digital

chain & execute utility programs

pdp15

digital equipment corporation

DEC-15-YWZB-DN2

CHAIN & EXECUTE

UTILITY PROGRAM

For additional copies, order No. DEC-15-YWZB-DN2 from Program Library, Digital Equipment Corporation, Maynard, Massachusetts, 01754 Price $3.00

First Printed MAY 1970

Revised SEPTEMBER 1970
Revised OCTOBER 1971

The material in this manual is intended for information purposes and is subject to change without notice. The latest revisions to this manual are indicated by a bar marked in the margin opposite the revised data.

PREFACE

The operation and use of the PDP-15 CHAIN & EXECUTE Utility Program is described in this manual. The information presented is valid for use in either the ADVANCED Software System (ADSS) or the Disk Operating System (DOS) environments. Differences between ADSS and DOS environments are indicated.

It was assumed, in the preparation of this manual, that the reader is familiar with the operation of the PDP-15 equipment and the contents of the software manual describing the features of the particular monitor system in use, that is:

a) for ADS users, PDP-15/20/30/40 ADVANCED Monitor Software System Manual, DEC-15-MR2B-D;

b) for DOS users, DOS Software System Users Manual, DEC-15-MRDA-D.

PDP-15 UTILITY PROGRAMS MANUAL, DEC-15-YWZB-D

The PDP-15 Utility Programs manual is comprised of a set of individual manuals, each of which describes the operation and use of a PDP-15 Utility Program. The manuals which make up the Utility Programs set are listed in the following Application Guide. In addition, the Application Guide also indicates the order number of each manual, and the specific PDP-15 Monitor Software Systems in which the program described may be used.

The Utility Manuals may be ordered either individually, by using the title and order number given with each manual, or as a set, by referencing "PDP-15 Utility Programs Manual, DEC-15-YWZB-D".

$$$$$

APPLICATION GUIDE

PDP-15 UTILITY PROGRAM MANUALS

PDP-15 Utility Program Manuals and the Application of Each

Title	Manual Order Number (DEC-15-YWZB_)	Applies to Monitor:		
		DOS	ADV	B/F
DDT Utility Program	DN1	✓	✓	✓
CHAIN & EXECUTE Utility Program	DN2	✓	✓	✓
SGEN ADVANCED Monitor	DN3		✓	
MTDUMP Utility Program	DN4	✓	✓	
PATCH Utility Program	DN5	✓	✓	✓
EDIT Utility Program	DN6	✓	✓	✓
UPDATE Utility Program	DN7	✓	✓	✓
LINKING LOADER	DN8	✓	✓	✓
PIP ADVANCED Monitor	DN9		✓	✓
SRCCOM Utility Program	DN11	✓	✓	✓
SGEN DOS Monitor	DN12	✓		
PIP DOS Monitor	DN13	✓		
Disk SAVE/RESTORE Programs	DN14	✓	✓	✓

CONTENTS

1. INTRODUCTION

The programs CHAIN and EXECUTE facilitate a user generated system of core overlays in a DOS, RSX or ADSS Monitor environment. This system of overlays consists of a resident main program, and may include other indicated resident routines, a resident blank COMMON storage area, and a set of subroutines which overlay each other as directed by the user. These subroutines are grouped into units called LINKS. Many or all LINKS may overlay each other, and several LINKS may overlay a larger LINK without overlaying each other. Cascading of sub-overlays is not limited.

A LINK is loaded into core when a subroutine within the LINK is called and remains resident until overlayed. A LINK's core image is not recorded or "swapped out" when it is overlayed. The same image is brought into core each time a LINK is loaded.

Subroutines are called and return control to the calling routine in the normal fashion, except when a calling routine will be overlayed by a called routine. In the latter case, no arguments can be conveyed via the call, and subroutine exit must be accomplished by calling another routine.

There is no imposed order in which routines must be called nor is there restriction of the routines callable by any routine.

The Overlay System, when recorded, is called an XCT file[1].

The program CHAIN is used to build an XCT file and the program EXECUTE supervises core residency during the execution of a CHAIN-built Overlay System.

For each call to a non-resident link, Execute must search the XCT file sequentially and circularly to find the called file. This can result in a larger percentage of the execution time begin spent in the search mode. The execution of complex overlay structures is speeded up appreciably when run under the RSX monitor since the Link Table entry for each external LINK contains the track sector address of that disk image.

2. THE OVERLAY SYSTEM

CHAIN-built Overlay Systems consist of 1) resident and non-resident code, 2) COMMON storage, and 3) a Link Table, all of which are described below.

The Overlay System is built onto an I/O device (XCT file) and the

[1]An "XCT file" is actually two files (see Appendix B).

core that will be required to run it need not be available (or existent) to build it.

Normally, I/O handlers are not included in an XCT file, but are loaded prior to execution per .DAT slot assignments and remain in core throughout a run. However, it is possible to include an I/O handler in a LINK by using a routine to make necessary .DAT slot assignment(s). This is illustrated in Example 9 (page 21).

In DOS systems, if I/O handlers are to be included in an overlay system, they must be transferred, using PIP, as single files to the user's UFD from the UFD<IOS>, before calling chain. During CHAIN operations the file (global) name of each handler must be cinluded in the proper LINK description command string.

2.1 Resident Routines

CHAIN requests a list of resident routines. All routines listed and any library routines they may call remain in core throughout a run. The first resident routine is the program to which initial control of the execution of the Overlay Systems is given by EXECUTE. All other routines in the Overlay System are either subroutines or co-routines. Resident subroutines may be called by any routine (resident or LINK component); therefore, the names assigned to resident routines must be unique throughout the Overlay System.

2.2 External Link Components

Each LINK consists of one or more subroutines, whose calling will result in the loading of the LINK when non-resident. These subroutines may be called by any routine, resident, or LINK component, and are called External Link Components. The names of External Link Components must also be unique throughout the Overlay System.

2.3 Internal Link Components

A LINK may also contain subroutines which are only to be called from within the LINK. These subroutines are called Internal Link Components, and may be used as internal components of other LINKS. The names of Internal Link Components must be unique only within their LINK.

2.4 Blank COMMON

Blank COMMON is universal COMMON. It is resident throughout a run and is available to all routines.

2.5 Labeled COMMON[1]

Labeled COMMON blocks declared by resident code are also resident and available to all routines. (This feature has been included to allow a resident initialized block to be available to all routines.)

A labeled COMMON block declared within a LINK with a block name that does not match a resident COMMON block name, is internal to the LINK. It is initialized (if BLOCK DATA) or cleared when the LINK is loaded, and overlayed when the LINK is overlayed. Using the same name for labeled COMMON blocks in different LINKS does not force an equivalence (see Example 3, page 15).

CHAIN provides a command option which causes LABELED COMMON BLOCKS of the same name to be allocated core only once, and to allow elements of LABELED COMMON to be referenced from any co-resident LINK. (Without this option, LABELED COMMON is considered internal to LINKS.) The option abbreviation accepted by CHAIN is "SAC" (Single Allocation of Commons). With the SAC option selected, a LABELED COMMON BLOCK is made a part of the first LINK listed in the Overlay Description that contains a declaration for the COMMON BLOCK.

This option facilitates an overlayable inter-LINK communication area. For example, SUB2, SUB3, and SUB4 each contains a

```
          COMMON /COMBLK/ A,B,C
```

statement. With the SAC option and the following Overlay Description, the LABELED COMMON BLOCK "COMBLK" would reside in SUB2, and would be available for communication between SUB3 and SUB4 (which overlay each other).

```
          SUB1:SUB2,SUB3
          SUB3:SUB4
```

CAUTION: Since there can be references to elements of a LABELED COMMON BLOCK from outside its LINK, a COMMON BLOCK may be overlayed while references to it still exist in other LINKS.

2.6 Restricting COMMON Areas (DOS Only)

CHAIN option "VTC" allows the user to restrict COMMON areas to bank boundaries. This is useful to the VT15 user who builds display files in COMMON, since the VT15 cannot cross bank boundaries directly (i.e.,

[1]See addendum (page 35)

13-bit addressing). There are two forms of the option and the giving of one cancels out the other form if it was given previously.

"VTC" option without names restricts all common areas to bank boundaries. The "VTC" option is delimited by being the first option in the command string or by a comma on the left, and by a comma or altmode on the right.

"VTC/NAM1,NAM2,...NAM3/" option with names restricts to bank boundaries only those COMMON areas named (Note: Blank common is .XX). More than one VTC option with names may be given in the command string and all names specified will be restricted. The option is delimited by being the first option in the command string or by a comma on the left, and a slash on the right. The name field within the option is delimited by a slash right after the "VTC" and the slash that terminates the option. Names in the list are separated by commas.

The VTC option will not restrict COMMON areas declared in BLOCK DATA SUBPROGRAMS.

The COMMON area is restricted to bank boundaries even if CHAIN is running in page relocation mode.

2.7 The Link Table

Also resident throughout a run is a table with an eleven-word entry for each External Link Component, a one-word buffer, and a transfer vector to an entry point to EXECUTE. This table is called the Link Table. Calls to External Link Components are loaded as JMS's to the appropriate Link Table entry. Transfer of control from the table is dependent on LINK residency (see Appendix A.)

3. SOURCE OF RELOCATABLE BINARY UNITS

When describing an Overlay System, the names of files are listed as containing either resident code of LINK components. The named files are read from the "User Program Device" (see paragraph 6). Library Routine names may also be used when describing resident code. The Library Routine names used are distinguished from file names by a "Library Indicator". The indicator is a pound sign (#) which may precede, follow, or appear within the name. The indicator is useful when it is desirable to force a Library Routine, which is not called by a resident routine to be included as resident code (for example, to "factor" a Library Routine out of all or many LINKS).

In the DOS system, CHAIN accepts library indicators (#) on both internal and external components. The name given must correspond to the GLOBAL name of the routine desired in the library.

In the ADSS system, the library indicator can be used only in the resident code list (RCL).

4. LINK ENTRY POINTS AND FILE NAMES

Each External Link Component has only one entry point whose calling will result in the loading of the LINK, if non-resident. This entry point name (GLOBAL symbol definition) must be the same as the filename of the file containing the binary unit of the External Component. The names of files containing either resident code or Internal LINK Components need not correspond to routine names (although normally they would).

The following must be considered to ensure the correct naming of files:

1) For External Components written in FORTRAN, the name of the source file should be the same as the subroutine name in the subroutine statement.
2) For External Components written in assembly language, the name of the source file should be the same as the label on the major entry point (the entry point should be defined as a GLOBAL synbol).
3) Internal Component files may be combined (using PIP) with an External Component file. The resultant filename, however, must be the same as the name of the External Component entry point.

5. THE RELOCATION PROCESS

Each LINK is relocated and output as a unit in the format described in Appendix C. The resident code, although not formally a LINK, is also relocated and output in LINK format, and is denoted LINK #000.[1]

CHAIN relocates into the XCT file rather than into core to avoid requiring sufficient core for itself and a LINK while it is being constructed. This results in the incorrect recording of 1) transfer vectors to routines not yet relocated (except External Link Components), 2) transfer vectors for elements of COMMON blocks not yet defined, and 3) string code address corrections (Standard Loader Codes 20 and 21). As this information is developed, it is stored in a core resident "Patch Table" and is recorded after all routines of the LINK or resident code have been relocated and output (see Appendix C).

[1]This standardization is imposed to minimize the size of EXECUTE. The Link Table is also recorded in LINK format and is denoted LINK #777 (actually #377777).

A core resident "Symbol Table" is used for global symbols, COMMON blocks, and COMMON symbols. Before any relocation, a GLOBAL symbol definition entry is made for each External Link Component such that calls to these routines will result in a transfer to the appropriate Link Table entry.

When relocating the resident code, a second Symbol Table entry is made for all names flagged with library indicators (i.e., #). This entry is a dummy GLOBAL symbol reference which is only recognized when searching for unresolved GLOBAL symbol references. It is used to ensure the inclusion of library routines not called by a resident routine.

In DOS systems, whenever a PGR (page) or BKR (bank) command option is specified, CHAIN performs a .USER macro for the <PAG> or <BNK> UIC respectively in .UFDT-1 to ensure that the correct library is searched during relocation.

After the relocation of each LINK, the Symbol Table is trimmed back to contain only entries made while relocating the resident code.

The Patch Table is constructed in increasing core from the first available register.[1] The Symbol Table is constructed in decreasing core writing over the lower third of CHAIN's code (which is no longer needed). The overlapping of the Symbol and Patch Tables results in a terminal error.

The image of the resident code and each LINK is recorded in the XCT file by relocating and outputting the routines from the indicated files on the user program device. If unresolved GLOBAL symbol references (references to yet unrelocated routine) exist, the user's Library (if existent) and the System Library are scanned, relocating and outputting any routines which contain a GLOBAL symbol definition that matches an unresolved GLOBAL symbol reference, thus resolving them. This Library search continues until all GLOBAL symbol references have been resolved or the libraries have been exhausted. The user's Library is scanned before the System Library. After each routine has been relocated and output, and if a load map has not been suppressed (NM option), a line will be typed out containing the routine's name (unless GM option) and the limits of core the routine will occupy.

[1]Determined from .SCOM+2.

In DOS systems, CHAIN calculates the number of 400_8 word-blocks needed to store the overlay system, by LINKs, as a core image. This information is stored in the environment indicator in bits Ø through 11 as a right-justified octal number. The number calculated does not include the LINK table (LINK 377777) or the resident code (LINK Ø).

6. I/O DEVICE ASSIGNMENTS

In either operating system CHAIN's I/O operations are accomplished via six .DAT slots. The slots and the functions assigned each are:

In ADSS operating systems CHAIN's I/O operations are accomplished via six .DAT slots. The slots and the functions assigned each are:

.DAT-1	System Library - The file .LIBR (.F4LIB under the B/F Monitor) is scanned to satisfy unresolved GLOBAL references.
.DAT-2	Command Input - Normally the console Teletype,[1] but may be the card or paper tape reader under Batch Mode.
.DAT-3	Typed output device - Must be the console Teletype.
.DAT-4	User program device - All routines listed during command input, which are not indicated as residing in a library, are relocated from this device.
.DAT-5	User's Library - The file .LIBR5 is scanned to satisfy unresolved GLOBAL symbol references. If a user's Library does not exist, .DAT-5 must be assigned "NONE".
.DAT-6	Relocated output device - the XCT file is written on this device.

CHAIN never has more than two files open at the same time. Whenever two files are open, one is open for input and the other for output. The handler functions are limited to: .INIT, .ENTER, .SEEK, .READ, .WRITE, .WAIT, and .CLOSE (i.e., DTB or DKB may be used.)

[1]Teletype is the registered trademark of the Teletype Corporation.

7. BUILDING AN OVERLAY SYSTEM

Before calling CHAIN, the user should be sure that the proper .DAT slot assignments have been made (see paragraph 6).

CHAIN is called by typing "CHAIN" following the Monitor's $ request. When loaded, CHAIN will type its name and version number and make the following requests:

```
NAME XCT FILE
LIST OPTIONS & PARAMETERS
DEFINE RESIDENT CODE
DESCRIBE LINKS & STRUCTURE
```

A response to each request via the command input device is necessary.

7.1 Command Input

CHAIN reads commands via the console Teletype or, in Batch Mode, via the card or paper tape reader. All input is accepted in logical lines, which consist of one or more physical lines. A carriage return (or card column 81) is used to continue a logical line onto the next physical line. An ALTMODE is used to terminate a logical line. A line (logical) consists of names (file, routine, option, parameter), library indicators (resident code definition only), and break characters (name terminators). Blanks are ignored. Names consist of 1-6 alphanumeric characters[1], and a library indicator (#) may appear preceding, following, or within a name.

The characters, equal sign, colon, comma, and slash are recognized as break characters, but are only accepted as valid break characters when appropriate.

When an error is detected, the entire logical line containing the error is rejected. IOPS ASCII editing (RUBOUT and ↑U) apply only to physical lines. A ↑P typein during command input will restart CHAIN.

An angle bracket (>) is typed out at the left margin to indicate the beginning of a line. A hyphen (-) is typed out at the left margin to indicate continuation of a line.

[1] RADIX-5Ø subset.

7.2 Conditional Messages

CHAIN's error messages attempt to indicate the source of an error. If recovery is possible, either by retyping a line (logical) or by typing ↑P and restarting CHAIN, CHAIN will type an angle bracket at the left margin to indicate input is requested. If recovery is not possible, CHAIN will exit to the Monitor after typing the error message.[1] A list of CHAIN and EXECUTE error messages is given in Appendix F --

Examples:

```
↑EXTERNAL NAME USED PRV -- XXXXX
>

↑IMPROPER BREAK CHAR -- /
>

TABLE OVERLAP
MONITOR VXX
$
```

Other messages indicate a required action. In these cases a ↑P typein is used to signal CHAIN that the operation has been performed.

Examples:

```
EOM, ↑P TO RESTART
LOAD XXXX & ↑P
```

7.3 XCT File Name

A 1-6 character file name terminated by an ALTMODE is required; i.e., a line containing a file name is the response to "NAME XCT FILE". This name is used when requesting EXECUTE to run the Overlay System under construction.

7.4 Option and Parameter List[2]

Options and environmental parameters are listed on a (logical) line, separated by commas. A zero length line (ALTMODE only) indicates no options or parameters specified. The following option and parameter abbreviations are recognized by CHAIN:

[1]The Symbol Table is built over the command input code, thus restart of CHAIN is not possible once relocation has begun.

[2]See addendum (page 30)

PGR	Components of the Overlay System are to be PAGE (4K) relocated, i.e., to run on a PDP-15.
BKR	Components of the Overlay System are to be BANK (8K) relocated, i.e., to run on a PDP-9 or PDP-15 under the "Bank Mode Monitor" (KM9-15).
FGD:x	Overlay System is to be built for Foreground use (B/F Monitor) where "x" is the lowest register used by the CHAIN, i.e., the base of the Overlay System.
FGD	Overlay System is to be built for Foreground use using a default base.
BGD	Overlay System is to be built for Background use under B/F Monitor or normal use under a Keyboard Monitor System.
8K 12K 16K 2ØK 24K 28K 32K	Core size of machine on which Overlay System is to be run (not needed for FGD Overlay Systems
PAR	Pause and type out LINK number after relocating the resident code (LINK #000).
PAL	Pause and type out LINK number after relocating each LINK (including resident code). ↑P typein to continue after a pause.
NM	No load map.
GM	If load map is output, names are to be file and global symbol names rather than program names.
SZ	The size of the Link Table, COMMON blocks, and routines are to be included in a load map. The size is listed following the core limits.
Default:	Background, Load Map, and core size and relocation mode (PAGE/BANK) of the Monitor under which the Overlay System is being constructed
SAC	Elements of LABELED COMMON may be referenced from any co-resident LINK.
VTC (DOS only)	This option used without names restricts all COMMON areas to bank boundaries; used with names, it restricts the named COMMON to bank boundaries.

If a conflict occurs, the latter (or right most) options or parameters are used. For example, the list

```
BKR, 12K, 20K, PGR
```

will cause CHAIN to build an Overlay System for a 2ØK PDP-15

NOTE

When building an Overlay System to run under another Monitor, care should be taken to use the correct library. Viz., ADSS BANK/PAGE or DOS BANK/PAGE.

7.5 Resident Code Definition

The names of files containing relocatable binary units of routines to be resident throughout a run and the names of Library Routines (flagged by library indicators (#)) to be resident throughout a run are listed on a line, separated by commas (,). EXECUTE transfers initial control to the entry point of the first resident routine relocated, i.e., the first routine of the first file listed, unless resident code is exclusively Library Routines. The response to "DEFINE RESIDENT CODE" must be at least one name.

7.6 LINK and Structure Definition

The Overlay structure is described in terms of LINK names. When a LINK is to consist of only one external component, the name of the file containing the external component may be used as the LINK name. However, when a LINK is to consist of more than one external component, the LINK must be named and defined.

7.6.1 LINK DEFINITIONS - Each LINK definition requires one line of command input consisting of the LINK's name followed by an equal sign (=) followed by the LINK definition.

A LINK definition is a list of the names of files which contain the relocatable binary units that comprise the LINK components. The individual file names listed are separated by commas (,); the two types of LINK components which may be used (external and internal) are separated within the definition by a slash (/). All external LINK component names must be listed before (to the left of) the slash separator; all internal LINK components must be listed after (to the right of) the slash. External LINK components are accepted only from files with names which match the external component name (i.e., GLOBAL symbol definition). In DOS external and internal link components can be retrieved from a library by the use of #, but the GLOBAL name must be used, in the command input.

NOTE

In ADSS, library indicators (#) can only be used in the resident code definition.

Example:

```
ABC=SUB1,SUB2/SUB3,SUB4
```

In the above example, SUB1 and SUB2 are external components of LINK ABC, and SUB3 and SUB4 are internal components of LINK ABC.

Rules for defining a LINK:

1. A LINK may not be a component of another LINK.
2. The names of the components of a LINK may not be used as LINK names[1]
3. A file name used in the resident code description cannot be used in a LINK definition.
4. A file name preceding a slash may be used only once.
5. A file name following a slash may be used in other LINK definitions (following a slash).

[1]When a LINK consists of only one component, the component's file name may be used as the LINK name in the "overlay structure description", but not in a LINK definition; i.e., it is not necessary to define a single component LINK but, if defined, the LINK name cannot be the component name.

7.6.2 OVERLAY STRUCTURE DESCRIPTION - An overlay structure is described using the names of defined LINKS, or the names of files containing LINK components and the operators colon (:) and comma (,), under the following set of rules:

1. A line is an independent statement processed from left to right.
2. A colon is read "is overlayed by".[1]
3. A comma is read "and".

Example:

```
SUB1:SUB2
SUB2:SUB3,SUB4
```

is interpreted as -- SUB1 is overlayed by (uses the same core as) SUB2, SUB2 is overlayed by SUB3 and SUB4, but SUB3 and SUB4 do not overlay each other.

4. A colon operator may not be used in a line after a comma has been used. This restriction prevents the following ambiguity:

```
SUB2:SUB3,SUB4:SUB5
```

The above line is rejected by CHAIN because it is not clear whether SUB5 overlays SUB3 or SUB4 or both. All four of the following examples are acceptable:

SUB2:SUB3,SUB4 SUB4:SUB5	SUB5 uses the same core as SUB4 but not the same core as SUB3.
SUB2:SUB3,SUB4 SUB3:SUB5	SUB5 uses the same core as SUB3 but not the same core as SUB4
SUB2:SUB5:SUB3,SUB4	SUB5 uses the same core as SUB3 _and_ SUB4. SUB3 and SUB4 are loaded individually (if non-resident) as called.
LINK=SUB3,SUB4 SUB2:LINK:SUB5	SUB5 uses the same core as SUB3 _and_ SUB4. Both SUB3 and SUB4 are loaded (if non-resident) whenever either is called.

5. A LINK name may appear only once preceding a colon and only once following another colon.
6. If a LINK name is used twice, it must be used following a colon before being used preceding another colon.
7. Several LINKS overlaying each other may be defined in one statement.

Example:

```
SUB1:SUB2:SUB3,SUB4
```
[1]

NOTE

This is a short method of defining the same overlay structure as in the first example, under rule 3.

[1]A loading order is not implied; just core mapping.

Rules 5 and 6, although they may appear restrictive, do not limit the user's description of an overlay structure, but do prevent multiple description of the position of a LINK in an overlay structure. A LINK may be both overlayed and overlaying, and it may not be possible or convenient to describe both conditions by using the LINK name only once, as follows:

```
        SUB1:SUB2:SUB3          SUB2 is overlaying SUB1 and is
                                overlayed by SUB3
```

Therefore, when a LINK is both overlaying and overlayed, its LINK name may be used twice, but the LINK(s) overlayed by it must be described before the LINK(s) by which it is overlayed.

Example:

```
        SUB2:SUB3,SUB4          SUB3 overlays SUB2
        SUB3:SUB5               SUB3 is overlayed by SUB5
```

NOTE

The description of an overlay structure only defines a desired core mapping; i.e., stating that SUB1 is overlayed by SUB2 means that both are to be relocated to the same core and cannot co-reside, but does not imply that SUB1 must be called before SUB2. There is no imposed order in which routines must be called, nor is there restriction of the routines callable by any routine.

7.7 Termination of Command Input

When the last line of overlay structure description has been input, command input is terminated by a zero length line (ALTMODE only). At this time, relocation of the resident code will begin and ↑P restart will no longer be possible because the command input code will be written over by the Symbol Table.

Command input may also be terminated after describing only resident code by typing an ALTMODE in response to the DESCRIBE LINKS & STRUCTURE request. This allows CHAIN & EXECUTE to be a useful alternative to the LINKING LOADER when the routines to be loaded and the loader cannot fit in core together, or when a job is to be run often and it is desirable to be able to load it with a simple command. Viz., E JOBNAM.

7.8 Examples

The following source code is used in examples 1 through 6.

```
C            FILE: MAIN
C            MAIN PROGRAM
C
             COMMON A,B
             CALL SUB1 (4,P)
             CALL SUB2 (2,Ø)
             IF (P-Q) 4Ø,1Ø,4Ø
1Ø           IF (A-P) 4Ø,2Ø,4Ø
2Ø           IF (B-Q) 4Ø,3Ø,4Ø
3Ø           IF (P+Q) 5Ø,4Ø,5Ø
4Ø           PAUSE
5Ø           STOP
             END

C            FILE: SUB1
C
             SUBROUTINE SUB1 (N,X)
             DIMENSION C(4)
             COMMON A /XXX/ C
             X=C(N)
             A=C(N)
             RETURN
             END

C            FILE: SUB2
C
             SUBROUTINE SUB2 (N,X)
             DIMENSION C(4)
             COMMON A,B /XXX/ C
             X=C(N)*C(N)
             B=C(N)*C(N)
             RETURN
             END

C            FILE: BDTA
C
             BLOCK DATA
             DIMENSION C(4)
             COMMON /XXX/ C
             DATA C(1),C(2),C(3),C(4)
             2       /1.Ø,2.Ø,3.Ø,4.Ø/
             END
```

NOTE

A complete list of error messages is given in Appendix F.

EXAMPLE 1

The angle bracket (>) in the left margin indicates the beginning of a logical line of command input; the remainder of the line is keyed in by the user and terminated by an ALTMODE. ALTMODE is a non-printing character. Both the minimum and maximum core locations occupied by the Link Table, a routine, or a COMMON block, are output in the Load Map as octal constants.

Block data initialized labeled COMMON Block XXX is resident and available to both SUB1 and SUB2.

```
CHAIN V4A

NAME XCT FILE
>XFN
LIST OPTIONS & PARAMETERS
>16K,PGR,SZ
DEFINE RESIDENT CODE
>MAIN,BDTA
DESCRIBE LINKS & STRUCTURE
>SUB1:SUB2
>                                   ←  ZERO LENGTH LINE TERM-
LINK TABLE                             INATES LINKS & STRUCTURE
       37607-37636 ØØØ3Ø               DESCRIPTION

RESIDENT CODE
MAIN   375Ø2-376Ø6 ØØ1Ø5
BDTA   37472-375Ø1 ØØØ1Ø            ←  XXX IS DECLARED AND
STOP   37457-37471 ØØØ13               INITIALIZED IN RESIDENT
PAUSE  37443-37456 ØØØ14               ROUTINE BDTA
SPMSG  3735Ø-37442 ØØØ73
REAL   364Ø3-37347 ØØ745
.CB    36363-364Ø2 ØØØ2Ø

LINK -- SUB1
SUB1   36323-36362 ØØØ4Ø
.DA    36254-36322 ØØØ47
.SS    36174-36253 ØØØ6Ø
INTEGE 36Ø14-36173 ØØ16Ø

LINK -- SUB2
SUB2   363Ø5-36362 ØØØ56
.DA    36236-363Ø4 ØØØ47
.SS    36156-36235 ØØØ6Ø
INTEGE 35776-36155 ØØ16Ø

BLANK COMMON
.XX    35772-35775 ØØØØ4

CORE REQ'D
       35772-37636 Ø1645
```

EXAMPLE 2

Both LINKs LK1 and LK2 contain the Block Data subprogram BDTA. I.e., SUB1 and SUB2 each has its own copy of initialized labeled COMMON Block XXX.

```
CHAIN V4A

NAME XCT FILE
>XFN
LIST OPTIONS & PARAMETERS
>16K,PGR
DEFINE RESIDENT CODE
>MAIN
DESCRIBE LINKS & STRUCTURE
>LK1=SUB1/BDTA
>LK2=SUB2/BDTA
>LK1:LK2
>
LINK TABLE
        37607-37636

RESIDENT CODE
MAIN    37502-37606
STOP    37467-37501
PAUSE   37453-37466
SPMSG   37360-37452
REAL    36413-37357
.CB     36373-36412

LINK -- LK1
SUB1    36333-36372
BDTA    36323-36332     ← XXX IS DECLARED AND
.DA     36254-36322       INITIALIZED IN INTERNAL
.SS     36174-36253       LINK COMPONENT BDTA
INTEGE  36014-36173       (see addendum, page 30)

LINK -- LK2
SUB2    36315-36372
BDTA    36305-36314     ← XXX IS DECLARED AND
.DA     36236-36304       INITIALIZED IN INTERNAL
.SS     36156-36235       LINK COMPONENT BDTA
INTEGE  35776-36155

BLANK COMMON
.XX     35772-35775

CORE REQ'D
        35772-37636
```

EXAMPLE 3

Only LINK LK1 has an initialized labeled COMMON Block XXX, the COMMON declaration in SUB2 has resulted in the allocation of core within LINK LK2 for an uninitialized labeled COMMON Block XXX (35750-57). The main program will pause.

```
CHAIN V4A

NAME XCT FILE
>XFN
LIST OPTIONS & PARAMETERS
>16K,PGR
DEFINE RESIDENT CODE
>MAIN
DESCRIBE LINKS & STRUCTURE
>LK1=SUB1/BDTA
>LK2=SUB2
>LK1:LK2
>
LINK TABLE
        37607-37636

RESIDENT CODE
MAIN    37502-37606
STOP    37467-37501
PAUSE   37453-37466
SPMSG   37360-37452
REAL    36413-37357
.CB     36373-36412

LINK -- LK1
SUB1    36333-36372
BDTA    36323-36332    ← INITIALIZED BLOCK XXX
.DA     36254-36322      (see addendum, page 30)
.SS     36174-36253
INTEGE  36014-36173

LINK -- LK2
SUB2    36315-36372
.DA     36246-36314
.SS     36166-36245
INTEGE  36006-36165
XXX     35776-36005    ← UNINITIALIZED BLOCK XXX

BLANK COMMON
.XX     35772-35775

CORE REQ'D
        35772-37636
```

EXAMPLE 4

Block Data subprograms do not have an entry point (file name defined as a Global Symbol), and therefore cannot be an External Link Component.

```
CHAIN V4A

NAME XCT FILE
>XFN
LIST OPTIONS & PARAMETERS
>16K,PGR
DEFINE RESIDENT CODE
>MAIN
DESCRIBE LINKS & STRUCTURE
>LK1=SUB1,BDTA
>LK2=SUB2
>LK1:LK2
>
LINK TABLE
       37574-37636

RESIDENT CODE
MAIN   37467-37573
STOP   37454-37466
PAUSE  37440-37453
SPMSG  37345-37437
REAL   36400-37344
.CB    36360-36377

LINK -- LK1
SUB1   36320-36357
BDTA   36310-36317
.DA    36241-36307
.SS    36161-36240
INTEGE 36001-36160
MISSING GLOBAL DEF -- BDTA
```

EXAMPLE 5

SUB1 and SUB2 have been misspelled in the LINK definitions. Since SUB1 and SUB2 are called by the main program, Global references for the symbols "SUB1" and "SUB2" are made. Neither SUB1 nor SUB2 exists in a Library, nor are they recognized as LINK components (because their names do not appear in a LINK definition or in the Overlay description). Therefore, the attempt to resolve the Global references fails.

```
CHAIN V4A

NAME XCT FILE
>XFN
LIST OPTIONS & PARAMETERS
>16K,PGR
DEFINE RESIDENT CODE
>MAIN
DESCRIBE LINKS & STRUCTURE
>LK1=S..1/BDTA
>LK2=S..2/BDTA
>LK1:LK2
>
LINK TABLE
       37607-37636

RESIDENT CODE
MAIN   37502-37606
STOP   37467-37501
PAUSE  37453-37466
SPMSG  37360-37452
REAL   36413-37357
.CB    36373-36412
UNRESOLVED GLOBAL(S):
SUB1
SUB2
```

EXAMPLE 6

Miscellaneous Recoverable Errors

```
CHAIN V4A

NAME XCT FILE
>XFN4567
↑ NAME LENGTH ERR
>XFN
LIST OPTIONS & PARAMETERS
>6K,BKR,SZ
↑ UNRECOGNIZED SYMBOL -- 6K
>16K,BKR,SZ
DEFINE RESIDENT CODE
>MAIN/BDTA
↑ IMPROPER BREAK CHAR -- /
>MAIN,BDTA
DESCRIBE LINKS & STRUCTURE
>LK1#=SUB1/BDTA
↑ LIB IND ON LINK NAME -- LK1
>LK1=SUB1/BDTA
↑ INTERNAL NAME USED PRV -- BDTA
>LK1=SUB1/LBDTA
>LK2=SUB1/LBDTA
↑ EXTERNAL NAME USED PRV -- SUB1
>LK2=SUB2/LBDTA
>LK1:LK2:MAIN
↑ RES ROUTINE NAME USED AS LINK NAME -- MAIN
>LK1:LK2
>LK1:SUB3
↑ NAME USED LEFT OF COLON TWICE -- LK1
>LK2:SUB3
>
LINK TABLE
        37574-37636 00043

RESIDENT CODE
MAIN    37467-37573 00105
BDTA    37457-37466 00010
 .       .       .      .
 .       .       .      .
 .       .       .      .
```

EXAMPLE 7

Both of the following cases have resulted in the same core allocation. (They would not have done so, if SUB4 were larger than SUB2). However, in Case II, SUB2 and SUB3 cannot be loaded individually (the entire LINK ABC is loaded whenever it is non-resident and either SUB2 or SUB3 is called). This may be useful in decreasing execution time, but also prevents SUB4 from calling SUB3 without overlaying itself.

CASE I

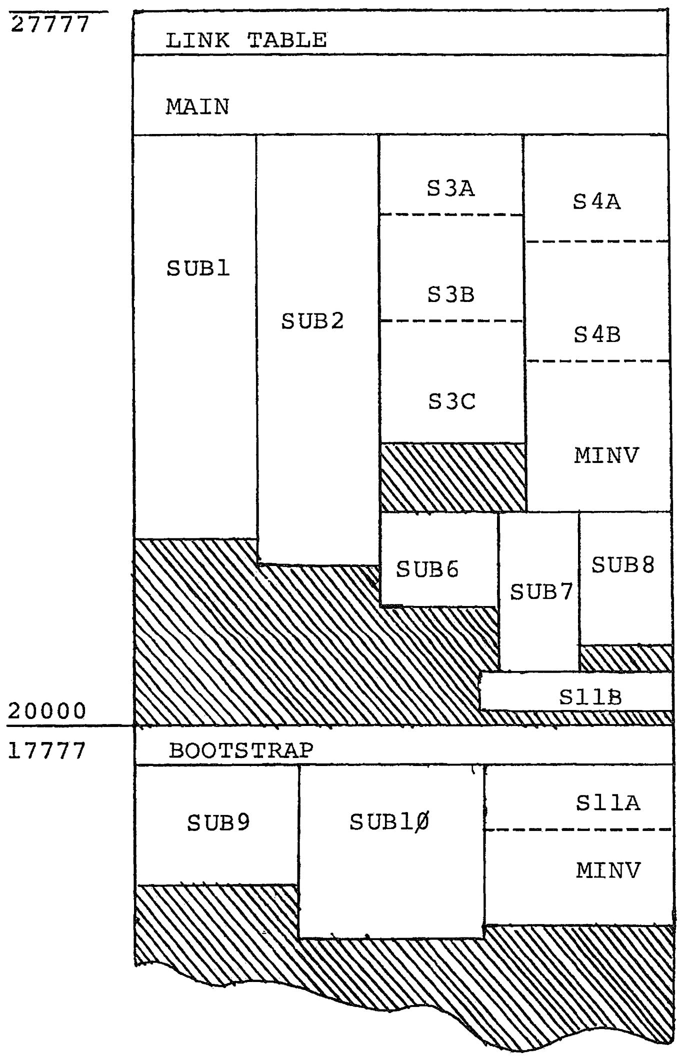
```
CHAIN V4A

NAME XCT FILE
>XFN
LIST OPTIONS & PARAMETERS
>16K,PGR
DEFINE RESIDENT CODE
>MAIN
DESCRIBE LINKS & STRUCTURE
>SUB1:SUB2,SUB3
>SUB2:SUB4
>
LINK TABLE
        37561-37636

RESIDENT CODE
MAIN    37415-37560

LINK -- SUB1
SUB1    36431-37414

LINK -- SUB2
SUB2    36741-37414

LINK -- SUB4
SUB4    37105-37414

LINK -- SUB3
SUB3    36347-36740

CORE REQ'D
        36347-37636
```

CASE II

```
CHAIN V4A

NAME XCT FILE
>XFN
LIST OPTIONS & PARAMETERS
>16K,PGR
DEFINE RESIDENT CODE
>MAIN
DESCRIBE LINKS & STRUCTURE
>ABC=SUB2,SUB3
>SUB1:ABC:SUB4
>
LINK TABLE
        37561-37636

RESIDENT CODE
MAIN    37415-37560

LINK -- SUB1
SUB1    36431-37414

LINK -- ABC
SUB2    36741-37414
SUB3    36347-36740

LINK -- SUB4
SUB4    37105-37414

CORE REQ'D
        36347-37636
```

```
CHAIN V4A

NAME XCT FILE
>XFN
LIST OPTIONS & PARAMETERS
>12K,PGR,SZ
DEFINE RESIDENT CODE
>MAIN
DESCRIBE LINKS & STRUCTURE
>LK3=S3A,S3B,S3C
>LK4=S4A,S4B/MINV
>LK11=S11A,S11B/MINV
>SUB1:SUB2:LK3,SUB6
>LK3:LK4
>SUB6:SUB7:SUB8
>SUB9:SUB1Ø:LK11
>
LINK TABLE
        27544-27777 ØØ234

RESIDENT CODE
MAIN    26415-27543 Ø1127

LINK -- SUB1
SUB1    21654-26414 Ø4541

LINK -- SUB2
SUB2    212ØØ-26414 Ø5215

LINK -- LK3
S3A     25215-26414 Ø12ØØ
S3B     23757-25214 Ø1236
S3C     22653-23756 Ø11Ø4

LINK -- LK4
S4A     24755-26414 Ø144Ø
S4B     23377-24754 Ø1356
MINV    22247-23376 Ø113Ø

LINK -- SUB6
SUB6    21Ø11-22246 Ø1236

LINK -- SUB7
SUB7    2Ø277-22246 Ø175Ø

LINK -- SUB8
SUB8    2Ø525-22246 Ø1522

LINK -- SUB9
SUB9    16Ø33-17636 Ø16Ø4

LINK -- SUB1Ø
SUB1Ø   15667-17636 Ø175Ø

LINK -- LK11
S11A    17357-17636 ØØ26Ø
S11B    2ØØ25-2Ø276 ØØ252
MINV    16227-17356 Ø113Ø

CORE REQ'D
        15667-27777 12111
```

EXAMPLE 8
CORE ALLOCATION ABOUT
MEMORY BOUND



EXAMPLE 9

Subroutine to cause the Line Printer Handler (LPA.) to be included in a LINK and to assign LPA to .DAT+5.

```
/ FILE: ALPA5
/
            .GLOBL  ALPA5,LPA.
.DAT=135
/
ALPA5       0
            LAC         LPA.
            DAC*        (.DAT+5)
            .INIT       5,1,0
            JMP*        ALPA5
            .END
```

```
C    FILE: OUTPUT
     SUBROUTINE OUTPUT (J,K)
     DIMENSION X(100),Y(100)
     COMMON X,Y
C
     CALL ALPA5
     DO 10 N=J,K
10   WRITE (5,20) X(N),Y(N)
20   FORMAT (2F10.2)
     RETURN
     END
```

FORTRAN subroutine "OUTPUT" requests the loading of the handler assigned to .DAT+5; therefore, .DAT+5 should be assigned to "NONE". In the case of a MACRO subroutine, the .IODEV 5 may be omitted, and the .DAT+5 assignment can be ignored.

```
CHAIN V4A

NAME XCT FILE
>XFN
LIST OPTIONS & PARAMETER
>16K,PGR
DEFINE RESIDENT CODE
>MAIN
DESCRIBE LINKS & STRUCTURE
PRINT=OUTPUT/ALPA5 (for ADSS)
PRINT=OUTPUT/ALPA5,LPA (for DOS)
>SUB1:SUB2:PRINT
>
LINK TABLE
         37574-37636
RESIDENT CODE
MAIN     37544-37573
STOP     37531-37543
SPMSG    37436-37530

LINK -- SUB1
SUB1     37361-37435
.SS      37301-37360
INTEGE   37121-37300
REAL     36154-37120
.CB      36134-36153

LINK -- SUB2
SUB2     37361-37435
.SS      37301-37360
INTEGE   37121-37300
REAL     36154-37120
.CB      36134-36153

LINK -- PRINT
OUTPUT   37345-37435
ALPA5    37333-37344
LPA.     36662-37332          ← Included
.DA      36613-36661            to satisfy
BCDIO    33621-36612            the GLOBAL
.SS      33541-33620            reference
FIOPS    33001-33540            made in
OTSER    32705-33000            ALPAS
INTEGE   32525-32704
REAL     31560-32524
.CB      31540-31557

BLANK COMMON
.XX      30717-31537

CORE REQ'D
         30717-37636
```

The above example is of use under a keyboard monitor. The user should be familiar with I/O under the B/F Monitor before attempting to include an I/O handler in a LINK of an Overlay System to be run under the B/F Monitor.

EXAMPLE 10

The example below illustrates the use of CHAIN as a virtual transparent loader. The term "virtual" is used since EXECUTE V4A (instructions) resides in core at execution time of the Overlay System. The example demonstrates the method actually used to provide an EXECUTE file of FOCAL for the 8K user on the PDP-15/20 system tape V5A so that, at execution time, DTE may be assigned for both library input and output to DECtape. User commands are underlined.

```
KM15 V5A

$R CHAIN

.DAT    DEVICE  USE

-6      DTB2    OUTPUT
-5      NONE    USER LIBRARY
-4      DTC2    USER PROGRAM(S)
-3      TTAØ    CONTROL AND ERROR MESSAGES
-2      TTAØ    COMMAND STRING
-1      DTCØ    SYSTEM LIBRARY

$A DTBØ -1,-4/DTB2 -6

$CHAIN

CHAIN V5A

NAME XCT FILE
>FOCAL<ALT MODE>
LIST OPTIONS & PARAMETERS
><ALT MODE>                /DEFAULT = CHAIN BUILDING
                           /MACHINE & CORE SIZE (8K PDP15)
DEFINE RESIDENT CODE
>FOCAL<ALT MODE>

DESCRIBE LINKS & STRUCTURE
><ALT MODE>
FOCAL   11762-17636
FNEW    1161Ø-12Ø37
.BH     11554-11665
DSQRT   11464-11631
DSIN    11451-11541
DCOS    1143Ø-11526
DATAN   11415-115Ø5
DEXP    114Ø2-11472
DLOG    11361-11457
.DD     11213-11436
.DB     11Ø73-1127Ø
.DE     1Ø772-1115Ø
.DF     1Ø633-11Ø47
.DC     1Ø564-1Ø71Ø
.DA     1Ø515-1Ø641
DOUBLE  1Ø312-1Ø572
REAL    Ø6736-Ø7777
```

Note: 1306_8 cells are available in an 8K PDP-15/20 system for FOCAL program development.

```
.CB      1Ø272-1Ø367
 CORE REQ'D
         Ø6736-17636
```

To further illustrate the use of EXECUTE where FOCAL XCT, FOCAL XCU are on DECtape unit Ø and the User FOCAL Library on unit 2:

```
KM15 V5A
$A DTEØ -4,-1/DTE2 3,5,7,10
$E FOCAL
EXECUTE V4A

FOCAL15 V9A

*
```

CHAIN V5A .DAT Slot Assignments

In order to allow the use of I/O handlers (e.g., DIB.) which include .FSTAT among their functions, CHAIN automatically ".SEEK's" explicitly named as well as implicitly named files (e.g., the user library (.LIBR5)), if the .DAT slot contains an assignment. Therefore, if no user library is referenced, .DAT slot -5 should be assigned to "NONE" or else a spurious IOPS13 (file not found) will result.

NOTE

When building an overlay system for "another machine" (e.g., PDP-9/15, EAE/NON-EAE), the library (.LIBR BIN) for the system on which overlays are to be executed must be used by CHAIN. Therefore, .DAT slot -1 must be assigned to a device with the appropriate .LIBR BIN file.

CHAIN-EXECUTE Restrictions Eliminated

Some restrictions in versions of EXECUTE earlier than V4A have been eliminated:

1. EXECUTE now clears the resident core area before loading
2. EXECUTE no longer clobbers the bootstrap when loading a LINK which occupies core both above and below the bootstrap (12K, 20K, 28K systems).

7.9 Using Non-File-Directoried Devices

When a file is required from a non-file-directoried device,[1] the file name is typed out in the following message:

```
     LOAD:    name & ↑P
```

Reading will begin following a ↑P typein.

If end of medium is detected, the following message will be typed out:

```
     EOM, ↑P TO RESTART
```

Reading will continue following a ↑P typein.

8. EXECUTION OF AN OVERLAY SYSTEM

The program EXECUTE oversees the execution of a CHAIN built system of overlays.

Before calling EXECUTE, the user should be sure that the "XCT file" is on the device assigned to .DAT-4, and that the .DAT slots used by the Overlay System are properly assigned.

EXECUTE is called by typing "EXECUTE" or "E" followed by the XCT file name, in response to the Monitor's $ request. The command and file name must be separated by at least one space.

The System Loader will open the XCT file to determine the .DAT slots used by the Overlay System, load EXECUTE and the I/O handlers assigned to the indicated .DAT slots, and transfer control to EXECUTE.

In DOS, EXECUTE will not announce itself. In ADSS EXECUTE will type its name and version number and then open the XCT file named in the call for EXECUTE, load the Link Table and resident code, and transfer control to the main program. The XCT file remains open throughout a run.

[1]To require minimum sized I/O handlers, CHAIN uses .INIT rather than .FSTAT to determine whether a device is file-oriented. If the maximum buffer size returned is greater than 63_{10} words, the device is assumed to be file-oriented. DECtape, disk, and magtape are file-oriented devices, and paper tape and cards are non-file-oriented devices.

The order in which the LINKS are loaded is not a function of EXECUTE. EXECUTE simply loads a LINK whenever a component subroutine is called and its LINK is non-resident.

EXECUTE will detect and indicate the following errors. All are terminal errors.

CAN'T FIT	Overlay System will not fit in available core.
CAN'T RUN	Overlay System either was relocated in the wrong mode (PAGE/BANK) or is the wrong type (BGD/FGD).
READ ERR	A portion of the XCT file is unreadable.

EXECUTE reads the XCT file via .DAT slot -4. The use of different handlers for the same I/O device (e.g., DTA and DTB) during a run, is generally not possible;[1] i.e., when transferring data to or from a device (including different units of a device) via more than one .DAT slot, each of these slots should be assigned to the same handler.[2]

When the XCT file has been recorded on a non-file-oriented device (e.g., paper tape), the following message is typed out to instruct the user to load the reader and type ↑P :

```
LD [XXX] & ↑P
```

where XXX is the LINK number of the required LINK. This message is output, initially, whenever a required file has been passed, or whenever an end-of-medium is detected.

9. CHAIN & EXECUTE UNDER THE I/O(PAPER TAPE) MONITOR

Under the I/O Monitor, .DAT slot re-assignment is not permitted and, at EXECUTE time, the XCT file is not read by a system loader which will load I/O handlers per .DAT slots used by the Overlay System. If the Overlay System requires other than paper tape I/O,

[1]The user should understand "Interrupt Setup" before attempting to use more than one handler for an I/O device.

[2]This restriction did not exist with previous versions of CHAIN and EXECUTE, where I/O handlers were included in "CHAINs" and no "CHAIN" would use EXECUTE's I/O device handler.

the additional handlers must be included in the Overlay System. This is illustrated in Example 9 (page 21).

When building an Overlay System onto paper tape, it is recommended that the PAL option be used to cause CHAIN to pause and type

```
PAUSE #XXX
```

after relocating and outputting each LINK. At each pause, the newly punched tape may be separated and labeled with the LINK number output in the pause typeout. EXECUTE will request LINKS by typing

```
LD [XXX] & ↑P
```

whenever an end-of-medium is detected or a required LINK has been passed.

APPENDIX A

THE LINK TABLE

The Link Table facilitates the interception of calls to external components of non-resident LINK. It consists of a transfer vector to an entry point in EXECUTE (EXUTV), a one-word buffer (BUF), and an eleven-word entry for each External Link Component.

CHAIN has altered calls to External Link Components to transfer control, via JMS to the first word of the appropriate Link Table entry. When a LINK is non-resident, its Link Table entries have the following format:

```
Ø
DAC    BUF
JMS*   EXUTV
DAC*   .+3
LAC    BUF
JMP*   .+2
ENTRY
ENTRY+1
LINK NUMBER
MIN. ADDRESS
MAX. ADDRESS
```

Thus, calling a non-resident subroutine transfers control to EXECUTE pointing to the fourth word of the Link Table entry for the External Link Component called.

EXECUTE then:

1. Fetches the LINK number from the Link Table and loads the LINK containing the called routine.
2. Places a LAC .-2 instruction in the third word of the Link Table entry for each component of the LINK just loaded.
3. Places a JMS* EXUTV instruction in the third word of the Link Table entry for each component of each LINK overlayed by the LINK just loaded.
4. Transfers control to the third word of the Link Table entry for the called routine, which effects the call.

While the LINK is resident, further calls of external components result in a transfer from the Link Table directly to the subroutine, i.e., the call overhead for resident external Link components is ten memory cycles (DAC BUF, LAC .-2, DAC* .+3, LAC BUF, JMP* .+2).

APPENDIX B

THE XCT FILE

The "XCT file" is actually two files, one with an "XCT" extension and the other with an "XCU" extension. Both have the file name response to CHAIN's "NAME XCT FILE" request.

CHAIN writes the XCU file first. It contains the resident code (LINK zero) and all of the actual LINKS numbered sequentially from one. The last information output is the XCT file which contains the Link Table, parameters required by EXECUTE, and the .IODEV information. The XCT file is written in the standard LINK format[1] and is designated LINK #377777. This allows EXECUTE to load the Link Table by simply loading LINK #377777.

The XCT file is opened and read by the System Loader (for the .IODEV information) and by EXECUTE before the XCU file is opened. Thus, by not including the XCT file information at the end of the XCU file (and having CHAIN output only one file), two passes to the end of a file, each time EXECUTE is loaded, have been eliminated. When CHAIN outputs to a file-oriented device, a dummy XCT file is written before the XCU file to reserve prime space for the actual XCT file.

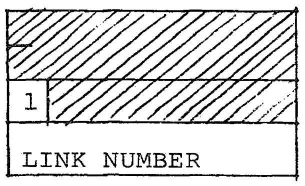
On paper tape, the XCT "file" is at the end of the tape and separated from the XCU "file" by a section of blank tape. This tape may be cut and spliced, or duplicated, to reverse positions of the files.

[1]The standard LINK format is described in Appendix C.

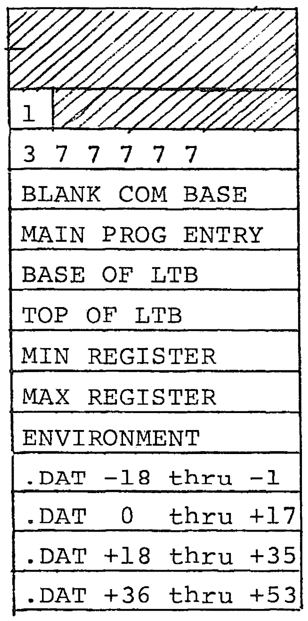
APPENDIX C

RECORD FORMATS

LINK ID (TYPE 1) RECORD



Special case:
LINK #377777



All LINKS begin with a type 1 record, normally followed by type 2 records, followed by at least one type 3 record.

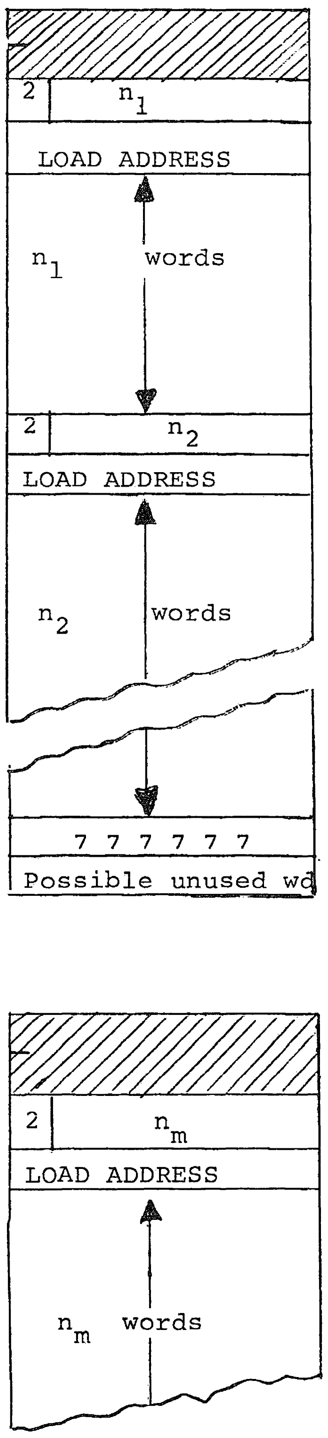
Loading LINK #377777 loads the Link Table.

Loading LINK #0 loads the resident code.

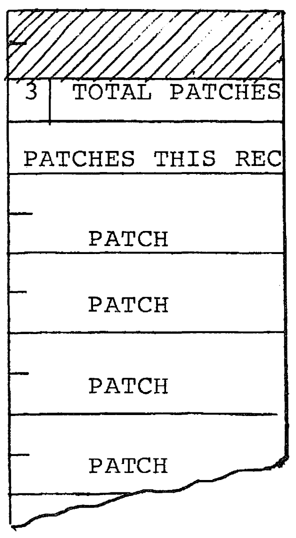
ENVIRONMENT WORD

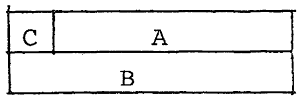
BIT	0	1
16	PGR	BKR
17	BGD	FGD

CONTIGUOUS BLOCKS (TYPE 2) RECORD



PATCH (TYPE 3) RECORD

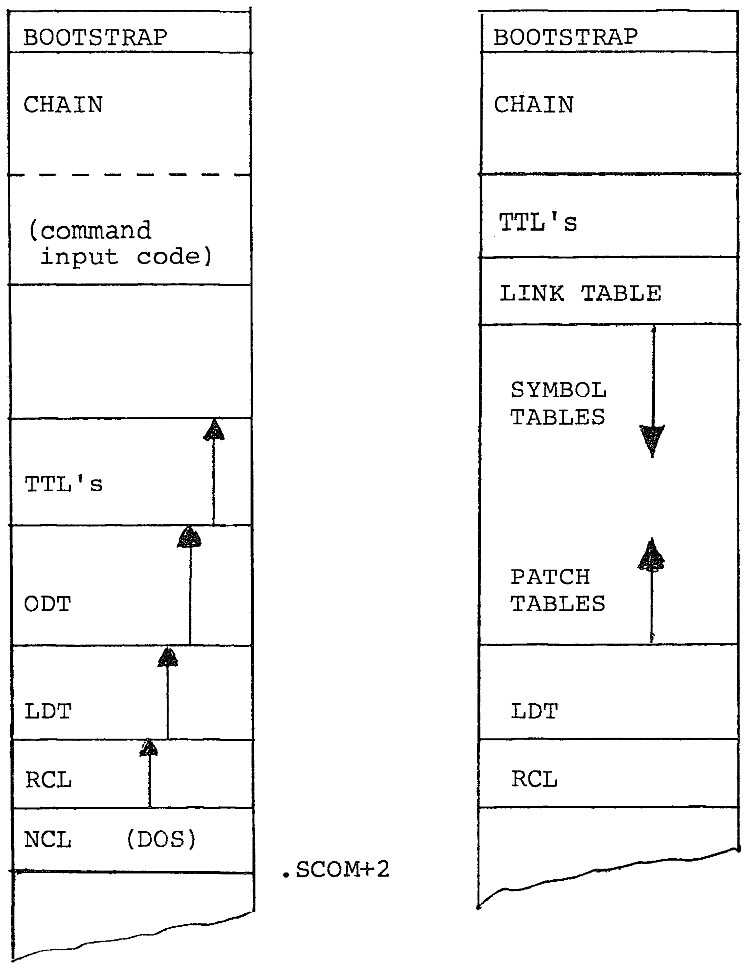




C=0	Store B at A
C=1	Store the address field (low order 12 or 13 bits) of B in the address field at A
C=2	Add the base of Blank Common to B and store sum at A

APPENDIX D

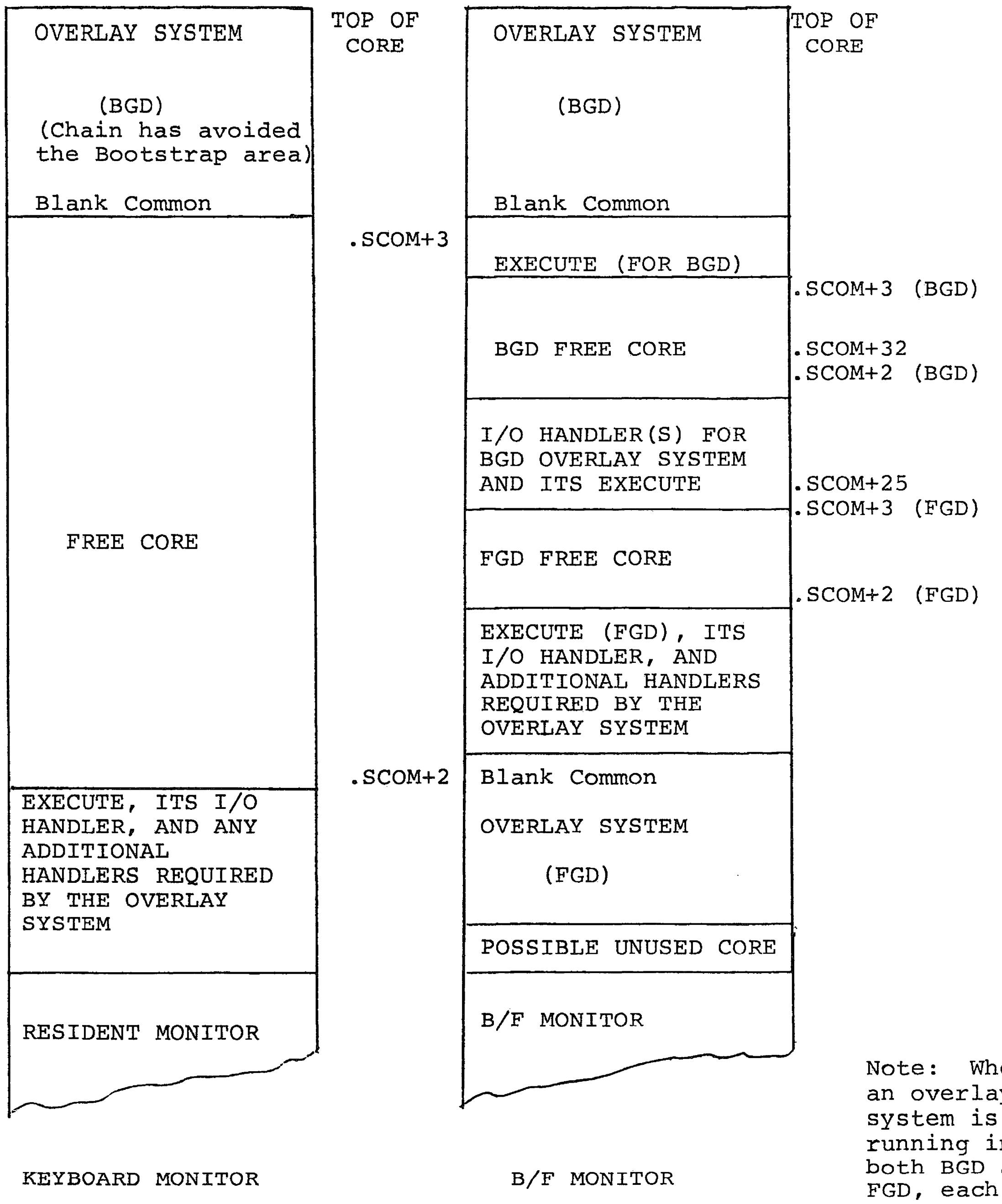
CORE ALLOCATION DURING CHAIN



The core used by the NCL (named COMMON list "UTC" option), RCL (Resident Code List, LDT (LINK Definition Table, ODT (Overlay Description Table), and the TTL's (Trunk-to-Twig Lists) is shown here to help when reading the program listing, which is not essential to effective use of CHAIN & EXECUTE.

APPENDIX E

CORE ALLOCATION DURING EXECUTE



Note: When an overlay system is running in both BGD and FGD, each has its own copy of EXECUTE

APPENDIX F

CHAIN AND EXECUTE ERROR MESSAGES

CHAIN performs comprehensive error checking and outputs messages, which are self-explanatory. Messages regarding command string errors immediately follow the questionable logical line (the line is rejected and must be retyped). Recoverable errors are indicated by an > following the message. Where possible, the offending character or name is output with the error message -- as may be noted from Example 6, paragraph 7.8. If the error is not recoverable, CHAION exits to the Monitor after typing the message. Other messages indicate a required action by the operator followed by a CTRL P character, as in the following example:

```
EOM, ↑P TO RESTART
LOAD prgnam & ↑P
```

The following abbreviations are used in error messages:

RES	RESIDENT
PRV	PREVIOUS
DEF	DEFINITION
LIB	LIBRARY
IND	INDICATOR
BLK	BLOCK
ERR	ERROR
ABS	ABSOLUTE
PROG	PROG

RECOVERABLE ERRORS:

↑UNRECOGNIZABLE SYMBOL

↑RES ROUTINE REQ'D (No resident routine was declared)

↑LINK NAME USED PRV (A legal name has 1-6 characters)

↑LINK NAME USED PRV (See paragraph 7.6.1 for correct use of LINK definitions)

↑IMPROPER BREAK CHAR (One of the break characters was used incorrectly)

↑INTERNAL NAME REPEATED IN LINE (See paragraph 2.3 for discussion of internal names)

↑EXTERNAL NAME USED PRV (See paragraph 2.2 for discussion of external names)

↑COMPONENT NAME USED AS LINK NAME (Names of components of a LINK may not be used as LINK names)

↑LINK DEF WITHIN OVERLAY DESCRIPTION

↑COLON MUST FOLLOW FIRST LINK NAME

↑MORE THAN ONE LINK OVERLAYED

↑NAME RIGHT OF COLON USED PRV (A LINK name may appear only once following a colon)

↑NAME USED MORE THAN TWICE

↑NAME USED LEFT OF COLON TWICE (A LINK name may appear only once preceding a colon)

↑LIB IND ON LINK NAME (# is a LIBRARY indicator)

ADSS Only { ↑LIB IND ON EXTERNAL NAME (# is a LIBRARY indicator)
↑LIB IND ON INTERNAL NAME (# is a LIBRARY indicator) }

↑INTERNAL NAME USED PRV (See paragraph 2.3)

↑RES ROUTINE USED AS A LINK NAME

↑NAME USED MORE THAN ONCE

<u>UNRECOVERABLE ERRORS:</u>

The following CHAIN errors are terminal. CHAIN will exit to the Monitor after typing the error message.

TABLE OVERLAP (See paragraph 5 for an explanation of tables generated during relocation)

CORE OVERFLOW (A core load greater than 32K is indicated)

READ ERROR (Terminal error on the input device)

ILLEGAL LOADER CODE (The input file contains an unrecognizable LOADER code)

LABELED COMMON BLK SIZE ERR (Labeled COMMON block now declared to be larger than previously declared to be)

UNRESOLVED GLOBAL(S): (A list of all unresolved GLOBALS is generated -- see paragraph 5)

ABS PROG (CHAIN must be able to relocate all programs and .ABS programs are not relocatable)

MISSING GLOBAL DEF (The missing GLOBAL definition will be listed)

DUPLICATE GLOBAL DEF (The duplicate GLOBAL will be listed)

EXECUTE ERRORS ARE ALL TERMINAL

CAN'T FIT (Overlay system will not fit in available core)

CAN'T RUN (Overlay system was relocated in the wrong mode -- PAGE/BANK -- or is the wrong type -- BGD/FGD)

READ ERR (A portion of the XCT file is unreadable)

HOW TO OBTAIN SOFTWARE INFORMATION

Announcements for new and revised software, as well as programming notes, software problems, and documentation corrections are published by Software Information Service in the following newsletters.

Digital Software News for the PDP-8 & PDP-12
Digital Software News for the PDP-11
Digital Software News for the PDP-9/15 Family

These newsletters contain information applicable to software available from Digital's Program Library, Articles in Digital Software News update the cumulative Software Performance Summary which is contained in each basic kit of system software for new computers. To assure that the monthly Digital Software News is sent to the appropriate software contact at your installation, please check with the Software Specialist or Sales Engineer at your nearest Digital office.

Questions or problems concerning Digital's Software should be reported to the Software Specialist. In cases where no Software Specialist is available, please send a Software Performance Report form with details of the problem to:

Software Information Service
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

These forms which are provided in the software kit should be fully filled out and accompanied by teletype output as well as listings or tapes of the user program to facilitate a complete investigation. An answer will be sent to the individual and appropriate topics of general interest will be printed in the newsletter.

Orders for new and revised software and manuals, additional Software Performance Report forms, and software price lists should be directed to the nearest Digital Field office or representative. U.S.A. customers may order directly from the Program Library in Maynard. When ordering, include the code number and a brief description of the software requested.

Digital Equipment Computer Users Society (DECUS) maintains a user library and publishes a catalog of programs as well as the DECUSCOPE magazine for its members and non-members who request it. For further information please write to:

DECUS
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

READER'S COMMENTS

Digital Equipment Corporation maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback -- your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability and readability.

Did you find errors in this manual? If so, specify by page.

How can this manual be improved?

Other comments?

Please state your position. ______________________ Date: __________

Name: ______________________ Organization: ______________________

Street: ______________________ Department: ______________________

City: ______________ State: ______________ Zip or Country __________

- - - - - - - - - - - - - - - - - Fold Here -

- - - - - - - - - - - - - - Do Not Tear - Fold Here and Staple - - - - - - - - - - - - - -