WICAT

Multi-user Control System (WMCS) LINK

Programmer Reference Manual

188-161-301 B

April 1984

WICATsystems

Copyright Statement

Receipt of this manual must not be construed as any kind of commitment, on the part of WICAT Systems Incorporated, regarding delivery or ownership of items manufactured by WICAT.

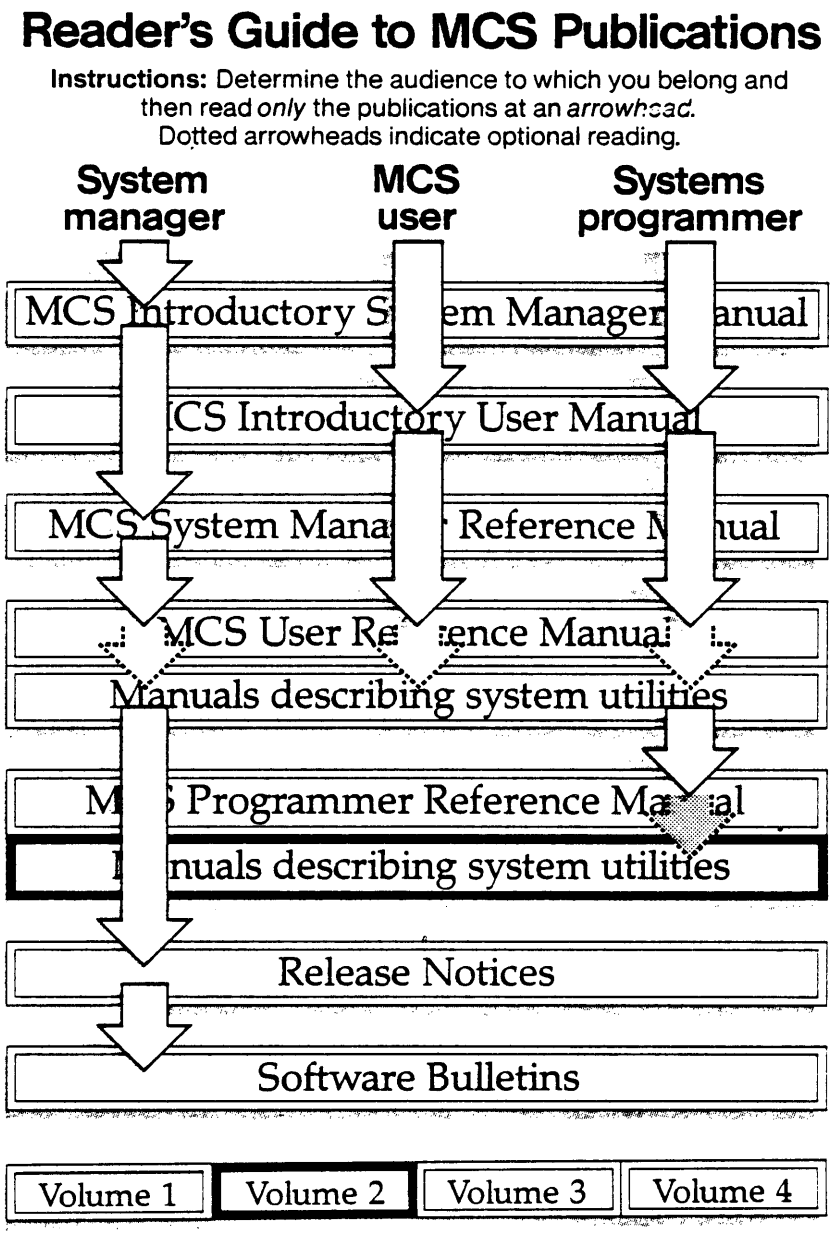
This manual is subject to change without notice.

Revision History

First Printing September 1981
Second Printing April 1984

The Purpose of This Manual

The manual explains how to use the LINK Program for program development.



An **introductory user manual** is a tutorial introduction to a software product. In other words, use of the product is explained in a step-by-step, user-friendly format that walks you through some fundamental aspects of the product. When you complete the introductory manual, you have the experiential basis for understanding the products's user reference manual.

User reference manuals are written for those who are new to the product (but who have read the introductory user manual) as well as for the experienced user.

System manager reference manuals contain information for those who perform administrative tasks associated with routine system operation.

Programmer reference manuals are written for programmers who at least understand programming fundamentals.

WMCS LINK Programmer Reference Manual

Table Of Contents

Table of Contents

TYPOGRAPHICAL CONVENTIONS USED IN THIS MANUAL

Uppercase letters within text indicate sample command line character strings, file designations, diagnostic messages, and reports. Such samples are in uppercase when run into the text so that you can distinguish what would be typed (or what would appear) on the screen. Examples set off from the text are in lowercase.

Bold facing indicates what you should type onto the screen, i.e., whereas uppercase characters indicate sample character strings, etc., that are part of an example, bold faced characters indicate what you must type as part of a procedure.

Square brackets, [], indicate a function key, the name of which appears in uppercase within the brackets, e.g., [RETRN], [CTRL], etc.

Underlining is used for emphasis.

CHAPTER 1

OVERVIEW

The WICAT Multi-user Control System (WMCS) LINK Program is a programming tool used to prepare the output of a compiler or assembler so that that output can be executed by the computer's hardware. Therefore, LINK produces a file that is an **executable image.**

The executable image has **sharable** and **nonsharable** segments. Sharable segments can be shared by more than one process.

LINK also creates a listing of symbols and addresses useful for cross referencing.

The description of the LINK Command, in the WICAT Multi-user Control System (WMCS) User Reference Manual, tells you how to execute LINK.

1.1 THE ROLE OF THE LINKER

The object modules created by language translators (including assemblers) are nonexecutable. External symbol references are unresolved. Unresolved symbol references include run-time modules required by high level languages as well as user-specified external declarations. The linker binds the symbol reference with the symbol definition and creates a memory image that can be read into memory and executed. Without a linker, modular programming is impossible, and language translators would be much more complex.

1.1.1 Program Modules

Modular programming, the process of combining separately compiled or assembled modules into an executable image, simplifies and enhances program development in the following ways:

1. Smaller modules are easier to write and maintain

because it is easier to find and fix errors in a smaller module.

2. Many modules are applicable to more than one task, and modules that have been used and tested in other programs can be included in new applications. This speeds program development.

3. Complicated programming requirements are simplified by breaking the requirements into smaller tasks, each of which can be allocated to a different person for implementation and debugging.

4. In some cases, the best language for one module may not be best for another part of the program. Modular programming allows modules written in different programming languages to be combined in a single application. Thus the programmer can use the language most suited to the application.

Figure 1.1 illustrates the role of the linker in modular programming wherein TIME is a computer program that interacts with the user to set the system clock on a WICAT system. The interaction is achieved through a main program written in WICAT Pascal. The clock is actually set by an assembly language program written in WICAT Assembler.

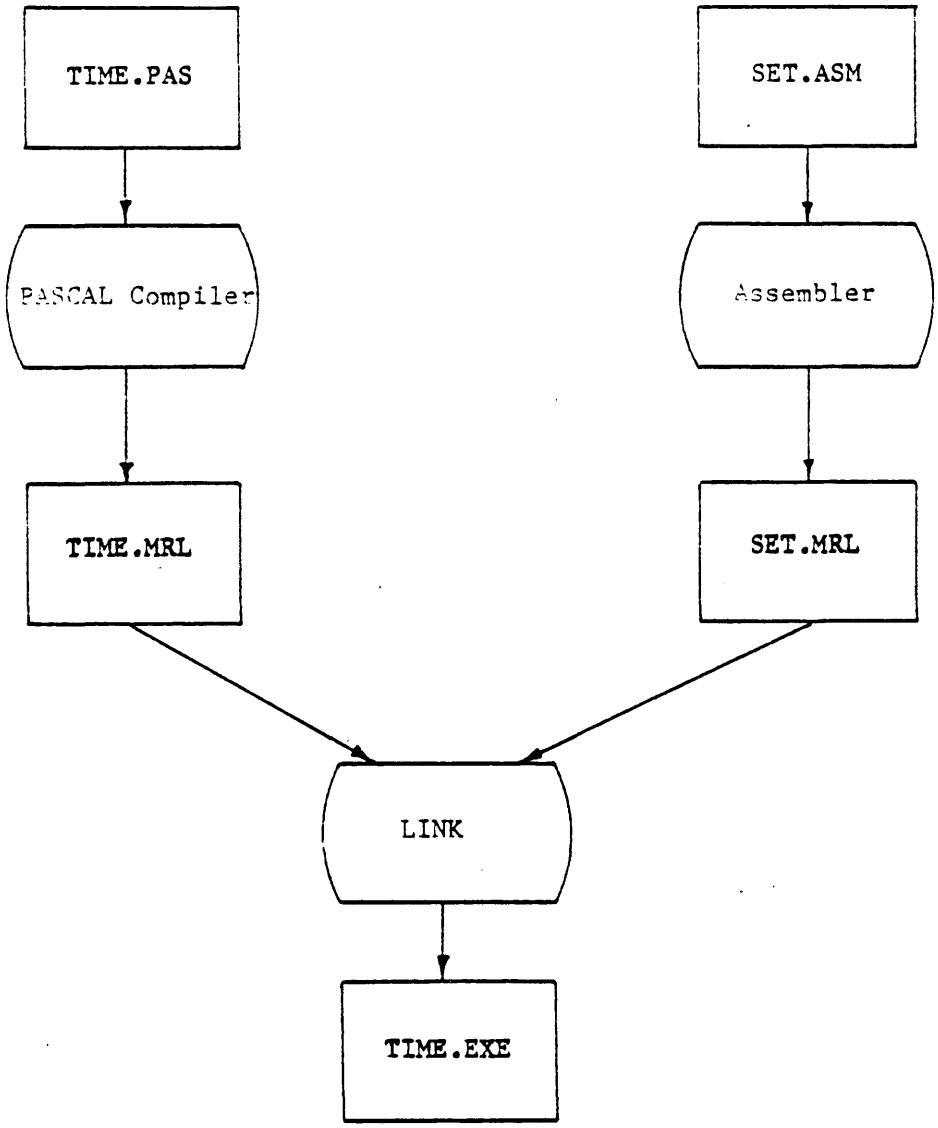


Figure 1.1. Modular Programming

1.1.2 Compilation And Assembly

LINK simplifies the work done by language translators because it assigns code to executable pages.

1.2 LINK FUNCTIONS

LINK produces two files:

1. An executable-image file (EXE is the file extension for this file).
2. A cross reference-map file (MCR is the file extension for this file).

These files are discussed in the following sections.

1.2.1 Executable-image File

An executable-image file is file type number 1. The record size of the file is 1024 bytes and the first record of the image file contains bit maps, each of which is 256 bytes long. These maps represent logical address space. Each of the bits in a map is a semaphore for a 1-Kbyte segment of address space. The first map defines the correspondence between the records on the disk and their ultimate location in memory. The second map defines the initial memory allocation. The third map indicates which pages can be shared.

The fourth map is not used on a WICAT computer.

1.2.2 Cross Reference Map File

This file is a listing file that specifies each module name and starting address.

CHAPTER 2

SYMBOLS AND REFERENCES

LINK's primary responsibility is to resolve symbolic references between modules.

2.1 DEFINITION

A **symbol** is an identifying label or name associated with one or more program statements or data area. A **reference** is the use of a symbol in a program statement of data definition.

Figure 2.1 illustrates a sample program called TEST. The program is written in Pascal and consists of one main routine and one subroutine.

```
1  PROGRAM TEST;
2
3  VAR   I    : INTEGER;
4
5  PROCEDURE INCREMENT;
6  BEGIN
7        I := I + 1;
8  END;
9
10  BEGIN
11        I := 0;
12        INCREMENT;
13  END.
```

Fig 2.1 Sample Program

These are the symbols used in TEST:

TEST	The name of the main routine
INCREMENT	The name of the subroutine
I	The name of a data area or variable

These are the references used in TEST:

line 7	I is referenced twice
line 11	I is referenced once
line 12	The subroutine INCREMENT is called

2.2 TYPES

Symbols are one of two types: local or global. LINK treats each type differently.

2.2.1 Local Symbols

Local symbols can be referenced only in the routine in which the local symbols are defined.

Note that for a program such as that in figure 2.1 (where there are no local symbols) the compiler or assembler resolves all references to local symbols.

2.2.2 Global Symbols

Global symbols can be referenced by routines other than the routine that defines them. For example, in figure 2.1 all symbols are global because they are defined by TEST (the main routine) and are therefore global to all subroutines. LINK resolves global symbol references (see chapter 4).

Global symbols are of two types:

1. Internally defined symbols, defined in the main program.

2. Externally defined symbols, defined in a routine that is external to the main program, and independently compiled (e.g., runtime routines used by Pascal).

CHAPTER 3

OBJECT MODULES

3.1 DEFINITION

The output (or object) file for the compiler or assembler is the input file for LINK. An object file consists of modules. In the case of the Pascal compiler, a discrete module is created for each procedure in the program along with one for the main program itself. An additional module is created for global variables. In chapter 2, figure 2.1, we examined a simple Pascal program. Figure 3.1 illustrates the modules created by the compiler.

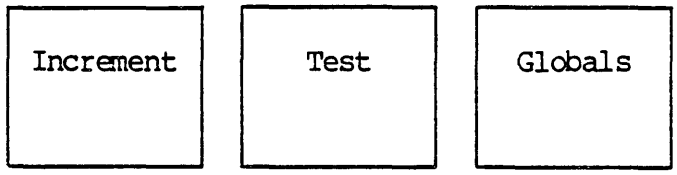


Fig. 3.1 Modules Created By PASCAL Compiler For TEST Program

A module can contain several records. An object record has the following hexadecimal format displayed in bytes:

Type	Byte Count	Relative Address	Code Byte 1	Code Byte 2	. . .	Code Byte n	Check Sum

Fig. 3.2 Object Record Format

This is an explanation of the terms appearing in figure 3.2:

Type	Type of object record described in 3.2, a label.
Byte Count	Number of bytes in the record following the record type (not including the byte

count itself).

Relative Address	Relative address of the code bytes in the module.
Code Byte	In the case of a code type record, the code bytes contain the actual hexadecimal code for the module; in the case of a symbol, the code bytes contain a name.
Checksum	The checksum is the one's complement of the sum of the bytes.

Diagrams showing the format of each type of record are found in appendix D.

Figure 3.3 shows the hexadecimal object file created by the Pascal compiler for program TEST (shown in figure 2.1). Note, the three modules relating to figure 3.1.

Type	Count	Relative Address	Code/ASCII String/Checksum	
RP	05	00000000	FA	Branch to start
R1	0C	00000000	2E53544152542E09	of main program
R2	0B	00000000	4EF900000000AD	
R7	0A	00000002	544553543083	
R8	05	00000006	F4	
RP	05	00000000	FA	INCREMENT module
R1	0F	00000000	494E4352454D454E54311A	
R2	17	00000000	4E56FFFC2D0D5279[0000004A]2A5E4E5E4E7503	
R7	16	00000008	2450415343414C2D474C4F42414C242D32A8	
R8	05	00000012	E8	
RP	05	00000000	FA	TEST module
R1	0A	00000000	544553543085	
R2	25	00000000	2F3C[00000006]2F3C[00000028]4EB9[00000000]4279[0000004A]2A4E4EB9[00000004]B	
R2	0B	00000020	4EB9[00000000]CD	
R7	16	00000002	2450415343414C2D474C4F42414C242D32AE	
R7	16	00000008	2450415343414C2D474C4F42414C242D32A8	
R7	0B	0000000E	52525230303060	
R7	16	00000014	2450415343414C2D474C4F42414C242D329C	
R7	0F	0000001C	494E4352454D454E5431FE	
R7	0B	00000022	5252523030314B	
R8	05	00000026	D4	
RI	05	00000000	FA	GLOBAL module
R1	16	00000000	2450415343414C2D474C4F42414C242D32B0	
R8	05	0000004C	AE	
R9	05	00000000	FA	

Fig. 3.3 Object File For TEST Program

Each record begins with a record label, 'R', followed by an integer between 0 and 9 or one of the alphabet characters 'I' or 'P'. References to global symbols (subroutines, variables, etc.) must be made using the M68000 absolute long addressing mode. In the object file, using figure 3.3 as an illustration, all references to internally and externally defined global symbols (refer to chapter 2) are enclosed in a box. Note, all references to internally defined global symbols have an offset into the global module while all references to externally defined global symbols have an address value of 0. These global references can be better understood by studying a listing of the assembly code generated by the compiler for TEST. This listing is illustrated in figure 3.4.

```
WICAT Pascal  Version 1.3

;procedure: increment(  1)
0000    4E56    link     a6,#-4
        FFFC
0004    2D0D    move.l   a5,-(a6)
0006    5279    addq.w   #1,74
        0000             global
        004A
000C    2A5E    move.l   (a6)+,a5
000E    4E5E    unlk     a6
0010    4E75    rts

;procedure: test(  0)
0000    2F3C    move.l   #6,-(sp)
        0000             global
        0006
0006    2F3C    move.l   #40,-(sp)
        0000             global
        0028
000C    4EB9    jsr      xxxxxx
        0000             RRR000
        0000
0012    4279    clr.w    74
        0000             global
        004A
0018    2A4E    move.l   a6,a5
001A    4EB9    jsr      xxxxxx
        0000             INCREMENT1
        0000
0020    4EB9    jsr      xxxxxx
        0000             RRR001
        0000
```

Fig. 3.4 Assembly Code Generated By Compiler For TEST Program

Note the following:

1. Global reference for I following address 0006 in module: INCREMENT

2. Global references for I in module: TEST, following address 0012.

3. External reference for runtime routine RRR000 following address C.

4. External reference for INCREMENT module following address 1A.

5. External reference of runtime routine RRR001 following address 20.

3.2 RECORD TYPES

There are four broad categories into which all object module records can be classified:

1. Delimiter.
2. Code.
3. Constant/label definitions.
4. References to global symbols.

3.2.1 Delimiter

There are six record types in this category, as defined by the second character in the type field:

RP Delimits a sharable segment; pure storage.

RI Delimits a nonsharable segment; impure storage.

R0 Identifies the main program. The program name is coded in the code bytes. This record delimits the sequence of records pertaining to the main program.

R1 Identifies a module or subroutine. The module name is coded in the code bytes. This record delimits the sequence of records pertaining to a module.

R8 This record marks the end of a module. The address of the first free location past the end of the module. LINK uses this offset from the start of the module to determine where it can start loading the next module. This record contains the amount of space used by the module.

R9 This record marks the end of the object file. An object file can contain several modules.

3.2.2 Code

An R2 label indicates that the record contains program code.

3.2.3 Constants/labels

There are four types of object records found in this category:

R3 Identifies a local constant. The name is coded in the code bytes. The relative address field contains the value of the constant.

R4 Identifies a local label. The name is coded in the code bytes. The relative address field contains an offset pointing to the labeled location.

R5 Identifies a global constant. The name is coded in the code bytes.

R6 Identifies a global label. The name is coded in the code bytes.

3.2.4 Global Symbols

Record type R7 identifies a reference to any global symbol. The relative address field is the location (within the module) of the reference to the global symbol.

CHAPTER 4

IMAGE CREATION

LINK reads relocatable object modules, resolves references between modules, and creates an image file that the operating system can load and execute.

As relocatable modules are read in, they are placed in a virtual address space that resembles the address space the program uses during execution. By convention, this address space is divided in half with the lower half reserved for program code and constants. The upper half is reserved for the stack and other variables. The purpose of this convention is to allow the non-changing or pure portions of programs to be shared among several users. See figure 4.1.

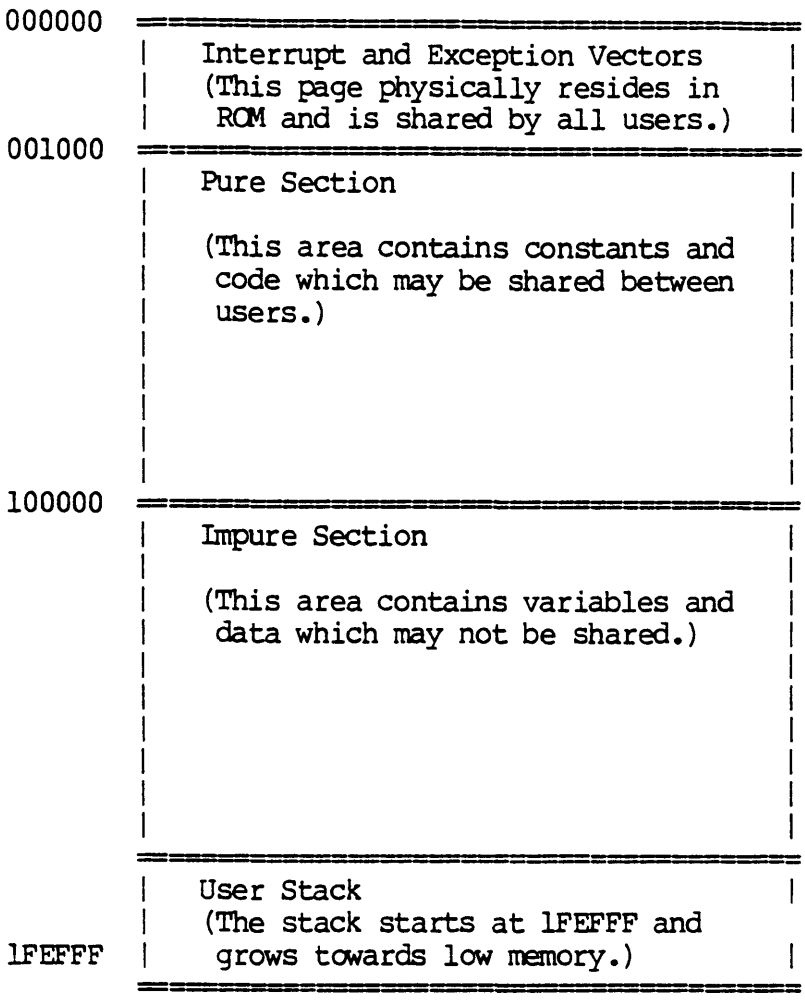


Fig. 4.1 Memory Map Of The 2-Mbyte Virtual Address Space

Space is allocated consecutively in these two segments. Allocation in each area is controlled by a base pointer that points to the lowest unallocated memory location. The pointer into the lower half of the address space is called Pure-Base and has an initial value of 1000 Hex. The pointer into the upper half of the address space is called Impure-Base and has an initial value of 100000 Hex.

As the relocatable modules are read in, they are placed into the virtual address space relative to one of these two bases. The base pointer chosen depends on the 'RP' or 'RI' type record that must be the first record in a module. If the record is type 'RP' the module goes into the lower half of the address space. If the record is type 'RI' the module goes into the upper half of the address space.

The end of a module is delimited by an 'R8' type of a record. The address field of an 'R8' type record contains a count of the number of bytes of memory used by the module. When a module has been completely read in, the base pointer is incremented by the value of the address

field in the R8 record. Thus the pointer once again points to the bottom of free memory.

Between the start and end of a module, three classes of records may be encountered. The first class causes data to be entered into the virtual memory address space. The second class causes symbols to become defined. The third class marks a reference to a symbol.

There is only one instance of the first record class. This is the 'R2' type of record. This record causes bytes of data to be stored in the virtual memory space.

The second and third record classes affect the linker symbol table that is defined by the series of Pascal statements in figure 4.2.

```
TYPE    Symbol-Table = Record
          Next       :  Symbol-Table;
          Name       :  Packed Array [1..20] of Char;
          Value      :  Address
          Referenced :  List-of-Undefined;
        End;

        List-of-Undefined = Record
          Next     :  List-of-Undefined;
          Value    :  Address;
        End;
```

Fig. 4.2 PASCAL Definition Of Linker Symbol Table

Pictorially, the symbol table corresponds to the diagram in figure 4.3.

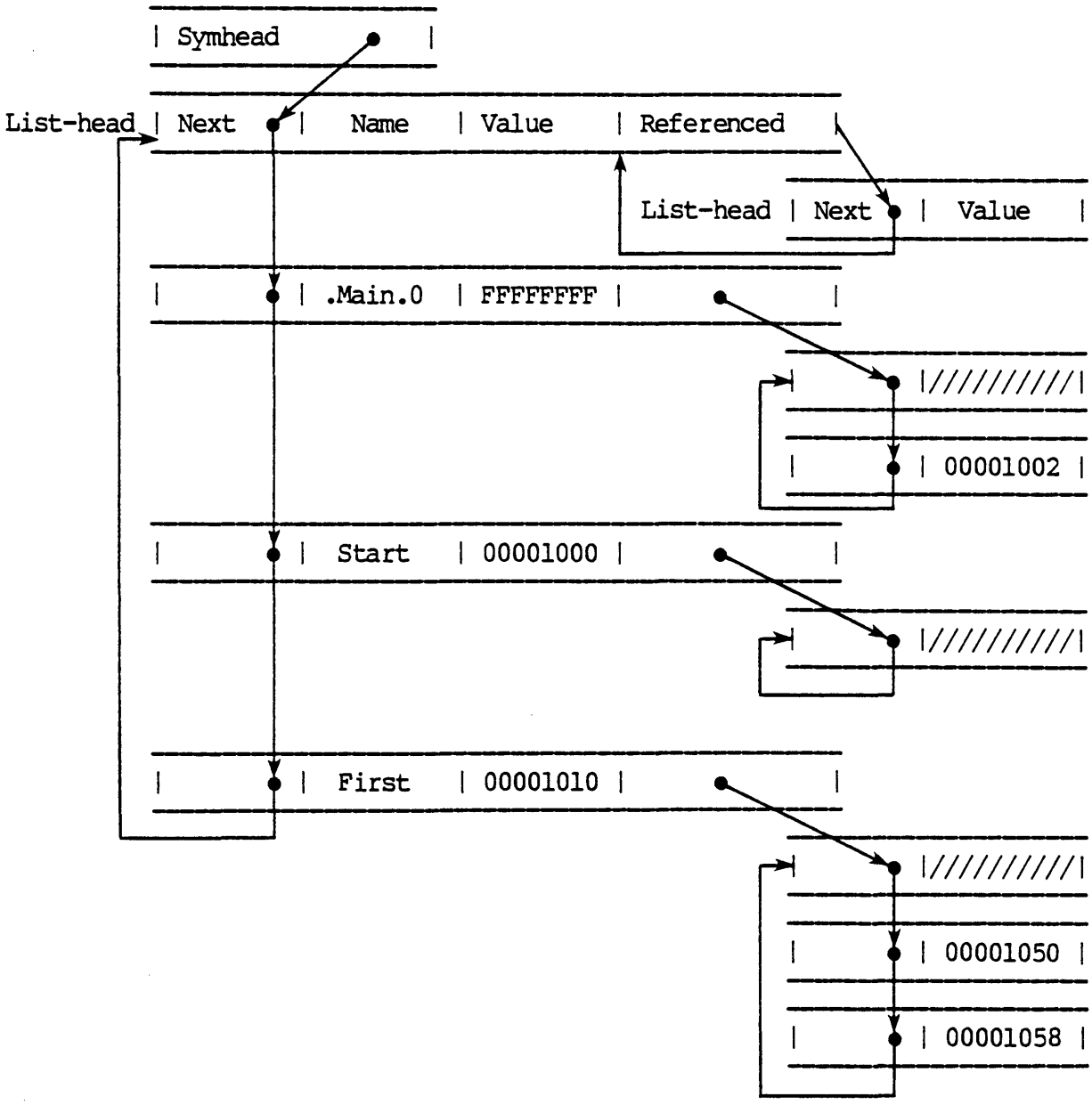


Fig. 4.3 Pictorial Representation of LINK Symbol

The second record class consists of types 'RQ', 'Rl', 'R5', and 'R6'. These result in a symbol table entry being built if one does not yet exist and cause the value field of the symbol table entry to become defined.

The third record class consists of only record type 'R7'. An occurence of a type 'R7' record causes a symbol table entry to be built if one does not yet exist. An entry is then made into a list of locations of undefined symbols that is associated with the symbol table entry.

After all of the relocatable modules are read in, LINK procedes to add the value field of each symbol table entry to all of the locations where it is referenced as specified in the associated list of references to undefined symbols.

After the undefined references have been resolved, the executable image file is created.

4.1 BIT MAP

The executable image file has a record length of 1024 bytes. The first record of this file contains bit maps each of which is 256 bytes long. These bit maps are:

1. Disk Memory Correspondence.
2. Initial Memory Allocation.
3. Shared Memory Page.
4. NOT USED.

These maps are arranged as shown in figure 4.4.

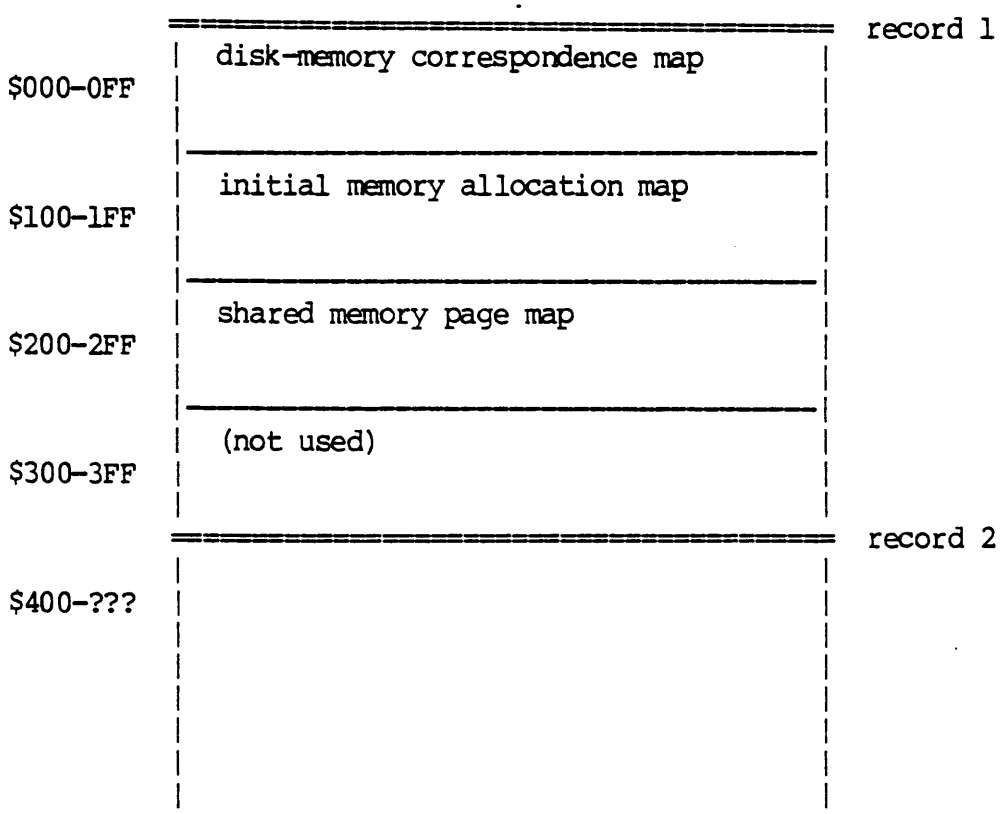


Fig. 4.4 Image File Format

4.1.1 Disk Memory Correspondence

The first bit map defines the relationship between the 2-Mbyte logical address space and the records of the executable image file. Each bit of this map corresponds to 1024 bytes of memory. If the bit is set, the next record from the executable image file is read into memory at the location corresponding to the bit position.

4.1.2 Initial Memory Allocation

The second bit map indicates which pages of memory should be allocated to the new process. Again, each bit corresponds to 1024 bytes in the 2-Mbyte address space. The operating system allocates memory to 4096-byte pages. Therefore, the setting of any of the bits on the page causes the full 4096 bytes to be allocated.

4.1.3 Shared Memory Page

The third bit map is the memory-protection bit map. The setting of any of the bits within the range corresponding to a 4096-byte page causes the full 4096 bytes to be write protected by the operating system. This process makes the page sharable because the contents cannot be changed.

There is space for a fourth bit map that is unused at this time and is assumed to be zero.

CHAPTER 5

LIBRARIES

LINK uses the content of a **library definition file** to resolve undefined references after all the user-specified files have been input.

5.1 DEFINING A LIBRARY

To define a library, you create a file containing a list of **equivalences.** An entry in this file is a symbolic name that can be referenced followed by the filename of an object module that defines the symbolic name.

You can create this file as a normal text file by using the VEW Program (read the Virtual Editing Window (VEW) User Reference Manual). The standard extension for a library definition file is .DEF. When referencing a file in an entry, you should enter the filename in uppercase and include the file extension.

A line that begins with an ampersand, @, indicates indirection through another library definition file. In this case, LINK responds as though the contents of the other file had been inserted at this point.

Figure 5.1 is an example of a library definition file.

```
... Symbolic Name ...|.......... File Name ..........
---------------------+-----------------------------------
        1         2  |         3         4         5
12345678901234567890 | 12345678901234567890123456789
---------------------+-----------------------------------
RRROOO               | _DS0/PRTLIB/INIT.MRL
RRR001               | _DS0/PRTLIB/EXIT.MRL
_PUTSTRING           | _DS0/PRTLIB.SVC/PUTSTR.MRL
@_DS0/RLIB/ANOTHER.DEF
```

Fig. 5.1 Library Definition File

5.2 PRELINK

When LINK is ready to use a file, it must translate the name of the file to get the File Control Block (FCB) number assigned to the file. You can often save time during the linking process by performing this translation as a separate step. The PRELINK Program does this. Execute PRELINK by typing either of the following command-line character strings and then striking [RETRN]:

```
> prelink linklib.def

    or

> prelink _ds0/syslib/linklib.def
```

PRELINK assumes the file has a .DEF extension if no file extension is specified in the PRELINK command-line character string.

PRELINK adds another column to the list of equivalences in the .DEF file. This column consists of the FCB numbers of the specified files. See figure 5.2 for an example of a library definition file after processing by PRELINK.

```
... Symbolic Name ...|.......... File Name ...........|.....FCB Number.....
---------------------+-----------------------------------+--------------------
        1         2  |         3         4         5   |         6         7
12345678901234567890 | 12345678901234567890123456789   | 12345678901234567890
---------------------+-----------------------------------+--------------------
RRROOO               | _DS0/PRTLIB/INIT.MRL              | _DS0//#96.1
RRR001               | _DS0/PRTLIB/EXIT.MRL              | _DS0//#97.3
_PUTSTRING           | _DS0/PRTLIB.SVC/PUTSTR.MRL        | _DS0//#106.1
@_DMO/RLIB/ANOTHER.DEF
```

Fig. 5.2 Library Definition File After Processing By Prelink

Note: PRELINK creates a new version of the library definition file in the same directory as the original file. To

conserve disk space, you may want to purge old library definition files after running PRELINK.

5.3 REFERENCING LIBRARIES

5.3.1 Default Library

LINK always references the file LINKLIB.DEF, found in directory /SYSLIB/ on the system disk. During initialization, LINK reads the contents of this file to make the names of library routines available to programs.

5.3.2 Other Libraries

To reference another library definition file, use the :LIBRARY switch on the command line. For example:

```
> link one,two,three :library=test
```

The foregoing command tells LINK to link files ONE, TWO, and THREE and reference TEST.DEF as a library file.

APPENDIX A

TROUBLESHOOTING

A.1 LINK DIAGNOSTIC MESSAGES

These are the diagnostic message associated with LINK:

Display Undefined Symbols

Checksum Error In Reading Relocatable Module

Unexpected End of File While Reading Relocatable Module

Address Of Undefined Reference Accesses Unallocated Disk Page

Open Error (WMCS diagnostic message number)

Read Error (WMCS diagnostic message number)

Write Error (WMCS diagnostic message number)

Close Error (WMCS diagnostic message number)

Delete Error (WMCS diagnostic message number)

A.2 MISCELLANEOUS ERRORS

Certain conditions lead to errors that do not give the preceding error messages. Some of these conditions are discussed in this section.

A.2.1 File Empty Or Does Not Exist

If a specified file is empty or does not exist, the message 'Unexpected end of file encountered while reading' is displayed followed by the name of the file. The solution is to use the TYPE Command to display the contents of the file. Be certain to specify the filename given in the diagnostic message.

A.2.2 Invalid Symbol Address

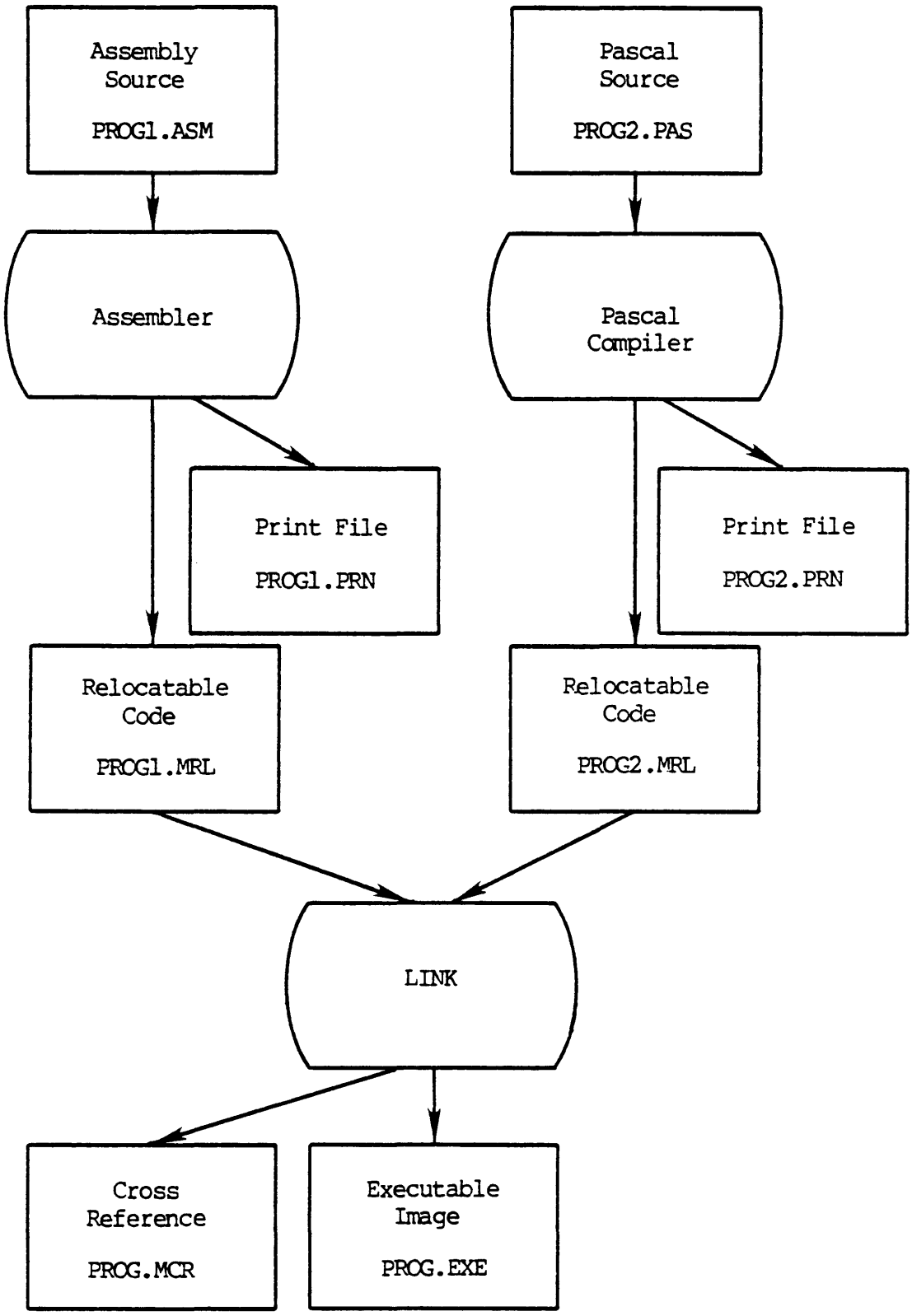
LINK does not check the validity of symbol addresses. If for some reason, a symbol has an offset beyond the end of its module, this error can happen to be caught during the process of adding the actual address to the location of an undefined reference. The solution is to examine the source modules for unreasonable address offsets.

APPENDIX B

EXAMPLE OF PROGRAM DEVELOPMENT

The diagram on the next page shows how files produced by the assembler and higher-level language compilers may be linked to form an executable image. A number of relocatable code (.MRL) files can be linked in one operation.

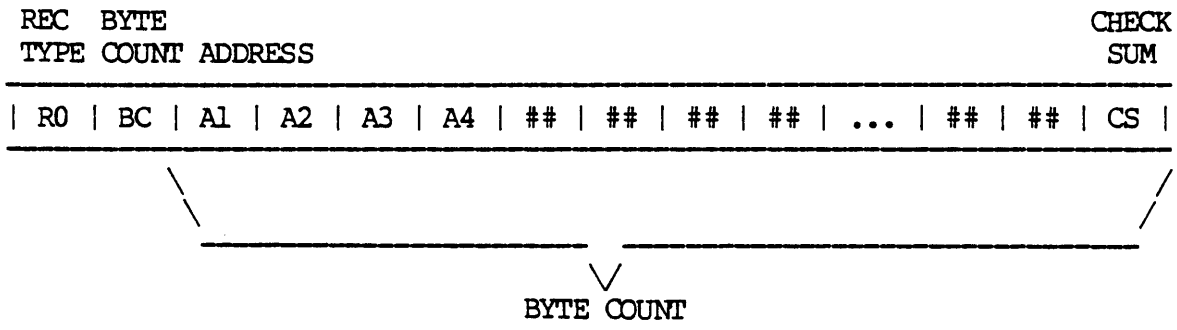
EXAMPLE OF PROGRAM DEVELOPMENT



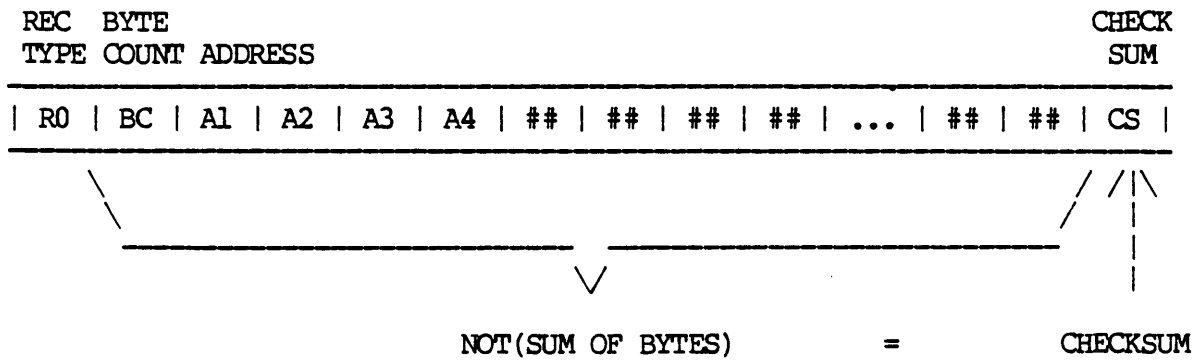
APPENDIX C

RELOCATABLE MODULE FORMAT

The relocatable source file contains a series of modules terminated by an R9 record. A module is a series of record types R2-R7 preceded either by an R0 or an R1 type record and terminated by an R8 type record.



The **byte count** is the number of bytes not including the byte count itself.



The **checksum** byte is the ones complement of the sum of the bytes. This form of checksum was chosen to be compatible with Motorola.

REC TYPE	BYTE COUNT	ADDRESS	PROGRAM NAME	CHECK SUM
R0	BC	A1 A2	A3 A4 ## ## ## ## ... ## ##	CS

Delimits the sequence of records pertaining to the main program. Otherwise, this type is treated the same as type 6.

REC TYPE	BYTE COUNT	ADDRESS	MODULE NAME	CHECK SUM
R1	BC	A1 A2	A3 A4 ## ## ## ## ... ## ##	CS

Delimits the sequence of records pertaining to a subroutine. Otherwise, this type is treated the same as type 6.

REC TYPE	BYTE COUNT	ADDRESS	CODE BLOCK	CHECK SUM
R2	BC	A1 A2	A3 A4 ## ## ## ## ... ## ##	CS

REC TYPE	BYTE COUNT	ADDRESS	LABEL	CHECK SUM
R3	BC	A1 A2	A3 A4 ## ## ## ## ... ## ##	CS

Local constant:

The label is defined to have the value of the address field.

REC TYPE	BYTE COUNT	ADDRESS	LABEL	CHECK SUM
R4	BC	A1 A2	A3 A4 ## ## ## ## ... ## ##	CS

Local label:

The label is defined to have the value of the address field plus the relocation constant.

REC TYPE	BYTE COUNT	ADDRESS	LABEL							CHECK SUM
R5	BC	A1	A2	A3	A4	##	##	##	##	...	##	##	CS

Global constant:

The label is defined to have the value of the address field.

REC TYPE	BYTE COUNT	ADDRESS	LABEL							CHECK SUM
R6	BC	A1	A2	A3	A4	##	##	##	##	...	##	##	CS

Global label:

The label is defined to have the value of the address field plus the relocation constant.

REC TYPE	BYTE COUNT	ADDRESS	LABEL							CHECK SUM
R7	BC	A1	A2	A3	A4	##	##	##	##	...	##	##	CS

The address field of this record is the location of the reference to the undefined label.

REC TYPE	BYTE COUNT	ADDRESS				CHECK SUM
R8	BC	A1	A2	A3	A4	CS

The foregoing kind of record marks the end of a module. This record contains the amount of space used by the module. The address field from the record is added to the current location pointer to form the address of the first free location past the end of the module. The address of the first free location past the end of the module then becomes the new location pointer.

RELOCATABLE MODULE FORMAT

REC TYPE	BYTE COUNT	ADDRESS				CHECK SUM
R9	BC	A1	A2	A3	A4	CS

The foregoing kind of record marks the end of file. The address field is undefined.

APPENDIX D

CROSS REFERENCE MAP LISTING

The following pages display a cross reference map listing from the sample program TEST.

CROSS REFERENCE MAP LISTING

WICAT Link Editor - Version 2.2

Linker map :

```
File : TEST.MRL                                  Base - Pure=  1000 Impure=100000
       .START.                 Module    1000                 6              0
       INCREMENT1              Module    1006                12              0
       TEST0                   Module    1018                26              0
       $PASCAL-GLOBAL$-2       Module  100000                 0             4C
                                                Used - Pure=   3E Impure=    4C
File : _DIO/PRTLIB/EXIT.MRL                      Base - Pure=  103E Impure=10004C
       RRR001                  Label     103E                40              0
                                                Used - Pure=   40 Impure=     0
File : _DIO/PRTLIB/INIT.MRL                      Base - Pure=  107E Impure=10004C
       RRR000                  Label     107E               126              0
                                                Used - Pure=  126 Impure=     0
File : _DIO/PRTLIB/RTGLOBAL.MRL                  Base - Pure=  11A4 Impure=10004C
       RTGLOBAL                Module  10004C
       HEAP                    Module  10005C                 0             20
                                                Used - Pure=    0 Impure=    20
File : _DIO/PRTLIB/REWRITE.MRL                   Base - Pure=  11A4 Impure=10006C
       RRR011                  Label     11A4                C4              0
                                                Used - Pure=   C4 Impure=     0
File : _DIO/PRTLIB/RESET.MRL                     Base - Pure=  1268 Impure=10006C
       RRR010                  Label     1268                EA              C
                                                Used - Pure=   EA Impure=     0
File : _DIO/PRTLIB/FBINIT.MRL                    Base - Pure=  1352 Impure=10006C
       RRR009                  Label     1352                4C              0
                                                Used - Pure=   4C Impure=     0
File : _DIO/PRTLIB/SHOWIO.MRL                    Base - Pure=  139E Impure=10006C
       SHOWIO                  Label     139E                60              0
                                                Used - Pure=   60 Impure=     0
File : _DIO/PRTLIB/ERROR.MRL                     Base - Pure=  13FE Impure=10006C
       ERROR                   Label     13FE                52              0
                                                Used - Pure=   52 Impure=     0
File : _DIO/PRTLIB/HANG.MRL                      Base - Pure=  1450 Impure=10006C
       HANG                    Label     1450                48              0
                                                Used - Pure=   48 Impure=     0
File : _DIO/PRTLIB/WHEX.MRL                      Base - Pure=  1498 Impure=10006C
       RRR039                  Label     1498                98              0
                                                Used - Pure=   98 Impure=     0
File : _DIO/PRTLIB/PUT.MRL                       Base - Pure=  1530 Impure=10006C
       RRR022                  Label     1530               17C              0
                                                Used - Pure=  17C Impure=     0
File : _DIO/PRTLIB/GET.MRL                       Base - Pure=  16AC Impure=10006C
       RRR021                  Label     16AC                7C              0
                                                Used - Pure=   7C Impure=     0
File : _DIO/PRTLIB/MBVALID.MRL                   Base - Pure=  1728 Impure=10006C
       RRR020                  Label     1728               1C0              0
                                                Used - Pure=  1C0 Impure=     0
```

```
File : _DIO/PRTLIB/CLOSE.MRL                    Base - Pure=  18E8 Impure=10006C
  ** Unable to open by FCB!
       RRR019                Label       18E8
       RRR019A               Label       18E8                  F4             0
                                                 Used - Pure=    F4 Impure=     0

       _HEAP                 Symbol    10006C                   0             0

Image filename = TEST.EXE
Total image size (all sizes given in hexadecimal) =
       9DC bytes pure code.
        6C bytes impure code.
         0 bytes reserved for dynamic space.
      1000 bytes reserved for stack.
```

CROSS REFERENCE MAP LISTING

WICAT Link Editor - Version 2.2

Symbol Cross Reference

$PASCAL-GLOBAL$-2	100000	100E	101A	1020	102C			
.START.	1000							
ERROR	13FE	1230	131E	13B0	13D2	145E	156A	16E6
		18B0	19A4					
HANG	1450	123E	132C	1578	16F4	18BE	19B2	
HEAP	10005C	1150						
INCREMENT1	1006	1034						
RRR000	107E	1026						
RRR001	103E	103A	1454					
RRR009	1352	1102	1130	198A				
RRR010	1268	1140						
RRR011	11A4	1112						
RRR019	18E8	11BA	127E					
RRR019A	18E8							
RRR020	1728	16C6						
RRR021	16AC	12E4						
RRR022	1530	1514						
RRR039	1498	13C8						
RTGLOBAL	10004C	1044	105A	1118	1146	13B6	1408	
SHOWIO	139E	1238	1326	1572	16EE	18B8	19AC	
TEST0	1018	1002						
_HEAP	10006C	115A						

APPENDIX E

BIT MAPS AND RECORDS

This appendix contains bit maps and records for the sample program TEST. These bit maps and records are a hexadecimal dump of the image file for the TEST program.

FILE: _DIO/TOM.CDS/TEST.EXE.1

	00000000 0E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	First
	00000070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Bit
	00000080 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Map
	00000090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000000A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000000B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000000C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000000D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000000E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000000F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000100 0E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Second
	00000180 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Bit
	00000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Map
	000001A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000001B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000001C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000001D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000001E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000001F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0F	
Record		
0	00000200 0E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000210 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000220 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000230 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	00000260 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Third
	00000270 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Bit
	00000280 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Map
	00000290 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000002A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000002B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000002C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
	000002D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	

```
| 000002E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 000002F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
|                                                            ——
|                                                            ——
| 00000300  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 00000310  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 00000320  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 00000330  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 00000340  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 00000350  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 00000360  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  | Fourth
| 00000370  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  | Bit
| 00000380  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  | Map
| 00000390  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 000003A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 000003B0  00 00 00 00 00 00 00 00 00 00 00 00 00 C0 00 00  |
| 000003C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 000003D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 000003E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
| 000003F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |
——                                                          ——
——
| 00000400  4E F9 00 00 10 18 4E 56 FF FC 2D 0D 52 79 00 10 NyNV|-Ry
| 00000410  00 4A 2A 5E 4E 5E 4E 75 2F 3C 00 10 00 06 2F 3C J*^N^Nu/</<
| 00000420  00 10 00 28 4E B9 00 00 10 7E 42 79 00 10 00 4A (N9~ByJ
| 00000430  2A 4E 4E B9 00 00 10 06 4E B9 00 00 10 3E 4E 56 *NN9N9>NV
| 00000440  FF F8 20 7C 00 10 00 50 20 50 2F 28 00 08 42 A7 x |P P/(B'
| 00000450  48 6E FF FC 4E 40 00 03 20 7C 00 10 00 4C 20 50 Hn|N@ |L P
| 00000460  2F 28 00 08 42 A7 48 6E FF FC 4E 40 00 03 42 A7 /(B'Hn|N@B'
| 00000470  42 A7 42 A7 48 6E FF FC 4E 40 00 10 60 FE 28 4F B'B'Hn|N@`~(O
| 00000480  42 81 20 0F 0C 80 00 1F FF EC 6E 44 20 2F 00 0C B lnD /
| 00000490  0C 80 00 00 0F FF 6E 38 20 6C 00 10 D1 C0 B1 FC n8 lQ@1|
| 000004A0  00 1F FF FF 6E 2A 4A 80 6F 26 0C 20 00 20 66 04 n*Jo&  f
| 000004B0  53 80 60 F2 42 28 00 01 53 80 6F 0A 0C 20 00 20 S`rB(So
| 000004C0  66 F6 53 80 60 02 53 88 48 68 00 01 52 81 60 D6 fvS`SHhR`V
| 000004D0  3F 01 48 6F 00 02 42 A7 42 A7 42 A7 2C 4F 2F 2C ?HoB'B'B',O/,
| 000004E0  00 08 2F 2C 00 04 2F 2C 00 00 4E 56 00 00 48 E7 /,/,NVHg
| 000004F0  FF FC 2F 2E 00 0C 42 67 3F 3C 00 01 3F 3C 00 01 |/.Bg?<?<
|
|
| 00000500  4E B9 00 00 13 52 2F 2E 00 0C 48 7A 00 80 42 67 N9R/.HzBg
| 00000510  4E B9 00 00 11 A4 20 7C 00 10 00 4C 20 AE 00 0C N9$ |L .
| 00000520  2F 2E 00 08 42 67 3F 3C 00 01 3F 3C 00 01 4E B9 /.Bg?<?<N9
| 00000530  00 00 13 52 2F 2E 00 08 48 7A 00 5E 42 67 4E B9 R/.Hz^BgN9
| 00000540  00 00 12 68 20 7C 00 10 00 50 20 AE 00 08 20 7C h |P . |
| 00000550  00 10 00 5C 21 48 00 00 21 7C 00 10 00 6C 00 04 \!H!|l
| 00000560  20 28 00 04 06 80 00 00 03 FF 02 80 00 1F FC 00  (|
| 00000570  90 A8 00 04 21 40 00 08 42 A8 00 0C 2D 6E 00 04 (!@B(-n
| 00000580  00 0C 4C DF 3F FF 4E 5E 50 8F 4E 75 53 59 53 24 L_?N^PNuSYS$
| 00000590  4F 55 54 50 55 54 00 00 53 59 53 24 49 4E 50 55 OUTPUTSYS$INPU
| 000005A0  54 00 00 00 4E 56 00 00 48 E7 80 C0 20 6E 00 0E TNVHg@ n
| 000005B0  4A 28 00 0D 67 08 2F 08 4E B9 00 00 18 E8 31 6E J(g/N9hln
```

```
      | 000005C0  00 08 00 0E 42 28 00 01 4A 68 00 18 56 E8 00 16 B(JhVh
      | 000005D0  22 6E 00 0A 20 09 4A 19 66 FC 90 89 44 80 53 80 "n Jf|DS
      | 000005E0  2F 2E 00 0A 2F 00 2F 3C 00 00 00 02 3F 28 00 14 /.//</?(
      | 000005F0  42 67 42 A7 2F 3C FF FF FF FF 2F 3C FF FF FF FF BgB'/</<
Record|
  1   |
      | 00000600  48 68 00 08 48 68 00 04 4E 40 00 00 4A A8 00 04 HhHhN@J(
      | 00000610  57 E8 00 0D 66 14 2D 6E 00 04 00 0E 4C DF 03 01 Whf-nL_
      | 00000620  4E 5E DF FC 00 00 00 0A 4E 75 48 7A 00 16 4E B9 N^_|NuHzN9
      | 00000630  00 00 13 FE 2F 08 4E B9 00 00 13 9E 4E F9 00 00 ~/N9Ny
      | 00000640  14 50 52 52 52 30 31 31 20 2D 2D 20 52 45 57 52 PRRR011 -- REWR
      | 00000650  49 54 45 20 2D 2D 20 20 43 52 45 41 54 20 46 41 ITE --  CREAT FA
      | 00000660  49 4C 45 44 0D 0A 00 00 4E 56 00 00 48 E7 80 C0 ILEDNVHg@
      | 00000670  20 6E 00 0E 4A 28 00 0D 67 08 2F 08 4E B9 00 00  nJ(g/N
      | 00000680  18 E8 31 6E 00 08 00 0E 42 28 00 16 22 6E 00 0A hlnB("n
      | 00000690  20 09 4A 19 66 FC 90 89 44 80 53 80 2F 2E 00 0A  Jf|DS/.
      | 000006A0  2F 00 2F 3C 00 00 00 01 3F 28 00 14 42 67 48 68 //<?(BgHh
      | 000006B0  00 08 48 68 00 04 4E 40 00 02 4A A8 00 04 66 3C HhN@J(f<
      | 000006C0  50 E8 00 0D 42 28 00 02 42 28 00 03 50 E8 00 00 PhB(B(Ph
      | 000006D0  21 7C FF FF FF FF 00 10 21 7C 00 00 00 01 00 1C !|!|
      | 000006E0  2F 08 4E B9 00 00 16 AC 2D 6E 00 04 00 0E 4C DF /N9,-nL_
      | 000006F0  03 01 4E 5E DF FC 00 00 00 0A 4E 75 0C A8 00 00 N^_|Nu(
      |
      |
      | 00000700  00 85 00 04 66 12 42 28 00 0D 50 E8 00 02 50 E8 fB(PhPh
      | 00000710  00 03 42 28 00 00 60 D0 48 7A 00 16 4E B9 00 00 B(`PHzN9
      | 00000720  13 FE 2F 08 4E B9 00 00 13 9E 4E F9 00 00 14 50 ~/N9NyP
      | 00000730  52 52 52 30 31 30 20 2D 2D 20 52 45 53 45 54 20 RRR010 -- RESET
      | 00000740  2D 2D 20 20 4F 50 45 4E 20 46 41 49 4C 45 44 0D --  OPEN FAILED
      | 00000750  0A 00 4E 56 00 00 2F 08 20 6E 00 0E 42 98 42 98 NV/ nBB
      | 00000760  42 98 42 98 42 98 42 98 42 98 42 98 20 6E 00 0E BBBBBB n
      | 00000770  31 6E 00 0C 00 18 31 6E 00 0A 00 14 50 E8 00 02 1nlnPh
      | 00000780  4A 6E 00 08 56 E8 00 0C 56 E8 00 03 2D 6E 00 04 JnVhVh-n
      | 00000790  00 0E 20 5F 4E 5E DF FC 00 00 00 0A 4E 75 4E 56  _N^_|NuNV
      | 000007A0  00 00 48 E7 FF FC 20 6E 00 08 48 7A 00 3A 4E B9 Hg| nHz:N9
      | 000007B0  00 00 13 FE 22 7C 00 10 00 4C 22 51 2F 09 2F 28 ~"|L"Q//(
      | 000007C0  00 04 3F 3C FF FF 4E B9 00 00 14 98 48 7A 00 28 ?<N9Hz(
      | 000007D0  4E B9 00 00 13 FE 2D 6E 00 04 00 08 4C DF 3F FF N9~-nL_?
      | 000007E0  4E 5E 58 8F 4E 75 49 2F 4F 20 52 45 53 55 4C 54 N^XNuI/O RESULT
      | 000007F0  20 3D 20 00 00 00 20 48 45 58 0D 0A 00 00 4E 56  =  HEXNV
      ——
      ——
      | 00000800  FF F8 48 E7 FF FC 20 7C 00 10 00 4C 20 50 42 86 xHg| |L PB
      | 00000810  22 6E 00 08 4A 19 67 04 52 86 60 F8 2F 28 00 08 "nJgR`x/(
      | 00000820  2F 3C FF FF FF FF 42 A7 2F 3C FF FF FF FF 2F 2E /<B'/</.
      | 00000830  00 08 2F 06 48 6E FF F8 48 6E FF FC 4E 40 00 05 /HnxHn|N@
      | 00000840  2D 6E 00 04 00 08 4C DF 3F FF 4E 5E 58 8F 4E 75 -nL_?N^XNu
      | 00000850  20 00 4E F9 00 00 10 3E 48 7A 00 0A 4E B9 00 00  Ny>HzN9
      | 00000860  13 FE 60 FE 45 58 45 43 55 54 49 4E 47 20 22 4A ~`~EXECUTING "J
      | 00000870  55 4D 50 20 54 4F 20 48 45 52 45 22 20 4C 4F 4F UMP TO HERE" LOO
      | 00000880  50 20 49 4E 20 50 41 53 43 41 4C 20 52 55 4E 54 P IN PASCAL P~~~
      | 00000890  49 4D 45 0D 0A 00 00 00 4E 56 FF EC 48 E7 F0 C0 IMENV1Hgp@
```

```
| 000008A0  20 6E 00 0E 70 14 22 4E 13 3C 00 20 53 80 66 F8  np"N< Sfx
| 000008B0  42 83 24 2E 00 0A 22 4E 22 02 E8 8A 02 81 00 00 B$."N"h
| 000008C0  00 0F 0C 01 00 09 6F 02 5E 01 06 01 00 30 13 01 o^0
| 000008D0  52 83 4A 82 66 E2 32 2E 00 08 48 C1 6A 02 22 03 RJfb2.HAj"
| 000008E0  B2 83 6C 0C 22 4E 20 01 13 3C 00 2A 53 80 6E F8 21"N <*Snx
| 000008F0  4A 81 67 28 B6 81 6A 08 11 7C 00 20 00 20 60 04 Jg(6j|  `
|
|
| 00000900  11 59 00 20 11 7C 00 01 00 00 11 7C 00 01 00 01 Y ||
| 00000910  2F 08 4E B9 00 00 15 30 53 81 60 D4 2D 6E 00 04 /N90S`T-n
| 00000920  00 0E 4C DF 03 0F 4E 5E DF FC 00 00 00 0A 4E 75 L_N^_|Nu
| 00000930  4E 56 FF EE 2F 08 20 6E 00 08 4A 28 00 00 67 1E NVn/ nJ(g
| 00000940  61 00 00 92 52 A8 00 10 42 28 00 00 42 28 00 01 aR(B(B(
| 00000950  2D 6E 00 04 00 08 20 5F 4E 5E 58 8F 4E 75 48 7A -n _N^XNuHz
| 00000960  00 1C 60 04 48 7A 00 4C 4E B9 00 00 13 FE 2F 08 `HzLN9~/
| 00000970  4E B9 00 00 13 9E 4E F9 00 00 14 50 52 52 52 30 N9NyPRRR0
| 00000980  32 32 20 2D 2D 20 50 55 54 20 2D 2D 20 20 46 49 22 — PUT —  FI
| 00000990  4C 45 20 43 4F 4D 50 4F 4E 45 4E 54 20 44 41 54 LE COMPONENT DAT
| 000009A0  41 20 49 53 20 4E 4F 54 20 56 41 4C 49 44 0D 0A A IS NOT VALID
| 000009B0  00 00 52 52 52 30 32 32 20 20 2D 2D 20 50 55 54 RRR022  — PUT
| 000009C0  20 2D 2D 20 20 57 52 49 54 45 20 46 41 49 4C 45  — WRITE FAILE
| 000009D0  44 0D 0A 00 48 E7 80 60 42 AE FF FC 4A 28 00 0C DHg`B.|J(
| 000009E0  67 08 2D 7C 00 00 00 03 FF FC 4A 68 00 18 67 00 g-||Jhg
| 000009F0  00 8A 42 6E FF EE 3D 68 00 18 FF F0 42 6E FF F2 Bnn=hpBnr
Record|
  2   |
      | 00000A00  3D 68 00 14 FF F4 20 28 00 10 90 A8 00 1C 52 80 =ht ((R
      | 00000A10  B0 AE FF EE 5C EE FF FA C0 E8 00 14 43 E8 00 20 0.n\nz@hCh
      | 00000A20  24 49 D5 C0 20 2E FF F2 60 02 14 D9 51 C8 FF FC $IU@ .r`YQH|
      | 00000A30  4A 2E FF FA 67 00 00 3E 2F 28 00 08 2F 28 00 1C J.zg>/(/(
      | 00000A40  2F 2E FF FC 2F 3C FF FF FF FF 48 68 00 20 20 2E /.|/<Hh  .
      | 00000A50  FF F2 D1 97 2F 2E FF EE 48 6E FF F6 48 68 00 04 rQ/.nHnvHh
      | 00000A60  4E 40 00 05 4A A8 00 04 66 00 FE FA 20 2E FF EE N@J(f~z .n
      | 00000A70  D1 A8 00 1C 4C DF 06 01 4E 75 2F 28 00 08 2F 28 Q(L_Nu/(/(
      | 00000A80  00 10 2F 2E FF FC 2F 3C FF FF FF FF 48 68 00 20 /.|/<Hh
      | 00000A90  2F 3C 00 00 00 01 48 6E FF F6 48 68 00 04 4E 40 /<HnvHhN@
      | 00000AA0  00 05 4A A8 00 04 67 CC 60 00 FE BA 4E 56 00 00 J(gL`~:NV
      | 00000AB0  2F 08 20 6E 00 08 4A 28 00 02 66 24 4A 28 00 00 / nJ(f$J(
      | 00000AC0  66 08 2F 08 4E B9 00 00 17 28 42 28 00 00 52 A8 f/N9(B(R(
      | 00000AD0  00 10 2D 6E 00 04 00 08 20 5F 4E 5E 58 8F 4E 75 -n _N^XNu
      | 00000AE0  48 7A 00 16 4E B9 00 00 13 FE 2F 08 4E B9 00 00 HzN9~/N9
      | 00000AF0  13 9E 4E F9 00 00 14 50 52 52 52 30 32 31 20 20 NyPRRR021
      |
      |
      | 00000B00  2D 2D 20 47 45 54 20 2D 2D 20 20 45 4F 46 20 57 — GET —  EOF W
      | 00000B10  41 53 20 54 52 55 45 20 50 52 49 4F 52 20 54 4F AS TRUE PRIOR TO
      | 00000B20  20 47 45 54 0D 0A 00 00 4E 56 FF EE 2F 08 20 6E  GETNVn/ n
      | 00000B30  00 08 4A 28 00 00 66 00 00 48 4A 28 00 02 66 00 J(fHJ(f
      | 00000B40  00 40 61 00 00 4A 4A 28 00 0C 67 34 42 28 00 03 @aJJ(g4B(
      | 00000B50  0C 28 00 0D 00 20 67 24 0C 28 00 0A 00 20 67 1C ( g$( g
      | 00000B60  0C 28 00 0B 00 20 67 14 0C 28 00 0C 00 20 67 0C ( g( g
      | 00000B70  4A 28 00 20 66 0A 11 7C 00 20 00 20 50 E8 00 03 J( f|  Ph
```

```
| 00000B80  2D 6E 00 04 00 08 20 5F 4E 5E 58 8F 4E 75 48 E7 -n _N^XNuHg
| 00000B90  80 60 42 AE FF FC 4A 28 00 0C 67 08 2D 7C 00 00 `B.|J(g-|
| 00000BA0  00 03 FF FC 4A 68 00 18 67 00 00 9C 42 6E FF F2 |JhgBnr
| 00000BB0  3D 68 00 14 FF F4 42 6E FF EE 3D 68 00 18 FF F0 =htBnn=hp
| 00000BC0  20 28 00 10 90 A8 00 1C 6B 08 B0 AE FF EE 6D 00  ((k0.n
| 00000BD0  00 4E 21 68 00 10 00 1C 2F 28 00 08 2F 28 00 1C N!h/(/(
| 00000BE0  2F 2E FF FC 2F 3C FF FF FF FF 48 68 00 20 20 2E /.|/<Hh  .
| 00000BF0  FF F2 D1 97 2F 2E FF EE 48 6E FF F8 48 68 00 04 rQ/.nHnxHh
|---
|---
| 00000C00  4E 40 00 04 20 28 00 04 67 0A 0C 80 00 00 00 8C N@ (g
| 00000C10  66 00 00 98 31 6E FF FA 00 1A 42 A8 00 04 61 00 flnzB(a
| 00000C20  00 66 4A 2E FF F6 57 E8 00 00 67 10 21 7C 00 00 fJ.vWhg!|
| 00000C30  00 8C 00 04 50 E8 00 02 60 06 12 DA 51 C8 FF FC Ph`ZQH|
| 00000C40  4C DF 06 01 4E 75 2F 28 00 08 2F 28 00 10 2F 2E L_Nu/(/(/.
| 00000C50  FF FC 2F 3C FF FF FF FF 48 68 00 20 2F 3C 00 00 |/<Hh /<
| 00000C60  00 01 48 6E FF F8 48 68 00 04 4E 40 00 04 20 28 HnxHhN@ (
| 00000C70  00 04 57 E8 00 00 67 C8 0C 80 00 00 00 8C 57 E8 WhgHWh
| 00000C80  00 02 67 BC 60 24 20 28 00 10 90 A8 00 1C 52 80 g<`$ ((R
| 00000C90  B0 68 00 1A 5E EE FF F6 C0 E8 00 14 43 E8 00 20 0h^nv@hCh
| 00000CA0  24 49 D5 C0 20 2E FF F2 4E 75 48 7A 00 16 4E B9 $IU@ .rNuHzN9
| 00000CB0  00 00 13 FE 2F 08 4E B9 00 00 13 9E 4E F9 00 00 ~/N9Ny
| 00000CC0  14 50 52 52 52 30 32 30 20 20 2D 2D 20 4D 42 56 PRRR020  — MBV
| 00000CD0  41 4C 49 44 20 2D 2D 20 20 52 45 41 44 20 20 46 ALID —  READ  F
| 00000CE0  41 49 4C 45 44 0D 0A 00 4E 56 FF FC 48 E7 80 80 AILEDNV|Hg
| 00000CF0  20 6E 00 08 4A 28 00 01 66 06 4A 28 00 16 67 68  nJ(fJ(gh
|
|
| 00000D00  42 80 4A 28 00 0C 67 04 30 3C 00 03 2F 28 00 08 BJ(g0</(
| 00000D10  2F 28 00 1C 4A 68 00 18 66 04 2E A8 00 10 2F 00 /(Jhf.(/
| 00000D20  2F 3C FF FF FF FF 48 68 00 20 4A 68 00 18 67 20 /<Hh Jhg
| 00000D30  30 28 00 14 D1 97 4A 28 00 17 66 0C 20 28 00 10 0(QJ(f (
| 00000D40  90 A8 00 1C 2F 00 60 0E 3F 28 00 18 42 67 60 06 (/`?(Bg`
| 00000D50  2F 3C 00 00 00 01 48 6E FF FC 48 68 00 04 4E 40 /<Hn|HhN@
| 00000D60  00 05 4A A8 00 04 66 36 2F 28 00 08 42 A7 48 68 J(f6/(B'Hh
| 00000D70  00 04 4E 40 00 03 2F 08 3F 28 00 18 3F 28 00 14 N@/?(?(
| 00000D80  42 67 1F 68 00 0C 00 01 4E B9 00 00 13 52 2D 6E BghN9R-n
| 00000D90  00 04 00 08 4C DF 01 01 4E 5E 58 8F 4E 75 48 7A L_N^XNuHz
| 00000DA0  00 16 4E B9 00 00 13 FE 2F 08 4E B9 00 00 13 9E N9~/N9
| 00000DB0  4E F9 00 00 14 50 52 52 52 30 31 39 20 20 2D 2D NyPRRR019  —
| 00000DC0  20 43 4C 4F 53 45 20 2D 2D 20 20 57 52 49 54 45  CLOSE —  WRITE
| 00000DD0  20 20 46 41 49 4C 45 44 0D 0A 00 00 00 00 00 00   FAILED
| 00000DE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000DF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Record|
  3   |
| 00000E00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000E10  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000E20  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000E30  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000E40  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000E50  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
| 00000E60  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000E70  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000E80  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000E90  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000EA0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000EB0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000EC0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000ED0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000EE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000EF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
|
|
| 00000F00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000F10  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000F20  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000F30  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000F40  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000F50  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000F60  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000F70  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000F80  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000F90  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000FA0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000FB0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000FC0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000FD0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000FE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00000FF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
—
—
| 00001000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000010A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000010B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000010C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000010D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000010E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000010F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
|
|
| 00001100  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001120  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001130  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
        | 00001140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001150  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001160  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001170  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001180  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001190  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000011A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000011B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000011C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000011D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000011E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000011F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Record  |
  4     |
        | 00001200  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001210  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001220  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001230  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001240  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001250  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001260  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001270  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001280  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001290  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000012A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000012B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000012C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000012D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000012E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000012F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        |
        |
        | 00001300  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001310  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001320  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001330  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001340  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001350  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001360  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001370  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001380  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 00001390  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000013A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000013B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000013C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000013D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000013E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        | 000013F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        ——
        ——
        | 00001400  00 00 10 02 00 00 10 0E 00 00 10 1A 00 00 10 20
        | 00001410  00 00 10 26 00 00 10 2C 00 00 10 34 00 00 10 3A &,4:
```

```
| 00001420  00 00 10 44 00 00 10 5A 00 00 11 02 00 00 11 12 DZ
| 00001430  00 00 11 18 00 00 11 30 00 00 11 40 00 00 11 46 0@F
| 00001440  00 00 11 50 00 00 11 5A 00 00 11 BA 00 00 12 30 PZ:0
| 00001450  00 00 12 38 00 00 12 3E 00 00 12 7E 00 00 12 E4 8>~d
| 00001460  00 00 13 1E 00 00 13 26 00 00 13 2C 00 00 13 B0 &,0
| 00001470  00 00 13 B6 00 00 13 C8 00 00 13 D2 00 00 14 08 6HR
| 00001480  00 00 14 54 00 00 14 5E 00 00 15 14 00 00 15 6A T^j
| 00001490  00 00 15 72 00 00 15 78 00 00 16 C6 00 00 16 E6 rxFf
| 000014A0  00 00 16 EE 00 00 16 F4 00 00 18 B0 00 00 18 B8 nt08
| 000014B0  00 00 18 BE 00 00 19 8A 00 00 19 A4 00 00 19 AC >$,
| 000014C0  00 00 19 B2 FF FF FF FF 00 00 00 00 00 00 00 00 2
| 000014D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000014E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000014F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
|
|
| 00001500  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001510  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001520  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001530  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001540  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001550  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001560  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001570  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001580  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001590  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000015A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000015B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000015C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000015D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000015E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000015F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Record|
  5   |
      | 00001600  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 00001610  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 00001620  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 00001630  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 00001640  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 00001650  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 00001660  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 00001670  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 00001680  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 00001690  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 000016A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 000016B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 000016C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 000016D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 000016E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      | 000016F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      |
      |
```

```
| 00001700  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001710  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001720  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001730  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001740  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001750  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001760  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001770  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001780  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 00001790  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000017A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000017B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000017C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000017D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000017E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
| 000017F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

WICAT Systems, Inc.
Product-documentation Comment Form

We are constantly improving our documentation, and we welcome specific comments on this manual.

Document Title: _____

Part Number: _____

Your Position:

- ☐ Novice user
- ☐ Experienced user
- ☐ Applications programmer
- ☐ System manager
- ☐ Systems analyst
- ☐ Hardware technician

Questions and Comments **Page No.**

Briefly describe examples, illustrations, or information that you think should be added to this manual.

_____ _____

_____ _____

_____ _____

What would you delete from the manual and why?

_____ _____

_____ _____

_____ _____

What areas need greater emphasis?

_____ _____

_____ _____

_____ _____

List any terms or symbols used incorrectly.

_____ _____

_____ _____

_____ _____

First Fold

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 00028 OREM, UTAH

POSTAGE WILL BE PAID BE ADDRESSEE

WICAT Systems, Inc.
Attn: Corporate Communications
1875 S. State St.
Orem, UT 84058

Second Fold

Tape