc. . .

Sort/Update Function — allow for the manipulation of data for reports and files.

Report Function— provide printed output of file data needed by the user.

Inquiry Function — supply visual display of master file and/or other information required by the user.

Selector Function — afford a common point of beginning and ending processing modules, while providing an index of available selections.

Backup and Restore Function — necessary to reduce error potential during backup and restore operations.

MAINTENANCE

DESCRIPTION

Maintenance functions allow the on-line addition, change, deletion, and inquiry of file records. Maintenance programs should be provided for all application master files.

All relevant record data should be displayed; however, total and accumulaton fields generated by the application should not be accessible for change. If such totals require manual entry at conversion time, a special version of maintenance must be provided. Fields accessible for change should be numbered.

Audit trails are generally not produced by maintenance functions; however, a hard copy option is often provided.

PROGRAMMING CONSIDERATIONS

After displaying the screen format in background, or clearing old data, maintenance functions allow for the selection of the following processing options to be performed on the file: 1—ADD, 2—CHANGE, 3—DELETE, 4—INQUIRE, 5—END.

Upon input, the data field area should be cleared, characters printed to indicate the field length and the bell rung. This may be accomplished efficiently by using a common input routine.

All possible validation should be performed on each field including reading other master files. Always display descriptions or names for codes or numbers entered. Invalid entries trigger a bell alarm and require re-entry of the field.

The file key data is always the first data input after the processing option is entered. Once the key data is entered the following procedure is generally followed.

Addition: The file is read to insure that this is not a duplicate entry.

Each field is entered in sequence. A IV entry causes the program to return to the previous field for re-entry.

Once all data fields are entered the transaction is treated as a change.

Changes: The record is extracted and the existing data is displayed.

The response to a "DATA CORRECT? ENTER CR TO PROCEED LINE NO. TO CHANGE NN TO VOID" message enables the selective change of accessible data fields. CR or Function Key I is a "NO CHANGE" entry for the data fields.

After all changes are entered the record is written to the file.

Deletes: The record is extracted and the existing data is displayed.

If the record has active totals or accumulations, the message "INVALID DELETE" is printed and the program returns for another transaction, (processing option entry).

The response to "CR TO DELETE 1 TO VOID", or other similar message provides the operator with a "last chance" decision option before removing the record from this file.

Inquiry: The file is read and the existing record data is displayed. A hard copy option is often provided.

The response to "CR TO PROCEED" causes the program to return for the next transaction, (processing option or the first key field for entry.)

Reference PROGRAMMING STANDARDS for Function Key usage.

ENTRY

DESCRIPTION

Entry functions allow either on-line or batch processing of transactions.

On-line entry functions facilitate immediate master file update requirements while performing validation on an individual transaction basis. Audit trails may be produced through the use of report files.

Batch processing entry functions provide for the addition, correction, and deletion of transactions using a batch file. This method allows balancing and correction operations to occur prior to the updating of the master file.

PROGRAMMING CONSIDERATIONS

The entire screen format is printed in background (or old data cleared) prior to beginning transaction entry. Upon input the data field area must be cleared and printed, to indicate the field length. This may be efficiently done by using a common input routine.

All possible validation should be performed on each field, including reading other files. Always display relevant descriptions, names or amounts obtained from validation reads for field entries. Invalid entries trigger a bell alarm and require re-entry of the erroneous field.

Each field is entered in sequence. A function IV entry during this phase causes the program to return to the previous field for re-entry. Once all fields have been entered a final "DATA CORRECT?" response should be requested before updating the affected files. If the data is not correct, the operator should be able to access any of the fields for correction as well as void the entire entry.

Reference PROGRAMMING STANDARDS for Function Key usage.

SORT/UPDATE

DESCRIPTION

Sorts and updates are always processed on-line. SORT files are particularly useful in most sorting situations. Updates usually involve master file updating from transaction or batch files. Sorts usually involve the creation of work files from master files.

PROGRAMMING CONSIDERATIONS

Password entries are sometimes required before beginning critical updates. Status flags can be used to indicate update in progress and run completion conditions. This method provides processing control in a multiple VDT environment.

Extract and/or LOCK should be used in updates and sorts and the INDor KEY= function should be used.

The READRECORD/WRITERECORD function can greatly decrease processing time in sorts and updates.

REPORTS

DESCRIPTION

Reports are generally produced on-line from permanent master or report files. Reports also function as part of batch processing when batch listing of "proofs" are required. Special situations arise where sort or work files must be used to generate the desired sequence. Updating of files should not take place while producing a report due to the increased run time and restart considerations.

If a report must be periodically produced, the frequency should appear in the report title. All report titles should be assigned and referenced consistently on all documentation.

Report headings should include company name, application description, report title, run date, period ending date (if any), page, system time, and report options selected (if any). Report headings should be uniform and centered based on the maximum print positions used. Individual field headings should be right and left justified for numeric and alpha fields. Special forms requirements should be noted on all documentation.

The availability of report options reduce duplication of effort in reports and programming, while increasing flexibility. Options for data selection based on date, status, or ranges (starting-ending) should be considered.

All report functions should include restart/reprint capabilities. No file should ever be cleared or processing status flags set until the operator indicates that the run was successfully completed.

PROGRAMMING CONSIDERATIONS

The printing of headings and totals should be in subroutines. Totals may be accumulated in arrays to reduce program coding.

All print statements should have error exits to a common error handling routine.

Report progress statements should have error exits to a standard error handling routine.

Report progress should be displayed by indicating the current key data or record count on the screen. The operator needs to know the current status.

INQUIRY

DESCRIPTION

Inquiry functions operate on-line and usually display master file data not available through maintenance programs. Hard copy options may be provided as required.

PROGRAMMING CONSIDERATIONS

The screen format is displayed in background. The screen foreground is cleared. The key data field is entered. If no corresponding record is found on file, an "INVALID ENTRY" message is usually displayed and the bell alarm rung. Where multi-screen displays are required, the operator should be provided with an end option before proceeding to the next screen.

SELECTORS

DESCRIPTION

Selectors display the selector identification time, date and all application function descriptions. The operator is able to select the application functions by number. Sub-selectors may be used to group function types when there are too many applications to fit on one selector screen.

PROGRAMMING CONSIDERATIONS

Upon completion, all programs should link back to the selector from which they were run. The master selector must have the capability to print and reset the system day.

When "END" is entered on the master selector, a BEGIN is executed and "**SYSTEM CLEAR**" is printed on the screen.

Selector program generators are often included in systems. In many cases, the available application functions for a selector are stored on a file and are available for maintenance.

BACK-UP/RESTORE PROCEDURE

The Back-Up/Restore Procedure should be presented as a processing selection on the system selector and is composed of the standard utility label controlled disc copy programs or other comparable procedure. This provides a controlled back-up environment.

This procedure should function automatically under program control. The operator should be told when and where to mount labelled discs. The system should validate that the correct packs are mounted in the correct disc drives before proceeding. Often, a logging feature can be used to record error conditions and recommend recovery techniques.

SECTION III – SYSTEM DESIGN

DESIGN APPROACH

The BASIC/FOUR system is very powerful and affords the analyst almost limitless design alternatives. In order to develop effective, well documented applications software systems, the analyst must implement systematic procedures which act as a guide during development. It is important to maintain a proper prospective of a project and to plan and control its development.

A system which is well documented and utilizes consistent design and programming techniques, is highly valued. However, a system must also fulfill the user's processing requirements and must be implemented in accordance with an installation schedule. In order to develop high quality applications software and install it in a timely fashion, it is necessary to function in an organized and methodical manner.

A system should be designed to take full advantage of the hardware and language capabilities available. Working in an on-line, interactive environment requires extra planning, especially since multiple user terminals are involved. The processing and file types available have a definite effect on the system design. Disc storage and processing time requirements may vary according to the file and processing types selected.

It is very important to consider the recommended design and programming standards during system design. The selection of possible design and technique alternatives is simplified through the implementation of the recommendations presented in this manual. This also helps the system development to flow smoothly into programming upon the completion of design.

System design must be integrated with the rest of the tasks and activities involved in applications software development. During the development of a system various stages of completion or, "phases", are reached. In order for a system to be produced in an organized and efficient manner, these phases must be anticipated and planned in advance, with each phase leading to the next.

Upon completion of system design sufficient documentation should have been generated to clearly define the system requirements, describe the application fuctions and to specify the technical methods which will provide the most efficient system possible. This documentation must guide and support the programming effort.

Successful applications software for the BASIC/FOUR system is created by using each of the following recommendations:

> Utilize the capabilities provided by the BASIC/FOUR system.
>
> Implement design and programming standards.
>
> Implement phased planning and estimating.
>
> Generate complete documentation which clearly defines all system requirements and functions.

SYSTEM DEVELOPMENT

The basic phases which occur during the system development cycle include:

> Requirements Definition and Analysis
> Preliminary Design
> Detail Design
> Software Construction
> Software Certification
> Installation

Upon completion of the development cycle, an additional phase, Maintenance, provides for the system operation and modification activities.

Operating in a development cycle consisting of a series of interrelated tasks and activities, provides the ability to develop applications software using a systematic approach. Each group of related tasks and activities are performed within specific periods of time, with one phase leading into another. In many cases, major activities may overlap from one phase to the next. The overall time dedicated to each phase varies greatly from project to project and must be controlled by the project manager.

The phased approach to applications software development has been proven to be very successful. It provides a basis for a common structured approach to software engineering which can be continually improved. By thoroughly defining the tasks and activities necessary in development, projects are made easier to plan, estimate and control. During the first three phases, Requirements Definition and Analysis, Preliminary Design and Detail Design, the users participate in selecting and evaluating system alternatives and design. The developer can plan the project in both general outline and detail formats. Individuals are allowed to specialize on phases and tasks and may interchange and transfer activities.

REQUIREMENTS DEFINITION AND ANALYSIS

This phase is usually performed as a pre-sale or feasibility study. In order to successfully determine the system requirements, a good line of communication must be established between the development group and the client. It is important to become familiar with the clients organization and to deal with the highest authority possible. Some of the requirements for the new system may sometimes become apparent by determining the inadequacies of the existing system. The data and transaction volumes, including the expected growth factor are determined. The hardware configuration which is to be installed is verified.

Once the clients problems and requirements are defined and documented, the analyst responds by outlining the tasks to be performed in order to satisfy those requirements. Once the analyst and the client agree on the system definition, a project estimate and a detail estimate for the next phase are prepared.

There are definite advantages to this approach. The developer and client are able to define and analyze the system requirements in a realistic, logical manner. Client participation in this phase provides the client with the opportunity to solidify his objectives. A complete statement of processing requirements is generated which can be used as a basis for further planning and development. (See Work Statement).

The number of costly changes is greatly reduced by having a good, documented requirements definition to work from.

PRELIMINARY DESIGN

The use of a "top down" design technique, from the management level down to a level of detail which identifies the major design features, is used to describe the appropriate design solutions. The design alternatives are evaluated on a cost-benefit basis and must be thoroughly reviewed by the development group as well as the client. Upon selection of an alternative, an implementation plan is developed, the project estimate is updated and the next phase is estimated in detail.

The "top down" design technique; designing a system beginning with the problem level and working down to the computing system level, aides in the analysis process. Cooperation between the analyst and client enables the selection of the optimum alternative solution. The participation of the development group helps to insure technical feasibility and system efficiency.

DETAIL DESIGN

The deisgn alternative selected in the Preliminary Design phase must be documented in detail in order to support the coding effort. The design should be logically tested, reviewed and documented to confirm its validity before coding is begun. The design is reviewed by the development team and the system logic and procedures are thoroughly checked. The client signs a system design acceptance letter and a detail plan and schedule is prepared.

The client and development group again work together during this phase to confirm the logic and correctness of the design. Errors which are detected and corrected now, are far less costly than if they are found after coding begins. The planning and testing of the system are greatly simplified by a thorough, well tested and documented system design.

SOFTWARE CONSTRUCTION

A complete and well documented system design greatly simplifies the coding effort. The coding generated in this phase must fulfill the system requirements as reflected in the system design. Program specifications, file record formats, VDT formats, report formats and a production schedule for the coding and testing of system functions are all critical during this phase. The program structure and coding techniques used should be consistant throughout the system. All detail documentation, including operator instructions, is continually being updated and/or written as coding proceeds.

A production schedule for program development greatly simplifies program and system debugging. Uniformity and consistency are also very critical factors in reducing debugging time as well as maintenance and modification time later. The use of common routines to accomplish this is very effective.

SOFTWARE CERTIFICATION

The system must satisfy comprehensive testing which uses "live" client data. Every program function is thoroughly tested by someone other than the author. System integration features are tested to insure accuracy. Also the documentation is reviewed to make sure it is complete. Client operations personnel training is begun.

Upon completion of the system testing by the development group, the programs are demonstrated to the client for his approval. There should be no surprises unveiled during the demonstration due to the client's involvement during development. The system requirements and design have been previously approved by the client, the system components have been completely tested and all input/output formats and other documentation have been previously reviewed with the client. Any minor changes requested are made and the client signs the program certification letter which activates shipment of the hardware for installation.

INSTALLATION

The application programs are transferred to the clients system. The system documentation is delivered and the operator training is completed using existing test data. The data files are loaded by means of manual entry or automated conversion. Once live processing begins an installation sign off is obtained which begins the pre-determined warranty period. An analyst is assigned to assist the customer in adjusting to the new system and resolve any minor problems which may arise.

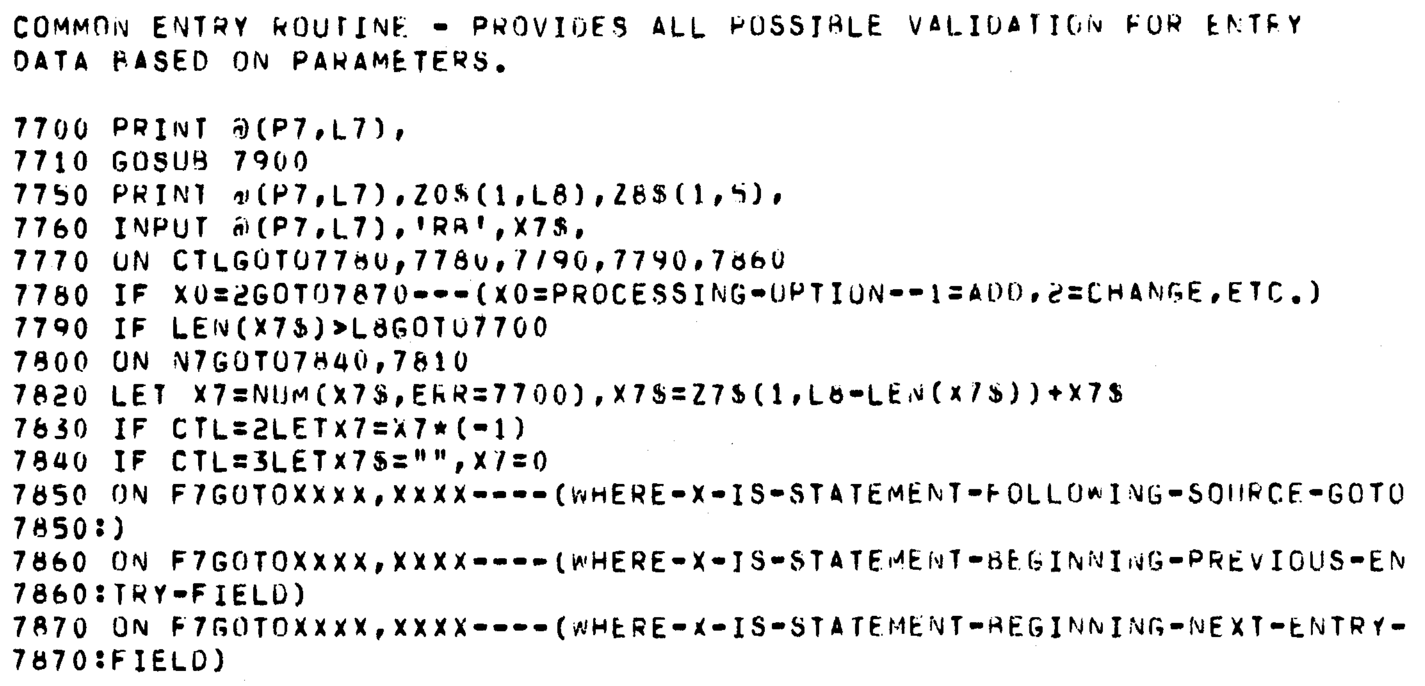
MAINTENANCE

Once the warranty period expires, the client may occasionally require assistance to satisfy additional requirements. The same procedures are used during this phase as are described for development. Changes and additions must be carefully designed, tested, analyzed, reviewed and documented. It is imperative that the system documentation is continually updated and kept current. No change should take place before the documentation is revised. If an existing program is being changed, a copy of the program should be stored and the live data files must be protected until the revised program is certified.

The systematic development of applications software is very rewarding in the Maintenance phase. Programmers not involved in the original development of the system are more easily able to work with a system which is well documented and programmed using consistant techniques.

SCHEDULING

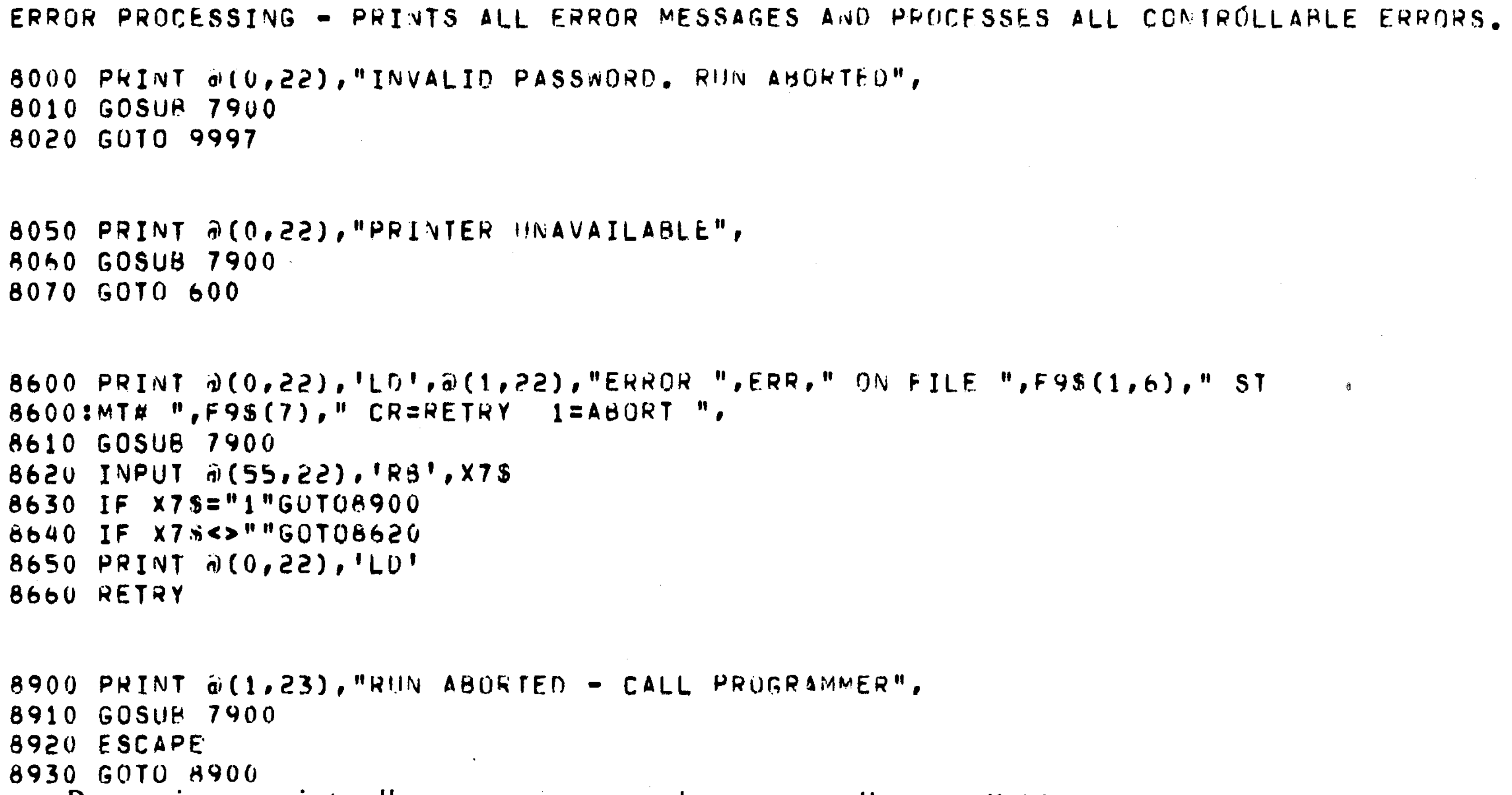
Many of the defined development activities overlap into various phases. The life cycle of the systematic applications software development activities and phases is reflected in the accompanying chart.



```
COMMON ENTRY ROUTINE - PROVIDES ALL POSSIBLE VALIDATION FOR ENTRY
DATA BASED ON PARAMETERS.

7700 PRINT @(P7,L7),
7710 GOSUB 7900
7750 PRINT @(P7,L7),Z0$(1,L8),Z8$(1,5),
7760 INPUT @(P7,L7),'RB',X7$,
7770 ON CTLGOTO7780,7780,7790,7790,7860
7780 IF X0=2GOTO7870---(X0=PROCESSING-OPTION--1=ADD,2=CHANGE,ETC.)
7790 IF LEN(X7$)>L8GOTO7700
7800 ON N7GOTO7840,7810
7820 LET X7=NUM(X7$,ERR=7700),X7$=Z7$(1,L8-LEN(X7$))+X7$
7630 IF CTL=2LETX7=X7*(-1)
7840 IF CTL=3LETX7$="",X7=0
7850 ON F7GOTOXXXX,XXXX----(WHERE-X-IS-STATEMENT-FOLLOWING-SOURCE-GOTO
7850:)
7860 ON F7GOTOXXXX,XXXX----(WHERE-X-IS-STATEMENT-BEGINNING-PREVIOUS-EN
7860:TRY-FIELD)
7870 ON F7GOTOXXXX,XXXX----(WHERE-X-IS-STATEMENT-BEGINNING-NEXT-ENTRY-
7870:FIELD)
```

Common Entry Routine – provides all possible validation for entry data based on parameters.

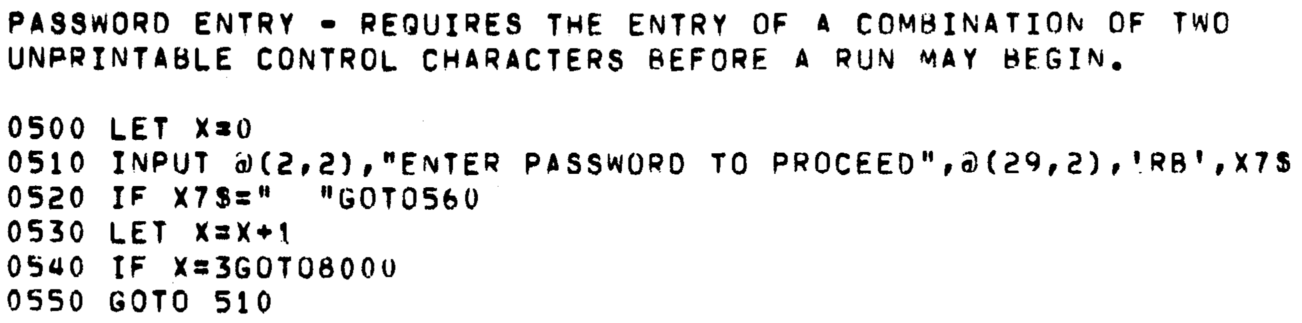
Figure 3-l. Common Entry Routine



```
ERROR PROCESSING - PRINTS ALL ERROR MESSAGES AND PROCESSES ALL CONTROLLABLE ERRORS.

8000 PRINT @(0,22),"INVALID PASSWORD. RUN ABORTED",
8010 GOSUB 7900
8020 GOTO 9997


8050 PRINT @(0,22),"PRINTER UNAVAILABLE",
8060 GOSUB 7900
8070 GOTO 600


8600 PRINT @(0,22),'LD',@(1,22),"ERROR ",ERR," ON FILE ",F9$(1,6)," ST
8600:MT# ",F9$(7)," CR=RETRY  1=ABORT ",
8610 GOSUB 7900
8620 INPUT @(55,22),'RB',X7$
8630 IF X7$="1"GOTO8900
8640 IF X7$<>""GOTO8620
8650 PRINT @(0,22),'LD'
8660 RETRY


8900 PRINT @(1,23),"RUN ABORTED - CALL PROGRAMMER",
8910 GOSUB 7900
8920 ESCAPE
8930 GOTO 8900
```

Error Processing – prints all error messages and processes all controllable errors.

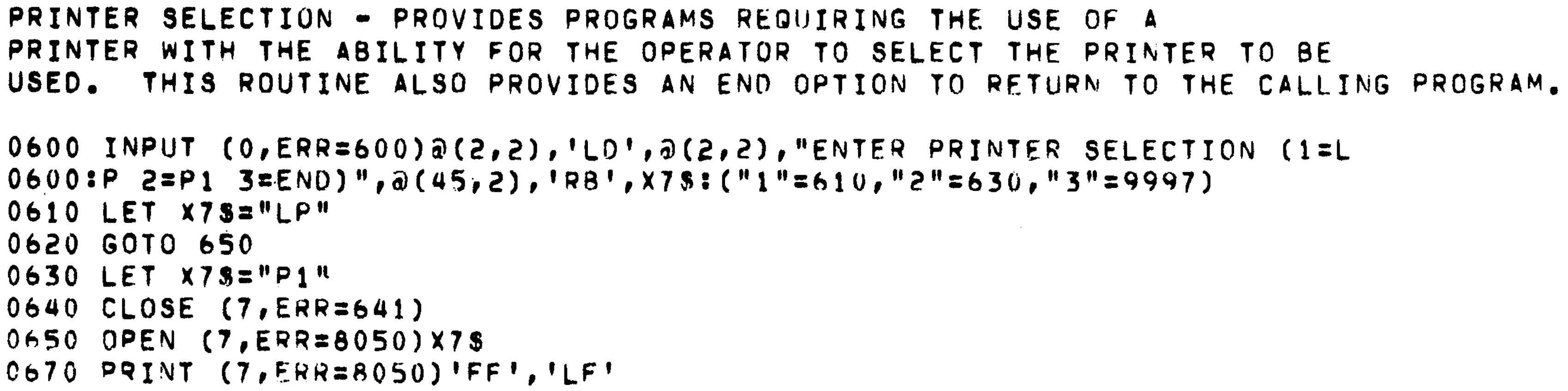
Figure 3-2. Error Processing Routine Example



```
PASSWORD ENTRY - REQUIRES THE ENTRY OF A COMBINATION OF TWO
UNPRINTABLE CONTROL CHARACTERS BEFORE A RUN MAY BEGIN.

0500 LET X=0
0510 INPUT @(2,2),"ENTER PASSWORD TO PROCEED",@(29,2),'RB',X7$
0520 IF X7$="  "GOTO560
0530 LET X=X+1
0540 IF X=3GOTO8000
0550 GOTO 510
```

Password Entry — requires the entry of a combination of two unprintable control characters before a run may begin.

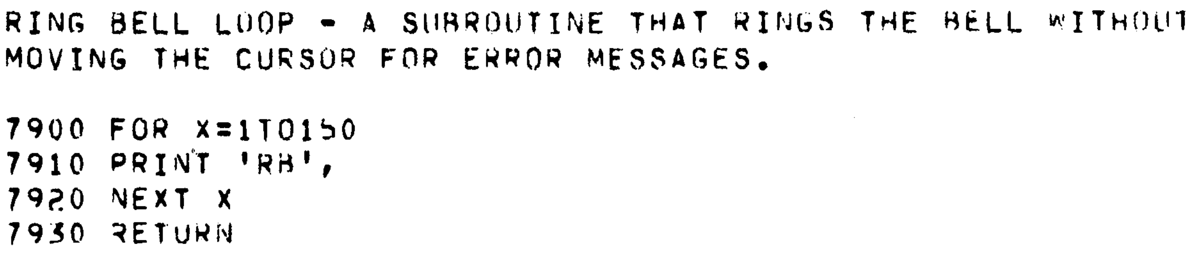
Figure 3-3. Password Entry Routine Example



```
PRINTER SELECTION - PROVIDES PROGRAMS REQUIRING THE USE OF A
PRINTER WITH THE ABILITY FOR THE OPERATOR TO SELECT THE PRINTER TO BE
USED.  THIS ROUTINE ALSO PROVIDES AN END OPTION TO RETURN TO THE CALLING PROGRAM.

0600 INPUT (0,ERR=600)@(2,2),'LD',@(2,2),"ENTER PRINTER SELECTION (1=L
0600:P 2=P1 3=END)",@(45,2),'RB',X7$:("1"=610,"2"=630,"3"=9997)
0610 LET X7$="LP"
0620 GOTO 650
0630 LET X7$="P1"
0640 CLOSE (7,ERR=641)
0650 OPEN (7,ERR=8050)X7$
0670 PRINT (7,ERR=8050)'FF','LF'
```

Printer Selection — provides programs requiring the use of a printer with the ability for the operator to select the printer to be used. This routine also provides an end option to return to the calling program.

Figure 3-4. Printer Selection Routine Example



```
RING BELL LOOP - A SUBROUTINE THAT RINGS THE BELL WITHOUT
MOVING THE CURSOR FOR ERROR MESSAGES.

7900 FOR X=1TO150
7910 PRINT 'RB',
7920 NEXT X
7930 RETURN
```

Ring Bell Loop — A sub-routine that rings the bell without moving the cursor for error messages.

Figure 3-5. Ring Bell Loop Sub-Routine Example

SECTION IV – USER DOCUMENTATION

In order to be complete, a system or program must be thoroughly documented as well as functional. Documentation is not generated exclusively for the programmer or analyst, but must be provided to the user upon installation of the system.

DESIGN DOCUMENTATION

The requirements for design documentation are:

A *work statement* defining the requirements of the system. This includes descriptions of: every system function, all source and output documents, transaction volumes, file sizes, and report frequency. Note that a more specific definition is done in the program specification stage.

> The work statement should be prepared by the software vendor, and may be included in the proposal to the Customer, then made a part of the Application Software Addendum (form BFC 1171) by reference.
>
> The work statement should consist of a minimum of four sections (Section V is optional).
>
> I. APPLICATIONS
>
> II. OPERATION DESCRIPTIONS
>
> III. VOLUMES
>
> IV. TERMS AND CONDITIONS
>
> V. VENDOR QUALIFICATIONS (optional)
>
> When writing the work statement, in the context of a proposal, use words like " . . . your Payroll System will include . . ." Use language which the customer can understand, and do not include flow charts unless they are extremely simple. The work statement need not be lengthy, but must define the scope of the job that will be done for the customer.
>
> Start a new page for each section of the work statement. Put a date on each page of each section, along with the name of the customer. Then, in the Consultant Order and Application Software Addendum, refer to the work statement by date.
>
> The applications section should contain a narrative, a list of input documents, a list of output documents, and an optional flow chart for each application. The flowchart must be extremely simple and should be included unless it is determined to be objectionable or confusing to the customer.
>
> If vendor wishes to include his qualifications in the work statement, put such qualifications at the end, in Section V.
>
> Sample work statements for a payroll application and an order entry application are shown in the Appendix.

A simple *system overview flowchart* should illustrate the major processing modules in the system. (See Figure 4-2). A comprehensive *processing flowchart* should reflect all input-process-output flow for all application functions (See Figure 4-1).

File Record Formats, BFC Form 1021 (See Figure 4-4) are required for all files and all record types used by the system and are accompanied by Data File Definition Sheets, BFC Form 1016 (See Figure 4-3).

VDT Screen Formats are necessary for all system functions. BFC Form 1022, accommodates all VDT models. However, if only model 7220 and 7230 screens are to be used, BFC Form 1275, is especially designed to allow typing on up to 25 lines with 80 characters each. A sample VDT screen format is provided (See Figure 4-5).

Report Formatting Sheets, BFC 1023, are required for all system reports (See Figure 4-6).

A *Program/File Matrix,* BFC 1261, must list all system applications and specify file activity for each application (See Figure 4-8).

The reservation and assignment of file and program names must be documented in a means which reflects the assignment scheme. The File/Program Assignment Matrix, BFC 1262 (See Figure 4-9), is available for this purpose.

Program Specifications, BFC 1263, describes the purpose of each program and its associated link programs, if any (See Figure 4-10). Special conditions and requirements should be included such as: processing conditions (flag status checking), algorithms and formulas. Files used and updated must be listed as well as a detailed description of the logical operation required. Care should be taken to cover the specifications in enough detail to support the programming phase.

Milestone Charts, BFC 1161, reflects the project schedule (See Figure 4-7).

System documentation is generated as part of the system design phase and is updated as required during the programming and debugging phases.

PROGRAM DOCUMENTATION

The requirements for program documentation are:

**M Utility Program Listings and Analysis,* (See BASIC/FOUR Utility Manual, BFC 5002 or BFC 5024).

**A Utility Directory Listing,* (See BASIC/FOUR Utility Manual, BFC 5002 or BFC 5024).

Program Flow Charts for complex programs only.

Program documentation is regarded as part of programming and must be completed before the program may be considered finished.

OPERATOR DOCUMENTATION

The requirements for Operator Documentation are:

Operator Instruction Narratives (See Figure 4-11) must describe the application function, the conditions and/or requirements of operation and the output produced.

Operator Instructions must provide the entry requirements for each screen field and any processing sequence requirements (See Figure 4-12).

Specify and document all cyclical processing requirements including the daily, weekly, monthly, quarterly and annual cycles. This information is critical in order to promote smooth operation of the system.

The best time for operator documentation to be completed is immediately after the program is written and debugged. However, operator documentation must be completed prior to the installation of the system.

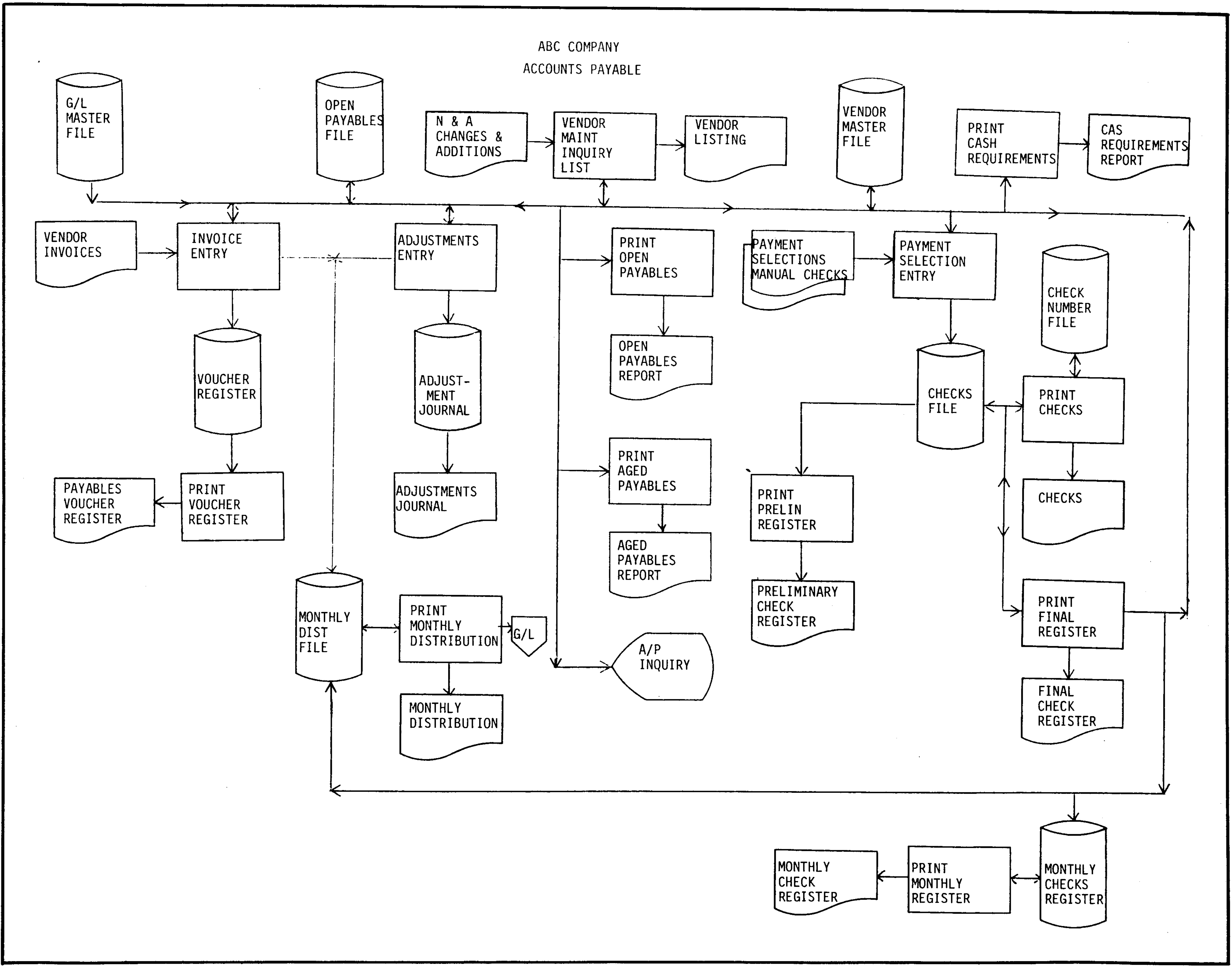


Figure 4-1. Simple System Flow Chart

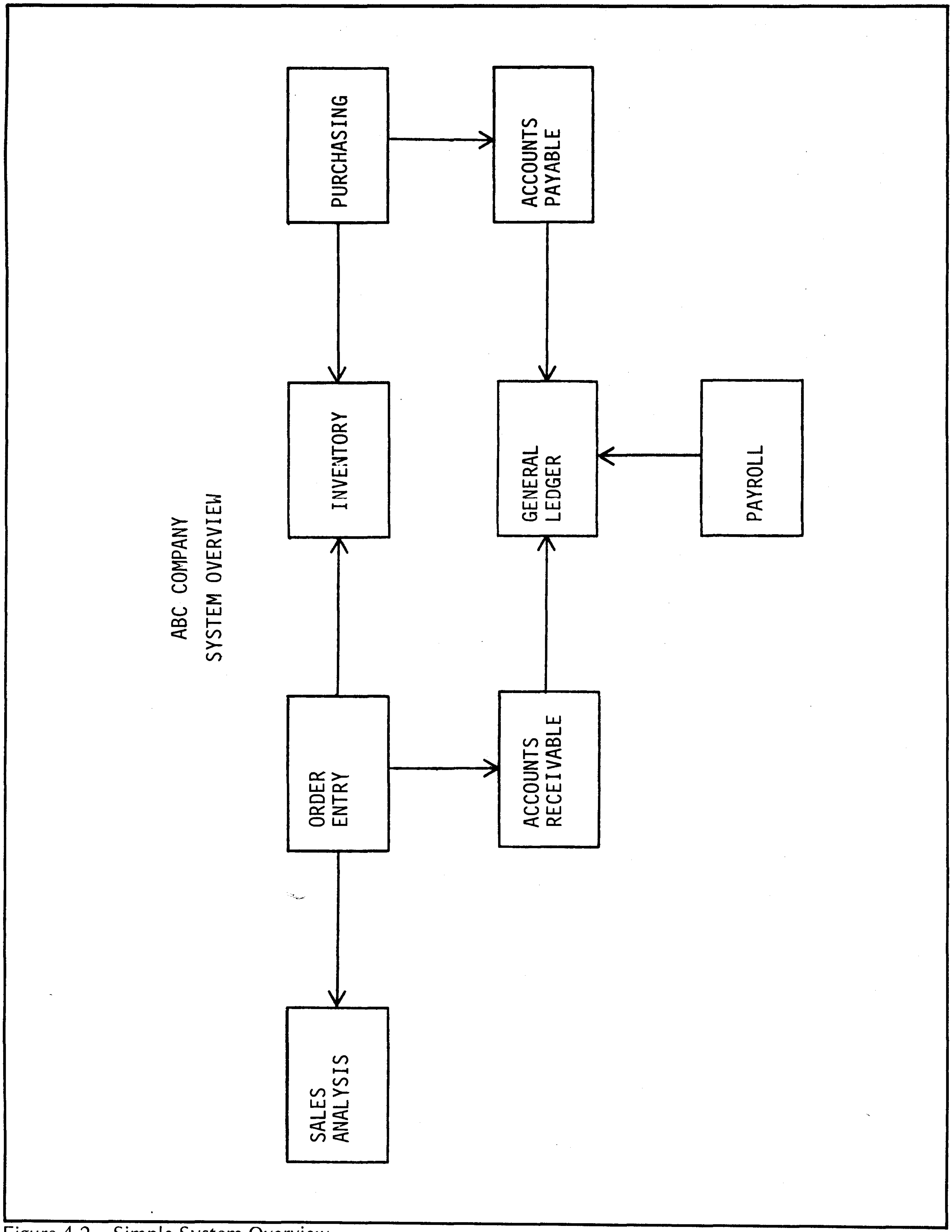
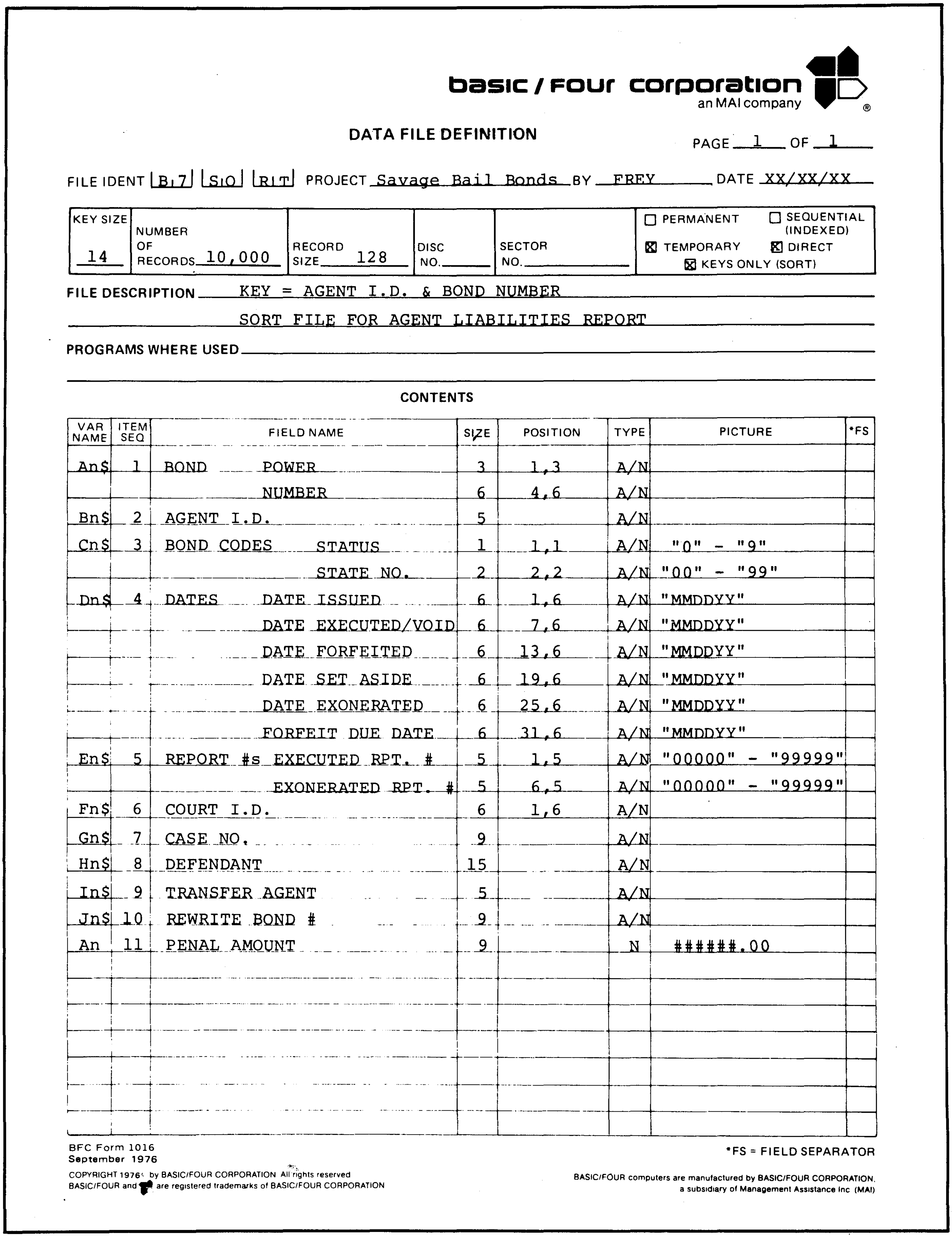


Figure 4-2. Simple System Overview



basic / four corporation
an MAI company ®

DATA FILE DEFINITION

PAGE 1 OF 1

FILE IDENT B 7 S O R T PROJECT Savage Bail Bonds BY FREY DATE XX/XX/XX

KEY SIZE 14	NUMBER OF RECORDS 10,000	RECORD SIZE 128	DISC NO.	SECTOR NO.	□ PERMANENT □ SEQUENTIAL (INDEXED) ☒ TEMPORARY ☒ DIRECT ☒ KEYS ONLY (SORT)

FILE DESCRIPTION KEY = AGENT I.D. & BOND NUMBER

SORT FILE FOR AGENT LIABILITIES REPORT

PROGRAMS WHERE USED

CONTENTS

VAR NAME	ITEM SEQ	FIELD NAME	SIZE	POSITION	TYPE	PICTURE	*FS
An$	1	BOND POWER	3	1,3	A/N		
		NUMBER	6	4,6	A/N		
Bn$	2	AGENT I.D.	5		A/N		
Cn$	3	BOND CODES STATUS	1	1,1	A/N	"0" - "9"	
		STATE NO.	2	2,2	A/N	"00" - "99"	
Dn$	4	DATES DATE ISSUED	6	1,6	A/N	"MMDDYY"	
		DATE EXECUTED/VOID	6	7,6	A/N	"MMDDYY"	
		DATE FORFEITED	6	13,6	A/N	"MMDDYY"	
		DATE SET ASIDE	6	19,6	A/N	"MMDDYY"	
		DATE EXONERATED	6	25,6	A/N	"MMDDYY"	
		FORFEIT DUE DATE	6	31,6	A/N	"MMDDYY"	
En$	5	REPORT #s EXECUTED RPT. #	5	1,5	A/N	"00000" - "99999"	
		EXONERATED RPT. #	5	6,5	A/N	"00000" - "99999"	
Fn$	6	COURT I.D.	6	1,6	A/N		
Gn$	7	CASE NO.	9		A/N		
Hn$	8	DEFENDANT	15		A/N		
In$	9	TRANSFER AGENT	5		A/N		
Jn$	10	REWRITE BOND #	9		A/N		
An	11	PENAL AMOUNT	9		N	######.00	

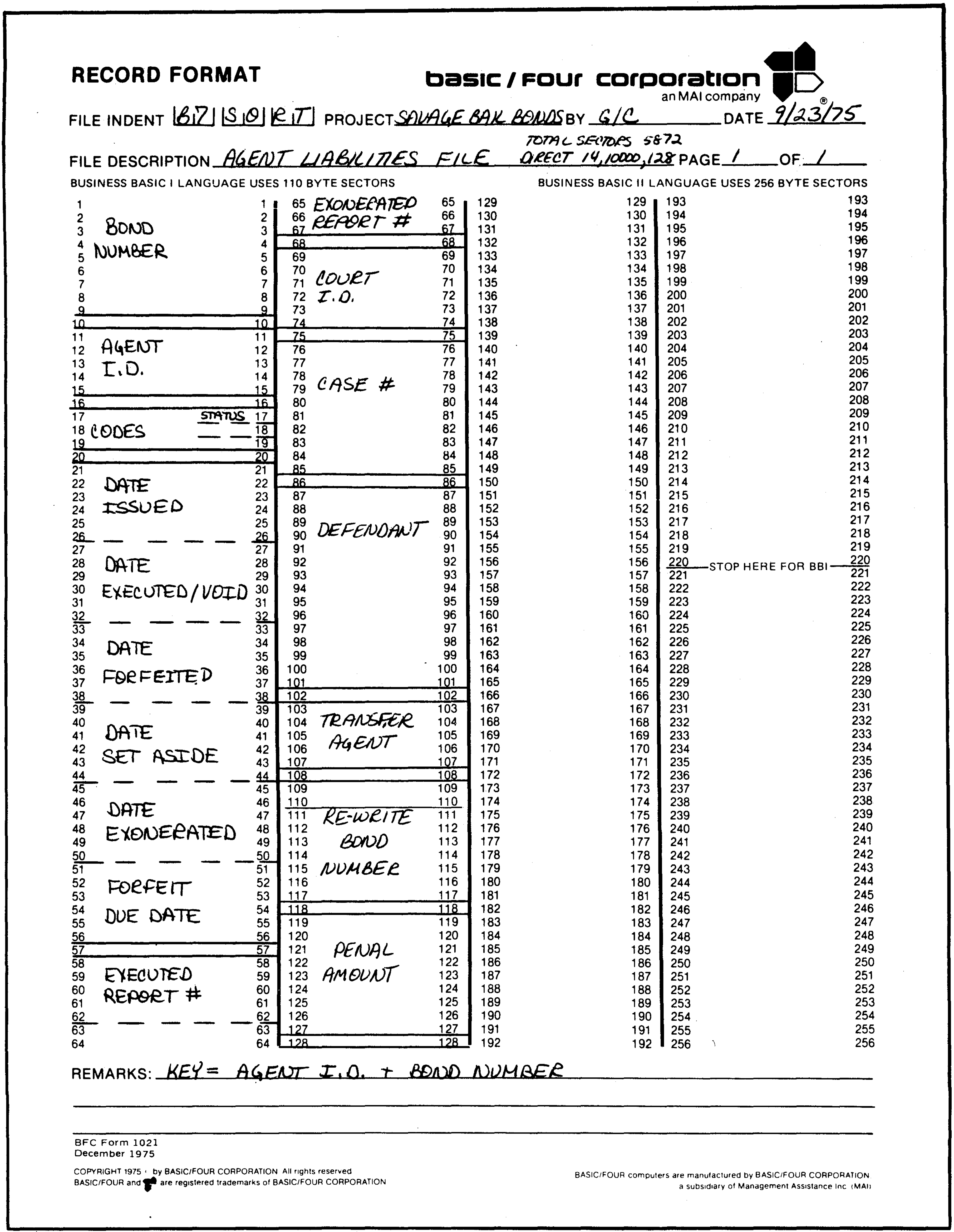
BFC Form 1016
September 1976

*FS = FIELD SEPARATOR

COPYRIGHT 1976 by BASIC/FOUR CORPORATION All rights reserved
BASIC/FOUR and are registered trademarks of BASIC/FOUR CORPORATION

BASIC/FOUR computers are manufactured by BASIC/FOUR CORPORATION,
a subsidiary of Management Assistance Inc (MAI)

Figure 4-3. Data File Definition Form (BFC 1016)



RECORD FORMAT

basic / FOUR corporation
an MAI company

FILE INDENT |B|7| |S|0| |R|T| PROJECT SAVAGE BAIL BONDS BY G/C DATE 9/23/75

TOTAL SECTORS 5872

FILE DESCRIPTION AGENT LIABILITIES FILE DIRECT 14,10000,128 PAGE 1 OF 1

BUSINESS BASIC I LANGUAGE USES 110 BYTE SECTORS — BUSINESS BASIC II LANGUAGE USES 256 BYTE SECTORS

1		1	65	EXONERATED	65	129	129	193		193
2		2	66	REPORT #	66	130	130	194		194
3	BOND	3	67		67	131	131	195		195
4		4	68		68	132	132	196		196
5	NUMBER	5	69		69	133	133	197		197
6		6	70		70	134	134	198		198
7		7	71	COURT	71	135	135	199		199
8		8	72	I.D.	72	136	136	200		200
9		9	73		73	137	137	201		201
10		10	74		74	138	138	202		202
11		11	75		75	139	139	203		203
12	AGENT	12	76		76	140	140	204		204
13	I.D.	13	77		77	141	141	205		205
14		14	78		78	142	142	206		206
15		15	79	CASE #	79	143	143	207		207
16		16	80		80	144	144	208		208
17	STATUS	17	81		81	145	145	209		209
18	CODES — —	18	82		82	146	146	210		210
19		19	83		83	147	147	211		211
20		20	84		84	148	148	212		212
21		21	85		85	149	149	213		213
22	DATE	22	86		86	150	150	214		214
23		23	87		87	151	151	215		215
24	ISSUED	24	88		88	152	152	216		216
25		25	89		89	153	153	217		217
26	— — — — —	26	90	DEFENDANT	90	154	154	218		218
27		27	91		91	155	155	219		219
28	DATE	28	92		92	156	156	220	STOP HERE FOR BBI	220
29		29	93		93	157	157	221		221
30	EXECUTED/VOID	30	94		94	158	158	222		222
31		31	95		95	159	159	223		223
32	— — — — —	32	96		96	160	160	224		224
33		33	97		97	161	161	225		225
34	DATE	34	98		98	162	162	226		226
35		35	99		99	163	163	227		227
36	FORFEITED	36	100		100	164	164	228		228
37		37	101		101	165	165	229		229
38	— — — — —	38	102		102	166	166	230		230
39		39	103		103	167	167	231		231
40		40	104	TRANSFER	104	168	168	232		232
41	DATE	41	105		105	169	169	233		233
42		42	106	AGENT	106	170	170	234		234
43	SET ASIDE	43	107		107	171	171	235		235
44	— — — — —	44	108		108	172	172	236		236
45		45	109		109	173	173	237		237
46		46	110		110	174	174	238		238
47	DATE	47	111	RE-WRITE	111	175	175	239		239
48	EXONERATED	48	112		112	176	176	240		240
49		49	113	BOND	113	177	177	241		241
50	— — — — —	50	114		114	178	178	242		242
51		51	115	NUMBER	115	179	179	243		243
52	FORFEIT	52	116		116	180	180	244		244
53		53	117		117	181	181	245		245
54	DUE DATE	54	118		118	182	182	246		246
55		55	119		119	183	183	247		247
56		56	120		120	184	184	248		248
57		57	121	PENAL	121	185	185	249		249
58		58	122		122	186	186	250		250
59	EXECUTED	59	123	AMOUNT	123	187	187	251		251
60		60	124		124	188	188	252		252
61	REPORT #	61	125		125	189	189	253		253
62	— — — — —	62	126		126	190	190	254		254
63		63	127		127	191	191	255		255
64		64	128		128	192	192	256		256

REMARKS: KEY = AGENT I.D. + BOND NUMBER

BFC Form 1021
December 1975

BASIC/FOUR computers are manufactured by BASIC/FOUR CORPORATION
a subsidiary of Management Assistance Inc (MAI)

Figure 4-4. File Record Format Form (BFC 1021)

Figure 4-5. VDT Formatting Sheet Form (BFC 1022)

VIDEO DISPLAY TERMINAL FORMATTING SHEET

basic / four

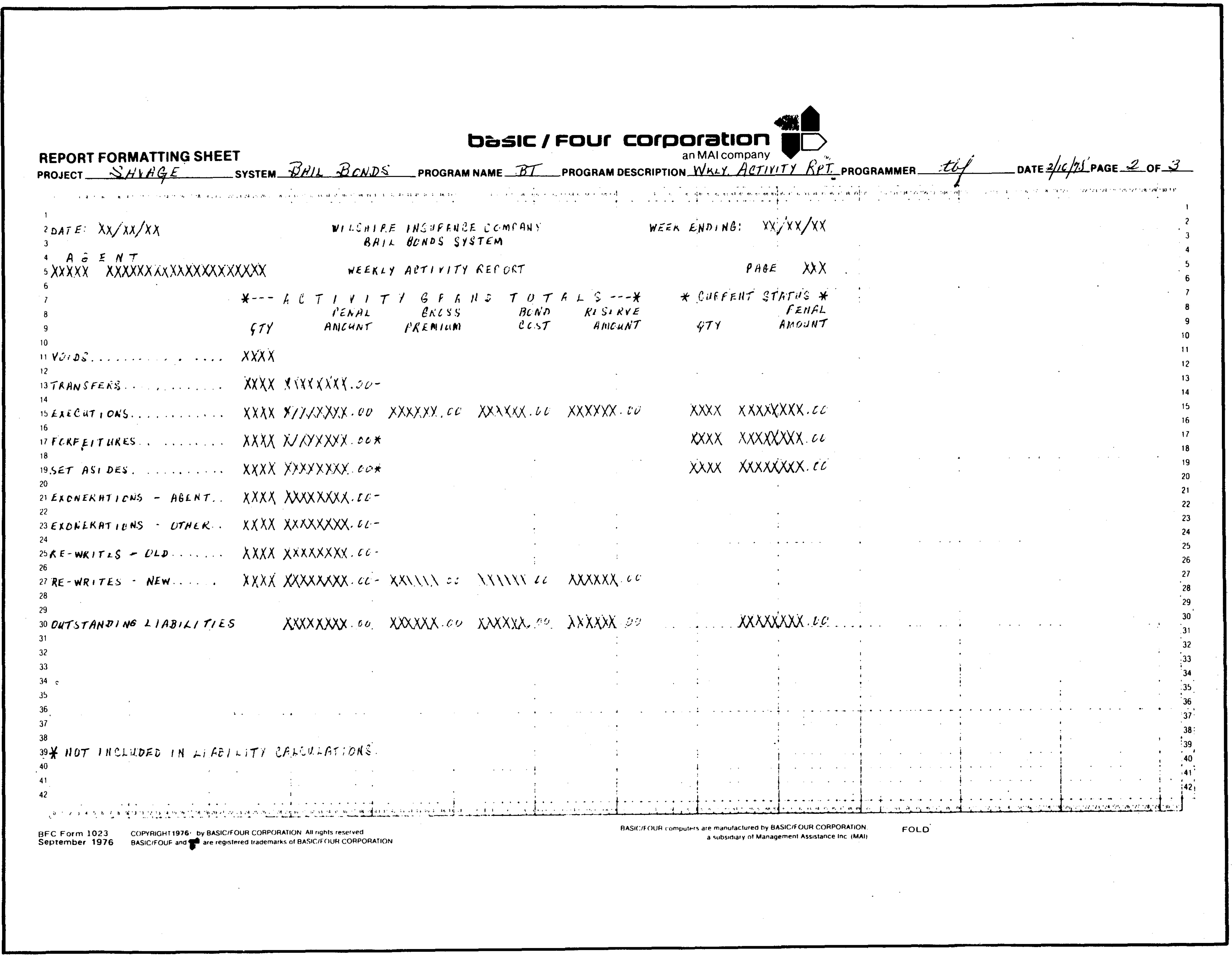
PROJECT SAVAGE PROGRAM NAME CJ, CK PROGRAMMER D

SYSTEM BAIL BONDS PROGRAM DESCRIPTION GEN. AGENT MAINTENANCE DATE 12/4/78 PAGE 1 OF 1

```
    0 1 2 3 4 5 6 7 8 9 10 11 12 ... 73
 0  * BAIL BONDS - GENERAL AGENT MAINTENANCE                                        0
 1                                                                                   1
 2  ENTER  1-ADD  2-CHANGE  3-DELETE  4-INQUIRE  5-END ☒                             2
 3                                                                                   3
 4     GENERAL AGENT NO.....  XXX   [MISSING DUPLICATE]                              4
 5  1. GENERAL AGENT NAME...  XXXXXXXXXXXXXXXXXXXXX         XXXXXXXXXXXXXXXXXXXX     5
 6  2. RETENTION %.........   XX.00                          XX.00                   6
 7     NUMBER OF AGENTS.....  XXX                            XXX                     7
 8                                                                                   8
 9                              *WILSHIRE*      * AMWEST *      * TOTAL *            9
10     NO. OF ACTIVE BONDS..         XXXX0          XXXX0           XXXX0            10
11     PENAL LIABILITY......    XXXXXXXX.00    XXXXXXXX.00     XXXXXXXX.00           11
12     GROSS PREMIUM........    XXXXXXXX.00    XXXXXXXX.00     XXXXXXXX.00           12
13                                                                                   13
14                                                                                   14
15                                                                                   15
16                                                                                   16
17                                                                                   17
18                                                                                   18
19                                                                                   19
20  ENTER LINE NO. TO CORRECT OR CR TO PROCEED........ □                             20
21                                                                                   21
22 [ ERROR MESSAGES                                                                  22
23                                                                                   23
24                                                                                   24
25                                                                                   25
26                                                                                   26
    0 1 2 3 4 5 6 7 8 9 10 11 12 ... 73
```

REMARKS: only numbered items may be changed.
number of active bonds and penal liability must be zero to delete.

BFC Form 1022
Revised: September 28, 1971



basic / FOUR corporation
an MAI company

REPORT FORMATTING SHEET

PROJECT SALVAGE SYSTEM BAIL BONDS PROGRAM NAME BT PROGRAM DESCRIPTION WKLY. ACTIVITY RPT. PROGRAMMER tbf DATE 2/10/75 PAGE 2 OF 3

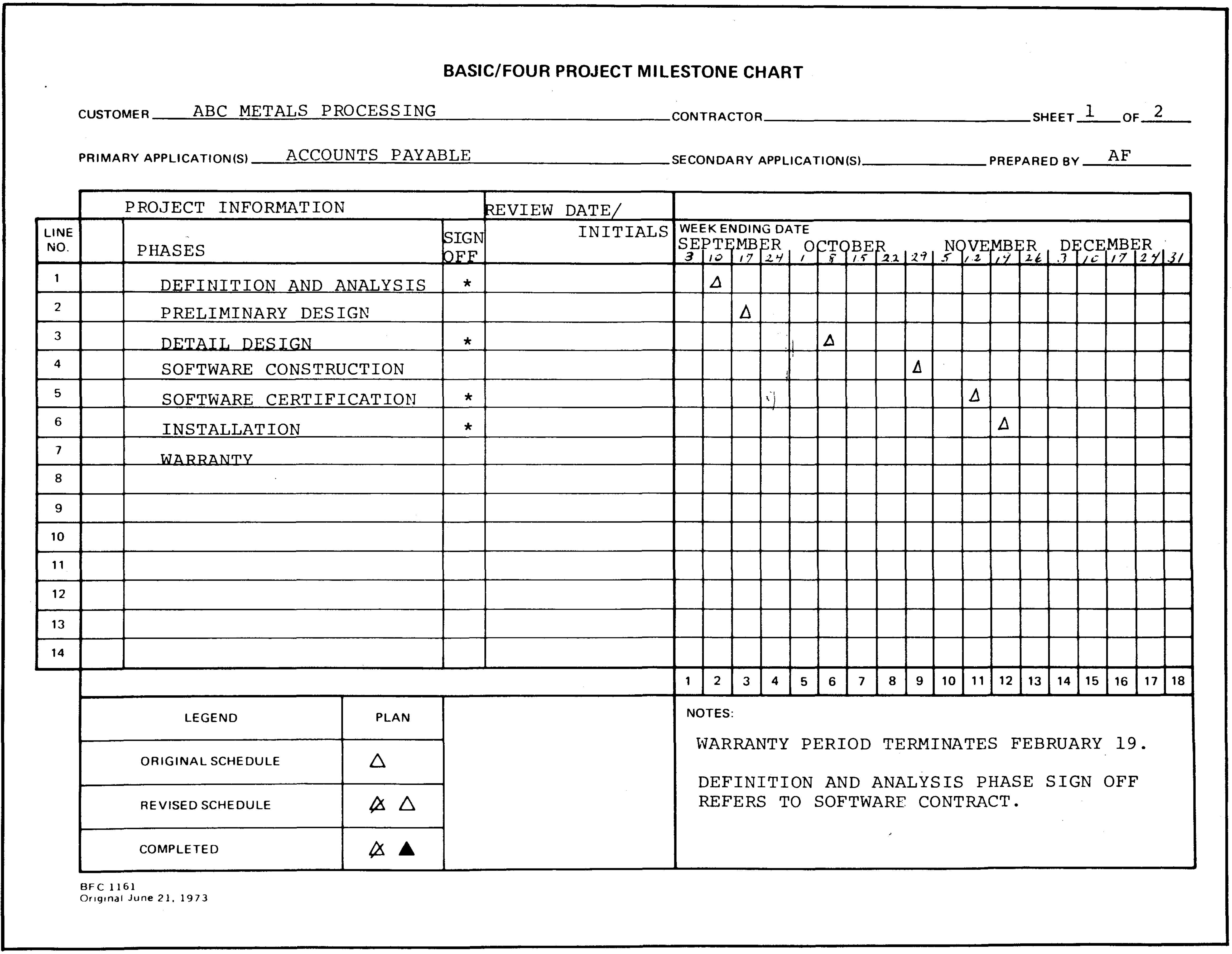
```
DATE: XX/XX/XX                           WILSHIRE INSURANCE COMPANY                   WEEK ENDING:  XX/XX/XX
                                            BAIL BONDS SYSTEM
 A G E N T
XXXXX  XXXXXXXXXXXXXXXXXXXX               WEEKLY ACTIVITY REPORT                         PAGE   XXX

                          *--- A C T I V I T Y   G R A N D   T O T A L S ---*        * CURRENT STATUS *
                                     PENAL        GROSS        BOND      RESERVE                  PENAL
                          QTY        AMOUNT       PREMIUM      COST      AMOUNT         QTY       AMOUNT

VOIDS...................  XXXX

TRANSFERS...............  XXXX  XXXXXXXX.00-

EXECUTIONS..............  XXXX  XXXXXXXX.00  XXXXXX.00  XXXXXX.00  XXXXXX.00        XXXX   XXXXXXXX.00

FORFEITURES.............  XXXX  XXXXXXXX.00*                                          XXXX   XXXXXXXX.00

SET ASIDES..............  XXXX  XXXXXXXX.00*                                          XXXX   XXXXXXXX.00

EXONERATIONS - AGENT..    XXXX  XXXXXXXX.00-

EXONERATIONS - OTHER..    XXXX  XXXXXXXX.00-

RE-WRITES - OLD.......    XXXX  XXXXXXXX.00-

RE-WRITES - NEW.....      XXXX  XXXXXXXX.00-  XXXXXX.00  XXXXXX.00  XXXXXX.00


OUTSTANDING LIABILITIES         XXXXXXXX.00  XXXXXX.00  XXXXXX.00  XXXXXX.00                 XXXXXXXX.00








* NOT INCLUDED IN LIABILITY CALCULATIONS
```

FOLD

BFC Form 1023
September 1976

Figure 4-6. Report Formatting Sheet Form (BFC 1023)

Figure 4-7. Milestone Chart (BFC 1161) (1 of 2)



BASIC/FOUR PROJECT MILESTONE CHART

CUSTOMER ABC METALS PROCESSING CONTRACTOR ______ SHEET 1 OF 2

PRIMARY APPLICATION(S) ACCOUNTS PAYABLE SECONDARY APPLICATION(S) ______ PREPARED BY AF

		PROJECT INFORMATION		REVIEW DATE/																		
					WEEK ENDING DATE																	
LINE NO.		PHASES	SIGN OFF	INITIALS	SEPTEMBER				OCTOBER					NOVEMBER				DECEMBER				
					3	10	17	24	1	8	15	22	29	5	12	19	26	3	10	17	24	31
1		DEFINITION AND ANALYSIS	*			△																
2		PRELIMINARY DESIGN					△															
3		DETAIL DESIGN	*							△												
4		SOFTWARE CONSTRUCTION											△									
5		SOFTWARE CERTIFICATION	*												△							
6		INSTALLATION	*													△						
7		WARRANTY																				
8																						
9																						
10																						
11																						
12																						
13																						
14																						
					1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

LEGEND	PLAN
ORIGINAL SCHEDULE	△
REVISED SCHEDULE	~~△~~ △
COMPLETED	~~△~~ ▲

NOTES:

WARRANTY PERIOD TERMINATES FEBRUARY 19.

DEFINITION AND ANALYSIS PHASE SIGN OFF REFERS TO SOFTWARE CONTRACT.

BFC 1161
Original June 21, 1973

Figure 4-7. Milestone Chart (BFC 1161) (2 of 2)

BASIC/FOUR PROJECT MILESTONE CHART

CUSTOMER ABC METALS PROCESSING CONTRACTOR ______ SHEET 2 OF 2

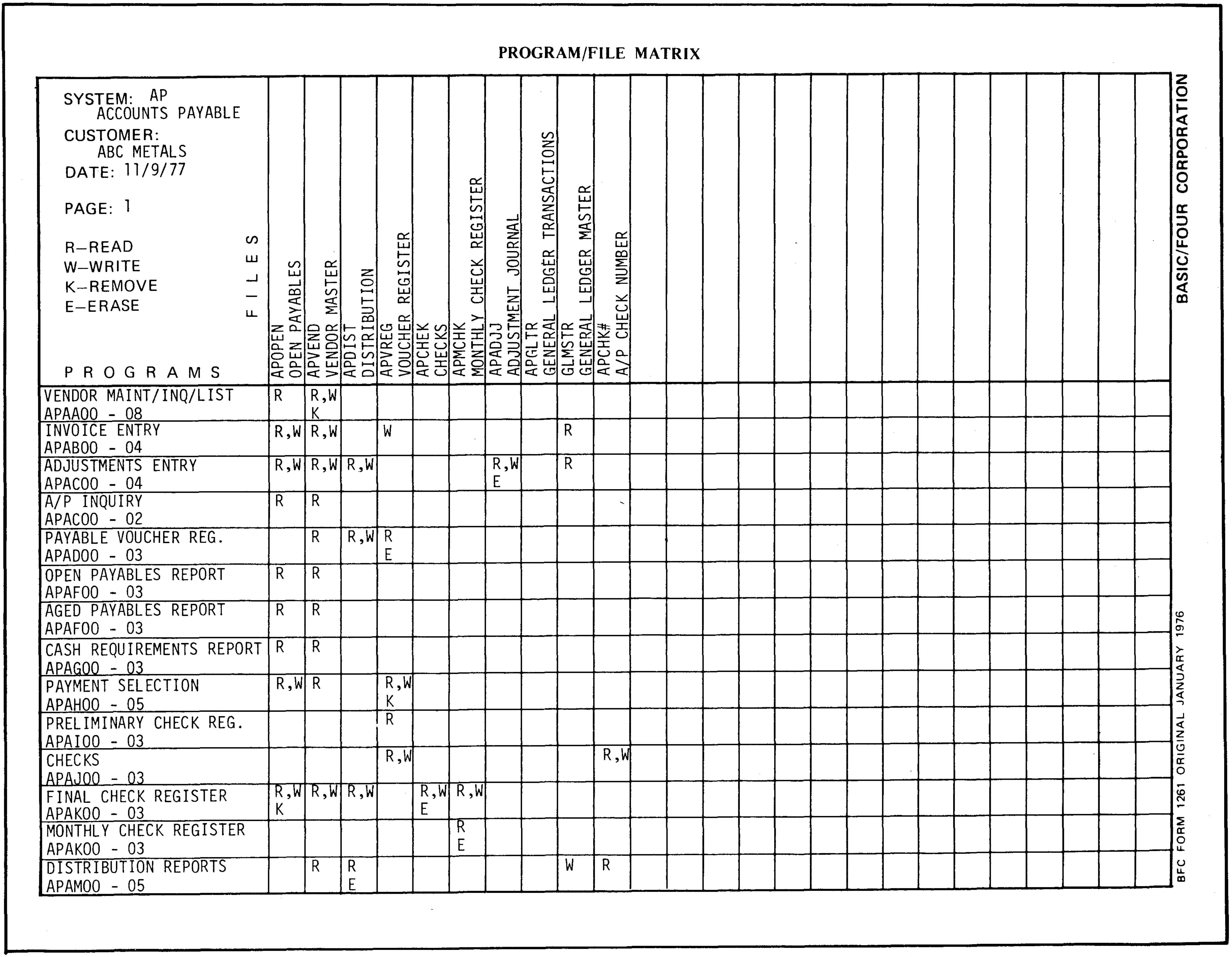
PRIMARY APPLICATION(S) ACCOUNTS PAYABLE I.D.=AP SECONDARY APPLICATION(S) ______ PREPARED BY AF

	PROJECT INFORMATION			REVIEW DATE/ INITIALS	WEEK ENDING DATE																	
LINE NO.	PROG NAME	PROGRAMMING SCHEDULE	PGMR		SEPTEMBER				OCTOBER					NOVEMBER				DECEMBER				
					3	10	17	24	1	8	15	22	29	5	12	19	26	3	10	17	24	31
1	AA	VENDOR MAINTENANCE/INQUIRY/LIST	SUE						△													
2	AB	INVOICE ENTRY	AL						△													
3	AC	ADJUSTMENTS ENTRY	AL							△												
4	AD	A/P INQUIRY	SUE							△												
5	AE	PAYABLE VOUCHER REGISTER	SUE							△												
6	AF	OPEN PAYABLES REPORT	SUE								△											
7	AG	AGED PAYABLES REPORT	SUE								△											
8	AH	CASH REQUIREMENTS REPORT	AL								△											
9	AI	PAYMENT SELECTION	AL								△											
10	AJ	PRELIMINARY CHECK REGISTER	AL									△										
11	AK	CHECKS	AL									△										
12	AL	FINAL CHECK REGISTER	AL									△										
13	AM	MONTHLY CHECK REGISTER	SUE									△										
14	AN	DISTRIBUTION REPORTS	SUE									△										
					1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

LEGEND	PLAN
ORIGINAL SCHEDULE	△
REVISED SCHEDULE	△̸ △
COMPLETED	△̸ ▲

NOTES:

BFC 1161
Original June 21, 1973



PROGRAM/FILE MATRIX

SYSTEM: AP
ACCOUNTS PAYABLE
CUSTOMER:
ABC METALS
DATE: 11/9/77

PAGE: 1

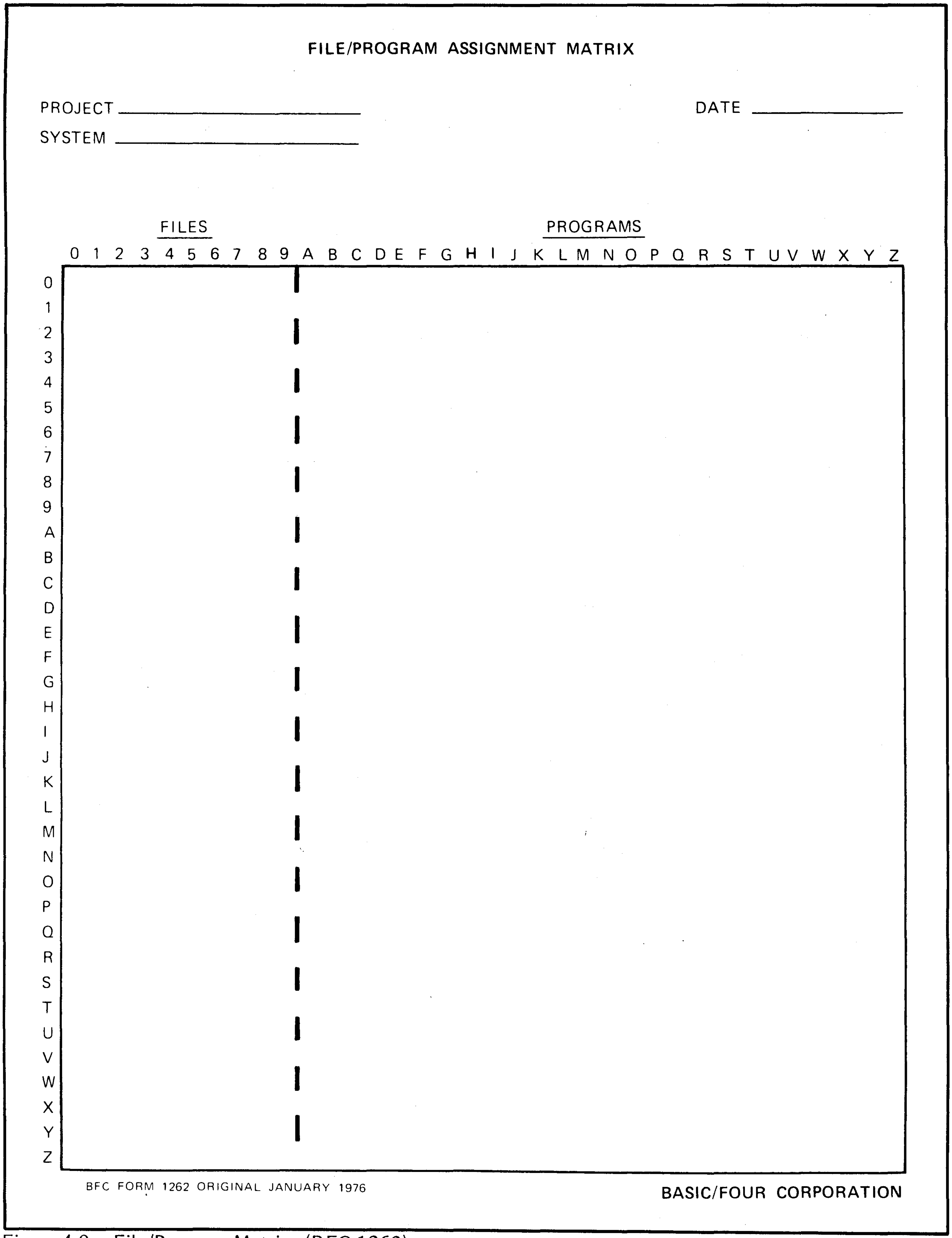
R–READ
W–WRITE
K–REMOVE
E–ERASE

PROGRAMS \ FILES	APOPEN OPEN PAYABLES	APVEND VENDOR MASTER	APDIST DISTRIBUTION	APVREG VOUCHER REGISTER	APCHEK CHECKS	APMCHK MONTHLY CHECK REGISTER	APADJJ ADJUSTMENT JOURNAL	APGLTR GENERAL LEDGER TRANSACTIONS	GLMSTR GENERAL LEDGER MASTER	APCHK# A/P CHECK NUMBER
VENDOR MAINT/INQ/LIST APAA00 - 08	R	R,W K								
INVOICE ENTRY APAB00 - 04	R,W	R,W		W					R	
ADJUSTMENTS ENTRY APAC00 - 04	R,W	R,W	R,W				R,W E		R	
A/P INQUIRY APAC00 - 02	R	R								
PAYABLE VOUCHER REG. APAD00 - 03		R	R,W	R E						
OPEN PAYABLES REPORT APAF00 - 03	R	R								
AGED PAYABLES REPORT APAF00 - 03	R	R								
CASH REQUIREMENTS REPORT APAG00 - 03	R	R								
PAYMENT SELECTION APAH00 - 05	R,W	R		R,W K						
PRELIMINARY CHECK REG. APAI00 - 03				R						
CHECKS APAJ00 - 03				R,W						R,W
FINAL CHECK REGISTER APAK00 - 03	R,W K	R,W	R,W		R,W E	R,W				
MONTHLY CHECK REGISTER APAK00 - 03						R E				
DISTRIBUTION REPORTS APAM00 - 05		R	R E						W	R

BASIC/FOUR CORPORATION

BFC FORM 1261 ORIGINAL JANUARY 1976

Figure 4-8. Program/File Matrix Form (BFC 1261)



FILE/PROGRAM ASSIGNMENT MATRIX

PROJECT ____________________ DATE ______________

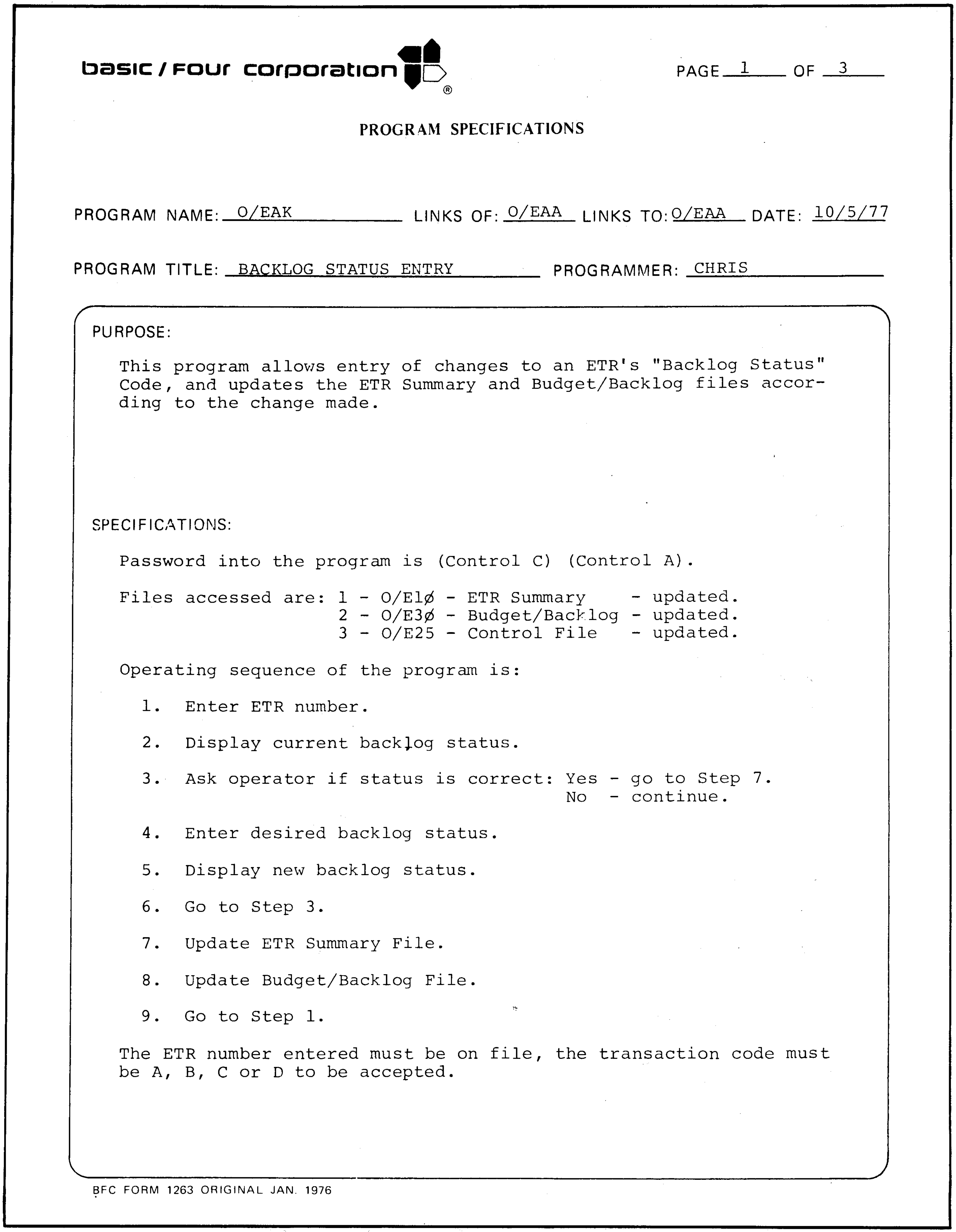
SYSTEM ____________________

	FILES										PROGRAMS																									
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0																																				
1																																				
2																																				
3																																				
4																																				
5																																				
6																																				
7																																				
8																																				
9																																				
A																																				
B																																				
C																																				
D																																				
E																																				
F																																				
G																																				
H																																				
I																																				
J																																				
K																																				
L																																				
M																																				
N																																				
O																																				
P																																				
Q																																				
R																																				
S																																				
T																																				
U																																				
V																																				
W																																				
X																																				
Y																																				
Z																																				

BFC FORM 1262 ORIGINAL JANUARY 1976

BASIC/FOUR CORPORATION

Figure 4-9. File/Program Matrix (BFC 1262)



basic / four corporation

PAGE 1 OF 3

PROGRAM SPECIFICATIONS

PROGRAM NAME: O/EAK LINKS OF: O/EAA LINKS TO: O/EAA DATE: 10/5/77

PROGRAM TITLE: BACKLOG STATUS ENTRY PROGRAMMER: CHRIS

PURPOSE:

This program allows entry of changes to an ETR's "Backlog Status" Code, and updates the ETR Summary and Budget/Backlog files according to the change made.

SPECIFICATIONS:

Password into the program is (Control C) (Control A).

```
Files accessed are: 1 - O/E1Ø - ETR Summary     - updated.
                    2 - O/E3Ø - Budget/Backlog  - updated.
                    3 - O/E25 - Control File    - updated.
```

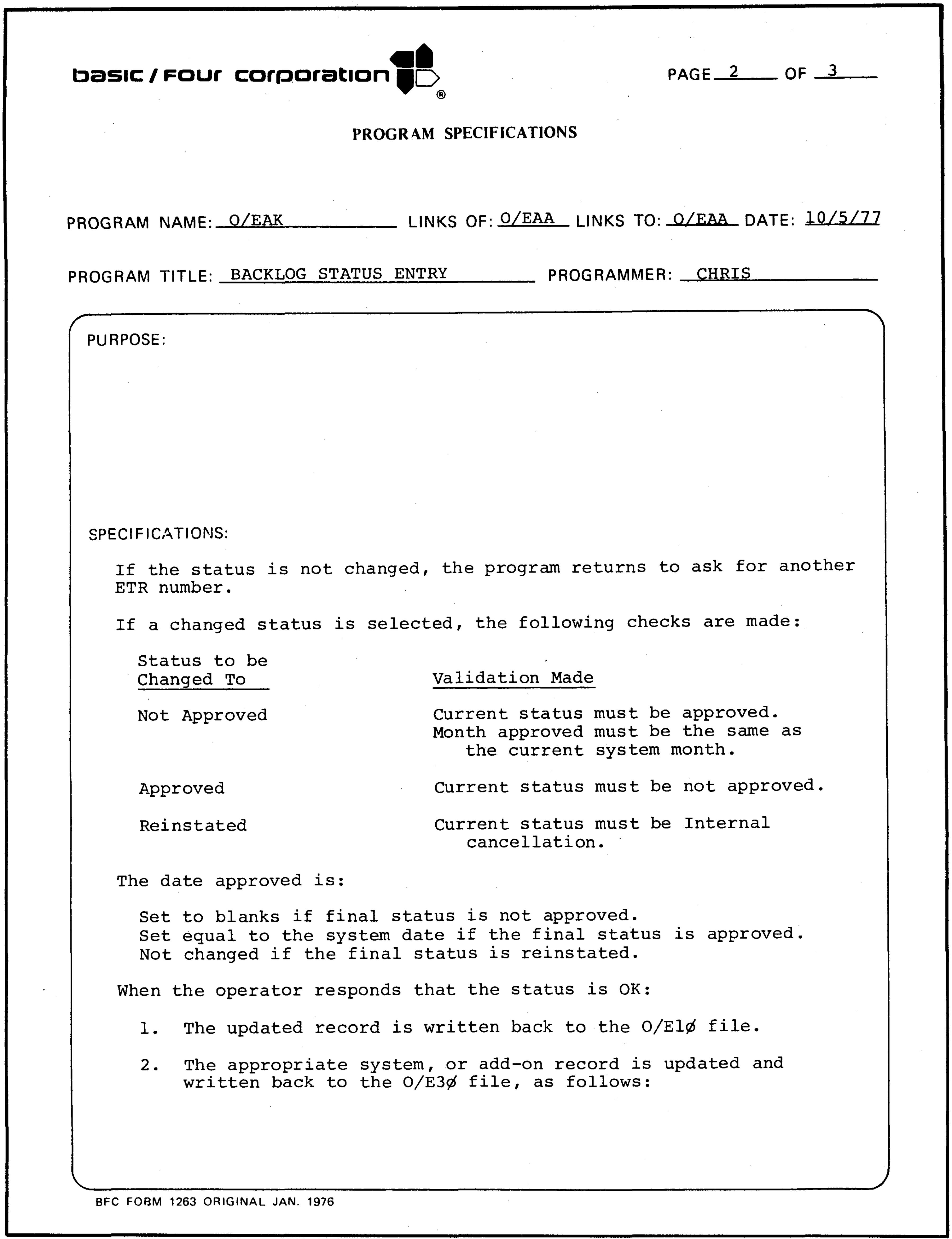
Operating sequence of the program is:

1. Enter ETR number.
2. Display current backlog status.
3. Ask operator if status is correct: Yes - go to Step 7.
No - continue.
4. Enter desired backlog status.
5. Display new backlog status.
6. Go to Step 3.
7. Update ETR Summary File.
8. Update Budget/Backlog File.
9. Go to Step 1.

The ETR number entered must be on file, the transaction code must be A, B, C or D to be accepted.

BFC FORM 1263 ORIGINAL JAN. 1976

Figure 4-10. Program Specifications Form (BFC 1263) (1 of 3)



basic / four corporation®

PAGE 2 OF 3

PROGRAM SPECIFICATIONS

PROGRAM NAME: O/EAK LINKS OF: O/EAA LINKS TO: O/EAA DATE: 10/5/77

PROGRAM TITLE: BACKLOG STATUS ENTRY PROGRAMMER: CHRIS

PURPOSE:

SPECIFICATIONS:

If the status is not changed, the program returns to ask for another ETR number.

If a changed status is selected, the following checks are made:

Status to be Changed To	Validation Made
Not Approved	Current status must be approved. Month approved must be the same as the current system month.
Approved	Current status must be not approved.
Reinstated	Current status must be Internal cancellation.

The date approved is:

Set to blanks if final status is not approved.
Set equal to the system date if the final status is approved.
Not changed if the final status is reinstated.

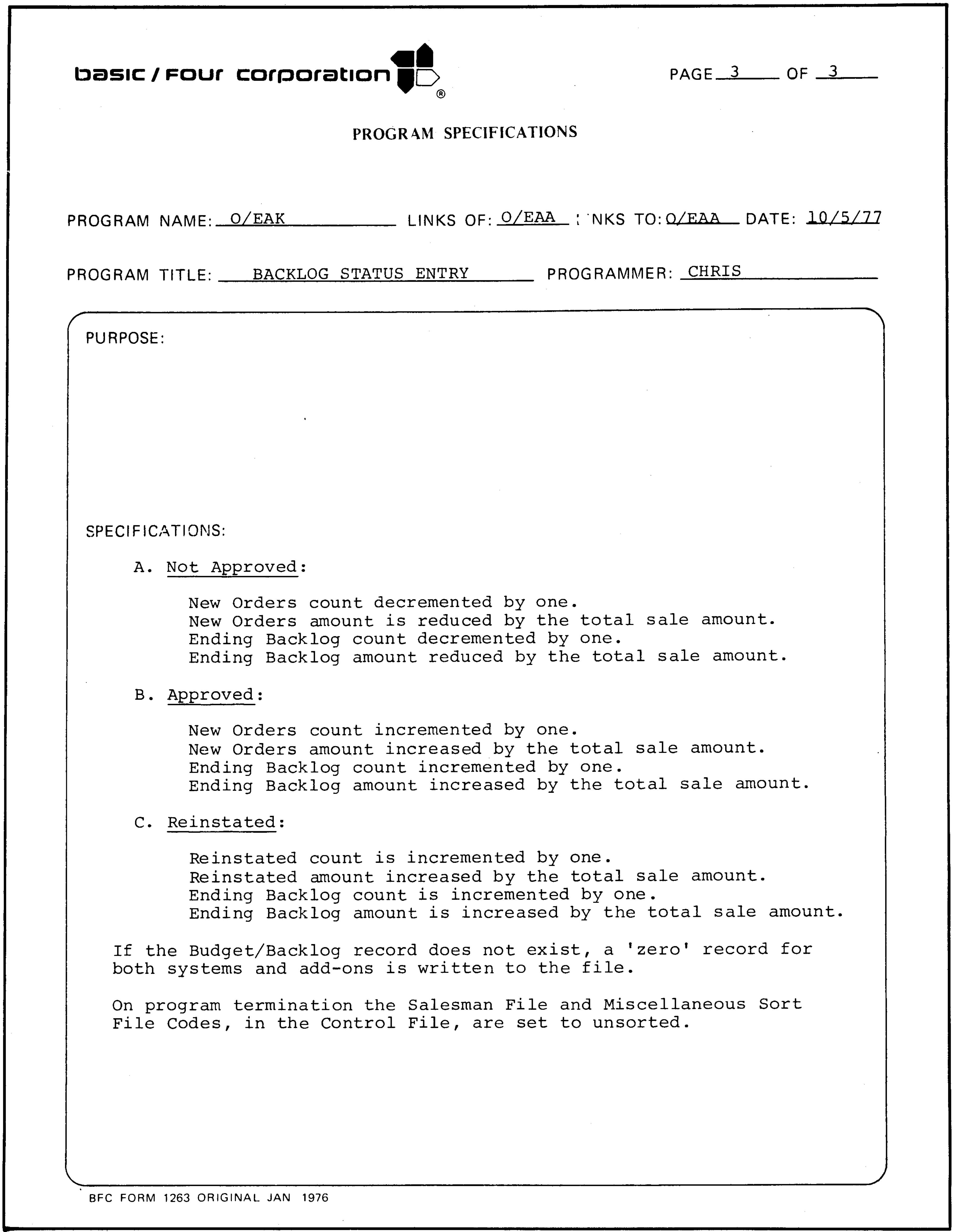
When the operator responds that the status is OK:

1. The updated record is written back to the O/E1Ø file.

2. The appropriate system, or add-on record is updated and written back to the O/E3Ø file, as follows:

BFC FORM 1263 ORIGINAL JAN. 1976

Figure 4-10. Program Specifications Form (BFC 1263) (2 of 3)



basic / four corporation®

PAGE 3 OF 3

PROGRAM SPECIFICATIONS

PROGRAM NAME: O/EAK LINKS OF: O/EAA LINKS TO: O/EAA DATE: 10/5/77

PROGRAM TITLE: BACKLOG STATUS ENTRY PROGRAMMER: CHRIS

PURPOSE:

SPECIFICATIONS:

A. Not Approved:

New Orders count decremented by one.
New Orders amount is reduced by the total sale amount.
Ending Backlog count decremented by one.
Ending Backlog amount reduced by the total sale amount.

B. Approved:

New Orders count incremented by one.
New Orders amount increased by the total sale amount.
Ending Backlog count incremented by one.
Ending Backlog amount increased by the total sale amount.

C. Reinstated:

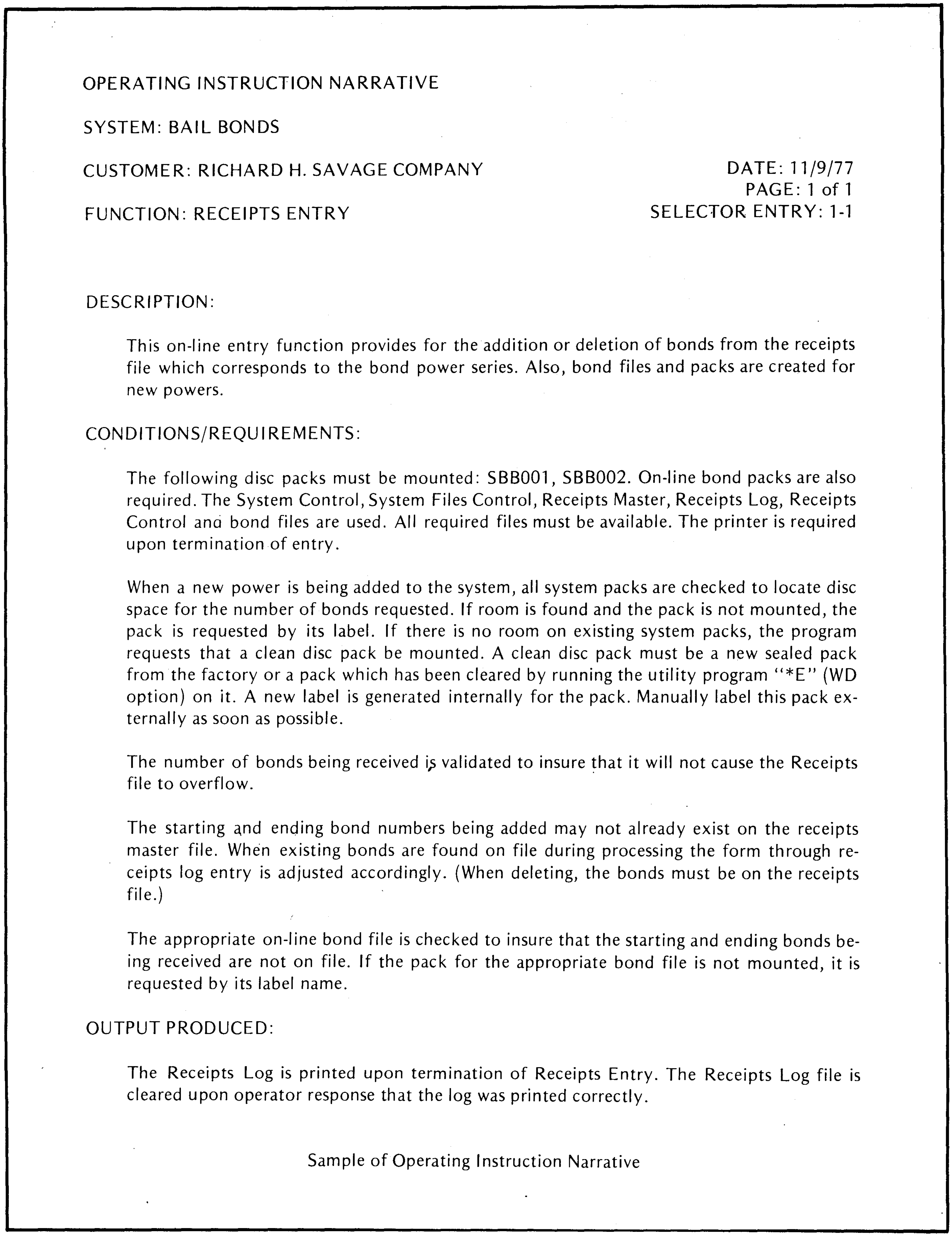
Reinstated count is incremented by one.
Reinstated amount increased by the total sale amount.
Ending Backlog count is incremented by one.
Ending Backlog amount is increased by the total sale amount.

If the Budget/Backlog record does not exist, a 'zero' record for both systems and add-ons is written to the file.

On program termination the Salesman File and Miscellaneous Sort File Codes, in the Control File, are set to unsorted.

BFC FORM 1263 ORIGINAL JAN 1976

Figure 4-10. Program Specifications Form (BFC 1263) (3 of 3)



OPERATING INSTRUCTION NARRATIVE

SYSTEM: BAIL BONDS

CUSTOMER: RICHARD H. SAVAGE COMPANY

FUNCTION: RECEIPTS ENTRY

DATE: 11/9/77
PAGE: 1 of 1
SELECTOR ENTRY: 1-1

DESCRIPTION:

This on-line entry function provides for the addition or deletion of bonds from the receipts file which corresponds to the bond power series. Also, bond files and packs are created for new powers.

CONDITIONS/REQUIREMENTS:

The following disc packs must be mounted: SBB001, SBB002. On-line bond packs are also required. The System Control, System Files Control, Receipts Master, Receipts Log, Receipts Control and bond files are used. All required files must be available. The printer is required upon termination of entry.

When a new power is being added to the system, all system packs are checked to locate disc space for the number of bonds requested. If room is found and the pack is not mounted, the pack is requested by its label. If there is no room on existing system packs, the program requests that a clean disc pack be mounted. A clean disc pack must be a new sealed pack from the factory or a pack which has been cleared by running the utility program "*E" (WD option) on it. A new label is generated internally for the pack. Manually label this pack externally as soon as possible.

The number of bonds being received is validated to insure that it will not cause the Receipts file to overflow.

The starting and ending bond numbers being added may not already exist on the receipts master file. When existing bonds are found on file during processing the form through receipts log entry is adjusted accordingly. (When deleting, the bonds must be on the receipts file.)

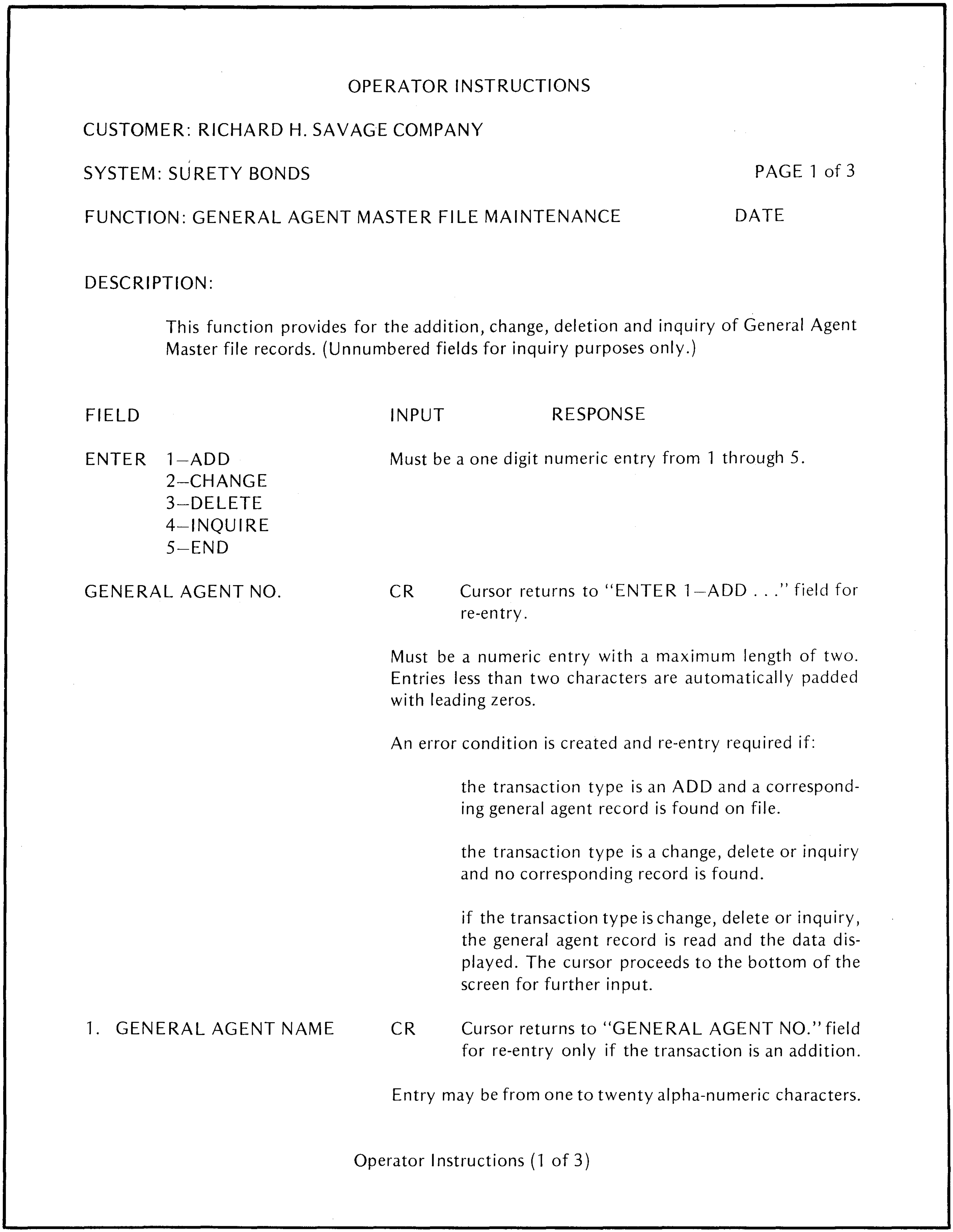
The appropriate on-line bond file is checked to insure that the starting and ending bonds being received are not on file. If the pack for the appropriate bond file is not mounted, it is requested by its label name.

OUTPUT PRODUCED:

The Receipts Log is printed upon termination of Receipts Entry. The Receipts Log file is cleared upon operator response that the log was printed correctly.

Sample of Operating Instruction Narrative

Figure 4-11. Sample of Operating Instruction Narrative



OPERATOR INSTRUCTIONS

CUSTOMER: RICHARD H. SAVAGE COMPANY

SYSTEM: SURETY BONDS PAGE 1 of 3

FUNCTION: GENERAL AGENT MASTER FILE MAINTENANCE DATE

DESCRIPTION:

This function provides for the addition, change, deletion and inquiry of General Agent Master file records. (Unnumbered fields for inquiry purposes only.)

FIELD INPUT RESPONSE

ENTER 1–ADD
2–CHANGE
3–DELETE
4–INQUIRE
5–END

Must be a one digit numeric entry from 1 through 5.

GENERAL AGENT NO.

CR Cursor returns to "ENTER 1–ADD . . ." field for re-entry.

Must be a numeric entry with a maximum length of two. Entries less than two characters are automatically padded with leading zeros.

An error condition is created and re-entry required if:

the transaction type is an ADD and a corresponding general agent record is found on file.

the transaction type is a change, delete or inquiry and no corresponding record is found.

if the transaction type is change, delete or inquiry, the general agent record is read and the data displayed. The cursor proceeds to the bottom of the screen for further input.

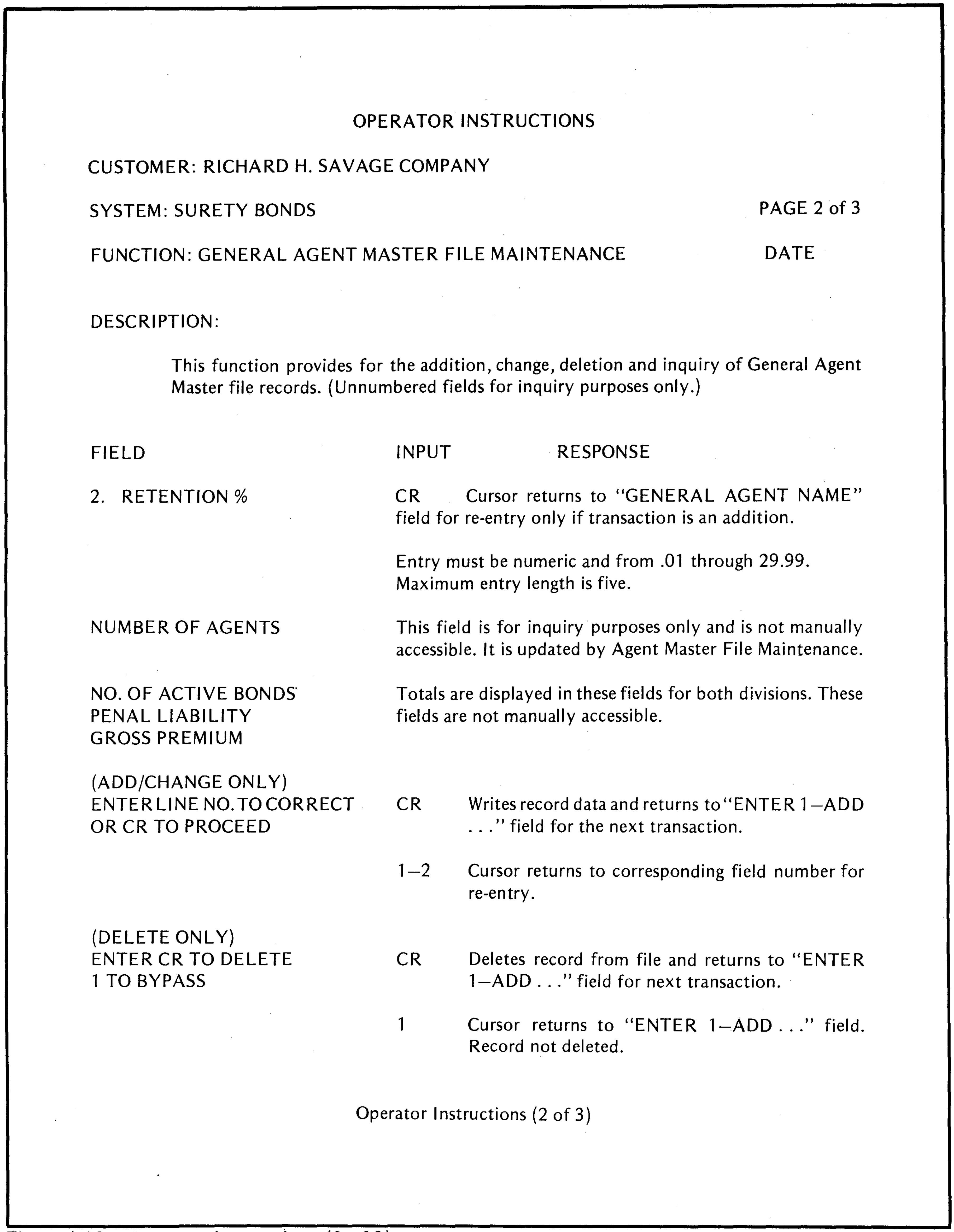
1. GENERAL AGENT NAME

CR Cursor returns to "GENERAL AGENT NO." field for re-entry only if the transaction is an addition.

Entry may be from one to twenty alpha-numeric characters.

Operator Instructions (1 of 3)

Figure 4-12. Operator Instructions (1 of 3)



OPERATOR INSTRUCTIONS

CUSTOMER: RICHARD H. SAVAGE COMPANY

SYSTEM: SURETY BONDS PAGE 2 of 3

FUNCTION: GENERAL AGENT MASTER FILE MAINTENANCE DATE

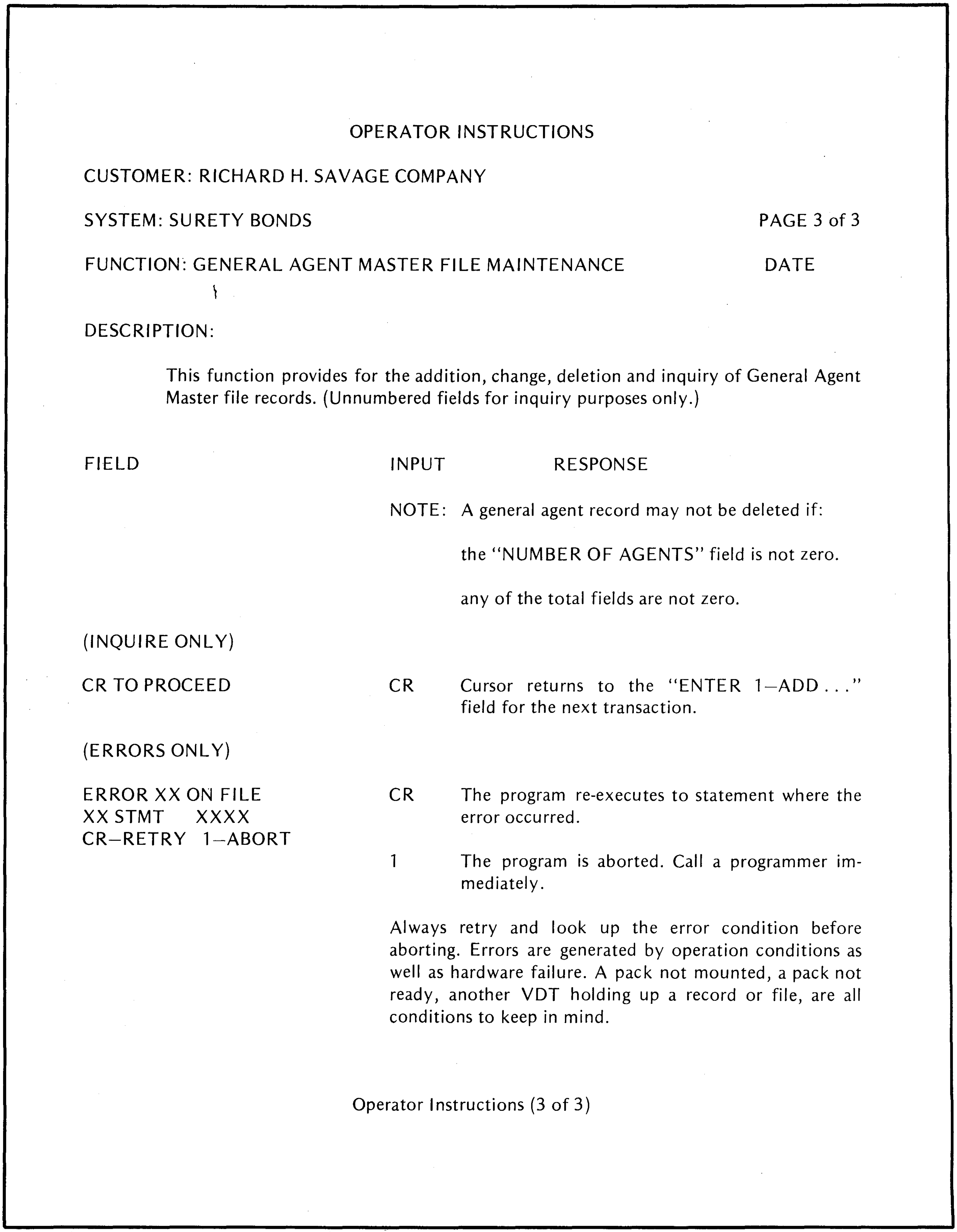
DESCRIPTION:

This function provides for the addition, change, deletion and inquiry of General Agent Master file records. (Unnumbered fields for inquiry purposes only.)

FIELD	INPUT	RESPONSE
2. RETENTION %	CR	Cursor returns to "GENERAL AGENT NAME" field for re-entry only if transaction is an addition.
		Entry must be numeric and from .01 through 29.99. Maximum entry length is five.
NUMBER OF AGENTS		This field is for inquiry purposes only and is not manually accessible. It is updated by Agent Master File Maintenance.
NO. OF ACTIVE BONDS PENAL LIABILITY GROSS PREMIUM		Totals are displayed in these fields for both divisions. These fields are not manually accessible.
(ADD/CHANGE ONLY) ENTER LINE NO. TO CORRECT OR CR TO PROCEED	CR	Writes record data and returns to "ENTER 1–ADD . . ." field for the next transaction.
	1–2	Cursor returns to corresponding field number for re-entry.
(DELETE ONLY) ENTER CR TO DELETE 1 TO BYPASS	CR	Deletes record from file and returns to "ENTER 1–ADD . . ." field for next transaction.
	1	Cursor returns to "ENTER 1–ADD . . ." field. Record not deleted.

Operator Instructions (2 of 3)

Figure 4-12. Operator Instructions (2 of 3)



OPERATOR INSTRUCTIONS

CUSTOMER: RICHARD H. SAVAGE COMPANY

SYSTEM: SURETY BONDS PAGE 3 of 3

FUNCTION: GENERAL AGENT MASTER FILE MAINTENANCE DATE

DESCRIPTION:

This function provides for the addition, change, deletion and inquiry of General Agent Master file records. (Unnumbered fields for inquiry purposes only.)

FIELD	INPUT	RESPONSE
	NOTE:	A general agent record may not be deleted if: the "NUMBER OF AGENTS" field is not zero. any of the total fields are not zero.
(INQUIRE ONLY)		
CR TO PROCEED	CR	Cursor returns to the "ENTER 1–ADD . . ." field for the next transaction.
(ERRORS ONLY)		
ERROR XX ON FILE XX STMT XXXX CR–RETRY 1–ABORT	CR	The program re-executes to statement where the error occurred.
	1	The program is aborted. Call a programmer immediately.

Always retry and look up the error condition before aborting. Errors are generated by operation conditions as well as hardware failure. A pack not mounted, a pack not ready, another VDT holding up a record or file, are all conditions to keep in mind.

Operator Instructions (3 of 3)

Figure 4-12. Operator Instructions (3 of 3)

APPENDIX I – SAMPLE WORK STATEMENT

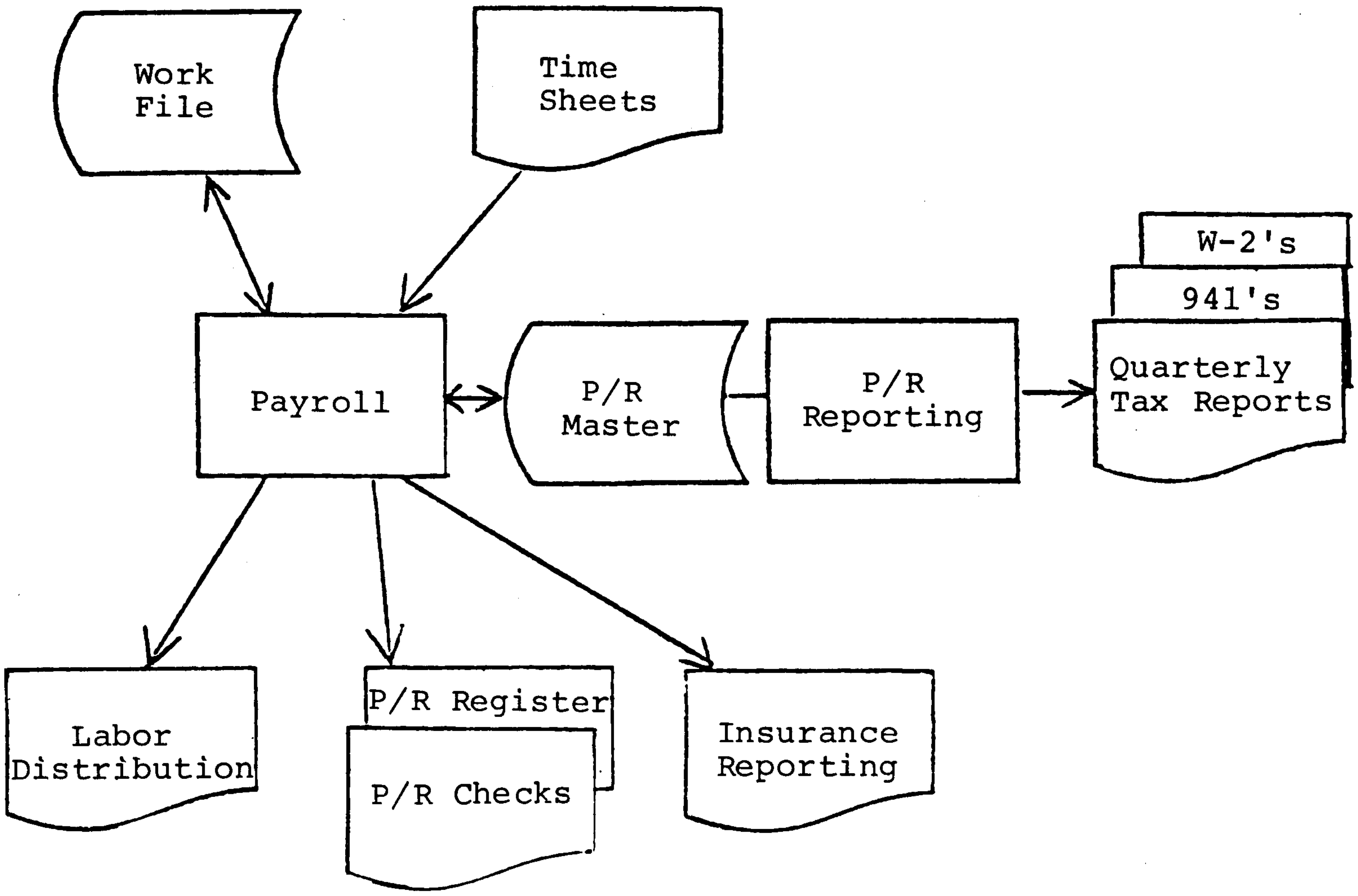
SECTION I - APPLICATIONS

CUSTOMER NAME ______________

DATE ______________

A. PAYROLL

1. System Flow



2. System Description

Your payroll system will process time sheets against a payroll master disc file on a weekly cycle. The payroll master will be updated on-line and the checks and register will be printed from the work file. Your system will also include labor distribution and insurance reporting as part of the weekly processing cycle. Quarterly, the payroll master will be processed to print the 941 and quarterly tax reports. Annually, the W-2's will be produced by the system.

Figure AI-1. Sample Work Statement (1 of 7)

CUSTOMER NAME ______________

DATE ______________

3. Input Documents

The source documents for your payroll system will be the following:

1. Hourly payroll time sheets
2. Salary payroll time sheets
3. Deductions
4. New employees, terminations, address changes

4. Output Documents

Your payroll system will produce the following documents:

1. Payroll Checks: weekly and semi-monthly
2. Payroll Register: weekly and semi-monthly
3. Payroll Check Register: weekly and semi-monthly
4. Insurance premium, union, pension reports: monthly
5. 941's and quarterly tax report, FDI, FUI, FUT, FICA, Federal Withholding Tax, State Withholding Tax
6. W-2's

NOTE: It is imperative that the vendor obtain a clear understanding of prospect's unusual payroll requirements (e.g., labor distribution, workman's compensation, union report, incentive considerations, multi-state, multi-company, and so on).

B. ORDER ENTRY

1. System Flow

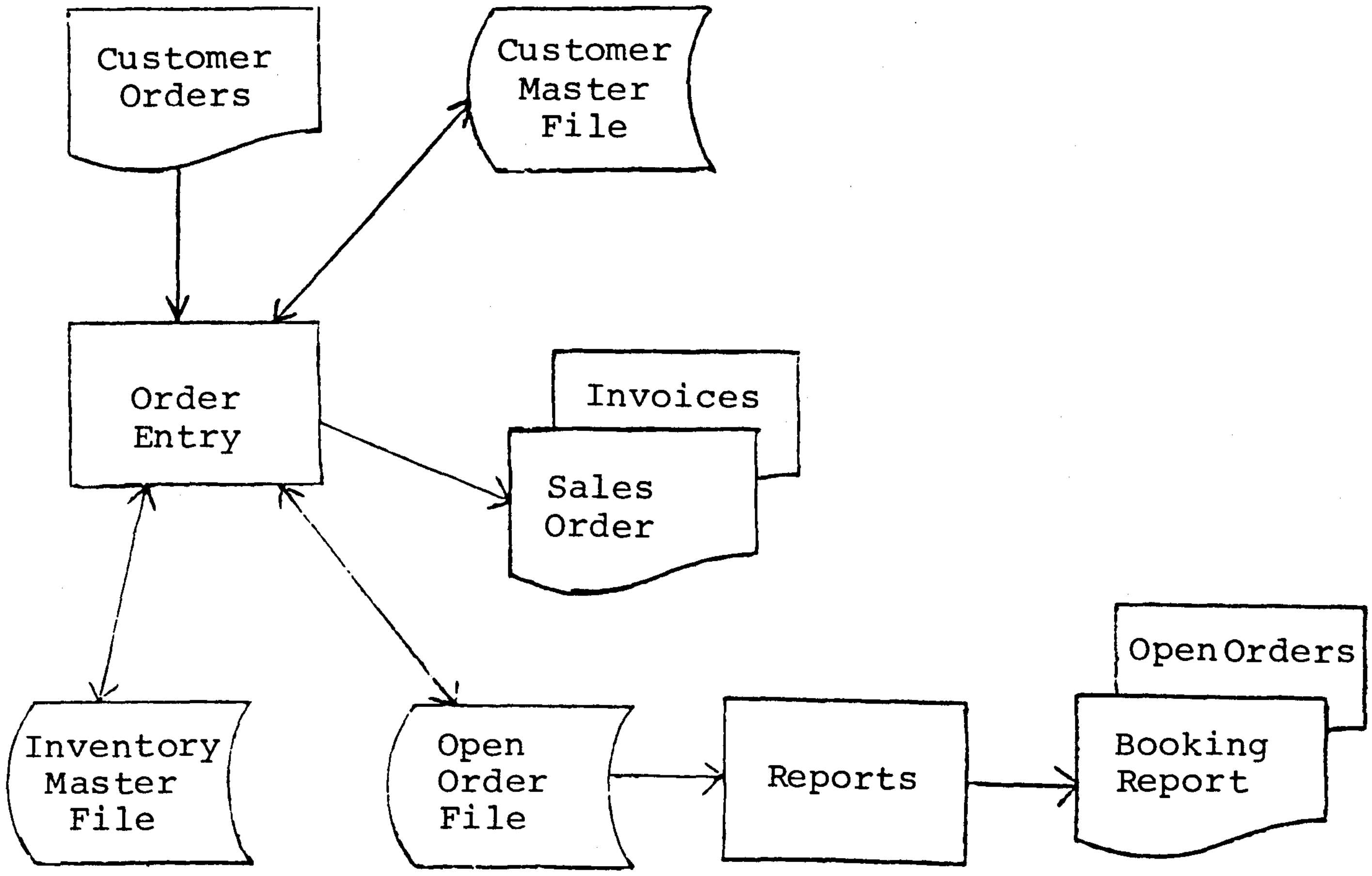
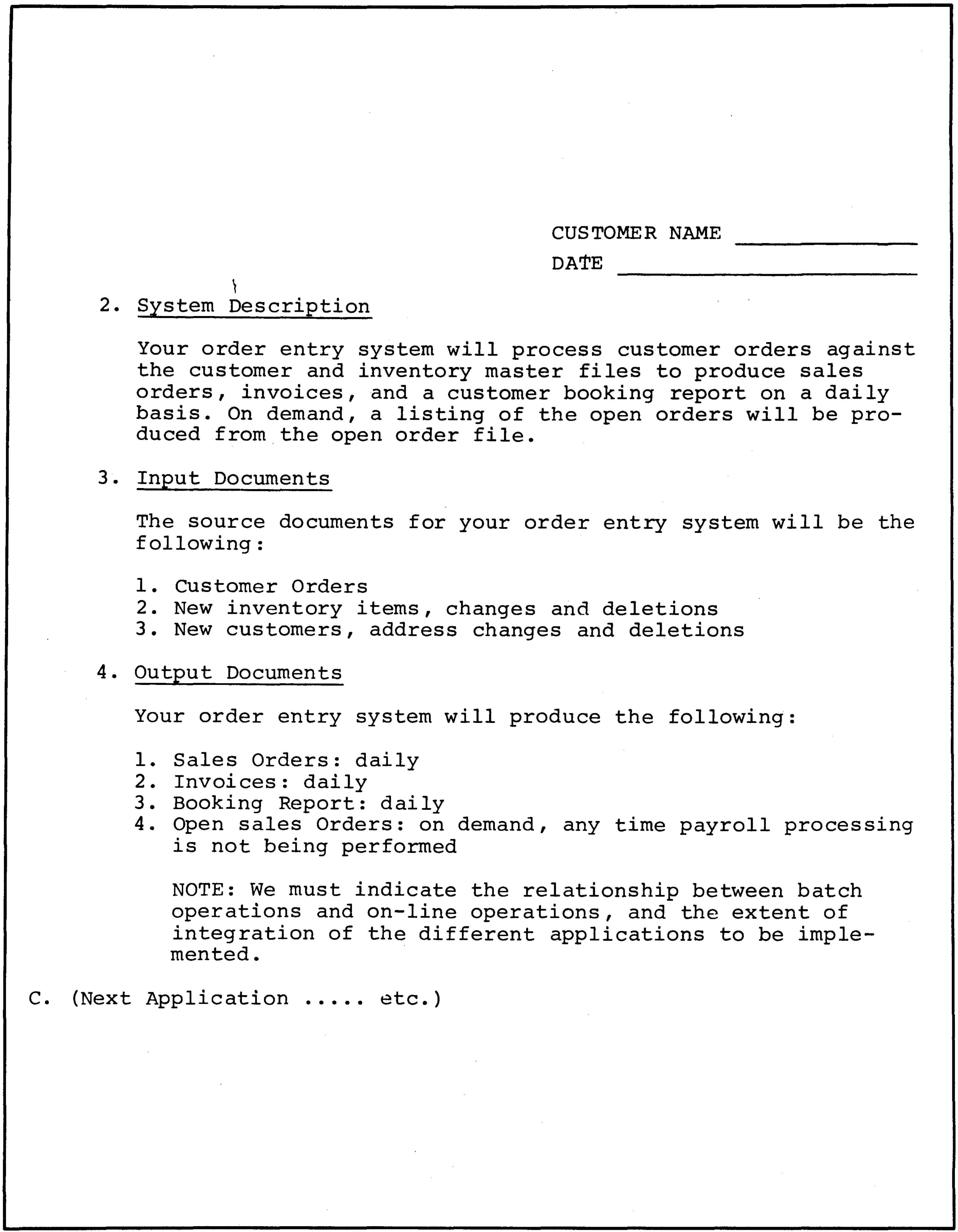


Figure A1-1. Sample Work Statement (2 of 7)



CUSTOMER NAME ______________

DATE ______________________

2. System Description

Your order entry system will process customer orders against the customer and inventory master files to produce sales orders, invoices, and a customer booking report on a daily basis. On demand, a listing of the open orders will be produced from the open order file.

3. Input Documents

The source documents for your order entry system will be the following:

1. Customer Orders
2. New inventory items, changes and deletions
3. New customers, address changes and deletions

4. Output Documents

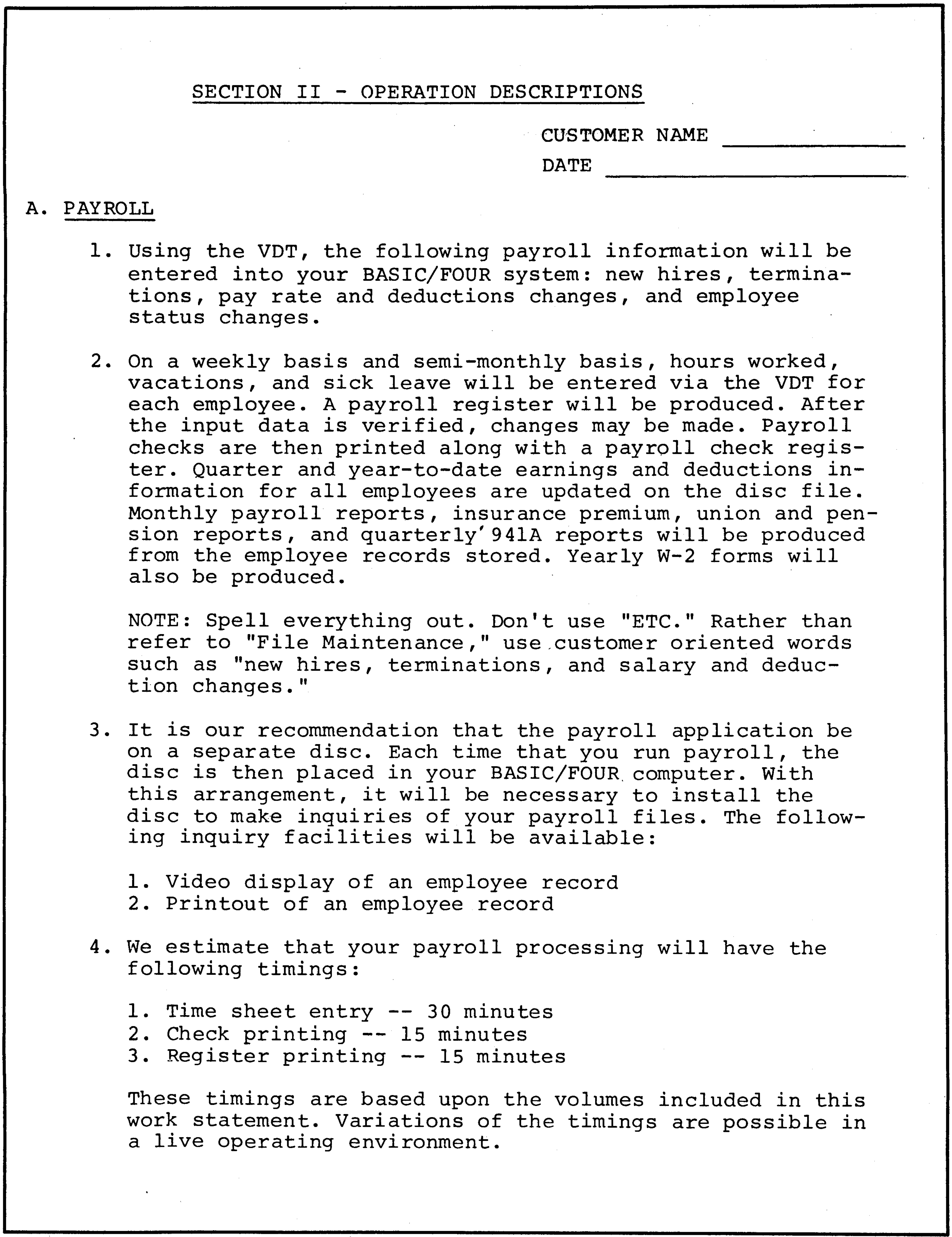
Your order entry system will produce the following:

1. Sales Orders: daily
2. Invoices: daily
3. Booking Report: daily
4. Open sales Orders: on demand, any time payroll processing is not being performed

NOTE: We must indicate the relationship between batch operations and on-line operations, and the extent of integration of the different applications to be implemented.

C. (Next Application ..... etc.)

Figure AI-1. Sample Work Statement (3 of 7)



SECTION II - OPERATION DESCRIPTIONS

CUSTOMER NAME ______________

DATE ______________________

A. PAYROLL

1. Using the VDT, the following payroll information will be entered into your BASIC/FOUR system: new hires, terminations, pay rate and deductions changes, and employee status changes.

2. On a weekly basis and semi-monthly basis, hours worked, vacations, and sick leave will be entered via the VDT for each employee. A payroll register will be produced. After the input data is verified, changes may be made. Payroll checks are then printed along with a payroll check register. Quarter and year-to-date earnings and deductions information for all employees are updated on the disc file. Monthly payroll reports, insurance premium, union and pension reports, and quarterly' 941A reports will be produced from the employee records stored. Yearly W-2 forms will also be produced.

   NOTE: Spell everything out. Don't use "ETC." Rather than refer to "File Maintenance," use customer oriented words such as "new hires, terminations, and salary and deduction changes."

3. It is our recommendation that the payroll application be on a separate disc. Each time that you run payroll, the disc is then placed in your BASIC/FOUR computer. With this arrangement, it will be necessary to install the disc to make inquiries of your payroll files. The following inquiry facilities will be available:

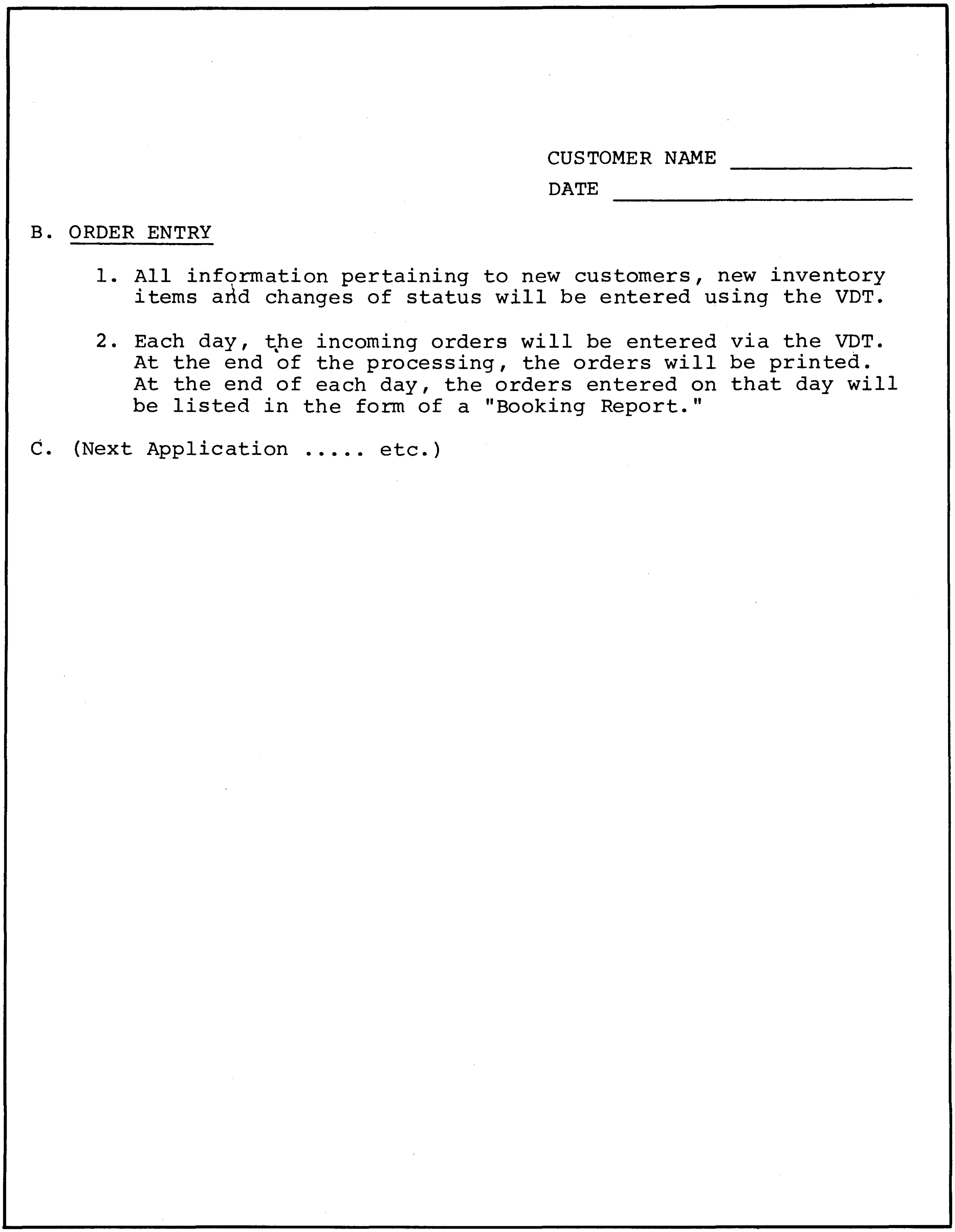
   1. Video display of an employee record
   2. Printout of an employee record

4. We estimate that your payroll processing will have the following timings:

   1. Time sheet entry -- 30 minutes
   2. Check printing -- 15 minutes
   3. Register printing -- 15 minutes

   These timings are based upon the volumes included in this work statement. Variations of the timings are possible in a live operating environment.

Figure AI-1. Sample Work Statement (4 of 7)



CUSTOMER NAME ______________

DATE ______________________

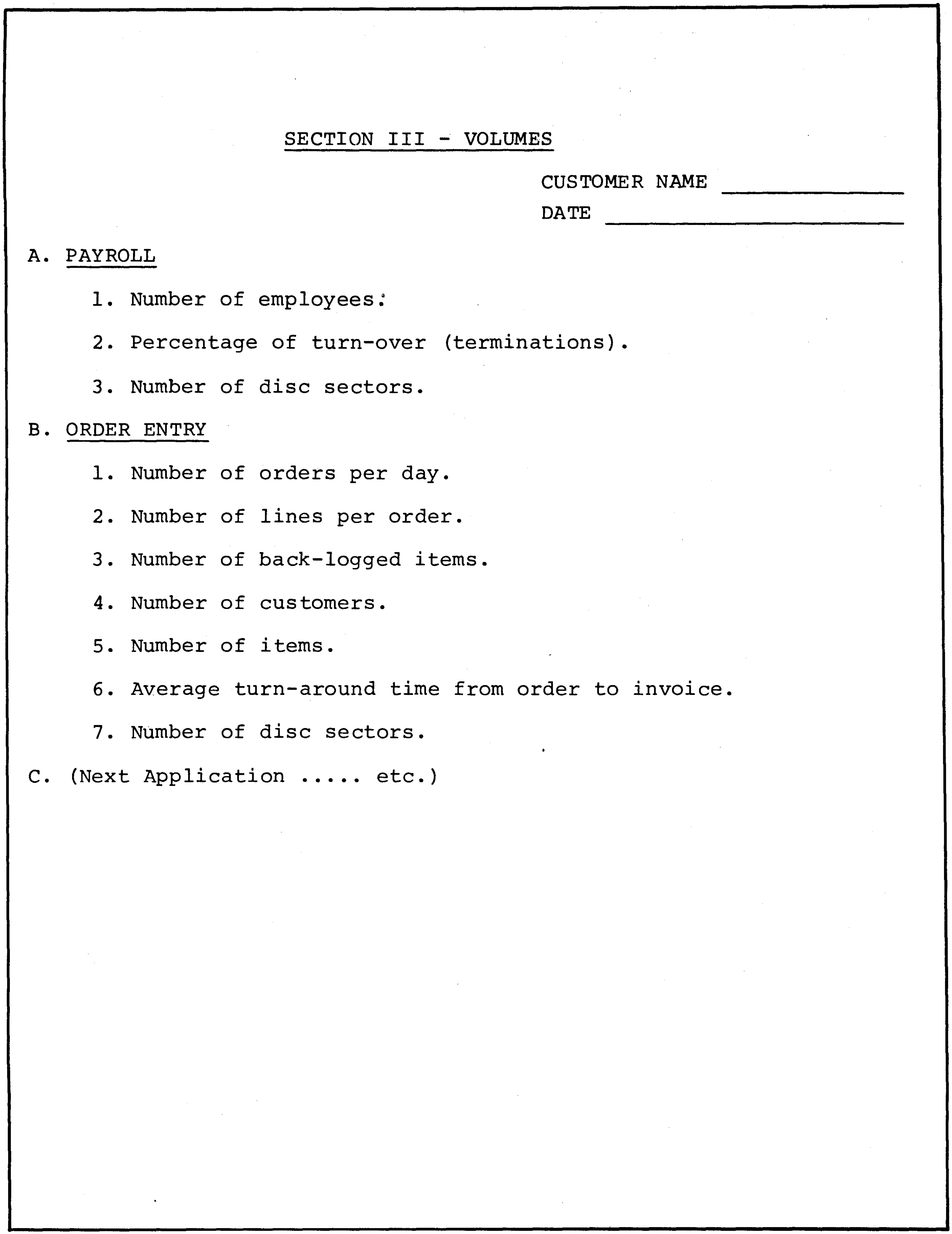
B. ORDER ENTRY

1. All information pertaining to new customers, new inventory items and changes of status will be entered using the VDT.

2. Each day, the incoming orders will be entered via the VDT. At the end of the processing, the orders will be printed. At the end of each day, the orders entered on that day will be listed in the form of a "Booking Report."

C. (Next Application ..... etc.)

Figure AI-1. Sample Work Statement (5 of 7)



SECTION III - VOLUMES

CUSTOMER NAME ______________

DATE ______________________

A. PAYROLL

1. Number of employees.

2. Percentage of turn-over (terminations).

3. Number of disc sectors.

B. ORDER ENTRY

1. Number of orders per day.

2. Number of lines per order.

3. Number of back-logged items.

4. Number of customers.

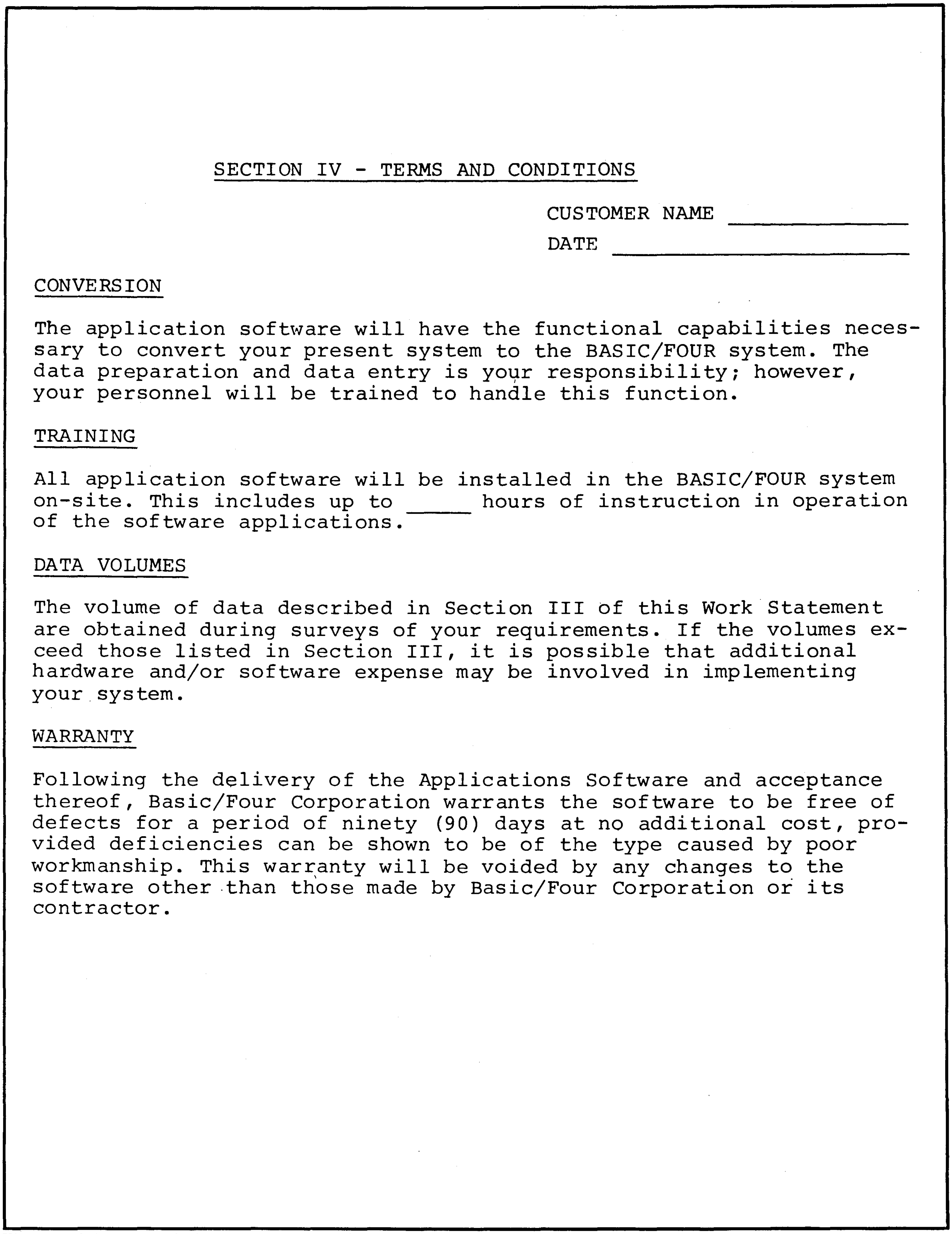
5. Number of items.

6. Average turn-around time from order to invoice.

7. Number of disc sectors.

C. (Next Application ..... etc.)

Figure AI-1. Sample Work Statement (6 of 7)



SECTION IV - TERMS AND CONDITIONS

CUSTOMER NAME ______________

DATE ______________________

CONVERSION

The application software will have the functional capabilities necessary to convert your present system to the BASIC/FOUR system. The data preparation and data entry is your responsibility; however, your personnel will be trained to handle this function.

TRAINING

All application software will be installed in the BASIC/FOUR system on-site. This includes up to _____ hours of instruction in operation of the software applications.

DATA VOLUMES

The volume of data described in Section III of this Work Statement are obtained during surveys of your requirements. If the volumes exceed those listed in Section III, it is possible that additional hardware and/or software expense may be involved in implementing your system.

WARRANTY

Following the delivery of the Applications Software and acceptance thereof, Basic/Four Corporation warrants the software to be free of defects for a period of ninety (90) days at no additional cost, provided deficiencies can be shown to be of the type caused by poor workmanship. This warranty will be voided by any changes to the software other than those made by Basic/Four Corporation or its contractor.

Figure AI-1. Sample Work Statement (7 of 7)

APPLICATIONS SOFTWARE DEVELOPMENT SCHEDULE

DEVELOPMENT FUNCTIONS	DEF AND ANALYSIS	PRELIMINARY DESIGN	DETAIL DESIGN	SOFTWARE CONSTRUCTION	SOFTWARE CERTIFICATION	INSTALL	MAINTENANCE
PRODUCTION ACTIVITIES	DEF AND ANALYSIS						
	PRELIMINARY DESIGN						
		FUNCTIONAL DESIGN					
			PROGRAM DESIGN				
				CODING			
			FUNCTIONAL TESTING PROGRAM DESIGN TESTING AND ANALYSIS	UNIT INTEGRATION AND TESTING			
	DEVELOP CERTIFICATION PLAN			DEVELOP TEST DATA	CERTIFICATION TESTING SITE TEST		
	DEVELOP CONVERSION/INSTALLATION PLAN					INSTALL	
	DEVELOP TRAINING PLAN			TRAINING PREPARATION AND TRAINING			
MAJOR DOCUMENTATION	*RQMTS DEF *RESPONSE TO RQMTS DEF	* PRELIM DESIGN	* FUNCTIONAL SPECIFICATIONS * PROGRAM SPECIFICATIONS * TEST PLAN * INSTALLATION PLAN * USER DOCUMENTATION	* PROGRAM INFO * TEST CASES * FINAL USER DOCUMENTATION * OPERATOR INSTRUCTIONS	* PROGRAMS CERTIFICATION SIGN OFF * FINAL INSTALLATION PLAN	* INSTALL REPORT	* DOCUMENTATION REVIEW AND UPDATE
ESTIMATING AND PROJECT PLANNING	*PRELIM DESIGN DETAIL ESTIMATE *PROJECT ESTIMATE	* DETAIL DESIGN DETAIL ESTIMATE * PROJECT ESTIMATE	* DETAILED PROJECT ESTIMATE				* ESTIMATES FOR CHANGES TO PROGRAMS AND DOCUMENTATION
REVIEWS	DEF AND ANALYSIS	PRELIM DESIGN	CRITICAL DESIGN	SOFTWARE CONSTRUCTION	CERTIFICATION	INSTALL	PERIODIC MAINT REVIEWS
				△ SIGN OFF		△ SIGN OFF	△ SIGN OFF

Figure AI-2. Applications Software Development Schedule

APPENDIX II – DESIGN STANDARDS

DESIGN STANDARDS

System design involves the definition and documentation of the requirements of a processing system. In order for the system to take full advantage of the hardware and software available, the analyst must design the system with the hardware and language capabilities in mind.

A system should be efficient as well as functional in order to be truly successful. The analyst should implement design techniques which capitalize on the available file types, language capabilities, and processing types, in order to fulfill the system requirements in the most efficient manner possible.

The design standards listed in this section are recommended rules to follow during the system design development phase. Each rule is accompanied by a reason or explanation. There may also be a narrative or coded example which represents a possible method for implementating the rule.

STANDARD RULE:

Design control feature for related application functions in order to insure accuracy and prevent operator error.

REASON:

Specific processing sequence is often required especially when using work files. These situations must be controlled so that they may not be run incorrectly.

METHOD EXAMPLES:

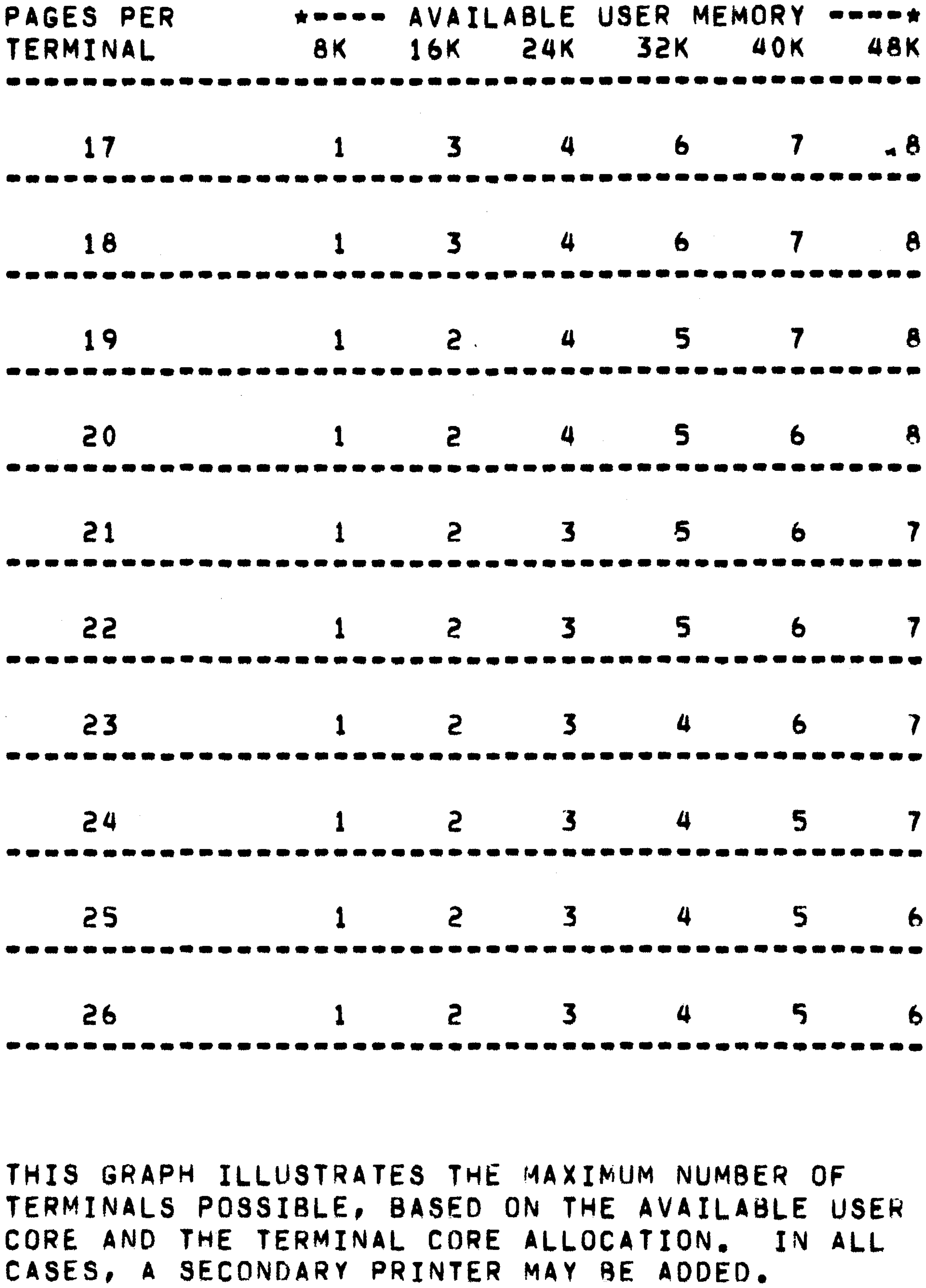
This may be done through the use of status flags, on a control file, which are constantly checked and updated to reflect the various stages of completion.

STANDARD RULE:

All program functions must be capable of operating in a maximum of 26 pages of memory (6656 bytes). The recommended breakdown for application programs is 18 pages (18 sectors, 4608 bytes) which leaves a proportionate 8 pages of memory (2048 bytes) for data area.

REASON:

Controlling the amount of user memory which is available for each task provides for efficient memory usage and encourages uniformity in program design. Since BBII allocates the available data area for each task, limiting the program size is a method to insure that sufficient data area will be available for processing.



```
PAGES PER       *---- AVAILABLE USER MEMORY ----*
TERMINAL          8K   16K   24K   32K   40K   48K
--------------------------------------------------

    17            1     3     4     6     7     8
--------------------------------------------------

    18            1     3     4     6     7     8
--------------------------------------------------

    19            1     2     4     5     7     8
--------------------------------------------------

    20            1     2     4     5     6     8
--------------------------------------------------

    21            1     2     3     5     6     7
--------------------------------------------------

    22            1     2     3     5     6     7
--------------------------------------------------

    23            1     2     3     4     6     7
--------------------------------------------------

    24            1     2     3     4     5     7
--------------------------------------------------

    25            1     2     3     4     5     6
--------------------------------------------------

    26            1     2     3     4     5     6
--------------------------------------------------


THIS GRAPH ILLUSTRATES THE MAXIMUM NUMBER OF
TERMINALS POSSIBLE, BASED ON THE AVAILABLE USER
CORE AND THE TERMINAL CORE ALLOCATION.  IN ALL
CASES, A SECONDARY PRINTER MAY BE ADDED.
```

Figure AII-1. Terminal Availability Graph

STANDARD RULE:

It is recommended that record size be defined as powers of two (i.e.: 2, 4, 8, 16, 32, 64, 128, 256, 512).

REASON:

Record sizes other than powers of two, adversely affect system operating efficiency. This is because additional disc rotations are required to read records which overlap sector boundaries.

STANDARD RULE:

When possible record sizes should be limited to two (2) sectors, (512 bytes) maximum.

REASON:

When working within a limited data area, records which occupy more than 512 bytes are impractical.

STANDARD RULE:

When related data does not fit on one record, design two separate files instead of multiple record formats.

REASON:

Multiple record types existing in the same file are not compatible with language translator programs or with EASY, the BASIC/FOUR report generator system.

STANDARD RULE:

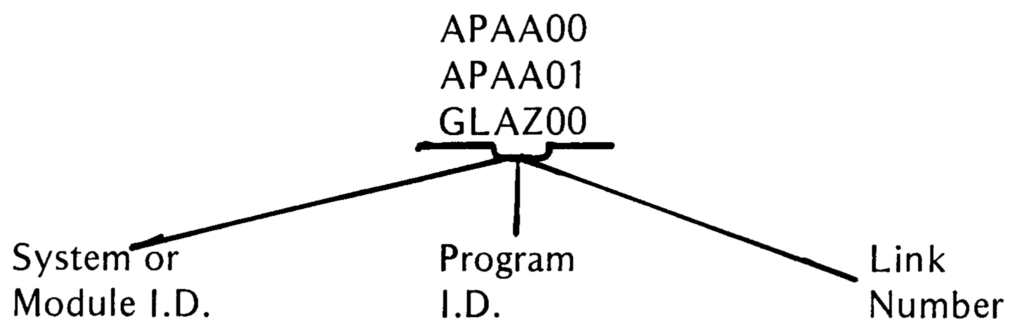
A scheme of file name assignments should be used consistently throughout the system and should be assigned as part of the systems design phase. This provides the ability to easily recognize a file or program.

REASON:

It is very important to be able to identify a program or file especially during the debugging phase.

METHOD EXAMPLES:

It is recommended that a program name include the system or module, program and overlay (link) identification.



APAA00
APAA01
GLAZ00

A data file name should include the system or module identification. The remaining four characters may be used to uniquely identify the file within the system, according to a consistent scheme.

STANDARD RULE:

The data files and program disc requirements should be defined as part of the system design phase. This includes establishing the necessary programs and files on the disc.

REASON:

This function should be completed prior to beginning programming in order to promote efficiency and organization in the coding/debugging phase.

STANDARD RULE:

Always include the key data in the record.

REASON:

In case a file directory is accidently destroyed or the key pointers are in error, a direct file may be easily created using the data records if the necessary key data is in the records.

STANDARD RULE:

Although the system allows variable length keys in DIRECT and SORT files, keys for a given file should be uniform in format and length.

REASON:

When it is necessary to read a DIRECT file by building the key with data from other files, inconsistent key lengths could cause a problem in accessing the desired record.

STANDARD RULE:

The next available sequence number should be stored on ordinal record zero (0) for INDEXED files. When using sequence numbers in DIRECT or SORT files store the next available sequence number in a control file.

REASON:

It is important to maintain strict control of sequence numbers in order to prevent the loss of data due to duplicate sequence numbers.

METHOD EXAMPLES:

When using sequence numbers, EXTRACT the control record, increment the next available sequence number and write back the control record immediately.

STANDARD RULE:

Program coding should always support a multi-user environment.

REASON:

Although an installation may only have one terminal originally, it is always possible that additional equipment may be added later.

METHOD EXAMPLES:

Multi-user capability is accomplished through the use of EXTRACT in maintenance and update functions as well as by using file LOCK in updates or other functions where only single users should be allowed.

STANDARD RULE:

The system time and date functions should have uniform format throughout the system.

REASON:

Although those functions are usually only updated once a day, they appear on all output and should be consistent.

METHOD EXAMPLES:

The time function should be set using four digits, representing twenty-four hours: HH:MM

10:00 12:00 14:00

The time should be output as follows:

10:00AM 12:00AM 1:00PM

The system date should be set and output as MO/DA/YR, and should be validated upon entry.

STANDARD RULE:

It is recommended that working variables used for such things as flags, line counts, page counts, error handling, print masks and printer selection be standard within a system. Standard variable assignments should be documented as part of systems design.

REASON:

Standard usage of working variables promotes consistency within a system even though several programmers may have worked on it. Also, the efficient assignment of data variables decreases the amount of user data required for program operation. Programs which use standard variables are easier to follow for someone, other than the author or designer, who must make modifications.

METHOD EXAMPLES:

A sample variable list is provided on page 63 for reference when studying the sample routines and examples in this manual. A list of such standard variables should be included as documentation for any system.

STANDARD RULE:

Always display the identification of the system and program function being executed. The program or link name may also be displayed.

REASON:

The operator must be kept informed on what is being processed in order to prevent ESCAPE or premature termination of the program. Also, the identification on the screen should agree with system documentation thereby encouraging the use of the documentation available.

STANDARD RULE:

Always provide the ability to return immediately to the selector before executing a program.

REASON:

Operators sometimes make incorrect selections and should be allowed to return to the selector without having to ESCAPE.

STANDARD RULE:

The current system time and date information should appear on all output including reports and screens.

REASON:

The time and date information is of great aide to an operator when filing reports. The ability to quickly determine which report of several is most current is often very important.

STANDARD RULE:

Always consider the device to be used when designing the I/O formats.

REASON:

Various devices have certain limitations which must be considered. For instance, when displaying data on an EDT, only 16 lines with 32 characters each may be displayed.

STANDARD RULE:

Common routines should be developed and used consistently where ever possible.

REASON:

The use of common routines for such things as error handling, data entry and printer selection, promotes programming efficiency and consistency. Common routines need only be written, keyed in and debugged once and then may be stored and merged into memory prior to keying in the rest of a program. (See Common Routines)

STANDARD RULE:

Implement a consistent scheme for the use of the Function Keys (motor bars).

REASON:

The use of the Function Keys increases operator efficiency especially when the 10-key numeric keys are used. When used, each of the Function Keys (I–IV) should have a predictable, pre-determined result.

METHOD EXAMPLES:

The Function Keys may be used effectively in entry and maintenance functions as follows:

CR or I – The new entry is accepted or "no change" occurs during selective field correction of existing data.

II – A numeric entry is automatically converted to a negative number.

III – The data field is set to null.

IV – The cursor returns to the previous data field for re-entry.

APPENDIX III – PROGRAMMING STANDARDS

PROGRAMMING STANDARDS

No two programmers enforce programming techniques in exactly the same manner. Whether or not the same methods are used is not as important as the overall consistency from program to program in a system.

The following list of standard rules are recommended in order to avoid serious problems and promote the development of reliable and consistent applications software. Each rule or goal is accompanied by a reason or explanation. In some cases, coded examples or written explanations of appropriate methods to accomplish the goal are also included.

STANDARD RULE:

Provide for all types of errors and use a common error routine which provides retry capability (especially for I/O activities). It is recommended to display the error code, the file name and the statement number as part of an error retry routine. Errors that are anticipated, such as 11-missing record, should be handled internally, by using the DOM= option.

REASON:

An operator should be allowed to RETRY error conditions without having the screen destroyed. Processing generated error conditions such as 11, should be anticipated by the programmer and code included to handle such a situation internally.

METHOD EXAMPLES:

See COMMON ROUTINES

STANDARD RULE:

Always accompany an ERASE directive with either a DIRECT, INDEXED or SORT directive in the same program.

REASON:

It is important to maintain strict control during file clear operations. This activity should be performed in a straight forward manner which is easy to follow.

If a file is not re-created immediately after being erased, it is possible that the disc space where the file was located could be allocated for another purpose before the proper file is re-defined.

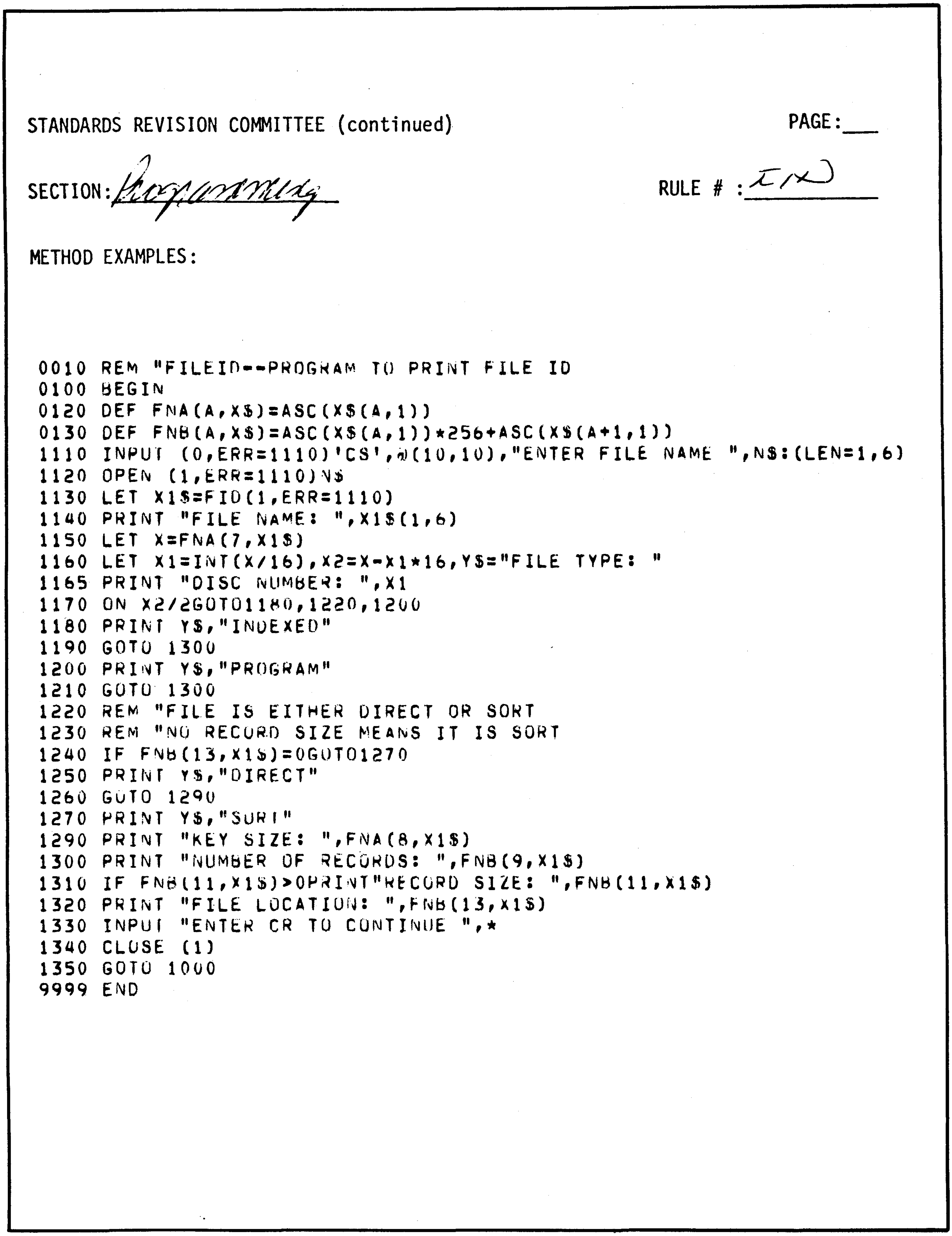
Also, files which are periodically cleared are often used by many programs. If such a file is not immediately re-defined after being erased, the code necessary to re-define the file must be included in all of the program functions which use it.

STANDARD RULE:

Use the FID function to obtain the file information necessary to ERASE and re-establish a file.

REASON:

When the FID function is used in programs, the "hard" coding of file information is eliminated. This makes a system more flexible. If the disc location of a file is changed, no program changes are required.



STANDARDS REVISION COMMITTEE (continued) PAGE:___

SECTION: Programming RULE # : I/X)

METHOD EXAMPLES:

```
0010 REM "FILEID--PROGRAM TO PRINT FILE ID
0100 BEGIN
0120 DEF FNA(A,X$)=ASC(X$(A,1))
0130 DEF FNB(A,X$)=ASC(X$(A,1))*256+ASC(X$(A+1,1))
1110 INPUT (0,ERR=1110)'CS',@(10,10),"ENTER FILE NAME ",N$:(LEN=1,6)
1120 OPEN (1,ERR=1110)N$
1130 LET X1$=FID(1,ERR=1110)
1140 PRINT "FILE NAME: ",X1$(1,6)
1150 LET X=FNA(7,X1$)
1160 LET X1=INT(X/16),X2=X-X1*16,Y$="FILE TYPE: "
1165 PRINT "DISC NUMBER: ",X1
1170 ON X2/2GOTO1180,1220,1200
1180 PRINT Y$,"INDEXED"
1190 GOTO 1300
1200 PRINT Y$,"PROGRAM"
1210 GOTO 1300
1220 REM "FILE IS EITHER DIRECT OR SORT
1230 REM "NO RECORD SIZE MEANS IT IS SORT
1240 IF FNB(13,X1$)=0GOTO1270
1250 PRINT Y$,"DIRECT"
1260 GOTO 1290
1270 PRINT Y$,"SORT"
1290 PRINT "KEY SIZE: ",FNA(8,X1$)
1300 PRINT "NUMBER OF RECORDS: ",FNB(9,X1$)
1310 IF FNB(11,X1$)>0PRINT"RECORD SIZE: ",FNB(11,X1$)
1320 PRINT "FILE LOCATION: ",FNB(13,X1$)
1330 INPUT "ENTER CR TO CONTINUE ",*
1340 CLOSE (1)
1350 GOTO 1000
9999 END
```

Figure AIII-1. FID Routine Example

STANDARD RULE:

The READRECORD directive in conjunction with the SIZ= option may be used to accept entry in response to direct questions.

REASON:

The use of READRECORD reduces key strokes because no CR entry is required as a field terminator. However, the use of READRECORD is impractical when more than 1 character must be entered, because it does not have all of the validation capabilities available with INPUT.

```
1000 PRINT @(10,10),"DO YOU WANT A HARD COPY? (Y/N) ",
1010 READRECORD (0,SIZ=1)@(45,10),X7$
1020 IF X7$="N"GOTO9000
1030 IF X7$<>"Y"GOTO1000
```

STANDARD RULE:

Always increment statement numbers by at least ten.

REASON:

This allows room for future corrections and modifications. The renumbering utility, *P and *Q support this technique.

STANDARD RULE:

Always complete a logic loop, (FOR/NEXT or GOSUB). EXIT TO is the only other acceptable method of exiting such logic.

REASON:

The operating system wll support as many open GOSUB and FOR/NEXT loops as the available data area will allow. An error is *not* generated if such routines are improperly exited (i.e.: by a GO TO). However, in order for a program to be easily modified, such routines should follow a logical pattern and come to a logical end.

METHOD EXAMPLES:

```
1050 GOSUB 7000
7000 FOR X = 1 TO 5
7010 IF A$(X,1)="0" EXIT TO 7030
7020 NEXT
7030 RETURN
```

STANDARD RULE:

The current record key or index data should always be available in a variable.

REASON:

In case of a system or program error, the availability of the current key or index may be critical in order to solve the problem. It is possible that the file pointer does not contain the applicable information.

METHOD EXAMPLES:

Use a key or index function when accessing a file. However, this does cause an additional disc access.

OR

If high volumes present a timing problem, using the key or index function may be eliminated. However, the current record key (or index) data must be available for error handling. This can also be done by storing the prior record key (or index) in a variable and executing a dummy read and a key (or index) function as part of the common error processing routine.

STANDARD RULE:

All date entries should be validated thoroughly.

REASON:

Dates are often used as the basis for payment selections, commissions, and management level reports.

METHOD EXAMPLES:

```
1010 INPUT (0,ERROR=1010)X$:(8,8)
1020 IF X$(1,2)< “01” OR X$(1,2)>  “12” OR X$(3,1)<>  “/”
     OR X$(4,2)<  “01” OR X$(4,2)>  “31” OR X$(6,1)<>  “/”
     OR X$(7,2)<  “76” OR X$(7,2)>  “90” GOTO 1010
```

If a valid date is stored as part of the system, use the month and year values for validation if feasible.

STANDARD RULE:

Indicate the program progress on the screen during sort, update or report processing functions.

REASON:

This enables the operator to estimate the time of completion. Also, this alerts the operator if a program is erroneously caught in a loop, or other type of error situation which does not generate a system error.

METHOD EXAMPLES:

This is done by printing the source file key, the index or a record count on the screen. If high record volume warrants, print the key or other data for every twenty-fifth record (or other convenient interval) and print the current key data as part of the common error handling routine.

STANDARD RULE:

Working variables usage should be recorded for each program.

REASON:

The existing variables name usage becomes critically important when changes must be made to a program.

STANDARD RULE:

It is recommended that data variables obtained by reading a file as well as data variables being prepared to be written to a file, reflect the current device number assignment of the file. String variable names then range from An$ through Zn$. Numeric variables range from An through Zn with the exception of arrays. (n represents device number)

REASON:

This allows for control of data variables, prevents accidental usage duplication and is of great aide to another programmer who may later have to work with the program.

METHOD EXAMPLES:

File Data Variables.

```
1050 READ (1,END=9000,ERR=8600,KEY=X1$)A1$,B1$,C1$,D1$,A1,B1,C1

1200 READRECORD (2,END=9000,ERR=8600,KEY=X2$)A2$
```

STANDARD RULE:

The variable used for file key or index values should be consistent and reflect the current device number assignment of the file.

REASON:

It is important that the key or index value can be easily identified.

METHOD EXAMPLES:

A variable that would seldom conflict in keeping with the file data variable assignment standard is X. Therefore, it is recommended that key data variables be Xn$ and index values be represented by Xn. (n is the file device number)

PROGRAM ORGANIZATION

Every program has an appointed task, whether it be the entry of data for storage on disc or the production of a report. Aside from the obvious purpose of the program, there are several logical processes which every complete program must include. It is logical, and not incompatible with the BASIC/FOUR system, that these processing operations be physically located within the program in the same sequence in which they occur. (See Diagram)

PROGRAM IDENTIFICATION

This mandatory section of code consists of remark statements which provide the program name and description, the system to which the program belongs, the revision date, author initials, link information, and (optional) the device number's assignment description. This section occupies statements 10 through 99 and is always the first coding in the program.

INITIALIZATION

This section of code performs "housekeeping" duties, such as clearing memory, screen set up, opening files, setting constants, and printer selection. It begins at statement number 100 and may continue through statement number 999. This section is omitted in the instance of "overlay" or "link" programs where initialization has already been performed.

PROCESSING

The processing section is the coding executed to accomplish the defined task, such as produce a report, accept data entry, or store the data on disc. The logic is executed repeatedly until the task is completed. This section of code occupies statements 1000 through 6999.

SUB ROUTINES

This section of code contains routines which may be repeatedly used from different locations in the program. File accesses, common data entry routines, and formula calculations are a few examples. This section occupies statements 7000 through 7999.

ERROR PROCESSING

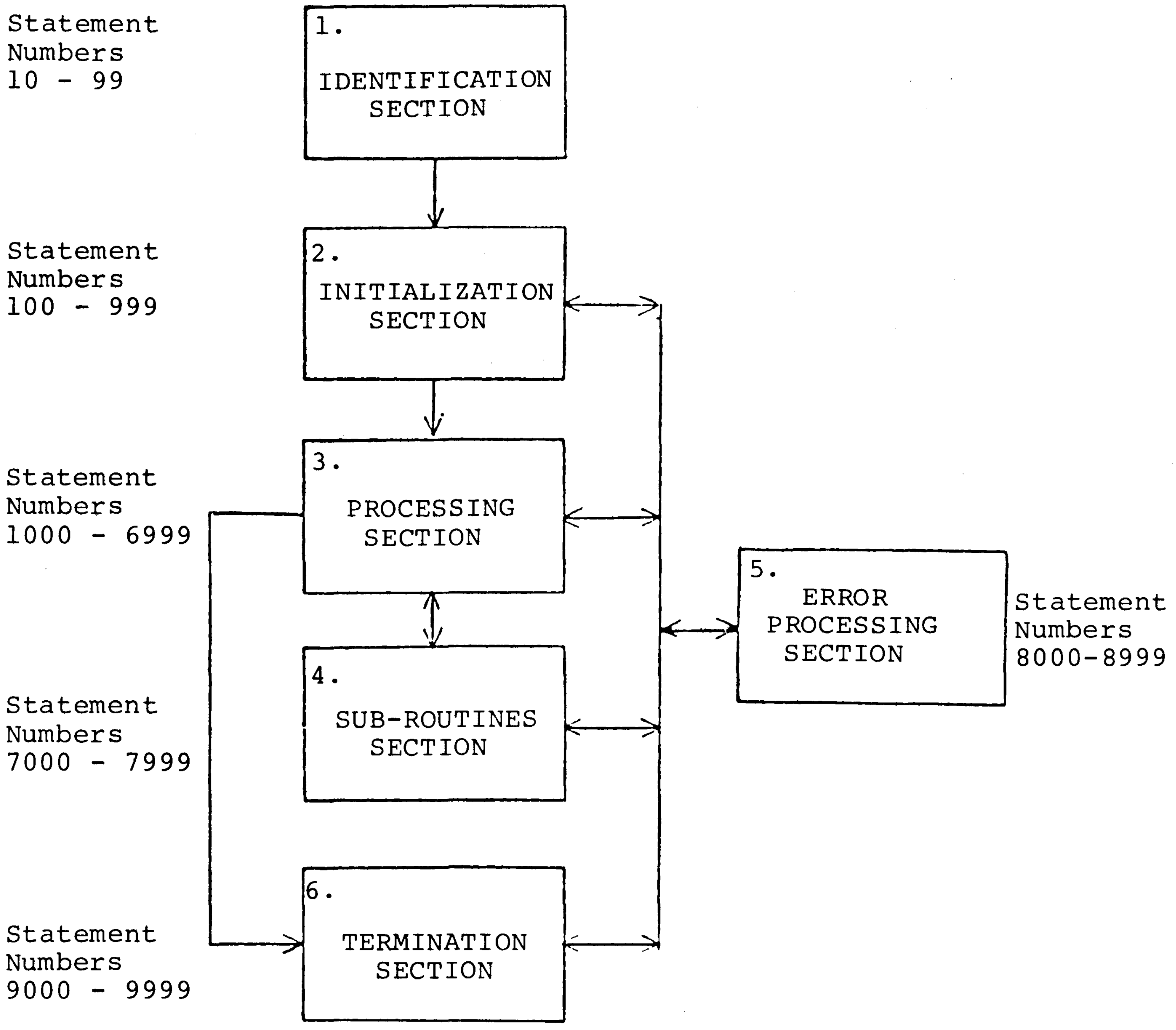
A complete program must contain logic to correctly handle all possible error conditions. An error must never be ignored. The Error Processing section contains all routines necessary for error handling and occupies statement number 8000 through 8999.

TERMINATION

This section performs the logical steps necessary to terminate the program. This includes setting flags, printing totals, and returning to the index program or program link. Also included in this section are mandatory program identification and "END" statements. The Termination section is located in statement numbers 9000 through 9999.

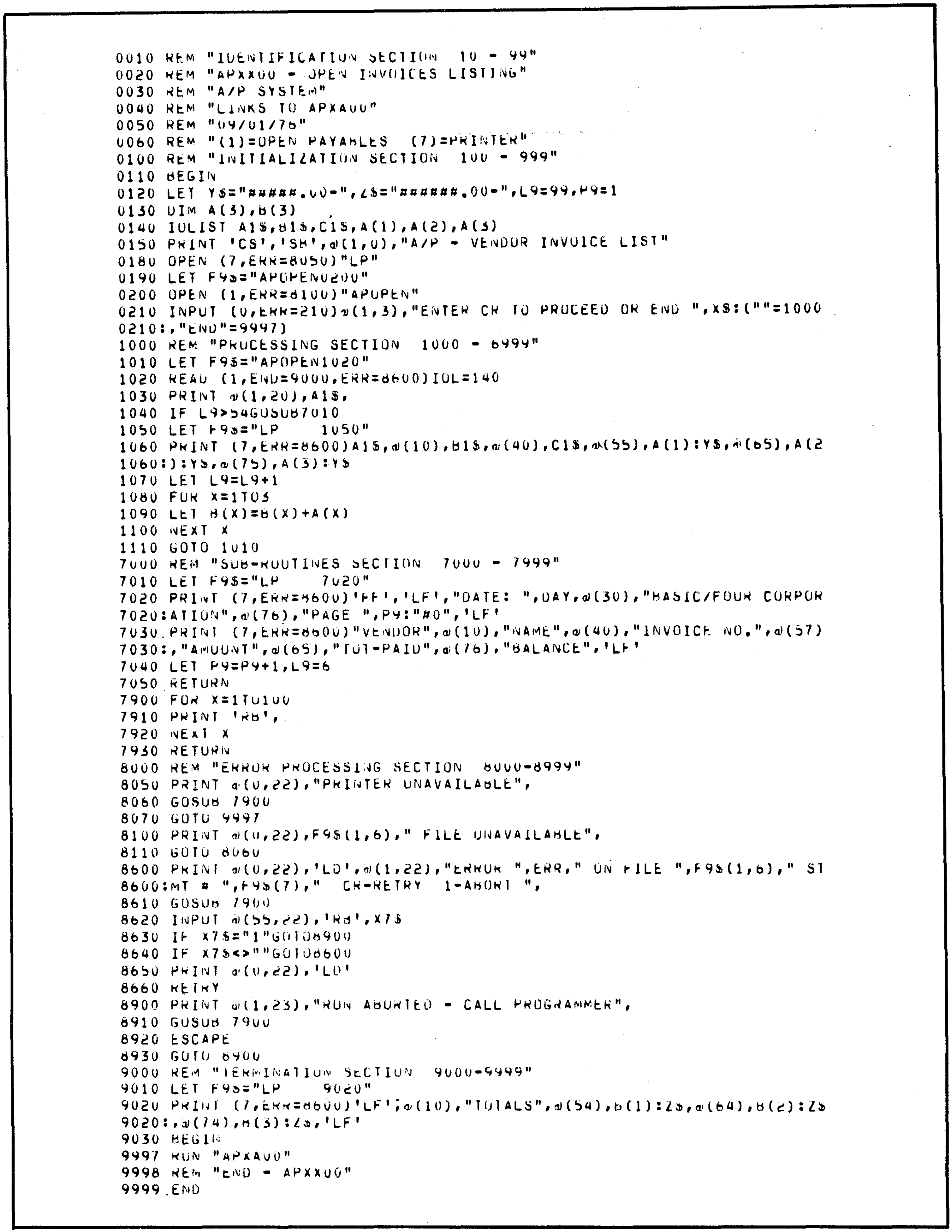
PROGRAM ORGANIZATION

DIAGRAM


Statement Numbers 10 - 99
1. IDENTIFICATION SECTION
Statement Numbers 100 - 999
2. INITIALIZATION SECTION
Statement Numbers 1000 - 6999
3. PROCESSING SECTION
Statement Numbers 7000 - 7999
4. SUB-ROUTINES SECTION
5. ERROR PROCESSING SECTION
Statement Numbers 8000-8999
Statement Numbers 9000 - 9999
6. TERMINATION SECTION

OPTION: If the above diagrammed statement numbering scheme is not used, "REM" statements must be used to head each logical section. Each section should consistently use the same numbering scheme throughout the system.

Figure AIII-2. Program Organization Diagram



```
0010 REM "IDENTIFICATION SECTION  10 - 99"
0020 REM "APXX00 - OPEN INVOICES LISTING"
0030 REM "A/P SYSTEM"
0040 REM "LINKS TO APXA00"
0050 REM "09/01/76"
0060 REM "(1)=OPEN PAYABLES  (7)=PRINTER"
0100 REM "INITIALIZATION SECTION  100 - 999"
0110 BEGIN
0120 LET Y$="#####.00-",Z$="######.00-",L9=99,P9=1
0130 DIM A(3),B(3)
0140 IOLIST A1$,B1$,C1$,A(1),A(2),A(3)
0150 PRINT 'CS','SB',@(1,0),"A/P - VENDOR INVOICE LIST"
0180 OPEN (7,ERR=8050)"LP"
0190 LET F9$="APOPEN0200"
0200 OPEN (1,ERR=8100)"APOPEN"
0210 INPUT (0,ERR=210)@(1,3),"ENTER CR TO PROCEED OR END ",X$:(""=1000
0210:,"END"=9997)
1000 REM "PROCESSING SECTION  1000 - 6999"
1010 LET F9$="APOPEN1020"
1020 READ (1,END=9000,ERR=8600)IOL=140
1030 PRINT @(1,20),A1$,
1040 IF L9>54GOSUB7010
1050 LET F9$="LP    1050"
1060 PRINT (7,ERR=8600)A1$,@(10),B1$,@(40),C1$,@(55),A(1):Y$,@(65),A(2
1060:):Y$,@(75),A(3):Y$
1070 LET L9=L9+1
1080 FOR X=1TO3
1090 LET B(X)=B(X)+A(X)
1100 NEXT X
1110 GOTO 1010
7000 REM "SUB-ROUTINES SECTION  7000 - 7999"
7010 LET F9$="LP    7020"
7020 PRINT (7,ERR=8600)'FF','LF',"DATE: ",DAY,@(30),"BASIC/FOUR CORPOR
7020:ATION",@(76),"PAGE ",P9:"#0",'LF'
7030 PRINT (7,ERR=8600)"VENDOR",@(10),"NAME",@(40),"INVOICE NO.",@(57)
7030:,"AMOUNT",@(65),"TOT-PAID",@(76),"BALANCE",'LF'
7040 LET P9=P9+1,L9=6
7050 RETURN
7900 FOR X=1TO100
7910 PRINT 'RB',
7920 NEXT X
7930 RETURN
8000 REM "ERROR PROCESSING SECTION  8000-8999"
8050 PRINT @(0,22),"PRINTER UNAVAILABLE",
8060 GOSUB 7900
8070 GOTO 9997
8100 PRINT @(0,22),F9$(1,6)," FILE UNAVAILABLE",
8110 GOTO 8060
8600 PRINT @(0,22),'LD',@(1,22),"ERROR ",ERR," ON FILE ",F9$(1,6)," ST
8600:MT # ",F9$(7),"  CR-RETRY  1-ABORT ",
8610 GOSUB 7900
8620 INPUT @(55,22),'RB',X7$
8630 IF X7$="1"GOTO8900
8640 IF X7$<>""GOTO8600
8650 PRINT @(0,22),'LD'
8660 RETRY
8900 PRINT @(1,23),"RUN ABORTED - CALL PROGRAMMER",
8910 GOSUB 7900
8920 ESCAPE
8930 GOTO 8900
9000 REM "TERMINATION SECTION  9000-9999"
9010 LET F9$="LP    9020"
9020 PRINT (7,ERR=8600)'LF',@(10),"TOTALS",@(54),B(1):Z$,@(64),B(2):Z$
9020:,@(74),B(3):Z$,'LF'
9030 BEGIN
9997 RUN "APXA00"
9998 REM "END - APXX00"
9999 END
```

Figure AIII-3. Sample Program Organization

COMMON ROUTINES

The following is a list of the variables which are used in the common routines coded examples.

VARIABLE NAME	DESCRIPTION	ROUTINE
F7	Input field flag	Entry

F9$	Operation device name
	FORMAT;
	F = File name
	S = Statement number
	BBII F9$="FFFFFFSSSS"
		Errors

L7	Cursor vertical position	Entry
L8	Maximum field length	Entry
L9	Line Count	
N7	Numeric entry flag 0=string 1=numeric	Entry
P7	Cursor horizontal position	Entry
P9	Page Count	
R8	Numeric entry maximum range	Entry
X	Miscellaneous	Bell, Password
X$	Miscellaneous	
X7	Input variable – numeric	Entry
X7$	Input variable – string	Entry, Errors, Printer Password
Z0$	Dashes (For VDT screens)	Entry
	OPTION: Use a single character such as \ at the end of the field to indicate length of field.	
Z7$	Zeros	Entry
Z8$	Blanks	Entry

Figure AIII-4. Common Routines

GLOSSARY OF TERMS

GLOSSARY OF TERMS

TERMS AND DEFINITIONS

ALPHANUMERIC – Characters which are either letters of the alphabet, numerals or special symbols.

APPLICATION PROGRAM – An application is a specific problem or job to be solved through the use of a computer or other data processing machinery. Therefore, an application program is one of several used to solve the specific problem.

Some examples are: Inventory Control, Payroll, Cost Accounting, etc.

ARRAY – An array is a group of numeric variables with the same name that are referenced through the use of a subscript. This can be thought of as being similar to a family of people, all of whom have the same surname, but who have different first names.

For example: Let us assume that the variable X has been defined as having three elements. These would be referenced as X(0), X(1) and X(2).

ASCII CODE – ASCII is the name of a standard code that assigns specific bit patterns to each sign, symbol, letter and operation in a specific set.

ASCII stands for: American Standard Code for Information Interchange. ASCII is the code used by the BASIC/FOUR computer.

BBI – The acronym for Business BASIC I, the first programming language level used on BASIC/FOUR systems.

BBII – The acronym for Business BASIC II, the programming language level used on BASIC/FOUR systems.

BIT – A contraction or abbreviation of binary digit. It signifies the smallest piece of smallest unit of information. In the computer, bits are either on (expressed by a 1) or off (expressed by 0). A group of eight bits is referred to as a byte.

BLANK – The character-code that will result in the printing of a space in a given position.

BOSS – An acronym for Basic Operating Software System.

BRANCHING – Branching within a computer program is the means whereby the normal sequence of execution is changed. A branch instruction causes the computer to execute an instruction other than the next one in sequence within the program.

Business BASIC Branch instructions are: GOTO . . . ON GOTO . . . GOSUB (See program sequence control).

BUSINESS BASIC – The interpretive, high level programming language used on BASIC/FOUR computers.

BYTE – The smallest addressable unit of information in memory. A byte is a group of eight binary bits. A byte can contain a value of 0 through 255.

The first four bits in a byte are referred to as the high order bits. The last four bits in a byte are referred to as the low order bits. One byte equals one character.

CENTRAL PROCESSOR – The portion of a computer that consists of the three main sections – arithmetic and control, input/output and memory.

CODING – To prepare a set of computer instructions required to perform a given action to solve a given problem.

CONCATENATION – Concatenation means to unite or join together. In Business BASIC, concatenation of alpha-numeric string information may be accomplished through the use of the + sign.

CONSTANTS – A constant is a quantity or data item that does not vary in value within a program.

CONTROL KEYS – The control keys (also called "Motor Bars") are used to reduce key strokes required for branching (see INPUT in Reference Manual).

CONTROL VARIABLE – In Business BASIC, a control variable is a numeric variable required as a parameter in FOR/NEXT loops. Execution of a FOR statement will set the control variable to a value. Each execution of a NEXT statement will increment the value. Each execution of a NEXT statement will increment the value in the control variable by the step value, if specified, or by 1. When the value in the control variable exceeds the end value, the FOR/NEXT loop is terminated.

COUNTER – A variable or a location in memory which can be set to an initial number and increased or decreased by an arbitrary number.

DATA FILES – Data represents information and information is the assigned meaning. Computers process and handle only data.

A file is a collection of related data. The data is present in the forms of records.

A data file may be in the form of punched cards, punched paper tape, magnetic tape or may be magnetically recorded on a disc.

DATA HANDLING FUNCTIONS – Data handling functions are provided to manipulate values contained in either string or numeric variables.

Functions are dividied into groups:
Those that examine a variable or provide in numeric form a part or characteristic of the variable.
Those that convert a variable from one form or code to another.

DATA LIST – In Business BASIC, the term data list refers to the collection of individual data items to be either entered from a file or output to a file.

The collection of items can be any combination of constant and/or variable information.

DEBUGGING – The process of determining the correctness of a computer routine, locating any errors, and correcting them. Also, the detection and correction of malfunctions in a computer itself.

DEBUGGING, ON-LINE – The act of debugging a program while time-sharing its execution with an on-line process program.

DEVICE – A device is any hardware mechanism that is created, formed, invented, devised or constructed by design.

Business BASIC refers to a device as a means of inputing information or outputting data.

Devices are: card readers, paper tape punch units, punch tape readers, magnetic tape units, video display terminals.

DIRECT FILE – A *direct* file is a type of file containing data records. As the file is created, and as records are written to the file initially, a string value is associated with each record. This value is called the key.

Direct files may be read:

RANDOMLY – using an index or key.
LOGICALLY – according to key sequence or according to record sequence using the index.

DIRECT MODE – Direct mode refers to the mode of operation in the Business BASIC language where statements keyed-in at a terminal are executed immediately after they are entered.

Statements to be executed immediately, or in direct mode are entered with no statement numbers.

DIRECTIVE – The Directive is the portion of a Business BASIC statement which specifies the operation to be performed.

Business BASIC Directives are English language word(s).

Some examples are: LET GOTO END STOP PRINT

DOCUMENTATION – Design, program and operator documentation which completely describe a system (See the USER DOCUMENTATION Section).

ERR TASK VARIABLE – ERR task variable is a numeric or simple variable which contains a code that represents the error status of the particular task.

The name of this variable is "ERR".

For example, to print its contents the following statement is used: PRINT ERR

ERROR HANDLING – Program coding routines which provide for all possible error conditions.

EXPONENTIATION – Exponentiation means to raise to a power. That is, to multiply a number by itself a specified number of times.

The symbol for exponentiation in Business BASIC is ↑ (up arrow). Example: 2↑2 means to raise 2 to the power of 2 (result is 4).

FIELD TERMINATOR – Business BASIC provides for field terminators. All data entered at a terminal must be followed by one field terminator unless READRECORD is used for input. (Also called control keys).

The field terminators are the following keys:

CR	–	Carriage Return
FUNCTION KEY I	–	FS (Field separator)
FUNCTION KEY II	–	GS (Group separator)
FUNCTION KEY III	–	RS (Record separator)
FUNCTION KEY IV	–	US (Unit separator)

FLOATING POINT – Floating Point is a system of representing numbers using a pair of numbers, one of which represents the digits in the number, the other which represents the power of ten by which the number is multiplied.

Floating Point notation is used in Business BASIC to represent very large or very small numbers.

Example: The number 1 in Floating Point notation is represented as 1E+01 (that is – 1 times 10 raised to the power of 1).

FLOWCHART – A diagram consisting of a set of symbols and connecting lines that show step-by-step progression through a procedure or system.

FORMAT MASK – Format Masks are the means provided by Business BASIC to print numbers in a formatted manner. A Format Mask is a string constant or the name of a string variable. The mask is appended to the number to be formatted by using a colon (:).

Example: PRINT 100:"00000" will display the number as 00100
PRINT 100:"#####" will display the number as 100

Format masks provide for proper insertion of commas, decimal points, dollar signs, positive and/or negative signs, etc.

HEXADECIMAL – Hexadecimal refers to a numbering system where there are sixteen possible digits. These digits are 0 through 9 and the letters A B C D E and F.

In the computer, a byte consists of 8 bits which can be divided into 4 high order bits and 4 low order bits. Each 4 bits contain a hexadecimal digit.

HORIZONTAL POSITION – Horizontal position refers to the position within a line on the terminal or printer.

The terminal has 80 horizontal positions referred to as 0 through 79.

Printers normally have 132 horizontal positions referred to as 0 through 131.

INDEXED FILE – An *indexed* file is a data file that is created by writing records sequentially starting with the first physical location for records and continuing to the end.

Indexed files may be accessed in two ways:

Sequentially, starting with the first physical location and continuing to the end.
By using a specific record's location. This is referred to as the index. The first record has an index of 0 (zero).

INTERACTIVE – A system with the ability to directly respond to user commands.

KEY – A key is the string value associated with each record of a direct file. It may be used to access individual records in the direct file.

LOGICAL OPERATORS – Logical Operators are used by Business BASIC to define the testing to be done in IF statements.

=	is equal to
<	less than
>	greater than
<> or><	not equal to
>= or=>	greater than or equal to
<= or=<	less than or equal to

LOOP – A loop is a group of statements in a program that are executed a specified number of times within the program at a given point.

The first statement in a Business BASIC loop is the FOR statement. The last statement in Business BASIC loop is the NEXT statement.

For example: The following loop is executed 5 times . . .

```
7100 FOR I=1T05
7110 . . . any BASIC statement
7120 NEXT I
```

MATHEMATICAL SYMBOLS – Mathematical symbols are characters recognized by Business BASIC to represent mathematical operations to be performed.

+	tells the computer to add
–	tells the computer to subtract
*	tells the computer to multiply
/	tells the computer to divide
∧	tells the computer to raise to a power

MNEMONIC CONSTANTS – A mnemonic constant consists of two characters, enclosed in primes (') and is used to cause a specific action to occur on a peripheral device or on a terminal.

NUMERIC ARRAYS – A numeric array is a group of numeric variables with the same name that are referenced through the use of a subscript. This can be thought of as similar to a family of people, all of whom have the same surname, but each of which have a different first name. (See ARRAY)

NUMERIC CONSTANT – A numeric constant is a number whose value does not vary within a program.

NUMERIC FIELD – A numeric field is an item of data present in a data record whose contents are always numeric information.

A numeric field is represented by using a simple variable name, numeric array, or a numeric constant.

ON-LINE – Directly interacting with a computer.

OVERLAY – An overlay is a continuation program. It is also often referred to as a LINK.

PARAMETER – A parameter is a required and/or optional value used in association with a statement directive to define in detail the action to be taken by the computer.

Parameters include or reference all variable or constant information that may be associated with the particular BASIC statement. The type of information included depends upon the particular directive.

Parameters often consist of variables.

PERIPHERAL DEVICE – A peripheral device is a means of inputting information to the computer or outputting information from the computer.

A BASIC/FOUR system may have any of the following peripheral devices: Card Reader, Paper Punch, Paper Tape Reader, Magnetic Tape Units, Video Display Terminals and Printer. (See DEVICE)

PRECISION STATEMENT – The precision statement, after it is executed, specifies the number of decimal places to the right of the decimal point that final results and intermediate results of mathematical expressions are to be carried.

Business BASIC provides for a minimum of 0 (none) through a maximum of 14 digits to the right of the decimal point.

PROGRAM – A program is a series of instructions which the computer follows to perform a given task. These instructions must be written in a language understood by both man and machine.

PROGRAM FILE – A program file is a disc file that contains a Business BASIC program.

PROGRAM INITIALIZATION – Program initialization refers to certain operations normally performed at the beginning of a computer program. Business BASIC provides three directives which can be used for initialization. These are BEGIN, CLEAR, and RESET.

The first statement in a Business BASIC program should be the BEGIN statement. This will *clear all data, close all files and reset any incomplete loops* and resets the precision to 2. The CLEAR statement only *clears data and resets incomplete loops* and resets the precision to 2. The RESET statement only *resets incomplete loops* and resets the precision to 2.

PROGRAM LANGUAGE – A programming language is basically the means of communication between man and computer. It is a *set of rules and conventions that govern the manner and sequence in which instructions are written or specified for execution by a computer.*

PROGRAM LOOP – A program loop is a group of statements in a program that are executed a specified number of times within the program at a given point. (See LOOP)

PROGRAM SEQUENCE CONTROL – Program sequence control statements change the order of processing statements within a program.

Business BASIC provides two statements that can alter the sequence of processing statements in a program. These are GOTO and ON/GOTO. (See BRANCHING)

PROGRAM TERMINATION – Business BASIC provides two directives to terminate processing of a program. The two directives function in the same manner. When executed, all files and/or devices opened are closed and thus made available for use. Execution of a program termination statement does not change the contents of any variables. The two statements provided are: STOP and END.

RANDOM – Permitting access to stored data in no particular sequence.

RECORD – A record consists of a group of data elements or fields that relate to a single major item. A collection of records is called a file.

For example: An inventory file might consist of records that would each relate to a single inventory item. Each record might contain the inventory item code, a description of the item, its price, the current quantity on hand, the quantity on order, etc.

ROUND – Round refers to a manner of adjusting a decimal number. When the computer rounds a number, the least significant digit is deleted, and the remaining part of the number is adjusted based on the following:

1. If the least significant digit is 4 or less, that digit is merely deleted to obtain the rounded result.

2. If the least significant digit is 5 or more, that digit is deleted and 1 is added to the next higher significant digit.

For example: 21.534 is 21.53 if rounded; 21.536 is 21.54 if rounded.

ROUTINE – A series of computer instructions which perform a specific, limited task.

SECTOR – The smallest accessible portion of a disc.

SEQUENTIAL – Sequential means an order. In Business BASIC, sequential refers to logical order of records in a disc file or a file present on a peripheral device.

An example of sequential is the manner in which a program is executed. That is, in statement number sequence. The first statement of a program is executed first, followed by the second, followed by the third and so forth.

SIGNIFICANT DIGIT – *Significant digit is the first non-zero digit in a number.*

For example: in the number 00030211, the first significant digit is 3.

The least significant digit in a number is that digit which contributes the smallest quantity to the value of the number.

For example: in the number 21.5436, the least significant digit is 6.

SIMPLE VARIABLE – A simple variable can contain numeric information only. It can be named as any single alphabetic character (A–Z) or as a single alphabetic character followed by a single numeric digit (0–9).

The following are all valid simple variable names: A J8 X9 G P1 (See VARIABLE).

SLEW – *Slew refers to the movement of paper in a printer through a distance greater than* the normal line spacing without printing.

SOFTWARE – The programs or routines, and supporting documentation, which instruct the operations of a computer.

STATEMENT – An instruction that defines an operation and which as a unit causes the computer to complete that operation.

STRING – A string is a collection of characters that are a mixture of numbers, alphabetic characters and/or special characters such as punctuation.

In Business BASIC, strings may be present in constant or variable form. (See STRING VARIABLE, STRING CONSTANT)

STRING CONSTANT – A string constant is a group of alphabetic and/or numeric characters, enclosed in quotation marks, that do not change within a program.

String constants are often used to communicate with an operator via the terminal.

For example: "This is a string constant" "So is this"

STRING FIELD – A string field is a data item within a record whose contents are a mixture of alphabetic and/or numeric information.

A string field may be represented by either a string constant or a string variable.

STRING VARIABLE – A string variable is a variable containing a mixture of alphabetic and numeric information. String variables may be named as any single alphabetic character (A–Z) followed by a dollar sign ($) or as a combination of a single alphabetic character (A–Z) followed by a single numeric digit (0–9) followed by a dollar sign ($).

Some examples of string variable names are: A$ A8$. N2$ U$ Z1$

SUBROUTINE – A subroutine is a series of program statements that are to be executed in more than one place within a given program. Subroutines may be called for execution at any point within the program.

The Business BASIC directive used to transfer program control to a subroutine is the GOSUB instruction.

Program control will be transferred back to the statement following the GOSUB statement when a return is executed. Return should be the last statement in a subroutine.

SUBSCRIPT – A subscript is a portion of a string or an array. To reference a portion of a string, the programmer must specify the starting position within the string. He may optionally specify the number of characters to be referenced.

For example: Assume the variable A$ contains 5 characters, ABCDE
A$(1,3) is the ABC
A$(5,1) is the E

SYNTAX – Syntax is the rules governing the structure of program statements or expressions in the Business BASIC language.

SYSTEM – A collection of parts or devices that forms and operates, as an organized whole through some form of regulated interaction.

SYSTEM DESIGN – The definition and documentation of the system requirements, function relationships and technical procedures, necessary to develope and support the system.

SYSTEM VARIABLES – There are two system variables in the Business BASIC language:

1. TIM is a numeric variable containing the current value of the built in computer clock expressed in ten thousands of an hour.

2. DAY is a string variable containing eight characters that are its current contents.

The system directives provided to change these variables are SETTIME and SETDAY respectively.

TASK – Any user program.