

÷.



Approved for Public Release: Distribution Unlimited



UNCLASSIFIED SECURITY CLASSIFICATION OF THIS PAGE

| | REPORT DOCUM | ENTATION PA | AGE | | |
|--|---|---|---|--|--|
| 1a. REPORT SECURITY CLASSIFICATION | | 1b. RESTRICTIVE | MARKINGS | <u> </u> | <u> </u> |
| UNCLASSIFIED | | | | | |
| 2a. SECURITY CLASSIFICATION AUTHORITY | | 3. DISTRIBUTION | N/AVAILABILITY | OF REPORT | |
| 26. DECLASSIFICATION/DOWNGRADING SCHED | JLE | | ved for P bution U | | |
| 4. PERFORMING ORGANIZATION REPORT NUMB | ER(S) | | ORGANIZATION | | |
| CISD-90/01 | | | | | |
| 6a. NAME OF PERFORMING ORGANIZATION | 6b. OFFICE SYMBOL | 7a. NAME OF M | ONITORING ORG | ANIZATION | |
| DTRC | (If applicable) | | | | |
| 6c. ADDRESS (City,State,and ZIP Code) | 3511 | | ity, State, and ZI | P. Codol | |
| | | 75. ADDress (C | ity, State, and Zi | | |
| Bethesda, MD 20084-5 | | | | | |
| 8a. NAME OF FUNDING/SPONSORING ORGANIZATION | 8b. OFFICE SYMBOL (If applicable) | 9. PROCUREMEN | T INSTRUMENT I | DENTIFICATION | INUMBER |
| 8c. ADDRESS (City,State,and ZIP Code) | _ I | 10. SOURCE OF | FUNDING NUMBER | RS | <u> </u> |
| | | PROGRAM ELEMENT NO. | PROJECT NO. | TASK NO. | WORK UNIT ACCESSION NO. |
| 12. PERSONAL AUTHOR(S) David V. Sommer, Sharo | T F Good | | | | |
| 13a. TYPE CF REPORT 13b. TIME | and the second | 1 | PORT (Year,Mont) /09/30 | h,Day) 15. PA | ge count 322 |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION | covered 193090 ^{to} inde | £ 90/ | /09/30 | I | 322 |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES | COVERED 93090 To inde 18. SUBJECT TERMS (C | E 90/ | 09/30 | d identify by bl | 322 |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION | COVERED 193090 To inde 18. SUBJECT TERMS (C Classifie | E 90/ | 09/30 if necessary and ig Cray | d identify by bi | 322 |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES | COVERED 193090 To inde 18. SUBJECT TERMS (C Classifie Computer | E 90/ | 19/30 if necessary and g Cray Cray | d identify by bl COS UNICOS | 322 |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES | COVERED 193090 To inde 18. SUBJECT TERMS (C Classifie Computer Control s | f 90/ | 19/30 if necessary and g Cray Cray | d identify by bi | 322 |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP | COVERED 193090 To inde 18. SUBJECT TERMS (C Classifie Computer Control s ry and identify by block | E 90/ Continue on reverse d computir tatements number) | if necessary and g Cray Cray DEC | d identify by bl COS UNICOS VAX/VMS | 322 |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to of the David Taylor Rese | COVERED 193090 To inde 18. SUBJECT TERMS (C Classifie Computer Control s ry and identify by block the Computer a carch Center | <u>f</u> 90/ Continue on reverse d computin tatements number) and Inform has instal | 109/30 if necessary and ng Cray Cray DEC nation Se led an I | d identify by bl COS UNICOS VAX/VMS rvices D ntegrate | 322 |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to of the David Taylor Rese Supercomputer Network. | COVERED 193090 To inde 18. SUBJECT TERMS (C Classifie Computer Control s ry and identify by block the Computer earch Center This manual | <u>f</u> 90/ Continue on reverse d computin tatements number) and Inform has instal provides a | 109/30 if necessary and ig Cray Cray DEC nation Se led an I in introd | d identify by bl COS UNICOS VAX/VMS rvices D ntegrate uction t | 322 lock number) Department ed to the |
| 13a. TYPE CF REPORT 13b. TIME Final FROM () 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in t of the David Taylor Rese Supercomputer Network. Network. Some informati | COVERED 193090 TO inde 18. SUBJECT TERMS (C Classifie Computer Control s ry and identify by block the Computer earch Center This manual on has been | <u>f</u> 90/ Continue on reverse d computin tatements number) and Inform has instal provides a distilled | 109/30 if necessary and ig Cray Cray DEC nation Se led an I in introd from man | d identify by bl COS UNICOS VAX/VMS rvices E ntegrate uction t y indivi | 322 lock number) epartment ed to the dual |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to of the David Taylor Rese Supercomputer Network. Network. Some informati documents and augmented | COVERED 193090 TO inde 18. SUBJECT TERMS (C Classified Computer Control s ry and identify by block the Computer earch Center This manual to reflect u | <u>E</u> <u>Continue on reverse</u> <u>d</u> computin <u>tatements</u> <u>number</u> and Inform has instal provides a distilled sage at DT | 19/30 if necessary and g Cray Cray DEC nation Se led an I in introd from man TRC. Con | d identify by bl COS UNICOS VAX/VMS rvices D ntegrate uction t y indivi trol sta | 322 Department ed to the dual tement |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to of the David Taylor Rese Supercomputer Network. Network. Some informati documents and augmented examples and description | COVERED 193090 TO inde 18. SUBJECT TERMS (C Classified Computer Control s ry and identify by block the Computer Earch Center This manual to reflect us as of hardward | <u>E</u> <u>Continue on reverse</u> <u>d</u> computin <u>tatements</u> <u>number</u> and Inform has instal provides a distilled sage at DT e and soft | 109/30 if necessary and ig Cray Cray DEC nation Se led an I in introd from man TRC. Con ware are | d identify by bl COS UNICOS VAX/VMS rvices E ntegrate uction t y indivi trol sta include | 322 Department ed to the dual tement ed, as is |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP 19. ABSTRACT (Continue on reverse if necessed The Computer Center in to of the David Taylor Reset Supercomputer Network. Network. Some informati documents and augmented examples and description information on moving fi | COVERED 193090 TO inde 18. SUBJECT TERMS (C Classifie Computer Control s ry and identify by block the Computer Earch Center This manual to reflect un to reflect un s of hardward les among the | <u>E</u> <u>Continue on reverse</u> <u>and computin</u> <u>tatements</u> <u>number</u> and Inform has instal provides a distilled sage at DT e and soft e CDC CYBE | 109/30 if necessary and ig Cray Cray DEC nation Se led an I an introd from man rC. Con ware are CR 860 (w | d identify by bl COS UNICOS VAX/VMS rvices E ntegrate uction t y indivi trol sta include ith the | 322 Department ed to the dual tement ed, as is Mass |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to f the David Taylor Rese Supercomputer Network. Network. Some informatid documents and augmented examples and description information on moving fis Storage System), the DEC | COVERED 193090 TO inde 18. SUBJECT TERMS (C Classifie Computer Control s Ty and identify by block the Computer Control s This manual con has been to reflect up as of hardward les among the CVAXcluster, | <u>f</u> 90/ Continue on reverse d computin tatements number) and Inform has instal provides a distilled sage at DT e and soft e CDC CYBE the secur | 109/30 if necessary and ig Cray Cray DEC nation Se led an I in introd from man TRC. Con ware are CR 860 (w the DEC VA | d identify by bl COS UNICOS VAX/VMS rvices D ntegrate uction t y indivi trol sta include ith the X, and t | 322 Department ed to the dual tement ed, as is Mass the CRAY |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to of the David Taylor Rese Supercomputer Network. Network. Some informati documents and augmented examples and description information on moving fi Storage System), the DEC X-MP, creating and exect | COVERED 193090 TO inde 18. SUBJECT TERMS (C Classifie Computer Control s Ty and identify by block the Computer arch Center This manual to reflect up to reflect up to reflect up to sof hardward les among the VAXcluster, ting batch jo | <u>e</u> <u>continue on reverse</u> <u>d</u> computin <u>tatements</u> <u>number</u> and Inform has instal provides a distilled sage at DT e and soft e CDC CYBE the secur obs, and u | A sif necessary and a if necessary and ng Cray DEC nation Se led an I an introd from man TRC. Con ware are DR 860 (w the DEC VA using the | d identify by bl COS UNICOS VAX/VMS rvices D ntegrate uction t y indivi trol sta include ith the X, and t interac | 322 Department ed to the dual tement ed, as is Mass the CRAY tive |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to f the David Taylor Rese Supercomputer Network. Network. Some informatid documents and augmented examples and description information on moving fis Storage System), the DEC | COVERED 193090 To inde 18. SUBJECT TERMS (C Classifie Computer Control s ry and identify by block the Computer arch Center This manual to reflect up to reflect up to sof hardward les among the CVAXcluster, ting batch jo | continue on reverse d computin tatements humber) and Inform has instal provides a distilled sage at DT e and soft e CDC CYBE the secur obs, and u X/MP, the | A if necessary and a if necessary and ng Cray DEC nation Se led an I in introd from man TRC. Con ware are TR 860 (w the DEC VA using the Mass St | d identify by bl COS UNICOS VAX/VMS rvices D ntegrate uction t y indivi trol sta include ith the X, and t interac | 322 Department ed to the dual tement ed, as is Mass the CRAY tive |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to of the David Taylor Rese Supercomputer Network. Network. Some informati documents and augmented examples and description information on moving fi Storage System), the DEC X-MP, creating and executor systems. Volume 1 20. DISTRIBUTION/AVAILABILITY OF ABSTRACT | COVERED 193090 TO inde 18. SUBJECT TERMS (C Classified Computer Control 5 Ty and identify by block the Computer This manual Con has been to reflect un to reflect un to reflect un to reflect un to sof hardward les among the VAXcluster, ting batch ju bes the Cray | <u>f</u> <u>90/</u> <u>continue on reverse</u> <u>d</u> computin <u>tatements</u> <u>number</u>) and Inform has instal provides a distilled sage at DT e and soft e CDC CYBE the secur obs, and u X/MP, the he CDC CYE | A if necessary and a if necessary and ng Cray DEC nation Se led an I in introd from man TRC. Con ware are TR 860 (w the DEC VA using the Mass St | d identify by bl COS UNICOS VAX/VMS rvices D ntegrate uction t y indivi trol sta include ith the X, and t interac orage Sy | 322 Department ed to the dual tement ed, as is Mass the CRAY tive |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to of the David Taylor Rese Supercomputer Network. Network. Some informatid documents and augmented examples and description information on moving fid Storage System), the DEC X-MP, creating and exects systems. Volume 1 description the DEC VAXes. Volume 2 | COVERED 193090 TO inde 18. SUBJECT TERMS (C Classified Computer Control 5 Ty and identify by block the Computer This manual Con has been to reflect un to reflect un to reflect un to reflect un to sof hardward les among the VAXcluster, ting batch jo bes the Cray | <u>e</u> <u>continue on reverse</u> <u>d</u> computin <u>tatements</u> <u>number</u> and Inform has instal provides a distilled sage at DT e and soft e CDC CYBE the secur obs, and u X/MP, the he CDC CYE | A stinecessary and a if necessary and a g Cray DEC nation Se led an I an introd from man TRC. Con ware are CR 860 (w the DEC VA ising the Mass St BER 860. SECURITY CLASS SIFIED | d identify by bl COS UNICOS VAX/VMS rvices D ntegrate uction t y indivi trol sta include ith the X, and t interac orage Sy IFICATION | 322 Department ed to the dual tement ed, as is Mass the CRAY tive stem and |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to of the David Taylor Rese Supercomputer Network. Network. Some informati documents and augmented examples and description information on moving fi Storage System), the DEC X-MP, creating and execut systems. Volume 1 20. DISTRIBUTION/AVAILABILITY OF ABSTRACT 20. DISTRIBUTION/AVAILABILITY OF ABSTRACT 20. DISTRIBUTION/AVAILABILITY OF ABSTRACT 20. DISTRIBUTION/AVAILABILITY OF ABSTRACT 21. NAME OF RESPONSIBLE INDIVIDUAL | COVERED 193090 TO inde 18. SUBJECT TERMS (C Classified Computer Control S Ty and identify by block the Computer This manual Con has been to reflect un to reflect un s of hardward les among the VAXcluster, ting batch ju bes the Cray | 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 21. ABSTRACT S 220. TELETHONE 220. TELETHONE | 109/30 a if necessary and ng Cray Cray DEC nation Se led an I in introd from man TRC. Con ware are CR 860 (w the DEC VA Ising the Mass St BER 860. SECURITY CLASS SIFIED (include Area Con | d identify by bl COS UNICOS VAX/VMS rvices D ntegrate uction t y indivi trol sta include ith the X, and t interac orage Sy IFICATION del 22c. OFFICE | 322 Department ed to the dual tement ed, as is Mass the CRAY tive stem and |
| 13a. TYPE CF REPORT 13b. TIME Final FROM 0 16. SUPPLEMENTARY NOTATION 17. COSATI CODES FIELD GROUP SUB-GROUP 19. ABSTRACT (Continue on reverse if necessar The Computer Center in to of the David Taylor Rese Supercomputer Network. Network. Some informatid documents and augmented examples and description information on moving fid Storage System), the DEC X-MP, creating and exects systems. Volume 1 description the DEC VAXes. Volume 2 | COVERED 193090 TO inde 18. SUBJECT TERMS (C Classified Computer Control S Ty and identify by block the Computer This manual Con has been to reflect un to reflect un s of hardward les among the VAXcluster, ting batch ju bes the Cray | 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 21. ABSTRACT S 220. TELETHONE 220. TELETHONE | A if necessary and a if necessary and IG Cray DEC nation Se led an I in introd from man TRC. Con Ware are CR 860 (w CE DEC VA Ising the Mass St BER 860. SECURITY CLASS SIFIED (Include Area Con 267-3343 | d identify by bl COS UNICOS VAX/VMS rvices E ntegrate uction t y indivi trol sta include ith the X, and t interac orage Sy IFICATION del 22c. OFFICE 351 | 322 Department ed to the dual tement ed, as is Mass the CRAY tive stem and |

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

18 (continued)

Hardware Interactive Mass Storage System Programming Secure computing Software Documentation Supercomputer

David Taylor Research Center Bethesda, Maryland 20084-5000

* x * * × * Computer Center * × Reference Manual × × Volume 1: Cray, MSS, DEC * π ÷ * ***************

by David V. Sommer Sharon E. Good

Scientific and Engineering User Support Branch Code 3511

| | Carderock | | An | napolis |
|---------|-----------|----------|-------|----------|
| Phone | (301) | 227-1907 | (301) | 267-3343 |
| Autovon | | 287-1907 | | 281-3343 |

For recorded message on computer status (301) 227-3043

Questions and requests for more detailed information should be directed to Code 3511, Bldg. 17, Rm. 226

Com_iuter and Information Systems Department Departmental Report

September 1990

. .

CISD-90/01

Through Revision 0 (Sept 1990)

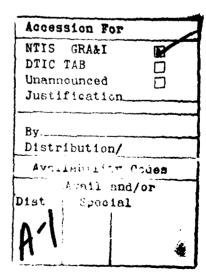
• •

Page i

90/09/30 Rev0

*** Revision Record ***

| Revision | Description |
|------------|--------------------|
| 0 (Sep 90) | Original printing. |



.

*** List of Effective Pages ***

| Title0i thru ix0 $1-1-1$ thru ix0 $1-2-1$ thru $1-2-8$ 0 $1-3-1$ thru $1-3-2$ 0 $1-4-1$ thru $1-4-2$ 0 $1-5-1$ 0 $2-1-1$ 0 $3-1-1$ thru $3-1-17$ 0 $3-2-1$ thru $3-2-8$ 0 $3-3-1$ thru $3-3-7$ 0 $3-4-1$ thru $3-4-6$ 0 $3-5-1$ thru $3-5-3$ 0 $3-6-1$ thru $3-6-16$ 0 $4-1-1$ thru $4-1-11$ 0 $5-1-1$ thru $5-2-12$ 0 $5-3-1$ 0 $5-4-1$ thru $5-4-3$ 0 $5-5-1$ thru $5-5-3$ 0 $5-6-1$ thru $5-6-3$ 0 $6-1-1$ thru $6-1-8$ 0 $7-1-1$ thru $7-1-18$ 0 $A-1$ thru $A-4$ 0 $B-1$ 0 $C-1$ thru $C-64$ 0 $D-1$ thru $C-64$ 0 $C-1$ thru $C-64$ 0 $C-1$ thru $G-9$ 0 $G-1$ thru $G-1$ 0 $G-1$ thru $G-1$ 0 $G-1$ thru $G-1$ 0 $G-1$ thru $G-1$ 0 <th>Rev</th> | Rev |
|--|-----|
| i thru ix 0 1-1-1 thru $1-1-5$ 0 1-2-1 thru $1-2-8$ 0 1-3-1 thru $1-3-2$ 0 1-4-1 thru $1-4-2$ 0 1-5-1 0 2-1-1 0 3-1-1 thru $3-1-17$ 0 3-2-1 thru $3-2-8$ 0 3-3-1 thru $3-7$ 0 3-4-1 thru $3-6-6$ 0 3-5-1 thru $3-5-3$ 0 3-6-1 thru $3-6-16$ 0 4-1-1 thru $4-1-11$ 0 5-1-1 thru $5-1-8$ 0 5-2-1 thru $5-2-12$ 0 5-3-1 0 5-4-1 thru $5-5-3$ 0 5-6-1 thru $5-6-3$ 0 5-6-1 thru $5-6-3$ 0 5-6-1 thru $5-6-3$ 0 5-1-1 thru $4-1-8$ 0 7-1-1 thru $4-1-8$ 0 7-1-1 thru $4-1-8$ 0 7-1-1 thru $7-1-18$ 0 A-1 thru $A-4$ 0 B-1 0 C-1 thru $C-64$ 0 D-1 thru $C-64$ 0 D-1 thru $C-64$ 0 C-1 thru $C-64$ 0 D-1 thru $F-4$ 0 G-1 thru $F-4$ 0 G-1 thru $G-9$ 0 G1-1 0 Index-1 thru Index-30 0 | |
| $\begin{array}{cccccccccccccccccccccccccccccccccccc$ | |
| 3-4-1 thru 3-4-6 0 3-5-1 thru 3-5-3 0 3-6-1 thru 3-6-16 0 4-1-1 thru 3-6-16 0 4-1-1 thru 3-6-16 0 5-1 thru 4-1-11 0 5-1 thru 5-1-8 0 5-2-1 thru 5-2-12 0 5-3-1 0 5-5-1 0 5-4-1 thru 5-4-3 0 5-5-1 thru 5-5-3 0 5-6-1 thru 5-6-3 0 6-1-1 thru 6-1-8 0 7-1-1 thru 7-1-18 0 A-1 thru 7-1-18 0 A-1 thru 0 0 E-1 thru 0 0 E-1 thru 0 0 E-1 thru 0 0 G-1 thru 0 0 G1-1 0 0 0 < | |
| 3-5-1 thru 3-5-3 0 3-6-1 thru 3-6-16 0 4-1-1 thru 3-6-16 0 4-1-1 thru 4-1-11 0 5-1-1 thru 5-1-8 0 5-2-1 thru 5-2-12 0 5-3-1 0 0 0 5-4-1 thru 5-4-3 0 5-5-1 thru 5-5-3 0 5-6-1 thru 5-6-3 0 6-1-1 thru 6-1-8 0 7-1-1 thru 7-1-18 0 A-1 thru 7-4 0 B-1 0 0 0 C-1 thru 0 0 E-1 thru 0 0 E-1 thru 0 0 F-1 thru 6-9 0 G-1 0 0 0 Index-1 thru 1 0 | |
| 3-6-1 thru 3-6-16 0 4-1-1 thru 4-1-11 0 5-1-1 thru 5-1-8 0 5-2-1 thru 5-2-12 0 5-3-1 0 5-4-1 thru 5-4-1 thru 5-4-3 0 5-5-1 thru 5-5-3 0 5-6-1 thru 5-6-3 0 6-1-1 thru 6-1-8 0 7-1-1 thru 7-1-18 0 A-1 thru 0 0 F-1 thru 0 0 F-1 thru 0 0 F-1 thru F-4 0 F-1 thru F-4 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 0 0 Index-1 thru Index-30 0 | |
| $\begin{array}{cccccccccccccccccccccccccccccccccccc$ | |
| 5-1-1 thru 5-1-8 0 $5-2-1 thru 5-2-12 0$ $5-3-1 0$ $5-4-1 thru 5-4-3 0$ $5-5-1 thru 5-5-3 0$ $5-6-1 thru 5-6-3 0$ $6-1-1 thru 6-1-8 0$ $7-1-1 thru 7-1-18 0$ $A-1 thru A-4 0$ $B-1 0$ $C-1 thru C-64 0$ $D-1 thru D-49 0$ $E-1 thru E-2 0$ $F-1 thru F-4 0$ $G-1 thru G-9 0$ $G1-1 0$ $Index-1 thru Index-30 0$ | |
| 5-2-1 thru 5-2-12 0 $5-3-1 0$ $5-4-1 thru 5-4-3 0$ $5-5-1 thru 5-5-3 0$ $5-6-1 thru 5-6-3 0$ $6-1-1 thru 6-1-8 0$ $7-1-1 thru 7-1-18 0$ $A-1 thru A-4 0$ $B-1 0$ $C-1 thru C-64 0$ $D-1 thru D-49 0$ $E-1 thru E-2 0$ $F-1 thru F-4 0$ $G-1 thru G-9 0$ $G1-1 0$ $Index-1 thru Index-30 0$ | |
| 5-3-1 0 5-4-1 thru 5-4-3 0 5-5-1 thru 5-5-3 0 5-6-1 thru 5-6-3 0 6-1-1 thru 6-1-8 0 7-1-1 thru 7-1-18 0 A-1 thru A-4 0 B-1 0 C-1 thru C-64 0 D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru G-9 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| 5-4-1 thru 5-4-3 0 5-5-1 thru 5-5-3 0 5-6-1 thru 5-6-3 0 6-1-1 thru 6-1-8 0 7-1-1 thru 7-1-18 0 A-1 thru A-4 0 B-1 0 0 0 C-1 thru C-64 0 D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G-1 thru G-9 0 G-1 thru G-9 0 G-1 thru G-9 0 G-1 thru Index-30 0 | |
| 5-5-1 thru 5-5-3 0 5-6-1 thru 5-6-3 0 6-1-1 thru 6-1-8 0 7-1-1 thru 7-1-18 0 A-1 thru A-4 0 B-1 0 0 0 C-1 thru C-64 0 D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 0 0 | |
| 5-6-1 thru 5-6-3 0 6-1-1 thru 6-1-8 0 7-1-1 thru 7-1-18 0 A-1 thru A-4 0 B-1 0 C-1 thru C-64 0 D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| 6-1-1 thru 6-1-8 0 7-1-1 thru 7-1-18 0 A-1 thru A-4 0 B-1 0 C-1 thru C-64 0 D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| 7-1-1 thru 7-1-18 0 A-1 thru A-4 0 B-1 0 C-1 thru C-64 0 D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| 7-1-1 thru 7-1-18 0 A-1 thru A-4 0 B-1 0 C-1 thru C-64 0 D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| B-1 0 C-1 thru C-64 0 D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| C-1 thru C-64 0 D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| D-1 thru D-49 0 E-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| E-1 thru E-2 0 F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| F-1 thru F-4 0 G-1 thru G-9 0 G1-1 0 Index-1 thru Index-30 0 | |
| G1-1 0 Index-1 thru Index-30 0 | |
| Index-1 thru Index-30 0 | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

Contents

Volume 1

| | Preface | |
|---|---|-------|
| | Revision Record | i |
| | List of Effective Pages | ii |
| | | |
| 1 | Introduction | 1-1-1 |
| | Hardware Configuration | 1-1-2 |
| | CRAY X-MP / 216 | 1-1-2 |
| | CDC CYBER 180 model 860 | 1-1-2 |
| | DEC VAXcluster | 1-1-3 |
| | DEC VAX | 1-1-3 |
| | DEC Secure VAXcluster | 1-1-3 |
| | DTNET Dial-in | 1-1-4 |
| | The Integrated Supercomputer Network | 1-1-5 |
| | User Interface With the Computer Center | 1-2-1 |
| | General Information | 1-2-1 |
| | Registering | 1-2-1 |
| | Multiple Accounts | 1-2-2 |
| | Cray COS | 1-2-2 |
| | VAX | 1-2-3 |
| | CDC NOS | 1-2-4 |
| | Mass Storage System | 1-2-5 |
| | Passwords, Passwords, Everywhere | 1-2-6 |
| | Trouble Forms | 1-2-8 |
| | Refunds | 1-2-8 |
| | ADP Control Center | 1-2-8 |
| | Software Available | 1-3-1 |
| | Prolog and Epilog Command Files | 1-4-1 |
| | VAX/VMS Command Files | 1-4-1 |
| | NOS Command Files | 1-4-1 |
| | NOS/VE Command Files | 1-4-2 |
| | Cray Command Files | 1-4-2 |
| | ANSI Standard Multi-file Tapes | 1-5-1 |
| | VAX | 1-5-1 |
| | NOS | 1-5-1 |
| | NOS/VE | 1-5-1 |
| 2 | The CRAY X-MP (UNICOS) | 2-1-1 |
| - | UNICOS Version 5.1 | 2-1-1 |
| | | 2.1.1 |
| 3 | The CRAY X-MP (COS) | 3-1-1 |
| | COS Version 1.17.1 | 3-1-1 |
| | Accessing the CRAY X-MP | 3-1-1 |



I

| Cray Datasets | 3-1-1 |
|--|----------------|
| Changing your Cray password | 3-1-2 |
| Batch Jobs | 3-1-3 |
| Batch Job Classes | 3-1-4 |
| SECURE Batch Job Class | 3-1-4 |
| From the VAXcluster | 3-1-5 |
| Killin, Batch Jobs | 3-1-6 |
| VAXcluster-to-Cray Examples | 3-1-6 |
| From the CDC CYBER 860 | 3-1-8 |
| Killing Batch Jobs | 3-1-8 |
| CYBER 860-to-Cray Examples | 3-1-9 |
| From a Running Cray Job | 3-1-10 |
| Examples | 3-1-10 |
| Interactive Jobs | 3-1-11 |
| From the VAXcluster | 3-1-11 |
| VMS Cray Station Commands | 3-1-12 |
| Examples | 3-1-15 |
| From the CDC CYBER 860 | 3-1-16 |
| ICF Prologue File | 3-1-16 |
| NOS ICF User Commands | 3-1-16 |
| Examples | 3-1-17 |
| Crew ICL Comments | 2_2_1 |
| Cray JCL Commands Job Definition and Control | 3-2-1 3-2-1 |
| | 3-2-1 |
| Dataset Definition and Control | 3-2-2 |
| Permanent Dataset Management | 3-2-2 |
| Permanent Dataset Staging Permanent Dataset Utilities | 3-2-2 |
| Local Dataset Utilities | 3-2-3 |
| | 3-2-3 |
| Dumps and Other Aids Logic Structure | 3-2-4 |
| Procedures | 3-2-4 |
| Programming Languages | 3-2-4 |
| Program Libraries | 3-2-5 |
| Object Libraries | 3-2-5 |
| Miscellaneous | 3-2-5 |
| JCL Expressions | 3-2-6 |
| Symbolic Variables | 3-2-6 |
| System Constants | 3-2-6 |
| COS-set Variables | 3-2-6 |
| User-set Variables | 3-2-7 |
| | 3-2-7 |
| Operators | 3-2-7 |
| Strings | J 4 / |
| Procedures | 3-3-1 |
| Simple Procedures | 3-3-1 |
| Complex Procedures | 3-3-1 |
| Prototype Statement | 3-3-2 |
| Temporary Datasets | 3-3-2 |
| Parameter Substitution | 3-3-3 |
| Apostrophes and Parentheses | 3-3-3 |
| DTRC "System" Procedures | 3-3-4 |
| DTRC Procedure Library | 3-3-4 |
| Examples | 3-3-5 |
| Simple Procedures | 3-3-5 |
| Complex Procedures | 3-3-5 |
| combrea records da | |

Page v

| Program Libraries UPDATE UPDATE Directives DECK and COMDECK Compile Directives Modification Directives Run Options Input Edit Directives Examples | 3-4-1 3-4-1 3-4-1 3-4-2 3-4-2 3-4-2 3-4-3 3-4-4 3-4-5 |
|--|---|
| Object Libraries BUILD Directives DTRC Object Libraries Examples | 3-5-1 3-5-1 3-5-1 3-5-2 |
| Loader SEGLDR Control Statement Message Levels Directive Segmentation Segmentation Directives Sample Tree Diagram Segmentation Cautions Compile, Load and Save an Absolute Program Simple Load Segmented LOAD | $3-6-1 3-6-1 3-6-1 3-6-2 3-6-10 3-6-10 3-6-14 3-6-15 3-6-16 3-6-16 3-6-16 3-6-16 3-6-16 \\ 3-6-1$ |
| The Mass Storage System MSS Security MSS File Purge MSS Backup for Critical Files Using the MSS from the Cray Using the MSS from the VAXcluster Using the MSS from the CDC CYBER 860 | 4-1-1 4-1-1 4-1-2 4-1-3 4-1-6 4-1-9 |
| DEC VAXcluster VMS VMS Version 5.3 Accessing the VAXcluster Login Password Logout Procedures System News Login Procedure File File Batch Jobs Killing Batch Jobs Accessing Other Networks Checking Host Accessibility Transferring VMS Files To and From OASYS Transferring VMS Files To and From CDC CYBER 860 Mail to Users at Other Sites Mail From Users at Other Sites | 5-1-1 5-1-1 5-1-2 5-1-2 5-1-2 5-1-2 5-1-3 5-1-4 5-1-4 5-1-4 5-1-5 5-1-5 5-1-6 5-1-6 5-1-7 5-1-8 |



4

5

Page vi

| Help Libraries | 5-2-1 |
|--|----------------|
| The System Help Library | 5-2-1 |
| DTRC Help Libraries | 5-2-1 |
| User Help Module | 5-2-2 |
| | 5-2-3 |
| Hints For Designing Help Displays | 5-2-3 |
| Selecting (Sub)topic Names | 5-2-3 |
| Modify a Help Library | |
| Compress a Help Library | 5-2-4 |
| List the Contents of a Help Library | 5-2-5 |
| Extract a Help Module | 5-2-5 |
| Accessing your Help Library | 5-2-5 |
| Adding Your Help Library to the System Helps | 5-2-6 |
| Using HELP | 5-2-6 |
| Sample Help Modules | 5-2-7 |
| A Program | 5-2-7 |
| A Subprogram | 5-2-9 |
| A Command Procedure | 5-2-10 |
| General Information | 5-2-11 |
| "HELP" module | 5-2-12 |
| Duran dama a | 5-3-1 |
| Procedures | 5-3-1 5-3-1 |
| DTRC Procedures | 5-3-1 |
| Object Libraries | 5-4-1 |
| DTRC Object Library | 5-4-1 |
| User Object Module | 5-4-1 |
| Modify an Object Library | 5-4-2 |
| Compress an Object Library | 5-4-2 |
| List the Contents of an Object Library | 5-4-3 |
| Extract an Object Module | 5-4-3 |
| Linking with an Object Library | 5-4-3 |
| Linking with an object library | |
| Text Libraries | 5-5-1 |
| DTRC Text Libraries | 5-5-1 |
| User Text Module | 5-5-1 |
| Create a Text Library | 5-5-1 |
| Modify a Text Library | 5-5-2 |
| Compress a Text Library | 5-5-3 |
| List the Contents of a Text Library | 5-5-3 |
| Extract a Text Module | 5-5-3 |
| | |
| Editors | 5-6-1 |
| The EDT Text Editor | 5-6-1 |
| Invoking EDT | 5-6-1 |
| Terminating EDT | 5-6-1 |
| The EVE Editor | 5-6-2 |
| Invoking EVE | 5-6-2 |
| The Screen | 5-6-2 |
| On-line Help | 5-6-2 |
| Terminating EVE | 5-6-2 |
| Why Use EVE Instead of EDT? | 5-6-3 |
| - | |
| | |
| Magnetic Tape | 6-1-1 |
| Terre Lebele | 6-1-1 |

i

| 6 | Magnetic Tape | 6-1-1 |
|---|---------------|-------|
| | Tape Labels | 6-1-1 |
| | Tape Formats | 6-1-1 |

| | 90/09/30 Rev0 | Page vii |
|---|--|---------------------|
| | Tape Care and Cleaning Tape Assignment | 6-1-2 6-1-3 |
| | Using Tapes on the DEC VAX Examples | 6-1-4 6-1-5 |
| | Using Tapes on the CYBER 860 NOS | 6-1-6 6-1-6 |
| | Examples NOS/VE | 6-1-6 6-1-7 |
| | Examples | 6-1-7 |
| 7 | Other Software | 7-1-1 |
| | Appendices | |
| A | Appendix A | A-1 |
| | ASCII Character Set CDC NOS Character Set | A-1 A-3 |
| В | Appendix B Cray UNICOS Commands | B-1 B-1 |
| с | Appendix C | C-1 |
| | Cray COS JCL Commands Strings | C-1 C-2 |
| | Some Common Parameters Permanent Dataset Utility Shorthand Nota ion | C-2 C-4 |
| | A Word About Continuations Summary of Cray JCL Commands | C-4 C-5 |
| D | Appendix D DEC VMS DCL Commands | D-1 |
| | Selected DEC VAX/VMS Commands | D-1 D-2 |
| | Selected DEC VAX/VMS Additions Cray Station Commands Cray Context Commands | D-6 D-30 D-32 |
| E | Appendix E References | E-1 E-1 |
| | Cray | E-1 |
| | DEC CDC NOS | E-1 E-1 |
| | CDC NOS/VE General | E-1 E-2 |
| | OCHET GT | E-2 |

FAppendix FF-1CCF Computer SystemsF-1Services and SupportF-4

| G | Appendix G | G-1 |
|---|-------------------------|-----|
| | Internal Data Structure | G-1 |
| | Internal Representation | G-6 |
| | CRAY X-MP | G-6 |
| | DEC VAX | G-7 |
| | CDC CYBER (NOS, NOS/BE) | G-8 |
| | CDC CYBER (NOS/VE) | G-9 |
| | | |
| | | |

Gl Glossary G1-1

Index

Index-1

Abstract

The Computer Center in the Computer and Information Services Department of the David Taylor Research Center has installed an Integrated Supercomputer Network. This manual provides an introduction to the Network. Some information has been distilled from many individual documents and augmented to reflect usage at DTRC. Control statement examples and descriptions of hardware and software are included, as is information on moving files among the CDC CYBER 860A (with the Mass Storage System), the DEC VAXcluster, the secure DEC VAX, and the CRAY X-MP, creating and executing batch jobs, and using the interactive systems. Volume 1 describes the CRAY X-MP, the Mass Storage System, and the DEC VAXes. Volume 2 describes the CDC CYBER 860A.

Administrative Information

The work described in this report was performed in the Scientific and Engineering Support Branch (3511) of the Computer and Information Services Department, David Taylor Research Center, under the sponsorship of the DTRC Computer Center (351).

***** Introduction *****

The DTRC Integrated Supercomputer Network consists of a CRAY X-MP/216 with five front-end computers: a DEC VAXcluster (two VAX 8550 processors), a secure DEC VAX 6410, a DEC VAX 8250, and a CDC CYBER 180/860. The Cray and VAXcluster can store and retrieve files on the Mass Storage System (MSS), which is part of the CDC CYBER 860.

The following operating systems are in use:

| CRAY X-MP | COS | version | 1.17.1 |
|---------------|--------|---------|----------|
| | UNICOS | | (future) |
| DEC VAX | VMS | version | 5.3-1 |
| CDC CYBER 860 | NOS | version | 2.7.1 |
| | NOS/VE | version | 1.5.1 |

The front-end computers support both batch processing of jobs submitted at central site, through remote batch terminals or from interactive terminals; and demand processing, which supports a variety of interactive terminals. In addition, batch jobs can be sent to the Cray for processing with the output returned for examination or printing.

This reference manual is divided into two volumes: one covering the CRAY X-MP, the Mass Storage System, and the DEC VAXes; the other covers the CDC CYBER 860. They are designed to provide the new user with enough information to use the Network to run simple batch jobs and to create and run programs and batch jobs interactively. Most of the frequently used control statements are described in detail in the Appendices. Magnetic tapes are discussed briefly. No attempt is made to describe all features of the operating systems or even all parameters of the control statements presented. More information can be found in the publications listed in Appendix E.

Before using the system, job order number(s) to be charged must be registered with Code 3502. Outside users must transfer funds to DTRC before receiving a job order number. Each individual user should have 4-character User Initials assigned (also by Code 3502).

*** *** Hardware Configuration ** ** CRAY X-MP/216 sn417 DTNET name: (UNICOS only, when available) TCP/IP name: sn417 (UNICOS only, when available) Ethernet address: 192.91.138.5 (UNICOS only, when available) Cray station ID: Cl 2 X-MP central processing units (over 200 MFLOPS each) 16M 64-bit words of central memory 4 model DD-49 disk storage units (4.8 Gbytes) 2 model DS-41 disk storage units (9.6 Gbytes) ** ** CDC CYBER 180 model 860 DTNET names: cdc860, nos TCP/IP names: cdc860, nos Ethernet address: 130.46.1.16 Cray station ID: N1 N (UNICOS only, when available) MFN Network ID: 1 CYBER 860A central processing unit (6.3 mips) 2M 60-bit word memory 25 peripheral processors 3 model 895 disk drives 4 model 679-5 nine-track tape drives (1600/6250 cpi) 2 model 679-3 nine-track tape drives (800/1600 cpi) 2 model 677-3 seven-track tape drives 1 model 405 card reader 1 model 415 card punch line printers (1200 1pm, upper/lower case) 2 model 585 1 model 7990 Mass Storage System (210 Gbytes) 3 model M861 storage modules

91/01/15 Rev0

** DEC VAXcluster ** VAXcluster nodes: DT3, DT4 DTNET names: dt3, dt4 TCP/IP names: dt3, dt4 Ethernet addresses: 130.46.1.12, .10 Cray station IDs: V3 ٧4 ٧_ (UNICOS only, when available) 2 VAX 8550 processors (6 mips each; DT3, DT4) -- each with 48 Mbyte 32-bit words of central memory 2 model SA482 disk storage array (5.0 Gbytes) 6 model RA81 disk drives (7.2 Gbytes) 1 model TA79 nine-track tape drives (1600/6250 cpi) 3 model TU79 nine-track tape drives (1600/6250 cpi) 2 model LP27 impact printers (800 lpm, upper/lower case) ** DEC VAX ** VAX node: DOE * DTNET name: doe TCP/IP name: doe Ethernet addresses: 130.46.1.13 1 VAX 8250 processors (1.2 mips) 16 Mbyte 32-bit words of central memory 4 model RA81 disk drives (1.6 Gbytes) 2 model TU81 nine-track tape drives (1600/6250 cpi) ** ** DEC Secure VAXcluster VAXcluster nodes: DBL07 Cray station IDs: (future) 1 VAX 6410 processors (7 mips) 64 Mbyte 32-bit words of central memory 2 model SA482 disk storage array (5.0 Gbytes) 6 model RA81 disk drives (2.4 Gbytes) 2 model TA78 nine-track tape drives (1600/6250 cpi) 1 model LP27 impact printers (800 lpm, upper/lower case)

* DOE is reserved for a special project and is not available for

general use.

Page 1-1-5

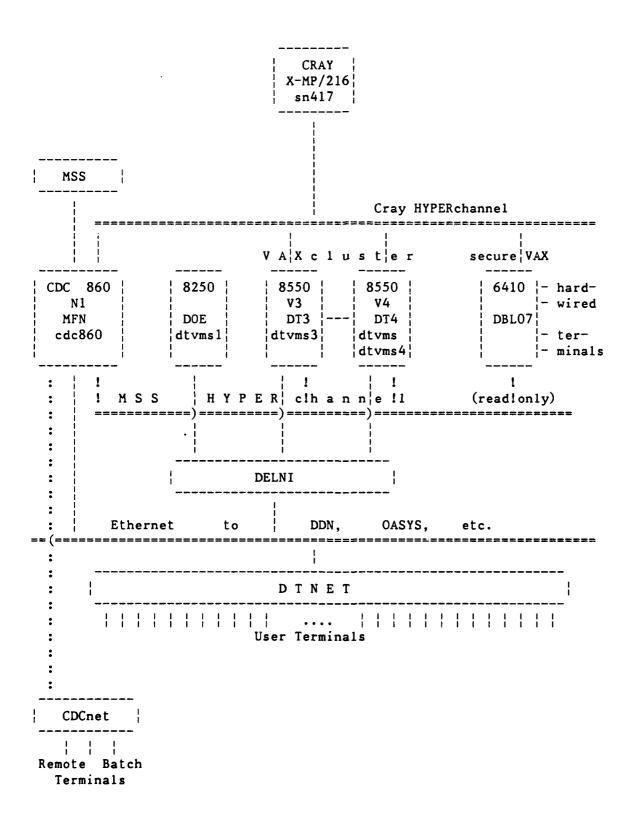
** DTNET Dial-in **

Access to Computer Center computers is via DTNET:

| Modem Speed | Carderock | Annapolis | |
|--------------------------|--|---------------------------------|------------------------------------|
| 1200 baud | (301) 227-5200 (32) | (301) 267-2010 (8) x4741 (8) | (within Annapolis site only) |
| 2400 baud 9600 baud * | (301) 227-5250 (32) (301) 227-3700 (16) | | - |

* While 9600 baud modems support lower speeds, access at a lower speed is not guaranteed.

The Integrated Supercomputer Network ***



Page 1-2-1

***** ***** User Interface With the Computer Center

*** *** General Information

The ADP Control Centers are located at Central Site at Carderock and Annapolis. You may submit tapes and pick up output from an ADP Control Center.

Computer and Information Systems Department Newsletter is our quarterly publication. The date of the latest on-line news update is printed at the start of each batch job (log) or interactive session. The NEWS command or procedure is used to view the current news file.

*** *** Registering

To register to use Computer Center computers, call our Business Office, Code 3502, at (301) 227-1361/1910. Be prepared to supply

- . your name
- . your DTRC code or non-DTRC company name and address and telephone
- . the job order number(s) to be charged for computer work
- . the computers on which you which to be registered
 - . CRAY X-MP
 - . DEC VAXcluster
 - . DEC secure VAX
 - . CDC CYBER 860 (NOS or NOS/VE)

Registration for the DEC VAXcluster or CRAY X-MP includes registration for the Mass Storage System (CDC CYBER 860 (NOS)).

You will be given

- . User Initials (if you are a new user)
- . the initial passwords (which MUST be changed during your first session) for each computer system for which you registered





Page 1-2-2

Rev0 90/09/30

** Multiple Accounts **

The following is a discussion of multiple accounts: how to log in, how to charge batch jobs, how to change permanent file accounting.

* Cray COS *

Interactive

and batch The account number is supplied by the AC= parameter of the ACCOUNT statement at the start of each batch or interactive session:

ACCOUNT, AC=....

Permanent

files

Files are normally charged to the interactive or batch session's account number. The "ALTACN,AC=ac." statement is used to change the number for future file saves. It remains in effect until another ALTACN statement is encountered, or until end-of-session.

The account number for an existing file can be changed to current session's account number by

ACCESS, DN=PROCLIB, OWN=PUBLIC. LIBRARY, DN=PROCLIB:*. NEWCHRG, OLD=oldchrgno, ID=id.

where ID=id changes only those files with matching ID ID changes only those files with a null ID no ID changes all files.

Alternatively, this can be done from a VAXcluster node which connects to the Cray. CNEWCHRG generates and submits a Cray job to make the changes.

CNEWCHRG upw old_ac [new_ac] [id] [wait] [type]

where upw is your user password ID=id changes only those files with matching ID ID changes only those files with a null ID no ID changes all files wait is WAIT to wait the the Cray job to complete type is TYPE to display the generated Cray job

Page 1-2-3

* VAX *

VAX users with more than one account are assigned a username/ password for each account. These usernames differ in the fifth character position, e.g., ABCD, ABCDA, ABCDB. The default login directory for each user is device: [username] where all files owned by the same individual are stored on the same device. For example,

U01: [ABCD] U01: [ABCDA] U01: [ABCDB]

The "usernames" belonging to a particular user are members of a VMS "group". By default on the VAXcluster, members of a group have Read and Execute access to all files owned by their fellow group members. User ABCDA wishing to access a file owned by ABCD simply references [ABCD] FILE.EXT .

These access rights can be changed by the SET PROTECTION and SET FILE /ACL commands. In addition, all members of these special "groups" have GRPPRV privilege which, when invoked, gives a member of the group full control, including file creation and deletion, over all files owned by all members of the group. GRPPRV is invoked by

\$ SET PROCess /PRIVileges=GRPPRV

(this would likely be in your LOGIN.COM)

Then to "copy" a file from one account to another, for example from ABCD to ABCDA, user ABCDA would

\$ COPY [ABCD] FILE.EXT []

or user ABCD would

\$ COPY FILE.EXT [ABCDA]

To simply "move" a file from one account to another, ABCDA would

The command MYACcount will indicate the account number of the current session or job, while MYACcount /ALL will provide a list of all user/account pairs in the group.

Page 1-2-4

Rev0 90/09/30

* CDC NOS *

- Interactive You are normally prompted for the account number for the session when you log in. (See Appendix I: CHVAL, CN.)
- Batch A batch job is charged to the number appearing on the actual (CHARGE, accountno.) or implied (/CHARGE) statement at the start of the job.

Permanent

files Files are normally charged to the interactive or batch session's account number. The CHARGE statement is used to change the number for future file saves. It remains in effect until another CHARGE statement is encountered, or until end-of-session.

The account number for an existing file can be changed to current session's account number by

CHANGE, pfn/CP.

To change several files, use

BEGIN, NEWCHRG, , fn.

where fn may have wildcards. To change all files, use

BEGIN, NEWCHRG.

• .• · ·

Page 1-2-5

* Mass Storage System *

The MSS is part of the CDC CYBER 860. The account numbers on files may be changed from the DEC VAXcluster, the CRAY X-MP, or on the 860.

From the DEC VAXcluster: \$ HFT ACCESS /Password=password \$ MSSNewchrg old_ac new_ac ^-- all MSS files with old_ac are changed

From the CRAY X-MP (COS): MSACCES, MPW=mss_pw. <-- may need AC=ac MSCHANG, MDN=mssfile, CP=newacctno.

On the CDC 860 (NOS) : CHANGE, mfn/CP.

From the CDC 860 (NOS/VE): MSACCES mss_pw MSCHANG mfn CP=1 *** Passwords, Passwords, Everywhere ***

Each computer system has its own password to gain access to it (while CDC NOS has two passwords, the CDC CYBER 860 at DTRC has only one). You MUST change these during your first session on each or you will be denied future access. For security, you are strongly urged to change your access passwords as soon as you can log into each computer. Passwords for all our computers expire in 90 days. For the VAX and CYBER, you must change your password during the session in which you are told it has expired. The new password must differ from the old one. On the VAX, it must differ from all others used during the previous 12 months.

To change your access passwords, use

. Cray COS (password is 4-15 characters, with at least one of the following characters: 0-9, \$, %, @)

. on the Cray

ACCOUNT, US=username, AC=joborderno, UPW=current_pw, NUPW=new_pw.

. from the VAXcluster

CSUDMIT /NUPW <-- use this form if you use CSUBMIT current_pw to submit jobs to the Cray new_pw new_pw

-or-

CNEWPW current_pw new_pw new_pw ac wait ^-- use this form if you do not use CSUBMIT

See also page 3-1-2.

 DEC VAXcluster (paseword is 6-12 characters) DEC secure VAX
 SET PASSWORD <-- you will be prompted for your current and new passwords

Page 1-2-7

. CDC CYBER 860 / MSS (NOS) (password is 4-7 characters, at least one rumber)

. from the VAXcluster

HFT PASSWORD <-- you will be prompted for your current and new passwords

. from the Cray (COS)

...SACCES, MPW=mpw. MSPASSW, OLD=oldpw, NEW=newpw.

. on the 860 (NOS)

PASSWOR, oldpw, newpw.

. from the 860 (NOS/VE)

MSACCES mpw MSPASSW oldpw newpw

. CDC CYBER 860 / MSS (NOS/VE) (password is up to 31 alphanumeric characters and underscores, starting with a letter)

Since NOS/VE is reached via NOS, your NOS login password gives you access to NOS/VE. However, to run NOS/VE batch jobs, you must have defined your batch password. To change it,

SET_PASSWORD <-- you will be prompted for your current and new passwords Page 1-2-8

*** Trouble Forms ***

A Trouble Form is used:

1) for refund requests

- 2) when problems are encountered
- 3) for suggestions, gripes and complaints.

The Trouble Form should include a succinct description of the problem and include as much documentation (dayfile or log, listings, dumps) as possible. It should be submitted to Code 3511 for processing.

Trouble Forms may be entered directly into the computer from any of the front-ends (VAXcluster, CYBER 860) using the GRIPE command. If supporting documentation (such as listings or dumps) is needed, please send it to Code 3511 (User Services).

*** Refunds ***

Requests for refunds on lost time must be accompanied by output of the run and a Trouble Form, and must be reported within five working days. Decisions on refunds will be made by Code 35.

*** ADP Control Center ***

The ADP Control Center has the following capabilities:

. assign, sell, clean, test and degauss magnetic tapes

. process Calcomp plots, and Xerox and microfiche output

. sell frequently-used terminal paper and ribbons

The following EAM facilities are available off-line at Central Site:

- . a small card interpreter
- . card punch
- . card verifier
- . shredder

See Appendix F for Computer Center telephone numbers.

***** Software Available *****

The following table lists the major software products and the computers where they are available. Type "HELP @CCF Software" on the VAXcluster for the latest version of this table.

| | DT3/DT4 8550 | DBL07 6410 | CRAY X-MP | CDC NOS | CDC NOS/VE |
|---------------|-----------------|---------------|--------------|------------|---------------|
| ABAQUS | x | - | x | - | _ |
| ACSL | x | - | - | - | - |
| ALGOL | - | - | - | x | - |
| APL | x | - | - | x | - |
| APT | - | - | - | × | - |
| Basic | x | x | - | x | x |
| C (CC) | x | ~ | x | - | - |
| Calcomp | x | - | x | x | (1) |
| CDD | x | x | - | - | - |
| Cobol | x | x | - | x | x |
| Datatrieve | x | x | - | - | - |
| DBMS | x | - | - | - | - |
| DECalc | x | x | _ | - | - |
| DISSPLA | x | - | x | x | - |
| DYNA3D | - | - | x | - | - |
| EISPACK | x | - | x | - | (1) |
| FMS | x | x | - | - | - |
| Fortran | x | x | x | x | x |
| FTP | x | - | - | - | - |
| GPSS | x | - | - | x | - |
| Hasp | - | - | - | x | - |
| HOTSPOT | - | - | - | x | - |
| IMSL | x | - | x | x | x |
| INGRES | (2) | - | - | - | - |
| Kermit | x | x | - | x | - |
| LINPACK | x | - | x | x | (1) |
| Macsyma | (2) | - | - | _ | _ |
| MODE4A/4C (20 | | - | - | x | - |

(1) - coming

(2) - DT4 only

Page 1-3-2

| | DT3/DT4 8550 | DBL07 6410 | CRAY X-MP | CDC NOS | CDC NOS/VE |
|------------------|-----------------|---------------|--------------|------------|---------------|
| Nastran | _ | - | x | | - |
| Pascal Patran | x x | x _ | x | x _ | x - |
| PCA | x | - | - | - | - |
| Pert Time | - | - | - | x | - |
| PL/I | x | x | - | - | - |
| PLOT10 | x | - | - | _ | - |
| Proj Mgt (PM) | x | - | - | - | - |
| RIM | x | - | - | - | - |
| Simscript | - | _ | - | x | - |
| SPICE | - | - | x | - | - |
| SPY | - | - | x | - | - |
| TAURUS | x | - | - | - | - |
| TELNET | x | (1) | - | - | - |
| WIN/TCP | x | (1) | - | - | - |
| XMODEM | - | - | - | x | - |

(1) coming

Page 1-4-1

***** Prolog and Epilog Command Files *****

A Prolog or Epilog Command File is a file containing commands which are executed automatically each time an event occurs. Prologs are executed at login, at the start of a batch job, at the invocation of an editor, etc. Each DTRC system has one or more such files. Epilogs are executed at logout, at the termination of an editor or utility. NOS/VE supports epilog files.

*** VAX/VMS Command Files ***

A system-level login rile is executed to define global symbols for all users. User file LOGIN.COM, if present, will be executed for each user login or batch job. This procedure file may set up symbols and logical names, establish process name, test whether batch or interactive, or other desired JCL. Type HELP LOGIN.COM_HINTS for suggestions to include in your LOGIN.COM file.

If file EDTINI.EDT exists, it will be executed each time EDT is entered and may be used to define special keys, macros, or other desired Editor commands.

If file EVE\$INIT.EVE exists, it will be executed each time EVE is entered and may be used to define special keys, macros, or other desired Editor commands.

*** NOS Command Files ***

A LOGINPR procedure file may be created containing JCL desired for each login or batch job. For instance, after testing for interactive, TRMDEF could be used to set special desired characteristics for your terminal. Execution of this command file is not automatic, but is triggered by a one-time execution of the UPROC command. If your prolog is long, you may wish to include RECOVER processing.

When the FSE editor is entered, the STARTUP procedure from FSEPROC is executed. The system version of this procedure is currently empty. If you wish to define editor directives which are always executed, make a copy of FSEPROC in your files with desired modifications. NOS will then execute your FSEPROC file instead of the system file.

LOGINPR and FSEPROC must be indirect files.

*** NOS/VE Command Files ***

A PROLOG command file may be created containing SCL desired for each login or batch job. For instance, after testing for interactive, CHANGE_TERMINAL_ATTRIBUTES could be used to set special desired characteristics for your terminal.

An EPILOG command file may be created to be executed at each logout or batch job termination.

When EDIT_FILE is entered, the commands in file SCU_EDITOR_PROLOG are executed. For instance, if you know the keypad or have a layout of it, you may wish to put "SET_SCREEN_OPTION MENU_ROWS=0" into this file to use the bottom line(s) of the screen for editing instead of for displaying the keys.

*** Cray Command Files ***

Before the Cray is used from NOS for the first time, and each time the NOS password is changed, CDEFINE should be used to create a CRAYDEF file. This file frees the user from including CYBER user and password information in each Cray JCL file.

If ICF (Cray Interactive Facility) will be used from NOS, a default PLAY file named ICFPROF may be created which will be executed at each ICF activation. It should contain commands such as /PERIOD to eliminate the need to end each Cray command with a period.

More information about VAX/Cray procedures such as CXCOMMANDS.COM will be included later.

***** ANSI Standard Multi-file Tapes *****

ANSI standard multi-file labelled tapes have each data file delimited by HDR1 and EOF1 labels. In the HDR1 label is the set ID field.

*** VAX ***

The set identifier field is set to the VSN when utilities such as COPY are used to write a multi-file reel.

*** NOS ***

The user defines the multi-file set identifier at the first write of the LABEL with the SI parameter. For ease of recall and compatibility with VAX, the VSN is recommended for the SI. All future reads/writes must use the previously-defined set identifier. If a VAX multi-file reel is read on NOS, the SI field must be set to VAX VSN. If FCOPY is used to create a multi-file tape to be read on a VAX, each FCOPY must be followed by WRITEF to properly create the EOF labels.

*** NOS/VE ***

The multi-file set identifier is created by using the FILE_SET_ IDENITIFIER parameter on the initial CHANGE_TAPE_LABEL_ATTRIBUTES command. The label for each successive file is entered on additional CHANGE_TAPE_LABEL_ATTRIBUTE commands. File specifications, i.e., minimum/maximum record and block sizes, are defined with the SET_FILE_ ATTRIBUTES command.

Page 2-1-1

***** The CRAY X-MP (UNICOS) *****

The CRAY X-MP/216 at DTRC is a powerful, general purpose computer having two central processing units (CPUs) which share files and are linked together. These CPUs share 16 million 64-bit words of memory. Each CPU achieves its extremely high processing rate (over 200 MFLOPS (million floating point operations per second)) using its scalar and vector capabilities.

*** UNICOS Version 5.1 ***

One operating system for the CRAY X-MP at DTRC is the Unix Cray Operating System (UNICOS), version 5.1, which supports both batch and interactive processing. UNICOS is expected to be available at DTRC within the next year.

***** The CRAY X-MP (COS) *****

The CRAY X-MP/216 at DTRC is a powerful, general purpose computer having two central processing units (CPUs) which share files and are linked together. These CPUs share 16 million 64-bit words of memory. Each CPU achieves its extremely high processing rate (over 200 MFLOPS (million floating point operations per second)) using its scalar and vector capabilities.

*** COS Version 1.17.1 ***

The operating system for the CRAY X-MP at DTRC is the Cray Operating System (COS), version 1.17.1, which supports both batch and interactive processing.

*** Accessing the CRAY X-MP ***

Batch jobs are normally submitted from one of the front-ends using: CSUBMIT or CRAY SUBMIT on the VAXcluster, or CSUBMIT on the CDC CYBER 860. They may also be submitted from a running batch or interactive job using the Cray SUBMIT command.

Interactive access is also from one of the front-ends using: CINT (station code version 4.02) on the VAXcluster, or ICF (Interactive Cray Facility) on the CDC CYBER 860 (NOS).

Both modes of access are described later in this chapter.

*** Cray Datasets ***

On the Cray, information is organized by COS into datasets, which may be on disk, memory-resident, or interactive. A dataset contains one or more files and may be temporary (available only to the job that created it) or permanent.

Each dataset has a disposition code to tell COS what to do with it when it is released. The 2-character alphanumeric disposition codes include SC (scratch - default), PR (print), IN (input), and ST (stage to the front end).

Jobs access local datasets, which may be temporary or permanent. Permanent datasets are made local by the ACCESS statement. Front end files are made local by the FETCH statement. *** Changing your Cray password ***

Your Cray access password may be changed from a batch job or interactively on the Cray, or from a procedure on the VAXcluster which creates and submits a Cray batch job for you.

Via DCL command CSUBMIT:

\$ CSUBMIT /NUPW

You will be prompted for your old and new passwords and a Cray job will be submitted on your behalf to change your password. The database on the VAXcluster will be updated with your new password.

If you submit jobs using CSUBMIT, use CSUBMIT to change your password. If you use CRAY SUBMIT or CNEWPW to make the change, the database will not be updated until you use CSUBMIT/NUPW.

Batch:

\$ CRAY SUBMIT mynewpw.job

where your file MYNEWPW.JOB contains:

JOB, JN=ssss. ACCOUNT, AC=ac, US=us, UPW=current_pw, NUPW=new_pw.

Interactive:

\$ CINT \
Cray Jobname: jobname > or CINT /JN=jobname /US=username
Cray Username: username /
!ACCOUNT,AC=ac,US=us,UPW=current_pw,NUPW=new_pw.
!^Z <-- ctrl-Z
Cint> QUIT
\$ <--- you are back in DCL</pre>

Via DCL command CNEWPW:

> (you will have file NUCRPW.CPR when it completes)

This procedure creates and deletes temporary file N\$U\$P\$W.JOB.

Page 3-1-3

*** Batch Jobs ***

Cray batch jobs are very similar to CDC batch jobs, but with different terminology. A batch job consists of one or more files. The first file is the JCL control statement file. It is followed by source or data files as needed by the JCL file. A typical job consisting of one source and one data file (*) looks like this:

> JOB, JN=jobname,.... ACCOUNT, AC=job_order_number, US=username, UPW=password.

<JCL statements>

/EOF

<source file>

/EOF

<-- end-of-file

<-- end-of-file

<data file>

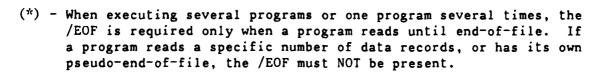
<eod>

<-- end-of-data

A Cray batch job has at least four datasets:

\$CS - the control statement file

- (part of \$IN, but not accessible to the user)
 \$IN the job input dataset. Accessible by its local name,
 \$IN, or as Fortran unit 5.
- \$0UT the job output dataset. Accessible by its local name, \$0UT, or as Fortran unit 6.
- \$LOG a history of the job. Not accessible to the user. \$LOG is appended to \$OUT when the batch job terminates.



Page 3-1-4

Rev0 91/01/15

** Batch Job Classes **

Batch jobs fall into five service classes: NORMAL, DEFER, BUDGET, PZERO, and SECURE. The US= parameter of the JOB statement specifies the job class (there is no default job class). SECURE jobs may be submitted only from the secure VAX (see below). Type HELP RATES (on the VAX), BEGIN, RATES (on CDC NOS), or RATES (on CDC NOS/VE) for the current rates.

Each class, except SECURE and interactive, is broken into four subclasses determined by the memory requested. The first letter of each subclass is:

| S | (small) | <= | 2 | Mwords |
|---|---------------|----|----|--------|
| M | (medium) | <= | 8 | Mwords |
| L | (large) | <= | 12 | Mwords |
| X | (extra large) | > | 12 | Mworde |

The subclasses for NORMAL are SNORM, MNORM, LNORM, XNORM; for DEFER: xDEFER; for BUDGET: xBUDGE; and for PZERO: xPZERO. The subclass names appear in the CRAY STATUS display.

The following chart shows for each job class: the maximum number of such jobs to be allowed to execute at the same time. They are listed in order of relative priority, highest priority first.

| | max | imu | m # | jo | bs | |
|-------------|-----|-----|-----|----|----|--|
| job class | IA | S | M | L | Х | |
| | | - | - | - | - | |
| Interactive | 14 | | | | | |
| NORMAL | | 6 | 4 | r | 1 | |
| NORMAL | | 0 | 4 | 2 | T | |
| DEFER | | 6 | 4 | 2 | 1 | |
| | | | | | | |
| BUDGET | | 6 | 4 | 2 | 1 | |
| | | ~ | • | | | |
| PZERO | | 2 | 3 | T | 1 | |
| SECURE | | | | | | |

** SECURE Batch Job Class **

Classified processing may be done on the CRAY X-MP by making prior arrangement with Operations. Access to the Cray is available only from a terminal in the secure computer room connected to the secure VAX (node name: DBLO7). Batch jobs submitted to the Cray from this terminal must have "US=SECURE" in the job statement; jobs with "US= any_other_class", or with US= omitted, will be rejected. Secure jobs may be submitted at any time but will be executed only during classified time. Secure jobs may access the Mass Storage System in read-only mode.

Page 3-1-5

** From the VAXcluster **

To use the Cray from the VAXcluster, log in to a node which can access the Cray, prepare your Cray batch job using any editor, and submit the job file(s) to the Cray using the CSUBMIT command:

\$ CSUBMIT filename -or- \$ CRAY SUBMIT filename

or

\$ CSUBMIT file1, file2,... -or- \$ CRAY SUBMIT file1, file2,...

where filename is a VAXcluster file containing the Cray job (default file extension: .JOB)

The output will be returned to your file jobname.CPR, where jobname is taken from the job statement of the Cray job (JN parameter).

CSUBMIT remembers your Cray password. It ignores an ACCOUNT statement, if present, and creates one from CSUBMIT qualifiers and your VAXcluster login username and account. CSUBMIT can also change your login password. CRAY SUBMIT requires that there be an ACCOUNT statement in the jobfile.

Files sent to the Cray must not have embedded tabs. See Appendix D: DETAB.

Page 3-1-6

/EOF

Rev0 90/09/30

* Killing Batch Jobs *

Cray jobs are identified by their Job Sequence Numbers (jsg). To find the jsq, use \$ CRAY STATUS -or-\$ CRAY **CRAY> STATUS** To kill a batch job, use \$ CRAY DROP isq <-- terminate executing job with EXIT processing -or-\$ CRAY KILL jsq <-- delete a job from the input queue -or-</pre> terminate executing job without EXIT processing -ordelete the output dataset from the output queue * × VAXcluster-to-Cray Examples 1) \$ CSUBMIT JOB1 \$ CRAY SUBMIT JOB1 -orwhere JOB1.JOB contains: JOB, JN=MYJOB. (1) ACCOUNT, US=username, UPW=password, AC=account. (1, 4)CFT. (1)SEGLDR, GO. (1) /EOF PROGRAM ADD (2) DO 10 I=1,5 (2) READ (5, *) N1, N2, N3 (2) N = N1 + N2 + N3(2) WRITE (6, *) N1, N2, N3, N (2) **10 CONTINUE** (2) END (2) /EOF 1 2 3 (3) 4 5 6 (3) 78 9 (3) 10 11 12 (3) 13 14 15 (3)

will submit the job to the Cray with the output returned in file MYJOB.CPR. The ACCOUNT statement (4) is omitted (or ignored) when using CSUBMIT.

Page 3-1-7

2) \$ CSUBMIT RUN2.JOB,RUN2.FOR,RUN2.DAT -or-\$ CRAY SUBMIT RUN2.JOB, RUN2.FOR, RUN2.DAT where RUN2.JOB contains the job control statements ((1) above) RUN2.FOR contains the Fortran source program ((2) above) RUN2.DAT contains the data ((3) above)vill submit the combined files to the Cray with the output returned in file MYJOB.CPR. Note that the /EOF records are not required in this format. 3) \$ CSUBMIT RUN3 \$ CRAY SUBMIT RUN3 -orwhere RUN3.JOB contains: JOB, JN=MYJOB. ACCOUNT, US=username, UPW=password, AC=account. (4) FETCH, DN=PROG3, TEXT='PROG3.FOR'. FETCH, DN=DATA3, TEXT='PROG3.DAT'. CFT, I=PROG3. SEGLDR, GO.

PROG3.FOR on the VAXcluster contains the program (2) above, with "OPEN (5, FILE='DATA3')" before the "DO 10 ..."

PROG3.DAT contains the data (3) above.

Page 3-1-8

Rev0 91/01/15

** From the CDC CYBER 860 **

To use the Cray from the CYBER 860, log in, prepare your Cray batch job using any editor, and submit the job file to the Cray using the CSUBMIT command (assumes you have set up the CRAYDEF file for your current CYBER 860 user name and password via a CDEFINE command):

| /CSUBMIT,lfn. | < | print at | Central Site |
|---------------------|---|----------|--------------------------|
| /CSUBMIT,1fn,RB=un. | < | put into | output queue for user un |
| /CSUBMIT,1fn,TO. | < | put into | your wait queue |

In the last two formats, use QGET to get the file from the queue. See Appendix D for additional parameters. To send the output elsewhere, use the Cray DISPOSE command (see Appendix C).

* Killing Batch Jobs *

Cray jobs are identified by their Job Sequence Numbers (jsq). To find the jsq, use

×

Page 3-1-9

*

1) /CSUBMIT,RUN1.

where local file RUN1 contains: JOB, JN=myjob. ACCOUNT, US=username, UPW=password, AC=account. FETCH, DN=prog3, SDN=myprog, TEXT='GET, myprog.CTASK.'. ^-- indirect file FETCH, DN≈mydata, TEXT='ATTACH, mydata.CTASK.'. <-- direct file CFT, I=prog3. SEGLDR, GO. -or-JOB, JN=myjob. ACCOUNT, US=username, UPW=password, AC=account. FETCH, DN=prog3, SDN=myprog, TEXT='GET, myprog.CTASK.'. ^-- indirect file MSACCES, US=user, MPW=mspass. <-- direct file MSFETCH, DN=mydata. CFT, I=prog3. SEGLDR,GO. MYPROG and MYDATA on the CDC CYBER 860 contain the program and

CYBER 860-to-Cray Examples

data (see page 3-1-7, example 3).

Page 3-1-10

Rev0 90/09/30

** From a Running Cray Job **

A batch job to be submitted from a running Cray job may reside either on the Cray or on one of the front-ends. From within the Cray job, ACCESS or FETCH the file to make it a local file, then SUBMIT it to the COS input queue. (See Appendix C for additional parameters for these Cray commands.)

* Examples *

1) The job is in a permanent dataset on the Cray:

```
JOB,....
ACCOUNT,....
...
ACCESS,DN=myjob,PDN=mypermjob.
SUBMIT,DN=myjob.
...
```

2) The job is in a file on the VAXcluster:

```
JOB,....
ACCOUNT,....
...
FETCH,DN=myjob,TEXT='myjob.job'.
-or-
FETCH,DN=myjob,MF=V3,TEXT='DT3"user pw"::UOn:[user]myjob.job'.
.^-- submitted from CYBER 860
or VAXcluster
SUBMIT,DN=myjob.
```

3) The job is in a file on the Mass Store (CDC CYBER 860):

JOB,.... ACCOUNT,.... MSACCES,US=user,MPW=mspass. MSFETCH,DN=myjob. SUBMIT,DN=myjob.

Page 3-1-11

*** Interactive Jobs ***

CRAY X-MP interactive access is via the Cray Station code on one of the front-ends.

** From the VAXcluster **

The CRAY X-MP is accessed via the VMS Cray Station, which may be entered by the CRAY or CINT command.

The CRAY command puts you into Cray context (indicated by the CRAY> prompt). Among other capabilities, you can examine and manipulate your jobs in the Cray queues.

The CINT command initiates an interactive session on the Cray. If not included in the CINT statement qualifiers, you will then be requested to supply:

| Cray Jobname: | < enter 1-7 alphanumeric characters as the |
|----------------|---|
| | jobname for this session (must be upper case |
| | to be able to fetch from the Mass Storage |
| | System; should be different for each session) |
| Cray Username: | < enter your User Initials |

The exclamation prompt (!) indicates that you have reached the Cray. Your first command must be your ACCOUNT statement. Any other commands will be ignored until a valid ACCOUNT statement is read.

!ACCOUNT, AC=1222233344, UPW=pw, US=userinit.

When you receive another ! prompt, your logon was successful. You may now use any of the commands in Appendix C. Every command MUST end with a terminator (.); if you forget, use the up-arrow to bring the command back and add the terminator.

To execute some Cray Station commands, use ²Z to interrupt CINT. At the Cint> prompt, enter the Station command. When it completes, you will be returned to your interactive session. Note that only a subset of the Cray Station commands may be executed in CINT.

To terminate the Cray interactive session, enter ². At the Cint[>] prompt, type QUIT to return to the VMS prompt and close the interactive session.

To leave the Cray Station, enter EXIT (or '2). This will bring you out of Cray context and back to the VMS prompt.

Page 3-1-12

Rev0 90/09/30

* VMS Cray Station Commands *

See Appendix D for the syntax of these commands. (CINT) indicates that the command may be executed interactively (via CINT) as well as from the Cray Station (CRAY).

\$ Create a temporary VMS subprocess, allowing you to enter DCL commands. To return to Cray context, type LOGOUT.

\$ (CINT only) Execute a single DCL command.

- + Display the next page of information in Cray context.
- Display the previous page of information in Cray context.
- @ (CINT) Execute an indirect station command file (containing station commands) in Cray context. (Synonym for PLAY.)

^C (CINT only) CTRL-C -- Same as ABORT.

- ^0 (CINT only) CTRL-0 -- Toggles interactive output on and off until the next Cray prompt.
- ^Z (CINT) CTRL-Z exit the current processing mode. In response to the Cray context prompt (CRAY>), it returns you to DCL; during a Cray interactive session, it returns you to command mode (Cint> prompt). While you are being prompted for command parameters, CTRL-Z cancels the command. You can also terminate the execution of an indirect station command file with CTRL-Z. In response to the Cint> prompt, you are returned to your interactive session.
- ABORT (CINT only) Interrupt the current interactive Cray job step and return to the "!" prompt after first displaying any COS output queued for the terminal.
- ATTACH (CINT only) Redirect Cray interactive terminal output to an alternate device (graphics).
- ATTENTION (CINT only) Interrupt the current interactive Cray job step and enters reprieve processing. If no reprieve processing, ATTENTION is the same as ABORT.
- BYE (CINT only) Terminate an interactive session. Depending on the command qualifiers, the COS interactive job may also be terminated.
- CLEAR Terminate any display command and clear the display portion of the screen.

COLLECT (CINT only) Store COS interactive output in a VMS file.

COMMENT Insert comments into an indirect station command file stream.

Page 3-1-13

DATASET Report the existence of a COS permanent dataset.

DELAY Suspend execution of an indirect station command file for a specified period of time.

DISCARD (CINT only) Discard all output from a Cray interactive session until the next COS prompt is issued.

DROP Terminate a Cray job and returns the associated output dataset. COS job execution enters reprieve processing after the next COS EXIT control statement.

EOF (CINT only) Send an end-of-file record to a connected COS interactive job. This command is normally required to terminate COS file input from the terminal.

EXIT (CINT only) Return you from Cray context or Cint command mode to DCL level.

HELP (CINT) Display information from the station help files or an index of all commands.

ISTATUS (CINT only) Return the status of your Cray interactive job, including the CPU time used and the last Cray logfile message.

JOB Display the status of a specific COS job.

- JSTAT Display the status of a specific job and its related tasks.
- KILL Terminate a Cray job immediately.

LOGFILE Provide access to the station logfile messages.

LOOP Restart execution of an indirect station command file at the beginning of the file. End looping with ^Z.

MESSAGE Send a message to the Cray job and station logfiles.

PAUSE Suspend the execution of an indirect station command file. Control is passed to the terminal, where you can terminate the command file by entering a command or resume it by entering a null line (<RET>).

PLAY (CINT only) Execute an indirect station command file. (Same as @.)

QUIT (CINT only) Terminate a Cray interactive session and the corresponding Cray interactive job. (Equivalent to BYE/ABORT.)

Rev0 90/09/30

- RECORD Start or stop the recording of terminal input to the specified file while in Cray context for later use with the PLAY or @ commands. Exiting Cray context automatically issues a RECORD/OFF.
- RELEASE Release a dataset in the output HOLDING queue due to VMS quota problems.

REMOVE Delete entries in the dataset staging queue.

- RERUN Immediately end the processing of a COS job and put job back into the input queue, unless the job has terminates or cannot be rerun.
- SET Define terminal working environment for the current session.
- SHOW Display information about the status of the station staging queue.
- SNAP Copy the current contents of the display region into the specified VMS file. If the command is issued from a terminal in line-by-line mode, the last display requested is recorded in the file.
- STATCLASS Display the current Cray job class structure.
- STATUS Display the Cray system status.
- STATUS (CINT only) Same as ISTATUS.
- STORAGE Initiate a COS disk status display providing the following information: device class or status; device name as it is known to COS; percentage of free space and permanent space on each device; number of recovered and unrecovered errors on each device; location of last error.
- SUBMIT Stage the specified VMS file to Cray to be put on the job input queue. The file must contain COS JCL (see CRAY HELP). The first record must be the JOB control statement. By default, the output from the COS job (known as a logfile) is sent to the directory from which the job was submitted.
- SUPPRESS Suppress the echoing of the next typed input line.

SWITCH Set or clear COS job sense switches.

Page 3-1-15

* Examples *

1) \$ cint Cray Jobname: ABCD001

> Cray Username: ABCD !ACCOUNT,AC=1222233344,UPW=mypw.

!<your Cray commands>
!^Z
Cint> <Station command>
!^Z
Cint> quit

2) \$ cint Cray Jobname: efgh002 Cray Username: efgh !ACCOUNT,AC=1222233344,UPW=mypw,US=efgh. <-- US=efgh needed</pre>

<same as example 1>

3) \$ cint /jn=struct /us=efgh

!ACCOUNT, AC=1222233344, UPW=mypw.

<same as example 1>

case to be able to fetch from the Mass Storage System) <-- user ABCD <-- US=abcd not needed since upper case was used in entry above <-- ctrl-Z to leave Cray</pre>

<-- any jobname (must be upper

- <-- returns to Cray when done
- <-- ctrl-Z to leave Cray
- <-- terminate Cray session

<-- US=efgh needed
 since lower case
 was used in entry
 above</pre>

<-- any jobname; user EFGH (since this is a VMS DCL command, it is converted to upper case) <-- US=efgh not needed since the VMS control statement is converted to

upper case

Page 3-1-16

** From the CDC CYBER 860 **

The CRAY X-MP is accessed via the NOS Interactive Cray Facility (ICF), which may be entered by the APPSW, ICF command from IAF. You enter ICF, log onto the Cray, do your thing (Cray or ICF commands), leave the Cray and ICF. You will then be at the NOS prompt.

Alternatively, you can specify ICF as the application when you log into NOS.

ICF commands have a prefix (normally a slash "/") and can be intermixed with Cray commands. To terminate the Cray session (and ICF), enter /BYE or /LOGOFF.

* ICF Prologue File *

A user prologue may be defined to be executed each time you start an ICF session. This is an 8/12-bit ASCII indirect access file named ICFPROF, with access granted to user SYSTEMX (PERMIT, ICFPROF, SYSTEMX.) We suggest you include "/PERIOD ON" so you don't have to type a period at the end of each Cray command.

* NOS ICF User Commands *

- /ABORT Send abort interrupt to the interactive Cray job (also user-break-2 key (normally %2).
- /ATTENTION Send attention interrupt to the interactive Cray job (also user-break-1 key (normally %1).
- /BYE Terminate this Cray interactive session. (Same as /LOGOFF)
- CONNECT Create a logical connection between this terminal and some other (slave) terminal.
- /DISCARD Discard output being sent from the Cray to this terminal.
- /ENDCONNECT Terminate a CONNECT.
- /ENDPLAY Terminate reading of a PLAY file.

/EOF Send an end-of-file to the Cray.

/HELP Display help information.

- /ICFSTATUS Display general information about the current status of ICF.
- /LOGOFF Terminate this Cray interactive session. (Same as /BYE)

/LOGON Initiate or reconnect to an existing Cray job.

| 90/09 | /30 | Rev0 |
|-------|-----|------|
|-------|-----|------|

Page 3-1-17

| /PERIOD | Set/reset | automatic | generation | o£ | а | terminator | on | COS |
|---------|-----------|-----------|------------|----|---|------------|----|-----|
| | commands. | | | | | | | |

- /PLAY Read data and commands from a NOS file in the user's catalog.
- /PREFIX Change the ICF command prefix letter.
- /QUIT Immediately terminate this Cray interactive session.
- /RESUME Resume the transmission of data to and from the Cray (negate the effect of SUSPEND).

/SUSPEND Suspend transmission of data to and from the Cray.

/STATUS Display Cray status.

/* An ICF comment line.

* Examples *

- - !account,ac=1222233344,upw=mypw. !<your Cray or ICF commands> !/bye

<a greeting>
!account,ac=1222233344,upw=mypw.
!<your Cray or ICF commands>
!/bye
T1210 - APPLICATION: iaf

<-- / is required; log onto DTRC Cray <-- US=abcd not needed <-- to leave Cray and ICF <-- log into ICF directly <-- / is required; log onto DTRC Cray <-- US=abcd not needed <-- to leave Cray and ICF <-- switch to another application such as IAF



***** Cray JCL Commands *****

The Cray Job Control Language (JCL) statements are grouped by function in this section. See Appendix C for a description of the syntax for each command. (DTRC) indicates a command or program added at DTRC. Some of the logic structure commands use JCL expressions, which are described later in this section.

*** Job Definition and Control ***

* Entire line is a comment.

ACCOUNT Validate a user's Job Order Number, user name and password.

ALTACN Validate an alternate account number for permanent datasets.

CALL Read control statements from another file.

CHARGES Report on job resources.

ECHO Control logfile messages.

EXIT On job abort, processing continues with the statement following the EXIT; if no abort, terminate job processing.

IOAREA Control access to a job's I/O area (containing the DSP and I/O buffers).

JOB First statement of a job -- gives job parameters.

JOBCOST (DTRC) Write a summary of job cost and system usage to \$LOG.

LIBRARY Specify search order for procedures during processing.

MEMORY Request new field length.

MODE Set/clear mode flags.

NORERUN Control a job's rerunability.

OPTION Specify user-defined options.

RERUN Control a job's rerunability.

RETURN Return from an alternate control statement file.

ROLLJOB Protect a job by writing it to disk.

SET Change value of a JCL symbolic variable.

SWITCH Turn pseudo sense switches on or off.

```
Page 3-2-2
```

Rev0 90/09/30

*** Dataset Definition and Control ***

ACCESS Make a permanent dataset local.

ASSIGN Create a dataset and assign dataset characteristics.

HOLD Dataset release occurs with implicit HOLD.

NOHOLD Cancel effect of HOLD.

RELEASE Relinquish access to a dataset from a job.

*** Permanent Dataset Management ***

ACCESS Make a permanent dataset local.

ADJUST Redefine size of a permanent dataset.

DELETE Remove a permanent dataset.

MODIFY Change a permanent dataset's characteristic information.

MSCHANG (DTRC) Change the attributes of a Mass Storage System file.

NEWCHRG (DTRC) Change permanent file account number.

PERMIT Grant/deny access to a permanent dataset.

SAVE Make a dataset permanent.

SCRUBDS Write over a dataset before release.

*** Permanent Dataset Staging ***

See Chapter 3 for staging to and from the Mass Storage System.

ACQUIRE Get a front-end dataset and make it permanent.

DISPOSE Stage dataset to the front-end; release a local dataset; change disposition characteristics.

FETCH Get a front-end dataset and make it local.

MSACCES (DTRC) Supply your Username and password to the Mass Storage System (MSS).

MSFETCH (DTRC) Fetch a file from the MSS.

MSPURGE (DTRC) Purge a file from the MSS.

MSSTORE (DTRC) Store a file on the MSS.

SUBMIT Send local dataset to COS input queue.

```
90/09/30 Rev0
```

*** Permanent Dataset Utilities ***

AUDIT Report on permanent datasets.

*** Local Dataset Utilities ***

BLOCK Convert an unblocked dataset to a blocked dataset.

- COPYD Copy blocked datasets.
- COPYF Copy blocked files.
- COPYNF Copy files from one blocked dataset to another.

COPYR Copy blocked records.

COPYU Copy unblocked datasets.

DS List local datasets.

NOTE Write text to a dataset.

QUERY Determine the current status and position of a local file.

REWIND Position a dataset at its beginning.

SKIPD Skip blocked datasets (position at EOD (after last EOF)).

SKIPF Skip blocked files from current position.

- SKIPR Skip blocked records from the current position.
- SKIPU Skip sectors on unblocked datasets.

UBBLOCK Convert a blocked dataset to an unblocked dataset.

WRITEDS Initialize a blocked dataset by writing a single file containing a specific number of records of a specific length.

*** Dumps and Other Aids ***

COMPARE Compare two datasets.

DEBUG Interpret a dump.

DUMPJOB Capture job information in dataset \$DUMP for display by DUMP.

DUMP Display job information previously captured by DUMPJOB.

FLODUMP Dump flowtrace table.

Page 3-2-3

FTREF Generate Fortran cross-reference.

ITEMIZE Report statistics about a library dataset.

PRINT Write value of JCL expression to the logfile.

SPY Generate a histogram of time usage within a program to locate inefficient code.

*** Logic Structure ***

ELSE IF-loop control.

ELSEIF IF-loop control.

END₁. IF-loop termination.

ENDLOOP LOOP termination.

EXITIF IF-loop control.

EXITLOOP LOOP control.

IF Begin a conditional block of code.

LOOP Start of an iterative control statement block.

*** Procedures ***

See Section 3-3 for additional information on the creation of procedures.

CALL Transfer control to a procedure.

"call by name" Execute a complex procedure in a library.

ENDPROC End of a procedure.

PROC Begin an in-line procedure definition block. This is followed by the procedure prototype statement which names the procedure and gives the formal parameter specifications.

RETURN Return control from a procedure to its CALLer.

*** Programming Languages ***

CFT Compile a Fortran source program.

CFT77 Alternate Fortran compiler (slower compile, faster execute).

PASCAL Compile a Pascal source program

Page 3-2-5

```
*** Program Libraries ***
```

See Section 3-4 for a discussion of program libraries (PL).

AUDPL Audit an UPDATE PL.

UPDATE Source and data maintenance.

*** Object Libraries ***

See Section 3-5 for a discussion of object libraries.

BUILD Generate and maintain library datasets.

SEGLDR Segment loader (see Section 3-6).

*** Miscellaneous ***

"call by name" Execute a program by its local file name.

SID Debug programs interactively or in batch.

SORT Sort/merge.

Page 3-2-6

Rev0 90/09/30

*** JCL Expressions ***

An expression is a string of operands and operators. It is evaluated from left to right, taking into account parentheses and operator hierarchy. Expressions allow the incrementing of counters, error code checking, and string comparison.

There are four types of operands:

. symbolic variables (see below)

. subexpressions (its value becomes an operand)

Expressions may be used in IF, ELSEIF, EXITIF, and EXITLOOP.

** Symbolic Variables **

There are 38 symbolic variables: 6 system constants, 7 variables set by COS, and 25 which can be set by the user.

* System Constants *

| Symbol | Range | Description |
|--------|---------|------------------------|
| FALSE | O | False |
| SID | literal | Mainframe ID (C1) |
| SYSID | literal | COS level ('COS n.nn') |
| TRUE | -1 | True |

SN and XM are also available.

* COS-set Variables *

| Symbol | Range | Description |
|----------|----------------|-----------------------------------|
| ABTCODE | 0-nnn | COS job abort code (ABnnn) |
| DATE | literal | mm/dd/yy |
| FL | 0-7777777 | current octal field length |
| FLM | 0-7777777 | JOB statement maximum octal FL |
| PDMST | 64-bits | status of most recent Permanent |
| | | Dataset Manager request |
| TIME | literal | hh:mm:ss |
| TIMELEFT | 64-bit integer | job time remaining (milliseconds) |

Page 3-2-7

| * User-set Variables * | | | | |
|--|---------------------------|--|--|--|
| Symbol | Range | Description | | |
| G0-G7 | 64-bits | 8 global pseudo-registers (can be used to pass data between procedures) | | |
| J0-J7 | 64-bits | 8 job (local) pseudo-registers (each procedure level has its own J registers) | | |
| JSR | 64-bits | Job Status Register containing the previous job step completion code | | |
| NOTEXT | 64-bits | text field not echoed (default: ON) | | |
| PDMFC SSW1-SSW6 | 64-bits 64-bits | most recent user-issued PDM request | | |
| | | ** Operators ** | | |
| Operators | may be | | | |
| | tic (+, -, n by 0 proc | *, /); Underflow and overflow are not detected; duces zero | | |
| | nal (.EQ., or O (FALSI | .NE., .LT., .GT., .LE., .GE.); returns -1 E) | | |
| . logical | (.OR., .AN | ND., .XOR., .NOT.); returns a 64-bit value | | |
| Operations are performed left to right, taking into account parentheses, with the hierarchy of operators: $(*, /)$, $(+, -)$, relational, .NOT., .AND., .OR., .XOR | | | | |
| | | ** Strings ** | | |
| | | of ASCII characters (040-176 octal) to be taker wo types of strings: | | |
| . literal | - de | elimited by apostrophes '' | | |

. parenthetical - delimited by parentheses -- (...)

Literal strings do not include the delimiters. An apostrophe within a literal string is represented by two apostrophes: '...''...'. A null string is indicated by two apostrophes: ''. A literal string is continued by placing an apostrophe and a continuation character at the end of the first line and an apostrophe at the start of the string on the next line:

> ... 'This Is A '^ 'Long String.' becomes

This Is A Long String.

Page 3-2-8

Rev0 90/09/30

÷.

Parenthetical strings do not include the delimiters. Spaces are removed; nested parentheses are not treated as separators; literal strings may appear in a parenthetical string. A parenthetical string is continued by placing a continuation character at the end of the first line and continuing the string on the next line:

> ... (This Is A ^ Long String.) becomes ThisIsALongString.

***** Procedures *****

A procedure is a group of control statements separate from the job control statement dataset (\$CS). Calling a procedure provides a simplified way to process that group of control statements. A procedure may be called by a job repeatedly or by another procedure.

There are two kinds of procedures in COS:

- . simple a sequence of control statements
- . complex a prototype statement (giving the name of the procedure and any parameters), the control statements, and optional data.

*** Simple Procedures ***

A simple procedure has no name or parameters and resides in a nonlibrary dataset. It is invoked by a CALL without the CNS parameter. Control is returned to the caller by a RETURN statement, the end of the first file in the dataset, or an EXIT (when not skipping because of an error condition). A simple procedure has no parameter substitution.

Any COS JCL statement, except PROC and ENDPROC, may be used in a simple procedure. One use might be to access all the datasets needed in several jobs without having to specify them in the individual jobs.

*** Complex Procedures ***

Complex procedures are named and may have parameters described in a prototype statement. Complex procedures are executed by

- . "call by name", which may include parameters for substitution in the procedure. The procedure is in \$PROC or a local dataset named in a LIBRARY statement.
- . CALL,DN=procfyl,CNS, followed by a line containing the procedure name and parameters for substitution. The procedure is the first file in a separate dataset; PROC and ENDPROC are not used.

Complex procedures may appear, delimited by PROC and ENDPROC, in the job control statement dataset (\$CS). When PROC is encountered, the procedure is written to \$PROC. Subsequent calls to the procedure may then be made using the procedure name (and any substitute parameters).

Rev0 90/09/30

A complex procedure has the general form:

PROC. <-- not for CALL prototype statement control statements ... &DA1A,dn1. data for first dataset ... &DATA,dnn. data for last dataset ENDPROC. <-- not for CALL

** Prototype Statement **

The prototype statement defines the name of the procedure and its formal parameters with their default value(s). It has the form:

name, p1, p2, ..., pn.

name - the name of the procedure (1-8 alphanumeric characters)

pi - a formal parameter specification

posi - positional

| keyi=dval:kval | keyi - formal keyword name dval - optional default value when keyi is omitted from the calling statement kval - optional default value when keyi is specified in the calling statement |
|----------------|--|
| | without a value |
| keyi= | - no defaults; the caller must supply a non-null value |
| keyi=: | - no defaults; allows keyi and keyi= |

** Temporary Datasets **

One or more temporary datasets may be included in a complex procedure following the control statement. Each starts with

&DATA,dn.

where dn is the required dataset name.

Page 3-3-3

** Parameter Substitution **

Formal parameters are used, preceded by an ampersand (&), within the body of the procedure. On execution, each is replaced by the value supplied or implied in the calling statement. ¶m is delimited by any character except A-Z, a-Z, 0-9, @, \$, or %. If the next character is one of these, the underline (_) is used at the delimiter and is removed at execution time.

If too few positional parameters are specified by the caller, null strings are used for the remaining parameters; if too many, the job aborts. Keyword parameters may appear in any order, however, all positional parameters must precede all keywords.

** Apostrophes and Parentheses **

Apostrophes in the calling statement denote literals and are not removed during substitutions; the outer set of parentheses are removed. If you are not sure how a parameter is used in the procedure, enclose it in parentheses.

The following shows parenthetical substitution:

| caller | after substitution |
|-----------------|--------------------|
| | |
| value | value |
| (valuel=value2) | valuel=value2 |
| valuel'.'value2 | valuel'.'value2 |
| valuel(.)value2 | valuel.value2 |

*** DTRC "System" Procedures ***

The following DTRC-written procedures have been added to COS as of the date of this page. See Appendix C for more information. For the current list, type "HELP @CRAY CONTENTS" on the VAX.

MSACCES Supply your Username and password to the Mass Storage System

MSAUDIT Sorted audit of Mass Storage System files.

MSCHANG Change the attributes of a Mass Storage System file.

MSFETCH Fetch a file from the Mass Storage System to the CRAY X-MP

MSPASSW Change your Mass Storage System access password.

MSPURGE Purge a Mass Storage System file from the CRAY X-MP

MSSTORE Store a CRAY X-MP file on the Mass Storage System

*** DTRC Procedure Library ***

One procedure library has been added to COS at DTRC:

PROCLIB, OWN=PUBLIC.

To use: ACCESS, PROCLIB, OWN=PUBLIC. LIBRARY, PROCLIB:*. procname,....

The following routines are the procedures in PROCLIB as of the date of this page. See Appendix C for more information. For the current list, type "HELP CORAY CONTENTS" on the VAX.

JOBCOST Write job cost summary to \$LOG

NEWCHRG Change the account number of permanent files

Page 3-3-5

*** *** Examples х× ** Simple Procedures 1) The first file of dataset GETLIBS contains: ACCESS, DN=MSPROC, OWN=PUBLIC. <-- the MSS procedures ACCESS, DN=DTLIB, OWN=PUBLIC. <-- the DTLIB subroutine library ACCESS, DN=SUBS. <-- your subroutine library This is executed by: CALL, DN=getlibs. ** ΧŻ Complex Procedures 2) As in example 1, but your subroutine library is to be identified by the caller: GETLIBS, SUBS. <-- prototype statement ACCESS, DN=MSPROC, OWN=PUBLIC. <-- the MSS procedures ACCESS, DN=DTLIB, OWN=PUBLIC. <-- the DTLIB subroutine library <-- your subroutine library ACCESS, DN=SUBS, PDN=&SUBS. When called by: CALL, DN=getlibs, CNS. getlibs, othersubs. the third ACCESS expands to ACCESS, DN=SUBS, PDN=othersubs. Note that the name of the procedure is unimportant, since it is the only procedure in the file. "getlibs, othersubs." could be replaced by "*, othersubs". When called by: CALL, DN=getlibs, CNS. getlibs, (hislib, OWN=him). the third ACCESS expands to ACCESS, DN=SUBS, PDN=hislib, OWN=him. When called by: CALL, DN=getlibs, CNS. getlibs, 'hislib, OWN=him'. the third ACCESS expands to ACCESS, DN=SUBS, PDN='hislib, OWN=him'. While this is legal (it says the permanent filename is "hislib,OWN=him"), it is probably an error and, if so, will abort the procedure.

3) Create a procedure library from procedures in the job stream.

ECHO, OFF. RELEASE, DN=\$PROC. <-- return existing \$PROC</pre> × <-- write first procedure to \$PROC</pre> PROC. prototype procedure body RETURN...procname EXIT. RETURN, ABORT...procname <-- end of first procedure ENDPROC. π PROC. <-- write next procedure to \$PROC</pre> prototype procedure body RETURN...procname EXIT. RETURN, ABORT...procname ENDPROC. <-- end of procedure * <-- more procedures • • • * ACCESS, DN=proclib, NA, UQ. <-- get original (existing) library</pre> SAVE, DN=\$PROC, PDN=proclib. <-- save new library DELETE, DN=proclib, NA. <-- delete original library <-- return new library RELEASE, DN=\$PROC. ACCESS, DN=proclib. <-- get new library with its own name <-- add it to the end of the library LIBRARY, DN=*:proclib. list -or-LIBRARY, DN=proclib:* <-- add it to the beginning of the library list ECHO, ON. . . . < use one of the procedures > . . .

Page 3-3-7

4) Create a procedure library from procedures in a separate file.

FETCH, DN=myprocs, TEXT='myprocs.pro'. <-- defaults to AC=ST
CALL, DN=myprocs.
SAVE, DN=\$PROC, PDN=proclib, PAM=R. </pre>

where VMS file MYPROCS.PRO contains:

* first procedure PROC. prototype procedure body ENDPROC. x next procedure PROC. prototype procedure body ENDPROC. * next procedure <-- more procedures ••• *

***** Program Libraries *****

Source programs and data may be in separate datasets or may be stored and maintained in program libraries. UPDATE creates and maintains these libraries while AUDPL (see Appendix C) audits them.

*** UPDATE ***

UPDATE is a program for creating and modifying a program library (PL). In addition, UPDATE will extract individual modules for input to a compiler or other program.

By default, 72 columns of information are retained. Fifteen additional characters are retained for each line: an 8-character identifier, a period (.), and a 6-digit sequence number, i.e., id.seq.

UPDATE supports two kinds of text modules or decks:

- a regular deck (beginning with a DECK directive)
- a common deck (beginning with a COMDECK directive) which may be included in decks with a CALL directive

Each type includes all lines following the deck directive until the next deck or modification directive.

History information is retained allowing the deletion, modification, or restoration or previous modifications.

See Appendix C for a description of the UPDATE control statement parameters.

*** UPDATE Directives ***

An UPDATE directive, which must be in upper case, has the following format:

m directive name [parameters]

where m is the master character (default: asterisk (*)). There are five categories of directives.

** DECK and COMDECK **

*DECK deck

First line of a new deck. <deck> is up to 8 characters, any ASCII character from 41 to 176 octal, except comma, period, blank, colon, equals.

*COMDECK cmdk (*CDK) First line of a new common deck.

(*DK)

Page 3-4-2

Rev0 90/09/30

** Compile Directives ** *CALL cmdk (*CA) Include the contents of a common deck. *CWEOF Write an EOF on the compile dataset if anything was written since the last EOF. *NOSEQ Do not write sequence numbers. *SEQ Write sequence numbers. *WEOF Write an EOF on the compile dataset. *WIDTH dw Change the data width (default: 72). *IF, *ELSEIF, *ELSE, and *ENDIF are also available. ** Modification Directives ** *BEFORE id.seq (*B) Insert before a line. (*CY) *COPY p,idl.seq1,id2.seq2 Copy a range of lines from deck or comdeck . (*D) *DELETE idl.seq1 <-- one line *DELETE idl.seq1, id2.seq2 <-- a range of lines *DELETE idl.seq1,.seq2 <-- same (short form) Delete a line or a range of lines. (*ID) *IDENT ident *IDENT ident,K=k1:k2:...,U=u1:u2:... Identify a set of modifications. You can require that other modification sets be known (K=) or unknown (U=). *INSERT id.seq (*I) Insert after a line. (*R) *RESTORE idl.seq1 <-- one line *RESTORE id1.seq1, id2.seq2 <-- a range of lines <-- same (short form) *RESTORE idl.seq1,.seq2 Restore a line or a range of lines.

** х× Run Options */comment A comment line. *COM_ILE p1, p2, ..., pj.pk, ..., pn (*C) Write one or more decks, including a range (pj.pk), to the compile and/or source datasets. Use UPDATE,K to force the output order. *COPY p,idl.seq1,id2.seq2,dn (*CY) *COPY p,idl.seq1,id2.seq2,dn,SEQ Copy a range of lines from deck or comdeck to dataset <dn>. SEQ will include sequence numbers. *LIST Resume listing input lines. UPDATE, L=0 overrides *LIST. *MASTER m Define a new master character for subsequent directives. (default: *) *NOLIST Stop listing input lines. *NOLIST overrides UPDATE, IN. *READ dn (*RD) Read input from another dataset. *REWIND dn Rewind a dataset. *SKIPF dn *SKIPF dn,n

Page 3-4-3

Skip file(s) in a local dataset.

*DECLARE and *DEFINE are also available.

90/09/30

Rev0

** * * Input Edit Directives *EDIT p1,p2,...,pn (*ED) Remove deleted and yanked lines from specific decks. These lines cannot be retrieved. This is useful for cleaning up a PL. *MOVEDK dk1:dk2 *MOVEDK dk1:. Position deck of common deck <dkl> immediately after deck or common deck <dk2> or at the beginning of the PL <.>. *PURGE id1, id2,..., idj.idk,..., idn.. Remove the effect of a modification set (idi), a range of datasets (idj.idk), or a set and all following (idn..). *PURGEDK dk Permanently remove a deck or common deck. *UNYANK id1, id2, ..., idj.idk, ..., idn.. Reactivate a deck, comdeck, or modification set previously yanked. *YANK idl, id2,..., idj.idk,..., idn.. Temporarily delete a deck, comdeck, or modification set previously yanked.

*SKIP and *ENDSKIP are also available.

90/09/30 Rev0 Page 3-4-5 *** *** Examples 1) Create a PL: JOB, JN=makep11. ACCOUNT,.... <-- omit if using VAXcluster CSUBMIT</pre> UPDATE, P=0, C=0. <-- no \$PL or \$CPL SAVE, DN=\$NPL, PDN=uyp1. /EOF *DK DECK1 s for deck DECK1> *DK DECK2 s for deck DECK2> *DK DECK3 s for deck DECK3> 2) Extract, compile and execute deck DECK2 from PL MYPL: JOB, JN=getp12. ACCOUNT,.... <-- omit if using VAXcluster CSUBMIT</pre> ACCESS, DN=\$PL, PDN=myp1. UPDATE. CFT77, I=\$CPL. SEGLDR, CMD='MAP, PART', GO. /EOF *C DECK2 3) Create a PL using a common deck, compile and execute: JOB, JN=makep13. ACCOUNT,.... <-- omit if using VAXcluster CSUBMIT</pre> UPDATE, P≈O. <-- no \$PL (required to create)</pre> SAVE, DN=\$NPL, PDN=myp1. CFT, I=\$CPL. SEGLDR, CMD='MAP, PART', GO. /EOF *CDK COM3 common / mycom / a, b, c real a, b, c *DK PROG3 program prog3 *CALL COM3 call sub print *, 'a,b,c=', a, b, c end *DECK SUB subroutine sub *CA COM3 a = 1.F = 2. c = 3.return end

```
Page 3-4-6
```

4) Update old source library to new, compile all decks and execute: JOB, JN=job4. <-- omit if using VAXcluster CSUBMIT</pre> ACCOUNT.... ACCESS, DN=\$PL, PDN=mylib. UPDATE, F, N. SAVE, DN=\$NPL, PDN=mylib. CFT, I=\$CPL. SEGLDR, GO. /EOF *IDENT DS0620 <-- correction must be unique (initials,date)</pre> *INSERT ALONE.57 <--- correct deck ALONE by insert after line 57 <FORTRAN statements> *DELETE FOUR.12.13 <-- correct deck FOUR replacing lines 12-13</pre> <new lines to replace deletions - optional> /EOF <data lines, if any> /EOF 5) Select routines from source subroutine library on MSS and compile with own program: JOB, JN=job5. ACCOUNT,.... <-- omit if using VAXcluster CSUBMIT ACCESS, DN=MSPROC, OWN=PUBLIC. LIBRARY, DN=MSPROC:*. MSACCES, UN=un, MPW=mymsspw. CFT77, ON=MSX. <-- compile own programs with listing</pre> MSFETCH, DN=LIBR, MDN=DTLIBPC, UN=NSYS. UPDATE, P=LIBR, Q, L=0. CFT77, I=\$CPL. SEGLDR, GO. <-- load and execute</pre> /EOF <own FORTRAN decks> /EOF *C rtnl,rtn6.rtn8 <-- select decks RTN1, 6, 7, 8 from library /EOF

***** Object Libraries *****

BUILD is a utility for creating and maintaining libraries of absolute and relocatable object modules. These libraries can then be used by the loader to locate the program to execute or the subprograms to be loaded with your program.

The BUILD control statement is described in Appendix C.

*** BUILD Directives ***

A directive consists of a keyword and, perhaps, a comma-separated list of dataset or module names. The keyword is separated from its list by a blank. Directives cannot be continue. Multiple directives, separated by a semicolon or period, may appear in one line.

- FROM dn1,dn2,...,dnn Single dataset for COPY, OMIT, LIST, or a list of datasets (copy dn1 thru dnn-1 to \$NBL, dnn is the same as if specified alone. If no COPY, OMIT, dnn is also copied. dni can be a library or sequential dataset (like \$BLD).
- OMIT fn1,fn2,...,fnn
 List of modules to be excluded. Each fni may be a single name
 or a group name, i.e., with wildcards (any 0 or more
 characters) or * (any single character).

COPY fnl,fn2,...,fnn List of modules to be copied. Each fni may be a single or group name, or a rename (ELM=OAK copies ELM and renames it OAK), or an inclusive range such as (first,last) or (first,) or (,last) or (,).

LIST

Immediately list characteristics of modules in input dataset.

*** DTRC Object Libraries ***

Two object libraries have been added to COS at DTRC:

DTLIB,OWN=PUBLIC - Subprograms written or maintained by the Computer Center To use: ACCESS,DN=DTLIB,OWN=PUBLIC. SEGLDR directive: LIB=DTLIB

UTILITY,OWN=PUBLIC - Programs written or maintained by the Computer Center To use: ACCESS,DN=UTILITY,OWN=PUBLIC. LIBRARY,UTILITY:*. program_name,... Page 3-5-2

LIST

*** *** Examples 1) Create a library of subprograms: JOB, JN=JOB1. ACCOUNT,.... CFT. BUILD, I=0, OBL=0. SAVE, DN=\$NBL, PDN=MYSUBLIB. /EOF <Fortran source subprograms> /EOF 2) Create a library of all subprograms from an UPDATE library: JOB, JN=JOB2. ACCOUNT,.... ACCESS, DN=\$PL, PDN=MYPL. UPDATE, F. CFT, I=\$CPL. BUILD, I=0, OBL=0. SAVE, DN=\$NBL, PDN=MYSUBLIB. /EOF 3) Add a subprogram to an existing library and have the output list in alphabetical order. JOB, JN=JOB3. ACCOUNT,.... ACCESS, DN=\$OBL, PDN=MYSUBLIB. CFT. BUILD, I=0, SORT. SAVE, DN=\$NBL, PDN=MYSUBLIB. /EOF <Fortran source subprograms> /EOF 4) Delete subprogram BADSUB from an existing library and list the contents of both old and new libraries. JOB, JN=JOB4. ACCOUNT.... ACCESS, DN=\$OBL, PDN=MYSUBLIB. BUILD, B=0. SAVE, DN=\$NBL, PDN=MYSUBLIB. /EOF OMIT BADSUB



5) List the contents of an existing library.

JOB, JN= JOB5. ACCOUNT,.... ACCESS, DN=SUBLIB, PDN=MYSUBLIB. BUILD, OBL=0, NBL=0, B=0. /EOF FROM SUBLIB; LIST.

Page 3-6-1

***** Loader *****

The loader is responsible for loading all programs, resolving any external references, and optionally initiating execution. Loading can produce either a single absolute module, or a (segmented) absolute program in which different parts of a program reside in memory only when needed.

*** SEGLDR ***

The primary loader is SEGLDR. It is controlled by directives which may appear as the next file in the input stream, in a separate file, or in the loader control statement.

** Control Statement **

See Appendix C for a fuller description of the SEGLDR control statement.

SEGLDR,I=dirfile,L=listfile,DN=binfil,LIB=library,ABS=absolute, CMD='directives',GO.

"SEGLDR." implies SEGLDR, I=\$IN, L=\$OUT, DN=\$BLD, ABS=\$ABD.

** Message Levels **

SEGLDR issues messages at the following levels:

- ERROR immediately terminates SEGLDR with no executable output
- WARNING no executable output but processing continues
- CAUTION executable output but a possible error was found
- NOTE SEGLDR has been misused or used ineffectively; executable output is still valid
- COMMENT does not affect execution

Page 3-6-2

Rev0 90/09/30

** Directives **

ł

Most SEGLDR directives have the format: keyword=value. Comments (anything following an asterisk (*)) may appear anywhere in the directives, including at the end of a directive line. Multiple comments on a line are separated by a semicolon (;). Elements of a list are comma-separated. Directives may be continued by splitting the line after a parameter (the comma is the last non-blank character in the line).

Naming files: ABS, BIN, DEFLIB, LIB, NODEFLIB.

Listing control: COMMENT, ECHO, MAP, TITLE, TRIAL.

Naming modules and common blocks: COMMONS, DUPORDER, DYNAMIC, FORCE, MODULES.

Error message control: DUPENTRY, DUPLOAD, MLEVEL, REDEF, USX.

Entry point and execution control: EQUIV, SET, XFER.

Global heap memory management: HEAP, LOWHEAP, STACK.

Memory allocation and presetting: ALIGN, ORG, PRESET.

Symbolic debugging: DRD, SYMBOLS.

Miscellaneous COS-dependent directives: ABORT, ABSNAME, BCINC, GRANT, NOECHO, NORED, PADINC, SECURE.

Miscellaneous global directives: CASE, CPUCHECK.

Additional information, including directives not discussed here, may be found in SR-0066 Segment Loader Reference Manual.

* comment A comment.

Examples: TITLE=GLOBAL DIRECTIVES *-----* Global directives *-----BIN=ABC TITLE=TREE DIRECTIVES *-----* Tree directives *_____ TREE ROOT(A,B) ENDTREE TITLE=SEGMENTS *-----SEGMENT=ROOT * ROOT directives

90/09/30 Rev0 Page 3-6-3 ABORT=ON | OFF Control SEGLDR error termination. Values: ON - abort if errors OFF - terminate normally even if errors Default: ABORT=ON ABS=dn The dataset to contain the absolute module. Default: \$ABD Examples: ABS=myprog ALIGN=IGNORE | MODULES | NORMAL Control the starting locations of modules and commmon blocks. Values: IGNORE - start each module's local or common block at the word following the previous one (ignore align bit) MODULES - start each module's local block and common block (if the align bit is set) at an instruction buffer boundary (32 words) NORMAL - start each module's local or common block with the align bit set at an instruction buffer boundary (32 words) Default: ALIGN=NORMAL BIN=dn1,dn2,... Datasets containing the relocatable modules to be loaded. Default: BIN=\$BLD Examples: BIN=myfile, yourfile, theirfyl BIN=oldfile CASE=UPPER | MIXED Control character conversion in directives. UPPER - convert to upper case Values: MIXED - do not convert Default: CASE=UPPER

COMMONS=blk1:siz1,blk2:siz2,... Specify the order to load common blocks. blki - name of a common block Values: sizi - n - decimal size 0 - first occurrence of this block sets the size (default: 0) Examples: COMMONS=myb1k:100000,data1 ^-- MYBLK is 100,000 words (no matter how it is defined); DATA1 has its first encountered length DEFLIB=deflib1, deflib2,..., deflibn Add libraries to SEGLDR's list of default libraries. If a specified library is already in the Remarks: default library list, it is moved to the end of the list. Libraries in DEFLIB are searched after the default system libraries; those in LIB are searched before. Examples: DEFLIB=mylib ^-- add MYLIB to the list = = = = = NODEFLIB; DEFLIB=mylib ^-- the default library list consists of just one library DRD Load for debugging. Symbol tables are written to \$DEBUG (or SYMBOLS=dn). Default: Normal load DUPENTRY=ERROR | WARNING | CAUTION | NOTE | COMMENT | IGNORE Specify the message level for a duplicate entry point. Default: DUPENTRY=CAUTION DYNAMIC=comblk DYNAMIC=// Name a common block to be located after the largest segment or the heap (if required). You control its size. It is always available to the program and cannot be preloaded with data. Values: a COMMON block name or // (blank common)

```
90/09/30
                 Rev0
                                                           Page 3-6-5
             Default: no dynamic common blocks
             Examples: DYNAMIC=ARRAYS
                                   ^-- common block /ARRAYS/ is dynamic
ECHO=ON | OFF
             Resume or suppress listing of input directives.
             Default:
                        ECHO=OFF
EQUIV=epname(syn1, syn2,...)
             Substitute a call to one entry point for a call to
             another.
             Values:
                        epname - the entry point to be used in the
                                 substitution
                              - an entry point to be replaced by
                        syni
                                 epname
             Examples:
                          . . .
                        CALL A
                          . . .
                        CALL B
                          ...
                        EQUIV=C(A,B)
                                    ^{--} replaces the calls to A and B
                                       by calls to C
FORCE=ON | OFF
             Control the forced loading of modules whose entry points
             are never called.
             Default: FORCE=OFF
HEAP=init+inc
             (Tasking) Allocate memory for dynamic management.
                        init - initial decimal number of words
             Values:
                        inc - size, in decimal words, of increment
                               when the heap overflows
                                0 - the heap size is fixed
                                (ignored if DYNAMIC directive is
                               specified)
             Examples: HEAP=10000+2000
```

LIB=1ib1,1ib2,... Libraries to be searched for routines not included in BIN= files or default libraries. Remarks: Libraries in DEFLIB are searched after the default system libraries; those in LIB are searched before. Examples: ACCESS, DTLIB, OWN=NSYS. <-- DTRC subroutine library <-- your subroutine ACCESS, sublib. library SEGLDR,CMD='LIB=sublib,DTLIB',... MAP=NONE | STAT | ALPHA | ADDRESS | PART | EPXRF | CBXRF | FULL Control the map listing. Values: NONE - no map

| NONE | | no map |
|---------|---|--------------------------------------|
| STAT | - | list load statistics: date/time, |
| | | longest branch length, last segment, |
| | | transfer entry point, stack and heap |
| | | information |
| ALPHA | - | STAT + block map for each segment |
| | | (modules in alphabetical order) |
| ADDRESS | - | ALPHA but modules in address order |
| PART | - | ALPHA + ADDRESS |
| EPXRF | - | STAT + entry point cross reference |
| CBXRF | - | STAT + common block cross reference |
| FULL | - | PART + EPXRF + CBXRF |
| | | |

Default: MAP=NONE

Examples: MAP=STAT MAP=EPXRF,CBXRF

MLEVEL=ERROR ¦ WARNING ¦ CAUTION ¦ NOTE ¦ COMMENT Print messages down to and including the level specified (has no effect if L=0). Default: MLEVEL=CAUTION

> Examples: MLEVEL=NOTE ^-- print error, warning, caution, and note messages

Page 3-6-7

MODULES=mod1:ds1,mod2:ds2,... The modules to be included and, optionally, the dataset containing a specific module. Values: modi - name of module to be loaded dsi - optional dataset containing the module Examples: MODULES=subl:sublib,sub2,sub3:yourlib MODULES=sub4, sub5 ^-- get SUB1 from SUBLIB; SUB3 from YOURLIB; SUB2, SUB4, SUB5 from the first dataset containing them NODEFLIB Do not search the default libraries. Search only BIN and LIB datasets. NOTE: Segmented loads must specify the file containing routine \$SEGRES. Examples: NODEFLIB; LIB=sublib, DTLIB, \$SCILIB ORDER=MODULES, COMMONS | COMMONS, MODULES | XMP.EMA Load modules before or after commons. XMP.EMA - most efficient allocation on X-MP Values: having more than 4 million words Defaults: ORDER=MODULES, COMMONS (<=4 million words) ORDER=XMP.EMA (> 4 million words) PRESET=ONES | ZEROS | INDEF | -INDEF | value Preset uninitialized data areas. Values: ONES - set to -1 ZEROS - set to 0 INDEF - set to octal 06050540000000000000000 -INDEF - set to octal 1605054000000000000000 value - 16-bit value placed in each parcel (0 < value < 177777 octal) Default: PRESET=ZEROS

Page 3-6-8

SET=epname:value Set the value of an entry point. epname - the entry point name Values: value - the value it is to have (overrides the value found in the relocatables) SET=\$RBUFLN:256 Examples: SET=\$WBUFLN:256 ^-- change the read or write buffer length to 256 characters (to change the read/write buffer area, use COMMONS=\$RFDCOM:265 or COMMONS=\$WFDCOM:265 (must be 9 more than the buffer length) STACK=init+inc (Tasking) Allocate part of heap memory to a stack for reentrant programs. Values: init - initial size, in decimal words, of a stack <128 - default used inc - size, in decimal words, of increments when the stack overflows 0 - stack overflow is not allowed Default: Static variables unless Tasking or ALLOC=STACK. Examples: STACK=5000+2000 SYMBOLS=ON | OFF | dn Specify program symbol table handling. ON - write symbol table to \$DEBUG Values: OFF - ignore symbol table dn - write symbol table to dn (dn may not be ON or OFF) Default: SYMBOLS=ON

Page 3-6-9

TITLE=titleTITLEDefine the second line of each page header. A page eject
is forced.Value:title - a string of 0-74 characters
(ends with end-of-line or semicolon)
omitted - clear the second header lineExamples:TITLE=This is a user title, really!TRIALDo not generate an executable module. Lets you get the
load map, determine optimal memory usage for data, or
get the total memory required.

Examples: TRIAL

USX=WARNING | CAUTION | IGNORE Specify how to treat unsatisfied externals.

> Values: WARNING - issue a warning message; do NOT write executable output CAUTION - issue a caution message; write executable output IGNORE - issue no message; write executable output

Default: USX=CAUTION

- XER=entry Specify the entry point at which the program is to start execution.
 - Values: entry the starting entry point name
 - Remarks: Use this to specify the name of the main program to be executed if it is in a library.
 - Default: The first primary entry point encountered -if none, "main" is used.

Page 3-6-10

Rev0 90/09/30

*** Segmentation ***

To make a large program fit into memory, it may be structured in segments, so that only a portion of the program resides in memory. By using the tree structure directives of SEGLDR, different arrangements of a program can be tried, without changing the program, until the best is achieved. ** Segmentation Directives ** Tree definition: TREE, tree definition, ENDTREE. Segment description: SEGMENT, BIN, COMMENT, COMMONS, DUP, ECHO, MODULES, SAVE, TITLE, ENDSEG. Global: COPY, SAVE, SLT. BIN=dn1, dn2,... Datasets containing the relocatable modules to be loaded. Only the first file of each dataset is processed. Default: BIN=\$BLD Examples: SEGMENT=birch BIN=myfile, yourfile, theirfyl BIN=oldfile ENDSEG ^-- all modules in datasets MYFILE, YOURFILE, THEIRFYL, and OLDFILE are loaded into segment BIRCH COMMONS=blkl:sizl,blk2:siz2,... Specify the order to load common blocks. Values: blki - name of a common block sizi - n - decimal size 0 - first occurrence of this block sets the size (default: 0) Examples: COMMONS=myb1k:100000,data1 ^-- MYBLK is allocated 100,000 words (no matter how it is defined): DATA1 has its first encountered length

Page 3-6-11

COPY Force the program to execute from a scratch file. This may speed program execution, especially of programs with segments which are loaded many times, because a faster form of I/O is used. SAVE=ON also forces the use of a scratch file.

Default: a scratch file is not used

DUP=mod(seg1,seg2,...)

Specify that a module is to be loaded into several segments. DUP must appear before the definitions of the segments into which the module is to be placed.

An alternate way is to list the module in the MODULES or COMMONS directive of each segment.

| Examples: | DUP=sub3(seg1,seg2) | root | | | |
|-----------|---------------------|--|------|--|--|
| | SEGMENT=seg1 | ······································ | | | |
| | MODULES=sub1 | | 1 | | |
| | COMMONS = com1 | seg1 | seg2 | | |
| | ENDSEG | | | | |
| | SEGMENT=seg2 | com1 | coml | | |
| | MODULES≖sub2 | subl | sub2 | | |
| | COMMONS=com1 | SUB3 | SUB3 | | |
| | ENDSEG | | | | |

ENDSEG End the definition of a segment of a tree structure.

Examples: see SEGMENT

ENDTREE End the definition of a tree structure.

Examples: see TREE

MODULES=mod1,mod2,...
(segment) List the modules to be put into the segment.
Values: modi - module name and optional dataset from
which it is to be loaded (mod:ds)
Examples: MODULES=m:binm,n,o
^-- load module M from dataset BINM
and modules N and O from the
first dataset which contains
them

SAVE=ON | OFF (Global) Specify whether all segments are to be saved (written to disk) before being overlaid. SAVE in a segment overrides the global SAVE. OFF - do not save each segment Values: ON - save each segment Default: SAVE=OFF Examples: SAVE=ON one TREE one(two,three) sub1 ENDTREE $\overline{}$ SEGMENT=one MODULES=sub1 two three SEGMENT=two MODULES=sub2, sub3 sub2 sub4 SEGMENT=three sub3 SAVE=OFF MODULES=sub4 ENDSEG SEGMENT=segname Begin the description of the contents of one segment of a tree. Examples: SEGMENT=oak MODULES=k,1,m COMMONS=//,oakcom ENDSEG TREE End the global directives and start the definition of a tree structure. Examples: TREE tree structure

ENDTREE

Page 3-6-13

f

h

g

j

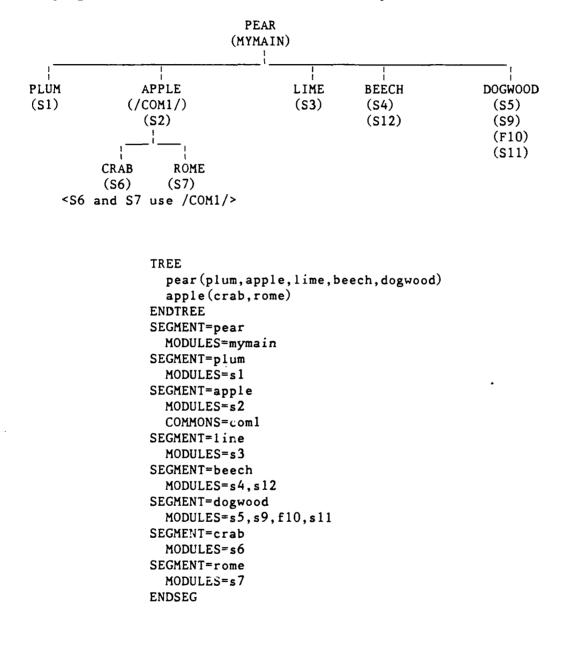
tree segment structure Define the tree structure, that is, the segments in each branch of the tree. The order of these definitions is unimportant. Syntax: segname(seg1,seg2,...) Examples: TREE а a(b,c)b(d,e) Ъ С c(f,g,h) f(i,j) ENDTREE

d

е

** Sample Tree Diagram **

A block data subprogram defines common /COM1/ which is to be loaded with program S2. /COM1/ is also referred to by S6 and S7.



Page 3-6-15

** Segmentation Cautions **

1. To develop a segmented job, several runs may be required, so relocatable object code should be SAVEd. Lommon blocks and some system routines may need to be included in lower segments to operate properly.

2. The load map should be checked carefully for any duplicate common block entries. The same common block may appear in more than one segment, each being considered a different common block. References are to the common block in the segment, if none, then to the one on the same branch. If a given common block is to appear only once in a program (the normal case), then it should be placed in the segment nearest to the root segment which can be referenced by all segments which use it.

```
Page 3-6-16
```

```
Rev0 90/09/30
```

*** Compile, Load and Save an Absolute Program *** ** ** Simple Load JOB, JN=jobname,.... ACCOUNT,.... CFT. SEGLDR, CMD= 'ABS=myprog'. SAVE, DN=myprog, PAM=R. <-- read only /EOF PROGRAM MYPROG (... . . . /EOF $\star\star$ ホホ Segmented LOAD JOB, JN=jobname,.... ACCOUNT,.... CFT. SAVE, DN=\$BLD, PDN=myprogob. <-- save relocatable modules for</pre> possible re-segmentation SEGLDR. <-- read only SAVE, DN=myprog, PAM=R. /EOF (CFT source program) /EOF ABS=myprog TREE . . . ENDTREE SEGMENT=... • • • ENDSEG SEGMENT=... . . . ENDSEG . . . /EOF

Page 4-1-1

***** The Mass Storage System *****

The Mass Storage System (MSS) is a large capacity on-line mass storage device. It is a cost effective extension to the Cray, CDC and VAXcluster disk systems and conventional magnetic tape storage. Specifically, the MSS, which is part of the CDC CYBER 860 (NOS), offers:

- . More than 20 times the on-line storage of the VAXcluster system; more than 40 times the on-line storage of the CRAY X-MP.
- . On-line access to files which previously had to be stored on magnetic tape because of size restrictions and/or infrequent use.
- . Reduced storage charges for these on-line files.

*** MSS Security ***

To provide adequate security for MSS users, you must submit your MSS (CYBER 860) password in any non-CDC (NOS) job or interactive session which will manipulate MSS files. To protect your MSS files, you must change this password at least every 90 days using the PASSWOR command on the CDC CYBER 860 (NOS), the HFT PASSWORD command on the VAXcluster, the MSPASSW command on the CRAY X-MP or the CDC 860 (NOS/VE).

*** MSS File Purge ***

MSS files may be purged by the Computer Center if the job order number is invalid or has been cancelled.

To recover purged files, call User Services, Code 3511, (301) 227-1907. A fee will be charged for this service. After the files have been restored, you must change to your valid job order number:

on 860 (NOS): CHANGE,mfn/CP or BEGIN,NEWCHRG on 860 (NOS/VE): MSCHANG mfn CP=1 on Cray (COS): see page 4-1-4 (MSCHANG) on VAXcluster: see page 4-1-8 (MSSNEWCHRG) Page 4-1-2

*** MSS Backup for Critical Files ***

In addition to normal file backup, critical direct files may be backed up and stored off-station. These files are available in the event of a catastrophe (such as fire) at the Carderock Computer Center.

For a file to be designated as "critical", it must have the attribute Backup Requirement (BR) set to critical (CR). This is done by specifying "BR=CR" if the file is critical, or "BR=Y" if it is not, when the file is made permanent. The default is BR=Y meaning on-station backup. For example (on NOS):

| DEFINE, 1fn=mfn/BR=CR. | < store a critical file | |
|------------------------|--|--|
| CHANGE,mfn/BR=CR. | < make a file critical | |
| CHANGE,mfn/BR=Y. | <pre>< make a file non-critical</pre> | |

Files designated for this off-station backup service will be charged a higher rate.

Page 4-1-3

*** *** Using the MSS from the Cray A description of the syntax of these commands may be found in Appendix C. ACQUIRE Transfer a file from the MSS as a local dataset and make it permanent on the Cray. Examples: ACQUIRE, DN=SOURCE, SDN=MYFILE, PDN=MYFILE, MF=N1, ^ TEXT='USER, user, pw. ATTACH, MYFILE. CTASK.'. ^-- transfer your direct MSS file MYFILE as local dataset SOURCE and make it a permanent dataset named MYFILE ACQUIRE, DN=DATA46, PDN=DATA46, MF=N1, ^ TEXT='USER, user, pw. '^ 'ATTACH, DATA46/UN=ABCD, PW=filepw.CTASK.'. ^-- transfer user ABCD's MSS file DATA46 (assuming you have permission to read the file) as local dataset DATA and make it a permanent dataset named DATA46 DISPOSE Transfer a Cray local dataset to the MSS. Examples: DISPOSE, DN=FT13, MF=N1, SDN=MYOUT13, DC=ST, ^ TEXT='USER,user,pw.'^ 'PURGE, MYOUT13/NA.'^ 'DEFINE, MYOUT13.'^ 'CTASK.'. ^-- local dataset FT13 is transferred to the MSS where it will be known as MYOUT13 FETCH Transfer a file from the MSS as a local dataset. It is released at the end of the job. Examples: FETCH, DN=SOURCE, SDN=MYFILE, MF=N1, ^ TEXT='USER, user, pw. ATTACH, MYFILE.CTASK.'. ^-- transfer your MSS file MYFILE as local dataset SOURCE FETCH, DN=ABDATA, MF=N1, TEXT='USER, user, pw. '^ 'ATTACH, ABDATA/UN=ABCD, PW=filepw.CTASK.'. ^-- transfer user ABCD's MSS file ABDATA as local dataset ABDATA FETCH, DN=SOURCE, SDN=MYFILE, MF=N1, ^ TEXT='USER, user, pw. GET, MYINDF. CTASK.'. ^-- transfer your CYBER 860 indirect file MYFILE as local dataset SOURCE

The following procedures provide access to the Mass Storage System. They have been made part of COS at DTRC.

MSACCES Supply your Username and password to the MSS. MSACCES is required before you can use the MSx commands.

Example: MSACCES, US=myid, MPW=mymsspw.

MSAUDIT Sorted audit of Mass Storage files.

Examples: MSAUDIT. <-- short audit of your MSS files

MSAUDIT,LO=X,SHOWPW=1. ^-- full audit showing each file's password, if any

MSAUDIT,L=audout,LO=X,UN=otheruser. ^-- full audit of another user's files you are allowed to see with output in local dataset AUDOUT

MSCHANG Change Mass Storage System file attributes.

Examples: MSCHANG, MDN=myfile, CT=PUBLIC. ^-- make your MSS file MYFILE public

MSCHANG, MDN=oldname, NMDN=newname.

MSCHANG, MDN=myfile, CP=1. ^-- change the account number of file MYFILE to your current MSS charge number

MSFETCH Fetch a direct file from the MSS.

Examples: MSFETCH, DN=infyl, MDN=mydata. ^-- your file in transparent mode

> MSFETCH, DN=prog, MDN=othrpgm, UN=ABCD, PW=pgmpw. ^-- another user's file

MSPASSW Change Mass Storage System access password.

Example: MSPASSW, OLD=oldpw, NEW=newpw.

Page 4-1-5

MSPURGE Purge an MSS file.

Example: MSPURGE, DN=myfyle.

MSSTORE Store a file on the MSS as a direct file.

Examples: MSSTORE, DN=out1, MDN=outfyl1, NA=1. ^-- overwrite if file already exists

> MSSTORE, DN=fy12, MDN=file2, DF=CB. ^-- file is stored in CDC Display Code

Rev0 90/09/30

*** using the MSS from the VAXcluster *** A description of the syntax of these commands may be found by typing "HELP <command>" on the VAXcluster. HYPERchannel (direct) File Transfer. HFT Examples: HFT ACCESS /U=ABCD /A=1222233344 /P=MSS_password ^-- gain access to the MSS HF1 CHANGE "MYFILE/AC=newac,CT=PU" ^-- change account number of MSS file MYFILE and make it public HFT DEFAULT ^-- display your current ACCESt values HFT DELETE MYFILE ^-- delete MSS file MYFILE HFT DIRECTORY ^-- audit your MSS file names HFT DIRECTORY "LO=F" ^-- full audit of your MSS files HFT FETCH MYPROG MYPROG.FOR ^-- fetch your MSS file MYPROG and make it permanent file MYPROG.FOR HFT PASSWORD old password new password new password repeated ^-- change your MSS password HFT PERMIT "MYFILE/UN=xxxx, M=R" ^-- give read access to user xxxx HFT STORE MYPROG.FOR "MYPROG/CT=S" HFT STORE MYPROC.FOR "MYPROG/CT=S" /DELETE ^-- store your file MYPROG.FOR on the MSS as MYPROG (/DELETE will delete your VAXcluster permanent file)

MSSAUDIT Audit your MSS files in a variety of formats. Examples: MSSAUDIT S <-- get a sorted short audit of your MSS files at the terminal MSSAUDIT F MSSAUDIT.LIS ^-- put a sorted full audit of your MSS files into file MSSAUDIT.LIS MSSAUDIT O UN=xxxx (0 = zero)^-- display a sorted list of the MSS files owned by user xxxx (assuming you have permission to see them) MSSBACKUP Store several files in a single file on the MSS, retaining each file's characteristics. Fetch individual files from the MSS file previously stored by MSSBACKUP. Examples: MSSBACKUP STORE *.* VMS0322 ^-- store all your files in a BACKUP file on the MSS (0322 is the date)MSSBACKUP LIST VMS0322 KEEP ^-- list the contents of the above BACKUP file on MSS at your terminal, keeping the .MSSBCK file for later FETCHes today MSSBACKUP FETCH VMS0322 RD* ^-- fetch the files beginning with RD (do not replace any existing versions) MSSB DELETE VMS0322 ^-- Delete the BACKUP file from MSS MSSDELETE Delete several MSS files.

> Examples: MSSDELETE MYFILE ^-- same as HFT DELETE "MYFILE"

> > MSSD F1,F2,F3,F4,F5 ^-- delete 5 MSS files

MSSNEWCHRG Change the account number on your MSS files. Examples: MSSNEWCHRG 1222233344 1234567890 ^-- change job order number for all files currently stored with account number 1-2222-333-44 to 1-2345-678-90

ì

Page 4-1-9

*** Using the MSS from the CDC CYBER 860 ***

The MSS is just a peripheral on the CDC CYBER 860 and is under the control of NOS. All NOS files on the CYBER 860, whether they reside on disk or the MSS, are accessed by the standard NOS permanent file commands.

NOS/VE does not have direct access to the MSS. The following commands provide access to the Mass Storage System. A description of their syntax may be found in Appendix H.

CHANGE_LINK_ATTRIBUTES (chala) Change individual link attributes for communication between dual-state partners.

> Examples: chala pw=mymsspw ^-- required to access the MSS (see also MSACCES)

DISPLAY_LINK_ATTRIBUTES (disla) Display individual link attribute values.

> Examples: disla CHARGE : FAMILY : nlfam PROJECT : USER : AMDS

GET_FILE (getf)

Copy a NOS file to NOS/VE.

Examples: getf my_file myfile ^-- Get MSS file MYFILE and store as NOS/VE file MY FILE

REPLACE_FILE (repf) Copy a NOS/VE file to NOS direct file, replacing any existing file.

> Examples: repf my_file myfile ^-- copies NOS/VE file MY_FILE to MSS as MYFILE, replacing any existing MSS/NOS file with the same name

Page 4-1-10

MSACCES Access the Mass Storage System. Examples: MSACCES mymsspw MSAUDIT Obtain a sorted audit of your MSS files. Examples: MSAUDIT <-- short audit of your MSS files</pre> MSAUDIT, LO=F, SPW=Y Full audit showing each file's password, if any MSAUDIT audout s UN=other ^-- short audit of another user's files you are allowed to see with output in file AUDOUT in MSAUDIT F='V***** ^-- all files starting with "V" your current catalog MSCATLIST The NOS CATLIST command. MSCHANG Change the attributes of a Mass Storage System file. Examples: MSCHANG myfile CT=PU ^-- make MYFILE a public file MSCHANG, myfile, CP=1 ^-- change the account number MSCHANG myfile NFN=newfile M=E ^-- change the name of the file and make it execute-only MSFETCH Fetch a file from the Mass Storage System. Examples: MSFETCH, mssfyll, inl ^-- MSSFYL1 is retrieved and stored as file IN1 in your current catalog MSFETCH mssfy12 in2 D64 ^-- MSSFYL2 is retrieved, converted from 64-character NOS Display Code and stored as file IN2 in your current catalog

```
MSPASSW Change your Mass Storage System access password.
         Examples: MSPASSW, OLD=mymsspw, NEW=numsspw
                       -or-
                    MSPASSW mymsspw numsspw
                      -or-
                    MSPASSW N=numsspw O=mymsspw
                                   ^-- the above are the same
MSPURGE Purge a file on the Mass Storatge System.
         Examples: MSPURGE F=mymss1
                                   ^-- purge MSS file MYMSS1
                    MSPURGE (f1, f2, f3, f4, f5)
                                   ^-- purge MSS files F1, F2, F3, F4,
                                       and F5
MSSTORE Store a file on the Mass Storage System.
         Examples: MSSTORE, in1, mssfyl1
                                   ^-- IN1 is stored as private, direct
                                       file MSSFYL1
                    MSSTORE in2 mssfyl2 D64 PW=fylepw
                                   ^-- IN2 is stored as private file
                                       MSSFYL2 (even if MSSFYL2 already
                                       exists) in CDC Display Code --
                                       FYLEPW is the password required
                                       for another user to access the
                                       file -- if MSSFYL2 does not
                                       exist, this will be a direct file
                    MSSTORE in3 mssfy13 A6
                                                I PU
                                   ^-- IN3 is stored as public, indirect
                                       file MSSFYL3 in 64-character
```

90/09/30

Rev0

Display Code

Page 4-1-11

Page 5-1-1

***** DEC VAXcluster -- VMS *****

The Digital Equipment Corporation (DEC) VAXcluster has two 8550 central processing units (CPUs) or nodes, each with 48 megabytes of memory, which share files and are linked together. Access is via DTNET. A separate VAX 6410 for secure processing is located at Carderock and is accessed in the secure computer room.

*** VMS Version 5.3 ***

The operating system for the DEC VAXcluster and the VAX 6410 at NTRC is called VMS, version 5.3-1.

Permanent files (user programs and data files retained for frequent use) reside on disk drives and the Mass Storage System. User files, if not specifically requested on a tape, will be assigned to available disk areas.

*** Accessing the VAXcluster ***

To access the VAXcluster, set your terminal to 8-bit, no parity, then:

. dial (301) 227-5200 <-- this will connect you with DTNET at 1200 baud (see page 1-1-4 for higher speeds)

. after the phone call completes, or if you are hardwired into DTNET, press the RETURN key until it displays the DTNET> prompt

. enter "connect dt4" (or "c dt4") to connect to DT4
 (similarly for DT3)

. in response to the Username: prompt, enter your User Initials

. in response to the Password: prompt, enter your login
 password (the default VAX prompt is \$)

*** Login Password ***

Your initial login password is your username, usually your user initials. This is entered in response to

Password:

the first time you log in. This password MUST be changed during your first session.

To change your login password, type

SET PASSWORD

You will be prompted for the current password, the new password, and the new password again (to insure there were no transmission problems).

Your password should be changed frequently, and must be changed at least every 90 days.

*** Logout Procedures ***

To terminate your session, get rid of any unwanted permanent files (remember that new versions of a file may be made frequently during the session with up to five retained and costing you money). You may also want to get rid of any journal files made by EDT (.JOU) or EVE (.TJL).

When this is done, or immediately, if the Central Site operator requests it, type LOGOUT. A time and usage summary of the session and a cost estimate will be displayed.

You will be returned to your DTNET session. To leave, type "L". This will disconnect the phone, if you have dialed in.

Note

If you do not type anything for about 13 minutes, you will be logged off VMS automatically. You are given a 5-minute warning.

*** System News ***

At login, a system bulletin may be displayed. For more details, type NEWS. To see earlier news items, type OLDNews. To see ancient news items, type VERYoldnews.

*** Login Procedure File ***

A Login Procedure File is a file in your home directory with the name LOGIN.COM. It contains commands to be executed each time you log in before you are given the \$ prompt. Commands and qualifiers should be spelled in full to allow for possible future changes in the operating system.

Any command may be in LOGIN.COM. You may want to see who is logged in (\$ SHOW USERS /FULL), or look at your home directory files (\$ DIRECTORY) or all your files (\$ DIRECTORY [...]), or define one or more of your HELP libraries (\$ ASSIGN UOn: [myid] mylib HLP\$LIBRARY_5). You should also define your home directory with a logical name (such as your first name, but NOT your username) using (\$ DEFINE myname UOn: [xxxx]). Then, you need only type myname: to refer to your home directory, which you may need to do frequently. For suggestions of other commands, symbols and logical names you might include, type "HELP LOGIN.COM Hints". Page 5-1-4

Rev0 90/09/30

*** Files ***

1. Because VMS at DTRC automatically deletes the low version number when more than 5 versions of a file are created, you should not use different versions of a file for different purposes. Instead use the file type field.

2. To reduce your file space and, therefore, your costs, you may wish to do a "PURGE [xxxx...]" every now and then to remove all low versions (or "PURGE [xxxx...] /KEEP=2" to keep the highest two versions.

3. When editing with EDT or EVE, a journal file is created of all your editing commands for use in re-editing your file if your editing session is aborted (^Y or a line disconnect). (If your editing session ends normally (EXIT or QUIT), the journal file is deleted.) You should check periodically for any journal files and delete them if they are no longer needed. Use the command "DIRectory /DATE [...]*.JOU,*.TJL" to see them.

*** Batch Jobs ***

A batch job is a procedure which is submitted by the SUBMIT command. By default, the job will be executed on either DT3 or DT4. If your job must run on a specific node, use the /QUEUE=DTn_BATCH qualifier (n is the desired node number). See page 1-3 1 ff for a table of the nodes on which specific software is available.

** Killing Batch Jobs **

When a job is SUBMITted, a message is displayed giving the "entry number". When the job goes into execution, it has a "process ID". To find out these numbers, use

| \$ SHOW | QUEUE | *BATCH | < | all | VAX | clus | ster | batch | queues | | |
|------------|--------|--------|---|------|------|------|-------|---------|----------|------|---------|
| \$ SHOW | SYSTEM | 1 | < | prod | ess | ID | and | entry | number | on | current |
| | | | | node | e (i | E tŀ | ne jo | ob is i | in execu | utic | n) |

To delete a job which has not gone into execution:

\$ DELETE SYS\$BATCH /ENTRY=entry

To delete an executing job:

- - <-- current node -- pid is
 last 4 digits of process
 ID (leading zeros may be
 omitted)</pre>

Page 5-1-5

*** Accessing Other Networks ***

DTRC also has access to the following networks:

DDN - the Defense Data Network (with connection to INTERNET) (host tables allow transfer to some other networks)

OASYS - the DTRC Office Automation System

The following can be reached from our DECnet using SET HOST:

NAVSEA node names: SEAHUB, SEAA, SEAB, SEAC, SEAD, SEAE

** Checking Host Accessibility **

Host accessibility may be verified on the VAX using the FINDHOST command. FINDHOST will search a downloaded version of the name server host tables. You may enter the address, the host name, or any portion of either the host name or address. A listing will be given of all entries that meet the specified search. The search string will be highlighted in the resulting list.

\$FINDHost <string>

-or-

SFH <string>

If the requested string is not matched, you will get a message that there were no matches. Call User Services for more information.

* * ** Transferring VMS Files To and From OASYS While logged into DTn: ftp dtrc.dt.navy.mil <-- File Transfer Protocol to OASYS (dtrc) via Ethernet -orftp dtrc login <-- to log into OASYS <-- your OASYS user name <user name> <password> <-- your OASYS password get <-- get a file from OASYS <OASYS filename> <VMS filename> put <--- send a file to OASYS <VMS filename> <OASYS filename> <-- leave ftp bye ** ** Transferring VMS Files To and From CDC CYBER 860 While logged into DTn: ftp cdc860 <-- connect to CDC CYBER 860 / MSS</pre> <CR> <-- your name <password> <-- your MSS password <-- if requested <account number> <-- get a file from MSS get <MSS filename> <VMS filename> <-- send a file to MSS put <VMS filename> e.g., to make a public indirect <MSS filename> file: put myfile.dat 'myfile/ia,ct=pu' <-- leave ftp bye

See also page 5-1-8.

Page 5-1-7

** Mail to Users at Other Sites **

Mail may be sent to users at other sites which are accessible via DDN. This is one way to transfer large (or small) text files. Use FTP, Kermit, or some other protocol for binary files.

While logged into VMS: \$ mail MAIL> send To: wins%"<user@hostname>" <-- where some typical hostnames ... are: dtrc.dt.navy.mil, dtoal.dt.navy.mil, icst-is.arpa, gwuvax.gwu.edu)

For example, to send a message to "sommer" on dtrc (OASYS B system) from node DT4:

\$ mail
MAIL> send
To: wins%"sommer@dtrc" <-- the brackets are optional
...</pre>

Mail is sent via the VMS mail utility and the Simple Mail Transfer Protocol (SMTP). The "To:" address has one of the following forms:

| Destination | Address Syntax | Utility |
|---|-----------------------------------|-------------------|
| same VAX | user | local VMS mail |
| same network | node::user | DECnet |
| another VAX | wins%" <user@host>"</user@host> | SMTP |
| remote host | wins%" <user@host>"</user@host> | SMTP |
| remote host routed through other hosts on your network | wins%"<@host,@host:user@host>" | SMTP |
| remote host on another network routed through a gateway | wins%"<@host,@gateway:user@host>" | SMTP |

Note that local VMS and DECnet mail is sent immediately; SMTP mail is sent every 20 minutes.

*

** Mail From Users at Other Sites **

The following are the Ethernet addresses for CCF computers as of the date of this page. Type "HELP @CCF Network_addr" on the VAX for an up-to-date list.

| | | computer | | address | | |
|---|----|----------------|--------|--------------|-------|---------|
| | | oasys | | 130.46.1.53 | | |
| | | dtoal | | 130.46.1.2 | | |
| | | dtrc | | 130.46.1.3 | | |
| | | dtoa3 | | 130.46.1.4 | | |
| | | dtix | | 130.46.1.5 | | |
| | | nems | | 130.46.1.6 | | |
| | | d t70 | | 130.46.1.7 | | |
| | | dt18 | | 130.46.1.8 | | |
| | | dtvms3 | | 130.46.1.12 | (DT3) | |
| | | dtvms | dtvms4 | 130.46.1.10 | (DT4) | |
| | × | cdc86 0 | nos | 130.46.1.16 | | |
| , | ** | sn417 | | 192.91.138.5 | (Cray | UNICOS) |
| | | | | | | |

To access these via DDN add ".dt.navy.mil", e.g., dtvms.dt.navy.mil. Thus, the address for mail to be sent to user ABCD on the VAXcluster via DDN is "abcd@dtvms.dt.navy.mil".

* - No mail. This is listed to show the network address.

** - Available only when UNICOS is up.

Page 5-2-1

***** Help Libraries *****

A help library (file type .HLB) contains help modules, that is, modules that provide information about a program, subprogram, procedure, or some general help information such as hints on how to do something. It is created and accessed using the following DCL commands:

LIBRARY Create, maintain, list, and extract modules from a help library.

HELP Display the desired helps.

*** The System Help Library ***

The system help library is read using the DCL command HELP. It provides help about the HELP program and lists many topics (VMS features, DCL commands, Hints, and other general information).

*** DTRC Help Libraries ***

Four help libraries have been added to VMS at DTRC:

CCF - General information about the Computer Center
 CRAY - Routines added to Cray at DTRC
 DTLIB - Subprograms in library DTLIB (Cray, CDC NOS, VMS)
 UTILITIES - Utility programs and procedures

When executing the HELP command, the additional help libraries are accessed by entering '@name', where 'name' is one of the help libraries listed above (e.g., @DTLIB) in response to 'Topic?'. For a table of contents of any of the above libraries, type

HELP @name Contents

*** User Help Module ***

A help module (default file type .HLP) is a file containing all the help information for one or more programs, procedures, etc. Column 1 of each line identifies the different sections of the help module. A digit indicates a keyword; a slash (/) indicates a qualifier; anything else is part of the help text. For example,

```
1 \text{ key}-1
                                    <-- HELP topic
     . . .
 help message text
    . . .
2 key-2
                                    <-- HELP sub-topic
     . . .
 help message text
     . . .
n key-n
     . . .
 help message text
    . . .
                                    <-- next HELP topic
1 \text{ key} - 1
```

A "1" line gives the topic name (up to 15 characters, avoid using b'anks; replace blanks with an underscore (_)). A "2" line is a sub-topic of the "1"-level topic; a "3" line is a sub-topic of the most recent "2"-level sub-topic; etc. Qualifiers (/ in column 1) will be listed separately by HELP and will all be displayed if the (sub)topic they qualify is selected.

A help module might look something like:

```
1 topic
<description of topic>
2 Qualifiers
 <optional description of qualifiers>
/topic qualifier 1
 <description of topic qualifier 1>
/topic_qualifier 2
 <description of topic qualifier 2>
/topic qualifier 3
<description of topic qualifier 3>
2 sub-topic 1
<description of sub-topic>
3 sub-topic of sub-topic 1
 <description of sub-sub-topic>
3 Qualifiers of sub-topic 1
 <optional description of qualifiers>
/sub-topic_l_qualifier_l
<description of qualifier_1 of sub topic 1>
/sub-topic_l_qualifier_2
<description of qualifier_2 of sub_topic l>
. . .
```

*** Hints For Designing Help Displays ***

While help messages can continue without interruption, you may wish to format the messages to fit the screen display. A topic ("1" in column 1) will have 17 lines in the first display; a sub-topic ("2" in column 1) will have 15 lines; a sub-sub-topic ("3" in column 1) will have 13 lines; etc. For all levels, the second and following displays have 20 lines. Level 1 lines should not exceed 78 columns; level 2 lines should not exceed 76 columns; level 3 lines, 74 columns; etc. Longer lines may "wrap around".

Every help library should have a module called "HELP" to describe the help library.

You may wish to have a table of contents module (suggested name "Contents") to list the routine names and give a short description of what each routine does.

If possible, the first help screen for a program, subprogram or procedure should contain all that is needed to use it. Definitions of parameters and qualifiers should be put into sub-topics.

*** Selecting (Sub)topic Names ***

While you may choose anything you want for topic and sub-topic names, we recommend the following conventions:

- . use upper case for routine names, parameters, and qualifiers (e.g., AUXPRINT, /CC, /HEADER, JGDATE, FLR below)
- . use lower case (first letter upper case) for general information (e.g., Parameters, Qualifiers, Examples, Admin_info below)
- . replace blanks with underscores (_) so that the name will be listed as a single element by HELP (e.g., Admin_info below)

*** Create a Help Library ***

The LIBRARY command is used to create a help library.

LIBRARY /HELP /CREATE help_library_name

-or-

LIBRARY /HELP /CREATE=(option,...) help library name

where help_library_name is the name of the library to be created. It will have the default filename help library name.HLB.

The following options may be specified:

- BLOCKS:n The number of 512-word blocks to be allocated. (default: 100)
- HISTORY:n The maximum number of library update history records to be maintained. (default: 20)
- KEYSIZE:n The maximum length of module names. (default: 15)
- MODULES:n The maximum number of modules the library can hold. (default: 256)

*** Modify a Help Library ***

The LIBRARY command is used to insert, and delete help library modules. Wildcards are allowed in module names.

LIBRARY /HELP /INSERT help_library_name help_module_name

LIBRARY /HELP /REPLACE help_library_name help_module_name

LIBRARY /HELP /DELETE= (module [,...]) help_library_name

'LIBRARY /HELP help_library_name help_module_name' is the same as if '/REPLACE' were specified. If '/LOG' is specified, a messages will be displayed for each operation done. (E.g., LIBR /HELP /LOG ...)

*** Compress a Help Library ***

After several inserts, deletes or replaces, there may be a lot of "dead space" in the library. To remove this, that is, to compress the library, use:

LIBRARY /HELP /COMPRESS help_library_name

-or-

5.0

LIBRARY /HELP /COMPRESS=(option,...) help_library_name

/LOG will list the modules as they are copied into the compressed library.

The options available are the same as for /CREATE.

Page 5-2-5

*** List the Contents of a Help Library ***

The LIBRARY command also lists the contents of a help library. The /LIST qualifier, which may be specified alone or with any of the above operations, will provide information about the library including a list of the modules in the library. If /FULL is also specified, the list of modules will include the date and time it was inserted into the library. If /HISTORY is specified, it will show who did what to the library and when. The number of history records retained is defined when the library was created or compressed.

For a list of the library without other operations, use

LIBRARY /HELP /LIST -or-LIBRARY /HELP /LIST /FULL -or-LIBRARY /HELP /LIST /FULL /HISTORY

The list will be displayed on SYS\$OUTPUT. To put the listing into a file, use /LIST=filespec.

To list information about specific modules, use /MODULE=(list) where <list> is a comma-separated list of module names with wildcards allowed. The default is /MODULE=*.

To list information about modules inserted after a certain time, use /SINCE (for those inserted today) or /SINCE=date_and_time (for those inserted after a specific date and/or time (e.g., /SINCE=09:00 for those after 9 AM today).

*** Extract a Help Module ***
To extract a help module to make some modifications to it, use
LIBRARY /HELP /EXTRACT=(module[,...])
/OUTPUT=file-spec

help_library_name

If /OUTPUT is specified, the modules are put into file <file-spec>. If /OUTPUT is omitted, they are put into file help_library_name.HLP.

Wildcards are allowed in module names.

*** Accessing your Help Library ***

To access you help library, use

HELP /LIBRARY=filespec [topic [sub-topic]]

where <filespec> must be complete (e.g., U09:[abcd]mylib), not just the filename.

*** Adding Your Help Library to the System Helps ***

The DCL HELP command supports many user libraries in addition to the system library. User libraries are added by assigning help library names to HLP\$LIBRARY_n, where n is omitted or a digit. HLP\$LIBRARY through HLP\$LIBRARY_4 are already defined at LOGIN. You may add your own help libraries starting with HLP\$LIBRARY_5. For example, you may wish to put

\$ DEFINE /NOLOG HLP\$LIBRARY_5 UOn:[myid]mylib1 \$ DEFINE /NOLOG HLP\$LIBRARY_6 UOn:[myid]mylib2

into your LOGIN.COM file so that your help library will always be part of the system HELP command for you. The first missing number (in this case "7") will end the list. These will be listed at the end of the last screen of the topic display. To access library "5" above, use "HELP \$mylib1", or "@mylib1" at the Topic? prompt.

*** Using HELP ***

The HELP command access the system help library ("HELP"), your library set ("HELP @libname"), or any other help library ("HELP /LIBRary= filespec").

On initial entry into a help library, the help module is displayed, if present, a list of topics, and, perhaps, the library set. At the "Topic?" prompt, enter the name of the topic for which you want help. Only as many characters as are needed to uniquely identify the topic are required. If the name is not unique, all matching topics are displayed.

After the topic has been displayed (may be more than one screen), a list of additional information (sub-topics) may also be shown. At the prompt, enter the sub-topic name.

When you have finished with a level, press RETURN to go up one level. Pressing RETURN at the "Topic?" prompt exits the HELP command. At any prompt (even in the middle of typing an entry, ^Z (CTRL-Z) will terminate HELP.

Enter a question mark (?) at any time to display the most recent (sub)topic again. The actual help displayed depends on how you got to the current level. The RETURN key should not be pressed with the "?", since the "?" is recognized immediately. (If a help library is entered from a program other than the HELP command, the RETURN is required after the "?".)

If you have forgotten the names of the additional (sub)topics, just enter something you know is not a (sub)topic name (in most cases, "ZZ" is sufficient). This will display an error message and show the valid (sub)topic names.

The up-arrow key may be used to bring back your most-recent entry, which may be edited and resubmitted.

Page 5-2-7 *** *** Sample Help Modules The following are sample help modules for a program, a subprogram, a procedure, general information; and a HELP help module. ** A Program $\star\star$ The following is a portion of the help module for the AUXPRINT program. **1 AUXPRINT** <-- topic List a file on an auxiliary printer (one attached to an interactive terminal). Format: ! Defaults AUXPRINT file-spec [/ [NO] CC /NOCC 1 1 [/ [NO] HEADER] /NOHEADER 1 [/LENGTH=1] 1 /LENGTH=66] [/SKIP=s ! /SKIP=0; ! /SKIP ==> /SKIP=10 [/WIDTH=w] ! /WIDTH=80; ! /WIDTH ==> /WIDTH=132 2 Parameter <-- sub-topic file-spec Specifies the name of the file to be printed. If omitted, you will be prompted for it. Defaults: extender - .DAT; filename - FOR002 2 Qualifiers <-- sub-topic The qualifiers may follow the command name or the file-spec. If a qualifier is specified more than once, only the final value is retained. /cc /CC /NOCC Specifies whether the file has carriage control in column 1 of each line. Default: /NOCC (that is, the file does not have carriage control in column 1)

90/09/30

Rev0

```
/HEADER
 /HEADER
 /NOHEADER
   Determines whether the listing will have a heading giving the
   date and file-spec.
 Default: /NOHEADER
2 Admin info
                                                          <-- sub-topic
                 VAX/VMS Fortran 77
  Language:
  Authors:
                 Dan Allen
                                 - DTRC Code 189.2
                 David V. Sommer - DTRC Code 3511
  Date written: 10/81 (da)
  Dates revised
    03/14/85 - dvs
                        - add qualifiers /CC /HEADER /LENGTH /SKIP
    03/14/85 - dvs - add qualifiers /CC /HEADER /1
10/22/85 - dvs - shorten /CC output by 1 line
               systems - change default to /NOHEADER
    03/07/86 - dvs
                        - add /WIDTH qualifier
                        - fix /CC processing when first top-of-page
                           is not first record
```

Page 5-2-9

** ** A Subprogram This illustrates a subprogram help module. We suggest that such a help have the following sub-topics: . Parameters (if the routine has them) (at least one example to show how to use the . Examples routine) . Admin info (to show the source language, author, a brief history, and anything else that might be appropriate) **1 JGDATE** <-- topic Convert any Gregorian date to a relative Julian number or vice versa. INTEGER jg, jd, gyear, gmonth, gday Usage: CALL JGDATE (jg, jd, gyear, gmonth, gday) The relative Julian number corresponding to a Gregorian date is the number of days since 11/24/-4713 (extrapolating the Gregorian calendar). This subroutine is useful in determining the elapsed number of days between any two calendar dates. It can also be used to find the calendar date so many days from any given date. 2 Farameters <-- sub-topic CALL JGDATE (jg, jd, gyear, gmonth, gday) - in - int - direction of conversion jg 1 - Gregorian to Relative Julian 2 - Relative Julian to Gregorian jg=1: jd - out - int - will contain relative Julian number gyear - in - int - Gregorian year (e.g., 1985) gmonth - in - int - Gregorian month (1-12) - in - int - Gregorian day (1-31) gday - in - int - relative Julian number jg=2: jd gyear - out - int - will contain Gregorian year (e.g., 1985) gmonth - out - int - will contain Gregorian month (1-12) gday - out - int - will contain Gregorian day (1-31) 2 Examples <-- sub-topic INTEGER jd, gy, gm, gd CALL JDDATE (1, jd, 1985, 2, 25) jd = jd + 1000CALL JGDATE (2, jd, gy, gm ,gd) This example will find the date 1000 days from 02/25/85.

Page 5-2-10

Rev0 90/09/30

2 Admin info <-- sub-topic Language: Fortran 77 David V. Sommer - DTRC Code 3511 Author: Date written: 1968 or earlier Dates revised 03/01/79 - implement on Burroughs 7700 02/01/85 - implement on DEC VAXcluster ** A Command Procedure ** The procedure FLR has the following definition for all users: \$ FLR :== @VSYS:FLR Without this definition, the "Format" would have **@VSYS:FLR** [filename] <-- topic 1 FLR Compile Fortran, Link and Run. Format: FLR [filename] If filename is omitted, you will be prompted for it. For execution, FOR005, FOR006 and SYS\$INPUT are assigned to the

terminal. Thus, all Fortran READ, PRINT, READ (5,..., WRITE (6,..., TYPE, and ACCEPT statements will read from or write to the terminal.

Ignore the system message "previous value of SYS\$INPUT has been superseded".

Page 5-2-11

** General Information **

The following is a portion of the help module for a discussion of the DTRC accounting for users with more than one account. This module has no sub-topics.

1 Many accounts

VAXcluster users with more than one account are assigned a username/ password for each account. These usernames differ in the fifth character position, e.g., CAWE, CAWEA, CAWEB. The default login directory for each user is device: [username] where all files owned by the same individual are stored on the same device. For example,

U01: [CAWE] U01: [CAWEA] U01: [CAWEB]

ACCESSING FILES OWNED BY YOUR ALTER EGO

The "usernames" belonging to a particular user are members of a VMS "group". By default on the VAXcluster, members of a group have Read and Execute access to all files owned by their fellow group members. User CAWEA wishing to access a file owned by CAWE simply references [CAWE] file.ext .

Of course, these access rights can be changed by the SET PROTECTION and SET FILE /ACL commands. In addition, all members of these special "groups" have GRPPRV privilege which, when invoked, gives a member of the group full control, including file creation and deletion, over all files owned by all members of the group. GRPPRV is invoked by

\$ SET PROCess /PRIVileges=GRPPRV

(this would likely be in your LOGIN.COM)

Then to "copy" a file from one account to another, for example CAWE to CAWEA, user CAWEA would

\$ COPY [CAWE] file.ext []

or user CAWE would

\$ COPY file.ext [CAWEA]

To simply "move" a file from one account to another, CAWEA would

\$ RENAME [CAWE]file.ext []
\$ SET FILE /OWNer_uic=CAWEA

Finally, the command MYACcount will indicate the account number of the current session or job, while MYACcount /ALL will provide a list of all user/account pairs in the group.

Page 5-2-12

;

** "HELP" module **

It is recommended, though not necessary, that your help library have a help module named HELP. Such a module will be displayed when you enter the library, and, therefore, should give a brief description of the library and, if appropriate, pointers to related libraries.

The following is the help module HELP for library @CCF:

1 HELP

The CCF help modules provide information of general interest to users of the DTRC Central Computing Facility.

Other help libraries available include:

@CRAY- DTRC additions to Cray@DTLIB- subprograms in library DTLIF (formerly NSRDC)@UTILITIES- utility programs and procedures

Last modified: 31-JUL-1990 13:05:35

***** Procedures *****

A procedure is a group of control statements in a file (default file type .COM). Calling a procedure provides a simplified way to process that group of control statements. A procedure may call another procedure.

Eight parameters, P1 through P8, are available for you (or another procedure) to pass data or other information to a procedure.

Both string and integer variables may be used in a procedure. Several lexical functions are available to interrogate the system, to manipulate variables, etc. Files may be read or written. An:, of course, DCL statements may be executed.

*** DTRC Procedures ***

Type HELP @UTILITIES CONTENTS for a list of procedures (and programs) which have been added to the DTRC VAX/VMS system.

Page 5-4-1

***** Object Libraries *****

An object library (file type .OLB) contains compiled subprograms for use in linking with a program.

The Librarian utility LIBRARY is used to create, maintain, list, and extract modules from an object library.

*** DTRC Object Library ***

One object library has been added to VMS at DTRC:

VSYS:DTLIB - Subprograms written or maintained by the Computer Center

To use: LINK yourobj, DTIB/LIB

*** User Object Module ***

An object module (file type .OBJ) is a file containing one or more compiled subprogram(s). They are produced by compiler such as FORTRAN, COBOL, PASCAL, etc.

*** Create an Object Library ***

The LIBRARY command is used to create an object library.

LIBRARY /CREATE object library name

-or-

LIBRARY /CREATE=(option,...) object_library_name

where object_library_name is the name of the library to be created. It will have the default filename object library name.OLB.

The following options may be specified:

- BLOCKS:n The number of 512-word blocks to be allocated. (default: 100)
- GLOBALS:n The maximum number of global symbols the library can contain. (default: 128)
- HISTORY:n The maximum number of library update history records to be maintained. (default: 20)

Rev0 90/09/30

KEYSIZE:n The maximum length of module names. (default: 15)

MODULES:n The maximum number of modules the library can hold. (default: 256)

*** Modify an Object Library ***

The LIBRARY command is used to insert, and delete object library modules. Wildcards are allowed in module names.

LIBRARY /INSERT object_library_name object_module_file

LIBRARY /REPLACE object library name object module file

LIBRARY /DELETE=(module[,...]) object library name

'LIBRARY object_library_name object_module_file' is the same as if '/REPLACE' were specified. If '/LOG' is specified, a message will be displayed for each operation. (E.g., LIBR /LOG ...)

If object_module_file contains several object modules, each will be a separate entity in the object library.

If the qualifier /NOGLOBALS is specified, the global symbols for the modules being inserted will not be put into the global symbol table.

*** Compress an Object Library ***

After several inserts, deletes or replaces, there may be a lot of "dead space" in the library. To remove this, that is, to compress the library, use:

LIBRARY /COMPRESS object library_name

-or-

LIBRARY /COMPRESS=(option,...) object_library name

/LOG will list the modules as they are copied into the compressed library.

In addition to the options available for /CREATE:

KEEP Copy the history records, etc., to the compressed library. (default: do not copy) *** List the Contents of an Object Library ***

The LIBRARY command also lists the contents of an object library. The /LIST qualifier, which may be specified alone or with any of the above operations, will provide information about the library including a list of the modules in the library. If /FULL is also specified, the list of modules will include the date and timeit was inserted into the library. If /HISTORY is specified, it will show who did what to the library and when. The number of history records retained is defined when the library was created or compressed.

For a list of the library without other operations, use

LIBRARY /LIST -or-LIBRARY /LIST /FULL -or-LIBRARY /LIST /FULL /HISTORY

The list will be displayed on SYS\$OUTPUT. To put the listing into a file, use /LIST=file-spec.

If the qualifier /NAMES is specified, the names of all global symbols will also be listed.

*** Extract an Object Module ***

To extract an object module to make some modifications to it, use

LIBRARY /EXTRACT=(module[,...] /OUTPUT=file-spec object_library_name

If /OUTPUT is specified, the modules are put into file <file-spec>. If /OUTPUT is omitted, they are put into file object module name.OBJ.

*** Linking with an Object Library ***

If your program uses subprograms in an object library, they can be linked using

LINK your obj, your lib/LIBrary

where your_obj is the object module for your program
your_lib is your object library
/LIBrary tells the linker that your lib is an object library

If you are linking more than one object file or using more than one object library, you might use one of the following forms:

> LINK obj1, obj2, lib1/LIB LINK obj1, obj2, lib1/LIB, lib2/LIB LINK obj1, obj2, lib1/LIB, obj3 LINK obj1, obj2, lib1/LIB, obj3, lib3/LIB

etc.

Page 5-5-1

***** Text Libraries *****

A text library (file type .TLB) contains text modules, that is, modules containing source programs, documents, notes, data, etc.

The Librarian utility LIBRARY is used to create, maintain, list, and extract modules from a text library.

*** DTRC Text Libraries ***

The following text libraries have been added in VSYS: at DTRC.

DTLIB - Source code for subprograms in library VSYS:DTLIB.OLB

- DTLIBCRAY Source code for subprograms in library DTLIB on the Cray
- INCLUDE Some common block and code segments to INCLUDE in a program or subprogram
- UTILITIES Source code for programs which have been added to VSYS:

*** User Text Module ***

A text module (default file type .TXT) is a file containing a source program, a document, some miscellaneous information, etc.

*** Create a Text Library ***

The LIBRARY command is used to create a text library.

LIBRARY /TEXT /CREATE text library name

-or-

LIBRARY /TEXT /CREATE=(option,...) text library name

where text_library_name is the name of the library to be created. It will have the default filename text library name.TLB.

The following options may be specified:

- BLOCKS:n The number of 512-word blocks to be allocated. (default: 100)
- HISTORY:n The maximum number of library update history records to be maintained. (default: 20)
- KEYSIZE:n The maximum length of module names. (default: 15)
- MODULES:n The maximum number of modules the library can hold. (default: 256)

*** Modify a Text Library ***

The LIBRARY command is used to insert, and delete text library modules.

LIBRARY /TEXT text_library_name text_module_file /INSERT LIBRARY /TEXT text_library_name text_module_file /INSERT /MODULE=module_name LIBRARY /TEXT text_library_name text_module_file /REPLACE

LIBRARY /TEXT text_library_name text_module_file /REPLACE /MODULE=module_name

LIBRARY /TEXT text library_file /DELETE=(module[,...])

"LIBRARY /TEXT text_library_name text_module_file" is the same as if "/REPLACE" were specified. If "/MODULE=..." is omitted, the module name will be the filename without the file type. If "/LOG" is specified, a message will be displayed for each operation. (E.g., LIBR /TEXT /LOG ...)

Wildcards are allowed in the module names when deleting.

Page 5-5-3

*** Compress a Text Library ***

After several inserts, deletes or replaces, there may be a lot of "dead space" in the library. To remove this, that is, to compress the library, use:

LIBRARY /TEXT /COMPRESS text library name

-or-

LIBRARY /TEXT /COMPRESS=(option,...) text_library_name

/LOG will list the modules as they are copied into the compressed library.

The options available are the same as for /CREATE.

*** List the Contents of a Text Library ***

The LIBRARY command also lists the contents of a text library. The /LIST qualifier, which may be specified alone or with any of the above operations, will provide information about the library including a list of the modules in the library. If /FULL is also specified, the list of modules will include the date and time it was inserted into the library. If /HISTORY is specified, it will show who did what to the library and when. The number of history records retained is defined when the library was created or compressed.

For a list of the library without other operations, use

LIBRARY /TEXT /LIST -or-LIBRARY /TEXT /LIST /FULL -or-LIBRARY /TEXT /LIST /FULL /HISTORY

The list will be displayed on SYS\$OUTPUT. To put the listing into a file, use /LIST=file-spec.

*** Extract a Text Module ***

To extract a text module to make some modifications to it, use

LIBRARY /TEXT /EXTRACT=(module[,...]) /OUTPUT=file-spec text_library_name

If /OUTPUT is specified, the modules are put into file <file-spec>. If /OUTPUT is omitted, they are put into file text_library_name.TXT.

Wildcards are allowed in the module names.

Page 5-6-1

***** Editors *****

VAX/VMS has two widely-user text editors: EDT and EVE; and a Text Processing Utility (TPU) which can be used to create your own editor. EVE is a editor written in TPU. This chapter gives an overview of EDT and EVE.

*** The EDT Text Editor ***

EDT is used to create or modify a file. There are three modes for using EDT: line, keypad (which uses the full screen), and non-keypad. Line mode is very similar to NETED on the CDC CYBER 176 or 750.

** Invoking EDT **

EDT is executed by:

\$ EDIT /EDT file
or
\$ EDIT file <--- /EDT is the default editor</pre>

where file may be a file specification or a logical name.

If the file is an existing file, the first line of the file will be displayed on the screen, followed by an * (the * is the prompt when in line mode). If the file does not exist, [EOB] will be displayed on the first line, followed by the * prompt. You are now ready to edit the file. A journal file of every command you enter is saved temporarily in filename.JOU. If EDT is terminated abnormally (including your session being disconnected), you can recover almost all of your editing by "EDIT /RECOVER file".

To change to screen mode, type "change" or "c" at the * prompt. To return to line mode, enter end-of-file (^2).

** On-line HELP **

Help is available in both line mode and keypad (change) mode. In line mode, at the * prompt, type "HELP" or "HELP command". Keypad mode uses the PF2 commmand to invoke the help utility. EDT will paint a picture of the keypad and prompt you to push the key for which you need help.

** Terminating EDT **

There are two ways to leave EDT: "EXIT", which saves the file; and "QUIT" which does not save it. If, for some reason, you wish to save the journal lile, "EXIT /SAVE" will save both the file and the journal file.

*** The EVE Editor ***

The Extensible VAX Editor, EVE, is a full-screen interactive text editor designed for use with VT100- and VT200-compatible terminals. Some features include multiple files and buffers, two windows, and some word processing commands. Advanced editing commands are entered through the use of a command line.

EVE has its own keypad. The EDT keypad may be used by typing "SET KEYPAD EDT" on the command line. In developing EVE, DEC has attempted to simplify the EDT keypad by reducing the number of keystrokes for each keypad command to one.

** Invoking EVE **

To begin an EVE session, enter

```
$ EVE
-or-
$ EVE file
```

A wildcard character, the asterisk, can be substituted for all or some of the characters in a long file name. If one file name matches the specification, that file is edited; otherwise, an error message is issued and no file is used. For example,

\$ EVE getty.txt
\$ EVE this_is_a_long_file_name.and_a_long_file_type
\$ EVE this *.and *

** The Screen **

The screen is divided into four parts. The first part, the window, contains the file's text. If the file is empty, you will only see the [End of file] notice. The second part, the status line, is highlighted, contains the current buffer name, mode, and direction. The third, the command line, displays advanced EVE commands. The fourth part, the message window, displays both informative and error messages.

** On-line Help **

EVE has both keypad help as well as an extensive "word processing" format help menu.

** Terminating EVE **

Increase two commands that allow you to leave the EVE environment. To terminate EVE and save the file, type end-of-file (^Z). This will create a new file or another version of an existing file. To leave EVE without saving your changes, press the DO key and then type QUIT. If your editing session ends abnormally, the "EVE /RECOVER file" command can be used to recover your session using journal file file.TJL.

Page 5-6-3

*** Why Use EVE Instead of EDT? ***

EDT users should consider switching to EVE for the following reasons:

- . EVE's use of windows allows editing multiple files simultaneously on the same screen. This is useful for making common changes to programs and subprograms or for moving lines from one file to another.
- . EVE has more ways of extending the basic editor and saving those extensions for future sessions than EDT.
- . EVE's string searching capability is much more flexible than EDT. It includes VMS and UNIX wildcard searching.
- . EVE offers "spawn" and "attach" and "DCL" commands to allow the user to work outside of the current process and return to the same active EVE session.
- . EVE supports the EDT keypad.

Page 6-1-1

***** Magnetic Tape *****

Magnetic tapes should be used for sequential data for such purposes as:

- . Transfer of information to and from other computers and
 - off-line peripherals
- . Files which are used infrequently
- . Back-up copies of disk files
- . Long-term storage of data

Tapes should not be used for scratch files or random information. For safety, two copies on different tapes should be maintained, or for data which is updated, a grandfather-father-son system is advised. It is not wise to mount a tape containing good data, read through it, and write new data at the end. Instead, copy the existing data to a second tape and add the new data to the second tape, retaining the first tape as a back-up.

Processing a file on tape will take considerably more I/O time than on disk and more elapsed time.

Information concerning the physical and logical characteristics of the tape is specified in control statements.

Nine-track tapes are supported on the DEC VAX and CDC CYBER 860 computers; 7-track tapes are supported on the CDC CYBER 860 (NOS only). There are no tape drives on the Cray, so tapes must be accessed via one of the front ends.

*** Tape Labels ***

Tapes may be labelled or unlabelled. Labels should always be used except when writing data for, or reading data from a computer which cannot handle ANSI standard labels.

In general, a labelled tape has volume and end-of-volume labels, and may also have user labels. Each file on the tape may have its own header and trailer labels.

*** Tape Formats ***

Generally, records on tape are fixed or variable length, blocked or unblocked, ASCII or EBCDIC (9-track), BCD (7-track), coded, or binary. Where possible, tapes written by or for another computer should be 9-track, 6250 or 1600 cpi, fixed length, blocked, ASCII. *** Tape Care and Cleaning ***

Tapes should be stored in closed containers in racks which give them vertical support. Tapes may not be spliced. They should be read and rewound at least every six months. Logs should be kept on contents, format, and creation dates of tapes.

If a tape has many parity errors, cleaning it may help. Even a brand new tape may need cleaning. This off-line process does not destroy the information on the tape. If a tape receives heavy usage, cleaning it after ten or more uses may reduce the incidence of parity errors. A tape can also be certified, which determines whether there are any areas on the tape which do not record properly. Certification DESTROYS current information on the tape (except VSN). To change the VSN, contact the Tape Librarian and request blank labelling or degaussing.

If, after a tape has been cleaned, it still has many parity errors, call User Services to have the tape drive cleaned. If the tape continues to have parity errors, it should be exchanged for a new tape. The information on the old tape is not recovered automatically in this case.

To have a tape cleaned or certified, submit an off-line work request to the Tape Librarian. Users who are not at the Carderock site should call (301) 227-1967.

When possible, slot tapes should be in the Computer Room environment for at least two hours before reading or writing. This allows temperature and humidity to stabilize and should minimize tapc problems.

Please notify Code 3511 (User Services), (301) 227-1907, of any unusual tape problems.

Page 6-1-3

*** Tape Assignment ***

Two classes of tape storage are provided in the Computer Center, 'Library' and 'Slot'. Tapes which are used frequently should be permanently stored in the NA cabinet, which is accessible from the CYBER 860 or VAXcluster. These tapes are assigned a permanent external label indicating location by cabinet, shelf and position, such as 'NA2499', and are referred to as 'Library' tapes. The volume serial number (VSN) of a Library tape is the same as the external label and should usually be a labelled tape.

Tapes which are seldom used on the CDC CYBER or VAX computers, which are being transferred between systems, or which are normally retained by the owner are assigned a temporary slot number for up to 24 hours at the computer on which they are to be used. At the end of the day's processing (or earlier at the user's request), these are returned to the ADP Control Center for pickup by the user and will require a new slot number assignment for the next use.

The VSN for a slot tape is 'SLOTxx=id' where xx is the assigned slot number id is the user's external sticker on the tape reel (six (6) one-inch-high characters, please, for easy reading by the operator)

Tapes belonging to remote users may be sent to the Tape Librarian. Special slots may be assigned for several weeks' continuous usage (on the CYBER 860, these are C1-9, Y1-9, B1-9, R1-9; on VAXcluster these are V1-9, A1-9, X1-9, S1-9).

All tapes to be used in a job must be supplied by the user as Library tapes and/or Slot tapes. No scratch tapes are available.

Tapes stocked by the Computer Center are of 2400-foot nominal length (10.5 in. diameter). Smaller tapes may be used. For remote slot assignment, assignment of library tapes, or to arrange for the purchase of tapes, contact the Tape Librarian, (301) 227-1967. CYBER procedure BEGIN, TPGET may be used to acquire NA tapes. Slot tapes may be signed in at the ADP Control Center. *** Using Tapes on the DEC VAX ***

The DEC VAXcluster has four 9-track tape drives (6250/1600 cpi).

The following VMS control statements are used to access or analyze magnetic tapes:

ALLOCATE Assign a tape drive to a logical name.

DEALLOCATE Return a previously allocated device and disassociate the job's logical name from the tape drive.

DISMOUNT Release a tape volume that was previously mounted.

INITIALIZE Initialize a magnetic tape.

MOUNT Mount a magnetic tape and, if labelled, check the label.

The following prodecures have been developed to handle the tape mounting and dismounting for you:

COPYD2T Copy disk files to a VAX tape using COPY.

COPYT2D Copy a VAX tape (written by COPY) to disk.

- FILEMANAGER An interactive procedure using the VMS BACKUP utility to create, add to, restore from, or list the contents of a backup tape.
- RFTAPE Read Foreign TAPE (copy tape-to-disk). Reads one or more files from a fixed, blocked or unblocked, ASCII or EBCDIC tape and saves them on disk.
- WFTAPE Write Foreign TAPE (copy disk-to-tape). Writes one or more disk files to a fixed, blocked or unblocked (ASCII or EBCDIC) tape.

Page 6-1-5

** Examples **

1. Initialize a VAX/VMS tape:

```
$! TAPINIT.COM : initialize VAX/VMS tape, default is 1600
S!
$
     if p3 .nes. "1600" .and. p3 .nes. "6250" then p3 = "1600"
$!
$
     allocate mu: tape
                                       ! get any available tape drive
$!
$
     mount tape: /foreign /density='p3' -
                 /comment="mount slot''pl' vsn=''p2' ringin"
$
     dismount tape /nounload
$
     initialize tape 'p2'
$
     deallocate tape
$
     exit
$!
$! p1 - 1- or 2-digit slot number or NONE
$! p2 - 6-character VSN
$! p3 - density (6250 or 1600) defaults to 1600
$!
$! created 06/23/88 by CASG
$! last modified 06/24/88 @ 1146 by CASG (add "?" for help)
$!
$! End of TAPINIT.COM
```

The above is a portion of the actual procedure to show just the defaulting of density and how to initialize a tape. To see the full procedure, which includes validation of each parameter, and allows "?" for help for the procedure and each parameter, type "TYPE VSYS:TAPINIT.COM".

Rev0 90/09/30

*** Using Tapes on the CYBER 860 ***

The CDC CYBER 860 has six 9-track tape drives (four for 6250/1600 cpi and two for 1600/800 cpi), and two 7-track tape drives (800/556 cpi). All drives are available to NOS; two 9-track (6250/1600 cpi) drives are available to NOS/VE.

** NOS **

The following NOS control statements are used to access or analyze magnetic tapes:

LABEL Mount a magnetic tape and, if labelled, check the label.

LISTLB List the labels of an ANSI-labelled tape.

RESOURC Specify that more than one tape drive is required.

TDUMP Octal and alphanumeric dump of all or part of a file.

VSN Associate a local file name with one or more volume serial numbers.

* Examples *

The following examples illustrate tape usage in batch jobs. Tapes may also be used interactively (without the job, USER and CHARGE statements).

1. Unlabelled NOS/BE tape to disk:

xxxx. job statement. USER,xxxx,upw. CHARGE,1234567890. DEFINE,disk/CT=PU. LABEL,tape,F=SI,LB=KU,VSN=NA9999,D=1600,PO=R,R. COPYBF,tape,disk,5. UNLOAD,tape.

2. Copy old stranger (foreign) tape to new - 6250 multifile:

xxxx. USER,xxxx,upw. CHARGE,1234567890. RESOURC,GE=1. <--- one additional tape drive LABEL,t4,VSN=NA9998,D=GE,PO=W,W,F=S,L=softwstr. VSN,t5=SLOTxx=CA9995. LABEL,t5,PO=R,R,F=S,D=GE,L=softwstr. COPY,t5,t4,EL=10,M=coded,PO=E. UNLOAD,t5.

Page 6-1-7

** NOS/VE **

The following NOS/VE control statements are used to access or analyze magnetic tapes:

CHANGE_TAPE_LABEL_ATTRIBUTES Change the current magnetic tape label attributes.

DETACH FILE

Detach one or more files from a job.

DISPLAY_BACKUP_LABEL_TAPE Display the current job default label type for a permanent file backup file on tape.

DISPLAY_TAPE_LABEL_ATTRIBUTES Display the current magnetic tape label attributes.

REQUEST_MAGNETIC_TAPE Associate a file with a magnetic tape.

SKIP_TAPE_MARK Position a tape backward or forward.

* Examples *

1. Read an unlabelled tape on VE:

/set_working_catalog \$user /change_block_label_type file_label=u /request_magnetic_tape file=\$local.tape/ external_vsn=mytape/ type=mt9\$1600 /copy_file \$local.tape myfilel /detach \$local.tape

```
2. Create a multi-file labelled tape:
  /reqmt file=$local.tapel
                             • •
          external vsn=TAPE02 ..
   ../
   ../
          recorded vsn=TAPE02 ..
   ../
          ring=false ..
  ../
          density=mt9$6250
   /chatla f=$local.tapel ..
           rf=true
   ../
           file identifier=file1 ..
   ../
           file_set_identifier=many1
   ../
   /set file attributes f=$local.tapel ..
  ../
                        block_type=user_specified_record ..
   ../
                        record_type=ansi fixed ...
   ../
                        maximum_record_length=80
                           $local.tape1
   /copf
           filel
   /chatla f=$local.tapel fi=file2
           file2
                           $local.tape1
   /copf
   /chatla f=$local.tape1 fi=file3
   /copf
           file3
                           $local.tape1
   /distla f=$local.tape1 do=current_file " display label written
   /detach $local.tape1
```

Page 7-1-1

***** Other Software *****

This chapter discusses various programming languages and other software packages available on the CCF computers.

ABAQUS A family of modeling capabilities based on the finite element method, designed to provide solutions to a wide range of mostly non-linear structural problems, and programmed around a common data management structure.

Execution: Cray COS: from the VAX: @VSYS:ABACRAY

DEC VAX/VMS: @VSYS:ABA

Post-processing of Cray or VAX runs is done on the VAX: @VSYS:ABAPLOT

Remarks: Processing is normally done on the Cray unless more memory is required than is available.

For Cray processing with .FIL output files, the .INP file must include "*FILE FORMAT, ASCII".

If a plot file is generated on the Cray, each *PLOT statement must include "OUTPUT=ASCII".

References: Machine-readable:

VMS: HELP ABAQUS

Contact: Pete Matula, (301) 227-1936 Mike Brown, (301) 227-1706 С A modern, block-structured programming language designed for portability, system programming, and general-purpose applications. It is derived from Algo1-60. Execution: Cray COS: CPP, 'inputfile'. ^-- C Pre-Processor cc. <-- C compiler SEGLDR,CMD='LIB=\$CLIB;STACK'. DEC VAX/VMS: \$ CC "C - A Reference Manual", Hardison and Steele References: Hardware manufacturers' reference manuals Machine-readable: VMS: HELP CC

CMS

(Code Management System) A source code library maintenance system which can be used for any ASCII file. CMS tracks the history of the file (changes, reason for change, who made the change and when). It can merge modifications; and stores the current and historic versions of the file.

Execution: DEC VAX/VMS: \$ CMS CMS>command -or-\$ CMS command

References: Machine-readable:

VMS: HELP CMS -or-\$ CMS CMS>help

<-- internal help

Page 7-1-4

Rev0 90/09/30

DataTrieve (DTR)

VAX DataTrieve is a data management system which runs on VMS. It is a tool for defining, storing, updating, and displaying data. The data may reside either in a relational database created through DTR or an existing RMS file. It provides interactive and program-callable access to data, a report writing facility, a graphics capablility, screen formatting support using FMS (Forms Management System), and distributed access on a network connected by DECnet.

Execution: DEC VAX/VMS: \$ DTR32

Remarks: Data formats, procedures, and other data structures are stored in the Common Data Dictionary (CDD).

Users wishing to use DTR must have a valid CDD path established for them by User Services.

References: Machine-readable:

VMS: HELP DATATRIEVE -or-\$ dtr32 DTR>help <-- internal help

Contact: User Services

DISSPLA (Display Integrated Software System and Plotting Language) A library of Fortran subroutines which facilitate plotting. It does not rely upon features particular to any type of graphic device.

Execution: Cray COS: (version 10.0)

ACCESS, DN=DISSPLA, OWN=PUBLIC. SEGLDR, CMD='LIB=DISSPLA'

To dispose the meta file DISPLOT for post-processing on the VAX:

DISPOSE, DN=DISPLOT, DF=BB, TEXT='DISPLOT.DAT'.

DEC VAX/VMS: (version 10.5)

\$ FORTRAN yourfile
\$ DISLINK yourfile
Other libraries (Y or N) <as you
need>

To post-process files created by "CALL COMPRS":

\$ RUN VSYS:DISPOP

Remarks: Cray post-processing must be done on the VAX.

References: DISSPLA User's Manual

Machine-readable:

VMS: HELP DISSPLA

Contact: User Services

Page 7-1-6

DTLIB A library of subprograms written or supported by the CCF. The contents of DTLIB (formerly call NSRDC) is different on each machine, but generally includes routines in the areas of: . character manipulation

- . sorting
- . date/time manipulation
- . debugging aids
- . extraction of job information
- . some of the Fortran 8x intrinsics

Usage: Cray COS: ACCESS, DN=DTLIB, OWN=PUBLIC. DEC VAX/VMS: \$ LINK <obj>, DTLIB/LIBrary CDC 860 NOS: ATTACH, DTLIB/UN=NSYS. References: Machine-readable: Cray: on VAX, "HELP @DTLIB" VAX: HELP @DTLIB

DYNA3D DYNA3D is an explicit three-dimensional finite element code for analyzing the large deformation dynamic response of inelastic solids and structures. A contact-impact algorithm permits gaps and sliding along material interfaces with friction. Using a specialization of this algorithm, such interfaces can be rigidly tied to admit variable zoning without the need of transition regions. Spatial discretization is achieved by the use of 8-node solid elements, 2-node beam elements, 4-node shell elements, 8-node solid shell elements, and rigid bodies. The equations-ofmotion are integrated in time by the central difference method. The 1989 version of DYNA3D contains thirty material models and ten equations of state to cover a wide range of material behavior.

Execution: Cray COS:Use VMS

DEC VAX/VMS: @VSYS:DYNA3D

Remarks: This DYNA3D procedure creates a Cray batch job from user responses to pertinent questions. There is an option to have the Cray job submitted by the procedure. In the Cray job, the binary plot files are restructured by program CVBIN so that they may be read by the TAURUS graphics post-processor on the VAXcluster.

References:

Cray:

Machine-readable:

VMS: HELP DYNA3D

| GPSS | General Purpo simulation pa | ose Simulation System (GPSS) is a generalized ackage. |
|------|--------------------------------|--|
| | Execution: | DEC VAX/VMS: \$ GPSS qualifiers parameters |
| | | CDC 860 NOS: ATTACH,GPSS/UN=APPLLIB. GPSS,parameters. ^ use FX for fixed format |
| | Remarks: | The VMS and CDC versions are different. |
| | References: | The IBM document. |
| | | General Purpose Simulation System Reference Manual, Simulation Software Ltd. (VAX/VMS version) |
| | | Machine-readable: |

VMS: HELP GPSS

IMSL (proprietary) The International Mathematical and Statistical Libraries package (edition 10) contains 948 subroutines in the following areas:

. 426 general applied mathematics routines

- . 351 statistics routines
- . 172 special functions

IMSL 10 was a major revision.

Major enhancements were made in many areas of numerical math. Most statistical analysis subprograms can print results, handle missing values, and implement advances in algorithms. There is no ERROR parameter in the argument list and no need for you to dimension work arrays. Workspace is allocated out of a common area. Informative messages are printed when errors occur. Matrices no longer require packing into onedimensional arrays. Some user-supplied external subprograms must now be functions.

CHARACTER variables are used in the routines and in the many intermediary routines not explicitly called by the application.

| Usage: | Cray COS: | ACCESS,DN=IMSL,OWN=PUBLIC. SEGLDR,,CMD='LIB=IMSL', |
|--------|--------------|---|
| | DEC VAX/VMS: | add "IMSL/LIBrary to the LINK statement |
| | CDC 860 NOS: | may be used only by FTN5 programs and is in two permanent files: |
| | | IMSLM - the Math routines IMSLSS - the Special function and Statistics routines |
| | | If both the mathematics and statistics packages are needed, you must use the following search order: |
| | | ATTACH, IMSLM, IMSLSS/UN=NSYS. LIBRARY, IMSLSS, IMSL. -or- LDSET, LIB=IMSLSS/IMSLM. |

.

| References: | The IMSL documentation is in three sections: |
|-------------|--|
| | MATH/Library V1.0 - general applied mathematics |
| | STAT/Library V1.0 - statistics |
| | SFUN/Library V2.0 - special functions |
| | Also, |
| | Update Guide - describes the differences with the previous version |
| | Machine-readable: |
| | DEC VAX/VMS: HELP IMSL |
| Contact: | User Services |

Page 7-1-11

INGRES A relational database management system warketed by Ingres Corporation. Transactions against the database are done through SQL (an ANSI standard query language) or through forms-based utilities accessed by name or through INGMENU, a user-friendly, forms-based interface to the INGRES utilities.

> Execution: DEC VAX/VMS: \$ SETINGRES ^-- once to define INGRES symbols \$ INGMENU <data_base>

\$ name <-- a specific utility</pre>

Remarks: You must be an authorized INGRES user before you may access any of the INGRES utilities, including INGMENU. Call User Services to register.

References: Machine-readable:

VMS: HELP INGRES

KERMIT File transfer system to/from microcomputers.

Execution: DEC VAX/VMS: KERMIT

CDC 860 NOS: GET, KERMIT/UN=NSYS. KERMIT.

Remarks: To use Kermit on the VAX or CDC CYBER, you must have Kermit on your PC (it might be a subset of PROCOMM).

> VAX files to be transferred should have carriage return carriage control. Files with Fortran carriage control or with Print control will not transfer properly.

References: Machine-readable:

VMS: \$ kermit Kermit-32> help <-- internal help

NOS: BEGIN, HELP,, KERMIT, outfyl. ^-- a 7-page document

Usage:

Page 7-1-13

LINPACK A package of 40 subroutines obtained from Argonne Laboratories. Lab. These subroutines analyze and solve classes of systems of simultaneous linear algebraic equations. Routines are included for:

- . general, banded, symmetric indefinite, symmetric positive definite, triangular, tridiagonal square, and Hermetian matrices
- . orthogonal-triangular and single value decompositions of rectangular matrices
- . least square problems
- . basic linear algebra problems

There are four versions:

| precision | prefix | VAXcluster | Cray | NOS |
|------------|--------|------------|------|-----|
| | | | | |
| single | S | | x | x |
| double | D | x | | |
| complex | С | | x | |
| complex*16 | Z | x | | |

Cray COS: part of SCILIB DEC VAX/VMS: LINK <obj>, VSYS:LINPACK/LIBrary CDC 860 NOS: GET,LINPACK/UN=NSYS. LDSET,LIB=LINPACK. -or-

LIBRARY, LINPACK.

References: "LINPACK Users' Guide", J. J. Dongara, J. R. Bunch, C. D. Moler, G. W. Stewart, SIAM, 1979.

Machine-readable documentation may be listed using:

DEC VAX/VMS: "HELP LINPACK" "HELP LINPACK x<routine>" where x<routine> is "x" followed by the single precision name

MMS (Module Management System) used to automate the assembly of software. MMS reads a system file and determines what has changed since the last "system build" and reassembles.

Execution: DEC VAX/VMS:

Remarks: Eliminates recompiling if the program has not changed since the system was last built.

References: Machine-readable:

VMS: HELP MMS

Page 7-1-15

NASTRAN A general-purpose finite element structural analysis program capable of performing a wide range of analysis on models of complex structures, including static stress analysis, natural frequency analysis, buckling analylsis, frequency response analysis, and transient response analysis.

> Execution: Cray COS: ACCESS, DN=NASTRAN, ID=RPK, OWN=PUBLIC. CALL, DN=NASTRAN, CNS. NASTRAN, I=mydata. ^-- simple execution (MYDATA must be ACCESSed prior to this)

References: DTNSRDC/CMLD-81-05: NASTRAN Theory and Application Course Supplement

Machine-readable:

VMS: HELP NASTRAN <-- for Cray version

Contact: Tony Quezon, (301) 227-1645

. . .

PASCAL A modern programming language designed for general-purpose applications. It is derived from Algol-60.

Execution: Cray COS: PASCAL DEC VAX/VMS: \$ PASCAL CDC 860 NOS: PASCAL. References: "Pascal - User Manual and Report", Jensen and Wirth Hardware manufacturers' reference manuals Machine-readable:

VMS: HELP PASCAL

Page 7-1-17

PCA (Performance and Coverage Analyzer) Pinpoints performance problems; analyzes programs written in several languages; reports on performance characteristics; can plot a program's use of resources using histograms or tables.

Execution: DEC VAX/VMS: \$ PCA

References: Machine-readable:

VMS: HELP PCA

See also: Cray COS: SPY

SPICE A general-purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analyses. Circuits may contain resistors, capacitors, inductors, mutual inductors, independent voltage and current sources, four types of dependent sources, transmission lines, the four most common semiconductor devices: diodes, BJT's, JFET's, and MOSFET's, and a Josephson Junctions model.

> Execution: Cray COS: ACCESS,PDN=SPICE,OWN=PUBLIC. SPICE. /EOF <SPICE data> References: SPICE 2G.2.5 (Program Reference), E. Cohen, University of California (420 pages) SPICE Version 2G User's Guide, 8 Nov 1982

SPICE Version 2G User's Guide, 8 Nov 1982 (73 pages)

Machine-readable:

VMS: . HELP SPICE . VSYS:SPICE.DOC (the User's Guide)

***** Appendix A *****

*** ASCII Character Set ***

| char | ASCII (hex) | EBCDIC (hex) | Display (octal) | char | ASCII (hex) | EBCDIC (hex) | Display (octal) |
|------------|----------------|-----------------|--------------------|----------------|----------------|-----------------|--------------------|
| NUL | 00 | 00 | | (((| 28 | 4D | 51 |
| SOH | 01 | 01 | |))) | 29 | 5D | 52 |
| STX | 02 | 02 | | *** | 27 2A | 50 50 | 47 |
| ETX | 03 | 03 | | +++ | 2B | 4E | 45 |
| EOT | 04 | 37 | | | 20 20 | 6B | 56 |
| ENQ | 05 | 2D | | • • • • | 20 2D | 60 | 46 |
| ACK | 06 | 2E | | | 20 2E | 4B | 57 |
| BEL | 07 | 2F | | | 2F | 61 | 50 |
| | | | | | | | |
| BS | 08 | 16 | | 000 | 30 | FO | 33 |
| HT | 09 | 05 | | 111 | 31 | F1 | 34 |
| LF | 0A | 25 | | 222 | 32 | F2 | 35 |
| VT | OB | OB | | 333 | 33 | F3 | 36 |
| FF | 00 | 00 | | 444 | 34 | F4 | 37 |
| CR | 0D | 0D | | 555 | 35 | F5 | 40 |
| SO | 0E | 0E | | 666 | 36 | F6 | 41 |
| SI | OF | OF | | 777 | 37 | F7 | 42 |
| DLE | 10 | 10 | | 888 | 38 | F8 | 43 |
| DC1 | 11 | 11 | | 999 | 39 | F9 | 44 |
| DC2 | 12 | 12 | | ::: | 3A | 7A | 63 |
| DC3 | 13 | 13 | | • • • • • • | 3B | 5E | 77 |
| DC4 | 14 | 30 | | <<< | 3C | 4C | 72 |
| NAK | 15 | 3D | | | 3D | 7E | 54 |
| SYN | 16 | 32 | | >>> | 3E | 6E | 73 |
| ETB | 17 | 26 | | ??? | 3F | 6F | 71 |
| CAN | 18 | 18 | | 666 | 40 | 7C | 74 |
| EM | 19 | 19 | | AAA | 41 | C1 | 01 |
| SUB | 1A | 3F | | BBB | 42 | C2 | 02 |
| ESC | 1B | 27 | | CCC | 43 | С3 | 03 |
| FS | 1C | 1C | | DDD | 44 | C4 | 04 |
| GS | 1D | 1D | | EEE | 45 | C5 | 05 |
| RS | 1E | 1E | | FFF | 46 | C6 | 06 |
| US | 1F | 1F | | GGG | 47 | С7 | 07 |
| space | 20 | 40 | 55 | ННН | 48 | C8 | 10 |
| 111 | 21 | 4F | 66 | III | 49 | С9 | 11 |
| ** ** ** | 22 | 7F | 64 | 111 | 4A | D1 | 12 |
| ### | 23 | 7B | 60 | KKK | 4B | D2 | 13 |
| \$\$\$ | 24 | 5B | 53 | LLL | 4C | D3 | 14 |
| %%% | 25 | 6C | | MMM | 4D | D4 | 15 |
| હુજુજુ | 26 | 50 | 67 | NNN | 4E | D5 | 16 |
| | 27 | 7D | 70 | 000 | 4F | D6 | 17 |

1 î 1

| char | ASCII (hex) | EBCDIC (hex) | Display (octal) | char | ASCII (hex) | EBCDIC (hex) | Display (octal) |
|---------|----------------|-----------------|--------------------|------------|----------------|-----------------|--------------------|
| PPP | 50 | D7 | 20 | hhh | 68 | 88 | |
| QQQ | 51 | D8 | 21 | iii | 69 | 89 | |
| RRR | 52 | D9 | 22 | jjj | 6A | 91 | |
| SSS | 53 | E2 | 23 | kkk | 6B | 92 | |
| TTT | 54 | E3 | 24 | 111 | 6C | 93 | |
| ບບບ | 55 | E4 | 25 | mmm | 6D | 94 | |
| VVV | 56 | E5 | 26 | nnn | 6E | 95 | |
| WWW | 57 | E6 | 27 | 000 | 6F | 96 | |
| XXX | 58 | Е7 | 30 | PPP | 70 | 97 | |
| YYY | 59 | E8 | 31 | 999 999 | 70 | 98 | |
| ZZZ | 59 5A | E9 | 32 | 444 rrr | 72 | 99 | |
| [[[| 5B | 4A | JZ | SSS | 73 | A2 | |
| | 50 50 | EO | 75 | ttt | 74 | A3 | |
| ììì | 5D | 5A | 75 | uuu | 75 | A3 A4 | |
| 7 Y Y Y | 5E | 5F | 76 | vvv | 76 | A5 | |
| | 5E 5F | 6D | 65 | www | 77 | A6 | |
| grave | 60 | 79 | · | xxx | 78 | А7 | |
| aaa | 61 | 81 | | ууу | 79 | A8 | |
| bbb | 62 | 82 | | 22Z | 7A | A9 | |
| ccc | 63 | 83 | | [[[| 7B | C0 | 61 |
| ddd | 64 | 84 | | | 70 70 | 6A | U1 |
| eee | 65 | 85 | | }}} | 70 7D | D0 | 62 |
| fff | 66 | 86 | | ~~~ | | Al | 52 |
| ggg | 67 | 87 | | DEL | 7E 7F | 07 | |

,

.

| | | *** C | DC NOS | Characte | r Set | *** | |
|-----------------|----------------|--------------|----------------------------|--------------------|------------------------|----------|----------------|
| Display Code | char- acter | punch 026 | punch 029 if diff | 7-track ext BCD | 9-tı ASCII (note | EBCDIC | note/name |
| | | | | | | | |
| 00 | ::: | 2-8 | | | 25 | 6C | colon (1,2) |
| 01 | AAA | 12-1 | | 61 | 41 | C1 | |
| 02 | BBB | 12-2 | | 62 | 42 | C2 | |
| 03 | CCC | 12-3 | | 63 | 43 | C3 | |
| 04 | DDD | 12-4 | | 64 | 44 | C4 | |
| 05 06 | EEE | 12-5 12-6 | | 65 | 45 | C5 | |
| 08 | FFF GGG | 12-6 12-7 | | 66 67 | 46 47 | C6 C7 | |
| 10 | ННН | 12-8 | | 70 | 47 | C8 | |
| 10 | III | 12-9 | | 70 | 48 49 | C8 C9 | |
| 12 | JJJ | 11-1 | | 41 | 49 4A | D1 | |
| 13 | KKK | 11-2 | | 42 | 4B | D2 | |
| 14 | LLL | 11-3 | | 43 | 4C | D3 | |
| 15 | MMM | 11-4 | | 44 | 4D | D4 | |
| 16 | NNN | 11-5 | | 45 | 4E | D5 | |
| 17 | 000 | 11-6 | | 46 | 4F | D6 | |
| 20 | PPP | 11-7 | | 47 | 50 | D7 | |
| 21 | QQQ | 11-8 | | 50 | 51 | D8 | |
| 22 | RRR | 11-9 | | 51 | 52 | D9 | |
| 23 | SSS | 0-2 | | 22 | 53 | E2 | |
| 24 | TTT | 0-3 | | 23 | 54 | E3 | |
| 25 | ບບບ | 0-4 | | 24 | 55 | E4. | |
| 26 | VVV | 0-5 | | 25 | 56 | E5 | |
| 27 | WWW | 0-6 | | 26 | 57 | E6 | |
| 30 | XXX | 0-7 | | 27 | 58 | E7 | |
| 31 | YYY | 0-8 | | 30 | 59 | E8 | |
| 32 33 | ZZZ | 0-9 | | 31 | 5A | E9 | (00) |
| 33 34 | 000 111 | 0 1 | | 12 01 | 30 31 | F0 F1 | (sometimes 00) |
| 34 | 222 | 2 | | 02 | 32 | F1 F2 | |
| 36 | 333 | 3 | | 02 | 33 | F2 F3 | |
| 37 | 444 | 4 | | 04 | 34 | F4 | |
| 40 | 555 | 5 | | 05 | 35 | F5 | |
| 41 | 666 | 6 | | 06 | 36 | F6 | |
| 42 | 777 | 7 | | 07 | 37 | F7 | |
| 43 | 888 | 8 | | 10 | 38 | F8 | |
| 44 | 999 | 9 | | 11 | 39 | F9 | |
| 45 | +++ | 12 | 12-6-8 | 60 | 2B | 4E | plus |
| 46 | | 11 | | 40 | 2D | 60 | minus |
| 47 | *** | 11-4-8 | | 54 | 2A | 5C | asterisk |
| 50 | 111 | 0-1 | | 21 | 2F | 61 | slash |
| 51 | (((| 0-4-8 | 12-5-8 | 34 | 28 | 4D | left paren |
| 52 |))) | 12-4-8 | 11-5-8 | 74 | 29 | 5D | right paren |
| 53 | \$\$\$ | 11-3-8 | | 53 | 24 | 5B | dollar |
| 54 | *** | 3-8 | 6-8 | 13 | 3D | 7E | equal |
| 55 | | | | 20 | 20 | 40 | blank |
| 56 | , , , | 0-3-8 | | 33 | 2C | 6B | comma |
| 57 | ••• | 12-3-8 | | 73 | 2E | 4B | period |

Page A-4

| Display Code | char- acter | punch 026 | punch 029 if diff | 7-track ext BCD | 9-tr ASCII (note | EBCDIC | note/name |
|-----------------|----------------|--------------|----------------------------|--------------------|------------------------|--------|---------------|
| 60 | <i>###</i> | 0-6-8 | 3-8 | 36 | 23 | 7B | pound |
| 61 | <u>ו</u> ווו | 7-8 | 12-2-8 | 17 | 5B | 4A | left bracket |
| 62 |]]] | 0-2-8 | 11-2-8 | 32 | 5D | 5A | right bracket |
| 63 | % % % | 2-8 | | | 25 | 6C | percent (1,2) |
| 64 | ***** | 4-8 | 7-8 | 14 | 22 | 7F | quote |
| 65 | | 0-5-8 | | 35 | 5F | 6D | underline |
| 66 | 111 | 11-2-8 | 12-7-8 | 52 | 21 | 4F | exclam (3) |
| 66 | 111 | 11-0 | | 52 | 21 | 4F | exclam (3) |
| 67 | 666 | 0-7-8 | 12 | 37 | 26 | 50 | ampersand |
| 70 | | 11-5-8 | 5-8 | 55 | 27 | 7D | apostrophe |
| 71 | ??? | 11-6-8 | 0-7-8 | 56 | 3F | 6F | question |
| 72 | <<< | 12-2-8 | 12-4-8 | 72 | 3 C | 4C | less than (3) |
| 72 | <<< | 12-0 | | 72 | 3C | 4C | less than (3) |
| 73 | >>> | 11-7-8 | 0-6-8 | 57 | 3E | 6E | greater than |
| 74 | 999 | 5-8 | 4-8 | 15 | 40 | 7C | at |
| 75 | /// | 12-5-8 | 0-2-8 | 75 | 5C | EO | reverse slant |
| 76 | ~ ^ ^ | 12-6-8 | 11-7-8 | 76 | 5E | 5F | caret |
| 77 | ;;; | 12-7-8 | 11-6-8 | 77 | 3B | 5E | semicolon (4) |
| 55 | | 6-8 | 0-4-8 | | 2 0 | 40 | blank (5) |

Notes:

- In the 63-character set (NOS/BE), Display Code 00 has no character, and 63 is the colon (:). In the 64-character set (NOS), 00 is the colon (:), and 63 is the percent (%).
- (2) On 7-track tape, this becomes zero (display 33).
- (3) Alternate punches.
- (4) Avoid a whole word of semicolons, which is a negative zero and is treated as an end-of-record.
- (5) On some terminals, this is transmitted as a binary zero.
 For these terminals, avoid putting this punch in columns 9-10, 19-20, ..., 79-80, as each will be interpreted as a zero-byte terminator.
- (6) When ASCII and EBCDIC tapes are read and converted to Display Code, lower case letters are folded into upper case. A number of other codes are also folded.

Page B-1

***** Appendix B *****

*** Cray UNICOS Commands ***

This appendix is reserved for a description of Cray UNICOS commands. It will be expanded when UNICOS is available at DTRC.

. . .

Page C-1

***** ***** Appendix C

*** *** Cray COS JCL Commands

Cray COS JCL commands have the following general syntax:

verb sepl paraml sep2 param2 ... sepn paramn term comments

verb is the name of the routine to be executed. It consists of an alphabetic character (A-Z, a-z, \$, %, \emptyset) followed by 0-6 alphanumeric characters for system, local dataset name and system dataset name verbs; or 1-8 alphanumeric characters for library-defined verbs.

sepi are separators and include: - VERB, parameter. (

- VERB(parameter).
- VERB, parameter. <-- use period if comma) - VERB(parameter) <-- use right paren if left paren
 - VERB(parameter, parameter)
- = - VERB(keyword=value)
- VERB(keyword=value1:value2) :
- ۸ - VERB(...parameters...^ <-- statement continued <-- on another line parameters)
- '...' VERB(keyword='string')
- (...) VERB(keyword=(value:value))
- parami are parameters, which may be positional or keyword. Positional parameters have one of the following formats: value valuel:value2:...:valuen Keyword parameters have one of the following formats: keyword
 - keyword=value keyword=valuel:value2:...:valuen
- term is the statement terminator. It is either a period VERB. VERB, parameters. or a right parenthesis VERB(parameters)

comments follow the terminator.

ホホホ *** Strings

The following string representations are used in this appendix:

aa...a 1 or more alphabetic characters

axx...x 1 or more alphanumeric characters, the first alphabetic

xxx...x 1 or more alphanumeric characters

nnn...n 1 or more decimal (unless otherwise stated) digits

* * * *** Some Common Parameters

The following parameters are used in many JCL commands. If they have a different meaning or a special condition, it will be mentioned in the individual description.

Alternate User Access Mode (see PAM=) AM=mode

| Dū=dc | Disposition code IN - input queue of dest: MT - magnetic tape at job PR - print at job origin SC - scratch the dataset ST - stage to mainframe mainframe) | b origin mainframe |
|---------|---|--|
| DF=df | Dataset format (blocking; BB - binary blocked BD - binary deblocked CB - character blocked | <pre>front-end conversion) (no reblocking, no conversion; for graphics output) (same as TR) (front-end converts to ASCII (VAX) or Display Code or ASCII (NOS))</pre> |
| | CD - character deblocked TR - transparent | (front-end converts to ASCII (VAX) or Display Code (NOS)) (no deblocking; no conversion; for object modules, etc.) |
| | (default: CB) | • |
| DN=dn | Local dataset name | (axxxxxx, 7 maximum) |
| ED=ed | Edition number (1-4095) | |
| ERR | Suppress error termination | n messages |
| EXO≖exo | Execute option ON - execute-only (canno OFF - not execute-only | ot be read or PSDUMPed) |

Page C-3

I = i dnIDN=idn Input dataset name (normal default: \$IN) iD-uid Additional permanent dataset ID (axxxxxx, 8 maximum) L=1dn Name of dataset to contain the listing (default: \$OUT) M=mn Maintenance control word (axxxxxx, 8 maximum) MF=mf Front-end computer N1 - CDC CYBER 180/860A (NOS) V3 - DEC VAXcluster node DT3 (VMS) (default: front-end of job origin) MSG Suppress normal termination messages NA No abort. If omitted, an error causes the job step to abort. nbo=0ODN=odn Output dataset name (normal default: \$0UT) OWN=owner Owner of the permanent dataset (not needed for your own files) PAM=mode Public Access Mode E - execute only (same effect as EXO=ON) M - maintenance only N - no public access R - read only W - write only Example: PAM=R:W gives read and write permission (default: N) PDN=pdn Permanent dataset name (xxxxxxxxxxx, 15 maximum; enclosed in quotes "..." if other than A-Z, 0-9) R = rdRead control word (axxxxxx, 8 maximum) Text (up to 240 character) to be passed to the front-end, TEXT='text' enclosed in apostrophes ('...') TID=tid Destination terminal (default: terminal of job origin) UQ Unique access (required to delete or modify a dataset) (default: multiple access) Write control word W≖wt (axxxxxx, 8 maximum)

*** Permanent Dataset Utility Shorthand Notation ***

In the permanent dataset utility commands, wildcards may be used in the PDN, PDS, ID, US, and OWN parameters. An asterisk "*" represents any single character; a minus sign "-" represents zero or more characters. They are illustrated with PDN=.

PDN=ABC- all permanent dataset names starting with ABC

PDN=A*** all 4-character permanent dataset names starting with A

PDN=-A*- all permanent dataset names containing the letter A followed by one or more other characters

PDN=- all permanent dataset names

PDN=***- all permanent dataset names having 3 or more characters

*** A Word About Continuations ***

If a COS JCL statement is too long to fit on one line, it may be continued by breaking the statement after a parameter, ending the line with a caret (^), and continuing the statement on the next line(s). For example,

> FETCH, DN=prog3, SDN=myprog, ^ TEXT='GET, myprog.CTASK.'.

If a text field (quoted string) is too long, it may be split anywhere by adding an apostrophe (') to close the partial string and a caret to end the first line, and starting the next line with an apostrophe immediately followed by the rest of the string. For example,

> DISPOSE, DN=FT14, SDN=myout14, DC=ST, MF=N1, TEXT='USER, user, pw.'^ 'PURGE, myout14/NA. DEFINE.myout14.CTASK.'.

```
-or-
```

DISPOSE, DN=FT14, SDN=myout14, DC=ST, MF=N1, ^ TEXT='USER, user, pw.'^ 'PURGE, myout1'/NA.'^ 'DEFINE, myout14.'^ 'CTASK.'.

*** Summary of Cray JCL Commands *** The following are Cray JCL statements, except as indicated by: (DTRC - x)A command, procedure or program added at DTRC. Unless otherwise noted, these are accessed by: ACCESS, DN=x, OWN=PUBLIC. LIBRARY, DN=x:*. name,.... x is one of: PROCLIB, UTILITY. * Entire line is a comment. Syntax: * <comments> Similar commands: NOS: COMMENT; * VMS: 1 * This is a comment ---Examples: ACCESS Make a permanent dataset local. Syntax: ACCESS, DN=dn, PDN=pdn, ID=uid, ED=ed, R=rd, W=wt, M=mn, UQ,NA,ERR,MSG,OWN=owner. Parameters: PDN=pdn - If omitted, dn is used. R = rd- required to read the dataset if R= on SAVE W=wt - required for ADJUST if W= on SAVE - required to DELETE the dataset if M= M=mn on SAVE Similar commands: NOS: ATTACH; GET VMS: no local file concept ACCESS, DN=mylocal, PDN=mypermfile. Examples: ACCESS, DN=my1c1, PDN=yourpermfile, OWN=yourid. ACCESS, DN=myfile, UQ. DELETE, DN=myfile. ACCOUNT Validate the user. Follows the JOB statement or, is the first interactive statement.

Syntax: ACCOUNT, AC=ac, US=us, UPW=upw, NUPW=nupw.

Parameters: AC=ac - Account number (required) (10 digits or "S" + 9 digits) - Username (your 4-character User US=us Initials) - User password (required) UPW=upw NUPW=nupw - New user password Remarks: This must be the first statement of an interactive session. When entered via CDC NOS ICF, US= may be omitted because it is supplied automatically. When entered via the DEC VMS Cray Station, US= may be omitted if you entered it in upper case in response to the CRAY USERNAME: prompt. See also: JOB; page 1-2-2; Appendix D: CNEWPW Similar commands: NOS: CHARGE VMS: no user-specified charging ACCOUNT, AC=1234567890, US=xxxx, UPW=mypass. Examples: ACCOUNT, AC=1234567890, US=xxxx, UPW=mypass,NUPW=nupass. ACQUIRE Get a front-end dataset and make it local and permanent. ACQUIRE, DN=dn, PDN=pdn, AC=ac, ID=uid, ED=ed, RT=rt, Syntax: R=rd,W=wt,M=mn,UQ,MF=mf,TEXT='text',DF=df, OWN=ov, PAM=mode, ERR, MSG. Parameters: AC=ac - acquisition code IN - input dataset IT - intertask communication ST - dataset staged from front end (MF=)(default: ST) - (defaults: 1 ED=ed (permanent dataset does not exist) highest (permanent dataset exists))

> RT=rt - retention period (1-4095 days) (default: 45)

Remarks: If the dataset is permanent, ACQUIRE is the same as ACCESS. If not, then it is the same as FETCH, SAVE, ACCESS.

See also: FETCH, MSFETCH

Similar commands: NOS: ATTACH; GET VM3. HFT FETCH

Examples: ACQUIRE, DN=myfile, PDN=myfile, TEXT='myfile.FOR'.

ADJUST Redefine size of a permanent dataset.

Syntax: ADJUST, DN=dn, NA, ERR, MSG.

Permissions required: write; UQ on ACCESS

Remarks: ADJUST attempts to close the file. Subsequent references in the same job must reopen it and begin at BOD.

Similar commands: NOS: APPEND VMS: lengthened automatically; cannot be shortened

Examples: ADJUST, DN=myfile, NA.

ALTACN Validate an alternate account number for permanent files.

Syntax: ALTACN, AC=ac.

Parameters: ac - the alternate account number

Remarks: ALTACN validates the supplied Job Order Number.

To use the validated number, specify the ACN parameter on the SAVE or MODIFY command.

See also: MODIFY, SAVE

Similar commands: NOS: CHANGE

Examples: ALTACN,AC=1222233344. <-- define the number ... SAVE,DN=newfy1,ACN. <-- use the number ACCESS,DN=oldfy1,PDN=myoldfy1,UQ,.... MODIFY,DN=oldfy1,ACN. <-- change the number

ASSIGN Create a local dataset and assign dataset characteristics.

Syntax: ASSIGN, DN=dn, LM=1m, A=alias, BS=bs, U.

Parameters: LM= - maximum number of 512-word blocks in the dataset (maximum: 296000; default: 40000)

A= - alternate unit name

Page C-8

- BS= octal number of 512-word blocks for the I/O buffer (default: 10 octal)
- U unblocked dataset (default: blocked)
- Remarks: See COS Reference Manual for additional parameters.

At system initiation, ASSIGN,DN=\$IN,A=FT05. ASSIGN,DN=\$OUT,A=FT06. are performed automatically. You may reassign them at any time.

A Fortran OPEN will not recognize an ASSIGNed dataset.

Similar commands: NOS, VMS: ASSIGN

Examples: ASSIGN, DN=myinput, A=FT11. ^-- Fortran program reading from unit 11 will read file MYINPUT instead

AUDIT Report on permanent datasets.

Syntax: AUDIT, L=1dn, PDN=pdn, ID=uid, OWN=own, ACN=acn, LO=opt:...:opt, SZ=dsz, ACC=opt:opt, X=mm/dd/yy:'hh:mm:ss', TCR=mm/dd/yy:'hh:mm:ss', TLA=mm/dd/yy:'hh:mm:ss', TLM=mm/dd/yy:'hh:mm:ss'.

Parameters: L= - list dataset name (default: \$OUT) PDN= - name of permanent dataset(s) to be listed ID= - list datasets with this ID ID - list datasets with null ID OWN= - list datasets with this ownership value ACN= - list datasets with this account number

Page C-9

L0= - list options: S - short list (PDN, ID, ED; 2 per line) (may not be mixed with other options) A - access tracking (owner name, count, time of last and first accesses) B - backup info (backup volume name, etc.) L - long list (PDN, ID, ED, size (words), retention time, access count, track access flag, public access mode (PAM), creation, last access, last dump time, device name, preferred residency (PR), current residency (CR). (default in batch if no LO) N - notes list P - permit list (permitted owner name, access mode, access count, time of last access, time of permit creation) R - retired dataset list (same as L, but only retired datasets) T - text list X - extended long list (L plus number of blocks and words allocated) SZ= - list datasets >= this size (in words) ACC= - access option parameters AM - those datasets belonging to OWN that you are allowed to see PAM - those datasets belonging to OWN having any form of public access (R:W:M:E) X= - list datasets expired as of this date Х - list datasets expired as of now TCR= - list datasets created since this date TCR - not allowed TCR=mm/dd/yy is sufficient TLA= - list datasets not accessed since this date TLA - not allowed TLA=mm/dd/yy is sufficient TLM= - list datasets modified since this date TLM - not allowed TLM=mm/dd/yy is sufficient Similar commands: NOS: CATLIST VMS: DIRECTORY; MSSAUDIT

AUDIT,LO=S Examples: <-- short audit AUDIT,LO=P <-- audit showing who can and has accessed the datasets AUDIT, LO=L:P:N <-- long audit, permitted users and notes AUDIT,LO=L <-- long audit AUDIT, OWN=PUBLIC. <-- list public files AUDPL Audit an UPDATE program library (PL). Syntax: AUDPL, P=pdn, I=idn, L=1dn, M=mdn, *=m, /=c, DW=dw, LW=1w, JU=ju, DK=1ist, PM=1ist, LO=string, CM, NA, NR. Parameters: P I L * / NR - see UPDATE M= - Modifications dataset name (will contain reconstructed modification sets) (default: \$MODS) M=0 - No modifications output DW= - Data width (number of characters written per line to M dataset) (default: up to DW value on UPDATE stmt) LW= - Listing width (number of characters written per line to L dataset (Values: divided into pages: 80, 132; continuous listing: C80, C132) (default: 132, divided into pages) JU= - Justification N - identifier name left-justified; sequence number right-justified; no period between L - entire sequence field left-justified with period between (default: identified name right-justified; sequence number preceded by a period and left-justfied) DK=dk1:dk2:...:dkn(1) $DK='dk1, dk2, \ldots, dkj. dkk, \ldots, dkn'$ (2) - Decks for A, C, D, H, I options and PM parameter (For (1): up to 100 decks; for (2): separate single decks with commas, and ranges of decks with periods) (Maximum string length: 96 characters) (default: options apply to all decks) DK - By itself is invalid

BLOCK

PM=id1:id2:...:idn (1)(2) PM='id1, id2,..., idj.idk,..., idn' - Pulled modification sets (reconstructs modification sets for the listed identifiers for the decks listed in DK) (Syntax: same as for DK=) PM - By itself is invalid LO=string - Listing options for ldn Text listing (for DK= decks, if specified) A - active lines C - conditional text directives (subset of option D) D - compile dataset generation directives (subset of option A) H - modification histories I - inactive lines Summary options (for the entire PL) K - deck line counts L - identifier list M - modification set cross-reference N - identifier list in ASCII order 0 - overlapping modification set list P - short summary of the PL S - status of modification set X - common deck cross-reference CM - Copy modifications (reconstructed modification sets) to 1dn and mdn NR - Do not rewind modifications or binary identifier list datasets at start or end of AUDPL Similar commands: NOS: UPDATE VMS: CMS; LIBRARIAN; INCLUDE (in Fortran) Examples: AUDPL, P=myp1, LO=P. _ _ _ AUDPL, P=myp1, PM=mod2a:mod3c:example, LO=AIKLMNOPSX. COPYF, I=\$MODS. Convert an unblocked dataset to a blocked dataset. Svntax: BLOCK, DN=1dn, BLKSIZE=size. (1)BLOCK, I=idn, O=odn, BLKSIZE=size. (2) Parameters: DN= - the dataset to be replaced (using an intermediate dataset \$UNBLK) (1dn is rewound before and after)

Page C-12

- BLKSIZE= record length in 64-bit words (non-foreign datasets only) ((2) - not permitted if previously assigned as foreign; record length and type are taken from the input ASSIGN)
- I= the unblocked input dataset (idn is not rewound before the copy)
- 0= the blocked output dataset (if previously opened (ASSIGN), odn is not rewound before; otherwise, odn is created)
- Remarks: For foreign datasets, the record length and type are taken from the ASSIGN.

BLOCK is intended primarily for postprocessing datasets created by or for certain stations.

Examples: BLOCK, DN=myfile. ^-- Replace MYFILE with blocked copy of itself = = = = BLOCK, I=myunblk, O=myblk.

^-- Copy unblocked file MYUNBLK as blocked file MYBLK

BUILD Generate and maintain library datasets.

Syntax: BUILD, I=idn, L=ldn, OBL=odn, B=bdn, NBL=ndn, SORT, NODIR, REPLACE.

| Parameters: | I=idn I I=0 | - | Directive dataset name (default: \$IN) Same as I=\$IN No directives |
|-------------|-------------------|---|--|
| | L=ldn L | | List dataset name (default: \$OUT) Same as L=\$OUT |
| | OBL=odn | - | Old object library dataset name (default: \$OBL) |
| | _ | | Same as OFL=\$OBL No old binary library |
| | B=bdn | - | Dataset with new object modules (default: \$BLD) |
| | В В=0 | | Same as B=\$BLD No modules to be added |

NBL=ndn - Output new object library dataset name (default: \$NBL) NBL - Same as NBL=\$NBL NBL=0 - No output written SORT - modules are to be output in alphabetical order (default: written in the order they were first read) NODIR - Do not append the directory to the output dataset - use to retrieve relocatables (default: append the directory) REPLACE - Modules in the new library are replaced and in the same order as in the old library (default: new modules follow the unreplaced modules in the new library) Directives: see page 3-5-1. See also: Section 3-5 Similar commands: NOS: LIBEDIT VMS: LIBRARIAN BUILD, OBL=0, I=0. Examples: SAVE, DN=\$NBL, PDN=mylib. ^-- create a new library from \$BLD - - - - -ACCESS, DN=\$OBL, PDN=mylib. BUILD, I=0. SAVE, DN=\$NBL, PDN=mylib. ^-- add modules from \$BLD to existing library ~ = = = = ACCESS, DN=mylib1. ACCESS, DN=my1ib2. ACCESS, DN=mylib3. BUILD, I, OLB=0, B=0.SAVE, DN=\$NBL, PDN=mylib4. - Directive: FROM mylib1, mylib2, mylib3 ^-- merge several libraries - if duplicate module names, last found is retained (or use rename form, if desired) - - - - -ACCESS, DN=\$OBL, pdn=mylib. BUILD, B=0. SAVE, DN=\$NBL, PDN=mylib. - Directive: OMIT badpgm ^-- remove a module from a library

ACCESS,DN=xyz,PDN=mylib. BUILD,I,OBL=xyz,B=0,NBL=\$BLD,NODIR. - Directive: COPY myprog ^-- extract module for loading

CALL Read control statements from the first file of another dataset or transfer control to a procedure.

Syntax:CALL, DN=dn.<-- read from another file</th>CALL, DN=dn, CNS.<-- call a procedure</td>

Parameters: DN=dn - the dataset containing the statements or procedure (rewound before use)

CNS - Crack Next Statement - the first statement in "dn" is the procedure header; the statement following the CALL is treated as the invocation of the procedure

See also: Section 3-3

- Similar commands: NOS: BEGIN VMS: @name
- Examples: Without CNS:

If the first file of dataset XYZ contains:

ACCESS, DN=INFYL, PDN=MYFILE. ACCESS, DN=FILE1, PDN=MYDATA.

Then CALL,DN=XYZ. will access both datasets. This might be useful if you have several jobs using the same files, or if you have the same processing to be done by many jobs.

With CNS:

If the first file of dataset XYZ contains:

G,FILE,DATA. ACCESS,DN=INFYL,PDN=&FILE. ACCESS,DN=FILE1,PDN=&DATA.

Then CALL,DN=XYZ,CNS. *,MYFILE,MYDATA will access the datasets MYFILE and MYDATA. Note that PROC and ENDPROC statements and the procedure name (G) are not used.

"call by name" Execute a program by its local file name. Syntax: dn. dn, parameters. Parameters: depends upon the local file being executed Similar commands: NOS: LGO or an lfn VMS: \$ name :== \$ dir:name \$ name Examples: ACCESS, DN=myobj. myobj. CFT Compile a Fortran source program. Syntax CFT, I=idn, L=1dn, B=bdn, C=cdn, E=m, EDN=edn, OPT=option,MAXBLOCK=mb,INT=i1,ALLOC=alloc, ON=string,OFF=string,TRUNC=nn,AIDS=aids, CPU=cpu:hdw,UNROLL=r,LOOPMARK[=1mmsgs], DEBUG, SAVEALL, ANSI. Parameters: I= - Input dataset name (default: \$IN) L= - Listable output (default: \$0UT) L=0 - List only fatal errors B= - Binary load module dataset name (default: \$BLD) B≠0 - No binary load modules C= - pseudo-CAL output dataset name (default: no dataset) E= - Highest level of messages to be suppressed 1 - comment2 - note3 - caution4 - warning 5 - error (default: 3) EDN= - Alternate error listing dataset (default: no dataset)

| ON= | - Options to be enabled |
|--------|--------------------------------------|
| _ | (default: C E L P Q R S T U V) |
| OFF= | - Options to be disabled |
| | (default: A B D F G H I J N O W X Z) |
| | A - abort if errors |
| | B - list sequence number of code |
| | generation block |
| | C - list common block names and |
| | lengths |
| | D - list DO-loop table |
| | E - recognize compiler directives |
| | F - FLOWTRACE |
| | |
| | G - list generated code (use only |
| | if requested by User Services) |
| | H - list only first statement of |
| | each program unit |
| | I - generate label symbol table |
| | J - one-trip DO-loops |
| | L - recognize listing control |
| | statements |
| | M - ignored |
| | N - put null symbols in symbol table |
| | 0 - identify out-of-bound array |
| | references |
| | P - allows double precision |
| | Q - abort on 100 fatal errors |
| | R - round multiply results |
| | S - list source code |
| | T - list symbol table |
| | U - enable recognition of INTEGER*2 |
| | declarations |
| | |
| | V - vectorize inner DO-loops |
| | W - do not use |
| | X - include cross-reference |
| | Y - ignored |
| | Z - put DEBUG symbol table on \$BLD |
| TRUNC= | - number of bits to be truncated |
| | (default: 0; maximum: 47) |
| AIDS= | - number of vectorization inhibition |
| | messages |
| | LOOPNONE - no messages |
| | LOOPPART - maximum of 3 per inner |
| | loop; 100 per compilation |
| | • • • • |
| | LOOPALL - all messages |
| | (default: LOOPPART) |

OPT=- options (no more than one from each of the following groups: OPT=opt:opt:...): . constant increment integer optimization: NOZEROINC - no incrementation by zero value variables ZEROINC - incrementation by zero value variables (default: NOZEROINC) . optimization for 1-line DO-loop replacement with \$SCILIB call: SAFEDOREP - no replacement if DO-loop has potential dependencies or equivalenced variables FULLDOREP - alway replace NODOREP - never replace (default: SAFEDOREP) . move invariant code outside of DO-loop: INVMOV - enable NOINVMOV - disable (default: INVMOV) . instructions moving over a branch instruction: UNSAFEIF - enable SAFEIF - disable (default: SAFEIF) . bottom loading of scalar loops: BL - enable NOBL - disable (default: BL) . B and T register allocation: BTREG - allocate maximum of 24 scalars to T regs NOBTREG - allocate to memory (default: NOBTREG) . compilation of loops with specific ambiguous dependencies in vector and scalar versions: CVL - enable NOCVL - disable (default: enabled) . update scalar temporaries in DO-loops: KEEPTEMP ~ enable KILLTEMP - disable (default: enable)

Page C-18

MAXBLOCK= - number of words in a block of code to optimize or vectorize MAXBLOCK=1 - disable (default: 2310) INT= - integer lengths 64 - full 64-bit integers 24 - short 24-bit integers (default: 64) ALLOC= - static memory allocation STATIC - all memory STACK - read-only constants and DATA, SAVE and common block entities HEAP - deferred implementation (default: STATIC) CPU= - mainframe type and hardware characteristics for running generated code cpu type: CRAY-XMP - 1, 2 or 4 processors CRAY-X1 - single-processor CRAY-X2 - dual-processor CRAY-X4 - quad-processor (default: compiling machine) hardware characteristics: [NO] EMA - extended memory [NO]CI - compressed index [NO]GS - gather/scatter [NO]CIGS - compressed index gather/ scatter [NO] VPOP - vector popcount functional unit [NO] AVL - two vector logical functional units [NO] BDM - bidirectional memory UNROLL= - iteration count for unrolling inner DO-loops (range: 0 <= r <= 9) (default: 3) UNROLL=0 - turn off unrolling LOOPMARK= - draw DO-loop brackets in source listing MSGS - reasons for not vectorizing NOMSGS - no messages (default: NOMSGS) LOOPMARK - same as LOOPMARK=NOMSGS

90/09/30 Rev0

CFT77

DEBUG - put sequence number labels in Debug Symbol Table (forces ON=IW and MAXBLOCK=1) (default: debugging turned off) SAVEALL - allocate user variables to static storage; compiler-generated variables to B or T registers ANSI - flag non-ANSI usage Remarks: CFT compiles faster than CFT77, but executes more slowly. Production programs should be compiled using CFT77 and the resulting \$BLD file saved. See also: CFT77 Similar commands: NOS: FTN5 VMS: FORTRAN Examples: CFT. CFT, I=\$CPL. <-- from UPDATE CFT, LOOPMARK=MSGS. CFT, B=myobj. Compile a Fortran 77 source program. Syntax CFT77, I=idn, L=1dn, B=bdn, C=cdn, E=m, OPT=option, INTEGER=i1, ALLOC=alloc, ON=string, OFF=string, TRUNC=nn, CPU=cpu:hdw, DEBUG, LIST, STANDARD. Parameters: I L B C ALLOC TRUNC CPU DEBUG - same as CFT E= - same as CFT, except E=5 not allowed OPT= - at most one from each of the following groups (OPT=opt:opt): . optimization: - attempt full optimization FULL OFF - no optimization (fast compile) NOVECT - scalar optimization only (default: FULL) . constant increment integer optimization: NOZEROINC - no incrementation by zero-value variables ZEROINC - incrementation by zero-value variables

(default: NOZEROINC)

INTEGER= - integer length 64 - full 64-bit integers 46 - short 46-bit integers (default: 46) ON= - M - enable the loopmark option (same as CFT, LOOPMARK=MSGS) (default: P Q R) - (default: A F G H J M O S X Z) OFF= LIST - full compilation listing (sets ON=CGSX) DO NOT USE -- specify ON=CMSX instead STANDARD - flag non-standard Fortran 77 usage Remarks: CFT77 compiles much more slowly than CFT, but may execute faster. OPT=OFF does not vectorize and will, therefore, run slower. Production programs should be compiled using CFT77 and the resulting \$BLD file saved. See also: CFT Similar commands: NOS: FTN5 VMS: FORTRAN Examples: CFT77. CFT77, I=\$CPL. <-- from UPDATE CFT77, B=myobj, ON=M. CHARGES Report on job resources. CHARGES, L=1dn, MSG=msgopt, SR=options. Syntax: Parameters: MSG - controls the display of messages in the system log ON - output to \$LOG and \$SYSLOG OFF - output not displayed in log SR - control display of system resources CPU - CPU, I/O wait, and CPU wait times since start of job DS - permanent dataset statistics (synonym: DISK) FSU - FSS (buffer memory in IOS) usage GRU - generic resource usage - job name and user number JNU JSQ - job sequence number - job size (memory) statistics MM (synonym: MEMORY)

(synonym: MEMORY) MULTI - % of time spent in each CPU NBF - number of blocks received from/ queued to a front end (synonym: FE)

90/09/30 Rev0

RDM - job and permanent data usage and limits TASK - CPU, I/O wait, and CPU wait times broken down by task; and totals for job WT - time spent waiting in input queue (synonym: QWAIT) (default: all statistics) Remarks: CHARGES is invoked automatically at job end. Similar commands: NOS: ENQUIRE VMS: CHARGES; ^T Examples: CHARGES,SR=DS:MM:TASK COMPARE Compare two datasets. Syntax: COMPARE, A=adn, B=bdn, L=1dn, DF=df, ME=maxe, CP=cpn, CS=csn, {CW=cw|CW=cw1:cw2},ABORT=ac. Parameters: - input dataset names - error if adn=bdn A= B≖ L= - name of dataset for list of differences (default: \$OUT; may not be same as adn or bdn) DF= - input dataset format B - binary - datasets compared logically with difference listed in octal T - text - differences printed as text (default: T) ME= - maximum number of differences to be printed (default: 100) CP= - amount of context printed, that is, the number of records on either side of a difference to be printed (applies only to DF=T) (default: 0) CS= - amount of context to be scanned, that is, the number of records on either side of a discrepancy to be scanned (applies only to DF=T) (default: 0)

C₩= - compare width - either compare columns 1 through cw or columns cwl through cw2 (default: CW=1:133) ABORT= - abort the job step after ac or more differences have been found ABORT - same as ABORT=1 (default: 1) Similar commands: NOS: VERIFY; VFYLIB **VMS: DIFFERENCES** ACCESS, DN=one, PDN=myfile1. Examples: ACCESS, DN=two, PDN=myfile2. COMPARE, A=one, B=two, CS=5. COPYD Copy blocked datasets. Syntax: COPYD, I=idn, O=odn, S=m. - shift count (number of ASCII blanks to be Parameters: S=m inserted at the start of each line) (maximum: 132) - same as S=1 S (default: 0) COPYF; COPYNF; COPYR; COPYU See also: Similar commands: NOS: COPY; COPYSBF VMS: COPY Examples: COPYD, I=myprog, S=25. ^-- copy shifted file to \$OUT (source program centered on wide paper) COPYF Copy blocked files. COPYF, I=idn, O=odn, NF=nf, S=m. Syntax: Parameters: I O S - same as COPYD NF=nf - decimal number of files to copy - copy through EOD NF (default: 1) Remarks: After the copy, both datasets are positioned after the EOF for the last file copied. If BFI=OFF is specified on the ASSIGN, compressed blanks are expanded.

See also: COPYD; COPYNF; COPYR; COPYU

90/09/30 Rev0

Page C-23

Similar commands: NOS: COPY; COPYBF; COPYCF; COPYSBF VMS: COPY

Examples: COPYF, I=FT02. <-- print Fortran unit 2 on \$0UT.

COPYNF Copy files from one blocked dataset to another.

Syntax: COPYNF, I=idn, O=odn, NF=n.

Parameters: I 0 - same as COPYD

NF=n - deciaml number of files to copy. NF - copy through EOD (default: 1)

Remarks: After the copy, the input dataset is positioned after the EOF for the last file copied; the output dataset is after the EOF of the last record copied.

See also: COPYD; COPYF; COPYR; COPYU

Similar commands: NOS: COPYBF; COPYCF

Examples: COPYNF, I=mydata, O=files, NF=3. ^-- copy 3 files from dataset MYDATA to dataset FILES

COPYR Copy blocked records. Syntax: COPYR, I=idn, O=odn, NR=nr, S=m. Parameters: I O S - same as COPYD NR=nr - decimal number of records to copy NR - copy through EOF (default: 1) Remarks: After the copy, both datasets are positioned at the end of the last record copied. If BFI=OFF is specified on the ASSIGN, compressed blanks are expanded.

See also: COPYD; COPYF; COPYNF; COPYU

Examples: COPYR, I=myfile, O=recs, NR=342.

Rev0 90/09/30

COPYU Copy unblocked datasets. Syntax: COPYU, I=idn, O=odn, NS=ns. Parameters: I 0 - same as COPYD NS=ns - number of sectors to copy NS - copy through EOD (default: 1) See also: COPYD; COPYF; COPYNF; COPYR COPYU, I=unfy11, O=unfy12, NS. Examples: &DATA Defines the beginning of data within a procedure. &DATA, dn. Syntax: Parameters: dn - the name of the dataset to contain the data which follows this statement Remarks: All lines following an &DATA up to the next &DATA or ENDPROC are written to the specified dataset. Similar commands: NOS: .DATA VMS: OPEN, WRITE, CLOSE Examples: PROC, MYPROC. . . . ENDPROC. &DATA, IN1. 1.73, 2.6, 4 4.62, 9.7, 6 0,0,0 &DATA, IN2. 06Test01 12Ship 472-396X DDA Dynamic Dump Analyzer (selectively examine the contents of a program memory dump). Syntax: DDA, I=idn, S=sdn, L=odn, DUMP=ddn, LOG=1dn, ECHO=edn. Parameters: I= - the directives to be executed (default: \$IN) S≠ - symbolic dataset name (default: \$DEBUG) L=

 - output listing (default: \$0UT) 90/09/30 Rev0

Page C-25

- DUMP= the dataset with the dump to be analyzed (default: \$DUMP)
- LOG= the dataset to receive a copy of all input to and output from the debugger (default: \$DBLOG)
- ECHO= the dataset to receive a copy of all input to the debugger (default: \$DBECHO)
- Remarks: Like DEBUG, DDA interprets the contents of a program memory dump created during abort exit processing. Unlike DEBUG, you can give directives to dynamically select the information to display.
- Directives: See SR-0311, COS Symbolic Debugging Package Reference Manual (formerly SR-0112).

See also: DEBUG

Similar commands: NOS: FTN5,DB=PMD VMS: FORTRAN/DEBUG

Examples: See DUMPJOB

DEBUG Interpret a dump.

Syntax: DEBUG, S=sdn, L=1dn, DUMP=ddn, CALLS=n, TASKS, SYMS=sym[:sym], NOTSYMS=nsym[:nsym], MAXDIM=dim, BLOCKS=blk[:blk], NOTBLKS=nblk[:nblk], RPTBLKS, PAGES=np.

Parameters: S= - Debug symbolic tables (default: \$DEBUG)

> L= - Listable output (default: \$OUT)

DUMP= - Dump dataset name (default: \$DUMP)

CALLS= - Number of routine levels to display (default: 50)

TASKS - Trace back through all existing tasks (default: only through tasks running when dump taken)

DELETE

SYMS= - List of symbols to be displayed (Maximum: 20 symbols) (default: all symbols) NOTSYMS= - List of symbols to be skipped (Maximum: 20 symbols) (default: all symbols displayed) MAXDIM= - Maximum number of each dimension to be displayed (default: 20:5:2:1:1:1:1) BLOCKS= - List of common blocks to include (Maximum: 20 symbols) BLOCKS - Include all common blocks NOTBLKS= - List of common blocks to exclude (overrides BLOCKS) (Maximum: 20 symbols) NOTBLKS - Exclude all but subprogram block RPTBLKS - Repeat blocks (display with each subprogram (default: display once) PAGES= - Page limit (default: 70) Similar commands: NOS: FTN5, DB=PMD VMS: FORTRAN/DEBUG See DUMPJOB. Examples: Remove a permanent dataset. Syntax: DELETE, DN=dn, NA, ERR, MSG, PARTIAL. DELETE, PDN=pdn, ID=uid, OWN=owner, ED=ed, M=mn, NA, ERR, MSG. Parameters: PARTIAL - delete the contents of the file, but not the information about the file - edition number (1-4095)ED=ed unsigned - specific edition +n - delete n highest editions - keep n highest editions -n ALL - all editions (default: highest edition) Remarks: The first form is used if the permanent file has already been ACCESSed.

The second form does not ACCESS the file.

See also: Appendic C: CDELETE Similar commands: VMS: CREATE a new version, PURGE/KEEP=1; DELETE; PURGE Examples: ACCESS, myfile, UQ. DELETE, DN=myfile, PARTIAL. - - - - -DELETE, PDN=myfile, ALL. ----DELETE, PDN=A**. ^-- delete all datasets with 3-character names starting with "A" DISPOSE Stage a dataset to the front-end; release a local dataset; change disposition characteristics. Syntax: DISPOSE, DN=dn, SDN=sdn, DC=dc, MF=mf, SF=sf, ID=uid, TID=tid,R=rd,W=wt,M=mn,TEXT='text',DF=df, WAIT NOWAIT, DEFER, NRLS. Parameters: DN=dn - required SDN=sdn - staged dataset name (1-15 characters) (default: dn; required for CYBER 860)

- DC=dc to 860: DC=ST is required to VAX: DC=PR with TEXT='any' makes a file with Fortran carriage control; DC=ST (with TEXT='any') makes a file with carriage return carriage control
- SF=sf special forms (1-8 alphanumeric characters) (default: no special forms)
- DF=df TR or CB or BB (default: CB)
- WAIT wait or don't wait until dataset has NOWAIT been staged to the front-end (default: NOWAIT)
- DEFER disposition occurs at end-of-job or when the dataset is RELEASEd
- NRLS after disposition, the dataset remains local (use WAIT)

See also: MSSTORE

DS

Similar commands: NOS: ROUTE VMS: FICHE (DTRC); PRINT; XEROX (DTRC) DISPOSE, DN=out1, DC=PR. Examples: ^-- to VAX (assumed job origin) - - - - -DISPOSE, DN=out2, SDN=mymss, MF=N1, DC=ST, ^ TEXT='USER, user, pw. '^ 'PURGE, mymss/NA.'^ 'DEFINE, mymss.'^ 'CTASK.',WAIT. ^-- send to MSS _ _ _ _ _ _ DISPOSE, DN=out3, MF=V3, ^ TEXT='myvax.dat',WAIT. ^-- send to VAXcluster _ _ _ _ _ _ DISPOSE, DN=DISPLOT, DC=ST, DF=BB, TEXT='plot.dat', ^ WAIT. ^-- DISSPLA output file to VAX for post processing List local datasets. DS. Syntax: Remarks: The information displayed includes alis, size, position (e.g., EOF), last operation, and open status. Similar commands: NOS: ENQUIRE, F Examples: DS. DSDUMP Dump a dataset in octal or hexadecimal. DSDUMP, I=idn, O=odn, DF=df, IW=n, NW=n, IR=n, NR=n, Syntax: IF=n,NF=n,IS=n,NS=n,Z,DB=db,DSZ=sz. Parameters: I= - (synonym: DN=idn) - dataset to receive the dump 0= (default: \$OUT) DF= - dataset format B - blocked U - unblocked (default: B) IW= - decimal/octal number of the initial word for each record/sector (defaults: 0 (Z specified); 1 (Z omitted))

NW= - decimal/octal number of words to dump (default: 1) - through end of record/sector NW IR= - decimal/octal number of the initial record for each input file - only if DF=B (defaults: 0 (Z specified); 1 (Z omitted)) NR= - decimal/octal number of records per file to dump - only if DF=B (default: 1) NR - all records in each file IF= - decimal/octal number of the initial file in idn - only if DF=B (defaults: 0 (Z specified); 1 (Z omitted)) NF= - decimal/octal number of files to dump only if DF=B (default: 1) NF=0 - all files in the dataset IS= - decimal/octal number of the initial sector - only if DF=U (defaults: 0 (Z specified); 1 (Z omitted)) NS= - decimal/octal number of sectors to dump only if DF=U (default: 1) - the zero-base for the initial-value Ζ parameters (IW, IR, IF, IS) - each Ix is relative to 0; Ζ output refers to word, record, file, and sector numbers start at 0 DSDUMP,..., IW=4096. is same as DSDUMP,...,Z,IW=4095. no Z - each Ix is relative to 1 (does not affect Nx parameters) DB= - numeric base for displaying the data words OCTAL or 0 - octal HEX or H - hexadecimal DSZ= - size of data items to dump WORD or W - words (64 bits) PARCEL or P - parcels (16 bits) (default: WORD) Similar commands: NOS: TDUMP Examples: DSDUMP, I=myfile, NW=25, NR=5, DB=H. ^-- hexadecimal dump of first 25 words of first 5 records of MYFILE

DUMP Display job information previously captured by DUMPJOB. DUMP, I=idn, O=odn, FWA=fwa, LWA=1wa, JTA, NXP, V, DSP, Syntax: FORMAT=f,CENTER. Parameters: I= - dataset containing the memory image (default: \$DUMP) FWA= - first word address to dump (default: word 0 of Job Communication Block (JCB)) - last word address to dump LWA= (default: 200 of JCB) - the limit address LWA LWA=0 - no memory JTA - dump Job Table Area (default: no JTA dump) NXP - dump No Exchange Package, B, T, cluster, and semaphore registers (default: these are dumped; NXP overrides V if both specified) - dump vector registers V (default: do not dump vector registers) DSP - dump Logical File Tables (LFTs) and Dataset Parameter Tables (DSPs) (default: do not dump LFTs and DSPs) FORMAT= - format for dumping FWA through LWA D - data - decimal integer and ASCII G - data - floating-point or exponential and ASCII I - instr - CAL mnemonics and ASCII M - data - each 16-bit parcel displayed as 1 hex and 4 octal digits C - data - octal integer and ASCII P - data - 16-bit parcel X - data - hex integer and ASCII CENTER - dump 100 (octal) words on each side of P-register address in P format See DUMPJOB. Examples: DUMPJOB Capture job information in dataset \$DUMP for display by DUMP or DEBUG or DDA. Syntax: DUMPJOB. Examples: ... EXIT. DUMPJOB.

DUMP,.... -or- DEBUG, BLOCKS,.... -or- 1 DA,....

90/09/30 Rev0

i

ł

Page C-31

| ECHO | Control logfile messages. | | | |
|---------|---|---|--|--|
| | Syntax: | ECHO,ON=class1::classm,OFF=class1::classn | | |
| | Parameters: | <pre>ON= - list of classes whose messages are to be written to the log file ("ON" is the same as "ON=ALL") OFF= - list of classes whose messages are NOT to be written to the log file ("OFF" is the same as "OFF=ALL") classi - ABORT - job failure EXPINF - dataset statistics messages JCL - messages in user's JCL PDMERR - PDM errors PDMINF - PDM dataset information ALL - all classes</pre> | | |
| | Remarks: | The ECHO state after returning from a procedure call is the same as before the call, regardless of any changes made in the procedure. Within a procedure, the ECHO state is that of the caller, unless changed within the procedure. | | |
| | Similar commands: VMS: /LOG on some commands | | | |
| | Examples: | ECHO, OFF. | | |
| ELSE | See IF. | | | |
| ELSEIF | See IF. | | | |
| ENDIF | See IF. | | | |
| ENDLOOP | See LOOP. | | | |
| ENDPROC | See PROC. | | | |
| EXIT | On job abort, processing continues with the statement follow: the EXIT; if no abort, terminate job processing. | | | |
| | Syntax: EX | KIT. | | |
| | Similar comm | mands: NOS: EXIT VMS: ON condition | | |
| | DI | KIT. UMPJOB. UMP. | | |

Rev0 90/09/30

EXITIF See IF.

EXITLOOP See LOOP.

FETCH Get a front-end dataset and make it local.

Syntax: FETCH, DN=dn, SDN=sdn, AC=ac, TEXT='text', MF=mf, DF=df.

Parameters: DN= - local dataset name

SDN= - staged dataset name (front-end dataset name) (default: dn)

AC= - acquisition code (where the dataset is to be acquired) IN - input (job) dataset - use SUBMIT to run the job IT - intertask communication MT - magnetic tape at the front end ST - staged dataset from the front end (default: ST)

- MF= mainframe computer identifier N1 - MSS V3 - DT3 (default: front end of job origin)
- DF= dataset format (BB, BD, CB, CD, TR) (default: CB)
- Remarks: FETCH defaults to DF=CB, MSFETCH defaults to DF=TR.

See also: MSFETCH

Similar commands: VMS: HFT FETCH (get an MSS file, DTRC)

Examples: FETCH, DN=SOURCE, TEXT='PROG.FOR'. = = = = = FETCH, DN=FT11, DF=TR,^ TEXT='[ABCD.SUBD1]CRAYBIN.DAT'. ^-- binary data file from a VAX subdirectory of user ABCD = = = = FETCH, DN=SORC, SDN=mssname, MF=N1,^ TEXT='USER, name, pw.^ 'GET, mssname.CTASK.'. ^-- get an indirect MSS (860) file FLODUMP Dump flowtrace table of a program abort. Syntax: FLODUMP, L=1dn. Parameters: L= - dataset to contain the report (default: \$OUT) Examples: . . . EXIT. DUMPJOB. FLODUMP. FTREF Generate Fortran cross-reference. Svntax: FTREF, I=idn, L=ldn, CB=op, TREE=op, ROOT=root, END=end, LEVL=n, DIR=dir, NORDER, MULTI. Parameters: I= - input dataset containing the crossreference table listing and Fortran source program (ON=XS) CB=- global common block cross references PART - routines using a common block FULL - use of common block variables NONE - no output information (default: PART) TREE= - static calling tree PART - entry names, external calls, calling routines, common block names FULL - PART plus static calling tree NONE - no output information (default: PART) ROOT= - if TREE=FULL, this defines the name of the routine to be used as the root of the tree (default: the routine not called by any other routine; if more than one, the first alphabetically) END= - if TREE=FULL, this defines the name of the routine to be used as the end of any branch of a tree (default: complete trees are generated) LEVEL= - if TREE-rolLL, this is the maximum length of any branch (default: the entire program)

90/09/30

Rev0

DIR= - dataset containing processing directives (default: no directives)

Page C-33

Rev0 90/09/30

NORDER - list subprograms in input order (default: list in alphabetical order) MULTI - summarize multitasking subroutine usage Directives: The following may be in the DIR= dataset: ROOT - list of modules to be used as roots of separate trees $ROOT, md1, md2, \ldots, mdn.$ SUBSET - list of modules to be processed SUBSET, md1, md2,..., mdn. (default: all modules) CHKBLK - list of common blocks to be checked for locked variables CHKBLK, b1k1, b1k2,..., b1kn. CHKMOD - list of external calls to be checked for calling from a locked area CHKMOD, mod1, mod2, ..., modn. Similar commands: NOS: FTN5,LO= VMS: FORTRAN /CROSS REFERENCE

HOLD Specify that dataset release occurs with implicit HOLD.

Syntax: HOLD, GRN=grn.

Parameters: GRN=grn - generic resource name

Remarks: This prevents return of resources to the system and is useful when dataset assignment is done by applications over which the user has no control.

See also: NOHOLD

IF Begin a conditional block of code.

Syntax: IF (expression) <do if true> ELSEIF (expression) <do if true> ELSE. <do if all other tests fail> ENDIF. EXITIF. EXITIF. EXITIF (expression) <-- exit unconditionally EXITIF (expression) <-- exit if exp is true Parameters: exp - a valid JCL expression 90/09/30 Rev0

Page C-35

Literal strings, '...', in an IF/ELSEIF Remarks: expression are limited to 8 characters (one machine word). Similar commands: NOS: IF; IFE VMS: IF ACCESS, DN=MYPROG, NA. Examples: IF (PDMST.NE.1) UPDATE (Q=MYPROG) CFT(I=SCPL,ON=A)NOTE (DN=SLIN, TEXT='ABS=MYPROG') ^-- create input directive file for SEGLDR SEGLDR (I=SLIN) SAVE (DN=MYPROG, NA) EXITIF. EXIT. *. *. Error while generating MYPROG * EXIT. ENDIF. MYPROG. Same as above, but in a procedure, with SEGLDR directives in a data file in the procedure: PROC. DOMYPROG. <-- omit NOTE command • • • ENDPROC. &DATA, SLIN . . . ABS=MYPROG Control access to a job's I/O area (containing the DSP and I/O IOAREA buffers). IOAREA, { LOCK } UNLOCK } Syntax: Parameters: LOCK - the limit address is set to the base of the DSPs, denying direct access to the user's DSP and I/O buffers. When locked, system I/O routines can gain access. UNLOCK - the limit address is set to JCFL, allowing access to these areas. Examples: IOAREA, LOCK.

ITEMIZE Report statistics about a library dataset. Svntax: ITEMIZE, DN=dn, L=1dn, NREW, MF=n, T, BL, E, B, X. Parameters: DN= - (default: \$OBL) NREW - no rewind (default: rewind before and after) NF= - number of files to be listed (default: 1) NF - all files Т - truncate lines after 80 characters (if specified, E, B, X may not be used) BL - burstable listing (each heading is at top of a page (default: page eject only when current page is nearly full) E - list all entry points (binary library datasets only) В - E plus code and common block information (B overrides E) Х - B plus external information (X overrides B) Restrictions: . an UPDATE PL is recognized only if it is the only item in a dataset . standard COS blocked datasets only Similar commands: NOS: ITEMIZE VMS: LIBRARIAN Examples: ITEMIZE, DN=myreloc ITEMIZE, DC=mylib, X. JOB First statement of a job - gives job parameters. Syntax: JOB, JN=jn, MFL=f1, T=t1, OLM=olm, US=jcn. Parameters: JN=jn - job name (1-7 alphanumeric characters) MFL=fl - maximum field length (decimal) for the job - fl is rounded up to the nearest multiple of 512 words, or the amount needed to load CSP (Control Statement Processor) (default: 768000) MFL - the system maximum (3, 532, 800)

- - this exceeds the DTRC maximum!
- OLM=olm maximum size of \$OUT; olm is the number of 512-word blocks (each block holds about 45 lines) (default: 8192; maximum: 65536)
- US=jcn job class (1-7 alphanumeric characters) jcn is one of: NORMAL, DEFER, BUDGET, PZERO, SECURE Job is dropped to a lower class if it doesn't fit the requested job class. (default: NORMAL, if it fits) (see page 3-1-4 for the job class limits and SECURE restrictions)
- Remarks: The JOB statement may be continued.

See also: ACCOUNT

- Examples: JOB, JN=jobnamel. ACCOUNT,.... <rest of job>
- JOBCOST (DTRC UTILITY) Write a summary of the job cost and system usage to \$LOG.

Syntax: JOBCOST

Remarks: A subroutine version is available in DTLIB.

- Examples: ACCESS, DN=UTILITY, OWN=PUBLIC. LIBRARY, DN=UTILITY:*. JOBCOST. <-- the cost to this point in job < execute your program > JOBCOST. <-- the cost of running your program
- LIBRARY Specify the library dataset search order for control statement verbs.

Syntax: LIBRARY, DN=dn1:dn2:...:dn64, V.

- Parameters: DN[∞] up to 64 library names to be searched an asterisk means add the listed names to the current searchlist
 - V list the current library searchlist in the logfile

Rev0 90/09/30

Page C-38

```
Similar commands: NOS: LIBRARY; LDSET, LIB= (not subs)
                      LIBRARY, DN=THISLIB: YOURLIB.
         Examples:
                                            ^-- the searchlist contains
                                                2 libraries
                      LIBRARY, DN=THATLIB:*, V.
                                            ^-- the searchlist now has
                                                3 libraries and are
                                                listed in the logfile
                      LIBRARY,,V.
                                           <-- list the current
                                                searchlist in the logfile
LOOP
         Start of an iterative control statement block.
         Syntax:
                      LOOP.
                       . . .
                       EXITLOOP.
                       EXITLOOP(expression)
                       . . .
                       ENDLOOP.
         Parameters: exp - a valid JCL expression
         Similar commands: NOS: WHILE
         Examples:
                      Merge two datasets for 60 records:
                         SET, J1=0.
                         SET, J2=60.
                         LOOP.
                           EXITLOOP(J2.EQ.0)
                           IF(J1.EQ.0)
                             COPYR, I=DSIN1, O=OUTDS.
                             SET, J1=1.
                           ELSE.
                             COPYR, I=DSIN2, O=OUTDS.
                             SET, J1=0.
                           ENDIF.
                           SET, J2=J2-1.
                         ENDLOOP.
                         REWIND, DN=DSIN1:DSIN2:OUTDS.
MEMORY
         Request new field length.
         Syntax:
                       MEMORY.
                       MEMORY, FL=f1.
                       MEMORY, FL=f1, { USER | AUTO }.
         Parameters: FL=f1 - the decimal number of words of field
                               length; "FL" allocates the job maximum
                       USER - field length is retained until the next
                               request
                       AUTO - field length is reduced automatically at
                               the end of each job step
```

90/09/30 Rev0

Page C-39

Similar commands: NOS: MFL Examples: MEMORY, FL. USER. <-- get and hold the maximum field length <-- resume automatic mode MEMORY, AUTO. (FL reduces after next job step) MEMORY, FL=32978. <-- get and hold 32978 words (user mode) MEMORY, FL=32978, AUTO. ^-- get 32978 words for next job step only MODE Set/clear mode flags. Syntax: MODE, FI=option, BT=option, EMA=option, AVL=option, ORI=option. Parameters: option - ENABLE or DISABLE FΙ - floating-point error interrupts (default: ENABLE) BT - bidirectional memory transfers (default: ENABLE) EMA - extended memory addressing (default: DISABLE) AVL - second vector logical function unit (default: DISABLE) ORI - operand range error interrupt (default: ENABLE) Similar commands: NOS: MODE VMS: ON condition MODIFY Change a permanent dataset's characteristics. Syntax: MODIFY, DN=dn, PDN=pdn, ID=uid, ED=ed, RT=rt, R=rd, W=wt, M=mn, NA, ERR, MSG, EXO=exo, PAM=mode, ACN. Parameters: RT=rt - new retention period RT= - reset to default ACN - use the alternate account number Remarks: If the file has control words (M=, R=, W=), they must all be specified in the ACCESS. See also: ALTACN; NEWCHRG; SAVE; Appendix D: CNEWCHRG Similar commands: NOS: CHANGE VMS: SET PROTECTION Examples: ACCESS, DN=mylocal, PDN=myperm, UQ, MODIFY, DN=mylocal, PAM=R.

MSACCES (DTRC) Supply username and password to the Mass Storage System.

Syntax: MSACCES, US=us, MPW=mpw, AC=ac.

Parameters: us - user initials/username (default: the executing VAX user initials)

mpw - MSS password

ac - account/charge number (default: the executing VAX account number)

Remarks: MSACCES is required before using the MSx commands.

Similar commands: VMS: HFT ACCESS (DTRC)

Examples: MSACCES, MPW=mymsspw. MSAUDIT,.... -or- MSCHANG,.... -or- MSFETCH,.... -or- MSPASSW,.... -or- MSPURGE,.... -or- MSSTORE..... = = = = MSACCES, US=other, MPW=otherpw, AC=otherac. ^-- access the MSS as another user

MSAUDIT (DTRC) Sorted audit of Mass Storage files.

Syntax: MSAUDIT, MPW=mpw, L=1dn, LO=10, SHOWPW=showpw, UN=un.

Parameters: mpw - your MSS password

| 10 | - list options F - full audit (4 lines per file + |
|----|--|
| | cost per month and per day) |
| | S - short audit (length, filename, |
| | CT, M (permissions), number of uses, indirect/direct) |
| | I - intermediate audit (short plus number of uses, date created, date last accessed, number of streams (direct files), password (if requested), charge number (your files), cost per day) |
| | (default: LO=F) |

- showpw enter anything to include each file's
 password (your files only) in the
 output list (LO=S or I)
 (default: passwords are not included)
- un Username (User Initials) of the owner of the MSS files to be audited (default: your own files)

Remarks: MSACCES is required before using the MSx commands. MSAUDIT provides a sorted listing of your files on the Mass Storage System (4 lines per file) and two shorter forms (1 line per file). See also: AUDIT Similar commands: NOS: BEGIN, AUDIT; CATLIST VMS: DIRECTORY/FULL; MSSAUDIT Examples: MSAUDIT, mymsspw. ^-- 4 lines per file listing of your MSS files on \$OUT _____ MSAUDIT, mymsspw, L=audout, LO=I, SHOWPW=x. ^-- 1 line per file listing (including each file's password) written to local file AUDOUT MSAUDIT, mymsspw, L=hisout, LO=S, UN=other. ^-- short listing of MSS files of user OTHER in file HISOUT MSCHANG Change Mass Storage System file attributes. Syntax: MSCHANG, MDN=mdn, NMDN=nmdn, PW=pw, CT=ct, M=m, BR=br, PR=pr,NA=na,AC=ac,CP=cp. Parameters: mdn - MSS filename whose attributes are to be changed nmdn - new MSS filename pw - new password 0 - clear the password - file permit Category Type ct ct meaning _____ P or PR or PRIVATE private S or SPRIV semiprivate PU or PUBLIC public - alternate user permission mode for m semiprivate and public files meaning m _____ -----______ E (EXECUTE) you can execute; others can read or execute concurrently (READ) all can read or execute R concurrently

Rev0 90/09/30

| | RU | (READUP) | all can read or execute; one (other) user can rewrite the file | | | |
|--------------|--|-----------------------------|--|--|--|--|
| | RA | (READAP) | all can read or execut; one (other) user can | | | |
| | RM | (READMD) | lengthen the file all can read or execute; one (other) user can lengthen or rewrite the file | | | |
| | U | (UPDATE) | all can read or execute; you can rewrite the file | | | |
| | А | (APPEND) | all can read or execute; you can lengthen the file | | | |
| | м | (MODIFY) | all can read or execute; you can lengthen or rewrite the file | | | |
| | W | (WRITE) | you can read, execute, lengthen, rewrite, or shorten the file; others have no concurrent access | | | |
| | | kup requirem | | | | |
| | br | | meaning | | | |
| | CR off-station backup Y on-station backup | | | | | |
| | pr - preferred residence pr meaning | | | | | |
| | M N | - | storage - MSS ence | | | |
| | | - abort | on errors abort on errors | | | |
| | | alternate u ut the file? | sers obtain information (Y or N) | | | |
| | one | currently i | is to be replaced by the n effect the account number | | | |
| | - | uired; the d unchanged. | efaults for the others is to | | | |
| Remarks: | MSACCES is | required be | fore using the MSx commands. | | | |
| Similar comm | | CHANGE HFT CHANGE | | | | |

Examples: MSACCES, UN=myid, MPW=mymsspw. MSCHANG, MDN=myfile, NMDN=newname. ^-- change MSS file MYFILE to NEWNAME MSCHANG, MDN=myfile, NMDN=newname, NA=1. ^-- change MSS file MYFILE to NEWNAME (don't abort if MSS file NEWNAME already exists) MSCHANG, MDN=myfile, CT=PU. ^-- make MSS file MYFILE public MSCHANG, MDN=myfile, PW=mypw. ^-- put a password on MSS file MYFILE MSCHANG, MDN=myfile, BR=CR. ^-- make MSS file MYFILE a critical file with off-station backup MSCHANG, MDN=myfile, BR=Y. ^-- make MSS file MYFILE a noncritical file with on-station backup MSCHANG, MDN=myfile, CP=1. ^-- change the account number to the one in effect on the MSS MSFETCH (DTRC) Fetch a file from the Mass Storage System. Syntax: MSFETCH, DN=dn, MDN=mdn, DF=df, UN=un, PW=pw. Parameters: dn - the local dataset name mdn - the MSS dataset (file) name (default: MDN=dn) df - data format TR - transparent (no conversion) CB - character blocked (convert from CDC display code) (default: DF=TR) un - Username (User Initials) of the owner of the MSS file (omit for your own files) pw - optional MSS file password Remarks: MSACCES is required before using the MSx commands. MSFETCH defaults to DF=TR, FETCH defaults to DF=CB.

See also:

ACQUIRE, FETCH

Rev0 90/09/30

Similar commands: NOS: ATTACH VMS: HFT FETCH (DTRC) Examples: MSACCES, UN=myid, MPW=mymsspw. MSFETCH, DN=in1, MDN=mymsfyl. MSFETCH, DN=in2, MDN=hisfy1, UN=him, DF=CB, PW=fylepw. IN1 is your file MYMSFYL transferred without conversion. IN2 is file HISFYL belonging to user HIM converted from CDC Display Code (FYLEPW is the password HIM requires for access to the file). MSPASSW Change Mass Storage System access password. MSPASSW, OLD=o1dpw, NEW=newpw. Syntax: Parameters: oldpw - your current MSS access password newpw - your new MSS access password Remarks: MSACCES is required before using the MSx commands. Similar commands: NOS: PASSWOR . VMS: HFT PASSWORD MSACCES, UN=myid, MPW=mymsspw. Examples: MSPASSW, OLD=mymsspw, NEW=newmsspw. MSPURGE (DTRC) Purge a file from the Mass Storage System. Syntax: MSPURGE, MDN=mdn. Parameters: mdn - the MSS dataset (file) name (default: MDN=dn) Remarks: MSACCES is required before using the MSx commands. Similar commands: NOS: PURGE VMS: HFT DELETE; MSSDELETE (both DTRC) Examples: MSACCES, UN=myid, MPW=mymsspw. MSPURGE, MDN=mssfy11.

MSSTORE (DTRC) Store a file on the Mass Storage System. Syntax: MSSTORE, DN=dn, MDN=mdn, DF=df, CT=ct, NA=na, PW=pw, BR=br, M=m, PR=pr, AC=ac. Parameters: dn - the local dataset name mdn - the MSS dataset (file) name (default: MDN=dn) df - data format TR - transparent (no conversion) CB - character blocked (convert from CDC display code) (default: DF=TR) ct - Category type P - private PU - public S - semi-private (default: CT=P) na - No Abort 0 - abort if file already exists on the MSS 1 - replace the old MSS file, if one exists (default: NA=0) pw - optional MSS file password br - backup requirements meaning br ___ CR off-station backup Y on-station backup (default: BR=Y) - alternate user permission mode for m semiprivate and public files m meaning E (EXECUTE) you can execute; others can read or execute concurrently R (READ) all can read or execute concurrently RU (READUP) all can read or execute; one (other) user can rewrite the file R^ (READAP) all can read or execute; one (other) user can lengthen the file

RM (READMD) all can read or execute; one (other) user can lengthen or rewrite the file (UPDATE) П all can read or execute; you can rewrite the file A (APPEND) all can read or execute; you can lengthen the file M (MODIFY) all can read or execute; you can lengthen or rewrite the file (WRITE) W you can read, execute, lengthen, rewrite, or shorten the file; others have no concurrent access br - backup requirements br meaning ___ _____ CR off-station backup Y on-station backup pr - preferred residence meaning рr ---Μ alternate storage - MSS Ν no preference (default: PR=N) ac - may alternate users obtain information about the file? (Y or N) Remarks: MSACCES is required before using the MSx commands. See also: DISPOSE Similar commands: NOS: DEFINE VMS: HFT STORE (DTRC) Examples: MSACCES, UN=myid, MPW=mymsspw. MSSTORE, DN=in1, MDN=mssfyl1. MSSTORE, DN=in2, MDN=mssfy12, BR=CR. MSSTORE, DN=in3, MDN=mssfy13, DF=CB, NA=1, PW=fy1epw. IN1 is stored as private file MSSFYL1. IN2 is stored as private file MSSFYL2 with offstation backup. IN3 is stored as private file MSSFYL3 (even is MSSFYL3 already exists) in CDC Display Code. FYLEPW is the password required for another user

to access the file.

90/09/30 Rev0

NEWCHRG

Page C-47

Syntax: NEWCHRG, OLD=oldchrgno, ID=id. Parameters: OLD= - the account number to be changed ID=id - change all files having this ID ID - change all files having a null ID (default: change all IDs) Remarks: NEWCHRG changes from the specified account number to the "current" number of the Cray job (from the ACCOUNT or most recent ALTACN statement). See also: ALTACN; MODIFY; Appendix D: CNEWCHRG Similar commands: NOS: BEGIN, NEWCHRG Examples: JOB, JN=.... ACCOUNT, AC=.... ACCESS, PROCLIB, OWN=PUBLIC. LIBRARY, PROCLIB:*. NEWCHRG, OLD=1222233344. ^-- change all files from account 1-2222-333-44 to the current one _ _ _ _ _ NEWCHRG, OLD=1222233344, ID=myid, ^-- change all files WITH ID=MYID from account 1-2222-333-44 to the current one - - - - -. . . ALTACN.AC=5666677788. NEWCHRG, OLD=12222433344. ^-- change all files from account 1-2222-333-44 to 5-6666-777-88 NOHOLD Cancel the effect of HOLD.

(DTRC - PROCLIB) Change permanent file account number.

Syntax: NOHOLD,GRN=grn. Parameters: GRN≖grn - generic resource name See also: HOLD

NORERUN Control a job's rerunability.

Syntax: NORERUN, option.

Rev0 90/09/30

Page C-48

Parameters: option - ENABLE - declare a job nonrerunable if any of the nonrerunable functions are done DISABLE - stop monitoring nonrerunable functions (if a job has already been declared nonrerunable, that status is not changed) (default: ENABLE) See also: RERUN Similar commands: NOS: NORERUN NORERUN, DISABLE. Examples: NOTE Write text to a dataset. NOTE, DN=dn, TEXT='text'. Syntax: Parameters: DN= - the dataset to be written (at its current position) DN - write to \$OUT TEXT= - up to 153 character to be written Similar commands: NOS: NOTE VMS: OPEN, WRITE, CLOSE NOTE, DN=UIN, TEXT= '*COMPILE myprog, mysub'. Examples: REWIND, UIN. UPDATE, I=UIN, OPTION Specify user-defined options. OPTION, LPP=n, PN={ p | ANY }, STAT=stat, BS=bsz, Syntax: ST=dev, DEF=pdev, XSZ=mxsz:mnsz, RDM, SEQ, UNB.BLK,NOF,OVF,SPD=sect,BFI=bfi, LM=mxsz,SZ=dsz. Parameters: LPP=n number of lines per page for job listings (0-255 decimal) LPP=0 do not change the current setting (default: 66) PN=p - select a processor (p is 1 or 2) PN=ANY - any available processor (if invalid, job aborts with an error message) (default: ANY)

| BS | buffer size (# of octal 512-word blocks in circular I/O buffer) (default: system defined; BS and UNB are mutually exclusive) |
|-----|--|
| ST | storage device type SCR - scratch PERM - permanent |
| DEF | - preferred device types |
| XSZ | maximum and minimum transfer sizes in octal sectors (default: system defined; normally half the buffer size) |
| RDM | ramdom dataset (default: sequential; RDM and SEQ are mutually exclusive) |
| SEQ | sequential dataset (default: sequential; RDM and SEQ are mutually exclusive) |
| UNB | - unblocked dataset (default: blocked; UNB and BS are mutually exclusive) |
| BLK | <pre>- blocked dataset (default: blocked; BLK and UNB are</pre> |
| NOF | do not overflow to another device (default: system defined; NOF and OVF are mutually exclusive) |
| OVF | - overflow allowed (default: system defined; NOF and OVF are mutually exclusive) |
| LN | - maximum number of decimal 512-word blocks for a dataset - job aborts if exceeded |
| SZ | number of decimal 512-word blocks to reserve for dataset when it is created (default: system defined) |
| BFI | blank field initiation (octal ASCII code signaling the beginning of a sequence of blanks) OFF - no blank compression (default: octal 33 (ESC)) |
| SPD | - dataset is striped |

Rev0 90/09/30

STAT= - the level of I/O statistics gathered
 for local datasets to appear in the
 user logfile
 (user level - accounting information
 system level - device information)
 ON - installation defined
 OFF - no statistics
 FULL - user and system info
 (default: OFF)
STAT - same as STAT=ON

Similar commands: VMS: SUBMIT /QUEUE=

PASCAL Compile a Pascal source program.

Syntax: PASCAL, I=idn, L=1dn, B=bdn, O=1ist, CPU=cpu:char.

Parameters: B= - generated binary load modules (default: \$BLD)

- CPU= Cray to execute the program cpu - CRAY-XMP CRAY-X1 - single-processor CRAY-X2 - dual-processor (default: the compiling machine)
 - char [NO] EMA - extended memory (24-bit A-register immediate loads; $common \ blocks > 4$ million words) [NO]CIGS - compressed index scatter/gather [NO] VPOP - vector population and parity [NO] READVL - vector length read instructions MEMSIZE=nK - (n * 1024) words MEMSIZE=nM - (n * 1048576) words[NO] BDM - bidirectional memory

Similar commands: NOS, VMS: PASCAL

Examples: PASCAL, I=mypasc.

PERMIT Grant/deny access to a permanent dataset.

Syntax: PERMIT, PDN=pdn, ID=uid, AM=am, RP, USER=ov, ADN=adn, NA, ERR, MSG.

Parameters: PDN=pdn - required

RP - remove the permissions

USER=ov - the name (User Initials) of the user to be granted/denied permission

ADN=adn - local dataset with the permit list

Similar commands: NOS: CHANGE; PERMIT VMS: SET PROTECTION; Access Control List

Examples: PERMIT, PDN=myfile, USER=abcd, AM=R. ^-- make file readonly for user ABCD = = = = PERMIT, PDN=myfile, USER=abcd. AM=N. ^-- remove all permissions for user ABCD

PRINT Write the value of a JCL expression to the logfile.

Syntax: PRINT(expression)

Parameters: exp - any valid JCL expression (maximum length: 8 characters)

Logfile format: UT060 decimal octal ASCII

Similar commands: NOS: DISPLAY VMS: WRITE SYS\$OUTPUT

Examples: SET(J1=J1+1) PRINT,J1.

PROC Begin an in-line procedure definition block. This is followed by the procedure prototype statement which names the procedure and gives the formal parameter specifications.

> Syntax: PROC, L=1dn, LIB=plib. name, p1, p2,..., pn ... ENDPROC.

Parameters: L - listing dataset to receive the echo of the definition block (default: \$LOG)

- name the name of the procedure (1-8 alphanumeric characters; should not be the same as a system verb)
- pi a formal parameter specification in one
 of the following formats:

pos - positional key=dvalue:kvalue - keyword key - formal keyword parameter dvalue - optional default value if the parameter is omitted kvalue - optional value if the parameter is specified with no value special cases: key= - specify a null value key=: - no defaults, but caller may specify key= or just key

See also: Section 3-3

Similar commands: NOS: .PROC VMS: always 8 parameters

Examples: PROC.

ENDPROC.

QUERY Determine the current status and position of a local file.

Syntax: QUERY, DN=dn, STATUS=status, POS=pos.

3 dn is open for I/O

POS= - the JCL symbolic variable name to receive the position of the dataset return values: value meaning _____ _____ -1 position indeterminate (not local, unblocked, closed) 0 BOD (beginning-of-data) 1 EOD (end-of-data) 2 EOF (end-of-file) EOR (end-of-record) 3 4 mid-record Remarks: In addition, a logfile message is generated: QUO01 - DN: 1dn STATUS: status POS: pos where status is UNKNOWN, CLOSED, OPEN-0, OPEN-1, OPEN-I/O is N/A, BOD, EOD, EOF, EOR, MID pos Similar commands: NOS: ENOUIRE VMS: no local file concept Examples: QUERY, DN=myfile, STATUS=J0, POS=J1. IF(J0.LT.0)COMMENT. file myfile is not local . . . ELSE. COMMENT. file myfile is local ENDIF. RELEASE Return a dataset. Syntax: RELEASE, DN=dn1:dn2:...:dn8, HOLD. Parameters: DN= - up to 8 dataset names HOLD - hold generic resource (do not return the allocation to the system pool) See also: HOLD, NOHOLD Similar commands: NOS: RETURN RELEASE, DN=temp:file1:out. Examples:

```
Page C-54
```

Rev0 90/09/30

RERUN Control a job's rerunability. Syntax: RERUN, option. Parameters: option - ENABLE - mark job as rerunable regardless of any nonrerunable functions which may have been performed so far in the job DISABLE - mark the job as nonrerunable (default: ENABLE) See also: NORERUN Similar commands: NOS: NORERUN Examples: RERUN, ENABLE. RETURN Return control from a procedure to its CALLer. RETURN. Syntax: RETURN, ABORT. Parameters: ABORT - cause COS to issue a job step abort Similar commands: NOS: REVERT VMS: EXIT Examples: See PROC. REWIND Position a dataset at its beginning. Syntax: REWIND, DN=dn1:dn2:...:dn8. Parameters: DN = -up to 8 datasets to be rewound Similar commands: NOS: REWIND Examples: REWIND, DN=temp:out:in1. ROLLJOB Protect a job by writing it to disk. Syntax: ROLLJOB. Remarks: There is no guarantee that a job will remain recoverable.

Examples: ROLLJOB.

90/09/30 Rev0

Page C-55

SAVE Make a local dataset permanent and define its characteristics. Syntax: SAVE, DN=dn, PDN=pdn, ID=uid, ED=ed, RT=rt, R=rd, W=wt, M=mn, UQ, NA, ERR, MSG, EXO=exo, PAM=mode, ADN=adn,ACN. Parameters: RT=rt - retention period RT =- set to default ADN=adn - local dataset with the permit list ACN - use the alternate account number See also: ALTACN, MODIFY Similar commands: NOS: DEFINE: SAVE **VMS:** CREATE Examples: SAVE, DN=out, PDN=ABCOUT. SAVE, DN=prog, PDN=mastprog, M=maint, PAM=R. ^-- the file is world-readable and YOU can't accidentally delete it SCRUBDS Write over a dataset before release. Syntax: SCRUBDS, DN=1fn. Parameters: lfn - the uniquely accessed file to be overwritten Remarks: SCRUBDS writes zeros over an existing dataset. Examples: ACCESS, DN=myfy1, PDN=myfy1e, UQ. SCRUBDS, DN=myfy1. SEGLDR Segment loader. Syntax: SEGLDR, I=idn, L=1dn, DN=bdn1:bdn2:...:bdn8, LIB=lib1:lib2:...:lib8,ABS=adn, CMD='directives'.GO. Parameters: I= - Dataset with SEGLDR directives (detault: \$IN) I - Same as I=\$IN L= Listable output (default: \$0UT) - Same as L=\$OUT L DN= - Up to 8 binary load dataset(s)

> - Same as DN=\$BLD (default: \$BLD)

DN

Page C-56

Rev0 90/09/30

- ABS= Datuset to contain the absolute program (default: \$ABD)
- CMD= Global directives to be processed; treated as first record read from J=idn; separate commands with semicolons (e.g., CMD='BIN=bdn;MAP=PART')
- GO Load and execute; ignored for a segmented load
- Remarks: By default, input load modules are read from \$BLD.

Directives: See section 3-6.

Similar commands: NOS: SEGLOAD VMS: virtual machine

Examples: CFT77,B=mycvj. SEGLDR,DN=myobj,LIB=mylib,CMD='MAP=PART',GO.

SET Change the value of a JCL variable.

Syntax: SET(symbol=expression)

Parameters: exp - a valid arithmetic, logical or literal assignment expression - may be delimited by parentheses

Remarks: The job-step aborts if the variable is unknown, is changable only by COS, or is a constant.

Similar commands: NOS: SET VMS: \$ name = value

Examples: SET(J1=J1+1) <-- increment procedure-local register J1 by 1

> SET(G1=(SYSID.AND.177777B)) ^-- put the low-order 2 characters of the current system revision level into global register G1

SET(G3=((ABTCODE.EQ.74).AND.(G2.EQ.0))) ^-- define global register G3

SID Debug programs interactively or in batch.

Syntax: SID=adn, I=idn, S=sdn, L=lsn, ECH=edn, CNT=n.

Page C-57

Parameters: adn ~ absolute dataset name (from LDR, AB=adn) **T** = - Input directives (default: \$IN) - Symbol dataset name S= (default: \$DEBUG) L= - Listable output (default: \$OUT) ECH= - Dataset for echoing input directives (default: no echoing) ECH - Same as ECH=1dn CNT= - Breakpoint interrupt count (default: 0 (no abort)) Similar commands: NOS: CID VMS: DEBUG SKIPD Skip blocked datasets (position at EOD (after last EOF)). Syntax: SKIPD, DN=dn. Parameters: DN - (default: \$IN) SKIPF, DN=dn, NF. Same as: Similar commands: NOS: SKIPEI VMS: OPEN with ACCESS=APPEND in program Examples: SKIPD, DN=myfile. SKIPF Skip blocked files from current position. SKIPF, DN=dn, NF=nf. Syntax: Parameters: DN=dn - (default: \$IN) NF=nf - decimal number of files to skip forward NF=-nf - decimal number of files to skip backward NF - position after the last EOF of the dataset (default: NF=1) Similar commands: NOS: SKIPF; SKIPFB; SKIPR Examples: SKIPF, DN=myfile.

SKIPR Skip blocked records from the current position.

Syntax: SKIPR, DN=dn, NR=nr.

Parameters: DN=dn - (default: \$IN)

NR=nr - decimal number of records to skip forward NR=-nr - decimal number of records to skip backward NR - position after the last EOF of the current file (default: NR=1)

Examples: SKIPR, DN=myfile.

SKIPU Skip sectors on unblocked datasets.

Syntax: SKIPU, DN=dn, NS=ns.

Parameters: DN=dn - no default

| NS=ns | | ecimal orward | number | of | sector | s to | skip | |
|--------|---|------------------------------|--------|----|--------|------|-------|-----|
| NS=-ns | | ecimal ackward | | of | sector | s to | skip | |
| NS | d | osition ataset default | | | e last | sect | or of | the |

- Examples: SKIPU, DN=myfile.
- SORT Sort/merge.
 - Syntax: SORT, S=sdn[:sdn...], M=mdn[:mdn...], O=odn, DIR=ddn, L=ldn, ECHO, RETAIN, NOVERF.
 - Parameters: S= Input dataset of up to 8 unsorted files M= - Input dataset of up to 8 sorted files to be merged (S or M or both must be specified)
 - 0= Output dataset (required)
 - DIR= Dataset with SORT directves (default: \$IN)
 - L= Listable output (default: \$OUT) L=0 - No listable output
 - ECHO Write directives to L=ldn (Not allowed if L=0)

Page C-59

RETAIN - Retain input order for equal keys NOVERF - Do not verify the sort (default: verify) Similar commands: NOS: SORT5 VMS: SORT SPY Generate a histogram on time usage within a program to locate inefficient code. Syntax: SPY, PREP, BS=bcktsz, D=dbugdn, S=scrtch, SUB=rtnl:rtn2:..:rtnn,TS=time. SPY, POST, ADDRESS, L=listdn, NOLABEL, NOLIB, S=scrtch, SUB=rtn1:rtn2:..:rtnn,MINHIT=n. Parameters: BS= - bucket size in words; each bucket begins on a word address that is a multiple of the bucket size (default: 4) D= - dataset containing the program's symbol table (default: SDEBUG) S= - dataset where SPY, PREP will write tables for SPY, POST to use SUB= - list of up to 20 routines to be analyzed - time slice in microseconds TS= (default: 500) ADDRESS - the report will be by address instead of by label L= - the output report listing dataset (default: \$OUT) NOLABEL - the bucket size will be an entire routine NOLIB - exclude library calls to routines whose names begin with "\$" MINHIT= - minimum number of hits required to generate a report line for a bucket or label (default: 1; 0 is NOT recommended)

Page C-60

SUBMIT

Rev0 90/09/30

Remarks: At SPY's request, COS reads the address of the current machine instruction. A group of addresses is called a bucket; accessing a bucket is called a hit. After execution, SPY generates a report of all buckets, including a bar graph showing where the time has been spent. Use SEGLDR to create the absolute; LDR mixes code and data making it more difficult to analyze. Similar commands: NOS: HOTSPOT VMS: PCA Examples: CFT.GN=IZ. -or- CFT77, ON=Z. -or- CAL, SYM. -or- PASCAL, 0=DM3. SEGLDR, ABS=myabs. <-- you must create an absolute program <-- prepare for SPY SPY.PREP. <-- run your program myabs. SPY, POST. <-- prepare the report EXIT. SPY, POST. Since an absolute module is always created, you could use SEGLDR. SPY, PREP. \$ABD. SPY, POST. EXIT. SPY, POST. Send a local dataset to the COS input queue. SUBMIT, DN=dn, SID=sf, DID=df, DEFER, NLRS. Syntax: Parameters: - Dataset containing the job (required) DN= SID= - Source front-end identifier (2 alphameric characters) DID= - Destination front-end identifier (2 alphameric characters) DEFER - Defer the SUBMIT until the dataset is released (default: SUBMIT occurs immediately) NLRS - Do not release the dataset after the SUBMIT; it remains local and read-only (default: dataset is released after the

SUBMIT)

Page C-61 90/09/30 Rev0 Similar commands: NOS: ROUTE, DC=IN; CSUBMIT VMS: SUBMIT; CRAY SUBMIT SUBMIT, DN=myjob1. Examples: Turn pseudo sense switches on/off. SWITCH Syntax: SWITCH, n=x. Parameters: n - switch number (1-6) x - switch position ON - turned on (set to 1) OFF - turned on (set to 0) Similar commands: NOS: SWITCH; OFFSW; ONSW SWITCH, 2=ON. Examples: UNBLOCK Convert a blocked dataset to an unblocked dataset. (1) UNBLOCK, DN=1dn. Syntax: (2) UNBLOCK, I=idn, O=odn. Parameters: DN= - the dataset to be replaced (using an intermediate dataset \$UNBLK) (1dn is rewound before and after) I = - the blocked input dataset (default: \$IN) (idn is not rewound before the copy) 0= - the unblocked output dataset (if previously marked to be unblocked (ASSIGN), odn is not rewound before; otherwise, odn is replaced) UNBLOCK is intended primarily for postprocessing Remarks: datasets created by or for certain stations. UNBLOCK, DN=myfile. Examples: ^-- Replace MYFILE with unblocked copy of itself UNBLOCK, I=myblk, O=myunblk. ^-- Copy blocked file MYBLK as unblocked file MYUNBLK

UPDATE Source and data maintenance. Syntax: UPDATE, P=pdn, I=idnl:idn2:...:idnn, C=cdn, N=ndn, L=1dn, E=edn, S=sdn, *=m, /=C, DW=dw, DC=dc,ML=n,&,opts. where & is one of: F Q[=d1:d2:...:dn] Q = 'd1, d2, ..., dj. dk, ..., dn'Parameters: P= - Program library dataset (default: \$PL) Ρ - Same as P=SPL P=0 - Required for a creation run I= - Input datasets with directives and text (Maximum: 100 datasets) (default: \$IN) - Same as I=SIN Ι I=O - No input dataset C= - Compile output dataset (default: SCPL) - Same as C=\$CPL С C=O - No compile output N= - New program library dataset (default: creation run: SNPL modification run: no new PL) - Same as C=\$CPL N N=0 - No new PL L= - Listable output (default: \$OUT) - Same as L=\$OUT L L=0 - No listable output E= - Error dataset name (default: \$OUT) E - Same as E=\$OUT E=0 - Errors written to L=1dn (If edn and 1dn are the same, 1dn is used and E=0) S= - Source output dataset (default: \$SR) S - Same as S=\$SR S=0 - No source output *=m - Master character for directives (defaults: creation run: modification run: read from the PL)

Page C-63

/=c - comment character (default: /) DW= - Data width (number of characters written per line to compile and source datasets (defaults: creation run: 72 modification run: dw when PL was created) DW - Same as DW=72 (creation) or use dw when PL was created (modification run) DC= - Declared modifications option: ON - mod declaration required OFF - mod declaration not required (default: OFF) ML= - Message level (highest severity level to suppress): 1 - comment2 - note 3 - caution4 - warning 5 - error(default: 3 - suppress COMMENT, NOTE, and CAUTION messages) F - Full UPDATE mode (default (F and Q omitted): normal UPDATE mode) 0= - Quick UPDATE mode (Maximum: 100 deck names) (default (F and Q omitted): normal UPDATE mode) opts - NA - no abort NR - no rewind of C and S files IF - write conditional text summary to ldn IN - write input to 1dn ID - write identifier summary to 1dn ED - write edited card summary to ldn CD - write compile dataset generation directives to 1dn UM - write unprocessed modifications to ldn and/or edn SQ - put sequencing in source output in columns dw+1 on (no effect on compile output) NS - no sequencing in compile output K - sequence decks according to Q Similar commands: NOS: UPDATE VMS: CMS; LIBRARIAN

Examples: UPDATE, I=mysorc, P=0, ID. ^-- create \$NPL, list identifiers = = = = = UPDATE. CFT, I=\$CPL. ... /EOF *COMPILE a,b,... /EOF

WRITEDS Initialize a blocked dataset.

Syntax: WRITEDS, DN=dn, NR=nr, RL=r1.

Parameters: DN=dn - required

NR=nr - required - decimal number of records to be written

- RL=r1 optional decimal record length (if non-zero, the first word of each record is the record number as a binary integer starting with 1) (default: 0 (a null record))
- Remarks: Writes a single file containing a specific number of records of a specific length. This is useful only for random (direct-access) files, which must be pre-formatted.

Examples: WRITEDS, DN=myfile, NR=1000, RL=125.

Page C-64

90/09/30 Rev0

Page D-1

***** Appendix D *****

*** DEC VMS DCL Commands ***

DEC VMS DCL (Digital Command Language) commands have the following general syntax:

verbparam1param2...! comments@filenameparam1param2...param8! commentsRUNfilename! comments

verb is the name of the routine to be executed. It consists of an alphabetic character (A-Z, a-z, \$, _) followed by 0-30 alphanumeric characters for the name of the command. A procedure (.COM) is executed using an at sign ("@") followed by the name of the procedure file. A user program is executed by the RUN statement.

parami are parameters, which may be positional or keyword.

comments follow an exclamation mark ("!") that is not part of a quoted parameter.

Because VMS has an extensive on-line help facility, the individual DCL commands are not described here. For a list of the help topics, type "HELP". For specific helps, type "HELP topic". The Computer Center maintains the following help libraries which are always available:

| HLP\$LIBRARY | @CCF | general information about the Computer Center |
|----------------|-------------------|---|
| HLP\$LIBRARY_1 | @DTLIB | subprograms in library DTLIB (Cray COS, CDC NOS, and DEC VAX/VMS) |
| HLP\$LIBRARY_2 | @UTILITIES | commands, programs, procedures, and packages added at DTRC |
| HLP\$LIBRARY_3 | @CRAY | DTRC additions to Cray |
| HLP\$LIBRARY_4 | | Reserved for future use |

Rev0 90/09/30

*** Selected DEC VAX/VMS Commands ***

The following are a few of the DEC VAX/VMS DCL commands:

ALLOCATE Assign a tape drive to a logical name.

Syntax:ALLOCATE device logical_nameParameters:device - the logical name of a specific or
generic tape drivelog_name - the name by which the tape is to be
known to the job (1-255 characters)Examples:\$ ALLOCATE MU: tape
^-- next available tape drive
starting with MU will be
assigned to logical name TAPE

DEALLOCATE Return a previously allocated device and disassociate the job's logical name from the tape drive.

Syntax: DEALLOCATE logical_name DEALLOCATE device name

DEALLOCATE /ALL

Parameters: log_name - the name by which the tape is known to the job dev_name - the name of the device (use if the device was not deallocated and the logical name is no longer defined)

Qualifiers: /ALL - deallocate all allocated devices

Examples: \$ DEALLOCATE tape ^-- deallocate the tape drive associated with logical name TAPE = = = = \$ DEALLOCATE \$2\$mua0

^-- deallocate tape drive mua0

DISMOUNT Release a tape volume that was previously mounted.

Syntax: DISMOUNT device_name

Parameters: device_name - the physical or logical name of the device to be dismounted

Qualifiers: /NOUNLOAD - Do not unload the tape (keeps the device and volume in a ready state (default: /UNLOAD)

Examples: \$ DISMOUNT /NOUNLOAD tape ^-- release file TAPE but keep the tape mounted for a future MOUNT

INITIALIZE Initialize a magnetic tape.

Syntax: INITIALIZE device vsn Parameters: device - the name given the tape in the ALLOCATE vsn - a 6-character volume serial number (all DTRC Network tapes are NAnnnn, where nnnn is a 4-digit number)

Remarks: HELP INITIALIZE for additional qualifiers

Examples: See page 6-1-6

MOUNT Mount a magnetic tape and, if labelled, check the label.

| Syntax: | \$ MOUNT | /BLOCKSIZE=mb1 | <pre>sn,] [logical_name] /COMMENT="string" /FOREIGN /RECORDSIZE=mr1 /[N0]WRITE</pre> |
|---------|----------|----------------|--|
| | | | |

Parameters: device - physical or logical name of the tape drive (for more than one tape, separate with commas or plus signs)

- vsn the volume serial number of the tape(s)
 as recorded in the tape's label record
 (0-6 characters)
 (not with /FOREIGN)

SET

| Qualifiers: | /BLOCKSIZE= | the default block size in bytes (range: 18-65,534; default: 2048) |
|-------------|----------------------|---|
| | /COMMENT = | specify additional information to the operator |
| | /DENSITY= | - the tape density (1600 or 6250) (default: the density of the first record of the volume) |
| | /FOREIGN | - an unlabelled tape |
| | /LABEL /NOLABEL | the tape has VAX/VMS ANSI labels the same as /FOREIGN (default: /LABEL) |
| | /RECORDSIZE= | - the number of characters in each record - normally used with /FOREIGN and /BLOCKSIZE (mrl <= mbl) |
| | /UNLOAD /NOUNLOAD | unload the tape when DISMOUNTed do not unload the tape (default: /UNLOAD) |
| | /WRITE /NOWRITE | the tape can be written the tape is read only (default: /WRITE) |
| Examples: | /REC | e: /FOREIGN /DENSITY=1600 - CORDSIZE=140 /BLOCKSIZE=5040 - mment="Please mount slot98 ", - "vsn=ABCD01 ring" ^ mount a slot tape for writing blocked records |
| | | DIOCKEN TECOLUS |
| | | be NA9999 /DENSITY=1600 ment="Pls mount with NO ring" |
| | 7 00 000 | <pre>^ mount a read-only tape</pre> |
| | = = = = = | |
| | tape. | 6 for an example of initializing a |
| | | eristics for subsequent use of a tion a magnetic tape. |
| Syntax: | SET MAGTAPE | device /DENSITY /END_OF_FILE /LOG /LOGSOFT /REWIND /SKIP=option |
| Parameters: | | logical name of a specific or ric tape drive |

90/09/30 Rev0

Qualifiers: /DENSITY - the default density (1600 or 6250) for writes to a foreign or unlabelled tare /END_OF_FILE - write a tape mark at the current position on the tape /LOG - display information about what was done /LOGSOFT - log soft errors on the error log file /REWIND - rewind the tape /SKIP= - position the tape option meaning --------BLOCK:n skip <n> blocks END_OF_TAPE position at the end-of-tape mark FILES:n skip <n> files RECORD:n skip <n> records /UNLOAD - rewind and unload the tape Similar commands: NOS: BKSP, REWIND, SKIPEI, SKIPF, SKIPFB, UNLOAD, WRITEF \$ MOUNT tape: /FOREIGN Examples: \$ SET MAGTAPE tape: /DENSITY=6250 ^-- mount a foreign tape and set the write density to 6250 cpi ----\$ SET MAGTAPE /SKIP=FILES:4 ^-- skip forward 4 files

Rev0 91/01/15

*** Selected DEC VAX/VMS Additions ***

The following are DTRC additions to DEC VAX/VMS:

APRINT (DTRC) Print one or more files on the printer in Annapolis.

| Syntax: | APRINT file(s) copies delete PRINT_quals |
|-------------|---|
| Parameters: | file - the file(s) to be printed |
| | copies - number of copies (default: 1) |
| | <pre>delete - DELETE to delete the file(s) after printing (default: keep the files)</pre> |
| | PRINT_quals - P4-P8 may be used for additional qualifiers for the PRINT statement |
| Remarks: | All PRINT qualifiers which we support are available. |
| | The files are places into terminal queue SYS\$ANAP. |
| See also: | CAPRINT |
| Examples: | APRINT myfile.out APRINT myfile.out 4 |

AUX

(DTRC) Turn an auxiliary printer on or off; for supported printers, send control characters to control character size and page eject.

Syntax: AUX option

Parameters: option - one of:

On/off: ON - turn printer on OFF - turn printer off Page eject: TOP - page eject (leave AUX ON) TOPOFF - page eject (leave AUX OFF) ALPS: ACC - condensed (17 cpi) - Pica (10 cpi) APC Brother 2024L: BCC - condensed characters (18 cpi) BCCOFF - condensed characters off - Elite (12 cpi) BEC BPC - Pica (10 cpi) BWC - wide characters (5 cpi) BWCOFF - wide characters off Okidata MicroLine 82 or 84: - condensed characters (15 cpi) 000 OLC - large (8.3 cpi) OPC - Pica (10 cpi) OWC - wide characters (5 cpi) VC132 - same as OCC VC80 - same as OPC

Remarks: Other printers which use the same control codes may use the corresponding options.

See also: AUXPRINT

Similar commands: NOS: BEGIN, AUX; BEGIN, AUXPRNT

Examples: \$ AUX ON \$ TYPE myfile.dat \$ AUX OFF = = = = = \$ AUX ON \$ AUX BCC <-- condensed on Brother 2024L \$ TYPE myprog.for \$ AUX BCCOFF \$ AUX OFF AUXPRINT (DTRC) Print one of more files on an axiliary printer (one attached to an interactive terminal) AUXPRINT files /ALLTYPES /CC /COPIES= /CS= Syntax: /DOUBLE SPACE /EJECT AT END /FF /HEADER /LOG /NEWPAGE /PS= /PW= /SHIFT= /SKIP= Parameters: files - the name of the file or commaseparated list of files to be printed -- wildcards are allowed (defaults: filename: FOR001; type: .DAT) Qualifiers: /ALLTYPES - controls the processing of certain file types (default: /NOALLTYPES) /CC - carriage control is in column 1 (default: /NOCC) /COPIES= - number of copies (default: 1) /CS= - character set (see AUX) ALPS: ACC, APC Brother 2024L: BCC, BDC, BEC, BPC, BWC Okidata MicroLine 82 or 84: OCC, OLC, OPC, OWC (default: however the printer is set) /DOUBLE - double spacing (default: /NODOUBLE) /EJECT - eject to a new page at the end of the last file printed (ignored for /CC) (default: /NOEJECT_AT_END) /FF - page eject for <FF> in column 1 (forces /NOCC; /NOFF ignores <FF>) (default: /FF) /HEADER - print a header with the filename before printing the file (default: /NOHEADER) /LOG - display information about the printing (default: /LOG) /NEWPAGE - start each file on a new page (default: /NONEWPAGE) /PS=- the page size (number of lines possible per page) (default: /PS=66) /PW= - the page width (maximum number of columns per line) -- less than 133 (defaults: /PW=80; /PW or /PW=0 implies /PW=132)

90/09/30 Rev0

Page D-9

- number of columns to shift each line /SHIFT= (for /CC, columns 2 on are shifted) (default: /SHIFT=0; /SHIFT implies /SHIFT=1) /SKIP= - number of lines to skip before printing the file (default: /SKIP=0: /SKIP implies /SKIP=10) Remarks: Other printers which use the same control codes as the ALPS, Brother 2024L or Okidata MicroLine 82 or 84 may use the /CS character sets. See also: AUX Similar commands: NOS: BEGIN, AUX; BEGIN, AUXPRNT Examples: AUXPRINT myprog.for ± = = AUXPRINT memo.txt /CS=BEC /CC ^-- print Elite characters on Brother 2024L printer, each line has carriage control in column 1 = = = = AUXPRINT a*.dat /CS=OCC /N /E ^-- print all .DAT files starting with A using condensed characters on an Okidata MicroLine 82 printer starting each file on a new page and ejecting a page after the last file - - - - -AUXPRINT myprog.lis /PW ^-- print a wide compilation listing (assumes wide paper is in the printer)

CAPRINT (DTRC) Convert the record attribute of a file having Fortran carriage control characters in column 1 of each line to "Fortran carriage control" and print on the remote printer in Annapolis.

> CAPRINT file copies keep PRINT quals Syntax: Parameters: file - the file to be printed copies - number of copies (default: 1) keep - any character will keep the converted file after it has been printed (default: delete the file) PRINT_quals - P4-P8 may be used for additional qualifiers for the PRINT statement Remarks: This is useful for printing CDC output files or any VAX file having column 1 carriage control but a different record attribute. The files are places into terminal queue SYS\$ANAP. See also: APRINT Examples: CAPRINT abcd.out ^-- convert and print file ABCD.OUT CAPRINT abcd.out 5 ^-- print 5 copies of ABCD.OUT CAPRINT abcd.out "" keep ^-- print 1 copy and keep the converted file (the next version of ABCD.OUT)

CNEWCHRG (DTRC) Change the account number on Cray permanent files from the VAXcluster.

Syntax: CNEWCHRG upw old_ac [new_ac] [id] [wait] [type]

Parameters: upw - Your user password for the generated ACCOUNT statement (the AC= value is taken from your current VAX/VMS session) (default: none - upw is required)

- type TYPE type the generated Cray job at your terminal other - do not type it
- Remarks: CNEWCHRG creates and submits a Cray job to make change.

CNEWCHRG works from any node of the VAXcluster. NEWCHRG is a Cray statement.

Any existing file NUCRAC.CPR is deleted before the Cray job is submitted.

This procedure creates and deletes all versions of file N\$U\$A\$C.JOB.

Similar commands: COS: ALTACN/MODIFY; NEWCHRG NOS: CHANGE Page D-11

CNEWCHRG myupw 1222233344 Examples: ^-- change all files from 1-2222-333-44 to the current (ACCOUNT) account number without waiting for it to complete Some time later: DIRectory NUCRAC.CPR ^-- see if the job has completed SET TERMinal /Width=132 TYpe NUCRAC.CPR ^-- look at it SET TERMinal /Width=80 DELete NUCRAC.CPR;* ^-- delete the file - - - - -CNEWCHRG myupw 1222233344 5666677788 "" WAIT ^-- change all files from 1-2222-333-44 to an alternate account number and wait for it to finish (note: the "" is the ID parameter - a null string to change all files) - - - - -CNEWCHRG myupw 1222233344 "" myid "" TYPE ^-- change ID=MYID files from 1-2222-333-44 to the current (login) account number without waiting for it to complete -type the generated job before submitting it This will display: JOB, JN=NUCRAC. ACCOUNT, US=myid, AC=myvmsaccount, UPW=myupw. ACCESS, DN=PROCLIB, OWN=PUBLIC. LIBRARY, DN=PROCLIB:*. NEWCHRG, OLD=1222233344, ID=myid. Look at it and delete it sometime later (see previous example).

CNEWPW (DTRC) Change your Cray password.

Syntax: CNEWPW old pw new pw new pw [ac] [wait]

Parameters: old pw - your current Cray password

new_pw - your new Cray password

- new_pw your new Cray password again for verification
- ac your Cray account number (if not the same as your current VMS login account number)
- Remarks: CNEWPW creates and submits a Cray job to make change.

CNEWPW works from any node of the VAXcluster. NUPW= is a parameter in the Cray ACCOUNT statement.

Any existing file NUCRPW.CPR is deleted before the Cray job is submitted.

This procedure creates and deletes all versions of file N\$U\$P\$W.JOB.

Similar commands: COS: ACCOUNT VMS: SET PASSWORD

Examples: CNEWPW myold mynew mynew ^-- change your password without waiting for it to be done Some time later: DIRectory NUCRPW.CPR ^-- see if the job has completed SET TERMinal /Width=132 <--- look at it TYpe NUCRPW.CPR SET TERMinal /Width=80 DELete NUCRPW.CPR;* <-- delete the file * = = = CNEWPW myold mynew mynew 1222233344 WAIT ^-- change your password for Job Order Number 1-2222-333-44 and wait for it to complete

CSUBMIT (DTRC) Submit a job to the Cray. CSUBMIT file(s) /AC=accountno /US=username Syntax: /UPW=password /NUPW /EOF=string /AFTER=time /LOG Parameters: file - one of: . a single file containing a complete Cray job . a comma- and/or plus-separated list of files which make up the Cray job (default filetype: .JOB) Qualifiers: /AC - if you have multiple account numbers, use /AC to specify an account number other than your current VAXcluster login account /AC is required if you use /US. (default: your VAXcluster login account number) /AFTER - specifies when the job is to be sent to the Cray (default: the job is queued for immediate submission to the Cray) /EOF - specifies the embedded COS end-of-file separator contained in the submitted job (if non-alphanumeric characters (including lower case letters) are used, they must be enclosed in quotes) For example, /EOF="E O F" means that lines containing just the 5-character string "E" space "O" space "F" are to be interpretted as end-of-file. /EOF=DAVE means that lines containing just the string "DAVE" are end-of-files. (default: /EOF="/EOF") /LOG - if you have CRAY SET TERM INFORM turned on and you do not want to see the message that your job has been queued, use /NOLOG (default: /LOG) /NUPW - indicates that your Cray password is to be changed

90/09/30 Rev0

Page D-15

You will be prompted for your current Cray password. If it does not match the database password, you are prompted for the password in the database. If they match, you are prompted for your new password, which will be put into the database.

Note: To do nothing more than change your password, use

CSUBMIT /NUPW

If you are changing only the password in the database, no Cray job will be generated. If you are changing your password on the Cray, a dummy job will be created and run with the output in file SETNUPW.CPR

Note: /NUPW cannot be used in a batch job

/UPW - the first time, your password will be entered into the database -- subsequently, use /UPW only if you are using a different /US

(default: /UPW=password in database)

/US - specify a different username for the Cray job (of course, you must be authorized to use the other username and must also supply /AC and /UPW)

Note: /US cannot be used in a batch job

(default: /US=the first 4 letters of your VAXcluster login username)

Remarks: This differs from CRAY SUBMIT in that jobs submitted using CSUBMIT do not need an ACCOUNT statement, CSUBMIT constructs it for you.

> The first time you use CSUBMIT, your password (/UPW) is added to a database. Every CSUBMIT then uses this database password to generate an ACCOUNT statement for you. Thus, your Cray job files no longer have your password, meaning that every time you change your password, you don't have to change all your Cray job files.

Rev0 90/09/30

Similarly, for your account number, a single Cray job may now be run under a different account, or even a different username, without changing the job file.

N.B. If there is an ACCOUNT statement in your job, it will be ignored and a new ACCOUNT statement will be generated.

Note that you must be logged into a VAXcluster node which connects to the Cray.

You can still use CRAY SUBMIT and RCSUBMIT to submit Cray jobs. However, these do not use (and cannot modify) the database, and, therefore, require ACCOUNT statements.

See also: CRAY SUBMIT; CNEWPW

Similar commands: NOS: CSUBMIT NOS/VE: SUBMIT_CRAY

Examples: \$ CSUBMIT myjob /UPW=mycraypw ^-- submit for the first time the Cray job in file MYJOB.JOB using CSUBMIT

- \$ CSUBMIT myjob ^-- submit the same job again

- \$ CSUBMIT myjob /US=other /AC=9888877766 /UPW=otherpw ^-- Submit the job as another user

Enter ABCD's New CRAY password. Password: <pw> Verification: <pw> %CX-S-SUB_OK, Job: SETNUPW queued for submission \$ \$ CSUBMIT /NUPW ^-- Change your Cray password (assuming you are user ABCD) when your actual Cray password is not the same as the one in the database -- perhaps you had changed it using CNEWPW or CRAY SUBMITted a job to change it Enter ABCD's current Cray password. Password: <pw> <-- your password entries are not echoed ABCD's current Cray password does not match the CSUBMIT password. Please enter ABCD's CSUBMIT password. Password: <pw> Enter ABCD's new Cray password. Password: <pw> Verification: <pw> <-- since your current S and new Cray passwords are the same. the database is updated, but no Cray job is created \$ CSUBMIT myjcl,myprog.for,mydatal.dat+ mydata2.dat,mydata3.dat ^-- Create and submit a job comprised of the following VAX/VMS files: MYJCL.JOB - Cray job control statements MYPROG.FOR - Cray Fortran program MYDATA1.DAT - first part of data file MYDATA2.DAT - second part of data file MYDATA3.DAT - separate data file \$ CSUBMIT myjob /EOF="The end" ^-- (upper case T, the rest is lower case) submit a job with the end-of-file lines as "The end" \$ CSUBMIT myjob /AFTER=18:00 (a) \$ CSUBMIT myjob /AFTER=TOMORROW (Ъ) \$ CSUBMIT myjob /AFTER=+00:05 (c) ^-- Submit the job a) after 6 PM, b) tomorrow, c) in 5 minutes

DETAB (DTRC) Remove tabs from a file or convert tab-format Fortran source lines to fixed-format. Syntax: DETAB in_file_spec out_file_spec /FORTRAN /LOG /INCREMENT=inc /TABS=tab_list Parameters: in_file_spec - the input file containing tabs out_file_spec - the output file with any tabs removed (default: next version of in_file_spec) Qualifiers: /FORTRAN - tab-format lines are converted to fixed-format (the first tab is set at column 7 (or 6 for continuation lines) and remaining tabs are converted to three blanks) Since tabs are collapsed to three blanks, it is unlikely that a DETABbed line will exceed 72 characters. If any lines do, you will be told how many and the length of the longest line. /NOFORTRAN - no reformatting is done /INCREMENT= - tabs are set every <inc> columns If both /TABS and /INCREMENT are specified, tabs are set at the column(s) specified by /TABS= and every <inc> columns after that. /LOG - list summary information and any warning messages (Default: /NOLOG) /TABS=n - set one tab at column n /TABS = (n1, n2, ..., nn)- set tabs at these columns If /INCREMENT=inc is not specified, then the tabs following the last defined tab stop, are each converted to a single blank.

91/05/15 Rev0

If /INCREMENT=inc is specified, then the tabs following the last defined tab stop will be every inc columns after the last defined tab stop.

(Defaults: /TABS=0 /INCREMENT=8 /NOFORTRAN)

Note: /FORTRAN overrides /TABS and /INCREMENT.

Remarks: This is useful for:

- . Preparing files to go to the Cray, Xerox 8700 or Microfiche, which don't recognize the tab character
- . Removing tabs in Fortran programs (for sending to another computer (such as the Cray and CYBER 860) which don't recognize the tab-format).
- . Changing the tab values while removing them (e.g., changing from every 8 columns, which is the VAX/VMS standard, to every 5 columns).

Examples: DETAB myprog.for /F

PRINTRM1 (DTRC) Print a file on the remote mini at Annapolis (RM1).

Remarks: Since RM1 is no longer available, use APRINT or CAPRINT to print in Annapolis.

(DTRC) Print a file on a CDC CYBER 860 central site. QPRINT Syntax: QPRINT vaxfile node /ASCII /DELETE /JOB=<job extension> /NAME=<job_name> /TID=<terminal_id> Parameters: vaxfile - file specification of the VAXcluster file to be printed on CDC node - the remote node on which the file is to be printed. One of: MFN - the CDC CYBER 860 Qualifiers: /ASCII - controls whether <vaxfile> is to be printed in upper and lower case (/ASCII) or just upper case (/NOASCII) (default: /NOASCII) /DELETE - controls whether <vaxfile> is deleted after it has been sent (default: /NODELETE) /JOB - the three alphanumeric characters to follow your user initials for the CDC jobname - if fewer than 3 characters, leading zeros are added (/JOB and /NAME are mutually exclusive) (default: /JOB=000) /NAME - the 1- to 7- alphanumeric character CDC job name - if fewer than 7 characters, it is padded on the right with zeros (/JOB and /NAME are mutually exclusive) (default: /NAME=xxxxext where xxxx are the executing user initials ext is the job and extension (/JOB))

/TID - Specifies where the file is to be printed -- no remote printers are currently supported (default: print at Central Site)

How it works: The CDC name of the output file is created from the /NAME or /JOB qualifier.

The output in your VAXfile is placed in the SYS\$QFT queue for transfer the HYPERchannel to the Mass Storage System (MSS) flagged for the node you requested.

Every 5 minutes or so, the queue transfer program on MFN checks for jobs coming to it and places them into their requested queues.

Remarks: CDC jobs may not have tabs or certain special characters. If /ASCII is not used, lower case will be folded into upper case. You may use RUN VSYS:CMP2FOR to remove tabs and change <FF> in column 1 to '1' before using QPRINT; the DETAB command may be used to remove tabs. Special characters not recognized by CDC will be converted to blanks by CDC.

The file must have Fortran carriage control.

Since RM1 is no longer available, use APRINT or CAPRINT to print in Annapolis.

Similar commands: COS: DISPOSE NOS: ROUTE

Examples: @VSYS:CMP2FOR myprog.lis ^-- prepare compilation listing for printing QPRINT myprog.lis MFN /NAME=xxxxABC /DELETE ^-- xxxx is the user initials; MYPROG.LIS will be deleted after is has been sent = = = = QPRINT /ASCII myprog.out MFN /JOB=1 ^-- print at with jobname xxxx001 in upper and lower case

Rev0 90/09/30

QSUBMIT (DTRC) Submit a job to a CDC CYBER 860 NOS input queue printing on the 860 Central Site Printer.

Syntax: QSUBMIT vaxfile node

Parameters: vaxfile - file specification of the VAXcluster file containing a CDC batch job (embedded end-of-records are indicated by a separate line containing only EOR in columns 1-3)

- node the remote node on which the job is to run -- one of: MFN - the CDC CYBER 860
- Remarks: CDC jobs may not have tabs or certain special characters; lower case will be folded into upper case. Special characters not recognized by CDC will be converted to blanks by CDC.
- How it works: The CDC job in your VAXfile is placed in the SYS\$QFT queue for transfer the HYPERchannel to the Mass Storage System (MSS) flagged for the node requested.

Every 5 minutes or so, the queue transfer program on MFN checks for jobs to it and places them into its input queue. The output is on a CDC central site printer.

As on CDC, if you want the job's output to be sent somewhere, then

ROUTE, OUTPUT, DC=PR, TID=<tid>, DEF.

should be placed in your CDC job to cause deferred routing of the entire file to another terminal ID.

- See also: QPRINT
- Similar commands: COS, NOS, VMS: SUBMIT NOS/VE: JOB; SUBMIT

Examples: QSUBMIT myfile.cdcjob MFN ^-- submits the CDC job in MYFILE.CDCJOB to the CDC CYBER 860 (MFN) NOS input queue with jobname from the job's JOB statement

RCAUDIT (DTRC) Create and submit a job to audit Cray files. Syntax: RCAUDIT cpw lo pdn id own acn sz wait Parameters: cpw - your Cray password 10 - list option ([S], A, B, L, N, P, R, T, X) 0 - use the default pdn - the file to be audited 0 - use the default (note: RCAUDIT cannot specifically audit file "0") (default: all files) - the ID for the file id 0 - null IDown - other owner's files 0 - use the default (default: your files) acn - restrict audit to this account number 0 - use the default (default: all account numbers) - restrict audit to files larger than this **SZ** many words 0 - use the default (default: all files) wait - WAIT - wait for the job to complete, display, delete the .CPR file (synonyms: YES, TRUE) other - do not wait (Cray job creates file RCAUD.CPR) Note: If P1 and P2 are both specified in the execute line, defaults are used for all other unspecified parameters. Remarks: Any existing file RCAUD.CPR is deleted before the Cray job is submitted. This procedure creates and deletes all versions of file R\$C\$A\$U\$D.JOB. See also: Appendix C: AUDIT Similar commands: NOS: BEGIN, AUDIT **VMS: DIRECTORY**

Examples: RCAUDIT mycraypw ^-- short audit of all my files (don't wait) ----RCAUDIT mycraypw 0 0 0 0 0 0 WAIT ^-- same (wait for completion) ~ - - -RCAUDIT mycraypw x ^-- "X" audit of all my files ~ _ _ _ _ _ RCAUDIT mycraypw x "A-" ^-- "X" audit of all files starting with "B" (the "" are needed because the "-" Cray wildcard is the VMS end-of-line continuation character) - - - -RCAUDIT mycraypw 0 0 0 0 1222233344 100000 ^-- Short audit of all my files larger than 100000 words under Job Order Number 1-2222-333-44 * = = * = RCAUDIT mycraypw x 0 0 abcd ^-- "X" audit of all ABCD's files (that I have permission to see) RCDELETE (DTRC) Delete a Cray permanent file. Syntax: RCDELETE pw pdn id ed m [wait] Parameters: pw - your Cray password pdn - the file to be deleted - the ID for the file id ed - the edition n - a specific edition +n - delete n highest editions -n - keep n highest editions ALL - delete all editions (default: delete the highest edition) - mainenance control word m wait - WAIT - wait for the job to complete, display, delete the .CPR file (synonyms: YES, TRUE) other - do not wait (Cray job creates file RCDEL.CPR) Remarks: Any existing file RCDEL.CPR is deleted before THE Cray job is submitted.

90/09/30 Rev0

RCGET

Page D-25

This procedure creates and deletes all versions of file R\$C\$D\$E\$L.JOB. See also: Appendix C: DELETE Similar commands: COS: DELETE, PDN= NOS, VMS: PURGE Examples: RCDELETE mycraypw abcde ^-- delete the highest edition of file ABCDE (don't wait) - - - - -RCDELETE mycraypw abcde 0 0 0 WAIT ^-- delete the highest edition of file ABCDE (wait for completion) _ _ _ _ _ _ RCDELETE mycraypw abcde zyx +3 ijk ^-- delete the high 3 editions of file ABCDE with ID=ZYX and maintenance control word IJK ----RCDELETE mycraypw abcde zyx -2 ijk ^-- keep the high 2 editions of file ABCDE with ID=ZYX and maintenance control word IJK RCDELETE mycraypw abcde 0 ALL (DTRC) Create and submit a job to get a Cray permanent dataset and save it as a VAX/VMS permanent file. Syntax: RCGET cpw VAXfile pdn id df ed r wait Parameters: cpw ~ your Cray password VAXfile - the VAX filespec for the file - use the default 0 .ext - "<pdn>.ext" (default: "<pdn>.") - the Cray file to be fetched pdn 0 - use the default (default: first 15 characters of the VAXfilename) (note: VAXfile and pdn may not both be "0") id - the ID of the file 0 - null ID(default: all files) df ~ data format (BB, CB, TR) 0 - use the default (default: CB)

ed - edition number 0 - use the default (default: the highest edition) - read control word r 0 - use the default (default: no read control word) - WAIT - wait for the job to complete, wait display, delete the .CPR file (synonyms: YES, TRUE) other - do not wait (Cray job creates file RCGET.CPR) (default: nowait) Remarks: Any existing file RCGET.CPR is deleted before THE Cray job is submitted. This procedure creates and deletes all versions of file R\$C\$G\$E\$T.JOB. See also: Similar commands: COS: DISPOSE Examples: RCGET mycraypw zyx.FOR abcde ^-- get Cray file ABCDE as VAX/VMS file ZYX.FOR (don't wait) _ _ _ _ _ _ RCGET mycraypw zyx.FOR abcde 0 0 0 WAIT ^-- same (wait for completion) RCGET mycraypw "" abcde ^-- get Cray file ABCDE as ABCDE. - - - - -RCGET mycraypw .FOR abcde ^-- Get Cray file ABCDE as ABCDE.FOR - - - - -RCGET mycraypw zyx.out abcde qrs 0 3 0 myreadcw ^-- get Cray file ABCDE, ID=QRS, ED=3 with read controlword as ZYX.OUT RCGET mycraypw plot.out plotout 0 BB 0 myreadcw ^-- Get Cray file PLOTOUT with DISSPLA output

RCSAVE (DTRC) Create and submit a job to save a VAX/VMS file as a Cray permanent dataset. Syntax: RCSAVE cpw VAXfile pdn id df pam m wait Parameters: cpw - your Cray password VAXfile - the VAX filespec for the file - use the default 0 .ext - "<pdn>.ext" (default: "<pdn>.") - the Cray file to be fetched pdn 0 - use the default (default: first 15 characters of the VAXfilename) (note: VAXfile and pdn may not both be "0") - the ID for the file id 0 - null ID(default: all files) - data format (BB, CB, TR) df 0 - use the default (default: CB) - public access mode pam E - execute only M - maintenance only N - no public access R - read only W - write only 0 - use the default (e.g., R:W gives read and write permission) (default: N) - maintenance control word m 0 - no maintenance control word - WAIT - wait for the job to complete, wait display, delete the .CPR file (synonyms: YES, TRUE) other - do not wait (Cray job creates file RCSAV.CPR) Note: If P1 and P2 are both specified in the execute line, defaults are used for all other unspecified parameters.

1 1 A

Rev0 90/09/30

Remarks: An LO=X audit is done for file <pdn>. Any existing file RCSAV.CPR is deleted before The Cray job is submitted. This procedure creates and deletes all versions of file R\$C\$S\$A\$V.JOB. See also: Similar commands: COS: VMS: Examples: RCSAVE mycraypw abcde.fgh zyx 0 0 r ^-- make my VAX/VMS file ABCDE.FGH a permanent dataset on the Cray with the name ZYX and having world read access (don't wait) - - - - -RCSAVE mycraypw abcde.fgh zyx 0 0 r 0 WAIT ^-- same (wait for completion) - - -RCSAVE mycraypw abcde.fgh "" "" r ^-- make my VAX/VMS file ABCDE.FGH a permanent dataset on the Cray with the name ABCDE and having world read access (the "" are place holders) = = = RCSAVE mycraypw abcde.fgh zyx qrs 0 0 ijk ^-- make my VAX/VMS file ABCDE.FGH a permanent dataset on the Cray with the name ZYX, ID=QRS and maintenance control word IJK (no permissions) RCSAVE mycraypw plot.out plotout 0 0 BB ^-- after RCGETting a binary blocked file (perhaps DISSPLA output),

send it back to the Cray

Page D-29

RCSUBMIT (DTRC) Submit a job to the Cray from any CCF VAXcluster node.

Syntax: RCSUBMIT job_file [password]

Parameters: job_name - the name of the file containing your Cray job

> password - your VAXcluster login password -for security, you may wish to omit this and be prompted for it (this is not used if you are on a node which is connected to the Cray)

Remarks: RCSUBMIT works from any node of the VAXcluster. CRAY SUBMIT works only on a node which connects directly with the Cray.

> KCSUBMIT and CRAY SUBMIT require an ACCOUNT statement in the job file. CSUBMIT does not, and ignores it if it is present. If you normally use CSUBMIT to submit your jobs, you should not use RCSUBMIT (or CRAY SUBMIT).

See also: CRAY SUBMIT; CSUBMIT

Similar commands: COS, VMS: SUBMIT NOS: ROUTE

Examples: RCSUBMIT crayjob myclustrpw ^-- from a node not connected to the Cray

where file CRAYJOB.JOB contains:

JOB, JN=test. ACCOUNT, AC=jobordrno, US=abcd, UPW=mypw. DISPOSE, DN=\$OUT, TEXT='node::', DEFER. ^-- at end-of-job, \$OUT will be put into file node::TEST.CPR FETCH, DN=test, TEXT='node::test.for' ^-- fetch program from node NODE CFT, I=test, L=0. FETCH, DN=FT05, TEXT='node::test.dat'. ^-- fetch data from node NODE SEGLDR.GO. DISPOSE, DN=FT04, TEXT='node::test.out'. ^-- send another output file of the program to node NODE * = = = CRAY SUBMIT crayjob ^-- from a node connected to the Cray (you can use RCSUBMIT, but this is faster)

*** Cray Station Commands ***

The VAX/VMS Cray Station provides the VMS user with access to the CRAY X-MP. The Cray Station is accessed via two commands: CRAY (all Station commands except interactive) and CINT (interactive access and a subset of the Station commands). The CRAY prompt is CRAY>; the CINT prompt is Cint>.

The following discussion of the Cray station commands is derived from the on-line helps for the CRAY and CINT commands. Type "CRAY HELP" or "CINT /HELP" at the DCL level, or "HELP" at the CRAY> of Cint> prompt for more detailed information.

| CRAY | | | ity or executes a single station s supplied as a parameter. |
|------|-------------|------------------|---|
| | Syntax: | \$ CRAY [station | o_command] /BREAKTHROUGH /REFRESH |
| | Parameters: | station_command | 1 - a single Cray station command to be executed |
| | | omitted | <pre>- you remain in Cray context until you enter EXIT</pre> |
| | Qualifiers: | /BREAKTHROUGH - | - a display refresh occurs during command input (valid for refresh mode only) (default: /NOBREAKTHROUGH) |
| | | /REFRESH - | enable display refreshing in a split screen Cray context (requires DEC CRT option enabled) |
| | | /NOREFRESH - | (requires DEC_CRT option enabled) - standard teletype environment (defaults: /REFRESH (VT100-type terminals) /NOREFRESH (non-VT100 terminals)) |
| | | CINT | |

See also: CINT

Similar commands: NOS: ICF

Examples: \$ CRAY

CINT From the DCL level, enter Cray interactive including a subset of the Cray context commands.

Syntax: \$ CINT /HELP /JN=jobname /MML=mm1 /UPPERCASE /PLAY=play_file /US=username

Qualifiers: /H - display help information without having to enter Cray interactive

- /J the interactive job name (first 7 characters used)
- /M maximum message length
- /P the play file to be run
- /US the username (1-15 characters)

Remarks:

See also: CRAY

Similar commands: NOS: ICF

Examples: \$ CINT Cray Jobname: myjob Cray Username: AMDS !ACCOUNT,....

Rev0 90/09/30

** Cray Context Commands **

The following commands may be executed at the CRAY> prompt. This identified with (CINT) may also be executed at the Cint> prompt.

| \$ | Create a temp commands. | porary VMS subprocess, allowing you to enter DCL |
|----|----------------------------|--|
| | Syntax: | <pre>\$ [dc1_command]</pre> |
| | Parameters: | dcl_command - any DCL command |
| | Remarks: | Since a subprocess is created, any logical names or process resources created in the subprocess will not be available from the main process. |
| | | To return to Cray context, type LOGOUT. |
| | Similar comma | ands: NOS ICF: |
| | Examples: | \$ show users |
| | | |
| + | Display the n | next page of information in Cray context. |
| | Syntax: | + |
| | Similar comma | ands: NOS ICF: |
| | Examples: | CRAY> + |
| | | |
| - | Display the p | previous page of information in Cray context. |
| | Syntax: | - |
| | Similar comma | ands: NOS ICF: |
| | Examples: | CRAY> - |
| Q | Fuere ter in | |
| e | Execute an in | ndirect station command file in Cray context. |
| | Syntax: | @file_spec |
| | Parameters: | <pre>file_spec - a VMS file containing station</pre> |
| | Remarks: | "@" is a synonym for the PLAY command. |
| | See also: | PLAY |

Similar commands: NOS ICF: /PLAY CRAY> @station.COM Examples: ABORT (CINT) Interrupt the current interactive Cray job step and return control to the COS Control Statement Processor (CSP). CSP will then issue the "!" prompt. Any COS output queued for the terminal will be displayed before the prompt is issued. Syntax: ABORT See also: DROP, KILL Similar commands: NOS ICF: ABORT CRAY> ABORT Examples: ATTACH (CINT) Redirect COS interactive terminal output to an alternate device. ATTACH [alt_device] /CHAR=(char,pos) Syntax: /MRS=max_rec_size /UFF /ON Parameters: alt device - the alternate device omitted - the current output device Qualifiers: /CHAR - route entire record to attached device if character <char> is in position <pos> of the current Cray interactive output record /MRS - route entire record (no carriage control) to attached device if the length of the current Cray interactive output record exceeds max_rec_size /OFF - do not route Cray interactive records to to attached device (all other parameters or qualifiers ignored) /ON - enable routing of Cray interactive records to an attached device Default: /ON Remarks: The device specified must not be in use and can be any device that accepts record I/O, such as a graphics terminal. Similar commands: NOS ICF: /CONNECT

. .

ATTENTION (CINT) Interrupt current interactive Cray job step and enter reprieve processing.

Syntax: ATTENTION

See also: ABORT

Remarks: If reprieve processing not specified, same as ABORT.

Similar commands: NOS ICF: /ATTENTION

Examples: CRAY> ATTENTION

BYE (CINT) Terminate an interactive session and, optionally, the COS interactive job.

Syntax: BYE /ABORT /SAVE

Qualifiers: /ABORT - terminate the associated COS interactive job

/SAVE - the associated COS interactive job remains active and output is saved; if the job reaches a COS threshold for output messages or requires input, the job is suspended; the terminal can be reconnected to the COS interactive job by the INTERACTIVE command

Remarks: BYE /ABORT is equivalent to QUIT.

See also: QUIT

Similar commands: NOS ICF: /BYE, /LOGOFF, /QUIT

Examples: CRAY> BYE

CLEAR Terminate any display command and clears the display portion of the screen.

Syntax: CLEAR

Remarks: CLEAR is only available when Cray context is in refresh mode.

Examples: CRAY> CLEAR

Page D-35

COLLECT (CINT) Store COS interactive output in a VMS file.

Syntax: COLLECT file spec /ECHO /OFF /ON

Parameters: file_spec - the VMS file to receive the COS interactive output

Qualifiers: /ECHO - display the output generated at the terminal as well as the VMS file /NOECHO - do not echo the generated output at the terminal; only into the VMS file (default: /ECHO)

- /OFF stop writing COS job output to a VMS file and close the VMS file (ignore other qualifiers
- /ON write COS job output to a VMS file
 (default: /ON)

Remarks: COLLECT can be used before the interactive job is initiated.

Examples: CRAY> COLLECT mycosfile.out

COMMENT Insert comments into an indirect station command file stream.

Syntax: COMMENT string

Parameters: string - any text

Remarks: The comment line can be 256 characters long, including "COMMENT".

See also: @, MESSAGE

Similar commands: NOS ICF: /*

Examples: COMMENT This is a comment

CONTROL_C 'CINT) CTRL-C (^C) performs the same function of the attention command.

Syntax:^C<--- ^ is the CTRL key</th>Remarks:Brings you back to the DCL prompt.See also:ABORT; ATTENTIONExamples:i ^C\$<-- leave Cray session abnormally</td>\$<-- you are back at the DCL level</td>

CONTROL_O (CINT) CTRL-O (^O) performs the same function as the discard command.

Syntax: ^0 <-- ^ is the CTRL key Remarks: ^0 toggles output on and off until the next Cray prompt. See also: DISCARD Examples: ! ^0

CONTROL Z (CINT) CTRL-Z (^Z) exits the current processing mode.

Syntax: ^Z <-- ^ is the CTRL key

Remarks: In response to the Cray context prompt (CRAY>), you are returned to DCL; in a Cray interactive session, you are returned to command mode. While you are being prompted for command parameters, CTRL-Z cancels the command.

CTRL-Z also terminates the execution of an indirect station command file.

See also: @

| Examples: | ! ^Z | < leave Cray session |
|-----------|------------|---------------------------------|
| | CRAY> QUIT | < terminate Cray session |
| | CRAY> ^Z | < terminate Cray context |
| | \$ | < you are back at the DCL level |

DATASET Test for the existence of a COS permanent dataset. Syntax: DATASET pdn /ID=id /ED=ed /OV=owner Parameters: pdn - name of PDS Qualifiers: /ID= - id of the dataset (1-8 characters) (default: null) /ED= - edition number of the dataset (1-4095) (default: current highest edition number) /OV= - owner of the dataset Examples: DATASET,myfile.

DELAY Suspend execution of an indirect station command file for a specified period of time.

Syntax: DELAY seconds Parameters: seconds - suspension time in seconds Examples: DELAY 20

DISCARD (CINT) Discard all output from a COS interactive session until the next COS prompt is issued.

Syntax: DISCARD

See also: ^O

Similar commands: NOS ICF: /DISCARD

Examples: DISCARD

DROP Terminate a COS job and return the associated output dataset. COS job execution enters reprieve processing after the next COS EXIT control statement.

Syntax: DROP jsq

Parameters: jsq - job sequence number

Remarks: Use STATUS to obtain the job sequence number (COS jsq).

KILL terminates the job immediately; DROP continues processing after an EXIT statement.

See also: ABORT, KILL

Examples: \$ CRAY CRAY> STATUS CRAY> DROP 9876

EOF (CINT) Sends an end-of-file record to a connected COS interactive job.

Syntax: EOF

Remarks: EOF is normally required to terminate COS file input from the terminal.

Similar commands: NOS ICF: /EOF

Examples: CRAY> EOF

Page D-37

Rev0 90/09/30

EXIT (CINT) Leave Cray context command mode and return to DCL. Syntax: EXIT ^Z Remarks: EXIT will close the file specified in a RECORD command, if it is still open.

See also: RECORD

Similar commands: NOS ICF: /EXIT

Examples: CRAY> EXIT

HELP (CINT) Display help information on the Cray station commands.

Syntax: HELP [station_command]

Parameters: station_command - a specific command for which help is desired omitted - a list of all available commands

Similar commands: NOS ICF: /HELP

Examples: \$ CRAY HELP = = = = = CRAY> HELP = = = = = CRAY> HELP CINT

ISTATUS (CINT) Get the status of your COS interactive job (with CPU time used and the last COS logfile message).

Syntax: ISTATUS

See also: JSTAT, STATUS

Examples: ISTATUS

JOB Display the status of a specific COS job.

Syntax: JOB jobname /JSQ=jsq

Parameters: jobname - the COS job name

Qualifiers: /JSQ= - the job sequence number from which to start the search for the job

Similar commands: NOS ICF: /STATUS

Examples: JOB myjob4

Page D-39

JSTAT Display the status of a specific job and its related tasks. JSTAT jsg /[NO]CYCLE /[NO]TRANSLATE Syntax: Parameters: jsq - the job sequence number Qualifiers: /CYCLE - cycle the display refresh through all the available information /NOCYCLE - display only the current page until you enter "+" or "-" (default: /NOCYCLE) - display the terminal ID field in /TRANSLATE the VMS UIC equivalent /NOTRANSLATE - display it in the station internal form (default: /TRANSLATE) Remarks: Use STATUS to obtain the COS job sequence number (j**sq**). ISTATUS, STATUS See also: Similar commands: NOS ICF: /STATUS Examples: JSTAT KILL Delete a job from the input queue, or immediately terminate an executing job, or delete the job's output dataset from the output queue. KILL jsq Syntax: Parameters: jsq - the job sequence number Remarks: Use STATUS to obtain the COS job sequence number (jsq). KILL terminates the job immediately; DROP continues processing after an EXIT statement. See also: ABORT, DROP Similar commands: NOS ICF: /ABORT Examples: CRAY> STATUS CRAY> KILL 9876

LOGFILE Provides access to the station logfile messages.

| Syntax: | LOGFILE | [file_spec] | <pre>/ACQUIRE /ALL /BEFORE=time /DISPOSE /ERROR /INTERACTIVE /JOB /MASTER /NETWORK /NODE=nodename /[NO]NOTIFY /OPERATOR /OUTPUT=file_spec /PRINT /RELEASE /SINCE=time /SUCCESS /STMSG /TRANSLATE</pre> |
|---------|---------|-------------|--|
|---------|---------|-------------|--|

Parameters: file_spec - An alternate station logfile to be displayed

Qualifiers: /ACQU - display ACQUIRE and FETCH messages

- /ALL display all messages
- /BEFO display messages from before a specified
 time
- /DISP display DISPOSE messages
- /ERRO display error messages
- /INTE display interactive processing messages
- /JOB display job submission messages
- /MAST display COS master operator messages
- /NETW display DECnet messages (all nodes)
- /NODE= display DECnet messages (one node)
- /NOTI you will be notified an asychronous LOGFILE operation is performed (requires /RELEASE) (default: /NONOTIFY)
- /OPER display operator messages

/SINC= - display messages since a specified time

Page D-41

/SUCC - display success, warning, and informational messages /STMS - display COS station messages and associated replies /TRAN - display terminal ID field (TID) as the VMS UIC equivalent /NOTR - display TID in the station internal form (default: /TRANSLATE) CRAY> LOGFILE jobname.LOG /SINCE=09:15 Examples: LOOP Restart execution of an indirect station command file at the beginning. Syntax: LOOP Remarks: CTRL-Z must be issued to terminate looping. Examples: CRAY> LOOP MESSAGE Send a message to the COS job logfile. MESSAGE string /JN=jobname Syntax: /JSQ=jsq Parameters: string - the message text (for embedded blanks, enclose in quotes ("...") Qualifiers: /JN= - the name of the COS job to receive the message (requires /JSQ) /JSQ= - the job sequence number of the COS job to receive the message COMMENT See also: Similar commands: NOS ICF: /* Examples: MESSAGE This is a message PAUSE Suspend execution of an indirect station command file. PAUSE Syntax: Control passes to the terminal, where you can Remarks: terminate the command file by entering a command or resume it by entering a null line (<RET>). Examples: PAUSE

PLAY (CINT) Execute an indirect station command file in Cray context.

Syntax: PLAY file_spec

Parameters: file_spec - a VMS file containing station commands

Remarks: PLAY files cannot themselves contain other (embedded) PLAY commands.

"@" is a synonym for the PLAY command.

Similar commands: NOS ICF: /PLAY

Examples: CRAY> PLAY station.COM

QUIT (CINT) Terminate a Cray interactive session and the corresponding COS interactive job.

Syntax: QUIT

Remarks: QUIT is the equivalent of BYE /ABORT.

See also: BYE

Similar commands: NOS ICF: /BYE, /LOGOFF, /QUIT

Examples:!^Z<-- leave Cray session</th>CRAY> QUIT<-- terminate the Cray session</td>CRAY> EXIT<-- terminate the Cray station</td>

RECORD Start or stop the recording of terminal input to a file while in Cray context for later use with the PLAY or @ commands.

Syntax: RECORD [file spec] /ON /OFF

Parameters: file_spec - the file into which terminal input is to be recorded

- Qualifiers: /ON start command recording (file_spec required)
 - /OFF end command recording (default: /ON)
- Remarks: Exiting Cray context automatically issues a RECORD/OFF.
- Examples: RECORD station.com /ON ... RECORD /OFF

Page D-43

REMOVE Delete entries in the dataset staging queue.

Syntax: REMOVE queue_id /LOCKED /SPOOL /STAGE

Parameters: queue_id - an 8-character hexadecimal number from the SHOW QUEUES display (leading zeros can be omitted)

Qualifiers: /LOCKED - controls whether or not locked entries are removed (default: /NOLOCKED)

/STAGE - remove an entry in the Cray staging queue

RERUN Immediately end the processing of a COS job and put it back into the input queue.

Syntax: RERUN jsq

Parameters: jsq - the job sequence number

Remarks: The job input dataset is saved and all output datasets associated with the job are deleted. The job input dataset is then rescheduled so the job can be rerun. No action is taken if the job execution is complete or if COS determines the job cannot be rerun.

Use STATUS to obtain the COS job sequence number (jsq).

SAVE Stages a VMS file to COS disk storage.

Syntax: SAVE file_spec /DELETE /DF=d /ED=ed /ID=id /MN=mn /PDN=pdn /RD=rd /RT=rt /US=us /WT

Parameters: file_spec - the file to be staged File_spec qualifiers: /DELE - delete the file when is has been successfully staged to the Cray /DF= - dataset format: CB, BB, or TR (default: CB)

/ED= - edition number (0-4095)
 (default: next higher number)

| | / I D= | - identification (1-8 alphameric chars) |
|-------------|---------------|---|
| | /mn= | - maintenance control word |
| | /PDN= | dataset name to be used (converted to uppercase) (default: the input file name) |
| | /RD≈ | - read permission control word |
| | /RT= | - the retention period, in days |
| | /US≃ | - the COS username |
| | /WT= | - the write permission control word |
| Examples: | SAVE m | yfile.dat /PDN=mydata /US=ABCD |
| INAL Define | the ter | minal working environment. |

SET TERMINAL Define the terminal working environme

SET TERMINAL FORTRAN

SET TERMINAL NOFORTRAN

Specify whether the terminal is to interpret output records from a COS interactive session as having FORTRAN carriage control.

Default: NOFORTRAN

SET TERMINAL INFORM

SET TERMINAL NOINFORM

Enable/disable the sending of station messages to the user logged on to VMS at a VAX terminal.

Default: NOINFORM

SET TERMINAL PAGE

SET TERMINAL PAGE=lines SET TERMINAL NOPAGE

Specify the number of lines of output before a page break.

Default: NOPAGE

Default for lines: determined by the scroll setting

SET TERMINAL REFRESH SET TERMINAL REFRESH=seconds <-- integer 0-60 SET TERMINAL NOREFRESH REFRESH provides a split-screen Cray context environment and is supported only on terminals with the DEC_CRT attribute. NOREFRESH provides a line-by-line Cray context environment.

Page D-45

÷

Defaults: REFRESH (VT100-type terminals) NOREFRESH (non-VT100-type terminals) SET TERMINAL SCROLL=lines Changes the Cray context window size. "lines" is the size of the command area (bottom window) and must be an integer from 3 to 13. Default for lines: 4 SET TERMINAL WIDTH=80 SET TERMINAL WIDTH=132 Changes the width of the terminal within Cray context. Default: 80 SHOW QUEUES Display entries in the dataset staging queue. SHOW QUEUES /ACQUIRE /ALL /CYCLE /JOB Syntax: /NODE=node id /OWNER /SAVE /STAGE /TRANSLATE Qualifiers: /ACQU - display all entries originating from COS (ACQUIRE or FETCH) (default: /ALL) /ALL - display all entries (same as /ACQUIRE/JOB/SAVE) (default: /ALL) /CYCL - cycle the display refresh through all the available information /NOCYC - display only the current page until you enter "+" or "-" (default: /NOCYCLE) /JOB - display entries originating from VMS (default: /ALL) /NODE= - display entries from a specific DECnet node (valid only from an attached station) /OWNER - display only your entries /SAVE - display entries for SAVEd datasets (default: /ALL) /STAGE - display all Cray staging entries

SNAP

/TRAN - display the terminal ID field in the VMS UIC equivalent /NOTRA - display it in the station internal form (default: /TRANSLATE) Remarks: The following fields are displayed: . Position in the staging queue (L is a locked entry i.e., one that is being processed) . Request type (JB=job, AC=acquire/fetch, SV=save) . Queue ID for use in the REQUEUE and RELEASE commands . VAX username of entry owner . Dataset transfer name (job name or dataset name) . Dataset terminal ID (TID) Similar commands: NOS ICF: /STATUS Examples: SHOW QUEUES /OWNER ^-- display all your entries Copy the current contents of the display region into a VMS file. Svntax: SNAP file spec / [NO] ESCAPE Parameters: file spec - VMS file to receive the snapshot

Qualifiers: /ESCAPE - retain escape sequences /NOESCAPE - remove escape sequences (default: /NOESCAPE)

- Remarks: In line-by-line mode, the last display requested is recorded.
- Examples: SNAP snap.job123

STATCLASS Display the current COS job class structure.

| Syntax: | STATCLASS | /[NO]CYCLE |
|-------------|--------------------|---|
| Qualifiers: | /CYCLE /NOCYCLE | cycle the display refresh through all the available information display only the current page until you enter "+" or "-" (default: /NOCYCLE) |
| | | |

Similar commands: NOS ICF: /ICFSTATUS, /STATUS

Examples: STATCLASS

| | 90/09 | 9/ 3 0 I | Rev0 | | Page D-47 |
|---|--------|-----------------|-------|---------|--|
|) | STATUS | (CINT) | Displ | ays the | COS system status. |
| | | Syntax: | | STATUS | /ALL /CLASS=class_id /CYCLE /EXECUTING /HOLD /ID=mainframe_id /INPUT /NODE=node_id /OUTPUT /OWNER /RECEIVING /SENDING /TRANSLATE /VAX |
| | | Qualifie | | | display all COS jobs display jobs and datasets of a specific job class (default: /ALL) |
| | | | | /CYCL | - cycle the display refresh through 11 |
| | | | | /NOCY | <pre>available information - display only the current page until you enter "+" or "-" (default: /NOCYCLE)</pre> |
| | | | | /EXEC | display the execution queue status (default: /EXECUTION) |
| | | | | /HOLD | - display COS datasets in the hold queue |
| | | | | /ID= | display jobs and datasets originating from a specific mainframe |
| | | | | /INPU | - display the input queue status |
| | | | | /NODE= | display the entries for a specific DECnet node |
| | | | | /OUTP | - display the output queue status |
| | | | | /OWNE | - display only your jobs and datasets |
| | | | | /RECE | display the Cray receiving queue status (default: /RECEIVING) |
| | | | | /SEND | display the Cray sending queue status (default: /SENDING) |
| | | | | /TRAN | - display terminal ID field (TID) as the VMS UIC equivalent |
| | | | | /NOTR | - display TID in the station internal form (default: /TRANSLATE) |
| | | | | /VAX | display only COS jobs related to this VAX/VMS station (or network of stations) |
| | | See also | : | ISTATUS | , JSTAT |
| 1 | | Simílar | comma | inds: N | OS ICF: /STATUS |
| | | Examples | : | STATUS | |

Rev0 90/09/30

SUBMIT Stage a VMS file to the COS input queue. Syntax: SUBMIT file spec /AFTER=time /EOF=eof /PRINT SUBMIT f1, f2,... /AFTER=time /EOF=eof /PRINT Parameters: file_spec - single VMS file with a complete COS job fl, f2, ... - two or more files to be combined to create a complete COS job Qualifiers: /AFTER= - specify when the job is to be sent to the Cray /EOF= - specify what represents an end-of-file (e.g., /EOF="E 0 F") (default: /EOF="/EOF") /PRINT - print the job's output file on COS job completion /NOPRINT - put the COS job's output into your VMS file COS jobname.CPR (default: /NOPRINT) Remarks: The file must contain a COS job. By default, the job's output (including the dayfile) is sent to the originating directory. See also: CSUBMI1; RCSUBMIT Similar commands: NOS: CSUBMIT CRAY> SUBMIT myjobl Examples: -or-\$ CRAY SUBMIT myjob1 CRAY> SUBMIT myjob2, myprog2.for, mydata2.dat -or-\$ CRAY SUBMIT myjob2, myprog2.for, mydata2.dat SUPPRESS (CINT) Suppress the echoing of the next typed input line.

Syntax: SUPPRESS

Examples: Cint> SUPPRESS

Page D-49

SWITCH Set or clear COS job sense switches.

Syntax: SWITCH jsq ssw /OFF SWITCH jsq ssw /ON

Parameters: jsq - the COS job sequence number

ssw - the sense switch number (1-6)

Qualifiers: /OFF - turn switch <ssw> off

/ON - turn switch <ssw> on

Remarks: These switches can be used for program synchronization on the Cray.

Examples: CRAY> STATUS <-- to get the jsq CRAY> SWITCH 9876 3 /ON <-- turn on switch 3

Page E-1

٠

***** Appendix E *****

*** References ***

The following manuals describe various features of the Cray, DEC and CDC systems.

** Cray **

| SR-0009 | Fortran (CFT) Reference Manual |
|---------|---------------------------------------|
| SR-0011 | COS Version ! Reference Manual |
| SR-0013 | UPDATE Reference Manual |
| SR-0018 | CFT77 Reference Manual |
| SV-0020 | DEC VAX/VMS Station Reference Manual |
| SR-0035 | CDC NOS Station Reference Manual |
| SR-0039 | COS Message Manual |
| SR-0060 | Pascal Reference Manual |
| SR-0066 | SEGLDR Reference Manual |
| SR-0113 | Programmer's Library Reference Manual |
| | |

** DEC **

| AA-001AE-GZ | DCL Dictionary |
|-------------|--|
| AA-LA16A-TE | EDT Reference Manual |
| AA-LA62A-TE | EVE Reference Manual |
| AA-D034E-TE | VAX Fortran Language Reference Manual |
| aa-la98a-te | VAX/VMS User's Manual |
| | Introduction to VAX/VMS by Terry Shannon |

** CDC NOS **

| 60460420 | NOS | Full Screen Editor |
|------------|-----|---|
| CMLD-88/15 | CDC | NOS Full Screen Editor (FSE) User's Guide |
| 60459680 | NOS | 2 Reference Set Volume 3: System Commands |

** CDC NOS/VE **

| 6046401 8 | NOS/VE Commands and Functions Quick Reference |
|------------------|---|
| 60464015 | NOS/VE File Editor |
| 60485913 | Fortran Version 1 for NOS/VE |
| 6046401 2 | Introduction to NOS/VE |
| 60464014 | NOS/VE System Usage |

Page E-2

** General **

CMLD-87-07 Fortran 77 Extensions - A Comparison CISD-90/01 Computer Center Reference Manual, Volume 1: Cray, MSS, DEC (this manual) CISD-90/02 Computer Center Reference Manual, Volume 2: CDC

Page F-1 .

```
***** Appendix F *****
```

```
*** CCF Computer Systems ***
```

Cray

C1

| Computer: | CRAY X-MP/216 |
|-------------------|---|
| Front-ends: | DEC VAXcluster (station code version 4.01), |
| | CDC CYBER 180/860 (N1) |
| Links to: | Mass Storage System (N1) |
| Operating system: | COS level 1.17 |
| Services: | batch, interactive |
| Schedule: | 24 hours a day, 7 days a week, except a few hours |
| | Tuesday and Thursday mornings for maintenance |
| Location: | Central site |

Page F-2

Rev0 90/09/30

DEC VAXcluster DT3 (V3) Computer: VAX 8550 CRAY X-MP (C1); CDC CYBER 180/860 with MSS Links to: (N1/MFN); DECnet to NAVSEA (SEAHUB, etc.) Operating system: VMS 5.3-1 Services: batch, interactive Schedule: 24 hours a day, 7 days a week, except a few hours Thursday morning for maintenance Location: Central site Network address: 130.46.1.12 (dtvms3.dt.navy.mil) DT4 (V4)VAX 8550 Computer: Links to: CRAY X-MP (C1); CDC CYBER 180/860 with MSS (N1/MFN); DECnet to NAVSEA (SEAHUB, etc.) Operating system: VMS 5.3-1 Services: batch, interactive Schedule: 24 hours a day, 7 days a week, except a few hours Thursday morning for maintenance Location: Central site Network address: 130.46.1.10 (dtvms.dt.navy.mil or dtvms4.dt.navy.mil) Secure DEC VAX SECURE VAX 6410 Computer: CRAY X-MP (C1) Links to: Operating system: VMS 5.3-1 secure batch, secure interactive Services: Schedule: 24 hours a day, 7 days a week, except a few hours for maintenance Central site Location: Control Data Corporation MFN (N1) Computer: CDC CYBER 180/8604 with Mass Storage System Cray Station ID: N1 Links via NOS to: CRAY X-MP (C1) Links via NOS from: CRAY X-MP (C1), DEC VAXcluster Operating systems: dual state with . NOS version 2.7.1 level 716 . NOS/VE version 1.5.1 level 739 Services: trillion-bit storage, local and remote batch, interactive Schedule: 24 hours a day. 7 days a week, except a few hours for maintenance Location: Central site Network address: 130.46.1.16 (cdc860.dt.navy.mil)

Page F-3

OASYS (Office Automation SYStem) composed of: OASYS Computer: Sequent S27 Links to: Mass Storage System Operating system: DYNIX v3.0.17.9 (BSD 4.2 + some 4.3 + some AT&T System V) OASYS (Office Automation) Services: 24 hours a day, 7 days a week, except a few hours Schedule: Wednesday night for backups Location: Central site Network address: 130.46.1.53 (oasys.dt.navy.mil) DTOA1 DEC VAX 11/780 Computer: Links to: Mass Storage System Operating system: Ultrix-32 Services: OASYS (Office Automation - primarily Carderock) Schedule: 24 hours a day, 7 days a week, except a few hours Thursday morning for maintenance Location: Central site Network address: 130.46.1.2 (dtoal.dt.navy.mil) DTRC Computer: DEC VAX 11/780 Links to: Mass Storage System Operating system: Ultrix-32 Services: OASYS (Office Automation - primarily Carderock) Schedule: 24 hours a day, 7 days a week, except a few hours Thursday morning for maintenance Central site Location: Network address: 130.46.1.3 (dtrc.dt.navy.mil) DTOA3 Computer: DEC VAX 11/780 Links to: Mass Storage System Operating system: Ultrix-32 CASYS Office Automation - primarily Annapolis) Services: 24 hours a day, 7 days a week, except a few hours Schedule: Thursday morning for maintenance Location: Central site Network address: 130.46.1.4 (dtoa3.dt.navy.mil)

Page F-4

*** *** Services and Support (301) 227-1910 Accounting for Computer Services: Code 3502 (301) 227-3043 Computer status (recorded message) (301) 227-1967 Dispatch desk (301) 227-1907 Manuals Cardercck: (301) 227-4901 Microcomputer support Annapolis: (301) 267-4987 (301) 227-1967 Tape Librarian (301) 227-1907 Training User Services (Scientific and Engineering Carderock: (301) 227-1907 User Support Branch - Code 3511) Annapolis: (301) 267-3343 Stan Willner (Head) Sharon Good Mike Kass Ed Kennedy Brenda Peters Dave Sommer (Annapolis) Administrative Personnel Computer & Information Services Department (G. Gray) (301) 227-1270 35 3501 Assistant for Technical Development and (301) 227-1428 Operations (L. Minor) (301) 227-1361 3502 Computer Department Business Office (301) 227-3454 3509 Administrative Office (D. Braxton) 351 Scientific & Engineering Systems Div. (S. Willner) (301) 227-1907 3511 S&E User Support Branch (S. Willner) (301) 227-1907 (301) 227-3642 3512 VAX/VMS Systems Branch (M. Brady) (301) 227-1271 3513 Cray/CDC Systems Branch (J. Wessel) 353 Office Automation Systems Division (R. Yearick) (301) 227-1428 (301) 227-1428 3531 Unix Systems and Programming Branch (R. Yearick) (301) 227-4901 3533 OA/Microcomputer Support Branch (P. Hayden) (301) 267-4987 (301) 227-1184 355 Information Systems Division (E. Kearney) (301) 227-1127 3551 Business Systems Branch (B. Crum) 3552 Special Project Branch (D. Singla) (301) 227–1184 357 Communications and Facilities Division (R. Weachter) (301) 227-1270 3571 Computer Facilities Branch (R. Weachter) (301) 227-3937 3572 Networks and Communications Branch (T. Smith) (301) 227-1400

Page G-1

***** Appendix G *****

*** Internal Data Structure ***

1. The following table summarizes word lengths on various computers:

| computer | op sys | bits/word | digit | ts/word | characters/word |
|--------------------------------|---------------------------|------------|-------|------------|---------------------------|
| CRAY X-MP | | 64 | 22 | octal | 8 |
| CDC CYBER 200 CDC CYBER 180 | NOS/VE | 64 | 16 | hex | 8 |
| CDC CYBER 180 CDC CYBER 170 | NOS & NOS/BE NOS/BE | 60 | 20 | octal | 10 |
| DEC VAX (when used in | Fortran) | 16) 32 | | hex hex | 2 4 |
| IBM | | 32 | 8 | hex | 4 |
| Burroughs 7700 | | 48 | 12 | hex | 6 |
| Unisys 1100 | | 36 | 12 | octal | 4 (ASCII) 6 (Fieldata) |

This affects the conversion of programs in four areas:

- a. The degree of precision of operations is different. Therefore, convergence factors may need to be increased or decreased in absolute value.
- b. Constants and data may need to be changed.
- c. Octal and hexadecimal constants used in masking operations are generally affected and require alteration according to their intended use.
- d. Since different computers may store a different number of characters per word, DATA statements that store a string of Hollerith characters may position the characters in different relative positions in different words. All variable formats (whether read in as data or created by the programmer) should be checked. Better yet, Fortran programs which store Hollerith data in INTEGER or REAL variables should be changed to use the Fortran 77 CHARACTER variables and never need to worry about this problem again. (You may have to worry about the maximum length of a CHARACTER variable, but not how it is stored.)

2. Internal representation of character data is ASCII in the CRAY X-MP and DEC VAX, Display Code in the CDC CYBER, and ASCII, EBCDIC or internal BCD in some other systems.

| CHARACTER string | machine | op sys | internal representation |
|------------------|---|---------------|---|
| ' ' (1 blank) | CRAY X-MP CDC 170 CDC 180 CDC 180 DEC VAX | NOS NOS/VE | * oct 20 hex 55 55 20 20 |
| '0' (1 zero) | CRAY X-MP CDC 170 CDC 180 DEC VAX | NOS | * oct 30 hex 33 30 30 |
| 'FILE48' | CRAY X-MP CDC 170 CDC 180 CDC 180 DEC VAX | NOS NOS/VE | <pre>* oct 46494C463438 hex 061014053743 061014053743 46494C453438 3834454C4946 (8 4 E L I F)</pre> |

 * - the octal representation depends on the position in the word

| Hollerith words | machine | op sys | internal machine repres | sentation |
|-------------------|-----------|--------|---|-----------|
| <blanks></blanks> | CRAY X-MP | | 0200401002004010020040 | oct |
| | | | 2020202020202020 | hex |
| | CDC 170 | | 555555555555555555555555555555555555555 | oct |
| | CDC 180 | NOS | 555555555555555555555555555555555555555 | oct |
| | CDC 180 | NOS/VE | 2020202020202020 | hex |
| | DEC VAX | | 20202020 | hex |
| <zeroes></zeroes> | CRAY X-MP | | 0300601403006014030060 | oct |
| | | | 303030303030303030 | hex |
| | CDC 170 | | 3333333333333333333333333 | oct |
| | CDC 180 | NOS | 3333333333333333333333333 | oct |
| | CDC 180 | NOS/VE | 303030303030303030 | hex |
| | DEC VAX | | 30303030 | hex |
| FILE48 | CRAY X-MP | | 0431112304246416020040 | oct |
| | | | 46494C4534382020 | hex |
| | CDC 170 | | 06101405374355555555 | oct |
| | CDC 180 | NOS | 06101405374355555555 | oct |
| | CDC 180 | NOS/VE | 46494C453438202C | hex |
| | DEC VAX | , | 454C4946 20203834 | hex |
| | | | (ELIF 84) < | - 2 words |

Page G-3

- 3. The character sequence for the CRAY X-MP, DEC VAXcluster and CDC 180 (NOS/VE) is ASCII. Note that numbers precede letters for alphabetic comparisons. The character sequences for CDC 180 (NOS) at DTRC is Display Code (64-character set). CDC NOS Fortran uses the Display Code sequence (letters before numbers); CDC NOS COBOL uses the ASCII6 sequence (numbers before letters). Cray, DEC VAX and CDC NOS/VE use the ASCII sequence.
- 4. CDC NOS uses some special bit configurations in floating point arithmetic to indicate indefinite and infinite operands. These errors could be caused by referencing program areas not initialized or areas overwritten due to inadequate storage reservation. The CPU will not do any further calculation if it encounters such a number and the job will abort with an error mode 2 or 4.

| + | infinity | 3777xxxxxxxxxxxxxx oct |
|---|------------|---|
| - | infinity | 4000xxxxxxxxxxxxxx |
| + | indefinite | 1777xxxxxxxxxxxxxx |
| - | indefinite | 6000xxxxxxxxxxxxxx |
| | | where 'x' is any octal digit, usually 0 |

5. CDC NOS/VE uses several exponents in floating point arithmetic to indicate zero:

+ zero 0xxx, 1000 thru 2FFF hex - zero 8xxx, 9000 thru AFFF

6. CDC NOS/VE uses special exponents in floating point arithmetic to indicate indefinite and infinite operands:

| + | infinity | D000 thru EFFF hex | |
|---|------------|--------------------------------|------|
| - | infinity | 5000 thru 6FFF | |
| + | indefinite | 7xxx | |
| - | indefinite | Fxxx | |
| | | where 'x' is any hexadecimal d | igit |

| Page | G-4 |
|------|-----|
|------|-----|

7. The word format of integers and floating point numbers differs on the various computers.

| CRAY X-MP 1, 1.0 -1, -1.0 2, 2.0 4, 4.0 | integer 00000000000000000000000000000000000 | floating point 040001400000000000000 401800000000000 1400014000 | hex oct hex oct hex |
|---|--|--|---------------------------------|
| DEC VAX | | | |
| 1, 1.0 | 0000001 | 00004080 | hex |
| -1, -1.0 | FFFFFFF | 0000C080 | |
| 2, 2.0 | 0000000 2 | 00004100 | |
| 4, 4.0 | 0000004 | 00004180 | |
| CDC CYBER (NOS) | | | |
| 1, 1.0 | 000000000 0000000001 | 1720400000 0000000000 | oct |
| -1, -1.0 | 7777777777 7777777776 | 6057377777 7777777777 | |
| 2, 2.0 | 000000000 000000000 2 | 1721400000 0000000000 | |
| 4, 4.0 | 000000000 000000004 | 1722400000 0000000000 | |
| CDC CYBER (NOS/ | VE) | | |
| 1, 1.0 | | 4001800000000000 | hex |
| -1, -1.0 | FFFFFFFFFFFFFFF | C00180000000000 | hex |
| 2, 2.0 | 000000000000000 2 | 4002800000000000 | hex |
| 4, 4.0 | 000000000000004 | 4003800000000000 | hex |
| Note the differ floating point) | ence in the format of neg numbers: | ative integers (and CYB | ER |
| CRAY X-MP. | DEC VAX. CDC NOS/VE | CDC NOS | |

| CRAY X-MP, DEC VAX, CDC NOS/VE | CDC NOS |
|---------------------------------------|---------------------------------------|
| | |
| two's complement of absolute value | one's complement of absolute value |

8. Logical variables are represented by:

| | CRAY X-MP, CDC | DEC VAX |
|-------|----------------|------------|
| TRUE | -1 | 1 in bit 0 |
| FALSE | 0 | 0 in bit 0 |

9. By default, your program area in central memory is set as follows:

| Cray CO | os | Cray UNICOS | DEC VMS | CDC NOS | CDC NOS/VE |
|---------|-----|-------------|---------|---------|------------|
| | | | | | |
| zero | ז'ר | Zero | zero | zero | zero |

Page G-5

Page G-6

Rev0 90/09/30

*** *** Internal Representation $\star\star$ CRAY X-MP ** Words in the CRAY X-MP are 64 bits long. Bits are numbered 0-63 or 63-0. Integer: bit 0 - the sign bit (0 = positive; 1 = negative) (23) bits 1:23 - the absolute value of the integer (22:0) $- \sim -10^{*}14$ to $\sim 10^{*}14$ range Integer (CFT, INTEGER=64): bit 0 - the sign bit (0 = positive; 1 = negative) (63) bits 1:63 - the absolute value of the integer (62:0) $- \sim -10^{**}19$ to $\sim 10^{**}19$ range bit (63) Real: 0 - the sign of the number bits 1:15 - the exponent (2000 bias) (62:48)bits 16:63 - the mantissa (47:0) - ~10**-2466 to ~10**2465 range precision $- \sim 14$ decimal digits Double: First word: bit 0 - the sign of the number (63) bits 1:15 - the exponent (2000 bias) (62:48) bits 16:63 - the high order part of the mantissa (47:0) Second word: bits 0:15 - unused (63:48)bits 16:63 - the low order part of the mantissa (47:0)range $- \sim 10^{**} - 8193$ to $\sim 10^{**} - 8189$

precision $- \sim 29$ decimal digits

Page G-7

** DEC VAX ** Bytes in the DEC VAX are 8 bits long with bits are numbered 7-0. A word (INTEGER*2 in Fortran) is 16 bits long (15-0). A long ord (INTEGER or INTEGER*4) is 32 bits long (31-0). Word (INTEGER*2): bit 15 - the sign bit (0 = positive; 1 = negative) bits 14:0 - the absolute value of the integer - -32,768 to 32,767 range Longword (INTEGER*4): - the sign bit (0 = positive; 1 = negative) bit 31 bits 30:0 - the absolute value of the integer - -2,147,483,648 to 2,147,483,647 range F float (REAL*4): bit 15 - the sign of the number bits 14:7 - the exponent (excess 128) bits 6:0 and 31:16 - the mantissa - ~.29*10**-8 to ~1.7*10**38 range precision $- \sim 7$ decimal digits D float (REAL*8, DOUBLE PRECISION): bit 15 - the sign of the number bits 14:7 - the exponent (excess 128) bits 6:0 and 63:48 and 47:32 and 31:16 - the mantissa range - ~.29*10**-8 to ~1.7*10**38 precision $- \sim 16$ decimal digits G_float (FORTRAN/G_floating): bit 15 - the sign of the number bits 14:4 - the exponent (excess 1024) 3:0 and bits 63:16 - the mantissa - ~.56*10**-308 to ~.9*10**308 range precision $- \sim 15$ decimal digits H_float (REAL*16): bit 15 - the sign of the number bits 14:0 - the exponent (excess 16,384) bits 127:16 - the mantissa - ~.84*10**-4932 to ~.59*10**4932 range precision $-\sim 33$ decimal digits

Page G-8

Rev0 90/09/30

** ** CDC CYBER (NOS, NOS/BE) Words in the CDC CYBER 170 and CYBER 180 (when running NOS or NOS/BE) are 60 bits long. Bits are numbered 59-0. Integer: bit 59 - the sign bit (0 = positive; 1 = negative) bits 58:0 - the absolute value of the integer Integer: bit 59 - the sign bit (0 = positive; 1 = negative) bits 47:0 - the absolute value of the integer (if used in multiplication or division) Real: bit 59 - the sign of the number bits 58:48 - the exponent (2000 bias) bits 47:0 - the mantissa with the binary point after bit 0 Double: (Double precision is performed in the software, not in the hardware) First word: bit 59 - the sign of the number bits 58:48 - the exponent (2000 bias) bits 47:0 - the high order part of the mantissa with the binary point after bit 0 Second word: bit 59 - the sign of the number bits 58:48 - the exponent (2000 bias) bits 47:0 - the low order part of the mantissa with the binary point after bit 0

Page G-9

** CDC CYBER (NOS/VE) **

Words in the CDC CYBER 180 (when running NOS/VE) are 64 bits long. Bits are numbered 0-63.

```
- the sign bit (0 = positive; 1 = negative)
Integer: bit 0
          bits 1:63
                      - the absolute value of the integer
          precision
                      - \sim 19 decimal digits
Real:
          bit O
                      - the sign of the number
                      - the exponent (4000 bias)
          bits 1:15
                         1:3 - the following FP (or DP) numbers
                                00x - FP zero
                                0x0 - FP zero
                                011 - standard FP number
                                100 - standard FP number
                                101 - FP infinity
                                110 - FP infinity
                                111 - FP indefinite
          bits 16:63 - the mantissa with the binary point before bit 16
                      -4.8*10**(-1234) to 5.2*10**(1232)
          range
          precision
                      - \sim 14 decimal digits
Double:
          First word:
          bit O
                      - the sign of the number
                      - the exponent (4000 bias)
          bits 1:15
                         1:3 - same as for real
          bits 16:63 - the high order part of the mantissa with the
                        binary point before bit 16
          Second word:
          bit 64
                      - same at bit 0
          bits 65:79 - same as bits 1:15
          bits 80-127 - the low order part of the mantissa with the
                        binary point after bit 0
                      - 4.8*10**(-1234) to 5.2*10**(1232)
          range
                      -\sim 29 decimal digits
          precision
```

Page G1-1

.

***** ***** Glossary Alphabetic (CDC - NOS) The letters A-Z. Alphabetic (CDC - NOS/VE) The letters A-Z, a-z. Alphabetic (Cray - COS) , , , , , and the letters A-Z, a-z. Alphabetic (DEC) \$, _ (underscore), and the letters A-Z, a-z (upper and lower case are the same). Alphanumeric Alphabetic and the digits 0-9. User initials (userid or username) The 4-character ID assigned to each user by Code 3502. This is used to identify jobs, for charge authorization, to identify permanent and MSS files, magnetic tapes, etc.

-Acc Page Index-1

***** Index ***** Note - NOS/BE system control statements are flagged with ". Intercom commands are flagged with @. UPDATE directives begin with *. Compiler options are flagged with \$, Primary references are flagged with an asterisk after the page number, for example, $1-1^*$. 3-2-1, C-5 * (comment) ! (comment) D-1 \$ (create VMS subprocess) 3-1-12 D-32 + (display next) - (display previous) D-32 @ (execute command file) 3-1-12, D-32* S (Execute DCL command) 3-1-12 @ (invoke procedure) D-1 3-1-12 - (last page) 3-1-12 + (next page) 3-1-11 ! (prompt) \$ (subprocess) D-32 1-3-1. 7-1-1* ABAQUS 3-3-3, 3-6-3, C-3, C-31, C-33 Abort ABORT 3-1-12, 3-1-16, 3-6-3, D-33* 3-6-3 ABS 3-6-1*, 3-6-3, G-1, G-4 Absolute ABTCODE 3-2-6 1-2-6, 3-1-1, 4-1-4, 4-1-10, 4-1-11, Access 5-1-1*, C-40, C-41, C-44, C-51, D-13 3-1-1, 3-1-10, 3-2-2, C-5*, C-6 ACCESS C-2*, C-3* Access mode C-3* Access, unique 4-1-8 Account ACCOUNT 1-2-6, 3-1-2, 3-1-3, 3-1-5, 3-1-11, 3-1-15, 3-2-1, C-5*, D-29 C-7, C-41, C-55, D-11 Account number 1-2-2* Account number, alternate 1-2-2*, C-47 Account number change F-4 Accounting

Page Index-2 Acc-AUD

)

Rev0

90/09/30

1-2-2* Accounts, multiple 3-2-2, 4-1-3*, C-6* ACQUIRE 1-3-1 ACSL 7-1-18 AC/DC analysis 1-1-2, 1-1-3, 5-1-8 Address, ethernet 5-1-5 Address, host F-2, F-3 Address, network 3-2-2, C-7* ADJUST F-4* Administrative personnel 1-2-1*, 1-2-8 ADP Control Center 7-1-13 Algebra, linear 1-3-1 ALGOL 3-6-3 ALIGN 6-1-4, D-2* ALLOCATE G1-1 Alphabetic G1-1 Alphanumeric 3-2-1, C-7* ALTACN 1-2-2* Alternate account number C-2* AM 3-3-3 Ampersand 1-2-1, D-6, D-10, F-4 Annapolis 6-1-1 ANSI standard label 1-3-1 APL 3-3-3, C-4* Apostrophe Applied mathematics 7-1-9 3-1-17 APPSW D-6* APRINT 1-3-1 APT 3-2-7 Arithmetic operator 3-2-7, 6-1-1, 6-1-4, A-1*, A-2*, ASCII A-3*, A-4*, G-2, G-3 3-2-2, 5-1-3, C-7*, C-12 ASSIGN 3-6-2 Asterisk D-1 At sign 3-1-12, D-33* ATTACH D-35 Attention 3-1-12, 3-1-16, D-34* ATTENTION C-41 Attribute, file Attribute, record D-10 4-1-4, 4-1-10 Attributes, file 4-1-9 Attributes, link 4-1-7, C-10, C-40, D-23 Audit 3-2-3, C-8* AUDIT 4-1-4, 4-1-10 Audit files 3-2-5, C-10* AUDPL

90/09/30 Rev0 Aut-C Page Index-3 Automatic logout 5-1-2 D-7* AUX D-7, D-8 Auxiliary printer D-8* AUXPRINT 4-1-2*, 4-1-7, 6-1-1, C-42, C-45, Backup C-46 6-1-4 BACKUP 1-3-1 Basic 1-1-1, 2-1-1, 3-1-1, 3-1-2, 3-1-3, Batch 3-1-4, 3-1-5, 3-1-8, 3-1-10, 5-1-4*, F-1, F-2 Batch job 3-1-3 3-1-6*, 3-1-8*, 5-1-4* Batch jobs, killing 6-1-1, A-3*, A-4*, G-2 BCD BEFORE 3-4-2 3-6-3, 3-6-10 BIN Binary 6-1-1 C-2 Binary blocked G-8, G-9 Binary point \$BLD C-55 BLOCK 3-2-3, C-11* Block data 3-6-14 6-1-1, 6-1-4 Blocked Blocked, binary C-2 Blocked, character C-2, C-43, C-45 C-11, C-22, C-23, C-57, C-61, C-64 Blocked dataset Blocked file C-22, C-57 C-23, C-58 Blocked record 3-1-4*, C-37 BUDGET job class C-35 Buffer 3-6-8 Buffer length BUILD 3-2-5, 3-5-1*, C-12* 5-1-2 Bulletin BYE 3-1-12, 3-1-16, D-34* ^C 3-1-12 1-3-1 С ^C 3-1-12 С 7-1-2*

1

| Page Index-4 C -CHA | | Rev0 | 90/09/30 |
|---|--------------------------|------------------------|--------------------|
| ^C (CTRL-C) | D-35* | | |
| Calcomp | 1-3-1 | | |
| Calcomp plot | 1-2-8 | | |
| CALL | | , 3-3-1, 3-3- | 5. 3-4-2. |
| | C-14* | , , | , - , , , , |
| Call by name | 3-2-4, 3-2-5 | 5, 3-3 -1, C-15 | i |
| CAPRINT | D-10* | | |
| Card interpreter | 1-2-8 | | |
| Card punch | 1-2-8 | | |
| Card verifier | 1-2-8 | | |
| Carderock | 1-2-1, 4-1-2 | 2, F-4 | |
| Caret | C-4* | | |
| Carriage control | D-10 | | |
| Carriage control, Fortran CASE | D-44 3-6-3 | | |
| Case, upper | 3-4-1 | | |
| CATLIST | 4-1-10 [*] | | |
| Caution | 3-6-1 | | |
| Caution, segmentation | 3-6-15 | | |
| cc | 1-3-1 | | |
| CCF | 5-2-1, D-1, | F-1 | |
| CCRM | E-2 | | |
| CDC | E-1, G1-1 | | |
| CDC CYBER 860 | 1-1-2 | | |
| CDC CYBER 860 (NOS) | 1-3-1 | | |
| CDC CYBER 860 (NOS/VE) | 1-3-1 | | |
| CDC output | D-10 | | |
| cdc860 | 1-1-2, F-2 | | |
| CDD CDEFINE | 1-3-1 1-4-2, 3-1-8 | 2 | |
| CDROP | 3-1-8 | , | |
| Central processing unit | 2-1-1, 3-1-1 | 1, 5-1-1 | |
| Central site | 3-1-8, F-1, | | |
| Central Site | 1-2-1* | - | |
| Central Site operator | 5-1-2 | | |
| Certify tape | 6-1-2 | | |
| CFT | 3-2-4, C-15* | | |
| CFT77 | 3-2-4, C-19 [*] | *, E-1 | |
| Change | C-39, C-56 | | |
| CHANGE | 4-1-1, 4-1-2 | 2 | |
| Change access password | 1-2-6 | | |
| | 1 2 2 * | | |
| Change account number CHANGE LINK ATTRIBUTES | 1-2-2* 4-1-9* | | |

CHANGE TAPE LABEL ATTRIBUTES 1-5-1, 6-1-7 CHARACTER G-1, G-2 Character blocked C-2, C-43, C-45 3-6-3 Character conversion 3-4-1 Character, master Character set, ASCII A-1 Character set, CDC A-3 C-7 Characteristic CHaracteristic C-55 Characteristic, dataset C-39 C-27 Characteristic, disposition 1-2-1, 4-1-1, 4-1-2, 4-1-8 Charge 3-2-1, C-20* CHARGES CINT 3-1-1, 3-1-2, 3-1-11, 3-1-15, D-31* CKILL 3-1-8 Class, job 3-1-4 3 - 1 - 4Class, service 3-1-4 Classified processing 1-2-8, 6-1-2 Clean magnetic tape CLEAR 3-1-12, D-34* 7-1-3* CMS CNEWCHRG D-11* 1-2-6, 3-1-2, D-13* CNEWPW 1-3-1 Cobo1 COBOL G-3 3-1-1 Code, disposition 7-1-3* Code Management System 6-1-1 Coded 3-1-12, D-35* COLLECT .COM 5-3-1 COMDECK 3 - 4 - 1D-1* Command D-35, D-37, D-41, D-42 Command file Command, station D-30 3-4-3, 3-6-1, 3-6-2, C-5, D-1 Comment COMMENT 3-1-12, D-35* Common 3-6-7 Common block 3-6-3, 3-6-4, 3-6-10, 3-6-15 COMMONS 3-6-4, 3-6-10, 3-6-14 Communication 4-1-9 COMPARE 3-2-3, C-21* 3-4-2, C-15, C-19, C-50 Compile COMPILE 3-4-3 1-2-8 Complaints

G-4

CHA-Com

Page Index-5

20/09/30

Complement, one's

Rev0

90/09/30 Page Index-6 Com-Cra Rev0 G-4 Complement, two's 3-2-7 Completion code Complex procedure 3-3-1, 3-3-5 5-2-4, 5-4-2, 5-5-3 Compress a library 1-2-1*, 4-1-1, 4-1-2, 5-2-1 Computer Center E-2 Computer Center Ref Man Computer service F-4 F-1 Computer Systems C-34 Conditional CONNECT 3-1-16 Constant G-1 Context, Cray D-30, D-31, D-32*, D-38, D-45 Continuation 3-2-8, 3-6-2, C-4* Control statement 3-1-3*, 3-3-1, 5-3-1 Control word, maintenance C-3 C-3 Control word, read C-3 Control word, write D-35* CONTROL C D-36* CONTROL O D-36* CONTROL Z G-1 Convergence factor Conversion G-1 Conversion, character 3-6-3 C-11, C-61, D-18 Convert Copy D 46 COPY 1-5-1, 3-4-2, 3-4-3, 3-6-11 COPYD 3-2-3, C-22* 6-1-4 COPYD2T 3-2-3, C-22* COPYF COPYNF 3-2-3, C-23* COPYR 3-2-3, C-23* COPYT2D 6-1-4 COPYU 3-2-3, C-24* 3-1-1*, E-1, F-1, G1-1 COS 3-1-10 COS input queue COS level 3-2-6 C-37 Cost, job 3-2-6 Counter .CPR 3-1-5 2-1-1, 3-1-1, 5-1-1 CPU Cray 1-3-1, 3-1-15, 4-1-3, D-11, D-13, D-14, D-23, D-24, D-25, D-27, E-1, F-1, G1-1 CRAY 5-2 1, D-1, D-30* Cray context 3-1-11, D-30, D-31, D-32*, D-38,

D-45 Cray context CRAY DROP 3-1-6 CRAY KILL 3-1-6 Cray station E-1 3-1-11, 3-1-12 Cray Station 1-1-2*, 1-1-3 Cray station ID CRAY STATUS 3-1-6 3-1-1, 3-1-2, 3-1-5, D-29 CRAY SUBMIT CRAY X-MP 1-1-2, 2-1-1*, 3-1-1*, F-1, G-6 CRAYDEF 1-4-2, 3-1-8 Create C-7 3-5-2, 5-2-3, 5-4-1, 5-5-1 Create a library Critical file 4-1-2 C-33 Cross-reference 3-1-3*, 3-3-1* \$CS CSTATUS 3-1-8 3-1-1, 3-1-2, 3-1-5, 3-1-8, D-14*, CSUBMIT D-29 CSUBMIT/NUPW 1-2-6* CTRL-C (^C) D-35* D-36* CTRL-O (^0) CTRL-Z 5-2-6 CTRL-Z (^Z) D-36* CWEOF 3-4-2 3-1-1, 3-1-8, 3-1-16, 5-1-6, 6-1-6, **CYBER 860** 6-1-7, D-20, D-22, F-2, G-8, G-9 C1 1-1-2*, F-1 Data G-1, G-2 3-3-2, C-24* DATA Database management system 7-1-4 7-1-11 Database mgt sys, relational Dataset 3-1-1, C-11, C-21, C-27, C-28, C-34, C-53, C-54, C-60, D-11, D-23, D-24, D-25, D-27 DATASET 3-1-13, D-36* C-22, C-23, C-57, C-61, C-64 Dataset, blocked Dataset characteristic C-39 3-2-2 Dataset definition C-2* Dataset format Dataset, library C-36 3-1-1, C-27, C-28, C-55 Dataset, local 3-1-1, 4-1-3, C-3, C-4, C-5, C-8, Dataset, permanent C-26, C-51, C-55, D-36

Page Index- Dat

Rev0

90/09/30

C-7 Dutaset size C-24, C-58, C-61 Dataset, unblocked 1-3-1 Datatrieve DataTrieve (DTR) 7-1-4* DATE 3-2-6 Dayfile 1-2-8 1-1-3, 1-3-1, 3-1-4 DBL07 DBMS 1-3-1 C−2* DC DCL D-1*, D-38, E-1 C-24*, C-30 DDA 1-1-5, 5-1-5, 5-1-7 DDN DEALLOCATE 6-1-4, D-2* Debug 3-6-4, C-56 DEBUG 3-2-3, C-25*, C-30 5-1-1*, E-1, F-2, G1-1 DEC 6-1-4, G-7 DEC VAX 1-1-3, 5-1-1*, F-2 DEC VAXcluster DECalc 1-3-1 DECK 3 - 4 - 1DECnet 5-1-5 3-1-4*, C-37 DEFER job class Define D-44 4-1-2, 5-1-3, 5-2-6 DEFINE 3-6-4 DEFLIB 1-2-8 Degauss magnetic tape 3-1-13, D-37* DELAY 4-1-7, D-24, D-39, D-43 Delete 3-2-2, 3-4-2, C-26* DELETE 3-2-7, 3-2-8, 3-3-3 Delimiter Demand 1-1-1 D-18* DETAB DETACH FILE 6-1-7 C-2* DF 1-1-5 Diagram, network G1-1 Digit 4-1-2, 4-1-4, 4-1-6 Direct file Directive, BUILD 3-5-1*, C-12 3-6-2* Directive, SEGLDR 3-6-10* Directive, segmentation 3-4-1* Directive, UPDATE DIRECTORY 5-1-3, 5-1-4 D-36 Discard DISCARD 3-1-13, 3-1-16, D-37* Disconnect 5-1-2

DISMOUNT 6-1-4, D-3* Dispatch F-4 Display code A-1, A-2, A-3, A-4, G-2, G-3 Display region D-46 DISPLAY BACKUP LABEL TAPE 6-1-7 DISPLAY LINK ATTRIBUTES 4-1-9* DISPLAY TAPE LABEL ATTRIBUTES 6-1-7 Dispose C-45 DISPOSE 3-1-8, 3-2-2, 4-1-3*, C-27* Disposition C−2* 3-1-1 Disposition code DISSPLA 1-3-1, 7-1-5* DN **C−2*** Document 5-5-1 Double precision G-6, G-7, G-8, G-9 DRD 3-6-4 DROP 3-1-13, D-37* DS 3-2-3, C-28* DSDUMP C-28* DTLIB 3-5-1, 5-2-1, 5-4-1, 5-5-1, 7-1-6*, D-1 DTLIBCRAY 5-5-1 DTn 5-1-6 DTNET 1-1-2, 1-1-4*, 1-1-5, 5-1-1, 5-1-2 dtoal F-3 • DTOA1 (OASYS) F-3 dtoa3 F-3 DTOA3 (OASYS) F-3 DTR 7-1-4* dtrc F-3 3 - 2 - 1DTRC DTRC (OASYS) F-3 dtvms F-2 F-2 dtvms3 dtvms4 F-2 DT1 1-1-3* 1-1-3*, 1-3-1, F-2 DT3 DT4 1-1-3*, 1-3-1, F-2 DT.NAVY.MIL 5-1-6, 5-1-8 dt.navy.mil F-2, F-3 Dual state F-2 4-1-9 Dual-state 3-2-3, C-24, C-25, C-28, C-33 Dump DUMP 3-2-3, C-30* **DUMP JOB** 3-2-3, C-30, C-30*

Rev0 90/09/30 Page Index-10 DUP-Epi DUP 3-6-11 DUPENTRY 3-6-4 3-6-5 Dynamic DYNAMIC 3-6-4 Dynamic Dump Analyzer (DDA) C-24 DYNA3D 1-3-1, 7-1-7* G-7 D_float EAM facilities 1 - 2 - 86-1-1, 6-1-4, A-1*, A-2*, A-3*. A-4*. EBCDIC G-2 D-48 Echo ECH0 3-2-1, 3-6-5, C-31* C-2* ED 3-4-4 EDIT Edition C−2 Editor 5-1-4, 5-6-1*, E-1 Editor (EDIT FILE) E-1 Editor (EDT) 5-1-2, 5-1-4, 5-6-1, E-1 Editor (EVE) 5-1-2, 5-1-4, 5-6-2, E-1 Editor (FSE) E-1 Editor (TPU) 5-1-4, 5-6-1 EDIT FILE 1-4-2 EDIT FILE (editor) E-1 5-1-2, 5-1-4, 5-6-1*, E-1 EDT (editor) EDTINI.EDT 1 - 4 - 11-3-1 EISPACK 3-6-9 Eject ELSE 3-2-4, C-34 3-2-4, 3-2-6, C-34 ELSEIF 3-1-16 ENDCONNECT 3-2-4, C-34 ENDIF 3-2-4, C-38* ENDLOOP ENDPLAY 3-1-16 3-2-4, 3-3-1, 3-3-2, C-51 ENDPROC 3-6-11, 3-6-14 ENDSEG ENDTREE 3-6-11, 3-6-13, 3-6-14 3-1-3, 3-1-11, D-37 End-of-file End-of-record A-4 3-6-4, 3-6-5, 3-6-8, 3-6-9 Entry point 1-5-1, 3-1-3*, 3-1-13, 3-1-16, 3-4-2, EOF D-37* EPILOG 1-4-2 Epilog file 1-4-1*

EQUIV 3-6-5 C-2* ERR 3-6-1, 3-6-3, C-39 Error Error code checking 3-2-6 1-1-5 Ethernet 1-1-2, 1-1-3, 5-1-8 Ethernet address 5-1-2, 5-1-4, 5-6-2*, E-1 EVE (editor) EVESINIT.EVE 1 - 4 - 1Exclamation mark D-1 C-15, D-42 Execute D-36 Exit EXIT 3-1-11, 3-1-13, 3-2-1, 3-3-1, C-31*, D-38* 3-2-4, 3-2-6, C-34 EXITIF 3-2-4, 3-2-6, C-38* EXITLOOP C-2* **EXO** 1-2-6 Expire (password) G-6, G-7, G-8, G-9 Exponent 3-2-6 Expression, JCL External, unsatisfied 3-6-9 Extract a module 5-2-5, 5-4-3, 5-5-3 G-5 FALSE 1 - 5 - 1FCOPY Fetch C-43 FETCH 3-1-1, 3-1-10, 3-2-2, 4-1-3*, C-6, C-32* Field length 3-2-6, C-38 4-1-1, 4-1-3, 4-1-7, 4-1-9, 4-1-10, File 5-1-3, 5-1-4, 6-1-1, D-11, D-23, D-24, D-25, D-27 File attribute C-41 File attributes 4-1-4, 4-1-10 C-22, C-57 File, blocked File, command D-35, D-37, D-41, D-42 4-1-2 File, critical File, direct 4-1-2, 4-1-4, 4-1-6 5-1-2, 5-1-4 File, journal File, local C-52 C-55 File, overwrite 1-2-2, 4-1-3, 4-1-9, 5-1-2 File, permanent C-41 File, rename 5-1-6 File, transfer File transfer 7-1-12

Page Index-12 Fil-Glo

Rev0 90/09/30

| File Transfer Protocol (FTP) | 5-1-6 |
|---|---|
| FILEMANAGER | 6-1-4 |
| Files | C-40 |
| Files, audit | 4-1-4, 4-1-10 |
| FILE-SET_IDENTIFIER | 1-5-1 |
| FINDHOST | 5-1-5* |
| Finite element | 7-1-7, 7-1-15 |
| Fixed length | 6-1-1 |
| Fixed-format | D-18 |
| FL | 3-2-6 |
| Flag, mode | C-39 |
| FLM | 3-2-6 |
| Floating point | G-4, G-7 |
| FLODUMP | 3-2-3, C-33* |
| Flowtrace table | C-33 |
| FMS | 1-3-1 |
| FORCE | 3-6-5 |
| Forced loading | 3-6-5 |
| Foreign | 6-1-4 |
| Foreign tape | 6-1-6 |
| Formal parameter Format, dataset Formats, tape Fortran | 3-3-2, 3-3-3 C-2* 6-1-1 1-3-1, C-15, C-19, C-33, D-10, D-18, E-1, E-2, G-3 |
| Fortran carriage control Fortran I/O record length Front-end FSE | D-44 3-6-8 1-1-1, 3-1-1, 3-1-10, 3-1-11, C-2, C-3, C-6, C-27, C-32, F-1 1-4-1 |
| FSE (editor) | E-1 |
| FSEPROC | 1-4-1 |
| FTP | 1-3-1, 5-1-6*, 5-1-7 |
| FTREF | 3-2-4, C-33* |
| FT05 | C-8 |
| FT06 | C-8 |
| Full screen | E-1 |
| Functions, special | 7-1-9 |
| F_float | G-7 |
| G register | 3-2-7 |
| Gateway | 5-1-7 |
| Gen. Purp. Sim. Sys. (GP55) | 7-1-8* |
| GET_FILE | 4-1-9* |
| Global symbol | 5-4-2 |

•

| GPSS GRIPE GO-G7 G_float $1-3-1, 7-1-8^*$ 1-2-8 GO-G7 G_float Hardware Configuration Hasp Header Heap $1-1-2$ 1-3-1 Header Heap Header HEAP $3-6-9$ Heap 3-6-4, 3-6-8 HEAP HELP $1-3-1, 3-1-13, 3-1-16, 5-1-3, 5-2-1, 5-2-5, D-38^*$ Help library $5-2-5, D-38^*$ Help Notule Hextdecimal G-1, G-2, G-4 HFT HFT ACCESS $4-1-6^*$ HFT DEFAULT HFT DEFAULT HFT DEFAULT HFT DEFAULT HFT DEFAULT HFT DEFAULT HFT STORE HFT DIRECTORY HFT PECTORY HFT PECH HFT PECH HFT STORE $3-2-6, 3-2-7$ Histogram C-59 History History History HIST STORE Hierarchy HOLD $3-2-6, 3-2-7$ Histogram History HIST STORE History History Hollerith Host address S-1-5 Host address S-1-5 Host address Host address Host table $5-1-5$ Host mame | 90/09/30 RevO | GPS-I Page Index-13 |
|---|---|---|
| Hasp $1-3-1$ Header $3-6-9$ Heap $3-6-4$, $3-6-8$ HEAP $3-6-5$ HELP $1-3-1, 3-1-13, 3-1-16, 5-1-3, 5-2-1, 5-2-5, D-38*$ Help library $5-1-3, 5-2-1^*, 5-2-3$ Help module $5-2-2$ Hexadecimal G^{-1}, G^{-2}, G^{-4} HFT $4-1-6^*$ HFT CHANCE $4-1-6^*$ HFT DELETE $4-1-6^*$ HFT DELETE $4-1-6^*$ HFT DELETE $4-1-6^*$ HFT DELETE $4-1-6^*$ HFT DIRECTORY $4-1-6^*$ HFT DIRECTORY $4-1-6^*$ HFT PERMIT $4-1-6^*$ HFT STORE $4-1-6^*$ Histogram $C-59$ History $3-2-6, 3-2-7$ History $3-1-3, 5-2-5, 5-4-1$ HDD $3-2-2, C-34^*, C-47$ Hollerith $6-1, G-2$ Host address $5-1-5$ Host table $5-1-5$ Host table $5-1-5$ Host name $5-1-7$ HYPERchannel $1-1-5$ HYPERchannel $1-1-5$ | GRIPE GO-G7 | 1-2-8 3-2-7 |
| Help module $5-2-2$ Hexadecimal $G-1, G-2, G-4$ HFT $4-1-6^*$ HFT $A-1-6^*$ HFT CHANCE $4-1-6^*$ HFT DEFAULT $4-1-6^*$ HFT DELETE $4-1-6^*$ HFT DIRECTORY $4-1-6^*$ HFT PETCH $4-1-6^*$ HFT PETCH $4-1-6^*$ HFT PERMIT $4-1-6^*$ HFT STORE $4-1-6^*$ Hierarchy $3-2-6, 3-2-7$ Histogram $C-59$ History $3-1-3, 5-2-5, 5-4-1$ HDLD $3-2-2, C-34^*, C-47$ Hollerith $G-1, G-2$ Home directory $5-1-5$ Host address $5-1-5$ Host table $5-1-5$ Host table $5-1-5$ Host mame $5-1-7$ HYPERchannel $1-1-5$ HYPERchannel File Transfer $4-1-6$ | Hasp Header Heap HEAP | 1-3-1 3-6-9 3-6-4, 3-6-8 3-6-5 1-3-1, 3-1-13, 3-1-16, 5-1-3, 5-2-1, |
| HFT ACCESS 4-1-6* HFT CHANGE 4-1-6* HFT DEFAULT 4-1-6* HFT DELETE 4-1-6* HFT DELETE 4-1-6* HFT DELETE 4-1-6* HFT DELETE 4-1-6* HFT PETCH 4-1-6* HFT PERMIT 4-1-6* HFT PERMIT 4-1-6* HFT STORE 4-1-6* Hierarchy 3-2-6, 3-2-7 Histogram C-59 Histogram C-59 History 3-1-3, 5-2-5, 5-4-1 HDLD 3-2-2, C-34*, C-47 HOLD 3-2-2, C-34*, C-47 Hollerith G-1, G-2 Home directory 5-1-5 Host address 5-1-5 Host name 5-1-5 Host name 5-1-5 Hostname 5-1-7 HOTSPOT 1-3-1 HYPERchannel 1-1-5 HYPERchannel File Transfer 4-1-6 | Help module | 5-1-3, 5-2-1*, 5-2-3 5-2-2 |
| HFT FETCH $4-1-6^{*}$ HFT PASSWORD $1-2-7$, $4-1-1$, $4-1-6^{*}$ HFT PERMIT $4-1-6^{*}$ HFT STORE $4-1-6^{*}$ Histogram $C-59$ History $3-2-6$, $3-2-7$ Histogram $C-59$ History $3-1-3$, $5-2-5$, $5-4-1$ HLP\$LIBRARYD-1HOLD $3-2-2$, $C-34^{*}$, $C-47$ HollerithG-1, G-2Home directory $5-1-5$ Host $5-1-5$ Host address $5-1-5$ Host table $5-1-5$ Host table $5-1-7$ HOTSPOT $1-3-1$ HYPERchannel $1-1-5$ HYPERchannel File Transfer $4-1-6$ | HFT ACCESS HFT CHANGE HFT DEFAULT | 4-1-6* 4-1-6* 4-1-6* |
| Histogram C-59 History 3-1-3, 5-2-5, 5-4-1 HLP\$LIBRARY D-1 HOLD 3-2-2, C-34*, C-47 Hollerith G-1, G-2 Home directory 5-1-3 Host 5-1-5 Host address 5-1-5 Host name 5-1-5 Host table 5-1-5 Hostname 5-1-7 HOTSPOT 1-3-1 HYPERchannel 1-1-5 HYPERchannel File Transfer 4-1-6 | HFT FETCH HFT PASSWORD HFT PERMIT | 4-1-6* 1-2-7, 4-1-1, 4-1-6* 4-1-6* |
| Home directory $5-1-3$ Host $5-1-5$ Host address $5-1-5$ Host name $5-1-5$ Host table $5-1-5$ Host table $5-1-7$ HoTSPOT $1-3-1$ HYPERchannel $1-1-5$ HYPERchannel File Transfer $4-1-6$ | Histogram History HLP\$LIBRARY | C-59 3-1-3, 5-2-5, 5-4-1 D-1 |
| Hostname5-1-7HOTSPOT1-3-1HYPERchannel1-1-5HYPERchannel File Transfer4-1-6 | Home directory Host Host address | 5-1-3 5-1-5 5-1-5 |
| | Hostname HOTSPOT HYPERchannel | 5-1-7 1-3-1 1-1-5 |
| H_float G-7 | H_float | G-7 |

.

I

| Page Index-14 | ICF-Job | | Re | ev0 | 90/09/30 |
|---|-------------|--|--------------------------------|---------------------|----------|
| ICF ICFPROC ICFSTATUS ID IDENT | | | 3-1-1, 3-1- 3-1-16* C-3* | -16 | |
| IDN IF IMSL \$IN INCLUDE | | | 3-2-6, C-34 7-1-9* | 4 * | |
| Indefinite Inefficient code Infinite INFORM Information | | G-3 C-59 G-3 D-44 5-2-1 | | | |
| INGRES Initialize INITIALIZE Input Input dataset | | 3-6-7, 6-1-4, | | | |
| Input queue INSERT Integer INTEGER*2 INTEGER*4 | | C-60 3-4-2 3-2-6, G-7 G-7 | G-4, G-6, (| G-7, G-8, G- | 9 |
| Interactive Interactive Cray H Internal data stru Internal represent | ucture | 4-1-1, | D-34, F-1, 3-1-16 | -1, 3-1-2, 3 F-2 | 3-1-11, |
| INTERNET Interpret Interpreter, card Interrupt Int'l Math & Stat | Libs (IMSL) | 5-1-5 C-24, (1-2-8 D-33, 1 7-1-9* | | | |
| IOAREA ISTATUS ITEMIZE Iterative I/O | | 3-2-1, 3-1-13 3-2-4, C-38 6-1-1, | , D-38* C-36* | | |
| | | | | | |

J register JCL 3-2-7 C-1 3-1-5, 3-1-8, 3-1-10, 3-1-11, 4-1-1, Job

C-47, C-54, D-14, D-22, D-29, D-39 Job 3-1-3, 3-1-13, 3-2-1, C-36*, D-38* JOB 3-1-3 Job, batch Job class 3 - 1 - 4D-46 Job class structure 3-3-1 Job control Job control language 3-2-1, 3-2-6 Job control statement 3-1-5 Job cost C-37 1-2-1, 4-1-1, 4-1-8 Job order number Job resource C-20 3-2-7 Job Status Register Job step 3-2-7 3-2-1, 3-3-4, C-37* JOBCOST .JOU 5-1-2 5-1-2, 5-1-4 Journal file 3-2-7 JSR JSTAT 3-1-13, D-39* J0-7J 3-2-7 Kermit 1-3-1, 5-1-7 KERMIT 7-1-12* 3-3-3, 3-6-2 Keyword 3-3-2, 3-3-3, C-1*, D-1 Keyword parameter 3-1-13, D-39* KILL Killing batch jobs 3-1-6*, 3-1-8*, 5-1-4* 1-5-1, 6-1-4, D-3 Label LABEL 1-5-1, 6-1-6 Label, ANSI standard 6-1-1 Label, tape 6-1-1 3-2-4 Language G1-1 Letter LIB 3-6-6 6-1-2, 6-1-3, F-4 Librarian, tape 3-6-4, 3-6-6, 3-6-7, C-12, E-1 Library 3-2-1, 5-2-1, 5-2-3, 5-2-4, 5-2-5, LIBRARY 5-4-1, 5-4-2, 5-4-3, 5-5-1, 5-5-2, 5-5-3, C-37* Library dataset C-36 5-1-3, 5-2-1*, 5-2-3 Library, help 3-2-5, 3-5-1*, 5-4-1*, 5-4-2, 5-4-3 Library, object 7-1-6, 7-1-9, 7-1-13 Library of subprograms

Page Index-16 Lib-Lon

n

Rev0

90/09/30

Library, program 3-2-5, 3-4-1, C-10, C-62 Library tape 6-1-3 5-5-1* Library, text 7-1-18 Linear 7-1-13 Linear algebra Link 5-4-1, F-1, F-2, F-3 LINK 5-4-3 Link attributes 4-1-9 1-3-1, 7-1-13* LINPACK 3-6-5, 5-2-5*, 5-4-3, 5-5-3, C-28 List LIST 3 - 4 - 3Listing C-48 LISTLB 6-1-6 3-2-6, 3-3-3 Literal 3-2-7 Literal string LLBUDG (Cray job class) 3-1-4 3-1-4 LLDEFER (Cray job class) LLNORM (Cray job class) 3 - 1 - 43-1-4 LLPZERO (Cray job class) 3-6-5 Load 3-6-6, 3-6-9, 3-6-15 Load map 3-6-1*, C-55 Loader Local 4-1-3, C-2, C-5, C-32 Local dataset 3-1-1, C-27, C-28, C-55, C-60 Local file C-52 Local file name C-15 C-35 Lock 1 - 2 - 8Log 3-1-3*, C-37 \$LOG C-51, D-41 Logfile LOGFILE 3-1-13, D-40* C-31 Logfile message 3 - 2 - 4Logic structure G-5 Logical Logical name 5-1-3, D-2 3-2-7 Logical operator 5-1-2 Login 5-1-3 Login Procedure File LOGINPR 1-4-1 LOGIN.COM 1-4-1, 5-1-3 LOGOFF 3-1-16 3-1-16, 3-1-17 LOGON LOGOUT 5-1-2 Logout, automatic 5-1-2 G-7 Longword

LOOP 3-1-13, 3-2-4, C-38*, D-41* Lost time 1 - 2 - 8LSBUDG (Cray job class) 3-1-4 LSDEFER (Cray job class) 3 - 1 - 4LSNORM (Cray job class) 3 - 1 - 4LSPZERO (Cray job class) 3-1-4 Macsyma 1-3-1 6-1-1* Magnetic tape Magnetic tape, clean 1-2-8 1-2-8 Magnetic tape, degauss Magnetic tape, purchase 1-2-8 1-2-8 Magnetic tape, test 5-1-7*, 5-1-8 Mail Mainframe 3-2-6 F-1, F-2, F-3 Maintenance C-3 Maintenance control word Mantissa G-6, G-7, G-8, G-9 E-1, F-4 Manuals MAP 3-6-6 Map, load 3-6-6, 3-6-9, 3-6-15 Mask G-1 1-1-1, 3-1-10, 3-2-2, 3-3-4, 4-1-1*, Mass Storage System C-40, F-2 MASTER 3-4-3 3 - 4 - 1Master character Mathematics 7-1-9 2-1-1, 3-1-1, 3-1-4, 3-6-5, 3-6-8 Memory MEMOR Y 3-2-1, C-38* C-58 Merge Message 3-6-1, 3-6-6, C-2, C-3, D-40, D-41, D-44, E-1 3-1-13, D-41* MESSAGE Message, logfile C-31 Message, recorded F-4 C-3* MF 1-1-2*, F-2 MFN Microcomputer 7-1-12, F-4 Microfiche D-19 MLEVEL 3-6-6 MMS 7-1-14* MODE 3-2-1, C-39* MODE4A/4C 1-3-1 MODIFY 3-2-2, C-39*

Page Index-18

Mod-Net

Rev0

90/09/30

5-2-4, 5-4-2, 5-5-2 Modify a library Module 3-6-3, 3-6-11 Module, absolute 3-6-1 Module, help 5-2-2 7-1-14 Module Mgt System (MMS) 5-4-1*, 5-4-2, 5-4-3 Module, object Module, relocatable 3-6-3 Module, text 3-4-1, 5-5-1* MODULES 3-6-7, 3-6-11, 3-6-14 6-1-4, D-3* MOUNT 3 - 4 - 4MOVEDK **MSACCES** 3-2-2, 3-3-4, 4-1-4*, 4-1-10*, C-40* MSAUDIT 3-3-4, 4-1-4*, 4-1-10*, C-40* 4-1-10* **MSCATLIST MSCHANG** 3-2-2, 3-3-4, 4-1-1, $4-1-4^*$, $4-1-10^*$, C-41* **MSFETCH** 3-2-2, 3-3-4, 4-1-4*, 4-1-10*, C-43* MSG C-3* MSPASSW 1-2-7, 3-3-4, 4-1-1, 4-1-4*, 4-1-11*, C-44* **MSPURGE** 3-2-2, 3-3-4, 4-1-5*, 4-1-11*, C-44* 3-1-10, 3-3-4, 4-1-1*, 4-1-3, F-2 MSS MSSAUDIT 4-1-7* 4-1-7* **MSSBACKUP** MSSBACKUP DELETE 4-1-7 MSSBACKUP FETCH 4-1-7 MSSBACKUP LIST 4-1-7 MSSBACKUP STORE 4-1-7 4-1-7* **MSSDELETE** 4-1-1, 4-1-8* **MSSNEWCHRG** MSSTORE 3-2-2, 3-3-4, 4-1-5*, 4-1-11*, C-45* 1-2-2* Multiple accounts Multi-file tape 1-5-1*, 6-1-8 C-3* NA Name 3-3-1 Name, host 5-1-5 Name, logical 5-1-3, D-2 5-1-5 Name server Nastran 1-3-2 7-1-15* NASTRAN NAVSEA 5-1-5 Negative G-6, G-7, G-8, G-9 5-1-1, 5-1-5* Network

F-2, F-3 Network address 1-1-5 Network diagram 1-1-2 Network ID NEWCHRG 3-2-2, 3-3-4, 4-1-1, C-47* 1-2-1*, 5-1-2* NEWS 1-2-1* Newsletter (CISD) C-3 No abort (NA) 1-1-3, 5-1-4 Node 3-6-7 NODEFLIB 3-2-2, C-47* NOHOLD 3-4-3 NOLIST 7-1-18 Non-linear 3-2-1, C-47* NORERUN 3-1-4*, C-37 NORMAL job class 1-1-2 nos 6-1-6, A-4, E-1, F-2, G-8, G1-1 NOS NOS station E-1 3-4-2 NOSEQ 6-1-6, A-4, G-8 NOS/BE 6-1-6, 6-1-7, E-1, F-2, G-9, G1-1 NOS/VE 3-6-1 Note 3-2-3, C-48* NOTE 3-2-7 NOTEXT 3-3-3 Null string 1-1-2*, F-2 N1 3-1-12 ^0 3-1-12 D-36* ^0 (CTRL-0) 1-1-5 OA VAXes F-3 oasys 5-1-5*, F-3 OASYS F-3 OASYS (OASYS) 3-2-5, 3-5-1*, 5-4-1*, 5-4-2, 5-4-3 Object library 5-4-1*, 5-4-2, 5-4-3 Object module G-1, G-2, G-4 Octal C-3* ODN

Page Index-20 Off-Pas

Pas

1

Rev0

90/09/30

| Office automation | F-3 |
|--|---|
| Office Automation System | 5-1-5*, F-3 |
| Off-line | 6-1-1 |
| Off-line work request | 6-1-2 |
| Off-station | 4-1-2 |
| OLDNews | 5-1-2 |
| One's complement | G-4 |
| Operating system | 1-1-1, F-1, F-2, F-3 |
| Operator, arithmetic | 3-2-7 |
| Operator, Central Site | 5-1-2 |
| Operator, logical | 3-2-7 |
| Operator, relational | 3-2-7 |
| OPTION | 3-2-1, C-48* |
| ORDER | 3-6-7 |
| ŞOUT | 3-1-3* |
| Output | 3-1-8, C-3, D-33, D-35, D-37 |
| Output, CDC | D-10 |
| Output dataset | 3-1-3 |
| Outside user | 1-1-1 |
| Overlay | 3-6-12 |
| Overwrite file | C-55 |
| OWN | C-3* |
| Owner | C-3 |
| Page break | D-44 |
| PAM | C-3* |
| Paper, terminal | 1-2-8 |
| Parameter | 3-3-1, 3-3-2, C-1*, D-1 |
| Parameter, formal | 3-3-2, 3-3-3 |
| Parameter, keyword | 3-3-2, 3-3-3, C-1* |
| Parameter, positional | 3-3-2, 3-3-3, C-1* |
| Parameter substitution | 3-3-1, 3-3-3* |
| Parentheses | 3-3-3 |
| Parenthetical string | 3-2-7, 3-2-8 |
| Parity | 5-1-1* |
| Parity error | 6-1-2 |
| Parmeter | 3-3-1 |
| Pascal | 1-3-2, E-1 |
| PASCAL | 3-2-4, 7-1-16*, C-50* |
| PASSWOR Password Password, Login | 1-2-7, 4-1-1 1-2-1, 1-2-6*, 3-1-2, 4-1-1, 4-1-4, 4-1-10, 4-1-11, 5-1-1, 5-1-2, C-40, C-41, C-44, D-13 5-1-2 |

1-3-2 Patran 3-1-13. D-41* PAUSE PCA 1-3-2, 7-1-17* PDMFC 3-2-7 PDMST 3-2-6 C-3* PDN Perf. & Coverage Anlyzr (PCA) 7-1-17 3 - 1 - 17PERIOD Peripheral 4-1-9 3-1-1, 4-1-3, C-3, C-4, C-5, C-8, Permanent dataset C-26, C-51, C-55, D-36 3-2-2 Permanent dataset management Permanent dataset staging 3-2-2 1-2-2, 4-1-3, 4-1-9, 5-1-2 Permanent file permission C-46 3-2-2, 4-1-6, C-2, C-3, C-7, C-41, Permission C-44, C-45, C-51, C-55, D-27, D-44 3-2-2, C-51* PERMIT F-4* Personnel, administrative 1-3-2 Pert Time 3-4-1* PL PLAY 1-4-2, 3-1-13, 3-1-17, D-42* Plot 1 - 2 - 87-1-5 Plotting PLOT10 1-3-2 PL/I 1-3-2 1-3-2 PM (Proj Mgt) Point, binary G-8, G-9 G-7 Point, floating Position C-52, C-54 Position a tape D-4 3-3-2, 3-3-3, C-1*, D-1 Positional parameter Positive G-6, G-7, G-8, G-9 7-1-5 Post-processing Precision G-6, G-7, G-9 PREFIX 3-1-17 3-6-7* PRESET 3-1-1, 3-1-8, D-6, D-10 Print PRINT 3-2-4, C-51* Print (on CDC CYBER 860) D-20 D-7, D-8 Printer, auxiliary 3-2-4, 3-3-1 PROC **\$PROC** 3-3-1* PROC 3-3-2. C-51* 3-2-4, 3-2-7, 3-3-1, 3-3-5, 5-3-1*, Procedure

C-14, C-24, C-51, C-54, D-1 Procedure PROCLIB 3-3-4*, C-5 3-2-5, 3-4-1, C-10, C-62 Program library Project Mgt 1-3-2 PROLOG 1-4-2 1-4-1* Prolog file 3-1-16* Prologue file Prompt (!) 3-1-11 3-3-1, 3-3-2*, 3-3-5, C-51 Prototype Punch A-4 1-2-8 Punch, card Purchase magnetic tape 1-2-8 4-1-1, 4-1-5, 4-1-11, C-44 Purge PURGE 3-4-4, 5-1-4 PURGEDK 3-4-4 3-1-4*, C-37 PZERO job class QGET 3-1-8 D-20* **QPRINT** D-22* QSUBMIT QUERY 3-2-3, C-52* D-39, D-43, D-45, D-48 Queue QUIT 3-1-11, 3-1-13, 3-1-17, D-42* RCAUDIT D-23* RCDELETE D-24* D-25* RCGET D-27* RCSAVE D-29* RCSUBMIT READ 3-4-3 C-3 Read control word Real G-6, G-8, G-9 REAL*16 G-7 REAL*4 G-7 REAL*8 G-7 RECORD 3-1-14, D-42* D-10 Record attribute Record, blocked C-23, C-58 Record length 3-6-8 F-4 Recorded message 4-1-1 Recover

> 3-6-8 E-1

Page Index-22 Pro-Ref

Reentrant

Reference

Rev0

90/09/30

ì

| Refresh | D-34, D-44 |
|--|----------------------------------|
| Refund request | 1-2-8 |
| Register | 3-2-7 |
| Register, G | 3-2-7 |
| Register, J | 3-2-7 |
| | |
| Register, Job Status | 3-2-7 |
| Register, vector | C-30 |
| Registration | 1-2-1* 7-1-11 |
| Relational database mgt sys Relational operator | 3-2-7 |
| Relational operator | 5-2-1 |
| Release | C-34 |
| RELEASE | 3-1-14, 3-2-2, C-53* |
| Relocatable | 3-6-10 |
| Relocatable module | 3-6-3 |
| Remove | C-26 |
| | |
| REMOVE | 3-1-14, D-43* |
| Rename file | C-41 |
| REPLACE_FILE | 4-1-9* |
| Representation, internal | G-2, G-6* |
| Reprieve | D-34, D-37 |
| DEGUEST MACHETIC TADE | 6-1-7 |
| REQUEST_MAGNETIC_TAPE RERUN | 3-1-14, 3-2-1, C-54*, D-43* |
| RESOURC | 6-1-6 |
| Resource, job | C-20 |
| Restart | D-41 |
| | |
| RESTORE | 3-4-2 |
| RESUME | 3-1-17 |
| Return | C-53 |
| RETURN | 3-2-1, 3-2-4, 3-3-1, C-54* |
| Rewind | D-5 |
| REWIND | 3-2-3, 3-4-3, C-54* |
| RFTAPE | 6-1-4 |
| Ribbons | 1-2-8 |
| RIM | 1-3-2 |
| RKE terminal | F-2 |
| | |
| ROLLJOB | 3-2-1, C-54* |
| Root segment | 3-6-15 |
| | |
| | |
| SAVE | 3-2-2, 3-6-12, C-6, C-55*, D-43* |
| Scalar | 2-1-1, 3-1-1 |
| Schedule | F-1, F-2, F-3 |
| Scratch | 3-1-1, 3-6-11 |
| Screen | D-34 |
| | |
| SCRUBDS | 3-2-2, C-55* |
| | |

Page Index-24

Rev0

90/09/30

1-4-2 SCU EDITOR PROLOG Search order C-37 Sector C-58 Secure 1-1-3, 5-1-1, F-2 SECURE F-2 SECURE job class 3-1-4*, C-37 Security 1-2-6, 4-1-1 SEGLDR 3-2-5, 3-6-1, 3-6-10*, C-55*, E-1 SEGLDR directive 3-6-2* 3-6-1, 3-6-11, 3-6-12, 3-6-13 Segment SEGMENT 3-6-12, 3-6-14 Segmentation 3-6-10, C-55 3-6-15 Segmentation caution 3-6-2, A-4 Semicolon Sense switch 3-2-7, C-61, D-49 C-1* Separator SEQ 3-4-2 G-3 Sequence F-3 Sequent Service F-1, F-2, F-4 Service class 3-1-4 SET 3-1-14, 3-2-1, 3-6-8, C-56* SET HOST 5-1-5 Set ID 1-5-1* D-4* SET MAGTAPE SET PASSWORD 1-2-6, 5-1-2* SET TERMINAL D-44*, D-45* SET FILE ATTRIBUTES 1-5-1 SET PASSWORD 1-2-7 SHOW 3-1-14 SHOW QUEUES D-45* SHOW SYSTEM 5-1-4 SHOW USERS 5-1-3 Shredder 1-2-8 SI 1 - 5 - 1SID 3-2-5, C-56* Sign G-6, G-7, G-8, G-9 Simple Mail Trans. Protocol 5-1-7 Simple procedure 3-3-1, 3-3-5 Simscript 1-3-2 Simulation 7-1-8 C-7 Size Size, dataset C-7 3-3-1 Skip SKIPD 3-2-3, C-57*

| SKIPF | 3-2-3, 3-4-3, C-57* |
|--------------------------|---|
| SKIPR | 3-2-3, C-58* |
| SKIPU | 3-2-3, C-58* |
| SKIP_TAPE_MARK | 6-1-7 |
| SLBUDG (Cray job class) | 3-1-4 |
| SLNORM (Cray job class) | $\begin{array}{c} 3-1-4 \\ 3-1-4 \\ 6-1-2, 6-1-3 \\ 3-1-4 \\ 5-1-7 \end{array}$ |
| SNAP | 3-1-14, D-46* |
| Software | 1-3-1* |
| SORT | 3-2-5, C-58* |
| Sort/Merge | C-58 |
| Source | 3-4-1, C-62 |
| Source program | 5-5-1 |
| Special functions | 7-1-9 |
| SPICE | 1-3-2, 7-1-18* |
| SPY | 1-3-2, 3-2-4, 7-1-17, C-59* |
| SQL | 7-1-11 |
| Sroll | D-44 |
| SSBUDG (Cray job class) | 3-1-4 |
| SSDEFER (Cray job class) | 3-1-4 |
| SSNORM (Cray job class) | 3-1-4 |
| SSPZERO (Cray job class) | 3-1-4 |
| SSW1-SSW6 | 3-2-7 |
| STACK | 3-6-8 |
| Stage | 3-1-1, 3-2-2, C-27, D-43, D-45, D-48 |
| Standard Query Language | 7-1-11 |
| Start execution | 3-6-9 |
| STATCLASS | 3-1-14, D-46* |
| Station command | D-30 |
| Station ID | 1-1-2*, 1-1-3 |
| Statistics | 7-1-9 |
| Status | C-52, D-38, D-39, D-47, F-4 |
| STATUS | 3-1-14, 3-1-17, D-47* |
| Storage | 6-1-1 |
| STORAGE | 3-1-14 |
| Storage, trillion-bit | F-2 |
| Store | 4-1-5, 4-1-11, C-45, D-35 |
| Stranger tape | 6-1-6 |
| String | C-2* |
| String comparison | 3-2-6 |
| String, literal | 3-2-7 |
| String, parenthetical | 3-2-7, 3-2-8 |

90/09/30 Page Index-26 Rev0 Str-Tap Structural analysis 7-1-15 3-2-4 Structure, logic Subexpression 3-2-6 3-1-1, 3-1-10, 3-1-14, 3-2-2, 5-1-4, SUBMIT C-60*, D-48* 5-2-1, 5-4-1 Subprogram 7-1-6, 7-1-9, 7-1-13 Subprogram library Substitute 3-6-5 3-3-1. 3-3-3* Substitution, parameter 5-2-6 Subtopic, help 5-2-2*, 5-2-3 Sub-topic, help 1-2-8 Suggestions C-37 Summary Supercomputer 1-1-1 F-4 Support D-48 Suppress SUPPRESS 3-1-14, D-48* D-37, D-41 Suspend SUSPEND 3-1-17 SWITCH 3-1-14, 3-2-1, C-61*, D-49* 3-2-7 Switch, sense 3-2-7, 5-1-3, 5-4-2 Symbol 5-4-2 Symbol, global Symbolic variable 3-2-6, C-52 SYMBOLS 3-6-8 SYSID 3-2-6 D-6*, D-10* SYSSANAP Tab 3-1-5, D-18 5-1-5 Table, host 3-6-4, 3-6-8 Table, symbol Tape 1-2-8, 1-5-1, 4-1-1, 6-1-4, 6-1-6, 6-1-7, A-4, D-2, D-3, D-4 Tape assignment 6-1-3 6-1-6 Tape, foreign 6-1-1 Tape formats Tape label 6-1-1 6-1-2, 6-1-3, F-4 Tape librarian 6-1-3 Tape, library 6-1-1* Tape, magnetic D-5 Tape mark 1-5-1*, 6-1-8 Tape, multi-file 6-1-2, 6-1-3 Tape, slot 6-1-6 Tape, stranger

1-3-2, 7-1-7* TAURUS TDUMP 6-1-6 1-2-1, 5-1-1, F-4 Telephone TELNET 1-3-2 TERMDEF 1 - 4 - 1Terminal 5-1-1*, D-42, D-44, D-45 Terminal paper 1-2-8 Terminate C-31, D-34, D-37, D-42 A-4, C-1* Terminator Test magnetic tape 1-2-8 Text C-3, C-48 C-3* TEXT Text library 5-5-1* 3-4-1, 5-5-1* Text module Time 3-1-4 TIME 3 - 2 - 6Time usage C-59 TIMELEFT 3-2-6 3-6-9 TITLE .TJL 5-1-2 Topic, help 5-2-2*, 5-2-3, 5-2-6 TPGET 6-1-3 TPU (editor) 5-1-4, 5-6-1 Training F-4 Transfer 4-1-3, C-14 Transfer, file 7-1-12 Transfer file 5-1-6 Transfer funds 1-1-1 C-2, C-43, C-45 Transparent TREE 3-6-12, 3-6-13, 3-6-14 Tree diagram 3-6-14 3-6-11, 3-6-12, 3-6-13 Tree structure TRIAL 3-6-9 Trillion-bit storage F-2 Trouble form 1-2-8 G-5TRUE G-4 Two's complement .TXT 5-5-1 3-2-3 UBBLOCK Ultrix-32 F-3 C-61* UNBLOCK 6-1-1, 6-1-4 Unblocked Unblocked dataset C-11, C-24, C-58, C-61

Rev0 Page Index-28 Und-War 3-3-3 Underline 5-2-3 Underscore 2-1-1* UNICOS Uninitialized data area 3-6-7 C-3*, C-7, C-39 Unique access (UQ) F-3 UNIX 6-1-1, 6-1-6, 6-1-7 Unlabelled C-35 Unlock Unsatisfied external 3-6-9 3-4-4 UNYANK 3-2-5, 3-4-1*, C-10, C-62*, E-1 UPDATE 3-4-1 Upper case 1-4-1 UPROC 3-1-11, 5-2-6 Up-arrow C−3* UQ 1-1-1, 5-1-1, 5-1-2, G1-1 User initials 1-2-1*, 3-1-11 User Initials User Services F-4 5-1-2, 5-1-3, G1-1 Username 3-6-9 USX 5-2-1, 5-5-1, D-1 UTILITIES 3-2-3 Utility 3-5-1, C-5 UTILITY C-5, C-7 Validate Variable 3-2-7, C-56 Variable length 6-1-1 3-2-6, C-52 Variable, symbolic G-7 VAX VAXcluster 1-1-3, 1-3-1, 3-1-1, 3-1-5, 3-1-11, 4-1-6, 5-1-1*, 6-1-4, F-2 2-1-1, 3-1-1 Vector C-30 Vector register 1-2-8 Verifier, card 5-1-2, 5-1-4* Version 5-1-2 **VERYoldnews** 5-1-1*, 6-1-4, D-1, D-43, E-1, F-2 VMS 3-1-11, 3-1-12 VMS Cray Station 1-5-1, 6-1-2, 6-1-3, 6-1-6 VSN 1-1-3*, F-2 **V**3 1-1-3*, F-2 ٧4

Warning

3-6-1, 5-1-2

90/09/30

| 90/09/30 RevO | WEO-200 Page Index-29 |
|--|---|
| WEOF WFTAPE Width WIDTH Wildcard | 3-4-2 6-1-4 D-45 3-4-2 5-2-5, 5-4-2, 5-5-2, 5-5-3, C-4* |
| Window WINS% WIN/TCP Word Word format | D-45 5-1-7 1-3-2 G-6, G-7, G-8, G-9 G-4 |
| Word length Write control word WRITEDS WRITEF | G-1 C-3 3-2-3, C-64* 1-5-1 |
| XER Xerox 8700 XMODEM | 3-6-9 D-19 1-3-2 |
| YANK | 3-4-4 |
| ^z · | 3-1-11, 3-1-12 |
| | 3-1-11, 3-1-12, 5-2-6 |
| ^Z (CTRL-Z) | D-36* |
| Zero Zero-byte | C-55, G-3, G-5 A-4 |
| 11/780, DEC VAX | F-3 |
| 2 00-UT | 1-3-1 |

| Page Index-30 | 63 -PRO | : | Rev0 | 90/09/30 |
|---|---------|---|------|----------|
| 63-character set 6410, DEC VAX 64-character set | | A-4* 5-1-1, F-2 A-4*, G-3 | | |
| 7-track | | 6-1-1, 6-1-6 | | |
| 8250, DEC VAX 8550, DEC VAX 860, CDC CYBER 8-bit | | 1-1-3 1-1-3, 5-1-1, F- 1-1-2 5-1-1 | 2 | |
| 9-track | | 6-1-1, 6-1-4, 6- | 1-6 | |
| SCS SDEBUG SDUMP SIN SLOG | | 3-1-3*, 3-3-1* 3-6-4 C-30 3-1-3* 3-1-3*, C-37 | | |
| \$OUT \$PROC | | 3-1-3* 3-3-1* | | |

•

Initial Distribution

٠

Copies:

12 Director Defense Technical Information Center (DTIC) Cameron Station Alexandria, Virginia 23314

Center Distribution

Copies:

| 1 | 35/3509 | Gray, G. R. |
|-----|---------|---------------------------|
| 1 | 3501 | Minor, L. R. |
| 1 | 3502 | Morris, J. |
| 1 | 351 | Strickland, J. D. |
| 150 | 3511 | Willner, S. E. |
| 20 | 3511 | Sommer, D. V. |
| 1 | 3512 | Brady, Mary |
| 1 | 3513 | Wessel, J. |
| 1 | 353 | Yearick, R. |
| 1 | 3533 | Hayden, H. P. |
| 1 | 3533 | Annapolis Computer Center |
| 1 | 355 | Kearney, E. |
| 1 | 3551 | Crum, Barbara |
| 1 | 3552 | Brady, Martha |
| 1 | 357 | - |
| 1 | 3571 | Knowles, H. |
| 1 | 3572 | Smith, T. |
| 1 | 522 | TIC (C) |
| 1 | 522.2 | TIC (A) |
| | | |

David Taylor Research Center

Bethesda, Maryland 20084-5000

CISD-90/01 30 September 1990 Computer & Information Services Department Departmental Report

COMPUTER CENTER REFERENCE MANUAL, VOLUME 1

David V. Sommer Sharon E. Good

> Approved for Public Release: Distribution Unlimited



CISD-90/01 COMPUTER CENTER REFERENCE MANUAL, VOLUME 1

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

•

| | REPORT DOCUM | ENTATION PA | GE | | |
|--|--|---|--|--|---|
| 1a. REPORT SECURITY CLASSIFICATION | | 16. RESTRICTIVE MARKINGS | | | |
| UNCLASSIFIED | | | | | |
| 28. SECURITY CLASSIFICATION AUTHORITY | | 3. DISTRIBUTIO | N/AVAILABILITY | OF REPORT | |
| 2b. DECLASSIFICATION/DOWNGRADING SCHE | DULE | | ed for P | | |
| 4. PERFORMING ORGANIZATION REPORT NUM | | | bution U | | |
| CISD-90/01 | (DEN(S) | S. WORLDRING | UNGANIZATION | | -1137 |
| CI3D 90/01 | | | | | |
| 68. NAME OF PERFORMING ORGANIZATION | NAME OF PERFORMING ORGANIZATION 66. OFFICE SYMBOL | | ONITORING ORG | ANIZATION | |
| DTRC (If applicable) | | | | | |
| | 3511 | | | | |
| 6c. ADDRESS (City,State,and ZIP Code) | | 7b. ADDRESS (C | ity, State, and ZI | P Code) | |
| | 5000 | | | | |
| Bethesda, MD 20084- | 5000 | 9. PROCUREMEN | T INSTRUMENT I | | |
| ORGANIZATION | (If applicable) | S. PROCONLINEN | | DENTITION | I NUMBER |
| | | | | | |
| 8c. ADDRESS (City,State,and ZIP Code) | | 10. SOURCE OF | FUNDING NUMBE | RS | |
| | | PROGRAM ELEMENT NO. | PROJECT NO. | TASK NO. | WORK UNIT |
| | | | | | |
| | on E. Good NE COVERED 093090 To inde | 1 | PORT (Year,Mont) /09/30 | h,Day) 15. PA | GE COUNT |
| 17. COSÀTI CODES FIELD GROUP SUB-GROUP | Computer | a computir | ng Cray Cray | COS UNICOS | lock number) |
| | <u>Control s</u> | | DEC | <u>VAX/VMS</u> | |
| 19. ABSTRACT (Continue on reverse if neces The Computer Center in of the David Taylor Res Supercomputer Network. Network. Some informat documents and augmented examples and description information on moving f Storage System), the DE | the Computer a search Center h This manual p tion has been o to reflect us ons of hardware tiles among the | and Inform has instal provides a listilled sage at DT e and soft e CDC CYBE | lled an I an introd from man TRC. Con ware are IR 860 (w | ntegrate uction t y indivi trol sta include ith the | ed cothe dual atement ed, as is Mass |
| systems. Volume 1 descr the DEC VAXes. Volume | uting batch jo tibes the Cray | bs, and u X/MP, the | sing the Mass St | interac | tive |
| systems. Volume 1 descr | uting batch jo ibes the Cray 2 describes the | bbs, and u X/MP, the ne CDC CYP | sing the Mass St | interac orage Sy | tive |
| systems. Volume 1 descrithe DEC VAXes. Volume | uting batch jo ibes the Cray 2 describes the | 21. ABSTRACT S | Mass St BER 860. | interac orage Sy | tive stem and |
| systems. Volume 1 descrithe DEC VAXes. Volume | uting batch jo ibes the Cray 2 describes the | 21. ABSTRACT S UNCLAS 22b. TELEPHONE | SECURITY CLASS SER 860. SECURITY CLASS SIFIED (Include Ares Co 267-3343 | interac orage Sy IFICATION de) 22c. OFFICE 35] | stem and |
| systems. Volume 1 descrithe DEC VAXes. Volume | uting batch jo ibes the Cray 2 describes the | 21. ABSTRACT S UNCLAS 22b. TELEPHONE | SECURITY CLASS SER 860. SECURITY CLASS SIFIED (Include Ares Co 267-3343 | interac orage Sy IFICATION de) 22c. OFFICE 35] | stem and |

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

18 (continued)

Hardware Interactive Mass Storage System Programming Secure computing Software Documentation Supercomputer ******

David Taylor Research Center Bethesda, Maryland 20084-5000

******** * * × × * × Computer Center \star * Reference Manual * Volume 1: Cray, MSS, DEC * × sie. $\frac{1}{2}$ sk. *****

by David V. Sommer Sharon E. Good

Scientific and Engineering User Support Branch Code 3511

CarderockAnnapolisPhone(301) 227-1907(301) 267-3343Autovon287-1907281-3343

For recorded message on computer status (301) 227-3043

Questions and requests for more detailed information should be directed to Code 3511, Bldg. 17, Rm. 226

Computer and Information Systems Department Departmental Report

September 1990

. .

CISD-90/01

Through Revision 0 (Sept 1990)

••

*** Revision Record ***

Revision Description 0 (Sep 90) Original printing.

1

*** List of Effective Pages ***

| Pages | Rev | Pages | Rev |
|-----------------------|-----|-------|-----|
| | | | |
| Title | 0 | | |
| i thru ix | 0 | | |
| 1-1-1 thru 1-1-5 | 0 | | |
| 1-2-1 thru 1-2-8 | 0 | | |
| 1-3-1 thru 1-3-2 | 0 | | |
| 1-4-1 thru 1-4-2 | 0 | | |
| 1-5-1 | 0 | | |
| 2-1-1 | C | | |
| 3-1-1 thru 3-1-17 | 0 | | |
| 3-2-1 thru 3-2-8 | 0 | | |
| 3-3-1 thru 3-3-7 | 0 | | |
| 3-4-1 thru 3-4-6 | 0 | | |
| 3-5-1 thru 3-5-3 | 0 | | |
| 3-6-1 thru 3-6-16 | 0 | | |
| 4-1-1 thru 4-1-11 | 0 | | |
| 5-1-1 thru 5-1-8 | 0 | | |
| 5-2-1 thru 5-2-12 | 0 | | |
| 5-3-1 | 0 | | |
| 5-4-1 thru 5-4-3 | 0 | | |
| 5-5-1 thru 5-5-3 | 0 | | |
| 5-6-1 thru 5-6-3 | 0 | | |
| 6-1-1 thru 6-1-8 | 0 | | |
| 7-1-1 thru 7-1-18 | 0 | | |
| A-1 thru A-4 | 0 | | |
| B-1 | 0 | | |
| C-1 thru C-64 | 0 | | |
| D-1 thru D-49 | 0 | | |
| E-1 thru E-2 | 0 | | |
| F-1 thru F-4 | Ō | | |
| G-1 thru G-9 | 0 | | |
| G1-1 | 0 | | |
| Index-1 thru Index-30 | 0 | | |

. . . .

.

.

Contents

Volume 1

| | Preface | |
|---|---|----------------|
| | Revision Record | i |
| | List of Effective Pages | ii |
| | - | |
| 1 | T . F . F F | 1_1_1 |
| 1 | Introduction | 1-1-1 1-1-2 |
| | Hardware Configuration | |
| | CRAY X-MP / 216 | 1-1-2 |
| | CDC CYBER 180 model 860 | 1-1-2 |
| | DEC VAXcluster | 1-1-3 |
| | DEC VAX | 1-1-3 |
| | DEC Secure VAXcluster | 1-1-3 |
| | DTNET Dial-in | 1-1-4 |
| | The Integrated Supercomputer Network | 1-1-5 |
| | User Interface With the Computer Center | 1-2-1 |
| | General Information | 1-2-1 |
| | Registering | 1-2-1 |
| | Multiple Accounts | 1-2-2 |
| | Cray COS | 1-2-2 |
| | VAX | 1-2-3 |
| | CDC NOS | 1-2-4 |
| | Mass Storage System | 1-2-5 |
| | Passwords, Passwords, Everywhere | 1-2-6 |
| | Trouble Forms | 1-2-8 |
| | Refunds | 1-2-8 |
| | ADP Control Center | 1-2-8 |
| | Software Available | 1-3-1 |
| | Prolog and Epilog Command Files | 1-4-1 |
| | VAX/VMS Command Files | 1-4-1 |
| | NOS Command Files | 1-4-1 |
| | NOS/VE Command Files | 1-4-2 |
| | Cray Command Files | 1-4-2 |
| | | |
| | ANSI Standard Multi-file Tapes | 1-5-1 |
| | VAX | 1-5-1 |
| | NOS | 1-5-1 |
| | NOS/VE | 1-5-1 |
| | | |
| 2 | The CRAY X-MP (UNICOS) | 2-1-1 |
| | UNICOS Version 5.1 | 2-1-1 |
| • | | 2-1-1 |
| 3 | The CRAY X-MP (COS) | 3-1-1 3-1-1 |
| | COS Version 1.17.1 | |
| | Accessing the CRAY X-MP | 3-1-1 |

| Cray Datasets Changing your Cray password Batch Jobs Batch Job Classes SECURE Batch Job Class From the VAXcluster Killing Batch Jobs VAXcluster-to-Cray Examples From the CDC CYBER 860 Killing Batch Jobs CYBER 860-to-Cray Examples From a Running Cray Job Examples Interactive Jobs From the VAXcluster VMS Cray Station Commands Examples From the CDC CYBER 860 ICF Prologue File NOS ICF User Commands Examples | $\begin{array}{c} 3-1-1\\ 3-1-2\\ 3-1-3\\ 3-1-4\\ 3-1-4\\ 3-1-5\\ 3-1-6\\ 3-1-6\\ 3-1-6\\ 3-1-8\\ 3-1-8\\ 3-1-8\\ 3-1-9\\ 3-1-10\\ 3-1-10\\ 3-1-11\\ 3-1-11\\ 3-1-11\\ 3-1-12\\ 3-1-15\\ 3-1-16\\ 3-1-16\\ 3-1-16\\ 3-1-17\end{array}$ |
|--|--|
| Cray JCL Commands Job Definition and Control Dataset Definition and Control Permanent Dataset Management Permanent Dataset Staging Permanent Dataset Utilities Local Dataset Utilities Dumps and Other Aids Logic Structure Procedures Program Libraries Object Libraries Miscellaneous JCL Expressions Symbolic Variables System Constants COS-set Variables User-set Variables Operators Strings | $\begin{array}{c} 3-2-1\\ 3-2-1\\ 3-2-2\\ 3-2-2\\ 3-2-2\\ 3-2-3\\ 3-2-3\\ 3-2-3\\ 3-2-3\\ 3-2-4\\ 3-2-4\\ 3-2-4\\ 3-2-4\\ 3-2-5\\ 3-2-5\\ 3-2-5\\ 3-2-5\\ 3-2-6\\ 3-2-6\\ 3-2-6\\ 3-2-6\\ 3-2-7\\ 3-2-7\\ 3-2-7\\ 3-2-7\end{array}$ |
| Procedures Simple Procedures Complex Procedures Prototype Statement Temporary Datasets Parameter Substitution Apostrophes and Parentheses DTRC "System" Procedures DTRC Procedure Library Examples Simple Procedures Complex Procedures | 3-3-1 3-3-1 3-3-2 3-3-2 3-3-2 3-3-3 3-3-3 3-3-3 3-3-4 3-3-4 3-3-4 3-3-5 3-3-5 3-3-5 3-3-5 |

Ì

4

5

Page v

| Program Libraries UPDATE UPDATE Directives DECK and COMDECK Compile Directives Modification Directives Run Options Input Edit Directives Examples | 3-4-1 3-4-1 3-4-1 3-4-2 3-4-2 3-4-2 3-4-3 3-4-4 3-4-5 |
|--|---|
| Object Libraries BUILD Directives DTRC Object Libraries Examples | 3-5-1 3-5-1 3-5-1 3-5-2 |
| Loader SEGLDR Control Statement Message Levels Directive Segmentation Jegmentation Directives Sample Tree Diagram Segmentation Cautions Compile, Load and Save an Absolute Program Simple Load Segmented LOAD | 3-6-1 3-6-1 3-6-1 3-6-2 3-6-10 3-6-10 3-6-14 3-6-15 3-6-16 3-6-16 3-6-16 |
| The Mass Storage System MSS Security MSS File Purge MSS Backup for Critical Files Using the MSS from the Cray Using the MSS from the VAXcluster Using the MSS from the CDC CYBER 860 | $\begin{array}{c} 4-1-1 \\ 4-1-1 \\ 4-1-2 \\ 4-1-2 \\ 4-1-3 \\ 4-1-6 \\ 4-1-9 \end{array}$ |
| DEC VAXcluster VMS VMS Version 5.3 Accessing the VAXcluster Login Password Logout Procedures System News Login Procedure File File Batch Jobs Killing Batch Jobs Accessing Other Networks Checking Host Accessibility Transferring VMS Files To and From OASYS Transferring VMS Files To and From CDC CYBER 860 Mail to Users at Other Sites Mail From Users at Other Sites | 5-1-1 5-1-1 5-1-2 5-1-2 5-1-2 5-1-3 5-1-4 5-1-4 5-1-4 5-1-4 5-1-5 5-1-5 5-1-5 5-1-6 5-1-6 5-1-7 5-1-8 |

6

| Help Libraries The System Help Library DTRC Help Libraries User Help Module Hints For Designing Help Displays Selecting (Sub)topic Names Modify a Help Library Compress a Help Library List the Contents of a Help Library Extract a Help Module Accessing your Help Library Adding Your Help Library to the System Helps Using HELP Sample Help Modules A Program A Subprogram A Command Procedure General Information "HELP" module | 5-2-1 $5-2-1$ $5-2-2$ $5-2-3$ $5-2-3$ $5-2-4$ $5-2-4$ $5-2-5$ $5-2-5$ $5-2-5$ $5-2-5$ $5-2-6$ $5-2-7$ $5-2-7$ $5-2-7$ $5-2-7$ $5-2-7$ $5-2-7$ $5-2-10$ $5-2-11$ $5-2-12$ |
|---|--|
| | J D 1 D |
| Procedures | 5-3-1 |
| DTRC Procedures | 5-3-1 |
| Object Libraries DTRC Object Library User Object Module Modify an Object Library Compress an Object Library List the Contents of an Object Library Extract an Object Module Linking with an Object Library Text Libraries DTRC Text Libraries User Text Module Create a Text Library Modify a Text Library Compress a Text Library List the Contents of a Text Library Extract a Text Module | 5-4-1 $5-4-1$ $5-4-2$ $5-4-2$ $5-4-3$ $5-4-3$ $5-5-1$ $5-5-1$ $5-5-1$ $5-5-1$ $5-5-1$ $5-5-2$ $5-5-3$ $5-5-3$ $5-5-3$ |
| Editors The EDT Text Editor Invoking EDT Terminating EDT The EVE Editor Invoking EVE The Screen On-line Help Terminating EVE Why Use EVE Instead of EDT? | 5-6-1 5-6-1 5-6-1 5-6-2 5-6-2 5-6-2 5-6-2 5-6-2 5-6-2 5-6-2 5-6-2 5-6-2 5-6-2 5-6-2 |
| Magnetic Tape Tape Labels Tape Formats | 6-1-1 6-1-1 6-1-1 |

| Tape Care and Cleaning | 6-1-2 |
|------------------------------|-------|
| Tape Assignment | 6-1-3 |
| Using Tapes on the DEC VAX | 6-1-4 |
| Examples | 6-1-5 |
| Using Tapes on the CYBER 860 | 6-1-6 |
| NOS | 6-1-6 |
| Examples | 6-1-6 |
| NOS/VE | 6-1-7 |
| Examples | 6-1-7 |
| - | |
| | |

7 Other Software

7-1-1

Appendices

| A | Appendix A | | | A-1 |
|---|------------|------------|-----|-----|
| | ASCII Cha | racter Set | | A-1 |
| | CDC NOS | Character | Set | A-3 |

BAppendix BB-1Cray UNICOS CommandsB-1

| C-1 |
|--------|
| C-1 |
| C-2 |
| C-2 |
| on C-4 |
| C-4 |
| C-5 |
| |

DAppendix DD-1DEC VMS DCL CommandsD-1Selected DEC VAX/VMS CommandsD-2Selected DEC VAX/VMS AdditionsD-6Cray Station CommandsD-30Cray Context CommandsD-32

EAppendix EE-1ReferencesE-1CrayE-1DECE-1CDC NOSE-1CDC NOS/VEE-1GeneralE-2

| F | Appendix F | F-1 |
|---|----------------------|-----|
| | CCF Computer Systems | F-1 |
| | Services and Support | F-4 |

| - | | ٠ | ٠ | |
|------|----|---|---|--|
| Page | V1 | 1 | 1 | |

'

| G | Appendix G | G-1 |
|---|-------------------------|-----|
| | Internal Data Structure | G-1 |
| | Internal Representation | G-6 |
| | CRAY X-MP | G-6 |
| | DEC VAX | G-7 |
| | CDC CYBER (NOS, NOS/BE) | G-8 |
| | CDC CYBER (NOS/VE) | G-9 |
| | | |
| | | |

| G1 | Glossary | G1-1 |
|----|----------|------|
| | | |

Index

Index-1

Abstract

The Computer Center in the Computer and Information Services Department of the David Taylor Research Center has installed an Integrated Supercomputer Network. This manual provides an introduction to the Network. Some information has been distilled from many individual documents and augmented to reflect usage at DTRC. Control statement examples and descriptions of hardware and software are included, as is information on moving files among the CDC CYBER 860A (with the Mass Storage System), the DEC VAXcluster, the secure DEC VAX, and the CRAY X-MP, creating and executing batch jobs, and using the interactive systems. Volume 1 describes the CRAY X-MP, the Mass Storage System, and the DEC VAXes. Volume 2 describes the CDC CYBER 860A.

Administrative Information

The work described in this report was performed in the Scientific and Engineering Support Branch (3511) of the Computer and Information Services Department, David Taylor Research Center, under the sponsorship of the DTRC Computer Center (351).

Page 1-1-1

***** Introduction *****

The DTRC Integrated Supercomputer Network consists of a CRAY X-MP/216 with five front-end computers: a DEC VAXcluster (two VAX 8550 processors), a secure DEC VAX 6410, a DEC VAX 8250, and a CDC CYBER 180/860. The Cray and VAXcluster can store and retrieve files on the Mass Storage System (MSS), which is part of the CDC CYBER 860.

The following operating systems are in use:

| CRAY X-MP | COS vers | sion 1.17.1 | |
|---------------|-------------|-------------|----------|
| | UNICOS | | (future) |
| DEC VAX | VMS vers | sion 5.3-1 | |
| CDC CYBER 860 | NOS vers | sion 2.7.1 | |
| | NOS/VE vers | sion 1.5.1 | |

The front-end computers support both batch processing of jobs submitted at central site, through remote batch terminals or from interactive terminals; and demand processing, which supports a variety of interactive terminals. In addition, batch jobs can be sent to the Cray for processing with the output returned for examination or printing.

This reference manual is divided into two volumes: one covering the CRAY X-MP, the Mass Storage System, and the DEC VAXes; the other covers the CDC CYBER 860. They are designed to provide the new user with enough information to use the Network to run simple batch jobs and to create and run programs and batch jobs interactively. Most of the frequently used control statements are described in detail in the Appendices. Magnetic tapes are discussed briefly. No attempt is made to describe all features of the operating systems or even all parameters of the control statements presented. More information can be found in the publications listed in Appendix E.

Before using the system, job order number(s) to be charged must be registered with Code 3502. Outside users must transfer funds to DTRC before receiving a job order number. Each individual user should have 4-character User Initials assigned (also by Code 3502). *** Hardware Configuration ***

** CRAY X-MP/216 **

DTNET name: sn417 (UNICOS only, when available) TCP/IP name: sn417 (UNICOS only, when available) Ethernet address: 192.91.138.5 (UNICOS only, when available) Cray station ID: C1

2 X-MP central processing units (over 200 MFLOPS each) 16M 64-bit words of central memory 4 model DD-49 disk storage units (4.8 Gbytes) 2 model DS-41 disk storage units (9.6 Gbytes)

** CDC CYBER 180 model 860 **

DTNET names: cdc860, nos TCP/IP names: cdc860, nos Ethernet address: 130.46.1.16 Cray station ID: N1 N_ (UNICOS only, when available) Network ID: MFN

1 CYBER 860A central processing unit (6.3 mips) 2M 60-bit word memory 25 peripheral processors 3 model 895 disk drives 4 model 679-5 nine-track tape drives (1600/6250 cpi) 2 model 679-3 nine-track tape drives (800/1600 cpi) 2 model 677-3 seven-track tape drives 1 model 405 card reader 1 model 415 card punch line printers (1200 lpm, upper/lower case) 2 model 585 1 model 7990 Mass Storage System (210 Gbytes) 3 model M861 storage modules

91/01/15 Rev0

Page 1-1-3

** DEC VAXcluster ** VAXcluster nodes: DT3, DT4 DTNET names: dt3, dt4 TCP/1P names: dt3, dt4 Ethernet addresses: 130.46.1.12, .10 Cray station IDs: V3 V4 (UNICOS only, when available) ۷_ 2 VAX 8550 processors (6 mips each; DT3, DT4) -- each with 48 Mbyte 32-bit words of central memory 2 model SA482 disk storage array (5.0 Gbytes) 6 model RA81 disk drives (7.2 Gbytes) 1 model TA79 nine-track tape drives (1600/6250 cpi) 3 model TU79 nine-track tape drives (1600/6250 cpi) 2 model LP27 impact printers (800 lpm, upper/lower case) ** ** DEC VAX DOE * VAX node: DTNET name: doe TCP/IP name: doe Ethernet addresses: 130.46.1.13 1 VAX 8250 processors (1.2 mips) 16 Mbyte 32-bit words of central memory 4 model RA81 disk drives (1.6 Gbytes) 2 model TU81 nine-track tape drives (1600/6250 cpi) ** DEC Secure VAXcluster ** VAXcluster nodes: DBL07 Cray station IDs: (future) 1 VAX 6410 processors (7 mips) 64 Mbyte 32-bit words of central memory 2 model SA482 disk storage array (5.0 Gbytes) 6 model RA81 disk drives (2.4 Gbytes) 2 model TA78 nine-track tape drives (1600/6250 cpi) 1 model LP27 impact printers (800 1pm, upper/lower case)

* DOE is reserved for a special project and is not available for

general use.

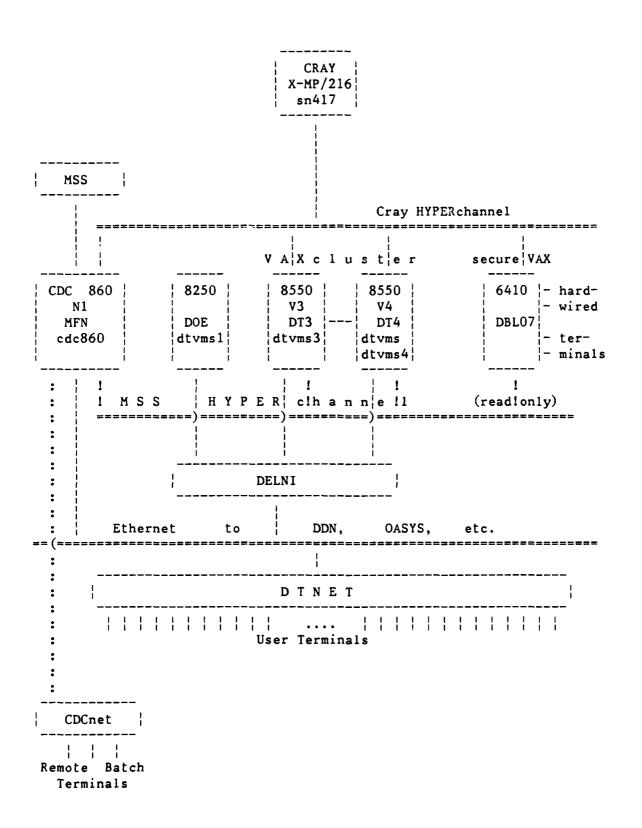
** DTNET Dial-in **

Access to Computer Center computers is via DTNET:

| Modem Speed | Carderock | Annapolis | |
|--------------------------|--|---------------------------------|------------------------------------|
| 1200 baud | (301) 227-5200 (32) | (301) 267-2010 (8) x4741 (8) | (within Annapolis site only) |
| 2400 baud 9600 baud * | (301) 227-5250 (32) (301) 227-3700 (16) | | - |

* While 9600 baud modems support lower speeds, access at a lower speed is not guaranteed.

*** The Integrated Supercomputer Network ***



Page 1-2-1

***** User Interface With the Computer Center *****

*** General Information ***

The ADP Control Centers are located at Central Site at Carderock and Annapolis. You may submit tapes and pick up output from an ADP Control Center.

Computer and Information Systems Department Newsletter is our quarterly publication. The date of the latest on-line news update is printed at the start of each batch job (log) or interactive session. The NEWS command or procedure is used to view the current news file.

*** Registering ***

To register to use Computer Center computers, call our Business Office, Code 3502, at (301) 227-1361/1910. Be prepared to supply

- . your name
- . your DTRC code or non-DTRC company name and address and telephone
- . the job order number(s) to be charged for computer work
- . the computers on which you which to be registered
 - . CRAY X-MP
 - . DEC VAXcluster
 - . DEC secure VAX
 - . CDC CYBER 860 (NOS or NOS/VE)

Registration for the DEC VAXcluster or CRAY X-MP includes registration for the Mass Storage System (CDC CYBER 860 (NOS)).

You will be given

- . User Initials (if you are a new user)
- . the initial passwords (which MUST be changed during your first session) for each computer system for which you registered

Page 1-2-2

Rev0 90/09/30

** Multiple Accounts **

The following is a discussion of multiple accounts: how to log in, how to charge batch jobs, how to change permanent file accounting.

* Cray COS *

Interactive

and batch The account number is supplied by the AC= parameter of the ACCOUNT statement at the start of each batch or interactive session:

ACCOUNT, AC=....

Permanent

files

Files are normally charged to the interactive or batch session's account number. The "ALTACN, AC=ac." statement is used to change the number for future file saves. It remains in effect until another ALTACN statement is encountered, or until end-of-session.

The account number for an existing file can be changed to current session's account number by

ACCESS, DN=PROCLIB, OWN=PUBLIC. LIBRARY, DN=PROCLIB:*. NEWCHRG, OLD=oldchrgno, ID=id.

where ID=id changes only those files with matching ID ID changes only those files with a null ID no ID changes all files.

Alternatively, this can be done from a VAXcluster node which connects to the Cray. CNEWCHRG generates and submits a Cray job to make the changes.

CNEWCHRG upw old_ac [new_ac] [id] [wait] [type]

where upw is your user password ID=id changes only those files with matching ID ID changes only those files with a null ID no ID changes all files wait is WAIT to wait the the Cray job to complete type is TYPE to display the generated Cray job

Page 1-2-3

* VAX *

VAX users with more than one account are assigned a username/ password for each account. These usernames differ in the fifth character position, e.g., ABCD, ABCDA, ABCDB. The default login directory for each user is device: [username] where all files owned by the same individual are stored on the same device. For example,

U01: [ABCD] U01: [ABCDA] U01: [ABCDB]

The "usernames" belonging to a particular user are members of a VMS "group". By default on the VAXcluster, members of a group have Read and Execute access to all files owned by their fellow group members. User ABCDA wishing to access a file owned by ABCD simply references [ABCD] FILE.EXT .

These access rights can be changed by the SET PROTECTION and SET FILE /ACL commands. In addition, all members of these special "groups" have GRPPRV privilege which, when invoked, gives a member of the group full control, including file creation and deletion, over all files owned by all members of the group. GRPPRV is invoked by

\$ SET PROCess /PRIVileges=GRPPRV

(this would likely be in your LOGIN.COM)

Then to "copy" a file from one account to another, for example from ABCD to ABCDA, user ABCDA would

\$ COPY [ABCD] FILE.EXT []

or user ABCD would

\$ COPY FILE.EXT [ABCDA]

To simply "move" a file from one account to another, ABCDA would

The command MYACcount will indicate the account number of the current session or job, while MYACcount /ALL will provide a list of all user/account pairs in the group. Page 1-2-4

Rev0 90/09/30

* CDC NOS *

- Interactive You are normally prompted for the account number for the session when you log in. (See Appendix I: CHVAL,CN.)
- Batch A batch job is charged to the number appearing on the actual (CHARGE, accountno.) or implied (/CHARGE) statement at the start of the job.

Permanent

files Files are normally charged to the interactive or batch session's account number. The CHARGE statement is used to change the number for future file saves. It remains in effect until another CHARGE statement is encountered, or until end-of-session.

The account number for an existing file can be changed to current session's account number by

CHANGE, pfn/CP.

To change several files, use

BEGIN, NEWCHRG, , fn.

where fn may have wildcards. To change all files, use

BEGIN, NEWCHRG.

. .

Page 1-2-5

* Mass Storage System *

The MSS is part of the CDC CYBER 860. The account numbers on files may be changed from the DEC VAXcluster, the CRAY X-MP, or on the 860.

From the DEC VAXcluster: \$ HFT ACCESS /Password=password \$ MSSNewchrg old_ac new_ac ^-- all MSS files with old_ac are changed

From the CRAY X-MP (COS): MSACCES,MPW=mss_pw. <-- may need AC=ac MSCHANG,MDN=mssfile,CP=newacctno.

On the CDC 860 (NOS) : CHANGE,mfn/CP.

From the CDC 860 (NOS/VE): MSACCES mss_pw MSCHANG mfn CP=1 *** Passwords, Passwords, Everywhere ***

Each computer system has its own password to gain access to it (while CDC NOS has two passwords, the CDC CYBER 860 at DTRC has only one). You MUST change these during your first session on each or you will be denied future access. For security, you are strongly urged to change your access passwords as soon as you can log into each computer. Passwords for all our computers expire in 90 days. For the VAX and CYBER, you must change your password during the session in which you are told it has expired. The new password must differ from the old one. On the VAX, it must differ from all others used during the previous 12 months.

To change your access passwords, use

. Cray COS (password is 4-15 characters, with at least one of the following characters: 0-9, \$, %, @)

. on the Cray

ACCOUNT, US=username, AC=joborderno, UPW=current pw, NUPW=new pw.

. from the VAXcluster

CSUBMIT /NUPW <-- use this form if you use CSUBMIT current_pw to submit jobs to the Cray new_pw new_pw

-or-

CNEWPW current_pw new_pw new_pw ac wait ^-- use this form if you do not use CSUBMIT

See also page 3-1-2.

 DEC VAXcluster (password is 6-12 characters) DEC secure VAX
 SET PASSWORD <--- you will be prompted for your current and new passwords

Page 1-2-7

. CDC CYBER 860 / MSS (NOS) (password is 4-7 characters, at least one number)

. from the VAXcluster

HFT PASSWORD <--- you will be prompted for your current and new passwords

. from the Cray (COS)

MSACCES, MPW=mpw. MSPASSW, OLD=oldpw, NEW=newpw.

. on the 860 (NOS)

PASSWOR, oldpw, newpw.

. from the 860 (NOS/VE)

MSACCES mpw MSPASSW oldpw newpw

. CDC CYBER 860 / MSS (NOS/VE) (password is up to 31 alphanumeric characters and underscores, starting with a letter)

Since NOS/VE is reached via NOS, your NOS login password gives you access to NOS/VE. However, to run NOS/VE batch jobs, you must have defined your batch password. To change it,

SET_PASSWORD <-- you will be prompted for your current and new passwords Page 1-2-8

Rev0 90/09/30

*** Trouble Forms ***

A Trouble Form is used:

- 1) for refund requests
- 2) when problems are encountered
- 3) for suggestions, gripes and complaints.

The Trouble Form should include a succinct description of the problem and include as much documentation (dayfile or log, listings, dumps) as possible. It should be submitted to Code 3511 for processing.

Trouble Forms may be entered directly into the computer from any of the front-ends (VAXcluster, CYBER 860) using the GRIPE command. If supporting documentation (such as listings or dumps) is needed, please send it to Code 3511 (User Services).

*** Refunds ***

Requests for refunds on lost time must be accompanied by output of the run and a Trouble Form, and must be reported within five working days. Decisions on refunds will be made by Code 35.

*** ADP Control Center ***

The ADP Control Center has the following capabilities:

. assign, sell, clean, test and degauss magnetic tapes

. process Calcomp plots, and Xerox and microfiche output

. sell frequently-used terminal paper and ribbons

The following EAM facilities are available off-line at Central Site:

- . a small card interpreter
- . card punch
- . card verifier
- . shredder

See Appendix F for Computer Center telephone numbers.

.

***** Software Available *****

The following table lists the major software products and the computers where they are available. Type "HELP @CCF Software" on the VAXcluster for the latest version of this table.

| | DT3/DT4 8550 | DBL07 6410 | CRAY X-MP | CDC NOS | CDC NOS/VE |
|-----------------|-----------------|---------------|--------------|------------|---------------|
| | | | | | |
| ABAQUS | x | - | x | - | - |
| ACSL | × - | - | - | - | - |
| ALGOL | | - | - | x | - |
| APL | x | - | - | X | - |
| APT | - | - | - | x | - |
| Basic | x | x | - | x | x |
| c (cc) | x | - | x | - | - |
| Calcomp | x | - | x | x | (1) |
| CDD | x | x | - | - | - |
| Cobol | x | x | - | x | x |
| Datatrieve | x | x | - | - | - |
| DBMS | x | - | - | - | - |
| DECalc | x | x | - | - | - |
| DISSPLA | x | - | x | x | - |
| DYNA 3 D | - | - | x | - | - |
| EISPACK | x | - | x | - | (1) |
| FMS | x | x | - | - | - |
| Fortran | x | x | x | x | x |
| FTP | x | - | - | - | - |
| GPSS | x | - | - | x | - |
| Hasp | _ | - | - | x | - |
| HOTSPOT | - | - | - | x | - |
| IMSL | x | - | x | x | x |
| INGRES | (2) | - | - | - | - |
| Kermit | x | x | - | x | - |
| LINPACK | x | - | x | x | (1) |
| Macsyma | (2) | _ | - | - | - |
| MODE4A/4C (20 | | - | - | x | - |

(1) - coming

(2) - DT4 only

Page 1-3-2

.

| | DT3/DT4 8550 | DBL07 6410 | CRAY X-MP | CDC NOS | CDC NOS/VE |
|---------------|-----------------|---------------|--------------|------------|---------------|
| Nastran | | - | x | | |
| Pascal | x | x | x | x | x |
| Patran | x | - | - | - | - |
| PCA | x | - | - | - | - |
| Pert Time | - | - | - | x | - |
| PL/I | x | x | - | - | - |
| PLOT10 | х | - | - | - | - |
| Proj Mgt (PM) | x | - | - | - | - |
| RIM | x | - | - | - | - |
| Simscript | - | - | - | x | - |
| SPICE | - | - | x | - | - |
| SPY | - | - | x | - | - |
| TAURUS | x | - | - | - | - |
| TELNET | x | (1) | - | - | - |
| WIN/TCP | x | (1) | - | - | - |
| XMODEM | - | - | - | x | - |

(1) coming

***** Prolog and Epilog Command Files *****

A Prolog or Epilog Command File is a file containing commands which are executed automatically each time an event occurs. Prologs are executed at login, at the start of a batch job, at the invocation of an editor, etc. Each DTRC system has one or more such files. Epilogs are executed at logout, at the termination of an editor or utility. NOS/VE supports epilog files.

*** VAX/VMS Command Files ***

A system-level login file is executed to define global symbols for all users. User file LOGIN.COM, if present, will be executed for each user login or batch job. This procedure file may set up symbols and logical names, establish process name, test whether batch or interactive, or other desired JCL. Type HELP LOGIN.COM_HINTS for suggestions to include in your LOGIN.COM file.

If file EDTINI.EDT exists, it will be executed each time EDT is entered and may be used to define special keys, macros, or other desired Editor commands.

If file EVE\$INIT.EVE exists, it will be executed each time EVE is entered and may be used to define special keys, macros, or other desired Editor commands.

*** NOS Command Files ***

A LOGINPR procedure file may be created containing JCL desired for each login or batch job. For instance, after testing for interactive, TRMDEF could be used to set special desired characteristics for your terminal. Execution of this command file is not automatic, but is triggered by a one-time execution of the UPROC command. If your prolog is long, you may wish to include RECOVER processing.

When the FSE editor is entered, the STARTUP procedure from FSEPROC is executed. The system version of this procedure is currently empty. If you wish to define editor directives which are always executed, make a copy of FSEPROC in your files with desired modifications. NOS will then execute your FSEPROC file instead of the system file.

LOGINPR and FSEPROC must be indirect files.

Rev0 90/09/30

*** NOS/VE Command Files ***

A PROLOG command file may be created containing SCL desired for each login or batch job. For instance, after testing for interactive, CHANGE_TERMINAL_ATTRIBUTES could be used to set special desired characteristics for your terminal.

An EPILOG command file may be created to be executed at each logout or batch job termination.

When EDIT_FILE is entered, the commands in file SCU_EDITOR_PROLOG are executed. For instance, if you know the keypad or have a layout of it, you may wish to put "SET_SCREEN_OPTION MENU_ROWS=0" into this file to use the bottom line(s) of the screen for editing instead of for displaying the keys.

*** Cray Command Files ***

Before the Cray is used from NOS for the first time, and each time the NOS password is changed, CDEFINE should be used to create a CRAYDEF file. This file frees the user from including CYBER user and password information in each Cray JCL file.

If ICF (Cray Interactive Facility) will be used from NOS, a default PLAY file named ICFPROF may be created which will be executed at each ICF activation. It should contain commands such as /PERIOD to eliminate the need to end each Cray command with a period.

More information about VAX/Cray procedures such as CXCOMMANDS.COM will be included later.

Page 1-5-1

***** ANSI Standard Multi-file Tapes *****

ANSI standard multi-file labelled tapes have each data file delimited by HDR1 and EOF1 labels. In the HDR1 label is the set ID field.

*** VAX ***

The set identifier field is set to the VSN when utilities such as COPY are used to write a multi-file reel.

*** NOS ***

The user defines the multi-file set identifier at the first write of the LABEL with the SI parameter. For ease of recall and compatibility with VAX, the VSN is recommended for the SI. All future reads/writes must use the previously-defined set identifier. If a VAX multi-file reel is read on NOS, the SI field must be set to VAX VSN. If FCOPY is used to create a multi-file tape to be read on a VAX, each FCOPY must be followed by WRITEF to properly create the EOF labels.

*** NOS/VE ***

The multi-file set identifier is created by using the FILE_SET_ IDENITIFIER parameter on the initial CHANGE_TAPE_LABEL_ATTRIBUTES command. The label for each successive file is entered on additional CHANGE_TAPE_LABEL_ATTRIBUTE commands. File specifications, i.e., minimum/maximum record and block sizes, are defined with the SET_FILE_ ATTRIBUTES command.

Page 2-1-1

***** The CRAY X-MP (UNICOS) *****

The CRAY X-MP/216 at DTRC is a powerful, general purpose computer having two central processing units (CPUs) which share files and are linked together. These CPUs share 16 million 64-bit words of memory. Each CPU achieves its extremely high processing rate (over 200 MFLOPS (million floating point operations per second)) using its scalar and vector capabilities.

*** UNICOS Version 5.1 ***

One operating system for the CRAY X-MP at DTRC is the Unix Cray Operating System (UNICOS), version 5.1, which supports both batch and interactive processing. UNICOS is expected to be available at DTRC within the next year.

***** The CRAY X-MP (COS) *****

The CRAY X-MP/216 at DTRC is a powerful, general purpose computer having two central processing units (CPUs) which share files and are linked together. These CPUs share 16 million 64-bit words of memory. Each CPU achieves its extremely high processing rate (over 200 MFLOPS (million floating point operations per second)) using its scalar and vector capabilities.

*** COS Version 1.17.1 ***

The operating system for the CRAY X-MP at DTRC is the Cray Operating System (COS), version 1.17.1, which supports both batch and interactive processing.

*** Accessing the CRAY X-MP ***

Batch jobs are normally submitted from one of the front-ends using: CSUBMIT or CRAY SUBMIT on the VAXcluster, or CSUBMIT on the CDC CYBER 860. They may also be submitted from a running batch or interactive job using the Cray SUBMIT command.

Interactive access is also from one of the front-ends using: CINT (station code version 4.02) on the VAXcluster, or ICF (Interactive Cray Facility) on the CDC CYBER 860 (NOS).

Both modes of access are described later in this chapter.

*** Cray Datasets ***

On the Cray, information is organized by COS into datasets, which may be on disk, memory-resident, or interactive. A dataset contains one or more files and may be temporary (available only to the job that created it) or permanent.

Each dataset has a disposition code to tell COS what to do with it when it is released. The 2-character alphanumeric disposition codes include SC (scratch - default), PR (print), IN (input), and ST (stage to the front end).

Jobs access local datasets, which may be temporary or permanent. Permanent datasets are made local by the ACCESS statement. Front end files are made local by the FETCH statement. *** Changing your Cray password ***

Your Cray access password may be changed from a batch job or interactively on the Cray, or from a procedure on the VAXcluster which creates and submits a Cray batch job for you.

Via DCL command CSUBMIT:

\$ CSUBMIT /NUPW

You will be prompted for your old and new passwords and a Cray job will be submitted on your behalf to change your password. The database on the VAXcluster will be updated with your new password.

If you submit jobs using CSUBMIT, use CSUBMIT to change your password. If you use CRAY SUBMIT or CNEWPW to make the change, the database will not be updated until you use CSUBMIT/NUPW.

Batch:

\$ CRAY SUBMIT mynewpw.job

where your file MYNEWPW.JOB contains:

JOB, JN=ssss. ACCOUNT, AC=ac, US=us, UPW=current_pw, NUPW=new_pw.

Interactive:

\$ CINT \
Cray Jobname: jobname > or CINT /JN=jobname /US=username
Cray Username: username /
!ACCOUNT,AC=ac,US=us,UPW=current_pw,NUPW=new_pw.
!^Z <--- ctrl-Z
Cint> QUIT
\$ <--- you are back in DCL</pre>

Via DCL command CNEWPW:

This procedure creates and deletes temporary file N\$U\$P\$W.JOB.

Page 3-1-3

*** Batch Jobs ***

Cray batch jobs are very similar to CDC batch jobs, but with different terminology. A batch job consists of one or more files. The first file is the JCL control statement file. It is followed by source or data files as needed by the JCL file. A typical job consisting of one source and one data file (*) looks like this:

> JOB, JN=jobname,.... ACCOUNT, AC=job_order_number, US=username, UPW=password.

<JCL statements>

/EOF

<source file>

/EOF

<-- end-of-file

<-- end-of-file

<data file>

<eod>

<-- end-of-data

A Cray batch job has at least four datasets:

\$CS - the control statement file

```
(part of $IN, but not accessible to the user)
$IN - the job input dataset. Accessible by its local name,
$IN, or as Fortran unit 5.
```

- \$0UT the job output dataset. Accessible by its local name, \$0UT, or as Fortran unit 6.
- \$LOG a history of the job. Not accessible to the user. \$LOG is appended to \$OUT when the batch job terminates.

^{(*) -} When executing several programs or one program several times, the /EOF is required only when a program reads until end-of-file. If a program reads a specific number of data records, or has its own pseudo-end-of-file, the /EOF must NOT be present.

Page 3-1-4

Rev0 91/01/15

** Batch Job Classes **

Batch jobs fall into five service classes: NORMAL, DEFER, BUDGET, PZERO, and SECURE. The US= parameter of the JOB statement specifies the job class (there is no default job class). SECURE jobs may be submitted only from the secure VAX (see below). Type HELP RATES (on the VAX), BEGIN, RATES (on CDC NOS), or RATES (on CDC NOS/VE) for the current rates.

Each class, except SECURE and interactive, is broken into four subclasses determined by the memory requested. The first letter of each subclass is:

| S | (small) | <= | 2 | Mwords |
|---|---------------|----|----|--------|
| М | (medium) | <= | 8 | Mwords |
| L | (large) | <= | 12 | Mwords |
| Х | (extra large) | > | 12 | Mwords |

The subclasses for NORMAL are SNORM, MNORM, LNORM, XNORM; for DEFER: xDEFER; for BUDGET: xBUDGE; and for PZERO: xPZERO. The subclass names appear in the CRAY STATUS display.

The following chart shows for each job class: the maximum number of such jobs to be allowed to execute at the same time. They are listed in order of relative priority, highest priority first.

| | maximum # | | | jobs | | |
|-------------|-----------|---|---|------|---|--|
| job class | IA | S | M | L | Х | |
| | | - | - | - | - | |
| Interactive | 14 | | | | | |
| NORMAL | | 4 | | 2 | 1 | |
| NUKMAL | | O | 4 | Z | T | |
| DEFER | | 6 | 4 | 2 | 1 | |
| | | Ū | • | - | - | |
| BUDGET | | 6 | 4 | 2 | 1 | |
| | | | | | | |
| PZERO | | 5 | 3 | 1 | 1 | |
| | | | | | | |
| SECURE | | | | | | |

** SECURE Batch Job Class **

Classified processing may be done on the CRAY X-MP by making prior arrangement with Operations. Access to the Cray is available only from a terminal in the secure computer room connected to the secure VAX (node name: DBLO7). Batch jobs submitted to the Cray from this terminal must have "US=SECURE" in the job statement; jobs with "US= any_other_class", or with US= omitted, will be rejected. Secure jobs may be submitted at any time but will be executed only during classified time. Secure jobs may access the Mass Storage System in read-only mode.

Page 3-1-5

** From the VAXcluster **

To use the Cray from the VAXcluster, log in to a node which can access the Cray, prepare your Cray batch job using any editor, and submit the job file(s) to the Cray using the CSUBMIT command:

\$ CSUBMIT filename -or- \$ CRAY SUBMIT filename

or

\$ CSUBMIT file1, file2,... -or- \$ CRAY SUBMIT file1, file2,...

where filename is a VAXcluster file containing the Cray job (default file extension: .JOB)

The output will be returned to your file jobname.CPR, where jobname is taken from the job statement of the Cray job (JN parameter).

CSUBMIT remembers your Cray password. It ignores an ACCOUNT statement, if present, and creates one from CSUBMIT qualifiers and your VAXcluster login username and account. CSUBMIT can also change your login password. CRAY SUBMIT requires that there be an ACCOUNT statement in the jobfile.

Files sent to the Cray must not have embedded tabs. See Appendix D: DETAB.

Page 3-1-6

× Killing Batch Jobs * Cray jobs are identified by their Job Sequence Numbers (jsq). To find the jsq, use \$ CRAY STATUS -or-\$ CRAY CRAY> STATUS To kill a batch job, use \$ CRAY DROP jsq <-- terminate executing job with EXIT processing -or-\$ CRAY KILL jsq <-- delete a job from the input queue -or-</pre> terminate executing job without EXIT processing -ordelete the output dataset from the output queue * × VAXcluster-to-Cray Examples 1) \$ CSUBMIT JOB1 \$ CRAY SUBMIT JOB1 -orwhere JOB1.JOB contains: (1)JOB, JN=MYJOB. (1, 4)ACCOUNT, US=username, UPW=password, AC=account. CFT. (1)SEGLDR, GO. (1)/EOF (2) PROGRAM ADD (2) DO 10 I=1,5 READ (5, *) N1, N2, N3 (2) N = N1 + N2 + N3(2) WRITE (6, *) N1, N2, N3, N (2) **10 CONTINUE** (2) (2) END /EOF 1 2 3 (3) 4 5 6 (3) 7 8 9 (3) 10 11 12 (3) 13 14 15 (3) /EOF

will submit the job to the Cray with the output returned in file MYJOB.CPR. The ACCOUNT statement (4) is omitted (or ignored) when using CSUBMIT.

Page 3-1-7

- 2) \$ CSUBMIT RUN2.JOB,RUN2.FOR,RUN2.DAT -or-\$ CRAY SUBMIT RUN2.JOB,RUN2.FOR,RUN2.DAT
 - where RUN2.JOB contains the job control statements ((1) above) RUN2.FOR contains the Fortran source program ((2) above) RUN2.DAT contains the data ((3) above)

will submit the combined files to the Cray with the output returned in file MYJOB.CPR. Note that the /EOF records are not required in this format.

3) \$ CSUBMIT RUN3 -or- \$ CRAY SUBMIT RUN3

where RUN3.JOB contains:

JOB, JN=MYJOB. ACCOUNT, US=username, UPW=password, AC=account. (4) FETCH, DN=PROG3, TEXT='PROG3.FOR'. FETCH, DN=DATA3, TEXT='PROG3.DAT'. CFT, I=PROG3. SEGLDR, GO.

PROG3.FOR on the VAXcluster contains the program (2) above, with "OPEN (5, FILE='DATA3')" before the "DO 10 ..."

PROG3.DAT contains the data (3) above.

Page 3-1-8

Rev0 91/01/15

** From the CDC CYBER 860 **

To use the Cray from the CYBER 860, log in, prepare your Cray batch job using any editor, and submit the job file to the Cray using the CSUBMIT command (assumes you have set up the CRAYDEF file for your current CYBER 860 user name and password via a CDEFINE command):

| /CSUBMIT,lfn. | < | prim | nt at | Central Site |
|---------------------|---|------|-------|--------------------------|
| /CSUBMIT,1fn,RB=un. | < | put | into | output queue for user un |
| /CSUBMIT,1fn,TO. | < | put | into | your wait queue |

In the last two formats, use QGET to get the file from the queue. See Appendix D for additional parameters. To send the output elsewhere, use the Cray DISPOSE command (see Appendix C).

* Killing Batch Jobs *

Cray jobs are identified by their Job Sequence Numbers (jsq). To find the jsq, use

\$ CSTATUS

To kill a batch job, use

\$ CDROP, jsq. <-- terminate executing job</pre>

-or-

\$ CKILL, jsq. <-- delete a job from the input queue -orterminate executing job keeping only the dayfile -ordelete an output dataset

.

*

*

CYBER 860-to-Cray Examples 1) /CSUBMIT,RUN1. where local file RUN1 contains: JOB, JN=myjob. ACCOUNT, US=username, UPW=password, AC=account. FETCH, DN=prog3, SDN=myprog, TEXT='GET, myprog.CTASK.'. ^-- indirect file FETCH, DN=mydata, TEXT='ATTACH, mydata.CTASK.'. <-- direct file CFT, I=prog3. SEGLDR,GO. -or-JOB, JN=myjob. ACCOUNT, US=username, UPW=password, AC=account. FETCH, DN=prog3, SDN=myprog, TEXT='GET, myprog.CTASK.'. ^-- indirect file MSACCES, US=user, MPW=mspass. <-- direct file MSFETCH, DN=mydata. CFT, I=prog3. SEGLDR, GO. MYPROG and MYDATA on the CDC CYBER 860 contain the program and data (see page 3-1-7, example 3).

Page 3-1-10

Rev0 90/09/30

** From a Running Cray Job ** A batch job to be submitted from a running Cray job may reside either on the Cray or on one of the front-ends. From within the Cray job, ACCESS or FETCH the file to make it a local file, then SUBMIT it to the COS input queue. (See Appendix C for additional parameters for these Cray commands.) * Examples * 1) The job is in a permanent dataset on the Cray: JOB,.... ACCOUNT,.... . . . ACCESS, DN=myjob, PDN=mypermjob. SUBMIT, DN=myjob. . . . 2) The job is in a file on the VAXcluster: JOB,.... ACCOUNT,.... . . . FETCH, DN=myjob, TEXT='myjob.job'. <-- submitted from VAXcluster</pre> -or-FETCH, DN=myjob, MF=V3, TEXT='DT3"user pw"::UOn:[user]myjob.job'. ^-- submitted from CYBER 860 or VAXcluster SUBMIT, DN=myjob. • • • 3) The job is in a file on the Mass Store (CDC CYBER 860): JOB,.... ACCOUNT,.... . . . MSACCES, US=user, MPW=mspass. MSFETCH, DN=myjob. SUBMIT, DN=myjob. . . .

Page 3-1-11

*** Interactive Jobs ***

CRAY X-MP interactive access is via the Cray Station code on one of the front-ends.

> ** From the VAXcluster **

The CRAY X-MP is accessed via the VMS Cray Station, which may be entered by the CRAY or CINT command.

The CRAY command puts you into Cray context (indicated by the CRAY> prompt). Among other capabilities, you can examine and manipulate your jobs in the Cray queues.

The CINT command initiates an interactive session on the Cray. Τf not included in the CINT statement qualifiers, you will then be requested to supply:

Cray Jobname: <-- enter 1-7 alphanumeric characters as the</pre> jobname for this session (must be upper case to be able to fetch from the Mass Storage System; should be different for each session) Cray Username: <-- enter your User Initials

The exclamation prompt (!) indicates that you have reached the Cray. Your first command must be your ACCOUNT statement. Any other commands will be ignored until a valid ACCOUNT statement is read.

!ACCOUNT, AC=1222233344, UPW=pw, US=userinit.

When you receive another ! prompt, your logon was successful. You may now use any of the commands in Appendix C. Every command MUST end with a terminator (.): if you forget, use the up-arrow to bring the command back and add the terminator.

To execute some Cray Station commands, use ^Z to interrupt CINT. At the Cint> prompt, enter the Station command. When it completes, you will be returned to your interactive session. Note that only a subset of the Cray Station commands may be executed in CINT.

To terminate the Cray interactive session, enter ^Z. At the Cint> prompt, type QUIT to return to the VMS prompt and close the interactive session.

To leave the Cray Station, enter EXIT (or ^Z). This will bring you out of Cray context and back to the VMS prompt.





* VMS Cray Station Commands *

See Appendix D for the syntax of these commands. (CINT) indicates that the command may be executed interactively (via CINT) as well as from the Cray Station (CRAY).

\$ Create a temporary VMS subprocess, allowing you to enter DCL commands. To return to Cray context, type LOGOUT.

\$ (CINT only) Execute a single DCL command.

- Display the next page of information in Cray context.
- Display the previous page of information in Cray context.
- @ (CINT) Execute an indirect station command file (containing station commands) in Cray context. (Synonym for PLAY.)

^C (CINT only) CTRL-C -- Same as ABORT.

- ^0 (CINT only) CTRL-0 -- Toggles interactive output on and off until the next Cray prompt.
- [^]Z (CINT) C^TRL-Z exit the current processing mode. In response to the Cray context prompt (CRAY>), it returns you to DCL; during a Cray interactive session, it returns you to command mode (Cint> prompt). While you are being prompted for command parameters, CTRL-Z cancels the command. You can also terminate the execution of an indirect station command file with CTRL-Z. In response to the Cint> prompt, you are returned to your interactive session.
- ABORT (CINT only) Interrupt the current interactive Cray job step and return to the "!" prompt after first displaying any COS output queued for the terminal.
- ATTACH (CINT only) Redirect Cray interactive terminal output to an alternate device (graphics).
- ATTENTION (CINT only) Interrupt the current interactive Cray job step and enters reprieve processing. If no reprieve processing, ATTENTION is the same as ABORT.
- BYE (CINT only) Terminate an interactive session. Depending on the command qualifiers, the COS interactive job may also be terminated.
- CLEAR Terminate any display command and clear the display portion of the screen.
- COLLECT (CINT only) Store COS interactive output in a VMS file.
- COMMENT Insert comments into an indirect station command file stream.

Page >-1-13

DATASET Report the existence of a COS permanent dataset.

DELAY Suspend execution of an indirect station command file for a specified period of time.

DISCARD (CINT only) Discard all output from a Cray interactive session until the next COS prompt is issued.

DROP Terminate a Cray job and returns the associated output dataset. COS job execution enters reprieve processing after the next COS EXIT control statement.

EOF (CINT only) Send an end-of-file record to a connected COS interactive job. This command is normally required to terminate COS file input from the terminal.

EXIT (CINT only) Return you from Cray context or Cint command mode to DCL level.

HELP (CINT) Display information from the station help files or an index of all commands.

ISTATUS (CINT only) Return the status of your Cray interactive job, including the CPU time used and the last Cray logfile message.

JOB Display the status of a specific COS job.

JSTAT Display the status of a specific job and its related tasks.

KILL Terminate a Cray job immediately.

LOGFILE Provide access to the station logfile messages.

LOOP Restart execution of an indirect station command file at the beginning of the file. End looping with ^Z.

MESSAGE Send a message to the Cray job and station logfiles.

PAUSE Suspend the execution of an indirect station command file. Control is passed to the terminal, where you can terminate the command file by entering a command or resume it by entering a null line (<RET>).

PLAY (CINT only) Execute an indirect station command file. (Same as @.)

QUIT (CINT only) Terminate a Cray interactive session and the corresponding Cray interactive job. (Equivalent to SYE/ABORT.)

- RECORD Start or stop the recording of terminal input to the specified file while in Cray context for later use with the PLAY or @ commands. Exiting Cray context automatically issues a RECORD/OFF.
- RELEASE Release a dataset in the output HOLDING queue due to VMS quota problems.

REMOVE Delete entries in the dataset staging queue.

- RERUN Immediately end the processing of a COS job and put job back into the input queue, unless the job has terminates or cannot be rerun.
- SET Define terminal working environment for the current session.
- SHOW Display information about the status of the station staging queue.
- SNAP Copy the current contents of the display region into the specified VMS file. If the command is issued from a terminal in line-by-line mode, the last display requested is recorded in the file.
- STATCLASS Display the current Cray job class structure.
- STATUS Display the Cray system status.
- STATUS (CINT only) Same as ISTATUS.
- STORAGE Initiate a COS disk status display providing the following information: device class or status; device name as it is known to COS; percentage of free space and permanent space on each device; number of recovered and unrecovered errors on each device; location of last error.
- SUBMIT Stage the specified VMS file to Cray to be put on the job input queue. The file must contain COS JCL (see CRAY HELP). The first record must be the JOB control statement. By default, the output from the COS job (known as a logfile) is sent to the directory from which the job was submitted.
- SUPPRESS Suppress the echoing of the next typed input line.

SWITCH Set or clear COS job sense switches.

Page 3-1-15

^{*} Examples ^{*}

 \$ cint Cray Jobname: ABCD001

> Cray Username: ABCD !ACCOUNT,AC=1222233344,UPW=mypw.

!<your Cray commands>
!^Z
Cint> <Station command>
!^Z
Cint> quit

<-- US=abcd not needed since
 upper case was used in
 entry above
<-- ctrl-Z to leave Cray
<-- returns to Cray when done
<-- ctrl-Z to leave Cray
<-- terminate Cray session</pre>

above

since lower case was used in entry

System) <-- user ABCD

<-- any jobname (must be upper</pre>

case to be able to fetch from the Mass Storage

2) \$ cint Cray Jobname: efgh002 Cray Username: efgh !ACCOUNT,AC=1222233344,UPW=mypw,US=efgh. <-- US=efgh needed</pre>

<same as example 1>

3) \$ cint /jn=struct /us=efgh <-- any jobname; user EFGH (since this is a VMS DCL command, it is converted to upper case) <-- US=efgh not needed since the VMS control statement is converted to upper case

<same as example 1>

** From the CDC CYBER 860 **

The CRAY X-MP is accessed via the NOS Interactive Cray Facility (ICF), which may be entered by the APPSW, ICF command from IAF. You enter ICF, log onto the Cray, do your thing (Cray or ICF commands), leave the Cray and ICF. You will then be at the NOS prompt.

Alternatively, you can specify ICF as the application when you log into NOS.

ICF commands have a prefix (normally a slash "/") and can be intermixed with Cray commands. To terminate the Cray session (and ICF), enter /BYE or /LOGOFF.

* ICF Prologue File *

A user prologue may be defined to be executed each time you start an ICF session. This is an 8/12-bit ASCII indirect access file named ICFPROF, with access granted to user SYSTEMX (PERMIT, ICFPROF, SYSTEMX.) We suggest you include "/PERIOD ON" so you don't have to type a period at the end of each Cray command.

* NOS ICF User Commands *

- /ABORT Send abort interrupt to the interactive Cray job (also user-break-2 key (normally %2).
- /ATTENTION Send attention interrupt to the interactive Cray job (also user-break-1 key (normally %1).
- /BYE Terminate this Cray interactive session. (Same as /LOGOFF)
- /CONNECT Create a logical connection between this terminal and some other (slave) terminal.
- /DISCARD Discard output being sent from the Cray to this terminal.

/ENDCONNECT Terminate a CONNECT.

/ENDPLAY Terminate reading of a PLAY file.

/EOF Send an end-of-file to the Cray.

/HELP Display help information.

- /ICFSTATUS Display general information about the current status of ICF.
- /LOGOFF Terminate this Cray interactive session. (Same as /BYE)

/LOGON Initiate or reconnect to an existing Cray job.

90/09/30

Rev0

Page 3-1-17

- /PERIOD Set/reset automatic generation of a terminator on COS commands.
- /PLAY Read data and commands from a NOS file in the user's catalog.

/PREFIX Change the ICF command prefix letter.

/QUIT Immediately terminate this Cray interactive session.

/RESUME Resume the transmission of data to and from the Cray (negate the effect of SUSPEND).

/SUSPEND Suspend transmission of data to and from the Cray.

/STATUS Display Cray status.

/* An ICF comment line.

> $_{\star}$ Examples *

- 1) /appsw.icf <--- / is the NOS prompt <a greeting> /logon <a greeting> !account,ac=1222233344,upw=mypw. !<your Cray or ICF commands> !/bye
- 2) FAMILY: ,abcd, pw, icf <a greeting> /logon <a greeting>

!account,ac=1222233344,upw=mypw. !<your Cray or ICF commands> 1/bye T1210 - APPLICATION: iaf

<-- / is required; log onto DTRC Cray <-- US=abcd not needed <-- to leave Cray and ICF <--- log into ICF directly <--- / is required; log onto DTRC Cray <-- US=abcd not needed <-- to leave Cray and ICF <-- switch to another

application such as IAF

Page 3-2-1

***** Cray JCL Commands *****

The Cray Job Control Language (JCL) statements are grouped by function in this section. See Appendix C for a description of the syntax for each command. (DTRC) indicates a command or program added at DTRC. Some of the logic structure commands use JCL expressions, which are described later in this section.

*** Job Definition and Control ***

* Entire line is a comment.

ACCOUNT Validate a user's Job Order Number, user name and password.

ALTACN Validate an alternate account number for permanent datasets.

CALL Read control statements from another file.

CHARGES Report on job resources.

ECHO Control logfile messages.

- EXIT On job abort, processing continues with the statement following the EXIT; if no abort, terminate job processing.
- IOAREA Control access to a job's I/O area (containing the DSP and I/O buffers).

JOB First statement of a job -- gives job parameters.

JOBCOST (DTRC) Write a summary of job cost and system usage to \$LOG.

LIBRARY Specify search order for procedures during processing.

MEMORY Request new field length.

MODE Set/clear mode flags.

NORERUN Control a job's rerunability.

OPTION Specify user-defined options.

RERUN Control a job's rerunability.

RETURN Return from an alternate control statement file.

ROLLJOB Protect a job by writing it to disk.

SET Change value of a JCL symbolic variable.

SWITCH Turn pseudo sense switches on or off.

Page 3-2-2

Rev0 90/09/30

*** Dataset Definition and Control ***

ACCESS Make a permanent dataset local.

ASSIGN Create a dataset and assign dataset characteristics.

HOLD Dataset release occurs with implicit HOLD.

NOHOLD Cancel effect of HOLD.

RELEASE Relinquish access to a dataset from a job.

*** Permanent Dataset Management ***

ACCESS Make a permanent dataset local.

ADJUST Redefine size of a permanent dataset.

DELETE Remove a permanent dataset.

MODIFY Change a permanent dataset's characteristic information.

MSCHANG (DTRC) Change the attributes of a Mass Storage System file.

NEWCHRG (DTRC) Change permanent file account number.

PERMIT Grant/deny access to a permanent dataset.

SAVE Make a dataset permanent.

SCRUBDS Write over a dataset before release.

*** Permanent Dataset Staging ***

See Chapter 3 for staging to and from the Mass Storage System.

ACQUIRE Get a front-end dataset and make it permanent.

DISPOSE Stage dataset to the front-end; release a local dataset; change disposition characteristics.

FETCH Get a front-end dataset and make it local.

MSACCES (DTRC) Supply your Username and password to the Mass Storage System (MSS).

MSFETCH (DTRC) Fetch a file from the MSS.

MSPURGE (DTRC) Purge a file from the MSS.

MSSTORE (DTRC) Store a file on the MSS.

SUBMIT Send local dataset to COS input queue.

Page 3-2-3

*** *** Permanent Dataset Utilities Report on permanent datasets. AUDIT *** *** Local Dataset Utilities BLOCK Convert an unblocked dataset to a blocked dataset. COPYD Copy blocked datasets. COPYF Copy blocked files. COPYNF Copy files from one blocked dataset to another. COPYR Copy blocked records. COPYU Copy unblocked datasets. List local datasets. DS Write text to a dataset. NOTE **OUERY** Determine the current status and position of a local file. REWIND Position a dataset at its beginning. SKIPD Skip blocked datasets (position at EOD (after last EOF)). SKIPF Skip blocked files from current position. SKIPR Skip blocked records from the current position. SKIPU Skip sectors on unblocked datasets. UBBLOCK Convert a blocked dataset to an unblocked dataset. Initialize a blocked dataset by writing a single file WRITEDS containing a specific number of records of a specific length. *** <u>**</u>* Dumps and Other Aids COMPARE Compare two datasets. DEBUG Interpret a dump. DUMPJOB Capture job information in dataset \$DUMP for display by DUMP. DUMP Display job information previously captured by DUMPJOB. FLODUMP Dump flowtrace table.

Rev0 90/09/30

FTREF Generate Fortran cross-reference.

ITEMIZE Report statistics about a library dataset.

PRINT Write value of JCL expression to the logfile.

SPY Generate a histogram of time usage within a program to locate inefficient code.

*** Logic Structure ***

ELSE IF-loop control.

ELSEIF IF-loop control.

ENDIF IF-loop termination.

ENDLOOP LOOP termination.

EXITIF IF-loop control.

EXITLOOP LOOP control.

IF Begin a conditional block of code.

LOOP Start of an iterative control statement block.

*** Procedures ***

See Section 3-3 for additional information on the creation of procedures.

CALL Transfer control to a procedure.

"call by name" Execute a complex procedure in a library.

ENDPROC End of a procedure.

PROC Begin an in-line procedure definition block. This is followed by the procedure prototype statement which names the procedure and gives the formal parameter specifications.

RETURN Return control from a procedure to its CALLer.

*** Programming Languages ***

CFT Compile a Fortran source program.

CFT77 Alternate Fortran compiler (slower compile, faster execute).

PASCAL Compile a Pascal source program.

*** Program Libraries ***

See Section 3-4 for a discussion of program libraries (PL).

AUDPL Audit an UPDATE PL.

UPDATE Source and data maintenance.

*** Object Libraries ***

See Section 3-5 for a discussion of object libraries.

BUILD Generate and maintain library datasets.

SEGLDR Segment loader (see Section 3-6).

*** Miscellaneous ***

"call by name" Execute a program by its local file name.

SID Debug programs interactively or in batch.

SORT Sort/merge.

Page 3-2-6

Rev0 90/09/30

*** JCL Expressions ***

An expression is a string of operands and operators. It is evaluated from left to right, taking into account parentheses and operator hierarchy. Expressions allow the incrementing of counters, error code checking, and string comparison.

There are four types of operands:

- . symbolic variables (see below)
- . subexpressions (its value becomes an operand)

Expressions may be used in IF, ELSEIF, EXITIF, and EXITLOOP.

** Symbolic Variables **

There are 38 symbolic variables: 6 system constants, 7 variables set by COS, and 25 which can be set by the user.

* System Constants *

| Symbol | Range | Description |
|--------|---------|------------------------|
| FALSE | 0 | False |
| SID | literal | Mainframe ID (C1) |
| SYSID | literal | COS level ('COS n.nn') |
| TRUE | -1 | True |

SN and XM are also available.

* COS-set Variables *

| Symbol | Range | Description |
|-----------------|------------------|--|
| ABTCODE DATE | 0-nnn literal | COS job abort code (ABnnn) mm/dd/yy |
| FL | 0-77777777 | mm/dd/yy current octal field length |
| FLM | 0-7777777 | JOB statement maximum octal FL |
| PDMST | 64-bits | status of most recent Permanent Dataset Manager request |
| TIME | literal | hh:mm:ss |
| TIMELEFT | 64-bit integer | job time remaining (milliseconds) |

Page 3-2-7

| * User-set Variables * | | | | |
|------------------------|---------|---|--|--|
| Symbol | Range | Description | | |
| G0-G7 | 64-bits | 8 global pseudo-registers (can be used to pass data between procedures) | | |
| J0-J7 | 64-bits | 8 job (local) pseudo-registers (each procedure level has its own J registers) | | |
| JSR | 64-bits | Job Status Register containing the previous job step completion code | | |
| NOTEXT | 64-bits | text field not echoed (default: ON) | | |
| PDMFC | 64-bits | most recent user-issued PDM request | | |
| SSW1-SSW6 | 64-bits | pseudo sense swiiches | | |

** Operators **

Operators may be

- . arithmetic (+, -, *, /); Underflow and overflow are not detected; division by 0 produces zero
- . relational (.EQ., .NE., .LT., .GT., .LE., .GE.); returns -1 (TRUE) or O (FALSE)
- . logical (.OR., .AND., .XOR., .NOT.); returns a 64-bit value

Operations are performed left to right, taking into account parentheses, with the hierarchy of operators: (*, /), (+, -),relational, .NOT., .AND., .OR., .XOR..

> ** Strings **

A string is a group of ASCII characters (040-176 octal) to be taken literally. There are two types of strings:

. literal - delimited by apostrophes -- '...'

. parenthetical - delimited by parentheses -- (...)

Literal strings do not include the delimiters. An apostrophe within a literal string is represented by two apostrophes: '...''...'. A null string is indicated by two apostrophes: ''. A literal string is continued by placing an apostrophe and a continuation character at the end of the first line and an apostrophe at the start of the string on the next line:

> ...'This Is A '^ 'Long String.' becomes This Is A Long String.

Parenthetical strings do not include the delimiters. Spaces are removed; nested parentheses are not treated as separators; literal strings may appear in a parenthetical string. A parenthetical string is continued by placing a continuation character at the end of the first line and continuing the string on the next line:

> ... (This Is A ^ Long String.) becomes ThisIsALongString.

Page 3-3-1

***** Procedures *****

A procedure is a group of control statements separate from the job control statement dataset (\$CS). Calling a procedure provides a simplified way to process that group of control statements. A procedure may be called by a job repeatedly or by another procedure.

There are two kinds of procedures in COS:

- . simple a sequence of control statements
- . complex a prototype statement (giving the name of the procedure and any parameters), the control statements, and optional data.

*** Simple Procedures ***

A simple procedure has no name or parameters and resides in a nonlibrary dataset. It is invoked by a CALL without the CNS parameter. Control is returned to the caller by a RETURN statement, the end of the first file in the dataset, or an EXIT (when not skipping because of an error condition). A simple procedure has no parameter substitution.

Any COS JCL statement, except PROC and ENDPROC, may be used in a simple procedure. One use might be to access all the datasets needed in several jobs without having to specify them in the individual jobs.

*** Complex Procedures ***

Complex procedures are named and may have parameters described in a prototype statement. Complex procedures are executed by

- . "call by name", which may include parameters for substitution in the procedure. The procedure is in \$PROC or a local dataset named in a LIBRARY statement.
- . CALL,DN=procfyl,CNS, followed by a line containing the procedure name and parameters for substitution. The procedure is the first file in a separate dataset; PROC and ENDPROC are not used.

Complex procedures may appear, delimited by PROC and ENDPROC, in the job control statement dataset (\$CS). When PROC is encountered, the procedure is written to \$PROC. Subsequent calls to the procedure may then be made using the procedure name (and any substitute parameters).

```
Page 3-3-2
```

A complex procedure has the general form:

PROC. <-- not for CALL prototype statement control statements ... &DATA,dn1. data for first dataset ... &DATA,dnn. data for last dataset ENDPROC. <--- not for CALL

** Prototype Statement **

The prototype statement defines the name of the procedure and its formal parameters with their default value(s). It has the form:

name, p1, p2, ..., pn.

name - the name of the procedure (1-8 alphanumeric characters)

pi - a formal parameter specification

posi - positional

** Temporary Datasets **

One or more temporary datasets may be included in a complex procedure following the control statement. Each starts with

&DATA, dn.

where dn is the required dataset name.

** Parameter Substitution **

Formal parameters are used, preceded by an ampersand (&), within the body of the procedure. On execution, each is replaced by the value supplied or implied in the calling statement. ¶m is delimited by any character except A-Z, a-z, 0-9, @, \$, or %. If the next character is one of these, the underline (_) is used as the delimiter and is removed at execution time.

If too few positional parameters are specified by the caller, null strings are used for the remaining parameters; if too many, the job aborts. Keyword parameters may appear in any order, however, all positional parameters must precede all keywords.

** Apostrophes and Parentheses **

Apostrophes in the calling statement denote literals and are not removed during substitutions; the outer set of parentheses are removed. If you are not sure how a parameter is used in the procedure, enclose it in parentheses.

The following shows parenthetical substitution:

| caller | after substitution |
|-----------------|--------------------|
| | |
| value | value |
| (valuel=value2) | valuel=value2 |
| valuel'.'value2 | valuel'.'value2 |
| valuel(.)value2 | valuel.value2 |

Page 3-3-4

Rev0 90/09/30

*** DTRC "System" Procedures ***

The following DTRC-written procedures have been added to COS as of the date of this page. See Appendix C for more information. For the current list, type "HELP @CRAY CONTENTS" on the VAX. MSACCES Supply your Username and password to the Mass Storage System MSAUDIT Sorted audit of Mass Storage System files. MSCHANG Change the attributes of a Mass Storage System file. MSFETCH Fetch a file from the Mass Storage System to the CRAY X-MP MSPASSW Change your Mass Storage System access password. **MSPURGE** Purge a Mass Storage System file from the CRAY X-MP MSSTORE Store a CRAY X-MP file on the Mass Storage System

*** DTRC Procedure Library ***

One procedure library has been added to COS at DTRC:

PROCLIB, OWN=PUBLIC.

To use: ACCESS, PROCLIB, OWN=PUBLIC. LIBRARY, PROCLIB:*. procname,....

The following routines are the procedures in PROCLIB as of the date of this page. See Appendix C for more information. For the current list, type "HELP @CRAY CONTENTS" on the VAX.

JOBCOST Write job cost summary to \$LOG

NEWCHRG Change the account number of permanent files

Page 3-3-5

*** *** Examples ** ** Simple Procedures 1) The first file of dataset GETLIBS contains: ACCESS, DN=MSPROC, OWN=PUBLIC. <-- the MSS procedures ACCESS, DN=DTLIB, OWN=PUBLIC. <-- the DTLIB subroutine library ACCESS, DN=SUBS. <-- your subroutine library This is executed by: CALL, DN=getlibs. ** **Complex Procedures** ** 2) As in example 1, but your subroutine library is to be identified by the caller: GETLIBS, SUBS. <-- prototype statement ACCESS, DN=MSPROC, OWN=PUBLIC. <-- the MSS procedures ACCESS, DN=DTLIB, OWN=PUBLIC. <-- the DTLIB subroutine library ACCESS, DN=SUBS, PDN=&SUBS. <-- your subroutine library When called by: CALL, DN=getlibs, CNS. getlibs, othersubs. the third ACCESS expands to ACCESS, DN=SUBS, PDN=othersubs. Note that the name of the procedure is unimportant, since it is the only procedure in the file. "getlibs, othersubs." could be replaced by "*, othersubs". When called by: CALL, DN=getlibs, CNS. getlibs,(hislib,OWN=him). the third ACCESS expands to ACCESS, DN=SUBS, PDN=hislib, OWN=him. When called by: CALL, DN=getlibs, CNS. getlibs, 'hislib, OWN=him'. the third ACCESS expands to ACCESS, DN=SUBS, PDN='hislib, OWN=him'. While this is legal (it says the permanent filename is "hislib,OWN=him"), it is probably an error and, if so, will abort the procedure.

3) Create a procedure library from procedures in the job stream.

. . . ECHO, OFF. <-- return existing \$PROC</pre> RELEASE, DN=\$PROC. * <-- write first procedure to \$PROC PROC. prototype procedure body RETURN...procname EXIT. RETURN, ABORT...procname ENDPROC. <-- end of first procedure * PROC. <-- write next procedure to \$PROC prototype procedure body RETURN...procname EXIT. RETURN, ABORT...procname <-- end of procedure ENDPROC. * <-- more procedures • • • × ACCESS, DN=proclib, NA, UQ. <-- get original (existing) library</pre> SAVE, DN=\$PROC, PDN=proclib. <-- save new library DELETE, DN=proclib, NA. <-- delete original library RELEASE, DN=\$PROC. <-- return new library ACCESS, DN=proclib. <-- get new library with its own name <-- add it to the end of the library LIBRARY, DN=*:proclib. list -or-LIBRARY, DN=proclib:* <-- add it to the beginning of the library list ECHO, ON. . . . < use one of the procedures > . . .

```
90/09/30 Rev0
```

prototype procedure body

ENDPROC.

*

••• * Page 3-3-7

4) Create a procedure library from procedures in a separate file.

next procedure

. . . FETCH, DN=myprocs, TEXT='myprocs.pro'. <-- defaults to AC=ST</pre> CALL, DN=myprocs. SAVE, DN=\$PROC, PDN=proclib, PAM=R. <-- others may use it</pre> . . . where VMS file MYPROCS.PRO contains: × first procedure PROC. prototype procedure body ENDPROC. * next procedure PROC.

<-- more procedures

Page 3-4-1

***** Program Libraries *****

Source programs and data may be in separate datasets or may be stored and maintained in program libraries. UPDATE creates and maintains these libraries while AUDPL (see Appendix C) audits them.

*** UPDATE ***

UPDATE is a program for creating and modifying a program library (PL). In addition, UPDATE will extract individual modules for input to a compiler or other program.

By default, 72 columns of information are retained. Fifteen additional characters are retained for each line: an 8-character identifier, a period (.), and a 6-digit sequence number, i.e., id.seq.

UPDATE supports two kinds of text modules or decks:

- a regular deck (beginning with a DECK directive)
- a common deck (beginning with a COMDECK directive) which may be included in decks with a CALL directive

Each type includes all lines following the deck directive until the next deck or modification directive.

History information is retained allowing the deletion, modification, or restoration or previous modifications.

See Appendix C for a description of the UPDATE control statement parameters.

*** UPDATE Directives ***

An UPDATE directive, which must be in upper case, has the following format:

m directive name [parameters]

where m is the master character (default: asterisk (*)). There are five categories of directives.

** DECK and COMDECK **

*DECK deck (*DK) First line of a new deck. <deck> is up to 8 characters, any ASCII character from 41 to 176 octal, except comma, period, blank, colon, equals.

*COMDECK cmdk (*CDK) First line of a new common deck. Page 3-4-2

Rev0 90/09/30

** Compile Directives ** (*CA) *CALL cmdk Include the contents of a common deck. *CWEOF Write an EOF on the compile dataset if anything was written since the last EOF. *NOSEQ Do not write sequence numbers. *SEQ Write sequence numbers. *WEOF Write an EOF on the compile dataset. *WIDTH dw Change the data width (default: 72). *IF, *ELSEIF, *ELSE, and *ENDIF are also available. ** ** Modification Directives *BEFORE id.seq (*B) Insert before a line. *COPY p,id1.seq1,id2.seq2 (*CY) Copy a range of lines from deck or comdeck . (*D) *DELETE idl.seq1 <-- one line *DELETE idl.seq1, id2.seq2 <-- a range of lines *DELETE idl.seq1,.seq2 <-- same (short form) Delete a line or a range of lines. *IDENT ident (*ID) *IDENT ident,K=k1:k2:...,U=u1:u2:... Identify a set of modifications. You can require that other modification sets be known (K=) or unknown (U=). *INSERT id.seq (*I) Insert after a line. (*R) *RESTORE idl.seq1 <-- one line *RESTORE idl.seq1, id2.seq2 <-- a range of lines <--- same (short form) *RESTORE idl.seq1,.seq2 Restore a line or a range of lines.

Page 3-4-3

** ** Run Options */comment A comment line. (*C) *COMPILE p1, p2, ..., pj. pk, ..., pn Write one or more decks, including a range (pj.pk), to the compile and/or source datasets. Use UPDATE,K to force the output order. *COPY p,idl.seq1,id2.seq2,dn (*CY) *COPY p,idl.seq1,id2.seq2,dn,SEQ Copy a range of lines from deck or comdeck to dataset <dn>. SEQ will include sequence numbers. *LIST Resume listing input lines. UPDATE, L=O overrides *LIST. *MASTER m Define a new master character for subsequent directives. (default: *) *NOLIST Stop listing input lines. *NOLIST overrides UPDATE, IN. *READ dn (*RD) Read input from another dataset. *REWIND dn Rewind a dataset. *SKIPF dn *SKIPF dn,n Skip file(s) in a local dataset. *DECLARE and *DEFINE are also available.

1

** Input Edit Directives ** (*ED) *EDIT p1,p2,...,pn Remove deleted and yanked lines from specific decks. These lines cannot be retrieved. This is useful for cleaning up a PL. *MOVEDK dkl:dk2 *MOVEDK dk1:. Position deck of common deck <dkl> immediately after deck or common deck <dk2> or at the beginning of the PL <.>. *PURGE id1, id2,..., idj.idk,..., idn.. Remove the effect of a modification set (idi), a range of datasets (idj.idk), or a set and all following (idn..). *PURGEDK dk Permanently remove a deck or common deck. *UNYANK id1, id2,..., idj.idk,..., idn.. Reactivate a deck, comdeck, or modification set previously yanked. *YANK id1, id2,..., idj. idk, ..., idn.. Temporarily delete a deck, comdeck, or modification set previously yanked. *SKIP and *ENDSKIP are also available.

end

*** *** Examples 1) Create a PL: JOB, JN=makep11. ACCOUNT,.... <-- omit if using VAXcluster CSUBMIT</pre> UPDATE, P=0,C=0. <-- no \$PL or \$CPL SAVE, DN=\$NPL, PDN=myp1. /EOF *DK DECK1 s for deck DECK1> *DK DECK2 s for deck DECK2> *DK DECK3 s for deck DECK3> 2) Extract, compile and execute deck DECK2 from PL MYPL: JOB, JN=getp12. <-- omit if using VAXcluster CSUBMIT ACCOUNT.... ACCESS, DN=\$PL, PDN=myp1. UPDATE. CFT77, I=\$CPL. SEGLDR, CMD='MAP, PART', GO. /EOF *C DECK2 3) Create a PL using a common deck, compile and execute: JOB, JN=makep13. ACCOUNT,.... <-- omit if using VAXcluster CSUBMIT</pre> UPDATE, P=0. <-- no \$PL (required to create)</pre> SAVE, DN=\$NPL, PDN=myp1. CFT, I=\$CPL. SEGLDR, CMD= 'MAP, PART', GO. /EOF *CDK COM3 common / mycom / a, b, c real a, b, c *DK PROG3 program prog3 *CALL COM3 call sub print *, 'a,b,c=', a, b, c end *DECK SUB subroutine sub *CA COM3 a = 1.b = 2.c = 3.return

4) Update old source library to new, compile all decks and execute: JOB, JN=job4. ACCOUNT.... <-- omit if using VAXcluster CSUBMIT ACCESS, DN=\$PL, PDN=mylib. UPDATE, F, N. SAVE, DN=\$NPL, PDN=mylib. CFT, I=\$CPL. SEGLDR.GO. /EOF *IDENT DS0620 <-- correction must be unique (initials,date)</pre> *INSERT ALONE.57 <-- correct deck ALONE by insert after line 57 <FORTRAN statements> *DELETE FOUR.12,13 <-- correct deck FOUR replacing lines 12-13</pre> <new lines to replace deletions - optional> /EOF <data lines, if any> /EOF 5) Select routines from source subroutine library on MSS and compile with own program: JOB, JN=job5. ACCOUNT,.... <-- omit if using VAXcluster CSUBMIT ACCESS, DN=MSPROC, OWN=PUBLIC. LIBRARY, DN=MSPROC:*. MSACCES, UN=un, MPW=mymsspw. CFT77.ON=MSX. <-- compile own programs with listing</pre> MSFETCH, DN=LIBR, MDN=DTLIBPC, UN=NSYS. UPDATE, P=LIBR, Q, L=0. CFT77, I=\$CPL. e SEGLDR,GO. <-- load and execute</pre> /EOF <own FORTRAN decks> /EOF *C rtn1, rtn6.rtn8 <-- select decks RTN1, 6, 7, 8 from library /EOF <data records, if any> /EOF

***** Object Libraries *****

BUILD is a utility for creating and maintaining libraries of absolute and relocatable object modules. These libraries can then be used by the loader to locate the program to execute or the subprograms to be loaded with your program.

The BUILD control statement is described in Appendix C.

*** BUILD Directives ***

A directive consists of a keyword and, perhaps, a comma-separated list of dataset or module names. The keyword is separated from its list by a blank. Directives cannot be continue. Multiple directives, separated by a semicolon or period, may appear in one line.

- FROM dn1,dn2,...,dnn Single dataset for COPY, OMIT, LIST, or a list of datasets (copy dn1 thru dnn-1 to \$NBL, dnn is the same as if specified alone. If no COPY, OMIT, dnn is also copied. dni can be a library or sequential dataset (like \$BLD).
- OMIT fn1,fn2,...,fnn
 List of modules to be excluded. Each fni may be a single name
 or a group name, i.e., with wildcards (any 0 or more
 characters) or * (any single character).
- COPY fn1,fn2,...,fnn List of modules to be copied. Each fni may be a single or grc.p name, or a rename (ELM=OAK copies ELM and renames it OAK), or an inclusive range such as (first,last) or (first,) or (,last) or (,).

LIST

Immediately list characteristics of modules in input dataset.

*** DTRC Object Libraries ***

Two object libraries have been added to COS at DTRC:

DTLIB,OWN=PUBLIC - Subprograms written or maintained by the Computer Center To use: ACCESS,DN=DTLIB,OWN=PUBLIC. SEGLDR directive: LIB=DTLIB UTILITY,OWN=PUBLIC - Programs written or maintained by the Computer Center To use: ACCESS,DN=UTILITY,OWN=PUBLIC.

LIBRARY,UTILITY:*. program_name,.... Page 3-5-2

*** *** Examples 1) Create a library of subprograms: JOB, JN=JOB1. ACCOUNT,.... CFT. BUILD, I=0, OBL=0. SAVE, DN=\$NBL, PDN=MYSUBLIB. /EOF <Fortran source subprograms> /EOF 2) Create a library of all subprograms from an UPDATE library: JOB, JN = JOB2. ACCOUNT,.... ACCESS, DN=\$PL, PDN=MYPL. UPDATE, F. CFT, I=\$CPL. BUILD, I=0, OBL=0. SAVE, DN=\$NBL, PDN=MYSUBLIB. /EOF 3) Add a subprogram to an existing library and have the output list in alphabetical order. JOB, JN=JOB3. ACCOUNT,.... ACCESS, DN=\$OBL, PDN=MYSUBLIB. CFT. BUILD, I=0, SORT. SAVE, DN=\$NBL, PDN=MYSUBLIB. /EOF <Fortran source subprograms> /EOF 4) Delete subprogram BADSUB from an existing library and list the contents of both old and new libraries. JOB, JN=JOB4. ACCOUNT,.... ACCESS, DN=\$OBL, PDN=MYSUBLIB. BUILD, B=0. SAVE, DN=\$NBL, PDN=MYSUBLIB. /EOF

OMIT BADSUB

LIST



5) List the contents of an existing library.

JOB, JN= JOB5. ACCOUNT,.... ACCESS,DN=SUBLIB,PDN=MYSUBLIB. BUILD.OBL=0,NBL=0,B=0. /EOF FROM SUBLIB; LIST.

***** Loader *****

The loader is responsible for loading all programs, resolving any external references, and optionally initiating execution. Loading can produce either a single absolute module, or a (segmented) absolute program in which different parts of a program reside in memory only when needed.

*** SEGLDR ***

The primary loader is SEGLDR. It is controlled by directives which may appear as the next file in the input stream, in a separate file, or in the loader control statement.

** Control Statement **

See Appendix C for a fuller description of the SEGLDR control statement.

SEGLDR,I=dirfile,L=listfile,DN=binfil,LIB=library,ABS=absolute, CMD='directives',GO.

"SEGLDR." implies SEGLDR, I=\$IN, L=\$OUT, DN=\$BLD, ABS=\$ABD.

** Message Levels **

SEGLDR issues messages at the following levels:

- ERROR immediately terminates SEGLDR with no executable output
- WARNING no executable output but processing continues
- CAUTION executable output but a possible error was found
- NOTE SEGLDR has been misused or used ineffectively; executable output is still valid
- COMMENT does not affect execution

Page 3-6-2

Rev0 90/09/30

** Directives **

Most SEGLDR directives have the format: keyword=value. Comments (anything following an asterisk (*)) may appear anywhere in the directives, including at the end of a directive line. Multiple comments on a line are separated by a semicolon (;). Elements of a list are comma-separated. Directives may be continued by splitting the line after a parameter (the comma is the last non-blank character in the line).

Naming files: ABS, BIN, DEFLIB, LIB, NODEFLIB.

Listing control: COMMENT, ECHO, MAP, TITLE, TRIAL.

Naming modules and common blocks: COMMONS, DUPORDER, DYNAMIC, FORCE, MODULES.

Error message control: DUPENTRY, DUPLOAD, MLEVEL, REDEF, USX.

Entry point and execution control: EQUIV, SET, XFER.

Global heap memory management: HEAP, LOWHEAP, STACK.

Memory allocation and presetting: ALIGN, ORG, PRESET.

Symbolic debugging: DRD, SYMBOLS.

Miscellaneous COS-dependent directives: ABORT, ABSNAME, BCINC, GRANT, NOECHO, NORED, PADINC, SECURE.

Miscellaneous global directives: CASE, CPUCHECK.

Additional information, including directives not discussed here, may be found in SR-0066 Segment Loader Reference Manual.

* comment A comment.

Examples: TITLE=GLOBAL DIRECTIVES *_____ * Global directives *_____ BIN=ABC TITLE=TREE DIRECTIVES *-----* Tree directives *-----TREE ROOT(A,B)ENDTREE TITLE=SEGMENTS *_____ SEGMENT=ROOT * ROOT directives

90/09/30 Rev0 Page 3-6-3 ABORT=ON | OFF Control SEGLDR error termination. Values: ON - abori if errors OFF - terminate normally even if errors Default: ABORT=ON ABS=dn The dataset to contain the absolute module. Default: \$ABD Examples: ABS=myprog ALIGN=IGNORE | MODULES | NORMAL Control the starting locations of modules and commmon blocks. Values: IGNORE - start each module's local or common block at the word following the previous one (ignore align bit) MODULES - start each module's local block and common block (if the align bit is set) at an instruction buffer boundary (32 words) NORMAL - start each module's local or common block with the align bit set at an instruction buffer boundary (32 words) Default: ALIGN=NORMAL BIN=dn1, dn2,... Datasets containing the relocatable modules to be loaded. Default: BIN=\$BLD Examples: BIN=myfile, yourfile, theirfyl BIN=oldfile CASE=UPPER | MIXED Control character conversion in directives. Values: UPPER - convert to upper case MIXED = do not convert Default: CASE=UPPER

COMMONS=blkl:siz1,blk2:siz2,... Specify the order to load common blocks. Values: blki - name of a common block sizi - n - decimal size 0 - first occurrence of this block sets the size (default: 0) Examples: COMMONS=myblk:100000,data1 ^-- MYBLK is 100,000 words (no matter how it is defined); DATA1 has its first encountered length DEFLIB=deflib1, deflib2,..., deflibn Add libraries to SEGLDR's list of default libraries. If a specified library is already in the Remarks: default library list, it is moved to the end of the list. Libraries in DEFLIB are searched after the default system libraries; those in LIB are searched before. Examples: DEFLIB=mylib ^-- add MYLIB to the list NODEFLIB: DEFLIB=mylib ^-- the default library list consists of just one library DRD Load for debugging. Symbol tables are written to \$DEBUG (or SYMBOLS=dn). Default: Normal load DUPENTRY=ERROR | WARNING | CAUTION | NOTE | COMMENT | IGNORE Specify the message level for a duplicate entry point. Default: DUPENTRY=CAUTION DYNAMIC=comblk DYNAMIC=// Name a common block to be located after the largest segment or the heap (if required). You control its size. It is always available to the program and cannot be preloaded with data. Values: a COMMON block name or // (blank common)

```
Default: no dynamic common blocks
             Examples: DYNAMIC=ARRAYS
                                   ^-- common block /ARRAYS/ is dynamic
FCHO=ON | OFF
             Resume or suppress listing of input directives.
             Default:
                        ECHO=OFF
EQUIV=epname(syn1, syn2,...)
             Substitute a call to one entry point for a call to
             another.
             Values:
                        epname - the entry point to be used in the
                                 substitution
                               - an entry point to be replaced by
                        syni
                                 epname
             Examples:
                          . . .
                        CALL A
                          . . .
                        CALL B
                          . . .
                        EQUIV=C(A, B)
                                    ^-- replaces the calls to A and B
                                       by calls to C
FORCE=ON OFF
             Control the forced loading of modules whose entry points
             are never called.
             Default: FORCE=OFF
HEAP=init+inc
             (Tasking) Allocate memory for dynamic management.
                        init - initial decimal number of words
             Values:
                        inc - size, in decimal words, of increment
                               when the heap overflows
                                0 - the heap size is fixed
                               (ignored if DYNAMIC directive is
                               specified)
```

Page 3-6-5

90/09/30

Rev0

Examples: HEAP=10000+2000

Page 3-6-6

Rev0 90/09/30

LIB=1ib1,1ib2,... Libraries to be searched for routines not included in BIN= files or default libraries. Remarks: Libraries in DEFLIB are searched after the default system libraries; those in LIB are searched before. Examples: ACCESS, DTLIB, OWN=NSYS. <-- DTRC subroutine library <-- your subroutine ACCESS, sublib. library SEGLDR,CMD='LIB=sublib,DTLIB',... MAP=NONE | STAT | ALPHA | ADDRESS | PART | EPXRF | CBXRF | FULL Control the map listing. Values: NONE - no map STAT - list load statistics: date/time, longest branch length, last segment, transfer entry point, stack and heap information ALPHA - STAT + block map for each segment (modules in alphabetical order) ADDRESS - ALPHA but modules in address order PART - ALPHA + ADDRESS EPXRF - STAT + entry point cross reference CBXRF - STAT + common block cross reference FULL - PART + EPXRF + CBXRF Default: MAP=NONE

Examples: MAP=STAT MAP=EPXRF,CBXRF

MLEVEL=ERROR | WARNING | CAUTION | NOTE | COMMENT Print messages down to and including the level specified (has no effect if L=0).

> Default: MLEVEL=CAUTION Examples: MLEVEL=NOTE ^-- print error, warning, caution, and note messages

Page 3-6-7

MODULES=mod1:ds1,mod2:ds2,... The modules to be included and, optionally, the dataset containing a specific module. Values: modi - name of module to be loaded dsi - optional dataset containing the module Examples: MODULES=sub1:sublib, sub2, sub3:yourlib MODULES=sub4, sub5 ^-- get SUB1 from SUBLIB; SUB3 from YOURLIB; SUB2, SUB4, SUB5 from the first dataset containing them NODEFLIB Do not search the default libraries. Search only BIN and LIB datasets. NOTE: Segmented loads must specify the file containing routine \$SEGRES. Examples: NODEFLIB; LIB=sublib, DTLIB, \$SCILIB ORDER=MODULES, COMMONS | COMMONS, MODULES | XMP.EMA Load modules before or after commons. Values: XMP.EMA - most efficient allocation on X-MP having more than 4 million words ORDER=MODULES, COMMONS (<=4 million words) Defaults: ORDER=XMP.EMA (> 4 million words) PRESET=ONES | ZEROS | INDEF | -INDEF | value Preset uninitialized data areas. Values: ONES - set to -1 ZEROS - set to O INDEF - set to octal 0605054000000000000000 -INDEF - set to octal 1605054000000000000000 value - 16-bit value placed in each parcel (0 < value < 177777 octal)Default: PRESET=ZEROS

Rev0 90/09/30

SET=epname:value Set the value of an entry point. Values: epname - the entry point name value - the value it is to have (overrides the value found in the relocatables) Examples: SET=\$RBUFLN:256 SET=SWBUFLN:256 ^-- change the read or write buffer length to 256 characters (to change the read/write buffer area, use COMMONS=\$RFDCOM:265 or COMMONS=\$WFDCOM:265 (must be 9 more than the buffer length) STACK=init+inc (Tasking) Allocate part of heap memory to a stack for reentrant programs. Values: init - initial size, in decimal words, of a stack <128 - default used inc - size, in decimal words, of increments when the stack overflows 0 - stack overflow is not allowed Default: Static variables unless Tasking or ALLOC=STACK. Examples: STACK=5000+2000 SYMBOLS=ON | OFF | dn Specify program symbol table handling. Values: ON - write symbol table to \$DEBUG OFF - ignore symbol table dn - write symbol table to dn (dn may not be ON or OFF) SYMBOLS=ON Default:

Page 3-6-9

TITLE=title TITLE Define the second line of each page header. A page eject is forced. Value: title - a string of 0-74 characters (ends with end-of-line or semicolon) omitted - clear the second header line Examples: TITLE=This is a user title, really! TRIAL Do not generate an executable module. Lets you get the load map, determine optimal memory usage for data, or get the total memory required. Examples: TRIAL USX=WARNING | CAUTION | IGNORE Specify how to treat unsatisfied externals. Values: WARNING - issue a warning message; do NOT write executable output CAUTION - issue a caution message; write executable output IGNORE - issue no message; write executable output USX=CAUTION Default: Specify the entry point at which the program is to start XER=entry execution. Values: entry - the starting entry point name Use this to specify the name of the main Remarks: program to be executed if it is in a library. Default: The first primary entry point encountered -if none, "main" is used.

Page 3-6-10

*** Segmentation ***

To make a large program fit into memory, it may be structured in segments, so that only a portion of the program resides in memory. By using the tree structure directives of SEGLDR, different arrangements of a program can be tried, without changing the program, until the best is achieved. ** ** Segmentation Directives Tree definition: TREE, tree definition, ENDTREE. Segment description: SEGMENT, BIN, COMMENT, COMMONS, DUP, ECHO, MODULES, SAVE, TITLE, ENDSEG. Global: COPY, SAVE, SLT. BIN=dn1, dn2,... Datasets containing the relocatable modules to be loaded. Only the first file of each dataset is processed. Default: BIN=\$BLD Examples: SEGMENT=birch BIN=myfile, yourfile, . theirfyl BIN=oldfile ENDSEG ^-- all modules in datasets MYFILE, YOURFILE, THEIRFYL, and OLDFILE are loaded into segment BIRCH COMMONS=blk1:siz1,blk2:siz2,... Specify the order to load common blocks. Values: blki - name of a common block sizi - n - decimal size 0 - first occurrence of this block sets the size (default: 0) Examples: COMMONS=myblk:100000,data1 ^-- MYBLK is allocated 100,000 words (no matter how it is defined); DATA1 has its first encountered length

Page 3-6-11

COPY Force the program to execute from a scratch file. This may speed program execution, especially of programs with segments which are loaded many times, because a faster form of I/O is used. SAVE=ON also forces the use of a scratch file.

Default: a scratch file is not used

DUP=mod(seg1,seg2,...)

Specify that a module is to be loaded into several segments. DUP must appear before the definitions of the segments into which the module is to be placed.

An alternate way is to list the module in the MODULES or COMMONS directive of each segment.

| Examples: | DUP≃sub3(seg1,seg2) | root | | |
|-----------|---------------------|------|------|--|
| | SEGMENT=seg1 | | I | |
| | MODULES=sub1 | 1 | | |
| | COMMONS=com1 | seg1 | seg2 | |
| | ENDSEG | | | |
| | SEGMENT=seg2 | com1 | coml | |
| | MODULES=sub2 | sub1 | sub2 | |
| | COMMONS=com1 | SUB3 | SUB3 | |
| | ENDSEG | | | |

ENDSEG End the definition of a segment of a tree structure.

Examples: see SEGMENT

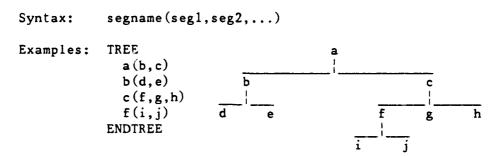
ENDTREE End the definition of a tree structure.

Examples: see TREE

MODULES=mod1,mod2,... (segment) List the modules to be put into the segment. Values: modi - module name and optional dataset from which it is to be loaded (mod:ds) Examples: MODULES=m:binm,n,o ^-- load module M from dataset BINM and modules N and O from the first dataset which contains them SAVE=ON OFF (Global) Specify whether all segments are to be saved (written to disk) before being overlaid. SAVE in a segment overrides the global SAVE. Values: OFF - do not save each segment ON - save each segment Default: SAVE=OFF Examples: SAVE=ON one TREE one(two,three) subl ENDTREE SEGMENT=one Ţ MODULES=sub1 two three SEGMENT=two MODULES=sub2, sub3 sub2 sub4 SEGMENT=three sub3 SAVE=OFF MODULES=sub4 ENDSEG SEGMENT=segname Begin the description of the contents of one segment of a tree. Examples: SEGMENT=oak MODULES=k,1,m COMMONS=//,oakcom ENDSEG TREE End the global directives and start the definition of a tree structure. Examples: TREE tree structure ENDTREE

tree segment structure

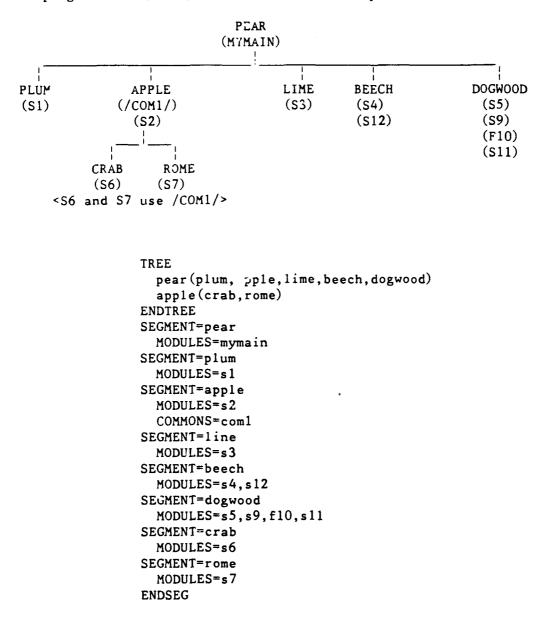
Define the tree structure, that is, the segments in each branch of the tree. The order of these definitions is unimportant.



Page 3-6-14

** Sample Tree Diagram **

A block data subprogram defines common /COM1/ which is to be loaded with program S2. /COM1/ is also referred to by S6 and S7.



Page 3-6-15

** Segmentation Cautions **

1. To develop a segmented job, several runs may be required, so relocatable object code should be SAVEd. Common blocks and some system routines may need to be included in lower segments to operate properly.

2. The load map should be checked carefully for any duplicate common block entries. The same common block may appear in more than one segment, each being considered a different common block. References are to the common block in the segment, if none, then to the one on the same branch. If a given common block is to appear only once in a program (the normal case), then it should be placed in the segment nearest to the root segment which can be referenced by all segments which use it.

```
***
                                                             ***
         Compile, Load and Save an Absolute Program
                       **
                                            **
                            Simple Load
JOB, JN=jobname,....
ACCOUNT,....
CFT.
SEGLDR,CMD='ABS=myprog'.
SAVE, DN=myprog, PAM=R.
                                 <-- read only
/EOF
      PROGRAM MYPROG (...
          . . .
/EOF
                     \star \star
                                             \star\star
                          Segmented LOAD
JOB, JN=jobname,....
ACCOUNT,....
CFT.
SAVE, DN=$BLD, PDN=myprogob.
                                    <-- save relocatable modules for</pre>
                                          possible re-segmentation
SEGLDR.
SAVE, DN=myprog, PAM=R.
                                      <-- read only
/EOF
   (CFT source program)
/EOF
ABS=myprog
TREE
  . . .
                 and the second
ENDTREE
SEGMENT=...
  ...
ENDSEG
SEGMENT=...
  . . .
ENDSEG
  . . .
/EOF
```

.

Page 3-6-16

90/09/30

Rev0

Page 4-1-1

***** The Mass Storage System *****

The Mass Storage System (MSS) is a large capacity on-line mass storage device. It is a cost effective extension to the Cray, CDC and VAXcluster disk systems and conventional magnetic tape storage. Specifically, the MSS, which is part of the CDC CYBER 860 (NOS), offers:

- . More than 20 times the on-line storage of the VAXcluster system; more than 40 times the on-line storage of the CRAY X-MP.
- . On-line access to files which previously had to be stored on magnetic tape because of size restrictions and/or infrequent use.
- . Reduced storage charges for these on-line files.

*** MSS Security ***

To provide adequate security for MSS users, you must submit your MSS (CYBER 860) password in any non-CDC (NOS) job or interactive session which will manipulate MSS files. To protect your MSS files, you must change this password at least every 90 days using the PASSWOR command on the CDC CYBER 860 (NOS), the HFT PASSWORD command on the VAXcluster, the MSPASSW command on the CRAY X-MP or the CDC 860 (NOS/VE).

*** MSS File Purge ***

MSS files may be purged by the Computer Center if the job order number is invalid or has been cancelled.

To recover purged files, call User Services, Code 3511, (301) 227-1907. A fee will be charged for this service. After the files have been restored, you must change to your valid job order number:

| on 860 (NOS): | CHANGE, mfn/CP or BEGIN, NEWCHRG |
|------------------|----------------------------------|
| on 860 (NOS/VE): | MSCHANG mfn CP=1 |
| on Cray (COS): | see page 4-1-4 (MSCHANG) |
| on VAXcluster: | see page 4-1-8 (MSSNEWCHRG) |

Page 4-1-2

Rev0 90/09/30

*** MSS Backup for Critical Files ***

In addition to normal file backup, critical direct files may be backed up and stored off-station. These files are available in the event of a catastrophe (such as fire) at the Carderock Computer Center.

For a file to be designated as "critical", it must have the attribute Backup Requirement (BR) set to critical (CR). This is done by specifying "BR=CR" if the file is critical, or "BR=Y" if it is not, when the file is made permanent. The default is BR=Y meaning on-station backup. For example (on NOS):

| DEFINE, 1fn=mfn/BR=CR. | < st | ore a | a cri | tical file |
|------------------------|------|-------|-------|--------------|
| CHANGE,mfn/BR=CR. | < ma | ke a | file | critical |
| CHANGE, mfn/BR=Y. | < ma | ke a | file | non-critical |

Files designated for this off-station backup service will be charged a higher rate.

Page 4-1-3

the site site *** Using the MSS from the Cray A description of the syntax of these commands may be found in Appendix C. ACQUIRE Transfer a file from the MSS as a local dataset and make it permanent on the Cray. Examples: ACQUIRE, DN=SOURCE, SDN=MYFILE, PDN=MYFILE, MF=N1, ^ TEXT='USER, user, pw. ATTACH, MYFILE. CTASK.'. ^-- transfer your direct MSS file MYFILE as local dataset SOURCE and make it a permanent dataset named MYFILE ACQUIRE, DN=DATA46, PDN=DATA46, MF=N1, ^ TEXT='USER, user, pw. '^ 'ATTACH, DATA46/UN=ABCD, PW=filepw.CTASK.'. ^-- transfer user ABCD's MSS file DATA46 (assuming you have permission to read the file) as local dataset DATA and make it a permanent dataset named DATA46 DISPOSE Transfer a Cray local dataset to the MSS. Examples: DISPOSE, DN=FT13, MF=N1, SDN=MYOUT13, DC=ST, ^ TEXT='USER,user,pw.'^ 'PURGE, MYOUT13/NA.'^ 'DEFINE, MYOUT13.'^ 'CTASK.'. ^-- local dataset FT13 is transferred to the MSS where it will be known as MYOUT13 FETCH Transfer a file from the MSS as a local dataset. It is released at the end of the job. Examples: FETCH, DN=SOURCE, SDN=MYFILE, MF=N1, ^ TEXT='USER, user, pw. ATTACH, MYFILE. CTASK.'. ^-- transfer your MSS file MYFILE as local dataset SOURCE FETCH, DN=ABDATA, MF=N1, TEXT='USER, user, pw. '^ 'ATTACH, ABDATA/UN=ABCD, PW=filepw.CTASK.'. ^-- transfer user ABCD's MSS file ABDATA as local dataset ABDATA FETCH, DN=SOURCE, SDN=MYFILE, MF=N1, ^ TEXT='USER, user, pw.GET, MYINDF.CTASK.'. ^-- transfer your CYBER 860 indirect file MYFILE as local dataset SOURCE

The following procedures provide access to the Mass Storage System. They have been made part of COS at DTRC.

MSACCES Supply your Username and password to the MSS. MSACCES is required before you can use the MSx commands.

Example: MSACCES, US=myid, MPW=mymsspw.

MSAUDIT Sorted audit of Mass Storage files.

Examples: MSAUDIT. <-- short audit of your MSS files

MSAUDIT,LO=X,SHOWPW=1. ^-- full audit showing each file's password, if any

MSAUDIT,L=audout,LO=X,UN=otheruser. ^-- full audit of another user's files you are allowed to see with output in local dataset AUDOUT

MSCHANG Change Mass Storage System file attributes.

Examples: MSCHANG, MDN=myfile, CT=PUBLIC. ^-- make your MSS file MYFILE public

MSCHANG, MDN=oldname, NMDN=newname.

MSCHANG, MDN=myfile, CP=1. ^-- change the account number of file MYFILE to your current MSS charge number

MSFETCH Fetch a direct file from the MSS.

Examples: MSFETCH, DN=infyl, MDN=mydata. ^-- your file in transparent mode

> MSFETCH, DN=prog, MDN=othrpgm, UN=ABCD, PW=pgmpw. ^-- another user's file

MSPASSW Change Mass Storage System access password.

Example: MSPASSW, OLD=oldpw, NEW=newpw.

Page 4-1-5

MSPURGE Purge an MSS file.

Example: MSPURGE, DN=myfyle.

MSSTORE Store a file on the MSS as a direct file.

Examples: MSSTORE, DN=out1, MDN=outfyl1, NA=1. ^-- overwrite if file already exists

> MSSTORE, DN=fy12, MDN=file2, DF=CB. ^-- file is stored in CDC Display Code

Rev0 90/09/30

*** Using the MSS from the VAXcluster ***

A description of the syntax of these commands may be found by typing "HELP <command>" on the VAXcluster.

HFT HYPERchannel (direct) File Transfer.

Examples: HFT ACCESS /U=ABCD /A=1222233344 /P=MSS_password ^-- gain access to the MSS

HFT CHANGE "MYFILE/AC=newac,CT=PU" ^-- change account number of MSS file MYFILE and make it public

HFT DEFAULT

^-- display your current ACCESS
values

HFT DELETE MYFILE ^-- delete MSS file MYFILE

HFT DIRECTORY ^-- audit your MSS file names

HFT DIRECTORY "LO=F" ^-- full audit of your MSS files

HFT FETCH MYPROG MYPROG.FOR ^-- fetch your MSS file MYPROG and make it permanent file MYPROG.FOR

HFT PASSWORD old password new password new password repeated ^-- change your MSS password

HFT PERMIT "MYFILE/UN=xxxx,M=R" ^-- give read access to user xxxx

HFT STORE MYPROG.FOR "MYPROG/CT=S" HFT STORE MYPROG.FOR "MYPROG/CT=S" /DELETE ^-- store your file MYPROG.FOR on the MSS as MYPROG (/DELETE will delete your VAXcluster permanent file)

Page 4-1-7

MSSAUDIT Audit your MSS files in a variety of formats. Examples: MSSAUDIT S <-- get a sorted short audit of your MSS files at the terminal MSSAUDIT F MSSAUDIT.LIS ^-- put a sorted full audit of your MSS files into file MSSAUDIT.LIS MSSAUDIT 0 UN=xxxx (0 = zero)^-- display a sorted list of the MSS files owned by user xxxx (assuming you have permission to see them) MSSBACKUP Store several files in a single file on the MSS, retaining each file's characteristics. Fetch individual files from the MSS file previously stored by MSSBACKUP. Examples: MSSBACKUP STORE *.* VMS0322 ^-- store all your files in a BACKUP file on the MSS (0322 is the date)MSSBACKUP LIST VMS0322 KEEP ^-- list the contents of the above BACKUP file on MSS at your terminal, keeping the .MSSBCK file for later FETCHes today MSSBACKUP FETCH VMS0322 RD* ^-- fetch the files beginning with RD (do not replace any existing versions) MSSB DELETE VMS0322 ^-- Delete the BACKUP file from MSS MSSDELETE Delete several MSS files. Examples: MSSDELETE MYFILE ^-- same as HFT DELETE "MYFILE"

> MSSD F1,F2,F3,F4,F5 ^-- delete 5 MSS files

Page 4-1-8

_ _ _ _

ł

Rev0 90/09/30

MSSNEWCHRG Change the account number on your MSS files. Examples: MSSNEWCHRG 1222233344 1234567890 ^-- change job order number for all files currently stored with account number 1-2222-333-44 to 1-2345-678-90

Page 4-1-9

*** Using the MSS from the CDC CYBER 860 ***

The MSS is just a peripheral on the CDC CYBER 860 and is under the control of NOS. All NOS files on the CYBER 860, whether they reside on disk or the MSS, are accessed by the standard NOS permanent file commands.

NOS/VE does not have direct access to the MSS. The following commands provide access to the Mass Storage System. A description of their syntax may be found in Appendix H.

CHANGE_LINK_ATTRIBUTES (chala) Change individual link attributes for communication between dual-state partners.

> Examples: chala pw=mymsspw ^-- required to access the MSS (see also MSACCES)

DISPLAY_LINK_ATTRIBUTES (disla) Display individual link attribute values.

> Examples: disla CHARGE : FAMILY : nlfam PROJECT : USER : AMDS

GET_FILE (getf) Copy a NOS file to NOS/VE.

> Examples: getf my_file myfile ^-- Get MSS file MYFILE and store as NOS/VE file MY_FILE

REPLACE_FILE (repf) Copy a NOS/VE file to NOS direct file, replacing any existing file.

> Examples: repf my_file myfile ^-- copies NOS/VE file MY_FILE to MSS as MYFILE, replacing any existing MSS/NOS file with the same name

```
Page 4-1-10
```

MSACCES Access the Mass Storage System. Examples: MSACCES mymsspw MSAUDIT Obtain a sorted audit of your MSS files. Examples: MSAUDIT <-- short audit of your MSS files</pre> MSAUDIT, LO=F, STW=Y ^-- full audit showing each file's password, if any MSAUDIT audout s UN=other ^-- short audit of another user's files you are allowed to see with output in file AUDOUT in MSAUDIT F='V*****' ^-- all files starting with "V" your current catalog MSCATLIST The NOS CATLIST command. MSCHANG Change the attributes of a Mass Storage System file. Examples: MSCHANG myfile CT=PU ^-- make MYFILE a public file MSCHANG, myfile, CP=1 ^-- change the account number MSCHANG myfile NFN=newfile M=E ^-- change the name of the file and make it execute-only MSFETCH Fetch a file from the Mass Storage System. Examples: MSFETCH, mssfyll, inl ^-- MSSFYL1 is retrieved and stored as file IN1 in your current catalog

MSFETCH mssfyl2 in2 D64 ^-- MSSFYL2 is retrieved, converted from 64-character NOS Display Code and stored as file IN2 in your current catalog

```
MSPASSW Change your Mass Storage System access password.
         Examples: MSPASSW, OLD=mymsspw, NEW=numsspw
                       -or-
                    MSPASSW mymsspw numsspw
                       -01 -
                    MSPASSW N=numsspw O=mymsspw
                                   ^{--} the above are the same
MSPURGE Purge a file on the Mass Storatge System.
         Examples: MSPURGE F=mymss1
                                   ^-- purge MSS file MYMSS1
                    MSPURGE (f1, f2, f3, f4, f5)
                                   ^-- purge MSS files F1, F2, F3, F4,
                                       and F5
MSSTORE Store a file on the Mass Storage System.
         Examples: MSSTORE, in1, mssfyll
                                   ^-- IN1 is stored as private, direct
                                       file MSSFYL1
                    MSSTORE in2 mssfy12 D64 PW=fy1epw
                                   ^-- IN2 is stored as private file
                                       MSSFYL2 (even if MSSFYL2 already
                                       exists) in CDC Display Code --
                                       FYLEPW is the password required
                                       for another user to access the
                                       file -- if MSSFYL2 does not
                                       exist, this will be a direct file
                    MSSTORE in3 mssfy13 A6
                                               I PU
                                   ^-- IN3 is stored as public, indirect
```

.

Rev0

90/09/30

file MSSFYL3 in 64-character Display Code

Page 4-1-11

Page 5-1-1

***** DEC VAXcluster -- VMS *****

The Digital Equipment Corporation (DEC) VAXcluster has two 8550 central processing units (CPUs) or nodes, each with 48 megabytes of memory, which share files and are linked together. Access is via DTNET. A separate VAX 6410 for secure processing is located at Carderock and is accessed in the secure computer room.

*** VMS Version 5.3 ***

The operating system for the DEC VAXcluster and the VAX 6410 at DTRC is called VMS, version 5.3-1.

Permanent files (user programs and data files retained for frequent use) reside on disk drives and the Mass Storage System. User files, if not specifically requested on a tape, will be assigned to available disk areas.

*** Accessing the VAXcluster ***

To access the VAXcluster, set your terminal to 8-bit, no parity, then:

. dial (301) 227-5200

<-- this will connect you with DTNET
at 1200 baud (see page 1-1-4 for
higher speeds)</pre>

- . after the phone call completes, or if you are hardwired into DTNET, press the RETURN key until it displays the DTNET> prompt
- . enter "connect dt4" (or "c dt4") to connect to DT4
 (similarly for DT3)
- . in response to the Username: prompt, enter your User Initials
- . in response to the Password: prompt, enter your login password (the default VAX prompt is \$)

*** Login Password ***

Your initial login password is your username, usually your user initials. This is entered in response to

Password:

the first time you log in. This password MUST be changed during your first session.

To change your login password, type

SET PASSWORD

You will be prompted for the current password, the new password, and the new password again (to insure there were no transmission problems).

Your password should be changed frequently, and must be changed at least every 90 days.

*** Logout Procedures ***

To terminate your session, get rid of any unwanted permanent files (remember that new versions of a file may be made frequently during the session with up to five retained and costing you money). You may also want to get rid of any journal files made by EDT (.JOU) or EVE (.TJL).

When this is done, or immediately, if the Central Site operator requests it, type LOGOUT. A time and usage summary of the session and a cost estimate will be displayed.

You will be returned to your DTNET session. To leave, type "L". This will disconnect the phone, if you have dialed in.

Note

If you do not type anything for about 13 minutes, you will be logged off VMS automatically. You are given a 5-minute warning.

*** System News ***

At login, a system bulletin may be displayed. For more details, type NEWS. To see earlier news items, type OLDNews. To see ancient news items, type VERYoldnews.

*** Login Procedure File ***

A Login Procedure File is a file in your home directory with the name LOGIN.COM. It contains commands to be executed each time you log in before you are given the \$ prompt. Commands and qualifiers should be spelled in full to allow for possible future changes in the operating system.

Any command may be in LOGIN.COM. You may want to see who is logged in (\$ SHOW USERS /FULL), or look at your home directory files (\$ DIRECTORY) or all your files (\$ DIRECTORY [...]), or define one or more of your HELP libraries (\$ ASSIGN UOn: [myid]mylib HLP\$LIBRARY_5). You should also define your home directory with a logical name (such as your first name, but NOT your username) using (\$ DEFINE myname UOn: [xxxx]). Then, you need only type myname: to refer to your home directory, which you may need to do frequently. For suggestions of other commands, symbols and logical names you might include, type "HELP LOGIN.COM_Hints".

Rev0 90/09/30

*** Files ***

1. Because VMS at DTRC automatically deletes the low version number when more than 5 versions of a file are created, you should not use different versions of a file for different purposes. Instead use the file type field.

2. To reduce your file space and, therefore, your costs, you may wish to do a "PURGE [xxxx...]" every now and then to remove all low versions (or "PURGE [xxxx...] /KEEP=2" to keep the highest two versions.

3. When editing with EDT or EVE, a journal file is created of all your editing commands for use in re-editing your file if your editing session is aborted (^Y or a line disconnect). (If your editing session ends normally (EXIT or QUIT), the journal file is deleted.) You should check periodically for any journal files and delete them if they are no longer needed. Use the command "DIRectory /DATE [...]*.JOU,*.TJL" to see them.

*** Batch Jobs ***

A batch job is a procedure which is submitted by the SUBMIT command. By default, the job will be executed on either DT3 or DT4. If your job must run on a specific node, use the /QUEUE=DTn_BATCH qualifier (n is the desired node number). See page 1-3-1 ff for a table of the nodes on which specific software is available.

** Killing Batch Jobs **

When a job is SUBMITted, a message is displayed giving the "entry number". When the job goes into execution, it has a "process ID". To find out these numbers, use

| \$ SHOW | QUEUE *BATCH | < | all VAXcluster batch queues |
|------------|--------------|---|--|
| \$ SHOW | SYSTEM | < | process ID and entry number on current node (if the job is in execution) |

To delete a job which has not gone into execution:

\$ DELETE SYS\$BATCH /ENTRY=entry

To delete an executing job:

- \$ DELETE node_BATCH /ENTRY=entry <-- any node -or-\$ STOP /ID=pid <-- current nod</pre>
 - <-- current node -- pid is
 last 4 digits of process
 ID (leading zeros may be
 omitted)</pre>

Page 5-1-5

*** Accessing Other Networks ***

DTRC also has access to the following networks:

DDN - the Defense Data Network (with connection to INTERNET) (host tables allow transfer to some other networks)

OASYS - the DTRC Office Automation System

The following can be reached from our DECnet using SET HOST:

NAVSEA node names: SEAHUB, SEAA, SEAB, SEAC, SEAD, SEAE

** Checking Host Accessibility **

Host accessibility may be verified on the VAX using the FINDHOST command. FINDHOST will search a downloaded version of the name server host tables. You may enter the address, the host name, or any portion of either the host name or address. A listing will be given of all entries that meet the specified search. The search string will be highlighted in the resulting list.

\$FINDHost <string>

-or-

\$FH <string>

If the requested string is not matched, you will get a message that there were no matches. Call User Services for more information.

** ** Transferring VMS Files To and From OASYS While logged into DTn: ftp dtrc.dt.navy.mil <-- File Transfer Protocol to OASYS -or-(dtrc) via Ethernet ftp dtrc . . . <-- to log into OASYS login <-- your OASYS user name <user name> <password> <-- your OASYS password <-- get a file from OASYS get <OASYS filename> <VMS filename> put <-- send a file to OASYS <VMS filename> <OASYS filename> bye <-- leave ftp ** Transferring VMS Files To and From CDC CYBER 860 ** While logged into DTn: ftp cdc860 <-- connect to CDC CYBER 860 / MSS</pre> <-- your name <CR> <password> <-- your MSS password <account number> <-- if requested get <-- get a file from MSS <MSS filename> <VMS filename> put <-- send a file to MSS <VMS filename> e.g., to make a public indirect <MSS filename> file: put myfile.dat 'myfile/ia,ct=pu' <-- leave ftp bye

See also page 5-1-8.

Page 5-1-7

** Mail to Users at Other Sites **

Mail may be sent to users at other sites which are accessible via DDN. This is one way to transfer large (or small) text files. Use FTP, Kermit, or some other protocol for binary files.

While logged into VMS: \$ mail MAIL> send To: wins%"<user@hostname>" <-- where some typical hostnames ... are: dtrc.dt.navy.mil, dtoal.dt.navy.mil, icst-is.arpa, gwuvax.gwu.edu)

For example, to send a message to "sommer" on dtrc (OASYS B system) from node DT4:

\$ mail MAIL> send To: wins%"sommer@dtrc" <-- the brackets are optional ...

Mail is sent via the VMS mail utility and the Simple Mail Transfer Protocol (SMTP). The "To:" address has one of the following forms:

| Destination | Address Syntax | Utility |
|---|-----------------------------------|-------------------|
| same VAX | user | local VMS mail |
| same network | node::user | DECnet |
| another VAX | wins%" <user@host>"</user@host> | SMTP |
| remote host | wins%" <user@host>"</user@host> | SMTP |
| remote host routed through other hosts on your network | wins%"<@host,@host:user@host>" | SMTP |
| remote host on another network routed through a gateway | wins%"<@host,@gateway:user@host>" | SMTP |

Note that local VMS and DECnet mail is sent immediately; SMTP mail is sent every 20 minutes.

** Mail From Users at Other Sites **

The following are the Ethernet addresses for CCF computers as of the date of this page. Type "HELP @CCF Network_addr" on the VAX for an up-to-date list.

| | computer | | address | |
|------|----------------|--------|--------------|---------------|
| | oasys | | 130.46.1.53 | |
| | dtoal | | 130.46.1.2 | |
| | dtrc | | 130.46.1.3 | |
| | dtoa3 | | 130.46.1.4 | |
| | dtix | | 130.46.1.5 | |
| | nems | | 130.46.1.6 | |
| | dt70 | | 130.46.1.7 | |
| | dt18 | | 130.46.1.8 | |
| | dtvms3 | | 130.46.1.12 | (DT3) |
| | dtvms | dtvms4 | 130.46.1.10 | (DT4) |
| * | cdc8 60 | nos | 130.46.1.16 | |
| * ** | sn417 | | 192.91.138.5 | (Cray UNICOS) |

To access these via DDN add ".dt.navy.mil", e.g., dtvms.dt.navy.mil. Thus, the address for mail to be sent to user ABCD on the VAXcluster via DDN is "abcd@dtvms.dt.navy.mil".

* - No mail. This is listed to show the network address.

** - Available only when UNICOS is up.

Page 5-2-1

***** Help Libraries *****

A help library (file type .HLB) contains help modules, that is, modules that provide information about a program, subprogram, procedure, or some general help information such as hints on how to do something. It is created and accessed using the following DCL commands:

LIBRARY Create, maintain, list, and extract modules from a help library.

HELP Display the desired helps.

*** The System Help Library ***

The system help library is read using the DCL command HELP. It provides help about the HELP program and lists many topics (VMS features, DCL commands, Hints, and other general information).

*** DTRC Help Libraries ***

Four help libraries have been added to VMS at DTRC:

CCF - General information about the Computer Center

- CRAY Routines added to Cray at DTRC
- DTLIB Subprograms in library DTLIB (Cray, CDC NOS, VMS)

UTILITIES - Utility programs and procedures

When executing the HELP command, the additional help libraries are accessed by entering '@name', where 'name' is one of the help libraries listed above (e.g., @DTLIB) in response to 'Topic?'. For a table of contents of any of the above libraries, type

HELP @name Contents

*** User Help Module ***

A help module (default file type .HLP) is a file containing all the help information for one or more programs, procedures, etc. Column 1 of each line identifies the different sections of the help module. A digit indicates a keyword; a slash (/) indicates a qualifier; anything else is part of the help text. For example,

```
1 key-1 <-- HELP topic
...
help message text
...
2 key-2 <-- HELP sub-topic
...
help message text
...
n key-n
...
help message text
...
1 key-1 <-- next HELP topic</pre>
```

A "1" line gives the topic name (up to 15 characters, avoid using blanks; replace blanks with an underscore (_)). A "2" line is a sub-topic of the "1"-level topic; a "3" line is a sub-topic of the most recent "2"-level sub-topic; etc. Qualifiers (/ in column 1) will be listed separately by HELP and will all be displayed if the (sub)topic they qualify is selected.

A help module might look something like:

```
1 topic
<description of topic>
2 Qualifiers
 <optional description of qualifiers>
/topic qualifier 1
<description of topic qualifier l>
/topic qualifier 2
<description of topic qualifier_2>
/topic_qualifier_3
<description of topic_qualifier_3>
2 sub-topic 1
 <description of sub-topic>
3 sub-topic_of_sub-topic_1
<description of sub-sub-topic>
3 Qualifiers of sub-topic 1
<optional description of qualifiers>
/sub-topic_1_qualifier_1
 <description of qualifier 1 of sub topic l>
/sub-topic l qualifier 2
 <description of qualifier 2 of sub topic l>
. . .
```

Page 5-2-3

*** Hints For Designing Help Displays ***

While help messages can continue without interruption, you may wish to format the messages to fit the screen display. A topic ("1" in column 1) will have 17 lines in the first display; a sub-topic ("2" in column 1) will have 15 lines; a sub-sub-topic ("3" in column 1) will have 13 lines; etc. For all levels, the second and following displays have 20 lines. Level 1 lines should not exceed 78 columns; level 2 lines should not exceed 76 columns; level 3 lines, 74 columns; etc. Longer lines may "wrap around".

Every help library should have a module called "HELP" to describe the help library.

You may wish to have a table of contents module (suggested name "Contents") to list the routine names and give a short description of what each routine does.

If possible, the first help screen for a program, subprogram or procedure should contain all that is needed to use it. Definitions of parameters and qualifiers should be put into sub-topics.

*** Selecting (Sub)topic Names ***

While you may choose anything you want for topic and sub-topic names, we recommend the following conventions:

- . use upper case for routine names, parameters, and qualifiers (e.g., AUXPRINT, /CC, /HEADER, JGDATE, FLR below)
- . use lower case (first letter upper case) for general information (e.g., Parameters, Qualifiers, Examples, Admin_info below)
- . replace blanks with underscores () so that the name will be listed as a single element by HELP (e.g., Admin_info below)

*** Create a Help Library ***

The LIBRARY command is used to create a help library.

LIBRARY /HELP /CREATE help library_name

-or-

LIBRARY /HELP /CREATE=(option,...) help_library_name

where help_library_name is the name of the library to be created. It will have the default filename help_library_name.HLB.

The following options may be specified:

- BLOCKS:n The number of 512-word blocks to be allocated. (default: 100)
- HISTORY:n The maximum number of library update history records to be maintained. (default: 20)
- KEYSIZE:n The maximum length of module names. (default: 15)
- MODULES:n The maximum number of modules the library can hold. (default: 256)

*** Modify a Help Library ***

The LIBRARY command is used to insert, and delete help library modules. Wildcards are allowed in module names.

LIBRARY /HELP /INSERT help_library_name help_module_name LIBRARY /HELP /REPLACE help_library_name help_module_name LIBRARY /HELP /DELETE=(module[,...]) help library name

'LIBRARY /HELP help_library_name help_module_name' is the same as if '/REPLACE' were specified. If '/LOG' is specified, a messages will be displayed for each operation done. (E.g., LIBR /HELP /LOG ...)

*** Compress a Help Library ***

After several inserts, deletes or replaces, there may be a lot of "dead space" in the library. To remove this, that is, to compress the library, use:

LIBRARY /HELP /COMPRESS help library name

-or-

LIBRARY /HELP /COMPRESS=(option,...) help_library_name

/LOG will list the modules as they are copied into the compressed library.

The options available are the same as for /CREATE.

Page 5-2-5

*** List the Contents of a Help Library ***

The LIBRARY command also lists the contents of a help library. The /LIST qualifier, which may be specified alone or with any of the above operations, will provide information about the library including a list of the modules in the library. If /FULL is also specified, the list of modules will include the date and time it was inserted into the library. If /HISTORY is specified, it will show who did what to the library and when. The number of history records retained is defined when the library was created or compressed.

For a list of the library without other operations, use

LIBRARY /HELP /LIST -or-LIBRARY /HELP /LIST /FULL -or-LIBRARY /HELP /LIST /FULL /HISTORY

The list will be displayed on SYS\$OUTPUT. To put the listing into a file, use /LIST=filespec.

To list information about specific modules, use /MODULE=(list) where <list> is a comma-separated list of module names with wildcards allowed. The default is /MODULE=*.

To list information about modules inserted after a certain time, use /SINCE (for those inserted today) or /SINCE=date_and_time (for those inserted after a specific date and/or time (e.g., /SINCE=09:00 for those after 9 AM today).

*** Extract a Help Module ***
To extract a help module to make some modifications to it, use
LIBRARY /HELP /EXTRACT=(module[,...])
/OUTPUT=file-spec

If /OUTPUT is specified, the modules are put into file <file-spec>. If /OUTPUT is omitted, they are put into file help library name.HLP.

help_library_name

Wildcards are allowed in module names.

*** Accessing your Help Library ***

To access you help library, use

HELP /LIBRARY=filespec [topic [sub-topic]]

where <filespec> must be complete (e.g., U09:[abcd]mylib), not just the filename.

*** Adding Your Help Library to the System Helps ***

The DCL HELP command supports many user libraries in addition to the system library. User libraries are added by assigning help library names to HLP\$LIBRARY_n, where n is omitted or a digit. HLP\$LIBRARY through HLP\$LIBRARY_4 are already defined at LOGIN. You may add your own help libraries starting with HLP\$LIBRARY_5. For example, you may wish to put

\$ DEFINE /NOLOG HLP\$LIBRARY_5 UOn:[myid]mylib1
\$ DEFINE /NOLOG HLP\$LIBRARY 6 UOn:[myid]mylib2

into your LOGIN.COM file so that your help library will always be part of the system HELP command for you. The first missing number (in this case "7") will end the list. These will be listed at the end of the last screen of the topic display. To access library "5" above, use "HELP \$mylib1", or "@mylib1" at the Topic? prompt.

*** Using HELP ***

The HELP command access the system help library ("HELP"), your library set ("HELP @libname"), or any other help library ("HELP /LIBRary= filespec").

On initial entry into a help library, the help module is displayed, if present, a list of topics, and, perhaps, the library set. At the "Topic?" prompt, enter the name of the topic for which you want help. Only as many characters as are needed to uniquely identify the topic are required. If the name is not unique, all matching topics are displayed.

After the topic has been displayed (may be more than one screen), a list of additional information (sub-topics) may also be shown. At the prompt, enter the sub-topic name.

When you have finished with a level, press RETURN to go up one level. Pressing RETURN at the "Topic?" prompt exits the HELP command. At any prompt (even in the middle of typing an entry, ^Z (CTRL-Z) will terminate HELP.

Enter a question mark (?) at any time to display the most recent (sub)topic again. The actual help displayed depends on how you got to the current level. The RETURN key should not be pressed with the "?", since the "?" is recognized immediately. (If a help library is entered from a program other than the HELP command, the RETURN is required after the "?".)

If you have forgotten the names of the additional (sub)topics, just enter something you know is not a (sub)topic name (in most cases, "ZZ" is sufficient). This will display an error message and show the valid (sub)topic names.

The up-arrow key may be used to bring back your most-recent entry, which may be edited and resubmitted.

Page 5-2-7

*** Sample Help Modules ***

The following are sample help modules for a program, a subprogram, a procedure, general information; and a HELP help module.

** A Program **

The following is a portion of the help module for the AUXPRINT program.

```
1 AUXPRINT <-- topic
List a file on an auxiliary printer (one attached to an
interactive terminal).
```

[/ [NO] CC

[/SKIP=s

[/WIDTH=w

[/LENGTH=1

[/ [NO] HEADER]

Format:

! Defaults

/NOHEADER

/SKIP ==> /SKIP=10

/WIDTH=132

! /LENGTH=66

! /WIDTH=80; ! /WIDTH ==>

! /SKIP=0;

! /NOCC

1

1

ł

]

]

]

AUXPRINT file-spec

2 Parameter <-- sub-topic file-spec Specifies the name of the file to be printed. If omitted, you will be prompted for it. Defaults: extender - .DAT; filename - FOR002 2 Qualifiers <-- sub-topic The qualifiers may follow the command name or the file-spec. If a qualifier is specified more than once, only the final value is retained. /CC /CC /NOCC Specifies whether the file has carriage control in column 1 of each line.

Default: /NOCC (that is, the file does not have carriage control in column 1) /HEADER

/HEADER /NOHEADER

Determines whether the listing will have a heading giving the date and file-spec.

Default: /NOHEADER

2 Admin_info

<-- sub-topic

Language: VAX/VMS Fortran 77

Authors: Dan Allen - DTRC Code 189.2 David V. Sommer - DTRC Code 3511

Date written: 10/81 (da)

Dates revised 03/14/85 - dvs - add qualifiers /CC /HEADER /LENGTH /SKIP 10/22/85 - dvs - shorten /CC output by 1 line systems - change default to /NOHEADER 03/07/86 - dvs - add /WIDTH qualifier - fix /CC processing when first top-of-page is not first record

Page 5-2-9

<-- topic

** ** A Subprogram

This illustrates a subprogram help module. We suggest that such a help have the following sub-topics:

. Parameters (if the routine has them)

- . Examples (at least one example to show how to use the routine)
- . Admin info (to show the source language, author, a brief history, and anything else that might be appropriate)

1 JGDATE

Convert any Gregorian date to a relative Julian number or vice versa.

INTEGER jg, jd, gyear, gmonth, gday Usage:

CALL JGDATE (jg, jd, gyear, gmonth, gday)

The relative Julian number corresponding to a Gregorian date is the number of days since 11/24/-4713 (extrapolating the Gregorian calendar).

This subroutine is useful in determining the elapsed number of days between any two calendar dates. It can also be used to find the calendar date so many days from any given date. <-- sub-topic 2 Parameters

CALL JGDATE (jg, jd, gyear, gmonth, gday)

- in - int - direction of conversion jg 1 - Gregorian to Relative Julian 2 - Relative Julian to Gregorian - out - int - will contain relative Julian number jg=1: jd gyear - in - int - Gregorian year (e.g., 1985) gmonth - in - int - Gregorian month (1-12) gday - in - int - Gregorian day (1-31) - in - int - relative Julian number jg=2: jd

gyear - out - int - will contain Gregorian year (e.g., 1985) gmonth - out - int - will contain Gregorian month (1-12) gday - out - int - will contain Gregorian day (1-31)

2 Examples

<-- sub-topic

INTEGER jd, gy, gm, gd CALL JDDATE (1, jd, 1985, 2, 25) jd = jd + 1000CALL JGDATE (2, jd, gy, gm ,gd)

This example will find the date 1000 days from 02/25/85.

Page 5-2-10

Rev0 90/09/30

2 Admin info <-- sub-topic Fortran 77 Language: David V. Sommer - DTRC Code 3511 Author: Date written: 1968 or earlier Dates revised 03/01/79 - implement on Burroughs 7700 02/01/85 - implement on DEC VAXcluster ** ** A Command Procedure The procedure FLR has the following definition for all users: S FLR :== @VSYS:FLR Without this definition, the "Format" would have @VSYS:FLR [filename] 1 FLR <-- topic Compile Fortran, Link and Run. Format: FLR [filename] If filename is omitted, you will be prompted for it. For execution, FOR005, FOR006 and SYS\$INPUT are assigned to the terminal. Thus, all Fortran READ, PRINT, READ (5,..., WRITE (6,..., TYPE, and ACCEPT statements will read from or write to the terminal.

Ignore the system message "previous value of SYS\$INPUT has been superseded".

Page 5-2-11

** General Information **

The following is a portion of the help module for a discussion of the DTRC accounting for users with more than one account. This module has no sub-topics.

1 Many accounts

VAXcluster users with more than one account are assigned a username/ parsword for each account. These usernames differ in the fifth character position, e.g., CAWE, CAWEA, CAWEB. The default login directory for each user is device: [username] where all files owned by the same individual are stored on the same device. For example,

U01: [CAWE] U01: [CAWEA] U01: [CAWEB]

ACCESSING FILES OWNED BY YOUR ALTER EGO

The "usernames" belonging to a particular user are members of a VMS "group". By default on the VAXcluster, members of a group have Read and Execute access to all files owned by their fellow group members. User CAWEA wishing to access a file owned by CAWE simply references [CAWE] file.ext.

Of course, these access rights can be changed by the SET PROTECTION and SET FILE /ACL commands. In addition, all members of these special "groups" have GRPPRV privilege which, when invoked, gives a member of the group full control, including file creation and deletion, over all files owned by all members of the group. GRPPRV is invoked by

\$ SET PROCess /PRIVileges=GRPPRV

(this would likely be in your LOGIN.COM)

Then to "copy" a file from one account to another, for example CAWE to CAWEA, user CAWEA would

\$ COPY [CAWE]file.ext []

or user CAWE would

\$ COPY file.ext [CAWEA]

To simply "move" a file from one account to another, CAWEA would

\$ RENAME [CAWE]file.ext []
\$ SET FILE /OWNer_uic=CAWEA

Finally, the command MYACcount will indicate the account number of the current session or job, while MYACcount /ALL will provide a list of all user/account pairs in the group.

Page 5-2-12

Rev0 90/09/30

** "HELP" module **

It is recommended, though not necessary, that your help library have a help module named HELP. Such a module will be displayed when you enter the library, and, therefore, should give a brief description of the library and, if appropriate, pointers to related libraries.

The following is the help module HELP for library @CCF:

1 HELP

The CCF help modules provide information of general interest to users of the DTRC Central Computing Facility.

Other help libraries available include:

@CRAY- DTRC additions to Cray@DTLIB- subprograms in library DTLIB (formerly NSRDC)@UTILITIES- utility programs and procedures

Last modified: 31-JUL-1990 13:05:35

Page 5-3-1

***** Procedures *****

A procedure is a group of control statements in a file (default file type .COM). Calling a procedure provides a simplified way to process that group of control statements. A procedure may call another procedure.

Eight parameters, Pl through P8, are available for you (or another procedure) to pass data or other information to a procedure.

Both string and integer variables may be used in a procedure. Several lexical functions are available to interrogate the system, to manipulate variables, etc. Files may be read or written. And, of course, DCL statements may be executed.

*** DTRC Procedures ***

Type HELP @UTILITIES CONTENTS for a list of procedures (and programs) which have been added to the DTRC VAX/VMS system.

Page 5-4-1

***** Object Libraries *****

An object library (file type .OLB) contains compiled subprograms for use in linking with a program.

The Librarian utility LIBRARY is used to create, maintain, list, and extract modules from an object library.

*** DTRC Object Library ***

One object library has been added to VMS at DTRC:

VSYS:DTLIB - Subprograms written or maintained by the Computer Center

To use: LINK yourobj, DTIB/LIB

*** User Object Module ***

An object module (file type .OBJ) is a file containing one or more compiled subprogram(s). They are produced by compiler such as FORTRAN, COBOL, PASCAL, etc.

*** Create an Object Library ***

The LIBRARY command is used to create an object library.

LIBRARY /CREATE object library name

-or-

LIBRARY /CREATE=(option,...) object_library_name

where object_library_name is the name of the library to be created. It will have the default filename object library_name.OLB.

The following options may be specified:

- BLOCKS:n The number of 512-word blocks to be allocated. (default: 100)
- GLOBALS:n The maximum number of global symbols the library can contain. (default: 128)
- HISTORY:n The maximum number of library update history records to be maintained. (default: 20)

Rev0 90/09/30

- KEYSIZE:n The maximum length of module names. (default: 15)
- MODULES:n The maximum number of modules the library can hold. (default: 256)

*** Modify an Object Library ***

The LIBRARY command is used to insert, and delete object library modules. Wildcards are allowed in module names.

LIBRARY /INSERT object_library_name object_module file

LIBRARY /REPLACE object_library_name object_module_file

LIBRARY /DELETE=(module[,...]) object library name

'LIBRARY object_library_name object_module_file' is the same as if '/REPLACE' were specified. If '/LOG' is specified, a message will be displayed for each operation. (E.g., LIBR /LOG ...)

If object_module_file contains several object modules, each will be a separate entity in the object library.

If the qualifier /NOGLOBALS is specified, the global symbols for the modules being inserted will not be put into the global symbol table.

*** Compress an Object Library ***

After several inserts, deletes or replaces, there may be a lot of "dead space" in the library. To remove this, that is, to compress the library, use.

LIBRARY /COMPRESS object library name

-or-

LIBRARY /COMPRESS=(option,...) object library name

/LOG will list the modules as they are copied into the compressed library.

In addition to the options available for /CREATE:

KEEP Copy the history records, etc., to the compressed library. (default: do not copy)

Page 5-4-3

*** List the Contents of an Object Library ***

The LIBRARY command also lists the contents of an object library. The /LIST qualifier, which may be specified alone or with any of the above operations, will provide information about the library including a list of the modules in the library. If /FULL is also specified, the list of modules will include the date and timeit was inserted into the library. If /HISTORY is specified, it will show who did what to the library and when. The number of history records retained is defined when the library was created or compressed.

For a list of the library without other operations, use

LIBRARY /LIST -or-LIBRARY /LIST /FULL -or-LIBRARY /LIST /FULL /HISTORY

The list will be displayed on SYS\$OUTPUT. To put the listing into a file, use /LIST=file-spec.

If the qualifier /NAMES is specified, the names of all global symbols will also be listed.

*** Extract an Object Module ***

To extract an object module to make some modifications to it, use

LIBRARY /EXTRACT=(module[,...] /OUTPUT=file-spec object_library_name

If /OUTPUT is specified, the modules are put into file <file-spec>. If /OUTPUT is omitted, they are put into file object module name.OBJ.

*** Linking with an Object Library ***

If your program uses subprograms in an object library, they can be linked using

LINK your_obj, your_lib/LIBrary

where your_obj is the object module for your program
 your_lib is your object library
 /LIBrary tells the linker that your lib is an object library

If you are linking more than one object file or using more than one object library, you might use one of the following forms:

> LINK obj1, obj2, lib1/LIB LINK obj1, obj2, lib1/LIB, lib2/LIB LINK obj1, obj2, lib1/LIB, obj3 LINK obj1, obj2, lib1/LIB, obj3, lib3/LIB

etc.

Page 5-5-1

***** Text Libraries *****

A text library (file type .TLB) contains text modules, that is, modules containing source programs, documents, notes, data, etc.

The Librarian utility LIBRARY is used to create, maintain, list, and extract modules from a text library.

*** DTRC Text Libraries ***

The following text libraries have been added in VSYS: at DTRC.

- DTLIB Source code for subprograms in library VSYS:DTLIB.OLB
- DTLIBCRAY Source code for subprograms in library DTLIB on the Cray
- INCLUDE Some common block and code segments to INCLUDE in a program or subprogram
- UTILITIES Source code for programs which have been added to VSYS:

*** User Text Module ***

A text module (default file type .TXT) is a file containing a source program, a document, some miscellaneous information, etc.

*** Create a Text Library ***

The LIBRARY command is used to create a text library.

LIBRARY /TEXT /CREATE text_library_name

-or-

LIBRARY /TEXT /CREATE=(option,...) text_library_name

where text_library_name is the name of the library to be created. It will have the default filename text library_name.TLB.

The following options may be specified:

- BLOCKS:n The number of 512-word blocks to be allocated. (default: 100)
- HISTORY:n The maximum number of library update history records to be maintained. (default: 20)
- KEYSIZE:n The maximum length of module names. (default: 15)
- MODULES:n The maximum number of modules the library can hold. (default: 256)

*** Modify a Text Library ***

The LIBRARY command is used to insert, and delete text library modules.

LIBRARY /TEXT text_library_name text_module_file /INSERT LIBRARY /TEXT text_library_name text_module_file /INSERT /MODULE=module_name LIBRARY /TEXT text_library_name text_module_file /REPLACE LIBRARY /TEXT text_library_name text_module_file /REPLACE /MODULE=module_name

LIBRARY /TEXT text_library_file /DELETE=(module[,...])

"LIBRARY /TEXT text_library_name text_module_file" is the same as if "/REPLACE" were specified. If "/MODULE=..." is omitted, the module name will be the filename without the file type. If "/LOG" is specified, a message will be displayed for each operation. (E.g., LIBR /TEXT /LOG ...)

Wildcards are allowed in the module names when deleting.

Page 5-5-3

*** Compress a Text Library ***

After several inserts, deletes or replaces, there may be a lot of "dead space" in the library. To remove this, that is, to compress the library, use:

LIBRARY /TEXT /COMPRESS text library name

-or-

LIBRARY /TEXT /COMPRESS=(option,...) text library name

/LOG will list the modules as they are copied into the compressed library.

The options available are the same as for /CREATE.

*** List the Contents of a Text Library ***

The LIBRARY command also lists the contents of a text library. The /LIST qualifier, which may be specified alone or with any of the above operations, will provide information about the library including a list of the modules in the library. If /FULL is also specified, the list of modules will include the date and time it was inserted into the library. If /HISTORY is specified, it will show who did what to the library and when. The number of history records retained is defined when the library was created or compressed.

For a list of the library without other operations, use

LIBRARY /TEXT /LIST -or-LIBRARY /TEXT /LIST /FULL -or-LIBRARY /TEXT /LIST /FULL /HISTORY

The list will be displayed on SYS\$OUTPUT. To put the listing into a file, use /LIST=file-spec.

*** Extract a Text Module ***

To extract a text module to make some modifications to it, use

LIBRARY /TEXT /EXTRACT=(module[,...]) /OUTPUT=file-spec text library name

If /OUTPUT is specified, the modules are put into file <file-spec>. If /OUTPUT is omitted, they are put into file text library name.TXT.

Wildcards are allowed in the module names.

Page 5-6-1

***** Editors *****

VAX/VMS has two widely-user text editors: EDT and EVE; and a Text Processing Utility (TPU) which can be used to create your own editor. EVE is a editor written in TPU. This chapter gives an overview of EDT and EVE.

*** The EDT Text Editor ***

EDT is used to create or modify a file. There are three modes for using EDT: line, keypad (which uses the full screen), and non-keypad. Line mode is very similar to NETED on the CDC CYBER 176 or 750.

** Invoking EDT **

EDT is executed by:

\$ EDIT /EDT file or \$ EDIT file <--- /EDT is the default editor</pre>

where file may be a file specification or a logical name.

If the file is an existing file, the first line of the file will be displayed on the screen, followed by an * (the * is the prompt when in line mode). If the file does not exist, [EOB] will be displayed on the first line, followed by the * prompt. You are now ready to edit the file. A journal file of every command you enter is saved temporarily in filename.JOU. If EDT is terminated abnormally (including your session being disconnected), you can recover almost all of your editing by "EDIT /RECOVER file".

To change to screen mode, type "change" or "c" at the * prompt. To return to line mode, enter end-of-file (^Z).

** On-line HELP **

Help is available in both line mode and keypad (change) mode. In line mode, at the * prompt, type "HELP" or "HELP command". Keypad mode uses the PF2 command to invoke the help utility. EDT will paint a picture of the keypad and prompt you to push the key for which you need help.

** Terminating EDT **

There are two ways to leave EDT: "EXIT", which saves the file; and "QUIT" which does not save it. If, for some reason, you wish to save the journal file, "EXIT /SAVE" will save both the file and the journal file. Page 5-6-2

*** The EVE Editor ***

The Extensible VAX Editor, EVE, is a full-screen interactive text editor designed for use with VT100- and VT200-compatible terminals. Some features include multiple files and buffers, two windows, and some wor_ processing commands. Advanced editing commands are entered through the use of a command line.

EVE has its own keypad. The EDT keypad may be used by typing "SET KEYPAD EDT" on the command line. In developing EVE, DEC has attempted to simplify the EDT keypad by reducing the number of keystrokes for each keypad command to one.

** Invoking EVE **

To begin an EVE session, enter

```
$ EVE
-or-
$ EVE file
```

A wildcard character, the asterisk, can be substituted for all or some of the characters in a long file name. If one file name matches the specification, that file is edited; otherwise, an error message is issued and no file is used. For example,

```
$ EVE getty.txt
$ EVE this_is_a_long_file_name.and_a_long_file_type
$ EVE this_*.and_*
```

** The Screen **

The screen is divided into four parts. The first part, the window, contains the file's text. If the file is empty, you will only see the [End of file] notice. The second part, the status line, is highlighted, contains the current buffer name, mode, and direction. The third, the command line, displays advanced EVE commands. The fourth part, the message window, displays both informative and error messages.

** On-line Help **

EVE has both keypad help as well as an extensive "word processing" format help menu.

** Terminating EVE **

There are two commands that allow you to leave the EVE environment. To terminate EVE and save the file, type end-of-file (^Z). This will create a new file or another version of an existing file. To leave EVE without saving your changes, press the DO key and then type QUIT. If your editing session ends abnormally, the "EVE /RECOVER file" command can be used to recover your session using journal file file.TJL.

Page 5-6-3

*** Why Use EVE Instead of EDT? ***

EDT users should consider switching to EVE for the following reasons:

- . EVE's use of windows allows editing multiple files simultaneously on the same screen. This is useful for making common changes to programs and subprograms or for moving lines from one file to another.
- . EVE has more ways of extending the basic editor and saving those extensions for future sessions than EDT.
- . EVE's string searching capability is much more flexible than EDT. It includes VMS and UNIX wildcard searching.
- . EVE offers "spawn" and "attach" and "DCL" commands to allow the user to work outside of the current process and return to the same active EVE session.
- . EVE supports the EDT keypad.

Page 6-1-1

***** Magnetic Tape *****

Magnetic tapes should be used for sequential data for such purposes as:

- . Transfer of information to and from other computers and off-line peripherals
- . Files which are used infrequently
- . Back-up copies of disk files
- . Long-term storage of data

Tapes should not be used for scratch files or random information. For safety, two copies on different tapes should be maintained, or for data which is updated, a grandfather-father-son system is advised. It is not wise to mount a tape containing good data, read through it, and write new data at the end. Instead, copy the existing data to a second tape and add the new data to the second tape, retaining the first tape as a back-up.

Processing a file on tape will take considerably more I/O time than on disk and more elapsed time.

Information concerning the physical and logical characteristics of the tape is specified in control statements.

Nine-track tapes are supported on the DEC VAX and CDC CYBER 860 computers; 7-track tapes are supported on the CDC CYBER 860 (NOS only). There are no tape drives on the Cray, so tapes must be accessed via one of the front ends.

*** Tape Labels ***

Tapes may be labelled or unlabelled. Labels should always be used except when writing data for, or reading data from a computer which cannot handle ANSI standard labels.

In general, a labelled tape has volume and end-of-volume labels, and may also have user labels. Each file on the tape may have its own header and trailer labels.

*** Tape Formats ***

Generally, records on tape are fixed or variable length, blocked or unblocked, ASCII or EBCDIC (9-track), BCD (7-track), coded, or binary. Where possible, tapes written by or for another computer should be 9-track, 6250 or 1600 cpi, fixed length, blocked, ASCII. *** Tape Care and Cleaning ***

Tapes should be stored in closed containers in racks which give them vertical support. Tapes may not be spliced. They should be read and rewound at least every six months. Logs should be kept on contents, format, and creation dates of tapes.

If a tape has many parity errors, cleaning it may help. Even a brand new tape may need cleaning. This off-line process does not destroy the information on the tape. If a tape receives heavy usage, cleaning it after ten or more uses may reduce the incidence of parity errors. A tape can also be certified, which determines whether there are any areas on the tape which do not record properly. Certification DESTROYS current information on the tape (except VSN). To change the VSN, contact the Tape Librarian and request blank labelling or degaussing.

If, after a tape has been cleaned, it still has many parity errors, call User Services to have the tape drive cleaned. If the tape continues to have parity errors, it should be exchanged for a new tape. The information on the old tape is not recovered automatically in this case.

To have a tape cleaned or certified, submit an off-line work request to the Tape Librarian. Users who are not at the Carderock site should call (301) 227-1967.

When possible, slot tapes should be in the Computer Room environment for at least two hours before reading or writing. This allows temperature and humidity to stabilize and should minimize tape problems.

Please notify Code 3511 (User Services), (301) 227-1907, of any unusual tape problems.

Page 6-1-3

*** Tape Assignment ***

Two classes of tape storage are provided in the Computer Center, 'Library' and 'Slot'. Tapes which are used frequently should be permanently stored in the NA cabinet, which is accessible from the CYBER 860 or VAXcluster. These tapes are assigned a permanent external label indicating location by cabinet, shelf and position, such as 'NA2499', and are referred to as 'Library' tapes. The volume serial number (VSN) of a Library tape is the same as the external label and should usually be a labelled tape.

Tapes which are seldom used on the CDC CYBER or VAX computers, which are being transferred between systems, or which are normally retained by the owner are assigned a temporary slot number for up to 24 hours at the computer on which they are to be used. At the end of the day's processing (or earlier at the user's request), these are returned to the ADP Control Center for pickup by the user and will require a new slot number assignment for the next use.

The VSN for a slot tape is 'SLOTxx=id' where xx is the assigned slot number id is the user's external sticker on the tape reel (six (6) one-inch-high characters, please, for easy reading by the operator)

Tapes belonging to remote users may be sent to the Tape Librarian. Special slots may be assigned for several weeks' continuous usage (on the CYBER 860, these are C1-9, Y1-9, B1-9, R1-9; on VAXcluster these are V1-9, A1-9, X1-9, S1-9).

All tapes to be used in a job must be supplied by the user as Library tapes and/or Slot tapes. No scratch tapes are available.

Tapes stocked by the Computer Center are of 2400-foot nominal length (10.5 in. diameter). Smaller tapes may be used. For remote slot assignment, assignment of library tapes, or to arrange for the purchase of tapes, contact the Tape Librarian, (301) 227-1967. CYBER procedure BEGIN, TPGET may be used to acquire NA tapes. Slot tapes may be signed in at the ADP Control Conter.

Page 6-1-4

*** Using Tapes on the DEC VAX ***

The DEC VAXcluster has four 9-track tape drives (6250/1600 cpi).

The following VMS control statements are used to access or analyze magnetic tapes:

ALLOCATE Assign a tape drive to a logical name.

DEALLOCATE Return a previously allocated device and disassociate the job's logical name from the tape drive.

DISMOUNT Release a tape volume that was previously mounted.

INITIALIZE Initialize a magnetic tape.

MOUNT Mount a magnetic tape and, if labelled, check the label.

The following prodecures have been developed to handle the tape mounting and dismounting for you:

COPYD2T Copy disk files to a VAX tape using COPY.

COPYT2D Copy a VAX tape (written by COPY) to disk.

- FILEMANAGER An interactive procedure using the VMS BACKUP utility to create, add to, restore from, or list the contents of a backup tape.
- RFTAPE Read Foreign TAPE (copy tape-to-disk). Reads one or more files from a fixed, blocked or unblocked, ASCII or EBCDIC tape and saves them on disk.
- WFTAPE Write Foreign TAPE (copy disk-to-tape). Writes one or more disk files to a fixed, blocked or unblocked (ASCII or EBCDIC) tape.

Page 6-1-5

** Examples **

1. Initialize a VAX/VMS tape:

```
$! TAPINIT.COM : initialize VAX/VMS tape, default is 1600
$1
     if p3 .nes. "1600" .and. p3 .nes. "6250" then p3 = "1600"
$
$1
Ŝ
     allocate mu: tape
                                      1 get any available tape drive
$!
$
     mount tape: /foreign /density='p3' -
                 /comment="mount slot''pl' vsn=''p2' ringin"
$
     dismount tape /nounload
$
     initialize tape 'p2'
$
     deallocate tape
S
     exit
$1
$! p1 - 1- or 2-digit slot number or NONE
$1 p2 - 6-character VSN
$! p3 - density (6250 or 1600) defaults to 1600
$!
1 created 06/23/88 by CASG
$! last modified 06/24/88 @ 1146 by CASG (add "?" for help)
$1
$! End of TAPINIT.COM
```

The above is a portion of the actual procedure to show just the defaulting of density and how to initialize a tape. To see the full procedure, which includes validation of each parameter, and allows "?" for help for the procedure and each parameter, type "TYPE VSYS:TAPINIT.COM".

هي در

*** Using Tapes on the CYBER 860 ***

The CDC CYBER 860 has six 9-track tape drives (four for 6250/1600 cpi and two for 1600/800 cpi), and two 7-track tape drives (800/556 cpi). All drives are available to NOS; two 9-track (6250/1600 cpi) drives are available to NOS/VE.

** NOS **

The following NOS control statements are used to access or analyze magnetic tapes:

LABEL Mount a magnetic tape and, if labelled, check the label.

LISTLB List the labels of an ANSI-labelled tape.

RESOURC Specify that more than one tape drive is required.

TDUMP Octal and alphanumeric dump of all or part of a file.

VSN Associate a local file name with one or more volume serial numbers.

* Examples *

The following examples illustrate tape usage in batch jobs. Tapes may also be used interactively (without the job, USER and CHARGE statements).

1. Unlabelled NOS/BE tape to disk:

xxxx. job statement. USER,xxxx,upw. CHARGE,1234567890. DEFINE,disk/CT=PU. LABEL,tape,F=SI,LB=KU,VSN=NA9999,D=1600,PO=R,R. COPYBF,tape,disk,5. UNLOAD,tape.

2. Copy old stranger (foreign) tape to new - 6250 multifile:

xxxx. USER,xxxx,upw. CHARGE,1234567890. RESOURC,GE=1. <--- one additional tape drive LABEL,t4,VSN=NA9998,D=GE,PO=W,W,F=S,L=softwstr. VSN,t5=SLOTxx=CA9995. LABEL,t5,PO=R,R,F=S,D=GE,L=softwstr. COPY,t5,t4,EL=10,M=coded,PO=E. UNLOAD,t5.

Page 6-1-7

** NOS/VE **

The following NOS/VE control statements are used to access or analyze magnetic tapes:

CHANGE_TAPE_LABEL_ATTRIBUTES Change the current magnetic tape label attributes.

DETACH FILE

Detach one or more files from a job.

DISPLAY_BACKUP_LABEL_TAPE Display the current job default label type for a permanent file backup file on tape.

DISPLAY_TAPE_LABEL_ATTRIBUTES Display the current magnetic tape label attributes.

REQUEST_MAGNETIC_TAPE Associate a file with a magnetic tape.

SKIP_TAPE_MARK Position a tape backward or forward.

* Examples

x

1. Read an unlabelled tape on VE:

/set_working_catalog \$user /change_block_label_type file_label≈u /request_magnetic_tape file=\$local.tape/ external_vsn=mytape .. type≈mt9\$1600 /copy_file \$local.tape myfile1 /detach \$local.tape

```
2. Create a multi-file labelled tape:
  /reqmt file=$local.tape1 ..
          external vsn=TAPE02 ..
  ../
          recorded_vsn=TAPE02 ..
  ../
  ../
          ring=false ...
          density=mt9$6250
   ../
  /chatla f=$local.tape1 ..
  ../
           rf=true
           file_identifier=file1 ..
   ../
          file set identifier=manyl
   ../
  /set_file_attributes f=$local.tapel ..
                        block_type=user_specified_record ..
  ../
   ../
                       record_type=ansi_fixed ..
   ../
                       maximum record_length=80
  /copf file1
                           $local.tape1
  /chatla f=$local.tape1 fi=file2
           file2
  /copf
                           $local.tape1
  /chatla f=$local.tape1 fi=file3
  /copf
           file3
                           $local.tape1
  /distla f=$local.tapel do=current_file " display label written
  /detach $local.tape1
```

Page 7-1-1

***** Other Software *****

This chapter discusses various programming languages and other software packages available on the CCF computers.

ABAQUS A family of modeling capabilities based on the finite element method, designed to provide solutions to a wide range of mostly non-linear structural problems, and programmed around a common data management structure.

Execution: Cray COS: from the VAX: @VSYS:ABACRAY

DEC VAX/VMS: @VSYS:ABA

Post-processing of Cray or VAX runs is done on the VAX: @VSYS:ABAPLOT

Remarks: Processing is normally done on the Cray unless more memory is required than is available.

For Cray processing with .FIL output files, the .INP file must include "*FILE FORMAT, ASCII".

If a plot file is generated on the Cray, each *PLOT statement must include "OUTPUT=ASCII".

References: Machine-readable:

VMS: HELP ABAQUS

Contact: Pete Matula, (301) 227-1936 Mike Brown, (301) 227-1706 С A modern, block-structured programming language designed for portability, system programming, and general-purpose applications. It is derived from Algo1-60. CPP, 'inputfile'. Execution: Cray COS: ^-- C Pre-Processor CC. <-- C compiler SEGLDR,CMD='LIB=\$CLIB;STACK'. DEC VAX/VMS: \$ CC References: "C - A Reference Manual", Hardison and Steele Hardware manufacturers' reference manuals Machine-readable: VMS: HELP CC

CMS

(Code Management System) A source code library maintenance system which can be used for any ASCII file. CMS tracks the history of the file (changes, reason for change, who made the change and when). It can merge modifications; and stores the current and historic versions of the file.

Execution: DEC VAX/VMS: \$ CMS CMS>command -or-\$ CMS command

References: Machine-readable:

VMS: HELP CMS -or-\$ CMS CMS>help

<-- internal help</pre>

Page 7-1-4

Rev0 90/09/30

DataTrieve (DTR)

VAX DataTrieve is a data management system which runs on VMS. It is a tool for defining, storing, updating, and displaying data. The data may reside either in a relational database created through DTR or an existing RMS file. It provides interactive and program-callable access to data, a report writing facility, a graphics capablility, screen formatting support using FMS (Forms Management System), and distributed access on a network connected by DECnet.

Execution: DEC VAX/VMS: \$ DTR32

Remarks: Data formats, procedures, and other data structures are stored in the Common Data Dictionary (CDD).

Users wishing to use DTR must have a valid CDD path established for them by User Services.

References: Machine-readable:

VMS: HELP DATATRIEVE -or-\$ dtr32 DTR>help

<-- internal help</pre>

DISSPLA (Display Integrated Software System and Plotting Language) A library of Fortran subroutines which facilitate plotting. It does not rely upon features particular to any type of graphic device.

Execution: Cray COS: (version 10.0)

ACCESS, DN=DISSPLA, OWN=PUBLIC. SEGLDR, CMD='LIB=DISSPLA'

To dispose the meta file DISPLOT for post-processing on the VAX:

DISPOSE, DN=DISPLOT, DF=BB, TEXT='DISPLOT.DAT'.

DEC VAX/VMS: (version 10.5)

To post-process files created by "CALL COMPRS":

\$ RUN VSYS:DISPOP

Remarks: Cray post-processing must be done on the VAX.

References: DISSPLA User's Manual

Machine-readable:

VMS: HELP DISSPLA

ł

DTLIB A library of subprograms written or supported by the CCF. The contents of DTLIB (formerly call NSRDC) is different on each machine, but generally includes routines in the areas of: . character manipulation

- . sorting
- . date/time manipulation
- . debugging aids
- . extraction of job information
- . some of the Fortran 8x intrinsics

Usage: Cray COS: ACCESS, DN=DTLIB, OWN=PUBLIC. DEC VAX/VMS: \$ LINK <obj>, DTLIB/LIBrary CDC 860 NOS: ATTACH, DTLIB/UN=NSYS. References: Machine-readable: Cray: on VAX, "HELP @DTLIB"

VAX: HELP @DTLIB

DYNA3D DYNA3D is an explicit three-dimensional finite element code for analyzing the large deformation dynamic response of inelastic solids and structures. A contact-impact algorithm permits gaps and sliding along material interfaces with friction. Using a specialization of this algorithm, such interfaces can be rigidly tied to admit variable zoning without the need of transition regions. Spatial discretization is achieved by the use of 8-node solid elements, 2-node beam elements, 4-node shell elements, 8-node solid shell elements, and rigid bodies. The equations-ofmotion are integrated in time by the central difference method. The 1989 version of DYNA3D contains thirty material models and ten equations of state to cover a wide range of material behavior.

Execution: Cray COS:Use VMS

DEC VAX/VMS: @VSYS:DYNA3D

Remarks: This DYNA3D procedure creates a Crey batch job from user responses to pertinent questions. There is an option to have the Cray job submitted by the procedure In the Cray job, the binary plot files are restructured by program CVBIN so that they may be read by the TAURUS graphics post-processor on the VAXcluster.

References:

Cray:

Machine-readable:

VMS: HELP DYNA3D

GPSS General Purpose Simulation System (GPSS) is a generalized simulation package. Execution: DEC VAX/VMS: \$ GPSS qualifiers parameters CDC 860 NOS: ATTACH, GPSS/UN=APPLLIB. GPSS, parameters. ^-- use FX for fixed format The VMS and CDC versions are different. Remarks: References: The IBM document. General Purpose Simulation System Reference Manual, Simulation Software Ltd. (VAX/VMS version) Machine-readable:

VMS: HELP GPSS

Page 7-1-9

IMSL (proprietary) The International Mathematical and Statistical Libraries package (edition 10) contains 948 subroutines in the following areas:

. 426 general applied mathematics routines

- . 351 statistics routines
- . 172 special functions

IMSL 10 was a major revision.

Major enhancements were made in many areas of numerical math. Most statistical analysis subprograms can print results, handle missing values, and implement advances in algorithms. There is no ERROR parameter in the argument list and no need for you to dimension work arrays. Workspace is allocated out of a common area. Informative messages are printed when errors occur. Matrices no longer require packing into onedimensional arrays. Some user-supplied external subprograms must now be functions.

CHARACTER variables are used in the routines and in the many intermediary routines not explicitly called by the application.

| Cray COS: | ACCESS,DN=IMSL,OWN=PUBLIC. SEGLDR,,CMD='LIB=IMSL', |
|--------------|---|
| DEC VAX/VMS: | add "IMSL/LIBrary to the LINK statement |
| CDC 860 NOS: | may be used only by FTN5 programs and is in two permanent files: |
| | IMSLM - the Math routines IMSLSS - the Special function and Statistics routines |
| | If both the mathematics and statistics packages are needed, you must use the following search order: |
| | ATTACH, IMSLM, IMSLSS/UN=NSYS. LIBRARY, IMSLSS, IMSL. -or- LDSET, LIB=IMSLSS/IMSLM. |
| | DEC VAX/VMS: |

| References: | The IMSL documentation is in three sections: |
|-------------|--|
| | MATH/Library V1.0 - general applied mathematics |
| | STAT/Library V1.0 - statistics |
| | SFUN/Library V2.0 - special functions |
| | Also, |
| | Update Guide - describes the differences with the previous version |
| | Machine-readable: |
| | DEC VAX/VMS: HELP IMSL |
| Contact: | User Services |

90/09/30 Rev0

Page 7-1-11

INGRES A relational database management system marketed by Ingres Corporation. Transactions against the database are done through SQL (an ANSI standard query language) or through forms-based utilities accessed by name or through INGMENU, a user-friendly, forms-based interface to the INGRES utilities.

> Execution: DEC VAX/VMS: \$ SETINGRES ^-- once to define INGRES symbols \$ INGMENU <data_base>

> > \$ name <-- a specific utility</pre>

Remarks: You must be an authorized INGRES user before you may access any of the INGRES utilities, including INGMENU. Call User Services to register.

References: Machine-readable:

VMS: HELP INGRES

Rev0 90/09/30

KERMIT File transfer system to/from microcomputers.

Execution: DEC VAX/VMS: KERMIT

CDC 860 NOS: GET,KERMIT/UN=NSYS. KERMIT.

Remarks: To use Kermit on the VAX or CDC CYBER, you must have Kermit on your PC (it might be a subset of PROCOMM).

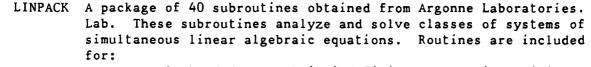
> VAX files to be transferred should have carriage return carriage control. Files with Fortran carriage control or with Print control will not transfer properly.

References: Machine-readable:

VMS: \$ kermit Kermit-32> help <-- internal help

NOS: BEGIN, HELP,, KERMIT, outfyl. ^-- a 7-page document

Usage:



- . general, banded, symmetric indefinite, symmetric positive definite, triangular, tridiagonal square, and Hermetian matrices
- . orthogonal-triangular and single value decompositions of rectangular matrices
- . least square problems
- . basic linear algebra problems

There are four versions:

| precision | prefix | VAXcluster | Cray | NOS |
|------------|--------|------------|------|-----|
| · · · | | | | |
| single | S | | x | x |
| double | D | x | | |
| complex | С | | x | |
| complex*16 | Z | x | | |

Cray COS: part of SCILIB

DEC VAX/VMS: LINK <obj>, VSYS:LINPACK/LIBrary

CDC 860 NOS: GET,LINPACK/UN=NSYS. LDSET,LIB=LINPACK. -or-LIBRARY,LINPACK.

References: "LINPACK Users' Guide", J. J. Dongara, J. R. Bunch, C. D. Moler, G. W. Stewart, SIAM, 1979.

Machine-readable documentation may be listed using:

DEC VAX/VMS: "HELP LINPACK" "HELP LINPACK x<routine>" where x<routine> is "x" followed by the single precision name

Page 7-1-14

MMS (Module Management System) used to automate the assembly of software. MMS reads a system file and determines what has changed since the last "system build" and reassembles.

Execution: DEC VAX/VMS:

Remarks: Eliminates recompiling if the program has not changed since the system was last built.

References: Machine-readable:

VMS: HELP MMS

90/09/30 Rev0

Page 7-1-15

NASTRAN A general-purpose finite element structural analysis program capable of performing a wide range of analysis on models of complex structures, including static stress analysis, natural frequency analysis, buckling analylsis, frequency response analysis, and transient response analysis.

> Execution: Cray COS: ACCESS, DN=NASTRAN, ID=RPK, OWN=PUBLIC. CALL, DN=NASTRAN, CNS. NASTRAN, I=mydata. ^-- simple execution (MYDATA must be ACCESSed prior to this)

References: DTNSRDC/CMLD-81-05: NASTRAN Theory and Application Course Supplement

Machine-readable:

VMS: HELP NASTRAN <-- for Cray version

Contact:

Tony Quezon, (301) 227-1645

PASCAL A modern programming language designed for general-purpose applications. It is derived from Algol-60.

Execution: Cray COS: PASCAL DEC VAX/VMS: \$ PASCAL CDC 860 NOS: PASCAL. References: "Pascal - User Manual and Report", Jensen and Wirth Hardware manufacturers' reference manuals Machine-readable:

VMS: HELP PASCAL

Page 7-1-17

PCA (Performance and Coverage Analyzer) Pinpoints performance problems; analyzes programs written in several languages; reports on performance characteristics; can plot a program's use of resources using histograms or tables. Execution: DEC VAX/VMS: \$ PCA References: Machine-readable:

VMS: HELP PCA

See also: Cray COS: SPY

SPICE A general-purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analyses. Circuits may contain resistors, capacitors, inductors, mutual inductors, independent voltage and current sources, four types of dependent sources, transmission lines, the four most common semiconductor devices: diodes, BJT's, JFET's, and MOSFET's, and a Josephson Junctions model.

> Execution: Cray COS: ACCESS, PDN=SPICE, OWN=PUBLIC. SPICE. /EOF <SPICE data>

References: SPICE 2G.2.5 (Program Reference), E. Cohen, University of California (420 pages)

SPICE Version 2G User's Guide, 8 Nov 1982 (73 pages)

Machine-readable:

VMS: . HELP SPICE . VSYS:SPICE.DOC (the User's Guide)

***** Appendix A *****

*** ASCII Character Set ***

| char | ASCII (hex) | EBCDIC (hex) | Display (octal) | cha | r ASCII (hex) | EBCDIC (hex) | Display (octal) |
|------------|----------------|-----------------|--------------------|------------|------------------|-----------------|--------------------|
| NUL | 00 | 00 | | (((| | 4D | 51 |
| SOH | 01 | 01 | |))) | | 5D | 52 |
| STX | 02 | 02 | | *** | | 5C | 47 |
| ETX | 03 | 03 | | +++ | | 4E | 45 |
| EOT | 04 | 37 | | ,,, | 2 C | 6B | 56 |
| ENQ | 05 | 2D | | | 2D | 60 | 46 |
| ACK | 06 | 2E | | • • • | 2E | 4B | 57 |
| BEL | 07 | 2 F | | /// | 2F | 61 | 50 |
| BS | 08 | 16 | | 000 | | FO | 33 |
| HT | 09 | 05 | | 111 | | F1 | 34 |
| LF | 0A | 25 | | 222 | | F2 | 35 |
| VT | OB | OB | | 333 | | F3 | 36 |
| FF | 0C | 0C | | 444 | | F4 | 37 |
| CR | 0D | 0D OF | | 555 | | F5 | 40 |
| S0 S1 | 0E 0F | 0E OF | | 666 777 | | F6 F7 | 41 42 |
| 51 | UF | Ur | | /// | 37 | r / | 42 |
| DLE | 10 | 10 | | 888 | 38 | F8 | 43 |
| DC1 | 11 | 11 | | 999 | 39 | , F9 | 44 |
| DC 2 | 12 | 12 | | ::: | 3A | 7A | 63 |
| DC3 | 13 | 13 | | ••• | | 5E | 77 |
| DC4 | 14 | 3C | | <<< | | 4C | 72 |
| NAK | 15 | 3D | | === | • | 7E | 54 |
| SYN | 16 | 32 | | >>> | | 6E | 73 |
| ETB | 17 | 26 | | ??? | 3F | 6F | 71 |
| CAN | 18 | 18 | | 000 | | 7C | 74 |
| EM | 19 | 19 | | AAA | | C1 | 01 |
| SUB | 1A | 3F | | BBB | | C2 | 02 |
| ESC | 1B | 27 | | CCC | | C3 | 03 |
| FS | 10 | 1C | | DDD | | C4 C5 | 04 05 |
| GS RS | 1D 1E | 1D 1E | | EEE FFF | | C6 | 05 |
| US | 1E 1F | 1E 1F | | GGG | | C7 | 07 |
| 03 | 11 | 11 | | 666 | • • • • | τ, | 07 |
| space | | 40 | 55 | ннн | | С8 | 10 |
| 111 | 21 | 4F | 66 | III | | С9 | 11 |
| ** * * * * | 22 | 7 F | 64 | JJJ | | D1 | 12 |
| ### | 23 | 7B | 60 | ККК | | D2 | 13 |
| \$\$\$ | 24 | 5B | 53 | LLL | | 53 | 14 |
| 7. 7. % | 25 | 6C | | MMM | | D4 | 15 |
| 666 | 26 | 50 | 67 | NNN | | D5 | 16 |
| * * * | 27 | 7D | 70 | 000 | 4F | D6 | 17 |

| char | ASCII (hex) | EBCDIC (hex) | Display (octal) | char | ASCII (hex) | EBCDIC (hex) | Display (octal) |
|-------|----------------|-----------------|--------------------|------|----------------|-----------------|--------------------|
| PPP | 50 | D7 | 20 | hhh | 68 | 88 | |
| QQQ | 51 | D8 | 21 | iii | 69 | 89 | |
| RRR | 52 | D9 | 22 | iii | 6A | 91 | |
| SSS | 53 | E2 | 23 | kkk | 6B | 92 | |
| TTT | 54 | E3 | 24 | 111 | 6C | 93 | |
| ບບບ | 55 | E4 | 25 | mmm | 6D | 94 | |
| VVV | 56 | E5 | 26 | nnn | 6E | 95 | |
| WWW | 57 | E6 | 27 | 000 | 6F | 96 | |
| XXX | 58 | E7 | 30 | PPP | 70 | 97 | |
| YYY | 59 | E8 | 31 | qqq | 71 | 98 | |
| ZZZ | 5a | E9 | 32 | rrr | 72 | 99 | |
|]]] | 5B | 4A | | ŚSS | 73 | A2 | |
| /// | 5C | EO | 75 | ttt | 74 | A3 | |
|]]] | 5D | 5A | | uuu | 75 | A4 | |
| ^ ^ ^ | 5E | 5F | 76 | vvv | 76 | А5 | |
| | 5F | 6D | 65 | www | 77 | A6 | |
| grave | 60 | 79 | | xxx | 78 | А7 | |
| aaa | 61 | 81 | | ууу | 79 | A8 | |
| bbb | 62 | 82 | | 22Z | 7A | А9 | |
| ccc | 63 | 83 | | {{{ | 7B | C0 | 61 |
| ddd | 64 | 84 | | | 7C | 6A | |
| eee | 65 | 85 | | }}} | 7D | D0 | 62 |
| fff | 66 | 86 | | ~~~ | 7E | A1 | |
| ggg | 67 | 87 | | DEL | 7F | 07 | |

90/09/30 Rev0

Page A-3

| | | *** (| CDC NOS | Characte | r Set | *** | |
|-----------------|----------------|--------------|----------------------------|--------------------|------------------------|------------------------|-----------------|
| Display Code | char- acter | punch 026 | punch 029 if diff | 7-track ext BCD | 9-ti ASCII (noto | rack EBCDIC e 6) | note/name |
| | | | | | | | |
| 00 | ::: | 2-8 | | <i>.</i> | 25 | 6C | colon (1,2) |
| 01 | AAA | 12-1 | | 61 | 41 | C1 | |
| 02 | BBB | 12-2 | | 62 62 | 42 | C2 | |
| 03 04 | CCC DDD | 12-3 12-4 | | 63 64 | 43 44 | C3 C4 | |
| 04 | EEE | 12-4 | | 65 | 44 | C5 | |
| 06 | FFF | 12-6 | | 66 | 46 | C6 | |
| 07 | GGG | 12-7 | | 67 | 47 | C7 | |
| 10 | ннн | 12-8 | | 70 | 48 | C8 | |
| 11 | III | 12-9 | | 71 | 49 | C9 | |
| 12 |]]] | 11-1 | | 41 | 4A | D1 | |
| 13 | KKK | 11-2 | | 42 | 4B | D2 | |
| 14 | LLL | 11-3 | | 43 | 4C | D3 | |
| 15 | MMM | 11-4 | | 44 | 4D | D4 | |
| 16 | NNN | 11-5 | | 45 | 4E | D5 | |
| 17 | 000 | 11-6 | | 46 | 4F | D6 | |
| 20 | PPP | 11-7 | | 47 | 50 | D7 ' | |
| 21 | QQQ | 11-8 | | 50 | 51 | D8 | |
| 22 | RRR | 11-9 | | 51 | 52 | D9 | |
| 23 | SSS | 0-2 | | 22 | 53 | E2 | |
| 24 | TTT | 0-3 | | 23 | 54 | E3 | |
| 25 | UUU | 0-4 | | 24 | 55 | E4 | |
| 26 27 | VVV WWW | 0-5 0-6 | | 25 26 | 56 57 | E5 E6 | |
| 30 | XXX | 0-7 | | 20 | 58 | E0 E7 | |
| 31 | YYY | 0-8 | | 30 | 59 | E8 | |
| 32 | ZZZ | 0-9 | | 31 | 57 5a | E9 | |
| 33 | 000 | Ő | | 12 | 30 | FO | (sometimes 00) |
| 34 | 111 | 1 | | 01 | 31 | F1 | (0000001000 00) |
| 35 | 222 | 2 | | 02 | 32 | F 2 | |
| 36 | 333 | 3 | | 03 | 33 | F3 | |
| 37 | 444 | 4 | | 04 | 34 | F4 | |
| 40 | 555 | 5 | | 05 | 35 | F5 | |
| 41 | 666 | 6 | | 06 | 36 | F6 | |
| 42 | 777 | 7 | | 07 | 37 | F7 | |
| 43 | 888 | 8 | | 10 | 38 | F 8 | |
| 44 | 999 | 9 | | 11 | 39 | F9 | |
| 45 | +++ | 12 | 12-6-8 | 60 | 2B | 4E | plus |
| 46 | <u></u> | 11 | | 40 | 2D | 60 | minus |
| 47 | *** | 11-4-8 | | 54 | 2A | 5C | asterisk |
| 50 | | 0-1 | | 21 | 2F | 61 | slash |
| 51 | (((| 0-4-8 | 12-5-8 | 34 | 28 | 4D | left paren |
| 52 52 |))) | 12-4-8 | 11-5-8 | 74 52 | 29 24 | 5D | right paren |
| 53 54 | \$\$\$ | 11-3-8 | <u> </u> | 53 | 24 2D | 5B 75 | dollar |
| 54 | #- | 3-8 | 6-8 | 13 20 | 3D 20 | 7E | equal |
| 56 | | 0-3-8 | | 33 | 20 2C | 40 6B | blank |
| 57 | ,,, | 12-3-8 | | 73 | 2C 2E | ов 4В | comma period |
| 51 | • • • | 12 3 0 | | | <i></i> | עד | herron |

| Display Code | char- acter | punch 026 | punch 029 if diff | 7-track ext BCD | 9-tı ASCII (note | EBCDIC | note/name |
|-----------------|----------------|----------------|----------------------------|--------------------|------------------------|--------|---------------|
| 60 | #### | 0-6-8 | 3-8 | 36 | 23 | 7B | pound |
| 61 |]]] | 7-8 | 12-2-8 | 17 | 5B | 4A | left bracket |
| 62 |]]] | 0-2-8 | 11-2-8 | 32 | 5D | 5A | right bracket |
| 63 | %%% | 2-8 | | | 25 | 6C | percent (1,2) |
| 64 | ** ** ** | 4-8 | 7-8 | 14 | 22 | 7F | quote |
| 65 | | 0-5-8 | | 35 | 5F | 6D | underline |
| 66 | 111 | 11-2-8 | 12-7-8 | 52 | 21 | 4F | exclam (3) |
| 66 | 111 | 11-0 | | 52 | 2 1 | 4F | exclam (3) |
| 67 | 888 | 0-7-8 | 12 | 37 | 26 | 50 | ampersand |
| 70 | | 11-5-8 | 5-8 | 55 | 27 | 7D | apostrophe |
| 71 | ??? | 11-6-8 | 0-7-8 | 56 | 3F | 6F | question |
| 72 | <<< | 12-2-8 | 12-4-8 | 72 | 3 C | 4C | less than (3) |
| 72 | <<< | 1 2-0 | | 72 | 3 C | 4C | less than (3) |
| 73 | >>> | ⊥ 1−7−8 | 0-6-8 | 57 | 3E | 6E | greater than |
| 74 | 666 | 5-8 | 4-8 | 15 | 40 | 7C | at |
| 75 | /// | 12-5-8 | 0-2-8 | 75 | 5C | EO | reverse slant |
| 76 | ~ ^ ^ | 12-6-8 | 11-7-8 | 76 | 5E | 5F | caret |
| 77 | ;;; | 12-7-8 | 11-6-8 | 77 | 3B | 5E | semicolon (4) |
| 55 | | 6-8 | 0-4-8 | | 20 | 40 | blank (5) |

Notes:

- In the 63-character set (NOS/BE), Display Code 00 has no character, and 63 is the colon (:). In the 64-character set (NOS), 00 is the colon (:), and 63 is the percent (%).
- (2) On 7-track tape, this becomes zero (display 33).
- (3) Alternate punches.
- (4) Avoid a whole word of semicolons, which is a negative zero and is treated as an end-of-record.
- (5) On some terminals, this is transmitted as a binary zero.
 For these terminals, avoid putting this punch in columns 9-10, 19-20, ..., 79-80, as each will be interpreted as a zero-byte terminator.
- (6) When ASCII and EBCDIC tapes are read and converted to Display Code, lower case letters are folded into upper case. A number of other codes are also folded.

Page B-1

•

***** Appendix B *****

*** Cray UNICOS Commands ***

This appendix is reserved for a description of Cray UNICOS commands. It will be expanded when UNICOS is available at DTRC.

90/09/30 Rev0 Page C-1

**** ***** Appendix C

*** *** Cray COS JCL Commands

Cray COS JCL commands have the following general syntax:

verb sep1 param1 sep2 param2 ... sepn paramn term comments

is the name of the routine to be executed. It consists of an verb alphabetic character (A-Z, a-z, \$, %, \emptyset) followed by 0-6 alphanumeric characters for system, local dataset name and system dataset name verbs; or 1-8 alphanumeric characters for library-defined verbs.

are separators and include: sepi - VERB.parameter. ,

- VERB(parameter).
- VERB,parameter. <-- use period if comma) - VERB(parameter) <-- use right paren if left paren
 - VERB(parameter, parameter)
- = - VERB(keyword=value)
- VERB(keyword=value1:value2) :
- ۸ - VERB(...parameters...^ <-- statement continued <-- on another line parameters)
- '...' VERB(keyword='string')

(...) - VERB(keyword=(value:value))

- parami are parameters, which may be positional or keyword. Positional parameters have one of the following formats: value value1:value2:...:valuen Keyword parameters have one of the following formats: keyword keyword=value
 - keyword=valuel:value2:...:valuen
- is the statement terminator. It is either a period term VERB. VERB, parameters. or a right parenthesis VERB(parameters)

comments follow the terminator.

Page C-2

Rev0 90/09/30

*** Strings ***

The following string representations are used in this appendix:

aa...a 1 or more alphabetic characters

axx...x 1 or more alphanumeric characters, the first alphabetic

xxx...x 1 or more alphanumeric characters

nnn...n 1 or more decimal (unless otherwise stated) digits

*** Some Common Parameters ***

The following parameters are used in many JCL commands. If they have a different meaning or a special condition, it will be mentioned in the individual description.

AM=mode Alternate User Access Mode (see PAM=)

| DC=dc | Disposition code IN - input queue of desti MT - magnetic tape at jok PR - print at job origin SC - scratch the dataset ST - stage to mainframe (mainframe) | o origin mainframe |
|-------|---|--|
| DF=df | Dataset format (blocking; BB - binary blocked | <pre>front-end conversion) (no reblocking, no conversion;</pre> |
| | - | for graphics output) |
| | BD - binary deblocked | (same as TR) |
| | CB - character blocked | (front-end converts to ASCII (VAX) or Display Code or ASCII (NOS)) |
| | CD - character deblocked | (front-end converts to ASCII (VAX) or Display Code (NOS)) |
| | TR - transparent | (no deblocking; no conversion; |

for object modules, etc.)

(default: CB)

DN=dn Local dataset name (axxxxx, 7 maximum)

ED=ed Edition number (1-4095)

ERR Suppress error termination messages

EXO=exo Execute option ON - execute-only (cannot be read or PSDUMPed) OFF - not execute-only

I=idn IDN=idn Input dataset name (normal default: \$IN) ID≂uid Additional permanent dataset ID (axxxxxx, 8 maximum) L=1dn Name of dataset to contain the listing (default: \$OUT) M=mn Maintenance control word (axxxxxx, 8 maximum) MF=mf Front-end computer N1 - CDC CYBER 180/860A (NOS) V3 - DEC VAXcluster node DT3 (VMS) (default: front-end of job origin) MSG Suppress normal termination messages NA No abort. If omitted, an error causes the job step to abort. 0=odn ODN-odn Output dataset name (normal default: \$OUT) OWN=owner Owner of the permanent dataset (not needed for your own files) PAM=mode Public Access Mode E - execute only (same effect as EXO=ON) M - maintenance only N - no public access R - read only W - write only Example: PAM=R:W gives read and write permission (default: N) PDN=pdn Permanent dataset name (xxxxxxxxxx, 15 maximum; enclosed in quotes "..." if other than A-Z.0-9) R=rdRead control word (axxxxxx, 8 maximum) Text (up to 240 character) to be passed to the front-end, TEXT='text' enclosed in apostrophes ('...') TID=tid Destination terminal (default: terminal of job origin) UQ Unique access (required to delete or modify a dataset) (default: multiple access) Write control word W=wt (axxxxxx, 8 maximum)

*** Permanent Dataset Utility Shorthand Notation ***

In the permanent dataset utility commands, wildcards may be used in the PDN, PDS, ID, US, and OWN parameters. An asterisk "*" represents any single character; a minus sign "-" represents zero or more characters. They are illustrated with PDN=.

PDN=ABC- all permanent dataset names starting with ABC

PDN= A^{***} all 4-character permanent dataset names starting with A

PDN=-A^{*-} all permanent dataset names containing the letter A followed by one or more other characters

PDN=- all permanent dataset names

PDN=***- all permanent dataset names having 3 or more characters

*** A Word About Continuations ***

If a COS JCL statement is too long to fit on one line, it may be continued by breaking the statement after a parameter, ending the line with a caret (^), and continuing the statement on the next line(s). For example,

FETCH, DN=prog3, SDN=myprog, ^
 TEXT='GET, myprog.CTASK.'.

If a text field (quoted string) is too long, it may be split anywhere by adding an apostrophe (') to close the partial string and a caret to end the first line, and starting the next line with an apostrophe immediately followed by the rest of the string. For example,

> DISPOSE, DN=FT14, SDN=myout14, DC=ST, MF=N1, TEXT='USER, user, pw.'^ 'PURGE, myout14/NA.DEFINE, myout14.CTASK.'.

-or-

DISPOSE, DN=FT14, SDN=myout14, DC=ST, MF=N1, ^ TEXT='USER, user, pw.'^ 'PURGE, myout14/NA.'^ 'DEFINE, myout14.'^ 'CTASK.'. 90/09/30 Rev0

χ

*** *** Summary of Cray JCL Commands The following are Cray JCL statements, except as indicated by: (DTRC - x)A command, procedure or program added at DTRC. Unless otherwise noted, these are accessed by: ACCESS, DN=x, OWN=PUBLIC. LIBRARY, DN=x:*. name,.... x is one of: PROCLIB, UTILITY. Entire line is a comment. * <comments> Syntax: Similar commands: NOS: COMMENT; * VMS: 1 * This is a comment ---Examples: Make a permanent dataset local. ACCESS Syntax: ACCESS, DN=dn, PDN=pdn, ID=uid, ED=ed, R=rd, W=wt, M=mn, UQ, NA, ERR, MSG, OWN=owner. Parameters: PDN=pdn - If omitted, dn is used. - required to read the dataset if R= on R=rdSAVE W=wt - required for ADJUST if W= on SAVE M=mn - required to DELETE the dataset if M= on SAVE Similar commands: NOS: ATTACH; GET VMS: no local file concept Examples: ACCESS, DN=mylocal, PDN=mypermfile. ACCESS, DN=mylc1, PDN=yourpermfile, OWN=yourid. ACCESS, DN=myfile, UQ. DELETE, DN=myfile. ACCOUNT Validate the user. Follows the JOB statement or, is the first interactive statement.

> Syntax: ACCOUNT, AC=ac, US=us, UPW=upw, NUPW=nupw.

- Account number (required)

(10 digits or "S" + 9 digits)

Parameters: AC=ac

US=us

- Username (your 4-character User Initials) UPW=upw - User password (required) NUPW=nupw - New user password Remarks: This must be the first statement of an interactive session. When entered via CDC NOS ICF, US= may be omitted because it is supplied automatically. When entered via the DEC VMS Cray Station, US= may be omitted if you entered it in upper case in response to the CRAY USERNAME: prompt. See also: JOB; page 1-2-2; Appendix D: CNEWPW Similar commands: NOS: CHARGE VMS: no user-specified charging Examples: ACCOUNT, AC=1234567890, US=xxxx, UPW=mypass. ACCOUNT, AC=1234567890, US=xxxx, UPW=mypass,NUPW=nupass. ACQUIRE Get a front-end dataset and make it local and permanent. Syntax: ACQUIRE, DN=dn, PDN=pdn, AC=ac, ID=uid, ED=ed, RT=rt, R=rd, W=wt, M≈mn, UQ, MF=mf, TEXT='text', DF=df, OWN=ov, PAM=mode, ERR, MSG. Parameters: AC=ac - acquisition code IN input dataset IT - intertask communication ST - dataset staged from front end (MF=)(default: ST) - (defaults: 1 ED=ed (permanent dataset does not exist) highest (permanent dataset exists)) RT=rt - retention period (1-4095 days) (default: 45) Remarks: If the dataset is permanent, ACQUIRE is the same as ACCESS. If not, then it is the same as FETCH, SAVE, ACCESS. See also: FETCH, MSFETCH

90/09/50 Rev0 Page C-7 Similar commands: NOS: ATTACH; GET VMS: HFT FETCH Examples: ACQUIRE, DN=myfile, PDN=myfile, TEXT='myfile.FOR'. ADJUST Redefine size of a permanent dataset. Syntax: ADJUST, DN=dn, NA, ERR, MSG. Permissions required: write; UQ on ACCESS Remarks: ADJUST attempts to close the file. Subsequent references in the same job must reopen it and begin at BOD. Similar commands: NOS: APPEND VMS: lengthened automatically; cannot be shortened Examples: ADJUST, DN=myfile, NA. ALTACN Validate an alternate account number for permanent files. Syntax: ALTACN, AC=ac. Parameters: ac - the alternate account number Remarks: ALTACN validates the supplied Job Order Number. To use the validated number, specify the ACN parameter on the SAVE or MODIFY command. See also: MODIFY, SAVE Similar commands: NOS: CHANGE Examples: ALTACN, AC=1222233344. <-- define the number SAVE, DN=newfy1, ACN. <-- use the number ACCESS, DN=oldfy1, PDN=myoldfy1, UQ, MODIFY, DN=oldfyl, ACN. <-- change the number ASSIGN Create a local dataset and assign dataset characteristics. Syntax: ASSIGN, DN=dn, LM=1m, A=alias, BS=bs, U. Parameters: LM= - maximum number of 512-word blocks in the dataset (maximum: 296000; default: 40000)

A= - alternate unit name

Page C-8

- BS= octal number of 512-word blocks for the I/O buffer (default: 10 octal)
- U unblocked dataset (default: blocked)
- Remarks: See COS Reference Manual for additional parameters.

At system initiation, ASSIGN,DN=\$IN,A=FT05. ASSIGN,DN=\$OUT,A=FT06. are performed automatically. You may reassign them at any time.

A Fortran OPEN will not recognize an ASSIGNed dataset.

Similar commands: NOS, VMS: ASSIGN

Examples: ASSIGN, DN=myinput, A=FT11. ^-- Fortran program reading from unit 11 will read file MYINPUT instead

AUDIT Report on permanent datasets.

Syntax: AUDIT,L=ldn,PDN=pdn,ID=uid,OWN=own,ACN=acn, LO=opt:...:opt,SZ=dsz,ACC=opt:opt, X=mm/dd/yy:'hh:mm:ss', TCR=mm/dd/yy:'hh:mm:ss', TLA=mm/dd/yy:'hh:mm:ss', TLM=mm/dd/yy:'hh:mm:ss'.

Parameters: L= - list dataset name (default: \$OUT) PDN= - name of permanent dataset(s) to be listed ID= - list datasets with this ID ID - list datasets with null ID OWN= - list datasets with this ownership value ACN= - list datasets with this account number

Page C-9

L0≖ - list options: S - short list (PDN, ID, ED; 2 per line) (may not be mixed with other options) A - access tracking (owner name, count, time of last and first accesses) B - backup info (backup volume name, etc.) L - long list (PDN, ID, ED, size (words), retention time, access count, track access flag, public access mode (PAM), creation, last access, last dump time, device name, preferred residency (PR), current residency (CR). (default in batch if no LO) N - notes list P - permit list (permitted owner name, access mode, access count, time of last access, time of permit creation) R - retired dataset list (same as L, but only retired datasets) T - text list X - extended long list (L plus number of blocks and words allocated) - list datasets >= this size (in words) SZ= ACC= - access option parameters AM - those datasets belonging to OWN that you are allowed to see PAM - those datasets belonging to OWN having any form of public access (R:W:M:E) Х= - list datasets expired as of this date Х - list datasets expired as of now TCR= - list datasets created since this date TCR - not allowed TCR=mm/dd/yy is sufficient TLA= - list datasets not accessed since this date TLA - not allowed TLA=mm/dd/yy is sufficient TLM= - list datasets modified since this date TLM - not allowed TLM=mm/dd/yy is sufficient Similar commands: NOS: CATLIST VMS: DIRECTORY; MSSAUDIT

AUDPL

| Examples: | AUDIT,LO=S< short auditAUDIT,LO=P< audit showing who can and has accessed the datasetsAUDIT,LO=L:P:N< long audit, permitted users and notesAUDIT,LO=L< long auditAUDIT,OWN=PUBLIC.< list public files |
|--------------|---|
| Audit an UPI | DATE program library (PL). |
| Syntax: | AUDPL,P=pdn,I=idn,L=ldn,M=mdn, [*] =m,/=c,DW=dw, LW=1w,JU=ju,DK=1ist,PM=list,LO=string, CM,NA,NR. |
| Parameters: | P I L * / NR - see UPDATE |
| | M= - Modifications dataset name (will contain reconstructed modification sets) (default: \$MODS) |
| | M=O - No modifications output |
| | DW= - Data width (number of characters written per line to M dataset) (default: up to DW value on UPDATE stmt) |
| | LW= - Listing width (number of characters written per line to L dataset (Values: divided into pages: 80, 132; continuous listing: C80, C132) (default: 132, divided into pages) |
| | <pre>JU= - Justification N - identifier name left-justified; sequence number right-justified; no period between L - entire sequence field left-justified with period between (default: identified name right-justified; sequence number preceded by a period and left-justfied)</pre> |
| | DK=dk1:dk2::dkn (1) DK='dk1,dk2,,dkj.dkk,,dkn' (2) - Decks for A, C, D, H, I options and PM parameter (For (1): up to 100 decks; for (2): separate single decks with commas, and ranges of decks with periods) (Maximum string length: 96 characters) (default: options apply to all decks) DK - By itself is invalid |

BLOCK

(1)PM=idl:id2:...:idn PM='idl,id2,...,idj.idk,...,idn' (2)- Pulled modification sets (reconstructs modification sets for the listed identifiers for the decks listed in DK) (Syntax: same as for DK=) PM - By itself is invalid LO=string - Listing options for ldn Text listing (for DK= decks, if specified) A - active lines C - conditional text directives (subset of option D) D - compile dataset generation directives (subset of option A) H - modification histories I - inactive lines Summary options (for the entire PL) K - deck line counts L - identifier list M - modification set cross-reference N - identifier list in ASCII order 0 - overlapping modification set list P - short summary of the PL S - status of modification set X - common deck cross-reference CM - Copy modifications (reconstructed modification sets) to 1dn and mdn NR - Do not rewind modifications or binary identifier list datasets at start or end of AUDPL Similar commands: NOS: UPDATE VMS: CMS; LIBRARIAN; INCLUDE (in Fortran) AUDPL, P=myp1, LO=P. Examples: * * * * * AUDPL, P=myp1, PM=mod2a:mod3c:example, LO=AIKLMNOPSX. COPYF, I=\$MODS. Convert an unblocked dataset to a blocked dataset. Syntax: BLOCK, DN=1dn, BLKSIZE=size. (1)(2) BLOCK, I=idn, O=odn, BLKSIZE=size. Parameters: DN= - the dataset to be replaced (using an intermediate dataset \$UNBLK) (1dn is rewound before and after)

Page C-12

- BLKSIZE= record length in 64-bit words (non-foreign datasets only) ((2) - not permitted if previously assigned as foreign; record length and type are taken from the input ASSIGN)
- I= the unblocked input dataset (idn is not rewound before the copy)
- 0= the blocked output dataset (if previously opened (ASSIGN), odn is not rewound before; otherwise, odn is created)
- Remarks: For foreign datasets, the record length and type are taken from the ASSIGN.

BLOCK is intended primarily for postprocessing datasets created by or for certain stations.

Examples: BLOCK, DN=myfile. ^-- Replace MYFILE with blocked copy of itself

> BLOCK, I=myunblk, O=myblk. ^-- Copy unblocked file MYUNBLK as blocked file MYBLK

BUILD Generate and maintain library datasets.

Syntax: BUILD, I=idn, L=1dn, OBL=odn, B=bdn, NBL=ndn, SORT, NODIR, REPLACE.

| Parameters: | I | Directive dataset name (default: \$IN) Same as I=\$IN No directives |
|-------------|-------------------|--|
| | L=ldn L | List dataset name (default: \$OUT) Same as L=\$OUT |
| | OBL | Old object library dataset name (default: \$OBL) Same as OBL=\$OBL No old binary library |
| | B=bdn B B=0 | Dataset with new object modules (default: \$BLD) Same as B=\$BLD No modules to be added |

90/09/30 Rev0

Page C-13

NBL=ndn - Output new object library dataset name (default: SNBL) NBL - Same as NBL=SNBL NBL=0 - No output written SORT - modules are to be output in alphabetical order (default: written in the order they were first read) NODIR - Do not append the directory to the output dataset - use to retrieve relocatables (default: append the directory) REPLACE - Modules in the new library are replaced and in the same order as in the old library (default: new modules follow the unreplaced modules in the new library) Directives: see page 3-5-1. See also: Section 3-5 Similar commands: NOS: LIBEDIT VMS: LIBRARIAN Examples: BUILD, OBL=0, I=0. SAVE, DN=\$NBL, PDN=mylib. ^-- create a new library from \$BLD _ _ _ _ _ ACCESS, DN=\$OBL, PDN=mylib. BUILD, I=0. SAVE, DN=\$NBL, PDN=mylib. ^-- add modules from \$BLD to existing library - - - - -ACCESS, DN=mylib1. ACCESS, DN=mylib2. ACCESS, DN=mylib3. BUILD, I, OLB=0, B=0. SAVE, DN=\$NBL, PDN=mylib4. - Directive: FROM mylib1, mylib2, mylib3 ^-- merge several libraries - if duplicate module names, last found is retained (or use rename form, if desired) _ _ _ _ * * ACCESS, DN=\$OBL, pdn=mylib. BUILD, B=0. SAVE, DN=\$NBL, PDN=mylib. - Directive: OMIT badpgm ^-- remove a module from a library

CALL

ACCESS, DN=xyz, PDN=mylib. BUILD, I, OBL=xyz, B=0, NBL=\$BLD, NODIR. - Directive: COPY myprog ^-- extract module for loading Read control statements from the first file of another dataset or transfer control to a procedure. Syntax: CALL, DN=dn. <-- read from another file CALL, DN=dn, CNS. <-- call a procedure</pre> Parameters: DN=dn - the dataset containing the statements or procedure (rewound before use) CNS - Crack Next Statement - the first statement in "dn" is the procedure header; the statement following the CALL is treated as the invocation of the procedure See also: Section 3-3 Similar commands: NOS: BEGIN VMS: @name Examples: Without CNS: If the first file of dataset XYZ contains: ACCESS, DN=INFYL, PDN=MYFILE. ACCESS, DN=FILE1, PDN=MYDATA. Then CALL, DN=XYZ. will access both datasets. This might be useful if you have several jobs using the same files, or if you have the same processing to be done by many jobs. With CNS: If the first file of dataset XYZ contains: G, FILE, DATA. ACCESS, DN=INFYL, PDN=&FILE. ACCESS, DN=FILE1, PDN=&DATA. Then CALL, DN=XYZ, CNS. *, MYFILE, MYDATA will access the datasets MYFILE and MYDATA. Note that PROC and ENDPROC statements and the procedure name (G) are not used.

90/09/30 Rev0

"call by name" Execute a program by its local file name. Syntax: dn. dn, parameters. Parameters: depends upon the local file being executed Similar commands: NOS: LGO or an 1fn VMS: \$ name :== \$ dir:name \$ name Examples: ACCESS, DN=myobj. myobj. CFT Compile a Fortran source program. Syntax CFT, I=idn, L=ldn, B=bdn, C=cdn, E=m, EDN=edn, OPT=option, MAXBLOCK=mb, INT=i1, ALLOC=alloc, ON=string,OFF=string,TRUNC=nn,AIDS=aids, CPU=cpu:hdw,UNROLL=r,LOOPMARK[=1mmsgs], DEBUG, SAVEALL, ANSI. Parameters: I= - Input dataset name (default: \$IN) L= - Listable output (default: \$OUT) L=0 - List only fatal errors - Binary load module dataset name B= (default: \$BLD) B=0 - No binary load modules C= - pseudo-CAL output dataset name (default: no dataset) Ē= - Highest level of messages to be suppressed 1 - comment2 - note3 - caution4 - warning 5 - error(default: 3) EDN≃ - Alternate error listing dataset (default: no dataset)

```
ON=
           - Options to be enabled
             (default: C E L P Q R S T U V)
OFF=
           - Options to be disabled
             (default: A B D F G H I J N O W X Z)
              A - abort if errors
              B - list sequence number of code
                  generation block
              C - list common block names and
                  lengths
              D - list DO-loop table
              E - recognize compiler directives
              F - FLOWTRACE
              G - list generated code (use only
                  if requested by User Services)
              H - list only first statement of
                  each program unit
              I - generate label symbol table
              J = one-trip DO-loops
              L - recognize listing control
                  statements
              M - ignored
              N - put null symbols in symbol table
              0 - identify out-of-bound array
                  references
              P - allows double precision
              Q - abort on 100 fatal errors
              R - round multiply results
              S - list source code
              T - list symbol table
              U - enable recognition of INTEGER*2
                  declarations
              V - vectorize inner DO-loops
              W - do not use
              X - include cross-reference
              Y - ignored
              Z - put DEBUG symbol table on $BLD
TRUNC=
           - number of bits to be truncated
             (default: 0; maximum: 47)
AIDS=
           - number of vectorization inhibition
             messages
              LOOPNONE - no messages
              LOOPPART - maximum of 3 per inner
                         loop; 100 per compilation
              LOOPALL - all messages
             (default: LOOPPART)
```

OPT= - options (no more than one from each of the following groups; OPT=opt:opt:...): . constant increment integer optimization: NOZEROINC - no incrementation by zero value variables ZEROINC - incrementation by zero value variables (default: NOZEROINC) . optimization for 1-line DO-loop replacement with \$SCILIB call: SAFEDOREP - no replacement if DO-loop has potential dependencies or equivalenced variables FULLDOREP - alway replace NODOREP - never replace (default: SAFEDOREP) . move invariant code outside of D0-100p: INVMOV - enable NOINVMOV - disable (default: INVMOV) . instructions moving over a branch instruction: UNSAFEIF - enable SAFEIF - disable (default: SAFEIF) . bottom loading of scalar loops: BL - enable NOBL - disable (default: BL) . B and T register allocation: BTREG - allocate maximum of 24 scalars to T regs NOBTREG - allocate to memory (default: NOBTREG) . compilation of loops with specific ambiguous dependencies in vector and scalar versions: CVL - enable NOCVL - disable (default: enabled) . update scalar temporaries in DO-loops: KEEPTEMP - enable KILLTEMP - disable (default: enable)

Page C-18

MAXBLOCK= - number of words in a block of code to optimize or vectorize MAXBLOCK=1 - disable (default: 2310) INT= - integer lengths 64 - full 64-bit integers 24 - short 24-bit integers (default: 64) ALLOC= - static memory allocation STATIC - all memory STACK - read-only constants and DATA, SAVE and common block entities HEAP - deferred implementation (default: STATIC) CPU= - mainframe type and hardware characteristics for running generated code cpu type: CRAY-XMP - 1, 2 or 4 processors CRAY-X1 - single-processor CRAY-X2 - dual-processor CRAY-X4 - quad-processor (default: compiling machine) hardware characteristics: [NO] EMA - extended memory . [NO]CI - compressed index [NO]GS - gather/scatter [NO]CIGS - compressed index gather/ scatter [NO] VPOP - vector popcount functional unit [NO] AVL - two vector logical functional units [NO] BDM - bidirectional memory UNROLL= - iteration count for unrolling inner DO-loops (range: 0 <= r <= 9) (default: 3) UNROLL=0 - turn off unrolling LOOPMARK= - draw DO-loop brackets in source listing MSGS - reasons for not vectorizing NOMSGS - no messages (default: NOMSGS) LOOPMARK - same as LOOPMARK=NOMSGS

90/09/30 Rev0

- DEBUG put sequence number labels in Debug Symbol Table (forces ON=IW and MAXBLOCK=1) (default: debugging turned off)
- SAVEALL allocate user variables to static storage; compiler-generated variables to B or T registers
- ANSI flag non-ANSI usage
- Remarks: CFT compiles faster than CFT77, but executes more slowly.

Production programs should be compiled using CFT77 and the resulting \$BLD file saved.

See also: CFT77

Similar commands: NOS: FTN5 VMS: FORTRAN

- Examples: CFT. CFT,I=\$CPL. <-- from UPDATE CFT,LOOPMARK=MSGS. CFT,B=myobj.
- CFT77 Compile a Fortran 77 source program.

Syntax CFT77, I=idn, L=1dn, B=bdn, C=cdn, E=m, OPT=option, INTEGER=i1, ALLOC=alloc, ON=string, OFF=string, TRUNC=nn, CPU=cpu:hdw, DEBUG, LIST, STANDARD.

Parameters: I L B C ALLOC TRUNC CPU DEBUG - same as CFT

E= - same as CFT, except E=5 not allowed OPT= - at most one from each of the following groups (OPT=opt:opt): . optimization: - attempt full optimization FULL - no optimization OFF (fast compile) NOVECT - scalar optimization only (default: FULL) . constant increment integer optimization: NOZEROINC - no incrementation by zero-value variables ZEROINC - incrementation by zero-value variables (default: NOZEROINC)

INTEGER= - integer length 64 - full 64-bit integers 46 - short 46-bit integers (default: 46) ON =- M - enable the loopmark option (same as CFT,LOOPMARK=MSGS) (default: P Q R) - (default: A F G H J M O S X Z) OFF= LIST - full compilation listing (sets ON=CGSX) DO NOT USE -- specify ON=CMSX instead STANDARD - flag non-standard Fortran 77 usage Remarks: CFT77 compiles much more slowly than CFT, but may execute faster. OPT=OFF does not vectorize and will, therefore, run slower. Production programs should be compiled using CFT77 and the resulting \$BLD file saved. See also: CFT Similar commands: NOS: FTN5 VMS: FORTRAN Examples: **CFT77.** CFT77, I=\$CPL. <-- from UPDATE CFT77,B=myobj,ON=M. CHARGES Report on job resources. Syntax: CHARGES, L=1dn, MSG=msgopt, SR=options. Parameters: MSG - controls the display of messages in the system log ON - output to \$LOG and \$SYSLOG OFF - output not displayed in log

SR - control display of system resources - CPU, I/O wait, and CPU wait times CPU since start of job DS - permanent dataset statistics (synonym: DISK) - FSS (buffer memory in IOS) usage FSU GRU - generic resource usage JNU - job name and user number JSQ - job sequence number MM - job size (memory) statistics (synonym: MEMORY) MULTI - % of time spent in each CPU - number of blocks received from/ NBF queued to a front end (synonym: FE) '

90/09/30 Rev0

RDM - job and permanent data usage and limits TASK - CPU, I/O wait, and CPU wait times broken down by task; and totals for job WT - time spent waiting in input queue (synonym: QWAIT) (default: all statistics) Remarks: CHARGES is invoked automatically at job end. Similar commands: NOS: ENQUIRE VMS: CHARGES; ^T Examples: CHARGES, SR=DS:MM:TASK COMPARE Compare two datasets. Syntax: COMPARE, A=adn, B=bdn, L=1dn, DF=df, ME=maxe, CP=cpn, CS=csn, {CW=cw { CW=cw1:cw2} , ABORT=ac. Parameters: A= - input dataset names - error if adn=bdn B= L= - name of dataset for list of differences (default: \$OUT; may not be same as adn or bdn) DF= - input dataset format B - binary - datasets compared logically with difference listed in octal T - text - differences printed as text (default: T) ME= - maximum number of differences to be printed (default: 100) CP= - amount of context printed, that is, the number of records on either side of a difference to be printed (applies only to DF=T) (default: 0) CS= - amount of context to be scanned, that is, the number of records on either side of a discrepancy to be scanned (applies only to DF=T) (default: 0)

CW= - compare width - either compare columns 1 through cw or columns cwl through cw2 (default: CW=1:133) ABORT= - abort the job step after ac or more differences have been found ABORT - same as ABORT=1 (default: 1) Similar commands: NOS: VERIFY; VFYLIB **VMS: DIFFERENCES** Examples: ACCESS, DN=one, PDN=myfile1. ACCESS, DN=two, PDN=myfile2. COMPARE, A=one, B=two, CS=5. COPYD Copy blocked datasets. Syntax: COPYD, I=idn, O=odn, S=m. Parameters: S=m - shift count (number of ASCII blanks to be inserted at the start of each line) (maximum: 132) S - same as S≈1 (default: 0) COPYF; COPYNF; COPYR; COPYU See also: Similar commands: NOS: COPY; COPYSBF VMS: COPY COPYD, I=myprog, S=25. Examples: ^-- copy shifted file to \$OUT (source program centered on wide paper) Copy blocked files. COPYF COPYF, I=idn, O=odn, NF=nf, S=m. Syntax: Parameters: I O S - same as COPYD NF=nf - decimal number of files to copy NF - copy through EOD (default: 1) Remarks: After the copy, both datasets are positioned after the EOF for the last file copied. If BFI=OFF is specified on the ASSIGN, compressed blanks are expanded.

See also: COPYD; COPYNF; COPYR; COPYU

90/09/30 Rev0

Page C-23

Similar commands: NOS: COPY; COPYBF; COPYCF; COPYSBF VMS: COPY

Examples: COPYF, I=FT02. <-- print Fortran unit 2 on \$0UT.

COPYNF Copy files from one blocked dataset to another.

Syntax: COPYNF, I=idn, O=odn, NF=n.

Parameters: I 0 - same as COPYD

NF=n - deciaml number of files to copy. NF - copy through EOD (default: 1)

Remarks: After the copy, the input dataset is positioned after the EOF for the last file copied; the output dataset is after the EOF of the last record copied.

See also: COPYD; COPYF; COPYR; COPYU

Similar commands: NOS: COPYBF; COPYCF

Examples: COPYNF, I=mydata, O=files, NF=3. ^-- copy 3 files from dataset MYDATA to dataset FILES

COPYR Copy blocked records. Syntax: COPYR, I=idn, O=odn, NR=nr, S=m. Parameters: I O S - same as COPYD NR=nr - decimal number of records to copy NR - copy through EOF (default: 1) Remarks: After the copy, both datasets are positioned at the end of the last record copied. If BFI=OFF is specified on the ASSIGN, compressed blanks are expanded. See also: COPYD; COPYF; COPYNF; COPYU

Examples: COPYR, I=myfile, O=recs, NR=342.

Rev0 90/09/30

COPYU Copy unblocked datasets. Syntax: COPYU, I=idn, O=odn, NS=ns. Parameters: I 0 - same as COPYD NS=ns - number of sectors to copy NS - copy through EOD (default: 1) COPYD; COPYF; COPYNF; COPYR See also: Examples: COPYU, I=unfy11, O=unfy12, NS. &DATA Defines the beginning of data within a procedure. &DATA, dn. Syntax: Parameters: dn - the name of the dataset to contain the data which follows this statement Remarks: All lines following an &DATA up to the next &DATA or ENDPROC are written to the specified dataset. Similar commands: NOS: .DATA VMS: OPEN, WRITE, CLOSE Examples: PROC, MYPROC. . . . ENDPROC. &DATA, IN1. 1.73, 2.6, 4 4.62, 9.7, 6 0,0,0 &DATA, IN2. 06Test01 12Ship 472-396X DDA Dynamic Dump Analyzer (selectively examine the contents of a program memory dump). DDA, I=idn, S=sdn, L=odn, DUMP=ddn, LOG=1dn, ECHO=edn. Syntax: - the directives to be executed Parameters: I= (default: SIN) S= - symbolic dataset name (default: \$DEBUG)

> L= - output listing (default: \$0UT)

90/09/30 Rev0

Page C-25

- LOG= the dataset to receive a copy of all input to and output from the debugger (default: \$DBLOG)
- ECHO= the dataset to receive a copy of all input to the debugger (default: \$DBECHO)
- Remarks: Like DEBUG, DDA interprets the contents of a program memory dump created during abort exit processing. Unlike DEBUG, you can give directives to dynamically select the information to display.
- Directives: See SR-0311, COS Symbolic Debugging Package Reference Manual (formerly SR-0112).

See also: DEBUG

Similar commands: NOS: FTN5,DB=PMD VMS: FORTRAN/DEBUG

Examples: See DUMPJOB

DEBUG Interpret a dump.

Syntax: DEBUG, S=sdn, L=ldn, DUMP=ddn, CALLS=n, TASKS, SYMS=sym[:sym], NOTSYMS=nsym[:nsym], MAXDIM=dim, BLOCKS=blk[:blk], NOTBLKS=nblk[:nblk], RPTBLKS, PAGES=np.

Parameters: S= - Debug symbolic tables (default: \$DEBUG)

> L= - Listable output (default: \$0UT)

- DUMP= Dump dataset name (default: \$DUMP)
- CALLS= Number of routine levels to display (default: 50)
- TASKS Trace back through all existing tasks (default: only through tasks running when dump taken)

DELETE

SYMS= - List of symbols to be displayed (Maximum: 20 symbols) (default: all symbols) NOTSYMS= - List of symbols to be skipped (Maximum: 20 symbols) (default: all symbols displayed) MAXDIM= - Maximum number of each dimension to be displayed (default: 20:5:2:1:1:1:1) BLOCKS= - List of common blocks to include (Maximum: 20 symbols) BLOCKS - Include all common blocks NOTBLKS= - List of common blocks to exclude (overrides BLOCKS) (Maximum: 20 symbols) NOTBLKS - Exclude all but subprogram block RPTBLKS - Repeat blocks (display with each subprogram (default: display once) PAGES= - Page limit (default: 70) Similar commands: NOS: FTN5, DB=PMD VMS: FORTRAN/DEBUG Examples: See DUMPJOB. Remove a permanent dataset. Syntax: DELETE, DN=dn, NA, ERR, MSG, PARTIAL. DELETE, PDN=pdn, ID=uid, OWN=owner, ED=ed, M=mn, NA, ERR, MSG. Parameters: PARTIAL - delete the contents of the file, but not the information about the file ED=ed - edition number (1-4095)unsigned - specific edition +n - delete n highest editions -n - keep n highest editions ALL - all editions (default: highest edition) Remarks: The first form is used if the permanent file has already been ACCESSed.

The second form does not ACCESS the file.

See also: Appendic C: CDELETE Similar commands: VMS: CREATE a new version, PURGE/KEEP=1; DELETE: PURGE Examples: ACCESS, myfile, UQ. DELETE, DN=myfile, PARTIAL. - - - - - -DELETE, PDN=myfile, ALL. _ _ _ _ _ DELETE, PDN=A**. ^-- delete all datasets with 3-character names starting with "A" DISPOSE Stage a dataset to the front-end; release a local dataset; change disposition characteristics.

Syntax: DISPOSE, DN=dn, SDN=sdn, DC=dc, MF=mf, SF=sf, ID=uid, TID=tid, R=rd, W=wt, M=mn, TEXT='text', DF=df, WAIT'NOWAIT, DEFER, NRLS.

Parameters: DN=dn - required

SDN=sdn - staged dataset name (1-15 characters) (default: dn; required for CYBER 860)

- DC=dc to 860: DC=ST is required to VAX: DC=PR with TEXT='any' makes a file with Fortran carriage control; DC=ST (with TEXT='any') makes a file with carriage return carriage control
- SF=sf special forms (1-8 alphanumeric characters) (default: no special forms)
- DF=df TR or CB or BB (default: CB)
- WAIT wait or don't wait until dataset has NOWAIT been staged to the front-end (default: NOWAIT)
- DEFER disposition occurs at end-of-job or when the dataset is RELEASEd
- NRLS after disposition, the dataset remains local (use WAIT)

See also: MSSTORE

DS

Rev0 90/09/30

. . .

Similar commands: NOS: ROUTE VMS: FICHE (DTRC); PRINT; XEROX (DTRC) Examples: DISPOSE, DN=out1, DC=PR. ^-- to VAX (assumed job origin) ----DISPOSE, DN=out2, SDN=mymss, MF=N1, DC=ST, ^ TEXT='USER,user,pw.'^ 'PURGE, mymss/NA.'^ 'DEFINE, mymss.'^ 'CTASK.',WAIT. ^-- send to MSS DISPOSE, DN=out3, MF=V3, ^ TEXT='myvax.dat',WAIT. ^-- send to VAXcluster = = = = DISPOSE, DN=DISPLOT, DC=ST, DF=BB, TEXT='plot.dat', ^ WAIT. ^-- DISSPLA output file to VAX for post processing List local datasets. Syntax: DS. Remarks: The information displayed includes alis, size, position (e.g., EOF), last operation, and open status. Similar commands: NOS: ENQUIRE, F Examples: DS. DSDUMP Dump a dataset in octal or hexadecimal. Syntax: DSDUMP, I=idn, O=odn, DF=df, IW=n, NW=n, IR=n, NR=n, IF=n,NF=n,IS=n,NS=n,Z,DB=db,DSZ=sz. Parameters: I= - (synonym: DN=idn) - dataset to receive the dump 0≖ (default: SOUT) DF= - dataset format B - blocked U - unblocked (default: B) IW= - decimal/octal number of the initial word for each record/sector (defaults: 0 (Z specified); 1 (Z omitted))

```
NW= - decimal/octal number of words to dump
                    (default: 1)
                  - through end of record/sector
             NW
             IR= - decimal/octal number of the initial record
                    for each input file - only if DF=B
                    (defaults: 0 (Z specified);
                               1 (Z omitted))
             NR= - decimal/octal number of records per file
                    to dump - only if DF=B
                    (default: 1)
             NR
                  - all records in each file
             IF= - decimal/octal number of the initial file in
                    idn - only if DF=B
                    (defaults: 0 (Z specified);
                               1 (Z omitted))
             NF= - decimal/octal number of files to dump -
                    only if DF=B
                    (default: 1)
             NF=0 - all files in the dataset
             IS= - decimal/octal number of the initial
                    sector - only if DF=U
                    (defaults: 0 (Z specified);
                               1 (Z omitted))
             NS= - decimal/octal number of sectors to dump -
                    only if DF=U
                    (default: 1)
             Ζ
                  - the zero-base for the initial-value
                    parameters (IW, IR, IF, IS)
                          - each Ix is relative to 0;
                     Ζ
                            output refers to word, record,
                            file, and sector numbers start
                            at 0
                              DSDUMP,..., IW=4096. is same as
                              DSDUMP, \ldots, Z, IW=4095.
                     no Z - each Ix is relative to 1
                    (does not affect Nx parameters)
             DB= - numeric base for displaying the data words
                     OCTAL or 0 - octal
                     HEX or H - hexadecimal
             DSZ= - size of data items to dump
                     WORD or W - words
                                          (64 bits)
                     PARCEL or P - parcels (16 bits)
                    (default: WORD)
Similar commands: NOS: TDUMP
Examples:
             DSDUMP, I=myfile, NW=25, NR=5, DB=H.
                          ^-- hexadecimal dump of first 25
                              words of first 5 records of
                              MYFILE
```

DUMP Display job information previously captured by DUMPJOB. DUMP, I=idn, O=odn, FWA=fwa, LWA=1wa, JTA, NXP, V, DSP, Syntax: FORMAT=f.CENTER. Parameters: I= - dataset containing the memory image (default: \$DUMP) - first word address to dump FWA= (default: word 0 of Job Communication Block (JCB)) LWA= - last word address to dump (default: 200 of JCB) - the limit address LWA LWA=0 - no memory JTA - dump Job Table Area (default: no JTA dump) NXP - dump No Exchange Package, B, T, cluster, and semaphore registers (default: these are dumped; NXP overrides V if both specified) ۷ - dump vector registers (default: do not dump vector registers) DSP - dump Logical File Tables (LFTs) and Dataset Parameter Tables (DSPs) (default: do not dump LFTs and DSPs) FORMAT= - format for dumping FWA through LWA D - data - decimal integer and ASCII G - data - floating-point or exponential and ASCII I - instr - CAL mnemonics and ASCII M - data - each 16-bit parcel displayed as 1 hex and 4 octal digits C - data - octa: integer and ASCII P - data - 16-bit parcel X - data - hex integer and ASCII CENTER - dump 100 (octal) words on each side of P-register address in P format Examples: See DUMPJOB. DUMPJOB Capture job information in dataset \$DUMP for display by DUMP or DEBUG or DDA. Syntax: DUMP JOB. Examples: ... EXIT. DUMPJOB. DUMP,.... -or- DEBUG, BLOCKS,.... -or- DDA,....

90/09/30 Rev0

Page C-31

ECHO Control logfile messages. Syntax: ECHO, ON=class1:...:classm, OFF=class1:...:classn Parameters: ON= - list of classes whose messages are to be written to the log file ("ON" is the same as "ON=ALL") OFF= - list of classes whose messages are NOT to be written to the log file ("OFF" is the same as "OFF=ALL") classi - ABORT - job failure EXPINF - dataset statistics messages JCL - messages in user's JCL PDMERR - PDM errors PDMINF - PDM dataset information - all classes ALL Remarks: The ECHO state after returning from a procedure call is the same as before the call, regardless of any changes made in the procedure. Within a procedure, the ECHO state is that of the caller, unless changed within the procedure. Similar commands: VMS: /LOG on some commands Examples: ECHO, OFF. ELSE See IF. See IF. ELSEIF ENDIF See IF. ENDLOOP See LOOP. ENDPROC See PROC. EXIT On job abort, processing continues with the statement following the EXIT; if no abort, terminate job processing. Syntax: EXIT. Similar commands: NOS: EXIT VMS: ON condition Examples: ... EXIT. DUMP JOB. DUMP. . . .

EXITIF See IF. EXITLOOP See LOOP. FETCH Get a front-end dataset and make it local. FETCH, DN=dn, SDN=sdn, AC=ac, TEXT='text', MF=mf, Svntax: DF=df. - local dataset name Parameters: DN= SDN= - staged dataset name (front-end dataset name) (default: dn) AC= - acquisition code (where the dataset is Lo be acquired) IN - input (job) dataset - use SUBMIT to run the job IT - intertask communication MT - magnetic tape at the front end ST - staged dataset from the front end (default: ST) - mainframe computer identifier MF= N1 - MSS V3 - DT3 (default: front end of job origin) DF= - dataset format (BB, BD, CB, CD, TR) (default: CB) Remarks: FETCH defaults to DF=CB, MSFETCH defaults to DF=TR. See also: **MSFETCH** Similar commands: VMS: HFT FETCH (get an MSS file, DTRC) Examples: FETCH, DN=SOURCE, TEXT='PROG.FOR'. - - - - -FETCH, DN=FT11, DF=TR, ^ TEXT='[ABCD.SUBD1]CRAYBIN.DAT'. ^-- binary data file from a VAX subdirectory of user ABCD FETCH, DN=SORC, SDN=mssname, MF=N1, ^ TEXT='USER, name, pw. ^ 'GET.mssname.CTASK.' ^-- get an indirect MSS (860) file

90/09/30 Rev0 rage C-33 FLODUMP Dump flowtrace table of a program abort. Syntax: FLODUMP, L=1dn. Parameters: L= - dataset to contain the report (default: \$OUT) Examples: . . . EXIT. DUMPJOB. FLODUMP. FTREF Generate Fortran cross-reference. FTREF, I=idn, L=1dn, CB=op, TREE=op, ROOT=root, Syntax: END=end, LEVL=n, DIR=dir, NORDER, MULTI. Parameters: I= - input dataset containing the crossreference table listing and Fortran source program (ON=XS) CB= - global common block cross references PART - routines using a common block FULL - use of common block variables NONE - no output information (default: PART) TREE= - static calling tree PART - entry names, external calls, calling routines, common block names FULL - PART plus static calling tree NONE - no output information (default: PART) ROOT= - if TREE=FULL, this defines the name of the routine to be used as the root of the tree (default: the routine not called by any other routine; if more than one, the first alphabetically) - if TREE=FULL, this defines the name of END= the routine to be used as the end of any branch of a tree (default: complete trees are generated) LEVEL= - if TREE=FULL, this is the maximum length of any branch (default: the entire program) - dataset containing processing directives DIR= (default: no directives)

HOLD

IF

NORDER - list subprograms in input order (default: list in alphabetical order) MULTI - summarize multitasking subroutine usage Directives: The following may be in the DIR= dataset: ROOT - list of modules to be used as roots of separate trees ROOT, md1, md2,..., mdn. SUBSET - list of modules to be processed SUBSET, md1, md2, ..., mdn. (default: all modules) CHKBLK - list of common blocks to be checked for locked variables CHKBLK, b1k1, b1k2,..., b1kn. CHKMOD - list of external calls to be checked for calling from a locked area CHKMOD, mod1, mod2, ..., modn. Similar commands: NOS: FTN5, LO= VMS: FORTRAN /CROSS REFERENCE Specify that dataset release occurs with implicit HOLD. Syntax: HOLD, GRN=grn. Parameters: GRN=grn - generic resource name Remarks: This prevents return of resources to the system and is useful when dataset assignment is done by applications over which the user has no control. See also: NOHOLD Begin a conditional block of code. Syntax: IF(expression) <do if true> ELSEIF(expression) <do if true> ELSE. <do if all other tests fail> ENDIF. EXITIF. <-- exit unconditionally EXITIF(expression) <-- exit if exp is true Parameters: exp - a valid JCL expression

Literal strings, '...', in an IF/ELSEIF Remarks: expression are limited to 8 characters (one machine word). Similar commands: NOS: IF: IFE VMS: IF Examples: ACCESS, DN=MYPROG, NA. IF (PDMST.NE.1) UPDATE (Q=MYPROG) CFT (I=\$CPL,ON=A) NOTE (DN=SLIN, TEXT='ABS=MYPROG') ^-- create input directive file for SEGLDR SEGLDR (I=SLIN) SAVE (DN=MYPROG, NA) EXITIF. EXIT. * *. Error while generating MYPROG * EXIT. ENDIF. MYPROG. ***************** Same as above, but in a procedure, with SEGLDR directives in a data file in the procedure: PROC. DOMYPROG. <-- omit NOTE command . . . ENDPROC. &DATA, SLIN . . . ABS=MYPROG Control access to a job's I/O area (containing the DSP and I/O IOAREA buffers). IOAREA, { LOCK | UNLOCK } Syntax: Parameters: LOCK - the limit address is set to the base of the DSPs, denying direct access to the user's DSP and I/O buffers. When locked, system I/O routines can gain arcess. UNLOCK - the limit address is set to JCFL, allowing access to these areas. Examples: IOAREA, LOCK.

ITEMIZE Report statistics about a library dataset. ITEMIZE, DN=dn, L=1dn, NREW, MF=n, T, BL, E, B, X. Syntax: Parameters: DN= - (default: \$OBL) NREW - no rewind (default: rewind before and after) NF= - number of files to be listed (default: 1) - all files NF - truncate lines after 80 characters Т (if specified, E, B, X may not be used) BL. - burstable listing (each heading is at top of a page (default: page eject only when current page is nearly full) E - list all entry points (binary library datasets only) В - E plus code and common block information (B overrides E) Х - B plus external information (X overrides B) Restrictions: . an UPDATE PL is recognized only if it is the only item in a dataset . standard COS blocked datasets only Similar commands: NOS: ITEMIZE VMS: LIBRARIAN Examples: ITEMIZE, DN=myreloc ITEMIZE, DC=mylib, X. JOB First statement of a job - gives job parameters. JOB, JN=jn, MFL=f1, T=t1, OLM=olm, US=jcn. Syntax: Parameters: JN=jn - job name (1-7 alphanumeric characters) MFL=f1 - maximum field length (decimal) for the job - fl is rounded up to the nearest multiple of 512 words, or the amount needed to load CSP (Control Statement Processor) (default: 768000) MFL - the system maximum (3, 532, 800)

- T=t1 job time limit (decimal seconds) (default: 30; max: 200000) T - the system maximum (~194 days!) NOTE: your job will not run because this exceeds the DTRC maximum!
- OLM=olm maximum size of \$OUT; olm is the number of 512-word blocks (each block holds about 45 lines) (default: 8192; maximum: 65536)
- US=jcn job class (1-7 alphanumeric characters) jcn is one of: NORMAL, DEFER, BUDGET, PZERO, SECURE Job is dropped to a lower class if it doesn't fit the requested job class. (default: NORMAL, if it fits) (see page 3-1-4 for the job class limits and SECURE restrictions)

Remarks: The JOB statement may be continued.

See also: ACCOUNT

Examples: JOB, JN=jobname1. ACCOUNT,.... <rest of job>

JOBCOST (DTRC - UTILITY) Write a summary of the job cost and system usage to \$LOG.

Syntax: JOBCOST

Remarks: A subroutine version is available in DTLIB.

Examples: ACCESS, DN=UTILITY, OWN=PUBLIC. LIBRARY, DN=UTILITY:*. JOBCOST. <-- the cost to this point in job < execute your program > JOBCOST. <-- the cost of running your program

LIBRARY Specify the library dataset search order for control statement verbs.

Syntax: LIBRARY, DN=dn1:dn2:....dn64, V.

Parameters: DN= - up to 64 library names to be searched - an asterisk means add the listed names to the current searchlist

V - list the current library searchlist in the logfile

Similar commands: NOS: LIBRARY; LDSET, LIB= (not subs) Examples: LIBRARY, DN=THISLIB: YOURLIB. ^-- the searchlist contains 2 libraries LIBRARY, DN=THATLIB:*, V. ^-- the searchlist now has 3 libraries and are listed in the logfile LIBRARY,,V. <-- list the current searchlist in the logfile Start of an iterative control statement block. LOOP Syntax: LOOP. . . . EXITLOOP. EXITLOOP(expression) . . . ENDLOOP. Parameters: exp - a valid JCL expression Similar commands: NOS: WHILE Examples: Merge two datasets for 60 records: SET, J1=0. SET, J2=60. LOOP. EXITLOOP (J2.EQ.0) IF(J1.EQ.0)COPYR, I=DSIN1, O=OUTDS. SET.J1=1. ELSE. COPYR, I=DSIN2, O=OUTDS. SET, J1=0. ENDIF. SET, J2=J2-1. ENDLOOP. REWIND, DN=DSIN1:DSIN2:OUTDS. MEMORY Request new field length. Syntax: MEMORY. MEMORY, FL=f1. MEMORY, FL=f1, { USER | AUTO }. Parameters: FL=fl - the decimal number of words of field length; "FL" allocates the job maximum USER - field length is retained until the next request AUTO - field length is reduced automatically at the end of each job step

90/09/30 Rev0

Page C-39

ļ

| | Similar comm | ands: NOS: MFL | | |
|--------|---|---|--|--|
| | Examples: | MEMORY,FL,USER. < get and hold the maximum field length | | |
| | | MEMORY,AUTO. < resume automatic mode (FL reduces after next job step) | | |
| | | MEMORY,FL=32978. < get and hold 32978 words (user mode) | | |
| | | MEMORY, FL=32978, AUTO. | | |
| | | ^ get 32978 words for next job step only | | |
| MODE | DE Set/clear mode flags. | | | |
| | Syntax: | MODE,FI=option,BT=option,EMA=option,AVL=option, ORI=option. | | |
| | Parameters: | option - ENABLE or DISABLE FI - floating-point error interrupts | | |
| | | <pre>(default: ENABLE) BT - bidirectional memory transfers (default: ENABLE)</pre> | | |
| | | EMA - extended memory addressing (default: DISABLE) | | |
| | | AVL - second vector logical function unit (default: DISABLE) | | |
| | | ORI - operand range error interrupt (default: ENABLE) | | |
| | Similar comm | ands: NOS: MODE VMS: ON condition | | |
| MODIFY | Change a permanent dataset's characteristics. | | | |
| | Syntax: | MODIFY,DN=dn,PDN=pdn,ID=uid,ED=ed,RT=rt,R=rd, W=wt,M=mn,NA,ERR,MSG,EXO=exo,PAM=mode,ACN. | | |
| | Parameters: | RT=rt - new retention period RT= - reset to default | | |
| | | ACN - use the alternate account number | | |
| | Remarks: | If the file has control words (M=, R=, W=), they must all be specified in the ACCESS. | | |
| | See also: | ALTACN; NEWCHRG; SAVE; Appendix D: CNEWCHRG | | |
| | Símilar comm | ands: NOS: CHANGE VMS: SET PROTECTION | | |
| | Examples: | ACCESS, DN=mylocal, PDN=myperm, UQ, | | |

MODIFY, DN=mylocal, PAM=R.

MSACCES (DTRC) Supply username and password to the Mass Storage System.

Syntax: MSACCES, US=us, MPW=mpw, AC=ac.

Parameters: us - user initials/username (default: the executing VAX user initials)

mpw - MSS password

ac - account/charge number (default: the executing VAX account number)

Remarks: MSACCES is required before using the MSx commands.

Similar commands: VMS: HFT ACCESS (DTRC)

Examples: MSACCES, MPW=mymsspw. MSAUDIT,.... -or- MSCHANG,.... -or- MSFETCH,.... -or- MSPASSW,.... -or- MSPURGE,.... -or- MSSTORE,.... = = = = = MSACCES, US=other, MPW=otherpw, AC=otherac. ^-- access the MSS as another user

MSAUDIT (DTRC) Sorted audit of Mass Storage files.

Syntax: MSAUDIT, MPW=mpw, L=ldn, LO=lo, SHOWPW=showpw, UN=un.

Parameters: mpw - your MSS password

| 10 | list options F - full audit (4 lines per file + cost per month and per day) |
|----|---|
| | S - short audit (length, filename, CT, M (permissions), number of uses, indirect/direct) |
| | I - intermediate audit (short plus number of uses, date created, date last accessed, number of streams (direct files), password (if requested), charge number (your files), cost per day) (default: LO=F) |

- showpw enter anything to include each file's
 password (your files only) in the
 output list (LO=S or I)
 (default: passwords are not included)
- un Username (User Initials) of the owner of the MSS files to be audited (default: your own files)

| 90/09/30 Rev(|) Page C- | 41 |
|---------------------|---|-------|
| Remarks: | MSACCES is required before using the MSx co MSAUDIT provides a sorted listing of your f on the Mass Storage System (4 lines per fil | iles |
| | two shorter forms (1 line per file). | |
| See also: | AUDIT | |
| Similar comm | mands: NOS: BEGIN, AUDIT; CATLIST VMS: DIRECTORY/FULL; MSSAUDIT | |
| Examples: | MSAUDIT,mymsspw. ^ 4 lines per file listing o your MSS files on \$OUT | f |
| | MSAUDIT,mymsspw,L=audout,LO=I,SHOWPW=x. ^ 1 line per file listing (including each file's pas written to local file AUDO | |
| | MSAUDIT,mymsspw,L=hisout,LO=S,UN=other. ^ short listing of MSS files user OTHER in file HISOUT | of |
| MSCHANG Change Mass | Storage System file attributes. | |
| Syntax: | MSCHANG, MDN=mdn, NMDN=nmdn, PW=pw, CT=ct, M=m, B PR=pr, NA=na, AC=ac, CP=cp. | R=br, |
| Parameters: | mdn - MSS filename whose attributes are to changed | be |
| | nmdn - new MSS filename | |
| | pw - new password 0 - clear the password | |
| | ct - file permit Category Type ct meaning | |
| | P or PR or PRIVATE private S or SPRIV semiprivate PU or PUBLIC public | |
| | m - alternate user permission mode for semiprivate and public files m meaning | |
| | E (EXECUTE) you can execute; oth can read or execute concurrently | ers |
| | R (READ) all can read or exec concurrently | ute |

Т

RU (READUP) all can read or execute; one (other) user can rewrite the file RA (READAP) all can read or execute; one (other) user can lengthen the file RM (READMD) all can read or execute; one (other) user can lengthen or rewrite the file U (UPDATE) all can read or execute; you can rewrite the file (APPEND) Α all can read or execute; you can lengthen the file (MODIFY) м all can read or execute; you can lengthen or rewrite the file (WRITE) you can read, execute, W lengthen, rewrite, or shorten the file; others have no concurrent access br - backup requirements br meaning ___ _____ CR off-station backup Y on-station backup pr - preferred residence pr meaning ____ Μ alternate storage - MSS Ν no preference - one of: na - abort on errors 0 non-0 - do not abort on errors (default: NA=0) - may alternate users obtain information ac about the file? (Y or N) ср - account number is to be replaced by the one currently in effect non-0 - change the account number MDN is required; the defaults for the others is to leave them unchanged. MSACCES is required before using the MSx commands. Similar commands: NOS: CHANGE

VMS: HFT CHANGE

Remarks:

90/09/30 Rev0

Page C-43

Examples: MSACCES, UN=myid, MPW=mymsspw. MSCHANG, MDN=myfile, NMDN=newname. ^-- change MSS file MYFILE to NEWNAME MSCHANG, MDN=myfile, NMDN=newname, NA=1. ^-- change MSS file MYFILE to NEWNAME (don't abort if MSS file NEWNAME already exists) MSCHANG, MDN=myfile, CT=PU. ^-- make MSS file MYFILE public MSCHANG, MDN=myfile, PW=mypw. ^-- put a password on MSS file MYFILE MSCHANG, MDN=myfile, BR=CR. ^-- make MSS file MYFILE a critical file with off-station backup MSCHANG, MDN=myfile, BR=Y. ^-- make MSS file MYFILE a noncritical file with on-station backup MSCHANG, MDN=myfile, CP=1. ^-- change the account number to the one in effect on the MSS MSFETCH (DTRC) Fetch a file from the Mass Storage System. Syntax: MSFETCH, DN=dn, MDN=mdn, DF=df, UN=un, PW=pw. Parameters: dn - the local dataset name mdn - the MSS dataset (file) name (default: MDN=dn) df - data format TR - transparent (no conversion) CB - character blocked (convert from CDC display code) (default: DF=TR) un - Username (User Initials) of the owner of the MSS file (omit for your own files) pw - optional MSS file password Remarks: MSACCES is required before using the MSx commands. MSFETCH defaults to DF=TR, FETCH defaults to DF=CB.

See also: ACQUIRE, FETCH Similar commands: NOS: ATTACH VMS: HFT FETCH (DTRC) Examples:

> MSACCES,UN=myid,MPW=mymsspw. MSFETCH,DN=in1,MDN=mymsfy1. MSFETCH,DN=in2,MDN=hisfy1,UN=him,DF=CB,PW=fy1epw.

IN1 is your file MYMSFYL transferred without conversion.

IN2 is file HISFYL belonging to user HIM converted from CDC Display Code (FYLEPW is the password HIM requires for access to the file).

MSPASSW Change Mass Storage System access password.

Syntax: MSPASSW,OLD=oldpw,NEW=newpw.

Parameters: oldpw - your current MSS access password

newpw - your new MSS access password

Remarks: MSACCES is required before using the MSx commands.

Similar commands: NOS: PASSWOR VMS: HFT PASSWORD

Examples: MSACCES, UN=myid, MPW=mymsspw. MSPASSW, OLD=mymsspw, NEW=newmsspw.

MSPURGE (DTRC) Purge a file from the Mass Storage System.

Syntax: MSPURGE, MDN=mdn.

Parameters: mdn - the MSS dataset (file) name (default: MDN=dn)

Remarks: MSACCES is required before using the MSx commands.

Similar commands: NOS: PURGE VMS: HFT DELETE; MSSDELETE (both DTRC)

Examples:

MSACCES,UN=myid,MPW=mymsspw. MSPURGE,MDN=mssfyl1. MSSTORE (DTRC) Store a file on the Mass Storage System. MSSfORE, DN=dn, mDN=mdn, DF=df, CT=ct, NA=na, PW=pw, Syntax: BR=br,M=m,PR=pr,AC=ac. Parameters: dn - the local dataset name mdn - the MSS dataset (file) name (default: MDN=dn) df - data format TR - transparent (no conversion) CB - character blocked (convert from CDC display code) (default: DF=TR) ct - Category type P - private PU - public S - semi-private (default: CT=P) na - No Abort 0 - abort if file already exists on the MSS 1 - replace the old MSS file, if one exists (default: NA=0) pw - optional MSS file password br - backup requirements br meaning _____ CR off-station backup Y on-station backup (default: BR=Y) - alternate user permission mode for m semiprivate and public files m meaning _____ E (EXECUTE) you can execute; others can read or execute concurrently (READ) R all can read or execute concurrently RU (READUP) all can read or execute; one (other) user can rewrite the file RA (READAP) all can read or execute; one (other) user can lengthen the file

RM (READMD) all can read or execute; one (other) user can lengthen or rewrite the file (UPDATE) all can read or execute; П you can rewrite the file (APPEND) all can read or execute; Α you can lengthen the file (MODIFY) all can read or execute; M you can lengthen or rewrite the file W (WRITE) you can read, execute, lengthen, rewrite, or shorten the file; others have no concurrent access br - backup requirements br meaning ___ _____ ______ CR off-station backup on-station backup Y pr - preferred residence pr meaning ------___ Μ alternate storage - MSS N no preference (default: PR=N) ac - may alternate users obtain information about the file? (Y or N) Remarks: MSACCES is required before using the MSx commands. DISPOSE See also: Similar commands: NOS: DEFINE VMS: HFT STORE (DTRC) Examples: MSACCES, UN=myid, MPW=mymsspw. MSSTORE, DN=in1, MDN=mssfy11. MSSTORE, DN=in2, MDN=mssfy12, BR=CR. MSSTORE, DN=in3, MDN=mssfy13, DF=CB, NA=1, PW=fy1epw. IN1 is stored as private file MSSFYL1. IN2 is stored as private file MSSFYL2 with offstation backup. IN3 is stored as private file MSSFYL3 (even is MSSFYL3 already exists) in CDC Display Code. FYLEPW is the password required for another user to access the file.

Syntax: NEWCHRG, OLD=oldchrgno, ID=id. Parameters: OLD= - the account number to be changed ID=id - change all files having this ID - change all files having a null ID ID (default: change all IDs) Remarks: NEWCHRG changes from the specified account number to the "current" number of the Cray job (from the ACCOUNT or most recent ALTACN statement). See also: ALTACN; MODIFY; Appendix D: CNEWCHRG Similar commands: NOS: BEGIN, NEWCHRG Examples: JOB, JN=.... ACCOUNT, AC=.... ACCESS, PROCLIB, OWN=PUBLIC. LIBRARY, PROCLIB:*.

NEWCHRG (DTRC - PROCLIB) Change permanent file account number.

NEWCHRG, OLD=1222233344. ^-- change all files from acc unt 1-2222-333-44 to the current one = = = = ... NEWCHRG, OLD=1222233344, ID=myid. ^-- change all files WITH ID=MYID from account 1-2222-333-44 to the current one = = = = ... ALTACN, AC=5666677788. NEWCHRG, OLD=12222433344. ^-- change all files from account 1-2222-333-44 to 5-6666-777-88

NOHOLD Cancel the effect of HOLD.

Syntax: NOHOLD,GRN=grn. Parameters: GRN=grn - generic resource name

See also: HOLD

NORERUN Control a job's rerunability.

Syntax: NORERUN, option.

Rev0 90/09/30

Page C-48

Parameters: option - ENABLE - declare a job nonrerunable if any of the nonrerunable functions are done DISABLE - stop monitoring nonrerunable functions (if a job has already been declared nonrerunable, that status is not changed) (default: ENABLE) See also: RERUN Similar commands: NOS: NORERUN Examples: NORERUN, DISABLE. NOTE Write text to a dataset. NOTE, DN=dn, TEXT='text'. Syntax: - the dataset to be written (at its current Parameters: DN= position) DN - write to \$OUT TEXT= - up to 153 character to be written Similar commands: NOS: NOTE VMS: OPEN, WRITE, CLOSE Examples: NOTE, DN=UIN, TEXT='*COMPILE myprog, mysub'. REWIND.UIN. UPDATE, I=UIN, OPTION Specify user-defined options. OPTION, LPP=n, PN={ p { ANY }, STAT=stat, BS=bsz, Syntax: ST=dev, DEF=pdev, XSZ=mxsz:mnsz, RDM, SEQ, UNB, BLK, NOF, OVF, SPD=sect, BFI=bfi, LM=mxsz,SZ=dsz. Parameters: LPP=n number of lines per page for job listings (0-255 decimal) LPP=0 do not change the current setting (default: 66) PN=p - select a processor (p is 1 or 2) PN=ANY - any available processor (if invalid, job aborts with an error message) (default: ANY)

BS - buffer size (# of octal 512-word blocks in circular I/O buffer) (default: system defined; BS and UNB are mutually exclusive) ST - storage device type SCR - scratch PERM - permanent DEF - preferred device types XSZ - maximum and minimum transfer sizes in octal sectors (default: system defined; normally half the buffer size) RDM - ramdom dataset (default: sequential; RDM and SEQ are mutually exclusive) SEQ - sequential dataset (default: sequential; RDM and SEQ are mutually exclusive) UNB - unblocked dataset (default: blocked; UNB and BS are mutually exclusive) BLK - blocked dataset (default: blocked; BLK and UNB are mutually exclusive) NOF - do not overflow to another device (default: system defined; NOF and OVF are mutually exclusive) OVF - overflow allowed (default: system defined; NOF and OVF are mutually exclusive) LN - maximum number of decimal 512-word blocks for a dataset - job aborts if exceeded SZ - number of decimal 512-word blocks to reserve for dataset when it is created (default: system defined) BFI - blank field initiation (octal ASCII code signaling the beginning of a sequence of blanks) OFF - no blank compression (default: octal 33 (ESC))

SPD - dataset is striped

STAT= - the level of I/C statistics gathered for local datasets to appear in the user logfile (user level - accounting information system level - device information) - installation defined ON OFF - no statistics FULL - user and system info (default: OFF) STAT - same as STAT=ON Similar commands: VMS: SUBMIT /QUEUE= PASCAL Compile a Pascal source program. PASCAL, I=idn, L=ldn, B=bdn, O=list, Syntax: CPU=cpu:char. Parameters: B= - generated binary load modules (default: \$BLD) 0= - Compiler options, separated by colons (default: A-:BP-:BREG=8:BT-:C-:D+:H2:^ H+24:L+:O+:P-:R+:RV-:S4:S+4:^ ST-:T+:TREG=8:U-:V+:X-:Z+) CPU= - Cray to execute the program cpu - CRAY-XMP CRAY-X1 - single-processor CRAY-X2 - dual-processor (default: the compiling machine) char - [NO]EMA - extended memory (24-bit A-register immediate loads; $common \ blocks > 4$ million words) [NO]CIGS - compressed index scatter/gather [NO] VPOP - vector population and parity [NO] READVL - vector length read instructions MEMSIZE=nK - (n * 1024) words MEMSIZE=nM - (n * 1048576) words - bidirectional memory [NO] BDM

Similar commands: NOS, VMS: PASCAL

Examples: PASCAL, I=mypasc.

| 90/09/30 | Rev0 |
|----------|------|
|----------|------|

PERMIT Grant/deny access to a permanent dataset.

Syntax: PERMIT, PDN=pdn, ID=uid, AM=am, RP, USER=ov, ADN=adn, NA, ERR, MSG.

Parameters: PDN=pdn - required

RP - remove the permissions

USER=ov - the name (User Initials) of the user to be granted/denied permission

ADN=adn - local dataset with the permit list

Similar commands: NOS: CHANGE; PERMIT VMS: SET PROTECTION; Access Control List

Examples: PERMIT, PDN=myfile, USER=abcd, AM=R. ^-- make file readonly for user ABCD = = = = PERMIT, PDN=myfile, USER=abcd, AM=N. ^-- remove all permissions for user ABCD

PRINT Write the value of a JCL expression to the logfile.

Syntax: PRINT(expression)

Parameters: exp - any valid JCL expression (maximum length: 8 characters)

Logfile format: UT060 decimal octal ASCII

Similar commands: NOS: DISPLAY VMS: WRITE SYS\$OUTPUT

Examples: SET(J1=J1+1) PRINT,J1.

PROC Begin an in-line procedure definition block. This is followed by the procedure prototype statement which names the procedure and gives the formal parameter specifications.

> Syntax: PROC, L=ldn, LIB=plib. name, p1, p2,..., pn ... ENDPROC.

Parameters: L - listing dataset to receive the echo of the definition block (default: \$LOG)

- LIB procedure library dataset to receive the definition body (default: \$PROC)
- name the name of the procedure (1-8 alphanumeric characters; should not be the same as a system verb)
- pi a formal parameter specification in one
 of the following formats:

pos - positional key=dvalue:kvalue - keyword key - formal keyword parameter dvalue - optional default value if the parameter is omitted kvalue - optional value if the parameter is specified with no value special cases: key= - specify a null value key=: - no defaults, but caller may specify key= or just key

See also: Section 3-3

Similar commands: NOS: .PROC VMS: always 8 parameters

Examples: PROC.

ENDPROC.

. . .

QUERY Determine the current status and position of a local file.

Syntax: QUERY, DN=dn, STATUS=status, POS=pos.

Parameters: STATUS= - the JCL symbolic variable name to receive the status of the dataset return values: value meaning ____ _____ -1 dn is not local 0 dn is closed dn is open for output 1 2 dn is open for input 3 dn is open for I/O

POS= - the JCL symbolic variable name to receive the position of the dataset return values: value meaning ____ -1 position indeterminate (not local, unblocked, closed) 0 BOD (beginning-of-data) 1 EOD (end-of-data) 2 EOF (end-of-file) 3 EOR (end-of-record) 4 mid-record Remarks: In addition, a logfile message is generated: QUO01 - DN: 1dn STATUS: status POS: pos where status is UNKNOWN, CLOSED, OPEN-O, OPEN-I, OPEN-I/O pos is N/A, BOD, EOD, EOF, EOR, MID Similar commands: NOS: ENQUIRE VMS: no local file concept Examples: QUERY, DN=myfile, STATUS=J0, POS=J1. IF(J0.LT.0)COMMENT. file myfile is not local . . . ELSE. COMMENT. file myfile is local . . . ENDIF. RELEASE Return a dataset. Syntax: RELEASE, DN=dn1:dn2:...:dn8, HOLD. Parameters: DN= - up to 8 dataset names HOLD - hold generic resource (do not return the allocation to the system pool) See also: HOLD, NOHOLD Similar commands: NOS: RETURN RELEASE, DN=temp:filel:out. Examples:

RERUN Control a job's rerunability. Syntax: RERUN, option. Parameters: option - ENABLE - mark job as rerunable regardless of any nonrerunable functions which may have been performed so far in the job DISABLE - mark the job as nonrerunable (default: ENABLE) See also: NORERUN Similar commands: NOS: NORERUN RERUN, ENABLE. Examples: RETURN Return control from a procedure to its CALLer. Syntax: **RETURN.** RETURN, ABORT. Parameters: ABORT - cause COS to issue a job step abort Similar commands: NOS: REVERT VMS: EXIT See PROC. Examples: REWIND Position a dataset at its beginning. REWIND, DN=dn1:dn2:...:dn8. Syntax: Parameters: DN = -up to 8 datasets to be rewound Similar commands: NOS: REWIND Examples: REWIND, DN=temp:out:in1. ROLLJOB Protect a job by writing it to disk. Syntax: ROLLJOB. There is no guarantee that a job will remain Remarks: recoverable.

Examples: ROLLJOB.

90/09/30 Rev0 Page C-55 SAVE Make a local dataset permanent and define its characteristics. SAVE, DN=dn, PDN=pdn, ID=uid, ED=ed, RT=rt, R=rd, W=wt, Syntax: M=mn, UQ, NA, ERR, MSG, EXO=exo, PAM=mode, ADN=adn.ACN. Parameters: RT=rt - retention period RT= - set to default ADN=adn - local dataset with the permit list ACN - use the alternate account number See also: ALTACN, MODIFY Similar commands: NOS: DEFINE; SAVE VMS: CREATE Examples: SAVE, DN=out, PDN=ABCOUT. _ _ _ _ SAVE, DN≈prog, PDN=mastprog, M=maint, PAM=R. ^-- the file is world-readable and YOU can't accidentally delete it SCRUBDS Write over a dataset before release. Syntax: SCRUBDS, DN=1fn. Parameters: 1fn - the uniquely accessed file to be overwritten Remarks: SCRUBDS writes zeros over an existing dataset. Examples: ACCESS, DN=myfy1, PDN=myfy1e, UQ. SCRUBDS, DN=myfy1. SEGLDR Segment loader. Syntax: SEGLDR, I=idn, L=1dn, DN=bdn1:bdn2:...:bdn8, LIB=1ib1:1ib2:...:lib8,ABS=adn, CMD='directives',GO. - Dataset with SEGLDR directives Parameters: I= (default: SIN) I - Same as I=\$IN L= - Listable output (default: \$OUT) - Same as L=\$OUT L

DN= - Up to 8 binary load dataset(s) DN - Same as DN=\$BLD (default: \$BLD)

Rev0 90/09/30

- LIB= Up to 8 relocatable object libraries to be searched
- ABS= Dataset to contain the absolute program (default: \$ABD)
- CMD= Global directives to be processed; treated as first record read from I=idn; separate commands with semicolons (e.g., CMD='BIN=bdn;MAP=PART')
- GO Load and execute; ignored for a segmented load

Remarks: By default, input load modules are read from \$BLD.

Directives: See section 3-6.

Similar commands: NOS: SEGLOAD VMS: virtual machine

Examples: CFT77,B=myobj. SEGLDR,DN=myobj,LIB=mylib,CMD='MAP=PART',GO.

SET Change the value of a JCL variable.

Syntax: SET(symbol=expression)

Parameters: exp - a valid arithmetic, logical or literal assignment expression - may be delimited by parentheses

Remarks: The job-step aborts if the variable is unknown, is changable only by COS, or is a constant.

Similar commands: NOS: SET VMS: \$ name = value

Examples: SET(J1=J1+1) <-- increment procedure-local register J1 by 1

> SET(G1=(SYSID.AND.177777B)) ^-- put the low-order 2 characters of the current system revision level into global register G1

SET(G3=((ABTCODE.EQ.74).AND.(G2.EQ.0))) ^-- define global register G3

SID Debug programs interactively or in batch.

Syntax: SID=adn, I=idn, S=sdn, L=1sn, ECH=edn, CNT=n.

Page C-57

Parameters: adn - absolute dataset name (from LDR, AB=adn) I= - Input directives (default: \$IN) S= - Symbol dataset name (default: \$DEBUG) - Listable output L= (default: \$OUT) ECH= - Dataset for echoing input directives (default: no echoing) ECH - Same as ECH=1dn CNT= - Breakpoint interrupt count (default: 0 (no abort)) Similar commands: NOS: CID VMS: DEBUG SKIPD Skip blocked datasets (position at EOD (after last EOF)). Syntax: SKIPD, DN=dn. Parameters: DN - (default: \$IN) Same as: SKIPF, DN=dn, NF. Similar commands: NOS: SKIPEI VMS: OPEN with ACCESS=APPEND in program SKIPD, DN=myfile. Examples: SKIPF Skip blocked files from current position. SKIPF, DN=dn, NF=nf. Syntax: Parameters: DN=dn - (default: \$IN) NF=nf - decimal number of files to skip forward NF=-nf - decimal number of files to skip backward - position after the last EOF of the NF dataset (default: NF=1) Similar commands: NOS: SKIPF; SKIPFB; SKIPR SKIPF, DN=myfile. Examples:

Page C-58

SKIPR Skip blocked records from the current position. Syntax: SKIPR, DN=dn, NR=nr. Parameters: DN=dn - (default: \$IN) NR=nr - decimal number of records to skip forward NR=-nr - decimal number of records to skip backward NR - position after the last EOF of the current file (default: NR=1) Examples: SKIPR, DN=myfile. SKIPU Skip sectors on unblocked datasets. Syntax: SKIPU, DN=dn, NS=ns. Parameters: DN=dn - no default NS=ns - decimal number of sectors to skip forward NS=-ns - decimal number of sectors to skip backward NS - position after the last sector of the dataset (default: NS=1) Examples: SKIPU, DN=myfile. SORT Sort/merge. Syntax: SORT, S=sdn[:sdn...], M=mdn[:mdn...], O=odn, DIR=ddn, L=1dn, ECHO, RETAIN, NOVERF. Parameters: - Input dataset of up to 8 unsorted files S≖ - Input dataset of up to 8 sorted files to M= be merged (S or M or both must be specified) 0= - Output dataset (required) DIR= - Dataset with SORT directves (default: \$IN) - Listable output L=

> L=0 - No listable output ECHO - Write directives to L=ldn (Not allowed if L=0)

(default: \$OUT)

90/09/30 Page C-59 Rev0 RETAIN - Retain input order for equal keys NOVERF - Do not verify the sort (default: verify) Similar commands: NOS: SORT5 VMS: SORT SPY Generate a histogram on time usage within a program to locate inefficient code. Syntax: SPY, PREP, BS=bcktsz, D=dbugdn, S=scrtch, SUB=rtn1:rtn2:..:rtnn,TS=time. SPY, POST, ADDRESS, L=1istdn, NOLABEL, NOLIB, S=scrtch, SUB=rtn1:rtn2:..:rtnn.MINHIT=n. Parameters: BS= - bucket size in words; each bucket begins on a word address that is a multiple of the bucket size (default: 4) D≖ - dataset containing the program's symbol table (default: SDEBUG) S= - dataset where SPY, PREP will write tables for SPY, POST to use SUB= - list of up to 20 routines to be analyzed TS= - time slice in microseconds (default: 500) ADDRESS - the report will be by address instead of by label L= - the output report listing dataset (default: SOUT) NOLABEL - the bucket size will be an entire routine NOLIB - exclude library calls to routines whose names begin with "\$" MINHIT= - minimum number of hits required to generate a report line for a bucket or label (default: 1; 0 is NOT recommended)

1

Remarks: At SPY's request, COS reads the address of the current machine instruction. A group of addresses is called a bucket; accessing a bucket is called a hit. After execution, SPY generates a report of all buckets, including a bar graph showing where the time has been spent. Use SEGLDR to create the absolute; LDR mixes code and data making it more difficult to analyze. Similar commands: NUS: HOTSPOT VMS: PCA Examples: CFT, ON=IZ. -or- CFT77, ON=Z. -or- CAL, SYM. -or- PASCAL, O=DM3. SEGLDR, ABS=myabs. <-- you must create an absolute program <-- prepare for SPY SPY, PREP. <-- run your program myabs. <-- prepare the report SPY, POST. EXIT. SPY.POST. Since an absolute module is always created, you could use SEGLDR. SPY, PREP. \$ABD. SPY, POST. EXIT. SPY, POST. SUBMIT Send a local dataset to the COS input queue. Syntax: SUBMIT, DN=dn, SID=sf, DID=df, DEFER, NLRS. Parameters: DN= - Dataset containing the job (required) SID= - Source front-end identifier (2 alphameric characters) DID= - Destination front-end identifier (2 alphameric characters) DEFER - Defer the SUBMIT until the dataset is released (default: SUBMIT occurs immediately) NLRS - Do not release the dataset after the SUBMIT; it remains local and read-only

(default: dataset is released after the

SUBMIT)

```
90/09/30 Rev0
                                                           Page C-61
         Similar commands: NOS: ROUTE, DC=IN; CSUBMIT
                            VMS: SUBMIT; CRAY SUBMIT
         Examples:
                      SUBMIT, DN=myjob1.
SWITCH
         Turn pseudo sense switches on/off.
         Syntax:
                      SWITCH, n=x.
         Parameters: n - switch number (1-6)
                      x - switch position
                           ON - turned on (set to 1)
                           OFF - turned on (set to 0)
         Similar commands: NOS: SWITCH; OFFSW; ONSW
         Examples:
                      SWITCH, 2=ON.
UNBLOCK Convert a blocked dataset to an unblocked dataset.
                      UNBLOCK, DN=1dn.
                                               (1)
         Syntax.
                      UNBLOCK, I=idn, O=odn.
                                               (2)
         Parameters: DN= - the dataset to be replaced (using an
                            intermediate dataset $UNBLK)
                            (1dn is rewound before and after)
                      I= - the blocked input dataset
                            (default: $IN)
                            (idn is not rewound before the copy)
                      0= - the unblocked output dataset
                            (if previously marked to be unblocked
                            (ASSIGN), odn is not rewound before;
                            otherwise, odn is replaced)
                      UNBLOCK is intended primarily for postprocessing
         Remarks:
                      datasets created by or for certain stations.
         Examples:
                      UNBLOCK, DN=myfile.
                                   ^-- Replace MYFILE with unblocked
                                       copy of itself
                      UNBLOCK, I=myb1k, O=myunb1k.
                                   ^-- Copy blocked file MYBLK as
                                       unblocked file MYUNBLK
```

UPDATE Source and data maintenance. UPDATE, P=pdn, I=idn1:idn2:...:idnn, C=cdn, N=ndn, Syntax: L=1dn, E=edn, S=sdn, *=m, /=C, DW=dw, DC=dc,ML=n,&,opts. where & is one of: F Q[=d1:d2:...:dn]Q = 'd1, d2, ..., dj. dk, ..., dn'Parameters: P= - Program library dataset (default: \$PL) - Same as P=\$PL Ρ P=0 - Required for a creation run I= - Input datasets with directives and text (Maximum: 100 datasets) (default: \$IN) - Same as I=\$IN T I=0 - No input dataset C= - Compile output dataset (default: \$CPL) С - Same as C=\$CPL C=O - No compile output N= - New program library dataset (default: creation run: SNPL modification run: no new PL) - Same as C=\$CPL N N=0 - No new PL L= - Listable output (default: \$OUT) I. - Same as L=\$OUT L=O - No listable output E= - Error dataset name (default: SOUT) - Same as E=\$OUT E E=0 - Errors written to L=1dn (If edn and 1dn are the same, 1dn is used and E=0) S= - Source output dataset (default: \$SR) S - Same as S≈\$SR S=0 - No source output *=m - Master character for directives (defaults: creation run: modification run: read from the PL)

/=c - comment character (default: /) DW= - Data width (number of characters written per line to compile and source datasets (defaults: creation run: 72 modification run: dw when PL was created) DW - Same as DW=72 (creation) or use dw when PL was created (modification run) DC= - Declared modifications option: ON - mod declaration required OFF - mod declaration not required (default: OFF) ML= - Message level (highest severity level to suppress): 1 - comment 2 - note 3 - caution4 - warning 5 - error (default: 3 - suppress COMMENT, NOTE, and CAUTION messages) - Full UPDATE mode F (default (F and Q omitted): normal UPDATE mode) 0= - Quick UPDATE mode (Maximum: 100 deck names) (default (F and Q omitted): normal UPDATE mode) opts - NA - no abort NR - no rewind of C and S files IF - write conditional text summary to ldn IN - write input to 1dn ID - write identifier summary to 1dn ED - write edited card summary to 1dn CD - write compile dataset generation directives to 1dn UM - write unprocessed modifications to ldn and/or edn SQ - put sequencing in source output in columns dw+l on (no effect on compile output) NS - no sequencing in compile output K - sequence decks according to Q Similar commands: NOS: UPDATE VMS: CMS; LIBRARIAN

Examples: UPDATE, I=mysorc, P=0, ID. ^-- create \$NPL, list identifiers = = = = = UPDATE. CFT, I=\$CPL. ... /EOF *COMPILE a,b,... /EOF

WRITEDS Initialize a blocked dataset.

Syntax: WRITEDS, DN=dn, NR=nr, RL=r1.

Parameters: DN=dn - required

NR=nr - required - decimal number of records to be written

- RL=rl optional decimal record length (if non-zero, the first word of each record is the record number as a binary integer starting with 1) (default: 0 (a null record))
- Remarks: Writes a single file containing a specific number of records of a specific length. This is useful only for random (direct-access) files, which must be pre-formatted.

Examples: WRITEDS, DN=myfile, NR=1000, RL=125.

***** Appendix D *****

*** DEC VMS DCL Commands ***

DEC VMS DCL (Digital Command Language) commands have the following general syntax:

verbparam1param2...! comments@filenameparam1param2...param8! commentsRUNfilename! comments

verb is the name of the routine to be executed. It consists of an alphabetic character (A-Z, a-Z, \$, _) followed by 0-30 alphanumeric characters for the name of the command. A procedure (.COM) is executed using an at sign ("@") followed by the name of the procedure file. A user program is executed by the RUN statement.

parami are parameters, which may be positional or keyword.

comments follow an exclamation mark ("!") that is not part of a quoted parameter.

Because VMS has an extensive on-line help facility, the individual DCL commands are not described here. For a list of the help topics, type "HELP". For specific helps, type "HELP topic". The Computer Center maintains the following help libraries which are always available:

| HLP\$LIBRARY | @CCF | general information about the Computer Center |
|----------------|-------------------|---|
| HLP\$LIBRARY_1 | @DTLIB | subprograms in library DTLIB (Cray COS, CDC NOS, and DEC VAX/VMS) |
| HLP\$LIBRARY_2 | @UTILITIES | commands, programs, procedures, and packages added at DTRC |
| HLP\$LIBRARY_3 | @CRAY | DTRC additions to Cray |
| HLP\$LIBRARY_4 | | Reserved for future use |

Rev0 90/09/30

assigned to logical name TAPE

*** Selected DEC VAX/VMS Commands ***

The following are a few of the DEC VAX/VMS DCL commands:

ALLOCATE Assign a tape drive to a logical name.

Syntax: ALLOCATE device logical_name Parameters: device - the logical name of a specific or generic tape drive log_name - the name by which the tape is to be known to the job (1-255 characters) Examples: \$ ALLOCATE MU: tape ^-- next available tape drive starting with MU will be

DEALLOCATE Return a previously allocated device and disassociate the job's logical name from the tape drive.

Syntax: DEALLOCATE logical_name DEALLOCATE device_name

DEALLOCATE /ALL

Parameters: log_name - the name by which the tape is known to the job dev_name - the name of the device (use if the device was not deallocated and the logical name is no longer defined)

Qualifiers: /ALL - deallocate all allocated devices

Examples: \$ DEALLOCATE tape ^-- deallocate the tape drive associated with logical name TAPE = = = = \$ DEALLOCATE \$2\$mua0

^-- deallocate tape drive mua0

90/09/30 Rev0 Page D-3 DISMOUNT Release a tape volume that was previously mounted. Syntax: DISMOUNT device name Parameters: device name - the physical or logical name of the device to be dismounted Qualifiers: /NOUNLOAD - Do not unload the tape (keeps the device and volume in a ready state (default: /UNLOAD) Examples: \$ DISMOUNT /NOUNLOAD tape ^-- release file TAPE but keep the tape mounted for a future MOUNT INITIALIZE Initialize a magnetic tape. Syntax: INITIALIZE device vsn Parameters: device - the name given the tape in the ALLOCATE Vsn - a 6-character volume serial number (all DTRC Network tapes are NAnnnn, where nnnn is a 4-digit number) Remarks: HELP INITIALIZE for additional qualifiers Examples: See page 6-1-6 MOUNT Mount a magnetic tape and, if labelled, check the label. \$ MOUNT device,... [vsn,...] [logical_name] Syntax: /BLOCKSIZE=mb1 /COMMENT="string" /DENSITY=den /FOREIGN /RECORDSIZE=mr1 /[NO]LABEL /[NO]UNLOAD /[NO]WRITE Parameters: device - physical or logical name of the tape drive (for more than one tape, separate with commas or plus signs) - the volume serial number of the tape(s) vsn as recorded in the tape's label record (0-6 characters) (not with /FOREIGN) log_name - the logical name to be used (not needed if is a logical name is used for DEVICE)

Rev0 90/09/30

Qualifiers: /BLOCKSIZE= - the default block size in bytes (range: 18-65,534; default: 2048) /COMMENT =- specify additional information to the operator /DENSITY= - the tape density (1600 or 6250) (default: the density of the first record of the volume) /FOREIGN - an unlabelled tape /LABEL - the tape has VAX/VMS ANSI labels /NOLABEL - the same as /FOREIGN (default: /LABEL) /RECORDSIZE= - the number of characters in each record - normally used with /FOREIGN and /BLOCKSIZE $(mr1 \le mb1)$ /UNLOAD - unload the tape when DISMOUNTed /NOUNLOAD - do not unload the tape (default: /UNLOAD) /WRITE - the tape can be written - the tape is read only /NOWRITE (default: /WRITE) Examples: \$ MOUNT tape: /FOREIGN /DENSITY=1600 -/RECORDSIZE=140 /BLOCKSIZE=5040 -/comment="Please mount slot98 ", -"vsn=ABCD01 ring" ^-- mount a slot tape for writing blocked records _ _ _ _ _ \$ MOUNT mytape NA9999 /DENSITY=1600 /comment="Pls mount with NO ring" ^-- mount a read-only tape See page 6-1-6 for an example of initializing a tape. SET MAGTAPE Define default characteristics for subsequent use of a magnetic tape device; position a magnetic tape. SET MAGTAPE device /DENSITY /END OF FILE Syntax: /LOG /LOGSOFT /REWIND /SKIP=option Parameters: device - the logical name of a specific or

generic tape drive

_ ____

| Qualifiers: | /DENSITY | - | the default der for writes to a unlabelled tape | - | |
|---|---|----|---|-------------------------------------|--|
| | /END_OF_FILE | - | write a tape ma position on the | ark at the current e tape | |
| | /LOG | - | display informa done | ation about what was | |
| | /logsoft | - | log soft errors on the error log file | | |
| | /REWIND | - | rewind the tap | e | |
| | /SKIP= | - | position the ta option | ape meaning | |
| | | | | skip <n> blocks</n> | |
| | | | END_OF_TAPE | position at the end-of-tape mark | |
| | | | FILES:n | skip <n> files</n> | |
| | | | RECORD:n | skip <n> records</n> | |
| /UNLOAD - rewind and unload the tape | | | | | |
| Similar commands: NOS: BKSP, REWIND, SKIPEI, SKIPF, SKIPFB, UNLOAD, WRITEF | | | | | |
| Examples: | \$ MOUNT tap | e: | /FOREIGN | | |
| | \$ SET MAGTAP | | tape: /DENSIT | | |
| | ^ mount a foreign tape and set the write density to 6250 cpi | | | | |
| | | | write densit | y 10 0250 cp1 | |
| | \$ SET MAGTAP | | /SKIP=FILES:4 | | |
| | | ^_ | skip forward | 4 files | |

*** Selected DEC VAX/VMS Additions ***

The following are DTRC additions to DEC VAX/VMS:

APRINT (DTRC) Print one or more files on the printer in Annapolis.

| Syntax: | APRINT file(s) copies delete PRINT_quals | | | | |
|-------------|---|--|--|--|--|
| Parameters: | file - the file(s) to be printed | | | | |
| | copies - number of copies (default: 1) | | | | |
| | <pre>delete - DELETE to delete the file(s) after printing (default: keep the files)</pre> | | | | |
| | PRINT_quals - P4-P8 may be used for additional qualifiers for the PRINT statement | | | | |
| Remarks: | All PRINT qualifiers which we support are available. | | | | |
| | The files are places into terminal queue SYS\$ANAP. | | | | |
| See also: | CAPRINT | | | | |
| Examples: | APRINT myfile.out APRINT myfile.out 4 | | | | |

90/09/30 Rev0

AUX

(DTRC) Turn an auxiliary printer on or off; for supported printers, send control characters to control character size and page eject.

Syntax: AUX option

Parameters: option - one of:

On/off: ON - turn printer on - turn printer off OFF Page eject: TOP - page eject (leave AUX ON) TOPOFF - page eject (leave AUX OFF) ALPS: ACC - condensed (17 cpi) APC - Pica (10 cpi) Brother 2024L: BCC - condensed characters (18 cpi) BCCOFF - condensed characters off BEC - Elite (12 cpi) BPC - Pica (10 cpi) BWC - wide characters (5 cpi) BWCOFF - wide characters off Okidata MicroLine 82 or 84: 000 - condensed characters (15 cpi) OLC - large (8.3 cpi) OPC - Pica (10 cpi) - wide characters (5 cpi) OWC VC132 - same as OCC VC80 - same as OPC

Remarks: Other printers which use the same control codes may use the corresponding options.

See also: AUXPRINT

Similar commands: NOS: BEGIN, AUX; BEGIN, AUXPRNT

Examples: \$ AUX ON \$ TYPE myfile.dat \$ AUX OFF = = = = = \$ AUX ON \$ AUX BCC <-- condensed on Brother 2024L \$ TYPE myprog.for \$ AUX BCCOFF \$ AUX OFF AUXPRINT (DTRC) Print one of more files on an axiliary printer (one attached to an interactive terminal). Syntax: AUXPRINT files /ALLTYPES /CC /COPIES= /CS= /DOUBLE SPACE /EJECT AT END /FF /HEADER /LOG /NEWPAGE /PS= /PW= /SHIFT= /SKIP= Parameters: files - the name of the file or commaseparated list of files to be printed -- wildcards are allowed (defaults: filename: FOR001; type: .DAT) Qualifiers: /ALLTYPES - controls the processing of certain file types (default: /NOALLTYPES) /CC - carriage control is in column 1 (default: /NOCC) /COPIES= - number of copies (default: 1) - character set (see AUX) /CS= ALPS: ACC, APC Brother 2024L: BCC, BDC, BEC, BPC, BWC Okidata MicroLine 82 or 84: OCC, OLC, OPC, OWC (default: however the printer is set) /DOUBLE - double spacing (default: /NODOUBLE) /EJECT - eject to a new page at the end of the last file printed (ignored for /CC) (default: /NOEJECT AT END) /FF - page eject for <FF> in column 1 (forces /NOCC; /NOFF ignores <FF>) (default: /FF) /HEADER - print a header with the filename before printing the file (default: /NOHEADER) /LOG - display information about the printing (default: /LOG) - start each file on a new page /NEWPAGE (default: /NONEWPAGE) - the page size (number of lines /PS=possible per page) (default: /PS=66) /PW= - the page width (maximum number of columns per line) -- less than 133 (defaults: /PW=80; /PW or /PW=0 implies /PW=132)

90/09/30 Rev0

Page D-9

/SHIFT= - number of columns to shift each line (for /CC, columns 2 on are shifted) (default: /SHIFT=0; /SHIFT implies /SHIFT=1) - number of lines to skip before /SKIP= printing the file (default: /SKIP=0; /SKIP implies /SKIP=10) Remarks: Other printers which use the same control codes as the ALPS, Brother 2024L or Okidata MicroLine 82 or 84 may use the /CS character sets. See also: AUX Similar commands: NOS: BEGIN, AUX; BEGIN, AUXPRNT AUXPRINT myprog.for Examples: _ _ _ AUXPRINT memo.txt /CS=BEC /CC ^-- print Elite characters on Brother 2024L printer, each line has carriage control in column 1 _ _ _ _ _ AUXPRINT a*.dat /CS=OCC /N /E ^-- print all .DAT files starting with A using condensed characters on an Okidata MicroLine 82 printer starting each file on a new page and ejecting a page after the last file - - - - -AUXPRINT myprog.lis /PW ^-- print a wide compilation listing (assumes wide paper is in the printer)

CAPRINT (DTRC) Convert the record attribute of a file having Fortran carriage control characters in column 1 of each line to "Fortran carriage control" and print on the remote printer in Annapolis.

> CAPRINT file copies keep PRINT_quals Syntax: Parameters: file - the file to be printed - number of copies copies (default: 1) - any character will keep the keep converted file after it has been printed (default: delete the file) PRINT quals - P4-P8 may be used for additional qualifiers for the PRINT statement This is useful for printing CDC output files or Remarks: any VAX file having column 1 carriage control but a different record attribute. The files are places into terminal queue SYS\$ANAP. See also: APRINT Examples: CAPRINT abcd.out ^-- convert and print file ABCD.OUT CAPRINT abcd.out 5 ^-- print 5 copies of ABCD.OUT CAPRINT abcd.out "" keep ^-- print 1 copy and keep the converted file (the next version of ABCD.OUT)

CNEWCHRG (DTRC) Change the account number on Cray permanent files from the VAXcluster. CNEWCHRG upw old ac Syntax: [new ac] [id] [wait] [type] Parameters: upw - Your user password for the generated ACCOUNT statement (the AC= value is taken from your current VAX/VMS session) (default: none - upw is required) old ac - the old account number (default: none - old ac is required) new ac - the new account number (if not your current VAX/VMS account number) id - optional Cray ID qualifier ID=id - a specific ID ID - the null ID (default: all IDs) - WAIT - wait for the job to complete, wait display, delete the .CPR file (synonyms: YES, TRUE) other - do not wait (Cray job creates file NUCRAC.CPR) type - TYPE - type the generated Cray job at your terminal other - do not type it Remarks: CNEWCHRG creates and submits a Cray job to make change. CNEWCHRG works from any node of the VAXcluster. NEWCHRG is a Cray statement. Any existing file NUCRAC.CPR is deleted before the Cray job is submitted. This procedure creates and deletes all versions of file N\$U\$A\$C.JOB. Similar commands: COS: ALTACN/MODIFY; NEWCHRG NOS: CHANGE

Examples: CNEWCHRG myupw 1222233344 ^-- change all files from 1-2222-333-44 to the current (ACCOUNT) account number without waiting for it to complete Some time later: DIRectory NUCRAC.CPR ^-- see if the job has completed SET TERMinal /Width=132 TYpe NUCRAC.CPR ^-- look at it SET TERMinal /Width=80 DELete NUCRAC.CPR;* ^-- delete the file e = = = = "" WAIT CNEWCHRG myupw 1222233344 5666677788 ^-- change all files from 1-2222-333-44 to an alternate account number and wait for it to finish (note: the "" is the ID parameter - a null string to change all files) - - - - -CNEWCHRG myupw 1222233344 "" myid "" TYPE ^-- change ID=MYID files from 1-2222-333-44 to the current (login) account number without waiting for it to complete -type the generated job before submitting it This will display: JOB, JN=NUCRAC. ACCOUNT, US=myid, AC=myvmsaccount, UPW=myupw. ACCESS, DN=PROCLIB, OWN=PUBLIC. LIBRARY, DN=PROCLIB:*. NEWCHRG, OLD=1222233344, ID=myid. Look at it and delete it sometime later (see

previous example).

CNEWPW (DTRC) Change your Cray password.

Syntax: CNEWPW old_pw new_pw new_pw [ac] [wait]

Parameters: old_pw - your current Cray password

- new pw your new Cray password
- new_pw your new Cray password again for verification
- ac your Cray account number (if not the same as your current VMS login account number)
- Remarks: CNEWPW creates and submits a Cray job to make change.

CNEWPW works from any node of the VAXcluster. NUPW= is a parameter in the Cray ACCOUNT statement.

Any existing file NUCRPW.CPR is deleted before the Cray job is submitted.

This procedure creates and deletes all versions of file N\$U\$P\$W.JOB.

Similar commands: COS: ACCOUNT VMS: SET PASSWORD

Examples: CNEWPW myold mynew mynew ^-- change your password without waiting for it to be done Some time later: DIRectory NUCRPW.CPR ^-- see if the job has completed SET TERMinal /Width=132 TYpe NUCRPW.CPR <-- look at it SET TERMinal /Width=80 DELete NUCRPW.CPR:* <-- delete the file = = = = = CNEWPW myold mynew mynew 1222233344 WAIT ^-- change your password for Job Order Number 1-2222-333-44 and wait for it to complete

CSUBMIT (DTRC) Submit a job to the Cray. CSUBMIT file(s) /AC=accountno /US=username Svntax: /UPW=password /NUPW /EOF=string /AFTER=time /LOG Parameters: file - one of: . a single file containing a complete Cray job . a comma- and/or plus-separated list of files which make up the Cray job (default filetype: .JOB) Qualifiers: /AC - if you have multiple account numbers, use /AC to specify an account number other than your current VAXcluster login account /AC is required if you use /US. (default: your VAXcluster login account number) /AFTER - specifies when the job is to be sent to the Cray (default: the job is queued for immediate submission to the Cray) /EOF - specifies the embedded COS end-of-file separator contained in the submitted job (if non-alphanumeric characters (including lower case letters) are used, they must be enclosed in quotes) For example, /EOF="E O F" means that lines containing just the 5-character string "E" space "O" space "F" are to be interpretted as end-of-file. /EOF=DAVE means that lines containing just the string "DAVE" are end-of-files. (default: /EOF="/EOF") /LOG - if you have CRAY SET TERM INFORM turned on and you do not want to see the message that your job has been queued, use /NOLOG (default: /LOG) /NUPW - indicates that your Cray password is to be changed

You will be prompted for your current Cray password. If it does not match the database password, you are prompted for the password in the database. If they match, you are prompted for your new password, which will be put into the database.

Note: To do nothing more than change your password, use

CSUBMIT /NUPW

If you are changing only the password in the database, no Cray job will be generated. If you are changing your password on the Cray, a dummy job will be created and run with the output in file SETNUPW.CPR

Note: /NUPW cannot be used in a batch job

/UPW - the first time, your password will be entered into the database -- subsequently, use /UPW only if you are using a different /US

(default: /UPW=password in database)

/US - specify a different username for the Cray job (of course, you must be authorized to use the other username and must also supply /AC and /UPW)

Note /US cannot be used in a batch job

(default: /US=the first 4 letters of your VAXcluster login username)

Remarks: This differs from CRAY SUBMIT in that jobs submitted using CSUBMIT do not need an ACCOUNT statement, CSUBMIT constructs it for you.

> The first time you use CSUBMIT, your password (UPW) is added to a database. Every CSUBMIT then uses this database password to generate an ACCOUNT statement for you. Thus, your Cray job files no longer have your password, meaning that every time you change your password, you don't have to change all your Cray job files.

Similarly, for your account number, a single Cray job may now be run under a different account, or even a different username, without changing the job file.

N.B. If there is an ACCOUNT statement in your job, it will be ignored and a new ACCOUNT statement will be generated.

Note that you must be logged into a VAXcluster node which connects to the Cray.

You can still use CRAY SUBMIT and RCSUBMIT to submit Cray jobs. However, these do not use (and cannot modify) the database, and, therefore, require ACCOUNT statements.

See also: CRAY SUBMIT; CNEWPW

Similar commands: NOS: CSUBMIT NOS/VE: SUBMIT_CRAY

Examples: \$ CSUBMIT myjob /UPW=mycraypw ^-- submit for the first time the Cray job in file MYJOB.JOB using CSUBMIT

\$ CSUBMIT myjob /AC=5666677788
 ^-- submit the job and charge it to
 another of my accounts

\$ CSUBMIT myjob /US=other /AC=9888877766 /UPW=otherpw ^-- Submit the job as another user

Enter ABCD's New CRAY password. Password: <pw> Verification: <pw> %CX-S-SUB_OK, Job: SETNUPW queued for submission S

S CSUBMIT /NUPW ^-- Change your Cray password (assuming you are user ABCD) when your actual Cray password is not the same as the one in the database -- perhaps you had changed it using CNEWPW or CRAY SUBMITted a job to change it Enter ABCD's current Cray password. Password: <pw> <-- your password entries are not echoed ABCD's current Cray password does not match the CSUBMIT password. Please enter ABCD's CSUBMIT password. Password: <pw> Enter ABCD's new Cray password. Password: <pw> Verification: <pw> <-- since your current \$ and new Cray passwords are the same, the database is updated, but no Cray job is created \$ CSUBMIT myjcl,myprog.for,mydatal.dat+ mydata2.dat,mydata3.dat ^-- Create and submit a job comprised of the following VAX/VMS files: - Cray job control MYJCL.JOB statements MYPROG.FOR - Cray Fortran program MYDATA1.DAT - first part of data file MYDATA2.DAT - second part of data file MYDATA3.DAT - separate data file \$ CSUBMIT myjob /EOF="The end" ^-- (upper case T, the rest is lower case) submit a job with the end-of-file lines as "The end" \$ CSUBMIT myjob /AFTER=18:00 (a) myjob /AFTER=TOMORROW (b) \$ CSUBMIT \$ CSUBMIT myjob /AFTER≠+00:05 (c) ^-- Submit the job a) after 6 PM, b) tomorrow, c) in 5 minutes

1

DETAB (DTRC) Remove tabs from a file or convert tab-format Fortran source lines to fixed-format. Syntax: DETAB in file_spec out file_spec /FORTRAN /LOG /INCREMENT=inc /TABS=tab list Parameters: in file spec - the input file containing tabs out file spec - the output file with any tabs removed (default: next version of in file spec) Qualifiers: /FORTRAN - tab-format lines are converted to fixed-format (the first tab is set at column 7 (or 6 for continuation lines) and remaining tabs are converted to three blanks) Since tabs are collapsed to three blanks, it is unlikely that a DETABbed line will exceed 72 characters. If any lines do, you will be told how many and the length of the longest line. /NOFORTRAN - no reformatting is done /INCREMENT= - tabs are set every <inc> columns If both /TABS and /INCREMENT are specified, tabs are set at the column(s) specified by /TABS= and every <inc> columns after that. /LOG - list summary information and any warning messages (Default: /NOLOG) /TABS=n - set one tab at column n /TABS = (n1, n2, ..., nn)- set tabs at these columns If /INCREMENT=inc is not specified, then the tabs following the last defined tab stop, are each converted to a single blank.

91/05/15 Rev0

If /INCREMENT=inc is specified, then the tabs following the last defined tab stop will be every inc columns after the last defined tab stop.

(Defaults: /TABS=0 /INCREMENT=8 /NOFORTRAN)

Note: /FORTRAN overrides /TABS and /INCREMENT.

Remarks: This is useful for:

- . Preparing files to go to the Cray, Xerox 8700 or Microfiche, which don't recognize the tab character
- . Removing tabs in Fortran programs (for sending to another computer (such as the Cray and CYBER 860) which don't recognize the tab-format).
- . Changing the tab values while removing them (e.g., changing from every 8 columns, which is the VAX/VMS standard, to every 5 columns).

Examples: DETAB myprog.for /F

PRINTRM1 (DTRC) Print a file on the remote mini at Annapolis (RM1).

Remarks: Since RM1 is no longer available, use APRINT or CAPRINT to print in Annapolis.

Rev0 91/01/15

QPRINT (DTRC) Print a file on a CDC CYBER 860 central site. QPRINT vaxfile node /ASCII Syntax: /DELETE /JOB=<job extension> /NAME=<job name> /TID=<terminal id> Parameters: vaxfile - file specification of the VAXcluster file to be printed on CDC node - the remote node on which the file is to be printed. One of: MFN - the CDC CYBER 860 Qualifiers: /ASCII - controls whether <vaxfile> is to be printed in upper and lower case (/ASCII) or just upper case (/NOASCII) (default: /NOASCII) /DELETE - controls whether <vaxfile> is deleted after it has been sent (default: /NODELETE) /JOB - the three alphanumeric characters to follow your user initials for the CDC jobname - if fewer than 3 characters, leading zeros are added (/JOB and /NAME are mutually exclusive) (default: /JOB=000) - the 1- to 7- alphanumeric character CDC /NAME job name - if fewer than 7 characters, it is padded on the right with zeros (/JOB and /NAME are mutually exclusive) (default: /NAME=xxxxext where xxxx are the executing user initials ext is the job and extension (/JOB)) /TID - Specifies where the file is to be printed -- no remote printers are

currently supported

(default: print at Central Site)

How it works: The CDC name of the output file is created from the /NAME or /JOB qualifier.

The output in your VAXfile is placed in the SYS\$QFT queue for transfer the HYPERchannel to the Mass Storage System (MSS) flagged for the node you requested.

Every 5 minutes or so, the queue transfer program on MFN checks for jobs coming to it and places them into their requested queues.

Remarks: CDC jobs may not have tabs or certain special characters. If /ASCII is not used, lower case will be folded into upper case. You may use RUN VSYS:CMP2FOR to remove tabs and change <FF> in column 1 to '1' before using QPRINT; the DETAB command may be used to remove tabs. Special characters not recognized by CDC will be converted to blanks by CDC.

The file must have Fortran carriage control.

Since RM1 is no longer available, use APRINT or CAPRINT to print in Annapolis.

Similar commands: COS: DISPOSE NOS: ROUTE

Examples: @VSYS:CMP2FOR myprog.lis ^-- prepare compilation listing for printing QPRINT myprog.lis MFN /NAME=xxxxABC /DELETE ^-- xxxx is the user initials; MYPROG.LIS will be deleted after is has been sent = = = = QPRINT /ASCII myprog.out MFN /JOB=1 ^-- print at with jobname xxxx001 in upper and lower case QSUBMIT (DTRC) Submit a job to a CDC CYBER 860 NOS input queue printing on the 860 Central Site Printer.

Syntax: QSUBMIT vaxfile node

Parameters: vaxfile - file specification of the VAXcluster file containing a CDC batch job (embedded end-of-records are indicated by a separate line containing only EOR in columns 1-3)

- node the remote node on which the job is to run -- one of: MFN - the CDC CYBER 860
- Remarks: CDC jobs may not have tabs or certain special characters; lower case will be folded into upper case. Special characters not recognized by CDC will be converted to blanks by CDC.
- How it works: The CDC job in your VAXfile is placed in the SYS\$QFT queue for transfer the HYPERchannel to the Mass Storage System (MSS) flagged for the node requested.

Every 5 minutes or so, the queue transfer program on MFN checks for jobs to it and places them into its input queue. The output is on a CDC central site printer.

As on CDC, if you want the job's output to be sent somewhere, then

ROUTE, OUTPUT, DC=PR, TID=<tid>, DEF.

should be placed in your CDC job to cause deferred routing of the entire file to another terminal ID.

See also: QPRINT

Similar commands: COS, NOS, VMS: SUBMIT NOS/VE: JOB; SUBMIT

Examples: QSUBMIT myfile.cdcjob MFN ^-- submits the CDC job in MYFILE.CDCJOB to the CDC CYBER 860 (MFN) NOS input queue with jobname from the job's JOB statement

RCAUDIT (DTRC) Create and submit a job to audit Cray files. Syntax: RCAUDIT cpw lo pdn id own acn sz wait Parameters: cpw - your Cray password - list option ([S], A, B, L, N, P, R, T, X) 10 0 - use the default pdn - the file to be audited 0 - use the default (note: RCAUDIT cannot specifically audit file "0") (default: all files) id - the ID for the file 0 - null IDown - other owner's files 0 - use the default (default: your files) acn - restrict audit to this account number 0 - use the default (default: all account numbers) - restrict audit to files larger than this sz many words 0 - use the default (default: all files) wait - WAIT - wait for the job to complete, display, delete the .CPR file (synonyms: YES, TRUE) other - do not wait (Cray job creates file RCAUD.CPR) Note: If P1 and P2 are both specified in the execute line, defaults are used for all other unspecified parameters. Remarks: Any existing file RCAUD.CPR is deleted before the Cray job is submitted. This procedure creates and deletes all versions of file R\$C\$A\$U\$D.JOB. See also: Appendix C: AUDIT Similar commands: NOS: BEGIN, AUDIT VMS: DIRECTORY

Examples: RCAUDIT mycraypw ^-- short audit of all my files (don't wait) RCAUDIT mycraypw 0 0 0 0 0 0 WAIT ^-- same (wait for completion) RCAUDIT mycraypw x ^-- "X" audit of all my files ----RCAUDIT mycraypw x "A-" ^-- "X" audit of all files starting with "B" (the "" are needed because the "-" Cray wildcard is the VMS end-of-line continuation character) _ _ _ _ _ RCAUDIT mycraypw 0 0 0 0 1222233344 100000 ^-- Short audit of all my files larger than 100000 words under Job Order Number 1-2222-333-44 - - - - -RCAUDIT mycraypw x 0 0 abcd ^-- "X" audit of all ABCD's files (that I have permission to see) RCDELETE (DTRC) Delete a Cray permanent file. Syntax: RCDELETE pw pdn id ed m [wait] Parameters: pw - your Cray password pdn - the file to be deleted - the ID for the file id - the edition ed n - a specific edition +n - delete n highest editions -n - keep n highest editions ALL - delete all editions (default: delete the highest edition) - mainenance control word m wait - WAIT - wait for the job to complete, display, delete the .CPR file (synonyms: YES, TRUE) other - do not wait (Cray job creates file RCDEL.CPR) Remarks: Any existing file RCDEL.CPR is deleted before THE Cray job is submitted.

RCGET

Page D-25

This procedure creates and deletes all versions of file R\$C\$D\$E\$L.JOB. See also: Appendix C: DELETE Similar commands: COS: DELETE, PDN= NOS, VMS: PURGE RCDELETE mycraypw abcde Examples: ^-- delete the highest edition of file ABCDE (don't wait) - - - - -RCDELETE mycraypw abcde 0 0 0 WAIT ^-- delete the highest edition of file ABCDE (wait for completion) RCDELETE mycraypw abcde zyx +3 ijk ^-- delete the high 3 editions of file ABCDE with ID=ZYX and maintenance control word IJK - = = = = RCDELETE mycraypw abcde zyx -2 ijk ^-- keep the high 2 editions of file ABCDE with ID=ZYX and maintenance control word IJK x = = = RCDELETE mycraypw abcde 0 ALL (DTRC) Create and submit a job to get a Cray permanent dataset and save it as a VAX/VMS permanent file. RCGET cpw VAXfile pdn id df ed r wait Syntax: Parameters: срw - your Cray password VAXfile - the VAX filespec for the file 0 - use the default .ext - "<pdn>.ext" (default: "<pdn>.") pdn - the Cray file to be fetched 0 - use the default (default: first 15 characters of the VAXfilename) (note: VAXfile and pdn may not both be "0") - the ID of the file i d 0 - null ID (default: all files) df - data format (BB, CB, TR) 0 - use the default

(default: CB)

ed - edition number 0 - use the default (default: the highest edition) - read control word r 0 - use the default (default: no read control word) wait - WAIT - wait for the job to complete, display, delete the .CPR file (synonyms: YES, TRUE) other - do not wait (Cray job creates file RCGET.CPR) (default: nowait) Remarks: Any existing file RCGET.CPR is deleted before THE Cray job is submitted. This procedure creates and deletes all versions of file R\$C\$G\$E\$T.JOB. See also: Similar commands: COS: DISPOSE Examples: RCGET mycraypw zyx.FOR abcde ^-- get Cray file ABCDE as VAX/VMS file ZYX.FOR (don't wait) = = = = = RCGET mycraypw zyx.FOR abcde 0 0 0 0 WAIT ^-- same (wait for completion) - - - - -RCGET mycraypw "" abcde ^-- get Cray file ABCDE as ABCDE. - - - - -RCGET mycraypw .FOR abcde ^-- Get Cray file ABCDE as ABCDE.FOR ----RCGET mycraypw zyx.out abcde qrs 0 3 0 myreadcw ^-- get Cray file ABCDE, ID=QRS, ED=3 with read controlword as ZYX.OUT - - - - -RCGET mycraypw plot.out plotout 0 BB 0 myreadcw ^-- Get Cray file PLOTOUT with DISSPLA output

RCSAVE (DTRC) Create and submit a job to save a VAX/VMS file as a Cray permanent dataset. Syntax: RCSAVE cpw VAXfile pdn id df pam m wait Parameters: cpw - your Cray password VAXfile - the VAX filespec for the file 0 - use the default .ext - "<pdn>.ext" (default: "<pdn>.") - the Cray file to be fetched pdn 0 - use the default (default: first 15 characters of the VAXfilename) (note: VAXfile and pdn may not both be "0") - the ID for the file id 0 - null ID(default: all files) df - data format (BB, CB, TR) 0 - use the default (default: CB) - public access mode pam E - execute only M - maintenance only N - no public access R - read only W - write only 0 - use the default (e.g., R:W gives read and write permission) (default: N) - maintenance control word m 0 - no maintenance control word wait - WAIT - wait for the job to complete, display, delete the .CPR file (synonyms: YES, TRUE) other - do not wait (Cray job creates file RCSAV.CPR) Note: If P1 and P2 are both specified in the execute line, defaults are used for all other unspecified parameters.

Remarks: An LO=X audit is done for file <pdn>. Any existing file RCSAV.CPR is deleted before The Cray job is submitted. This procedure creates and deletes all versions of file R\$C\$S\$A\$V.JOB. See also: Similar commands: COS: VMS: RCSAVE mycraypw abcde.fgh zyx 0 0 r Examples: ^-- make my VAX/VMS file ABCDE.FGH a permanent dataset on the Cray with the name ZYX and having world read access (don't wait) RCSAVE mycraypw abcde.fgh zyx 0 0 r 0 WAIT ^-- same (wait for completion) - - - - -RCSAVE mycraypw abcde.fgh "" "" r ^-- make my VAX/VMS file ABCDE.FGH a permanent dataset on the Cray with the name ABCDE and having world read access (the "" are place holders) - - - - -RCSAVE mycraypw abcde.fgh zvx qrs 0 0 ijk ^-- make my VAX/VMS file ABCDE.FGH a permanent dataset on the Cray with the name ZYX, ID=QRS and maintenance control word IJK (no permissions) - - - - -RCSAVE mycraypw plot.out plotout 0 0 BB ^-- after RCGETting a binary blocked file (perhaps DISSPLA output),

send it back to the Cray

Page D-29

RCSUBMIT (DTRC) Submit a job to the Cray from any CCF VAXcluster node.

Syntax: RCSUBMIT job file [password]

Parameters: job_name - the name of the file containing your Cray job

> password - your VAXcluster login password -for security, you may wish to omit this and be prompted for it (this is not used if you are on a node which is connected to the Cray)

Remarks: RCSUBMIT works from any node of the VAXcluster. CRAY SUBMIT works only on a node which connects directly with the Cray.

> RCSUBMIT and CRAY SUBMIT require an ACCOUNT statement in the job file. CSUBMIT does not, and ignores it if it is present. If you normally use CSUBMIT to submit your jobs, you should not use RCSUBMIT (or CRAY SUBMIT).

See also: CRAY SUBMIT; CSUBMIT

Similar commands: COS, VMS: SUBMIT NOS: ROUTE

Examples: RCSUBMIT crayjob myclustrpw ^-- from a node not connected to the Cray

where file CRAYJOB.JOB contains:

JOB, JN=test. ACCOUNT, AC=jobordrno, US=abcd, UPW=mypw. DISPOSE, DN=\$OUT, TEXT='node::', DEFER. ^-- at end-of-job, \$OUT will be put into file node::TEST.CPR FETCH, DN=test, TEXT='node::test.for' ^-- fetch program from node NODE CFT, I=test, L=0. FETCH, DN=FT05, TEXT='node::test.dat'. ^-- fetch data from node NODE SEGLDR.GO. DISPOSE, DN=FT04, TEXT='node::test.out'. ^-- send another output file of the program to node NODE - - - - -CRAY SUBMIT crayjob ^-- from a node connected to the Cray (you can use RCSUBMIT, but this is faster)

*** Cray Station Commands ***

The VAX/VMS Cray Station provides the VMS user with access to the CRAY X-MP. The Cray Station is accessed via two commands: CRAY (all Station commands except interactive) and CINT (interactive access and a subset of the Station commands). The CRAY prompt is CRAY>; the CINT prompt is Cint>.

The following discussion of the Cray station commands is derived from the on-line helps for the CRAY and CINT commands. Type "CRAY HELP" or "CINT /HELP" at the DCL level, or "HELP" at the CRAY> or Cint> prompt for more detailed information.

CRAY Enter the Cray context utility or executes a single station command when that command is supplied as a parameter. Syntax: \$ CRAY [station command] /BREAKTHROUGH /REFRESH Parameters: station_command - a single Cray station command to be executed omitted - you remain in Cray context until you enter EXIT Qualifiers: /BREAKTHROUGH - a display refresh occurs during command input (valid for refresh mode only) (default: /NOBREAKTHROUGH) /REFRESH - enable display refreshing in a split screen Cray context (requires DEC CRT option enabled) /NOREFRESH - standard teletype environment (defaults: /REFRESH (VT100-type terminals) /NOREFRESH (non-VT100 terminals))

See also: CINT

Similar commands: NOS: ICF

Examples: \$ CRAY

Page D-31

CINT From the DCL level, enter Cray interactive including a subset of the Cray context commands. Syntax: \$ CINT /HELP /JN=jobname /MML=mm1 /UPPERCASE /PLAY=play_file /US=username Qualifiers: /H - display help information without having to enter Cray interactive /J - the interactive job name (first 7 characters used) /M - maximum message length /UP - controls whether input is converted to uppercase (default: /NOUPPERCASE) /P - the play file to be run /US - the username (1-15 characters)Remarks: See also: CRAY Similar commands: NOS: ICF Examples: \$ CINT Cray Jobname: myjob Cray Username: AMDS !ACCOUNT,....

1

** Cray Context Commands **

The following commands may be executed at the CRAY> prompt. This identified with (CINT) may also be executed at the Cint> prompt.

| \$ | Create a temp commands. | oorary VMS subprocess, allowing you to enter DCL |
|----|----------------------------|--|
| | Syntax: | <pre>\$ [dcl_command]</pre> |
| | Parameters: | dcl_command - any DCL command |
| | Remarks: | Since a subprocess is created, any logical names or process resources created in the subprocess will not be available from the main process. |
| | | To return to Cray context, type LOGOUT. |
| | Similar comma | ands: NOS ICF: |
| | Examples: | \$ show users |
| | - | |
| + | Display the r | next page of information in Cray context. |
| | Syntax: | + |
| | Similar comma | ands: NOS ICF: |
| | Examples: | CRAY> + |
| _ | Display the | previous page of information in Cray context. |
| | | nevious page of information in clay context. |
| | Syntax: | - |
| | Similar comma | ands: NOS ICF: |
| | Examples: | CRAY> - |
| 0 | Execute an ir | ndirect station command file in Cray context. |
| | Syntax: | @file_spec |
| | Parameters: | file_spec - a VMS file containing station commands |
| | Remarks: | "@" is a synonym for the PLAY command. |
| | See also: | PLAY |

Page D-33

Similar commands: NOS JCF: /PLAY

Examples: CRAY> @station.COM

ABORT (CINT) Interrupt the current interactive Cray job step and return control to the COS Control Statement Processor (CSP). CSP will then issue the "!" prompt. Any COS output queued for the terminal will be displayed before the prompt is issued.

Syntax: ABORT

See also: DROP, KILL

Similar commands: NOS ICF: ABORT

Examples: CRAY> ABORT

ATTACH (CINT) Redirect COS interactive terminal output to an alternate device.

Syntax: ATTACH [alt_device] /CHAR=(char,pos) /MRS=max_rec_size /OFF /ON

- Parameters: alt_device the alternate device omitted - the current output device
- Qualifiers: /CHAR route entire record to attached device if character <char> is in position <pos> of the current Cray interactive output record
 - /MRS route entire record (no carriage control)
 to attached device if the length of the
 current Cray interactive output record
 exceeds max_rec_size

 - /ON enable routing of Cray interactive records to an attached device

Default: /ON

Remarks: The device specified must not be in use and can be any device that accepts record I/O, such as a graphics terminal.

Similar commands: NOS ICF: /CONNECT

ATTENTION (CINT) Interrupt current interactive Cray job step and enter reprieve processing.

Syntax: ATTENTION

See also: ABORT

Remarks: If reprieve processing not specified, same as ABORT.

Similar commands: NOS ICF: /ATTENTION

Examples: CRAY> ATTENTION

BYE (CINT) Terminate an interactive session and, optionally, the COS interactive job.

Syntax: BYE /ABORT /SAVE

Qualifiers: /ABORT - terminate the associated COS interactive job

/SAVE - the associated COS interactive job remains active and output is saved; if the job reaches a COS threshold for output messages or requires input, the job is suspended; the terminal can be reconnected to the COS interactive job by the INTERACTIVE command

Remarks: BYE /ABORT is equivalent to QUIT.

See also: QUIT

Similar commands: NOS ICF: /BYE, /LOGOFF, /QUIT

Examples: CRAY> BYE

CLEAR Terminate any display command and clears the display portion of the screen.

Syntax: CLEAR

Remarks: CLEAR is only available when Cray context is in refresh mode.

Examples. CRAY> CLEAR

Page D-35

COLLECT (CINT) Store COS interactive output in a VMS file.

Syntax: COLLECT file spec /ECHO /OFF /ON

Parameters: file_spec - the VMS file to receive the COS interactive output

Qualifiers: /ECHO - display the output generated at the terminal as well as the VMS file /NOECHO - do not echo the generated output at the terminal; only into the VMS file (default: /ECHO)

> /OFF - stop writing COS job output to a VMS file and close the VMS file (ignore other qualifiers

/ON - write COS job output to a VMS file
 (default: /ON)

Remarks: COLLECT can be used before the interactive job is initiated.

Examples: CRAY> COLLECT mycosfile.out

COMMENT Insert comments into an indirect station command file stream.

Syntax:COMMENT stringParameters:string - any textRemarks:The comment line can be 256 characters long,
including "COMMENT".See also:@, MESSAGE

Similar commands: NOS ICF: /* Examples: COMMENT This is a comment

CONTROL_C (CINT) CTRL-C (^C) performs the same function of the attention command.

Syntax:^C<-- ^ is the CTRL key</th>Remarks:Brings you back to the DCL prompt.See also:ABORT; ATTENTIONExamples:! ^C\$<-- leave Cray session abnormally</td>\$<-- you are back at the DCL level</td>

CONTROL_O (CINT) CTRL-O (^O) performs the same function as the discard command.

Syntax: ^0 <-- ^ is the CTRL key Remarks: ^0 toggles output on and off until the next Cray prompt. See also: DISCARD

Examples: ! ^O

CONTROL_Z (CINT) CTRL-Z (^Z) exits the current processing mode.

Syntax: ^Z <-- ^ is the CTRL key

Remarks: In response to the Cray context prompt (CRAY>), you are returned to DCL; in a Cray interactive session, you are returned to command mode. While you are being prompted for command parameters, CTRL-Z cancels the command.

CTRL-Z also terminates the execution of an indirect station command file.

See also:

6

| Examples: | ! ^Z | < leave Cray session |
|-----------|------------|---------------------------------|
| | CRAY> QUIT | < terminate Cray session |
| | CRAY> ^Z | < terminate Cray context |
| | \$ | < you are back at the DCL level |

DATASET Test for the existence of a COS permanent dataset. Syntax: DATASET pdn /ID=id /ED=ed /OV=owner Parameters: pdn - name of PDS Qualifiers: /ID= - id of the dataset (1-8 characters) (default: null) /ED= - edition number of the dataset (1-4095) (default: current highest edition number) /OV= - owner of the dataset Examples: DATASET,myfile.

Page D-37

DELAY Suspend execution of an indirect station command file for a specified period of time.

Syntax: DELAY seconds

Parameters: seconds - suspension time in seconds

Examples: DELAY 20

DISCARD (CINT) Discard all output from a COS interactive session until the next COS prompt is issued.

Syntax: DISCARD

See also: ^0

Similar commands: NOS ICF: /DISCARD

Examples: DISCARD

DROP Terminate a COS job and return the associated output dataset. COS job execution enters reprieve processing after the next COS EXIT control statement.

Syntax: DROP jsq

Parameters: jsq - job sequence number

Remarks: Use STATUS to obtain the job sequence number (COS jsq).

KILL terminates the job immediately; DROP continues processing after an EXIT statement.

See also: ABORT, KILL

Examples: \$ CRAY CRAY> STATUS CRAY> DROP 9876

EOF (CINT) Sends an end-of-file record to a connected COS interactive job.

Syntax: EOF

Remarks: EOF is normally required to terminate COS file input from the terminal.

Similar commands: NOS ICF: /EOF

Examples: CRAY> EOF

Page D-38

÷

Rev0 90/09/30

EXIT (CINT) Leave Cray context command mode and return to DCL. Syntax: EXIT ^Z Remarks: EXIT will close the file specified in a RECORD command, if it is still open. See also: RECORD Similar commands: NOS ICF: /EXIT Examples: CRAY> EXIT HELP (CINT) Display help information on the Cray station commands. Syntax: HELP [station_command] Parameters: station_command - a specific command for which help is desired omitted - a list of all available commands Similar commands: NOS ICF: /HELP \$ CRAY HELP Examples: - - - - - -CRAY> HELP **CRAY> HELP CINT** ISTATUS (CINT) Get the status of your COS interactive job (with CPU time used and the last COS logfile message). Syntax: ISTATUS See also: JSTAT, STATUS Examples: ISTATUS JOB Display the status of a specific COS job. JOB jobname /JSQ=jsq Syntax: Parameters: jobname - the COS job name Qualifiers: /JSQ= - the job sequence number from which to start the search for the job Similar commands: NOS ICF: /STATUS Examples: JOB myjob4

Page D-39

JSTAT Display the status of a specific job and its related tasks. Syntax: JSTAT jsq /[NO]CYCLE /[NO]TRANSLATE Parameters: jsq - the job sequence number Qualifiers: /CYCLE - cycle the display refresh through all the available information /NOCYCLE - display only the current page until you enter "+" or "-" (default: /NOCYCLE) /TRANSLATE - display the terminal ID field in the VMS UIC equivalent /NOTRANSLATE - display it in the station internal form (default: /TRANSLATE) Remarks: Use STATUS to obtain the COS job sequence number (jsq). ISTATUS, STATUS See also: Similar commands: NOS ICF: /STATUS Examples: JSTAT Delete a job from the input queue, or immediately terminate an KILL executing job, or delete the job's output dataset from the output queue. Syntax: KILL jsq Parameters: jsq - the job sequence number Remarks: Use STATUS to obtain the COS job sequence number (j**sq**). KILL terminates the job immediately; DROP continues processing after an EXIT statement. See also: ABORT, DROP Similar commands: NOS ICF: /ABORT Examples: CRAY> STATUS CRAY> KILL 9876

LOGFILE Provides access to the station logfile messages.

- Syntax: LOGFILE [file_spec] /ACQUIRE /ALL /BEFORE=time /DISPOSE /ERROR /INTERACTIVE /JOB /MASTER /NETWORK /NODE=nodename /[NO]NOTIFY /OPERATOR /OUTPUT=file_spec /PRINT /RELEASE /SINCE=time /SUCCESS /STMSG /TRANSLATE
- Parameters: file_spec An alternate station logfile to be displayed
- Qualifiers: /ACQU display ACQUIRE and FETCH messages
 - /ALL display all messages
 - /BEFO display messages from before a specified
 time
 - /DISP display DISPOSE messages
 - /ERRO display error messages
 - /INTE display interactive processing messages
 - /JOB display job submission messages
 - /MAST display COS master operator messages
 - /NETW display DECnet messages (all nodes)
 - /NODE= display DECnet messages (one node)
 - /NOTI you will be notified an asychronous LOGFILE operation is performed (requires /RELEASE) (default: /NONOTIFY)
 - /OPER display operator messages

 - /RELE close the existing logfile and create a
 new version

/SINC= - display messages since a specified time

Page D-41

- /SUCC display success, warning, and informational messages
- /STMS display COS station messages and associated replies
- /TRAN display terminal ID field (TID) as the VMS UIC equivalent /NOTR - display TID in the station internal form (default: /TRANSLATE)
- Examples: CRAY> LOGFILE jobname.LOG /SINCE=09:15
- LOOP Restart execution of an indirect station command file at the beginning.

Syntax: LOOP

Remarks: CTRL-Z must be issued to terminate looping.

Examples: CRAY> LOOP

MESSAGE Send a message to the COS job logfile.

Syntax: MESSAGE string /JN=jobname /JSQ=jsq

Parameters: string - the message text (for embedded blanks, enclose in quotes ("...")

Qualifiers: /JN= - the name of the COS job to receive the message (requires /JSQ)

/JSQ= - the job sequence number of the COS job
 to receive the message

See also: COMMENT

Similar commands: NOS ICF: /*

Examples: MESSAGE This is a message

PAUSE Suspend execution of an indirect station command file.

Syntax: PAUSE

Remarks: Control passes to the terminal, where you can terminate the command file by entering a command or resume it by entering a null line (<RET>).

Examples: PAUSE

PLAY (CINT) Execute an indirect station command file in Cray context.

Syntax: PLAY file_spec

Parameters: file_spec - a VMS file containing station commands

Remarks: PLAY files cannot themselves contain other (embedded) PLAY commands.

"@" is a synonym for the PLAY command.

Similar commands: NOS ICF: /PLAY

Examples: CRAY> PLAY station.COM

QUIT (CINT) Terminate a Cray interactive session and the corresponding COS interactive job.

Syntax: QUIT

Remarks: QUIT is the equivalent of BYE /ABORT.

See also: BYE

Similar commands: NOS ICF: /BYE, /LOGOFF, /QUIT

| Examples: | !^Z | < | leave Cray session | | | | |
|-----------|------------|---|----------------------------|--|--|--|--|
| | CRAY> QUIT | < | terminate the Cray session | | | | |
| | CRAY> EXIT | < | terminate the Cray station | | | | |

RECORD Start or stop the recording of terminal input to a file while in Cray context for later use with the PLAY or @ commands.

Syntax: RECORD [file_spec] /ON /OFF

Parameters: file_spec - the file into which terminal input is to be recorded

- Qualifiers: /ON start command recording (file_spec required)
 - /OFF ~ end command recording
 (default: /ON)
- Remarks: Exiting Cray context automatically issues a RECORD/OFF.
- Examples: RECORD station.com /ON ... RECORD /OFF

Page D-43

REMOVE Delete entries in the dataset staging queue.

Syntax: REMOVE queue id /LOCKED /SPOOL /STAGE

Parameters: queue_id - an 8-character hexadecimal number from the SHOW QUEUES display (leading zeros can be omitted)

- Qualifiers: /LOCKED controls whether or not locked entries are removed (default: /NOLOCKED)

 - /STAGE remove an entry in the Cray staging
 queue
- RERUN Immediately end the processing of a COS job and put it back into the input queue.

Syntax: RERUN jsq

Parameters: jsq - the job sequence number

Remarks: The job input dataset is saved and all output datasets associated with the job are deleted. The job input dataset is then rescheduled so the job can be rerun. No action is taken if the job execution is complete or if COS determines the job cannot be rerun.

Use STATUS to obtain the COS job sequence number (jsq).

SAVE Stages a VMS file to COS disk storage.

Syntax: SAVE file_spec /DELETE /DF=d /ED=ed /ID=id /MN=mn /PDN=pdn /RD=rd /RT=rt /US=us /WT

Parameters: file_spec - the file to be staged

- - /DF= dataset format: CB, BB, or TR (defa.iit: CB)

Page D-44

/ID= - identification (1-8 alphameric chars) /MN =- maintenance control word /PDN= - dataset name to be used (converted to uppercase) (default: the input file name) /RD=- read permission control word /RT= - the retention period, in days /US= - the COS username /WT= - the write permission control word Examples: SAVE myfile.dat /PDN=mydata /US=ABCD SET TERMINAL Define the terminal working environment. SET TERMINAL FORTRAN SET TERMINAL NOFORTRAN Specify whether the terminal is to interpret output records from a COS interactive session as having FORTRAN carriage control. Default: NOFORTRAN SET TERMINAL INFORM SET TERMINAL NOINFORM Enable/disable the sending of station messages to the user logged on to VMS at a VAX terminal. Default: NOINFORM SET TERMINAL PAGE SET TERMINAL PAGE=lines SET TERMINAL NOPAGE Specify the number of lines of output before a page break. Default: NOPAGE Default for lines: determined by the scroll setting SET TERMINAL REFRESH SET TERMINAL REFRESH=seconds <-- integer 0-60</pre> SET TERMINAL NOREFRESH

> REFRESH provides a split-screen Cray context environment and is supported only on terminals with the DEC_CRT attribute. NOREFRESH provides a line-by-line Cray context environment.

90/09/30 Rev0 Defaults: REFRESH (VT100-type terminals) NOREFRESH (non-VT100-type terminals) SET TERMINAL SCROLL=lines Changes the Cray context window size.

"lines" is the size of the command area (bottom window) and must be an integer from 3 to 13.

Default for lines: 4

SET TERMINAL WIDTH=80 SET TERMINAL WIDTH=132 Changes the width of the terminal within Cray context.

Default: 80

SHOW QUEUES Display entries in the dataset staging queue.

| Syntax: | SHOW | QUEUES | /ACQUIRE /ALL | | /CYCLE | /JOB |
|---------|------|--------|---------------|---------|--------|-------|
| | | | /NODE=no | de_id | /OWNER | /SAVE |
| | | | /STAGE | /TRANSI | LATE | |

Qualifiers: /ACQU - display all entries originating from COS (ACQUIRE or FETCH) (default: /ALL)

- /ALL display all entries
 (same as /ACQUIRE/JOB/SAVE)
 (default: /ALL)
- /CYCL cycle the display refresh through all
 the available information
- /NOCYC display only the current page until you
 enter "+" or "-"
 (default: /NOCYCLE)

- /OWNER display only your entries

/STAGE - display all Cray staging entries

Page D-45

SNAP

/TRAN - display the terminal ID field in the VMS UIC equivalent /NOTRA - display it in the station internal form (default: /TRANSLATE) Remarks: The following fields are displayed: . Position in the staging queue (L is a locked entry i.e., one that is being processed) . Request type (JB=job, AC=acquire/fetch, SV=save) . Queue ID for use in the REQUEUE and RELEASE commands . VAX username of entry owner . Dataset transfer name (job name or dataset name) . Dataset terminal ID (TID) Similar commands: NOS ICF: /STATUS SHOW QUEUES /OWNER Examples: ^-- display all your entries Copy the current contents of the display region into a VMS file. SNAP file spec / [NO] ESCAPE Syntax: Parameters: file_spec - VMS file to receive the snapshot Qualifiers: /ESCAPE - retain escape sequences /NOESCAPE - remove escape sequences (default: /NOESCAPE) Remarks: In line-by-line mode, the last display requested is recorded. SNAP snap.job123 Examples:

STATCLASS Display the current COS job class structure.

| Syntax: | STATCLASS | /[NO]CYCLE |
|-------------|-----------|--|
| Qualifiers: | /CYCLE | cycle the display refresh through all the available information |
| | /NOCYCLE | display only the current page until you enter "+" or "-" (default: /NOCYCLE) |

Similar commands: NOS ICF: /ICFSTATUS, /STATUS

Examples: STATCLASS

Page D-47

.

ł

| STATUS | (CINT) Disp | lays the COS system status. |
|--------|-------------|---|
| | Syntax: | STATUS /ALL /CLASS=class_id /CYCLE /EXECUTING /HOLD /ID=mainframe_id /INPUT /NODE=node_id /OUTPUT /OWNER /RECEIVING /SENDING /TRANSLATE /VAX |
| | Qualifiers: | <pre>/ALL - display all COS jobs /CLAS= - display jobs and datasets of a specific job class (default: /ALL)</pre> |
| | | <pre>/CYCL - cycle the display refresh through all available information</pre> |
| | | /NOCY - display only the current page until you enter "+" or "-" (default: /NOCYCLE) |
| | | /EXEC - display the execution queue status (default: /EXECUTION) |
| | | /HOLD - display COS datasets in the hold queue |
| | | <pre>/ID= - display jobs and datasets originating from a specific mainframe</pre> |
| | | /INPU - display the input queue statur |
| | | <pre>/NODE= - display the entries for a specific DECnet</pre> |
| | | /OUTP - display the output queue status |
| | | /OWNE - display only your jobs and datasets |
| | | <pre>/RECE - display the Cray receiving queue status (default: /RECEIVING)</pre> |
| | | <pre>/SEND - display the Cray sending queue status (default: /SENDING)</pre> |
| | | /TRAN - display terminal ID field (TID) as the VMS UIC equivalent |
| | | /NOTR - display TID in the station internal form (default: /TRANSLATE) |
| | | /VAX - display only COS jobs related to this VAX/VMS station (or network of stations) |
| | See also: | ISTATUS, JSTAT |

Similar commands: NOS ICF: /STATUS

Examples: STATUS

Rev0 90/09/30

SUBMIT Stage a VMS file to the COS input queue. Syntax: SUBMIT file spec /AFTER=time /EOF=eof /PRINT SUBMIT f1, f2,... /AFTER=time /EOF=eof /PRINT Parameters: file spec - single VMS file with a complete COS job fl,f2,... - two or more files to be combined to create a complete COS job Qualifiers: /AFTER= - specify when the job is to be sent to the Cray /EOF= - specify what represents an end-of-file (e.g., /EOF="E 0 F") (default: /EOF="/EOF") /PRINT - print the job's output file on COS job completion /NOPRINT - put the COS job's output into your VMS file COS_jobname.CPR (default: /NOPRINT) The file must contain a COS job. By default, the Remarks: job's output (including the dayfile) is sent to the originating directory. CSUBMIT; RCSUBMIT See also: Similar commands: NOS: CSUBMIT Examples: CRAY> SUBMIT myjob1 -or-\$ CRAY SUBMIT myjob1 = = CRAY> SUBMIT myjob2, myprog2.for, mydata2.dat -or-\$ CRAY SUBMIT myjob2, myprog2.for, mydata2.dat SUPPRESS (CINT) Suppress the echoing of the next typed input line. Syntax: SUPPRESS

Examples: Cint> SUPPRESS

Page D-49

SWITCH Set or clear COS job sense switches.

Syntax: SWITCH jsq ssw /OFF SWITCH jsq ssw /ON

Parameters: jsq - the COS job sequence number

ssw - the sense switch number (1-6)

Qualifiers: /OFF - turn switch <ssw> off

/ON - turn switch <ssw> on

Remarks: These switches can be used for program synchronization on the Cray.

Examples: CRAY> STATUS <--- to get the jsq CRAY> SWITCH 9876 3 /ON <--- turn on switch 3

***** Appendix E *****

*** References ***

The following manuals describe various features of the Cray, DEC and CDC systems.

** Cray **

| SR-0009 | Fortran (CFT) Reference Manual |
|---------|---------------------------------------|
| SR-0011 | COS Version 1 Reference Manual |
| SR-0013 | UPDATE Reference Manual |
| SR-0018 | CFT77 Reference Manual |
| SV-0020 | DEC VAX/VMS Station Reference Manual |
| SR-0035 | CDC NOS Station Reference Manual |
| SR-0039 | COS Message Manual |
| SR-0060 | Pascal Reference Manual |
| SR-0066 | SEGLDR Reference Manual |
| SR-0113 | Programmer's Library Reference Manual |
| | |

** DEC **

| AA-001AE-GZ | DCL Dictionary |
|--------------|--|
| AA-LA16A-TE. | EDT Reference Manual |
| AA-LA62A-TE | EVE Reference Manual |
| AA-D034E-TE | VAX Fortran Language Reference Manual |
| AA-LA98A-TE | VAX/VMS User's Manual |
| | Introduction to VAX/VMS by Terry Shannon |

** CDC NOS **

| 60460420 | NOS Full Screen Editor |
|------------|---|
| CMLD-88/15 | CDC NOS Full Screen Editor (FSE) User's Guide |
| 60459680 | NOS 2 Reference Set Volume 3: System Commands |

** CDC NOS/VE **

| 60464018 | NOS/VE Commands and Functions Quick Reference |
|----------|---|
| 60464015 | NOS/VE File Editor |
| 60485913 | Fortran Version 1 for NOS/VE |
| 60464012 | Introduction to NOS/VE |
| 60464014 | NOS/VE System Usage |



•

Page E-2

** General **

CMLD-87-07 Fortran 77 Extensions - A Comparison CISD-90/01 Computer Center Reference Manual, Volume 1: Cray, MSS, DEC

(this manual)

CISD-90/02 Computer Center Reference Manual, Volume 2: CDC

.

Cray

C1

e .+

٠

Page F-1

```
***** Appendix F *****
*** CCF Computer Systems ***
```

| Computer: | CRAY X-MP/216 |
|-------------------|---|
| Front-ends: | DEC VAXcluster (station code version 4.01), |
| | CDC CYBER 180/860 (N1) |
| Links to: | Mass Storage System (N1) |
| Operating system: | COS level 1.17 |
| Services: | batch, interactive |
| Schedule: | 24 hours a day, 7 days a week, except a few hours |
| | Tuesday and Thursday mornings for maintenance |
| Location: | Central site |

Page F-2

Rev0 90/09/30

DEC VAXcluster DT3 (V3) VAX 8550 Computer: Links to: CRAY X-MP (C1); CDC CYBER 180/860 with MSS (N1/MFN); DECnet to NAVSEA (SEAHUB, etc.) Operating system: VMS 5.3-1 Services: batch, interactive Schedule: 24 hours a day, 7 days a week, except a few hours Thursday morning for maintenance Location: Central site Network address: 130.46.1.12 (dtvms3.dt.navy.mil) DT4 (V4)VAX 8550 Computer: Links to: CRAY X-MP (C1); CDC CYBER 180/860 with MSS (N1/MFN); DECnet to NAVSEA (SEAHUB, etc.) Operating system: VMS 5.3-1 Services: batch, interactive Schedule: 24 hours a day, 7 days a week, except a few hours Thursday morning for maintenance Location: Central site Network address: 130.46.1.10 (dtvms.dt.navy.mil or dtvms4.dt.navy.mil) Secure DEC VAX SECURE Computer: VAX 6410 CRAY X-MP (C1) Links to: Operating system: VMS 5.3-1 secure batch, secure interactive Services: 24 hours a day, 7 days a week, except a few hours Schedule: for maintenance Location: Central site Control Data Corporation MFN (N1) CDC CYBER 180/860A with Mass Storage System Computer: Cray Station ID: N1 Links via NOS to: CRAY X-MP (C1) Links via NOS from: CRAY X-MP (C1), DEC VAXcluster Operating systems: dual state with . NOS version 2.7.1 level 716 . NOS/VE version 1.5.1 level 739 Services: trillion-bit storage, local and remote batch, interactive Schedule: 24 hours a day, 7 days a week, except a few hours for maintenance Location: Central site Network address: 130.46.1.16 (cdc860.dt.navy.mil)

Page F-3

OASYS (Office Automaticn SYStem) composed of: OASYS Computer: Sequent S27 Links to: Mass Storage System Operating system: DYNIX v3.0.17.9 (BSD 4.2 + some 4.3 + some AT&T System V) Services: OASYS (Office Automation) Schedulat 24 hours a day, 7 days a week, except a few hours Wednesday night for backups Central site Location: Network address: 130.46.1.53 (oasys.dt.navy.mil) DTOA1 Computer: DEC VAX 11/780 Mass Storage System Links to: Operating system: Ultrix-32 OASYS (Office Automation - primarily Carderock) Services: Schedule: 24 hours a day, 7 days a week, except a few hours Thursday morning for maintenance Location: Central site Network address: 130.46.1.2 [coal.dt.navy.mil] DTRC Computer: DEC VAX 11/780 Links to: Mass Storage System Operating system: Ultrix-32 OASYS (Office Automation - primarily Carderock) Services: Schedule: 24 hours a day, 7 days a week, except a few hours Thursday morning for maintenance Location: Central site Network address: 130.46.1.3 (dtrc.dt.navy.mil) DTGA3 DEC VAX 11/780 Computer: Links to: Mass Storage System Operating system: Ultrix-32 OASYS Office Automation - primarily Annapolis) Services: 24 hours a day, 7 days a week, except a few hours Schedule: Thursday morning for maintenance Location: Central site Network address: 130.46.1.4 (dtoa3.dt.navy.mil)

Page F-4

*** *** Services and Support Accounting for Computer Services: Code 3502 (301) 227-1910 (301) 227-3043 Computer status (recorded message) Dispatch desk (301) 227-1967 Manuals (301) 227-1907 Microcomputer support Carderock: (301) 227-4901 Annapolis: (301) 267-4987 Tape Librarian (301) 227-1967 (301) 227-1907 Training User Services (Scientific and Engineering User Support Branch - Code 3511) Carderock: (301) 227-1907 Annapolis: (301) 267-3343 Stan Willner (Head) Sharon Good Mike Kass Ed Kennedy Brenda Peters Dave Sommer (Annapolis) Administrative Personnel 35 Computer & Information Services Department (G. Gray) (301) 227-1270 3501 Assistant for Technical Development and Operations (L. Minor) (301) 227-1428 (301) 227-1361 3502 Computer Department Business Office 3509 Administrative Office (D. Braxton) (301) 227-3454 351 Scientific & Engineering Systems Div. (S. Willner) (301) 227-1907 3511 S&E User Support Branch (S. Willner) (301) 227-1907 (301) 227-3642 3512 VAX/VMS Systems Branch (M. Brady) 3513 Cray/CDC Systems Branch (J. Wessel) (301) 227-1271 (301) 227-1428 353 Office Automation Systems Division (R. Yearick) 3531 Unix Systems and Programming Branch (R. Yearick) (301) 227-1428 3533 OA/Microcomputer Support Branch (P. Hayden) (301) 227-4901 (301) 267-4987 (301) 227-1184 355 Information Systems Division (E. Kearney) 3551 Business Systems Branch (B. Crum) (301) 227-1127 (301) 227-1184 3552 Special Project Branch (D. Singla) 357 Communications and Facilities Division (R. Weachter) (301) 227-1270 3571 Computer Facilities Branch (R. Weachter) (301) 227-3937 3572 Networks and Communications Branch (T. Smith) (301) 227-1400

Page G-1

***** Appendix G *****

*** Internal Data Structure ***

1. The following table summarizes word lengths on various computers:

| computer | op sys | bits/word | digi | ts/word | cha: | racters/word |
|--------------------------------|-----------------|------------|------------|------------|--------|-----------------------|
| CRAY X-MP | | 64 | 22 | octal | | 8 |
| CDC CYBER 200 CDC CYBER 180 | NOS/VE | 64 | 16 | hex | | 8 |
| CDC CYBER 180 | NOS & NOS/BE | 60 | 20 | octal | | 10 |
| CDC CYBER 170 | NOS/BE | | | | | |
| DEC VAX (when used in | Fortran | 16) 32 | - | hex hex | | 2 4 |
| | i or cran, | | _ | _ | | - |
| IBM | | 32 | 8 | hex | | 4 |
| Burroughs 7700 | | 48 | 1 2 | hex | | 6 |
| Unisys 1100 | | 36 | 1 2 | octal | 4 6 | (ASCII) (Fieldata) |

This affects the conversion of programs in four areas:

- a. The degree of precision of operations is different. Therefore, convergence factors may need to be increased or decreased in absolute value.
- b. Constants and data may need to be changed.
- c. Octal and hexadecimal constants used in masking operations are generally affected and require alteration according to their intended use.
- d. Since different computers may store a different number of characters per word, DATA statements that store a string of Hollerith characters may position the characters in different relative positions in different words. All variable formats (whether read in as data or created by the programmer) should be checkcd. Better yet, Fortran programs which store Hollerith data in INTEGER or REAL variables should be changed to use the Fortran 77 CHARACTER variables and never need to worry about this problem again. (You may have to worry about the maximum length of a CHARACTER variable, but not how it is stored.)

2. Internal representation of character data is ASCII in the CRAY X-MP and DEC VAX, Display Code in the CDC CYBER, and ASCII, EBCDIC or internal BCD in some other systems.

| CHARACTER string | machine | op sys | internal representation |
|------------------|---|---------------|---|
| ' ' (1 blank) | CRAY X-MP CDC 170 CDC 180 CDC 180 DEC VAX | NOS NOS/VE | * oct 20 hex 55 55 20 20 |
| '0' (1 zero) | CRAY X-MP CDC 170 CDC 180 DEC VAX | NOS | * oct 30 hex 33 30 30 |
| 'FILE48' | CRAY X-MP CDC 170 CDC 180 CDC 180 DEC VAX | NOS NOS/VE | * oct 46494C463438 hex 061014053743 061014053743 46494C453438 3834454C4946 (8 4 E L I F) |

* - the octal representation depends on the position in the word

| Hollerith words | machine | op sys | internal machine repres | sentation |
|-------------------|-----------|--------|---|-----------|
| <blanks></blanks> | CRAY X-MP | | 0200401002004010020040 | oct |
| | | | 2020202020202020 | hex |
| | CDC 170 | | 555555555555555555555555555555555555555 | oct |
| | CDC 180 | NOS | 555555555555555555555555555555555555555 | oct |
| | CDC 180 | NOS/VE | 2020202020202020 | hex |
| | DEC VAX | | 20202020 | hex |
| <zeroes></zeroes> | CRAY X-MP | | 0300601403006014030060 | oct |
| | | | 303030303030303030 | hex |
| | CDC 170 | | 3333333333333333333333333 | oct |
| | CDC 180 | NOS | 3333333333333333333333333 | oct |
| | CDC 180 | NOS/VE | 3030303030303030 | hex |
| | DEC VAX | | 30303030 | hex |
| FILE48 | CRAY X-MP | | 0431112304246416020040 | oct |
| | | | 46494C4534382020 | hex |
| | CDC 170 | | 06101405374355555555 | oct |
| | CDC 180 | NOS | 06101405374355555555 | oct |
| | CDC 180 | NOS/VE | 46494C4534382020 | hex |
| | DEC VAX | | 454C4946 20203834 | hex |
| | | | (ELIF 84) < | - 2 words |

- 3. The character sequence for the CRAY X-MP, DEC VAXcluster and CDC 180 (NOS/VE) is ASCII. Note that numbers precede letters for alphabetic comparisons. The character sequences for CDC 180 (NOS) at DTRC is Display Code (64-character set). CDC NOS Fortran uses the Display Code sequence (letters before numbers); CDC NOS COBOL uses the ASCII6 sequence (numbers before letters). Cray, DEC VAX and CDC NOS/VE use the ASCII sequence.
- 4. CDC NOS uses some special bit configurations in floating point arithmetic to indicate indefinite and infinite operands. These errors could be caused by referencing program areas not initialized or areas overwritten due to inadequate storage reservation. The CPU will not do any further calculation if it encounters such a number and the job will abort with an error mode 2 or 4.

| + | infinity | 3777xxxxxxxxxxxxxx oct |
|---|------------|--|
| - | infinity | 4000xxxxxxxxxxxxx |
| + | indefinite | 1777***** |
| - | indefinite | 6000xxxxxxxxxxxxx |
| | | where 'x' is any octal digit, usually 0. |

5. CDC NOS/VE uses several exponents in floating point arithmetic to indicate zero:

+ zero 0xxx, 1000 thru 2FFF hex - zero 8xxx, 9000 thru AFFF

6. CDC NOS/VE uses special exponents in floating point arithmetic to indicate indefinite and infinite operands:

+ infinity D000 thru EFFF hex
- infinity 5000 thru 6FFF
+ indefinite 7xxx
- indefinite Fxxx
where 'x' is any hexadecimal digit

7. The word format of integers and floating point numbers differs on the various computers.

| CRAY X-MP | integer | floating point | |
|---|---|-------------------------|-----|
| 1, 1.0 | 000000000000000000000000000000000000000 | 04000140000000000000000 | oct |
| • | 0000000000000001 | 4018000000000000 | hex |
| -1, -1.0 | 17777777777777777777777777777 | 14000140000000000000000 | oct |
| • | FFFFFFFFFFFFFF | C00180000000000 | hex |
| 2, 2.0 | 0000000000000000000000000002 | 04000240000000000000000 | oct |
| · | 0000000000000002 | 4002800000000000 | hex |
| 4, 4.0 | 000000000000000000000000000000000000000 | 04000440000000000000000 | oct |
| · | 000000000000004 | 400380000000000 | hex |
| | | | |
| DEC VAX | | | |
| 1, 1.0 | 0000001 | 00004080 | hex |
| -1, -1.0 | FFFFFFFF | 0000C080 | |
| 2, 2.0 | 0000002 | 00004100 | |
| 4, 4.0 | 0000004 | 00004180 | |
| | | | |
| CDC CYBER (NOS) | | 1700/00000 0000000000 | |
| 1, 1.0 | 000000000 000000001 | 1720400000 0000000000 | oct |
| -1, -1.0 | 7777777777 777777776 | 6057377777 7777777777 | |
| 2, 2.0 | 000000000 000000002 | 1721400000 0000000000 | |
| 4, 4.0 | 000000000 000000004 | 1722400000 0000000000 | |
| CDC CYBER (NOS/ | VE) | | |
| 1, 1.0 | | 400180000000000 | hex |
| -1, -1.0 | FFFFFFFFFFFFFFF | C00180000000000 | hex |
| 2, 2.0 | 000000000000002 | 4002800000000000 | hex |
| 4, 4.0 | | 4003800000000000 | hex |
| | | | |
| Note the difference in the format of negative integers (and CYBER | | | |
| floating point) | numbers: | | |
| | | | |

| CRAY X-MP, DEC VAX, CDC NOS/VE | CDC NOS |
|--------------------------------|-------------------|
| | |
| two's complement of absolute | one's complement |
| value | of absolute value |

8. Logical variables are represented by:

| | CRAY X-MP, CDC | DEC VAX |
|-------|----------------|------------|
| TRUE | -1 | 1 in bit 0 |
| FALSE | 0 | 0 in bit 0 |

9. By default, your program area in central memory is set as follows:

| Cray COS | Cray UNICOS | DEC VMS | CDC NOS | CDC NOS/VE |
|----------|-------------|---------|---------|------------|
| | | | | |
| zero * | zero | zero | zero | zero |

*** Internal Representation ***

** CRAY X-MP **

Words in the CRAY X-MP are 64 bits long. Bits are numbered 0-63or 63-0. 0 - the sign bit (0 = positive; 1 = negative) (23) Integer: bit bits 1:23 - the absolute value of the integer (22:0) $- \sim -10^{**}14$ to $\sim 10^{**}14$ range Integer (CFT, INTEGER=64): 0 - the sign bit (0 = positive; 1 = negative) (63) bit bits 1:63 - the absolute value of the integer (62:0) $- \sim -10^{**}19$ to $\sim 10^{**}19$ range Real: bit 0 - the sign of the number (63) bits 1:15 - the exponent (2000 bias) (62:48)bits 16:63 - the mantissa (47:0) - ~10**-2466 to ~10**2465 range precision $- \sim 14$ decimal digits First word: Double: bit 0 - the sign of the number (63) bits 1:15 - the exponent (2000 bias) (62:48)bits 16:63 - the high order part of the mantissa (47:0) Second word: bits 0:15 - unused (63:48)bits 16:63 - the low order part of the mantissa (47:0) - ~10**-8193 to ~10**8189 range precision - \sim 29 decimal digits

Page G-7

Bytes in the DEC VAX are 8 bits long with bits are numbered 7-0. A word (INTEGER*2 in Fortran) is 16 bits long (15-0). A longword (INTEGER or INTEGER*4) is 32 bits long (31-0). Word (INTEGER*2): bit 15 - the sign bit (0 = positive; 1 = negative) bits 14:0 - the absolute value of the integer - -32,768 to 32,767 range Longword (INTEGER*4): bit 31 - the sign bit (0 = positive; 1 = negative) bits 30:0 - the absolute value of the integer range - -2,147,483,648 to 2,147,483,647 F float (REAL*4): 15 bit - the sign of the number bits 14:7 - the exponent (excess 128) 6:0 and bits 31:16 - the mantissa - ~.29*10**-8 to ~1.7*10**38 range precision $- \sim 7$ decimal digits D float (REAL*8, DOUBLE PRECISION): bit 15 - the sign of the number bits 14:7 - the exponent (excess 128) bits 6:0 and 63:48 and 47:32 and 31:16 - the mantissa - ~.29*10**-8 to ~1.7*10**38 range precision $- \sim 16$ decimal digits G_float (FORTRAN/G_floating): bit 15 - the sign of the number bits 14:4 - the exponent (excess 1024) bits 3:0 and 63:16 - the mantissa - ~.56*10**-308 to ~.9*10**308 range precision $- \sim 15$ decimal digits H float (REAL*16): bit 15 - the sign of the number bits 14:0 - the exponent (excess 16,384) bits 127:16 - the mantissa - ~.84*10**-4932 to ~.59*10**4932 range precision $-\sim 33$ decimal digits

 $\star\star$

DEC VAX **

Page G-8

Rev0 90/09/30

** CDC CYBER (NOS, NOS/BE) **

Words in the CDC CYBER 170 and CYBER 180 (when running NOS or NOS/BE) are 60 bits long. Bits are numbered 59-0. - the sign bit (0 = positive; 1 = negative) Integer: bit 59 bits 58:0 - the absolute value of the integer Integer: bit 59 - the sign bit (0 = positive; 1 = negative) bits 47:0 - the absolute value of the integer (if used in multiplication or division) Real: bit 59 - the sign of the number bits 58:48 - the exponent (2000 bias) bits 47:0 - the mantissa with the binary point after bit 0 Double: (Double precision is performed in the software, not in the hardware) First word: bit 59 - the sign of the number bits 58:48 - the exponent (2000 bias) bits 47:0 - the high order part of the mantissa with the binary point after bit 0 Second word: bit 59 - the sign of the number bits 58:48 - the exponent (2000 bias) bits 47:0 - the low order part of the mantissa with the binary point after bit 0

 $\star\star$

Page G-9

**

CDC CYBER (NOS/VE) Words in the CDC CYBER 180 (when running NOS/VE) are 64 bits long. Bits are numbered 0-63. Integer: bit 0 - the sign bit (0 = positive; 1 = negative) bits 1:63 - the absolute value of the integer precision $- \sim 19$ decimal digits Real: bit O - the sign of the number bits 1:15 - the exponent (4000 bias) 1:3 - the following FP (or DP) numbers 00x - FP zero 0x0 - FP zero 011 - standard FP number 100 - standard FP number 101 - FP infinity 110 - FP infinity 111 - FP indefinite bits 16:63 - the mantissa with the binary point before bit 16 - 4.8*10**(-1234) to 5.2*10**(1232) range precision $- \sim 14$ decimal digits Double: First word: bit O - the sign of the number bits 1:15 - the exponent (4000 bias) 1:3 - same as for real bits 16:63 - the high order part of the mantissa with the binary point before bit 16 Second word: bit 64 - same at bit 0 bits 65:79 - same as bits 1:15 bits 80-127 - the low order part of the mantissa with the binary point after bit 0 - 4.8*10**(-1234) to 5.2*10**(1232) range precision $-\sim 29$ decimal digits

90/09/30 Rev0

**** **** Glossary Alphabetic (CDC - NOS) The letters A-Z. Alphabetic (CDC - NOS/VE) The letters A-Z, a-z. Alphabetic (Cray - COS) , , , , , and the letters A-Z, a-z. Alphabetic (DEC) , (underscore), and the letters A-Z, a-Z (upper andlower case are the same). Alphanumeric Alphabetic and the digits 0-9. User initials (userid or username) The 4-character ID assigned to each user by Code 3502. This is used to identify jobs, for charge authorization, to identify permanent and MSS files, magnetic tapes, etc. 90/09/30 Rev0 -Acc Page Index-1 ***** Index ***** Note - NOS/BE system control statements are flagged with ". Intercom commands are flagged with @. UPDATE directives begin with *. Compiler options are flagged with \$. Primary references are flagged with an asterisk after the page number, for example, 1-1*.

* (comment) 3-2-1, C-5 ! (comment) D-1 \$ (create VMS subprocess) 3 - 1 - 12+ (display next) D-32 - (display previous) D-32 @ (execute command file) 3-1-12, D-32* 3-1-12 \$ (Execute DCL command) @ (invoke procedure) D-1 3-1-12 - (last page) + (next page) 3-1-12 3-1-11 ! (prompt) \$ (subprocess) D-32 1-3-1, 7-1-1* ABAQUS Abort 3-3-3, 3-6-3, C-3, C-31, C-33 ABORT 3-1-12, 3-1-16, 3-6-3, D-33* ABS 3-6-3 3-6-1*, 3-6-3, G-1, G-4 Absolute ABTCODE 3-2-6 1-2-6, 3-1-1, 4-1-4, 4-1-10, 4-1-11, Access 5-1-1*, C-40, C-41, C-44, C-51, D-13 3-1-1, 3-1-10, 3-2-2, C-5*, C-6 ACCESS C-2*, C-3* Access mode C-3* Access, unique 4-1-8 Account ACCOUNT 1-2-6, 3-1-2, 3-1-3, 3-1-5, 3-1-11, 3-1-15, 3-2-1, C-5*, D-29 C-7, C-41, C-55, D-11 Account number 1-2-2* Account number, alternate 1-2-2*, C-47 Account number change Accounting F-4

Rev0

90/09/30

Accounts, multiple 1-2-2* 3-2-2, 4-1-3*, C-6* ACQUIRE ACSL 1-3-1 AC/DC analysis 7-1-18 Address, ethernet 1-1-2, 1-1-3, 5-1-8 Address, host 5-1-5 Address, network F-2, F-3 3-2-2, C-7* ADJUST Administrative personnel F-4* 1-2-1*, 1-2-8 ADP Control Center Algebra, linear 7-1-13 ALGOL 1-3-1 ALIGN 3-6-3 ALLOCATE 6-1-4, D-2* Alphabetic G1-1 Alphanumeric G1-1 ALTACN 3-2-1, C-7* 1-2-2* Alternate account number AM C-2* 3-3-3 Ampersand Annapolis 1-2-1, D-6, D-10, F-4 ANSI standard label 6-1-1 1-3-1 APL 3-3-3, C-4* Apostrophe Applied mathematics 7-1-9 APPSW 3-1-17 APRINT D-6* APT 1-3-1 Arithmetic operator 3-2-7 ASCII 3-2-7, 6-1-1, 6-1-4, A-1*, A-2*, A-3*, A-4*, G-2, G-3 ASSIGN 3-2-2, 5-1-3, C-7*, C-12 3-6-2 Asterisk At sign D-1 ATTACH 3-1-12, D-33* Attention D-35 ATTENTION 3-1-12, 3-1-16, D-34* Attribute, file C-41 Attribute, record D-10 Attributes, file 4-1-4, 4-1-10 Attributes, link 4-1-9 4-1-7, C-10, C-40, D-23 Audit AUDIT 3-2-3, C-8* Audit files 4-1-4, 4-1-10 AUDPL 3-2-5, C-10*

| 90/09/30 Rev0 | Aut-C Page Index-3 |
|--|--|
| Automatic logout AUX Auxiliary printer | 5-1-2 D-7* D-7, D-8 |
| AUXPRINT | D-8* |
| Backup | 4-1-2*, 4-1-7, 6-1-1, C-42, C-45, C-46 |
| BACKUP | 6-1-4 |
| Basic | 1-3-1 |
| Batch | 1-1-1, 2-1-1, 3-1-1, 3-1-2, 3-1-3, |
| | 3-1-4, 3-1-5, 3-1-8, 3-1-10, 5-1-4*, F-1, F-2 |
| Batch job | 3-1-3 |
| Batch jobs, killing | 3-1-6*, 3-1-8*, 5-1-4* |
| BCD | 6-1-1, A-3*, A-4*, G-2 |
| BEFORE | 3-4-2 |
| BIN | 3-6-3, 3-6-10 |
| Binary | 6-1-1 |
| Binary blocked | C-2 |
| Binary point | G-8, G-9 |
| | , |
| \$BLD | C-55 |
| BLOCK | 3-2-3, C-11* |
| Block data | 3-6-14 |
| Blocked | 6-1-1, 6-1-4 |
| Blocked, binary | C-2 |
| Blocked, character | C-2, C-43, C-45 |
| Blocked dataset | C-11, C-22, C-23, C-57, C-61, C-64 |
| Blocked file | C-22, C-57 |
| Blocked record | C-23, C-58 |
| BUDGET job class | 3-1-4*, C-37 |
| | |
| Buffer Buffer length | C-35 3-6-8 |
| Buffer length BUILD | 3-6-6 3-2-5, 3-5-1*, C-12* |
| Bulletin | 5-1-2 |
| BYE | 3-1-12, $3-1-16$, D-34* |
| 0.5 | 5 1 12, 5 1 10, 5 54 |
| | |
| ^C | 3-1-12 |
| | |
| с | 1-3-1 |
| ^C | 3-1-12 |
| c | 7-1-2* |
| - | · • • |

| Page Index-4 C -CHA | Rev0 90/09/30 |
|---|--|
| ^C (CTRL-C) | D-35* |
| Calcomp Calcomp plot CALL Call by name | $ \begin{array}{c} 1-3-1 \\ 1-2-8 \\ 3-2-1, 3-2-4, 3-3-1, 3-3-5, 3-4-2, \\ C-14^{*} \\ 3-2-4, 3-2-5, 3-3-1, C-15 \\ \hline $ |
| CAPRINT | D-10* |
| Card interpreter | 1-2-8 |
| Card punch | 1-2-8 |
| Card verifie: | 1-2-8 |
| Carderock | 1-2-1, 4-1-2, F-4 |
| Caret | C-4* |
| Carriage control | D-10 |
| Carriage control, Fortran | D-44 |
| CASE | 3-6-3 |
| Case, upper | 3-4-1 |
| CATLIST | 4-1-10* |
| Caution | 3-6-1 |
| Caution, segmentation | 3-6-15 |
| CC | 1-3-1 |
| CCF | 5-2-1, D-1, F-1 |
| CCRM | E-2 |
| CDC | E-1, G1-1 |
| CDC CYBER 860 | 1-1-2 |
| CDC CYBER 860 (NOS) | 1-3-1 |
| CDC CYBER 860 (NOS/VE) | 1-3-1 |
| CDC output | D-10 |
| cdc860 | 1-1-2, F-2 |
| CDD | 1-3-1 |
| CDEFINE | 1-4-2, 3-1-8 |
| CDROP | 3-1-8 |
| Central processing unit | 2-1-1, 3-1-1, 5-1-1 |
| Central site | 3-1-8, F-1, F-2, F-3 |
| Central Site | 1-2-1* |
| Central Site operator | 5-1-2 |
| Certify tape | 6-1-2 |
| CFT | 3-2-4, C-15*, E-1 |
| CFT77 | 3-2-4, C-19*, E-1 |
| Change | C-39, C-56 |
| CHANGE | 4-1-1, 4-1-2 |
| Change access password | 1-2-6 |
| Change account number | 1-2-2* |
| CHANGE_LINK_ATTRIBUTES | 4-1-9* |

| 90/09/30 | Rev0 |
|----------|------|
|----------|------|

Page Index-5

CHANGE TAPE LABEL ATTRIBUTES 1-5-1, 6-1-7 CHARACTER G-1, G-2 Character blocked C-2, C-43, C-45 Character conversion 3-6-3 3-4-1 Character, master Character set, ASCII A-1 A-3 Character set, CDC C-7 Characteristic C-55 CHaracteristic C-39 Characteristic, dataset C-27 Characteristic, disposition 1-2-1, 4-1-1, 4-1-2, 4-1-8 Charge CHARGES 3-2-1, C-20* CINT 3-1-1, 3-1-2, 3-1-11, 3-1-15, D-31* CKILL 3-1-8 Class, job 3 - 1 - 43-1-4 Class, service 3-1-4 Classified processing Clean magnetic tape 1-2-8, 6-1-2 3-1-12, D-34* CLEAR CMS 7-1-3* CNEWCHRG D-11* 1-2-6, 3-1-2, D-13* CNEWPW 1 - 3 - 1Cobol COBOL G-3 3-1-1 Code, disposition 7-1-3* Code Management System 6-1-1 Coded COLLECT 3-1-12, D-35* .COM 5-3-1 COMDECK 3 - 4 - 1Command D-1* Command file D-35, D-37, D-41, D-42 D-30 Command, station 3-4-3, 3-6-1, 3-6-2, C-5, D-1 Comment COMMENT 3-1-12, D-35* 3-6-7 Common 3-6-3, 3-6-4, 3-6-10, 3-6-15 Common block COMMONS 3-6-4, 3-6-10, 3-6-14 Communication 4-1-9 COMPARE 3-2-3, C-21* 3-4-2, C-15, C-19, C-50 Compile COMPILE 3 - 4 - 3Complaints 1-2-8 G-4 Complement, one's

Page Index-6 Com-Cra

a

Rev0

90/09/30

Complement, two's G-4 3-2-7 Completion code 3-3-1, 3-3-5 Complex procedure 5-2-4, 5-4-2, 5-5-3 Compress a library Computer Center 1-2-1*, 4-1-1, 4-1-2, 5-2-1 Computer Center Ref Man E-2 F-4 Computer service F-1 Computer Systems C-34 Conditional CONNECT 3-1-16 G-1 Constant D-30, D-31, D-32*, D-38, D-45 Context, Cray 3-2-8, 3-6-2, C-4* Continuation 3-1-3*, 3-3-1, 5-3-1 Control statement Control word, maintenance C-3 C-3 Control word, read C-3 Control word, write D-35* CONTROL C CONTROL 0 D-36* D-36* CONTROL Z G-1 Convergence factor Conversion G-1 Conversion, character 3-6-3 Convert C-11, C-61, D-18 Copy D-46 1-5-1, 3-4-2, 3-4-3, 3-6-11 COPY COPYD 3-2-3, C-22* 6-1-4 COPYD2T COPYF 3-2-3, C-22* 3-2-3, C-23* COPYNF COPYR 3-2-3, C-23* COPYT2D 6-1-4 3-2-3, C-24* COPYU 3-1-1*, E-1, F-1, G1-1 COS 3-1-10 COS input queue COS level 3-2-6 C-37 Cost, job 3-2-6 Counter 3-1-5 .CPR 2-1-1, 3-1-1, 5-1-1 CPU Cray 1-3-1, 3-1-15, 4-1-3, D-11, D-13, D-14, D-23, D-24, D-25, D-27, E-1, F-1, G1-1 5-2-1, D-1, D-30* CRAY Cray context 3-1-11, D-30, D-31, D-32*, D-38,

90/09/30

Rev0

Page Index-7

D-45 Cray context CRAY DROP 3 - 1 - 63-1-6 CRAY KILL Cray station E-1 3-1-11, 3-1-12 Cray Station 1-1-2*, 1-1-3 Cray station ID CRAY STATUS 3-1-6 3-1-1, 3-1-2, 3-1-5, D-29 CRAY SUBMIT CRAY X-MP 1-1-2, 2-1-1*, 3-1-1*, F-1, G-6 CRAYDEF 1-4-2, 3-1-8 C-7 Create 3-5-2, 5-2-3, 5-4-1, 5-5-1 Create a library Critical file 4-1-2 Cross-reference C-33 3-1-3*, 3-3-1* \$CS **CSTATUS** 3-1-8 CSUBMIT 3-1-1, 3-1-2, 3-1-5, 3-1-8, D-14*, D-29 1-2-6* CSUBMIT/NUPW D-35* CTRL-C (^C) D-36* CTRL-0 (^0) 5-2-6 CTRL-Z CTRL-Z (^Z) D-36* CWEOF 3-4-2 CYBER 860 3-1-1, 3-1-8, 3-1-16, 5-1-6, 6-1-6, 6-1-7, D-20, D-22, F-2, G-8, G-9 **C1** 1-1-2*, F-1 G-1, G-2 Data DATA 3-3-2, C-24* Database management system 7-1-4 7-1-11 Database mgt sys, relational Dataset 3-1-1, C-11, C-21, C-27, C-28, C-34, C-53, C-54, C-60, D-11, D-23, D-24, D-25, D-27 DATASET 3-1-13, D-36* Dataset, blocked C-22, C-23, C-57, C-61, C-64 Dataset characteristic C-39 3-2-2 Dataset definition Dataset format C-2* Dataset, library C-36 3-1-1, C-27, C-28, C-55 Dataset, local Dataset, permanent 3-1-1, 4-1-3, C-3, C-4, C-5, C-8, C-26, C-51, C-55, D-36



Page Index-8 I

Rev0

90/09/30

C-7 Dataset size C-24, C-58, C-61 Dataset, unblocked Datatrieve 1-3-1 7-1-4* DataTrieve (DTR) DATE 3-2-6 1-2-8 Davfile DBL07 1-1-3, 1-3-1, 3-1-4 DBMS 1-3-1 DC C-2* DCL D-1*, D-38, E-1 C-24*, C-30 DDA 1-1-5, 5-1-5, 5-1-7 DDN 6-1-4, D-2* DEALLOCATE 3-6-4, C-56 Debug 3-2-3, C-25*, C-30 DEBUG 5-1-1*, E-1, F-2, G1-1 DEC 6-1-4, G-7 DEC VAX 1-1-3, 5-1-1*. F-2 DEC VAXcluster 1-3-1 DECalc 3 - 4 - 1DECK 5-1-5 DECnet 3-1-4*, C-37 DEFER job class Define D-44 4-1-2, 5-1-3, 5-2-6 DEFINE DEFLIB 3-6-4 Degauss magnetic tape 1-2-8 3-1-13, D-37* DELAY Delete 4-1-7, D-24, D-39, D-43 DELETE 3-2-2, 3-4-2, C-26* 3-2-7, 3-2-8, 3-3-3 Delimiter 1-1-1 Demand D-18* DETAB DETACH FILE 6-1-7 C-2* DF Diagram, network 1-1-5 G1-1 Digit Direct file 4-1-2, 4-1-4, 4-1-6 Directive, BUILD 3-5-1*, C-12 3-6-2* Directive, SEGLDR 3-6-10* Directive, segmentation 3-4-1* Directive, UPDATE 5-1-3, 5-1-4 DIRECTORY D-36 Discard 3-1-13, 3-1-16, D-37* DISCARD 5-1-2 Disconnect

1

| DISMOUNT | 6-1-4, D-3* |
|-------------------------------|---|
| Dispatch | F-4 |
| Display code | A-1, A-2, A-3, A-4, G-2, G-3 |
| Display region | D-46 |
| DISPLAY_BACKUP_LABEL_TAPE | 6-1-7 |
| | |
| DISPLAY_LINK_ATTRIBUTES | 4-1-9* |
| DISPLAY_TAPE_LABEL_ATTRIBUTES | 6-1-7 |
| Dispose | C-45 |
| DISPOSE | 3-1-8, 3-2-2, 4-1-3*, C-27* |
| Disposition | C-2* |
| - | |
| Disposition code | 3-1-1 |
| DISSPLA | 1-3-1, 7-1-5* |
| DN | C-2* |
| Document | 5-5-1 |
| Double precision | G-6, G-7, G-8, G-9 |
| - | |
| DRD | 3-6-4 |
| DROP | 3-1-13, D-37* |
| DS | 3-2-3, C-28* |
| DSDUMP | C-28* |
| DTLIB | 3-5-1, 5-2-1, 5-4-1, 5-5-1, 7-1-6*, |
| | , |
| | D-1 |
| DTLIBCRAY | 5-5-1 |
| DTn | 5-1-6 |
| DTNET | $1-1-2$, $1-1-4^*$, $1-1-5$, $5-1-1$, $5-1-2$ |
| dtoal | F-3 |
| | |
| DTOA1 (OASYS) | F-3 |
| dtoa3 | F-3 |
| DTOA3 (OASYS) | F-3 |
| DTR | 7-1-4* |
| dtrc | F-3 |
| | |
| DTRC | 3-2-1 |
| DTRC (OASYS) | F-3 |
| dtvms | F-2 |
| dtvms3 | F-2 |
| dtvms4 | F-2 |
| | |
| DT1 | 1-1-3* |
| DT3 | 1-1-3*, 1-3-1, F-2 |
| DT4 | 1-1-3*, 1-3-1, F-2 |
| DT.NAVY.MIL | 5-1-6, 5-1-8 |
| dt.navy.mil | F-2, F-3 |
| | , |
| Dual state | F-2 |
| Dual-state | 4-1-9 |
| Dump | 3-2-3, C-24, C-25, C-28, C-33 |
| DUMP | 3-2-3, C-30* |
| DUMP JOB | 3-2-3 , C-30, C-30* |
| | ,, |

Page Index-10 DUP-Epi Rev0 90/09/30 DUP 3-6-11 3-6-4 DUPENTRY 3-6-5 Dynamic DYNAMIC 3-6-4 Dynamic Dump Analyzer (DDA) C-24 DYNA3D 1-3-1, 7-1-7* G-7 D float EAM facilities 1-2-8 6-1-1, 6-1-4, A-1*, A-2*, A-3*, A-4*, EBCDIC G-2 D-48 Echo ECH0 3-2-1, 3-6-5, C-31* C-2* ED EDIT 3-4-4 Edition C-2 Editor 5-1-4, 5-6-1*, E-1 Editor (EDIT FILE) E-1 Editor (EDT) 5-1-2, 5-1-4, 5-6-1, E-1 Editor (EVE) 5-1-2, 5-1-4, 5-6-2, E-1 Editor (FSE) E-1 5-1-4, 5-6-1 Editor (TPU) 1-4-2 EDIT FILE EDIT FILE (editor) E-1 EDT (editor) 5-1-2, 5-1-4, 5-6-1*, E-1 EDTINI.EDT 1-4-1 EISPACK 1-3-1 3-6-9 Eject 3-2-4, C-34 ELSE 3-2-4, 3-2-6, C-34 ELSEIF ENDCONNECT 3-1-16 ENDIF 3-2-4, C-34 3-2-4, C-38* ENDLOOP ENDPLAY 3-1-16 3-2-4, 3-3-1, 3-3-2, C-51 ENDPROC ENDSEG 3-6-11, 3-6-14 ENDTREE 3-6-11, 3-6-13, 3-6-14 3-1-3, 3-1-11, D-37 End-of-file End-of-record A-4 3-6-4, 3-6-5, 3-6-8, 3-6-9 Entry point EOF 1-5-1, 3-1-3*, 3-1-13, 3-1-16, 3-4-2, D-37* EPILOG 1-4-2 1-4-1* Epilog file

90/09/30 Rev0

3-6-5 EOUIV C-2* ERR 3-6-1, 3-6-3, C-39 Error Error code checking 3-2-6 1 - 1 - 5Ethernet 1-1-2, 1-1-3, 5-1-8 Ethernet address EVE (editor) 5-1-2, 5-1-4, 5-6-2*, E-1 EVESINIT.EVE 1-4-1 Exclamation mark D-1 Execute C-15, D-42 D-36 Exit 3-1-11, 3-1-13, 3-2-1, 3-3-1, C-31*, EXIT D-38* 3-2-4, 3-2-6, C-34 EXITIF EXITLOOP 3-2-4, 3-2-6, C-38* C-2* EXO Expire (password) 1-2-6 Exponent G-6, G-7, G-8, G-9 Expression, JCL 3-2-6 External, unsatisfied 3-6-9 ·5-2-5, 5-4-3, 5-5-3 Extract a module G-5 FALSE FCOPY 1 - 5 - 1C-43 Fetch FETCH 3-1-1, 3-1-10, 3-2-2, 4-1-3*, C-6, . C-32* 3-2-6, C-38 Field length 4-1-1, 4-1-3, 4-1-7, 4-1-9, 4-1-10, File 5-1-3, 5-1-4, 6-1-1, D-11, D-23, D-24, D-25, D-27 File attribute C-41 4-1-4, 4-1-10 File attributes C-22, C-57 File, blocked D-35, D-37, D-41, D-42 File, command File, critical 4-1-2 File, direct 4-1-2, 4-1-4, 4-1-6 5-1-2, 5-1-4 File, journal C-52 File, local File, overwrite C-55 1-2-2, 4-1-3, 4-1-9, 5-1-2File, permanent C-41 File, rename 5-1-6 File, transfer File transfer 7-1-12





Page Index-12 Fil-Glo

Rev0 90/09/30

| File Transfer Protocol (FTP) | 5-1-6 |
|---|---|
| FILEMANAGER | 6-1-4 |
| Files | C-40 |
| Files, audit | 4-1-4, $4-1-10$ |
| FILE-SET_IDENTIFIER | 1-5-1 |
| FINDHOST | 5-1-5* |
| Finite element | 7-1-7, 7-1-15 |
| Fixed length | 6-1-1 |
| Fixed-format | D-18 |
| FL | 3-2-6 |
| Flag, mode | C-39 |
| FLM | 3-2-6 |
| Floating point | G-4, G-7 |
| FLODUMP | 3-2-3, C-33* |
| Flowtrace table | C-33 |
| FMS | 1-3-1 |
| FORCE | 3-6-5 |
| Forced loading | 3-6-5 |
| Foreign | 6-1-4 |
| Foreign tape | 6-1-6 |
| Formal parameter Format, dataset Formats, tape Fortran | 3-3-2, 3-3-3 C-2* 6-1-1 1-3-1, C-15, C-19, C-33, D-10, D-18, E-1, E-2, G-3 |
| Fortran carriage control Fortran I/O record length Front-end FSE | D-44 3-6-8 1-1-1, 3-1-1, 3-1-10, 3-1-11, C-2, C-3, C-6, C-27, C-32, F-1 1-4-1 |
| FSE (editor) | E-1 |
| FSEPROC | 1-4-1 |
| FTP | 1-3-1, 5-1-6*, 5-1-7 |
| FTREF | 3-2-4, C-33* |
| FT05 | C-8 |
| FTO6 | C-8 |
| Full screen | E-1 |
| Functions, special | 7-1-9 |
| F_float | G-7 |
| G register | 3-2-7 |
| Gateway | 5-1-7 |
| Gen. Purp. Sim. Sys. (GPSS) | 7-1-8* |
| GET_FILE | 4-1-9* |
| Global symbol | 5-4-2 |

| 90/09/30 Rev0 | GPS-I Page Index-13 |
|----------------------------|--------------------------------------|
| GPSS | 1-3-1, 7-1-8* |
| GRIPE | 1-2-8 |
| G0-G7 | 3-2-7 |
| G_float | G-7 |
| Hardware Configuration | 1-1-2 |
| Hasp | 1-3-1 |
| Header | 3-6-9 |
| Неар | 3-6-4, 3-6-8 |
| HEAP | 3-6-5 |
| HELP | 1-3-1, 3-1-13, 3-1-16, 5-1-3, 5-2-1, |
| | 5-2-5, D-38* |
| Help library | 5-1-3, 5-2-1*, 5-2-3 |
| Help module | 5-2-2 |
| Hexadecimal | G-1, G-2, G-4 |
| HFT | 4-1-6* |
| HFT ACCESS | 4-1-6* |
| HFT CHANGE | 4-1-6* |
| HFT DEFAULT | 4-1-6* |
| HFT DELETE | 4-1-6* |
| HFT DIRECTORY | 4-1-6* |
| HFT FETCH | 4-1-6* |
| HFT PASSWORD | 1-2-7, 4-1-1, 4-1-6* |
| HFT PERMIT | 4-1-6* |
| HFT STORE | 4-1-6* |
| Hierarchy | 3-2-6, 3-2-7 |
| Histogram | C-59 |
| History | 3-1-3, 5-2-5, 5-4-1 |
| HLP\$LIBRARY | D-1 |
| HOLD | 3-2-2, C-34*, C-47 |
| Hollerith | G-1, G-2 |
| Home directory | 5-1-3 |
| Host | 5-1-5 |
| Host address | 5-1-5 |
| Host name | 5-1-5 |
| Host table | 5-1-5 |
| Hostname | 5-1-7 |
| HOTSPOT | 1-3-1 |
| HYPERchannel | 1-1-5 |
| HYPERchannel File Transfer | 4-1-6 |
| H_float | G-7 |
| | |

I

Rev0 90/09/30

.

٠

| ICF | 1-4-2, 3-1-1, 3-1-16 |
|--|---|
| ICFPROC | 1-4-2, 3-1-16* |
| ICFSTATUS | 3-1-16 |
| ID | 3-4-2, C-3* |
| IDENT | 3-4-2 |
| IDN | C-3* |
| IF | 3-2-4, 3-2-6, C-34* |
| IMSL | 1-3-1, 7-1-9* |
| \$IN | 3-1-3* |
| INCLUDE | 5-5-1 |
| Indefinite | G-3 |
| Inefficient code | C-59 |
| Infinite | G-3 |
| INFORM | D-44 |
| Information | 5-2-1 |
| INGRES | 1-3-1, 7-1-11* |
| Initialize | 3-6-7, C-64 |
| INITIALIZE | 6-1-4, D-3* |
| Input | D-42, D-43, D-48 |
| Input dataset | 3-1-3 |
| Input queue | C-60 |
| INSERT | 3-4-2 |
| Integer | 3-2-6, G-4, G-6, G-7, G-8, G-9 |
| INTEGER*2 | G-7 |
| INTEGER*4 | G-7 |
| Interactive Interactive Cray Facility Internal data structure Internal representation | 1-1-1, 2-1-1, 3-1-1, 3-1-2, 3-1-11, 4-1-1, D-34, F-1, F-2 3-1-1, 3-1-16 G-1* G-2*, G-6* |
| INTERNET | 5-1-5 |
| Interpret | C-24, C-25 |
| Interpreter, card | 1-2-8 |
| Interrupt | D-33, D-34 |
| Int'l Math & Stat Libs (IMSL) | 7-1-9* |
| IOAREA | 3-2-1, C-35* |
| ISTATUS | 3-1-13, D-38* |
| ITEMIZE | 3-2-4, C-36* |
| Iterative | C-38 |
| I/O | 6-1-1, C-48 |

3-2-7 J register JCL C-1 3-1-5, 3-1-8, 3-1-10, 3-1-11, 4-1-1, Job

C-47, C-54, D-14, D-22, D-29, D-39 Job 3-1-3, 3-1-13, 3-2-1, C-36*, D-38* JOB 3-1-3 Job, batch Job class 3-1-4 D-46 Job class structure Job control 3-3-1 3-2-1, 3-2-6 Job control language Job control statement 3-1-5 C-37 Job cost Job order number 1-2-1, 4-1-1, 4-1-8 Job resource C-20 Job Status Register 3-2-7 Job step 3-2-7 3-2-1, 3-3-4, C-37* JOBCOST .JOU 5-1-2 5-1-2, 5-1-4 Journal file JSR 3-2-7 JSTAT 3-1-13, D-39* J0-7J 3-2-7 Kermit 1-3-1, 5-1-7 7-1-12* KERMIT 3-3-3, 3-6-2 Keyword 3-3-2, 3-3-3, C-1*, D-1 Keyword parameter KILL 3-1-13, D-39* Killing batch jobs 3-1-6*, 3-1-8*, 5-1-4* 1-5-1, 6-1-4, D-3 Label LABEL 1-5-1, 6-1-6 Label, ANSI standard 6-1-1 6-1-1 Label, tape 3-2-4 Language G1-1 Letter LIB 3-6-6 Librarian, tape 6-1-2, 6-1-3, F-4 3-6-4, 3-6-6, 3-6-7, C-12, E-1 Library 3-2-1, 5-2-1, 5-2-3, 5-2-4, 5-2-5, LIBRARY 5-4-1, 5-4-2, 5-4-3, 5-5-1, 5-5-2, 5-5-3, C-37* Library dataset C-36 5-1-3, 5-2-1*, 5-2-3 Library, help 3-2-5, 3-5-1*, 5-4-1*, 5-4-2, 5-4-3 Library, object 7-1-6, 7-1-9, 7-1-13 Library of subprograms

90/09/30

Library, program 3-2-5, 3-4-1, C-10, C-62 6-1-3 Library tape 5-5-1* Library, text 7-1-18 Linear Linear algebra 7-1-13 Link 5-4-1, F-1, F-2, F-3 LINK 5-4-3 Link attributes 4-1-9 1-3-1, 7-1-13* LINPACK List 3-6-5, 5-2-5*, 5-4-3, 5-5-3, C-28 3 - 4 - 3LIST Listing C-48 LISTLB 6-1-6 3-2-6, 3-3-3 Literal Literal string 3-2-7 LLBUDG (Cray job class) 3-1-4 LLDEFER (Cray job class) 3-1-4 LLNORM (Cray job class) 3-1-4 3-1-4 LLPZERO (Cray job class) 3-6-5 Load 3-6-6, 3-6-9, 3-6-15 Load map 3-6-1*, C-55 Loader 4-1-3, C-2, C-5, C-32 Local Local dataset 3-1-1, C-27, C-28, C-55, C-60 Local file C-52 Local file name C-15 C-35 Lock 1-2-8 Log 3-1-3*, C-37 SLOG C-51, D-41 Logfile LOGFILE 3-1-13, D-40* Logfile message C-31 3-2-4 Logic structure Logical G-5 Logical name 5-1-3, D-2 3-2-7 Logical operator 5-1-2 Login 5-1-3 Login Procedure File LOGINPR 1-4-1 1-4-1, 5-1-3 LOGIN.COM LOGOFF 3-1-16 3-1-16, 3-1-17 LOGON 5-1-2 LOGOUT 5-1-2 Logout, automatic Longword G-7

90/09/30 Rev0

Page Index-17

LOOP 3-1-13, 3-2-4, C-38*, D-41* Lost time 1 - 2 - 8LSBUDG (Cray job class) 3-1-4 LSDEFER (Cray job class) 3-1-4 LSNORM (Cray job class) 3-1-4 LSPZERO (Cray job class) 3-1-4 Macsyma 1-3-1 6-1-1* Magnetic tape Magnetic tape, clean 1-2-8 Magnetic tape, degauss 1 - 2 - 8Magnetic tape, purchase 1-2-8 Magnet; c tape, test 1-2-8 5-1-7*, 5-1-8 Mail Mainframe 3-2-6 Maintenance F-1, F-2, F-3 Maintenance control word C-3 Mantissa G-6, G-7, G-8, G-9 Manuals E-1, F-4 MAP 3-6-6 Map, load 3-6-6, 3-6-9, 3-6-15 Mask G-1 Mass Storage System 1-1-1, 3-1-10, 3-2-2, 3-3-4, 4-1-1*, C-40, F-2 MASTER 3-4-3 Master character 3 - 4 - 17-1-9 fathematics Memory 2-1-1, 3-1-1, 3-1-4, 3-6-5, 3-6-8 MEMORY 3-2-1, C-38* Merge C-58 3-6-1, 3-6-6, C-2, C-3, D-40, D-41, Message D-44, E-1 MESSAGE 3-1-13, D-41* Message, logfile C-31 Message, recorded F-4 C-3* MF MFN 1-1-2*, F-2 Microcomputer 7-1-12, F-4 Microfiche D-19 MLEVEL 3-6-6 MMS 7-1-14* MODE 3-2-1, C-39* MODE4A/4C 1-3-1 MODIFY 3-2-2, C-39*

Page Index-18

Network

90/09/30

Modify a library 5-2-4, 5-4-2, 5-5-2 3-6-3, 3-6-11 Module Module, absolute 3-6-1 Module, help 5-2-2 7-1-14 Module Mgt System (MMS) 5-4-1*, 5-4-2, 5-4-3 Module, object Module, relocatable 3-6-3 3-4-1, 5-5-1* Module, text MODULES 3-6-7, 3-6-11, 3-6-14 MOUNT 6-1-4, D-3* MOVEDK 3-4-4 3-2-2, 3-3-4, 4-1-4*, 4-1-10*, C-40* MSACCES 3-3-4, 4-1-4*, 4-1-10*, C-40* MSAUDIT 4-1-10* MSCATLIST **MSCHANG** 3-2-2, 3-3-4, 4-1-1, $4-1-4^*$, $4-1-10^*$, C-41* 3-2-2, 3-3-4, 4-1-4*, 4-1-10*, C-43* MSFETCH C-3* MSG MSPASSW 1-2-7, 3-3-4, 4-1-1, 4-1-4*, 4-1-11*, C-44* 3-2-2, 3-3-4, 4-1-5*, 4-1-11*, C-44* MSPURGE 3-1-10, 3-3-4, 4-1-1*, 4-1-3, F-2 MSS MSSAUDIT 4-1-7* 4-1-7* MSSBACKUP MSSBACKUP DELETE 4-1-7 MSSBACKUP FETCH 4-1-7 MSSBACKUP LIST 4-1-7 MSSBACKUP STORE 4-1-7 4-1-7* MSSDELETE MSSNEWCHRG 4-1-1, 4-1-8* MSSTORE 3-2-2, 3-3-4, 4-1-5*, 4-1-11*, C-45* 1-2-2* Multiple accounts 1-5-1*, 6-1-8 Multi-file tape NA C-3* 3-3-1 Name Name, host 5-1-5 Name, logical 5-1-3, D-2 Name server 5-1-5 Nastran 1-3-2 7-1-15* NASTRAN NAVSEA 5-1-5 Negative G-6, G-7, G-8, G-9

5-1-1, 5-1-5*

90/09/30 Rev0

1

Net-ODN Page Index-19

F-2, F-3 Network address Network diagram 1-1-5 1-1-2 Network ID 3-2-2, 3-3-4, 4-1-1, C-47* NEWCHRG NEWS 1-2-1*, 5-1-2* 1-2-1* Newsletter (CISD) No abort (NA) C-3 Node 1-1-3, 5-1-4 NODEFLIB 3-6-7 NOHOLD 3-2-2, C-47* NOLIST 3-4-3 7-1-18 Non-linear 3-2-1, C-47* NORERUN NORMAL job class 3-1-4*, C-37 1-1-2 nos NOS 6-1-6, A-4, E-1, F-2, G-8, G1-1 NOS station E-1 3-4-2 NOSEQ NOS/BE 6-1-6, A-4, G-8 NOS/VE 6-1-6, 6-1-7, E-1, F-2, G-9, G1-1 3-6-1 Note NOTE 3-2-3, C-48* NOTEXT 3-2-7 3-3-3 Null string 1-1-2*, F-2 N1 ^0 3-1-12 3-1-12 ^0 (CTRL-0) D-36* 1-1-5 OA VAXes F-3 oasys 5-1-5*, F-3 OASYS OASYS (OASYS) F-3 **Object library** 3-2-5, 3-5-1*, 5-4-1*, 5-4-2, 5-4-3 Object module 5-4-1*, 5-4-2, 5-4-3 Octal G-1, G-2, G-4 ODN C-3*

Rev0 90/09/30

| Office automation | F-3 |
|--|---|
| Office Automation System | 5-1-5*, F-3 |
| Off-line | 6-1-1 |
| Off-line work request | 6-1-2 |
| Off-station | 4-1-2 |
| OLDNews | 5-1-2 |
| One's complement | G-4 |
| Operating system | 1-1-1, F-1, F-2, F-3 |
| Operator, arithmetic | 3-2-7 |
| Operator, Central Site | 5-1-2 |
| Operator, logical | 3-2-7 |
| Operator, relational | 3-2-7 |
| OPTION | 3-2-1, C-48* |
| ORDER | 3-6-7 |
| \$OUT | 3-1-3* |
| Output | 3-1-8, C-3, D-33, D-35, D-37 |
| Output, CDC | D-10 |
| Output dataset | 3-1-3 |
| Outside user | 1-1-1 |
| Overlay | 3-6-12 |
| Overwrite file | C-55 |
| OWN | C-3* |
| Owner | C-3 |
| Page break | D-44 |
| PAM | C-3* |
| Paper, terminal | 1-2-8 |
| Parameter | 3-3-1, 3-3-2, C-1*, D-1 |
| Parameter, formal | 3-3-2, 3-3-3 |
| Parameter, keyword | 3-3-2, 3-3-3, C-1* |
| Parameter, positional | 3-3-2, 3-3-3, C-1* |
| Parameter substitution | 3-3-1, 3-3-3* |
| Parentheses | 3-3-3 |
| Parenthetical string | 3-2-7, 3-2-8 |
| Parity | 5-1-1* |
| Parity error | 6-1-2 |
| Parmeter | 3-3-1 |
| Pascal | 1-3-2, E-1 |
| PASCAL | 3-2-4, 7-1-16*, C-50* |
| PASSWOR Password Password, Login | 1-2-7, 4-1-1 1-2-1, 1-2-6*, 3-1-2, 4-1-1, 4-1-4, 4-1-10, 4-1-11, 5-1-1, 5-1-2, C-40, C-41, C-44, D-13 5-1-2 |

Patran 1-3-2 3-1-13, D-41* PAUSE PCA 1-3-2, 7-1-17* 3-2-7 PDMFC PDMST 3-2-6 C-3* PDN Perf. & Coverage Anlyzr (PCA) 7-1-17 3-1-17 PERIOD Peripheral 4-1-9 3-1-1, 4-1-3, C-3, C-4, C-5, C-8, Permanent dataset C-26, C-51, C-55, D-36 3-2-2 Permanent dataset management Permanent dataset staging 3-2-2 1-2-2, 4-1-3, 4-1-9, 5-1-2 Permanent file C-46 permission 3-2-2, 4-1-6, C-2, C-3, C-7, C-41, Permission C-44, C-45, C-51, C-55, D-27, D-44 3-2-2, C-51* PERMIT F-4* Personnel, administrative 1-3-2 Pert Time 3-4-1* PL 1-4-2, 3-1-13, 3-1-17, D-42* PLAY Plot 1-2-8 Plotting 7-1-5 1-3-2 PLOT10 PL/I 1-3-2 PM (Proj Mgt) 1-3-2 G-8, G-9 Point, binary Point, floating G-7 Position C-52, C-54 D-4 Position a tape 3-3-2, 3-3-3, C-1*, D-1 Positional parameter Positive G-6, G-7, G-8, G-9 7-1-5 Post-processing Precision G-6, G-7, G-9 PREFIX 3-1-17 3-6-7* PRESET Print 3-1-1, 3-1-8, D-6, D-10 PRINT 3-2-4, C-51* Print (on CDC CYBER 860) D-20 D-7, D-8 Printer, auxiliary 3-2-4, 3-3-1 PROC 3-3-1* **\$PROC** PROC 3-3-2, C-51* Procedure 3-2-4, 3-2-7, 3-3-1, 3-3-5, 5-3-1*, Page Index-22 Pro-Ref

Rev0

1

90/09/30

Procedure C-14, C-24, C-51, C-54, D-1 PROCLIB 3-3-4*, C-5 Program library 3-2-5, 3-4-1, C-10, C-62 1-3-2 Project Mgt PROLOG 1-4-2 1-4-1* Prolog file Prologue file 3-1-16* 3-1-11 Prompt (!) 3-3-1, 3-3-2*, 3-3-5, C-51 Prototype Punch A-4 1-2-8 Punch, card Purchase magnetic tape 1-2-8 4-1-1, 4-1-5, 4-1-11, C-44 Purge PURGE 3-4-4, 5-1-4 PURGEDK 3-4-4 3-1-4*, C-37 PZERO job class 3-1-8 QGET D-20* OPRINT D-22* QSUBMIT 3-2-3, C-52* **QUERY** Queue D-39, D-43, D-45, D-48 3-1-11, 3-1-13, 3-1-17, D-42* QUIT RCAUDIT D-23* RCDELETE D-24* D-25* RCGET D-27* RCSAVE D-29* RCSUBMIT READ 3-4-3 Read control word C-3 G-6, G-8, G-9 Rea1 REAL*16 G-7 REAL*4 G-7 REAL*8 G-7 RECORD 3-1-14, D-42* Record attribute D-10 C-23, C-58 Record, blocked Record length 3-6-8 F-4 Recorded message 4-1-1 Recover 3-6-8 Reentrant E-1 Reference

90/09/30 Rev0

Ref-SCR Page Index-23

I

| Refresh | D-34, D-44 |
|--------------------------|--|
| Refund request | 1-2-8 |
| Register | 3-2-7 |
| Register, G | 3-2-7 |
| Register, J | 3-2-7 |
| Registration | 3-2-7 C-30 1-2-1* 7-1-11 3-2-7 |
| Release | C-34 |
| RELEASE | 3-1-14, 3-2-2, C-53* |
| Relocatable | 3-6-10 |
| Relocatable module | 3-6-3 |
| Remove | C-26 |
| REMOVE | 3-1-14, D-43* |
| Rename file | C-41 |
| REPLACE_FILE | 4-1-9* |
| Representation, internal | G-2, G-6* |
| Reprieve | D-34, D-37 |
| REQUEST_MAGNETIC_TAPE | 6-1-7 |
| RERUN | 3-1-14, 3-2-1, C-54*, D-43* |
| RESOURC | 6-1-6 |
| Resource, job | C-20 |
| Restart | D-41 |
| RESTORE | 3-4-2 |
| RESUME | 3-1-17 |
| Return | C-53 |
| RETURN | 3-2-1, 3-2-4, 3-3-1, C-54* |
| Rewind | D-5 |
| REWIND | 3-2-3, 3-4-3, C-54* |
| RFTAPE | 6-1-4 |
| Ribbons | 1-2-8 |
| RIM | 1-3-2 |
| RKE terminal | F-2 |
| ROLLJOB | 3-2-1, C-54* |
| Root segment | 3-6-15 |
| SAVE | 3-2-2, 3-6-12, C-6, C-55*, D-43* |
| Scalar | 2-1-1, 3-1-1 |
| Schedule | F-1, F-2, F-3 |
| Scratch | 3-1-1, 3-6-11 |
| Screen | D-34 |
| SCRUBDS | 3-2-2, C-55* |

Page Index-24 SCU-SKI

Rev0

90/09/30

1-4-2 SCU EDITOR PROLOG Search order C-37 Sector C-58 Secure 1-1-3, 5-1-1, F-2 SECURE F-2 SECURE job class 3-1-4*, C-37 Security 1-2-6, 4-1-1 SEGLDR 3-2-5, 3-6-1, 3-6-10*, C-55*, E-1 SEGLDR directive 3-6-2* 3-6-1, 3-6-11, 3-6-12, 3-6-13 Segment SEGMENT 3-6-12, 3-6-14 3-6-10, C-55 Segmentation Segmentation caution 3-6-15 Semicolon 3-6-2, A-4 Sense switch 3-2-7, C-61, D-49 C-1* Separator 3-4-2 SEQ Sequence G-3 F-3 Sequent Service F-1, F-2, F-4 Service class 3 - 1 - 4SET 3-1-14, 3-2-1, 3-6-8, C-56* SET HOST 5-1-5 Set ID 1-5-1* D-4* SET MAGTAPE 1-2-6, 5-1-2* SET PASSWORD D-44*, D-45* SET TERMINAL SET FILE ATTRIBUTES 1-5-1 SET PASSWORD 1-2-7 SHOW 3-1-14 D-45* SHOW OUEUES SHOW SYSTEM 5-1-4 SHOW USERS 5-1-3 1-2-8 Shredder SI 1-5-1 3-2-5, C-56* SID Sign G-6, G-7, G-8, G-9 Simple Mail Trans. Protocol 5-1-7 3-3-1, 3-3-5 Simple procedure Simscript 1-3-2 Simulation 7-1-8 Size C-7 Size, dataset C-7 3-3-1 Skip 3-2-3, C-57* SKIPD

| | 90/09 | 9/30 | Rev0 |
|--|-------|------|------|
|--|-------|------|------|

SKI-Str Page Index-25

| SKIPF | 3-2-3, 3-4-3, C-57* |
|--------------------------|--------------------------------------|
| SKIPR | 3-2-3, C-58* |
| SKIPU | 3-2-3, C-58* |
| SKIP_TAPE_MARK | 6-1-7 |
| SLBUDG (Cray job class) | 3-1-4 |
| SLDEFER (Cray job class) | 3-1-4 |
| SLNORM (Cray job class) | 3-1-4 |
| Slot tape | 6-1-2, 6-1-3 |
| SLPZERO (Cray job class) | 3-1-4 |
| SMTP | 5-1-7 |
| SNAP | 3-1-14, D-46* |
| Software | 1-3-1* |
| SORT | 3-2-5, C-58* |
| Sort/Merge | C-58 |
| Source | 3-4-1, C-62 |
| Source program | 5-5-1 |
| Special functions | 7-1-9 |
| SPICE | 1-3-2, 7-1-18* |
| SPY | 1-3-2, 3-2-4, 7-1-17, C-59* |
| SQL | 7-1-11 |
| Sroll | D-44 |
| SSBUDG (Cray job class) | 3-1-4 |
| SSDEFER (Cray job class) | 3-1-4 |
| SSNORM (Cray job class) | 3-1-4 |
| SSPZERO (Cray job class) | 3-1-4 |
| SSW1-SSW6 | 3-2-7 |
| STACK | 3-6-8 |
| Stage | 3-1-1, 3-2-2, C-27, D-43, D-45, D-48 |
| Standard Query Language | 7-1-11 |
| Start execution | 3-6-9 |
| STATCLASS | 3-1-14, D-46* |
| Station command | D-30 |
| Station ID | 1-1-2*, 1-1-3 |
| Statistics | 7-1-9 |
| Status | C-52, D-38, D-39, D-47, F-4 |
| STATUS | 3-1-14, 3-1-17, D-47* |
| Storage | 6-1-1 |
| STORAGE | 3-1-14 |
| Storage, trillion-bit | F-2 |
| Store | 4-1-5, 4-1-11, C-45, D-35 |
| Stranger tape | 6-1-6 |
| String | C-2* |
| String comparison | 3-2-6 |
| String, literal | 3-2-7 |
| String, parenthetical | 3-2-7, 3-2-8 |

Rev0 Page Index-26 Str-Tap 7-1-15 Structural analysis 3-2-4 Structure, logic Subexpression 3-2-6 3-1-1, 3-1-10, 3-1-14, 3-2-2, 5-1-4, SUBMIT C-60*, D-48* 5-2-1, 5-4-1 Subprogram 7-1-6, 7-1-9, 7-1-13 Subprogram library 3-6-5 Substitute 3-3-1. 3-3-3* Substitution, parameter Subtopic, help 5-2-6 5-2-2*, 5-2-3 Sub-topic, help 1-2-8 Suggestions C-37 Summary Supercomputer 1-1-1 F-4 Support D-48 Suppress SUPPRESS 3-1-14, D-48* Suspend D-37, D-41 SUSPEND 3-1-17 3-1-14, 3-2-1, C-61*, D-49* SWITCH 3-2-7 Switch, sense 3-2-7, 5-1-3, 5-4-2 Symbol 5-4-2 Symbol, global Symbolic variable 3-2-6, C-52 SYMBOLS 3-6-8 SYSID 3-2-6 D-6*, D-10* **SYS\$ANAP** Tab 3-1-5, D-18 5-1-5 Table, host Table, symbol 3-6-4, 3-6-8 1-2-8, 1-5-1, 4-1-1, 6-1-4, 6-1-6, Tape 6-1-7, A-4, D-2, D-3, D-4 6-1-3 Tape assignment Tape, foreign 6-1-6 6-1-1 Tape formats Tape label 6-1-1 6-1-2, 6-1-3, F-4 Tape librarian 6-1-3 Tape, library Tape, magnetic 6-1-1* Tape mark D-5 1-5-1*, 6-1-8 Tape, multi-file Tape, slot 6-1-2, 6-1-3 6-1-6 Tape, stranger

90/09/30

90/09/30 Rev0



Page Index-27

1-3-2. 7-1-7* TAURUS 6-1-6 TDUMP 1-2-1, 5-1-1, F-4 Telephone TELNET 1-3-2 TERMDEF 1-4-1 Terminal 5-1-1*, D-42, D-44, D-45 Terminal paper 1-2-8 C-31, D-34, D-37, D-42 Terminate Terminator A-4, C-1* 1-2-8 Test magnetic tape C-3, C-48 Text C-3* TEXT 5-5-1* Text library Text module 3-4-1, 5-5-1* Time 3-1-4 TIME 3-2-6 C-59 Time usage TIMELEFT 3-2-6 TITLE 3-6-9 .TJL 5-1-2 5-2-2*, 5-2-3, 5-2-6 Topic, help TPGET 6-1-3 TPU (editor) 5-1-4, 5-6-1 Training F-4 Transfer 4-1-3, C-14 Transfer, file 7-1-12 Transfer file 5-1-6 Transfer funds 1-1-1 Transparent C-2, C-43, C-45 TREE 3-6-12, 3-6-13, 3-6-14 3-6-14 Tree diagram 3-6-11, 3-6-12, 3-6-13 Tree structure TRIAL 3-6-9 Trillion-bit storage F-2 Trouble form 1-2-8 TRUE G-5 Two's complement G-4 5-5-1 .TXT 3-2-3 UBBLOCK F-3 Ultrix-32 C-61* UNBLOCK 6-1-1, 6-1-4 Unblocked Unblocked dataset C-11, C-24, C-58, C-61

Page Index-28 Und-War

Rev0 90/09/30

| Underline | 3-3-3 |
|---|---|
| Underscore | 5-2-3 |
| UNICOS | 2-1-1* |
| Uninitialized data area | 3-6-7 |
| Unique access (UQ) | C-3*, C-7, C-39 |
| UNIX | F-3 |
| Unlabelled | 6-1-1, 6-1-6, 6-1-7 |
| Unlock | C-35 |
| Unsatisfied external | 3-6-9 |
| UNYANK | 3-4-4 |
| UPDATE | 3-2-5, 3-4-1*, C-10, C-62*, E-1 |
| Upper case | 3-4-1 |
| UPROC | 1-4-1 |
| Up-arrow | 3-1-11, 5-2-6 |
| UQ | C-3* |
| User initials | 1-1-1, 5-1-1, 5-1-2, G1-1 |
| User Initials | 1-2-1*, 3-1-11 |
| User Services | F-4 |
| Username | 5-1-2, 5-1-3, G1-1 |
| USX | 3-6-9 |
| UTILITIES | 5-2-1, 5-5-1, D-1 |
| Utility | 3-2-3 |
| UTILITY | 3-5-1, C-5 |
| Validate | C-5, C-7 |
| Variable | 3-2-7, C-56 |
| Variable length | 6-1-1 |
| Variable, symbolic | 3-2-6, C-52 |
| VAX | G-7 |
| VAXcluster Vector Vector register Verifier, card | 1-1-3, 1-3-1, 3-1-1, 3-1-5, 3-1-11, 4-1-6, 5-1-1*, 6-1-4, F-2 2-1-1, 3-1-1 C-30 1-2-8 |
| Version | 5-1-2, 5-1-4* |
| VERYoldnews | 5-1-2 |
| VMS | 5-1-1*, 6-1-4, D-1, D-43, E-1, F-2 |
| VMS Cray Station | 3-1-11, 3-1-12 |
| VSN | 1-5-1, 6-1-2, 6-1-3, 6-1-6 |
| V3 | 1-1-3*, F-2 |
| V4 | 1-1-3*, F-2 |

Warning

3-6-1, 5-1-2

| | 90/09/30 Rev0 | WEO-200 Page Index-29 |
|---|--|---|
| 1 | WEOF WFTAPE Width WIDTH Wildcard | 3-4-2 6-1-4 D-45 3-4-2 5-2-5, 5-4-2, 5-5-2, 5-5-3, C-4* |
| | Window WINS% WIN/TCP Word format | D-45 5-1-7 1-3-2 G-6, G-7, G-8, G-9 G-4 |
| | Word length Write control word WRITEDS WRITEF | G-1 C-3 3-2-3, C-64* 1-5-1 |
| | XER Xerox 8700 XMODEM | 3-6-9 D-19 1-3-2 |
| | YANK | 3-4-4 |
| | ^Z | 3-1-11, 3-1-12 |
| | | 3-1-11, 3-1-12, 5-2-6 |
| | ^Z (CTRL-Z) | D-36* |
| | Zero Zero-byte | C-55, G-3, G-5 A-4 |
| | 11/780, DEC VAX | F-3 |
| | 200-UT | 1-3-1 |

,

| Page Index-30 | 63 -PRO | Rev0 | 90/09/30 |
|---|---------|---|----------|
| 63-character set 6410, DEC VAX 64-character set | | A-4* 5-1-1, F-2 A-4*, G-3 | |
| 7-track | | 6-1-1, 6-1-6 | |
| 8250, DEC VAX 8550, DEC VAX 860, CDC CYBER 8-bit | | 1-1-3 1-1-3, 5-1-1, F-2 1-1-2 5-1-1 | |
| 9-track | | 6-1-1, 6-1-4, 6-1-6 | |
| SCS SDEBUG SDUMP SIN SLOG | | 3-1-3*, 3-3-1* 3-6-4 C-30 3-1-3* 3-1-3*, C-3/ | |
| \$OUT \$PROC | | 3-1-3* 3-3-1* | |

Initial Distribution

Copies:

Director 12 Defense Technical Information Center (DTIC) Camoron Station Alemandria, Virginia 23314

Center Distribution

Copies:

| 1 | 35/3509 | Gray, G. R. |
|-----|---------|---------------------------|
| 1 | 3501 | Minor, L. R. |
| 1 | 502د | Morris, J. |
| 1 | 351 | Strickland, J. D. |
| 150 | 3511 | Willner, S. E. |
| 20 | 3511 | Sommer, D. V. |
| 1 | 3512 | Brady, Mary |
| | 3513 | Wessel, J. |
| | 353 | Yearick, R. |
| 1 | 3533 | Hayden, H. P. |
| 1 | 3533 | Annapolis Computer Center |
| 1 | 355 | Kearney, E. |
| 1 | 3551 | Crum, Barbara |
| 1 | 3552 | Brady, Martha |
| 1 | 357 | |
| 1 | 3571 | Knowles, H. |
| 1 | 3572 | Smith, T. |
| 1 | 522 | TIC (C) |
| 1 | 522.2 | TIC (A) |
| | | |



.

