SCOPE 3.1.5 for the CONTROL DATA® 6400, 6500, and 6600 computers supervises the assembly, compilation, and execution of a wide variety of jobs. SCOPE scheduling increases job throughput. In addition to input/output functions, storage assignment, accounting, and operator communications, SCOPE provides these special features:

- Extended core storage allocation
- Random access on both 6603 and 6638 disks
- Usage of 854 disk pack and 865 drum as system devices
- Tape error recovery
- Extended character set
- Tape labeling and automatic reel switching
- Linking loader with segment and overlay capabilities
- Optimum use of input/output equipment and priority processing
- Implementation of file action macros and system action requests
- Operating environment information during program execution
- Minimized use of control points for system functions
- Job checkpoint/restart
- Multi-file reels and multi-reel files
- Debugging aids

**Library Programs**

<table>
<thead>
<tr>
<th>Library Program</th>
<th>SIMSCRIPT</th>
<th>EXPORT/IMPORT</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPASS</td>
<td>TTY Respond</td>
<td>PERT/TIME</td>
</tr>
<tr>
<td>FORTRAN</td>
<td>APT</td>
<td>SORT/MERGE</td>
</tr>
<tr>
<td>FORTRAN Extended</td>
<td>ALGOL-60</td>
<td>OPTIMA</td>
</tr>
<tr>
<td>COBOL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CONTROL STATEMENTS

n, Tt, CMfl, ECb, Pp.                              Job Identification

n     Alphanumeric job name, 1-7 characters beginning with a letter

Tt    Central processor time limit in seconds, 1-777778

fl    Central memory field length, 1-3600008

b     Extended core storage 10008-word blocks, 1-77778

p     Priority level, \(1 \leq p \leq 2^k - 1, \quad k \leq 8\) according to installation option; \(1 = \text{lowest priority}\)

LOADER(name)                                    Loader Selection

name   Name of loader

PPLOADR Peripheral Processor Loader

CPLOADR Central Processor Loader

LOAD(lnf)                                       File Loading

lnf    Logical file name

EXECUTE(name, p1, p2, ..., pn)                  Execution

name   Program entry point

p_i    Program parameter forms:
       \(p_i, \quad p_i = 0, \quad p_i = q_i\)
       \(p_i\) and \(q_i\) are 7-character strings
       \(1 \leq i \leq 53\)
name, \(p_1, p_2, \ldots, p_n\)  

Program Call

name  

Program entry point

\(p_i\)  

Program parameter forms:

\(p_1, p_1 = 0, p_1 = q_i\)

\(p_i\) and \(q_i\) are 7-character strings

\(1 \leq i \leq 53\)

NOGO.  

Load Completion

Load, print memory map, but do not execute.

REQUEST, lfn, dt, dc, x, y, eq.  

Equipment Request

lfn  

Alphanumeric logical file name beginning with a letter, 1-7 characters.

dt  

Device type written as yxx where \(y = \text{no.}\) of devices and \(xx = \text{equipment types}\).

\(xx\) TYPES

CP Card punch

LP Line printer

MT Mag. tape, 1/2" density depends on labeling

LO Mag. tape, 1/2", 200 bpi

HI Mag. tape, 1/2", 556 bpi

HY Mag. tape, 1/2", 800 bpi

CR Card reader

Dnnnnn Mass storage unit, nnnn = FET code

dc

File disposition code

PR Print

P1 Print on 501/505

P2 Print on 512

PU Punch Hollerith

PB Punch binary
CK      Checkpoint dump
P8      Punch binary (full 80 columns)
MF      Multi-file tape

x      Magnetic tape data format
blank  SCOPE standard
X      External, SCOPE 2.0 compatible
S      Stranger tape

y      Magnetic tape label format
blank  Unlabeled
E or N SCOPE standard

eq     Equipment number

COMMON, ifn.  Common File

ifn    Alphanumeric logical file name, beginning with a letter, 1-7 characters

RELEASE, ifn.  Common File Release

ifn    Alphanumeric logical file name, beginning with a letter, 1-7 characters

SWITCH, n.  Pseudo Sense Switch

Set switch n = 1-6

MODE, n.  Arithmetic Exit Mode

n    Exit mode, set to 7 unless altered by this control statement:

n = 0  Disable exit mode

n = 1  Address out of range; reference outside established limits of central memory or extended core storage, or negative word count in an extended core storage communication.
n = 2  Floating point arithmetic; infinite operand
n = 3  Address or operand out of range
n = 4  Floating point arithmetic, operand indefinite
n = 5  Indefinite operand or address out of range
n = 6  Indefinite operand or operand out of range
n = 7  Indefinite operand, operand out of range, or address out of range

REMOVES_if__prome.
COMMENTS COMMENTS

Dayfile Comments

Remarks listed as comments on dayfile

EXIT.

Process following control cards if job terminated abnormally except for compilation and assembly errors.

EXIT(S)

Process following control cards if job terminated abnormally.

REDUCE.

Reduce Field Length

MAP(p)

Map Control

p

Mapping specification

ON Full map after loading
OFF No map
PART No entry addresses in map
SEGMENTS AND OVERLAYS

SEGZERO(sn, pn₁, pn₂, …, pnₙ) First Segment

SEGMENT(sn, pn₁, pn₂, …, pnₙ) Subsequent Segments

SECTION(sname, pn₁, pn₂, …, pnₙ) Section

OVERLAY(fn, l₁, l₂, Cnnnnnn) Overlay

sn Relocatable segment name
pn₁ Relocatable section or subprogram names
sname Relocatable section name
fn Absolute file name
l₁ Primary level number (octal); 0 for first overlay
l₂ Secondary level number (octal); 0 for first overlay
Cnnnnnn Optional; begin loading nnnnnn words from the start of blank common
FILE UPDATING

UPDATE (parameter list)

UPDATE.

Parameters may be absent, present, or (except F and Q) present and equal to a value; parameters may appear in any order.

P = fname
Old program library; if omitted, OLDPL assumed

N = fname
New program library; if omitted, no new program library written

I = fname
File containing control cards; if omitted, UPDATE assumes INPUT

L = fname
Listable output file; if omitted, OUTPUT assumed

C = fname
Card images to be assembled on named file; if omitted, COMPILER assumed; if C = 0, no compile file written

S = fname
Source deck listing on named file; if omitted no file written

F
Full assembly option; if omitted, only COMPILER and modifications listed

Q
Speed updating corrections, *COMPILER must include all routines to be modified; common deck modifications must be specified by user

File Manipulation Cards

*REWIND fname
*SKIP fname, rent
*READ fname
*LABEL label name
Creation

*DECK dname
*COMDECK dname
*END

Assembly

*COMPILE a,b,c,...,d
*DECK dname
*COMDECK dname
*WEOR n
*CALL dname

Correction and Updating

*IDENT idnam
*PURGE idnam
*DELETE a,n
*DELETE a,n,b,m
*RESTORE a,n
*RESTORE a,n,b,m
*INSERT a,n
*YANK a
/*any comments
*ADDFILE fname, a,n

**LABEL MACRO**

lfn  LABEL  fln, ed, ret, create, reel, mfn, pos

lfn Logical file name
fln File label name
ed Edition number
ret Retention cycle
create Creation date
reel Reel number
mfn Multi-file name
pos Position number
FET CREATION MACROS

Sequential coded file

```
\textbf{lnf} FILEC fwa,f, (WSA = \texttt{addr}_w, l_w), (OWN = eoi, err)
LBL, DTY = dt, DSC = \texttt{dc}, UPR, EPR,
UBC = \texttt{ubc}, MLR = \texttt{mlrs}
```

Sequential binary file

```
\textbf{lnf} FILEB fwa,f, (WSA = \texttt{addr}_w, l_w), (OWN = eoi, err)
LBL, DTY = dt, DSC = \texttt{dc}, UPR, EPR,
UBC = \texttt{ubc}, MLR = \texttt{mlrs}
```

Random coded file

```
\textbf{lnf} RFILEC fwa,f, (WSA = \texttt{addr}_w, l_w), (IND = \texttt{addr}_i, l_i)
(OWN = eoi, err), LBL, DTY = dt,
DSC = \texttt{dc}, UPR, EPR
```

Random binary file

```
\textbf{lnf} RFILEB fwa,f, (WSA = \texttt{addr}_w, l_w), (IND = \texttt{addr}_i, l_i)
(OWN = eoi, err), LBL, DTY = dt,
DSC = \texttt{dc}, UPR, EPR
```

Required parameters

```
\textbf{lnf} Logical file name
fwa First word address of FET
f Number of words in FET
```

Optional parameters:

```
dt Device type
dc Disposition code
\texttt{addr}_w First word address of working storage area
l_w Number of words in working storage area
\texttt{addr}_i First word address of index buffer
l_i Number of words in index buffer
eoi End-of-information address for OWNCOD: routine
err Error address for OWNCODE routine
```
LBL
LABEL definition macro follows FILE macro

UPR
User processing of end-of-reel conditions

EPR
User processing of error conditions

CHECKPOINT/RESTART OPERATIONS

CKP.
Checkpoint Dump

Save currently active files

RESTART, name, #.
Restart Checkpoint Dump

RESTART, #, name.

RESTART, name.

RESTART, #.

RESTART.

name
Name of dump file; CCCCCCC assumed if omitted

# Checkpoint number for restart
FILE ACTION REQUESTS

REQUEST param

Assign Equipment During Execution

First word address of two word parameter list:

<table>
<thead>
<tr>
<th>59</th>
<th>27</th>
<th>23</th>
<th>17</th>
<th>11</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>logical file name</td>
<td>status</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-------------------</td>
<td>--------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dc</td>
<td>dt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pyqx</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

status

| 000001 | Request completed |
| 22 8   | Illegal function  |
| 24 8   | FNT full          |
| 25 8   | No equipment logically available |
| 26 8   | No equipment available |
| 30 8   | Duplicate file name |
| 31 8   | Duplicate check file name |

pyqx

Used only when dt specified 1/2" mag. tape

p=1    External tape
p=0    SCOPE 3.0 tape
y=1    Two tapes
y=0    One tape
q=1    SCOPE file label
q=0    Unlabeled
x=1    Existing file
x=0    New file
dc

File disposition code

0000  No special action
0001  Checkpoint file
0002  Multifile tape
0003-7  Reserved
0010  Punch coded output
0011  Reserved
0012  Punch binary output
0013  Reserved
0014  Punch 80 column binary output
0015-37  Reserved
0040  Printed output
0041-1777  Reserved
2000-3777  File being processed by RESPOND
4000-5777  File being processed by EXPORT/IMPORT
6000-7777  Reserved

dt

Device type

Bits  11-6  5-0

00  SCOPE selected

AA  01  6603-I disk†††

00  System default, same as 03
01  Inner zone only  Alternate sector half-track
02  Outer zone only  
03  Both zones  
†04  Both zones  Sequential
†05  Inner zone only  sector full-track
†06  Outer zone only  
07  CDC reserved
10  Eight sector allocation (RESPON.
11-77  CDC reserved

† Codes are defined but supporting software is not provided by SCOPE.
†† Codes 0701 and 4000-7777 require a device assigned by REQUEST card or function before file is opened.
††† 6603-I disk is a basic 6603 with or without field option 10098 (disk speedup) installed; 6603-II is a 6603 with both field options 10098 and 10124 (speedup augment) installed.
| AB | 02 | 6638 disk |
|    | 00 | System default, same as 03 |
|    | 01 | Alternate sector halftrack |
|    | 02 | CDC reserved |
|    | 03 | Same as 01 |
|    | 04-07 | CDC reserved |
|    | 10 | Eight sector allocation |
|    | 11-77 | CDC reserved |
|    | 03 | Data cell |
| AC | 04 | 6603-II disk††† xx same as |
|    | 05,06 | CDC reserved for 6603-I |
| AP | 07 | 3234/854 disk |
|    | 00 | System default, same as 03 |
|    | ††, †01 | Private pack, same as 03 |
|    | 02 | CDC reserved |
|    | 03 | Alternate triplets of sectors, one track |
|    | 04-77 | CDC reserved |
|    | 10,11 | CDC reserved |
| AD | 12 | 3637/865 drum |
|    | 00 | Standard allocation is 64 words per PRU (1 PRU = 3 sectors), 2 PRU's per record block |
|    | 01-77 | Reserved for system |
|    | 13-17 | CDC reserved |
| †††† AX | 20 | ECS |
|    | 21-27 | CDC reserved |
|    | 30-37 | Reserved for installations, mass storage only |

†Codes are defined but supporting software is not provided by SCOPE.
††Codes 0701 and 4000-7777 require a device assigned by REQUEST card or function before file is opened.
†††6603-I disk is a basic 6603 with or without field option 10098 (disk speedup) installed; 6603-II is a 6603 with both field options 10098 and 10124 (speedup augment) installed.
††††The interim ECS system was developed by Graham Campbell, Kurt Fuchsel, and Sidney Heller, of the Brookhaven National Laboratory, Upton, New York. Work performed at Brookhaven National Laboratories is supported by the U.S. Atomic Energy Commission.
60x 1/2-inch 7-track, magnetic tape

(Right 6 bits in binary)

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xxxx00</td>
<td>HI density 556 bpi</td>
</tr>
<tr>
<td>xxxx01</td>
<td>LO density 200 bpi</td>
</tr>
<tr>
<td>xxxx10</td>
<td>HI density 800 bpi</td>
</tr>
<tr>
<td>xxxx11</td>
<td>CDC reserved</td>
</tr>
<tr>
<td>xx00xx</td>
<td>Unlabeled</td>
</tr>
<tr>
<td>xx01xx</td>
<td>SCOPE standard label (USASI)</td>
</tr>
<tr>
<td>xx10xx</td>
<td>CDC reserved (optional label)</td>
</tr>
<tr>
<td>xx11xx</td>
<td>CDC reserved</td>
</tr>
<tr>
<td>00xxxx</td>
<td>SCOPE standard data format</td>
</tr>
<tr>
<td>01xxxx</td>
<td>X data format</td>
</tr>
<tr>
<td>10xxxx</td>
<td>CDC reserved (S data format)</td>
</tr>
<tr>
<td>11xxxx</td>
<td>CDC reserved (L data format)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>41-43</td>
</tr>
<tr>
<td>†TR</td>
<td>44</td>
</tr>
<tr>
<td>†TP</td>
<td>45</td>
</tr>
<tr>
<td>--</td>
<td>46-47</td>
</tr>
<tr>
<td>LP</td>
<td>50</td>
</tr>
<tr>
<td>L1</td>
<td>51</td>
</tr>
<tr>
<td>L2</td>
<td>52</td>
</tr>
<tr>
<td>53-55</td>
<td>CDC reserved</td>
</tr>
<tr>
<td>--</td>
<td>56-57</td>
</tr>
<tr>
<td>CR</td>
<td>60</td>
</tr>
<tr>
<td>--</td>
<td>61-65</td>
</tr>
<tr>
<td>--</td>
<td>66-67</td>
</tr>
<tr>
<td>CP</td>
<td>70</td>
</tr>
<tr>
<td>DS</td>
<td>71</td>
</tr>
<tr>
<td>†GC</td>
<td>72</td>
</tr>
<tr>
<td>†HC</td>
<td>73</td>
</tr>
<tr>
<td>†FM</td>
<td>74</td>
</tr>
<tr>
<td>†PL</td>
<td>75</td>
</tr>
<tr>
<td>--</td>
<td>76-77</td>
</tr>
</tbody>
</table>

† Codes are defined but supporting software is not provided by SCOPE.

†† Codes 0701 and 4000-7777 require a device assigned by REQUEST card or function before file is opened.
For the following requests, these definitions are applicable:

**lnf**
Logical file name

**recall**
If non-blank, control returns to calling program after operation is completed; otherwise, control returns after accepting the request.

**l**
If non-blank, information is skipped until an end-of-record with levelnumber ≥ specified level number is read; if absent, this field is set to zero.

**OPEN lfn,x,recall**
Ready File For Processing

**x**
File operation: READ, WRITE, READNR, WRITENR, ALTER, REEL, REELNR

**CLOSE lfn,x,recall**
Set File to Close

**x**
File action:
- x absent, set at beginning-of-information
- x = NR
- x = UNLOAD

**CLOSER lfn,x,recall**
Terminate Processing/Control Labeling (magnetic tape only)

**x**
- x absent, rewind
- x = NR
- x = UNLOAD

**EVICT lfn,recall**
Release File Mass Storage Space

**READ lfn,recall**
Read Into Circular Buffer

**READSKP lfn,l,recall**
Read Into Circular Buffer

**RPHR lfn,recall**
Read Single Physical Record

1/2" magnetic tape only; no conversion

**READNS lfn,recall**
Read Mass Storage Non-Stop
READIN lfn, x
  x  Absent  Deblock into working storage area
        /name/  Read record using name index
     logical record number
        Read record using name or number index

WRITE lfn, recall
WRITER lfn, l, recall
WRITEF, lfn, recall
  Write From Circular Buffer
  Write with Level Number
  Write with Logical End-of-File

WPHR lfn, recall
  Write Single Physical Record
  1/2" magnetic tape only

WRITOUT lfn, x
  x  Absent  Block from working storage area
       /name/  Write using name index
    logical record number
       Write using name or number index

REWRITE lfn, recall
REWRI TER lfn, l, recall
REWRI TEF lfn, recall
  Rewrite Mass Storage
    Mass Storage Rewrite with Logical End-of-File
    only

WRITIN lfn, x
  x
    blank  Working storage to buffer
    /name/  Working storage to named record
    m  Working storage to numbered record

SKIPF lfn, n, l, recall
  Bypass Logical Records-Forward
  n  Number of logical records or record groups to be skipped; if absent, 1 is assumed

BKSP lfn, recall
  Bypass Logical Record-Reverse
BKSPRU lfn, n, recall  Bypass Physical Record Unit—Reverse
n  Number of PRUs to be bypassed; if absent, 1 is assumed
SKIPB lfn, n, l, recall  Bypass Logical Records—Reverse
n  Number of logical records or record group to be skipped; if absent, 1 is assumed.
l  Level number
REWIND lfn, recall  Rewind File
UNLOAD lfn, recall  Upload File

SYSTEM ACTION REQUESTS

For the following requests, this definition is applicable:

recall  If non-blank, control returns to calling program after operation is completed; otherwise, control returns after accepting the request

MEMORY type, status, recall  Obtain or Change Field Length

    type  Field length reference:
     CM  0
     ECS  1

    status  Field length alteration:
         0  No alteration
    any number  Alter field length to equal value of number

RECALL lfn  Generate Calling Sequence

    lfn  Base address of FET - job relinquishes central processor. When lfn is specified, control returns to program when I/O request is completed for that file; otherwise control returns next time around monitor loop
MESSAGE addr,x,recall

addr
Location where message is stored in display code

x
x = 0, message displayed and entered into dayfile
x ≠ 0, message displayed but not entered into dayfile

ENDRUN
Terminate Run Normally

ABORT
Terminate Job Abnormally

TIME status,recall
Time to Status

CLOCK status,recall
Clock to Status

DATE status,recall
Date to Status

JDATE status,recall
Julian Date to Status

LOADER param
Request to Loader

param
Location of user-established load sequence parameter list

<table>
<thead>
<tr>
<th>lfn</th>
<th>sl</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>r</td>
<td>p</td>
</tr>
<tr>
<td>u</td>
<td>v</td>
</tr>
<tr>
<td>m</td>
<td>k</td>
</tr>
<tr>
<td>s</td>
<td>f</td>
</tr>
<tr>
<td>c</td>
<td>lwa</td>
</tr>
<tr>
<td>59</td>
<td>53</td>
</tr>
<tr>
<td>47</td>
<td>43</td>
</tr>
<tr>
<td>41</td>
<td>39</td>
</tr>
<tr>
<td>37</td>
<td>35</td>
</tr>
<tr>
<td>17</td>
<td>fwa</td>
</tr>
</tbody>
</table>

lfn
File from which programs are loaded; entry point; subprogram name; or zero

sl
Location of a list of sections or subprograms, or a segment

l₁
Segment level if s ≠ 0, v = 0, and 0 ≤ l₁ ≤ 63; Primary overlay level if s = 0, v ≠ 0

l₂
Secondary level

r
Reset bit; if r ≠ 0, loader tables cleared
p  Partial map bit; if p ≠ 0, on-line partial core map given
u  Library overlay flag; if u ≠ 0, overlay load from system library
v  Overlay flag; if v ≠ 0, overlay load operation requested
m  NOMAP flag; if m ≠ 0, maps of segment or overlay load suppressed
k  Search key; if k ≠ 0, lfn is entry point name
s  Segment flag; if s ≠ 0, segment loading operation requested
f  Fill flag; if f ≠ 0, unsatisfied external symbols filled with out-of-bounds references
C  Complete flag; if c ≠ 0, load necessary subroutines from system library
lwa Last location, relative to RA, available for the loading operation; if lwa = 0, limit of program loading is first word of LOADEF tables
fwa Initial location, relative to RA, at which to begin loading; if fwa = 0, loading occurs at next available location

LOADREQ, param  Overlay Request

param

Zero or blank  Rewind file named in RA + 64 and load for execution
Non-blank  Location of user-established load sequence parameter list; same as for LOADER request.
LIBRARY PREPARATION AND MAINTENANCE

Definitions for the following requests:

s Source file
D Destination file
p Record name
r Residence: DS, CM
e Edition: 0-63
n \(1 \leq n \leq 2^{17}-1\); in SKIPF;
   n may be a name
x Count

EDITLIB. \hspace{1cm} \text{Call Statement}

System directory saved on common file

EDITLIB(RESTORE) \hspace{1cm} \text{Alternate Call Statement}

System directory replaced by contents of common file

MOVE(p, r) \hspace{1cm} \text{Change Record Residence}

DELETE(p) \hspace{1cm} \text{Delete Record from System Directory}

LIST(s) \hspace{1cm} \text{Write Out Directory}

READY(d) \hspace{1cm} \text{Prepare To Create System Library}

\(d = \text{SYSTEM}\) System directory manipulated
\(d = \text{SYSTEM,*}\) Empty directory created
\(d \neq \text{SYSTEM}\) Model of an empty directory and an empty
   scratch file prepared

TRANSFER(s, n, x) \hspace{1cm} \text{Copy System Records}

ADD(p, s, r, e) \hspace{1cm} \text{Add Record}

s \neq \text{SYSTEM} \hspace{0.5cm} \text{Pre-positioning of } s \text{ necessary}
S = \text{SYSTEM} \hspace{0.5cm} \text{Pre-positioning unnecessary}
p may be single record name or: \(p_1 - p_2, p - *

ADDBCD(p, s, r, e) \hspace{1cm} \text{Add Coded Record as Overlay}

ADDCOS(p, s, r, e) \hspace{1cm} \text{Add Record Without a Prefix}

ADDTEXT(p, s, r, e) \hspace{1cm} \text{Add Compile File From Update}
DELETE(p)  Delete Record
LENGTH (p)  Request Field Length
COMPLETE.  Complete File

READY statement must precede COMPLETE statement.

REWIND(s)  Rewind File
SKIPB(s, n)  Backspace On File
SKIPF(s, n)  Skip Logical Records

UTILITY FUNCTIONS

COPY(file1, file2)  Copy To End-of-Information
COPYBF(file1, file2, n)  Copy Binary File
COPYCF(file1, file2, n)  Copy Coded (BCD) File
COPYSBF(file1, file2, n)  Copy Shifted Binary File
COPYBR(file1, file2, n)  Copy Binary Record
COPYCR(file1, file2, n)  Copy Coded Record
COPYL(file1, file2, file3)  Copy Library
COPYN(p1, out, in1, in2, ... , in10)  Copy Logical Records

<table>
<thead>
<tr>
<th>p1</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Include id fields</td>
</tr>
<tr>
<td>Nonzero</td>
<td>Omit id fields</td>
</tr>
</tbody>
</table>

n = number of records/files

REWIND(file1)
SKIPF(file1, ± n)
SKIPR(file1, ± n)
WEOF(file1)
Record Identification Card: \( p_1, p_2, p_3 \)

- \( p_1 \): Name or number of beginning record
- \( p_2 \): Last record to copy
- \( p_3 \): Source file

name Copy \( p_1 \) to \( p_2 \)
integer \( 10 \) Number of records
* Copy to end-of-file
** Copy to double end-of-file
/ Copy to zero length record
0 or blank Copy \( p_1 \)

UNLOAD(file1) Unload file
REWIND(file1) Rewind file

LBC. Begin loading binary corrections at reference address + 100
LBC,address. Begin loading at address
LOC. Load octal corrections
LOC,address. Clear from reference address to address before modifying

LOC(address_1, address_2) Clear from address_1 to address_2 before modifying

PBC. Begin punching at reference address + 100_8; deck length in words specified by contents of RA + 117_8
PBC,address. Punch from reference address to address
PBC(address_1, address_2) Punch from address_1 to address_2
WBR,n,rl.

Write Binary Record

Begin writing from reference address + 100

n   File label must be TAPEn,n=1–7

rl  Record length in words; if omitted
    length is taken from lower 18-bits
    of RA + 117

RBR,n.

Read Binary Record

Begin reading into RA + 100

n   File label must be TAPEn;n=1–7

RFL,nfl.

Request Field Length

nfl  New field length in words

COMPARE(file1,file2,n,level,errors,records)

file1  File to be compared

file2  File to be compared

n  Number of records in file1

level  Minimum e-o-r level

errors  Number of discrepancies per record to list

records  Number of counted records to be processed
DEBUGGING AIDS

TRACE, p₁, p₂, ..., pₙ.

Parameters

p₁, p₂, ..., pₙ

ID = iiii iii

Optional alphanumeric identifier, 1-7 characters

Initial address

IA = e
IA = e + n

IA₁ = e - n
e Entry point name

IAC = c
IAC = c + n

IAC₁ = c - n

n Octal integer ≤ 777777

Last address

LA = e
LA = e + n

LA₁ = e - n

LAC = c
LAC = c + n

LAC₁ = c - n

Frequency

F₁ = n Trace from nth time IA encountered
F₂ = n Stop tracing nth time IA encountered
F₃ = n Trace every nth time IA encountered

n = octal integer

Register trigger

TR = P, An, Bn, or Xn  n = register number, 1-7

Masking trigger

TM = m, k₁, k₂, ..., kₙ

m  5 or 10 digit octal mask

k₁ Match key
Location trigger

TL = e
TL = e + n  e  Entry point name
TL1 = e - n  c  Labeled common block name
TLC = c
TLC = c + n  n  Octal integer \leq 777777
TLC1 = c - n  b  nth location in blank common
TLB = b

Register Dump
RD  Include register dump

Storage location reference

OL = e, i
OL = e + n, i
OL = e - n, i  Write i words, 1 \leq i \leq 100
OLC = c, i
OLC = c + n, i
OLC1 = c - n, i
OLB = b, i

Register designator

OR = r, i  r  Register An, Bn, Xn; write i words
           beginning at address in r
n = 0-7

\text{SNAP, } p_1, p_2, \ldots, p_n
\text{Parameters}

\text{Snapshot Dump}

p_1, p_2, \ldots, p_n
\text{Optional alphanumeric identifiers, 1-7 characters}

\text{ID = iiiiiii}
Trap location
IA = e
IA = e + n  c  Entry point name
IA = a
IA = e - n  c  Labeled common block name
IAC = c  n  Octal integer
IAC = c + n  a  Absolute address relative to RA
IAC1 = c - n

First word address of dump area  Last word address of dump area
FWA = e          LWA = e
FWA = e + n      LWA = e + n
FWA = n          LWA1 = e - n
FWA = a          LWAC = c
FWA = e - n      LWAC1 = c - n
FWA1 = n
FWAC = c          LWAB = b
FWAC = c + n      LWA = n
FWAC = n          LWA = a
FWAC1 = c - n    LWA = c + n
FWAC1 = n          LWA = n
FWAB = b          LWAC1 = n

b  bth location in blank common

Interval between dumped words
INT = n  n  Positive octal integer

Dump format
F  code character
  Characters
O  Octal
M  Octal with mnemonic operation codes
I  Integer
S  Single precision floating point
F  I format if exponent zero; S otherwise
D  Double precision floating point
C  Display code
R  Before or after above characters for register dump
Frequency
F1 = n
F2 = n
F3 = n

Dump nth time IA encountered.
Stop dump nth time IA encountered.
Dump every nth time IA encountered.
n Octal integer

Entry point
UR = p, r_1, ..., r_n
p User program entry point before dump taken
r_i Parameters

Post Mortem Dumps

DMP.
Exchange package and p-77 through p + 77

DMP, address.
Reference address through parameter address

DMP(address_1, address_2)
Dump address_1 through address_2;
dump absolute address_1 through absolute address_2 if 4 in high order address position.

DMPECS(address_1, address_2, f, lfn)
Dump Extended Core Storage

address_1, address_2
Dump from closest multiple of 10^8 greater than or equal to address_1 to closest multiple of 10^8 greater than address_2

f Print format per line
0 or 1 4 words in octal and display code
2 2 words in octal parcels/display
3 2 word octal bytes/display code
4 2 words octal/display code

lfn Dumpfile; OUTPUT assumed if omitted
<table>
<thead>
<tr>
<th>DEBUG (p)</th>
<th>Debug Control Card</th>
</tr>
</thead>
<tbody>
<tr>
<td>p</td>
<td>Dump</td>
</tr>
<tr>
<td>C</td>
<td>Labeled dump followed by a change dump when DMP encountered</td>
</tr>
<tr>
<td>T</td>
<td>Load TRACE and SNAP with (0, 0) overlay; load TRACE with SEGZERO with segment mode</td>
</tr>
<tr>
<td>S</td>
<td>Load TRACE and SNAP with (0, 0) overlay; load SNAP with SEGZERO in segment mode</td>
</tr>
</tbody>
</table>
FORTRAN COMPILE AND PUNCH

source statements

PROGRAM BOB (INPUT, OUTPUT, TAPE1)

RUN(P)

RA6600, T100, CM45000, EC40, P7.
FORTRAN Compile and Execute

PROGRAM OVA (INPUT, OUTPUT)

RUN.

JOB456, T500, CM50000, EC30, P3.
COMPASS SOURCE DECKS

Peripheral Processor Program

Central Processor Program

Absolute Central Processor Program

END

source statements

PERIPH
IDENT PPCODE, 1000B

ENTRY entry name

IDENT CPCODE

source statements

END

source statements

ABS
IDENT, ABSCP, 100B
COMPASS ASSEMBLIES

COMPASS source statements

COMPASS.
JOB, X6, T100, CM45000.

Compass assembly for LOAD-AND-GO

data

COMPASS source statements

DRIVE.
COMPASS(B=DRIVE)
WWW124, T500, CM45000, EC37, P3.
COMPASS ASSEMBLY
FROM PROGRAM LIBRARY

*COMPILE PROGA, PROGB
CORRECTIONS TO DECKS
PROGA AND PROGB
*IDENT CORR
COMPASS (I=COMPILE)
UPDATE(Q, P=PROGLIB)
REQUEST PROGLIB.
LBI, CM55000, T1000.
COMPASS ASSEMBLY AND
FORTRAN COMPILATION

*COMPILE FORT

*CORRECTIONS TO FORT

*IDENT COR2

*COMPILE COMP

CORRECTIONS TO COMP

*IDENT COR1

RUN($,,,$COMPIL)

COMPASS (I=ASSFMBL)

UPDATE(Q, P=OPL, C=COMPIL)

UPDATE(Q, P=OPL, C=ASSEMBL)

REQUEST OPL.

LBI, CM55000, T1000.
TRACE RUN

COMPASS source deck
COPYCF (SNACE, OUTPUT)
REWIND (SNACE)
LGO.
TRACE (parameters)
COMPASS.
LBJ, CM55000, T1000.
TRACE RUN

COMPASS source deck
COPYCR(SNACE, OUTPUT)
REWIND(SNACE)
LGO.
SNAP(parameters)
TRACE(parameters)
TRACE(parameters)
COMPASS.
RMN, CM55000, T1000.
PREPUNCHED BINARY PROGRAM

data

binary deck

DMP(20000)

EXIT.

INPUT.

GM1111, P6, CM50000, T400.