

HP OpenVMS

DCL ディクショナリ： A-M

AA-R1EAF-TE

2005 年 4 月

本書は，OpenVMS DCL コマンドおよびレキシカル関数について，例を挙げながら詳しく説明しています。

改訂 / 更新情報:

『OpenVMS DCL ディクショナリ』 V7.3-2 の改訂版です。

ソフトウェア・バージョン:

OpenVMS Alpha V8.2

OpenVMS I64 V8.2

OpenVMS VAX V7.3

日本ヒューレット・パッカード株式会社

© 2005 Hewlett-Packard Development Company, L.P.

本書の著作権は日本ヒューレット・パッカート株式会社が保有しており、本書中の解説および図、表は日本ヒューレット・パッカートの文書による許可なしに、その全体または一部を、いかなる場合にも再版あるいは複製することを禁じます。

また、本書に記載されている事項は、予告なく変更されることがありますので、あらかじめご承知おきください。万一、本書の記述に誤りがあった場合でも、日本ヒューレット・パッカートは一切その責任を負いかねます。

本書で解説するソフトウェア(対象ソフトウェア)は、所定のライセンス契約が締結された場合に限り、その使用あるいは複製が許可されます。

日本ヒューレット・パッカートは、日本ヒューレット・パッカートまたは日本ヒューレット・パッカートの指定する会社から納入された機器以外の機器で対象ソフトウェアを使用した場合、その性能あるいは信頼性について一切責任を負いかねます。

Microsoft, MS-DOS, Visual C++, Windows, および Windows NT は、米国およびその他の国の Microsoft 社の商標です。

Intel, Itanuim, Intel Itanuim Processor Family は米国およびその他の国の Intel 社の商標です。

Motif, OSF/1, および UNIX は米国およびその他の国の The Open Group の商標です。

Java およびすべての Java ベースのマークは、米国およびその他の国の Sun Microsystems 社の商標です。

その他のすべての商標および登録商標は、それぞれの所有者が保有しています。

原典：HP OpenVMS DCL Dictionary:A-M

© 2005 Hewlett-Packard Development Company, L.P.

本書は、日本語 VAX DOCUMENT V 2.1を用いて作成しています。

目次

まえがき	vii
! (コメント区切り文字)	DCLI-1
= (割り当て文)	DCLI-2
:= (文字列割り当て文)	DCLI-6
@ (プロシージャの実行)	DCLI-10
ACCOUNTING	DCLI-16
ALLOCATE	DCLI-17
ANALYZE/AUDIT	DCLI-21
ANALYZE/CRASH_DUMP	DCLI-22
ANALYZE/DISK_STRUCTURE	DCLI-23
ANALYZE/ERROR_LOG/ELV (Alpha/I64 のみ)	DCLI-24
ANALYZE/IMAGE	DCLI-25
ANALYZE/MEDIA	DCLI-38
ANALYZE/OBJECT	DCLI-39
ANALYZE/PROCESS_DUMP	DCLI-49
ANALYZE/RMS_FILE	DCLI-56
ANALYZE/SSLOG (Alpha/I64 のみ)	DCLI-57
ANALYZE/SYSTEM	DCLI-58
APPEND	DCLI-59
ASSIGN	DCLI-65
ASSIGN/MERGE	DCLI-73
ASSIGN/QUEUE	DCLI-75
ATTACH	DCLI-77
BACKUP	DCLI-79
CALL	DCLI-80
CANCEL	DCLI-85
CHECKSUM	DCLI-88
CLOSE	DCLI-94
CONNECT	DCLI-97
CONTINUE	DCLI-101
CONVERT	DCLI-103
CONVERT/DOCUMENT	DCLI-104
CONVERT/RECLAIM	DCLI-118
COPY	DCLI-119
COPY/FTP	DCLI-131
COPY/RCP	DCLI-134
CREATE	DCLI-136
CREATE/DIRECTORY	DCLI-140
CREATE/FDL	DCLI-144

CREATE/MAILBOX (Alpha/I64 のみ)	DCLI-145
CREATE/NAME_TABLE	DCLI-148
CREATE/TERMINAL	DCLI-152
DEALLOCATE	DCLI-159
DEASSIGN	DCLI-161
DEASSIGN/QUEUE	DCLI-166
DEBUG	DCLI-167
DECK	DCLI-172
DEFINE	DCLI-175
DEFINE/CHARACTERISTIC	DCLI-183
DEFINE/FORM	DCLI-185
DEFINE/KEY	DCLI-189
DELETE	DCLI-194
DELETE/BITMAP (Alpha/I64 のみ)	DCLI-200
DELETE/CHARACTERISTIC	DCLI-201
DELETE/ENTRY	DCLI-203
DELETE/FORM	DCLI-206
DELETE/INTRUSION_RECORD	DCLI-208
DELETE/KEY	DCLI-210
DELETE/MAILBOX (Alpha/I64 のみ)	DCLI-212
DELETE/QUEUE	DCLI-213
DELETE/QUEUE/MANAGER	DCLI-215
DELETE/SYMBOL	DCLI-217
DEPOSIT	DCLI-219
DIFFERENCES	DCLI-224
DIRECTORY	DCLI-235
DISABLE AUTOSTART	DCLI-250
DISCONNECT	DCLI-253
DISMOUNT	DCLI-255
DUMP	DCLI-261
EDIT/ACL	DCLI-272
EDIT/EDT	DCLI-273
EDIT/FDL	DCLI-278
EDIT/SUM	DCLI-279
EDIT/TECO	DCLI-280
EDIT/TPU	DCLI-284
ENABLE AUTOSTART	DCLI-285
ENDSUBROUTINE	DCLI-288
EOD	DCLI-289
EOJ	DCLI-291
EXAMINE	DCLI-292
EXCHANGE	DCLI-296
EXCHANGE/NETWORK	DCLI-297
EXIT	DCLI-308
FONT	DCLI-312
GOSUB	DCLI-313
GOTO	DCLI-316

HELP	DCLI-319
HELP/MESSAGE	DCLI-328
IF	DCLI-335
INITIALIZE	DCLI-339
INITIALIZE/QUEUE	DCLI-358
INQUIRE	DCLI-376
INSTALL	DCLI-379
JAVA	DCLI-380
JOB	DCLI-381
レキシカル関数	DCLI-388
F\$CONTEXT	DCLI-391
F\$CSID	DCLI-398
F\$CVSI	DCLI-400
F\$CVTIME	DCLI-402
F\$CVUI	DCLI-405
F\$DELTA_TIME	DCLI-407
F\$DEVICE	DCLI-408
F\$DIRECTORY	DCLI-411
F\$EDIT	DCLI-413
F\$ELEMENT	DCLI-415
F\$ENVIRONMENT	DCLI-417
F\$EXTRACT	DCLI-421
F\$FAO	DCLI-424
F\$FID_TO_NAME (Alpha/I64 のみ)	DCLI-431
F\$FILE_ATTRIBUTES	DCLI-432
F\$GETDVI	DCLI-436
F\$GETENV (Alpha のみ)	DCLI-449
F\$GETJPI	DCLI-450
F\$GETQUI	DCLI-458
F\$GETSYI	DCLI-480
F\$IDENTIFIER	DCLI-490
F\$INTEGER	DCLI-492
F\$LENGTH	DCLI-494
F\$LICENSE (Alpha/I64 のみ)	DCLI-495
F\$LOCATE	DCLI-496
F\$MESSAGE	DCLI-498
F\$MODE	DCLI-501
F\$MULTIPATH (Alpha/I64 のみ)	DCLI-503
F\$PARSE	DCLI-505
F\$PID	DCLI-509
F\$PRIVILEGE	DCLI-512
F\$PROCESS	DCLI-514
F\$SEARCH	DCLI-515
F\$SETPRV	DCLI-519
F\$STRING	DCLI-524
F\$TIME	DCLI-525
F\$TRNLNM	DCLI-527

F\$TYPE	DCLI-533
F\$UNIQUE (Alpha/I64 のみ)	DCLI-535
F\$USER	DCLI-537
F\$VERIFY	DCLI-538
LIBRARY	DCLI-541
LICENSE	DCLI-542
LINK	DCLI-543
LOGIN プロシージャ	DCLI-544
LOGOUT	DCLI-549
MACRO	DCLI-551
MAIL	DCLI-552
MERGE	DCLI-553
MESSAGE	DCLI-554
MONITOR	DCLI-555
MOUNT	DCLI-556

索引

表

DCLI-1 CPU 時間制限値の指定と処理	DCLI-365
DCLI-2 ワーキング・セット省略時の値，超過値，および制限値の決定	DCLI-373
DCLI-3 レキシカル関数の要約	DCLI-388
DCLI-4 FAO ディレクティブの要約	DCLI-426
DCLI-5 F\$FILE_ATTRIBUTES 項目	DCLI-432
DCLI-6 F\$GETDVI 項目	DCLI-437
DCLI-7 F\$GETJPI 項目	DCLI-451
DCLI-8 F\$GETQUI キーワード	DCLI-460
DCLI-9 F\$GETQUI 項目	DCLI-462
DCLI-10 F\$GETSYI 項目	DCLI-481
DCLI-11 F\$MESSAGE キーワード	DCLI-498
DCLI-12 コンテキスト・シンボル・タイプ	DCLI-533
DCLI-13 テープに対するキーワード	DCLI-572

まえがき

対象読者

本書は、HP OpenVMS オペレーティング・システムを使用するすべてのユーザを対象に書かれています。本書には、すべての DCL (DIGITAL Command Language) コマンドおよびレキシカル関数の説明が含まれています。コマンドに制限事項がある場合、または特定の特権が必要な場合は、そのように記述されています。

本書は、読者が『OpenVMS ユーザーズ・マニュアル』の内容を理解していることを前提にしています。

本書の構成

本書は、各コマンドおよびレキシカル関数の詳細な説明で構成されています。各コマンドはアルファベット順に並んでおり、各ページの一番上にはコマンド名が記述されています。レキシカル関数は、「レキシカル関数」(JOB コマンドの説明の後)内にまとめてあります。それぞれの関数は、その中でアルファベット順に並んでいます。

『OpenVMS DCL ディクショナリ』のハードコピー・マニュアルは上巻/下巻の2分冊で構成されています。上巻には、A から M で始まるコマンド(レキシカル関数を含みます)の説明が記述されています。下巻には、N から Z で始まるコマンドの説明が記述されています。

廃止された DCL コマンド、およびそれに代わるコマンドを、本書の付録 A (ハードコピー・マニュアルの場合は下巻に含まれる)に示します。

言語コンパイラやその他の OpenVMS オプションのソフトウェア製品のコマンドは、本書には含まれていません。これらのコマンドについては、製品とともに提供されるドキュメントを参照してください。

関連資料

OpenVMS オペレーティング・システムの概要、および DCL の使用方法については、『OpenVMS ユーザーズ・マニュアル』を参照してください。『OpenVMS ユーザーズ・マニュアル』は、初心者や会話型コンピュータ・システムの経験が少ないユーザを対象にしています。

『OpenVMS ユーザーズ・マニュアル』は、DCL コマンド言語の概要を説明します。また、DCL コマンドやレキシカル関数を使用したコマンド・プロシージャを作成する際の自習書としても使用できます。

各ユーティリティについての詳細は、それぞれのユーティリティのマニュアルを参照してください。それぞれのユーティリティのマニュアルには、ユーティリティを起動する DCL コマンド、ユーティリティ起動中に使用できるコマンド、および参照情報が記述されています。『OpenVMS DCL デクショナリ』は、各ユーティリティの簡単な説明とフォーマットについてのみ説明しています。

メッセージについての詳細は、オンラインの Help Message ユーティリティを参照してください。

HP OpenVMS 製品およびサービスについての追加情報は、弊社の Web サイトにアクセスしてください。URL は次のとおりです。

<http://www.hp.com/jp/openvms>

または

<http://www.hp.com/go/openvms>

本書で使用する表記法

本書では、次の表記法を使用しています。

表記法	意味
Ctrl/x	Ctrl/x という表記は、Ctrl キーを押しながら別のキーまたはポインティング・デバイス・ボタンを押すことを示します。
PF1 x	PF1 x という表記は、PF1 に定義されたキーを押してから、別のキーまたはポインティング・デバイス・ボタンを押すことを示します。
Return	例の中で、キー名が四角で囲まれている場合には、キーボード上でそのキーを押すことを示します。テキストの中では、キー名は四角で囲まれていません。 HTML 形式のドキュメントでは、キー名は四角ではなく、括弧で囲まれています。
...	例の中の水平方向の反復記号は、次のいずれかを示します。 <ul style="list-style-type: none">• 文中のオプションの引数が省略されている。• 前出の 1 つまたは複数の項目を繰り返すことができる。• パラメータや値などの情報をさらに入力できる。

表記法	意味
.	垂直方向の反復記号は、コードの例やコマンド形式の中の項目が省略されていることを示します。このように項目が省略されるのは、その項目が説明している内容にとって重要ではないからです。
()	コマンドの形式の説明において、括弧は、複数のオプションを選択した場合に、選択したオプションを括弧で囲まなければならないことを示しています。
[]	コマンドの形式の説明において、大括弧で囲まれた要素は任意のオプションです。オプションをすべて選択しても、いずれか 1 つを選択しても、あるいは 1 つも選択しなくても構いません。ただし、OpenVMS ファイル指定のディレクトリ名の構文や、割り当て文の部分文字列指定の構文の中では、大括弧に囲まれた要素は省略できません。
[]	コマンド形式の説明では、括弧内の要素を分けている垂直棒線はオプションを 1 つまたは複数選択するか、または何も選択しないことを意味します。
{ }	コマンドの形式の説明において、中括弧で囲まれた要素は必須オプションです。いずれか 1 つのオプションを指定しなければなりません。
太字	太字のテキストは、新しい用語、引数、属性、条件を示しています。
<i>italic text</i>	イタリック体のテキストは、重要な情報を示します。また、システム・メッセージ (たとえば内部エラー <i>number</i>)、コマンド・ライン (たとえば <i>PRODUCER=name</i>)、コマンド・パラメータ (たとえば <i>device-name</i>) などの変数を示す場合にも使用されます。
UPPERCASE TEXT	英大文字のテキストは、コマンド、ルーチン名、ファイル名、ファイル保護コード名、システム特権の短縮形を示します。
Monospace type	モノスペース・タイプの文字は、コード例および会話型の画面表示を示します。 C プログラミング言語では、テキスト中のモノスペース・タイプの文字は、キーワード、別々にコンパイルされた外部関数およびファイルの名前、構文の要約、または例に示される変数または識別子への参照などを示します。
—	コマンド形式の記述の最後、コマンド・ライン、コード・ラインにおいて、ハイフンは、要求に対する引数がその後の行に続くことを示します。
数字	特に明記しない限り、本文中の数字はすべて 10 進数です。10 進数以外 (2 進数、8 進数、16 進数) は、その旨を明記してあります。

! (コメント区切り文字)

コマンド行で、この記号の後はすべてコメントであり、コマンドの一部として処理されないことを示します。

フォーマット

!

例

```
$ !
$ WRITE SYS$OUTPUT "hello"      ! This command should output "hello".
hello
$ FOO = " "                      ! This command defines FOO as a blank.
$ FOO WRITE SYS$OUTPUT "hello" ! This command should output "hello".
hello
$ FOO = "!"                     ! This command defines FOO as a !.
$ FOO WRITE SYS$OUTPUT "hello" ! This command should be ignored.
$
```

= (割り当て文)

= (割り当て文)

文字列または整数値に対するシンボル名を定義します。

フォーマット

シンボル名=[=]式

シンボル名[ビット位置, サイズ] [=]置換式

注意

DCL コマンド名としてすでに使用されているシンボル名を割り当てないでください。IF, THEN, ELSE, および GOTO のようなシンボル割り当ては、コマンド・プロセスの実行を妨げる可能性がありますので、行わないでください。

パラメータ

シンボル名

シンボル名に対して、1 文字から 255 文字までの文字列を定義します。シンボル名には、DEC 補助文字 (DEC MCS) 文字セットの英数字、アンダースコア(_), またはドル記号(\$)を含めることができます。ただし、シンボル名は英字 (大文字と小文字は同じとみなします)、アンダースコア, またはドル記号から始めなければなりません。割り当て文に等号を 1 つ(=)指定すると、シンボル名を現在のコマンド・レベルのローカル・シンボル・テーブルに登録します。割り当て文に等号 2 つ(==)を指定すると、シンボル名をグローバル・シンボル・テーブルに登録します。

式

割り当て文の右辺に値を指定します。このパラメータは文字列、整数値、シンボル名、レキシカル関数、またはこれらの要素の組み合わせで構成できます。式に含まれる各構成要素を評価し、結果をシンボルに割り当てます。リテラル文字列は、すべて二重引用符(" ")で囲まなければなりません。シンボルを含む式を指定した場合には、シンボルの値を使用して式を評価します。

式を評価した結果は、文字列値または符号付きの整数値になります。式を評価した結果が文字列になる場合には、そのシンボルに文字列値を割り当てます。式を評価した結果が整数値の場合には、シンボルに整数値を割り当てます。整数値が 4 バイトのバッファに収まり切れない場合でも、エラー・メッセージは出力されません。

式中の演算子の概略と、式の指定方法についての詳細、および式の評価方法についての詳細は、『OpenVMS ユーザーズ・マニュアル』を参照してください。

DCL は、1024 バイトのバッファを使用して割り当て文を記憶し式を評価します。シンボル名、式、および式の計算の長さは 1024 バイトを超えることはできません。

[ビット位置、サイズ]

シンボル名の現在の 32 ビットの値に、2 進数値としての式の値を挿入することを指定します。この種の割り当て文は、シンボル名の現在の値を評価した後で、指定したビット数を置換式の結果と置き換えます。ビット位置は、ビット 0 を基準として相対位置であり、置き換えを実行する先頭の位置を指定します。置き換えるシンボルが整数の場合には、ビット位置は 32 より小さい値でなければなりません。またビット位置とサイズの合計も 32 以下でなければなりません。

置き換えるシンボルが文字列の場合には、ビット位置は 6152 より小さい値でなければなりません。各文字は 8 ビットを使って表現するため、768 文字目までならどの文字列からでも、置き換えを開始できます (768 文字目は、6144 というビット位置から始まります)。ビット位置とサイズの合計は、6152 以下でなければなりません。

サイズとは、置き換えるビット数です。32 より大きいサイズを指定すると、DCL はサイズを 32 に切り捨てます。

かぎ括弧は、必ず指定しなければなりません。シンボル名とかぎ括弧の間に、空白を入れることはできません。ビット位置とサイズの値は、整数式として指定します。

置換式

変更するシンボルを置き換えるために使用する、値を指定します。置換式は、整数式として使用します。

変更するシンボルが整数の場合には、置換式は、シンボルに割り当てられた値と置き換えられるビット・パターンを定義します。変更するシンボルが文字列の場合には、置換式の結果は、文字列の指定されたビットを置き換えるビット・パターンを定義します。変更しているシンボルが未定義の場合には、置換式の結果は空文字列を置き換えます。

説明

割り当て文を使用してシンボルを定義すると、コマンド言語を拡張することができます。会話型コマンド・レベルでは、シンボルを使用してコマンド、またはコマンド行の同義語を定義できます。コマンド・プロシージャ・ファイルでは、シンボルを使用して条件式に変数を代入することができます。

定義できるシンボルの最大数は、以下により決まります。

- 現在のプロセスでシンボル・テーブルおよびラベルを含めるためにコマンド・インタプリタが使用できる領域のサイズ

領域のサイズは、各プロセスごとにシステム・パラメータ CLISYMTBL で決まります。

= (割り当て文)

- シンボル名のサイズとその値

コマンド・インタプリタは、シンボル名とその値に対して領域を割り当てます。
また、各シンボルには数バイト多めに割り当てられます。

例

1. `$ LIST == "DIRECTORY"`

この割り当て文は、DCL コマンド DIRECTORY に対するグローバル・シンボル定義として、ユーザ定義同義語 LIST を定義します。

2.

```
$ COUNT = 0
$ LOOP:
$   COUNT = COUNT + 1
$   IF P'COUNT' .EQS. "" THEN EXIT
$   APPEND/NEW &P'COUNT' SAVE.ALL
$   DELETE &P'COUNT';*
$   IF COUNT .LT. 8 THEN GOTO LOOP
$ EXIT
```

このコマンド・プロシージャ COPYDEL.COM は、パラメータとして指定したファイルを SAVE.ALL というファイルに追加しています。追加後に、追加したファイルを削除します。8 個までファイル名が使用でき、各々シンボル P1, P2, ... に割り当てられます。

このコマンド・プロシージャはカウンタを使用して参照され、ループ内では、空文字列かどうかを IF 文で検査してから処理を行っています。この IF 文では、シンボル COUNT の値が文字 P に結合され、ループの 1 回目では P1 を、2 回目では P2 を、というように各々検査されます。P'COUNT' の評価後に、P1 や P2 等に相当するファイル名の置換が IF コマンドのコンテキストの中で自動的行われます。

APPEND や DELETE コマンドは、入力パラメータとしてファイル指定を必要とするので、自動的な置換を行いません。&P'COUNT' のアンパサンド(&)は、これらのコマンドでシンボル置換を強制的に行います。これらのコマンドがループ内で最初に検索された時には、COUNT はその現在の値に置換されます。コマンドの実行時には、アンパサンドで別の置換を行います。つまり、最初のファイル指定が P1 で置換され、2 番目のファイル指定は P2 で置換されます。

このコマンド・プロシージャを起動するには、次のコマンドを使用します。

```
$ @COPYDEL ALAMO.TXT BEST.DOC
```

ファイル ALAMO.TXT と BEST.DOC は、それぞれ SAVE.ALL に追加された後に削除されます。

```
3. $ A = 25
   $ CODE = 4 + F$INTEGER("6") - A
   $ SHOW SYMBOL CODE
      CODE = -15  HEX = FFFFFFFF1  Octal = 1777761
```

この例は、2つの割り当て文を含んでいます。最初の割り当て文は、25という値をシンボルAに割り当てます。2番目の割り当て文は、整数(4)、レキシカル関数(F\$INTEGER(6))、およびシンボルAを含む式を評価します。式の結果である-15が、シンボルCODEに割り当てられます。

```
4. $ FILENAME = "JOBSEARCH" - "JOB"
   $ FILETYPE = ".OBJ"
   $ FILESPEC = FILENAME + FILETYPE
   $ TYPE 'FILESPEC'
```

最初のコマンドでは、シンボルFILENAMEに“SEARCH”を割り当てます。文字列“SEARCH”は、式で実行された文字列削除演算の結果です。2番目のコマンドでシンボルFILETYPEに“.OBJ”を割り当て、3番目のコマンドでシンボルFILENAMEとFILETYPEを足して、FILESPECを作成しています。

シンボルFILENAMEとFILETYPEの値が連結されているので、FILESPECには文字列“SEARCH.OBJ”が割り当てられます。そしてシンボルFILESPECは、TYPEコマンドのパラメータとして使用されます。一重引用符(‘’)で、シンボルFILESPECをその値であるSEARCH.OBJに展開するように指定しています。そのため、TYPEコマンドでファイルSEARCH.OBJをタイプします。

```
5. $ BELL[0,32] = %X07
   $ SHOW SYMBOL BELL
      BELL = ""
```

この例では、シンボルBELLを算術置換文で作成しています。シンボルBELLは未定義であったため、空文字列に16進数の7が挿入されます。この値はASCIIコードでは、ターミナルのベルを鳴らすコードです。SHOW SYMBOL BELLコマンドを実行すると、ターミナルのベルが鳴ります。

シンボルBELLに整数値が定義済みであった場合には、BELLを表示すると、その新しい値が表示されます。

```
6. $ $=34
   %DCL-W-NOCOMD, no command on line - reenter with alphabetic first
   character
   $ $$=34
   $ SHOW SYMBOL $$
   %DCL-W-UNDSYM, undefined symbol - check validity and spelling
   $ SHOW SYMBOL $
   $ = 34  Hex = 00000022  Octal = 00000000042
```

ドル記号(\$)で始まるシンボルを使う場合、DCLが最初のドル記号を捨ててしまうので、ドル記号を2つ続けて(\$\$)使用してください。

:= (文字列割り当て文)

:= (文字列割り当て文)

文字列値に対してシンボル名を定義します。

フォーマット

シンボル名:=`[=]`文字列

シンボル名`[オフセット, サイズ]` :=`[=]`置換文字列

注意

DCL コマンド名としてすでに使用されているシンボル名を割り当てないでください。IF, THEN, ELSE, および GOTO のようなシンボルの割り当ては、コマンド・プロシージャの実行を妨げる可能性がありますので、行わないでください。

パラメータ

シンボル名

シンボル名に対して、1 文字から 255 文字までの文字列を定義します。シンボル名には、DEC 補助文字 (DEC MCS) 文字セットの英数字、アンダースコア(_), またはドル記号(\$)を含めることができます。ただし、シンボル名は英字 (大文字と小文字は同じとみなします)、アンダースコア, またはドル記号から始めなければなりません。割り当て文に等号を 1 つ(=)指定すると、シンボル名を現在のコマンド・レベルのローカル・シンボル・テーブルに登録します。割り当て文に等号 2 つ(==)を指定すると、シンボル名をグローバル・シンボル・テーブルに登録します。

文字列

シンボルに割り当てられる文字列値を指定します。この文字列には、英数字または特殊文字を含めることができます。DCL は、文字列割り当て文を記憶するために、1024 バイトのバッファを使用します。したがって、シンボル名、文字列、および文字列内に含まれるシンボル置換の長さは、1024 文字以下でなければなりません。

文字列割り当て文(:=)では、文字列リテラルを引用符(")で囲む必要はありません。この場合には、文字列値は自動的に大文字に変換されます。また、文字列の前後のスペースやタブは削除され、文字の間の複数のスペースやタブは 1 つのスペースに変換されます。

大文字への変換をしないようにしたり、または文字列に含まれる必要なスペースとタブをそのまま保存するには、文字列を二重引用符で囲まなければなりません。文字列の内部で二重引用符を使用する場合には、文字列全体を二重引用符で囲み、さらに文

:= (文字列割り当て文)

字列の中で二重引用符を指定する場所に二重引用符を 2 つ指定します。次の例を参照してください。

```
$ TEST := "this      is a ""test"" string"
$ SHOW SYMBOL TEST
TEST = "this      is a "test" string"
```

この例では、スペース、小文字、および二重引用符は、シンボル定義の中にそのまま保存されます。

シンボル割り当てを複数行に継続するには、継続文字としてハイフン(-)を使用します。次の例を参照してください。

```
$ LONG_STRING := THIS_SYMBOL_ASSIGNMENT_IS_A_VERY_LONG-
_$ _SYMBOL_STRING
```

文字列割り当て文を使って、空文字列をシンボルに割り当てるためには、割り当て文の右辺に文字列を指定しないようにします。次の例を参照してください。

```
$ NULL :=
```

文字列は、文字列リテラルとして指定するか、または文字列リテラルに評価されるシンボルかレキシカル関数として指定します。シンボルやレキシカル関数を使用する場合には、シンボル置換を要求するために一重引用符(' ')で囲みます。シンボル置換についての詳細は、『OpenVMS ユーザーズ・マニュアル』を参照してください。

また、フォーリン・コマンドを定義するために、文字列割り当て文を使用することもできます。フォーリン・コマンドについての詳細は『OpenVMS ユーザーズ・マニュアル』を参照してください。

[オフセット、サイズ]

シンボル値の一部が置換文字によって上書きされることを指定します。この形式の文字列割り当て文は、シンボルに割り当てられている値を求め、その値の一部(オフセットとサイズによって定義されている部分)を置換文字列と置き換えます。この場合、かぎ括弧は必ず必要であり、シンボル名と左側のかぎ括弧の間には、スペースを指定できません。

オフセットは、シンボル名の文字列値の先頭を基準にして、置換される最初の文字の相対位置を指定します。オフセットの値は 0 から始まります。

コマンドに指定したオフセットが、変更される文字列に含まれる最後の文字のオフセットより大きい場合には、文字列の最後と追加される置換文字列のオフセットの間に、スペースが挿入されます。指定できるオフセットの最大値は、768 です。

サイズは、置換される文字数を指定します。サイズの値は 1 から始まります。

:= (文字列割り当て文)

オフセットとサイズは、整数式として指定します。整数式についての詳細は『OpenVMS ユーザーズ・マニュアル』を参照してください。サイズとオフセットを加算した値は、769 より小さくなくてはなりません。

置換文字列

変更される文字列を上書きするために使用される文字列を指定します。置換文字列がサイズ引数より短い場合には、指定されたサイズと等しくなるように、置換文字列の右側に空白が挿入されます。その後で、置換文字列を使って、シンボル名に割り当てられた文字列が上書きされます。置換文字列がサイズ引数より長い場合には、置換文字列の右側の部分が指定されたサイズまで切り捨てられます。

置換文字列は、文字列リテラルとして指定するか、または文字列リテラルに評価されるシンボルやレキシカル関数として指定することができます。シンボルやレキシカル関数を使用する場合には、シンボル置換を要求するために、一重引用符(' ')で囲まなければなりません。シンボル置換についての詳細は『OpenVMS ユーザーズ・マニュアル』を参照してください。

例

1. `$ TIME := SHOW TIME`
`$ TIME`
14-DEC-2001 11:55:44

この例では、TIME というシンボルが、SHOW TIME というコマンド文字列と等しいと定義されます。この場合、シンボル名がコマンド文字列の最初の単語として指定されているため、コマンド・インタプリタは自動的にシンボル名を文字列値と置き換え、SHOW TIME コマンドを実行します。

2. `$ STAT := $DKA1:[TEDESCO]STAT`
`$ STAT`

この例では、STAT をフォーリン・コマンドとして定義する方法が示されています。STAT というシンボルは、ドル記号で始まってその後にファイル指定が続く文字列に等しいものとして定義されています。コマンド・インタプリタは、ファイル指定が実行可能なイメージのファイル指定であること、つまり、ファイル・タイプが EXE のファイルであると仮定します。

この後 STAT とタイプすると、コマンド・インタプリタはイメージを実行します。

:= (文字列割り当て文)

```
3. $ A = "this is a big    space."
   $ SHOW SYMBOL A
   A = "this is a big    space."
   $ B := 'A'
   $ SHOW SYMBOL B
   B = "THIS IS A BIG SPACE."
```

この例では、割り当て文と文字列割り当て文が比較されています。シンボル A は割り当て文を使って定義されているため、小文字や複数のスペースはそのまま保存されます。シンボル B は文字列割り当て文を使って定義されています。一重引用符('')が必要な点に注意してください。一重引用符を使用しないと、B というシンボル名はリテラル文字列 A に等しいものとして定義されてしまいます。文字列割り当て文を使用すると、シンボル A の値がシンボル B に割り当てられる際に、文字は大文字に変換され、複数のスペースは 1 つのスペースに変換されてしまいます。

```
4. $ FILE_NAME := MYFILE
   $ FILE_NAME[0,2] := OL
   $ SHOW SYMBOL FILE_NAME
   FILE_NAME = "OLFILE"
```

この例では、シンボル FILE_NAME に割り当てられた文字列の最初の 2 文字を、文字列 OL に変更する置換式が示されています。オフセット 0 は文字列の最初の文字を指定し、サイズ指定の 2 は、文字列の長さを指定しています。

```
5. $ FILE_NAME := MYFILE
   $ FILE_TYPE := .TST
   $ FILE_NAME[F$LENGTH(FILE_NAME),4] := 'FILE_TYPE'
   $ SHOW SYMBOL FILE_NAME
   FILE_NAME = "MYFILE.TST"
```

この例では、シンボル FILE_NAME に文字列 MYFILE を、シンボル FILE_TYPE に文字列.TST を各々代入しています。3 番目の割り当て文では、レキシカル関数 F\$LENGTH を用いて、上書きを始めるオフセット値を特定しています。シンボル FILE_TYPE は、置換文字列(.TST)を参照するために用いられます。シンボルの展開を要求するために、一重引用符('')を指定しています。

レキシカル関数 F\$LENGTH は、シンボル FILE_NAME の文字列の長さを返します。この値は、オフセットとして使用されます。シンボル FILE_TYPE の文字列の 4 文字を、FILE_NAME の文字列の最後に追加しています。シンボル FILE_NAME の結果は、文字列 MYFILE.TST になります。

@ (プロシージャの実行)

コマンド・プロシージャを実行します。または、コマンド・インタプリタに対して、特定のファイルまたは装置からのコマンド入力を読み込むよう要求します。

フォーマット

@ ファイル指定 [パラメータ[,...]]

パラメータ

ファイル指定

前のコマンドに対する入力を読み込まれる装置またはファイル、あるいは実行されるコマンド・プロシージャを指定します。省略時のファイル・タイプは.COMです。アスタリスク(*)やパーセント記号(%)などのワイルドカード文字を、ファイル指定で使用することはできません。

パラメータ[,...]

コマンド・プロシージャに渡される1つから8つの省略可能なパラメータを指定します。これらのパラメータには、文字列値を入力した順に(P1, P2, ... P8)の8つのシンボルが割り当てられます。シンボルは、指定されたコマンド・プロシージャの内部でのみ有効です。各パラメータは、1つまたは複数の空白で区切ります。空パラメータを指定するには、連続する2つの引用符("")を使用します。パラメータには、英数字または特殊文字を含む文字列値を指定することができます。ただし次の制約があります。

- コマンド・インタプリタは英字を大文字に変換し、各パラメータの区切り文字に空白を使用します。空白や小文字を含むパラメータを渡すには、パラメータを引用符で囲む必要があります。
- 最初のパラメータがスラッシュ(/)から始まる場合には、パラメータを引用符(" ")で囲まなければなりません。
- リテラルとしての引用符や空白を含むパラメータを渡すには、文字列全体を二重引用符で囲み、文字列の中で連続する2つの二重引用符を指定します。たとえば、コマンド・プロシージャ TEST.COM が次のコマンド行を含んでいたとします。

```
$ WRITE SYS$OUTPUT P1
```

この時、DCL プロンプト(\$)に対して次のコマンドを入力します。

```
$ @TEST "Never say ""quit"""
```

TEST.COM というプロシージャを実行すると、パラメータ P1 には次の文字列が割り当てられます。

```
Never say "quit"
```

文字列に引用符が含まれており、空白が含まれていない場合には、引用符は文字列の中でそのまま保存され、引用符で囲まれた英字は小文字のまま保存されます。たとえば、DCL プロンプトに対して次のコマンドを入力します。

```
$ @TEST abc"def"ghi
```

TEST.COM というプロシージャを実行すると、パラメータ P1 には次の文字列が割り当てられます。

```
ABC"def"GHI
```

シンボルをパラメータとして使用する場合には、シンボル置換を実行するためにシンボルを一重引用符(' ')で囲む必要があります。次の例を参照してください。

```
$ NAME = "JOHNSON"
$ @INFO 'NAME'
```

一重引用符を使用すると、“JOHNSON”という値が NAME というシンボルと置き換えられます。したがって、“JOHNSON”というパラメータが、P1 として INFO.COM に渡されます。

説明

@コマンドは、以下を含むコマンド・プロシージャを実行する際に使用します。

- DCL コマンド行、またはデータ、あるいはその両方
- 特定のコマンド行に対する修飾子、またはパラメータ、あるいはその両方

コマンド、またはデータ、あるいはその両方を含むコマンド・プロシージャを実行するには、コマンド行の最初に@コマンドを入力し、その次にコマンド・プロシージャ・ファイルの名前を指定します。コマンド・プロシージャには DCL コマンド、および現在実行中のコマンドまたはプログラムへの入力データを含めることができます。コマンド・プロシージャ内のすべての DCL コマンドは、ドル記号(\$)で始めなければなりません。コマンド行がハイフン(-)により継続している場合は、その継続行はドル記号から始まらなくても構いません。

コマンド・プロシージャ中で、行の最初にドル記号がなく、継続行でもないコマンド行は、現在実行中のコマンドまたはプログラムへの入力データとして扱われます。DECK コマンドを使用すると、データのレコード位置にドル記号があることを指定できます。

@ (プロシージャの実行)

コマンド・プロシージャ内に、他のコマンド・プロシージャを実行する@コマンドを含めることもできます。トップ・レベルのコマンド・プロシージャも含め、最大で 16 回までコマンド・プロシージャをネストさせることができます。SUBMIT コマンドを使用する、またはシステム・カード・リーダにコマンド・プロシージャを含むカード・デッキを置くことにより、コマンド・プロシージャをバッチ・ジョブとしてキューに登録することもできます。

特定のコマンド行に対する修飾子、またはパラメータ、あるいはその両方を含むコマンド・プロシージャを実行するには、通常、コマンド行で修飾子またはパラメータを指定する位置に@コマンドを置きます。次に、修飾子またはパラメータを含むコマンド・プロシージャ・ファイル名を指定します。

コマンド・プロシージャ・ファイルがコマンドに対するパラメータで始まる場合は、@コマンドの前に空白が必要です。次の例を参照してください。

```
$ CREATE TEST.COM
TIME
Ctrl/Z
$ SHOW @TEST
14-DEC-2001 17:20:26
```

反対に、コマンド・プロシージャ・ファイルがコマンドに対する修飾子で始まる場合は、@コマンドの前に空白を入れてはいけません。

```
$ CREATE TEST_2.COM
/SIZE
Ctrl/Z
$ DIR@TEST_2

Directory WORK$:[SCHEDULE]

JANUARY.TXT;8      14-DEC-2001 15:47:45.57
FEBRUARY.TXT;7     14-DEC-2001 15:43:16.20
MARCH.TXT;6        14-DEC-2001 11:11:45.74
.
.
.
Total of 11 files.
```

コマンド・プロシージャ・ファイルにパラメータ、修飾子、あるいはその両方が含まれている場合は、コマンド行をドル記号で始めてはいけません。@ファイル指定に続くコマンド行のデータは、プロシージャに対するパラメータとして扱われます。

修飾子

/OUTPUT=ファイル指定

コマンド・プロシージャの出力を書き込むファイル名を指定します。省略時の設定では、出力は現在の SYS\$OUTPUT 装置に書き込まれます。省略時の出力ファイル・

タイプは.LIS です。アスタリスク(*)およびパーセント記号(%)ワイルドカード文字を、ファイル指定で使用することはできません。システムからの応答とエラー・メッセージは、指定したファイルと SYS\$COMMAND の両方に書き込まれます。/OUTPUT 修飾子を指定する場合は、コマンド・プロシージャのファイル指定のすぐあとに修飾子を指定しなければなりません。そのように指定しないと、/OUTPUT 修飾子はコマンド・プロシージャに渡されるパラメータとして解釈されます。

SYS\$OUTPUT の定義を変更すれば、コマンド・プロシージャからの出力先を変更することができます。コマンド・プロシージャの 1 行目として次のコマンドを指定すると、出力は指定したファイルに送られます。

```
$ DEFINE SYS$OUTPUT ファイル指定
```

このプロシージャが終了すると、SYS\$OUTPUT はもとの等価文字列に戻されます。この方法では、コマンド・プロシージャを実行するときに/OUTPUT 修飾子を使用した場合と同じ結果になります。

例

```
1. $ CREATE DOFOR.COM
   $ ON WARNING THEN EXIT
   $ IF P1.EQS." " THEN INQUIRE P1 FILE
   $ FORTRAN/LIST 'P1'
   $ LINK 'P1'
   $ RUN 'P1'
   $ PRINT 'P1'
   Ctrl/Z
   $ @DOFOR AVERAGE
```

この例では、DOFOR.COM というコマンド・プロシージャが示されています。このコマンド・プロシージャは、プログラムをコンパイルし、リンクし、実行するために、それぞれ FORTRAN コマンド、LINK コマンド、および RUN コマンドを実行します。ON コマンドはコマンドの実行結果が警告状態やエラー状態である場合には、プロシージャの実行を継続しないことを要求しています。

DOFOR.COM を実行するときに、FORTRAN プログラムのファイル指定をパラメータ P1 として渡すことができます。プロシージャを実行するときに P1 の値を指定しなかった場合には、INQUIRE コマンドがターミナルにプロンプト・メッセージを表示し、そのプロンプトに対して入力した値が、シンボル P1 に割り当てられます。この例では、ファイル名 AVERAGE が P1 に割り当てられています。FORTRAN コマンド、LINK コマンド、RUN コマンド、および PRINT コマンドには、それぞれ省略時のファイル・タイプが設定されているため、ファイル・タイプは含まれていません。

@ (プロシージャの実行)

2. \$ @MASTER/OUTPUT=MASTER.LOG

このコマンドは、MASTER.COM という名前のプロシージャを実行します。出力はすべて MASTER.LOG ファイルに書き込まれます。

3. \$ CREATE FILES.COM
*.FOR, *.OBJ
Ctrl/Z
\$ DIRECTORY @FILES

この例は FILES.COM コマンド・プロシージャを示しています。このファイルには、DCL コマンド行へのパラメータが含まれています。ファイル全体が DCL により、コマンド入力として処理されます。このプロシージャを DIRECTORY コマンドの後で実行して、現在の省略時のディレクトリ内の FORTRAN ソース・ファイルとオブジェクト・ファイルの一覧を表示させています。

4. \$ CREATE QUALIFIERS.COM
/DEBUG/SYMBOL_TABLE/MAP/FULL/CROSS_REFERENCE
Ctrl/Z
\$ LINK SYNAPSE@QUALIFIERS

この例では、LINK コマンドに対する修飾子を含んでいる QUALIFIERS.COM というコマンド・プロシージャが示されています。LINK コマンドを入力する時に、リンクするファイルのファイル指定のすぐ後に、コマンド・プロシージャを指定します。この時、ファイル指定と@コマンドの間には空白を入れてはいけません。

5. \$ CREATE SUBPROCES.COM
\$ RUN 'P1' -
/BUFFER_LIMIT=1024 -
/FILE_LIMIT=4 -
/PAGE_FILES=256 -
/QUEUE_LIMIT=2 -
/SUBPROCESS_LIMIT=2 -
'P2' 'P3' 'P4' 'P5' 'P6' 'P7' 'P8'
Ctrl/Z
\$ @SUBPROCES LIBRA /PROCESS_NAME=LIBRA

この例は、SUBPROCES.COM というコマンド・プロシージャを示しています。このプロシージャは、イメージを実行するためのサブプロセスを生成する RUN コマンドを実行し、また、サブプロセス生成のためのクォータを定義する修飾子を含んでいます。実行するイメージ名は、パラメータ P1 で渡されます。パラメータ P2 から P8 は、追加する修飾子を指定するために使用できます。

この例では、P1 に LIBRA というファイル名が割り当てられます。これがサブプロセスで実行するイメージの名前です。P2 には/PROCESS_NAME=LIBRA という修飾子が割り当てられます。これが RUN コマンドに追加される修飾子です。


```

6. $ CREATE EDOC.COM
   $ ASSIGN SYS$COMMAND: SYS$INPUT
   $ NEXT:
   $     INQUIRE NAME "File name"
   $     IF NAME.EQS." " THEN EXIT
   $     EDIT/TPU 'NAME'.DOC
   $     GOTO NEXT
   Ctrl/Z
   $ @EDOC

```

この EDOC.COM というプロシージャは、EVE エディタを起動します。編集セッション終了時に、ラベル NEXT でプロシージャはループします。各ループにおいて、編集するファイル名を要求します。省略時のファイル・タイプは.DOC です。INQUIRE コマンドへの応答に空行が入力されると、EXIT コマンドでプロシージャが終了します。

ASSIGN コマンドで、プロシージャ実行中に SYS\$INPUT の等価名を変更しています。この変更により、EVE エディタがコマンド・プロシージャ・ファイルではなく、ターミナルから入力データを読むことができます。SYS\$INPUT が変更されない場合には、省略時の入力データ・ストリームはコマンド・プロシージャ・ファイルになります。コマンド・プロシージャ終了時に、SYS\$INPUT は解除され元の設定に戻ります。

```

7. ! PEOPLE.DAT
   ! A set of data with embedded key qualifiers for the SORT command.
   !
   ! Usage: SORT@PEOPLE.DAT
   !
   /KEY=(POS:10,SIZE:10) sys$input people.out
   Fred      Flintstone  555-1234
   Barney    Rubble      555-2244
   Wilma     Flintstone  555-1234
   Betty     Rubble      555-2244
   George    Slate      555-8911
   Dino      Dinosaur    555-1234
   $!
   $ purge people.out
   $ type people.out

```

ソートされた名簿を PEOPLE.OUT ファイルに作成し、それを表示します。この例では、DCL コマンドの途中で "@" を使用した場合に、DCL がそのファイル全体をコマンド入力として処理することを示します。

ACCOUNTING

Accounting ユーティリティを起動します。Accounting ユーティリティはリソースの使用に関するレポートを作成します。

Accounting ユーティリティについての詳細は、『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』あるいはオンライン・ヘルプを参照してください。

フォーマット

ACCOUNTING [ファイル指定[,...]]

ALLOCATE

装置の割り当てを解除するまで、またはそのプロセスを終了するまで、装置へ排他的にアクセスすることができます。オプションで、装置に論理名をつけることができます。

読み込み (R)、書き込み (W)、または制御アクセス権が必要です。

フォーマット

ALLOCATE 装置名[:][,...] [論理名[:]]

パラメータ

装置名[:][,...]

物理装置名、または物理装置名に変換される論理名を指定します。装置名は汎用名でも構いません。コントローラまたはユニット番号が指定されていない場合、指定した部分を満たすすべての装置が割り当てられます。複数の装置を指定した場合は、最初の使用可能な装置が割り当てられます。

論理名[:]

1 から 255 文字までの文字列を指定します。文字列に空白を含めたい場合は、文字列全体を一重引用符(' ')で囲みます。最後のコロン(:)は使用されません。ここで指定した論理名は、プロセス論理名になります。論理名を明示的に削除するまで、またはこのプロセスを終了するまで、論理名は有効です。

修飾子

/GENERIC

/NOGENERIC (省略時の設定)

最初のパラメータが、装置名ではなく装置型であることを示します。装置型の例としては RX50、RD52、TK50、RC25、RCF25、RL02 が挙げられます。指定した名前と型を持つ、最初のまだ割り当てられていない装置を占有します。

/[NO]GENERIC 修飾子は、ALLOCATE コマンドの装置名パラメータの前に指定します。たとえば DCL プロンプト (\$) に対して次のコマンドを実行すると、RK07 装置を占有できます。

```
$ ALLOCATE/GENERIC RK07 DISK
```

/GENERIC 修飾子で指定できる装置型の一部を、次の表に示します。どの装置が使用できるかは、現在使用している OpenVMS のバージョンに対応した SPD をご覧ください。

装置の種類				
ディスク装置				
EF51	EF52	EF53	EF54	EF58
ESE20	ESE25	ESE52	ESE56	ESE58
EZ31	EZ31L	EZ32	EZ32L	EZ33
EZ33L	EZ34	EZ35	EZ51	EZ52
EZ53	EZ54	EZ56R	EZ58	HSZ10
HSZ15	HSZ20	HSZ40	ML11	RA60
RA70	RA71	RA72	RA73	RA80
RA81	RA82	RA90	RA92	RAH72
RB02	RB80	RC25	RCF25	RD26
RD31	RD32	RD33	RD51	RD52
RD53	RD54	RF30	RF31	RF31F
RF32	RF35	RF36	RF37	RF70
RF71	RF72	RF73	RF74	RF75
RFF31	RFH31	RFH32	RFH35	RFH72
RFH73	RK06	RK07	RL01	RL02
RM03	RM05	RM80	RP04	RP05
RP06	RP07	RP07HT	RX01	RX02
RX04	RX18	RX23	RX23S	RX26
RX33	RX33S	RX35	RX50	RZ01
RZ13	RZ14	RZ15	RZ16	RZ17
RZ18	RZ22	RZ23	RZ23L	RZ24
RZ24L	RZ25	RZ25L	RZ26	RZ26B
RZ26L	RZ26M	RZ27	RZ27B	RZ27L
RZ28	RZ28B	RZ28L	RZ29	RZ29B
RZ31	RZ34L	RZ35	RZ35L	RZ36
RZ36L	RZ37	RZ38	RZ55	RZ55L
RZ56	RZ56L	RZ57	RZ57I	RZ57L
RZ58	RZ59	RZ72	RZ73	RZ73B
RZ74	RZ74B	RZ75	RZ75B	RZF01
コンパクト・ディスク装置				
RRD40	RRD40S	RRD42	RRD43	RRD44
RRD50	RV20	RV60	RV80	RW504
RW510	RW514	RW516	RWZ01	RWZ21
RWZ31	RWZ51	RWZ52	RWZ53	RWZ54

装置の種類				
テープ装置				
TA78	TA79	TA81	TA85	TA86
TA87	TA90	TA90E	TA91	TAD85
TAPE9	TD34	TD44	TE16	TF30
TF70	TF85	TF86	TK50	TK50S
TK60	TK70	TK70L	TKZ09	TKZ60
TL810	TL820	TLZ04	TLZ06	TLZ07
TLZ6	TLZ7	TM32	TS11	TSZ05
TSZ07	TSZ08	TU45	TU56	TU58
TU77	TU78	TU80	TU81	TZ30
TZ30S	TZ85	TZ857	TZ86	TZ865
TZ867	TZ87	TZ875	TZ877	TZ88
TZ885	TZ887	TZ89	TZ895	TZ897
TZK10	TZK11	TZX0		

/LOG (省略時の設定)

/NOLOG

装置の名前が割り当てられたことを示すメッセージを表示します。他の装置に割り当てられている論理名を別の装置に割り当てようとすると、論理名が指す装置が変更されることを示すメッセージが表示されます。

例

1. \$ ALLOCATE DMB2:
%DCL-I-ALLOC, _DMB2: allocated

この例で ALLOCATE コマンドは、コントローラ B ユニット 2 である、RK06 /RK07 ディスク・ドライブの占有を要求しています。システムの応答は、装置の占有が成功したことを示しています。

2. \$ ALLOCATE MT,MF: TAPE:
%DCL-I-ALLOC, _MTB2: allocated
.
.
.
\$ SHOW LOGICAL TAPE:
TAPE: = _MTB2: (process)
\$ DEALLOCATE TAPE:
\$ DEASSIGN TAPE:

この例の ALLOCATE コマンドは、MT または MF で始まる名前のテープ装置の占有と、それに論理名 TAPE を割り当てよう要求しています。ALLOCATE コマンドは MT で名前が始まる使用可能なテープ装置を探し、占有した装置名とともに応答します。MT で名前が始まるテープ装置がない場合、ALLOCATE コマンドは MF で名前が始まるテープ装置を探します。これ以降、ユーザのプログラ

ムまたはコマンド文字列中の装置 TAPE への参照は、装置名 MTB2 に変換されません。

このテープ装置が必要なくなったら、DEALLOCATE コマンドを使用して割り当てを解除し、DEASSIGN コマンドを使用して論理名を削除します。ALLOCATE コマンドではコロンをつけて論理名 TAPE を指定しましたが、論理名テーブルのエントリにはコロンは含まれていません。

3. \$ ALLOCATE/GENERIC RL02 WORK
 %DCL-I-ALLOC, _DLA1: allocated
 %DCL-I-SUPERSEDE, previous value of WORK has been superseded

この例の ALLOCATE コマンドは、すべての RL02 ディスク装置の占有と、それに論理名 WORK を割り当てるよう要求しています。終了メッセージは、占有した装置を表示し、論理名 WORK の割り当てが変更されたことを示します。

4. \$ ALLOCATE \$TAPE1
 %DCL-I-ALLOC, _MUA0: allocated

この例の ALLOCATE コマンドは、論理名 \$TAPE1 に関連付けられているテープ装置 MUA0 を占有します。

5. \$ ALLOCATE /GENERIC RX50 ACCOUNTS

この例の ALLOCATE コマンドは、最初の使用可能なディスクセット・ドライブを占有し、それにプロセス論理名 ACCOUNTS を割り当てます。

ANALYZE/AUDIT

Audit Analysis ユーティリティを起動します。このユーティリティは、機密保護監査ログ・ファイル、または機密保護アーカイブ・ファイルから、情報を選択的に抽出し表示します。

Audit Analysis ユーティリティについての詳細は、『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』あるいはオンライン・ヘルプを参照してください。

フォーマット

ANALYZE/AUDIT [ファイル指定]

ANALYZE/CRASH_DUMP

システム・ダンプ・アナライザ (SDA)・ユーティリティを起動します。このユーティリティは、システム・ダンプ・ファイルを分析します。/CRASH_DUMP 修飾子は必須です。

Alpha システムにおけるシステム・ダンプ・アナライザ・ユーティリティについての詳細は、『OpenVMS System Analysis Tools Manual』あるいはオンライン・ヘルプを参照してください。VAX システムにおけるシステム・ダンプ・アナライザ・ユーティリティについての詳細は、『OpenVMS VAX System Dump Analyzer Utility Manual』¹を参照してください。

フォーマット

ANALYZE/CRASH_DUMP ファイル指定

説明

システム・ダンプ・アナライザ (SDA)・ユーティリティを起動します。このユーティリティは、システム・ダンプ・ファイルを分析します。/CRASH_DUMP 修飾子は必須です。

OpenVMS Alpha システム

ANALYZE/CRASH_DUMP コマンドはプロセス・ダンプにも使用できます。ただし、プロセス・ダンプには、ダンプ内のすべての情報にアクセスできる ANALYZE /PROCESS コマンドの方が適しています。

¹ このマニュアルはアーカイブされています。すでにメンテナンスされておらず、OpenVMS のドキュメント・セットにも含まれていません。ただし、<http://www.hp.com/go/openvms/doc>からオンラインで参照するか、オンライン・ヘルプで参照することができます。

ANALYZE/DISK_STRUCTURE

Analyze/Disk_Structure ユーティリティを起動します。このユーティリティは、次の操作を行います。

- Files-11 オンディスク構造レベル 1、レベル 2、およびレベル 5 ディスク・ボリュームの、可読性と有効性をチェックします。
- エラーと整合性を報告します。

/DISK_STRUCTURE 修飾子は必須です。

Analyze/Disk_Structure ユーティリティについての詳細は、『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』あるいはオンライン・ヘルプを参照してください。

フォーマット

ANALYZE/DISK_STRUCTURE 装置名:[/qualifier]

ANALYZE/ERROR_LOG/ELV (Alpha/I64 のみ)

1 つあるいは複数のエラー・ログ・ファイルを選択して内容を報告する Error Log Viewer (ELV) を起動します。このユーティリティは OpenVMS Version 7.3 以降が稼動しているシステム上で書き込まれたエラー・ログを扱う場合に最も便利なツールです。Error Log Viewer についての詳細は、『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』またはオンライン・ヘルプを参照してください。

OpenVMS Version 7.2*システムで書き込まれたエラー・ログに関しては、DECevent ユーティリティを起動する DIAGNOSE コマンドを使用する必要があります。DECevent はサポートされなくなりましたが、必要なユーザは、ソフトウェアと関連マニュアルを次の Freeware の Web サイトからダウンロードできます。

<http://h71000.www7.hp.com/openvms/freeware/>

OpenVMS Version 7.2 よりも前のバージョンで書き込まれたエラー・ログに関しては、ANALYZE/ERROR_LOG コマンドを使用します。このコマンドは、Error Log Report Formatter (ERF) を起動します。ERF のマニュアルは、次の Freeware の Web サイトにあります。

<http://h71000.www7.hp.com/openvms/freeware/>

フォーマット

ANALYZE/ERROR_LOG/ELV [コマンド]

ANALYZE/IMAGE

OpenVMS VAX および Alpha システムの実行可能イメージ・ファイルや共用可能イメージ・ファイルと、OpenVMS I64 システムの ELF (Executable and Linkable Format) イメージ・ファイルや共用可能イメージ・ファイルの内容を分析し、ファイル中のエラーを見つけます。分析には、I64 および Alpha システム上で変換されたイメージも含まれます。/IMAGE 修飾子は必須です。

イメージ・ファイルに関する一般的な情報については、『OpenVMS Linker Utility Manual』のリンクの説明を参照してください。オブジェクト・ファイルの内容を分析する場合は、ANALYZE/OBJECT コマンドを使用します。

フォーマット

ANALYZE/IMAGE ファイル指定[,...]

パラメータ

ファイル指定[,...]

分析したいイメージ・ファイル名を1つまたは複数指定します。少なくとも1つはファイル名を指定しなければなりません。複数のファイル名を指定する場合は、ファイル名をコンマ(,)またはプラス記号(+)で区切ります。省略時のファイル・タイプは.EXE です。

ファイル指定でアスタリスク(*)およびパーセント記号(%)ワイルドカード文字を使用することができます。

説明

ANALYZE/IMAGE コマンドは、OpenVMS VAX および Alpha システムの実行可能イメージ・ファイルや共用可能イメージ・ファイル、および OpenVMS I64 システムの ELF (Executable and Linkable Format) イメージ・ファイルや共用イメージ・ファイルの構成要素の説明を提供します。また、イメージ・ファイルの主な部分の構造が正しいかどうか確認します。ただし、ANALYZE/IMAGE コマンドでエラーが検出されなくても、プログラムの実行時にエラーが発生する可能性があります。

OpenVMS I64 システムでは、ANALYZE/IMAGE コマンドは、ヘッダ情報を調べることで、自動的に I64、Alpha、および VAX のイメージを区別します。

複数のエラーが検出された場合は、最も重大なレベルの最初のエラーが返されます。たとえば警告 (A) と 2 つのエラー (B および C) が検出された場合は、イメージ終了状態として最初のエラー (B) が返されます。イメージ終了状態は、イメージ終了時に DCL シンボル \$STATUS に置かれます。

注意

I64 のイメージとオブジェクトでは、Analyze ユーティリティは、分析対象のファイルがイメージ・ファイルなのかオブジェクト・ファイルなのかを判断します。ELF イメージ・ファイルに対して ANALYZE/OBJECT コマンドを使用することもできますが、ELF イメージ・ファイルには ANALYZE/IMAGE を使用し、ELF オブジェクト・ファイルには ANALYZE/OBJECT を使用するよう にしてください。

ANALYZE/IMAGE の出力を解析するには、ELF イメージの出力形式が変わる可能性がある点に注意してください。

ANALYZE を修飾子なしで使用すると、省略時の設定は/OBJECT となります。したがって、この省略時の設定を使用して出力ファイル中のイメージを分析すると、ユーティリティはそれを正しく「オブジェクト・ファイルの分析」として判断します。

OpenVMS VAX および Alpha バージョンの ANALYZE/IMAGE では、自プラットフォーム以外のすべてのイメージを分析できるわけではありません。たとえば、VAX 上では I64 のイメージを分析できず、古いバージョンの VAX 上では Alpha のイメージを分析できません。

I64 プラットフォーム上で I64 イメージを分析する場合でも、ANALYZE/IMAGE は VAX または Alpha だけで有効な修飾子を受け付けますが、そのような修飾子は無視されます。

プラットフォームに依存して、ANALYZE/IMAGE コマンドは、メタ情報 (たとえば、ELF, EIHD, IHD など) を調べることで、I64 イメージと VAX および Alpha イメージを区別します。

ANALYZE/IMAGE コマンドを実行すると、イメージファイルに関する次に示す情報が分かります。

- イメージのアーキテクチャと種別 — OpenVMS のプラットフォームと、イメージが実行可能であるか、共用可能であるか
- イメージ名 — イメージまたは共用可能イメージの名前
- イメージの識別情報 — リンク操作で指定した識別情報
- 作成リンクの識別情報 — イメージを生成したリンク
- リンク日付と時刻 — リンク操作の日付と時刻
- イメージ変換アドレス — イメージ実行時に制御を渡すアドレス
- イメージ・バージョン — イメージの更新レベル (メジャー ID およびマイナー ID)

- イメージ・シンボル・ベクタの位置およびサイズ (Alpha および I64 のみ)
- 必要な共用可能イメージのリスト — 共用可能イメージの依存関係
- デバッグ・シンボル・テーブル (DST) の位置 — イメージ・ファイル中の DST の位置。/DEBUG または /TRACEBACK コマンド修飾子でリンクされている実行可能イメージ中でのみ、DST 情報が得られます (VAX および Alpha のみ)。
- デバッグ情報とトレースバック情報の位置と解釈方法 — データの情報と形式が格納されているセクション (DWARF) (I64 のみ)
- グローバル・シンボル・テーブル (GST) の位置 — イメージ・ファイル中の GST の位置。共用可能イメージ・ファイル中でのみ、GST 情報が得られます (VAX および Alpha のみ)。
- グローバル・シンボル・テーブル (.symtab) の位置 — イメージ・ファイル中の GST の位置。共用可能イメージ・ファイル中でのみ、GST 情報が得られます (I64 のみ)。
- パッチ情報 — イメージがパッチされている (再コンパイル、または再アセンブル および再リンクなしに変更されている) かどうか。パッチされている場合は、実際のパッチ・コードが表示されます (VAX および Alpha のみ)。
- イメージ・セクション記述子 (ISD) — 属性に基づいて OpenVMS Cluster システムにグループ化されたイメージ・バイナリ部分。ISD は、イメージのアドレス領域を初期化する時に、イメージ・アクティベータが必要とする情報を含んでいます。たとえば、ISD は ISD が共用可能であるかどうか、ISD が読み込み可能または書き込み可能であるかどうか、ISD が基底または位置独立であるかどうか、および割り当てられるメモリ量などの情報が含まれています (VAX のみ)。
- 内部テーブルの要約 — イメージに含まれるプログラム・セグメントとセクションの一覧 (I64 のみ)
- フィックスアップ・ベクタ — 共用可能イメージ参照が位置独立であることを保証するために、イメージ・アクティベータが必要とする情報 (VAX および Alpha のみ)。
- フィックスアップ情報 — 共用可能イメージの参照の位置独立性を保障するために、イメージ・アクティベータが必要とする情報 (I64 のみ)
- システム・バージョン・カテゴリ — エグゼクティブ (I64 および Alpha のシステム共用可能イメージ、または VAX のシステム・シンボル・テーブル) に対してリンクされたイメージについて、イメージが最初にリンクされた時のシステム・バージョン・カテゴリの値、および現在実行中のシステムの値を表示します。イメージは最後にリンクされたので、これらの値を使用するとシステムの変更が分かります。

ANALYZE/IMAGE コマンドには、コマンド修飾子と位置修飾子があります。VAX および Alpha のイメージでの省略時の設定では、位置修飾子 (たとえば /GST や /HEADER) を指定しないと、イメージ全体が分析されます。位置修飾子を指定する

と、(いつも有効な) /HEADER 修飾子を除いた他のすべての位置修飾子、およびユーザが明示的に要求したすべての修飾子を、分析から除外します。

ELF イメージを分析する際の省略時の動作は、Alpha または VAX のイメージを分析する際の動作とは異なります。ELF イメージの場合、主要な ELF テーブルの要約が表示されます。この情報を使用して、分析するセグメントまたはセクション(あるいはその両方)を選択することができます。エラーを見つけるには、すべてのセクションとセグメントを選択して、イメージ全体を分析します。

修飾子

/FIXUP_SECTION (VAX および Alpha のみ)

位置修飾子

イメージのフィックスアップ・セクション中のすべての情報を分析することを指定します。

ANALYZE/IMAGE コマンドの後に/FIXUP_SECTION 修飾子を指定すると、パラメータ・リスト中の各イメージ・ファイルのフィックスアップ・セクションが分析されます。

ファイル指定の後に/FIX_SECTION 修飾子を指定すると、そのイメージ・ファイルのフィックス・セクションのみが分析されます。

/FLAGVALUES=(キーワード[,...]) (I64 のみ)

ELF モジュールのいくつかのフィールドは、ビット・フラグになっています。可能な場合、これらのビット・フラグ値が調べられ、個別に表示されます。省略時には、1 (オン) に設定されているフラグ値だけが表示されます。

キーワードは以下のとおりです。

キーワード	説明
ON	キーワード ON を指定すると、値が 1 のフラグがすべて表示されます。
OFF	キーワード OFF を指定すると、値が 0 のフラグがすべて表示されます。
ALL	キーワード ALL を指定すると、すべてのフラグが表示されます。これに対しキーワード ON と OFF は、各フラグ・ビットの値を示します。

/GST (VAX および Alpha のみ)

位置修飾子

すべてのグローバル・テブル・レコードを分析することを指定します。この修飾子は、共用可能イメージに対してのみ有効です。

ANALYZE/IMAGE コマンドの後に/GST 修飾子を指定すると、パラメータ・リスト中の各イメージ・ファイルのグローバル・シンボル・テブルが分析されます。

ファイル指定の後に/GST 修飾子を指定すると、そのファイルのグローバル・シンボル・テーブルのみが分析されます。

/HEADER (VAX および Alpha のみ)

位置修飾子

すべてのヘッダ・アイテムおよびセクション記述子を分析することを指定します。

/INTERACTIVE

/NOINTERACTIVE (省略時の設定)

分析を会話型で行うかどうかを指定します。会話型モードでは、各アイテムを分析するたびに画面に結果が表示され、継続するかどうか質問が表示されます。

/MODULE [= (モジュール名[,...])] (I64 のみ)

指定したモジュールに対するデバッグ情報やトレースバック情報だけを出力します。/SECTIONS 修飾子にキーワード ALL, DEBUG, または TRACE を指定して、デバッグ情報やトレースバック情報を要求する必要があります。デバッグ情報やトレースバック情報を選択して出力する場合は、さらにモジュール名も指定することができます。

モジュール名を指定しないと、指定可能なモジュールに関するデバッグ・メタ情報またはトレースバック・メタ情報だけが出力されます。その場合、それ以外のデバッグ選択またはトレースバック選択は無効になります。

注意

この修飾子は、ANALYZE/IMAGE だけで有効です。ANALYZE/OBJECT では I64 イメージの情報も出力できますが、/MODULE 修飾子は指定できません。

/OUTPUT=ファイル指定

イメージを分析した結果を格納する出力ファイルを指定します。アスタリスク (*) やパーセント記号 (%) ワイルドカード文字を、ファイル指定に使用することはできません。ファイル・タイプを指定しファイル名を省略した場合は、省略時の設定によりファイル名は ANALYZE になります。省略時のファイル・タイプは ANL です。この修飾子を省略した場合は、分析結果は現在の SYS\$OUTPUT 装置に出力されます。

/PAGE_BREAK=キーワード (I64 のみ)

レポート・ファイルにページ・ブレイク (改ページ) を入れるかどうかと、入れる位置を指定します。この修飾子は、/OUTPUT を使用してレポート・ファイルを出力する場合にだけ有用です。/INTERACTIVE を指定して会話型の分析を行う場合には無視されます。

キーワードは以下のとおりです。

キーワード	説明
NONE	ページ・ブレイクなしのレポートを作成します。
PRINTABLE_REPORT	ページ・ブレイクを含む印刷可能なレポートをリスティング・ファイルとして作成します。ページごとの行数は、プリンタ・ページの省略時の行数になります。この動作は、修飾子を指定しない場合の ANALYZE_IMAGE の省略時の動作です。
SEPARATE_INFORMATION	セクション情報ごとにページ・ブレイクを挿入します。

/PATCH_TEXT (VAX のみ)

位置修飾子

すべてのパッチ・テキスト・レコードを分析することを指定します。ANALYZE /IMAGE コマンドの後に/PATCH_TEXT 修飾子を指定すると、パラメータ・リスト中の各イメージ・ファイルのパッチ・テキスト・レコードが分析されます。

ファイル指定の後に/PATCH_TEXT 修飾子を指定すると、そのファイルのパッチ・テキスト・レコードだけが分析されます。

/SECTIONS [= (キーワード [...])] (I64 のみ)

表示対象の個々のプログラム・セクションまたはセクション種別を選択します。

注意

この修飾子とキーワードは、表示対象セクションのリストに含めるものを指定することしかできません。この修飾子を否定の目的で使用したり、除外リストを指定するのに使用することはできません。値を指定しない場合、省略時のキーワードは HEADERS です。

キーワードは以下のとおりです。

キーワード	説明
ALL	モジュール内の全セクションの詳細な分析結果を表示する。このキーワードを指定すると、大量の出力が生成されることがある点に注意すること。
CODE	種別が SHT_PROGBITS で、実行可能フラグ (セクション・ヘッダ内の SHDR\$M_SHF_EXECINSTR) が設定されているすべてのセクションの全内容を表示する。セクション・データは機械語命令として表示される。

キーワード	説明
DEBUG [=(接尾辞[,...])]	<p>デバッグ情報を含むセクションを分析して表示する。</p> <p>さらに、デバッグ・セクション名の接尾辞のリストを使用して、DEBUG 情報を選択的に出力することができる。デバッグ・セクション名は、要約テーブル中に ".debug_suffix" として表示される。接尾辞は以下のものが指定できる。</p> <ul style="list-style-type: none"> • ABBREV—DEBUG の短縮形の出力 • ARANGES—DEBUG のアドレス・ルックアップ・テーブルの出力 • FRAME— アンwind用の DEBUG フレーム・ディスクリプタの出力 • INFO—DEBUG シンボルの出力 • LINE—DEBUG ソース・ライン情報の出力 • PUBNAMES—DEBUG 名前ルックアップ・テーブルの出力 • PUBTYPES—DEBUG タイプ・ルックアップ・テーブルの出力
EXTENSIONS	種別が SHT_IA64_EXT のセクションを分析して表示する。データは 16 進形式で表示される。
GROUP	種別が SHT_GROUP のセクションを分析して表示する。この種別のセクションは、そのグループに属する各セクションのセクション番号のリストからなる。
HEADERS	省略時のキーワード。ELF ヘッダとセクション・ヘッダの詳細が表示される。
LINKAGES	種別が SHT_VMS_LINKAGES のセクションを分析して表示する。データはリンケージ・ディスクリプタのリストとして表示される。
NOBITS	種別が SHT_NOBITS のセクションを分析して表示する。この種別のセクションに関連付けられたモジュール・データはない。
NOTE	種別が SHT_NOTE のセクションを分析して表示する。このセクションのデータは、フォーマットされた OpenVMS ノート・エントリのリストとして表示される。
NULL	種別が PT_NULL のセクションをすべて表示する。この種別のセグメントについては、データは表示されない。
NUMBERS= (番号[,...])	<p>以下のように個別のセクションを表示する。</p> <ul style="list-style-type: none"> • 選択されたセクションは、ヘッダと内容が詳しく表示される。セクション番号がモジュール内に存在しない場合は、情報メッセージが表示される。 • 1 つ以上の番号を指定することができる。 • セクション番号は、10 進、8 進 (%O 接頭辞を使用)、または 16 進 (%X 接頭辞を使用) で指定可能。

キーワード	説明
STRTAB	種別が SHT_STRTAB のセクションを分析して表示する。このセクション・データは、文字列テーブルとして表示される。
SYMTAB	種別が SHT_SYMTAB のセクションを表示する。このセクションのデータは、シンボル・テーブルとして表示される。
SYMBOL_VECTOR	この種別のセクションは、共用可能イメージ・ファイルにだけ現れる。存在する場合、動的なセグメント DT_VMS_SYMVEC タグと同じデータを指す。
TRACE [=(接尾辞[,...])]	<p>トレースバック情報を含むセクションを分析して表示する。</p> <p>さらに、トレース・セクション名の接尾辞のリストを使用して、TRACE 情報を選択的に出力できる。トレース・セクション名は、要約テーブル中に ".trace_suffix" として表示される。接尾辞は以下のものが指定できる。さらに、デバッグ・セクションとトレースバック・セクションで共通のセクション ".debug_line" が 1 つあるため、以下のように接尾辞 "line" も指定できる。</p> <ul style="list-style-type: none"> • ABBREV—TRACE の短縮形の出力 • ARANGES—TRACE のアドレス・ルックアップ・テーブルの出力 • INFO—TRACE シンボルの出力 • LINE—TRACE ソース・ライン情報の出力
UNWIND	種別が SHT_IA64_UNWIND のセクションを分析して表示する。この種別のセクションには、それぞれ種別が SHT_PROGBITS のアンワインド情報セクションが関連付けられており、このセクションも表示される。

/SEGMENTS [=(キーワード[,...])] (l64 のみ)

個別のプログラム・セグメントまたは指定した種別のプログラム・セグメントを選択して表示します。

注意

この修飾子とキーワードは、表示対象セグメントのリストに含めるものを指定することしかできません。この修飾子を否定の目的で使用したり、除外リストを指定するのに使用することはできません。値を指定しない場合の省略時のキーワードは HEADERS です。

キーワードは以下のとおりです。

キーワード	説明
ALL	すべてのプログラム・セグメントの情報を分析して表示する。出力が大量になることがある点に注意すること。

キーワード	説明
CODE	すべての実行可能セグメント(セグメント・ヘッダでPHDR\$M_PF_Xビットが設定されているセグメント)を分析して表示する。セグメント・データは、機械語命令として表示される。
DYNAMIC	種別が PT_DYNAMIC のセグメントを分析して表示する。
EXTENSIONS	種別が IA_64_ARCHEXT のセグメントを分析して表示する。
HEADERS	省略時のキーワード。ELF ヘッダおよびセグメント・ヘッダの詳細を分析して表示する。
LOAD	種別が PT_LOAD のセグメントを分析して表示する。セグメント・ヘッダが、このセグメントが実行可能セグメントであることを示す(セグメント・ヘッダで PHDR\$M_PF_X ビットが設定されている)場合は、内容は機械語命令として表示される。そうでない場合は、内容が 16 進データとしてフォーマットされる。
NULL	種別が PT_NULL のセグメントを分析して表示する。この種別のセグメントについては、データは表示されない。
NUMBERS=(番号[,...])	<p>以下のように、個別のセグメントを分析して表示する。</p> <ul style="list-style-type: none"> • 選択されたセグメントは、ヘッダと内容が詳しく表示される。セクション番号がモジュール内に存在しない場合は、情報メッセージが表示される。 • 1 つ以上の番号を指定することができる。 • セグメント番号は、10 進、8 進(%O 接頭辞を使用)、または 16 進(%X 接頭辞を使用)で指定する。

/SELECT=(キーワード[,...])

特定のイメージ・ファイルまたはオブジェクト・ファイルの情報を収集し、選択されたキーワード項目を指定された順序で表示します。

Analyze は、/SELECT 修飾子で選択可能なすべての情報に対して DCL シンボルを作成します。シンボル名は、接頭辞 ANALYZE\$と、格納されている情報を示す名前からなります。シンボルの値は選択した情報であり、通常は SYS\$OUTPUT に出力されます。実際には、出力される情報はすべて各シンボルにそのまま入ります。選択しなかった情報については、対応するシンボルに空文字列が入ります。

キーワードは次のとおりです。

キーワード	説明
ARCHITECTURE	アーキテクチャ情報を DCL シンボル ANALYZE\$ARCHITECTURE に書き込む。ファイルが OpenVMS I64 のイメージ・ファイルである場合は、"OpenVMS IA64"を戻す。ファイルが OpenVMS Alpha のイメージ・ファイルである場合には "OpenVMS Alpha"を戻す。ファイルが OpenVMS VAX のイメージ・ファイルの場合には、"OpenVMS VAX"を戻す。

キーワード	説明
BUILD_IDENTIFICATION	構築識別情報を DCL シンボル ANALYZE\$BUILD_IDENTIFICATION に書き込む。OpenVMS I64 および Alpha のイメージ・ファイルでは、イメージ・ファイルに格納されているイメージ構築識別を二重引用符で囲んで戻す。OpenVMS VAX のイメージ・ファイルでは、隣接する二重引用符で表現されるヌル文字列が戻される。
FILE_TYPE	ファイル・タイプ情報を DCL シンボル ANALYZE\$FILE_TYPE に書き込む。ファイルが OpenVMS I64, Alpha, または VAX のイメージ・ファイルの場合, "Image"を戻す。
IDENTIFICATION [=キーワード]	<p>キーワードは次のとおり。</p> <ul style="list-style-type: none"> • IMAGE (省略時の設定) — イメージ識別情報を DCL シンボル ANALYZE\$IDENTIFICATION に書き込む。イメージ・ファイルに格納されているイメージ識別を二重引用符で囲んで戻す。それ以外の場合は "Unknown"を戻す。 • LINKER — リンカ識別情報を DCL シンボル ANALYZE\$LINKER_IDENTIFICATION に書き込む。イメージをリンクするために使用するリンカの識別を戻す。
IMAGE_TYPE	イメージ種別情報を DCL シンボル ANALYZE\$IMAGE_TYPE に書き込む。ファイルが共用可能イメージ・ファイルである場合には "Shareable"を戻す。ファイルが OpenVMS I64, Alpha, または OpenVMS VAX の実行可能 (共用可能でない) イメージ・ファイルである場合には "Executable"を戻す。
LINK_TIME	リンク時刻情報を DCL シンボル ANALYZE\$LINK_TIME に書き込む。イメージ・ファイルに格納されているイメージ・リンク時刻を二重引用符で囲んで戻す。
NAME	イメージ名を DCL シンボル ANALYZE\$NAME に書き込む。イメージ・ファイルの場合は、イメージ・ヘッダに格納されているイメージ名を二重引用符で囲んで戻す。
VERSION_NUMBERS (Alpha/I64 のみ)	イメージがシステム・ベース・イメージおよびシステム・コンポーネントに依存している場合は、ANALYZE はイメージから得たバージョン番号を DCL シンボルに書き込む。シンボル名にはコンポーネント名が付けられる。シンボルの値にはマイナー・バージョン番号とメジャー・バージョン番号が含まれる。イメージが、ANALYZE が動作しているプラットフォームと同じプラットフォーム用なら、動作中のシステムから得たバージョン番号も書き込まれ比較される。

注意

Analyze ユーティリティは、複数のファイル进行处理することが可能です。しかし、DCL シンボルは一組しかないので、シンボルには最後に分析したファイルの情報だけが入ります。エラーが起きると、シンボルの値は未定義となります。まず Analyze のエラーを確認し、その後シンボルを使用してください。

 例

1. \$ ANALYZE/IMAGE LINEDT

この例の ANALYZE/IMAGE コマンドは説明を作成し、イメージ LINEDT.EXE のエラー分析を行います。出力は、現在の SYS\$OUTPUT 装置に送られます。

2. \$ ANALYZE/IMAGE/OUTPUT=LIALPHEX/FIXUP_SECTION/PATCH_TEXT
LINEDT, ALPRIN (VAX および Alpha のみ)

この例の ANALYZE/IMAGE コマンドは説明を作成し、また、ファイル LIALPHEX.ANL 内の LINEDT.EXE と ALPRIN.EXE のフィックスアップ・セクションおよびパッチ・レコード・テキストのエラー分析を作成します。出力は、LIALPHEX.ANL ファイルに送付されます。

3. \$ ANALYZE/IMAGE/SELECT=(ARCH,FILE,NAME,IDENT,BUILD,LINK) *.EXE
DISK:[DIRECTORY]ALPHA.EXE;1
OpenVMS ALPHA
Image
"Test image ALPHA"
"A11-27"
"X5SC-SSB-0000"
14-JUN-2004 07:16:19.24
DISK:[DIRECTORY]VAX.EXE;1
OpenVMS VAX
Image
"Test image VAX"
"V11-27"
"
15-JUN-2004 13:18:40:70

この例は、Alpha システムでの実行ファイル ALPHA.EXE および VAX.EXE について要求された情報を表示しています。

4. \$ ANALYZE/IMAGE/SELECT=(ARCHITECTURE,IDENT,NAME) HELLO 1

ANALYZE/IMAGE

```
USER:[JOE]HELLO.EXE;1
OpenVMS IA64
"V1.0"
"HELLO"
$
$ SHOW SYMBOL ANALYZE$*
ANALYZE$ARCHITECTURE = "OpenVMS IA64"
ANALYZE$BUILD_IDENTIFICATION = ""
ANALYZE$FILE_TYPE = ""
ANALYZE$IDENTIFICATION = ""V1.0""
ANALYZE$IMAGE_TYPE = ""
ANALYZE$LINKER_IDENTIFICATION = ""
ANALYZE$LINK_TIME = ""
ANALYZE$NAME = ""HELLO""
$
$ ANALYZE/IMAGE/SELECT=(IDENT=(IMAGE,LINKER),IMAGE,LINK) HELLO 2
USER:[JOE]HELLO.EXE;1
"V1.0"
"Linker I01-54"
Executable
7-JUN-2004 11:47:08.10
$
$ SHOW SYMBOL ANALYZE$*
ANALYZE$ARCHITECTURE = ""
ANALYZE$BUILD_IDENTIFICATION = ""
ANALYZE$FILE_TYPE = ""
ANALYZE$IDENTIFICATION = ""V1.0""
ANALYZE$IMAGE_TYPE = "Executable"
ANALYZE$LINKER_IDENTIFICATION = "Linker I01-54"
ANALYZE$LINK_TIME = "7-JUN-2004 11:47:08.10"
ANALYZE$NAME = ""
$
$ ANALYZE/IMAGE/SELECT=FILE HELLO.* 3
USER:[JOE]HELLO.C;1
%ANALYZE-E-ILLFIL, Illegal file format encountered
USER:[JOE]HELLO.EXE;1
Image
USER:[JOE]HELLO.MAP;1
%ANALYZE-E-ILLFIL, Illegal file format encountered
USER:[JOE]HELLO.OBJ;1
Object
$
$ SHOW SYMBOL ANALYZE$*
ANALYZE$ARCHITECTURE = ""
ANALYZE$BUILD_IDENTIFICATION = ""
ANALYZE$FILE_TYPE = "Object"
ANALYZE$IDENTIFICATION = ""
ANALYZE$IMAGE_TYPE = ""
ANALYZE$LINKER_IDENTIFICATION = ""
ANALYZE$LINK_TIME = ""
ANALYZE$NAME =
$
```

この I64 の例では、実行可能ファイル HELLO.EXE について要求された情報が表示されています。以下の説明は、例の中の各 ANALYZE/IMAGE コマンド行の右端にある番号に対応しています。

- 1 選択した情報だけが DCL シンボルに設定されます。シンボル内の情報は、SYS\$OUTPUT に出力される情報と同じであり、引用符付きの文字列が出力される場合には、引用符付きの文字列がシンボルに設定されます。
- 2 新しいリンカ識別を選ぶ場合には、IDENT とキーワード・リストを使用する必要があります。
- 3 ワイルドカードを使用すると、分析対象のファイルにエラー（たとえば、ファイル形式不正エラーなど）があっても、Analyze は停止しません。DCL シンボルには、最後に分析したファイルの情報だけが入ります。

ANALYZE/MEDIA

Bad Block Locator ユーティリティを起動します。このユーティリティはブロックでアドレス指定できる装置を分析し、データを格納できないブロック位置を記録します。

Bad Block Locator ユーティリティについての詳細は、『OpenVMS Bad Block Locator Utility Manual』（ドキュメンテーション CD-ROM に用意されています）、あるいはオンライン・ヘルプを参照してください。

フォーマット

ANALYZE/MEDIA 装置

ANALYZE/OBJECT

OpenVMS VAX および Alpha システムのオブジェクト・ファイルと、OpenVMS I64 システムの ELF (Executable and Linkable Format) オブジェクト・ファイルの内容を分析し、エラーを検出します。/OBJECT 修飾子は必須です。

オブジェクト・ファイルについての一般的な情報は、『OpenVMS Linker Utility Manual』のリンクの説明を参照してください。イメージ・ファイルの内容を分析する場合は、ANALYZE/IMAGE コマンドを使用してください。

フォーマット

ANALYZE/OBJECT ファイル指定[,...]

パラメータ

ファイル指定[,...]

分析したいオブジェクト・ファイル、またはオブジェクト・モジュール・ライブラリ(省略時のファイル・タイプは.OBJ)を指定します。複数のファイル名を指定する場合は、ファイル名をコンマ(,)またはプラス記号(+)で区切ります。ファイル指定でアスタリスク(*)およびパーセント記号(%)ワイルドカード文字を使用することができます。

説明

ANALYZE/OBJECT コマンドは、1 つまたは複数のファイルに含まれている、1 つまたは複数のオブジェクト・モジュールの内容を記述します。また、部分的なエラー分析を行うこともできます。オブジェクト・モジュールのすべてのレコードが、その内容、形式、指定順序において、I64、Alpha または VAX のオブジェクト言語に準拠しているかどうか分析します。

OpenVMS I64 システムでは、オブジェクト・モジュール・ヘッダの形式を調べることにより ANALYZE/OBJECT コマンドは、I64、Alpha、VAX のオブジェクトを自動的に区別します。

ANALYZE/OBJECT コマンドは、コンパイラ、デバッガや、オペレーティング・システムのオブジェクト・モジュールを扱うその他のソフトウェアのプログラマ向けです。ANALYZE/OBJECT コマンドは、オブジェクト・モジュールにより生成された ELF オブジェクト形式 (I64) またはオブジェクト言語レコード (VAX および Alpha) を、Linker ユーティリティが受け入れることができるかチェックし、ファイル中の

エラーを検出します。また、オブジェクト・ファイルまたはオブジェクト・モジュール・ライブラリ中のレコードの説明を提供します。リンカ、および Alpha と VAX のオブジェクト言語についての詳細は、『OpenVMS Linker Utility Manual』を参照してください。I64 オブジェクト形式の情報は、今後のリリースで提供されます。

注意

I64 のイメージとオブジェクトでは、Analyze ユーティリティは、分析対象のファイルがイメージ・ファイルなのかオブジェクト・ファイルなのかを判断します。ELF オブジェクト・ファイルに対して ANALYZE/IMAGGE コマンドを使用することもできますが、ELF イメージ・ファイルには ANALYZE/IMAGE を使用し、ELF オブジェクト・ファイルには ANALYZE/OBJECT を使用するようにしてください。

OpenVMS VAX および OpenVMS Alpha バージョンの ANALYZE/OBJECT では、自プラットフォーム以外のすべてのオブジェクト(たとえば、VAX や Alpha 上での I64 オブジェクト)を分析できるわけではありません。

ELF オブジェクトに対する ANALYZE/OBJECT の出力形式は変更される可能性があります。また、ELF オブジェクトを分析するときの省略時の動作は、Alpha や VAX のオブジェクトを分析するときの動作とは異なります。ELF オブジェクトでは、主要な ELF テーブルの要約が表示されます。この情報を使用して、詳しく分析するセクションを選択することができます。エラーを見つけるには、すべてのセクションを選択して、イメージ全体を分析する必要があります。

I64 プラットフォーム上で I64 オブジェクトを分析する場合でも、ANALYZE/OBJECT は VAX または Alpha だけで有効な修飾子を受け付けますが、そのような修飾子は無視されます。

ANALYZE/OBJECT コマンドは、最初のレコードから最後のレコードまで、順番にオブジェクト・モジュールを分析します。各レコードのフィールドは、最初のフィールドから最後のフィールドまで、順番に分析されます。オブジェクト・モジュールの分析が終わったら、各種レコードの内容と形式と、OpenVMS I64、Alpha または OpenVMS VAX オブジェクト言語で記述される各種レコードの必須な内容と形式を比較します。分析出力に診断メッセージが含まれている場合は、こうして比較することが特に重要です。

ANALYZE/OBJECT は、オブジェクト・モジュールの以下の情報を表示します。

- モジュールのアーキテクチャと種別
- モジュール名
- モジュールのバージョン
- モジュール作成日と時刻
- 言語プロセッサの作成者

オブジェクト・モジュールのリンクは、オブジェクト・モジュールの分析とは異なり、オブジェクトの内容は解釈されません。その代わり、メタ情報の整合性だけがチェックされます。その結果、分析した時にエラーがなくても、リンクした時にエラーが発生する可能性があります。特に VAX および Alpha オブジェクトに対しては、分析時には次の項目をチェックしません。

- TIR コマンドのデータ引数の形式が正しいかどうか。
- “Store Data” TIR コマンドが、有効なアドレス・リミットに格納しているか。

そのため、分析時にはエラーがなかったオブジェクト・モジュールも、リンク時にチェックする必要があります。

エラーが検出された場合は、最も重大なレベルの最初のエラーが返されます。たとえば警告 (A) と 2 つのエラー (B および C) が検出された場合は、イメージ終了状態として最初のエラー (B) が返されます。イメージ終了状態は、イメージ終了時に DCL シンボル \$STATUS に置かれます。

ANALYZE/OBJECT コマンドは、位置修飾子を使用します。位置修飾子とは、コマンド行で指定される位置により機能が異なる修飾子です。コマンド行ですべての入力ファイルより前に位置修飾子を指定すると、その修飾子はすべての入力ファイルに影響を与えます。たとえば次のコマンドは、ファイル A、B、および C のグローバル・シンボル・ディレクトリ・レコードを分析するように指定しています。

```
$ ANALYZE/OBJECT/GSD A,B,C
```

反対に、パラメータ・リスト中の 1 つのファイルに対して位置修飾子が指定された場合は、その修飾子はそのファイルにのみ影響を与えます。たとえば次のコマンドは、ファイル B(だけ) のグローバル・シンボル・ディレクトリ・レコードを分析するように指定しています。

```
$ ANALYZE/OBJECT A,B/GSD,C
```

VAX および Alpha のオブジェクトでは、一般に、オブジェクト・モジュールのすべてのレコードが分析されます。ただし、/DBG、/EOM、/GSD、/LNK、/MHD、/TBT、または/TIR 修飾子が指定された場合は、修飾子により識別されるレコード・タイプのみが分析されます。その他のレコード・タイプはすべて無視されます。

省略時の設定では (適切な修飾子を使用して明示的に分析に制限を指定しない限り)、すべてのレコード・タイプが分析されます。

注意

VAX および Alpha のオブジェクトでは、どのような修飾子を指定しても、モジュールの終端 (EOM) レコードとモジュール・ヘッダ (MHD)・レコードは分析されます。

164 オブジェクトでは、どのような修飾子を指定しても、ELF ヘッダ、セクション・ヘッダ・テーブル、ノート・セクションが常に分析されます。

修飾子

/DISASSEMBLE (I64 のみ)

位置修飾子

種別が SHT_PROGBITS で、実行可能フラグ (セクション・ヘッダ内の SHDR\$_M_SHF_EXECINSTR) が設定されているすべてのセクションを表示します。セクションデータは機械語命令として表示され、ラベルや分岐ターゲットなどはシンボル化されます。シンボル化には、シンボル・テーブル中のすべてのローカル・シンボルとグローバル・シンボルが使用されます。出力は、コンパイラが生成する機械語のリストと同様になります。

注意

この修飾子は、オブジェクトに対してだけ指定可能です。I64 イメージに含まれているシンボルは、グローバル・シンボルだけです。さらに、この修飾子で生成される出力は、ANALYZE/OBJECT/SECTIONS=CODE の出力とは異なります。後者でも同じセクションの機械語が出力されますが、シンボル化は行われません。

/DBG (VAX および Alpha のみ)

位置修飾子

すべてのデバッグ情報レコードを分析することを指定します。パラメータ・リスト中のすべてのファイルのデバッグ情報を分析したい場合は、/OBJECT 修飾子の直後に /DBG 修飾子を指定します。選択的にデバッグ情報を分析したい場合は、デバッグ情報を分析したいファイルを指定した直後に /DBG 修飾子を指定します。

/EOM (VAX および Alpha のみ)

位置修飾子

MHD レコード、EOM レコード、およびコマンドで明示的に指定したレコードだけを分析することを指定します。パラメータ・リスト中のすべてのファイルにこれを適用させたい場合は、/OBJECT 修飾子の直後に /EOM 修飾子を指定します。

/EOM 修飾子を選択的に指定したい場合は、適用させたいファイルを指定した直後に /EOM 修飾子を指定します。

注意

モジュールの終端レコードは、EOM または EOMW レコードになることができます。詳細は『OpenVMS Linker Utility Manual』を参照してください。

/FLAGVALUES=(キーワード[,...]) (l64 のみ)

ELF モジュール内のいくつかのフィールドは、ビット・フラグになっています。可能な場合、これらのビット・フラグ値が調べられ、個別に表示されます。省略時には、1 (オン) が設定されているフラグ値だけが表示されます。

キーワードは以下のとおりです。

キーワード	説明
ON	値が 1 のすべてのフラグが表示されます。
OFF	値が 0 のすべてのフラグが表示されます。
ALL	すべてのフラグ値が表示されます。これに対しキーワード ON と OFF は、各フラグ・ビットの値を示します。

/GSD (VAX および Alpha のみ)

位置修飾子

すべてのグローバル・シンボル・ディレクトリ (GSD)・レコードを分析することを指定します。

パラメータ・リスト中の各ファイルの GSD レコードを分析したい場合は、/OBJECT 修飾子の直後に/GSD 修飾子を指定します。

選択的に GSD レコードを分析したい場合は、GSD レコードを分析したいファイルを指定した直後に/GSD 修飾子を指定します。

/INCLUDE [=(モジュール[,...])]

指定したファイルがオブジェクト・モジュール・ライブラリの場合は、この修飾子を使用して、分析のためライブラリ内の選択したオブジェクト・モジュールをリストします。リストを省略したりアスタリスク(*)を指定したりすると、すべてのモジュールが分析されます。モジュールを 1 つだけ指定する場合は、括弧を省略できます。

/INTERACTIVE

/NOINTERACTIVE (省略時の設定)

分析を会話型で行うかどうかを指定します。会話型モードでは、各アイテムを分析するたびに画面に結果が表示され、継続するかどうか質問が表示されます。

/LNK (VAX および Alpha のみ)

位置修飾子

すべてのリンク・オプション指定 (LNK) レコードを分析することを指定します。

パラメータ・リスト中の各ファイルの LNK レコードを分析したい場合は、/OBJECT 修飾子の直後に/LNK 修飾子を指定します。

選択的に LNK レコードを分析したい場合は、LNK レコードを分析したいファイルを指定した直後に/LNK 修飾子を指定します。

/MHD (VAX および Alpha のみ)
位置修飾子

MHD レコード、EOM レコード、およびコマンドで明示的に指定したレコードだけを分析することを指定します。パラメータ・リスト中のすべてのファイルにこれを適用させたい場合は、/OBJECT 修飾子の直後に/MHD 修飾子を指定します。

/MHD 修飾子を選択的に指定したい場合は、適用させたいファイルを指定した直後に/MHD 修飾子を指定します。

/OUTPUT [=ファイル指定]

オブジェクトの分析を格納する出力ファイルを指定します。省略時の設定では SYS\$OUTPUT 装置に出力されます。ファイル・タイプを指定しファイル名を省略した場合は、省略時の設定によりファイル名は ANALYZE になります。省略時のファイル・タイプは .ANL です。

アスタリスク (*) やパーセント記号 (%) ワイルドカード文字を、ファイル指定に使用することはできません。

/PAGE_BREAK=キーワード (I64 のみ)

レポート・ファイルにページ・ブレイク (改ページ) を入れるかどうかと、入れる位置を指定します。この修飾子は、/OUTPUT を使用してレポート・ファイルを出力する場合にだけ有効です。/INTERACTIVE を指定して会話型の分析を行う場合には無視されます。

キーワードは以下のとおりです。

キーワード	説明
NONE	ページ・ブレイクなしのレポートを作成します。
PRINTABLE_ REPORT	ページ・ブレイクを含む印刷可能なレポートをリスティング・ファイルとして作成します。ページごとの行数は、プリンタ・ページの省略時の行数になります。この動作は、修飾子を指定しない場合の ANALYZE_OBJECT の省略時の動作です。
SEPARATE_ INFORMATION	セクション情報ごとにページ・ブレイクを挿入します。

/SECTIONS [= (キーワード[,...])] (I64 のみ)

表示対象の個々のプログラム・セクションまたはセクション種別を選択します。

注意

この修飾子とキーワードは、表示するセクションのリストに含めるものを指定することしかできません。この修飾子を否定の目的で使用したり、除外リストを指定するのに使用することはできません。値を指定しない場合の省略時のキーワードは HEADERS です。

キーワードは以下のとおりです。

キーワード	説明
ALL	モジュール内の全セクションの詳細な分析結果を表示する。このキーワードを指定すると、大量の出力が生成されることがある点に注意すること。
CODE	種別が SHT_PROGBITS で、実行可能フラグ (セクション・ヘッダ内の SHDR\$M_SHF_EXECINSTR) が設定されているセクションをすべて表示する。セクション・データは、機械語命令として表示される。
DEBUG [(=接尾辞[,...])]	<p>デバッグ情報を含むセクションを分析して表示する。</p> <p>さらに、デバッグ・セクション名接尾辞のリストを使用して、DEBUG 情報を選択的に出力することができる。デバッグ・セクション名は、要約テーブル中に ".debug_suffix" として表示される。接尾辞は以下のものが指定できる。</p> <ul style="list-style-type: none"> • ABBREV—DEBUG の短縮形の出力 • ARANGES—DEBUG のアドレス・ルックアップ・テーブルの出力 • FRAME— アンwind用の DEBUG フレーム・ディスクリプタの出力 • INFO—DEBUG シンボルの出力 • LINE—DEBUG ソース・ライン情報の出力 • PUBNAMES—DEBUG 名前ルックアップ・テーブルの出力 • PUBTYPES—DEBUG タイプ・ルックアップ・テーブルの出力
EXTENSIONS	種別が SHT_IA64_EXT のセクションを分析して表示する。データは 16 進形式で表示される。
GROUP	種別が SHT_GROUP のセクションを分析して表示する。この種別のセクションは、そのグループに属するセクションのセクション番号のリストからなる。
HEADERS	省略時のキーワード。ELF ヘッダおよびセクション・ヘッダの詳細を表示する。
LINKAGES	種別が SHT_VMS_LINKAGES のセクションを分析して表示する。データはリンケージ・ディスクリプタのリストとして表示される。
NOBITS	種別が SHT_NOBITS のセクションを分析して表示する。この種別のセクションに関連付けられたモジュール・データはない。
NOTE	種別が SHT_NOTE のセクションを分析して表示する。このセクション・データは、フォーマットされた OpenVMS ノート・エントリのリストとして表示される。
NULL	種別が PT_NULL のセクションをすべて表示する。この種別のセグメントについては、データは表示されない。

キーワード	説明
NUMBERS= (番号[,...])	<p>以下のように個別のセクションを表示する。</p> <ul style="list-style-type: none"> 選択されたセクションは、ヘッダと内容が詳しく表示される。セクション番号がモジュール内に存在しない場合は、情報メッセージが表示される。 1 つ以上の番号を指定することができる。 セクション番号は、10 進、8 進 (%O 接頭辞を使用)、または 16 進 (%X 接頭辞を使用) で指定可能。
PROGBITS	<p>アンwind・セクション以外の、種別が SHT_PROGBITS のセクションをすべて表示する。</p> <p>種別が SHT_PROGBITS のセクションに対するフォーマット方法は、セクション・ヘッダの EXECINSTR フラグ (SHDR\$M_SHF_EXECINSTR) によって変わる。このビットが立っている場合は、セクション・データは機械語命令として表示される。ビットが立っていない場合は、16 進データとして表示される。</p> <p>アンwind・セクションは、/SECTIONS=UNWIND を指定した場合に表示される。</p>
RELOCATIONS	<p>種別が SHT_RELA のセクションを分析して表示する。このセクションのデータは、再配置エントリのテーブルとして表示される。</p>
STRTAB	<p>種別が SHT_STRTAB のテーブルを分析して表示する。このセクションのデータは、文字列テーブルとして表示される。</p>
SYMTAB	<p>種別が SHT_SYMTAB のセクションを表示する。このセクション・データは、シンボル・テーブルとして表示される。</p>
TRACE [(=接尾辞[,...])]	<p>トレースバック情報を含むセクションを分析して表示する。</p> <p>さらに、トレース・セクション名の接尾辞のリストを使用して、TRACE 情報を選択的に出力できる。トレース・セクション名は、要約テーブル中に ".trace_suffix" として表示される。接尾辞は以下のものが指定できる。さらに、デバッグ・セクションとトレースバック・セクションで共通のセクション ".debug_line" が 1 つあるため、以下のように接尾辞 "line" も指定できる。</p> <ul style="list-style-type: none"> ABBREV—TRACE の短縮形の出力 ARANGES—TRACE のアドレス・ルックアップ・テーブルの出力 INFO—TRACE シンボルの出力 LINE—TRACE ソース・ライン情報の出力
UNWIND	<p>種別が SHT_IA64_UNWIND のセクションを分析して表示する。この種別のセクションには、種別が SHT_PROGBITS のアンwind情報セクションが関連付けられている。関連付けられたセクションも表示される。</p>

/SELECT=(キーワード[,...])

特定のオブジェクト・ファイルの情報を収集し、選択したキーワード項目を指定された順序で表示します。

注意

/SELECT 修飾子はオブジェクト・ファイルにもイメージ・ファイルにも使用でき、同じキーワードを使って選択することができます。しかし、リンクの日付や時刻など、いくつかの情報はオブジェクトには存在しません。このため、いくつかのキーワードについては、Analyze ユーティリティは "Unknown" を戻します。次の表には、キーワード (オブジェクト・ファイルで有用なもの) と戻り値だけを挙げます。

Analyze は、/SELECT 修飾子を使って選択可能なすべての情報に対し、DCL シンボルを作成します。シンボル名は、接頭辞 ANALYZE\$ と、格納されている情報を示す名前からなります。シンボルの値は選択した情報であり、通常は SYS\$OUTPUT に出力されます。実際には、出力される情報はすべて各シンボルにそのまま入ります。選択しなかった情報については、対応するシンボルに空文字列が入ります。

キーワードは次のとおりです。

キーワード	説明
ARCHITECTURE	アーキテクチャ情報を DCL シンボル ANALYZE\$ARCHITECTURE に書き込む。ファイルが OpenVMS I64 のオブジェクト・ファイルである場合は、"OpenVMS IA64" を戻す。ファイルが OpenVMS Alpha のオブジェクト・ファイルである場合は、"OpenVMS Alpha" を戻す。ファイルが OpenVMS VAX のオブジェクト・ファイルである場合は、"OpenVMS VAX" を戻す。
FILE_TYPE	ファイル・タイプ情報を DCL シンボル ANALYZE\$FILE_TYPE に書き込む。ファイルが OpenVMS I64, Alpha, または VAX のオブジェクト・ファイルである場合は、"Object" を戻す。

/TBT (VAX および Alpha のみ)

位置修飾子

すべてのモジュール・トレースバック (TBT) ・レコードを分析することを指定します。

パラメータ・リスト中の各ファイルの TBT レコードを分析したい場合は、/OBJECT 修飾子の直後に /TBT 修飾子を指定します。

選択的に TBT レコードを分析したい場合は、TBT レコードを分析したいファイルを指定した直後に /TBT 修飾子を指定します。

/TIR (VAX および Alpha のみ)

位置修飾子

すべてのテキスト情報および再配置 (TIR) レコードを分析することを指定します。

パラメータ・リスト中の各ファイルの TIR レコードを分析したい場合は、/OBJECT 修飾子の直後に/TIR 修飾子を指定します。

選択的に TIR レコードを分析したい場合は、TIR レコードを分析したいファイルを指定した直後に/TIR 修飾子を指定します。

例

1. \$ ANALYZE/OBJECT/INTERACTIVE LINEDT

この例で ANALYZE/OBJECT コマンドは説明を作成し、オブジェクト・ファイル LINEDT.OBJ を部分的にエラー分析します。/INTERACTIVE 修飾子が指定されているので、出力はターミナルに送られます。各アイテムを分析するたびに結果は画面に表示され、継続するかどうか質問が表示されます。

2. \$ ANALYZE/OBJECT/OUTPUT=LIOBJ/DBG LINEDT (VAX および Alpha のみ)

この例で ANALYZE/OBJECT コマンドは、ファイル LINEDT.OBJ のデバッグ情報レコードだけを分析します。出力は、ファイル LIOBJ.ANL に書き込まれます。

3. \$ ANALYZE/OBJECT/SELECT=(ARCH,FILE) *.OBJ
DISK:[DIRECTORY]ALPHA.OBJ;1
OpenVMS ALPHA
Object
DISK:[DIRECTORY]VAX.OBJ;1
OpenVMS VAX
Object

この例は、オブジェクト・ファイル ALPHA.OBJ および VAX.OBJ について要求された情報を表示しています。

ANALYZE/PROCESS_DUMP

OpenVMS デバッガを起動して、イメージの実行に失敗した時に作成されたプロセス・ダンプ・ファイル进行分析します。ダンプ・ファイルを作成する場合は、RUN または SET PROCESS コマンドに/DUMP 修飾子を指定します。

Alpha システムでは、DUMP/PROCESS コマンドを使用することによって、プロセスを強制的にダンプすることもできます。

ANALYZE/PROCESS_DUMP コマンドは、Alpha または VAX のいずれかのイメージのプロセス・ダンプ・ファイルを表示するために OpenVMS デバッガを起動します。(DEBUG コマンドに関する情報を含め) デバッガについての詳細は、『OpenVMS デバッガ説明書』を参照してください。

ダンプ・ファイルに対して読み込み (R) アクセス権が必要です。

フォーマット

ANALYZE/PROCESS_DUMP ダンプ・ファイル

パラメータ

ダンプ・ファイル
デバッガで分析するダンプ・ファイルを指定します。

説明

ANALYZE/PROCESS_DUMP コマンドは、実行時に失敗したイメージのダンプ・ファイル調べます。OpenVMS Debugger が自動的に起動されます。プロセスのダンプ・ファイルを作成するためには、イメージを起動する時に RUN コマンドに/DUMP 修飾子を指定するか、またはイメージを起動する前に SET PROCESS/DUMP コマンドを指定する必要があります。Alpha システムでは、DUMP/PROCESS コマンドを使用できます。

OpenVMS VAX システムの場合

この項は、バージョン 7.2 またはそれ以前の Alpha システムにも適用されます。

注意

ダンプを作成したシステムで、プロセス・ダンプ进行分析してください。異なるシステムにダンプ・ファイルを移動させると、正しく分析できない可能性があります。

ダンプ・イメージをロードできても、構成が異なるために、ANALYZE/PROCESS_DUMP コマンドの実行に失敗することがあります。たとえば、オペレーティング・システムのバージョンが異なっても分析はされますが、その結果は保証されません。

この他に、P1 空間内の制御領域の構成、ダンプ時に実行中のプロセス、および ANALYZE/PROCESS_DUMP コマンドを実行するプロセスについての制限事項があります。各プロセスのためのユーザ・スタックの基底位置は、割り当てられた空間のサイズにより異なり、プロセスに互換性があるかどうかを決めます。ダンプを分析するために割り当てられた空間のサイズは、ダンプを作成したプロセスに割り当てられた空間のサイズより小さくしなければなりません。オペレーティング・システムのバージョンは同じでも、異なるシステムでダンプを分析する場合は、割り当てられた空間のサイズに影響を与える 1 つまたは複数のシステム・パラメータを変更して、割り当てられた空間のサイズを小さくすることができます。

システム・パラメータ IMGIOCNT は、動的に変更することができます。その他のシステム・パラメータを適用させるためには、再ブートする必要があります。

Alpha システムでは、OpenVMS Debugger でダンプしたイメージを分析できないことがあります。たとえばダンプしたイメージの PC が無効なアドレスに設定された場合、またはダンプしたイメージのスタックが不良プロセス記述子により破損した場合は、Delta Debugger (DELTA) を使用してダンプを分析しなければなりません。デバグガとして DELTA を使用するには、Install ユーティリティを起動してイメージ SYS\$LIBRARY:DELTA をインストールする必要があります。Install ユーティリティについての詳細は、『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』を参照してください。

OpenVMS Alpha システムの場合

この項は、バージョン 7.3 またはそれ以降の OpenVMS Alpha システムに適用されます。

これで、ダンプが生成されたシステム以外のシステム上でダンプ・ファイルを分析できるようになります。ただし、ベース・イメージのリンクの日付と時刻が等しくない場合には、ダンプが生成されたシステムからファイル SYS\$BASE_IMAGE.EXE をコピーし、これを論理名 SDA\$READ_DIR でポイントする必要があります。次に例を示します。

```
$ COPY other_node::SYS$LOADABLE_IMAGES:SYS$BASE_IMAGE.EXE my_disk$:[my_dir]
$ DEFINE/USER SDA$READ_DIR my_disk$:[my_dir],SYS$SYSROOT:[SYS$LDR],SYS$SYSROOT:[SYS$LIB]
$ ANALYZE/PROCESS_DUMP mycrash.dmp
```

ダンプが生成されたシステム以外のシステムで、スレッド化されたプロセス・ダンプを分析する場合には、ダンプが生成されたシステム上の PTHREAD\$RTL と PTHREAD\$DBGSHR (DECthread デバグ・アシスタント) もコピーし、ポイントしなければなりません。次に例を示します。

```
$ COPY other_node::SYS$LOADABLE_IMAGES:SYS$BASE_IMAGE.EXE my_disk$:[my_dir]
$ COPY other_node::SYS$SHARE:PTHREAD$RTL.EXE my_disk$:[my_dir]
$ COPY other_node::SYS$SHARE:PTHREAD$DBGSHR.EXE my_disk$:[my_dir]
$ DEFINE/USER SDA$READ_DIR my_disk$:[my_dir],SYS$SYSROOT:[SYS$LDR],SYS$SYSROOT:[SYS$LIB]
$ DEFINE/USER PTHREAD$RTL my_disk$:[my_dir]PTHREAD$RTL.EXE
$ DEFINE/USER PTHREAD$DBGSHR my_disk$:[my_dir]PTHREAD$DBGSHR.EXE
$ ANALYZE/PROCESS_DUMP mycrash.dmp
```

デバッガでプロセス・ダンプを分析できない場合には、System Dump Analyzer (SDA) ユーティリティを使用してください。詳細は、オンライン・ヘルプの ANALYZE /CRASH コマンドを参照してください。次に例を示します。

```
$ ANALYZE/CRASH mycrash.dmp

OpenVMS (TM) Alpha system dump analyzer
...analyzing a compressed process dump...

Dump taken on 19-OCT-1999 12:03:40.95
SDA> ..
.
.
```

修飾子

/FULL

VAX および Alpha システム上で、デバッガ・コマンド SHOW IMAGE、SHOW THREAD/ALL、および SHOW CALL によって表示される情報を表示します。

/IMAGE=ダンプ・ファイル

VAX システム上で、分析に使用するプロセス・コンテキストを設定するために起動するイメージを指定します。/NOIMAGE 修飾子を使用すると、分析には DELTA デバッガが使用されます。

省略時の設定では、シンボルは、ダンプの時点で実行されていたイメージと同じ名前のイメージから取得されます。

/IMAGE_PATH[=ディレクトリ指定]ダンプ・ファイル
/NOIMAGE_PATH

Alpha システム上で、デバッガがデバッガ・シンボル・テーブル (DST) ファイルを探すために使用する検索パスを指定します。以前のデバッガと同様に、デバッガは保存済みのプロセス・イメージ・リストからイメージ・リストを構築します。イメージを設定すると (メイン・イメージが自動的に設定されます)、デバッガはそのイメージをオープンして DST ファイルを探そうと試みます。

/IMAGE_PATH=ディレクトリ指定修飾子を指定すると、デバッガは指定されたディレクトリの中で DST ファイルを探します。デバッガはまずディレクトリ指定をディレクトリ検索リストの論理名として変換しようと試みます。これに失敗すると、デバッガはディレクトリ指定をディレクトリ指定として解釈し、そのディレクトリの中で

対応する.DSFまたは.EXE ファイルを探します。.DSF ファイルの方が.EXE ファイルよりも優先されます。.DSF または.EXE ファイルの名前はイメージと一致していなければなりません。

/IMAGE_PATH=ディレクトリ指定修飾子を指定しなかった場合、デバッガはまずダンプ・ファイルを含んでいるディレクトリの中で DST ファイルを探します。これに失敗すると、デバッガはディレクトリ SYS\$SHARE を、次にディレクトリ SYS\$MESSAGE を探します。デバッガがイメージの DST ファイルを見つけられなかった場合、デバッガが利用できるシンボリック情報はグローバルおよびユニバーサル・シンボル名に限定されます。

バージョン 7.3 とそれ以降のデバッガは、ダンプ・ファイルのイメージ指定と DST ファイルのリンクの日付と時刻が一致しているかどうかをチェックし、一致していなければ警告を発します。

ダンプ・ファイル・パラメータは、分析するプロセス・ダンプ・ファイルの名前です。プロセス・ダンプ・ファイルのファイル・タイプは.DMP でなければならず、DST ファイル・タイプは.DSF または.EXE でなければならないことに注意してください。

制限事項

論理名を使用してイメージの検索をリダイレクトしたり、/IMAGE_PATH 修飾子を同時に使用することはできません。/IMAGE_PATH 修飾子を使用する場合は、元の場所にはないすべてのイメージが、このパスで検索できるところに置かれている必要があります。個別のイメージ論理名 ("DEFINE SH SYS\$LOGIN:SH.EXE" 中の "SH" など) は処理されません。

また、コンマで区切られたディレクトリ・リストは処理されないで、/IMAGE_PATH 修飾子にディレクトリ検索パスを直接入力することはできません。ただし、ディレクトリ検索パスに変換される論理名を指定することはできます。

/INTERACTIVE

/NOINTERACTIVE (省略時の設定)

VAX システムでは、ターミナルの画面いっぱいに表示されたら、情報の表示を一時停止することを指定します。次の情報を表示したい場合は、RETURN キーを押します。省略時の設定では、情報は一時停止せずに表示されます。

/MISCELLANEOUS

VAX システムでは、ダンプ時のプロセス情報とレジスタを表示します。表示されるプロセス情報についての詳細は、システム・サービス\$GETJPI を参照してください。

/RELOCATION

VAX システムでは、ダンプ中に格納されているデータ構造が P0 空間にマップされときのアドレスを表示します (このようなデータ構造の例としては、スタックがあります)。デバッガが P1 空間のこれらのデータ構造を使用できるように、ダンプ中のデータ構造は P0 空間にマップされなければなりません。

例

```

1. $ ANALYZE/PROCESS/FULL ZIPLIST

R0 = 00018292 R1 = 8013DE20 R2 = 7FFE6A40 R3 = 7FFE6A98
R4 = 8013DE20 R5 = 00000000 R6 = 7FFE7B9A R7 = 0000F000
R8 = 00000000 R9 = 00000000 R10 = 00000000 R11 = 00000000
SP = 7FFAEF44 AP = 7FFAEF48 FP = 7FFAEF84
FREE_P0_VA 00001600 FREE_P1_VA 7FFAC600
Active ASTs 00 Enabled ASTs 0F
Current Privileges FFFFFFF80 1010C100
Event Flags 00000000 E0000000
Buffered I/O count/limit 6/6
Direct I/O count/limit 6/6
File count/limit 27/30
Process count/limit 0/0
Timer queue count/limit 10/10
AST count/limit 6/6
Enqueue count/limit 30/30
Buffered I/O total 7 Direct I/O total 18

Link Date 27-DEC-2001 15:02:00.48 Patch Date 17-NOV-2001 00:01:53.71
ECO Level 0030008C 00540040 00000000 34303230
Kernel stack 00000000 pages at 00000000 moved to 00000000
Exec stack 00000000 pages at 00000000 moved to 00000000
Vector page 00000001 page at 7FFEFE00 moved to 00001600
PIO (RMS) area 00000005 pages at 7FFE1200 moved to 00001800
Image activator context 00000001 page at 7FFE3400 moved to 00002200
User writable context 0000000A pages at 7FFE1C00 moved to 00002400
Creating a subprocess
      VAX DEBUG Version 5.4
DBG>

```

この例は、VAX システムで ANALYZE/PROCESS コマンドに/FILL 修飾子を指定した場合の出力を示しています。指定したファイル ZIPLIST には、回復不可能なエラーが検出されたプロセスのダンプが含まれています。DBG>プロンプトは、デバッガがコマンドを受け入れる用意ができていることを示します。

```

2. $ ANALYZE/PROCESS/FULL WECRASH.DMP

OpenVMS Alpha Debug64 Version X7.3-010
%SYSTEM-F-IMGDMP, dynamic image dump signal at PC=001D0F8CB280099C, PS=001D0028
break on unhandled exception preceding WECRASH\th_run\%LINE 26412 in THREAD 8
%DEBUG-W-UNAOPNSRC, unable to open source file DSKD$:[IMGDMP]WECRASH.C;11
-RMS-F-DEV, error in device name or inappropriate device type for operation
26412: Source line not available

```

ANALYZE/PROCESS_DUMP

image name	set	base address	end address
CMA\$TIS_SHR	no	000000007B8CA000	000000007B8D7FFF
CODE0		FFFFFFFF80500000	FFFFFFFF805033FF
DATA1		000000007B8CA000	000000007B8CB3FF
DATA2		000000007B8CC000	000000007B8D13FF
DATA3		000000007B8D2000	000000007B8D21FF
DATA4		000000007B8D4000	000000007B8D41FF
DATA5		000000007B8D6000	000000007B8D63FF
DECC\$SHR	no	000000007BE7A000	000000007BF0DFFF
CODE0		FFFFFFFF8055C000	FFFFFFFF806C9DFF
DATA1		000000007BE7A000	000000007BEACFFF
DATA2		000000007BEBA000	000000007BEC2DFF
DATA3		000000007BECA000	000000007BED77FF
DATA4		000000007BEDA000	000000007BEDA9FF
DATA5		000000007BEEA000	000000007BEEA1FF
DATA6		000000007BEFA000	000000007BEFE7FF
DATA7		000000007BF0A000	000000007BF0D1FF
DPML\$SHR	no	000000007BB92000	000000007BBD1FFF
CODE0		FFFFFFFF80504000	FFFFFFFF8055B5FF
DATA1		000000007BB92000	000000007BBAC1FF
DATA2		000000007BBAE000	000000007BBDBBFF
DATA3		000000007BBBE000	000000007BBBE1FF
DATA4		000000007BBC0000	000000007BBCC9FF
DATA5		000000007BBCE000	000000007BBCE3FF
DATA6		000000007BBD0000	000000007BBD07FF
LIBOTS	no	000000007B5AA000	000000007B5B1FFF
DATA1		000000007B5AA000	000000007B5AC5FF
DATA2		000000007B5AE000	000000007B5AFBFF
DATA3		000000007B5B0000	000000007B5B01FF
LIBRTL	no	000000007B558000	000000007B5A9FFF
CODE0		FFFFFFFF8041C000	FFFFFFFF804BD7FF
DATA1		000000007B558000	000000007B5669FF
DATA2		000000007B568000	000000007B5697FF
DATA3		000000007B578000	000000007B5845FF
DATA4		000000007B588000	000000007B5881FF
DATA5		000000007B598000	000000007B59A5FF
DATA6		000000007B5A8000	000000007B5A99FF
PTHREAD\$RTL	no	000000007BBD2000	000000007BC27FFF
DATA0		000000007BBD2000	000000007BBD A1FF
DATA1		000000007BBD C000	000000007BBD F3FF
DATA2		000000007BBE0000	000000007BBE2FFF
DATA3		000000007BBE4000	000000007BC1E1FF
DATA4		000000007BC20000	000000007BC20BFF
DATA5		000000007BC22000	000000007BC247FF
DATA6		000000007BC26000	000000007BC275FF
*WECRASH	yes	0000000000010000	00000000000403FF

total images: 7

Thread Name	State	Substate	Policy	Pri
1 default thread	blocked	join	2 SCHED_OTHER	11
2 thread 0: counting	ready VP 0		SCHED_OTHER	11
3 thread 1: dumping	ready VP 0		SCHED_OTHER	11
4 thread 2	blocked	delay	SCHED_OTHER	11
5 thread 3	blocked	delay	SCHED_OTHER	11
6 thread 4	blocked	delay	SCHED_OTHER	11
7 thread 5: counting	ready VP 0		SCHED_OTHER	11
8 thread 6: dumping	running		SCHED_OTHER	11
9 thread 7	blocked	delay	SCHED_OTHER	11
10 thread 8	blocked	delay	SCHED_OTHER	11
11 thread 9	blocked	delay	SCHED_OTHER	11

module name	routine name	line	rel PC	abs PC
*WECRASH	th_run	26411	0000000000000244	0000000000030244
SHARE\$PTHREAD\$RTL_DATA0			000000000001F15C	000000007BC0315C
SHARE\$PTHREAD\$RTL_DATA0			000000000000F494	000000007BBF3494
			0000000000000000	0000000000000000

----- the above looks like a null frame in the same scope as the frame below

module name	routine name	line	rel PC	abs PC
SHARE\$PTHREAD\$RTL_DATA0			?	?

```

DBG>
DBG> set source/latest sys$disk:[]
DBG> examine/source .pc-4
module WECRASH
  26411:      lib$signal(SS$_IMGDMP);
DBG>

```

この例は、Alpha システムで /FULL 修飾子の使用による、マルチスレッド・プロセス・ダンプ上の ANALYZE/PROCESS コマンドの出力を示しています。

ANALYZE/RMS_FILE

Analyze/RMS_File ユーティリティを起動します。このユーティリティを使用すると、OpenVMS RMS ファイルの内部構造を検査し分析することができます。/RMS_FILE 修飾子は必須です。

Analyze/RMS_File ユーティリティについての詳細は、『OpenVMS Record Management Utilities Reference Manual』あるいはオンライン・ヘルプを参照してください。

フォーマット

ANALYZE/RMS_FILE ファイル指定[,...]

ANALYZE/SSLOG (Alpha/I64 のみ)

SSLOG.DAT ファイルを分析します。このファイルにはシステム・サービスのログ・データが格納されています。/SSLOG 修飾子は必須です。

詳細は、オンライン・ヘルプの ANALYZE/SSLOG、または『OpenVMS System Analysis Tools Manual』のシステム・サービス・ログに関する章を参照してください。

フォーマット

ANALYZE/SSLOG [修飾子]/[ファイル指定]/

ANALYZE/SYSTEM

システム・ダンプ・アナライザ・ユーティリティを起動します。このユーティリティは、実行中のシステムを分析します。/SYSTEM 修飾子は必須です。

Alpha システムおよび I64 システムにおけるシステム・ダンプ・アナライザ・ユーティリティについての詳細は、『OpenVMS System Analysis Tools Manual』またはオンライン・ヘルプを参照してください。VAX システムにおけるシステム・ダンプ・アナライザ・ユーティリティについての詳細は、『OpenVMS VAX System Dump Analyzer Utility Manual』¹を参照してください。

フォーマット

ANALYZE/SYSTEM

¹ このマニュアルはアーカイブされています。すでにメンテナンスされておらず、OpenVMS のドキュメント・セットにも含まれていません。ただし、<http://www.hp.com/go/openvms/doc>からオンラインで参照するか、オンライン・ヘルプで参照することができます。

APPEND

指定された出力ファイルに、1 つまたは複数の指定された入力ファイルの内容を追加します。

フォーマット

APPEND 入力ファイル[,...] 出力ファイル

パラメータ

入力ファイル[,...]

追加する 1 つ、または複数の入力ファイルの名前を指定します。入力ファイルは、すべて指定された順に、出力ファイルの最後に追加されます。複数の入力ファイルを指定する場合には、コンマ(,)またはプラス記号(+)で区切ります(コンマとプラス記号は、同じ意味に解釈されます)。

入力ファイルには、ワイルドカード文字(*と%)を使用することができます。

出力ファイル

入力ファイルが追加されるファイルの名前を指定します。

少なくとも 1 つの出力ファイルを指定しなければなりません。装置やディレクトリを指定しなかった場合には、APPEND コマンドは、現在の省略時の装置およびディレクトリを使用します。省略したファイル指定要素に対しては、APPEND コマンドは、入力ファイルの対応する要素を使用します。

出力ファイルの指定時にアスタリスク・ワイルドカード文字(*)を使用すると、APPEND コマンドは、指定した入力ファイルの対応する要素を使用します。複数の入力ファイルを追加している場合には、最初の入力ファイルの対応する要素を使用します。

説明

APPEND コマンドの構文と機能は、COPY コマンドの構文と機能に似ています。通常、APPEND コマンドは、1 つまたは複数のファイルの内容を既存のファイルの最後に追加します。この時、バージョン番号は増えません。/NEW_VERSION 修飾子を指定すると、その名前を持つファイルが存在しない場合は、新しい出力ファイルが作成されます。

DECwindows 複合ドキュメントに APPEND コマンドを使用する場合には、特に注意してください。詳細は『Guide to OpenVMS File Applications』を参照してください。

修飾子

/ALLOCATION=ブロック数

出力ファイルの初期占有サイズを、1 ブロック 512 バイトのブロック数で設定します。/ALLOCATION 修飾子が指定されていない場合や、/ALLOCATION 修飾子にブロック数を指定しない場合には、出力ファイルの初期占有サイズは、入力ファイルのサイズによって決定されます。

占有サイズは、/NEW_VERSION 修飾子が指定され、新しいファイルが実際に作成される場合にだけ適用されます。

/BACKUP

/BEFORE または /SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、最新のバックアップの日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/CREATED、/EXPIRED、および /MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として /CREATED 修飾子が使用されます。

/BEFORE[=時刻]

指定された時刻以前の時刻属性をもつファイルを選択します。絶対時刻、または絶対時刻とデルタ時間の組み合わせを指定します。また、BOOT、LOGIN、TODAY(省略時の設定)、TOMORROW、および YESTERDAY というキーワードも指定できます。適用する時刻属性は、/BACKUP、/CREATED(省略時の設定)、/EXPIRED、または /MODIFIED 修飾子のいずれかで指定します。

時刻指定の詳細は、『OpenVMS ユーザーズ・マニュアル』、またはオンライン・ヘルプのトピック Date を参照してください。

/BLOCK_SIZE=n

COPY が使用する省略時のブロック・サイズ (124) を指定変更します。1 ~ 127 の範囲の値を指定することができます。

/BY_OWNER[=利用者識別コード]

ファイル所有者の利用者識別コード (UIC) が、指定された所有者 UIC と一致する場合にだけ、そのファイルを選択します。/BY_OWNER 修飾子だけを指定し、UIC を省略した場合には、現在のプロセスの UIC が使用されます。

UIC を指定する場合には、『OpenVMS ユーザーズ・マニュアル』に説明されている標準的な UIC 形式を使用します。

/CONFIRM

/NOCONFIRM (省略時の設定)

ファイルに対する各 APPEND 操作の実行を確認するために、操作の前に確認を要求します。システムがプロンプトを表示したら、次のいずれかの応答を入力します。

YES	NO	QUIT
TRUE	FALSE	<input type="text" value="Ctrl/Z"/>
1	0	ALL
	<input type="text" value="Return"/>	

応答には、大文字と小文字を任意に組み合わせて使用することができます。単語による応答は、1文字以上(たとえば、TRUE の場合は T、TR、または TRU)に短縮することができます。肯定応答は、YES、TRUE、1 です。否定応答は、NO、FALSE、0、Return です。QUIT と Ctrl/Z は、その時点でコマンドの処理を停止する時に使用します。ALL を応答すると、コマンドは処理を継続しますが、プロンプトは表示されなくなります。上記に示されていない応答を入力すると、DCL はエラー・メッセージを出力し、同じプロンプトがもう一度表示されます。

/CONTIGUOUS

/NOCONTIGUOUS

出力ファイルが物理的に連続したディスク・ブロックを使用するかどうかを指定します。何も指定されていない場合には、APPEND コマンドは、対応する入力ファイルと同じ属性で出力ファイルを作成し、十分な連続領域が無くてもエラー・メッセージは表示しません。この修飾子は、/NEW_VERSION 修飾子とともに使用します。

入力ファイルが連続している場合には、連続した領域に出力ファイルを作成しようとしますが、連続した出力ファイルを作成するための十分な領域がない場合でもエラーは報告しません。属性の異なる複数の入力ファイルを追加する場合には、出力ファイルは、連続したファイルになることも非連続のファイルになることもあります。確実に連続した領域に出力ファイルを作成したい場合は、/CONTIGUOUS 修飾子を使用する必要があります。

/CREATED (省略時の設定)

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、作成日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/EXPIRED、および/MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/EXCLUDE=(ファイル指定[,...])

指定されているファイル(1 つまたは複数)と一致するファイルを、APPEND 操作から除外することを指定します。ファイル指定にはディレクトリを含むことはできませんが、装置を含むことはできません。ファイル指定の中で、ワイルドカード文字(*と%)を使用することができますが、相対バージョン番号を指定して特定のバージョンを除外することはできません。1 つのファイルだけを指定する場合には、括弧を省略できます。

/EXPIRED

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、満了日時をもとにファイルを選択します (満了日は、SET FILE /EXPIRATION_DATE コマンドで設定します)。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/CREATED、および/MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/EXTENSION=ブロック数

ファイルを拡張するたびに、出力ファイルに追加されるブロック数を指定します。/EXTENSION 修飾子を指定すると、/NEW_VERSION 修飾子も指定されていると解釈されるため、/NEW_VERSION 修飾子を指定する必要はありません。この修飾子は、/NEW_VERSION 修飾子と同時に指定します。

拡張サイズは、新しいファイルが実際に作成される場合にだけ使用されます。

/LOG**/NOLOG (省略時の設定)**

APPEND コマンドが、追加される各ファイルのファイル指定を表示するかどうかを制御します。/LOG 修飾子を使用した場合には、APPEND コマンドは、各追加操作を実行したあとで、追加されたブロック数またはレコード数と、入力ファイルと出力ファイルのファイル名を表示します。

/MODIFIED

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、最新の変更日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/CREATED、および/EXPIRED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/NEW_VERSION**/NONEW_VERSION (省略時の設定)**

指定した出力ファイルが存在しない場合に、APPEND コマンドが、新しい出力ファイルを作成するかどうかを制御します。省略時の設定では、指定した出力ファイルは既存のファイルでなければなりません。指定した出力ファイルが存在しない場合には、新しい出力ファイルを作成するために/NEW_VERSION 修飾子を使用します。出力ファイルが存在する場合には、/NEW_VERSION 修飾子は無視され、入力ファイルはその出力ファイルに追加されます。

/PROTECTION=(所有区分[: アクセス][,...])

出力ファイルに対して適用される保護を定義します。

- 所有区分は、システム (S)、所有者 (O)、グループ (G) またはワールド (W) から指定します。
- アクセス・コードは、読み込み (R)、書込み (W)、実行 (E) または削除 (D) から指定します。

出力ファイルが存在する場合には、省略時の保護属性（指定されていない保護属性を含む）は、そのファイルの現在の保護設定から適用され、新しい出力ファイルが作成される場合には、現在の省略時の保護設定が使用されます。この修飾子は、/NEW_VERSION 修飾子と同時に指定します。

保護コード指定についての詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

/READ_CHECK

/NOREAD_CHECK (省略時の設定)

すべてのレコードが正しく読み込まれたかどうか確認するために、入力ファイルの各レコードを 2 回ずつ読み込むことを、APPEND コマンドに指定します。

/SINCE[=時刻]

指定された時刻以降の時刻属性をもつファイルを選択します。絶対時刻、または絶対時刻とデルタ時間の組み合わせを指定します。また、BOOT、LOGIN、TODAY(省略時の設定)、TOMORROW、および YESTERDAY というキーワードも指定できます。適用する時刻属性は、/BACKUP、/CREATED(省略時の設定)、/EXPIRED、または/MODIFIED 修飾子のいずれかで指定します。

時刻指定の詳細は、『OpenVMS ユーザーズ・マニュアル』、またはオンライン・ヘルプのトピック Date を参照してください。

/WRITE_CHECK

/NOWRITE_CHECK (省略時の設定)

各レコードが正しく追加され、出力ファイルからそのレコードが正しく読み込めることを確認するために、レコードが書き出された後、そのレコードを読み込むことを APPEND コマンドに指定します。

例

1. \$ APPEND TEST3.DAT TESTALL.DAT

この APPEND コマンドは、省略時のディスクおよびディレクトリにある TEST3.DAT というファイルの内容を、同様に省略時のディスクおよびディレクトリにある TESTALL.DAT というファイルに追加します。

2. \$ APPEND/NEW_VERSION/LOG *.TXT MEM.SUM
 %APPEND-I-CREATED, USE\$:[MAL]MEM.SUM;1 created
 %APPEND-S-COPIED, USE\$:[MAL]A.TXT;2 copied to USE\$:[MAL]MEM.SUM;1 (1 block)
 %APPEND-S-APPENDED, USE\$:[MAL]B.TXT;3 appended to USE\$:[MAL]MEM.SUM;1 (3 records)
 %APPEND-S-APPENDED, USE\$:[MAL]G.TXT;7 appended to USE\$:[MAL]MEM.SUM;1 (51 records)

APPEND コマンドは、.TXT ファイル・タイプのすべてのファイルを、MEM.SUM という名前のファイルに追加します。/LOG 修飾子は、追加された各入力ファイルの指定の表示を要求します。MEM.SUM ファイルが存在しない場合は、APPEND コマンドは出力されるとおりに作成します。出力に示されるブ

ロックまたはレコード数は、ターゲット・ファイルの合計ではなく、ソース・ファイルを参照します。

3. \$ APPEND/LOG A.DAT, B.MEM C.*
 %APPEND-S-APPENDED, USE\$:[MAL]A.DAT;4 appended to USE\$:[MAL]C.DAT;4 (2 records)
 %APPEND-S-APPENDED, USE\$:[MAL]B.MEM;5 appended to USE\$:[MAL]C.DAT;4 (29 records)

APPEND コマンドは、ファイル A.DAT および B.MEM を、すでに存在している C.DAT ファイルに追加します。

4. \$ APPEND/LOG A.* B.*
 %APPEND-S-APPENDED, USE\$:[MAL]A.DAT;5 appended to USE\$:[MAL]B.DAT;1 (5 records)
 %APPEND-S-APPENDED, USE\$:[MAL]A.DOC;2 appended to USE\$:[MAL]B.DAT;1 (1 record)

入力ファイル指定と出力ファイル指定は、ともにファイル・タイプ・フィールドにワイルドカードが使用されています。APPEND コマンドは、ファイル名 A の各ファイルを、ファイル名として既存のファイルに追加します。最初の入力ファイルのファイル・タイプによって、出力ファイル・タイプが決まります。

5. \$ APPEND BOSTON"BILL_BESTON YANKEE"::DEMO1.DAT, DEMO2.DAT
 \$ _To: DALLAS::DISK1:[MODEL.TEST]TEST.DAT

この APPEND コマンドは、リモート・ノード BOSTON 上のファイル DEMO1.DAT と DEMO2.DAT を、リモート・ノード DALLAS 上のファイル TEST.DAT に追加します。

ASSIGN

論理名を作成し、指定された論理名に 1 つまたは複数の等価文字列を割り当てます。既に定義されている論理名を指定した場合には、古い等価名は新しい等価名で置き換えられます。

フォーマット

ASSIGN 等価名[,...] 論理名[:]

パラメータ

等価名[,...]

1 文字から 255 文字までの文字列を指定します。等価名は通常、ファイル名や装置名、他の論理名であり、特定の論理名テーブル内の論理名に割り当てられます。文字列に大文字の英数字、ドル記号(\$)、またはアンダースコア文字(_)以外の文字が含まれている場合には、文字列を二重引用符(" ")で囲む必要があります。等価文字列に二重引用符が含まれている場合には、2 つの連続した二重引用符("")を指定します。1 つの論理名に複数の等価名を指定すると、サーチ・リストが生成されます。1 つの論理名は、最大 128 個までの等価名を持つことができます。

ファイル指定として使用される等価名を指定する場合には、等価名がファイル指定としてそのまま使用されるときに必要な句読点(コロン(:)、かぎ括弧([])、ピリオド(.))を含む必要があります。したがって、装置名を等価名として指定する場合には、装置名の最後にコロンを指定しなければなりません。

ASSIGN コマンドを使用すると、同じ論理名を複数の等価名に与えることができます。1 つの論理名に対して複数の等価名を指定する場合には、サーチ・リストが生成されます。サーチ・リストについての詳細は、『OpenVMS ユーザーズ・マニュアル』を参照してください。

論理名[:]

論理名文字列を指定します。論理名文字列には、1 文字から 255 文字までを含むことができます。指定した論理名テーブル内の等価名を表すのに論理名を選択することができます。

論理名に大文字の英数字、ドル記号、アンダースコア以外の文字が含まれている場合には、論理名を二重引用符で囲まなければなりません。論理名に二重引用符が含まれる場合には、論理名全体を二重引用符で囲み、二重引用符が必要な位置に連続した 2 つの二重引用符を指定します。論理名の最後にコロンを指定した場合、その名前を論理名テーブルに登録する前に、システムがコロンを削除します。(この点は DEFINE コマンドと異なります。DEFINE コマンドでは、コロンはそのまま保存されます。)

プロセス・ディレクトリ論理名テーブル (LNM\$PROCESS_DIRECTORY) またはシステム・ディレクトリ論理名テーブル (LNM\$SYSTEM_DIRECTORY) に登録される論理名は、1 文字から 31 文字の長さでなければなりません。この長さには、ドル記号やアンダースコア文字も含まれます。プロセス・ディレクトリまたはシステム・ディレクトリに登録する論理名が、論理名テーブル名に変換される場合には、名前の中の英文字はすべて大文字である必要があります。省略時の設定では、論理名はプロセス論理名テーブルに登録されます。

論理名に英数字、ドル記号およびアンダースコア文字以外の文字を含める場合には、名前を二重引用符で囲みます。論理名に二重引用符を含める場合には、名前を二重引用符で囲み、二重引用符の必要な部分に 2 つの連続した二重引用符を置きます。論理名を二重引用符で囲むと、英字の大文字と小文字の区別は保持されます。

説明

ASSIGN コマンドは、1 つまたは複数の等価名を表わす論理名を定義して、論理名テーブルにエントリを作成します。等価名は、装置名、他の論理名、ファイル指定、またはその他の任意の文字列です。

論理名を格納したい論理名テーブルを指定するには、/PROCESS、/JOB、/GROUP、/SYSTEM、または/TABLE 修飾子を使用します。複数の修飾子を指定した場合は、最後に指定した修飾子だけが有効です。テーブルを指定しない場合は、省略時の設定により/TABLE=LNM\$PROCESS (または/PROCESS) に格納されます。

作成する論理名のアクセス・モードを指定するには、/USER_MODE、/SUPERVISOR_MODE、または/EXECUTIVE_MODE 修飾子を使用します。複数の修飾子を指定した場合は、最後に指定した修飾子だけが有効です。アクセス・モードを指定しない場合は、スーパーバイザ・モード名が作成されます。論理名は、その論理名を格納しているテーブルと同じモードかまたは外側のモードで作成できます (ユーザ・モードが一番外側のモードで、エグゼクティブ・モードが一番内側のモードです)。

名前ごとにアクセス・モードが異なっていれば、同じ論理名テーブルに同じ名前を持つ複数の論理名を格納することができます (ただし、テーブル内の既存の論理名が NO_ALIAS 属性を持つ場合は、このテーブルで同じ名前を使用して外側のモードの論理名を作成できません)。

既存の論理名と同じテーブルで同じモードの同じ名前を持つ論理名を作成すると、新しい論理名で既存の論理名が置き換えられます。

DEFINE コマンドを使用して、論理名を作成することもできます。テーブルから論理名を削除するには、DEASSIGN コマンドを使用します。

注意

SYS\$SYSTEM: 内の実行可能イメージのファイル名と同じ論理名は割り当てないでください。このような論理名を使用すると、そのイメージを起動できなくなります。

論理名の作成と使用の方法については、『OpenVMS ユーザーズ・マニュアル』を参照してください。

修飾子

/CLUSTER_SYSTEM

この修飾子を使用するためには、SYSTEM アカウントでログインするか、SYSNAM (システム論理名) 特権または SYSPRV (システム) 特権を持っている必要があります。

LNMSYSCLUSTER テーブルにクラスタ・ワイド論理名を割り当てます。

/EXECUTIVE_MODE

SYSNAM (システム論理名) 特権が必要です。

指定されたテーブルにエグゼクティブ・モードの論理名を作成します。

/EXECUTIVE_MODE 修飾子を指定しても SYSNAM 特権が与えられていない場合には、ASSIGN コマンドはその修飾子を無視して、スーパーバイザ・モードの論理名を作成します。論理名のモードは、登録する論理名テーブルのモードと同じか、またはより低いモードでなければなりません。

/GROUP

SYSPRV (システム特権) または GRPNAM (グループ論理名) 特権が必要です。

論理名をグループ論理名テーブルに登録します。UIC (利用者識別コード) のグループ番号が等しい他のユーザは、その論理名にアクセスすることができます。/GROUP 修飾子は/TABLE=LNMSGROUP の同意語です。

/JOB

論理名をジョブ単位の論理名テーブルに登録します。論理名を作成しているプロセスと同じジョブの階層構造に含まれるプロセスはすべて、その論理名にアクセスすることができます。/JOB 修飾子は/TABLE=LNMSJOB の同意語です。

/LOG (省略時の設定)

/NOLOG

既存の名前を置換する論理名を定義するときに、メッセージが表示されるかどうかを制御します。

/NAME_ATTRIBUTES[=(キーワード[,...])]

論理名の属性を指定します。省略時の設定では、属性は何も設定されません。属性として、次のキーワードを指定することができます。

CONFINE	SPAWN コマンドでサブプロセスを作成した時、論理がサブプロセスにコピーされないことを指定します。このキーワードは、利用者固有のテーブルに論理名を作成する場合にだけ、意味をもちます。
NO_ALIAS	より低い特権の（外側の）アクセス・モードでは、このテーブルに同じ名前の論理名を作成できないことを指定します。同じ名前を持つ他の論理名が、このテーブルより低い特権のアクセス・モードで既に存在する場合には、その論理名は削除されます。

キーワードを 1 つだけしか指定しない場合には、括弧を省略することができます。指定した属性だけが設定されます。

/PROCESS (省略時の設定)

論理名をプロセス論理名テーブルに登録します。/PROCESS 修飾子は /TABLE=LNМ\$PROCESS の同意語です。

/SUPERVISOR_MODE (省略時の設定)

スーパーバイザ・モードの論理名を、指定されたテーブルに作成します。

/SYSTEM

SYSNAM（システム論理名）または SYSPRV（システム特権）特権が必要です。

論理名をシステム論理名テーブルに登録します。システムのすべてのユーザが、その論理名にアクセスすることができます。/SYSTEM 修飾子は /TABLE=LNМ\$SYSTEM の同意語です。

/TABLE=テーブル名

共用可能な論理名テーブルの名前を指定する場合には、そのテーブルに対して書き込み（W）アクセス権が必要です。

論理名が登録される論理名テーブルの名前を指定します。/TABLE 修飾子を使用すれば、ユーザが定義した論理名テーブル（CREATE/NAME_TABLE コマンドによって作成されるテーブル）や、プロセス論理名テーブル、ジョブ論理名テーブル、グループ論理名テーブル、システム論理名テーブルのいずれも指定することができ、あるいはプロセス論理名ディレクトリ・テーブルまたはシステム論理名ディレクトリ・テーブルを指定することもできます。

複数の等価名を持つ論理名を使ってテーブル名を指定すると、その論理名は最初に検出されたテーブルに登録されます。たとえば、ASSIGN/TABLE=LNМ\$FILE_DEV を指定した時、LNМ\$FILE_DEV は LNМ\$PROCESS と LNМ\$JOB、LNМ\$GROUP、LNМ\$SYSTEM に等しいと定義されているので、この場合には、論理名が LNМ\$PROCESS に登録されます。

/TABLE 修飾子を明示的に指定しなかった場合には、省略時の設定として、/TABLE=LNМ\$PROCESS が使用されます。

/TRANSLATION_ATTRIBUTES[=(キーワード[,...])]
 等価名修飾子

論理名を等価文字列に変換する際の、1 つまたは複数の属性を指定します。変換属性に対しては、次のキーワードを指定できます。

CONCEALED	等価文字列が隠し装置名であることを指定します。 隠し装置名が定義されると、装置を参照するメッセージの中で、等価文字列ではなく論理名が表示されます。CONCEALED 属性を指定した場合、等価文字列は物理装置名でなければなりません。
TERMINAL	等価文字列の反復変換を行わないことを指定します。したがって、論理名変換は現在の等価文字列で終了します。

キーワードを1 つだけしか指定しない場合には、括弧を省略することができます。指定した属性だけが設定されます。

同じ論理名に対して複数の異なる等価文字列を指定する場合は、各等価文字列に対して異なる変換属性を指定できます。

/USER_MODE

ユーザ・モードの論理名を、指定したテーブルに作成します。

プロセス論理名テーブルに作成されたユーザ・モード論理名は、ただ1 つのイメージ実行においてだけ使用されます。すなわち、プロセスの中でのイメージの実行の終了時に（つまり、イメージまたはユーザ・プログラムを実行する DCL コマンドが終了した後で）、ユーザ・モード・エントリは、論理名テーブルから削除されます。また、ユーザ・モード論理名は、コマンド・プロシージャを起動および実行している時に、自動的に削除されます。

例

1. \$ ASSIGN \$DISK1:[CREMERS.MEMOS] MEMOSD

この ASSIGN コマンドは、ファイル指定の一部
分\$DISK1:[CREMERS.MEMOS]を論理名 MEMOSD に割り当てます。

2. \$ ASSIGN/USER_MODE \$DISK1:[FODDY.MEMOS]WATER.TXT TM1

この ASSIGN コマンドは、論理名 TM1 にファイル指定を割り当てています。イメージの実行後に、この論理名は自動的に削除されます。

3. \$ ASSIGN XXX1:[HEROLD] ED
\$ PRINT ED:TEST.DAT
Job 274 entered on queue SYS\$PRINT

この ASSIGN コマンドは、ED という論理名を、XXX1 というディスクの
[HEROLD] というディレクトリ名に割り当てます。このあと、ED という論理名を参照すると、この論理名が指定されたディスクおよびディレクトリとして使

用されます。PRINT コマンドは、XXX1:[HEROLD]TEST.DAT というファイルを印刷するジョブを、システム・プリンタのキューに登録します。

4. \$ ASSIGN YYY2: TEMP:
\$ SHOW LOGICAL TEMP
"TEMP" = "YYY2:" (LNM\$PROCESS_TABLE)
\$ DEASSIGN TEMP

この ASSIGN コマンドは、TEMP という論理名を YYY2 という装置に割り当てます。TEMP は、スーパーバイザ・モードで作成され、プロセス論理名テーブルに登録されます。SHOW LOGICAL コマンドは、論理名の割り当てが実行されたかどうかを確認します。この ASSIGN コマンドでは、TEMP という論理名の最後にコロンが指定されていますが、コマンド・インタプリタは論理名テーブルにその論理名を登録する前に、コロンを削除します。したがって、このあとの DEASSIGN コマンドでは、TEMP だけを指定し、コロンは省略することができます。SHOW LOGICAL コマンドでは、コロンを省略(たとえば、SHOW LOGICAL TEMP)しなければなりません。

5. \$ MOUNT TTT1: MASTER TAPE
\$ ASSIGN TAPE:NAMES.DAT PAYROLL
\$ RUN PAYROLL
.
.
.

この例では、装置 TTT1: にマウントされたボリューム(ラベル名 MASTER)に論理名 TAPE を MOUNT コマンドで割り当てています。ASSIGN コマンドで、論理装置 TAPE 上のファイル NAMES.DAT に論理名 PAYROLL を割り当てています。したがって、プログラムが論理名 PAYROLL で参照する OPEN 要求を出すと、ボリューム・ラベル名 MASTER というテープ上のファイル NAMES.DAT がオープンされます。

6. \$ CREATE/NAME_TABLE TABLE1
\$ ASSIGN/TABLE=LNM\$PROCESS_DIRECTORY TABLE1, -
_\$ LNM\$PROCESS, LNM\$JOB, LNM\$GROUP, LNM\$SYSTEM LNM\$FILE_DEV
\$ ASSIGN/TABLE=TABLE1 -
_\$ /TRANSLATION_ATTRIBUTES=CONCEALED DKA1: WORK_DISK

CREATE/NAME_TABLE コマンドは、TABLE1 というプロセス固有の論理名テーブルを作成します。

最初の ASSIGN コマンドにより、ファイル指定または装置名の論理名変換で、TABLE1 が最初に検索されます。これは、TABLE1 が、LNM\$FILE_DEV という論理名に対する等価文字列の最初の項目であるためです。論理名 LNM\$FILE_DEV は、装置またはファイル指定の変換時に、論理名テーブルの省略時の検索順序を決定します。

2 番目の ASSIGN コマンドは、WORK_DISK という論理名を DKA1 という物理装置に割り当て、その論理名を TABLE1 に登録します。この論理名は、隠し属性を持っています。したがって、システム・メッセージには、WORK_DISK という論理名が表示されます。

7. \$ ASSIGN/TABLE=LNMS\$PROCESS/TABLE=LNMS\$GROUP DKA0: SYSFILES
\$ SHOW LOGICAL SYSFILES
"SYSFILES" = "DKA0:" (LNMS\$GROUP_000240)

この ASSIGN コマンドには、矛盾する修飾子が含まれています。このような場合には、最後に指定された修飾子を使用します。したがって、SHOW LOGICAL コマンドからの応答は、論理名がグループ論理名テーブルに登録されたことを示しています。

8. \$ ASSIGN/TABLE=LNMS\$GROUP 'F\$TRNLNM("SYS\$COMMAND")' TERMINAL
%DCL-I-SUPERSEDE, previous value of TERMINAL has been superseded

この例では、レキシカル関数 F\$TRNLNM で論理名 SYS\$COMMAND を変換し、その結果を使用して等価名 TERMINAL を定義しています。ASSIGN コマンドのメッセージは、論理名 TERMINAL が既にグループ論理名テーブルに定義済みであったので、以前のエントリを上書きしたことを示しています。

このコマンドを LOGIN.COM ファイルで使用すれば、各ターミナル・セッション開始時に論理名 TERMINAL が再定義されます。現在のプロセスやそのサブプロセスでは、論理名 TERMINAL を使用して現在の端末にメッセージを出力できます。

9. \$ ASSIGN DALLAS::DMA1: DATA

論理名 DATA にリモート・ノード DALLAS 上の装置 DMA1 を割り当てています。これ以降の論理名 DATA の参照は、リモート・ノード上のディスクに対するものとなります。

10. \$ CREATE AVERAGE.COM
\$ ASSIGN/USER_MODE SYS\$COMMAND: SYS\$INPUT
\$ EDIT/EDT AVERAGE.FOR
\$ FORTRAN AVERAGE
\$ LINK AVERAGE
\$ RUN AVERAGE
87
80
90
9999
\$ EXIT
Ctrl/Z
\$ @AVERAGE.COM

CREATE コマンドでコマンド・プロシージャ AVERAGE.COM を作成しています。

コマンド・プロシージャ内の ASSIGN コマンドは、/USER_MODE 修飾子で一時的に SYS\$INPUT の値を変更しています。EDT エディタが起動されると、ターミナルから入力を受け取ります。このようにして、プログラム AVERAGE.FOR を会話形式で作成または変更できます。

EDT を終了すると、ユーザ・モードの SYS\$INPUT は削除され、元の定義に戻ります (入力ストリームはコマンド・プロシージャで与えられます)。したがって、プログラム AVERAGE への入力は、コマンド・プロシージャ内から入力されます。

ASSIGN/MERGE

1つのキューからすべてのジョブを削除し、そのジョブを他の既存のキューに登録します。実行中のジョブに対しては影響しません。

両方のキューに対する管理 (M) 権が必要です。

フォーマット

ASSIGN/MERGE ターゲット・キュー[:] ソース・キュー[:]

パラメータ

ターゲット・キュー[:]

ジョブに登録するキューの名前を指定します。

ソース・キュー[:]

新しいキューに移されるジョブが登録されている、キューの名前を指定します。

説明

ASSIGN/MERGE コマンドは、あるキューから待ち状態のジョブを削除して、別のキューに登録します。このコマンドは、ターゲット・キューおよびソース・キュー内の実行中のジョブには影響しません。ソース・キューで現在実行中のジョブは、そのキューで実行を終了します。このコマンドは、バッチ・キューにも使用できますが、一般にプリント・キューに使用します。

ASSIGN/MERGE コマンドは、特にライン・プリンタの動作が不調な場合に便利です。ASSIGN/MERGE コマンドを入力すると、既存のジョブを別の印刷装置に変更できます。ジョブの損失または中断なしにマージ操作を実行するには、STOP/QUEUE/NEXT コマンドでソース・キューを停止させます。次に、STOP/QUEUE/REQUEUE コマンドを入力して、ソース・キューに入っている現在のジョブをターゲット・キューに移動します (STOP/QUEUE/REQUEUE コマンドでジョブを移動できなかった場合は、STOP/QUEUE/RESET コマンドを使用してキューの制御を回復してください)。STOP コマンドを入力してから、ASSIGN/MERGE コマンドを入力してください。

例

1. \$ STOP/QUEUE/NEXT LPB0
 \$ STOP/QUEUE/REQUEUE=LPA0 LPB0
 \$ ASSIGN/MERGE LPA0 LPB0

STOP/QUEUE/NEXT コマンドは、LPB0 というキューにおいて他のジョブが実行されないようにします。STOP/QUEUE/REQUEUE コマンドは、LPB0 で現在実行されているジョブを LPA0 というキューに再登録します。ASSIGN/MERGE コマンドは、ジョブを LPB0 というプリント・キューから削除し、LPA0 プリント・キューに移動します。

ASSIGN/QUEUE

論理キューを1つの実行キューに割り当てます。ASSIGN/QUEUEコマンドは、プリント・キューまたはターミナル・キューに対してだけ使用できます。

両方のキューに対する管理 (M) 権が必要です。

フォーマット

ASSIGN/QUEUE キュー名[:] 論理キュー名[:]

パラメータ

キュー名[:]

指定した論理キュー名が割り当てられる、実行キューの名前を指定します。論理キュー、総称キュー、バッチ・キューの指定はできません。

論理キュー名[:]

指定した実行キューに割り当てられる論理キューの名前を指定します。

説明

ASSIGN/QUEUE コマンドは、論理キューと実行キューの間に、1対1対応を設定します。論理キューに登録されたジョブは、印刷するために、常に指定された実行キューに登録されます。

ASSIGN/QUEUE コマンドを入力すると、論理キューは実行できません。

論理キューを初期化したら、ASSIGN/QUEUE コマンドを使用して論理キューを既存の実行キューと対応付けてください。論理キューを設定するには、次の手順に従ってください。

1. INITIALIZE/QUEUE コマンドで、論理キューを初期化します (/START 修飾子は使用しないでください)。
2. 既存の実行キューに論理キュー名を割り当てます。
3. START/QUEUE コマンドで、論理キューを開始します。

論理キューに対して START/QUEUE コマンドを入力すると、ジョブを処理するために論理キューに送ることができます。

例

1. \$ INITIALIZE/QUEUE/DEFAULT=FLAG=ONE/START LPA0
 \$ INITIALIZE/QUEUE TEST_QUEUE
 \$ ASSIGN/QUEUE LPA0 TEST_QUEUE
 \$ START/QUEUE TEST_QUEUE

この例では、まず、LPA0 というプリント・キューが初期化され開始されます。LPA0 キューは、フラグ・ページが各ジョブの前に印刷されるように設定されています。2 番目の INITIALIZE/QUEUE コマンドは、TEST_QUEUE という論理キューを作成します。ASSIGN/QUEUE コマンドは、LPA0 というプリント・キューに TEST_QUEUE という論理キューを割り当てます。START/QUEUE コマンドは、論理キューを開始します。

2. \$ INITIALIZE/QUEUE/START LPB0

この例では、論理キューが初期化されていないので、ASSIGN/QUEUE コマンドは必要ありません。プリント・キューが初期化されています。つまり、LPB0 が、ライン・プリンタの名前です。このように、INITIALIZE/QUEUE/START コマンドを入力したあと、LPB0 にジョブを登録することができます。

ATTACH

現在処理中のプロセスから、指定したプロセスへ制御を移します (元のプロセスは、この後ハイバネートします)。

使用している端末にメールボックスが対応づけられている場合は、ATTACH および SPAWN コマンドは使用できません。

フォーマット

ATTACH [プロセス名]

パラメータ

プロセス名

制御を移す親プロセス、またはサブプロセスの名前を指定します。ここで指定するプロセスは既に存在し、現在処理中のジョブの一部で、現在処理中のプロセスと同じ入力ストリームを共有していなければなりません。ただし現在のプロセス、またはNOWAIT 修飾子を使用して作成したサブプロセスは指定できません。

プロセス名は、1 から 15 文字の英数字で指定します。指定したプロセスに接続できない場合は、エラー・メッセージが表示されます。

プロセス名パラメータは、/IDENTIFICATION 修飾子と同時に指定できません。

説明

ATTACH コマンドを使用すると、入力ストリームを別のプロセスに接続できます。また、1 つのサブプロセスから別のサブプロセス、または親プロセスに制御を移すことができます。

ATTACH コマンドを入力すると、親プロセス、または“アタッチ元”プロセスはハイバネート状態になり、入力ストリームは指定した“アタッチ先”プロセスに接続されます。ATTACH コマンドを使用すると、同一ジョブ内のサブプロセス (SPAWN/WAIT または別の ATTACH コマンドによりハイバネート状態になっている) に、制御を移すことができます。現在処理中のプロセス、現在処理中のジョブの一部ではないプロセス、存在しないプロセスに制御を移すことはできません。これらのプロセスに制御を移そうとすると、エラー・メッセージが表示されます。

また SPAWN/WAIT コマンドと ATTACH コマンドを使用すると、作成したサブプロセスを終了しなくても親プロセスに制御を戻すことができます。詳細は、 SPAWN コマンドの説明を参照してください。

修飾子

/IDENTIFICATION=pid

端末制御を移すプロセスのプロセス ID(PID) を指定します。先行の 0 は省略できます。/IDENTIFICATION 修飾子は、プロセス名と同時に指定できません。

/IDENTIFICATION 修飾子を省略した場合は、プロセス名を指定しなければなりません。

例

1. \$ ATTACH JONES_2

この例で ATTACH コマンドは、端末制御をサブプロセス JONES_2 に移しています。

2. \$ ATTACH/IDENTIFICATION=30019

この例で ATTACH コマンドは、現在処理中のプロセスから、PID が 30019 であるプロセスに制御を移します。/IDENTIFICATION 修飾子が指定されているので、プロセス名は省略されています。

BACKUP

Backup コーティリティ (BACKUP) を起動し、次のいずれかのバックアップ操作を行います。

- ディスク・ファイルのコピーを作成する。
- ディスクまたは磁気テープ上に、BACKUP により作成されたファイル中のデータとしてディスク・ファイルを保存 (セーブ) する (BACKUP により作成されたファイルをセーブ・セットと呼ぶ)。
- BACKUP セーブ・セットからディスク・ファイルを回復 (リストア) する。
- BACKUP セーブ・セットに保存されているディスク・ファイルまたはファイルを、他のディスク・ファイルと比較する。
- BACKUP セーブ・セットに保存されているファイルに関する情報を出力装置またはファイルに表示する。

BACKUP を使用してシステム・ディスクをバックアップすることはできません。システム・ディスクはブートストラップを使用しなければなりません。

BACKUP およびシステム・ディスクのバックアップについての詳細は、
『OpenVMS システム管理者マニュアル』および『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』あるいはオンライン・ヘルプを参照してください。

フォーマット

BACKUP 入力指定 出力指定

CALL

コマンド・プロシージャ内でラベルを付けられたサブルーチンへ制御を移します。

フォーマット

CALL ラベル [パラメータ[...]]

パラメータ

ラベル

コマンド行の最初に現われる，1 ～ 255 桁の英数字からなるラベル名です。ラベルにブランクを含めることはできません。CALL コマンドが実行されるとき，指定されたラベルに後続するコマンドへ制御が移ります。

ラベルは，現在のコマンド・プロシージャの CALL 文の前に，あるいは後に置くことができます。コマンド・プロシージャ内のラベルは，コロン (:) で終了してください。サブルーチンのラベルは一意でなければなりません。

より内側のプロシージャ・レベルのラベルは，外側のプロシージャレベルからはアクセスできません。次の例を参照してください。

```
$CALL B
$A: SUBROUTINE
$ B: SUBROUTINE
$ ENDSUBROUTINE
$ENDSUBROUTINE
```

この例では，サブルーチン A 内のラベル B は，サブルーチン A より外側ではアクセスできません。

パラメータ[...]

コマンド・プロシージャに渡す 1 つから 8 つのパラメータを指定します。空パラメータは連続する 2 つの引用符 ("") を用いて指定します。これらのパラメータは，文字列値を入力した順に，P1，P2，..., P8 と 8 つまでのシンボルに割り当てられます。シンボルは，指定されたコマンド・プロシージャの内部だけで有効です。各パラメータは，1 つまたは複数のスペースで区切ります。

パラメータは，英数字または特殊文字を含む文字列値として指定することができますが，次の制約があります。

- コマンド・インタプリタは英字を大文字に変換し，各パラメータを区切るためにスペースを使用します。したがって，スペースや小文字を含むパラメータを渡すためには，パラメータを引用符(“ ”)で囲む必要があります

- 最初のパラメータがスラッシュ文字(/)から始まる場合には、パラメータを二重引用符で囲まなければなりません。
- リテラルとしての引用符、およびスペースを含むパラメータを渡すには、文字列全体を二重引用符で囲み、文字列の中に連続する 2 つの二重引用符を指定します。次の例を参照してください。

```
$ CALL SUB1 "Never say ""quit"""
```

制御が SUB1 に移るとき、パラメータ P1 には次の文字列が割り当てられます。

```
Never say "quit"
```

文字列に引用符が含まれており、スペースが含まれていない場合には、引用符はそのまま保存され、引用符で囲まれた英字は小文字のまま保存されます。次の例を参照してください。

```
$ CALL SUB2 abc"def"ghi
```

制御が SUB2 に移るとき、パラメータ P1 には次の文字列を割り当てられます。

```
ABCdefGHI
```

シンボルをパラメータとして使用する場合には、シンボル置換を実行するために、シンボルを一重引用符('')で囲む必要があります。次の例を参照してください。

```
$ NAME = "JOHNSON"
$ CALL INFO 'NAME'
```

一重引用符を使用すると、"JOHNSON"という値が"NAME"というシンボルと置き換えられます。したがって、"JOHNSON"というパラメータが P1 として、サブルーチン INFO に渡されます。

説明

CALL コマンドは、コマンド・プロシージャ内でラベルを付けられたサブルーチンへ制御を移します。CALL コマンドは@ (プロシージャ実行) コマンドに同様に、新しいプロシージャ・レベルを作成します。CALL コマンドを使用する利点は、プロシージャを処理するためにファイルをオープンしたりクローズする必要がないことです。また、複数のプロシージャは複数のファイルではなく 1 つのファイル中にあるので、CALL コマンドを使用するとプロシージャの管理が簡単です。

CALL コマンドを使用してサブルーチンに制御を移すと、新しいプロシージャ・レベルが作成され、指定された引数の値に P1 から P8 までのシンボルが割り当てられます。EXIT コマンドが実行されるまで、サブルーチンでの実行が続きます。EXIT コマンドが実行されると、制御は CALL コマンドの次のコマンド行に移ります。

プロシージャは、最大で 32 レベルまでネストさせることができます。この中には、コマンド・プロシージャの任意の組み合わせやサブルーチンの呼び出しを含めることができます。ネストしたサブルーチン構造内で定義されたローカル・シンボルやラベルは、@コマンドでルーチンを起動した場合と同様に扱われます。つまり、ラベルはそれを定義したサブルーチン・レベルでのみ有効です。

より外側のレベルで定義されたローカル・シンボルは、それより内部のネスト・レベルのサブルーチンでは使用できます。つまり、ローカル・シンボルの読み込みはできますが書き込みはできません。より外側のサブルーチン・レベルでローカルなシンボルに値を割り当てると、現在のサブルーチン・レベルに新しいシンボルが作成されます。この時、より外側のプロシージャ・レベルのシンボルは変更されません。

SUBROUTINE および ENDSUBROUTINE コマンドは、サブルーチンの開始および終了を定義します。サブルーチンへのエントリ・ポイントを定義するラベルは、SUBROUTINE コマンドの直前または同一コマンド行上で指定しなければなりません。

1 つのサブルーチンには、1 つのエントリ・ポイントしか含めることができません。サブルーチンは、最初の実行可能文 SUBROUTINE コマンドで開始しなければなりません。プロシージャ内で EXIT コマンドを指定しないと、ENDSUBROUTINE コマンドが EXIT コマンドと同じ処置を行います。

SUBROUTINE コマンドは、実行されるコンテキストに依存する 2 つの異なる処理を行います。CALL コマンドの結果として SUBROUTINE コマンドを実行する場合は、新しいプロシージャ・レベルを開始し、CALL コマンドで指定したように P1 から P8 までのパラメータを定義し、サブルーチンの実行を開始します。CALL コマンドを使用せずに起動されたプロシージャの実行フローで SUBROUTINE 動詞が検出された場合は、SUBROUTINE 以降のすべてのコマンドは、対応する ENDSUBROUTINE コマンドが検出されるコマンドまでスキップされます。

注意

SUBROUTINE および ENDSUBROUTINE コマンドは、3 文字以下に短縮することはできません。

修飾子

/OUTPUT=ファイル指定

すべての出力を、指定したファイルまたは装置に書き込みます。省略時設定では、出力は現在の SYS\$OUTPUT という論理装置に送られ、出力ファイル・タイプは LIS です。システムからの応答とエラー・メッセージは、指定したファイルと SYS\$COMMAND の両方に書き込まれます。/OUTPUT を指定する場合には、修飾子

は CALL コマンドのすぐあとに指定しなければなりません。出力ファイル指定には、ワイルド・カード文字は使用できません。

コマンド・プロシージャからの出力先を変更するには、SYS\$OUTPUT の定義を変更することもできます。コマンド・プロシージャの 1 行目として次のコマンドを指定すると、出力は指定したファイルに送られます。

```
$ DEFINE SYS$OUTPUT filespec
```

このプロシージャが終了すると、SYS\$OUTPUT は元の等価文字列に戻されます。この方法を使うと、コマンド・プロシージャの実行時に/OUTPUT 修飾子を使用した場合と同じ結果になります。

例

```
1. $
   $! CALL.COM
   $
   $! Define subroutine SUB1
   $!
   $ SUB1: SUBROUTINE
       .
       .
       .
   $     CALL SUB2           !Invoke SUB2 from within SUB1
       .
       .
       .
   $     @FILE               !Invoke another procedure command file
       .
       .
       .
   $     EXIT
   $     ENDSUBROUTINE      !End of SUB1 definition
   $!
   $! Define subroutine SUB2
   $!
   $ SUB2: SUBROUTINE
       .
       .
       .
   $     EXIT
   $     ENDSUBROUTINE      !End of SUB2 definition
   $!
   $! Start of main routine. At this point, both SUB1 and SUB2
   $! have been defined but none of the previous commands have
   $! been executed.
   $!
   $ START:
   $     CALL/OUTPUT=NAME$.LOG SUB1 "THIS IS P1"
```

CALL

```
.  
. .  
$ CALL SUB2 "THIS IS P1" "THIS IS P2"  
. .  
$ EXIT !Exit this command procedure file
```

このコマンド・プロシージャは、ラベルの付いたサブルーチンへ制御を移すための CALL コマンドの使い方を示しています。この例は、サブルーチンあるいは別のコマンド・ファイルを、サブルーチンとして呼べることを示します。

CALL コマンドは、出力ファイルを NAMES.LOG として、サブルーチン SUB1 を起動します。また、他のユーザにファイルへの書き込みアクセスを認めます。サブルーチン SUB2 は SUB1 に呼ばれます。プロシージャは SUB2 を実行し、次にコマンド・プロシージャ FILE.COM を起動するために@（プロシージャ実行）コマンドを用います。

SUB1 のコマンドがすべて実行されると、メイン・プロシージャの CALL コマンドは次に SUB2 を呼び出します。SUB2 が実行されるまで、プロシージャは継続します。

CANCEL

RUN コマンドまたは\$SCHDWK システム・サービスでスケジューリングされたウェイクアップ要求を含め、指定したプロセスに対するウェイクアップ要求を取り消します。

次のいずれか 1 つが必要です。

- プロセスの所有権
- 所有権はないが同一グループ内のプロセスに対してスケジューリングされたウェイクアップ要求を取り消すための GROUP 特権
- システム上のすべてのプロセスに対してスケジューリングされたウェイクアップ要求を取り消すための WORLD 特権

フォーマット

CANCEL `[[ノード名::]プロセス名]`

パラメータ

ノード名::

指定したプロセスが実行中のノードの名前を指定します。

現在のプロセスとは異なる OpenVMS Cluster システム上のノード名を指定することはできません。

プロセス名

ウェイクアップ要求を取り消したいプロセスの名前を指定します。プロセス名には、15 文字までの英数字を使用できます。

ここで指定するプロセスは、現在のプロセスと同一グループ内のプロセスでなければなりません。

説明

CANCEL コマンドは、指定したプロセスに対してスケジューリングされたウェイクアップ要求を取り消します。

CANCEL コマンドは、指定したプロセスを削除することはありません。CANCEL コマンドを発行した時に、そのプロセスでイメージを実行している場合は、イメージの実行終了後、そのプロセスは終了するのではなく、ハイバネートします。

CANCEL

ウェイクアップ要求が取り消されたハイバネートしているプロセスを削除するには、STOP コマンドを使用します。SHOW PROCESS コマンドで/SUBPROCESS 修飾子を指定すると、サブプロセスが削除されたかどうか確認できます。

ローカル・プロセス名は、リモート・プロセス名と似ています。そのため ATHENS::SMITH と指定すると、システムは、ノード ATHENS のプロセス SMITH をチェックする前に、ローカル・ノードでプロセス ATHENS::SMITH をチェックします。

/IDENTIFICATION=pid 修飾子を使用して、プロセス名を指定することもできます。/IDENTIFICATION 修飾子とともにプロセス名パラメータを使用すると、修飾子がパラメータを上書きします。プロセス名パラメータも/IDENTIFICATION 修飾子も指定しないと、CANCEL コマンドは現在のプロセス(つまり CANCEL コマンドを発行するプロセス)に対してスケジューリングされているウェイクアップ要求を取り消します。

修飾子

/IDENTIFICATION=pid

プロセス識別番号 (PID) でプロセスを識別します。PID を指定する時、先行の 0 は省略できます。

例

1. \$ CANCEL CALENDAR

この例で CANCEL コマンドは、プロセス名が CALENDAR であるプロセスに対するウェイクアップ要求を取り消します。STOP コマンドで削除されるまで、CALENDAR プロセスはハイバネートします。

2. \$ RUN/SCHEDULE=14:00 STATUS
%RUN-S-PROC_ID, identification of created process is 0013012A

・
・
・

\$ CANCEL/IDENTIFICATION=13012A

この例で RUN コマンドは、イメージ STATUS を実行するプロセスを作成します。このプロセスはハイバネートし、14:00 に実行を開始するようスケジューリングされています。このプロセスが STATUS を実行する前に、CANCEL コマンドはウェイクアップ要求を取り消しています。


```
3. $ RUN/PROCESS_NAME=LIBRA/INTERVAL=1:00    LIBRA
    %RUN-S-PROC_ID, identification of created process is 00130027
    .
    .
    .
    $ CANCEL LIBRA
    $ STOP LIBRA
```

この例の RUN コマンドは、LIBRA という名前のサブプロセスを作成して、イメージ LIBRA.EXE を 1 時間に 1 回実行します。

次に CANCEL コマンドでウェイクアップ要求を取り消します。このプロセスは存在しつづけますが、STOP コマンドで削除するまで、ハイバネートします。

CHECKSUM

CHECKSUM コマンドは、OpenVMS ファイルに対するチェックサムを計算するためのユーティリティを実行します。チェックサムの結果は、DCL シンボル CHECKSUM\$CHECKSUM で参照できます。

フォーマット

CHECKSUM ファイル指定

パラメータ

ファイル指定

チェックサムを計算するファイルの名前を指定します。ファイルの指定では、アスタリスク(*)とパーセント記号(%)のワイルドカードが使用できます。

説明

CHECKSUM ユーティリティは、OpenVMS のファイルに対して、ファイル、イメージ、オブジェクトのチェックサムを計算します。ファイル・チェックサムの場合は、使用するアルゴリズムによって、ファイルの内部レコード構造に従うかどうかが決まります。イメージまたはオブジェクトのチェックサムの場合は、ユーティリティは常にイメージまたはオブジェクトの構造に従います。

修飾子/FILE、/IMAGE、および/OBJECT により、どの種類のチェックサムを計算するかが決まります。修飾子ごとに省略時のファイル・タイプ(.DAT、.EXE、または.OBJ)が決まり、表示される情報量も決まります。省略時の設定である/FILE では、ファイルのレコード構造に従った XOR のファイル・チェックサムとなります。省略時のファイル・タイプは.DAT となり、SYS\$OUTPUT には情報は何も出力されません。

ファイル・チェックサムの場合、CHECKSUM がどのアルゴリズムを使用して計算を実行するか指定できます。省略時の設定では、Alpha および VAX の XOR によるレコード・ベースのアルゴリズムが使用されます。代わりに、CRC アルゴリズムまたは MD5 アルゴリズムを選択することもでき、どちらもファイル全体の内容を使用してチェックサムが計算されます。CRC アルゴリズムは、ELF-64 ファイルで使用されるアルゴリズムと同じで、PKZIP などの一般的な圧縮ツールでも使用されています(つまり、ZIP ファイル中のファイル・チェックサムは、CHECKSUM ユーティリティによって得られたチェックサムと比較することができます)。MD5 アルゴリズム

は MD5 ダイジェストであり、MD5.EXE や md5sum のような、パブリック・ドメインのツールを使用して取得できます。

イメージのチェックサムは、Alpha/VAX プラットフォームと I64 プラットフォームで異なります。オブジェクトのチェックサムは I64 プラットフォームだけで利用できます。プラットフォーム修飾子/ALPHA、/I64、または/VAX を使用して、ネイティブでないイメージまたはオブジェクトのチェックサムを計算することができます。

ELF-64 イメージとオブジェクトのチェックサムでは、CHECKSUM はすべて CRC-32 アルゴリズムを使用します。CRC は、AUTODIN II、Ethernet、FDDI CRC とも呼ばれ、VAX CRC 命令の一部として文書化されています。イメージまたはオブジェクトのチェックサムは、OpenVMS I64 のオブジェクト・ファイルとイメージ・ファイルで使用される ELF-64 のデータ構造に従います。これらのチェックサムでは、不変のデータだけが計算で使用されます。タイムスタンプやバージョンといった可変データは、計算から除外されます。これは、異なるコンパイルおよびリンク操作の結果を比較するためです。

Alpha および VAX のイメージでは、CHECKSUM は XOR アルゴリズムを使用します。イメージのチェックサムは Alpha および VAX のイメージ構造に従い、不変のデータだけが計算で使用されます。タイムスタンプなどの可変データは計算から除外されます。これは、異なるリンク操作の結果を比較するためです。Alpha システムと VAX システムでは、オブジェクトの不変データに基づいてオブジェクト・ファイルのチェックサムを計算することはできません。

修飾子

/ALGORITHM=オプション

/ALGORITHM=XOR (省略時の設定)

ファイル・チェックサムで使用するアルゴリズムを選択します。省略時の設定はレコード内のデータに対する XOR アルゴリズムで、OpenVMS Alpha システムおよび VAX システムの以前の Checksum ユーティリティで使用されていたものと同じです。オプションには以下のものがあります。

- CRC — ファイル内の全バイトに対する CRC-32 アルゴリズム (レコード構造があっても無視される)。このアルゴリズムは、AUTODIN II、Ethernet、FDDI CRC とも呼ばれる。
- MD5 — Ronald L. Rivest により公開された MD5 ダイジェスト (RFC 1321) で、ファイル内の全バイトが対象となる (レコード構造があっても無視される)。
- XOR — ファイルのレコード構造に従った、全データに対する XOR アルゴリズム。

/ALPHA

Alpha タイプのチェックサムを計算します。I64 システムで/IMAGE 修飾子とともに指定した場合にだけ有効です (つまり, I64 システム上で Alpha イメージのチェックサムを計算する場合です)。この修飾子は Alpha プラットフォームでは省略時の設定であり, VAX プラットフォームでは無効です。

/FILE (省略時の設定)

ファイル・チェックサムを計算します。

省略時の設定では, XOR アルゴリズム (/ALGORITHM=XOR) を使用してチェックサムが計算されます。/FILE 修飾子では, 省略時のファイル・タイプは.DAT となります。省略時の設定では, 符号なし 10 進数のチェックサム値が DCL シンボル CHECKSUM\$CHECKSUM に格納され, 画面には出力されません。/SHOW=DATA を指定すると, 指定した入力ファイルの完全なファイル名と, 符号なし 10 進数値のファイル・チェックサムが出力されます。

プラットフォーム修飾子/ALPHA, /I64, または/VAX は, ファイル・チェックサムの結果には影響しません。しかし, /ALPHA または/VAX を指定すると, /SHOW 修飾子は指定できません。これは, Alpha システムと VAX システムの元の Checksum コーティリティでは, これらの修飾子が使用できなかったためです。

/I64

I64 タイプのチェックサムを計算します。Alpha システム上で/IMAGE または/OBJECT とともに使用した場合にだけ有効です (つまり, Alpha システム上で I64 イメージまたはオブジェクトのチェックサムを計算する場合です)。/I64 修飾子は, I64 プラットフォームでは省略時の設定であり, VAX プラットフォームでは無効です。

/IMAGE

イメージの全バイトのチェックサムを計算します。イメージ・バイトだけをチェックサムに含めるため, イメージ構造に従って計算されます。リンクのバージョンやリンク日付といった可変データは除外されます。

I64 イメージの場合 (つまり I64 形式のファイル), CRC チェックサムが計算され, 以下の内容を含む追加情報も SYS\$OUTPUT に出力されます。

- 完全なファイル名とイメージ・セグメントに対するチェックサム
- ヘッドのチェックサムとイメージ全体のチェックサム

出力値は 16 進数表記で表示されます。DCL シンボル CHECKSUM\$CHECKSUM にも, 結果が 16 進表記で格納されます。

Alpha および VAX イメージでは, XOR チェックサムが計算され, 以下に示す追加情報が SYS\$OUTPUT に出力されます。

- 完全なファイル名とイメージ・セクションに対するチェックサム

- ヘッダのチェックサムとイメージ全体のチェックサム

チェックサム値は 16 進表記で出力されます。しかし、DCL シンボル CHECKSUM\$CHECKSUM に格納される結果は、符号なしの 10 進表記です。

注意

Alpha および VAX のイメージでは、DCL シンボル CHECKSUM\$CHECKSUM のチェックサム値が 10 進になっており、以前のチェックサム・ツールとの互換性が保たれています。

/IMAGE 修飾子を指定すると、省略時のファイル・タイプは.EXE となります。I64 イメージでは、この修飾子を指定すると、/SHOW 修飾子に対するキーワード値 HEADERS と SEGMENTS が、特に指定しなくても有効になります。

/OBJECT

すべての I64 オブジェクト・バイトの CRC チェックサムを計算します。

/OBJECT 修飾子では、ELF-64 オブジェクト構造に従って、オブジェクト・バイトだけがチェックサムの対象となります。言語プロセッサのバージョンや生成日といった可変データは除外されます。

以下の内容を含む追加情報が SYS\$OUTPUT に出力されます。

- 指定された入力ファイルの完全なファイル名
- 各オブジェクト・セクション、ヘッダ、オブジェクト全体に対するチェックサム

出力されるチェックサム値は、16 進表記です。DCL シンボル CHECKSUM\$CHECKSUM に格納される結果も、16 進表記です。

/OBJECT 修飾子では、省略時のファイル・タイプは.OBJ となります。また、/SHOW 修飾子に対するキーワード値 HEADERS と SECTIONS が、特に指定しなくても有効になります。

この修飾子は、VAX プラットフォームでは無効です。Alpha プラットフォームでは、/I64 修飾子と同時に使用した場合にだけ有効です。

/OUTPUT[=ファイル指定]

/NOOUTPUT

/OUTPUT 修飾子は、コマンドの出力をどこに送るかを制御します。/NOOUTPUT 修飾子を指定すると、出力は行われません。

/OUTPUT およびファイル指定 (/OUTPUT=ファイル指定) を指定すると、出力は現在の出力装置 SYS\$OUTPUT ではなく、指定されたファイルに書き込まれます。この修飾子を指定しないか、ファイルを指定せずに /OUTPUT 修飾子を指定した場合は、出力は SYS\$OUTPUT に送られます。

/OUTPUT 修飾子を使用しても、結果(つまり、DCL シンボル CHECKSUM\$CHECKSUM)には影響しません。

/SHOW=(オプション[,...])

どのチェックサムと追加情報を装置に出力するかを制御します。

この修飾子に対するオプションは以下のとおりです。

- ALL — 該当するオプションをすべて設定するが、以下の制限がある。
 - ファイル・チェックサムの場合、DATA キーワードだけが指定可能
 - イメージのチェックサムの場合、すべてのキーワードが指定可能
 - オブジェクトのチェックサムでは、SEGMENT キーワードは指定不可
- DATA — 完全なファイル名とファイル・チェックサムを出力する。互換性のため、このオプションは/FILE で使用可能。
- EXCLUDED — イメージまたはオブジェクトのチェックサムから除外されたデータをフォーマットする。
- HEADERS — すべての I64 ヘッダのチェックサムを出力する。このオプションは、/IMAGE と/OBJECT では省略時の設定である。
- SECTIONS — すべての ELF-64 セクションのチェックサムを出力する。このオプションは、/OBJECT では省略時の設定である。
- SEGMENTS — すべての ELF-64 プログラム・セグメントのチェックサムを出力する。このオプションは、/IMAGE では省略時の設定である。

/SHOW 修飾子は、VAX プラットフォームでは無効です。

/VAX

VAX タイプのチェックサムを計算します。I64 または Alpha システムで、/IMAGE を使用し、非 VAX システム上での VAX イメージのチェックサムを計算する場合にだけ有用です。この修飾子は、VAX プラットフォームでは無効です。

例

1.

CHECKSUM/IMAGE コマンドの出力は、I64 プラットフォームと Alpha プラットフォームで異なる結果となります。イメージの構造が違いため、チェックサムに対する名前が変わります。

- Alpha のチェックサムでは、セクション番号は BLISS 定数%D'1' として出力されるのに対し、I64 のチェックサムは 10 進数として出力される。
- Alpha のチェックサムでは、チェックサムは BLISS 定数%X'6C5404CB' として出力されるのに対し、I64 のチェックサムは DCL スタイルの 16 進数で出力される。

- Alpha の DCL シンボルは符号なしの 10 進数値であるのに対し, I64 の DCL シンボルは 16 進数値である。

Alpha システムでの例

```
2. $ CHECKSUM/IMAGE HELLO.EXE

file DISK$USER:[JOE]HELLO.EXE;10
image section %D'1' checksum is %X'6C5404CB'
image section %D'2' checksum is %X'E29D6A3A'
image section %D'3' checksum is %X'114B0786'
image header checksum is %X'00000204'
checksum of all image sections is %X'9F826977'

$ SHOW SYMBOL CHECKSUM$CHECKSUM
CHECKSUM$CHECKSUM = "2676124023"
```

I64 システムでの例

```
3. $ CHECKSUM/IMAGE FOOBAR.EXE

File DISK$USER:[JOE]FOOBAR.EXE;3
Checksum program segment 0: %X18E293D7
Checksum program segment 1: %XE29D6A3A
Checksum program segment 2: %XA6D02DD5
Checksum program segment 3: %X30130E3E
Checksum dynamic segment %X0F704080
Elf header checksum: %X7A6AC80F
Elf program header checksum: %XBF6B41D8
Elf section header checksum: %X6C770CF6
Elf (object/image) checksum: %X2EEE7726

$ SHOW SYMBOL CHECKSUM$CHECKSUM
CHECKSUM$CHECKSUM = "2EEE7726"
```

CLOSE

OPEN コマンドによりオープンされているファイルをクローズし、関連する論理名を解除します。

フォーマット

CLOSE 論理名[:]

パラメータ

論理名[:]

OPEN コマンドによりファイルがオープンされた時に、そのファイルに割り当てる論理名を指定します。

説明

コマンド・レベルでの読み込みまたは書き込みのためにオープンされているファイルは、CLOSE コマンドで終了するまで、またはそのプロセスが終了するまで、オープンされています。ファイルをオープンしたコマンド・プロシージャが、オープンしたファイルをクローズせずに終了すると、ファイルはオープンなままです。コマンド・インタプリタが自動的にそのファイルをクローズさせることはありません。

修飾子

/DISPOSITION=オプション

ファイルをクローズするときにどのような操作を実行するかを指定します。オプションは、次のとおりです。

DELETE	ファイルを削除する。
KEEP (default)	ファイルを保存する。
PRINT	ファイルをプリントする。
SUBMIT	ファイルをキューに登録する。

/ERROR=ラベル

ファイルをクローズする時にエラーが発生した場合に、制御を戻すコマンド・プロシージャ内のラベルを指定します。この修飾子に対して指定されているエラー・ルーチンは、ON コマンドに指定されている動作より優先します。エラーが発生した時に、

指定したラベルに正しく制御が渡されると、グローバル・シンボル\$STATUS にエラーの種類を示すコードが設定されます。

/LOG (省略時の設定)

/NOLOG

DCL でオープンされていないファイルをクローズしようとする、警告メッセージを生成します。/ERROR 修飾子を指定した場合には、/LOG 修飾子は無効になります。

DCL でファイルがオープンされていない場合には、エラー分岐が実行され、メッセージは表示されません。

例

```
1. $ OPEN/READ INPUT_FILE TEST.DAT
   $ READ_LOOP:
   $ READ/END_OF_FILE=NO_MORE INPUT_FILE DATA_LINE
   .
   .
   .
   $ GOTO READ_LOOP
   $ NO_MORE:
   $ CLOSE INPUT_FILE
```

OPEN コマンドは、TEST.DAT というファイルをオープンし、そのファイルに INPUT_FILE という論理名を割り当てます。READ コマンドの/END_OF_FILE 修飾子は、ファイルの最後に到達したときに、コマンド・インタプリタが NO_MORE というラベルの行に制御を渡すことを要求しています。CLOSE コマンドは、入力ファイルをクローズします。

```
2. $ @READFILE
   Ctrl/Y
   $ STOP
   $ SHOW LOGICAL/PROCESS
   .
   .
   .
   "INFILE" = "_DB1"
   "OUTFILE" = "_DB1"
   $ CLOSE INFILE
   $ CLOSE OUTFILE
```

CTRL/Y は、READFILE.COM というコマンド・プロシージャの実行に割り込みをかけます。そのあと、STOP コマンドが、プロシージャを停止します。SHOW LOGICAL/PROCESS コマンドは、プロセス論理名テーブルに現在登録されている名前を表示します。このコマンドで表示される名前には、READFILE.COM プロシージャの中で、OPEN コマンドによって割り当てられた INFILE と OUTFILE という論理名も含まれています。

CLOSE

CLOSE コマンドは、これらのファイルをクローズし、論理名の割り当てを解除します。

CONNECT

ユーザが使用している物理端末を、他のプロセスと接続している仮想端末と接続します。

ユーザは、ユーザの利用者識別コード (UIC) を持つプロセスに接続している仮想端末に接続しなければなりません。他の物理端末は、仮想端末に接続しません。

フォーマット

CONNECT 仮想端末名

パラメータ

仮想端末名

ユーザが接続している仮想端末の名前を指定します。仮想端末名は、文字 VTA で始まります。SHOW USERS コマンドを実行すると、プロセスに接続している仮想端末の名前が表示されます。

説明

サブプロセスを作成する SPAWN コマンドや、サブプロセスに接続する ATTACH コマンドとは異なり、CONNECT コマンドは別のプロセスに接続します。

通信回線を使用しているシステムにログ・インしている場合、CONNECT コマンドは便利です。回線上にノイズがありキャリア・シグナルを失った場合、ユーザのプロセスは終了しません。再度ログ・インした後に元のプロセスに再接続し、2 つ目のプロセスからログ・アウトすることができます。

CONNECT コマンドを使用するためには、OpenVMS Alpha システム上で System Manager ユーティリティ (SYSMAN)、および OpenVMS VAX システム上で System Generation ユーティリティ (SYSGEN) で、仮想端末機能が許可されていなければなりません。

利用者が使用しているシステムで仮想端末機能が許可されている場合は、SET TERMINAL/DISCONNECT/PERMANENT コマンドを使用して、特定の物理端末の仮想端末属性を許可することができます。この属性を許可すると、ある利用者がその物理端末にログ・インする時に仮想端末が作成されます。物理端末は仮想端末に接続され、仮想端末はプロセスに接続されます。

新しい仮想端末に対しては、まず TTY_DEFCHAR2 システム・パラメータの TT2\$V_DISCONNECT ビットを設定し、システムを再ブートする必要があります。ttddriver を使用して仮想装置 VTA0 を作成すると、これらの処理が行われます。次の例を参照してください。Alpha システムの場合は、次のように入力します。

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> IO CONNECT/NOADAPTER/DRIVER=SYS$LOADABLE_IMAGES:SYS$TTDRIVER VTA0:
```

VAX システムの場合は、次のように入力します。

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> CONNECT /NOADAPTER/DRIVER=TTDRIVER VTA0:
```

/NOLOGOUT 修飾子を指定していない場合は、物理端末と仮想端末との接続が切断されると、ユーザは現在のプロセスからログ・アウトしてしまいます (またこのプロセスで実行中のイメージは停止します)。

/NOLOGOUT 修飾子を指定していると、プロセスは仮想端末に接続されたままです。このプロセスでイメージを実行している場合は、プロセスが端末からの入力を必要とする、または端末に書き込みを行おうとするまで、イメージは引き続き実行されます。プロセスが端末からの入力を必要とする時、または端末に書き込みを行おうとする時は、物理端末が仮想端末に再接続するまで、プロセスは待ち状態になります。

仮想端末を使用していなくても、仮想端末に接続することができます。ただし現在のプロセスをログ・アウトするためには、CONNECT コマンドに/LOGOUT 修飾子を指定しなければなりません。仮想端末から他の仮想端末に接続する場合は、/NOLOGOUT 修飾子を指定すると、現在のプロセスも残すことができます。

修飾子

/CONTINUE

/NOCONTINUE (省略時の設定)

他のプロセスに接続する直前に、現在のプロセスで CONTINUE コマンドを実行するかどうかを制御します。この修飾子を指定すると、ユーザが他のプロセスに接続した後で、割り込みをかけられたイメージの処理を続けることができます。

/CONTINUE 修飾子を/LOGOUT 修飾子と同時に指定することはできません。

/LOGOUT (省略時の設定)

/NOLOGOUT

仮想端末を使用して他のプロセスに接続する時に、現在のプロセスをログ・アウトします。

仮想端末に接続していないプロセスから CONNECT コマンドを実行する場合は、/LOGOUT 修飾子を指定しなければなりません。/LOGOUT 修飾子を指定しないと、DCL はエラー・メッセージを表示します。

/LOGOUT 修飾子を/CONTINUE 修飾子と同時に指定することはできません。

例

1. \$ RUN AVERAGE

Ctrl/Y

\$ CONNECT/CONTINUE VTA72

RUN コマンドを使用して、イメージ AVERAGE.EXE を実行します。この RUN コマンドは、仮想端末に接続していない端末から発行しています。次に Ctrl/Y を押して、イメージに割り込みをかけます。その後 CONNECT コマンドに /CONTINUE 修飾子を指定します。これにより CONTINUE コマンドが発行され、イメージの実行が継続され、ユーザは他の仮想端末に接続します。ユーザは後で元のプロセスに再度接続することができます。

2. \$ SHOW USERS/FULL

VAX/VMS User Processes at 22-DEC-2001 14:11:56.91

Total number of users = 51, number of processes = 158

Username	Node	Process Name	PID	Terminal
KIDDER	BUKETT	KIDDER	29A0015E	FTA3:
KIDDER	BUKETT	_FTA4:	29A0015F	FTA4:
KIDDER	RACEY1	KIDDER	05800062	FTA5:
KIDDER	RACEY1	DECW\$MWM	0580005D	MBA44: Disconnected
KIDDER	RACEY1	DECW\$SESSION	05800059	
KIDDER	RACEY1	VUE\$KIDDER_2	0580005E	(subprocess of 05800059)
KIDDER	RACEY1	VUE\$KIDDER_3	0580005F	MBA51: Disconnected
KIDDER	RACEY1	VUE\$KIDDER_4	05800060	MBA53: Disconnected
SMITH	BUKETT	SMITH	29A002C1	FTA7:
SMITH	BUKETT	SMITH_1	29A006C2	(subprocess of 29A002C1)
SMITH	BUKETT	SMITH_2	29A00244	(subprocess of 29A002C1)
SMITH	HAMLET	SMITH	24800126	FTA6:
SMITH	HAMLET	DECW\$BANNER	24800155	(subprocess of 24800126)
SMITH	HAMLET	DECW\$MWM	2480011F	MBA170: Disconnected
SMITH	HAMLET	DECW\$SESSION	2480011D	FTA5:

.
.

.

\$ CONNECT VTA273

SMITH logged out at 22-DEC-2001 14:12:04.53

\$

この例は、キャリア・シグナルを失った後、元のプロセスに再度接続する方法を示しています。まず再度ログ・インして新しいプロセスを作成します。ログ・インしたら、SHOW USERS/FULL コマンドを実行して、元のプロセスの仮想端末名を確認します。次に CONNECT コマンドを実行して、元のプロセスを実行していた仮想端末に接続します。特に修飾子を指定していないので、CONNECT コマンドを発行したプロセスは、ログ・アウトします。

CONNECT

元のプロセスに再度接続したら、キャリア・シグナルを失った時に実行していたイメージの実行を続けます。この例では、接続が切断された時、ユーザ SMITH は会話型レベルでした。

CONTINUE

Ctrl/Y または Ctrl/C を押すことにより割り込まれた DCL コマンド、プログラム、コマンド・プロシージャの実行を再開します。他のイメージやコマンド・プロシージャを起動した後では、再開できません。

フォーマット

CONTINUE

説明

CONTINUE コマンドを使用すると、Ctrl/Y または Ctrl/C を押して中断させたイメージ処理、あるいはコマンド・プロシージャを再開できます。別のイメージを実行するコマンドを入力した場合、またはコマンド・プロシージャを起動した場合には、イメージの実行を再開できません。ただし、別のイメージを実行しないコマンドの後には、CONTINUE を使用できます。これらのコマンドのリストについては、『OpenVMS ユーザーズ・マニュアル』を参照してください。

CONTINUE コマンドは、「C」1文字に短縮できます。

CONTINUE コマンドは、コマンド・プロシージャで、IF コマンドまたは ON コマンドのターゲット・コマンドとして使用できます。CONTINUE コマンドは、GOTO コマンドのターゲット・ラベルの後で使用しても、ターゲット・コマンドになります。また、CONTINUE コマンドは、VAX Fortran PAUSE 文または VAX COBOL-74 STOP リテラル文を実行したプログラムの処理再開にも使用できます。

例

```
1. $ RUN MYPROGRAM_A
   Ctrl/Y
$ SHOW TIME
  14-DEC-2001 13:40:12
$ CONTINUE
```

RUN コマンドは、MYPROGRAM_A というプログラムを実行します。プログラム実行中に CTRL/Y を押すと、イメージに割り込みがかけられます。SHOW TIME コマンドで、現在の日付と時刻を表示しています。このあとの CONTINUE コマンドは、MYPROGRAM_A というイメージの実行を再開します。

CONTINUE

2. \$ ON SEVERE_ERROR THEN CONTINUE

このコマンド・プロシージャ文は、コマンドやプログラムの実行により警告状態、エラー状態、重大なエラー状態を示す値が戻された場合でも、プロシージャの実行を継続することをコマンド・インタプリタに指示します。この ON 文は、エラーまたは重大なエラーが発生した際にはプロシージャを終了するという省略時の動作を無効にします。

CONVERT

Convert ユーティリティを起動します。このユーティリティはあるファイルのレコードを別のファイルにコピーし、入力ファイルの編成や形式を出力ファイルの編成や形式に変更します。

Convert ユーティリティについての詳細は、『OpenVMS Record Management Utilities Reference Manual』あるいはオンライン・ヘルプを参照してください。

フォーマット

CONVERT 入力ファイル指定[,...] 出力ファイル指定

CONVERT/DOCUMENT

CDA にサポートされている変更可能な入力ファイルを，変更可能なまたは最終的な形式の出力ファイルに変換します。

注意

DECwindows Motif for OpenVMS がシステムにインストールされている時のみ，このコマンドを使用することができます。

フォーマット

CONVERT/DOCUMENT 入力ファイル指定 出力ファイル指定

パラメータ

入力ファイル指定

変換する入力ファイル名を指定します。省略時のファイル・タイプは.DDIF です。

出力ファイル指定

出力ファイル名を指定します。省略時のファイル・タイプは.DDIF です。

説明

CONVERT/DOCUMENT コマンドを使用すると，文書の形式を変換することができます。入力ファイル(そのファイルを読むためのアプリケーションと互換性のない形式のファイル)と出力ファイル(別の形式で作成されるファイル)の，名前と形式を指定します。

入力ファイル形式のための入力コンバータが存在し，出力ファイル形式のための出力コンバータが存在している場合は，入力ファイルのファイル形式を別のファイル形式に変換することができます。省略時の設定の入力ファイルおよび出力ファイルの形式は，DDIF (DIGITAL Document Interchange Format) です。DDIF は，テキスト，グラフィック，イメージなどを含めることができる複合文書の格納および変換のための標準形式です。

他のいくつかのコンバータと同様に DDIF 入力コンバータおよび DDIF 出力コンバータは，CDA Base Services for DECwindows Motif for OpenVMS といっしょにインストールされます。入力ファイル形式を他の出力ファイル形式に変換するとき，変更を最小限に抑えるようなオプションを指定できるコンバータもあります。

CONVERT/DOCUMENT コマンドに/OPTIONS 修飾子を指定する前に、必要なオプションを書いたオプション・ファイルを作成しておきます。

すべてのコンバータは、メッセージ・ログ・オプションをサポートしています。変換中の情報メッセージおよびエラー・メッセージは、ユーザが指定したファイルに書き込まれます。

修飾子

/FORMAT=形式名

入力ファイルまたは出力ファイルをコード化する形式を指定します。省略時の入力ファイルおよび出力ファイルの形式は DDIF です。

CDA Base Services for DECwindows Motif for OpenVMS とともに提供される入力コンバータ、およびサポートされるファイル形式の省略時のファイル・タイプを、次に示します。

入力形式	ファイル・タイプ
DDIF	.DDIF
DTIF	.DTIF
TEXT	.TXT

CDA Base Services for DECwindows Motif for OpenVMS とともに提供される出力コンバータ、およびサポートされるファイル形式の省略時のファイル・タイプを、次に示します。

出力形式	ファイル・タイプ
DDIF	.DDIF
DTIF	.DTIF
TEXT	.TXT
PS	.PS
ANALYSIS	.CDA\$ANALYSIS

弊社が提供する CDA Converter Library は、いくつかの文書、グラフィック、イメージ、データ・テーブル入力コンバータ、およびデータ・テーブル出力コンバータを提供しています。ソフトウェア・ベンダも、CDA に準拠したアプリケーションやコンバータを提供しています。システムで利用できるコンバータについては、システム管理者に相談してください。

分析出力コンバータ

分析出力コンバータは、入力ファイルの中間表現の分析を行います。分析出力ファイルには、指定したオブジェクトと入力ファイルに格納された値が含まれます。アプリケーション・プログラマは、デバッグのために分析出力ファイルを使用することができます。

アプリケーションのエンド・ユーザは分析出力ファイルを使用して、複数のサブファイルへの参照やリンクを入力ファイルに含めるかどうか、決めることができます。各サブファイルは、ネットワークを介して別々にコピーされなければなりません。これは、入力ファイルをネットワークを介して転送しても、サブファイルは自動的に転送されないためです。

分析出力ファイルで、文字列“ERF_”という文字列を検索することができます。入力ファイルである DDIF 複合文書とリンクしたイメージ・ファイル“griffin.img”で、文字列“ERF_”を検索した例を次に示します。

```
ERF_LABEL ISO LATIN1 "griffin.img" ! Char. string.
ERF_LABEL TYPE RMS_LABEL TYPE "$RMS:
ERF_CONTROL COPY_REFERENCE ! Integer = 1
```

分析出力ファイルは、プログラマが使用することを仮定しています。入力ファイル中のコード化された情報を変更することは前提としておらず、ファイルの内容を調べることを前提としています。上記の例では、分析出力ファイル中の、リンクされたファイルへの参照を検索する方法を示しています。

DDIF 入力コンバータ

DDIF 入力コンバータは、DDIF 入力ファイルを、指定した出力ファイル形式に引き続き変換される中間表現に変換します。データのマッピング、変換に関する制限事項、外部ファイル参照、および DDIF 入力コンバータに関連した文書構文エラーについては以下を参照してください。

- データのマッピング

DDIF 入力ファイルの情報を、中間表現に直接マップします。

- 変換に関する制限事項

中間表現に変換する時、DDIF 入力ファイルの情報が消失することはありません。

ただし DDIF 入力ファイルで、DDIF 入力コンバータが理解するバージョンより新しいバージョンの DDIF 文法を使用している場合は、新文法で示される要素は消失します。

- 外部ファイル参照

DDIF 入力ファイル中の外部ファイル参照はすべて、中間表現に変換されます。

出力コンバータが要求すればコンバータ・カーネルは外部参照を解決できますが、DDIF 入力コンバータは外部参照を解決を試みません。

- 文書構文エラー

DDIF 入力ファイルに文書構文エラーがあると、回復不可能な入力処理エラーが発生します。DDIF 入力コンバータが文書構文エラーを検出すると、変換処理は停止され、それ以降の入力処理は行われません。

DDIF 出力コンバータ

DDIF 出力コンバータは、入力ファイルの中間表現から DDIF 出力ファイルを作成します。データのマッピング、および DDIF 出力コンバータに関連する変換の制限事項を、次に示します。

- データのマッピング

入力ファイルの中間表現中の情報を、DDIF 出力ファイルに直接マップします。

- 変換に関する制限事項

DDIF 出力ファイルに変換する時、入力ファイルの中間表現の情報が消失することはありません。

DTIF 入力コンバータ

DTIF 入力コンバータは、DTIF 入力ファイルを、指定した出力ファイル形式に引き続き変換される中間表現に変換します。データのマッピング、変換に関する制限事項、外部ファイル参照、および DTIF 入力コンバータに関連した文書構文エラーについては以下を参照してください。

- データのマッピング

DTIF 入力ファイルの情報を、中間表現に直接マップします。

- 変換に関する制限事項

中間表現に変換する時、DTIF 入力ファイルの情報が消失することはありません。

ただし DTIF 入力ファイルで、DTIF フロント・エンドが理解するバージョンより新しいバージョンの DTIF 文法を使用している場合は、新文法で示される要素は消失します。

- 外部ファイル参照

DTIF 入力ファイル中の外部ファイル参照はすべて、中間表現に変換されます。

DTIF 入力コンバータは、外部参照の解決を試みません。

- 文書構文エラー

DTIF 入力ファイルに文書構文エラーがあると、回復不可能な入力処理エラーが発生します。DTIF 入力コンバータが文書構文エラーを検出すると、変換処理は停止され、それ以降の入力処理は行われません。

DTIF 出力コンバータ

DTIF 出力コンバータは、入力ファイルの中間表現から DTIF 出力ファイルを作成します。データのマッピング、および DTIF 出力コンバータに関連する変換の制限事項を、次に示します。

- データのマッピング

入力ファイルの中間表現中の情報を、DTIF 出力ファイルに直接マップします。

- 変換に関する制限事項

DTIF 出力ファイルに変換する時、入力ファイルの中間表現の情報が消失することはありません。

- 外部ファイル参照

DTIF 出力コンバータは、入力ファイルの中間表現に格納されている外部ファイル参照を変換しますが、外部参照の解決は試みません。

テキスト入力コンバータ

テキスト入力コンバータは、テキスト (ISO Latin1) 入力ファイルを、指定した出力ファイル形式に引き続き変換される中間表現に変換します。データのマッピング、変換に関する制限事項、外部ファイル参照、テキスト入力コンバータに関連した文書構文エラーについては以下を参照してください。

- データのマッピング

テキスト入力ファイルの情報を、中間表現に直接マップします。ライン・ブレイクとフォーム・フィールドは、DDIF 指示文にマップされます。1 行の空白行、または連続した複数の空白行は、段落の終端を示すものと解釈されます。

キャラクタ・セル端末または端末エミュレータで、DEC MCS ファイルとしてテキスト入力ファイルを入力すると、次のように変換されます。

変換前の文字	変換後の文字
Concurrency sign	分音符
大文字の O と E の合字	乗算記号
分音符の付いた大文字の Y	鋭アクセントの付いた大文字の Y
小文字の o と e の合字	除算記号
分音符の付いた小文字の y	鋭アクセントの付いた Y

- 変換に関する制限事項

中間表現に変換する時、テキスト入力ファイルの情報が消失することはありません。これは、テキスト・ファイルには構造情報が入っていないからです。

印刷不可能な文字はすべて、空白文字に変換されます。たとえば ANSI エスケープ文字は、空白文字に変換され解釈を試みません。

- 外部ファイル参照

テキスト・ファイルには、外部ファイル参照は含まれていません。

- 文書構文エラー

テキスト・ファイルには構文がないため、テキスト入力コンバータにより構文エラーが報告されることはありません。

テキスト出力コンバータ

テキスト出力コンバータは、入力ファイルの中間表現からテキスト出力ファイルを作成します。データのマッピング、およびテキスト出力コンバータに関連する変換の制限事項を、次に示します。

- データのマッピング

入力ファイルの中間表現の中のすべての Latin1 テキストは、テキスト出力ファイルに変換されます。

入力ファイルをテキスト出力ファイルに変換する場合は、テキスト出力ファイルにはテキスト、およびライン・フィード、ページ・ブレイク、タブなどの最小限のフォーマットしか入れられないことに注意してください。テキスト出力コンバータは、(出力ファイルがさらに変換される可能性を考慮して) フォーマット情報を保持します。ページングは最も近いキャラクタ・セル(行、カラム)位置に変換されます。

- 変換に関する制限事項

テキスト出力ファイルに変換される時に、入力ファイルの中間表現の中のすべてのグラフィック、イメージ、およびテキスト属性は消失します。

モノスペース・フォントが使用されているので、レイアウトを保持するために上書きが行われ、テキストが消失する可能性があります。文書のフォーマット情報に指定したページ幅よりも、指定したページ幅が狭い場合は、行が切り捨てられる可能性もあります。OVERRIDE_FORMAT 処理オプションを使用すれば、このようなことは起こりません。それは、OVERRIDE_FORMAT 処理オプションを指定すると、文書のフォーマット情報は無視されるからです。

PostScript 出力コンバータ

PostScript 出力コンバータは、入力ファイルの中間表現から PostScript 出力ファイルを作成します。データのマッピング、および PostScript 出力コンバータに関連する変換の制限事項を、次に示します。

- データのマッピング

入力ファイルの中間表現中の情報を、PostScript 出力ファイルに直接マップします。

- 変換に関する制限事項

PostScript 出力ファイルに変換する時、入力ファイルの中間表現の情報が消失することはありません。

/MESSAGE_FILE=ファイル指定
/NOMESSAGE_FILE (省略時の設定)

文書変換に関するメッセージを記録します。入力コンバータおよび出力コンバータからのメッセージは、ファイル指定に指定したファイルに出力されます。ファイル指定を行わないと、メッセージは SYS\$ERROR に出力されます。省略時の設定は /NOMESSAGE_FILE です。

/OPTIONS=オプション・ファイル名

変換時に、入力ファイルと出力ファイルに適用させたい処理オプションを含むテキスト・ファイルを指定します。オプション・ファイルの省略時のファイル・タイプは /CDA\$OPTIONS です。

オプション・ファイルの作成

CONVERT/DOCUMENT コマンドで /OPTIONS 修飾子を指定する前に、オプション・ファイルを作成しておきます。オプション・ファイルは、オペレーティング・システム上でファイル・タイプが省略時のファイル・タイプ CDA\$OPTIONS であるテキスト・ファイルです。

オプション・ファイルには、入力ファイル形式と出力ファイル形式に適用させるすべての処理オプションを指定します。処理オプションを指定すると、入力ファイルを形式の異なる出力ファイルに変換する時に、最小限の変更に抑えることができます。

オプション・ファイルは必須ではありません。ファイルを変換する時、省略時の処理オプションは自動的に適用されます。ただし省略時以外の設定を使用する場合は、オプション・ファイルが必要です。

オプション・ファイルを作成する時は、以下のガイドラインに従ってください。

- オプション・ファイルの各行は、入力形式または出力形式に対するキーワードで始めなければなりません。そのあとに空白やタブ(複数可)、またはスラッシュ(/)を続けます。

DDIF や DTIF のように、いくつかのファイル形式では、入力コンバータと出力コンバータがあります。形式キーワードで _INPUT や _OUTPUT を指定すると、処理オプションを入力形式にのみ、または出力形式にのみ適用させるよう制限することができます。

- 同一入力形式または同一出力形式に対して複数のオプションがある場合は、各行には処理オプションを 1 つだけ指定してください。
- 処理オプションを指定するには、大文字の英字、小文字の英字、数字 (0 ~ 9)、ドル記号 (\$)、およびアンダースコア (_) を使用します。
- 処理オプションに対して指定された値の前に、1 つまたは複数の空白やタブを入れます。

オプション・ファイルのエントリ例を次に示します。

PS PAPER_HEIGHT 10

この例では形式キーワードの_OUTPUT は必須ではありません。これは、PostScript は出力形式でのみ使用可能だからです。省略時の設定では、PAPER_HIGHT の値はインチ単位で指定します。

オプション・ファイルに、特定の変換のためのコンバータに適用されないオプションを含めると、これらのオプションは無視されます。

入力形式または出力形式で無効なオプション、あるいはオプションで無効な値を指定すると、エラー・メッセージが返されます。処理オプションに適用される制限事項を、次の節に示します。

分析出力での処理オプション

分析出力コンバータは、次のオプションをサポートします。

- COMMENT DEFAULT_VALUES

省略時の値により生成された行の先頭に、コメント文字(!)を挿入します。コメント文字は、対応する集合体括弧や配列括弧の前にも挿入されます。

- COMMENT INHERITED_VALUES

継承した値により生成される行の先頭に、コメント文字(!)を挿入します。コメント文字は、対応する集合体括弧や配列括弧の前にも挿入されます。

- TRANSLATE_BYTE_STRINGS

省略時の設定を変更します。BYTE STRING 型のデータの場合、バイト文字列中のすべての文字が印刷可能な文字(16 進値で 20 ~ 7E)であれば、分析出力は 16 進数を変換した値を表示しません。TRANSLATE_BYTE_STRINGS オプションを指定すると、この機能が変更されます。

- IMAGE_DATA

省略時の設定を変更します。DDIF\$_IDU_PLANE_DATA (ビットマップのイメージ)のバイト文字列データの場合、以前は分析出力は 16 進数を変換した値と ASCII を変換した値を表示しました。これらはどちらも、多くのユーザにとって特定の値ではありませんでした。新しいバージョンでは、どちらの表示も次のコメントと置き換えられます。

```
! *** Bit-mapped data not displayed here ***
```

16 進数での表示を行う場合は、IMAGE_DATA オプションを指定します。このオプションを指定すると、ASCII に変換されません。

- INHERITANCE

属性継承を有効にして、分析が表示されます。継承属性は、出力中で“[Inherited value.]”というマークが付けられます。このオプションを指定すると、外部参照はメインの文書にインポートされます。

テキスト出力での処理オプション

テキスト出力コンバータは、次のオプションをサポートします。

- ASCII_FALLBACK [ON,OFF]

テキスト出力コンバータが、テキストを 7 ビットの ASCII で出力することを指定します。文字のフォールバック表現は、ASCII 標準に記述されています。このオプションを指定しないと、省略時の設定により OFF が使用されます。値を指定せずにこのオプションを指定すると、省略時の設定により ON が使用されます。

- CONTENT_MESSAGES [ON,OFF]

テキスト出力コンバータが、入力ファイルの中間表現中でテキストではない要素を検出するたびに、メッセージを出力ファイルに書き込むことを指定します。このオプションを指定しないと、省略時の設定により OFF が使用されます。値を指定せずにこのオプションを指定すると、省略時の設定により ON が使用されます。

- HEIGHT 値

テキスト出力ファイルでの、1 ページあたりの最大行数を指定します。0 を指定すると、1 ページあたりの行数は、文書で指定した行数と同じになります。OVERRIDE_FORMAT とともに指定する、または文書にページ・サイズが指定されていないと、このオプションで指定した値を使用して、フォーマットします。省略時の設定の値は 66 です。

- OVERRIDE_FORMAT [ON,OFF]

HEIGHT および WIDTH 処理オプションで指定したページのサイズにテキストをフォーマットできるように、文書に含まれているフォーマット情報を無視するかどうかを指定します。このオプションを指定しないと、省略時の設定により OFF が使用されます。値を指定せずにこのオプションを指定すると、省略時の設定により ON が使用されます。

- SOFT_DIRECTIVES [ON,OFF]

テキスト出力ファイルを作成する時に、文書に含まれているソフト指示文に従うかどうかを指定します。このオプションを指定しないと、省略時の設定により OFF が使用されます。値を指定せずにこのオプションを指定すると、省略時の設定により ON が使用されます。

- WIDTH 値

テキスト出力ファイルでの、1 ページあたりの最大カラム数を指定します。0 を指定すると、1 ページあたりのカラム数は、文書で指定したカラム数と同じになります。OVERRIDE_FORMAT とともに指定する、または文書にページ・サイズが指定されていないと、このオプションで指定した値を使用してフォーマットします。ここで指定した値を越えるテキストは、切り捨てられます。省略時の値は 80 です。

PostScript 出力コンバータでの処理オプション

PostScript 出力コンバータは、次のオプションをサポートします。

- PAPER_SIZE サイズ

PostScript 出力ファイルをフォーマットする時に使用される、用紙のサイズを指定します。サイズ引数で有効な値を次に示します。

キーワード	サイズ
A0	841 x 1189 ミリ (33.13 x 46.85 インチ)
A1	594 x 841 ミリ (23.40 x 33.13 インチ)
A2	420 x 594 ミリ (16.55 x 23.40 インチ)
A3	297 x 420 ミリ (11.70 x 16.55 インチ)
A4	210 x 297 ミリ (8.27 x 11.70 インチ)
A	8.5 x 11 インチ (216 x 279 ミリメートル)
B	11 x 17 インチ (279 x 432 ミリメートル)
C	17 x 22 インチ (432 x 559 ミリメートル)
D	22 x 34 インチ (559 x 864 ミリメートル)
E	34 x 44 インチ (864 x 1118 ミリメートル)
LEDGER	11 x 17 インチ (279 x 432 ミリメートル)
LEGAL	8.5 x 14 インチ (216 x 356 ミリメートル)
LETTER	8.5 x 11 インチ (216 x 279 ミリメートル)
LP	13.7 x 11 インチ (348 x 279 ミリメートル)
VT	8 x 5 インチ (203 x 127 ミリメートル)

省略時の設定の用紙サイズは A(8.5 x 11 インチ) です。

- PAPER_HEIGHT 高さ

あらかじめ定義されている値以外の、用紙のサイズを指定します。省略時の設定の用紙の高さは 11 インチです。

- PAPER_WIDTH 幅

あらかじめ定義されている値以外の、用紙のサイズを指定します。省略時の設定の用紙の幅は 8.5 インチです。

- PAPER_TOP_MARGIN 上マージン

用紙の上部のマージン幅を指定します。省略時の設定は 0.25 インチです。

- PAPER_BOTTOM_MARGIN 下マージン

用紙の下部のマージン幅を指定します。省略時の設定は 0.25 インチです。

- PAPER_LEFT_MARGIN 左マージン

用紙の左のマージン幅を指定します。省略時の設定は 0.25 インチです。

- PAPER_RIGHT_MARGIN 右マージン

用紙の右のマージン幅を指定します。省略時の設定は 0.25 インチです。

- PAPER_ORIENTATION 方向

出力された PostScript ファイルで使用する用紙の方向を指定します。方向引数で有効な値を次に示します。

キーワード	説明
PORTRAIT	縦向き。用紙の長い方の辺が、垂直軸と並行になる向き。
LANDSCAPE	横向き。用紙の長い方の辺が、水平軸と並行になる向き。

省略時の設定は PORTRAIT です。

- EIGHT_BIT_OUTPUT [ON,OFF]

PostScript 出力コンバータに、8 ビットで出力するかどうかを指定します。省略時の値は ON です。

- LAYOUT [ON,OFF]

DDIF 文書で指定されたレイアウトで処理するかどうかを指定します。省略時の値は ON です。

- OUTPUT_BUFFER_SIZE 出力バッファ・サイズ

出力バッファのサイズを指定します。64 から 256 までの値を指定しなければなりません。省略時の値は 132 です。

- PAGE_WRAP [ON,OFF]

下マージンを越えたテキストを、改ページするかどうかを指定します。省略時の値は ON です。

- SOFT_DIRECTIVES [ON,OFF]

出力を編集するために、DDIF ファイルのソフト指示文を処理するかどうかを指定します (ソフト指示文は改行、改ページ、タブなどの編集コマンドを指定します)。PostScript 出力コンバータがソフト指示文を処理すると、省略時の値は ON です。

- WORD_WRAP [ON,OFF]

右マージンを越えるテキストのラッピングを行うかどうかを指定します。省略時の値は ON です。OFF を指定すると、テキストは右マージンを越えてしまいます。

ドメイン・コンバータ

入力コンバータのある、CDA をサポートする表ファイル形式に適用させる処理オプションを含む、オプション・ファイルを作成できます。表ファイル形式の例としては、データ・テーブルやスプレッドシートがあります。

表入力ファイルを文書出力ファイルに変換するには、DTIF_TO_DDIF 形式名を使用し、次に処理オプションを指定します。特定の表入力ファイル形式や文書出力ファイル形式への処理オプションに加えて、DTIF_TO_DDIF 処理オプションを指定します。

レポートや他の文書に表のテキスト表現を入れられるように、表入力ファイルを文書出力ファイルに変換したい場合もあります。ただし表入力ファイルを文書出力ファイルに変換すると、セル・ボーダ、ヘッダ、グリッド行、すべての式、およびフォント・タイプが消失することに注意してください。

ドメイン・コンバータは、次のオプションをサポートします。

- COLUMN_TITLE
カラム属性のカラム・タイトルを、カラムの上中央に表示します。
- CURRENT_DATE
現在の日付と時刻を、ページの左下の隅に表示します。これらは、文書に省略時で設定されている形式にしたがって表示されます。
- DOCUMENT_DATE
文書ヘッダにある文書の日付と時刻を、ページの左上の隅に表示します。これらは、文書に省略時で設定されている形式にしたがって表示されます。
- DOCUMENT_TITLE
文書ヘッダの文書タイトルを、各ページの上中央に表示します。
- PAGE_NUMBER
現在のページ番号を、ページの右上の隅に表示します。
- PAPER_SIZE サイズ
PostScript 出力ファイルをフォーマットする時に使用する、用紙のサイズを指定します。サイズ引数で有効な値を次に示します。

キーワード	サイズ
A0	841 x 1189 ミリ (33.13 x 46.85 インチ)
A1	594 x 841 ミリ (23.40 x 33.13 インチ)
A2	420 x 594 ミリ (16.55 x 23.40 インチ)
A3	297 x 420 ミリ (11.70 x 16.55 インチ)
A4	210 x 297 ミリ (8.27 x 11.70 インチ)
A5	148 x 210 ミリ (5.83 x 8.27 インチ)
A	8.5 x 11 インチ (216 x 279 ミリ)
B	11 x 17 インチ (279 x 432 ミリ)
B4	250 x 353 ミリ (9.84 x 13.90 インチ)
B5	176 x 250 ミリ (6.93 x 9.84 インチ)
C	17 x 22 インチ (432 x 559 ミリ)
C4	229 x 324 ミリ (9.01 x 12.76 インチ)
C5	162 x 229 ミリ (6.38 x 9.02 インチ)
D	22 x 34 インチ (559 x 864 ミリ)
DL	110 x 220 ミリ (4.33 x 8.66 インチ)

キーワード	サイズ
E	34 x 44 インチ (864 x 1118 ミリ)
10x13_ENVELOPE	13 x 254 ミリ (15600 x 10 インチ)
9x12_ENVELOPE	12 x 229 ミリ (14400 x 9 インチ)
BUSINESS_ENVELOPE	9.5 x 105 ミリ (11400 x 4.13 インチ)
EXECUTIVE	10 x 191 ミリ (12000 x 7.5 インチ)
LEDGER	11 x 17 インチ (279 x 432 ミリ)
LEGAL	8.5 x 14 インチ (216 x 356 ミリ)
LETTER	8.5 x 11 インチ (216 x 279 ミリ)
LP	13.7 x 11 インチ (348 x 279 ミリ)
VT	8 x 5 インチ (203 x 127 ミリ)

用紙サイズ A(8.5 x 11 インチ) が省略時の設定です。

- PAPER_HEIGHT 高さ

あらかじめ定義されている値以外の、用紙のサイズを指定します。省略時の設定の用紙の高さは 11 インチです。

- PAPER_WIDTH 幅

あらかじめ定義されている値以外の、用紙のサイズを指定します。省略時の設定の用紙の幅は 8.5 インチです。

- PAPER_TOP_MARGIN 上マージン

用紙の上部のマージン幅を指定します。省略時の設定は 0.25 インチです。

- PAPER_BOTTOM_MARGIN 下マージン

用紙の下部のマージン幅を指定します。省略時の設定は 0.25 インチです。

- PAPER_LEFT_MARGIN 左マージン

用紙の左のマージン幅を指定します。省略時の設定は 0.25 インチです。

- PAPER_RIGHT_MARGIN 右マージン

用紙の右のマージン幅を指定します。省略時の設定は 0.25 インチです。

- PAPER_ORIENTATION 方向

出力された PostScript ファイルで使用される用紙の方向を指定します。方向引数で有効な値を次に示します。

キーワード	説明
PORTRAIT	縦向き。用紙の長い方の辺が、垂直軸と並行になる向き。
LANDSCAPE	横向き。用紙の長い方の辺が、水平軸と並行になる向き。

省略時の設定は PORTRAIT です。

例

1. \$ CONVERT/DOCUMENT/OPTIONS=MY_OPTIONS.CDA\$OPTIONS -
_MY_INPUT.DTIF/FORMAT=DTIF MY_OUTPUT.DDIF/FORMAT=DDIF

このコマンドは MY_INPUT.DTIF という名前で DTIF 形式の入力ファイルを , MY_OUTPUT.DDIF という名前で DDIF 形式の出力ファイルに変換します。オプション・ファイル MY_OPTIONS.CDA\$OPTIONS も指定しています。

CONVERT/RECLAIM

Convert/Reclaim ユーティリティを起動します。このユーティリティは、新しいレコードを Prolog 3 索引ファイルに書き込めるように、Prolog 3 索引ファイルの空バケットを使用できるようにします。/RECLAIM 修飾子は必須です。

Convert/Reclaim ユーティリティについての詳細は、『OpenVMS Record Management Utilities Reference Manual』あるいはオンライン・ヘルプを参照してください。

フォーマット

CONVERT/RECLAIM ファイル指定

COPY

1 つまたは複数の既存のファイルから新しいファイルを作成します。COPY コマンドによって、次の操作を実行できます。

- 1 つの入力ファイルを 1 つの出力ファイルにコピーする。
- 複数の入力ファイルを 1 つのファイルに連結する。
- 複数の入力ファイルを複数の出力ファイルにコピーする。

フォーマット

COPY 入力ファイル指定[,...] 出力ファイル指定

パラメータ

入力ファイル指定[,...]

コピーする、1 つまたは複数の既存ファイルの名前を指定します。ワイルドカード文字 (*と%) を使用することができます。装置やディレクトリを指定しない場合には、現在の装置とディレクトリが使用されます。複数の入力ファイルを指定する場合には、各ファイル指定をコンマ (,) またはプラス記号 (+) で区切らなければなりません。

出力ファイル指定

入力ファイルがコピーされる、出力ファイルの名前を指定します。

出力ファイル指定には、少なくとも 1 つのファイル指定フィールドを指定しなければなりません。装置またはディレクトリを指定していない場合には、現在の省略時の装置とディレクトリが使用されます。他のフィールド (ファイル名、ファイル・タイプ、バージョン番号) を省略した場合には、COPY コマンドは入力ファイルの対応するフィールドの値を使用します。複数の入力ファイルが指定されている場合には、一般に、最初の入力ファイルの各フィールドを使用します。

ファイル名、ファイル・タイプ、またはバージョン番号の代わりに、アスタリスク・ワイルドカード文字 (*) を使用することができます。COPY コマンドでは、出力ファイルの名前を決定するために、その際に参照される入力ファイルの対応するフィールドが使用されます。

説明

COPY コマンドは、1 つまたは複数の既存ファイルから、新しいファイルを作成します。装置とディレクトリを指定しない場合は、現在の省略時の装置とディレクトリが使用されます。COPY コマンドには、次の機能があります。

- 入力ファイルを出力ファイルにコピーする。
- 2 つ以上の入力ファイルを 1 つの出力ファイルに連結する。
- 入力ファイルのグループを出力ファイルのグループにコピーする。

COPY コマンドは、省略時の設定では、1 つの出力ファイルを作成します。複数の入力ファイルを指定した場合には、最初入力ファイルが出力ファイルにコピーされ、2 番目以降の入力ファイルが出力ファイルの最後に追加されます。出力ファイル指定フィールドを省略したり、フィールドにアスタリスク(*)ワイルドカード文字を使用すると、最初または唯一の入力ファイルの対応するフィールドを使用して出力ファイルに名前が付けられます。

最大レコード長を持つ複数の入力ファイルを指定すると、出力ファイルには最初の入力ファイルの最大レコード長が与えられます。以降の入力ファイルに出力ファイルの最大レコード長より長いレコードがあると、COPY コマンドは矛盾したファイル属性を示すメッセージを発行して、次のファイルのコピーを開始します。

複数の出力ファイルを作成するには、以下の少なくとも 1 つを使用して複数の入力ファイルを指定します。

- 出力ディレクトリ指定、ファイル名、ファイル・タイプ、またはバージョン番号フィールドで、アスタリスク(*)ワイルドカード文字を使用する。
- 出力ファイル指定として、ノード名、装置名、またはディレクトリ指定だけを使用する。
- /NOCONCATENATE 修飾子

COPY コマンドで複数の出力ファイルを作成する場合には、出力ファイル名には各入力ファイルから対応するフィールドが使用されます。また、出力ファイル指定でアスタリスク(*)ワイルドカード文字を使用すると、複数の出力ファイルを作成できます。次に例を示します。

```
$ COPY A.A;1, B.B;1 *.C
```

この COPY コマンドは、現在の省略時ディレクトリに、A.C;1 および B.C;1 というファイルを作成します。複数の入力ファイルと出力ファイルを指定する場合には、/LOG 修飾子を使用すると、ファイルが正しくコピーされたことを確認できます。

DECwindows 複合ドキュメントに COPY コマンドを使用する場合には、特に注意してください。詳細は『Guide to OpenVMS File Applications』を参照してください。

バージョン番号

入力ファイルと出力ファイルのバージョン番号を指定しない場合、(省略時の設定により) 出力ファイルには次のいずれかのバージョン番号が付けられます。

- 入力ファイルのバージョン番号
- 同じファイル名とファイル・タイプを持つ既存ファイルの最大バージョン番号に 1 を足したバージョン番号

アスタリスク(*)ワイルドカード文字で出力ファイルのバージョン番号を指定すると、出力ファイルのバージョン番号として、対応する入力ファイルのバージョン番号が使用されます。

出力ファイルのバージョン番号を明示的に指定すると、出力ファイル指定にはその番号が使用されます。指定したバージョン番号より大きいバージョンの出力ファイルが存在する場合は、警告メッセージが表示されファイルがコピーされます。同じバージョンの出力ファイルが存在する場合は、メッセージが表示され、入力ファイルはコピーされません。

ファイル保護と作成日/更新日

出力ファイル名のいずれかの部分を明示的に指定した場合、COPY コマンドでは出力ファイルを新しいファイルと見なします。新しいファイルの作成日は、現在の時刻と日付にセットされます。

1 つまたは複数のアスタリスク(*)とパーセント記号(%)ワイルドカード文字で出力ファイルを指定すると、入力ファイルの作成日が使用されます。

COPY コマンドでは、常に出力ファイルの更新日が現在の時刻と日付に設定されます。バックアップ日は、0 に設定されます。ファイル・システムによって、出力ファイルに新しい満了日が割り当てられます(保持が許可されていると満了日が設定され、許可されていないと 0 に設定されます)。

出力ファイルの保護とアクセス制御リスト(ACL)は、次のパラメータによって次の順序で決定されます。

- 出力ファイルの既存バージョンの保護
- 出力ディレクトリの省略時の保護と ACL
- プロセスの省略時のファイル保護

(BACKUP コマンドは、入力ファイルの作成日、更新日、ファイル保護を使用することに注意してください。)

出力ファイルの保護を変更するには、/PROTECTION 修飾子を使用します。

通常、出力ファイルの所有者は、出力ファイルの作成者と同じです。ただし、拡張特権を持つ利用者が出力ファイルを作成する場合、所有者は、親ディレクトリまたは、存在する場合は出力ファイルの前バージョンの所有者になります。

拡張特権は、次のいずれかです。

- SYSPRV (システム特権) または BYPASS
- システム利用者識別コード (UIC)
- 親ディレクトリ (または出力ファイルの前バージョン) の所有者が、新しい出力ファイルと同じグループに属する場合は、GRPPRV (グループ特権)
- 親ディレクトリ (または出力ファイルの前バージョン) の所有者を表す識別子 (リソース属性を伴う)

ディレクトリ・ファイルのコピー

ディレクトリ・ファイルをコピーすると、指定したディレクトリ名の新しい空のディレクトリが作成されます。指定したディレクトリのファイルは、新しいディレクトリにコピーされません。ディレクトリ・ファイルのコピーの例については、「例」を参照してください。

修飾子

/ALLOCATION=ブロック数

出力ファイルの初期占有サイズを、1 ブロック 512 バイトとして、n で指定されたブロック数に設定します。/ALLOCATION 修飾子を指定しない場合や、ブロック数を指定しない場合には、出力ファイルの初期占有サイズは、コピーされる入力ファイルのサイズによって決定されます。

/BACKUP

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、最新のバックアップの日時をもとにファイルを選択します。この修飾子とは他の時刻属性を指定する修飾子、/CREATED、/EXPIRED、および/MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/BEFORE[=時刻]

指定された時刻以前の時刻属性をもつファイルを選択します。絶対時刻、または絶対時刻とデルタ時間の組み合わせを指定します。また、BOOT、LOGIN、TODAY(省略時の設定)、TOMORROW、および YESTERDAY というキーワードも指定できます。適用する時刻属性は、/BACKUP、/CREATED(省略時の設定)、/EXPIRED、または/MODIFIED 修飾子のいずれかで指定します。

時刻指定の詳細は、『OpenVMS ユーザーズ・マニュアル』、またはオンライン・ヘルプのトピック Date を参照してください。

/BLOCK_SIZE=n

COPY が使用する省略時のブロック・サイズ (124) を指定変更します。1 ~ 127 の範囲の値を指定することができます。

/BY_OWNER[=uic]

ファイルの利用者識別コード (UIC) が指定した所有者 UIC と一致するファイルを選択します。/BY_OWNER 修飾子だけを指定し UIC を省略した場合には、現在のプロセスの UIC であると解釈されます。

UIC を指定する場合には、『OpenVMS システム・セキュリティ・ガイド』に説明されている標準的な UIC 形式を使用します。

/CONCATENATE (省略時の設定)

/NOCONCATENATE

出力ファイルのフィールドにワイルドカード文字が使用されていない時に、すべての入力ファイルから 1 つの出力ファイルを作成するかどうかを制御します。

/NOCONCATENATE 修飾子を指定すると、複数の出力ファイルを作成します。入力ファイル指定にワイルドカード文字を指定した場合には、すべての入力ファイルを連結した、1 つの出力ファイルが作成されます。

Files-11 ディスク上構造レベル 2 ディスク、およびレベル 5 ディスクからファイルを連結する場合には、COPY コマンドは、アルファベット順にファイルを連結します。ファイル・バージョン・フィールドにワイルドカード文字を指定した場合には、ファイルはバージョン番号の大きい順にコピーされます。Files-11 ディスク上構造レベル 1 ディスクからのファイルを連結する場合には、ランダムな順序で連結されます。

/CONFIRM

/NOCONFIRM (省略時の設定)

そのファイルに対する COPY 操作の実行を確認するために、各 COPY 操作の前に問い合わせを行います。システムがプロンプトを表示したあと、次のいずれかの応答を入力します。

YES	NO	QUIT
TRUE	FALSE	Ctrl/Z
1	0	ALL

Return

応答として単語を入力する場合には、大文字と小文字を任意に組み合わせることができます。単語の応答は、1 文字または複数の文字 (たとえば、TRUE は T, TR, または TRU) に短縮できます。肯定応答は、YES, TRUE, および 1 です。否定応答は NO, FALSE, 0, Return です。QUIT または Ctrl/Z は、その時点でコマンドの処理を停止することを示します。ALL と応答すると、コマンドはプロセスを継続しますが、そのあとプロンプトは表示されなくなります。上記のリストに含まれていない応答をタイプすると、DCL がエラー・メッセージを出力し、同じプロンプトがもう一度表示されます。

/CONTIGUOUS

/NOCONTIGUOUS

ファイルが連続する物理ディスク・ブロックを使用するかどうかを指定します。省略時の設定では、COPY コマンドは、対応する入力ファイルと同じ属性の出力ファイルを作成します。また、省略時設定では、連続したディスク・ブロックが十分でない場

合でも、エラーを報告しません。属性の異なる複数の入力ファイルをコピーする場合には、出力ファイルは連続した領域にコピーされるとは限りません。確実に連続した領域にファイルをコピーするためには、/CONTIGUOUS 修飾子を使用します。

ファイルをテープに、またはテープからコピーする場合には、/CONTIGUOUS 修飾子は無効です。これは、テープ上のファイル・サイズは、ディスクにコピーされるまで判断することができないためです。テープからファイルをコピーする時に、そのファイルを連続して領域にコピーしたい場合には、COPY コマンドを 2 度使用します。つまり、ファイルをテープからコピーするための COPY コマンドと、連続したファイルを作成するための COPY コマンドです。

/CREATED (省略時の設定)

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、作成日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/EXPIRED、および/MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/EXCLUDE=(ファイル指定[,...])

指定するファイル (1 つまたは複数) と一致するファイルを、COPY 操作から除外することを指定します。ファイル指定にディレクトリを含むことは可能ですが、装置を含むことはできません。ファイル指定にワイルドカード文字 (*と%) を使用できます。しかし特定バージョンを除外するために相対バージョン番号を使用することはできません。1 つのファイルだけを指定する場合には、括弧を省略できます。

/EXPIRED

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、満了日時をもとにファイルを選択します (満了日は、SET FILE /EXPIRATION_DATE コマンドで設定します)。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/CREATED、および/MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/EXTENSION=n

ファイルが拡張されるたびに出力ファイルに追加される、ブロック数を指定します。/EXTENSION 修飾子を指定しない場合には、出力ファイルの省略時の拡張属性は、対応する入力ファイルの拡張属性によって決定されます。

/LOG

/NOLOG (省略時の設定)

COPY コマンドが、コピーされた各ファイルのファイル情報を表示するかどうかを制御します。

/LOG 修飾子を指定すると COPY コマンドは、各コピー操作を実行するたびに次の情報を表示します:

- 入力ファイルと出力ファイルのファイル指定

- コピーされたブロック数またはレコード数（ファイルがブロック単位で、またはレコード単位でコピーされるのかにより異なる）
- 作成された新しいファイルの総数

/MODIFIED

/BEFORE または /SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、最新の変更日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/CREATED、および /EXPIRED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として /CREATED 修飾子が使用されます。

/OVERLAY

/NOOVERLAY (省略時の設定)

ファイルに新しい領域を割り当てるのではなく、入力ファイルのデータを既存の出力ファイルにコピーして、既存のデータを上書きすることを要求します。ディスク上のファイルの物理的な位置は変更されません。ただし、RMS 索引順編成ファイルと相対編成ファイルでは、出力ファイルに入力ファイルよりも少ないブロックしか割り当てられていない場合には、EMS-E-EOF エラーでコピーは失敗します。

ファイル構造を持たない装置に出力ファイルが書き込まれる場合には、/OVERLAY 修飾子は無視されます。

/PROTECTION=(所有区分[: アクセス・コード][,...])

出力ファイルに対して適用される保護を定義します。

- 所有区分パラメータは、システム (S)、所有者 (O)、グループ (G) またはワールド (W) から指定します。
- アクセス・コード・パラメータは、読み込み (R)、書込み (W)、実行 (E) または削除 (D) から指定します。

出力ファイルが存在する場合には、指定されていない保護属性は、そのファイルの現在の保護設定から適用されます。出力ファイルが存在しない場合には、現在の省略時の保護設定が使用されます。

保護コードの指定についての詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

/READ_CHECK

/NOREAD_CHECK (省略時の設定)

入力ファイルの各レコードが正しく読みとられたかどうかを確認するために、指定した入力ファイルの各レコードを、2 回ずつ読み取ることを要求します。

/REPLACE

/NOREPLACE (省略時の設定)

出力ファイルと同じファイル指定のファイルが既に存在する場合には、既存のファイルが削除されることを要求します。COPY コマンドは、出力ファイルに対して新しい領域を割り当てます。一般に、/REPLACE 修飾子を使用する場合には、ファイル指

定にバージョン番号まで指定します。省略時設定では、同じファイル指定のファイルがすでに存在する場合には、COPY コマンドはバージョン番号が 1 だけ大きな新しいバージョンのファイルを作成します。/NOREPLACE 修飾子が指定されている場合には、同じバージョン番号のファイルが存在すると、エラーとなります。

/SINCE[=時刻]

指定された時刻以降の時刻属性をもつファイルを選択します。絶対時刻、または絶対時刻とデルタ時間の組み合わせを指定します。また、BOOT、LOGIN、TODAY(省略時の設定)、TOMORROW、および YESTERDAY というキーワードも指定できます。適用する時刻属性は、/BACKUP、/CREATED(省略時の設定)、/EXPIRED、または/MODIFIED 修飾子のいずれかで指定します。

時刻指の詳細は、『OpenVMS ユーザーズ・マニュアル』、またはオンライン・ヘルプのトピック Date を参照してください。

/STYLE=キーワード

表示するファイル名の書式を指定します。

この修飾子のキーワードは CONDENSED および EXPANDED です。意味は次の表のとおりです。

キーワード	説明
CONDENSED (省略時の設定)	ファイル名を 255 文字長の文字列に適合するように表示します。このファイル名の場合、ファイル指定に DID あるいは FID 短縮形を含むことが可能です。
EXPANDED	ファイル名をディスクに格納されているとおりに表示します。このファイル名の場合、ファイル指定に DID あるいは FID 短縮形は含みません。

キーワード CONDENSED と EXPANDED を同時に指定することはできません。この修飾子は、確認が要求された場合に、出力メッセージに表示されるファイル名の書式を指定します。

EXPANDED キーワードが指定されていない場合、ファイル・エラーは CONDENSED ファイル指定で表示されます。

詳細は『OpenVMS ユーザーズ・マニュアル』を参照してください。

/TRUNCATE (省略時の設定)

/NOTRUNCATE

COPY コマンドが、コピー時に出力ファイルをエンド・オブ・ファイル (EOF) で切り捨てるかどうかを制御します。この操作は順編成ファイルにのみ使用できます。

省略時の設定では、出力ファイルのサイズは入力ファイルの実際のサイズで決定されます。/NOTRUNCATE を指定すると、出力ファイルのサイズは入力ファイルの占有サイズで決定されます。

/VOLUME=n

マルチボリューム・セットの指定された相対ボリューム番号に、出力ファイルを登録することを要求します。省略時設定では、ファイルは、マルチボリューム・セットの任意の位置に作成されます。

/WRITE_CHECK

/NOWRITE_CHECK (省略時の設定)

レコードが正しくコピーされ、そのファイルをあとで正しく読み込むことができるかどうかを確認するために、出力ファイルにレコードを書き込んだあと、出力ファイルの各レコードを読み込むことを COPY コマンドに要求します。

注意

TK50 テープ・ドライブのようなハードウェア装置では、これらのハードウェア機能の一部として、データの一貫性をチェックします。これらの装置では、/WRITE_CHECK を使用する必要はありません。どの装置が自動書き込みチェックを提供しているかについては、使用しているハードウェアのドキュメントを参照してください。

例

1. \$ COPY TEST.DAT NEWTEST.DAT

この COPY コマンドは、省略時のディスクおよびディレクトリから TEST.DAT というファイルの内容を、同じディスクおよび同じディレクトリの NEWTEST.DAT という名前のファイルにコピーします。NEWTEST.DAT という名前のファイルがすでに存在する場合には、新しいバージョンを作成します。

2. \$ COPY ALPHA.TXT TMP
\$ COPY ALPHA.TXT .TMP

最初の COPY コマンドは、ALPHA.TXT というファイルを TMP.TXT という名前のファイルにコピーします。COPY コマンドは、出力ファイルのファイル・タイプとして入力ファイルのものを使用します。2 番目の COPY コマンドは、ALPHA.TMP という名前のファイルを作成します。この COPY コマンドでは、入力ファイルの名前を出力ファイルのファイル名として使用します。

3. \$ COPY/LOG TEST.DAT NEW.DAT;1/REPLACE
%COPY-I-REPLACED, DKA0:[MAL]NEW.DAT;1 being replaced
%COPY-S-COPIED, DKA0:[MAL]TEST.DAT;1 copied to DKA0:[MAL]NEW.DAT;1 (1 block)

この例では/REPLACE 修飾子を指定して、出力ファイルの既存のバージョンと新しいファイルを置き換えるよう COPY コマンドに要求しています。COPY コマンドからの最初のメッセージは、既存のファイルを置き換えたことを示しています。出力ファイルのバージョン番号は、明示的に指定しなければなりません。明示的に指定しないと COPY コマンドは、ファイル NEW.DAT の新しいバージョンを作成します。

4. \$ COPY *.COM [MALCOLM.TESTFILES]

この例の COPY コマンドは、現在の省略時のディレクトリ内で、ファイル・タイプが.COM であるファイルの最新バージョンを、サブディレクトリ [MALCOLM.TESTFILES] にコピーします。

5. \$ COPY/LOG *.TXT *.OLD
 %COPY-S-COPIED, DKA0:[MAL]A.TXT;2 copied to DKA0:[MAL]A.OLD;2 (1 block)
 %COPY-S-COPIED, DKA0:[MAL]B.TXT;2 copied to DKA0:[MAL]B.OLD;2 (1 block)
 %COPY-S-COPIED, DKA0:[MAL]G.TXT;2 copied to DKA0:[MAL]G.OLD;2 (4 blocks)
 %COPY-S-NEWFILES, 3 files created

この例の COPY コマンドは、ファイル・タイプが.TXT であるファイルの最新バージョンを、新しいファイルにコピーします。新しいファイルのファイル名は、既存のファイルのファイル名と同じです。ただしファイル・タイプは.OLD です。COPY コマンドからの最後のメッセージは、新たに 3 つのファイルが作成されたことを示しています。

6. \$ COPY/LOG A.DAT,B.MEM C.*
 %COPY-S-COPIED, DKA0:[MAL]A.DAT;5 copied to DKA0:[MAL]C.DAT;11 (1 block)
 %COPY-S-COPIED, DKA0:[MAL]B.MEM;2 copied to DKA0:[MAL]C.MEM;24 (58 records)
 %COPY-S-NEWFILES, 2 files created

2 つの入力ファイル指定はコンマで区切られています。出力ファイル指定のアスタリスク(*)ワイルドカード文字は、2 つの出力ファイルを作成することを指定しています。各コピー操作において COPY コマンドは、入力ファイルのファイル・タイプを使用して、出力ファイルのファイル・タイプを決めます。

7. \$ COPY/LOG *.TXT TXT.SAV
 %COPY-S-COPIED, DKA0:[MAL]A.TXT;2 copied to DKA0:[MAL]TXT.SAV;1 (1 block)
 %COPY-S-APPENDED, DKA0:[MAL]B.TXT;2 appended to DKA0:[MAL]TXT.SAV;1 (3 records)
 %COPY-S-APPENDED, DKA0:[MAL]G.TXT;2 appended to DKA0:[MAL]TXT.SAV;1 (51 records)
 %COPY-S-NEWFILES, 1 file created

この例の COPY コマンドは、ファイル・タイプが.TXT であるすべてのファイルの最新バージョンを、TXT.SAV という 1 つのファイルにコピーします。最初の入力ファイルがコピーされた後、COPY コマンドは、他の入力ファイルはその出力ファイルに追加されることを示すメッセージを表示します。

この例で/NOCONCATENATE 修飾子を使用すると、COPY コマンドは各入力ファイルに対して 1 つずつ TXT.SAV を作成します。各 TXT.SAV ファイルは、バージョン番号が異なります。

8. \$ COPY MASTER.DOC DKA1:[BACKUP]

この例の COPY コマンドは、ファイル MASTER.DOC の最新バージョンを装置 DKA1 にコピーします。ディレクトリ [BACKUP] に MASTER.DOC という名前のファイルがない場合は、COPY コマンドは入力ファイルのバージョン番号を出力ファイルに割り当てます。このコマンドを実行するためには、装置 DKA1 上の [BACKUP] ディレクトリへの書き込み (W) アクセス権が必要です。

9. \$ COPY SAMPLE.EXE DALLAS::DISK2:[000,000]SAMPLE.EXE/CONTIGUOUS

この例の COPY コマンドは、ローカル・ノードにあるファイル SAMPLE.EXE を、リモート・ノード DALLAS 上の SAMPLE.EXE にコピーします。
/CONCATENATE 修飾子が指定されているので、出力ファイルは物理的に連続したディスク領域に作成されます。このコマンドを実行するためには、リモート・ノード DALLAS の装置 DISK2 への書き込み (W) アクセス権が必要です。

10. \$ COPY *.* PRTLND:.*.*

この COPY コマンドは、ローカル・ノードのユーザ・ディレクトリ内にあるすべてのファイルを、PRTLND というリモート・ノードにコピーします。新しいファイルには、入力ファイルと同じ名前が与えられます。このコマンドが正しく機能するには、PRTLND というリモート・ノードの省略時のディレクトリに対して、書き込み (W) アクセス権が必要です。

11. \$ COPY BOSTON::DISK2:TEST.DAT;5
_To: DALLAS"SAM SECReturn"::DISK0:[MODEL.TEST]TEST.DAT/ALLOCATION=50

この COPY コマンドは、ノード BOSTON の DISK2 という装置の TEST.DAT;5 というファイルを DALLAS というリモート・ノードの TEST.DAT という名前の新しいファイルにコピーします。/ALLOCATION 修飾子は、ノード DALLAS の TEST.DAT という新しいファイルに対して、最初に 50 ブロックを割り当てます。リモート・ディレクトリをアクセスするために、"SAM SECReturn"というアクセス制御文字列を使用しています。

12. \$ MOUNT TAPED1: VOL025 TAPE:
\$ COPY TAPE:.*.* *

この例で MOUNT コマンドは、ラベルが VOL025 であるボリュームを、磁気テープ装置 TAPED1 にマウントし、その装置に論理名 TAPE を割り当てるよう指定しています。

COPY コマンドは、入力ファイル指定に論理名 TAPE を使用して、磁気テープ上のすべてのファイルを、現在の省略時のディスクおよびディレクトリ上にコピーするよう要求しています。コピーされたファイルはすべて、ファイル名とファイル・タイプを保持します。

13. \$ ALLOCATE CR:
_CR1: ALLOCATED
\$ COPY CR1: CARDS.DAT
\$ DEALLOCATE CR1:

この例の ALLOCATE コマンドは、排他的なカード・リーダの使用を要求しています。ALLOCATE コマンドからの応答は、カード・リーダ CR1 の装置名を示しています。

カード・リーダを占有したら、カードのデッキをリーダに入れ、入力ファイルとしてカード・リーダを指定して COPY コマンドを入力できます。COPY コマンドはカードをファイル CARDS.DAT に読み込みます。カード・デッキの終端 (EOF)

は、EOF カード (12-11-0-1-6-7-8-9 overpunch) により示されていなければなりません。

DEALLOCATE コマンドは、カード・リーダを解放します。

14. \$ COPY [SMITH]MONKEY.DIR [JONES]
\$ COPY [SMITH.MONKEY]*.* [JONES.MONKEY]*.*

この例では、COPY コマンドを用いて、[JONES]MONKEY.DIR ディレクトリ・ファイルとして登録される新しい空のディレクトリ[JONES.MONKEY]を作成します。COPY コマンドで[JONES]MONKEY.DIR ディレクトリ・ファイルを作成したら、[JONES.MONKEY]ディレクトリ上にファイルをコピーあるいは作成できます。

例中の 2 番目の COPY コマンドは、[SMITH.MONKEY]ディレクトリから[JONES.MONKEY]ディレクトリにファイルをコピーします。

15. \$ COPY [SMITH]CATS.DIR [SMITH]DOGS.DIR

この例では、COPY コマンドを用いて、新しい空のファイル[SMITH]DOGS.DIRを作成します。[SMITH]CATS.DIR ファイルと同じ属性を持つディレクトリ・ファイルを作成するには、この COPY コマンドを使用してください。以下のコマンドを入力しても、同じ結果が得られます。

```
$ CREATE/DIRECTORY [SMITH.DOGS]
```

16. \$ COPY [SMITH]TIGER.DIR [SMITH.ANIMALS]
\$ COPY [SMITH.TIGER]*.* [SMITH.ANIMALS.TIGER]*.*
\$ DELETE [SMITH.TIGER]*.*;
\$ SET SECURITY/PROTECTION=(WORLD:DELETE) TIGER.DIR
\$ DELETE TIGER.DIR;

この例ではまず、COPY コマンドを用いて新しい空のディレクトリ・ファイル[SMITH.ANIMALS]TIGER.DIRを作成します。続くコマンドでは、[SMITH.TIGER]ディレクトリから[SMITH.ANIMALS.TIGER]ディレクトリへファイルをコピーし、さらに元の TIGER.DIR ディレクトリを削除します。TIGER.DIR はディレクトリ・ファイルなので、これを削除する前に、保護コードの DELETE を設定しておかなければなりません。

COPY/FTP

FTP ユーティリティを起動して、TCP/IP 接続経由で異なるファイル・システムを持つホスト間でファイルを転送します。

フォーマット

COPY/FTP 入力ファイル指定 出力ファイル指定

パラメータ

入力ファイル指定

コピーする既存のファイル(コピー元ファイル)の名前を指定します。

出力ファイル指定

入力ファイルをコピーする先の出力ファイル(コピー先ファイル)の名前を指定します。

説明

COPY/FTP コマンドは、ファイル転送プロトコル(FTP)を使用して、リモート・ノードからファイルをコピーします。このコマンドが提供するサービスは、FTP のアーキテクチャに基づく機能のサブセットです(提供される FTP プログラムの使用法については、ベンダのマニュアルを参照してください)。

OpenVMS 間の転送

両方のマシンが OpenVMS 構造化転送をサポートしている場合、/BINARY、/ASCII、および/FDL 修飾子は無視されます。協調動作する OpenVMS FTP のクライアントとサーバは、正しい OpenVMS 属性を使用して自動的にファイルを転送します。

COPY/FTP では、一般にリモート・ファイル指定でアスタリスク・ワイルドカード文字(*)がサポートされています。

修飾子

/ANONYMOUS

単一または複数のノードに匿名アクセスします。/ANONYMOUS は、省略時

のリモート・アクセスです。リモート・ノードに対するパスワードの形式は、"user@fullyqualifiednodename"でなければなりません。

/ASCII

ASCII ファイル(テキスト・ファイル)を識別するために使用します。/ASCII は、省略時の設定です。

/BINARY

バイナリ・ファイルを識別するために必要です。

/FDL

この修飾子はオプションです。FDL(ファイル定義言語) ファイルとの会話を行います。ローカルな OpenVMS システムにファイルがコピーされる時は、リモート FDL ファイルが検索され、ファイル定義が復元されます。ファイルがローカルな OpenVMS システムの外にコピーされる時は、要求したファイルに加え、FDL ファイルが作成され、コピーされます。/FDL が指定されてもベンダのアプリケーションがサポートしていない場合は、警告メッセージが出力される可能性があります。

/LOG

ファイルが転送された場合、SYS\$OUTPUT にメッセージを表示します。

/NOSTRUVMS

STRU OpenVMS 転送の試行を明示的に無効にするために使用します。そうでない場合、サーバによっては試行時に強制終了する可能性があります。

/PASSIVE=オプション

FTP クライアントとサーバのどちらがデータ接続を開始するかを制御します。この修飾子を指定しなかった場合は、Internet Protocol に応じた値が使用されます。IPv4 の場合は OFF、IPv6 の場合は ON が使用されます。

/PASSIVE オプションは、次の表のとおりです。

オプション	説明
OFF	FTP サーバがデータ接続を開始します。
ON (省略時の値)	FTP クライアントがデータ接続を開始します。 これは、FTP クライアントとサーバ間にファイアウォールがあるためにサーバからの接続ができない場合に使用されます。 ON が省略時の値となるのは、/PASSIVE が指定された場合のみです。

この修飾子が機能するためには、基盤となる TCP/IP ネットワーク製品がこの修飾子を認識し、FTP パッシブをサポートしている必要があります。

/PASSIVE 修飾子は FTP PASV コマンドと同値です。

/VERBOSE

/NOVERBOSE

(バナー・メッセージを含む) すべてのメッセージが端末に表示されるかどうかを指定します。省略時の設定では、メッセージの表示を禁止します。

例

1. `$ COPY/FTP/FDL/ANON rms_indexed_file.idx -
remotehost5:="/public/rms.idx.file"`

この例では、TCP/IP 接続を介して、OpenVMS RMS ファイル `rms_indexed_file.idx` から `remotehost5` 上のリモート・ファイル `public/rms.idx.file` へ転送を行います。リモート・ノードへのアクセスは匿名で行われ、FLD ファイルが作成され、`rms_indexed_file.inx` とともにコピーされます。

2. `$ COPY/FTP/VERBOSE sys$login:login.com -
xdelta.zko.dec.com"username password":sys$login:login.tmp`

この例では、TCP/IP 接続を介して、OpenVMS RMS ファイル `sys$login:login.com` からリモート・ファイル `sys$login:login.tmp` へ、リモート・システムのユーザ名とパスワードを指定して転送を行います。

3. `$ COPY/FTP/LOG RESULTS.LOG -
_To: grad.uq.edu.au"JONES BYRONBAY":DKA200$:[JONES.DATA]`

この例では、COPY/FTP コマンドで RESULTS.LOG ファイルを、`uq.edu.au` インターネット・ドメイン内にある `grad` ノード上のユーザ・アカウント `JONES` とパスワード `BYRONBAY` を指定して `DKA200$:[JONES.DATA]RESULTS.LOG` へ転送を行います。

COPY/RCP

RCP ユーティリティを起動して、TCP/IP 接続経由で、ホスト間でファイルをコピーします。

フォーマット

COPY/RCP 入力ファイル指定 出力ファイル指定

パラメータ

入力ファイル指定

コピーする既存のファイル(コピー元ファイル)の名前を指定します。

出力ファイル指定

入力ファイルをコピーする先の出力ファイル(コピー先ファイル)の名前を指定します。

説明

COPY/RCP コマンドは、RCP ユーティリティを使用して、リモート・ホストとの間で1つまたは複数のファイル(またはディレクトリ階層)をコピーします。

TCP/IP 用の OpenVMS DCL コマンドは、DECnet ネットワーク接続用の DCL コマンドと同じリモート・ファイル指定書式をサポートしています。一部のファイル・トランザクション・アプリケーションのインプリメントでは、コピー元ファイルとコピー先ファイルがともにリモートのファイル指定であるファイル転送がサポートされています。

完全なリモート・ファイル指定形式は、次のとおりです。

```
node"username password account"::filename.ext
```

ファイルが OpenVMS 以外のシステムに存在する場合、ファイル名を二重引用符で囲みます。たとえば、U32 という Tru64 UNIX ノード上の/usr/users/user/Ordersというファイルにアクセスするには、次のように指定します。

```
U32"user password"::"/usr/users/user/Orders"
```

UNIX®システムは、大文字と小文字を区別するファイル指定をサポートしていることに注意してください。

修飾子

/AUTHENTICATE

リモート・ノードにアクセスするのに利用される，Kerberos 認証を指定します。

/LOG

ファイルが転送されたときに，SYS\$OUTPUT にメッセージを表示します。

/PRESERVE

ファイルの保護コードを保持します。

/RECURSIVE

サブディレクトリのコピーを要求します。

/TRUNCATE=USERNAME

ユーザ名を 8 文字で切り捨てます。

/USERNAME=ユーザ名

リモート・ユーザ名を指定する，オプションの修飾子です。省略時の操作では，ローカル端末のユーザ名と同じユーザ名を用いて，リモート・システムにログインします。このコマンドは，/USERNAME の値として引用符号に囲まれたパラメータをサポートします。

例

1. `$ COPY/RCP local_file.c remotehost4"Smith smpw"::rem_file.c`

この例では，TCP/IP 接続を介して，local_file.c をリモート・ホスト上の rem_file.c へコピーします。

CREATE

順編成ディスク・ファイル，または順編成ファイルを作成します。

フォーマット

CREATE ファイル指定[,...]

パラメータ

ファイル指定[,...]

作成する 1 つまたは複数の入力ファイルの名前を指定します。ワイルドカード文字は使用できません。ファイル名，またはファイル・タイプを省略しても CREATE コマンドからは省略時の設定は与えられないため，ファイル名やファイル・タイプは，空文字列になります。既存のファイル名を指定した場合には，新しいバージョンが作成されます。

説明

CREATE コマンドは，新しい順編成ディスク・ファイルを作成します。会話型モードでは，コマンド行の入力後に入力する行は，新しく作成されるファイルのレコードになります。ファイル入力を終了するには，Ctrl/Z を押します。

コマンド・プロシージャ・ファイルから CREATE コマンドを入力する場合，システムは，レコードの最初の位置がドル記号(\$)になるまで，コマンド・プロシージャ・ファイル以降のすべてのレコードを新しいファイルに読み込みます。ファイル入力は，レコードの最初の位置がドル記号になるか，コマンド・プロシージャの終了とともに終了します。

CREATE コマンドで既存のファイル指定を使用すると，新しく作成されるファイルのバージョン番号は，同じ指定を持つ既存ファイルより大きいバージョン番号になります。

CREATE コマンドを使用して論理名サーチ・リストにファイルを作成すると，ファイルは論理名変換で生成された最初のディレクトリだけに作成されます。

通常，出力ファイルの所有者は，出力ファイルの作成者と同じです。ただし，拡張特権を持つ利用者が出力ファイルを作成すると，所有者は親ディレクトリの，または出力ファイルの前バージョンの所有者になります。

拡張特権には、次の特権が含まれます。

- SYSPRV (システム特権) または BYPASS
- システム利用者識別コード (UIC)
- GRPPRV (グループ特権)。親ディレクトリ (または出力ファイルの前バージョン) の所有者が新しい出力ファイルの作成者と同じグループに属する場合
- 親ディレクトリ (または出力ファイルの前バージョン) の所有者を表す識別子 (リソース属性付き)

修飾子

/LOG

/NOLOG (省略時の設定)

コマンド実行として作成される新しい各ファイルのファイル指定を表示します。

/OWNER_UIC=uic

自分の UIC と異なる UIC を指定するためには、SYSPRV (システム特権) 特権が必要です。

作成するファイルの所有者の利用者識別コード (UIC) を指定します。UIC は、『OpenVMS ユーザーズ・マニュアル』に説明されている、標準的な UIC 形式を使用して指定します。

/PROTECTION=(所有区分[: アクセス・コード][,...])

ファイルに適用される保護を定義します。

- 所有区分は、システム (S)、所有者 (O)、グループ (G) またはワールド (W) から指定します。
- アクセス・コードは、読み込み (R)、書込み (W)、実行 (E)、または削除 (D) から指定します。

各アクセス・カテゴリの値を指定しない場合や、/PROTECTION 修飾子を省略すると、CREATE コマンドは、指定されていない各カテゴリごとに以下の保護を適用します。

ファイルが既に存在している	適用される保護
Yes	既存ファイルの保護
No	現在の省略時の保護

注意

アクセスを指定しないでファイルを作成しようとすると、そのファイルにはシステムの省略時の RMS 保護値が適用されます。アクセスを指定しないファ

CREATE

イルを作成するには、SET SECURITY/PROTECTION コマンドを使用してください。

保護コードの指定についての詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

/VOLUME=n

各ファイルが、マルチボリューム・セットの指定された相対ボリューム番号に作成されます。省略時の設定では、ファイルは、マルチボリューム・セットの任意の場所に作成されます。

例

1. \$ CREATE MEET.TXT
John, Residents in the apartment complex will hold their annual
meeting this evening. We hope to see you there, Regards, Elwood

省略時のディレクトリにテキスト・ファイル MEET.TXT を作成します。このテキスト・ファイルには、Ctrl/Z が入力されるまでの行が入っています。

2. \$ CREATE A.DAT, B.DAT
Input line one for A.DAT...
Input line two for A.DAT...
.
.
.

Input line one for B.DAT...
Input line two for B.DAT...
.
.
.

\$

ターミナルから CREATE コマンドを入力した後では、システムは CTRL/Z によって最初の入力を終了するまで、A.DAT という順編成ファイルに入力行を読み込みます。次の入力データは、B.DAT という 2 番目のファイルに入力されます。この場合も、CTRL/Z により入力は終了します。

```

3. $ FILE = F$SEARCH("MEET.TXT")
   $ IF FILE .EQS. ""
   $ THEN CREATE MEET.TXT
       John, Residents in the apartment complex will hold their annual
       meeting this evening. We hope to see you there, Regards, Elwood
   $ ELSE TYPE MEET.TXT
   $ ENDIF
   $ EXIT

```

この例では、省略時のディスクとディレクトリにファイル MEET.TXT があるかどうか検索します。ファイルが存在しなければ、CREATE コマンドを使って MEET.TXT というファイルを作成します。

```

4. $ SET DEFAULT DKA500:[TEST]
   $ SET PROCESS /CASE=CASE_LOOKUP=SENSITIVE /PARSE_STYLE=EXTENDED
   $ CREATE COEfile.txt
      Ctrl/Z
   $ CREATE COEFILE.TXT
      Ctrl/Z
   $ CREATE CoEfile.txt
      Ctrl/Z
   $ DIRECTORY

Directory DKA500:[TEST]

CoEfile.txt;1
COEFILE.TXT;1
COEfile.txt;1

```

この例では、DKA500 は ODS-5 ディスクです。プロセスを CASE_LOOKUP=SENSITIVE に設定し、大文字大文字小文字の区別が異なるだけで名前は同じ複数のファイルを作成すると、DCL は 2 目以降のファイルを新しいファイルとして処理し、この例のようにリストします。

CREATE/DIRECTORY

1 つまたは複数の、新しいディレクトリまたはサブディレクトリを作成します。
/DIRECTORY 修飾子は省略できません。

第 1 レベル・ディレクトリを作成する場合は、マスタ・ファイル・ディレクトリ (MFD) への書き込み (W) アクセス権が必要です。システム・ボリュームでは、一般に SYSTEM ユーザか、SYSPRV 特権または BYPASS 特権を持つユーザのみが MFD への書き込みアクセス権を持っています。

サブディレクトリを作成しようとするディレクトリへの書き込み (W) アクセス権が必要です。

フォーマット

CREATE/DIRECTORY ディレクトリ指定[,...]

パラメータ

ディレクトリ指定[,...]

作成する 1 つまたは複数のディレクトリ、あるいはサブディレクトリの名前を指定します。装置名 (およびコロンの[:]) を指定することもできます。省略時の設定は、現在の省略時のディレクトリです。ワイルドカード文字は使用できません。サブディレクトリを作成する場合には、ディレクトリ・レベルの各名前をピリオド(.) で区切ります。

1 つの CREATE/DIRECTORY コマンドで、一連のネスティングしたサブディレクトリを作成できます。たとえば、コマンドの入力時には[a.b]も[a]も存在しない場合でも、[a.b.c]というサブディレクトリを作成することが可能です。各サブディレクトリは、最上位レベルから順に下位レベルへ作成されます。

説明

CREATE/DIRECTORY コマンドは、新規にディレクトリとサブディレクトリを作成します。新しい第 1 レベルのディレクトリを作成するには、特殊な特権が必要です (上記の制限を参照)。一般に各利用者は、そのディレクトリにサブディレクトリを作成するための特権を持っています。ディレクトリ間を移動するには、SET DEFAULT コマンドを使用します。

修飾子

/ALLOCATION=n

指定したディレクトリに割り当てる初期ブロック数を指定します。省略時の設定では、割り当ては1ブロックです。

この修飾子は、MAIL.DIR;1などの大きなディレクトリを作成するのに便利です。後に必要となる、ディレクトリの動的拡張をしなくて済むため性能が向上します。

この修飾子はFiles-11 ODS-1、ODS-3、ODS-4 ボリュームには適用されません。

/LOG

/NOLOG (省略時の設定)

CREATE/DIRECTORY コマンドが各ディレクトリを作成したあと、作られた各ディレクトリを表示するか否かを制御します。

/OWNER_UIC=オプション

自分以外のユーザ識別コード (UIC) を指定するには、SYSPRV (システム特権) 特権が必要です。

作成されているディレクトリを所有する利用者識別コード (UIC) を指定します。省略時の設定は、作成者の UIC です。UIC の指定位置にキーワード PARENT を指定すると、一段上の親ディレクトリの UIC がコピーされます。特権を持ったユーザがサブディレクトリを作成する場合、省略時の設定では、親ディレクトリ (トップ・レベルディレクトリの場合は MFD) の所有者がそのディレクトリの所有者となります。ディレクトリの作成時に/OWNER_UIC 修飾子を指定しない場合には、このコマンドは、次に示すように所有者を割り当てます。

- 英数字形式またはサブディレクトリ形式でディレクトリ名を指定する場合には、省略時の所有者は作成者の UIC となります (UIC の省略時の値が、親ディレクトリの UIC になるような特権を作成者が持つ場合を除きます)。
- UIC 形式でディレクトリ名を指定する場合には、省略時の所有者はディレクトリ名に指定されている UIC となります。

UIC は、『OpenVMS ユーザーズ・マニュアル』に説明されている標準的な UIC 形式を使用して指定してください。

/PROTECTION=(所有区分[: アクセス・コード][,...])

ディレクトリに適用される保護を定義します。

- 所有区分は、システム (S)、所有者 (O)、グループ (G) またはワールド (W) から指定します。
- アクセス・コードは、読み込み (R)、書込み (W)、実行 (E)、または削除 (D) から指定します。

省略時の保護は、親ディレクトリ（一段上のディレクトリ、またはトップ・レベル・ディレクトリのマスタ・ディレクトリ）の保護から削除 (D) アクセスを除いたものです。

第 1 レベル・ディレクトリを作成している場合には、その 1 つ上のレベルのディレクトリは、MFD です (MFD の保護は、INITIALIZE コマンドによって設定されます)。

保護コードの指定についての詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

`/VERSION_LIMIT=n`

ディレクトリに存在できるファイルのバージョン数を指定します。ここで指定した制限を超えると、最小バージョンが自動的に削除されます。`/VERSION_LIMIT=0` と指定する場合には、バージョン・リミットは設定されません。指定できるバージョンの最大数は、32,767 です。省略時の設定では、親ディレクトリ (1 つ上のレベルのディレクトリ) に対して設定されているバージョン数が使用されます。

バージョン・リミットの設定を変更すると、新しいリミットは、設定が変更されたあとで作成されたファイルに対してだけ適用されます。変更する前に作成されたファイルの新しいバージョンに対しては、前のバージョン・リミットが適用されます。

`/VOLUME=n`

ディレクトリ・ファイルを、マルチボリューム・セットの内の指定された相対ボリューム番号に書き込むことを要求します。`/VOLUME` 修飾子を省略した場合には、ファイルは、マルチボリューム・セットの中の任意の場所に書き込まれます。

例

1. `$ CREATE/DIRECTORY/VERSION_LIMIT=2 $DISK1:[ACCOUNTS.MEMOS]`

この `CREATE/DIRECTORY` コマンドは、MEMOS というサブディレクトリを `$DISK1` という装置上の ACCOUNTS ディレクトリに作成します。このディレクトリには、2 つを越えるバージョンのファイルは保持できません。

2. `$CREATE/DIRECTORY/PROTECTION=(SYSTEM:RWED,OWNER:RWED,GROUP,WORLD) -
$[KONSTANZ.SUB.HLP]`

この `CREATE/DIRECTORY` コマンドは、VAX 上で `[KONSTANZ.SUB.HLP]` という名前のサブディレクトリを作成します。このディレクトリの保護は、システムと所有者には、読み込み (R)、書き込み (W)、実行 (E)、削除 (D) を許しますが、グループおよびびワールドには、すべてのアクセスを禁じています。

3. \$ CREATE/DIRECTORY DISK2:[GOLDSTEIN]

この CREATE/DIRECTORY コマンドは、[GOLDSTEIN]というディレクトリを DISK2 という装置上に作成します。第 1 レベル・ディレクトリを作成するためには、特別の特権が必要です。

4. \$ CREATE/DIRECTORY [HOFFMAN.SUB]
\$ SET DEFAULT [HOFFMAN.SUB]

この CREATE/DIRECTORY コマンドは、[HOFFMAN.SUB]という名前のサブディレクトリを作成します。このディレクトリ・ファイルは、[HOFFMAN]という名前のディレクトリに登録されます。SET DEFAULT [HOFFMAN.SUB]コマンドは、現在の省略時のディレクトリをこのサブディレクトリに変更します。したがって、このあと作成されるファイルはすべて、[HOFFMAN.SUB]に登録されます。

5. \$ CREATE/DIRECTORY [BOAEN.SUB1.SUB2.SUB3]

この例では、トップ・レベル・ディレクトリ[BOAEN]と、3つのサブディレクトリ ([BOAEN.SUB1], [BOAEN.SUB1.SUB2]と[BOAEN.SUB1.SUB2.SUB3]) を作成しています。

CREATE/FDL

FDL ファイルの指定を使用して新しい空のデータ・ファイルを作成するために、Create/FDL ユーティリティ (CREATE/FDL) を起動します。/FDL 修飾子は省略できません。

Create/FDL ユーティリティについての詳細は、『OpenVMS Record Management Utilities Reference Manual』あるいはオンライン・ヘルプを参照してください。

フォーマット

CREATE/FDL=*FDL* ファイル指定[/ファイル指定/

CREATE/MAILBOX (Alpha/I64 のみ)

MBAという名前の仮想メールボックスを作成し、それに入出力チャネル番号を割り当てます。/MAILBOX 修飾子は必須です。

注意

以下の特権が必要です。

- TMPMBX (一時的メールボックス) : 一時的メールボックス (省略時の設定) を作成するために必要
 - CMEXEC (エグゼクティブ・モードへの変更) : 一時的メールボックスを作成するために必要 (省略時の設定)。注意: この要件は一時的なものなので、将来のリリースでは削除されます。
 - PRMMBX (パーマネント・メールボックス) : パーマネント・メールボックスを作成するために必要 (/PERMANENT 修飾子を使用した場合)
 - SYSNAM (システム論理名) : システム論理名テーブルにメールボックスに対する論理名を作成するために必要。
 - GRPNAM (グループ論理名) : グループ論理名テーブルにメールボックスに対する論理名を作成するために必要。
-

メールボックスを削除するには、DELETE/MAILBOX コマンドを使用します。

フォーマット

CREATE/MAILBOX 論理名

パラメータ

論理名

新しいメールボックスに対する論理名を指定します。システムによりメールボックスが作成され、論理名がそのメールボックスを指すように設定されます。

説明

CREATE/MAILBOX コマンドは仮想メールボックスを作成します。

修飾子

/BUFFER_SIZE=n

システムの動的メモリのバイト数を指定します。このメモリは、メールボックスにメッセージを送る際のバッファとして使用されます。/BUFFER_SIZE を指定しないか 0 を指定した場合、オペレーティング・システムは省略時の設定値としてシステム・パラメータ DEFMBXBUFQUO の値を使用します。

/LOG

/NOLOG (省略時の設定)

新たに作成したメールボックスの名前を表示します。

/MESSAGE_SIZE=n

メールボックスに送信可能なメッセージの最大サイズをバイト単位で指定します。最大値は 65535 です。/MESSAGE_SIZE を指定しないか 0 を指定した場合は、オペレーティング・システムは省略時の設定値としてシステム・パラメータ DEFMBXMXMSG の値を使用します。

/PERMANENT

メールボックスがパーマナント・メールボックスであることを指定します。省略時の設定では、メールボックスは一時的メールボックスです。

/PROTECTION=(所有者[: アクセス][,...])

メールボックスに対する保護を指定します。

- 所有者パラメータには、system (S)、owner (O)、group (G)、world (W)を指定します。
- アクセスパラメータには、読み込み(R)、書き込み(W)、論理入出力(L)、物理入出力(P)を指定します。

保護を指定しない場合、メールボックス・テンプレートが使用されます。

詳しい保護コードの指定方法については、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

/TEMPORARY (省略時の設定)

メールボックスが一時的メールボックスであることを指定します。/PERMANENT を指定しない限り、省略時の設定は一時的メールボックスです。

例

1.

```
$CREATE/MAILBOX/PERMANENT/MESSAGE_SIZE=512/LOG MY_MAILBOX
%CREATE-I-CREATED, MBA38: created
$SHOW DEVICE MBA38/FULL
Device MBA38:, device type local memory mailbox, is online,
    record-oriented device, shareable, mailbox device.

Error count          0  Operations completed          0
Owner process        ""  Owner UIC                    [SYSTEM]
Owner process ID 00000000  Dev Prot  S:RWPL,O:RWPL,G:RWPL,W:RWPL
Reference count      0  Default buffer size          512
```

この例では、MY_MAILBOX という論理名を使用してパーマネント・メールボックスを作成しています。SHOW DEVICE コマンドにより、メールボックスの属性がすべて表示されます。

CREATE/NAME_TABLE

新しい論理名テーブルを作成します。/NAME_TABLE 修飾子は省略できません。

フォーマット

CREATE/NAME_TABLE テーブル名

パラメータ

テーブル名

作成する論理名テーブルの名前を指定します。テーブル名は1文字から31文字までの長さで、使用できる文字は、英数字、ドル記号(\$)、およびアンダースコア(_)です。テーブル名は大文字でなくてはなりません。小文字を使用した名前を指定すると、すべて大文字に変換されます。このテーブル名は、プロセス・ディレクトリ論理名テーブル(LNM\$PROCESS_DIRECTORY)とシステム・ディレクトリ論理名テーブル(LNM\$SYSTEM_DIRECTORY)のどちらかに、論理名として登録されます。

説明

CREATE/NAME_TABLE コマンドは新しい論理名テーブルを作成します。そのテーブルがプロセス固有のものであれば、テーブル名はLNM\$PROCESS_DIRECTORY ディレクトリ・テーブルに格納されます。共有可能なものである場合には、LNM\$SYSTEM_DIRECTORY ディレクトリ・テーブルに格納されます。

すべての新テーブルは親テーブルを持ち、親テーブルによって新テーブルがプロセス固有のものか共有可能なものかが決まります。プロセス固有のテーブルを作成するには、/PARENT_TABLE 修飾子を使用してプロセス固有テーブル(プロセス・ディレクトリ・テーブル)の名前を指定します。共有可能なテーブルを作成するには、親テーブルを共有可能テーブルとして指定します。

親テーブルを明示的に指定しない場合には、CREATE/NAME_TABLE コマンドは、親テーブルがLNM\$PROCESS_DIRECTORY であるプロセス固有テーブルを作成します。つまり、テーブル名はプロセス・ディレクトリに入れられます。

すべてのテーブルにはサイズ・クォータがあります。このクォータは、テーブルの潜在的な成長を押さえるか、またはテーブルのサイズが仮想的には無制限であることを示すことができます。/QUOTA 修飾子についての記述で、クォータの指定方法が説明されています。

作成するテーブルのアクセス・モードの指定には、/USER_MODE、/SUPERVISOR_MODE、または/EXECUTIVE_MODE 修飾子を使用します。これらの修飾子を 1 つ以上指定した場合には、最後に指定した修飾子だけが有効です。アクセス・モードを指定しない場合には、スーパーバイザ・モードのテーブルが作成されます。

論理名テーブルの削除には、DEASSIGN コマンドを使用します。この時、削除するテーブルの名前を指定し、/TABLE 修飾子にテーブルの名前が入れられたディレクトリ・テーブルを指定します。

論理名テーブルについての詳細は、『OpenVMS システム管理者マニュアル』を参照してください。

修飾子

/ATTRIBUTES[=(キーワード[,...])]

論理名テーブルの属性を指定します。キーワードを 1 つだけ指定する場合には、括弧を省略できます。/ATTRIBUTES 修飾子を指定しない場合には、属性は設定されません。

属性に対しては、次のキーワードを指定できます。

CONFINE	テーブルが、SPAWN コマンドによって生成されたサブプロセスにコピーされないことを指定します。このキーワードは、プロセスに固有な論理名テーブルを作成している場合にだけ使用できます。CONFINE 属性を持つテーブルが作成されると、そのテーブルに登録される名前はすべて、CONFINE 属性をもちます。
NO_ALIAS	より特権の低いアクセス・モードでは、現在のディレクトリに同じ名前（論理名または論理名テーブルの名前）に登録できないことを指定します。NO_ALIAS を指定しない場合には、そのテーブルには、より特権の低いアクセス・モードをもつ同じ名前を、「別名」としてつけることができます。NO_ALIAS 属性を持つテーブルの作成時に、同じモード、またはより特権の低いモードで、同じ名前がその論理名ディレクトリ・テーブルにすでに存在する場合には、この名前は削除されます。
SUPERSEDE	既存のテーブルと同じ名前、同じアクセス・モード、および同じディレクトリ・テーブルの新しいテーブルを指定する場合には、既存のテーブルを削除し、新しいテーブルを作成することを指定します。同じ名前のテーブルがすでに存在するかどうかとは無関係に、新しいテーブルは必ず作成されます（SUPERSEDE 属性を指定しない場合には、既存のテーブルが存在すれば、新しいテーブルは作成されません）。 /LOG 修飾子を指定した場合や、/LOG 修飾子に対する省略時の設定を使用した場合には、結果を示すメッセージが表示されます。

/EXECUTIVE_MODE

SYSNAM（システム理論名）特権が必要です。

エグゼクティブ・モードの論理名テーブルを作成します。エグゼクティブ・モードの論理名テーブル作成時に SYSNAM 特権を持たない場合には、スーパーバイザ・モードの論理名テーブルが作成されます。

CREATE/NAME_TABLE

/LOG (省略時の設定)
/NOLOG

SUPERSEDE 属性が指定されている場合や、同じテーブルがすでに存在していても SUPERSEDE 属性が指定されていない場合に、コマンドの結果を示す情報メッセージを表示するかどうかを制御します。省略時の設定は/LOG で、つまり情報メッセージが表示されます。

/PARENT_TABLE=テーブル

親テーブルに対する作成 (C) アクセス権とシステム・ディレクトリへの書き込み (W) アクセス権か、または SYSPRV 特権が必要です。

親テーブルの名前を指定します。親テーブルは、作成するテーブルが固有のものなのか共有可能なものなのかを決定します。親テーブルを指定しない場合には、省略時のテーブルとして LNM\$PROCESS_DIRECTORY が使用されます。共有可能テーブルの親テーブルは、LNM\$SYSTEM_DIRECTORY です。親テーブルは、作成するテーブルと同じかそれよりも高いアクセス・モードでなければなりません。

/PROTECTION=(所有区分[: アクセス・コード][,...])

共有可能テーブルに適用される保護を定義します。

- 所有区分は、システム (S), 所有者 (O), グループ (G) またはワールド (W) から指定します。
- アクセス・コードは、読み込み (R), 書き込み (W), 作成 (C) または削除 (D) から指定します。

保護コードの指定についての詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

/PROTECTION 修飾子は、共有可能論理名テーブルに対してだけ適用されます。プロセス固有の論理名テーブルには、適用されません。

/QUOTA=バイト数

論理名テーブルの上限サイズをバイト数で指定します。新しいテーブルに登録される各論理名のサイズは、この上限値から計算されます。新しいテーブルのクォータは、親テーブルのクォータ・ホルダから静的に減算されます。親テーブルのクォータ・ホルダは、テーブル階層を上向きにたどった際に出会う最初の論理名テーブルであり、クォータの明示値とクォータ・ホルダを持っています。/QUOTA 修飾子を指定しない場合や、/QUOTA=0 を指定する場合には、親テーブルのクォータ・ホルダが作成するテーブルのクォータ・ホルダとなり、新しいテーブルに論理名が登録されるたびに、動的に領域が減らされます。論理名からは空白文字が取り除かれます。/QUOTA 修飾子を指定しない場合や、/QUOTA=0 を指定する場合には、テーブルは無制限のクォータを持ちます。

/SUPERVISOR_MODE (省略時の設定)

スーパーバイザ・モードの論理名テーブルを作成します。モードを指定しない場合には、スーパーバイザ・モードの論理名テーブルが作成されます。

/USER_MODE

ユーザ・モードの論理名テーブルを作成します。モードを指定しない場合には、スーパーバイザ・モードの論理名テーブルが作成されます。

注意

コマンド・プロシージャを起動し終了すると、ユーザ・モード論理名は自動的に削除されます。

例

1.

```
$ CREATE/NAME_TABLE TEST_TAB
$ SHOW LOGICAL TEST_TAB
%SHOW-S-NOTRAN, no translation for logical name TEST_TAB
$ SHOW LOGICAL/TABLE=LNМ$PROCESS_DIRECTORY TEST_TAB
```

この CREATE/NAME_TABLE コマンドは、TEST_TAB という新しいテーブルを作成します。何も指定されていないので、テーブル名は省略時の設定によって、プロセス・ディレクトリに登録されます。最初の SHOW LOGICAL コマンドは省略時の設定によって、プロセス・ディレクトリ・テーブルを検索しないため、TEST_TAB という名前を見つけることができません。したがって、プロセス・ディレクトリを検索するために/TABLE 修飾子を使用します。

2.

```
$ CREATE/NAME_TABLE/ATTRIBUTES=CONFINE EXTRA
$ DEFINE/TABLE=EXTRA MYDISK DISK4;
$ DEFINE/TABLE=LNМ$PROCESS_DIRECTORY LNМ$FILE_DEV -
_$ EXTRA, LNМ$PROCESS, LNМ$JOB, LNМ$GROUP, LNМ$SYSTEM
$ TYPE MYDISK:[COHEN]EXAMPLE1.LIS
```

この例は、EXTRA という新しい論理名テーブルを CONFINE 属性で作成しています。EXTRA テーブルとその中の論理名は、サブプロセスに引き継がれません。

次に、論理名 MYDISK がテーブル EXTRA に登録されています。MYDISK をファイル指定に使用するには、テーブル EXTRA が RMS によるファイル解析時に検索されなければなりません。このため、EXTRA を等価名の 1 つとして持つプロセス固有の論理名 LNМ\$FILE_DEV を定義しています(システムは、装置とファイル指定の論理名展開に LNМ\$FILE_DEV を検索すべきテーブルへのポイントとして使用し、プロセス固有の LNМ\$FILE_DEV が定義されていれば、システムのそれに優先して使用します)。LNМ\$FILE_DEV の定義後は、EXTRA, プロセス・テーブル, ジョブ・テーブル, グループ・テーブル, システム・テーブルの順に検索され、MYDISK を DISK4 に展開される等価名としてファイル指定に使用できます。

CREATE/TERMINAL

他のターミナル・タイプをエミュレートするウィンドウを作成します。

注意

現在は、DECterm ウィンドウのみがサポートされています。

フォーマット

CREATE/TERMINAL [コマンド文字列]

パラメータ

コマンド文字列

作成するサブプロセスのコンテキストで実行されるコマンド文字列を指定します。/DETACH または/NOPROCESS と同時に指定することはできません。CREATE/TERMINAL コマンドは、SPAWN コマンドと同様に使用できます。

説明

CREATE/TERMINAL コマンドは現在のプロセスのサブプロセスを作成します。サブプロセスの作成時には、プロセス永久ファイルやイメージ、またはプロシージャ・コンテキストは親プロセスからはコピーされません。サブプロセスは、コマンド・レベル 0 (現在のプロンプトの DCL レベル) にセットされます。

/PROCESS 修飾子を指定しない場合、サブプロセスの名前は、親プロセスと同じ名前に一意の数字を加えたものになります。たとえば、親プロセスの名前が SMITH である場合には、サブプロセスの名前は SMITH_1 や SMITH_2 等になります。

サブプロセスでは親プロセスの LOGIN.COM は実行されません。これは、サブプロセスの初期化を速く行うために、コンテキストが別々にコピーされるためです。/WAIT 修飾子が指定されると、サブプロセスが終了し ATTACH コマンドを使用することで親プロセスに制御が戻るまで、親プロセスはハイバネート状態になります。

サブプロセスを終了し親プロセスに戻るには、LOGOUT コマンドを使用します。ATTACH コマンドを使用して、親プロセスを含むサブプロセス階層構造内の他のプロセスに制御を移すことができます (SHOW PROCESS/SUBPROCESS コマンドを

使用すると、サブプロセス階層構造内のプロセスが表示され、現在のプロセスが表示されます)。

注意

サブプロセス階層構造は CREATE/TERMINAL コマンドを使用して確立されるので、階層構造内のプロセスを終了する場合には注意しなければなりません。プロセスの終了時には、階層構造のその時点より下に位置するすべてのサブプロセスが自動的に終了します。たとえば、SPAWN/NOWAIT CREATE/TERMINAL コマンドは、DECterm ウィンドウが作成されるとすぐに終了するサブプロセスを生成します。このプロセスが終了すると、DECterm ウィンドウは消去されます。したがって、代わりに SPAWN/NOWAIT CREATE/TERMINAL/WAIT コマンドを使用すればプロセスを継続できます。

CREATE/TERMINAL コマンドの修飾子は、コマンドの動詞のすぐ後に指定しなければなりません。コマンド文字列パラメータは最後の修飾子の後に指定し、コマンド行の最後まで続けます。

修飾子

/APPLICATION_KEYPAD

作成するターミナル・ウィンドウを APPLICATION_KEYPAD 属性に設定します。/APPLICATION_KEYPAD と /NUMERIC_KEYPAD 修飾子のどちらも指定されない場合は、親プロセスの設定が引き継がれます (/NUMERIC_KEYPAD も参照してください)。

/BIG_FONT

作成するターミナル・ウィンドウの初期化時に、(リソース・ファイルに指定されたように) ビッグ・フォントを選択するように指定します。/BIG_FONT 修飾子を /LITTLE_FONT 修飾子と同時に指定するとエラーになります。どちらの修飾子も指定されない場合は、初期フォントはビッグ・フォントになります。

/BROADCAST

/NOBROADCAST

作成するターミナル・ウィンドウのブロード・キャスト・メッセージを有効にするかどうかを指定します。省略時の設定は、親プロセスの設定を引き継ぎます。

/CARRIAGE_CONTROL

/NOCARRIAGE_CONTROL

キャリッジ・リターンとライン・フィード文字を、サブプロセスのプロンプト文字列の出力前に出力するかどうかを指定します。省略時の設定では、CREATE/TERMINAL コマンドは親プロセスの設定をコピーします。/CARRIAGE_CONTROL 修飾子は、/NODETACH 修飾子の指定時にのみ使用できます。

/CLI=CLI ファイル指定
/NOCLI

サブプロセスで使用するコマンド言語インタプリタ (CLI) を指定します。省略時の CLI は、親プロセスと同じです (SYSUAF に定義されています)。/CLI 修飾子を指定した場合は、親プロセスの属性がサブプロセスにコピーされます。指定する CLI は、SYS\$SYSTEM: に置かれファイル・タイプが .EXE であるものでなければなりません。この修飾子は、/NODETACH 修飾子の指定時にのみ使用できます。

/CONTROLLER=ファイル指定

ターミナル・ウィンドウ・コントロール・イメージの名前を指定します。この名前を指定することで、基本製品でサポートされていない言語の制御等を行うコントローラとウィンドウを関係付けることができます。DECterm ウィンドウに対する省略時の設定は、SYS\$SYSTEM:DECW\$TERMINAL.EXE です。また、装置およびディレクタリの省略時設定は SYS\$SYSTEM であり、省略時のファイル・タイプは .EXE です。

注意

\$PARSE が返すファイル名の "name" フィールドを基に、メール・ボックスの論理名を決定します。たとえば、"name" フィールドが DECW\$TERMINAL の場合、メール・ボックス論理名は、DECW\$TERMINAL_MAILBOX_node::0.0 になります。また、旧バージョンとの互換性のために、DECW\$DECTERM_MAILBOX_host::0.0 も同じメール・ボックスを指すように定義されます。

/DEFINE_LOGICAL=({論理名, TABLE=テーブル名} [...])

作成する仮想ターミナルの名前を指す論理名を指定します。リスト形式で、論理名、または TABLE= に続いて論理名テーブルを指定します。論理名テーブルを指定した場合は、以降指定される論理名がそのテーブルに登録されます。省略時の設定は、プロセス論理名テーブルです。

/DETACH
/NODETACH (省略時の設定)

独立プロセスを作成するかサブプロセスを作成するかを指定します。/DETACH 修飾子を使用する (独立プロセスを作成する) 場合は、コマンド文字列パラメータを指定できません。

/DISPLAY=ディスプレイ名

ターミナル・ウィンドウが作成されるディスプレイ名を指定します。このパラメータが省略された場合には、論理名 DECW\$DISPLAY が使用されます。

/ESCAPE
/NOESCAPE

作成するターミナル・ウィンドウの ESCAPE (エスケープ文字) 属性の設定または解除を行います。省略時には、親プロセスの設定を引き継ぎます。

/FALLBACK
/NOFALLBACK

作成するターミナル・ウィンドウの FALLBACK (フォール・バック文字) 属性の設定または解除を行います。省略時には、親プロセスの設定を引き継ぎます。

/HOSTSYNC (省略時の設定)
/NOHOSTSYNC

作成するターミナル・ウィンドウの HOSTSYNC (ホスト同期) 属性の設定または解除を行います。省略時には、親プロセスの設定を引き継ぎます。

/INPUT=ファイル指定

新プロセスで SYS\$INPUT の代わりに使用する入力ファイルまたは装置を指定します。省略時設定では、作成されるターミナル・ウィンドウが使用されます。

/DETACH と同時に指定しても、指定しなくても使用できます。

/INSERT

作成するターミナル・ウィンドウの行編集機能の省略時設定を、挿入モードにします。/INSERT も/OVERSTRIKE 修飾子も指定されない場合は、親プロセスの設定が引き継がれます (/OVERSTRIKE も参照してください)。

/KEYPAD (省略時の設定)
/NOKEYPAD

親プロセスのキーパッド定義とキーパッド状態を引き継ぐかどうかを指定します。

/NODETACH 修飾子を指定した場合でのみ有効です。

/LINE_EDITING
/NOLINE_EDITING

作成するターミナル・ウィンドウの行編集機能を有効にします。省略時の設定では、親プロセスの設定が引き継がれます。

/LITTLE_FONT

作成するターミナル・ウィンドウの初期化時に、(リソース・ファイルに指定されたように) リトル・フォントを選択するように指定します。/LITTLE_FONT 修飾子を/BIG_FONT 修飾子と同時に指定するとエラーになります。どちらの修飾子も指定されない場合は、初期フォントはビッグ・フォントになります。

/LOGGED_IN (省略時の設定)
/NOLOGGED_IN

ユーザ名とパスワードを聞く (/NOLOGGED_IN) か、あるいは自動ログインをする (/LOGGED_IN) かを指定します。/DETACH 修飾子とともにのみ使用できます。

/LOGICAL_NAMES (省略時の設定)
/NOLOGICAL_NAMES

作成するターミナル・ウィンドウが、親プロセスの論理名を引き継ぐかどうかを指定します。/NODETACH 修飾子とともにのみ使用できます。

/NOTIFY

/NONOTIFY (省略時の設定)

作成するターミナル・ウィンドウの終了時に、親プロセスに対してブロードキャスト・メッセージを送って通知するかどうかを指定します。/NODETACH 修飾子とともにのみ使用できます。

/NUMERIC_KEYPAD

作成するターミナル・ウィンドウを NUMERIC_KEYPAD(数字キーパッド) ターミナル属性に設定します。/NUMERIC_KEYPAD も/APPLICATION_KEYPAD 修飾子も指定されない時は、親プロセスの設定が引き継がれます (/APPLICATION_KEYPAD も参照してください)。

/OVERSTRIKE

作成するターミナル・ウィンドウの行編集機能の省略時設定を、上書きモードにします。/INSERT も/OVERSTRIKE 修飾子も指定されない場合は、親プロセスの設定が引き継がれます (/INSERT も参照してください)。

/PASTHRU

/NOPASTHRU

作成するターミナル・ウィンドウの PASTHRU ターミナル属性を設定または解除します。省略時には、親プロセスの設定が引き継がれます。

/PROCESS (省略時の設定)

/PROCESS=プロセス名

/NOPROCESS

作成されるプロセスまたはサブプロセスのプロセス名を指定します。/NOPROCESS 修飾子を指定すると、プロセスなしでウィンドウが作成されます。

プロセス名を指定せずに/PROCESS 修飾子を指定すると、親プロセス名に一意の数字を付けた名前が付けられます。省略時のプロセス名は、ユーザ名_n の形式です。既存のプロセス名を指定した場合には、エラーになります。この修飾子は、/DETACH あるいは/NODETACH 修飾子とともに指定します。

/PROMPT=プロンプト

作成するターミナル・ウィンドウのプロンプト文字列を指定します。/NODETACH 修飾子とともにのみ使用できます。

/READSYNC

/NOREADSYNC

作成するターミナル・ウィンドウの READSYNC ターミナル属性を設定または解除します。省略時には、親プロセスの設定が引き継がれます。

/RESOURCE_FILE=ファイル指定

作成するターミナル・ウィンドウが、省略時設定の DECW\$USER_DEFAULTS:DECW\$TERMINAL_DEFAULT.DAT の代わりに使用する、リソース・ファイルを指定します。

/SYMBOLS (省略時の設定)

/NOSYMBOLS

サブプロセスが親プロセスの DCL シンボルを引き継ぐかどうかを指定します。

/NODETACH 修飾子とともにのみ指定できます。

/TABLE=コマンド・テーブル

サブプロセスが代わりに使用するコマンド・テーブル名を指定します。/NODETACH

修飾子とともにのみ指定できます。

/TTSYNC

/NOTTSYNC

作成するターミナル・ウィンドウの TTSYNC ターミナル属性を設定または解除します。省略時には、親プロセスの設定が引き継がれます。

/TYPE_AHEAD

/NOTYPE_AHEAD

作成するターミナル・ウィンドウの TYPE_AHEAD ターミナル属性を設定または解除します。省略時には、親プロセスの設定が引き継がれます。

/WAIT

/NOWAIT (省略時の設定)

別の DCL コマンド入力前に、サブプロセスの終了を待つかどうかを指定します。

/NOWAIT 修飾子を指定すれば、サブプロセスの終了を待たずに次のコマンドを実行できます。/NODETACH 修飾子とともにのみ指定できます。

/WINDOW_ATTRIBUTES=(パラメータ[,...])

作成するターミナル・ウィンドウの初期属性を指定します。リソース・ファイルから読み込まれるウィンドウ属性の省略時設定値に優先します。次のパラメータを指定できます。

パラメータ	説明
BACKGROUND	背景色
FOREGROUND	前景色
WIDTH	このパラメータは使われません。 COLUMNS を使用してください。
HEIGHT	このパラメータは使われません。 ROWS を使用してください。
X_POSITION	X 位置 (ピクセル単位)
Y_POSITION	Y 位置 (ピクセル単位)
ROWS	ウィンドウの、文字セル単位での行数。
COLUMNS	ウィンドウの、文字セル単位での桁数。
INITIAL_STATE	ウィンドウの初期状態。 ICON または WINDOW。
TITLE	ウィンドウのタイトル文字列
ICON_NAME	ウィンドウのアイコン名
FONT	使用するフォント名。 /LITTLE_FONT 修飾子を指定するか/LITTLE_FONT も/BIG_FONT 修飾子も指定しない場合は、リソース・ファイルに設定されているリトル・フォント名に優先します。その他の場合は、ビッグ・フォント名に優先します。フォント名は、論理名でも、完全なフォント・セット内のベース・フォントでも構いません。

例

1.

```
$ CREATE/TERMINAL=DECTERM/DETACH -
_$ /DISPLAY=MYNODE::0 -
_$ /WINDOW_ATTRIBUTES=( -
_$ ROWS=36, -
_$ COLUMNS=80, -
_$ TITLE="REMOTE TERMINAL", -
_$ ICON_NAME="REMOTE TERMINAL" )
```

この例では、ノード MYNODE:: 上に DECterm ウィンドウ内に独立プロセスを作成しています。ウィンドウは 36 行 80 桁で、そのタイトルとアイコン名は "Remote terminal" に設定されています。

2.

```
$ CREATE/TERMINAL=DECTERM -
_$ /NOPROCESS -
_$ /DEFINE_LOGICAL=(TABLE=LNMSGROUP,DBG$INPUT,DBG$OUTPUT)
```

この例では、プロセス名を指定せずに DECterm を作成しています。独立プロセスのデバッグのため論理名 DBG\$INPUT, DBG\$OUTPUT をグループ論理名テーブルに定義しています (これには GROUP 特権が必要です)。

DEALLOCATE

占有している装置を、他のプロセスで使用できるようにします(ただしその装置に対応付けられている論理名の解除は行いません)。DEALLOCATE コマンドは、使用中の装置は占有を解除しません。

フォーマット

DEALLOCATE 装置名[:]

パラメータ

装置名[:]

占有を解除する装置の名前を指定します。ここで指定する装置名は、物理装置名でも使用していない論理名でも構いません。物理装置名を指定する場合、省略時のコントローラは A、省略時のユニットは 0 です。このパラメータは/ALL 修飾子と同時に指定できません。

修飾子

/ALL

ユーザのプロセスで現在占有している装置のうち、現在使用されていないすべての装置の占有を解除します。この修飾子は、装置名パラメータと同時に指定できません。

例

1. \$ DEALLOCATE DMB1:

この例の DEALLOCATE コマンドは、コントローラ B の RK06/RK07 装置のユニット 1 の占有を解除します。

2. \$ ALLOCATE MT: TAPE
%DCL-I-ALLOC, _MTB1: allocated

·
·
·

\$ DEALLOCATE TAPE:

この例の ALLOCATE コマンドは、すべての磁気テープ・ドライブを占有し、それに論理名 TAPE を割り当てるよう要求しています。ALLOCATE コマンドの応答は、装置 MTB1 の占有に成功したことを示しています。DEALLOCATE コマ

DEALLOCATE

ンドは、論理名 TAPE を指定してテープ・ドライブの占有を解除することを指定しています。

3. \$ DEALLOCATE/ALL

この例の DEALLOCATE コマンドは、現在占有しているすべての装置の占有を解除します。

DEASSIGN

ALLOCATE コマンド， ASSIGN コマンド， DEFINE コマンド， または MOUNT コマンドによって設定された論理名の割り当てを解除します。また， DEASSIGN コマンドは， CREATE/NAME_TABLE コマンドで作成された論理名テーブルを削除します。

フォーマット

DEASSIGN [論理名[:]]

パラメータ

論理名[:]

割り当てを解除する論理名を指定します。論理名は， 1 文字から 255 文字までの長さです。論理名に英数字，ドル記号(\$)，あるいはアンダースコア(_)以外の文字が含まれている場合には，引用符(" ")で囲みます。/ALL 修飾子を使用する場合を除き，論理名パラメータは必ず指定します。

論理名パラメータの最後にコロン(:)を指定しても，コマンド・インタプリタはそのコロンを無視します (ASSIGN コマンドおよび ALLOCATE コマンドは，コロンが指定されている場合でも論理名を論理名テーブルに登録するときに，論理名からそのコロンを削除します)。論理名が 1 つまたは複数の続くコロンを含む場合は，DEASSIGN 論理名パラメータに追加のコロンを 1 つ付ける必要があります。(たとえば，論理名 FILE: を割り当て解除するには，DEASSIGN FILE:: と入力します。)

論理名テーブルを削除するには，論理名パラメータとしてテーブル名を指定します。またそのテーブル名が登録されている論理名ディレクトリ・テーブルを指定するために，/TABLE 修飾子を使用します。

説明

DEASSIGN コマンドは，ALLOCATE，ASSIGN，DEFINE，または MOUNT コマンドのいずれかで行った論理名の割り当てを解除します。また，DEASSIGN コマンドは，CREATE/NAME_TABLE コマンドで作成した論理名テーブルを削除します。DEASSIGN で/ALL 修飾子を使用すると，指定したテーブルのすべての論理名の割り当てを解除できます。テーブルを指定しないで/ALL 修飾子を使用すると，(コマンド・インタプリタで作成された名前を除いて) プロセス・テーブル内のすべての名前の割り当てが解除されます。つまり，指定したアクセス・モードまたは外側のアクセス・モードで入力されたすべての名前の割り当てが解除されます。

論理名の割り当てを解除したい論理名テーブルを指定するには、/PROCESS、/JOB、/GROUP、/SYSTEM、または/TABLE 修飾子を使用します。複数の修飾子を指定した場合は、最後に指定した修飾子だけが有効です。指定した論理名のエントリが複数の論理名テーブルに存在する場合、コマンド行に指定した最後の論理名テーブルだけから名前が削除されます。論理名テーブルを指定しない場合、省略時の設定は/TABLE=LN\$PROCESS 修飾子です。

共有可能論理名を削除するには、論理名テーブルに対する書き込み (W) アクセス権が必要です。共有可能論理名テーブルを削除するには、親テーブルに対する書き込み (W) アクセス権と、対象とする論理名テーブルに対する削除 (D) アクセス権が必要です。

割り当てを解除したい論理名のアクセス・モードを指定するには、/USER_MODE、/SUPERVISOR_MODE、または/EXECUTIVE_MODE 修飾子を使用します。複数の修飾子を指定した場合は、最後に指定した修飾子だけが有効です。モードを指定しないと、スーパーバイザ・モード名が削除されます。論理名の割り当てを解除すると、同じ論理名テーブル内に外側のアクセス・モードで作成された同じ名前も削除されます。

エグゼクティブ・モード論理名の割り当てを解除するには、SYSNAM (システム論理名) 特権が必要です。

SYSMAN 特権を持たないで/EXECUTIVE_MODE 修飾子を指定すると、修飾子は無視され、スーパーバイザ・モード論理名の割り当てが解除されます。

システムからログアウトすると、すべてのプロセス・プライベート論理名と論理名テーブルが削除されます。イメージを終了すると、プロセス論理名テーブル内の利用者モード・エントリの割り当てが解除されます。システムからログオフすると、ジョブ・テーブル内の論理名とジョブ・テーブル自体が削除されます。

他のすべての共有可能論理名テーブル内の名前は、利用者モード名、スーパーバイザモード名、エグゼクティブ・モード名のいずれであっても、明示的に割り当てが解除されるまで残っています。共有可能論理名テーブル内の名前を削除するには、そのテーブルに対する書き込み (W) アクセス権が必要です。

論理名テーブルを削除すると、テーブル内のすべての論理名も削除されます。子孫テーブルも削除されます。共有可能論理名テーブルを削除するには、そのテーブルに対する削除 (D) アクセス権が必要です。

修飾子

/ALL

指定した論理名テーブルと同じアクセス・モード、またはそれより外側の (特権の低い) アクセス・モードで登録された、すべての論理名が削除されることを指定しま

す。論理名テーブルを指定しない場合には、省略時の値として、プロセス・テーブルである LNM\$PROCESS が使用されます。/ALL を指定する場合には、論理名パラメータは入力できません。

/CLUSTER_SYSTEM

SYSTEM アカウントでログインするか、クラスタ・ワイド論理名の割り当てを解除するための SYSNAM (システム論理名) または SYSPRV (システム) 特権を持っている必要があります。

LNМ\$SYSCLUSTER テーブルから論理名の割り当てを解除します。

/EXECUTIVE_MODE

エグゼクティブ・モードの論理名の割り当てを解除するためには、SYSNAM (システム論理名) 特権が必要です。

指定したアクセス・モード、またはそれより外側の (特権の低い) アクセス・モードで登録された、エントリだけが削除されることを指定します。エグゼクティブ・モードに対する SYSNAM 特権を持たない場合には、スーパーバイザ・モードの論理名の割り当てを解除します。

/GROUP

グループ論理名テーブルからエントリを削除するためには、GRPNAM (グループ論理名) または SYSPRV 特権が必要です。

指定した論理名が、グループ論理名テーブルに登録されていることを示します。

/GROUP 修飾子は、/TABLE=LNМ\$GROUP 修飾子の同意語です。

/JOB

指定した論理名が、ジョブ論理名テーブルに登録されていることを示します。/JOB 修飾子は、/TABLE=LNМ\$JOB 修飾子の同意語です。論理名テーブルを明示的に指定しない場合には、省略時の設定として /PROCESS 修飾子が使用されます。

ログイン時にシステムが作成するジョブ論理名、たとえば、SYS\$LOGIN、SYS\$LOGIN_DEVICE、および SYS\$SCRATCH などの割り当ては、解除してはいけません。ただし、これらの論理名に対して、ユーザ自身が新しい等価名を割り当てている場合 (つまり、より特権の低いアクセス・モードで新しい論理名を作成している場合) には、明示的に作成した論理名の割り当てを解除することが可能です。

/PROCESS (省略時の設定)

指定した論理名が、プロセス論理名テーブルに登録されていることを示します。

/PROCESS 修飾子は /TABLE=LNМ\$PROCESS 修飾子と同意語です。

コマンド・インタプリタが作成した論理名、たとえば SYS\$INPUT、SYS\$OUTPUT、および SYS\$ERROR などの割り当ては、解除することができません。ただし、これらの論理名に対して新しい等価名を割り当てている場合 (つまり、より特権の低いアクセス・モードで新しい論理名を作成している場合) には、明示的に作成した論理名の割り当てを解除することが可能です。

/SUPERVISOR_MODE (省略時の設定)

指定した論理名テーブルに登録されている論理名の中で、スーパーバイザ・モードで作成された論理名を削除します。/SUPERVISOR_MODE 修飾子を指定すると、DEASSIGN コマンドは、ユーザ・モードの同じ名前の論理名の割り当ても解除します。

/SYSTEM

指定した論理名が、システム論理名テーブルに登録されていることを示します。

/SYSTEM 修飾子は、/TABLE=LNM\$SYSTEM 修飾子の同意語です。

/TABLE=テーブル名

削除する論理名が含まれている論理名テーブルの名前を指定します。省略時の設定は LNM\$PROCESS です。プロセス・テーブル、ジョブ・テーブル、グループ・テーブル、システム・テーブル、あるいはディレクトリ・テーブルの 1 つ、または利用者定義テーブルの名前を指定することができます (プロセス、ジョブ、グループ、システムの各論理名テーブルは、それぞれ LNM\$PROCESS, LNM\$JOB, LNM\$GROUP, LNM\$SYSTEM という論理名で指定しなければなりません)。

/TABLE 修飾子を使用すれば、論理名テーブルを削除することもできます。プロセスに固有なテーブルを削除するためには、次のように指定します。

```
$ DEASSIGN/TABLE=LNM$PROCESS_DIRECTORY テーブル名
```

共有可能テーブルを削除するためには、次のように指定します。

```
$ DEASSIGN/TABLE=LNM$SYSTEM_DIRECTORY テーブル名
```

共有可能な論理名テーブルを削除するためには、そのテーブルに対して削除 (D) アクセスが可能であるか、または共有可能テーブルの名前が登録されているディレクトリ・テーブルに対して書き込み (W) アクセスが可能でなければなりません。

/TABLE 修飾子を指定しない場合には、省略時の設定として

/TABLE=LNM\$PROCESS 修飾子が使用されます。

/USER_MODE

ユーザ・モードで作成されたプロセス論理名テーブルの論理名を削除します。

/USER_MODE 修飾子を指定する場合、DEASSIGN コマンドは、ユーザ・モードの論理名の割り当てのみを解除します。また、ユーザ・モード論理名は、コマンド・プロシージャの起動および実行中に、自動的に削除されます。

例
1. \$ DEASSIGN MEMO

この例では、プロセス論理名 MEMO を解除しています。

2. \$ DEASSIGN/ALL

この例では、ユーザ・モードおよびスーパーバイザ・モードの論理名をすべて解除しています。しかし、コマンド・インタプリタによってプロセス論理名テーブルに登録されたエグゼクティブ・モードの論理名(たとえば、SYS\$INPUT, SYS\$OUTPUT, SYS\$ERROR, SYS\$DISK, および SYS\$COMMAND 等)は消しません。

3. \$ DEASSIGN/TABLE=LNK\$PROCESS_DIRECTORY TAX

この例では、論理名テーブル TAX とその下層のテーブルを削除します。論理名テーブルを削除するには、/TABLE=LNK\$PROCESS_DIRECTORY または/TABLE=LNK\$SYSTEM_DIRECTORY 修飾子を指定しなければなりません。これは、すべてのテーブル名はこれらのディレクトリに格納されているためです。

4. \$ ASSIGN USER_DISK: COPY
 \$ SHOW LOGICAL COPY
 "COPY" = "USER_DISK:" (LNK\$PROCESS_TABLE)
 \$ DEASSIGN COPY

この ASSIGN コマンドは、論理名 COPY と装置 USER_DISK を等価にし、プロセス論理名テーブルに置きます。DEASSIGN コマンドは、論理名を削除します。

5. \$ DEFINE SWITCH: TEMP
 \$ DEASSIGN SWITCH::

この DEFINE コマンドは、SWITCH: という論理名を、プロセス論理名テーブルに登録します。最後のコロンは、論理名の一部として保存されます。この論理名を DEASSIGN コマンドで削除するためには、2つのコロンが必要です。これは、DEASSIGN コマンドが最後のコロンを1つ削除してしまうので、論理名に含まれる文字と一致するようにもう1つのコロンが必要なためです。

6. \$ ASSIGN/TABLE=LNK\$GROUP DKA1: GROUP_DISK
 \$ DEASSIGN/PROCESS/GROUP GROUP_DISK

この例では、グループ論理名 GROUP_DISK を作成しています。次の DEASSIGN コマンドの修飾子は矛盾していますが、/GROUP 修飾子が後に指定されているためうまく論理名が削除されています。

7. \$ ASSIGN DALLAS::USER_DISK: DATA
 .
 .
 .
 \$ DEASSIGN DATA

この ASSIGN コマンドは、DALLAS というリモート・ノードの USER_DISK という装置指定に対して、DATA という論理名を割り当てます。この後、DATA という論理名を参照すると、DALLAS というリモート・ノードの USER_DISK というディスクが参照されます。DEASSIGN コマンドは、この DATA という論理名の割り当てを解除します。

DEASSIGN/QUEUE

プリント・キューまたはターミナル・キューから，論理キューの割り当てを解除し，論理キューを停止します。DEASSIGN/QUEUE コマンドは，バッチ・キューには使用できません。

キューに対する管理 (M) アクセス権が必要です。

フォーマット

DEASSIGN/QUEUE 論理キュー名[:/]

パラメータ

論理キュー名[:/]

特定のプリント・キュー，またはターミナル・キューから割り当てを解除する，論理キューの名前を指定します。

説明

DEASSIGN/QUEUE コマンドを入力すると，論理キュー内のジョブは，ASSIGN/QUEUE コマンドでキューを別のプリント・キューまたは装置に再割り当てされるまで，待ち状態になります。

例

```
1. $ ASSIGN/QUEUE LPA0 ASTER
   .
   .
   .
   $ DEASSIGN/QUEUE ASTER
   $ ASSIGN/MERGE LPB0 ASTER
```

この ASSIGN/QUEUE コマンドは，ASTER という論理キューを LPA0 というプリント・キューに割り当てます。このあと DEASSIGN/QUEUE コマンドを使用して，この論理キューの割り当てを解除します。ASSIGN/MERGE コマンドは，ASTER に含まれるジョブを，LPB0 というプリント・キューに再割り当てします。

DEBUG

OpenVMS Debugger を起動します。

OpenVMS Debugger についての詳細は『OpenVMS デバッガ説明書』を参照してください。

DCL レベルからデバッガ・コマンドのヘルプを参照したい場合は、次のコマンドを入力します。

```
$ HELP/LIBRARY=SYS$HELP:DBG$HELP DEBUG
```

フォーマット

DEBUG

ヒープ・アナライザ

ヒープ・アナライザには、メモリの使用状況をリアルタイムで図に示す機能があります。示された図を検討することで、性能を改善できる可能性のあるアプリケーション部分を素早く判別することができます。たとえば回数の多すぎるメモリ割り当て、大きなメモリ・ブロック、フラグメンテーションの顕在化、あるいはメモリ・リークなどに気付きます。

デバッガからのヒープ・アナライザの実行についての詳細は、『OpenVMS デバッガ説明書』を参照してください。

OpenVMS I64 では、スタンドアロン・ヒープ・アナライザはデバッガから起動します。OpenVMS Alpha および OpenVMS VAX では、スタンドアロン・ヒープ・アナライザは次のコマンドで起動します。

```
$ DEFINE/USER/NAME=CONFINE LIBRTL SYS$LIBRARY:LIBRTL_INSTRUMENTED
$ RUN/NODEBUG program
```

修飾子

/CLIENT

DEBUG クライアントの Motif インタフェースを起動します。クライアントから、サーバによって表示されるネットワーク・バインディング文字列を使用して接続します。サーバに最初に接続したクライアントは 1 次クライアントであり、そのサーバに接続できる 2 次クライアントの数を制御します。

/KEEP

保持デバッガ (Kept Debugger) を起動します。保持デバッガは、1 つのイメージを何度もデバッグしたり、デバッガを終了せずに特定の複数のイメージをデバッグしたりすることができます。

保持デバッガを起動するには、DEBUG/KEEP コマンドを実行します。

/RESUME (省略時の設定)

デバッグしているプログラムの実行を Ctrl/Y で中断した後、非保持デバッガを再起動します (LINK コマンドで/NOTRACEBACK 修飾子を指定してリンクしたプログラムを中断することはできません)。

Ctrl/Y でプログラムの中断をしていない場合は、DEBUG/RESUME コマンドを実行しても何も起こりません。

/SERVER [=([BINDING_INFO=ファイル指定] [,PROTOCOLS=(プロトコル[,...])])]

DEBUG サーバを起動します。DEBUG サーバには、同一の OpenVMS ノードあるいはリモート OpenVMS ノード、または Microsoft® Windows® 95 あるいは Microsoft Windows NT®が稼動している PC ノード上の最大 30 までのクライアントが同時に接続することができます。

省略可能な BINDING_INFO キーワードを指定する場合は、サーバ・バインディング識別文字列が書き込まれるファイル指定を指定します。このキーワードを指定しない場合は、ファイルは作成されません。

省略可能な PROTOCOLS キーワードを指定する場合は、DEBUG サーバに接続するために有効とするネットワーク・プロトコルを指定します。指定されたプロトコルだけが有効となります。このキーワードを指定しない場合は、すべてのプロトコルが有効となります。プロトコル引数には、次のキーワードのいずれか 1 つあるいは複数を指定することができます。

ALL

[NO]DECNET

[NO]TCP_IP

[NO]UDP

サーバに最初に接続したクライアントは 1 次クライアントとなります。1 次クライアントの接続後に接続したクライアントは 2 次クライアントとなります。1 次クライアントは、そのサーバに接続できる 2 次クライアントの数を制御します。

サーバは、一連の RPC バインディング文字列を表示して、クライアントがサーバに接続するために経由するポート番号を識別します。ポート番号は、識別文字列の最後の角括弧 ([]) の中表示されます。

クライアントから接続する場合の最も簡単なポート識別文字は、サーバのノード名と角括弧に囲まれたポート番号で構成されます。次の例はすべて正しいバインディング識別文字列です。

```
NODNAM[1234]
NCACN_IP_TCP:16.32.16.25[1112]
16.32.16.25[1112]
NCACN_DNET_NSP:63.1004[RPC20A020DD0001]
```

注意

デバッグ・サーバを起動するには、ライト・データベースに DBG\$ENABLE_SERVER 識別子を持っている必要があります。デバッグ・サーバを使用する場合は注意してください。一度デバッグ・サーバを起動すると、ネットワーク上の任意のクライアントがそのデバッグ・サーバに接続することができます。

システム管理者は、DBG\$ENABLE_SERVER 識別子を許可する前に、ライト・データベースにアクセスするための書き込みアクセスを持つアカウントから DEBUG/SERVER コマンドを実行してこの識別子を作成しておかなければなりません。システム管理者は一度だけこの識別子を作成する必要があります。以降は、Authorize ユーティリティを起動して、ライト・データベースのユーザ・アカウントに DBG\$ENABLE_SERVER 識別子を許可することができます。

/TARGET_ARCHITECTURE[=オプション]

/KEEP とともに使用し、System Code Debugger を使用してデバッグを実行する対象の OpenVMS ソフトウェアを実行しているシステムのアーキテクチャを指定します。オプションは、次のとおりです。

HOST (省略時の設定)	ターゲット・アーキテクチャが、ホストのアーキテクチャと同じ(つまり、DEBUG/KEEP コマンドを入力するシステムである)。
ALPHA	ターゲット・アーキテクチャが、Alpha プロセッサである。
IA64	ターゲット・アーキテクチャが、Intel® Itanium® プロセッサである。

例

1. \$ FORTRAN/DEBUG/NOOPTIMIZE WIDGET
\$ LINK/DEBUG WIDGET
\$ RUN WIDGET

[Debugger Banner and Version]

```
%DEBUG-I-INITIAL, language is FORTRAN, module set to WIDGET
DBG>
```

FORTTRAN および LINK コマンドは/DEBUG 修飾子を指定して、デバッガ・シンボル・テーブル情報とプログラム WIDGET.FOR をコンパイルしています。このプログラムはデバッグ情報でコンパイルおよびリンクされているので、RUN コマンドでプログラムが開始されるとイメージ・アクティベータは自動的にデバッガ

を起動します。デバッガが起動された時点では、プログラム・コードは展開されていません。

```
2. $ FORTRAN/DEBUG/NOOPTIMIZE WIDGET
$ LINK/DEBUG WIDGET
$ RUN/NODEBUG WIDGET

NAME:
NAME:
NAME:
^Y
$ DEBUG/RESUME

[ Debugger Banner and Version ]

%DEBUG-I-INITIAL, language is FORTRAN, module set to WIDGET
DBG>
```

FORTRAN および LINK コマンドは/DEBUG 修飾子を指定して、デバッガ・シンボル・テーブル情報でプログラム WIDGET.FOR をコンパイルしています。RUN コマンドはイメージ WIDGET.EXE の実行を始めます。このイメージはループし、制御不可能です。Ctrl/Y でプログラムに割り込みをかけ、DEBUG/RESUME コマンドで制御をデバッガに移します。

```
3. $ CC/DEBUG/NOOPTIMIZE ECHOARGS
$ LINK/DEBUG ECHOARGS
$ ECHO == "$ sys$disk:[]echoargs.exe"
$ DEBUG/KEEP

[ Debugger Banner and Version ]

DBG> RUN/COMMAND="ECHO"/ARGUMENTS="fa sol la mi"
%DEBUG-I-INITIAL, language is C, module set to ECHOARGS
%DEBUG-I-NOTATMAIN, type G0 to get to start of main program
DBG>
.
.
.
DBG> RERUN/ARGUMENTS="fee fii foo fum"
%DEBUG-I-INITIAL, language is C, module set to ECHOARGS
%DEBUG-I-NOTATMAIN, type G0 to get to start of main program
DBG>
.
.
.
DBG> RUN/ARGUMENTS="a b c" ECHOARGS
%DEBUG-I-INITIAL, language is C, module set to ECHOARGS
%DEBUG-I-NOTATMAIN, type G0 to get to start of main program
DBG>
```

CC および LINK コマンドは/DEBUG 修飾子を指定して、デバッガ・シンボル・テーブル情報でプログラム ECHOARGS.C をコンパイルしています。

シンボル定義コマンドは、デバッグ・セッション中にフォーリン・コマンドを定義します。

DEBUG/KEEP コマンドは、保持デバuggを起動します。

最初の RUN コマンドは、/COMMAND 修飾子を使用してイメージ・ファイルを起動するフォーリン・コマンドを指定し、/ARGUMENTS 修飾子を使用して引数の文字列を指定します。

RERUN コマンドは同一イメージ・ファイルを再起動し、/ARGUMENTS 修飾子を使用して新しい引数の文字列を指定します。

2 番目の RUN コマンドは新しいイメージ・ファイル、および新しい引数の文字列を指定します。

4. \$ PASCAL/DEBUG/NOOPTIMIZE 8QUEENS
 \$ LINK/DEBUG 8QUEENS
 \$ DEFINE/USER/NAME=CONFINE LIBRTL SYS\$LIBRARY:LIBRTL_INSTRUMENTED
 \$ RUN/NODEBUG 8QUEENS

[Heap Analyzer window is displayed]

PASCAL および LINK コマンドは/DEBUG 修飾子を指定して、デバugg・シンボル・テーブル情報でプログラム 8QUEENS.PAS をコンパイルします。

DEFINE コマンドにより、ヒープ・アナライザはメモリ割り当てとメモリ割り当ての解除に関する情報を収集する LIBRTL にアクセスします。

RUN/NODEBUG コマンドはヒープ・アナライザを起動しますが、デバuggは起動しません。

5. \$ DEBUG/SERVER=(PROTOCOLS=(TCP_IP,DECNET))

 %DEBUG-I-SPEAK: TCP/IP: YES, DECnet: YES, UDP: NO
 %DEBUG-I-WATCH: Network Binding: ncacn_ip_tcp:16.32.16.25[1112]
 %DEBUG-I-WATCH: Network Binding: ncacn_dnet_nsp:63.1004[RPC20A020DD0001]
 %DEBUG-I-AWAIT: Ready for client connection...

DEBUG/SERVER コマンドは、ネットワーク・プロトコルの TCP/IP および DECnet を指定して、デバugg・サーバに接続します。バインディング文字列は TEMP.TMP ファイルに保存されることに注意してください。TYPE コマンドを使用して、TEMP.TMP の内容を表示することができます。

DECK

コマンド，またはプログラムの入力ストリームの先頭を示します。

フォーマット

DECK

説明

DECK コマンドは，コマンドまたはプログラムの入力になるデータに印を付けます。DECK コマンドを使用できるのは，入力データが必要なコマンドまたはプログラムを実行する要求の後だけです。

コマンド・プロシージャでは，入力ストリーム内の任意のデータ・レコードの空白でない最初の文字がドル記号の場合，このコマンドが必要です。また，コマンド・プロシージャでは，DECK コマンドの前にドル記号を付けなければなりません。ドル記号は，入力レコードの最初の文字位置 (カラム 1) になければなりません。

DECK コマンドは，単一データ・ストリームに対してだけファイルの終端 (EOF) 指示子を定義します。DECK コマンドを使用すると，ドル記号で始まるデータ・レコードを入力ストリームに入れることができます。入力ストリームに 1 つまたは複数のデータの集合を入れるには，DECK コマンドの後にそれぞれを EOF 指示子で終了させたデータの集合を続けます。

/DOLLARS 修飾子で指定した EOF 指示子が検出されると，EOF 指示子は省略時の設定，つまりドル記号で始まるレコードに再設定されます。現在のコマンド・レベルの実際の EOF 指示子が検出された場合も，省略時の設定が再設定されます。

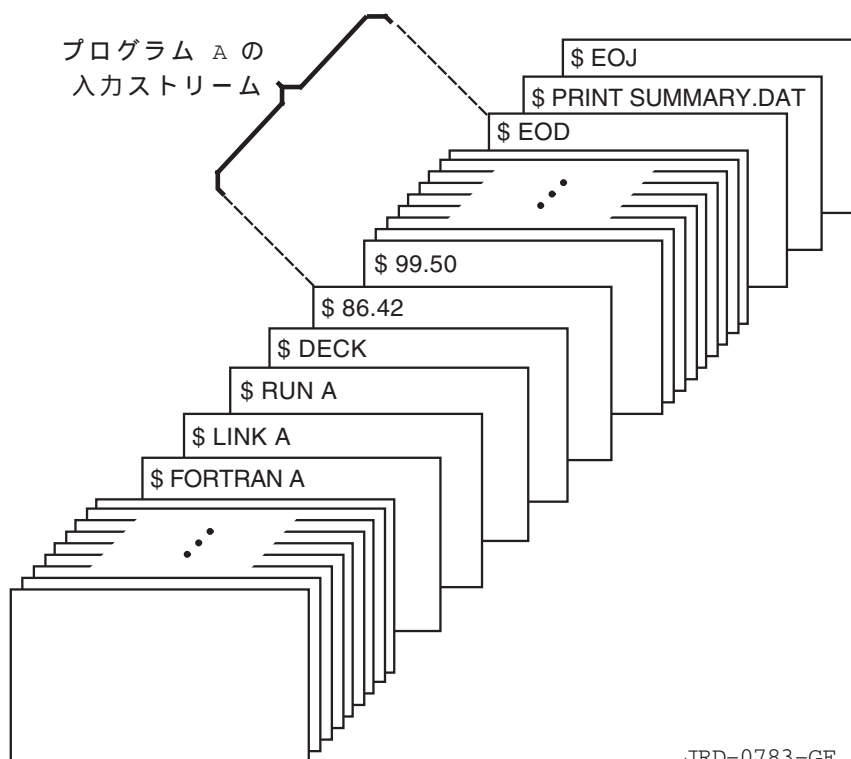
修飾子

/DOLLARS[=文字列]

指定した 1 ~ 15 文字の文字列を，ファイル終了 (EOF) 指示子として設定します。\$EOD という文字列から始まる 1 つまたは複数のレコードが，入力データに含まれている場合に，文字列を指定します。リテラルとしての小文字や，複数の空白あるいはタブを含むファイル終了指示子を指定する場合には，引用符 (" ") で囲まなければなりません。/DOLLARS 修飾子を指定しない場合や，/DOLLARS 修飾子だけを指定して文字列を省略する場合には，ファイルの終了 (EOF) を示すために，EOD コマンドを使用しなければなりません。

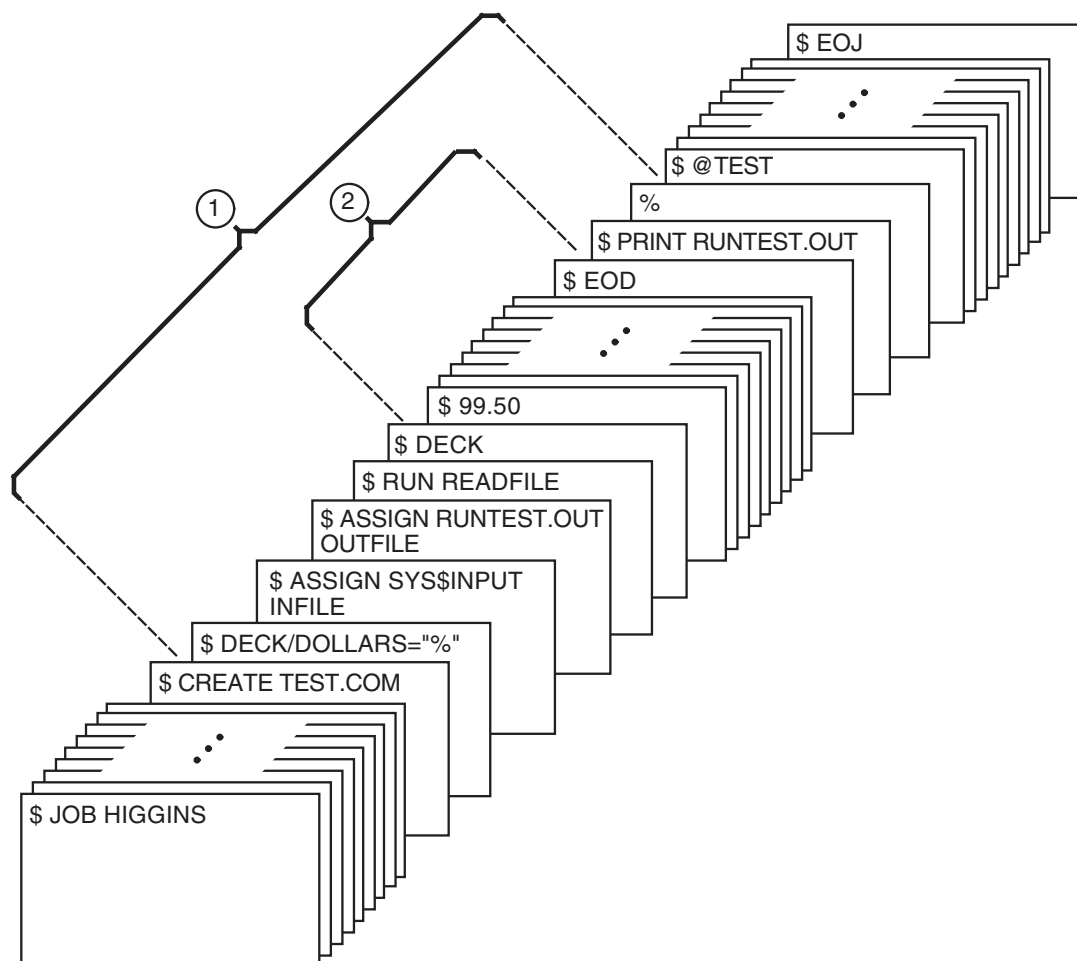
例

1.



この例の Fortran および LINK コマンドは、プログラム A をコンパイルしリンクします。プログラム A が実行されると、論理装置 SYS\$INPUT からプログラム A が読み込んだすべてのデータは、コマンド・ストリームから読み込まれたデータです。DECK コマンドは、入力ストリームのレコードの最初の文字位置 (カラム 1) にドル記号を入れられることを示しています。EOD コマンドは、データの終端 (end-of-file) を示します。

2.



- ① CREATE コマンドの入カストリーム
- ② プログラム READFILE の入カストリーム

JRD-0784-GE

この例の CREATE コマンドは、入カストリームに入カされた行からコマンド・
 プロシージャ・ファイル TEST.COM を作成します。DECL/DOLLARS コマンド
 は、パーセント記号(%)が CREATE コマンドの EOF 指示子であることを示して
 います。これにより文字列 \$EOD は入カレコードとして読み込まれ、RUN コマ
 ンドに対する入カの終わりを示します。

DEFINE

論理名に等価名を対応させます。

フォーマット

DEFINE 論理名 等価名[,...]

パラメータ

論理名

論理名文字列を指定します。論理名文字列には、1 文字から 255 文字までを含むことができます。次の規則が適用されます。

- 論理名が、プロセスまたはシステムの論理名ディレクトリ・テーブル (LNM\$SYSTEM_DIRECTORY および LNM\$PROCESS_DIRECTORY) に入力された場合、論理名は 1 文字から 31 文字までの英数字 (ドル記号(\$)) とアンダースコア(_)文字を含む) だけで構成されます。論理名が論理名テーブル名に変換される場合は、名前の中の英文字はすべて大文字である必要があります。
- 論理名の最後にコロン(:)を指定した場合、DEFINE コマンドは、そのコロンを論理名の一部として保存します。(この点は、ASSIGN コマンドと異なっています。ASSIGN コマンドでは、論理名テーブルに名前を登録する前に、コロンが削除されます。)省略時の設定では、論理名はプロセス論理名テーブルに登録されます。
- 文字列に英数字、ドル記号、またはアンダースコア以外の文字が含まれている場合には、その文字列を引用符(" ")で囲みます。論理名に引用符が含まれる場合には、論理名全体を引用符で囲み、論理名の中で引用符(")が必要な位置に連続する 2 つの二重引用符("")を指定します。論理名を引用符で囲む場合には、英字の大文字と小文字の区別もそのまま保存されます。

等価名[,...]

1 文字から 255 文字までの文字列を指定します。

- 文字列に英数字、ドル記号、またはアンダースコア以外の文字が含まれる場合には、文字列全体を引用符で囲まなければなりません。等価文字列に引用符が含まれる場合には、文字列全体を引用符で囲み、引用符(")が必要な位置に連続する 2 つの二重引用符("")を指定します。2 つ以上の等価名を指定すれば、サーチ・リストを作成できます。1 つの論理名は、最大 128 個までの等価名を持つことができます。

- ファイル指定として使用される等価名を指定する場合には、その等価名が直接使用されるときに必要な区切り文字(コロン、かぎ括弧、ピリオド)も含まなければなりません。したがって、等価名として装置名を指定する場合には、等価名の最後にコロンを指定します。

DEFINE コマンドでは、同じ論理名を複数の等価名に割り当てることができます。たとえば、同じ論理名を使用して異なるディスクの異なるディレクトリを、アクセスすることができ、また異なるディレクトリの異なるファイルをアクセスできます。

説明

DEFINE コマンドは、1つまたは複数の等価名を表わす論理名を定義して、論理名テーブル内のエントリを作成します。等価名には、装置名、別の論理名、ファイル指定、または他の文字列を使用できます。

論理名の使用を、特定のプロセス、特定のジョブ、特定のグループに制限することができます。また、論理名をシステム全体あるいは OpenVMS Cluster システム全体で使用することもできます。これは、論理名が作成されるテーブルに依存します。テーブルを指定するには、/PROCESS、/JOB、/GROUP、/SYSTEM、/TABLE の修飾子のいずれかを使用します。

最初の4つの修飾子は、それぞれプロセス、ジョブ、グループ、システムの論理名テーブルを表わしています。/TABLE 修飾子は、任意のタイプのテーブルを指定するために使用されます。/TABLE 修飾子は、クラスタ全体の論理名テーブルを指定する場合に使用される唯一のものです。

/PROCESS、/JOB、/GROUP、/SYSTEM、/TABLE の修飾子の複数を指定した場合は、最後に指定した修飾子だけが有効となります。これらの修飾子のいずれも指定しない場合は、論理名はユーザのプロセス論理名テーブルに追加されます。

作成する論理名のアクセス・モードを指定するには、/USER_MODE、/SUPERVISOR_MODE、または/EXECUTIVE_MODE 修飾子を使用します。複数の修飾子を指定した場合は、最後に指定した修飾子だけが有効です。アクセス・モードを指定しないと、スーパーバイザ・モード名が作成されます。名前を入れるテーブルと同じモードまたは外側のモードで、論理名を作成できます(利用者モードが一番外側のモードで、エグゼクティブ・モードが一番内側のモードです)。

それぞれの論理名のアクセス・モードが異なる限り、同じ論理名テーブルに同じ名前を持つ複数の論理名をいれることができます(ただし、テーブル内の既存の論理名が NO_ALIAS 属性を持つ場合は、このテーブルで同じ名前を使用して外側のモードの論理名を作成できません)。

既存の名前と同じモードで、同じテーブルに同じ名前を持つ論理名を作成すると、新しい論理名で既存の論理名が置き換えられます。

ASSIGN コマンドを使用して、論理名を作成することもできます。テーブルから論理名を削除するには、DEASSIGN コマンドを使用します。

注意

SYS\$SYSTEM: 内の実行可能イメージのファイル名と同じ論理名を割り当てないでください。そのイメージが起動できなくなります。

等価名を指定しない(すなわちインデックスを指定しない)で論理名を作成する場合は、\$CRELNM システム・サービスを使用します。

等価名として ODS-5 ファイル名を指定する場合は、『OpenVMS システム管理者マニュアル(上巻)』を参照してください。

アプリケーションでの使用を除く、論理名および論理名テーブルについての完全な説明については、『OpenVMS ユーザーズ・マニュアル』を参照してください。アプリケーションでの論理名の使用については、『OpenVMS Programming Concepts Manual』を参照してください。クラスタ全体の論理名の管理については、『OpenVMS Cluster システム』を参照してください。また、論理名の変換に使用されるレキシカル関数 F\$TRNLNM についての本書の説明も参照してください。

修飾子

/CLUSTER_SYSTEM

この修飾子を使用するためには、SYSTEM アカウントでログインするか、SYSNAM (システム論理名) または SYSPRV (システム) 特権を持っている必要があります。

LNMSYSCLUSTER テーブルにクラスタ・ワイド論理名を定義します。

/EXECUTIVE_MODE

エグゼクティブ・モードの論理名を作成するためには、SYSNAM (システム論理名) 特権が必要です。

指定したテーブルに、エグゼクティブ・モードの論理名を作成します。

/EXECUTIVE_MODE 修飾子を指定しても、SYSNAM 特権を持たない場合には、DEFINE コマンドはその修飾子を無視し、スーパーバイザ・モードの論理名を作成します。論理名のアクセス・モードは、論理名を登録するテーブルのモードと同じか、それより低いものでなければなりません。

/GROUP

グループ論理名テーブルに論理名を登録するためには、GRPNAM (グループ論理名) 特権または SYSNAM (システム論理名) 特権が必要です。

DEFINE

論理名をグループ論理名テーブルに登録します。登録者の UIC (利用者識別コード) のグループ番号が同じである他の利用者は、この論理名をアクセスできます。/GROUP 修飾子は、/TABLE=LNМ\$GROUP 修飾子の同意語です。

/JOB

論理名を、ジョブ論理名テーブルに登録します。この論理名を作成したプロセスと同じジョブ階層構造に含まれるプロセスはすべて、この論理名をアクセスできます。/JOB 修飾子は、/TABLE=LNМ\$JOB 修飾子の同意語です。

/LOG (省略時の設定)

/NOLOG

既存の名前を無効にする論理名の定義時に、メッセージを表示するか否かを制御します。

/NAME_ATTRIBUTES[=(キーワード[,...])]

論理名に対して、属性を指定します。省略時の設定では、属性は設定されません。属性に対して指定できるキーワードは、次のとおりです。

CONFINE 論理名が、SPAWN コマンドによって生成されたサブプロセスにコピーされないことを指定します。このキーワードは、プロセス固有のテーブルに論理名を作成する場合にだけ意味をもちます。

CONFINE 属性は、論理名が登録される論理名テーブルの属性からも与えられます。論理名テーブルが "CONFINE" 属性を持つ場合には、そのテーブルに含まれる論理名もすべて "CONFINE" 属性をもちます。

NO_ALIAS より低い特権の（外側の）アクセス・モードでは、同じ名前の論理名をこのテーブルに登録できないことを指定します。同じ名前を持つ他の論理名が、より低い特権のアクセス・モードで、このテーブルにすでに登録されている場合には、その名前は削除されます。

キーワードを 1 つしか指定しない場合には、括弧を省略できます。また、指定した属性だけが設定されます。

/PROCESS (省略時の設定)

論理名を、プロセス論理名テーブルに登録します。/PROCESS 修飾子は、/TABLE=LNМ\$PROCESS 修飾子の同意語です。

/SUPERVISOR_MODE (省略時の設定)

指定したテーブルに、スーパーバイザ・モードの論理名を作成します。論理名のアクセス・モードは、論理名を登録するテーブルのモードと同じか、それより低いものでなければなりません。

/SYSTEM

システム論理名テーブルに名前を登録するためには、書き込み (W) アクセス権または SYSNAM (システム論理名) 特権が必要です。

論理名をシステム論理名テーブルに登録します。システム上のすべてのユーザが、この論理名をアクセスできます。/SYSTEM 修飾子は、/TABLE=LNМ\$SYSTEM 修飾子の同意語です。

/TABLE=テーブル名

共有可能な論理名テーブルの名前を指定するには、テーブルへの書き込み (W) アクセス権が必要です。

論理名を登録する論理名テーブルの名前を指定します。/TABLE 修飾子を使用すれば、利用者定義論理名テーブル (CREATE/NAME_TABLE コマンドによって作成されたもの) を指定できます。また、プロセス論理名テーブル、ジョブ論理名テーブル、グループ論理名テーブル、システム論理名テーブル、クラスタ論理名テーブルと、プロセスまたはシステムの論理名ディレクトリ・テーブルを指定できます。

複数の等価文字列が与えられている論理名を使用して、テーブル名を指定する場合には、論理名は最初に検出されたテーブルに登録されます。たとえば、DEFINE /TABLE=LNМ\$FILE_DEV を指定し、LNМ\$FILE_DEV が LNМ\$PROCESS、LNМ\$JOB、LNМ\$GROUP、および LNМ\$SYSTEM と等しいと定義されている場合には、その論理名は LNМ\$PROCESS に登録されます。

省略時の設定では、/TABLE=LNМ\$PROCESS が使用されます。

/TRANSLATION_ATTRIBUTES[=(キーワード[,...])]

等価名修飾子。

論理名の等価文字列に、1 つまたは複数の属性を指定します。変換属性に対して指定できるキーワードは、次のとおりです。

CONCEALED	等価文字列が隠し装置名であることを示します。隠し装置名を定義した場合、システムは、その装置を参照するメッセージの中で、等価文字列ではなく、論理名を表示します。
TERMINAL	等価文字列が、反復変換されないことを示します。したがって、論理名変換は、現在の等価文字列の変換で終了しなければなりません。

キーワードを 1 つしか指定しない場合には、括弧を省略できます。また、指定した属性だけが設定されます。

1 つの論理名に対して複数の等価文字列が存在する場合、各等価文字列は、それぞれ異なる変換属性を持つことができます。

/USER_MODE

指定したテーブルに、ユーザ・モードの論理名を作成します。

プロセス論理名テーブル (1 つまたは複数) に作成されるユーザ・モードの論理名は、1 つのイメージの実行のためだけに使用されます。たとえば、コマンド・プロシージャ内で、実行中のイメージが SYS\$INPUT を再定義できるように、ユーザ・モードの論理名を作成できます。ユーザ・モードの論理名は、プロセス内で実行されているイメージが終了する時 (つまり、イメージを実行している DCL コマンドまたは利用者のプログラムが実行を終了した後で)、プロセス論理名テーブルから削除されます。また、ユーザ・モードの論理名は、コマンド・プロシージャを起動および実行している時に自動的に削除されます。

例

1. `$ DEFINE/USER_MODE TM1 $DISK1:[ACCOUNTS.MEMOS]WATER.TXT`

この例では、DEFINE コマンドで TM1 をファイル指定に等価なものとして定義しています。次のイメージ実行後に、論理名 TM1 は自動的に削除されます。

2. `$ DEFINE CHARLIE XXX1:[CHARLES]`
`$ PRINT CHARLIE:TEST.DAT`
 Job 274 entered on queue SYS\$PRINT

この例では、DEFINE で論理名 CHARLIE を、ディスク XXX1 のディレクトリ名[CHARLES]に関連付けています。PRINT コマンドは、ファイル XXX1:[CHARLES]TEST.DAT のコピーをシステム・プリンタのキューに登録します。

3. `$ DEFINE PROCESS_NAME LIBRA`
`$ RUN WAKE`

この DEFINE コマンドは、PROCESS_NAME という論理名をプロセス論理名テーブルに登録し、LIBRA という等価名に割り当てます。この論理名は、スーパーバイザ・モードで作成されます。WAKE というプログラムは、PROCESS_NAME という論理名を変換し、LIBRA という名前のプロセスに対して特殊な動作を実行します。

4. `$ DEFINE TEMP: XXX1:`
`.`
`.`
`.`
`$ DEASSIGN TEMP::`

この DEFINE コマンドは、TEMP: という論理名に XXX1: という等価文字列を割り当て、この論理名をプロセス論理名テーブルに登録します。コロンは、論理名の一部として扱われます。DEASSIGN コマンドは、論理名を削除します。DEASSIGN コマンドでは、この論理名を指定するために 2 つのコロンが必要です。1 つのコロンが、DEASSIGN コマンドによって削除されるためです。したがって、もう 1 つのコロンは論理名の一部として処理されます。

5. `$ DEFINE PORTLAND PRTLND::YYY0:[DECNET.DEMO.COM]`

この DEFINE コマンドは、PRTLND::YYY0:[DECNET.DEMO.COM] という等価文字列を使用して、プロセス論理名テーブルに PORTLAND という論理名に登録します。その後の論理名 PORTLAND の参照では、指定されたノード、ディスク、サブディレクトリが参照されます。

6. `$ DEFINE LOCAL "BOSTON"JAY_SABLE JKS":::"`

この DEFINE コマンドは、BOSTON"JAY_SABLE JKS"::: というリモート・ノード等価名を使用して、プロセス論理名テーブルに LOCAL という論理名に登録します。現在のノードでの DCL コマンド文字列処理に関する規則に従って、3 組の引用符を使用しています。引用符によって、等価名に含まれるアクセス制御情報が 1 組の引用符で囲まれるようにしています。

7. `$ DEFINE MYDISK XXX0:[MYDIR], YYY0:[TESTDIR]`

この例では、論理名 MYDISK を XXX0:[MYDIR] と YYY0:[TESTDIR] のサーチ・リストとして、プロセス論理名テーブルに登録しています。

8. `$ DEFINE/TABLE=LNMS$CLUSTER_TABLE FIRENZE FIRENZE::FIESOLE:[ETRUSCAN]`

この例では、DEFINE コマンドが FIRENZE をディレクトリ指定 FIRENZE::FIESOLE:[ETRUSCAN] に対応させ、新しい論理名 (FIRENZE) およびその等価文字列 (FIRENZE::FIESOLE:[ETRUSCAN]) を省略時のクラスタ全体のテーブルに置きます。新しい論理名は、クラスタ内のすべてのノードに自動的に通知されます。

9. `$ CREATE/NAME_TABLE TABLE1`
`$ DEFINE/TABLE=LNMS$PROCESS_DIRECTORY LNM$FILE_DEV -`
`_ $ TABLE1, LNM$PROCESS, LNM$JOB, LNM$GROUP, LNM$SYSTEM`
`$ DEFINE/TABLE=TABLE1 -`
`_ $ /TRANSLATION_ATTRIBUTES=CONCEALED WORK_DISK DKA1:`

この例では、プロセス固有の論理名テーブル TABLE1 を作成しています。

最初の DEFINE コマンドは、TABLE1 が装置名やファイル指定の検索時に最初に検索されるように、LNMS\$FILE_DEV を定義しています。これは、TABLE1 が論理名 LNM\$FILE_DEV の等価名の最初の項目だからです。論理名 LNM\$FILE_DEV は、装置やファイル指定の検索時には常に、論理名テーブルの検索順序の省略時設定を決定します。

次の DEFINE コマンドは、論理名 WORK_DISK を物理装置 DKA1 の等価名として TABLE1 に登録しています。論理名 WORK_DISK は、隠し装置名の属性を与えられているので、システム・メッセージにもそのまま変換されずに表示されます。

10. `$ CREATE/NAME_TABLE SPECIAL`
`$ DEFINE/TABLE=LNMS$PROCESS_DIRECTORY LNM$FILE_DEV -`
`_ $ SPECIAL, LNM$PROCESS, LNM$JOB, LNM$GROUP, LNM$SYSTEM`
`$ DEFINE/TABLE=LNMS$PROCESS_DIRECTORY TAB SPECIAL`
`$ DEFINE/TABLE=TAB REPORT [CHELSEA]STORES`
`$ SHOW LOGICAL/TABLE=SPECIAL REPORT`
`"REPORT" = "[CHELSEA]STORES" (SPECIAL)`

この例では、CREATE/NAME_TABLE コマンドで SPECIAL という論理名テーブルを作成しています。このテーブルは、プロセス・ディレクトリ LNM\$PROCESS_DIRECTORY に登録されます。

最初の DEFINE コマンドは、SPECIAL が装置名やファイル指定の検索時に最初に検索されるように、LNMS\$FILE_DEV を定義しています。これは、SPECIAL が論理名 LNM\$FILE_DEV の等価名の最初の項目だからです。論理名 LNM\$FILE_DEV は、装置やファイル指定の検索時には常に、論理名テーブルの検索順序の省略時設定を決定します。論理名 LNM\$FILE_DEV は、プロセス・ディレクトリ LNM\$PROCESS_DIRECTORY に登録しています。

DEFINE

次の DEFINE コマンドでは、論理名 TAB を定義しています。TAB は、論理名テーブルを示す SPECIAL という文字列に変換されます。TAB は論理名テーブルに繰り返し変換されるので、プロセス・ディレクトリに登録しなければなりません。

次に、論理名 REPORT が、論理名テーブル TAB に登録されています。TAB は、テーブル SPECIAL に翻訳されるので、論理名 REPORT はテーブル SPECIAL に登録されています。SHOW LOGICAL コマンドで、論理名 REPORT がどこに登録されたか確認できます。

TAB は、別のテーブルを指すように再定義できるので、テーブル名 TAB を使用する別のプログラムを走らせる場合に、実際のテーブルを別のテーブルに変更することもできます。

DEFINE/CHARACTERISTIC

キュー属性に数値を割り当てます。/CHARACTERISTIC 修飾子は、省略できません。すでに属性に数値が定義されている場合には、既存の属性の割り当てを変更するには、その属性の定義を削除して再定義しなければなりません。

OPER（オペレータ）特権が必要です。

注意

属性番号 1 つに 1 つの属性名のみ定義できます。

フォーマット

DEFINE/CHARACTERISTIC 属性名 属性番号

パラメータ

属性名

定義する属性に名前を割り当てます。属性名は、既存の属性名であるか、1文字から31文字までの長さの新しい属性名です。アルファベットの大文字と小文字、数字、ドル記号(\$)、およびアンダースコア(_)を含むことができます。ただし、属性名には、アルファベットを少なくとも1文字は含まなければなりません。各数値に対して定義できる属性名は1つだけです。

属性番号

定義する属性に、番号を割り当てます。番号は、0 から 127 までの範囲です。

説明

システム管理者またはオペレータは、DEFINE/CHARACTERISTIC コマンドを使用して、システムのキューの特定の特性に名前と番号を割り当てます。特性は、環境に対して意味を持つ印刷ジョブまたはバッチ・ジョブの任意の属性を参照できます。特性の名前と番号は任意ですが、その特性に対して一意でなければなりません。

注意

V6.0 より前のバージョンの OpenVMS は、DEFINE/CHARACTERISTIC コマンドを使用すると、複数の特性名を 1 つの番号に定義できました。ただし、この機能はサポートされていませんでした。

DEFINE/CHARACTERISTIC コマンドでは、複数の特性名を 1 つの番号に定義できなくなりました。ただし、キューの構成によって単一の番号に複数の特性名を定義する必要がある場合、論理名を定義することで同じ結果を得ることができます。たとえば、次のようなコマンドを入力できます。

```
$ DEFINE/CHARACTERISTIC SECOND_FLOOR 2
$ DEFINE/SYSTEM/EXECUTIVE_MODE SALES_FLOOR SECOND_FLOOR
$ DEFINE/SYSTEM/EXECUTIVE_MODE SALES_DEPT SECOND_FLOOR
```

この例では、特性名 SECOND_FLOOR が、特性番号 2 に割り当てられます。次に、論理名 SALES_FLOOR と SALES_DEPT が、特性名 SECOND_FLOOR と等価であると定義されます。その結果、論理名 SALES_FLOOR と SALES_DEPT が、それぞれで特性名 SECOND_FLOOR と特性番号 2 と等価になります。これらの論理名は、任意の/CHARACTERISTIC=属性名修飾子に対して、属性名の値として指定できます。

OpenVMS Cluster 環境では、論理名が必要なすべてのノードで、論理名を定義しなければなりません。

特性を定義すると、印刷ジョブまたはバッチ・ジョブ、および実行キューに対応付けることができます。ジョブに特性を指定する場合についての詳細は、PRINT および SUBMIT コマンドの/CHARACTERISTICS 修飾子の説明を参照してください。

現在システムに定義されている特性を調べるには、SHOW QUEUE /CHARACTERISTICS コマンドを使用します。特定のキューに指定されている特性を調べるには、SHOW QUEUE/FULL コマンドを使用します。キューに特性を対応付ける場合についての詳細は、INITIALIZE/QUEUE、SET QUEUE、および START/QUEUE コマンドの/CHARACTERISTICS 修飾子の説明を参照してください。

DELETE/CHARACTERISTIC コマンドは、定義済みの特性を削除します。

キュー特性を指定する場合についての詳細は、『OpenVMS システム管理者マニュアル』を参照してください。

例

1. \$ DEFINE/CHARACTERISTIC REDINK 3

この例では、属性 REDINK を属性番号 3 に定義しています。PRINT /CHARACTERISTICS=REDINK (または PRINT/CHARACTERISTICS=3) コマンド実行時には、プリンタ・キューが REDINK または 3 の属性を定義されている場合にのみジョブは出力されます。

DEFINE/FORM

プリンタのフォーム名に対して、フォーム番号と属性を定義します。/FORM 修飾子は省略できません。フォーム名または番号を変更する場合には、一度削除してから再定義しなければなりません。DEFINE/FORM 修飾子の値のみを変える場合には、フォーム名と番号が同じであれば、DEFINE/FORM コマンドで新しい値を指定すれば変更できます。

OPER（オペレータ）特権が必要です。

フォーマット

DEFINE/FORM フォーム名 フォーム番号

パラメータ

フォーム名

定義するフォームに名前を割り当てます。フォーム名は、既存のフォーム・タイプか、または1文字から31文字までの長さの文字列です。文字列には、アルファベットの太文字と小文字、数字、ドル記号(\$)、およびアンダースコア(_)を含むことができます。ただし、名前にはアルファベット文字が少なくとも1文字含まれていなければなりません。

フォーム番号

定義するフォームに、0～9999の範囲の番号を割り当てます。省略時の DEFAULT フォーム、つまりシステムのブートストラップ時に自動的に定義されるフォームには、0という番号が割り当てられます。

説明

システム管理者または、オペレータは、DEFINE/FORM コマンドを使用して、プリンタ・キューまたは端末キューで使用するために、用紙の種類と印刷領域に名前と番号を割り当てます。新しいキュー・ファイルを作成した場合、システムは、フォーム番号が0で、すべて属性が省略時の設定になっている DEFAULT フォームを定義します。

一部の DEFINE/FORM 修飾子は、印刷領域を指定します。/MARGIN 修飾子と /WIDTH 修飾子の LEFT オプションと RIGHT オプションは、1行当たり文字数を決定します。/MARGIN 修飾子と /WIDTH 修飾子の RIGHT オプションを使用すると、テキスト行を折り返す位置を指定できます（ただし、テキストを埋めるため、またはフ

フォーマッティングのために、/MARGIN 修飾子と/WIDTH 修飾子の LEFT オプションと RIGHT オプションを使用することはできません。

DEFINE/FORM コマンドを使用して、異なる用紙の種類を指定することもできます。/DESCRIPTION 修飾子を使用すると、フォーム名をより完全に記述できます。

フォームを定義すると、プリント・ジョブおよび出力実行キューと対応付けることができます。ジョブにフォームを指定する場合についての詳細は、PRINT/FORM コマンドの説明を参照してください。

システムに定義されているフォームを調べるには、SHOW QUEUE/FORM コマンドを使用します。特定のキューに現在マウントされているフォームと、そのキューの省略時のフォームとして指定されているフォームを調べるには、SHOW QUEUE/FULL コマンドを使用します。キューにフォームを対応付ける場合についての詳細は、INITIALIZE/QUEUE、SET QUEUE、および START/QUEUE コマンドの/DEFAULT および/FORM_MOUNTED 修飾子の説明を参照してください。

プリント・ジョブを制御するためにフォームを使用する方法についての詳細は、『OpenVMS システム管理者マニュアル』を参照してください。

修飾子

/DESCRIPTION=文字列

最大 255 文字の文字列を指定します。この文字列はフォームに関する情報をオペレータに提供するために使用されます。省略時の文字列は、指定されたフォーム名です。

ここで指定される文字列は、フォーム・タイプを詳しく定義するために使用します。たとえば、LETTER1、LETTER2、および LETTER3 というフォーム名を定義している場合には、/DESCRIPTION 修飾子を使用することで LETTER1 は標準的なレター用紙 (8.5 × 11 インチ) であり、LETTER2 は小さなレター用紙 (6 × 9 インチ) であり、LETTER3 は社長個人用のレター用紙であることを、利用者およびオペレータに知らせることができます。

小文字、空白、または他の非アルファベット文字を含む文字列は、二重引用符 (" ") で囲みます。

/LENGTH=n

フォーム・ページの 1 ページの物理的な長さを、行数で指定します。省略時設定のページ長は、66 行です。これは、標準的なページの長さが 11 インチであり、1 インチに 6 行が印字されると考えた場合の値です。パラメータ n は、1 から 255 までの正の整数です。

プリント・シンピオントは、装置の 1 ページの長さをこの値に設定します。これは、機械的な改ページ機能のないプリンタに対して改ページ (フォーム・フィード) を行う際に、何行の空白行を出力するかを計算するのに用いられます。

/MARGIN=(オプション[,...])

BOTTOM, LEFT, RIGHT, および TOP という 4 つの余白オプションの中から、1 つまたは複数を指定します。

BOTTOM=n	1 ページにおいて、印刷された最後の部分から物理的なページの (紙の) 最後までに残す空白行の行数を指定します。n の値は正の整数であり、/LENGTH 修飾子の値より小さな値でなければなりません。省略時の値は 6 であり、これは一般に用紙の下余白が 1 インチであることを示します。
LEFT=n	印字可能な左端の位置と実際の印字領域の間に、余白として残す桁数を指定します。n の値は正の整数であり、/WIDTH 修飾子の値より小さな値でなければなりません。省略時の値は 0 であり、これは実際の印字領域がプリンタで印字できる用紙の左端から始まることを示します。
RIGHT=n	/WIDTH 修飾子の設定と実際の印字領域の間に、余白として残す桁数を指定します。n の値は正の整数であり、/WIDTH 修飾子より小さい値でなければなりません。RIGHT オプションの値は、/WIDTH の値から左側へ順にカウントされます。省略時の値は 0 であり、これは /WIDTH の値の桁位置まで印字されることを示します。
TOP=n	用紙上の物理的なページの上端と実際に印字される領域の上端との間に残す、空白行の行数を指定します。n の値は正の整数で、/LENGTH 修飾子の値までの範囲です。省略時の値は 0 で、これは用紙の上余白が作成されないことを示します。

/PAGE_SETUP=(モジュール[,...])

/NOPAGE_SETUP (省略時の設定)

各ページの前に、装置を設定する 1 つまたは複数のモジュールを指定します。モジュールは、装置制御ライブラリに登録されていなければなりません。フォームがマウントされている間は、システムは各ページを印刷する前に、装置制御ライブラリから指定されたモジュールを取り出し、そのモジュールをプリンタにコピーします。

/SETUP=(モジュール[,...])

各ファイルの印刷開始時に装置を適切にセット・アップする、1 つまたは複数のモジュールを指定します。モジュールは、装置制御ライブラリに登録されているものです。フォームがマウントされている間は、システムは各ファイルを印刷する前に、装置制御ライブラリから指定されたモジュールを取りだし、そのモジュールをプリンタにコピーします。

装置制御モジュールについての詳細は、『OpenVMS システム管理者マニュアル』の "Batch and Print Operations" の章を参照してください。

/SHEET_FEED

/NOSHEET_FEED (省略時の設定)

各物理ページの最後で、プリント・ジョブを一時停止し、新しい用紙を挿入できるようにすることを指定します。

/STOCK=文字列

フォームに対応する用紙の種類 (ストック) を指定します。/STOCK 修飾子に指定できる文字列パラメータは、1 文字から 31 文字までの長さであり、ドル記号、アンダースコア、およびすべての英数字を含むことができます。/STOCK 修飾子を指定する場合には、フォームに対応させるストックの名前を指定しなければなりません。/STOCK 修飾子を指定しない場合には、ストック名はフォーム名と同じになります。

どのような文字列でも指定できますが、同じ種類の用紙を使用するフォームを作成する場合には、同じ種類の用紙を参照する DEFINE/FORM コマンドで、/STOCK に対する文字列がすべて同じになるようにしてください。

同じ種類の用紙を複数のフォームが使用し、それらフォームでは余白指定や自動改行の位置、あるいはページ・サイズなどが各用紙ごとに異なる場合に、この修飾子は便利です。これらのフォームの1つを要求するジョブは、同じキューで印刷されます。フォームに対応したストック文字列の変更は、そのフォームがそのジョブまたはキューでも参照されていない場合にのみ、行うことができます。

/TRUNCATE (省略時の設定)
/NOTRUNCATE

現在の行の長さ (/WIDTH または/MARGIN=RIGHT 修飾子で指定された長さ) を越える文字を切り捨てます。/TRUNCATE 修飾子を指定する場合には、/WRAP 修飾子は指定できません。/NOTRUNCATE 修飾子と/NOWRAP 修飾子のどちらも指定した場合には、プリンタは、可能な位置までなるべく多くの文字を1行に印字します。この修飾子の組み合わせは、ある種のグラフィックス出力に対して効果的です。

/WIDTH=n

用紙上の桁位置または文字の位置によって、用紙の物理的な幅を指定します。パラメータ n は、1 から 65,535 までの正の整数で、省略時の値は 132 です。

この値を越える行は、/WRAP 修飾子が有効であれば自動改行して次の行に印刷され、/TRUNCATE 修飾子が有効であれば切り捨てられます (/NOTRUNCATE と/NOWRAP 修飾子の両方が有効な場合には、可能な位置まで行は印刷されます)。

行の改行(ラップ)を判断する時には、/MARGIN=RIGHT 修飾子の指定が/WIDTH 修飾子に優先されます。

/WRAP
/NOWRAP (省略時の設定)

現在の行の長さ (/WIDTH または/MARGIN=RIGHT 修飾子で指定された長さ) を越える文字を、次の行に改行して印刷(ラップ)します。/WRAP 修飾子を指定する場合には、/TRUNCATE 修飾子は指定できません。/NOWRAP 修飾子と/NOTRUNCATE 修飾子のどちらも指定した場合には、プリンタは、可能な位置までなるべく多くの文字を1行に印字します。この修飾子の組み合わせは、ある種のグラフィックス出力に対して効果的です。

例

1. \$ DEFINE/FORM /MARGIN=(TOP=6,LEFT=10) CENTER 3

この例では、上余白6行で左余白10桁のフォーム CENTER を定義しています。下余白(6行)と右余白(10桁)は、省略時設定のままです。また、フォームは番号3に割り当てられます。

DEFINE/KEY

等価文字列およびいくつかの属性を、ターミナル・キーボードのキーに割り当てます。

フォーマット

DEFINE/KEY キー名 等価文字列

パラメータ

キー名

定義するキーの名前を指定します。VT52 ターミナルの定義可能なキーは、すべて数値キーパッドにあります。VT100 シリーズのターミナルでは、数値キーパッドのすべてのキーと 4 つの矢印キーを定義できます。また、LK201 キーボードを備えたターミナルでは、以下の 3 種類のキーを定義できます。

- 数値キーパッドのキー
- 編集キーパッドのキー（上矢印と下矢印キーを除く）
- ターミナルの一番上のファンクション・キー（F1 から F5 キーを除く）

次に示す表は、最初のカラムにキー名が示されています。残りの 3 つのカラムは、キ一定義が可能な 3 種類のターミナル・キーボード上でのキーの名称を示したものです。

キー名	LK201	VT100 シリーズ	VT52
PF1	PF1	PF1	[青色]
PF2	PF2	PF2	[赤色]
PF3	PF3	PF3	[灰色]
PF4	PF4	PF4	- -
KP0, KP1, ..., KP9	0, 1, ..., 9	0, 1, ..., 9	0, 1, ..., 9
Period	.	.	.
Comma	,	,	(なし)
Minus	-	-	(なし)
Enter	Enter	ENTER	ENTER
Left	←	←	←
Right	→	→	→
Find (E1)	Find	—	—

キー名	LK201	VT100 シリーズ	VT52
Insert Here (E2)	Insert Here	—	—
Remove (E3)	Remove	—	—
Select (E4)	Select	—	—
Prev Screen (E5)	Prev Screen	—	—
Next Screen (E6)	Next Screen	—	—
Help	Help	—	—
Do	Do	—	—
F6, F7, ..., F20	F6, F7, ..., F20	—	—

定義可能なキーの中には、常に定義できるキーがあります。また、KP0 から KP9 までのキーや、PERIOD、COMMA および MINUS のように、定義できるように前もって設定しなければならないキーもあります。これらのキーを使用する場合には、前もって、SET TERMINAL/APPLICATION コマンド、または SET TERMINAL/NONUMERIC コマンドを入力しなければなりません。

LK201 キーボードでは、上および下矢印キーと、F1 から F5 までのキーに定義することはできません。右および左矢印キーと、F6 から F14 までのキーは、コマンド行の編集のために予約されています。これらのキーを定義する前には、SET TERMINAL/NOLINE_EDITING コマンドを入力しなくてはなりません。また、CTRL/V を押せば F7 から F14 のキーを使用可能にすることができます。F6 は CTRL/V で使用可能にはならないので注意してください。

等価文字列

キーを押した時に処理される文字列を指定します。文字列にスペースや小文字が含まれる場合には、等価文字列を引用符 (" ") で囲まなければなりません。

説明

DEFINE/KEY コマンドを使用すると、特定の端末のキーに定義を割り当てることができます。端末には、VT52、VT100 シリーズ、および LK201 キーボードの端末があります。

これらの端末のキーパッドにあるキーを定義するには、まず SET TERMINAL/APPLICATION または SET TERMINAL/NONUMERIC コマンドを入力します。このように端末を設定した場合、システムはキーパッド・キーからのキーストロークを異なる方法で解釈します。たとえば、SET TERMINAL/NONUMERIC が有効になっていると、キーパッドの 1 キーを押してもシステムに文字“1”は送信されません。

等価文字列定義には、各種の情報を入れることができます。通常、等価文字列定義は、DCL コマンドで構成します。たとえば、0 キーに SHOW TIME コマンドを割り当てることができます。0 を押すと、システムは、現在の日付と時刻を表示します。

他の定義は、コマンド行に追加するテキスト文字列で構成することができます。テキスト文字列を挿入するようにキーを定義する場合、文字列を挿入した後にさらにデータを入力できるように /NOTERMINATE 修飾子を使用します。

多くの場合、エコー機能を使用したいことがあります。省略時の設定は、/ECHO です。/ECHO が設定されている場合、キーを押すたびに画面にそのキーの定義が表示されます。

/STATE 修飾子を使用すると、端末で利用できるキー定義の数を増やすことができます。定義ごとに対応する状態が異なる限り、同じキーに任意の数の定義を割り当てることができます。状態の名前には、任意の英数字、ドル記号、およびアンダスコアを使用することができます。状態に対応する定義の種類を記憶しやすいように、できるだけ覚えやすく入力しやすい状態名を作成してください。たとえば、SETSHOW という状態を作成できます。この状態のキー定義は、各種の DCL SET および SHOW コマンドを参照できます。EDT エディタに慣れている場合は、状態を GOLD として定義できます。次に、/IF_STATE 修飾子を使用して、GOLD として定義されたキーと組み合わせて使用するキーに異なる定義を割り当てることができます。

SET KEY コマンドは、キーパッド状態を変更します。キーの定義と状態を表示するには、SHOW KEY コマンドを使用します。

修飾子

/ECHO (省略時の設定)

/NOECHO

キーを押した後、等価文字列が画面に表示されるかどうかを指定します。/NOECHO 修飾子を /NOTERMINATE 修飾子とともに使用することはできません。

/ERASE

/NOERASE (省略時の設定)

キーを押す、等価文字列がスクリーンに表示される際、同じコマンド行にすでに入力されていた文字列が消去されるかどうかを指定します。

/IF_STATE=(状態名,...)

/NOIF_STATE

1 つまたは複数の状態を指定します。その状態の中の 1 つは、キー定義に対して有効でなければなりません。/NOIF_STATE 修飾子は、/IF_STATE=現在の状態の指定と同じ意味です。状態名は英数字文字列です。状態は、/SET_STATE 修飾子または SET KEY コマンドによって設定されます。状態名を 1 つしか指定しない場合には、括弧を省略できます。複数の状態名を指定することにより、指定したすべての状態で同じ機能を持つキーを定義できます。

/LOCK_STATE

/NOLOCK_STATE (省略時の設定)

/SET_STATE 修飾子によって設定した状態を、他の状態に明示的に変更されるまで有効のままにすることを指定します (省略時の設定では、/SET_STATE 修飾子は、次に押された定義可能キーに対してか、あるいは次に入力された読み込み終了キーに対してのみ有効です)。/LOCK_STATE 修飾子は、/SET_STATE 修飾子がともに指定されている場合にだけ使用できます。

/LOG (省略時の設定)

/NOLOG

キー定義が正しく作成されたことを示すメッセージを、システムが表示するか否かを制御します。

/SET_STATE=状態名

/NOSET_STATE (省略時の設定)

キーが押された時、指定された状態名が設定されます (省略時の設定では、キーが押された時に現在のロック状態はリセットされます)。キー定義にこの修飾子を指定しない場合には、SET KEY コマンドを使用して現在の状態を変更することができます。状態名は、任意の英数字文字列で、引用符で囲んだ文字列で状態を指定します。

/TERMINATE

/NOTERMINATE (省略時の設定)

キーを押した時に、現在の等価文字列が即時処理されるかどうかを指定します (文字列を入力し、Return を押すのと同様です)。省略時の設定では、等価文字列が処理される前に、他のキーを押すことができます。したがって、コマンド行の中や、プロンプトの後、あるいは入力しているテキスト中に他の文字を挿入できる、キー定義を作成することができます。

例

```
1. $ DEFINE/KEY PF3 "SHOW TIME" /TERMINATE
   %DCL-I-DEFKEY, DEFAULT key PF3 has been defined
   $ PF3
   $ SHOW TIME
   14-DEC-2001 14:43:59
```

この DEFINE/KEY コマンドは、キーパッドの PF3 キーが、SHOW TIME コマンドを実行するよう定義します。DEFAULT とは、省略時の状態のことです。

```
2. $ DEFINE/KEY PF1 "SHOW " /SET_STATE=GOLD/NOTERMINATE/ECHO
   %DCL-I-DEFKEY, DEFAULT key PF1 has been defined
   $ DEFINE/KEY PF1 " DEFAULT" /TERMINATE/IF_STATE=GOLD/ECHO
   %DCL-I-DEFKEY, GOLD key PF1 has been defined
   $ PF1
   $ PF1
   $ SHOW DEFAULT
   DISK1:[JOHN.TEST]
```

最初の DEFINE/KEY コマンドは、PF1 キーが SHOW という文字列である

と定義しています。後続のキーに対して、状態が GOLD に設定されます。
/NOTERMINATE 修飾子は、キーが押されたときに文字列を処理しないようにシステムに指示を与えます。2 番目の DEFINE/KEY コマンドは、キーパッドが GOLD 状態の時の PF1 キーの等価文字列を定義します。キーパッドが GOLD 状態の場合に PF1 を押すと、このコマンド行での読み込み操作が終了します。

PF1 キーを 2 回押すと、システムは SHOW DEFAULT コマンドを表示し、このコマンドを処理します。

この例の 2 行目の DEFAULT という単語は、PF1 キーがすでに省略時の状態で定義されていることを示しています。2 番目の DEFINE/KEY コマンドの DEFAULT という単語の前のスペースに注意してください。このスペースが省略されていると、システムは、DEFAULT を SHOW コマンドに対するキーワードとして認識できません。

3. \$ SET KEY/STATE=ONE
 %DCL-I-SETKEY, keypad state has been set to ONE
 \$ DEFINE/KEY PF1 "ONE"
 %DCL-I-DEFKEY, ONE key PF1 has been defined
 \$ DEFINE/KEY/IF_STATE=ONE PF1 "ONE"
 %DCL-I-DEFKEY, ONE key PF1 has been defined

上記の例は、ONE 状態に対する PF1 キーに "ONE" を定義する 2 つの方法を示しています。

キー定義の望ましい方法は、2 番目の DEFINE/KEY コマンドです。この方法では、キー定義と同じコマンドに状態を指定することで、エラーの発生する可能性を無くしています。

DELETE

大容量記憶ディスク・ボリュームから，1 つまたは複数のファイルを削除します。

ファイルに対する削除 (D) アクセス権と，親ディレクトリに対する書き込み (W) アクセス権が必要です。削除するファイルがディレクトリである場合には，そのディレクトリは空でなければなりません。

フォーマット

DELETE ファイル指定[,...]

パラメータ

ファイル指定[,...]

大容量記憶ディスク・ボリュームから削除する，1 つまたは複数のファイルの名前を指定します。最初のファイル指定には，省略時のディレクトリまたは特定のディレクトリと，ファイル名，ファイル・タイプ，バージョン番号を含めなければなりません。2 番目以降のファイル指定には，バージョン番号を含めなければなりません。この場合は，前に処理したファイル指定から省略されたフィールドが適用されます。ファイル指定のどのフィールドにも，ワイルドカード文字を使用できます。

ディレクトリや装置名を省略する場合には，現在の省略時の装置および省略時のディレクトリが使用されます。

セミコロン (;) の後にファイル・バージョン番号が指定されていない場合や，バージョン番号が 0 の場合，あるいはファイル指定のバージョン番号に 1 つまたは複数のスペースが含まれている場合には，ファイルの最新バージョンが削除されます。

複数のファイルを削除するためには，ファイル指定を，コンマ (,)，またはプラス記号 (+) で区切ります。

説明

DELETE コマンドは，1 つまたは複数のファイルを，マス・ストレージ・ディスク・ボリュームから削除します。このコマンドは，ファイルに対する削除 (D) アクセス権と，親ディレクトリに対する書き込み (W) アクセス権が必要です。削除したいファイルがディレクトリである場合には，そのディレクトリは空でなければなりません。

修飾子

/BACKUP

/BEFORE または /SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、最新のバックアップの日時をもとにファイルを選択します。この修飾子以外の時刻属性を指定する修飾子、/CREATED、/EXPIRED、および /MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として /CREATED 修飾子を使用されます。

/BEFORE[=時刻]

指定された時刻以前の時刻属性を持つファイルを選択します。絶対時刻、または絶対時刻とデルタ時間の組み合わせを指定します。また、BOOT、LOGIN、TODAY(省略時の設定)、TOMORROW、および YESTERDAY というキーワードも指定できます。適用する時刻属性は、/BACKUP、/CREATED(省略時の設定)、/EXPIRED、または /MODIFIED 修飾子のいずれかで指定します。

時刻指定の詳細は、『OpenVMS ユーザーズ・マニュアル』、またはオンライン・ヘルプのトピック Date を参照してください。

/BY_OWNER[=uic]

ファイルの所有者の利用者識別コード (UIC) が、指定した所有者 UIC と一致する場合に、ファイルを選択します。/BY_OWNER 修飾子だけが指定されており、UIC が省略されている場合には、現在のプロセスの UIC が省略時の値として使用されます。

UIC は、『OpenVMS システム・セキュリティ・ガイド』に説明されている標準的な UIC 形式で指定します。

/CONFIRM

/NOCONFIRM (省略時の設定)

各ファイルに対しての DELETE 操作の実行を確認するために、DELETE 操作の前に、プロンプトが表示されるかどうかを制御します。システムがプロンプトを表示したら、次のいずれの応答を入力します。

YES	NO	QUIT
TRUE	FALSE	Ctrl/Z
1	0	ALL

Return

単語で応答する場合には、大文字と小文字を任意に組み合わせることができます。また単語による応答は、1 文字または複数の文字に短縮できます (たとえば、TRUE は T、TR、または TRU に省略できます) が、短縮しても一意でなければなりません。肯定応答は、YES、TRUE、および 1 です。否定応答は、NO、FALSE、0、Return です。QUIT と入力したり Ctrl/Z を押すと、その時点でコマンドの処理の停止を要求します。ALL を応答した場合には、コマンドは処理は継続しますが、その後プロンプトは表示されなくなります。上記のリストに示されていない応答を入力すると、DCL はエラー・メッセージを発行し、同じプロンプトが再表示されます。

DELETE

/CREATED (省略時の設定)

/BEFORE または /SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、作成日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/EXPIRED、および /MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として /CREATED 修飾子が使用されます。

/ERASE

/NOERASE (省略時の設定)

ファイルを削除すると、そのファイルが記憶されていた領域は、将来使用するためにシステムに戻されます。新しいデータがもう一度書き込まれるまで、その位置に記憶されていたデータは、まだシステムに存在したまま残ります。/ERASE 修飾子を使用すると、データが記憶されていた場所にシステムが指定したパターンが上書きされるため、データはシステムから完全に消去されます。

/EXCLUDE=(ファイル指定[...])

指定したファイルを、DELETE 操作から除外することを指定します。ファイル指定には、ディレクトリを含むことができますが、装置は含むことはできません。ファイル指定には、ワイルドカード文字(*と%)を使用できますが、特定のバージョンを除外するための相対バージョン番号の使用はできません。ファイルを 1 つしか指定しない場合には、括弧を省略できます。

/EXPIRED

/BEFORE または /SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、満了日時をもとにファイルを選択します(満了日は、SET FILE /EXPIRATION_DATE コマンドで設定します)。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/CREATED、および /MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として /CREATED 修飾子が使用されます。

/GRAND_TOTAL (Alpha/I64 のみ)

削除されたファイルの数、ブロック数、またはバイト数の合計を表示します。結果は、現在の省略時の設定に応じて、ブロック単位またはバイト単位で表示されます。現在の省略時の設定を表示するには、SHOW PROCESS/UNITS を使用します。省略時の設定を変更するには、DCL コマンド SET PROCESS/UNITS=BYTES または SET PROCESS/UNITS=BLOCKS を実行します。

/IGNORE=INTERLOCK (Alpha/I64 のみ)

書き込みアクセスしたファイルを削除用としてマークできます。これによってファイル名エントリが削除され、最後のユーザによってファイルが閉じられたときにファイルが削除されます。

/LOG

/NOLOG (省略時の設定)

削除したあとで、各ファイルのファイル指定を DELETE コマンドが表示するかどうかを制御します。

/MODIFIED

/BEFORE または /SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、最新の変更日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/CREATED、および /EXPIRED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として /CREATED 修飾子が使用されます。

/SINCE[=時刻]

指定された時刻以降の時刻属性を持つファイルを選択します。絶対時刻、または絶対時刻とデルタ時間の組み合わせを指定します。また、BOOT、LOGIN、TODAY(省略時の設定)、TOMORROW、および YESTERDAY というキーワードも指定できます。適用する時刻属性は、/BACKUP、/CREATED(省略時の設定)、/EXPIRED、または /MODIFIED 修飾子のいずれかで指定します。

時刻指定の詳細は、『OpenVMS システム・セキュリティ・ガイド』、またはオンライン・ヘルプのトピック Date を参照してください。

/STYLE=キーワード

ファイル削除中に表示するファイル名の書式を指定します。

この修飾子のキーワードは CONDENSED および EXPANDED です。意味は次の表のとおりです。

キーワード	説明
CONDENSED (省略時の設定)	ファイル名を 255 文字長の文字列に適合するように表示します。このファイル名の場合、ファイル指定に DID あるいは FID 短縮形を含むことが可能です。
EXPANDED	ファイル名をディスクに格納されているとおりに表示します。このファイル名の場合、ファイル指定に DID あるいは FID 短縮形は含みません。

キーワード CONDENSED と EXPANDED を同時に指定することはできません。この修飾子は、確認が要求された場合に、出力メッセージに表示されるファイル名の書式を指定します。

EXPANDED キーワードが指定されていない場合、ファイル・エラーは CONDENSED ファイル指定で表示されます。

詳細は『OpenVMS ユーザーズ・マニュアル』を参照してください。

例

1. \$ DELETE COMMON.SUM;2

この DELETE コマンドは、現在の省略時のディスクおよびディレクトリから、COMMON.SUM;2 というファイルを削除します。

DELETE

2. `$ DELETE *.OLD;*`

この DELETE コマンドは、ファイル・タイプが OLD であるすべてのファイルの、すべてのバージョンを、省略時のディスク・ディレクトリから削除します。

3. `$ DELETE ALPHA.TXT;*, BETA;*, GAMMA;*`

この例は、ファイル ALPHA.TXT, BETA.TXT と GAMMA.TXT の全バージョンを削除します。ファイル・タイプは、最初のファイルのものが一時的な省略時の設定として使用されますが、バージョン番号は必ず指定しなければなりません(ここではワイルドカードとして指定されています)。

4. `$ DELETE /BEFORE=15-APR/LOG *.DAT;*`

```
%DELETE-I-FILDEL, DISK2:[MAIN]ASSIGN.DAT;1 deleted (5 block)
%DELETE-I-FILDEL, DISK2:[MAIN]BATCHAVE.DAT;3 deleted (4 blocks)
%DELETE-I-FILDEL, DISK2:[MAIN]BATCHAVE.DAT;2 deleted (4 blocks)
%DELETE-I-FILDEL, DISK2:[MAIN]BATCHAVE.DAT;1 deleted (4 blocks)
%DELETE-I-FILDEL, DISK2:[MAIN]CANCEL.DAT;1 deleted (2 blocks)
%DELETE-I-FILDEL, DISK2:[MAIN]DEFINE.DAT;1 deleted (3 blocks)
%DELETE-I-FILDEL, DISK2:[MAIN]EXIT.DAT;1 deleted (1 block)
%DELETE-I-TOTAL, 7 files deleted (23 blocks)
```

この例では、ファイル・タイプが DAT で、今年の 4 月 15 日以前に作成もしくは更新された、全ファイルの全バージョンを削除しています。/LOG 修飾子を指定すれば、ファイル名だけでなく、削除されたファイルの総数も表示されます。

5. `$ DELETE A.B;`

この DELETE コマンドは、ファイル A.B の最新バージョンを削除します。

6. `$ DELETE/CONFIRM/SINCE=TODAY [MEIER.TESTFILES]*.OBJ;*`

```
DISK0:[MEIER.TESTFILES]AVERAG.OBJ;1, delete? [N]:Y
DISK0:[MEIER.TESTFILES]SCANLINE.OBJ;4, delete? [N]:N
DISK0:[MEIER.TESTFILES]SCANLINE.OBJ;3, delete? [N]:N
DISK0:[MEIER.TESTFILES]SCANLINE.OBJ;2, delete? [N]:N
DISK0:[MEIER.TESTFILES]WEATHER.OBJ;3, delete? [N]:Y
```

この DELETE コマンドは、[MEIER.TESTFILES]というサブディレクトリに含まれる、ファイル・タイプが OBJ であるファイルのすべてのバージョンを調べ、その中で今日作成または変更されたファイルを見つけます。各ファイルを削除する前に、そのファイルを削除するかどうかを確認するプロンプトを表示します。省略時の応答である N が、かぎ括弧に囲まれて示されます。

7. `$ DIRECTORY [.SUBTEST]`

```
%DIRECT-W-NOFILES, no files found
$ SET SECURITY/PROTECTION=(OWNER:DELETE) SUBTEST.DIR
$ DELETE SUBTEST.DIR;1
```

ディレクトリ・ファイル SUBTEST.DIR の削除前に、DIRECTORY コマンドでディレクトリ下にファイルがないことを確認しています。SET SECURITY /PROTECTION コマンドでディレクトリ・ファイルの保護を再定義し、削除可能にしています。その後、DELETE コマンドで削除しています。

8. `$ DELETE DALLAS"THOMAS SECRET":DISK0:[000,000]DECODE.LIS;1`

この DELETE コマンドは、リモート・ノード DALLAS の、装置 DISK0 上の [000,000] というディレクトリから、ファイル DECODE.LIS;1 を削除します。リモート・ノード名のあとに、ユーザ名とパスワードが指定されています。

9. `$ DELETE NODE12::"DISK1:DEAL.BIG"`
`$ DELETE NODE12::DISK1:DEAL.BIG;`

この 2 つの DELETE コマンドは、ともにリモート・ノード QUEBEC 上の装置 ZZZ1 のファイル DEAL.BIG を削除します。DELETE コマンドはそのファイル指定にバージョン番号を必要としますが、バージョン番号をサポートしないノード上にファイルがあるため (QUEBEC は、RT-11 ノードです)、ファイル指定を二重引用符で囲む (" ") かセミコロン (;) だけを指定します。

10. `$ DELETE/GRAND_TOTAL *.txt;*`
`%DELETE-I-TOTAL, 61 files deleted (274KB)`

この例の出力では、61 個のファイル、合計 274KB が削除されたことが表示されています。プロセスは現在、ファイル・サイズをバイト単位で表示するように設定されています。ブロック単位で表示するように変更するには、SET PROCESS/UNITS=BLOCKS コマンドを使用します。

DELETE/BITMAP (Alpha/I64 のみ)

このコマンドを使用すると、システム管理者は、アクティブなビットマップを削除して、メモリ・リソースを開放することができます。ミニコピー・ビットマップを削除すると、仮想ユニットの以前のメンバは、完全なコピー操作でしか追加できなくなります。ビットマップの詳細は、『Volume Shadowing for OpenVMS 説明書』を参照してください。

装置の所有権、または VOLPRO (ボリューム保護) 特権が必要です。

フォーマット

DELETE/BITMAP *n[,n,...]*

パラメータ

n[,n,...]

削除するビットマップのビットマップ ID を、1 つ以上指定します。

修飾子

/LOG

/NOLOG (省略時の設定)

各ビットマップを削除する際に、表示するかどうかを指定します。

例

1. \$ SHOW DEVICE /BITMAP DSA12

Device Name	BitMap ID	Size (Bytes)	Percent Populated	Type of Bitmap	Master Node	Active
DSA12:	00020007	8364	0%	Minimerge	NODE1	Yes
	00040008	8364	0%	Minimerge	NODE2	Yes

\$ DELETE/BITMAP 00020007

この例では、SHOW DEVICE コマンドの出力で2つのビットマップが表示されています。DELETE コマンドでは、ID が 00020007 のビットマップを削除しています。

DELETE/CHARACTERISTIC

DEFINE/CHARACTERISTIC コマンドを使用してすでに設定されている、キュー属性の定義を削除します。/CHARACTERISTIC 修飾子は省略できません。

OPER (オペレータ) 特権が必要です。

フォーマット

DELETE/CHARACTERISTIC 属性名

パラメータ

属性名

削除する属性の名前を指定します。

説明

DELETE/CHARACTERISTIC コマンドは、システム属性テーブルから属性を削除します。

属性の名前または番号を変更するには、その属性を一度削除して再定義する必要があります。

修飾子

/LOG

/NOLOG (省略時の設定)

削除された属性名を削除後に表示するかどうかを制御します。

例

```
1. $ DEFINE/CHARACTERISTIC BLUE 7
    .
    .
    .
    $ DELETE/CHARACTERISTIC BLUE
    $ DEFINE/CHARACTERISTIC BLUE_INK 7
```

この DEFINE/CHARACTERISTIC コマンドは、属性 BLUE を属性番号 7 で、プリンタの青インクを表すようにします。この属性名を変更するため、DELETE/CHARACTERISTIC コマンドを入力し、その後で DEFINE/CHARACTERISTIC コマンドに同じ属性番号 7 を使用して、属性名を BLUE_INK に変更しています。

DELETE/ENTRY

キューから、1つまたは複数のプリント・ジョブまたはバッチ・ジョブを削除します。/ENTRY 修飾子は省略できません。

キューに対する管理 (M) アクセス権、または指定ジョブに対する削除 (D) アクセス権が必要です。

フォーマット

DELETE/ENTRY=(エントリ番号[,...]) [キュー名[:]]

パラメータ

エントリ番号[,...]

キューから削除するジョブのエントリ番号 (またはエントリ番号のリスト) を指定します。エントリ番号を1つしか指定しない場合には、括弧を省略できます。キュー名を省略した場合は、複数のキューからエントリを削除できます。

プリントおよびバッチ・ジョブには、システムで一意的なエントリ番号が与えられます。PRINT や SUBMIT コマンドは、省略時の設定では、ジョブが正常にキューに登録された場合にエントリ番号を表示し、ローカル・シンボル\$ENTRY を最新のジョブのエントリ番号に設定します。ジョブのエントリ番号を確認するには、SHOW ENTRY または SHOW QUEUE コマンドを使用します。

キュー名[:]

ジョブが存在するキューの名前を指定します。キュー名には、ジョブが登録されたキュー、またはジョブが実行されているキューを指定できます。キュー名は省略可能ですが、指定された場合には、エントリが指定キューにあるかどうか削除前にチェックされます。

説明

DELETE/ENTRY コマンドは、キューから1つまたは複数のジョブを削除します。DELETE/ENTRY コマンドでキュー名と複数のエントリ番号を指定する場合、すべてのジョブが同じキューに入っている必要があります。

ジョブは、現在実行中であっても他の状態であっても、削除できます。たとえば DELETE/ENTRY は、保留状態や待ち状態のジョブを削除できます。

 修飾子

/LOG

/NOLOG (省略時の設定)

削除するバッチまたはプリント・ジョブのエントリ番号を表示するかどうかを制御します。

 例

1. \$ PRINT/HOLD ALPHA.TXT
Job ALPHA (queue SYS\$PRINT, entry 110) holding
.
.
.
\$ DELETE/ENTRY=110 SYS\$PRINT

この PRINT コマンドは、ALPHA.TXT というファイルを印刷するジョブを、保留状態でキューに登録します。このジョブの実行は、SET ENTRY/RELEASE コマンドが入力されるまで延期されます。システムは、ジョブ名、エントリ番号、ジョブが登録されたキュー名、および状態を表示します。その後、DELETE/ENTRY コマンドでキュー SYS\$PRINT からそのエントリを削除することを指定しています。

2. \$ SUBMIT/AFTER=18:00 WEATHER
Job WEATHER (queue SYS\$BATCH, entry 203) holding until 14-DEC-2001 18:00
\$ SUBMIT/HOLD/PARAMETERS=SCANLINE DOFOR
Job DOFOR (queue SYS\$BATCH, entry 210) holding
.
.
.
\$ DELETE/ENTRY=(203,210)/LOG
%DELETE-W-SEARCHFAIL, error searching for 203
-JBC-E-NOSUCHENT, no such entry
%DELETE-I-DELETED, entry 210 aborting or deleted

SUBMIT コマンドは、コマンド・プロシージャ WEATHER.COM と DOFOR.COM をバッチ・キューに登録しています。WEATHER.COM のは、午後 6:00 に実行するように、DOFOR.COM は、SET ENTRY/RELEASE コマンドを入力するまで実行されないように保留状態で、それぞれ登録されています。その後、DELETE/ENTRY/LOG コマンドで両方のエントリをキューから削除しています。エントリが削除されたことを示すメッセージが表示されます。

DELETE/ENTRY/LOG コマンドが入力される前に、ジョブ WEATHER (エントリ 203) は終了していたため、このエントリは既に存在していません。キューにエントリがないことを示すメッセージが表示されています。一方、ジョブ DOFOR (エントリ 210) は、DELETE/ENTRY/LOG コマンドの入力時には保留状態であ

ったため、このエントリはキューから削除され、そのことを示すメッセージが表示されています。

```
3. $ PRINT CHAPTER8.MEM
Job CHAPTER8 (queue SYS$PRINT, entry 25) pending on queue
SYS$PRINT
.
.
.
$ SHOW QUEUE SYS$PRINT
Printer queue SYS$PRINT, on PARROT::PARROT$LPA0,
mounted form DEFAULT
```

Entry	Jobname	Username	Status
-----	-----	-----	-----
24	CHAPTER7	SMITH	Pending
25	CHAPTER8	SMITH	Pending

```
$ DELETE/ENTRY=25
```

この PRINT コマンドは、CHAPTER8.MEM というファイルを印刷するジョブを、SYS\$PRINT というプリント・キューに登録します。この後で SMITH というユーザは、このファイルを印刷する前に、もう一度編集しなければならないことに気付きました。そこでユーザ SMITH は、SHOW QUEUE コマンドを使用して、ジョブがまだ実行されていないことと、そのジョブのエントリ番号が 25 であることを確認し、その後 DELETE/ENTRY コマンドを入力することにより、キューからそのジョブを削除しています。

DELETE/FORM

DEFINE/FORM コマンドによってすでに設定されている，プリント・キューまたはターミナル・キューのフォーム・タイプの定義を削除します。/FORM 修飾子は省略できません。

OPER（オペレータ）特権が必要です。

フォーマット

DELETE/FORM フォーム名

パラメータ

フォーム名
削除するフォーム名を指定します。

説明

DELETE/FORM コマンドは，システム・フォーム・テーブルからフォーム定義を削除します。フォームを削除する場合には，フォームでマウントされたキュー内のフォーム，またはそのフォームを要求するジョブでマウントされたキュー内のフォームに対する，未解決の参照があってははいけません。フォームのすべての参照を調べるには，SHOW QUEUE/FULL コマンドを使用します。

フォームの名前または番号を変更するには，フォームを削除して再定義しなければなりません。フォームの名前と番号を変更しない限り，異なる値で DEFINE/FORM コマンドを再入力すると，任意の DEFINE/FORM 修飾子の値を修正することができます。

修飾子

/LOG
/NOLOG (省略時の設定)
削除後に削除されたフォーム名を表示するかどうかを制御します。

例

1. \$ DELETE/FORM CENTER

この DELETE/FORM コマンドは、フォーム CENTER を削除しています。

2. \$ DEFINE/FORM -
_ \$ /DESCRIPTION="letter size continuous form paper" CFLET 7
.
.
.
\$ DELETE/FORM CFLET
\$ DEFINE/FORM -
_ \$ /DESCRIPTION="letter size continuous form paper" LETTER_CONT 7

この DEFINE/FORM コマンドは、8.5 × 11 インチの連続用紙を意味するように、フォーム CFLET を 7 番に定義しています。その後、フォーム名を変更するために、フォーム CFLET を削除してから、LETTER_CONT という新しいフォームを定義しています。

DELETE/INTRUSION_RECORD

侵入データベースからエントリを削除します。

CMKRNL（カーネルへのモード変更）と SECURITY 特権が必要です。

フォーマット

DELETE/INTRUSION_RECORD 侵入元

パラメータ

侵入元

ログ・インしようとしているユーザの装置またはリモート・システムの名前を指定します。侵入元は、他のオペレーティング・システム・ドメインの形式で表現される可能性があります。たとえば、大文字と小文字の区別があったり、DCL の文法規則に従わなかったりするかも知れません。このような場合には、侵入元パラメータを引用符で囲みます。

説明

侵入データベースからエントリを削除するには、DELETE/INTRUSION_RECORD コマンドを使用します。たとえば、ユーザ HAMMER が期限切れのパスワードを使用して端末 TTA24 に何度もログインしようすると、SHOW INTRUSION コマンドは次のエントリを表示します。

Intrusion	Type	Count	Expiration	Source
TERM_USER	INTRUDER	9	10:29:39.16	TTA24:HAMMER

ログイン失敗の上限に到達しているため、端末はシステムからロックアウトされます。ユーザ HAMMER がシステム管理者に連絡し、ログイン失敗の原因がパスワードの期限切れであることが確認できれば、DELETE/INTRUSION コマンドを使用して侵入データベースからレコードを削除することができます。

修飾子

/NODE=(ノード名[,...])

指定されたノードに関連するノード情報を削除します。指定されたノードがノード情

報リスト内にのみあるノードの場合，侵入レコードも削除されます。

例

1. `$ DELETE/INTRUSION_RECORD TTC2:`

この DELETE/INTRUSION_RECORD コマンドは，TTC2 への侵入の試みによって作成されたすべての侵入レコードを削除します。正当なユーザは誰もログインに失敗しないため，ユーザ名は指定されていません。

2. `$ DELETE/INTRUSION_RECORD "AV34C2/LC-2-10":FORGETFUL`

この例では，侵入元はターミナル・サーバに接続されたローカル・ターミナルです。侵入データベースから記録を削除するには，ターミナル・ポート名を引用符で囲みます。これにより，オペレーティング・システムが，スラッシュ (/) を修飾子では無い異質の文字であることを認識できます。

3. `$ DELETE/INTRUSION_RECORD NODE1::HAMMER`

このコマンドは，NODE1 ノードの HAMMER というユーザによって作成されたすべてのエントリを削除します。

4. `$ DELETE/INTRUSION_RECORD/NODE=(CAPPY,INDI)`

```
$ SHOW INTRUSION
NETWORK      SUSPECT      2  26-JUL-2001 08:51:25.66  BARNEY::HAMMER
Node: TSAVO   Count:    2
```

このコマンドは，CAPPY ノードおよび INDI ノードの侵入エントリを削除します。

5. `$ DELETE/INTRUSION_RECORD/NODE=FOOBAR`

```
$ SHOW INTRUSION
NETWORK      SUSPECT      2  26-JUL-2001 08:51:25.66  BARNEY::HAMMER
Node: TSAVO   Count:    2
```

このコマンドは，FOOBAR ノードの侵入エントリを削除します。

6. `$ DELETE/INTRUSION_RECORD/NODE=TSAVO`

```
$ SHOW INTRUSION
%SHOW-F-NOINTRUDERS, no intrusion records match specification
```

このコマンドは，TSAVO ノードの侵入エントリを削除しようと試みますが，このノードに侵入レコードはありません。

DELETE/KEY

DEFINE/KEY コマンドによって設定されたキー定義を削除します。/KEY 修飾子は省略できません。

フォーマット

DELETE/KEY [キー名]

パラメータ

キー名

定義を削除するキーの名前を指定します。このパラメータは、/ALL 修飾子と同時に指定することはできません。

修飾子

/ALL

指定された状態のすべてのキー定義が削除されることを指定します。省略時には、現在の状態のすべてのキー定義が削除されます。/ALL 修飾子を使用した場合には、キー名は指定できません。/STATE 修飾子を使用すれば、1 つまたは複数の状態を指定できます。

/LOG (省略時の設定)

/NOLOG

指定されたキー定義が削除されたことを示すメッセージが、表示されるかどうかを制御します。

/STATE=(状態名[,...])

/NOSTATE (省略時の設定)

指定されたキー定義が削除される、状態の名前を指定します。省略時には、現在の状態のキー定義が削除されます。状態名を 1 つしか指定しない場合には、括弧を省略できます。状態名は、任意の英数字文列です。

例

```
1. $ DELETE/KEY/ALL
%DCL-I-DELKEY, DEFAULT key PF1 has been deleted
%DCL-I-DELKEY, DEFAULT key PF2 has been deleted
%DCL-I-DELKEY, DEFAULT key PF3 has been deleted
%DCL-I-DELKEY, DEFAULT key PF4 has been deleted
$
```

この例では、省略時の状態に対して PF1 キーから PF4 キーまでがすでに定義されていると仮定しています。この DELETE/KEY コマンドは、省略時の状態である現在の状態の、すべてのキー定義を削除します。

```
2. $ DEFINE/KEY PF3 "SHOW TIME" /TERMINATE
%DCL-I-DEFKEY, DEFAULT key PF3 has been defined
$ PF3
$ SHOW TIME
14-DEC-2001 14:43:59
.
.
.
$ DELETE/KEY PF3
%DCL-I-DELKEY, DEFAULT key PF3 has been deleted
$ PF3
$
```

この DEFINE/KEY コマンドは、キーパッドの PF3 キーに SHOW TIME コマンドを定義しています。その後で、DELETE/KEY コマンドの使用により、PF3 キーに対する定義を削除します。したがって、この後 PF3 キーを押しても、システム・プロンプトが表示されるだけです。

DELETE/MAILBOX (Alpha/I64 のみ)

指定されたメールボックスを削除します。

PRMMBX (パーマネント・メールボックス) 特権が必要です。

フォーマット

DELETE/MAILBOX 名前

パラメータ

名前

削除するメールボックス装置の名前 (MBA_n) , またはメールボックスを指す論理名を指定します。

修飾子

/LOG

/NOLOG (省略時の設定)

メールボックスに削除マークを付けたときに通知を表示します。

例

```
1. $SHOW LOGICAL MY_MBX
   "MY_MBX" = "MBA37:" (LNM$SYSTEM_TABLE)
$SHOW DEVICE MBA37

Device          Device          Error
Name            Status          Count
MBA37:          Online          0
$DELETE/MAILBOX/LOG MBA37
%DELETE-I-MBXDEL, Mailbox MBA37 has been marked for deletion
$SHOW DEV MBA37
%SYSTEM-W-NOSUCHDEV, no such device available
```

この例では、論理名 MY_MBX が指しているメールボックス MBA37 の、削除前後の状態を表示しています。

DELETE/QUEUE

INITIALIZE/QUEUE コマンドで作成されたプリント/バッチ・キューを削除します。キュー内のジョブは同時にすべて削除されます。キューは、停止状態でなければなりません。/QUEUE 修飾子は省略できません。

キューに対する管理 (M) アクセス権が必要です。

フォーマット

DELETE/QUEUE キュー名[:]

パラメータ

キュー名[:]

削除されるキューの名前を指定します。

説明

キューを削除するには、次の手順に従ってください。

1. STOP/QUEUE/NEXT コマンドを使用して、指定されたキューを停止します。

STOP/QUEUE/NEXT コマンドは、すべての実行中のジョブが終了してから、指定されたキューを停止させます。すべての実行中のジョブが終了するまで待ってください。

2. 指定されたキューの未解決の参照がないことを確認します。

汎用キューが、指定されたキューをターゲットとする実行キューを参照する場合は、ターゲットとする実行キューのリストから指定されたキューを削除しなければなりません。

論理キューが、指定されたキューを参照する場合は、論理キューの割り当てを解除しなければなりません。

指定されたキューが汎用キューである場合、当初汎用キューに入れられ、そのターゲットとするキューにまだ存在しているジョブは、指定されたキューの参照に含まれます。指定されたキューを削除する前に、当初指定されたキューに受け渡され、現在そのターゲットとするキューで実行されているジョブを削除するか、またはこれらのジョブが終了するまで待たなければなりません。

3. 指定されたキューから別のキューにジョブを移動するには、SET ENTRY /REQUEUE または ASSIGN/MERGE コマンドを使用します。指定されたキューを削除した場合、そのキューに残っているすべてのジョブも削除されます。
4. DELETE/QUEUE コマンドを入力します。

修飾子

/LOG

/NOLOG (省略時の設定)

指定されたキューが削除されたことを示すメッセージが、表示されるかどうかを制御します。

例

1.

```
$ INITIALIZE/QUEUE/DEFAULT=FLAG/START/ON=LPA0 LPA0_QUEUE
.
.
.
$ STOP/QUEUE/NEXT LPA0_QUEUE
$ DELETE/QUEUE LPA0_QUEUE
```

最初のコマンドは、LPA0_QUEUE というプリンタ・キューを初期化し、スタートします。STOP/QUEUE/NEXT コマンドは、キューを停止します。次に、DELETE/QUEUE コマンドを実行することにより、キューを削除します。

DELETE/QUEUE/MANAGER

1 ノードまたは OpenVMS Cluster システムからキュー・マネージャを削除します。削除するキュー・マネージャが管理しているキューおよびジョブはすべて削除されます。最初にキュー・マネージャを止めます。/NAME_OF_MANAGER 修飾子は省略できません。

OPER (オペレータ) および SYSNAM (システム論理名) 特権が必要です。

フォーマット

DELETE/QUEUE/MANAGER/NAME_OF_MANAGER=キュー・マネージャ名

パラメータ

なし

説明

キュー・マネージャを削除するには、次の手順に従ってください。

1. STOP/QUEUE/MANAGER/CLUSTER/NAME_OF_MANAGER= キュー・マネージャ名コマンドを使用して、指定されたキュー・マネージャを停止します。
2. キュー・マネージャの名前を指定して、DELETE/QUEUE/MANAGER/NAME_OF_MANAGER コマンドを入力します。

修飾子

/NAME_OF_MANAGER=文字列

削除するキュー・マネージャを指定します。/NAME_OF_MANAGER 修飾子は省略できません。キュー・マネージャには、最大 31 文字の文字列または論理名を指定します。

例

1. `$ DELETE/QUEUE/MANAGER/NAME_OF_MANAGER=BATCH_MANAGER`

この DELETE/QUEUE/MANAGER/NAME_OF_MANAGER コマンドは、
BATCH_MANAGER というキュー・マネージャを削除します。キュー・データベースの共有マスタ・ファイルからの指定キュー・マネージャへの参照を削除し、
BATCH_MANAGER に関係付けられたキューおよびジャーナル・ファイルを削除します。

DELETE/SYMBOL

ローカル・シンボル・テーブル，またはグローバル・シンボル・テーブルから，1 つまたはすべてのシンボル定義を削除します。/SYMBOL 修飾子は省略できません。

フォーマット

DELETE/SYMBOL [シンボル名]

パラメータ

シンボル名

削除するシンボルの名前を指定します。/ALL 修飾子が指定されていない限り，シンボル名パラメータは必ず指定しなければなりません。シンボル名パラメータを指定する場合には，/ALL 修飾子は指定できません。シンボル名は，1 文字から 255 文字までの長さです。省略時の設定では，DELETE/SYMBOL コマンドは，シンボルが現在のコマンド・プロシージャのローカル・シンボル・テーブルに登録されていると仮定します。

説明

DELETE/SYMBOL コマンドは，シンボル・テーブルからシンボル定義を削除します。グローバル・シンボル・テーブルもローカル・シンボル・テーブルも指定しないと，ローカル・シンボル・テーブルからシンボルが削除されます。/GLOBAL 修飾子と/LOCAL 修飾子の両方を指定した場合は，最後に指定した修飾子だけが有効になります。/SYMBOL 修飾子は，必ず DELETE コマンドの直後に指定しなければなりません。

修飾子

/ALL

指定されたシンボル・テーブルの，すべてのシンボル名が削除されることを指定します。/LOCAL または/GLOBAL 修飾子を指定しない場合には，現在のコマンド・レベルで定義された，すべてのシンボルが削除されます。/ALL 修飾子を指定する場合には，シンボル名パラメータは指定できません。

/GLOBAL

シンボル名が，現在のプロセスのグローバル・シンボル・テーブルから削除されることを指定します。

/LOCAL (省略時の設定)

シンボル名が、現在のプロセスのローカル・シンボル・テーブルから削除されることを指定します。

/LOG

/NOLOG (省略時の設定)

削除された各シンボルを示す、情報メッセージが表示されるか否かを制御します。

例

1. `$ DELETE/SYMBOL/ALL`

この DELETE/SYMBOL コマンドは、現在のコマンド・レベルのすべてのシンボル定義を削除します。

2. `$ DELETE/SYMBOL/LOG KUDOS`

`%DCL-I-DELSYM, LOCAL symbol KUDOS has been deleted`

この DELETE/SYMBOL コマンドは、現在のプロセスのローカル・シンボル・テーブルから、シンボル FOO を削除します、さらに、/LOG 修飾子が指定されているため、削除されたシンボルを示す情報メッセージが表示されます。

3. `$ DELETE/SYMBOL/GLOBAL PDEL`

この DELETE/SYMBOL コマンドは、現在のプロセスのグローバル・シンボル・テーブルから、PDEL という名前のシンボルを削除します。

DEPOSIT

仮想メモリ中の指定した位置の内容を置換し、新しい内容を表示します。

DEPOSIT コマンドと同時に EXAMINE コマンドを使用すると、会話型でプログラムをデバッグすることができます。DCL コマンドの DEPOSIT コマンドは、OpenVMS Debugger の DEPOSIT コマンドに似ています。

内容を変更したい仮想メモリ位置へのユーザ・モード読み込み (R) 権および書き込み (W) アクセス権が必要です。

フォーマット

DEPOSIT 位置=データ[,...]

パラメータ

位置

内容を変更したい仮想メモリ中の仮想アドレスの開始点、または仮想アドレスの範囲 (2 番目に指定するアドレスは、最初に指定したアドレスより大きくなければなりません) を指定します。ここでは整数値、シンボル名、レキシカル関数、またはこれらの任意の組み合わせを含む、有効な整数式を指定します。基数修飾子は、アドレスを解釈する基数を決めます。省略時の設定は 16 進形式です。シンボル名は常に、それが定義された基数で解釈されます。基数演算子 %X, %D, または %O は、位置の前に指定できます。16 進値は数字 (または前に %X をつけた数字) で始めなければなりません。

ここで指定する位置は、プロセスで現在実行中のイメージの、仮想アドレス領域内ではなければなりません。

DEPOSIT および EXAMINE コマンドは、現在のメモリ位置へのポインタを保守します。DEPOSIT コマンドはこのポインタを、修正した最後のバイトの次のバイトに設定します。後続の EXAMINE および DEPOSIT コマンドでピリオド(.)を使用すると、このポインタを参照することができます。DEPOSIT コマンドを使用して指定したデータを格納できない場合は、ポインタは変更しません。EXAMINE コマンドは、ポインタの値を変更しません。

データ[,...]

指定した位置に格納するデータを指定します。省略時の設定では、データは 16 進形式で指定されているとみなします。あとで 2 進形式に変換したり、指定した位置に書き込むことができます。

複数のデータを指定する場合は、各データをコンマ(,)で区切ります。DEPOSIT コマンドは、指定されたアドレスから連続的な位置にデータを書き込みます。

ASCII 以外のデータを格納する場合は、有効な整数式を使用して複数のデータ項目を指定することができます。

ASCII データを格納する場合は、データ項目は 1 つしか指定できません。等号の右にあるすべての文字は、文字列であるとみなされます。これらの文字は大文字に変換され、空白は圧縮されます。

説明

DEPOSIT コマンドの実行が終了すると、データが格納された仮想メモリアドレスと、その位置の新しい内容が表示されます。次の例を参照してください。

```
address: contents
```

現在のアクセス・モードで、指定したアドレスの読み込みはできても書き込みができない場合、DEPOSIT コマンドはその位置の元の内容を表示します。指定したアドレスからの読み込みも書き込みもできない場合は、DEPOSIT コマンドはデータ・フィールドにアスタリスク(*)を表示します。DEPOSIT コマンドは、その位置(修正した最後のバイトの次のバイト)にポインタを設定します。

数値のリストを指定すると、アクセス違反が発生する前に、いくつかの値は正常に格納されます。ASCII データを格納している間にアクセス違反が発生すると、何も格納されません。

基数修飾子: DEPOSIT または EXAMINE コマンドで、コマンド・インタプリタが数値リテラルを解釈する基数です。省略時の基数は、16 進です。この場合、コマンド行に指定されたすべての数値リテラルは、16 進数であるとみなされます。コマンド行で基数修飾子を変更すると、その基数が、後続の EXAMINE および DEPOSIT コマンドの省略時の基数になります。これは、他の修飾子でその設定を変更するまで続きます。次の例を参照してください。

```
$ DEPOSIT/DECIMAL 900=256
00000384: 256
```

DEPOSIT コマンドは、位置 900 と値 256 の両方を、10 進数として解釈しています。後続のすべての DEPOSIT および EXAMINE コマンドは、アドレスやデータに指定した数字を 10 進数とみなします。DEPOSIT コマンドは、常にアドレス位置を 16 進数で表示することに注意してください。

= (割り当て文) コマンドで定義されたシンボル値は、常にそれが定義された基数で解釈されます。

格納位置または格納データとして指定する 16 進数値は、数字 (0 ~ 9) で始めなければなりません。数字 (0 ~ 9) で始めないとコマンド・インタプリタは、シンボル名が入力されたとみなし、シンボルを置換しようとします。

DEPOSIT コマンドを入力する時に基数演算子 %X, %D, または %O を使用すると、現在の省略時の設定を変更することができます。次の例を参照してください。

```
$ DEPOSIT/DECIMAL %X900=10
```

このコマンドは、16 進数 900 として指定した位置を、10 進数値 10 として格納します。

長さ修飾子: DEPOSIT コマンドの、省略時の長さの単位はロングワードです。データ値のリストを指定した場合、データは指定した位置から始まる連続したロングワードに格納されます。コマンド行で長さ修飾子を変更すると、その長さが、後続の EXAMINE および DEPOSIT コマンドの省略時の長さになります。これは、他の修飾子でその設定を変更するまで続きます。

ASCII 値を格納する場合、長さ修飾子は無視されます。

修飾子の位置に関する制限事項: DEPOSIT コマンドは、式を算術的に分析します。そのため、前にスラッシュ (/) が必要な修飾子は、正確に解釈されるためには、コマンド名の直後に指定しなければなりません。

修飾子

/ASCII

指定したデータが ASCII であることを示します。

データ項目は 1 つだけ指定できます。等号 (=) の右にあるすべての文字は、文字列であるとみなされます。引用符 (") で囲まないと、メモリに書き込まれる前にこれらの文字は、すべて大文字に変換され複数の空白は 1 つ空白に圧縮されます。

DEPOSIT コマンドは、仮想メモリに格納する前に、データを 2 進形式に変換します。/ASCII が指定されている場合、または省略時の設定が ASCII モードの場合、ユーザが指定した位置は 16 進形式であるとみなされます。

/BYTE

一度に 1 バイトずつデータを格納するよう要求します。

/DECIMAL

データが 10 進形式であることを示します。DEPOSIT コマンドは、仮想メモリに格納する前に、データを 2 進形式に変換します。

DEPOSIT

/HEXADECIMAL

データが 16 進形式であることを示します。DEPOSIT コマンドは、仮想メモリに格納する前に、データを 2 進形式に変換します。

/LONGWORD

一度に 1 ロングワードずつデータを格納するよう要求します。

/OCTAL

データが 8 進形式であることを示します。DEPOSIT コマンドは、仮想メモリに格納する前に、データを 2 進形式に変換します。

/WORD

一度に 1 ワードずつデータを格納するよう要求します。

例

1. \$ RUN MYPROG

.
.
.

Ctrl/Y

```
$ EXAMINE %D2145876444
7FE779DC: 0000000000
$ DEPOSIT .=17
7FE779DC: 0000000017
$ CONTINUE
```

RUN コマンドで、イメージ MYPROG.EXE を実行します。次に Ctrl/Y を押してプログラムを中断します。/HEXADECIMAL および/LONGWORD 修飾子の省略時の設定が有効であるとみなされ、DEPOSIT コマンドは仮想メモリ位置 2145876444 にロングワード値 17(10 進数で 23) を格納します。

EXAMINE コマンドは、現在のメモリ位置(ここでは仮想アドレス 2145876444)にポインタを設定しているので、DEPOSIT コマンドでピリオド(.)を使用して、この位置を参照することができます。

CONTINUE コマンドで、イメージの実行を再開させます。

2. \$ DEPOSIT/ASCII 2C00=FILE: NAME: TYPE:
00002C00: FILE: NAME: TYPE:...

この例で DEPOSIT コマンドは、16 進形式で位置 2C00 に文字データを格納し、修正した後でこの位置の内容を表示します。現在の省略時の長さはロングワードなので、DEPOSIT コマンドからの応答は、完全なロングワードを表示します。反復記号(...)は、DEPOSIT コマンドにより修正されなかった情報を含むデータの、最後のロングワードより後にロングワードがあることを示します。


```

3. $ EXAMINE 9C0          ! Look at Hex location 9C0
000009C0: 8C037DB3
$ DEPOSIT .=0            ! Deposit longword of 0
000009C0: 00000000
$ DEPOSIT/BYTE .=1       ! Put 1 byte at next location
000009C4: 01
$ DEPOSIT .+2=55         ! Deposit 55 next
000009C7: 55
$ DEPOSIT/LONG .=0C,0D,0E ! Deposit longwords
000009C8: 0000000C 0000000D 0000000E

```

上記の例の一連の DEPOSIT コマンドは、DEPOSIT コマンドで現在の位置ポインタを変更する方法を示しています。/BYTE 修飾子を指定した後、すべてのデータは格納されバイトで表示されます。これは/LONGWORD 修飾子がシステムの省略時の設定を格納するまで続きます。

```

4. $ BASE=%X200          ! Define a base address
$ LIST=BASE+%X40         ! Define offset from base
$ DEPOSIT/DECIMAL LIST=1,22,333,4444
00000240: 00000001 00000022 00000333 00004444
$ EXAMINE/HEX LIST:LIST+0C ! Display results in hex
00000240: 00000001 00000016 0000014D 0000115C

```

割り当て文は基底アドレスを 16 進形式で定義し、基底アドレスから 16 進オフセットにラベルを定義します。DEPOSIT コマンドは値のリストを読み込み、指定した位置から各値をロングワードに格納します。EXAMINE コマンドは、これらの値を 16 進数で表示するよう要求しています。

DIFFERENCES

2つのディスク・ファイルの内容を比較し、一致しないレコードのリストを表示します。

フォーマット

DIFFERENCES 第1ファイル [第2ファイル]

パラメータ

第1ファイル

比較のために入力する最初のファイルの名前を指定します。ファイルには、ファイル名とファイル・タイプを指定しなければなりません。ファイルの中で、ワイルドカード文字は使用できません。

第2ファイル

比較のために2番目に入力するファイルの名前を指定します。省略したフィールドに対しては、第1ファイルの対応するフィールドが使用されます。ファイルの中で、ワイルドカード文字は使用できません。

第2ファイルを指定しない場合には、DIFFERENCES コマンドは、第1ファイルの1つ前のバージョンを使用します。

説明

2つのファイルが同じかどうか、また異なる場合どのように異なるかを判定するには、DIFFERENCES コマンドを使用します。DIFFERENCES コマンドは、指定された2つのファイルをレコードごとに比較し、異なる場合は差をリストする出力ファイルを生成します。

DIFFERENCES コマンドの修飾子は、次のように機能別に分類できます。

- DIFFERENCES コマンドにレコードごとのデータを無視するように要求する修飾子

/COMMENT_DELIMITERS
/IGNORE

これらの修飾子を使用すると、コメントを表す文字を定義したり、ファイルを比較する場合に無視する文字または文字のクラスを指定したりすることができます。たとえば、DIFFERENCES コマンドに、余分な空白行や行の中の余分なスペースを無視させることができます。

省略時の設定では、DIFFERENCES コマンドは、各レコード内のすべての文字を比較します。

- 違いの一覧に含まれる情報の書式を制御する修飾子

```
/CHANGE_BAR
/IGNORE
/MERGED
/MODE
/PARALLEL
/SEPARATED
/SLP
/WIDTH
```

省略時の設定では、比較するファイルに存在する違いがマージされます。2つのファイルの一致しない各レコードがリストされてから、一致する次のレコードがリストされます。

省略時の設定では、リストされるレコードごとに行番号が示され、無視するように指定されたすべての文字を除去したレコードがリストされます。

修飾子を組み合わせて指定すると、複数の書式で比較をリストする出力を要求できます。SLP 出力は、他のすべての出力の種類とは互換性がないことに注意してください。パラレル出力を生成できるのは、ASCII モードだけです。

- 比較の範囲を制御する修飾子

```
/MATCH
/MAXIMUM_DIFFERENCES
/WINDOW
```

省略時の設定では、マスター入力ファイル内のすべてのレコードが読み込まれ、更新入力ファイル内の一致するレコードが探されます。2つの入力ファイルの一致の検出は、一致が見つかるか、2つのファイルの最後に到達するまで続きます。2つのファイルのセクションが一致すると見なされるのは、各ファイルの連続した3つのレコードが同じと判定される場合だけです。

省略時の設定では、DIFFERENCES コマンドの出力は、現在の SYS\$OUTPUT 装置に書き出されます。出力を別のファイルまたは装置に書き出すには、/OUTPUT 修飾子を使用します。

DIFFERENCES コマンドは、終了状態で終了します。比較の結果は、次の重大度で示されます。

SUCCESS	ファイルは同じです。
INFORMATIONAL	ファイルは異なります。
WARNING	DIFFERENCES の利用者指定最大数を超過しました。
ERROR	仮想メモリの不足のため、比較を完了できません。

SUCCESS 以外のすべての重大度は、2つの入力ファイルが異なることを表します。

修飾子

`/CHANGE_BAR=[([変更バー文字],[[NO]NUMBER])]`

指定された文字を使用して違いをマークします。`/CHANGE_BAR` 修飾子が表示する出力は、`/CHANGE_BAR` 修飾子を指定した位置により異なります。次の例では、`/CHANGE_BAR` 修飾子の位置による結果について説明します。

次の例では、前のバージョン `input.file` とは異なる行すべての先頭にシャープ記号(`#`)の付いた最新のバージョンを表示します。

```
$ DIFFERENCES input.file/CHANGE_BAR=#
```

次の例では、`input.file;1` と異なる行すべての先頭にシャープ記号(`#`)の付いた `input.file;2` を表示します。

```
$ DIFFERENCES input.file;1 input.file;2 /CHANGE_BAR=#
```

次の例では、`input.file;2` と異なる行すべての先頭にシャープ記号(`#`)を付けた `input.file;1` を表示します。

```
$ DIFFERENCES input.file;1/CHANGE_BAR=# input.file;2
```

次の例では、`input.file;2` と異なる行すべての先頭にパーセント記号(`%`)を付けた `input.file;1` と、`input.file;1` と異なる行すべての先頭にシャープ記号(`#`)を付けた `input.file;2` を表示します。

```
$ DIFFERENCES input.file;1/CHANGE_BAR=% input.file;2/CHANGE_BAR=#
```

- 変更バー文字を指定しないと、ASCII 出力モードでは、省略時の設定の変更バー文字は感嘆符(!)です。
- 16 進出力モードおよび 8 進出力モード (詳細は `/MODE` 修飾子を参照) では、変更バー文字は無視されます。そのかわりにレコードの見出しに「***CHANGE***」という文字列が表示されます。キーワード `NONUMBER` を指定すると、リストに行番号は表示されません。
- `NUMBER` または `NONUMBER` の指定により、変更バー・リストに行番号を付けるかどうかを制御できます。どちらも指定しない場合には、省略時の値は `/[NO]NUMBER` コマンド修飾子により決まります。
- オプションを 1 つだけ指定する場合、括弧は省略できます。

- 変更バー文字として感嘆符(!)を指定する場合には、/CHANGE_BAR=(“!”,NUMBER)のように、引用符で囲む必要があります。

/COMMENT_DELIMITER[(区切り文字[,...])]

指定されたコメント区切り文字より右の文字を無視します。

1文字だけ指定する場合は、括弧は省略できます。引用符で囲んでいない小文字は、自動的に大文字に変換されます。英数字以外の文字(たとえば!や,)は、引用符で囲まなければなりません。複数文字のコメント文字は使用できません。文字自体、または次に示すキーワードのいずれか1つをタイプすることにより、最大32のコメント文字を指定できます。一意に認識できれば、キーワードは2文字までに短縮できます(1文字だと、区切り文字であると認識されます)。

キーワード	文字
COLON	コロン(:)
COMMA	コンマ(,)
EXCLAMATION	感嘆符(!)
FORM_FEED	フォーム・フィード
LEFT	左かぎかっこ([)
RIGHT	右かぎかっこ(])
SEMI_COLON	セミコロン(;))
SLASH	スラッシュ(/)
SPACE	スペース
TAB	水平タブ

/COMMENT_DELIMITER 修飾子を指定すると、/IGNORE=COMMENTS 修飾子も暗黙に含まれます。

ある文字の大文字と小文字のどちらも区切り文字として使用する場合には、その文字を2回指定し、1回は大文字で、もう1回は小文字で指定します。/COMMENT_DELIMITER 修飾子にコメント文字もキーワードも指定しないと、DIFFERENCES コマンドはファイル・タイプにより省略時の設定のコメント文字を想定します。あるファイル・タイプ(.COB および.FOR)では、行の1桁目にある場合のみ、省略時の設定のコメント文字は有効な区切り文字とみなされます。

ファイル・タイプと、省略時の設定のコメント文字を次の表に示します。

ファイル・タイプ	省略時の設定のコメント文字
.B2S, .B32, .BAS, .BLI	!
.CBL, .CMD	! and ;
.COB	1 桁目の*または/
.COM, .COR	!
.FOR	任意の位置の!, および1桁目のC, D, c, d

ファイル・タイプ	省略時の設定のコメント文字
.HLP	!
.MAC, .MAR	;
.R32, .REQ	!

/EXACT

/PAGE=SAVE および/SEARCH 修飾子とともに使用し、正確に一致する文字列検索を指定します。この場合、検索文字列は二重引用符(“”)で囲まなければなりません。

/SEARCH 修飾子を指定せずに/EXACT 修飾子を指定した場合は、Find(E1) キーを押すと文字列検索が有効になります。

/HIGHLIGHT[=キーワード]

/PAGE=SAVE および/SEARCH 修飾子とともに指定し、一致した検索文字列の強調表示方法を指定します。一致した検索文字列があった場合は、行全体が強調されます。キーワードには BOLD, BLINK, REVERSE, および UNDERLINE を指定できます。省略時の設定は BOLD です。

/IGNORE=(オプション[,...])

比較するときに、指定した文字、文字列、またはレコードを無視することを指定します。/IGNORE 修飾子は、比較レコードが入力ファイルに記憶されている形と同じようにリスト・ファイルに出力されるのか、指定した文字を無視して編集されたレコードとして出力されるのかも制御します。キーワードを 1 つだけ指定する場合は、括弧は省略できます。キーワード・パラメータは、文字またはキーワードを参照します。最初のキーワードは、(無視する文字があれば)比較の際に無視する文字を決め、2 番目のキーワードは無視した文字を出力に含めるかどうかを決めます。/IGNORE 修飾子のオプションとして指定できるキーワードを次に示します。

キーワード	無視される項目
BLANK_LINES	データ間のブランク行
CASE	比較されるテキストの大文字小文字
COMMENTS	コメント区切り文字の後に続くデータ (/COMMENT_DELIMITER 修飾子を使用して、1 つまたは複数のコメント区切り文字を指定します)。
FORM_FEEDS	フォーム・フィード文字
HEADER[=n]	ファイルの中の、レコードの先頭文字がフォーム・フィードであるレコード以降の n レコードを、ヘッダ・レコードとして定義します。最初のレコードにフォーム・フィードが含まれていない場合は、最初のレコードは無視されません。n はヘッダ・サイズを表し、省略時の値は 2 です。フォーム・フィード 1 つだけからなるレコードは n にカウントされません。
SPACING	データ行の内部に含まれる複数のスペース、またはタブ (連続するスペースまたはタブは 1 つのスペースに変更されます)。

キーワード	無視される項目
TRAILING_SPACES	データ行の最後のスペース文字およびタブ文字
キーワード	出力で無視される項目の状態
EDITED	無視された文字が、削除された状態で出力レコードが作成される。
EXACT	入力ファイルと正確に同じ状態で出力レコードが作成される。
PRETTY	出力レコードは、形式化される。

各データ行に対して、まず COMMENTS、FORM_FEEDS、HEADER、および SPACING がチェックされ、そのあと TRAILING_SPACES と BLANK_LINES がチェックされます。したがって、COMMENTS、TRAILING_SPACES、および BLANK_LINES を無視することを DIFFERENCES に指示すると、コメントのあとに続くいくつかのスペース、またはブランク行のレコードは、完全に無視されます。

省略時の設定では、DIFFERENCES コマンドは、各ファイルのすべての文字を比較し、すべての相違点を報告します。また、無視された文字をすべて削除した状態で、出力ファイルにレコードを出力します。

/PARALLEL 修飾子を指定した場合は、出力の形式化が常に実行されます。次の表は、変換される文字に対応する出力を示しています。

文字	形式化された出力
タブ (Ctrl/I)	1 個から 8 個のスペース
Return (Ctrl/M)	<CR>
ライン・フィード (Ctrl/J)	<LF>
垂直タブ (Ctrl/K)	<VT>
フォーム・フィード (Ctrl/L)	<FF>
他のプリントされない文字	. (ピリオド)

/MATCH=サイズ

一致していることを解釈するために、必要なレコード数を指定します。省略時の設定では、DIFFERENCES は一致しないレコードを検出したあと、一致するレコードが 3 つ連続して検出されると、そのファイルは一致しているものとして処理を続けます。この省略時の値である 3 を変更するには、/MATCH 修飾子を使用します。

DIFFERENCES が相違点を検出したあとで、誤った位置を「一致している」と解釈している場合は、/MATCH の値を大きくします。

/MAXIMUM_DIFFERENCES=n

一致しないレコードが指定された数だけ検出されると、DIFFERENCES が終了することを指定します。

DIFFERENCES

一致しないレコード総数は、異なる各セクションごとに異なるレコードの最大数を求め、それを加算することによって判断できます。

DIFFERENCES は、指定した一致しないレコードの最大数に到達すると、その最大数に到達する前に検出したレコードだけを出力します。また 1 つのリスト形式だけを出力し、警告メッセージを表示します。

省略時の設定では、一致しないレコードの最大数は設定されていません (指定された入力ファイルに含まれるすべてのレコードが比較されます)。

`/MERGED[=n]`

出力ファイルに相違点のマージされたリストが含まれることを要求します。n の値は、一致しないレコードの各リストのあとで、出力される一致したレコード数を示します。この値は、一致サイズに等しい、あるいはそれより小さい値で 10 進数でなければなりません (`/MATCH=サイズ修飾子` を参照)。省略時の値では、DIFFERENCES コマンドは、一致しないレコードの各セットのあとで、一致した 1 つのレコードを含む、マージされた (相違点について 2 つのレコードが記録された) リストを作成します (`/MERGED=1`)。 `/MERGED`、`/SEPARATED` および `/PARALLEL` のいずれも指定されない場合は、一致しないレコードの各セットのあとで、一致した 1 つのレコードを含む、マージされたリストを作成します

省略時の n の値を変更する場合や、他のタイプの出力と相違点についての 2 つのレコードが記録されたリストを作成する場合には、`/MERGED` 修飾子を使用します。

`/MODE=(基数[,...])`

出力リストの形式を指定します。次のキーワードを指定することにより、1 つまたは複数の基数モードによる出力を作成できます。ASCII (省略時の設定)、HEXADECIMAL、OCTAL キーワード (1 つまたは複数) は括弧で囲みます。これらのキーワードは、1 文字または複数の文字に省略できます。

省略時の設定では、DIFFERENCES は出力を ASCII モードで作成します。複数の基数モードを指定する場合には、出力リストには、指定した各基数で、ファイルの比較が示されます。複数の基数モードを指定する場合には、各モードをコンマで区切らなければなりません。

`/PARALLEL` 修飾子、または `/SLP` 修飾子を指定すると、それらの形式に対して、`/MODE` 修飾子は無視されます。

`/NUMBER` (省略時の設定)

`/NONUMBER`

リストに含まれるレコードに、行番号を付けるかどうかを制御します。省略時の値では、行番号はリストに含まれます。

`/OUTPUT[=ファイル指定]`

相違点を示した出力リストが書き込まれる、出力ファイルを指定します。`/OUTPUT` 修飾子を省略する場合には、出力は現在の `SYS$OUTPUT` 装置に書き込まれます。`/OUTPUT` 修飾子だけを使用しファイル指定を省略すると、出力は、第 1 ファイ

ルと同じ名前のファイルに書き込まれます。この場合のファイル・タイプは DIF になります。ファイル指定に、ワイルドカード文字は使用できません。

/OUTPUT を指定すると、『OpenVMS ユーザーズ・マニュアル』に説明されているような、出力ファイル指定に対して適用される、省略時の値を使用できます。省略時の出力ファイル・タイプは DIF です。

/PAGE[=キーワード]

/NOPAGE (省略時の設定)

画面上の差分情報の表示を制御します。

次のキーワードを指定できます。

CLEAR_SCREEN	ページ・モードで表示 (毎回画面を消去する)
SCROLL	スクロール・モードで表示 (毎回画面を消去しない)
SAVE[=n]	n ページ分の履歴を保持する (前ページに戻ることも可能)

/PAGE=SAVE 修飾子を使用すると、最大 5 画面 (最大 255 カラムまで) 分の履歴を保存できます。ページ内では、次のキーを使って画面の移動などができます。

キー・シーケンス	意味
Up arrow key, Ctrl/B	1 行スクロール・アップ
Down arrow key	1 行スクロール・ダウン
Left arrow key	1 カラム左シフト
Right arrow key	1 カラム右シフト
Find (E1)	文字列検索を起動
Insert Here (E2)	半画面右シフト
Remove (E3)	半画面左シフト
Select (E4)	80/132 カラム切り替え
Prev Screen (E5)	前ページに移動
Next Screen (E6), Return, Enter, Space	次ページに移動
F10, Ctrl/Z	終了 (ユーティリティによっては異なることがあります)。
Help (F15)	ヘルプ・テキストを表示
Do (F16)	最新 (現在) の画面と (履歴内で) 最古画面の切り替え
Ctrl/W	再表示

/PAGE 修飾子は、/OUTPUT 修飾子と同時に指定できません。

/PARALLEL[=n]

一致しないレコードの平行・リスト (横に並べた形式のリスト) を出力することを指定します。n の値は、一致しないレコードの各リストのあとに出力される、一致したレコード数を示します。この値は 10 進正数であり、MATCH に指定されている値以下でなければなりません (/MATCH=サイズ修飾子を参照)。

DIFFERENCES

省略時の設定では、DIFFERENCES コマンドは、一致しないレコードの各リストのあとに、一致したレコードを加えたリストを作成しません。また DIFFERENCES は、マージされた相違点のリストだけを作成します。

/SEARCH="文字列"

/PAGE=SAVE 修飾子とともに指定して、表示される情報内で検索したい文字列を指定します。スペース文字等を含む場合は、検索文字列は引用符で囲まなければなりません。

情報が表示されている時に Find(E1) キーを押すと、検索文字列を動的に変更することができます。この場合は、引用符は必要ありません。

/SEPARATED[=MASTER, REVISION]

指定したファイルから一致しないレコードだけのリストを作成し、それを順番に出力します。第 1 ファイルからのリストだけを作成するには、キーワード MASTER を指定します。第 2 ファイルからのリストだけを作成するには、キーワード REVISION を指定します。

省略時の設定では、DIFFERENCES は、相違点のマージされたリストだけを作成します。

/SLP

DIFFERENCES が、SLP エディタへの入力に適した出力ファイルを作成することを指定します。/SLP 修飾子を指定する場合には、/MERGED 修飾子、/PARALLEL 修飾子、/SEPARATED 修飾子および/CHANGE_BAR 修飾子という出力ファイル修飾子は指定できません。

SLP 修飾子によって作成される出力ファイルは、第 1 入力ファイルを更新するために、SLP への入力として使用できます (第 1 入力ファイルを第 2 入力ファイルと一致するように変更できます)。

/SLP だけを指定し、/OUTPUT 修飾子を指定しない場合には、DIFFERENCES は、第 1 入力ファイルと同じファイル名でファイル・タイプが DIF であるファイルに、出力ファイルを書き込みます。

/WIDTH=n

出力リストの 1 行の幅を指定します。省略時の設定では出力は 132 文字の幅ですが、出力がターミナルに送られるときは例外です。この場合には、出力の幅は、ターミナルの行の幅によって制御されます。

ターミナルの 1 行の幅を変更するには、SET TERMINAL コマンドを使用します。

/WINDOW=サイズ

一致しないレコードのリストを作成し、第 1 入力ファイルの次のレコードに対して処理を継続する前に、検索するレコードの数を制御します。省略時の設定では、DIFFERENCES は、一致しないレコードをリストする前に、両方の入力ファイルの最後まで検索します。

ウィンドウ・サイズは、相違セクションの最小サイズであり、これは DIFFERENCES コマンドが 2 つの入力ファイル間の同期を失う原因となります。

/WRAP

/NOWRAP (省略時の設定)

/PAGE=SAVE 修飾子とともに指定し、画面の幅より長い行を改行表示するかどうかを指定します。/WRAP 修飾子を指定した場合は、はみだす部分は次の行に改行されます。

/NOWRAP 修飾子を指定した場合は、はみだした部分は左右にスクロールしながら見ることができます。

例

```
1. $ DIFFERENCES EXAMPLE.TXT
*****
File DISK1:[CHRIS.TEXT]EXAMPLE.TXT;2
  1  DEMONSTRATION
  2  OF V7.3 DIFFERENCES
  3  UTILITY
*****
File DISK1:[CHRIS.TEXT]EXAMPLE.TXT;1
  1  DEMONSTRETION
  2  OF VMS DIFFERENCES
  3  UTILITY
*****
Number of difference sections found: 1
Number of difference records found: 2
DIFFERENCES/ IGNORE=()/MERGED=1-
      DISK1:[CHRIS.TEXT]EXAMPLE.TXT;2-
      DISK1:[CHRIS.TEXT]EXAMPLE.TXT;1
```

この DIFFERENCES コマンドは、現在の省略時のディレクトリの EXAMPLE.TXT というファイルの、2 つの最新バージョンの内容を比較します。DIFFERENCES は、各レコードのすべての文字を比較し、その結果をターミナルに表示します。

DIFFERENCES

2. \$ DIFFERENCES/PARALLEL/WIDTH=80/COMMENT_DELIMITER="V" EXAMPLE.TXT

```
-----
File DISK1:[CHRIS.TEXT]EXAMPLE.TXT;2 | File DISK1:[CHRIS.TEXT]EXAMPLE.TXT;1
----- 1 ----- 1 -----
DEMONSTRATION | DEMONSTRETION
-----

Number of difference sections found: 1
Number of difference records found: 1
DIFFERENCES/IGNORE=(COMMENTS)/COMMENT_DELIMITER=("V")/WIDTH=80/PARALLEL-
    DISK1:[CHRIS.TEXT]EXAMPLE.TXT;2-
    DISK1:[CHRIS.TEXT]EXAMPLE.TXT;1
```

上記の例と同様にファイルを比較します。ただし、行の最初に“V”のつくすべての文字列は無視されます。また、違いのパラレル・リストを 80 桁で表示します。

3. \$ DIFFERENCES/WIDTH=80/MODE=(HEX,ASCII) EXAMPLE.TXT/CHANGE_BAR

```
File DISK1:[CHRIS.TEXT]EXAMPLE.TXT;2
  1 ! DEMONSTRATION
  2 ! OF V7.3 DIFFERENCES
  3 UTILITY
*****
*****
File DISK1:[CHRIS.TEXT]EXAMPLE.TXT;2
RECORD NUMBER 1 (00000001) LENGTH 14 (0000000E) ***CHANGE***
    204E 4F495441 5254534E 4F4D4544 DEMONSTRATION .. 000000
RECORD NUMBER 2 (00000002) LENGTH 19 (00000013) ***CHANGE***
    4E455245 46464944 20302E33 5620464F OF V7.3 DIFFEREN 000000
                                534543 CES..... 000010
RECORD NUMBER 3 (00000003) LENGTH 7 (00000007)
    595449 4C495455 UTILITY..... 000000
*****

Number of difference sections found: 1
Number of difference records found: 2
DIFFERENCES /WIDTH=80/MODE=(HEX,ASCII)
    DISK1:[CHRIS.TEXT]EXAMPLE.TXT;2/CHANGE_BAR-
    DISK1:[CHRIS.TEXT]EXAMPLE.TXT;1
```

例 1 と同様にファイルを比較します。ただし、違いを 16 進形式と ASCII 形式でリストします。また、省略時の設定の変更バーを出力で使用するよう指定しています。16 進形式での省略時の設定の変更バーは、***CHANGE***です。ASCII 形式での省略時の設定の変更バーは、感嘆符です。

4. \$ DIFFERENCES/OUTPUT BOSTON::DISK2:TEST.DAT OMAHA::DISK1:[PGM]TEST.DAT

この DIFFERENCES コマンドは、リモート・ノードにある 2 つのファイルを比較し違いを表示します。第 1 ファイルは、リモート・ノード BOSTON にある TEST.DAT です。第 2 ファイルは、リモート・ノード OMAHA にある TEST.DAT です。出力ファイルは DISK1:[PGM]TEST.DIF です。

DIRECTORY

ファイルのリスト，またはファイルやファイル・グループに関する情報を表示します。

ファイル名が既知のファイルを見るには，ファイルに対する実行 (E) アクセス権が必要です。ファイルを読んだり，ファイルのリストを表示したり，ファイル名にワイルドカード文字を使用する場合には，ファイルに対する読み込み (R) アクセス権が必要です。

フォーマット

DIRECTORY [ファイル指定[,...]]

DIRECTORY/FTP ディレクトリ指定

パラメータ

ファイル指定[,...]

表示する 1 つまたは複数のファイルを指定します。ファイル指定の構文によって，どのファイルが表示されるかが，次のように決定されます。

- ファイル指定を入力しない場合には，DIRECTORY コマンドは，現在の省略時のディレクトリに含まれるファイルの，すべてのバージョンのリストを表示します。
- 装置名だけを指定する場合には，DIRECTORY コマンドは，省略時のディレクトリ指定を使用します。
- ファイル指定にファイル名，ファイル・タイプ，バージョン番号が含まれない場合には常に，指定したディレクトリに含まれる，すべてのファイルのすべてのバージョンのリストが表示されます。
- ファイル指定にファイル名またはファイル・タイプの少なくともどちらかが含まれ，バージョン番号が含まれない場合には，DIRECTORY コマンドはすべてのバージョンのリストを表示します。
- ファイル指定にファイル名だけが含まれる場合には，DIRECTORY コマンドは，ファイル・タイプやバージョン番号とは無関係に，現在の省略時のディレクトリの中で，その名前を持つすべてのファイルのリストを表示します。
- ファイル指定にファイル・タイプだけが含まれる場合には，DIRECTORY コマンドは，ファイル名やバージョン番号と無関係に，現在の省略時のディレクトリの中で，そのファイル・タイプを持つ，すべてのファイルのリストを表示します。

ファイル指定のディレクトリ、ファイル名、ファイル・タイプ、またはバージョン番号には、アスタリスク (*) およびパーセント記号 (%) のワイルドカード文字が使用できます。この時には、条件を満たすすべてのファイルが表示されます。複数のファイルを指定する場合には、各ファイルをコンマ (,) またはプラス記号 (+) で区切ります。

ディレクトリ指定

標準の DECnet リモート・ファイル指定を行います。UNIX システムのように大文字と小文字の区別をつける場合や外部文字情報を保存するには、また、装置あるいはディレクトリ指定は、二重引用符で囲まれたファイル指定を行ってください。詳細は、/FTP 修飾子の説明を参照してください。

説明

DIRECTORY コマンドは、ディレクトリに入っているファイルをリストします。修飾子を使用すると、ファイル名とともに追加情報が表示されます。

DIRECTORY コマンドの出力は、特定の書式修飾子とその省略時の設定に依存します。修飾子には、/COLUMNS、/DATE、/FULL、/OWNER、/PROTECTION、および/SIZE があります。ただし、ファイルは常にアルファベット順にリストされ、最高バージョンが最初にリストされます。

修飾子とその機能を学習する際には、組み合わせて動作する修飾子と、一方が他方を上書きする修飾子に注意してください。たとえば、/FULL 修飾子を指定すると、システムは複数のカラムにすべての情報を表示できません。したがって、/COLUMNS 修飾子と/FULL 修飾子を同時に指定すると、要求したカラム数は無視されます。

また、実行時ライブラリに提供されている国際日時書式ルーチンを使用して、システムに定義されている他の言語と書式を選択することもできます。『OpenVMS RTL Library (LIB\$) Manual』を参照してください。

修飾子

/ACL

各ファイルのアクセス制御リスト (ACL) が、表示されるかどうかを制御します。指定がなければ、DIRECTORY コマンドはファイルごとに ACL を表示しません。隠しオプションを付けて作成されたアクセス制御エントリ (ACE) は、SECURITY 特権が有効になっている場合にのみ表示されます。/ACL 修飾子は、/COLUMNS 修飾子に優先します。

詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

/BACKUP

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、最新のバックアップの日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/CREATED、/EXPIRED、および/MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/BEFORE[=時刻]

指定された時刻以前の時刻属性を持つファイルを選択します。絶対時刻、または絶対時刻とデルタ時間の組み合わせを指定します。また、BOOT、LOGIN、TODAY(省略時の設定)、TOMORROW、および YESTERDAY というキーワードも指定できます。適用する時刻属性は、/BACKUP、/CREATED(省略時の設定)、/EXPIRED、または/MODIFIED 修飾子のいずれかで指定します。

時刻指定の詳細は、『OpenVMS ユーザーズ・マニュアル』、またはオンライン・ヘルプのトピック Date を参照してください。

/BRIEF (省略時の設定)

各ファイルのファイル名、タイプおよびバージョン番号だけを出力することを指定します。この簡略形式では、ファイルは各行の左から右に、アルファベット順でバージョン番号の大きい順に出力されます。/ACL、/DATE、/FILE_ID、/FULL、/NOHEADING、/OWNER、/PROTECTION、/SECURITY、および/SIZE 修飾子を使用して、簡略形式の表示を拡張することができます。

/BY_OWNER[=uic]

ファイル所有者の利用者識別コード (UIC) が、指定した所有者 UIC と一致するファイルだけを選択します。/BY_OWNER 修飾子だけを指定し UIC を省略すると、現在のプロセスの UIC が省略時の値として使用されます。

UIC は、『OpenVMS ユーザーズ・マニュアル』に説明されている標準的な UIC 形式で指定します。

詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

/CACHING_ATTRIBUTE

選択したファイルのキャッシング属性を表示します。

/COLUMNS=n

簡略表示のカラム数を指定します。省略時の設定では、カラム数は 4 です。必要な数だけカラムを要求できますが、/WIDTH 修飾子の値によって制限されます。/COLUMNS 修飾子は、/ACL 修飾子や/FULL 修飾子、/SECURITY 修飾子とともに使用することはできません。

実際に表示されるカラム数は、各カラムに対して指定した情報の量と/WIDTH 修飾子で指定した表示値により決定されます。システムは、/COLUMNS 修飾子に対して指定したカラム数とは無関係に、省略時の値の範囲内または指定された表示幅の範囲内で表示できるだけのカラムを表示します。

複数のカラムを指定し、各カラムに追加情報を表示することを要求した場合にだけ、`DIRECTORY` コマンドは長いファイル名を途中で切り捨てます。省略時のファイル名サイズは 19 文字です。この省略時の値を変更するためには、`/WIDTH` 修飾子を使用します。ファイル名が途中で切り捨てられる場合には、システムはファイル名フィールド・サイズより 1 文字だけ少ない文字数でファイル名を表示し、最後の位置に縦線を挿入します。たとえば、ファイル名が `SHOW_QUEUE_CHARACTERISTICS` であり、各カラムにファイル名とサイズの両方を表示することを `DIRECTORY` コマンドに対して要求した場合には、そのファイルは `"SHOW_QUEUE_CHARACT | 120"` として表示されます。

`/CREATED` (省略時の設定)

`/BEFORE` または `/SINCE` 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、作成日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、`/BACKUP`、`/EXPIRED`、および `/MODIFIED` 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として `/CREATED` 修飾子を使用されます。

`/DATE[=オプション]`

`/NODATE` (省略時の設定)

指定したファイルごとに、作成日、最終変更日、満了日、バックアップ日、有効日、あるいは記録日を表示することを指定します。この修飾子を省略すると、`/NODATE` 修飾子を使用されます。`/DATE` 修飾子だけを指定し、オプションを省略すると、作成日が表示されます。次のオプションが指定できます。

オプション	説明
<code>ACCESSED</code>	最終アクセス日を表示する。 詳細は『Guide to OpenVMS File Applications』を参照。
<code>ALL</code>	<code>CREATED</code> (作成日)、 <code>MODIFIED</code> (最終変更日)、 <code>EXPIRED</code> (満了日)、 <code>BACKUP</code> (バックアップ日)、 <code>EFFECTIVE</code> (有効日)、 <code>RECORDING</code> (記録日) という順序で、すべての日付を表示する。
<code>ATTRIBUTES</code>	属性の最終変更日を表示する。 詳細は『Guide to OpenVMS File Applications』を参照。
<code>BACKUP</code>	各ファイルの最新のバックアップ日を表示する。
<code>CREATED</code>	各ファイルの作成日を表示する。
<code>DATA_</code>	データの最終変更日を表示する。
<code>MODIFIED</code>	詳細は『Guide to OpenVMS File Applications』を参照。
<code>EFFECTIVE</code>	ファイルの内容が正しい有効日を表示する (ISO 9660)。
<code>EXPIRED</code>	各ファイルの満了日を表示する。
<code>MODIFIED</code>	ファイルが最後に変更された日を表示する。
<code>RECORDING</code>	記憶媒体上の記録日を表示する (ISO 9660)。

`/EXACT`

`/PAGE=SAVE` および `/SEARCH` 修飾子とともに使用し、大文字と小文字を区別した文字列検索を指定します。この場合、検索文字列は二重引用符で囲まなければなりません。

/SEARCH 修飾子を指定せずに/EXACT 修飾子を指定した場合，"Find キー" (E1) を押すと文字列検索が有効になります。

/EXCLUDE=(ファイル指定[,...])

指定したファイルを， DIRECTORY コマンドの操作から除外することを指示します。ファイル指定にディレクトリを含めることはできますが，装置名を含めることはできません。

ファイル指定には，ワイルドカード文字 (*と%) を使用できますが，特定のバージョンを除外するために相対バージョン番号を指定することはできません。

1 つのファイルしか指定しない場合には，括弧を省略できます。

/EXPIRED

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると，満了日時をもとにファイルを選択します (満了日は， SET FILE /EXPIRATION_DATE コマンドで設定します)。この修飾子は他の時刻属性を指定する修飾子，/BACKUP，/CREATED，および/MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には，省略時の設定として/CREATED 修飾子を使用されます。

/FILE_ID

ファイル識別番号 (FID) を表示するかどうかを制御します。省略時には， DIRECTORY コマンドは，/FULL 修飾子が指定された場合を除き， FID を表示しません。

/FTP

FTP ユーティリティの directory (dir あるいは ls) コマンドを実行します。 DIRECTORY/FTP コマンドは FTP ユーティリティを用いて， TCP/IP 接続を介して，指定したリモート・ディレクトリの内容のリストをローカル・ホストに出力します。書式は次のようになります。

```
$ DIR/FTP nodename"username password":::directory_pathname
```

ディレクトリ・パス名を省略した場合，ユーザのホーム・ディレクトリの内容が表示されます。ノード名だけを入力した場合は， ANONYMOUS ディレクトリの内容が表示されます。

/FULL

ファイルごとに，次の項目を表示します。

- ファイル名
- ファイル・タイプ
- バージョン番号
- ファイル識別番号 (FID)
- 使用されているブロック数
- 割り当てられているブロック数
- ファイル所有者ユーザ識別子 (UIC)

作成日
 最新の変更日とリビジョン番号
 満了日
 最新のバックアップ日
 使用の有効日
 メディア上への記録日
 ファイル編成
 シェルブ状態
 キャッシング属性
 ファイル属性
 レコード形式
 レコード属性
 RMS 属性
 ジャーナル情報
 ファイル保護
 アクセス制御リスト (ACL)
 クライアント属性
 セマンティクス・タグ値 (適用されている場合)

/GRAND_TOTAL

選択されたすべてのファイルとディレクトリの総数だけを表示します。ディレクトリごとの総ファイル数と、個々のファイル情報の両方は、出力されません (ディレクトリ数の合計の表示については、/TRAILING 修飾子を参照してください)。

/HEADING

/NOHEADING

装置とディレクトリ指定から構成される見出し行の表示を制御します。省略時の出力形式では、この見出し行は表示されます。/NOHEADING を指定する場合、出力は 1 カラムの形式になります。さらに、どのファイルに関しても、出力に装置とディレクトリ指定を含んだ完全なファイル指定が表示されます。/NOHEADING 修飾子の指定は、/COLUMNS 修飾子に優先します。

コマンド・プロシージャの中で完全なファイル指定のリストを作成して、後でそのリストを操作できるようにする場合は、/NOHEADING 修飾子と /NOTRAILING 修飾子を組み合わせて使用すると便利です。

/HIGHLIGHT[=キーワード]

/PAGE=SAVE および/SEARCH 修飾子とともに使用し、一致した検索文字列の強調表示方法を指定します。一致した検索文字列があった場合は、行全体が強調されます。キーワードには、BOLD, BLINK, REVERSE, および UNDERLINE を指定できます。省略時の設定は BOLD です。

/MODIFIED

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、最新の変更日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/CREATED、および/EXPIRED 修飾子とは同時に

指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/OUTPUT[=ファイル指定]
/NOOUTPUT

コマンドの出力の送り先を制御します。省略時には、出力は、現在の SYS\$OUTPUT 装置に送られます。ワイルドカード文字は使用できません。

/OUTPUT 修飾子に部分的なファイル指定 (たとえば/OUTPUT=[KIER]) を入力すると、DIRECTORY が省略時のファイル名となり、.LIS が省略時のファイル・タイプになります。/NOOUTPUT を指定すると、何も出力されません。

出力が、表示するディレクトリと同じディレクトリ下のファイルに書かれる場合、出力ファイルもディレクトリ・リストに表示されます。

/OWNER
/NOOWNER (省略時の設定)

ファイルの所有者のユーザ識別コード (UIC) が表示されるかどうかを制御します。

所有者フィールドの省略時のサイズは 20 文字です。ファイル所有者の UIC がこの長さを越える場合には、情報の一部が切り捨てられます。このフィールドのサイズは/WIDTH=OWNER を指定し、OWNER フィールドの値を与えることによって変更できます。詳細は/WIDTH 修飾子の説明を参照してください。

/PAGE[=キーワード]
/NOPAGE (省略時の設定)

ディレクトリ情報の画面表示をページャを介して行います。

次のキーワードを指定できます。

CLEAR_SCREEN	ページモードで表示 (毎回画面を消去する)
SCROLL	スクロールモードで表示 (毎回画面を消去しない)
SAVE[=n]	n ページ分の履歴を保持する (前ページに戻ることも可能)

/PAGE=SAVE 修飾子を指定すると最大 5 画面 (最大 255 カラムまで) 分の履歴を保存できます。ページャ内では以下のキーを使って画面の移動などができます。

キー・シーケンス	説明
Up arrow key, Ctrl/B	1 行スクロールアップ
Down arrow key	1 行スクロールダウン
Left arrow key	1 カラム左シフト
Right arrow key	1 カラム右シフト
Find (E1)	文字列検索を起動
Insert Here (E2)	半画面右シフト
Remove (E3)	半画面左シフト
Select (E4)	80/132 カラム切替え

キー・シーケンス	説明
Prev Screen (E5)	前ページに移動
Next Screen (E6), Return, Enter, Space	次ページに移動
F10, Ctrl/Z	終了 (ユーティリティによっては異なる)
Help (F15)	ヘルプ・テキストを表示
Do (F16)	最新 (現在) 画面と (履歴内で) 最古画面の切り替え
Ctrl/W	再表示

/PAGE 修飾子は/OUTPUT 修飾子とは同時に使用できません。

/PRINTER

表示をファイルに落とし、このファイルを/OUTPUT 修飾子によって指定したファイル名でキュー SYS\$PRINT に登録します。/PRINTER 修飾子だけを指定し/OUTPUT 修飾子を省略すると、出力は DIRECTORY.LIS という名前の一時ファイルに送られます。このファイルは、印刷するためにキューに登録され、その後削除されます。

/PROTECTION

/NOPROTECTION (省略時の設定)

各ファイルのファイル保護が、表示されるかどうかを制御します。

/SEARCH="文字列 "

/PAGE=SAVE 修飾子とともに使用し、表示される情報内で検索したい文字列を指定します。文字列にスペース文字等を入れたい場合は、検索文字列は二重引用符で囲まなければなりません。

情報が表示されている時に Find(E1) キーを押すと、検索文字列を動的に変更することができます。この場合は、引用符は必要ありません。

/SECURITY

ファイルの機密保護に関する情報が表示されるかどうかを制御します。/SECURITY 修飾子を使用した場合、/ACL 修飾子、/OWNER 修飾子、および/PROTECTION 修飾子の 3 つをすべて指定した時と同じ情報が表示されます。隠しオプションを付けて作成されたアクセス制御エントリ (ACE) は、SECURITY 特権が有効になっている場合にのみ表示されます。

詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

/SELECT=(キーワード[,...])

表示するファイルを選択します。次に示すキーワードのいずれか 1 つを指定します。

ACL	ACL が設定されている/いない (NOACL) ファイルを選択
NOACL	

CACHING_ATTRIBUTE=(オプション[,...])

指定したキャッシング属性のあるファイルを表示。有効なオプションは次のとおりです。

NO_CACHING
WRITETHROUGH

FILE=(オプション[,...])

ファイル指定の 1 部のみを表示。表示する/しない部分を以下の中から指定します。

[NO]NODE
[NO]DEVICE
[NO]DIRECTORY
[NO]NAME
[NO]TYPE
[NO]VERSION

/SELECT=FILE と/FULL 修飾子は同時に指定できません。
オンライン/シェルブドのファイルを選択

ONLINE
NOONLINE

PRESHELVED
NOPRESHELVED

プリシェルブド/プリシェルブドでないファイルを選択

SHELVABLE
NOSHELVABLE

シェルブ可能/不能なファイルを選択

SIZE=(オプション[,...])

サイズに基づいてファイルを選択。有効なオプションを次に示します。

オプション	説明
MAXIMUM= <i>n</i>	指定したパラメータ (<i>n</i>) よりブロック数が少ないファイルを選択します。 <i>n</i> の省略時の値は、 1,073,741,823 です。 MINIMUM= <i>n</i> を同時に指定すれば、表示するファイルのサイズの範囲を指定できます。
MINIMUM= <i>n</i>	指定したパラメータ (<i>n</i>) よりブロック数の多いファイルを選択します。 <i>n</i> の省略時の値は、 0 です。 MAXIMUM= <i>n</i> を同時に指定すれば、表示するファイルのサイズの範囲を指定できます。
MINIMUM= <i>n</i> , MAXIMUM= <i>n</i>	ブロック・サイズが MAXIMUM と MINIMUM で指定される範囲内であるファイルを選択します。
UNUSED[= <i>n</i>]	ファイルの使用領域とファイルの割り当て済みサイズとの差が、ディスクのクラスタ・サイズを超えるファイル名を表示します。値を指定すると、その値を超える未使用領域を持つファイルのファイル名が表示されます。

VERSION=(オプション[, オプション]) (Alpha/I64 のみ)

以下のオプションのどちらかまたは両方で指定した範囲に入るバージョン番号を持つファイルを、すべて表示します。

MINIMUM=数値
MAXIMUM=数値

`/SHELVED_STATE`
ファイルがシェルブド、プリシェルブド、オンラインのいずれであるかを表示します。

`/SINCE[=時刻]`
指定された時刻以降の時刻属性を持つファイルを選択します。絶対時刻、または絶対時刻とデルタ時間の組み合わせを指定します。また、`BOOT`、`LOGIN`、`TODAY`(省略時の設定)、`TOMORROW`、および `YESTERDAY` というキーワードも指定できます。適用する時刻属性は、`/BACKUP`、`/CREATED`(省略時の設定)、`/EXPIRED`、または `/MODIFIED` 修飾子のいずれかで指定します。

時刻指定の詳細、『OpenVMS ユーザーズ・マニュアル』、またはオンライン・ヘルプのトピック `Date` を参照してください。

`/SIZE[=オプション]`
`/NOSIZE` (省略時の設定)
各ファイルのブロック・サイズを表示します。`/SIZE` だけを指定し、オプションを省略した場合は、ファイルが使用している (`USED`) ブロック・サイズが表示されます。指定できるオプションは、次のとおりです。

<code>ALL</code>	使用されているブロック数と割り当てられているブロック数の両方を、ファイル・サイズとして表示します。
<code>ALLOCATION</code>	割り当てられているブロック数を、ファイル・サイズとして表示します。
<code>UNITS[=オプション]</code>	<p><code>SET PROCESS/UNITS</code> で指定されている現在の省略時の設定を上書きし、ブロック数やバイト数を任意に指定してファイル・サイズを表示することができます。</p> <p><code>UNITS</code> キーワードで有効なオプションは、<code>BLOCKS</code> および <code>BYTES</code> です。</p> <p><code>UNITS</code> にオプションを付けずに指定すると、省略時の設定値は変更されません。</p>
<code>USED</code>	使用されているブロック数を、ファイル・サイズとして表示します。

このフィールドの幅は、`/WIDTH` 修飾子の `SIZE` の値を与えることにより変更できます。

`/STYLE=キーワード,[キーワード]`
ディレクトリ内容を表示する場合に、表示するファイル名の書式を指定します。

この修飾子のキーワードは `CONDENSED` および `EXPANDED` です。意味は次の表のとおりです。

キーワード	説明
<code>CONDENSED</code> (省略時の設定)	ファイル名を 255 文字長の文字列に適合するように表示します。このファイル名の場合、ファイル指定に <code>DID</code> あるいは <code>FID</code> 短縮形を含むことが可能です。

キーワード	説明
EXPANDED	ファイル名をディスクに格納されているとおりに表示します。このファイル名の場合、ファイル指定に DID あるいは FID 短縮形は含みません。

CONDENSED および EXPANDED の両方のキーワードを指定した場合、ファイル指定が 2 列で表示されます。列のサイズは表示の幅に依存していて、ファイル名は各列の内部で自動改行されます。

EXPANDED キーワードが指定されていない場合、ファイル・エラーは CONDENSED ファイル指定で表示されます。

詳細は『OpenVMS システム管理者マニュアル(上巻)』を参照してください。

/TIME[=オプション]

/NOTIME (省略時の設定)

/DATE 修飾子と同じく、各々の指定されたファイルごとに、バックアップ日、作成日、満了日および変更日があります。省略時の設定は/NOTIME 修飾子です。オプションを指定しないで/TIME 修飾子を使用した場合は、作成日が表示されます。使用可能なオプションを次に示します。

オプション	説明
ALL	CREATED (作成日)、EXPIRED (満了日)、BACKUP(バックアップ日)、および MODIFIED (最終変更日)を指定する。
BACKUP	各ファイルの最新のバックアップ日を表示する。
CREATED	各ファイルの作成日を表示する。
EFFECTIVE	ファイルの内容が正しい有効日を表示する。
EXPIRED	各ファイルの満了日を表示する。
MODIFIED	ファイルが最後に変更された日を表示する。
RECORDING	記憶媒体上の記録日を表示する。

/TOTAL

ディレクトリ名とファイルの総数だけを表示します。

省略時には、出力形式は/BRIEF 修飾子によって決定されます。この場合には、総数だけでなくファイル名、ファイル・タイプ、バージョン番号もすべて表示されます。

/TRAILING

/NOTRAILING

下記の情報が要約された形で、後続行に出力されるかどうかを制御します。

- 表示されたファイルの総数
- ディレクトリごとに使用されているブロックの総数
- 割り当てられているブロックの総数

- ディレクトリの総数と、すべてのディレクトリによって使用されているブロック数または割り当てられているブロック数 (複数のディレクトリが表示されている場合のみ)

省略時の出力形式には、この要約情報のほとんどが含まれます。/SIZE 修飾子と /FULL 修飾子は、要約情報の中にどの情報を表示するかをより正確に指定します。

/TRAILING 修飾子が単独で使用されると、ディレクトリ内のファイルの数が表示されます。/SIZE 修飾子を同時に指定する場合には、/SIZE 修飾子に指定したサイズ・オプション (FULL または ALLOCATION のどちらか) に応じて、ブロック数も表示されます。/FULL 修飾子を同時に指定すると、使用されているブロック数および割り当てられているブロック数も表示されます。複数のディレクトリを表示する場合、要約情報にはディレクトリの総数、使用されているブロック数、割り当てられているブロック数が含まれます。

/VERSIONS=n

選択された各ファイルの最新の n バージョンが表示されることを指定します。

/VERSIONS 修飾子を省略した場合には、各ファイルのすべてのバージョンが表示されます。指定できる値は、1 以上の値です。

/WIDTH=(キーワード[,...])

表示の幅を規定します。キーワードを 1 つだけしか指定しない場合には、括弧を省略できます。次のキーワードを指定できます。

DISPLAY=n DIRECTORY 表示の全体の幅を指定します。n の値は、1 ~ 255 までの範囲の整数です。n の省略時の値は 0 で、DIRECTORY コマンドが、その表示の幅をターミナルの幅に設定することを示します。表示幅がターミナルの幅を越えている場合には、情報は途中で切り捨てられます。

FILENAME=n ファイル名を指定するためのフィールドの幅を指定します。省略時の値は 19 文字です。各カラムに、ファイル名の他の情報も表示することを指定した場合、ファイル名が n を越えると、ファイル名を切り捨てずに表示したあとで次の行に改行し、他の情報を表示します (/COLUMNS 修飾子を参照してください)。

OWNER=n 所有者を指定するためのフィールドの幅を指定します。省略時の値は 20 文字です。所有者のユーザ識別子 (UIC) が所有者フィールドの長さを越える場合には、情報は途中で切り捨てられます。

SIZE=n サイズ・フィールドの幅を指定します。V6.0 より前のバージョンの OpenVMS では、省略時の値は 6 です。OpenVMS V6.0 およびそれ以降では、省略時の値は 7 です。ファイル・サイズがサイズ・フィールドの長さを越える場合には、情報は途中で切り捨てられます。

/WRAP

/NOWRAP (省略時の設定)

/PAGE=SAVE 修飾子とともに使用し、画面の幅より長い行を改行表示するかどうかを指定します。/WRAP 修飾子を指定した場合、はみ出す部分は次行に表示されません。

/NOWRAP を指定した場合、はみ出した部分は左右にスクロールしながら見ることができます。

例

1. \$ DIRECTORY AVERAGE.*

Directory DISK\$DOCUMENT:[SOUDER]

AVERAGE.EXE;6 AVERAGE.FOR;6 AVERAGE.LIS;4 AVERAGE.OBJ;12

Total of 4 files.

この例では、DIRECTORY コマンドで AVERAGE というファイル名のすべてのファイル・タイプのファイルを表示します。

2. \$ DIRECTORY/SIZE=USED/DATE=CREATED/VERSIONS=1/PROTECTION -
_ \$ AVERAGE

Directory DISK\$DOCUMENT:[SLOUGH]

AVERAGE.EXE;6	6	19-DEC-2001 15:43:02.10	(RE,RE,RWED,RE)
AVERAGE.FOR;6	2	19-DEC-2001 10:29:53.37	(RE,RE,RWED,RE)
AVERAGE.LIS;4	5	19-DEC-2001 16:27:27.19	(RE,RE,RWED,RE)
AVERAGE.OBJ;6	2	19-DEC-2001 16:27:44.23	(RE,RE,RWED,RE)

Total of 4 files, 15 blocks.

この例では、DIRECTORY コマンドは、現在の省略時のディレクトリに含まれる AVERAGE という名前のすべてのファイルの最新バージョンの、ファイルの使用ブロック数、作成日と保護コードを表示します。

3. \$ DIRECTORY/FULL DISK\$GRIPS_2:[VMS.TV]DEMO.EXE

Directory DISK\$GRIPS_2:[VMS.TV]

```
DEMO.EXE;1                      File ID:  (36,11,0)
Size:          390/390           Owner:    [0,0]
Created: 12-NOV-2001 11:45:19.00
Revised: 14-DEC-2001 15:45:19.00 (34)
Expires:  <None specified>
Backup:   28-NOV-2001 04:00:12.22
Effective: <None specified>
Recording: <None specified>
File organization: Sequential
Shelved state:   Online
Caching attribute: Writethrough
File attributes: Allocation: 390, Extend: 0, Global buffer count: 0,
                  Version limit: 0, Backups disabled, Not shelveable
Record format:   Fixed length 512 byte records
Record attributes: None
RMS attributes:  None
Journaling enabled: None
File protection: System:RE, Owner:RE, Group:RE, World:RE
Access Cntrl List: None
Client attributes: None
```

Total of 1 file, 390/390 blocks.

この例は、DIRECTORY/FULL コマンドを説明しています。

4. \$ DIRECTORY/VERSIONS=1/COLUMNS=1 AVERAGE.*

この DIRECTORY コマンドは、現在の省略時のディレクトリ内でファイル名 AVERAGE を持つ最高バージョンの各ファイルだけを表示します。1 カラムの簡略形式で表示され、見出し行と後続行が付きます。

5. \$ DIRECTORY BLOCK%%

この DIRECTORY コマンドは、省略時の装置およびディレクトリに含まれるファイルの中から、名前が BLOCK から始まり、そのあとに任意の 3 文字が続くファイルのすべてのバージョンとすべてのファイル・タイプを表示します。省略時の出力形式は簡略形式であり、4 つのカラムが表示され、見出し行と後続行も表示されます。

6. \$ DIRECTORY/EXCLUDE=(AVER.DAT;*,AVER.EXE;*) [*...]AVER

この DIRECTORY コマンドは、省略時の装置の全ディレクトリにあって、ファイル名 AVER を持つすべてのファイルの全バージョンを表示します。ただし、AVER.DAT と AVER.EXE を除いています。

7. \$ DIRECTORY/SIZE=ALL FRESNO::DISK1:[TAMBA]*.COM

リモート・ノード FRESNO の装置 DISK1 のディレクトリ TAMBA 下にある、ファイル・タイプ COM のファイルの全バージョンを表示します。このリストには、ファイルの使用しているブロック・サイズ、および割り当てられているブロック・サイズも含まれます。

8. \$ DIRECTORY-
_ \$ /MODIFIED/SINCE=14-DEC-2001:01:30/SIZE=ALL/OWNER-
_ \$ /PROTECTION/OUTPUT=UPDATE/PRINTER [A*]

省略時の装置の A で始まる名前のトップ・ディレクトリ下で、2001 年 12 月 14 日の午前 1 時 30 分以降に更新されたファイルを表示します。全バージョンのファイルが、使用サイズ、割り当てサイズ、最新更新日付、所有者、保護コードとともに表示されます。この出力は、ファイル UPDATE.LIS に書かれ、自動的に省略時のプリンタ・キューに登録された後、削除されます。

9. \$ DIRECTORY/SHELVED_STATE

Directory MYDISK:[THOMPSON]

MYFILE.TXT;2	Online
NOT_SHELVED.TXT;1	Online
SHELVED.TXT	Shelved

Total of 3 files.

この例ではファイルがシェルブド、プリシェルブド、オンライン、リモートかを表示しています。

10. \$ DIRECTORY *.PS

Directory MYDISK:[TEST]

REPORT.PS;1 1197

Total of 1 file, 1197 blocks.

\$ DIRECTORY/SIZE=UNITS=BYTES *.PS

Directory \$1\$DKC600:[TEST]

REPORT.PS;1 598KB

Total of 1 file, 598KB

省略時の設定では、最初の DIRECTORY コマンドにより、ファイル・サイズがブロック単位で表示されます。次に 2 番目の DIRECTORY コマンドにより、ファイル・サイズがバイト単位で表示されます。

DISABLE AUTOSTART

1 ノード上で、特定のキュー・マネージャが管理する、すべての自動起動キューの自動起動機能を禁止します。省略時の設定では、このコマンドは/QUEUES 修飾子を使用します。

OPER (オペレータ) 特権が必要です。

自動起動キューについての詳細は、『OpenVMS システム管理者マニュアル』のバッチ・キューおよびプリント・キューに関する章を参照してください。

フォーマット

DISABLE AUTOSTART[/QUEUES]

パラメータ

なし

説明

DISABLE AUTOSTART/QUEUES コマンドが入力されると、キュー・マネージャは該当ノードで次の操作を行います。

- シャットダウンに備えて、キュー・マネージャが管理するすべての自動起動キューに“停止待ち状態”のマークをつける。
- キュー・マネージャの自動起動キューがそのノードにフェイルオーバーするのを抑止する。
- 該当ノードのキュー・マネージャの自動起動キューで現在実行中のいずれかのジョブが終了したら、自動起動が許可されていて、キューのフェイルオーバー・リスト内で次に使用可能なノードがあればそのノードに、キューをフェイルオーバーさせる(自動起動キューのフェイルオーバー・リストについての詳細は、INITIALIZE/QUEUE コマンドの/AUTOSTART_ON 修飾子を参照してください)。

フェイルオーバー・リストのないノード上の自動起動キュー、または自動起動のためのフェイルオーバー・ノードでない自動起動キューは、現在のジョブが終了すると停止します。このような停止したキューは、自動起動機能は残ります。該当ノード、またはキューがフェイルオーバーするノードに ENABLE AUTOSTART コマンドを入力されると、キュー・マネージャはこのような停止した自動起動キューを再起動させます。

省略時の設定では、このコマンドは、コマンドを入力したノードに対して有効です。異なるノード上の自動起動を禁止するには、/ON_NODE 修飾子を使用します。

DISABLE AUTOSTART/QUEUES コマンドは、ノードをシャットダウンするコマンド・プロシージャ SHUTDOWN.COM に含まれています。SHUTDOWN.COM を使用せずにあるノードをシャットダウンし、そのノードで自動起動キューが有効な場合は、まず DISABLE AUTOSTART/QUEUES コマンドを入力します。

DISABLE AUTOSTART/QUEUES コマンドは、自動起動キューに対してのみ影響があります。

修飾子

/NAME_OF_MANAGER=名前

禁止したい自動起動キューを制御しているキュー・マネージャ名を指定します。この修飾子を使用すると、キューの集合に対して異なる自動起動機能を使用することができます。

/NAME_OF_MANAGER 修飾子を省略すると、省略時のキュー・マネージャ名 SYS\$QUEUE_MANAGER が使用されます。複数のキュー・マネージャについての詳細は、『OpenVMS システム管理者マニュアル』を参照してください。

/ON_NODE=ノード名

OpenVMS Cluster システム内のノードを指定します。この修飾子を使用すると、このコマンドを入力したノード以外のノード上の、自動起動を禁止することができます。

/QUEUES

キューの自動起動を禁止することを指定します (この修飾子は省略時の設定により使用されます)。

例

```
1. $ INITIALIZE/QUEUE/BATCH/START/AUTOSTART_ON=SATURN:: BATCH_1
   $ ENABLE AUTOSTART/QUEUES
   .
   .
   .
   $ DISABLE AUTOSTART/QUEUES
```

この例では INITIALIZE/QUEUE コマンドは、ノード SATURN で実行できる自動起動キュー BATCH_1 を作成します。/START 修飾子は、キューの自動起動を有効にします。(ノード SATURN 上で実行される) ENABLE AUTOSTART

/QUEUES コマンドは該当ノードの自動起動を許可します。これによりキュー（および該当ノード上で有効な他の自動起動キュー）はジョブの実行を開始します。

（ノード SATURN 上で実行される）DISABLE AUTOSTART コマンドは該当ノード上の自動起動キューを停止し、他のキューが該当ノードにフェイルオーバーするのを抑止します。

/NAME_OF_MANAGER 修飾子が指定されていないので、これらのコマンドは省略時のキュー・マネージャ SYS\$QUEUE_MANAGER が管理するキューにのみ有効です。

BATCH_1 は 1 つのノードでのみ実行するように設定されているので、キューは他のノードにフェイルオーバーできず停止します。ただしキューは自動起動が有効であり、ノード SATURN に対して ENABLE AUTOSTART コマンドが入力されると、このキューはスタートします。STOP/QUEUE/NEXT または STOP/QUEUE/RESET コマンドでキューの自動起動を無効にしない限り、BATCH_1 をリスタートさせるために START/QUEUE コマンドを実行する必要はありません。

2. \$DISABLE AUTOSTART/QUEUES/ON_NODE=JADE

この例の DISABLE AUTOSTART/QUEUES コマンドは、OpenVMS Cluster ノード JADE の自動起動を禁止します。このコマンドは、クラスタ内の任意のノードから入力できます。

DISCONNECT

物理端末と仮想端末の間の接続を切断します。物理端末の接続が切断された後、その物理端末を使用する仮想端末およびプロセスはシステムに残ります。

物理端末と仮想端末が接続していなければなりません。

フォーマット

DISCONNECT

パラメータ

なし

説明

DISCONNECT コマンドを使用して、仮想端末および対応するプロセスから物理端末との接続を切断します。仮想端末とプロセスはシステム上に残るので、CONNECT コマンドを使用して後でそのプロセスに再接続することができます (仮想端末、および仮想端末への接続方法についての詳細は、CONNECT コマンドの説明を参照してください)。仮想端末と接続しているプロセスを終了するには、LOGOUT コマンドを使用します。

仮想端末との接続を切断した後、物理端末を使用して再度ログインすることができます。

使用している物理端末が仮想端末に接続している時のみ、DISCONNECT コマンドを使用できます。

修飾子

/CONTINUE

/NOCONTINUE (省略時の設定)

他のプロセスに接続する直前に、現在のプロセスで CONTINUE コマンドを実行するかどうかを制御します。この修飾子を指定すると、プロセスが端末からの入力を必要とする、または端末に書き込みを行おうとするまで、接続を切断した後も割り込みをかけられたイメージの実行を続けることができます。プロセスが端末からの入力を必

要とする時、または端末に書き込みを行おうとする時は、物理端末が仮想端末に再度接続するまで、プロセスは待ち状態になります。

例

1. \$ DISCONNECT

このコマンドは物理端末と仮想端末との接続を切断しますが、プロセスをログ・アウトしません。接続を切断した後でも、物理端末を使用して再度ログ・インできます。

2. \$ RUN PAYROLL

\$ DISCONNECT/CONTINUE

この例で RUN コマンドは、仮想端末に接続している物理端末から発行されています。イメージ PAYROLL.EXE に割り込みをかけられた後、DISCONNECT コマンドを使用して、プロセスをログ・アウトせずに物理端末と仮想端末の接続を切断します。/CONTINUE 修飾子を指定しているため、プロセスが端末からの入力が必要とする、または端末への書き込みを行おうとするまで、イメージ PAYROLL.EXE の実行を続けることができます。プロセスが端末からの入力が必要とする時、または端末に書き込みを行おうとする時は、物理端末が仮想端末に再度接続するまで、プロセスは待ち状態になります。ただし、物理端末を使用して再度ログインし、他の操作を行うことができます。

DISMOUNT

マウントしたディスクまたはテープ・ボリュームをクローズし、その装置に対応する論理名を削除します。

グループおよびシステムのボリュームをディスマウントするには、GRPNAM (グループ論理名) および SYSNAM (システム論理名) 特権が必要です。

フォーマット

DISMOUNT 装置名[:]

パラメータ

装置名[:]

ボリュームを含む装置名を指定します。論理名または物理名で指定します。物理名を指定した場合は、制御装置の省略時の設定は A に、ユニットの省略時の設定は 0 になります。

現在装置にマウントされているボリュームが、ディスク・ボリューム・セットまたはテープ・ボリューム・セットのメンバである場合は、/UNIT 修飾子を指定しない限り、セットのボリュームはすべてディスマウントされます。

説明

(\$DISMOU システム・サービスを起動する) DISMOUNT コマンドは、Files-11 構造のボリュームがディスマウントされないようにする条件について調べます。この条件は、次の 4 つに分類できます。

- スワップ・ファイルとページ・ファイルがインストールされている
- イメージがインストールされている
- 装置がボリュームにスプールされている
- ユーザ・ファイル (上記の 3 つに該当しないファイル) がオープンされている

これらの条件が 1 つも満たされない場合、DISMOUNT コマンドは次の操作を実行します。

- マウントされたボリュームのユーザ・リストからボリュームを削除し、そのボリュームに対応する論理名 (割り当てられている場合) を削除し、マウント・カウントから減らします。

- マウント・カウントを減らして0になった場合、DISMOUNT コマンドは、そのボリュームにディスマウントのマークをつけます。

ボリュームがアイドル状態になると、つまり、そのボリュームのファイルをオープンしているユーザはいないと DISMOUNT コマンドが判断した後は、DISMOUNT コマンドはFiles-11構造のボリュームにマークをつけて、すぐにディスマウントします。

- マウント・カウントを減らしても0にならない場合、DISMOUNT コマンドは、そのボリュームにディスマウントのマークをつけません(これは、そのボリュームが共用としてマウントされているからです)。この場合には、DISMOUNT コマンドを発行しても、そのプロセスはボリュームでアクセスできず、論理名が削除されます。
- ボリュームをディスマウントした後、システムに非ページング・プールが返されます。/GROUP または/SYSTEM 修飾子を使用してボリュームをマウントした場合も、ページング・プールは返されます。

オープンされたファイル、またはボリュームがディスマウントできない他の条件が検出されると、DISMOUNT コマンドはボリュームにディスマウントのマークをつけません。その代わりに、ボリュームがディスマウントできないことを示すメッセージを表示し、その後でディスマウントできない条件と各条件のインスタンス数を示すメッセージを表示します。

/OVERRIDE=CHECKS 修飾子を指定すると、オープンされたファイルやその他の条件に関係なく、ボリュームにマークをつけてディスマウントできます。たとえば、ボリュームにディスマウントのマークをつけると、新しいファイルはオープンできなくなります。また、ファイル・システムのキャッシュもフラッシュされます。システムをシャットダウンして、ファイル・システムのキャッシュをディスクに書き込まなければならない場合、この処理は特に重要になります。

ボリュームがFiles-11ボリューム・セットの一部であり、/UNIT 修飾子を指定していない場合、ボリューム・セット全体がディスマウントされます。

/SHARE 修飾子を指定してボリュームをマウントした場合、そのボリュームをマウントしたユーザがそのボリュームをディスマウントする、またはログアウトするまで、実際にはそのボリュームはディスマウントされません。ただし、DISMOUNT コマンドは、装置に対応する論理名を削除します。

ALLOCATE コマンドを使用して割り当てた装置は、DISMOUNT コマンドを使用してボリュームをディスマウントしても割り当てられたままです。装置が MOUNT コマンドで暗黙に割り当てられた場合は、DISMOUNT コマンドを使用してその割り当てを解除します。

/GROUP または/SYSTEM 修飾子を使用してマウントしたボリュームは、他のユーザが現在そのボリュームにアクセス中であってもディスマウントされます。グループお

よびシステムのボリュームをディスマウントするには、それぞれ GRPNAM 特権と SYSNAM 特権が必要です。

修飾子

/ABORT

/GROUP 修飾子も/SYSTEM 修飾子も指定せずにマウントしたボリュームに対してこの修飾子を指定するには、ボリュームの所有権、または VOLPRO (ボリューム保護) ユーザ特権が必要です。別のプロセスがプライベートにマウントしているボリュームの場合には、さらに SHARE ユーザ特権が必要です。

マウントしたユーザに関係なく、ディスマウントするボリュームを指定します。/ABORT 修飾子の主な目的は、マウント・チェックを終了することです。DISMOUNT/ABORT コマンドは、未処理の入出力要求もすべて取り消します。/SHARE 修飾子を指定してボリュームをマウントした場合、/ABORT 修飾子はマウントしたユーザに関係なく、ボリュームをディスマウントします。

/CLUSTER

複合アーキテクチャの OpenVMS Cluster システムを介して、ボリュームをディスマウントします。DISMOUNT/CLUSTER を指定すると、DISMOUNT コマンドは、ローカル・ノードの Files-11 構造のボリュームのディスマウントを妨げるような、オープンされたファイルやその他の条件がないかどうかを調べます。オープンされたファイルや他の条件が検出されなかった場合は、DISMOUNT コマンドは OpenVMS Cluster の他のノードについて条件を調べます。DISMOUNT コマンドがいずれかのノードで条件を検出した場合は、エラー・メッセージを表示して、エラーが発生した装置およびノードを示し、その後でオープンされたファイルまたはその他の条件を示すエラー・メッセージを表示します。

ローカル・ノードでボリュームを正常にディスマウントした後、DISMOUNT コマンドは既存の OpenVMS Cluster 環境内にある他のすべてのノードのボリュームをディスマウントします。システムがクラスタのメンバでない場合は、/CLUSTER 修飾子は作用しません。

/FORCE_REMOVAL ddcu:

指定したシャドウ・セット・メンバをシャドウ・セットから除外します。

装置との接続が切断されたときに、シャドウ・セットのマウント・チェックが行われている場合は、/FORCE_REMOVAL ddcu: を使用して、指定したシャドウ・セット・メンバ (ddcu:) をただちにシャドウ・セットから除外することができます。この修飾子を省略すると、その装置はマウント・チェックが完了するまでディスマウントされません。

この修飾子を /POLICY=MINICOPY (=OPTIONAL) 修飾子と同時に使用することはできません。

指定される装置は、コマンドが発行されたノードにマウントされているシャドウ・セットのメンバでなければなりません。

/OVERRIDE=CHECKS

そのボリューム内でファイルがオープンされている場合でも、Files-11構造のボリュームにディスマウントのマークをつけます。DISMOUNT/OVERRIDE=CHECKSを指定すると、DISMOUNT コマンドは、ディスマウントを妨げるオープンされたファイルまたはその他の条件を示すメッセージを表示し、その後で、ボリュームにディスマウントのマークをつけたことを示すメッセージを表示します。

このコマンドは、装置上のオープンされたファイルをクローズしません。ファイルをオープンしているすべてのプロセスがファイルを正しくクローズするまで、あるいはそれらのプロセスが完全にランダウンされるまで、装置を正しくディスマウントすることはできません。

DISMOUNT/OVERRIDE=CHECKS コマンドを実行してからディスマウント操作が終了するまでには、かなりの時間を要する場合があります。必ず、ディスマウントが終了するまで待ってから、ボリュームを取り外してください。SHOW DEVICE コマンドで、ディスマウントが終了したかどうか確認できます。ボリュームのディスマウントの最終処理はファイル・システムで行われ、実際にディスマウントできるのはそのボリュームのオープンされたファイルをすべてクローズしてからです。また、そのボリュームに既知ファイル・リストのエントリがある場合、そのボリュームをディスマウントすることはできません。

このコマンドを使用することにより、装置にはディスマウントのマークがつけられます。これにより、すでにオープンされたファイルのクローズ中に、その装置上のファイルをプロセスがオープンすることを防ぎます。

/POLICY=[NO]MINICOPY[=(OPTIONAL)] (Alpha/I64 のみ)

シャドウイング・ミニコピー機能の設定と使用を制御します。

ビットマップを作成するには、LOG_IO (論理 I/O) 特権が必要です。

MINICOPY キーワードの具体的な意味は、次のように DISMOUNT コマンドのコンテキストに依存します。

1. マルチメンバ・シャドウ・セットからの1つのメンバのディスマウントである場合には、シャドウ・セットへのすべての書き込みを追跡するための書き込みビットマップが作成されます。この書き込みビットマップは、削除されたメンバを後からミニコピーを使ってシャドウ・セットに戻すときに使用できます。

書き込みビットマップを作成することができず、キーワード OPTIONAL が指定されていなければ、ディスマウントは失敗し、メンバは削除されません。

/POLICY 修飾子を省略するか、/POLICY=NOMINICOPY を指定した場合には、ビットマップは作成されません。

2. クラスタ内のシャドウ・セットの最後のディスマウントである場合には、シャドウ・セットで将来ミニコピー操作を行えるかどうかを確認されます。

シャドウ・セットが1つのメンバしか持っていないか、またはマージ状態にあり、OPTIONAL が指定されていなかった場合には、ディスマウントは失敗します。

NOMINICOPY と MINICOPY のどちらも指定しなければ、MINICOPY=OPTIONAL と同じ意味になり、セットはそれ以前のチェックとは関係なくディスマウントされます。

詳細情報については『Volume Shadowing for OpenVMS 説明書』を参照してください。

/UNIT

指定した装置のボリューム・セットから、ボリュームを1つだけディスマウントします。省略時の設定では、セット内のボリュームをすべてディスマウントします。

注意

ボリューム・セットのルート・ボリュームにはマスタ・ファイル・ディレクトリ (MFD) があるので、ディスマウントしないでください。MFD にアクセスできない場合は、ボリューム・セット内のファイルにアクセスできないことがあります。

/UNLOAD

/NOUNLOAD

ボリュームがマウントされている装置を、物理的にアンロードするかどうかを決定します。/UNLOAD または /NOUNLOAD 修飾子を指定せずに DISMOUNT コマンドを指定した場合は、MOUNT コマンドで指定した修飾子 (/UNLOAD または /NOUNLOAD) により、ボリュームを物理的にアンロードするかどうかが決まります。

例

1. \$ MOUNT MTA0: PAYVOL TAPE

・
・
・

\$ DISMOUNT TAPE

この例で MOUNT コマンドは、PAYVOL というボリューム ID を持つテープを装置 MTA0: にマウントし、その装置に論理名 TAPE を割り当てます。省略時の設定では、ボリュームは共用可能ではありません。

この DISMOUNT コマンドはボリュームへのアクセスを解放し、装置の割り当てを解除して、論理名 TAPE を削除します。

2. \$ MOUNT/SHARE DKA3: DOC_FILES

·
·
·

\$ DISMOUNT DKA3:

この例で MOUNT コマンドは、DOC_FILES というラベルのボリュームを装置 DKA3 にマウントします。他のユーザは、MOUNT コマンドを実行してその装置にアクセスできます。DISMOUNT コマンドを実行すると、このコマンドを発行したプロセスはその装置にアクセスできません。他のユーザがこのボリュームにアクセスできる場合、このボリュームは、そのユーザのプロセスによりマウントされたままになっています。

3. \$ DISMOUNT/NOUNLOAD DMA2:

この例の DISMOUNT コマンドは、このボリュームをディスマウントします。
/NOUNLOAD 修飾子を指定しているので、このボリュームはレディ状態になります。

4. \$ MOUNT/BIND=PAYROLL DMA1:,DMA2: PAYROLL01,PAYROLL02

·
·
·

\$ DISMOUNT/UNIT DMA2:

この例の MOUNT コマンドは、2つのボリューム・セット PAYROLL をマウントします。DISMOUNT コマンドは PATROLL01 にはアクセスできるようにして、PAYROLL2 だけをディスマウントします。ボリューム・セットのマスタ・ファイル・ディレクトリ (MFD) はルート・ボリュームにあるため、ボリューム・セットのルート・ボリューム (この場合は PAYROLL1) をディスマウントしてはいけません。

5. \$ DISMOUNT \$10\$DJA100

```
%DISM-W-CANNOTDMT, $10$DJA100: cannot be dismounted
%DISM-W-INSWPGFIL, 4 swap or page files installed on volume
%DISM-W-SPOOLEDEV, 3 devices spooled to volume
%DISM-W-INSTIMAGE, 7 images installed on volume
%DISM-W-USERFILES, 6 user files open on volume
```

この例で DISMOUNT コマンドは、装置 \$10\$DJA100 のディスマウントを妨げている、オープンされたファイルやその他の条件を表示します。

6. \$ DISMOUNT/CLUSTER \$10\$DJA100

```
%DISM-W-RMTDMTFail, $10$DJA100: failed to dismount on node SALT
%DISM-W-FILESOPEN, volume has files open on remote node
%DISM-W-RMTDMTFail, $10$DJA100: failed to dismount on node PEPPER
%DISM-W-FILESOPEN, volume has files open on remote node
%DISM-W-CANNOTDMT, $10$DJA100: cannot be dismounted
```

この例で DISMOUNT コマンドは、エラーが発生した装置 \$10\$DJA100、および ノード SALT と PEPPER を示すメッセージを表示した後に、ボリュームのオープンされたファイルを示すメッセージを表示します。

DUMP

10 進，16 進，8 進形式，ASCII 形式，または書式付きデータ構造で，ファイル，ディレクトリ，ディスク・ボリューム，磁気テープ・ボリューム，または CD-ROM ボリュームの内容を表示します。このコマンドは，プロセス・ダンプを作成するのに使用できます。

フォーマット

DUMP ファイル指定[...]

パラメータ

ファイル指定[...]

ダンプするファイルまたは装置名を指定します。

ここで指定した装置がディスクでも，テープでも，またはネットワーク装置でもない場合，あるいは装置が/FOREIGN 修飾子を指定してマウントされている場合には，ファイル指定には装置名だけが使用できます。

指定した装置が/FOREIGN 修飾子を指定しないでマウントされているネットワーク装置，ディスク装置，またはテープ装置の場合には，ファイル指定にワイルドカード文字のアスタリスク(*)とパーセント記号(%)を使用できます。

Files-11 C/D 形式の規格は，マウントされているボリューム，およびフォーリン・マウントされているボリュームでインプリメントされています。

説明

省略時の設定では，DUMP コマンドは ASCII 文字と 16 進ロングワードの両方で出力を編集します。基数修飾子 (/OCTAL，/DECIMAL，/HEXADECIMAL)，または長さ修飾子 (/BYTE，/WORD，/LONGWORD) を使用すると，別の形式を指定できます。

ファイルのダンプ

入力媒体が，/FOREIGN 修飾子を指定せずにマウントしたネットワーク装置，ディスク装置，またはテープ装置である場合，DUMP コマンドはファイルに作用します。ファイルは，レコード単位またはブロック単位でダンプできます。ワイルドカード文字のアスタリスク(*)とパーセント記号(%)を指定して，処理する複数のファイルを選択することもできます。

ボリュームのダンプ

入力媒体がディスク装置でもテープ装置でもない場合、または/FOREIGN 修飾子を指定してマウントされている場合、DUMP コマンドは、非ファイル構造 (NFS) 媒体としての入力装置に作用します。ディスク装置は、512 バイトの論理ブロックごとにダンプされます。その他の装置は、物理ブロックごとにダンプされます。入力媒体の再位置付けは行われません。そのため、1 つの DUMP コマンドでテープの連続したブロックをダンプすることができます。

LOG_IO(論理入出力) 特権を持っている場合には、Files-11 構造のボリュームのどのブロックでもダンプできます。たとえば、/BLOCKS 修飾子を使用して、システム・ディスクの 100 番ブロックをダンプできます。

プロセスのダンプ

/PROCESS 修飾子を使用すると、DUMP コマンドは、プロセス・ダンプ・ファイルの作成を試みます。

ダンプの読み込み

ASCII 形式では、読み込みは左から右に行われます。16 進表現、10 進表現、および 8 進表現では、読み込みは右から左に行われます。

修飾子の数値指定

/BLOCKS、/RECORDS、および/NUMBER 修飾子の数値は 10 進数で指定するか、または 16 進数、8 進数、10 進数を表すためにそれぞれの先頭に%X、%O、%D を付けて指定します。たとえば、10 進数の値 24 を指定する有効な方法を次に示します。

```
24
%X18
%O30
%D24
```

修飾子

/ALLOCATED

ファイルに割り当てられたすべてのブロックをダンプに含めます。省略時の設定では、ファイルの終端[EOF]に続くブロックはダンプに含まれません。

入力が/FOREIGN 修飾子を指定しないでマウントされているディスクの場合には、/ALLOCATE 修飾子を指定できます。/ALLOCATE 修飾子と/RECORDS 修飾子は、同時に指定できません。

/BLOCKS[=(オプション[,...])]

指定された 1 つまたは複数のブロックを、一度に 1 ブロックずつダンプします。ネットワーク装置を除くすべての装置では、これが省略時の設定の動作です。

ブロック番号は、ファイルの先頭を基準にした整数値を指定します。通常、ブロックには1から始まる番号が付いています。ディスク装置が/FOREIGN 修飾子を使用してマウントされている場合は、ブロックには0から始まる番号が付けられます。次のオプションのいずれかを指定して、ダンプするブロックの範囲を選択します。

START:n	ダンプする最初のブロックの番号を指定します。省略時の設定は、最初のブロックです。
END:n	ダンプする最後のブロックの番号を指定します。省略時の設定は、最後のブロック、またはファイルの終端 (EOF) ブロックです。これは、/ALLOCATE 修飾子を指定したかどうかによって決まります。
COUNT:n	ダンプするブロック数を指定します。COUNT オプションは、END オプションの代わりに使用できます。両方を同時に指定することはできません。

オプションを1つだけ指定する場合は、括弧を省略できます。

/BLOCKS 修飾子と/RECORDS 修飾子を同時に指定することはできません。

Files-11構造のボリュームの任意のブロックをダンプするには、/BLOCKS 修飾子を使用します。この操作を行うためには、LOG-IO(論理入出力) 特権が必要です。

/BYTE

ダンプをバイト単位で編集します。/BYTE、/LONGWORD、および/WORD 修飾子を同時に指定することはできません。省略時の形式は、ロングワードです。

/DECIMAL

ファイルを10進数でダンプします。/DECIMAL、/HEXADECIMAL (省略時の設定)、および/OCTAL 修飾子を同時に指定することはできません。

/DESCRIPTOR[=(オプション[,...])]

指定したISO 9660 ボリューム記述子をダンプします。/NOFORMATTED を指定している場合、ブロック単位で編集します。

指定できる記述子オプションを次に示します。

BOOT:n	n番目の Boot Record を検索します。
PVD:n	n番目の Primary Volume Descriptor を検索します。
SVD:n	n番目の Supplementary Volume Descriptor を検索します。
VPD:n	n番目の Volume Partition Descriptor を検索します。
VDST:n	n番目の Volume Descriptor Set Terminator を検索します。

オプションを1つだけ指定する場合は、括弧を省略できます。

ISO 9660 記述子は、ボリュームの最初からの位置で指定します。特に指定しない限り、省略時の値は1です。ISO 9660 ボリュームは、指定した記述子を探し編集して出力するために、ボリューム記述子セット・シーケンスの最初から最後まで順番に検索されます。

/DIRECTORY

指定したファイルのデータ・ブロックを、Files-11ディスク構造レベル1，2，または5ディレクトリ・レコード，ISO 9660，あるいはHigh Sierraディレクトリ・レコードに，書式付きディスク構造としてダンプします。

/EXACT

/PAGE=SAVE 修飾子と**/SEARCH** 修飾子とともに使用し，正確に一致した文字列を検索することを指定します。この場合，検索文字列は二重引用符(“ ”)で囲まなければなりません。

/SEARCH 修飾子を指定せずに**/EXTRACT** 修飾子を指定した場合は，Find (E1) キーを押すと文字列検索が有効になります。

/FILE_HEADER

有効なFiles-11ヘッダの各データ・ブロックを，選択した基数と長さの形式ではなく，Files-11ヘッダ形式でダンプします。

/FORMATTED (省略時の設定)**/NOFORMATTED**

Files-11形式でファイル・ヘッダをダンプします。**/NOFORMATTED** 修飾子は，8進形式でファイル・ヘッダをダンプします。**/HEADER** 修飾子を指定した場合，この修飾子は便利です。

/HEADER

ファイル・ヘッダとアクセス制御リスト (ACL) をダンプします。ファイル・ヘッダだけをダンプするには，**/BLOCK=(COUNT:0)** も指定します。**/HEADER** 修飾子は，**/FOREIGN** 修飾子を使用してマウントされた装置に対しては無効です。

表示形式を変更するには，**/FORMATTED** 修飾子を使用します。

/FILE_HEADER 修飾子とともに**/HEADER** 修飾子を指定すると，解釈された形式でFiles-11ファイル・ヘッダを出力できます。

省略時の設定では，ファイル・ヘッダは表示されません。

/HEXADECIMAL (省略時の設定)

ファイルを16進数でダンプします。**/DECIMAL**，**/HEXADECIMAL**(省略時の設定)，および**/OCTAL** 修飾子は同時に指定できません。

/HIGHLIGHT[=キーワード]

/PAGE=SAVE 修飾子および**/SEARCH** 修飾子とともに使用し，一致した検索文字列がの強調表示方法を指定します。一致した検索文字列があった場合は，行全体が強調表示されます。キーワードにはBOLD，BLINK，REVERSE，UNDERLINEが指定できます。省略時の設定はBOLDです。

/IDENTIFIER=ファイル識別子

指定したボリュームから、ファイル識別子 (FID) 番号によって選択されたファイルをダンプします。詳細は、DIRECTORY コマンドの/FILE_ID 修飾子を参照してください。

/LONGWORD (省略時の設定)

ダンプをロングワード単位で編集します。/BYTE、/LONGWORD、および/WORD 修飾子を同時に指定することはできません。

/MEDIA_FORMAT=キーワード

データ構造をダンプする形式を指定します。この修飾子を指定した場合は、次のキーワードのいずれか 1 つを使用しなければなりません。

CDROM ISO 9660 メディア形式でダンプすることを指定します。これは、/MEDIA_FORMAT 修飾子を指定しなかった場合の省略時の設定です。

CDROM_HS High Sierra メディア形式でダンプすることを指定します。

/NUMBER[=n]

出力行へのバイト・オフセットの割り当て方法を指定します。/NUMBER 修飾子を指定すると、バイト・オフセットはnから始まり、ダンプ中継続的に増加します。/NUMBER 修飾子を省略すると、最初のバイト・オフセットはゼロになります。省略時の設定では、各ブロックまたはレコードの先頭で、バイト・オフセットはゼロに再設定されます。

/OCTAL

ファイルを 8 進数でダンプします。/DECIMAL、/HEXADECIMAL (省略時の設定)、および/OCTAL 修飾子を同時に指定することはできません。

/OUTPUT[=ファイル指定]

ダンプ用の出力ファイルを指定します。ファイルを指定しないと、省略時の設定により、ダンプするファイルのファイル名とファイル・タイプ.DMP が指定されます。/OUTPUT 修飾子を指定しない場合は、ダンプは SYS\$OUTPUT に出力されます。/OUTPUT および/PRINTER 修飾子は同時に指定できません。

/PAGE[=キーワード]

/NOPAGE (省略時の設定)

ダンプ情報の画面表示を制御します。

/PAGE 修飾子とともに次のキーワードを使用できます。

CLEAR_SCREEN 各ページが表示される前に、画面を消去します。

SCROLL 情報を一度に 1 行ずつ表示します。

SAVE[=n] 情報の画面を移動できるようにします。nは、格納するページ数です。

/PAGE=SAVE 修飾子を使用すると、画面上の情報の間を移動できます。

/PAGE=SAVE 修飾子では、最大 255 カラムの情報を 5 画面まで格納できます。

/PAGE=SAVE 修飾子を使用する場合には、次のキーを使用して情報間を移動できます。

キー	説明
Up arrow key, Ctrl/B	1 行上にスクロールします。
Down arrow key	1 行下にスクロールします。
Left arrow key	1 つ左にスクロールします。
Right arrow key	1 つ右にスクロールします。
Find (E1)	情報が表示された時に探す文字列を指定します。
Insert Here (E2)	半画面右にスクロールします。
Remove (E3)	半画面左にスクロールします。
Select (E4)	80 カラム・モードと 132 カラム・モードを切り替えます。
Prev Screen (E5)	1 つ前のページの情報を表示します。
Next Screen (E6), Return, Enter, Space	次のページの情報を表示します。
F10, Ctrl/Z	終了します (設定が異なるユーティリティもあります)。
Help (F15)	ユーティリティのヘルプ・テキストを表示します。
Do (F16)	最も古いページと最新のページの表示を切り替えます。
Ctrl/W	再表示します。

/PAGE 修飾子は/OUTPUT 修飾子と同時に指定できません。

/PATH_TABLE

ISO 9660 パス・テーブル形式でデータ・ブロックをダンプします。

/PRINTER

ファイル名がダンプするファイルと同じでファイル・タイプが.DMP であるファイルとして、ダンプの出力を SYS\$PRINT にキュー登録します。/PRINTER 修飾子を指定しない場合は、ダンプは SYS\$OUTPUT に送られます。ワイルドカード文字のアスタリスク(*)とパーセント記号(%)は使用できません。/OUTPUT 修飾子と/PRINTER 修飾子は同時に指定できません。

/PROCESS [/ID=pid] [プロセス名]

プロセス・ダンプの作成を試みます。省略時のプロセスは、現在のプロセスになります。目的のプロセス・ダンプを作成するには、プロセス ID またはプロセス名のどちらかを指定します。

/RECORDS[=(オプション[,...])]

1 度に 1 ブロックずつではなく、1 度に 1 レコードずつファイルをダンプします。省略時の設定では、入力ネットワーク装置を除くすべての装置で、一度に 1 ブロックずつダンプします。

各レコードには、1 から始まる番号が付けられます。

次のいずれかのオプションを指定して、ダンプするレコードの範囲を選択します。

START:n	ダンプする最初のレコードの番号を指定します。省略時の設定は、最初のレコードです。
END:n	ダンプする最後のレコードの番号を指定します。省略時の設定は、最後のレコードです。
COUNT:n	ダンプするレコードの数を指定します。COUNT オプションは、END オプションの代わりに使用できません。両方を同時に指定することはできません。

オプションを 1 つだけ指定する場合は、括弧を省略できます。

/RECORDS 修飾子を指定する場合は、/ALLOCATED 修飾子や/BLOCKS 修飾子は指定できません。

/SEARCH="文字列 "

/PAGE=SAVE 修飾子とともに指定して、表示された情報の中で検索したい文字列を指定します。文字列にスペース文字等を入れたい場合は、文字列全体を二重引用符で囲まなければなりません。

情報が表示されている間に Find キー (E1) を押すと、検索文字列も動的に変更することができます。この場合は、二重引用符は必要ありません。

/STYLE=キーワード

ファイル・ダンプ中に表示するファイル名の書式を指定します。

この修飾子のキーワードは CONDENSED および EXPANDED です。意味は次の表のとおりです。

キーワード	説明
CONDENSED (省略時の設定)	ファイル名を 255 文字長の文字列に適合するように表示します。このファイル名の場合、ファイル指定に DID あるいは FID 短縮形を含むことが可能です。
EXPANDED	ファイル名をディスクに格納されているとおりに表示します。このファイル名の場合、ファイル指定に DID あるいは FID 短縮形は含みません。

キーワード CONDENSED と EXPANDED を同時に指定することはできません。この修飾子は、出力ヘッダに表示されるファイル名の書式を指定します。

EXPANDED キーワードが指定されていない場合、ファイル・エラーは CONDENSED ファイル指定で表示されます。

詳細は『OpenVMS ユーザーズ・マニュアル』を参照してください。

/VALIDATE_HEADER

Files-11の/DIRECTORY レコードを確認します。

/WIDTH=n

nに 80 または 132 を指定して、ダンプ出力を 80 または 132 カラムに編集します。

DUMP

/WORD

ダンプをワード単位で編集します。/BYTE, /LONGWORD, および/WORD 修飾子を同時に指定することはできません。

/WRAP

/NOWRAP (省略時の設定)

/PAGE=SAVE 修飾子とともに指定して、画面幅を制限し、画面幅より長い行を次の行へ自動的に改行します。

/NOWRAP 修飾子は画面の幅以上に行を拡張しますが、/PAGE=SAVE 修飾子で用意された(左右の)スクロール機能を使用すれば、画面に表示されていない部分も見ることができます。

例

```
1. $ DUMP TEST.DAT
Dump of file DISK0:[MOORE]TEST.DAT;1 on 14-DEC-2001 15:43:26.08
File ID (3134,818,2)   End of file block 1 / Allocated 3
Virtual block number 1 (00000001), 512 (0200) bytes
 706D6173 20612073 69207369 68540033 3.This is a samp 000000
 73752065 62206F74 20656C69 6620656C le file to be us 000010
 61786520 504D5544 2061206E 69206465 ed in a DUMP exa 000020
 00000000 00000000 0000002E 656C706D mple..... 000030
 00000000 00000000 00000000 00000000 ..... 000040
 00000000 00000000 00000000 00000000 ..... 000050
 00000000 00000000 00000000 00000000 ..... 000060
.
.
.
 00000000 00000000 00000000 00000000 ..... 0001E0
 00000000 00000000 00000000 00000000 ..... 0001F0
```

この DUMP コマンドは、16 進ロングワード形式と ASCII 形式の両方で、ファイルの先頭のブロックから始まる TEST.DAT の内容を表示します。

```

2. $ DUMP TEST.DAT/OCTAL/BYTE
Dump of file DISK0:[SCHELL]TEST.DAT;1 on 14-DEC-2001 15:45:33.58
File ID (74931,2,1)   End of file block 1 / Allocated 3
Virtual block number 1 (00000001), 512 (0200) bytes
151 040 163 151 150 124 000 063 3.This i 000000
160 155 141 163 040 141 040 163 s a samp 000010
040 145 154 151 146 040 145 154 le file 000020
163 165 040 145 142 040 157 164 to be us 000030
040 141 040 156 151 040 144 145 ed in a 000040
141 170 145 040 120 115 125 104 DUMP exa 000050
377 377 000 056 145 154 160 155 mple.... 000060
000 000 000 000 000 000 000 000 ..... 000070
000 000 000 000 000 000 000 000 ..... 000100
000 000 000 000 000 000 000 000 ..... 000110
.
.
.
000 000 000 000 000 000 000 000 ..... 000760
000 000 000 000 000 000 000 000 ..... 000770

```

この DUMP コマンドは、8 進バイトと ASCII 文字の両方で編集して、ファイルの最初のブロックから始まる TEST.DAT のイメージを表示します。

```

3. $ DUMP NODE3::DISK2:[STATISTICS]RUN1.DAT

```

このコマンド行では、リモート・ノード NODE3 にあるファイル RUN1.DAT をダンプします。省略時の DUMP 形式が使用されます。

```

4. $ DUMP/HEADER/BLOCK=COUNT=0 SYS$SYSTEM:DATASHARE.EXE

```

```

Dump of file SYS$SYSTEM:DATASHARE.EXE on 12-NOV-2001 16:06:46.75
File ID (16706,59,0)   End of file block 410 / Allocated 411

```

File Header

Header area

```

Identification area offset:      40
Map area offset:                 100
Access control area offset:      255
Reserved area offset:           255
Extension segment number:        0
Structure level and version:     2, 1
File identification:              (16706,59,0)
Extension file identification:    (0,0,0)
VAX RMS attributes
Record type:                     Fixed
File organization:               Sequential
Record attributes:               <none specified>
Record size:                     512
Highest block:                   411
End of file block:               410
End of file byte:                414
Bucket size:                     0
Fixed control area size:         0
Maximum record size:             512
Default extension size:          0

```

DUMP

```

Global buffer count:      0
Directory version limit:  0
File characteristics:     Contiguous best try
Caching attribute:       Writethrough
Map area words in use:    3
Access mode:              0
File owner UIC:           [1,4]
File protection:          S:RWED, O:RWED, G:RE, W:
Back link file identification: (7149,80,0)
Journal control flags:    <none specified>
Active recovery units:    None
Highest block written:    411
Client attributes:        None

Identification area
File name:                DATASHARE.EXE
Revision number:          1
Creation date:             12-AUG-2001 14:06:49.84
Revision date:             12-AUG-2001 14:06:53.20
Expiration date:          <none specified>
Backup date:              <none specified>

Map area
Retrieval pointers
Count:      411      LBN: 1297155

Checksum:      30710

```

この例で DUMP コマンドは、指定したファイルのファイル・ヘッダをダンプします。このファイルはFiles-11 ODS-2 9660 媒体に記録されているため、ファイル・ヘッダはFiles-11 File Header 形式で表示されます。Files-11 Header に埋め込まれているのは、VAX RMS属性ブロックです。

5. \$ DUMP/HEADER/BLOCK=COUNT=0 DISK\$GRIPS_2:[000000]AAREADME.TXT;
 Dump of file DISK\$GRIPS_2:[000000]AAREADME.TXT;1 on 15-DEC-2001
 10:07:29.70
 File ID (4,6,0) End of file block 29 / Allocated 29
 ISO 9660 File Header
 Length of Directory Record: 48
 Extended Attribute Length: 1
 Location of Extent (LSB/MSB): 312/312
 Data Length of File Section (LSB/MSB): 14640/14640
 Recording Date and Time 10-DEC-2001 16:22:30 GMT(0)
 File Flags RECORD, PROTECTION
 Interleave File Unit size: 0
 Interleave Gap size: 0
 Volume Sequence # of extent (LSB/MSB): 1/1
 File Identifier Field Length: 14
 File Identifier: AAREADME.TXT;1
 System Use
 5458542E 454D4441 45524141 0E010000 01000018 001E1610 100B5930 39000000
 ...90Y.....AAREADME.TXT 000000
 00313B
 ;1..... 000020


```

Extended Attribute record
  Owner Identification (LSB/MSB):      7/7
  Group Identification (LSB/MSB):     246/246
  Access permission for classes of users S:R, O:R, G:RE, W:RE
  File Creation Date/Time:            5-OCT-2001 14:17:49.29 GMT(0)
  File Modification Date/Time:        6-NOV-2001 16:22:30.96 GMT(0)
  File Expiration Date/Time:          00-00-0000 00:00:00.00 GMT(0)
  File Effective Date/Time:           00-00-0000 00:00:00.00 GMT(0)
  Record Format                        Fixed
  Record Attributes                    CRLF
  Record Length (LSB/MSB):            80/80
  System Identifier:
  System Use
  Extended Attribute Version:          1
  Escape Sequence record length:       0
  Application Use Length (LSB/MSB):    0/0
  Application Use

VAX RMS attributes
  Record type:                        Fixed
  File organization:                  Sequential
  Record attributes:                  Implied carriage control
  Record size:                        80
  Highest block:                      29
  End of file block:                  29
  End of file byte:                  304
  Bucket size:                        0
  Fixed control area size:            0
  Maximum record size:                80
  Default extension size:             0
  Global buffer count:                0
  Directory version limit:            0

```

DUMP/HEADER コマンドは、指定したファイルのファイル・ヘッダをダンプします。このファイルは ISO 9660 媒体に記録されているため、ファイル・ヘッダは ISO 9660 File Header 形式で表示されます。またオプションの ISO 9660 Extended Attribute Record (XAR) もあるため、それ也表示されます。最後に、DUMP/HEADER コマンドの要求どおりに、VAX RMS属性が表示されます。

EDIT/ACL

オブジェクトのアクセス制御リスト (ACL) を作成または変更するために、アクセス制御リスト・エディタ (ACL エディタ) を起動します。これには/ACL 修飾子が必要です。

ACL エディタについての詳細は、『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』または『OpenVMS システム・セキュリティ・ガイド』またはオンライン・ヘルプを参照してください。

フォーマット

EDIT/ACL オブジェクト指定

EDIT/EDT

EDT 会話型テキスト・エディタを起動します。/EDT 修飾子は必須です。

EDT コマンドの情報は、Ctrl/Z を押しながら EDT コマンド・プロンプトで HELP と入力することによって EDT 内から使用できます。キーパッド・ヘルプは、コマンド・ヘルプの他に、PF2 を押すことによっても使用できます。EDT コマンドおよび修飾子の情報を含む、EDT コマンドの説明については、『OpenVMS ユーザーズ・マニュアル』を参照してください。

フォーマット

EDIT/EDT ファイル指定

パラメータ

ファイル指定

EDT エディタを使用して作成、または編集するファイルを指定します。指定したファイルが存在しない場合には、EDT がそのファイルを作成します。

EDT エディタがファイルを作成する場合、そのファイルに、省略時のファイル・タイプは与えられません。したがって、ファイル・タイプは、指定しない場合には空文字列になります。ファイルは、Files-11 構造のボリューム上のディスク・ファイルでなければなりません。

ファイル指定の中で、ワイルドカード文字は使用できません。

説明

EDT は、テキスト・ファイルを作成または編集します。EDT を使用すると、キーパッド、行、非キーパッドという 3 つのモードで、テキストを入力または編集できます。キーパッドによる編集は、画面用であり、VT300 シリーズ、VT200 シリーズ、VT100 端末、および VT52 端末で使用できます。画面指向エディタを使用すると、一度に数行のテキストを参照し、カーソルをテキストの任意の方向に移動できます。行編集は、すべての端末で動作します。実際に、VT300 シリーズ、VT200 シリーズ、VT1000、または VT52 以外の端末では、EDT では行編集しか使用できません。番号が付いた行の編集に慣れている場合は、行編集の方が使いやすいたことがあります。非キーパッド・モードは、VT300 シリーズ、VT200 シリーズ、VT100 端末、および VT52 端末で利用できるコマンド指向の画面エディタです。行モードおよび非キーパ

ッド・モードを使用して、キーパッド・モードで使用するキーを再定義することができます。

EDT の起動時の省略時の設定は、行モードです。既存のファイルを編集する場合、EDT はファイルの第 1 行の行番号とテキストを表示します。新しいファイルを作成する場合、EDT は次のメッセージを表示します。

```
Input file does not exist
[EOB]
```

いずれの場合でも、次に EDT は行モードプロンプト (アスタリスク(*)) を表示します。

EDT エディタについての詳細は、『OpenVMS EDT Reference Manual』(ドキュメンテーション CD-ROM に用意されています) を参照してください。

修飾子

```
/COMMAND[=ファイル指定]
/NOCOMMAND
```

EDT がスタート・アップ・コマンド・ファイルを使用するかどうかを指定します。/COMMAND ファイル修飾子のあとに、等号(=)に続けてコマンド・ファイルを指定します。コマンド・ファイルの省略時のファイル・タイプは、EDT です。ファイル指定の中で、ワイルドカード文字は使用できません。

次のコマンド行は、MEMO.DAT という名前のファイルを編集するために EDT を起動し、EDT が XEDTINI.EDT という名前のスタート・アップ・コマンド・ファイルを使用することを指定しています。

```
$ EDIT/COMMAND=XEDTINI.EDT MEMO.DAT
```

"/COMMAND=コマンド・ファイル"修飾子を指定しない場合には、EDT は EDTSYS という論理名の定義を調べます。EDTSYS が定義されていない場合には、システム全体で使用される SYS\$LIBRARY:EDTSYS.EDT というスタート・アップ・コマンド・ファイルを使用します。このファイルが存在しない場合には、EDTINI という論理名の定義を調べます。EDTINI が定義されていない場合には、省略時のディレクトリから EDTINI.EDT という名前のファイルを探します。これらのファイルがどれも存在しない場合には、省略時の状態で編集セッションを開始します。

システム全体で使用されるスタート・アップ・コマンド・ファイルも、省略時のディレクトリにある EDTINI.EDT ファイルも、EDT が実行しないようにするには、次のように /NOCOMMAND 修飾子を使用します。

```
$ EDIT/EDT/NOCOMMAND MEMO.DAT
```

/CREATE (省略時の設定)

/NOCREATE

指定した入力ファイルが存在しない時に、EDT が新しいファイルを作成するかどうかを制御します。

一般に、指定したディレクトリから要求したファイル名を見つけることができない場合、EDT は入力ファイル指定と一致する名前の新しいファイルを作成します。EDT コマンド行に/NOCREATE 修飾子を指定し、存在しないファイルのファイル指定を入力すると、EDT はエラー・メッセージを表示し、以下のように、DCL コマンド・レベルに戻ります。

```
$ EDIT/EDT/NOCREATE NEWFILE.DAT
Input file does not exist
$
```

/JOURNAL[=ジャーナル・ファイル]

/NOJOURNAL

EDT が、編集セッションをジャーナル・ファイルに保存するかどうかを制御します。ジャーナル・ファイルは、編集セッション中に入力されるキーストロークを記録します。ジャーナル・ファイルの省略時のファイル名は、入力ファイルの名前と同じで、ファイル・タイプはJOU です。/JOURNAL 修飾子を使用すれば、ジャーナル・ファイルに対して、入力ファイルとは別のファイル指定を使用できます。

次のコマンド行は、MEMO.DAT という名前のファイルを編集するために EDT を起動し、ジャーナル・ファイルとして SAVE.JOU という名前を指定しています。

```
$ EDIT/EDT/JOURNAL=SAVE MEMO.DAT
```

他のディレクトリのファイルを編集している時に、ジャーナル・ファイルをそのディレクトリに作成する場合には、/JOURNAL 修飾子に、そのディレクトリ名を含むファイル指定をしなければなりません。そうしない場合、EDT はジャーナル・ファイルを省略時のディレクトリに作成します。

ジャーナル・ファイルが作成されるディレクトリは、書き込みアクセスを禁止されています。

EDT が編集セッションの記録を保存しないようにしたい場合には、次に示すように、EDT コマンド行に/NOJOURNAL 修飾子を指定します。

```
$ EDIT/EDT/NOJOURNAL MEMO.DAT
```

ジャーナル・ファイルを作成したあと、ジャーナル・ファイルに含まれているコマンドを処理するように、EDT/RECOVER コマンドを使用します。ファイル指定の中で、ワイルドカード文字は使用できません。

```
/OUTPUT=出力ファイル
/NOOUTPUT
```

EDT が編集セッションの最後に、出力ファイルを作成するかどうかを指定します。入力ファイルと出力ファイルの省略時のファイル指定は、同じです。出力ファイルに、入力ファイルと異なるファイル指定を与える場合には、/OUTPUT 修飾子を使用します。

次のコマンド行は、MEMO.DAT という名前のファイルを編集するために EDT を起動し、この編集セッションで作成される出力ファイルに、OUTMEM.DAT という名前を与えます。

```
$ EDIT/EDT/OUTPUT=OUTMEM.DAT MEMO.DAT
```

次の例に示すように、他のディレクトリにファイルを作成する場合には、出力ファイル指定の一部としてディレクトリ情報を含むことができます。

```
$ EDIT/EDT/OUTPUT=[BARRETT.MAIL]MEMO.DAT MEMO.DAT
```

/NOOUTPUT 修飾子は、出力ファイルが作成されないようにします。しかし、ジャーナル・ファイルの作成には影響を与えません。出力ファイルを作成しない場合には、次のように/NOOUTPUT を使用します。

```
$ EDIT/EDT/NOOUTPUT MEMO.DAT
```

システムが割り込みをかけても、ジャーナル・ファイルはまだ保存されているため、編集セッションをもう一度実行する必要はありません。/NOOUTPUT が指定されている場合でも、出力ファイルを作成する場合には、WRITE というライン・モード・コマンドを使って、セッションを終了する前にテキストを外部ファイルに書き込みます。

ファイル指定の中で、ワイルドカード文字は使用できません。

```
/READ_ONLY
/NOREAD_ONLY (省略時の設定)
```

EDT がジャーナル・ファイルを保存し、出力ファイルを作成するかどうかを指定します。/NOREAD_ONLY 修飾子が指定されると、EDT はジャーナル・ファイルを保存し、EXIT というライン・モード・コマンドを実行する時に、出力ファイルを作成します。/READ_ONLY 修飾子を使用した結果は、/NOJOURNAL 修飾子と/NOOUTPUT 修飾子の両方を使用した場合と同じです。

次のコマンド行は、CALENDAR.DAT という名前のファイルを編集するために EDT を起動しますが、ジャーナル・ファイルも出力ファイルも作成しません。

```
$ EDIT/EDT/READ_ONLY CALENDAR.DAT
```

ファイル内を検索するだけで変更しない場合には、/READ_ONLY 修飾子を使用します。ファイルを変更する場合には、ライン・モード・コマンドである WRITE コマンドを使って変更結果を保存します。ただし、ジャーナル・ファイルは作成されません。

/RECOVER

/NORECOVER (省略時の設定)

編集セッションを開始するときに、EDT がジャーナル・ファイルを読み取るかどうかを指定します。

/RECOVER 修飾子を指定すると、EDT はジャーナル・ファイルを読み取り、そのファイルに含まれるコマンドを処理します。この場合の構文は、次のとおりです。

```
$ EDIT/EDT/RECOVER MEMO.DAT
```

ジャーナル・ファイルのファイル・タイプが JOU でない場合や、ファイル名が入力ファイルの名前と同じでない場合には、次の例に示すように、/JOURNAL 修飾子と /RECOVER 修飾子の両方を指定しなければなりません。

```
$ EDIT/EDT/RECOVER/JOURNAL=SAVE.XXX MEMO.DAT
```

/NORECOVER 修飾子は EDT の省略時の設定であるため、コマンド行にこの修飾子を指定する必要はありません。

例

1. \$ EDIT/EDT/OUTPUT=NEWFILE.TXT OLDFILE.TXT
 1 This is the first line of the file OLDFILE.TXT.
 *

この EDIT コマンドは、OLDFILE.TXT というファイルを編集するために、EDT エディタを起動します。この時 EDT は、EDTSYS という論理名の割り当てを調べます。EDTSYS が定義されていない場合には、EDT は、システムのスタート・アップ・コマンド・ファイルである SYS\$LIBRARY:EDTSYS.EDT を使用します。このファイルが存在しない場合には、EDTINI という論理名の割り当てを調べます。EDTINI が定義されていない場合には、省略時のディレクトリから、EDTINI.EDT という名前のファイルを見つけます。これらのファイルがどれも存在しない場合には、省略時の状態で編集セッションを開始します。セッションが終了したとき、編集されたファイルの名前は、NEWFILE.TXT になります。

2. \$ EDIT/EDT/RECOVER OLDFILE.TXT

この EDIT コマンドは、異常終了した前回の編集セッション回復するために EDT エディタを起動します。EDT は、OLDFILE.TXT というファイルをオープンし、OLDFILE.JOU というジャーナル・ファイルを処理します。ジャーナル・ファイルが処理された後で、会話型編集を再開できます。

EDIT/FDL

File Definition Language (FDL) ファイルの作成と変更のために、Edit/FDL ユーティリティ (EDIT/FDL) を起動します。/FDL 修飾子は必須です。

Edit/FDL ユーティリティについての詳細は、『OpenVMS Record Management Utilities Reference Manual』またはオンライン・ヘルプを参照してください。

フォーマット

EDIT/FDL ファイル指定

EDIT/SUM

SUMSLP バッチ型エディタを起動して、1つの入力ファイルを複数のファイルのエディタ・コマンドで更新します。

SUMSLP ユーティリティについての詳細は、『OpenVMS SUMSLP Utility Manual』（ドキュメンテーション CD-ROM に用意されています）、またはオンライン・ヘルプを参照してください。

フォーマット

EDIT/SUM 入力ファイル

EDIT/TECO

TECO 会話型テキスト・エディタを起動します。

フォーマット

EDIT/TECO [ファイル指定]

EDIT/TECO/EXECUTE=コマンド・ファイル [引数]

パラメータ

ファイル指定

TECO エディタを使用して作成または編集されるファイルを指定します。ファイルが存在しない場合には、/NOCREATE 修飾子を指定していない限り、TECO によって作成されます。ファイル指定にワイルドカード文字は使用できません。

ファイル指定なしに/MEMORY 修飾子を指定する（省略時の設定）と、TECO は論理名 TEC\$MEMORY で示されるファイルを編集します。TEC\$MEMORY が等価文字列を持たない場合、あるいは/NOMEMORY 修飾子が指定されている場合には、TECO はコマンド・モードに入り、既存ファイルの編集を行いません。

/MEMORY 修飾子とファイル指定の両方を指定すると、そのファイル指定は、論理名 TEC\$MEMORY と等価になります。

引数

/EXECUTE 修飾子の説明を参照してください。

説明

TECO エディタは、テキスト・ファイルを作成または編集します。TECO の使用方法についての詳細は、『Standard TECO Text Editor and Corrector for the VAX, PDP-11, PDP-10, and PDP-8』マニュアル（ドキュメンテーション CD-ROM に用意されています）を参照してください。

修飾子

/COMMAND[=ファイル名]
/NOCOMMAND

スタートアップ・コマンド・ファイルを使用するか否かを制御します。/COMMAND 修飾子の後には、等号(=)を使用して、そのコマンド・ファイルを指定することができます。コマンド・ファイルの省略時のファイル・タイプは TEC です。

スタートアップ・コマンド・ファイル XTECOINI.TEC を使って、ファイル MEMO.DAT を編集するには、次のコマンドを入力します。

```
$ EDIT/TECO/COMMAND=XTECOINI.TEC MEMO.DAT
```

/COMMAND 修飾子を指定しなかった場合、あるいはコマンド・ファイルを指定せずに /COMMAND 修飾子を指定した場合には、TECO は TEC\$INIT 論理名の定義を検索します。TEC\$INIT が定義されていない場合は、スタートアップ・コマンドは実行されません。

論理名 TEC\$INIT には、TECO コマンド文字列か、あるいはドル記号(\$)に続くファイル指定を定義できます。論理名 TEC\$INIT が TECO コマンド文字列であればそのまま実行され、ファイル指定であれば、そのファイルの内容が TECO コマンドとして実行されます。詳細は『Standard TECO Text Editor and Corrector for the VAX, PDP-11, PDP-10, and PDP-8』マニュアル(ドキュメンテーション CD-ROM に用意されています)を参照してください。

スタートアップ・コマンド・ファイルを実行しないようにするには、次の例に示すように、/NOCOMMAND 修飾子を使用します。

```
$ EDIT/TECO/NOCOMMAND MEMO.DAT
```

スタートアップ・コマンド・ファイルのファイル指定には、ワイルドカード文字は使用できません。

/CREATE (省略時の設定)
/NOCREATE

指定された入力ファイルを検出できなかった場合に、新しいファイルを作成します。/MEMORY 修飾子が指定され、かつ入力ファイルが指定されなかった時には、論理名 TEC\$MEMORY に指定されたファイルが作成されます。通常は、TECO は、指定されたディレクトリの中に要求されたファイル名が見つからないと、入力ファイル指定と一致する新しいファイルを作成します。TECO コマンド行に /NOCREATE 修飾子を指定し、しかも存在しないファイルの指定をタイプすると、TECO はエラー・メッセージを表示して、DCL コマンド・レベルに戻ります。/CREATE および /NOCREATE 修飾子は、/EXECUTE 修飾子と同時に指定することはできません。

/EXECUTE=コマンド・ファイル[引数]

TECO を起動し、コマンド・ファイルに記述されている TECO マクロを実行します。引数は、マクロの実行開始時に、テキスト・バッファに置かれます。空白文字

および特殊文字は引用符 (" ") で囲まなければなりません。TECO マクロについての詳細は、『Standard TECO Text Editor and Corrector for the VAX, PDP-11, PDP-10, and PDP-8』マニュアル(ドキュメンテーション CD-ROM に用意されています)を参照してください。

/EXECUTE 修飾子は、/CREATE および/MEMORY 修飾子と同時に指定することはできません。

/MEMORY (省略時の設定)
/NOMEMORY

EDIT/TECO コマンドに対するファイル指定が省略されたときに、TECO で最後に編集されたファイル (TEC\$MEMORY 論理名で示される) が編集対象ファイルとなるように指定します。

/OUTPUT=出力ファイル
/NOOUTPUT (省略時の設定)

編集セッション終了時にどのように出力ファイル名を指定するかを制御します。省略時の設定では、出力ファイルは入力ファイルと同じ名前ですが、バージョン番号は 1 つ大きくなります。入力ファイルと異なるファイル指定を出力ファイルに与えるときは、/OUTPUT 修飾子を使用してください。

次のコマンド行は、TECO を起動してファイル MEMO.DAT を編集し、OUTMEMO.DAT に書き出します。

```
$ EDIT/TECO/OUTPUT=OUTMEM.DAT MEMO.DAT
```

出力ファイル指定には、ディレクトリ指定も含めることができ、次の例に示すように出力を他のディレクトリに送ることができます。

```
$ EDIT/TECO/OUTPUT=[BARRRET.MAIL]MEMO.DAT MEMO.DAT
```

出力ファイルのファイル指定には、ワイルドカード文字は使用できません。

/READ_ONLY
/NOREAD_ONLY (省略時の設定)

出力ファイルを作成するかどうかを制御します。省略時の設定では、出力ファイルが作成されます。/READ_ONLY 修飾子を指定すると、出力ファイルは作成されません。

例

1. \$ EDIT/TECO/OUTPUT=NEWFILE.TXT OLDFILE.TXT

この EDIT コマンドは、TECO エディタを起動して、ファイル OLDFILE.TXT を編集します。TECO は TEC\$INIT 論理名定義を検索します。TEC\$INIT が定義されていないと、TECO は、コマンド・ファイルを使用せずに編集セッションを

開始します。セッションの終了時に、編集されたファイルに NEWFILE.TXT という名前が付けられます。

2. \$ EDIT/TECO/EXECUTE=FIND_DUPS "TEMP, ARGS, BLANK"

この例では、/EXECUTE 修飾子により、ファイル FIND_DUPS.TEC に記述されている TECO マクロが実行され、引数文字列 "TEMP, ARGS, BLANK" が、テキスト・バッファに置かれます。

EDIT/TPU

DEC テキスト処理ユーティリティ (DEC Text Processing Utility [DECTPU]) を起動します。省略時の設定では、EVE (拡張可能多機能エディタ [Extensible Versatile Editor]) エディタを起動します。DECTPU は、テキスト・エディタおよびその他のアプリケーション作成のための、構造化プログラミング言語およびその他の構成要素を提供します。

EVE を使用した編集についての詳細は、『OpenVMS ユーザーズ・マニュアル』またはオンライン・ヘルプを参照してください。

フォーマット

EDIT/TPU [入力ファイル]

ENABLE AUTOSTART

指定したキュー・マネージャが管理するすべての自動起動キューの、自動起動機能を 1 ノード上で許可します。省略時の設定では、このコマンドは/QUEUES 修飾子を使用します。

OPER (オペレータ) 特権が必要です。

自動起動キューについての詳細は、『OpenVMS システム管理者マニュアル』のバッチ・キューおよびプリント・キューに関する章を参照してください。

フォーマット

ENABLE AUTOSTART[/QUEUES]

パラメータ

なし

説明

キューの自動起動を許可すると、キュー・マネージャは、ノード上で停止しているすべての処理中の自動起動キューを自動的に起動させます。また、そのノードにフェイルオーバーしたすべての自動起動キューを自動的に起動させます。省略時の設定では、ENABLE AUTOSTART コマンドはコマンドを入力したノードに対して有効です。異なるノード上の自動起動を許可するには、/ON_NODE 修飾子を指定します。

省略時の設定では、このコマンドは省略時のキュー・マネージャ SYS\$QUEUE_MANAGER が管理する自動起動キューに対して有効です。異なるキュー・マネージャが管理する自動起動キューの自動起動を許可するには、/NAME_OF_MANAGER 修飾子を使用します。

INITIALIZE/QUEUE コマンドに/START 修飾子を指定して、または START /QUEUE コマンドで自動起動キューを起動し、STOP/QUEUE/NEXT または STOP /QUEUE/RESET コマンドで停止していない場合、自動起動キューは処理中です。

ノードをブートする時に、ENABLE AUTOSTART コマンドを入力するまで自動起動は禁止されています。たいていの場合は、利用者システム固有のスタートアップ・コマンド・プロシージャ、またはキューのスタートアップ・コマンド・プロシージャ

にこのコマンドを含めて、ノードをブートするたびに該当ノードの自動起動キューを起動させます。

修飾子

`/NAME_OF_MANAGER=名前`

許可したい自動起動キューを制御しているキュー・マネージャ名を指定します。この修飾子を使用すると、キューの集合に対して異なる自動起動機能を使用することができます。

`/NAME_OF_MANAGER` 修飾子を省略すると、省略時のキュー・マネージャ名 `SYS$QUEUE_MANAGER` が使用されます。

複数のキュー・マネージャについての詳細は、『OpenVMS システム管理者マニュアル』のキュー・マネージャに関する章を参照してください。

`/ON_NODE=ノード名`

OpenVMS Cluster システム内のノードを指定します。この修飾子を使用すると、このコマンドを入力したノード以外のノード上の、自動起動を許可することができます。

`/QUEUES`

キューの自動起動を許可することを指定します (この修飾子は省略時の設定により使用されます)。

例

```
1. $ INITIALIZE/QUEUE/BATCH/START-
   _$ /AUTOSTART_ON=SATURN:: BATCH_1
   $ ENABLE AUTOSTART/QUEUES
      .
      .
      .
   $ DISABLE AUTOSTART/QUEUES
```

この例では `INITIALIZE/QUEUE` コマンドは、ノード SATURN で実行できる自動起動キュー BATCH_1 を作成します。`/START` 修飾子は、キューの自動起動を有効にします。(ノード SATURN 上で実行される) `ENABLE/AUTOSTART/QUEUES` コマンドは該当ノードの自動起動を許可します。これによりキュー (および該当ノード上で有効な他の自動起動キュー) はジョブの実行を開始します。

(ノード SATURN 上で実行される) `DISABLE AUTOSTART` コマンドは該当ノード上の自動起動キューを停止し、他のキューが該当ノードにフェイルオーバーするのを抑止します。

/NAME_OF_MANAGER 修飾子が指定されていないので、これらのコマンドは省略時のキュー・マネージャ SYS\$QUEUE_MANAGER が管理するキューにのみ有効です。

BATCH_1 は 1 つのノードでのみ実行するよう設定されているので、キューは他のノードにフェイルオーバーできず停止します。ただしキューは自動起動が有効であり、ノード SATURN に対して ENABLE AUTOSTART コマンドが入力されると、このキューはスタートします。STOP/QUEUE/NEXT または STOP/QUEUE/RESET コマンドでキューの自動起動を無効にしない限り、BATCH_1 をリスタートさせるために START/QUEUE コマンドを実行する必要はありません。

```
2. $ INITIALIZE/QUEUE/BATCH/START-
   _$ /AUTOSTART_ON=(NEPTUN::,SATURN::) BATCH_1
   $ ENABLE AUTOSTART/QUEUES/ON_NODE=NEPTUN
   $ ENABLE AUTOSTART/QUEUES/ON_NODE=SATURN
   .
   .
   .
   $ STOP/QUEUES/ON_NODE=NEPTUN
```

この例の INITIALIZE/QUEUE コマンドは、自動起動キュー BATCH_1 を作成します。/START 修飾子は、キューの自動起動を有効にします。

最初の ENABLE AUTOSTART/QUEUES コマンドにより、ノード NEPTUN 上でキューの実行が開始されます。2 番目の ENABLE AUTOSTART/QUEUES コマンドはノード SATURN の自動起動を許可し、SATURN 上で停止しているすべての有効な自動起動キュー、および SATURN にフェイルオーバーしたすべての自動起動キューをスタートさせます。

この後、ノード NEPTUN を OpenVMS Cluster システムから削除する場合を考えてみます。STOP/QUEUES/ON_NODE コマンドはノード NEPTUN 上のすべてのキューを停止し、自動起動キュー BATCH_1 はノード SATURN にフェイルオーバーします。このキューは自動起動機能が有効であり、ノード SATURN では自動起動が許可されているので、BATCH_1 は SATURN 上で自動的にスタートします。

/NAME_OF_MANAGER 修飾子が指定されていないので、このコマンドは省略時のキュー・マネージャ SYS\$QUEUE_MANAGER が管理するキューに対してのみ有効です。

ENDSUBROUTINE

コマンド・プロシージャ内のサブルーチンの終了を示します。

ENDSUBROUTINE コマンドについての詳細は、CALL コマンドの説明またはオンライン・ヘルプを参照してください。

フォーマット

ENDSUBROUTINE

EOD

コマンドまたはプログラムが、会話型ターミナル以外の入力装置からデータを読み込む場合に、データ・ストリームの終わりを示します。

フォーマット

\$ EOD

説明

コマンド・プロシージャまたはバッチ・ジョブの EOD (end-of-deck) コマンドの機能は、次のとおりです。

- ドル記号(\$)で始まる入力データ行を終了させます。DECK コマンドは、ドル記号で始まる以降の行をコマンドでなくデータとして解釈することを示します。EOD コマンドは、データ行の最後を示します。
- コマンド・ストリームの中に、複数の入力ファイルが他のコマンドが介在しないで含まれている場合に、入力ファイルを終了させます。データを読み取るプログラムまたはコマンドは、EOD コマンドが読まれた時に、ファイルの終端(end-of-file) 条件を受け取ります。

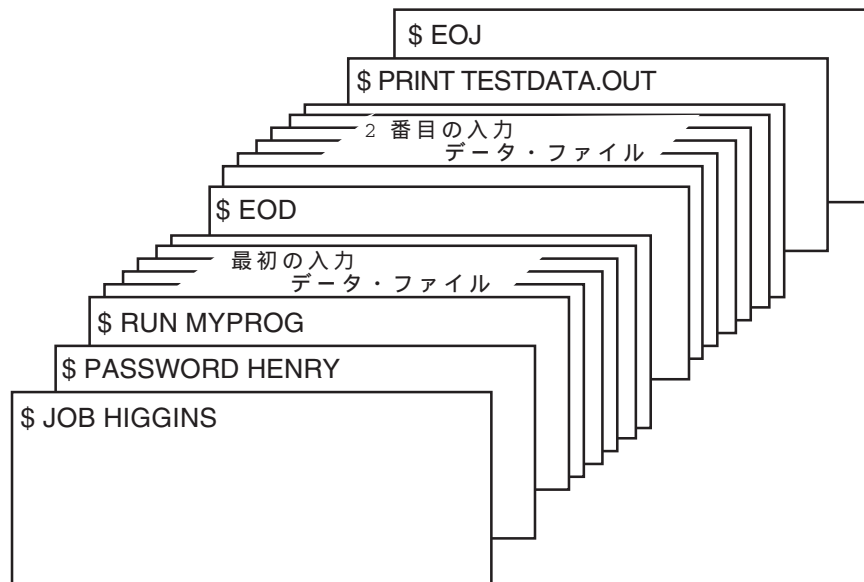
EOD コマンドの前には必ず、ドル記号を付けてください。ドル記号は、入力レコードの最初の文字位置(カラム 1) になければなりません。

例

```
1. $ CREATE WEATHER.COM
   $ DECK
   $ FORTRAN WEATHER
   $ LINK WEATHER
   $ RUN WEATHER
   $ EOD
   $ @WEATHER
```

このコマンド・プロシージャは、WEATHER.COM というコマンド・プロシージャを作成します。DECK コマンドと EOD コマンドによって囲まれている行が、WEATHER.COM というファイルに書き込まれます。そのあと、コマンド・プロシージャは WEATHER.COM を実行します。

2.



JRD-0785-GE

プログラム MUPROG には 2 つの入力ファイルが必要です。これらのファイルは論理装置 SYS\$INPUT から読み込まれます。EDO コマンドは 1 つ目のデータ・ファイルの終了と、2 つ目のデータ・ファイルの開始を示します。ドル記号で始まる次の行 (この例では PRINT コマンド) は、2 つ目のデータ・ファイルの終了を示します。

EOJ

カード・リーダからキューに登録される，バッチ・ジョブの終わりを示します。

フォーマット

\$ EOI

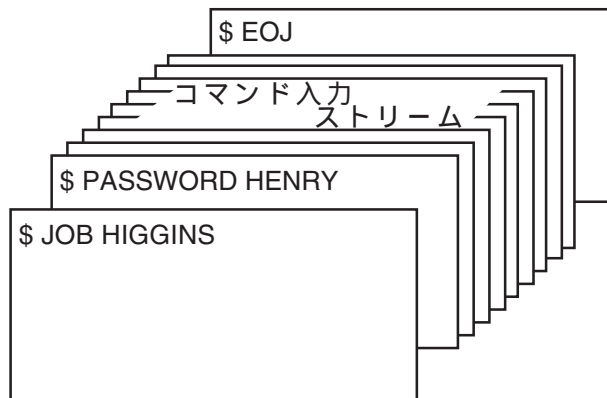
説明

EOJ (end-of-job) コマンドは，カード・リーダ経由で登録されたバッチ・ジョブの終わりを示します。EOJ カードは，必須ではありません。ただし，存在する場合，コマンド行の空白でない最初の文字は，ドル記号(\$)でなければなりません。EOJ コマンドを他の文脈で発行すると，プロセスをログアウトします。EOJ コマンドには，短縮形はありません。

EOJ カードは，EOF カードと等価です。

例

1.



JRD-0786-GE

JOB および PASSWORD コマンドは，カード・リーダ経由で登録されたバッチ・ジョブの開始を示します。EOJ コマンドは，そのジョブの終わりを示します。

EXAMINE

仮想メモリの内容を表示します。

内容を確認する仮想メモリの記憶位置へのユーザ・モード読み取り(R)アクセス権が必要です。

フォーマット

EXAMINE 記憶位置[: 記憶位置]

パラメータ

記憶位置[: 記憶位置]

内容を確認する仮想アドレスまたは仮想アドレスの範囲(最初に小さいアドレスを、2番目に大きいアドレスを指定)を指定します。アドレスの範囲を指定する場合には、開始アドレスと終了アドレスをコロン(:)で区切ります。

記憶位置には算術演算子、論理演算子、または事前に割り当てられたシンボルを含む任意の有効な計算式を指定できます。基数修飾子を使用して、アドレスを解釈するための基数を決めます。省略時の設定は16進数です。シンボル名は常に、それらが定義されたときの基数で解釈されます。記憶位置の前には、基数演算子%X、%D、または%Oを付けることができます。16進数は、数字(または%X)で開始する必要があります。

DEPOSIT および EXAMINE コマンドは、現在のメモリ記憶位置のポインタを保持します。EXAMINE コマンドは、EXAMINE コマンドを指定したときに表示した最後の記憶位置にこのポインタを設定します。EXAMINE または DEPOSIT コマンドにピリオド(.)を付けて実行すれば、この記憶位置を参照できます。

説明

EXAMINE コマンドは、仮想メモリの内容を表示します。アドレスは16進形式で表示され、内容は指定した基数で表示されます。次の例を参照してください。

```
address: contents
```

指定したアドレスにユーザ・モードでアクセスできない場合は、内容のフィールドに4つのアスタリスク(*)が表示されます。

基数修飾子: DEPOSIT または EXAMINE コマンドの基数の省略時の設定によって、コマンドによる数値リテラルの解釈方法が決まります。省略時の設定の基数は 16 進数値なので、コマンド行のすべての数値リテラルは、16 進数値であると仮定されます。EXAMINE コマンドに基数修飾子を指定した場合は、別の修飾子によって上書きされるまで、その基数が以降の EXAMINE コマンドおよび DEPOSIT コマンドの省略時の設定になります。次の例を参照してください。

```
$ EXAMINE/DECIMAL 900
00000384: 0554389621
```

EXAMINE コマンドは、記憶位置 900 を 10 進数として解釈し、その記憶位置の内容を 10 進数で表示します。以降のすべての DEPOSIT コマンドおよび EXAMINE コマンドでは、アドレスやデータに入力される数を 10 進数であると仮定します。EXAMINE コマンドは常に、アドレス記憶位置を 16 進数で表示する点に注意してください。

= (代入文) コマンドによって定義されたシンボルは常に、それらが定義されたときの基数で解釈されます。

表示する記憶位置または格納するデータとして入力する 16 進数値は、数字 (0 ~ 9) で始まらなければなりません。数字 (0 ~ 9) 以外で始めるとコマンド・インタプリタは、シンボル名を入力したか、またはシンボルの置換を行うものと仮定します。

現在の省略時の設定を変更するには、EXAMINE コマンドを入力する時に基数演算子 %X, %D, または %O を使用します。次の例を参照してください。

```
$ EXAMINE/DECIMAL %X900
00000900: 321446536
```

このコマンドは、16 進数値の 900 として指定された記憶位置にあるデータを 10 進数形式で表示することを要求しています。

長さ修飾子: EXAMINE コマンドの省略時の設定の長さの単位はロングワードです。EXAMINE コマンドは、各ロングワードの間にブランクを入れて、一度に 1 ロングワードずつデータを表示します。EXAMINE コマンドで長さ修飾子を使用した場合は、別の修飾子によって上書きされるまで、その長さが以降の EXAMINE コマンドおよび DEPOSIT コマンドでの、メモリ記憶位置の省略時の設定の長さになります。

修飾子の配置に関する制約: EXAMINE コマンドは、算術的に式を解釈します。そのため、コマンド名の直後に指定された場合のみ、修飾子は正しく解釈されます。

修飾子

/ASCII

指定した記憶位置のデータを ASCII 形式で表示します。

対応する ASCII コードがない 2 進値は、ピリオド(.)として表示されます。

/ASCII 修飾子を指定した場合、または ASCII モードが省略時の設定である場合は、コマンド行で指定される数値リテラルの省略時の設定の基数として、16 進数が使用されます。

/BYTE

指定された記憶位置にあるデータを一度に 1 バイトずつ表示します。

/DECIMAL

指定された記憶位置の内容を 10 進形式で表示します。

/HEXADECIMAL

指定された記憶位置の内容を 16 進形式で表示します。

/LONGWORD

指定された記憶位置にあるデータを一度に 1 ロングワードずつ表示します。

/OCTAL

指定された記憶位置の内容を 10 進形式で表示します。

/WORD

指定された記憶位置にあるデータを一度に 1 ワードずつ表示します。

例

```
1. $ RUN MYPROG
   Ctrl/Y
   $ EXAMINE 2678
   0002678: 1F4C5026
   $ CONTINUE
```

この例で RUN コマンドは、イメージ MYPROG.EXE の実行を開始します。MYPROG の実行中に Ctrl/Y を押して実行を中断します。EXAMINE コマンドによって、仮想メモリ記憶位置 2678(16 進値)の内容が表示されます。

```
2. $ BASE = %X1C00
   $ READBUF = BASE + %X50
   $ ENDBUF = BASE + %XA0
   $ RUN TEST
   Ctrl/Y
   $ EXAMINE/ASCII READBUF:ENDBUF
   00001C50: BEGINNING OF FILE MAPPED TO GLOBAL SECTION
   .
   .
   .
```

この例では、プログラム TEST.EXE を実行する前に、プログラムの基底アドレスならびにラベル READBUF と ENDBUF のシンボリック名が定義されます。これらのシンボリック名は、すべて基数演算子 %X を使用して 16 進形式で表されま

す。READBUF および ENDBUF は、プログラムの基底アドレスからのオフセットを定義しています。

プログラムの実行中に Ctrl/Y を押してプログラムを中断します。EXAMINE コマンドによって、指定されたメモリ記憶位置にあるすべてのデータが ASCII 形式で表示されます。

EXCHANGE

Exchange ユーティリティ (EXCHANGE) を起動します。このユーティリティは、オペレーティング・システムが通常認識できない形式で書き込まれた、大容量記憶ボリュームを操作します。

EXCHANGE を使用すると、次の操作を行うことができます。

- フォーリン・ボリュームの作成
- ファイルをボリュームに転送、ボリュームからファイルを転送
- ボリュームのディレクトリの一覧作成

RT-11ディスクのようなブロック・アドレス指定可能な装置の場合、EXCHANGE は、ファイル名の変更やファイルの削除などの操作も実行できます。EXCHANGE は、フォーリン・ボリュームのイメージであるFiles-11ファイルも操作できます。これらのファイルは仮想装置と呼ばれます。

EXCHANGE についての詳細は、『OpenVMS Exchange Utility Manual』（ドキュメンテーション CD-ROM に用意されています）、またはオンライン・ヘルプを参照してください。

フォーマット

EXCHANGE [サブコマンド] [ファイル指定] [ファイル指定]

EXCHANGE/NETWORK

このコマンドを使用すると、OpenVMS オペレーティング・システムと、OpenVMS のファイル編成をサポートしないオペレーティング・システムとの間でファイルを転送できるようになります。転送は、OpenVMS システムと OpenVMS 以外のオペレーティング・システムのノードを接続する DECnet ネットワーク通信リンク上で行われます。

DECnet サービスを使用すると、EXCHANGE/NETWORK コマンドで次の操作を行うことができます。

- OpenVMS ノードと OpenVMS 以外のシステムのノード間でファイルを転送する。
- 入力ファイルのグループを出力ファイルのグループとして転送する。
- OpenVMS 以外の 2 つのノードが、EXCHANGE/NETWORK コマンドを実行する OpenVMS ノードに DECnet で接続されている場合に、これらのノード間でファイルを転送する。

フォーマット

EXCHANGE/NETWORK 入力ファイル指定[,...] 出力ファイル指定

パラメータ

入力ファイル指定[,...]

転送したい既存のファイル名を指定します。ワイルドカード文字のアスタリスク(*)とパーセント記号(%)が使用できます。2 つ以上のファイルを指定する場合には、ファイル指定をコンマ(,)で区切ります。

出力ファイル指定

入力ファイルを転送する出力ファイル名を指定します。

出力ファイル指定には、フィールドを最低 1 つ指定しなければなりません。装置またはディレクトリを省略すると、現在の省略時の設定の装置およびディレクトリが使用されます。その他に不明なフィールド(ファイル名、ファイル・タイプ、およびバージョン番号)がある場合は、EXCHANGE/NETWORK コマンドによって、入力ファイル指定の対応するフィールドに置き換えられます。

EXCHANGE/NETWORK コマンドは、指定した各入力ファイルにつき新しい出力ファイルを 1 つ作成します。

ファイル名、ファイル・タイプ、またはバージョン番号の代わりに、ワイルドカード文字のアスタリスク(*)を使用できます。EXCHANGE/NETWORK コマンドは、関連する入力ファイルの対応するフィールドを使用して、出力ファイルをに名前を付けます。また、出力ファイル指定でワイルドカード文字のアスタリスク(*)を使用すると、EXCHANGE/NETWORK で 2 つ以上の出力ファイルを作成するよう指定することができます。次の例を参照してください。

```
$ EXCHANGE/NETWORK A.A,B.B MYPC::*.C
```

この EXCHANGE/NETWORK コマンドは、OpenVMS 以外のターゲット・ノード MYPC にファイル A.C および B.C を作成します。

ワイルドカード文字のアスタリスク(*)とパーセント記号(%), およびバージョン番号についての詳細は、次の「説明」を参照してください。

説明

EXCHANGE/NETWORK コマンドには、次の制限事項があります。

- ファイルが転送できるのは、ディスク装置間だけです。転送するファイルをディスク装置に永久的に格納しない場合は、このコマンドを実行する前にファイルをディスクに移動するか、またはコマンドが終了したあとディスクからファイルを取り出す必要があります。
- リモート・システムには、512 バイトのブロック・サイズが必要です。ここで 1 バイトは 8 ビット長です。
- ファイルを転送するノードは、DECnet Data Access Protocol (DAP) をサポートしていなければなりません。

OpenVMS レコード管理サービス (RMS) を使用すると、オペレーティング・システムは OpenVMS RMS ファイルのレコードにアクセスできます。どちらも OpenVMS ノードである 2 つのノード間で OpenVMS RMS ファイルを転送するには、他の DCL コマンド (COPY, APPEND, CONVERT など) から適切なものを 1 つ使用します。これらのコマンドは、RMS ファイル編成を認識して、ファイルの転送時に RMS レコード構造を保持するように設計されています。

OpenVMS ノードと OpenVMS 以外のノード間でファイルを転送するには、EXCHANGE/NETWORK コマンドを使用します。このコマンドを使用すると、ファイル編成の違いが原因で転送ができなかったり、予想しない結果になったりするのを防ぐことができます。COPY コマンドを使用した場合でも、コピーされたファイルの内容や属性は確実に保持されますが、EXCHANGE/NETWORK コマンドにはそれ以上の利点があります。EXCHANGE/NETWORK コマンドを使用すると、ファイル転送時にレコード属性を明示的に制御でき、数種類のオペレーティング・システムでファイルを使用可能にすることができます。

EXCHANGE/NETWORK コマンドは、同じ DECnet ネットワークに接続された OpenVMS ノードと OpenVMS 以外のノード間でファイルを転送します。OpenVMS 以外のシステムが OpenVMS ファイル編成をサポートしていない場合でも、EXCHANGE/NETWORK コマンドを使用すれば、転送時にファイルおよびレコードの各属性を変更したり廃棄したりできます。ただし、ターゲット・システムが OpenVMS ノードである場合は、この節で後述するように、ファイル定義言語 (FDL) ファイルを使用すれば、出力ファイルのファイル属性およびレコード属性を変更することができます。EXCHANGE/NETWORK コマンドには、多様な転送を正しく処理できるように、いくつかの省略時の設定が用意されています。ただし、転送の種類によっては、両方のノードのファイル形式またはレコード形式の必要条件を知っている必要もあります。

OpenVMS のファイル属性およびレコード属性

OpenVMS 環境のすべての RMS ファイルには、ファイル属性およびレコード属性を示す、ファイル属性およびレコード属性と呼ばれる情報が含まれています。ファイル属性には、ファイル編成、ファイルの保護、ファイル割り当て情報などの項目があります。レコード属性には、レコード形式、レコード・サイズ、索引編成ファイルのキー定義、キャリッジ制御情報などの項目があります。これらの属性が、OpenVMS RMS ファシリティが用いるデータ形式やアクセス方法を定義します。

OpenVMS ファイル編成をサポートしない OpenVMS 以外のオペレーティング・システムには、ファイルのファイル属性およびレコード属性を保存する機能がありません。ファイル属性およびレコード属性を保存または処理できない OpenVMS 以外のシステムに OpenVMS ファイルを転送すると、これらの情報の大部分が廃棄されてしまいます。これらの属性をファイルから削除すると、OpenVMS システムで再使用しなければならない場合に、そのファイルは元どおりの使い方はできません。

OpenVMS ノードへのファイルの転送

OpenVMS 以外のシステムから OpenVMS システムにファイルを転送すると、ファイルは通常、省略時の設定のファイル属性およびレコード属性を持つと仮定されます。ただし、ファイル定義言語 (FDL) ファイルを使用して、ファイルの属性を指定することもできます。また、CDA ドキュメントを転送する場合には、EXCHANGE/NETWORK コマンドの後に次のコマンドを入力します。

```
$ SET FILE/SEMANTICS=[ddif,dtif] ドキュメント名.doc
```

/FDL 修飾子で FDL ファイルを指定する場合には、FDL ファイルが出力ファイルの属性を決めます。OpenVMS 以外のシステムから OpenVMS システムにファイルを転送し、互換性のあるファイル属性およびレコード属性を設定する時、この機能は便利です。ただし、FDL ファイルを使用する場合、必要な属性を決めるのはユーザーです。

FDL ファイルについての詳細は、『OpenVMS Record Management Utilities Reference Manual』を参照してください。

OpenVMS 以外のノードへのファイルの転送

EXCHANGE/NETWORK コマンドは、OpenVMS ファイル編成をサポートしない OpenVMS 以外のシステムへの転送を行う場合に、OpenVMS ファイルに対応するファイル属性およびレコード属性を廃棄します。転送時のファイル属性およびレコード属性の消失によって、ほとんどのアプリケーションで使用できない出力ファイルが作成される可能性があります。

転送モードの選択

EXCHANGE/NETWORK コマンドには、AUTOMATIC、BLOCK、RECORD、および CONVERT の、4 つの転送モード・オプションがあります。ほとんどのファイル転送には、AUTOMATIC が使用できます。AUTOMATIC 転送モード・オプションを使用すると、ブロック入出力またはレコード入出力のいずれかを使用して、ファイルを転送できます。入出力の選択は、入力ファイルのファイル編成とオペレーティング・システムによります。

BLOCK 転送モード・オプションを選択すると、EXCHANGE/NETWORK コマンドは、ブロック入出力アクセス用に入力ファイルと出力ファイルの両方をオープンします。これによって、入力ファイルは、ブロック単位で出力ファイルに転送されます。実行可能なイメージを転送する場合には、この転送モードを使用します。また、このモードは、ファイル内容を正確に保持しなければならない場合に使用すると便利です。これは、別のシステムで一時的にファイルを保存する場合や、共同で動作するアプリケーションがシステムに存在する場合に必要となります。

RECORD 転送モード・オプションを選択すると、EXCHANGE/NETWORK コマンドは、レコード入出力アクセス用に入力ファイルと出力ファイルの両方をオープンします。これによって、入力ファイルは、逐次アクセスで出力ファイルに転送されます。この転送モードは主に、テキスト・ファイルを転送する場合に使用します。

CONVERT 転送モード・オプションを選択すると、EXCHANGE/NETWORK コマンドは、RECORD アクセス用に入力ファイルをオープンし、BLOCK アクセス用に出力ファイルをオープンします。これによって、レコードは入力ファイルから読み込まれ、ブロックにパックされてから、出力ファイルに書き込まれます。この転送モードは主に、暗黙のキャリッジ制御なしでファイルを転送する場合に使用します。たとえば、DIGITAL Standard Runoff (DSR) で作成したファイルを DECnet DOS システムに転送するには、CONVERT 転送モード・オプションを使用しなければなりません。結果の出力ファイルを OpenVMS ノードに返送するには、AUTOMATIC 転送モード・オプションを使用します。

ワイルドカード文字

ファイル指定には、ワイルドカード文字のアスタリスク(*)とパーセント記号(%)を使用できます。また、その動作内容は、OpenVMS ノードに対して他の OpenVMS システム・コマンドを使用した場合と同じです。

2 つ以上の入力ファイルが指定されて、出力ファイル指定にワイルドカード文字のアスタリスク(*)やパーセント記号(%)が指定されていない場合には、最初の入力ファイルが出力ファイルにコピーされます。後続の各入力ファイルは転送された後、同じフ

ファイル名でバージョン番号が大きい出力ファイル名になります。複数の入力ファイルが、1つの出力ファイルに連結されるわけではありません。また、バージョン番号をサポートしないフォーリン・システムにファイルを転送する場合には、最後に指定した入力ファイルに対応する出力ファイルが1つだけ作成されます。

複数の出力ファイルを作成するには、複数の入力ファイルを指定します。次のいずれかの方法を使用してください。

- 出力ファイル名、ファイル・タイプ、またはバージョン番号の各フィールドでワイルドカード文字のアスタリスク(*)を1つ使用します。
- 出力ファイル指定として、ノード名、装置名、またはディレクトリ名だけを使用します。

複数の出力ファイルを作成する場合、EXCHANGE/NETWORK コマンドは、対応する入力ファイルのフィールドを使用して、出力ファイルを指定します。

複数の入力ファイルと出力ファイルを指定する場合には、/LOG 修飾子を使用すると、意図したとおりにファイルがコピーされていることを確かめることができます。

バージョン番号

ターゲット・ノードのファイル形式でバージョン番号が使用できる場合には、次のガイドラインを参照してください。

入力ファイル、および出力ファイルのバージョン番号が指定されていない場合、EXCHANGE/NETWORK コマンドは、出力ファイルに次のいずれかのバージョン番号を割り当てます(省略時の設定)。

- 入力ファイルのバージョン番号
- 同じファイル名およびファイル・タイプを持つ既存のファイルの、最高のバージョン番号より1つ大きいバージョン番号

出力ファイルのバージョン番号をワイルドカード文字のアスタリスク(*)で指定した場合、EXCHANGE/NETWORK コマンドは、対応する入力ファイルのバージョン番号を、出力ファイルのバージョン番号として使用します。

出力ファイルのバージョン番号を明示的に指定した場合、EXCHANGE/NETWORK コマンドは通常、出力ファイル指定にそのバージョン番号を使用します。ただし、同じかまたはそれ以上のバージョンの出力ファイルが存在している場合には、警告メッセージが発行されずにファイルがコピーされて、バージョン番号は、現在ある最高のバージョン番号より1つ大きい値に設定されます。

ファイルの保護および作成日/更新日

出力ファイル名の一部が明示的に指定されている場合、EXCHANGE/NETWORK コマンドは、出力ファイルを新しいファイルとして処理します。出力ノードが OpenVMS システムである場合、新しいファイルの作成日は、現在の日時に設定されます。ただし、出力ファイル指定が、ワイルドカード文字のアスタリスク(*)とパー

セント記号(%)のみで構成されている場合、出力ファイルは新しいファイルとして処理されないため、出力ファイルの作成日は入力ファイルの作成日と同じになります。

出力ファイルの更新日は常に、現在の日時に設定されます。バックアップの日付はゼロに設定されます。出力ファイルには、新しい満了日が割り当てられます。保持が許可されている場合は、満了日はファイル・システムによって設定されます。保持が禁止されている場合、満了日はゼロに設定されます。

ターゲット・ノードが OpenVMS ノードである場合には、出力ファイルの保護およびアクセス制御リスト (ACL) は、次のパラメータによって以下の順序で決まります。

1. 出力ファイルの既存バージョンの保護
2. 出力ディレクトリの省略時の設定の保護および ACL
3. プロセスでの省略時の設定のファイルの保護

ACL の概要については、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

OpenVMS システムでは、出力ファイルの所有者は通常、出力ファイルの作成者と同じです。ただし、拡張された特権を持つユーザが出力ファイルを作成する場合、その所有者は親ディレクトリの所有者か、または出力ファイルの前バージョンが存在する場合にはその所有者になります。

拡張特権には、次のものがあります。

- SYSPRV(システム特権)または BYPASS
- システム利用者識別コード (UIC)
- GRPPRV (グループ特権)。親ディレクトリ(または出力ファイルの前バージョン)の所有者が、新しい出力ファイルの作成者と同じグループに属する場合。
- 親ディレクトリ(または出力ファイルの前バージョン)の所有者を表す識別子 (リソース属性を持つ)

修飾子

/BACKUP

/BEFORE または /SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、最新のバックアップの日時をもとにファイルを選択します。この修飾子とは他の時刻属性を指定する修飾子、/CREATED、/EXPIRED、および /MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として /CREATED 修飾子が使用されます。

/BEFORE[=時刻]

指定された時刻以前の時刻属性をもつファイルを選択します。絶対時刻、または絶対時刻とデルタ時間の組み合わせを指定します。また、BOOT、LOGIN、TODAY(省略時の設定)、TOMORROW、およびYESTERDAYというキーワードも指定できます。適用する時刻属性は、/BACKUP、/CREATED(省略時の設定)、/EXPIRED、または/MODIFIED 修飾子のいずれかで指定します。

時刻指定の詳細は、『OpenVMS ユーザーズ・マニュアル』、またはオンライン・ヘルプのトピック Date を参照してください。

/BY_OWNER[=uic]

指定した所有者 UIC と一致する所有者 (UIC) を持つファイルだけを選択します。省略時の UIC (利用者識別コード) は、現在処理中のプロセスの UIC です。

『OpenVMS システム・セキュリティ・ガイド』の説明に従って、標準形式を使用して UIC を指定します。

/CONFIRM

/NOCONFIRM (省略時の設定)

各ファイルを転送する前に、ファイルの転送について確認するよう要求するかどうかを制御します。次の応答が有効です。

YES	NO	QUIT
TRUE	FALSE	Ctrl/Z
1	0	ALL

Return

応答には、大文字と小文字の任意の組み合わせが使用できます。一意に認識できれば単語での応答を 1 つまたは複数の文字に短縮しても構いません。たとえば、TRUE の場合は T、TR、または TRU に短縮できます。肯定の応答は、YES、TRUE、または 1 です。否定の応答は、NO、FALSE、0 を押して、Return を押します。QUIT を入力するか、または Ctrl/Z を押すと、その時点でコマンドの処理を終了することを意味します。ALL と応答すると、コマンドは処理を続けますが、プロンプトは再表示されません。上記のリストにない応答を入力すると、DCL はエラー・メッセージを発行してプロンプトを再表示します。

/CREATED (省略時の設定)

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、作成日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/EXPIRED、および/MODIFIED 修飾子とは同時に指定できません。これら 4 つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/EXCLUDE=(ファイル指定[,...])

指定したファイルをファイル転送の操作から除外します。ファイル指定にディレクトリは指定できませんが、装置は指定できません。ファイル指定には、ワイルドカード文字のアスタリスク(*)とパーセント記号(%)を使用できます。ただし、特定のバージョ

ンを除外する場合には、相対的なバージョン番号は使用できません。1 ファイルだけを指定する場合は、括弧を省略できます。

/EXPIRED

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、満了日時をもとにファイルを選択します(満了日は、SET FILE /EXPIRATION_DATE コマンドで設定します)。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/CREATED、および/MODIFIED 修飾子とは同時に指定できません。これら4つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/FDL=FDL ファイル指定

ファイル定義言語(FDL) ファイルの記述を用いて、出力ファイルの属性を指定します。出力ファイルに特別な属性を必要とする場合には、この修飾子を使用します。FDL ファイルについての詳細は、『OpenVMS Record Management Utilities Reference Manual』を参照してください。

/FDL 修飾子を使用すると、転送モードを暗黙のうちにブロック単位にします。ただし、/TRANSFER_MODE 修飾子で指定した転送モードが優先します。

/LOG

/NOLOG (省略時の設定)

EXCHANGE/NETWORK コマンドが、コピーされる各ファイルのファイル指定を表示するかどうかを制御します。

/LOG 修飾子を使用すると、EXCHANGE/NETWORK コマンドは各コピー操作について次の項目を表示します。

- 入力ファイルと出力ファイルのファイル指定
- コピーされるブロック数またはレコード数(ファイルがブロック単位または逐次アクセスのいずれでコピーされるかにより異なる)

/MODIFIED

/BEFORE または/SINCE 修飾子を適用する時刻属性を指定します。この修飾子を指定すると、最新の変更日時をもとにファイルを選択します。この修飾子は他の時刻属性を指定する修飾子、/BACKUP、/CREATED、および/EXPIRED 修飾子とは同時に指定できません。これら4つの修飾子のいずれも指定しない場合には、省略時の設定として/CREATED 修飾子が使用されます。

/SINCE[=時刻]

指定された時刻以降の時刻属性をもつファイルを選択します。絶対時刻、または絶対時刻とデルタ時間の組み合わせを指定します。また、BOOT、LOGIN、TODAY(省略時の設定)、TOMORROW、およびYESTERDAYというキーワードも指定できます。適用する時刻属性は、/BACKUP、/CREATED(省略時の設定)、/EXPIRED、または/MODIFIED 修飾子のいずれかで指定します。

時刻指定の詳細は、『OpenVMS ユーザーズ・マニュアル』，またはオンライン・ヘルプのトピック Date を参照してください。

/STYLE=キーワード

表示するファイル名の書式を指定します。

この修飾子のキーワードは CONDENSED および EXPANDED です。意味は次の表のとおりです。

キーワード	説明
CONDENSED (省略時の設定)	ファイル名を 255 文字長の文字列に適合するように表示します。このファイル名の場合，ファイル指定に DID あるいは FID 短縮形を含むことが可能です。
EXPANDED	ファイル名をディスクに格納されているとおりに表示します。このファイル名の場合，ファイル指定に DID あるいは FID 短縮形は含みません。

キーワード CONDENSED と EXPANDED を同時に指定することはできません。この修飾子は，確認が要求された場合に，出力メッセージに表示されるファイル名の書式を指定します。

EXPANDED キーワードが指定されていない場合，ファイル・エラーは CONDENSED ファイル指定で表示されます。

詳細は『OpenVMS システム管理者マニュアル(上巻)』を参照してください。

/TRANSFER_MODE=オプション

転送に使用する入出力方式を指定します。この修飾子は，すべてのファイル形式で使用できます。次のオプションのいずれかを指定できます。

オプション	機能
AUTOMATIC	EXCHANGE/NETWORK コマンドが，適切な転送モードを選択します。これが省略時の設定の転送モードです。
BLOCK	ブロック入出力用に入力ファイルと出力ファイルの両方をオープンし，ファイルをブロック単位で転送します。
CONVERT[=オプション[,...]]	入力ファイルからレコードを読み込み，ブロックにバックして，ブロック・モードで出力ファイルに書き込みます。次の表にリストしたオプションによって，転送時にどのような付加情報が挿入されるかが決まります。
RECORD	レコード入出力用に入力ファイルと出力ファイルをオープンして，ファイルを逐次アクセスで転送します。ターゲット・システムは，レコード操作をサポートしていなければなりません。また，入力ファイルは，レコード入出力アクセスでオープンしなければなりません。

CONVERT 転送モードでは、次の 4 つのオプションを使用して、レコードへの特別な文字の挿入を制御できます。

オプション	機能
CARRIAGE_CONTROL	入力ファイルのキャリッジ制御情報が解釈され、実際の文字に展開されて各レコードに書き込まれます。
COUNTED	各レコードのバイト長がレコードの先頭に書き込まれます。この長さには、各レコードの FIXED_CONTROL、CARRIAGE_CONTROL、および RECORD_SEPARATOR の情報のすべてが含まれます。
FIXED_CONTROL	固定長制御部レコード (VFC) 情報を持つすべての可変長が、データの一部として出力ファイルに書き込まれます。COUNTED オプションが指定されている場合には、この情報の前にレコード長情報が書き込まれます。
RECORD_SEPARATOR= 区切り文字	<p>各レコードに 1 バイトまたは 2 バイトのレコード区切り文字が挿入されます。レコード区切り文字は、レコード内の最後の文字です。区切り文字には、次の 3 つのいずれかを指定できます。</p> <ul style="list-style-type: none"> • CR: キャリッジ・リターンだけを指定します。 • LF: 行送りだけを指定します。 • CRLF: キャリッジ・リターンと行送りを指定します。

例

1. \$ EXCHANGE/NETWORK VMS_FILE.DAT KUDOS::FOREIGN_SYS.DAT

この例で EXCHANGE/NETWORK コマンドは、現在の省略時の設定の装置およびディレクトリにある VMS_FILE.DAT ファイルを、OpenVMS 以外のノード KUDOS 上の FOREIGN_SYS.DAT ファイルに転送します。/TRANSFER_MODE 修飾子が明示的に指定されなかったため、EXCHANGE/NETWORK コマンドは自動的に、転送モードをブロック入出力またはレコード入出力のいずれかに決めます。

2. \$EXCHANGE/NETWORK/TRANSFER_MODE=BLOCK-
\$:KUDOS:FOREIGN_SYS.DAT VMS_FILE.DAT

この例で EXCHANGE/NETWORK コマンドは、OpenVMS 以外のノード KUDOS にある FOREIGN_SYS.DAT ファイルを、現在の省略時の設定の装置およびディレクトリにある VMS_FILE.DAT ファイルに転送します。転送モードとして、ブロック入出力を指定します。

3. \$ EXCHANGE/NETWORK/FDL=VMS_FILE_DEFINITION.FDL-
\$:KUDOS:REMOTE_FILE.TXT VMS_FILE.DAT

この例で EXCHANGE/NETWORK コマンドは、ノード KUDOS の REMOTE_FILE.TXT ファイルを VMS_FILE.DAT ファイルに転送します。出力ファイル VMS_FILE.DAT のファイル属性は、ファイル定義言語 (FDL) ソース・ファイルの VMS_FILE_DEFINITION.FDL から取得されます。/FDL 修飾子が指定され/TRANSFER_MODE 修飾子が省略されているので、転送モードには、省略時の設定であるブロック入出力を使用します。

FDL ファイルの作成についての詳細は、『OpenVMS Record Management Utilities Reference Manual』を参照してください。

4. \$ EXCHANGE/NETWORK -
\$ /TRANSFER_MODE=CONVERT=(CARRIAGE_CONTROL,COUNTED, -
\$ RECORD_SEPARATOR=CRLF, FIXED_CONTROL) -
\$ PRINT_FILE.TXT KUDOS::*

この例で EXCHANGE/NETWORK コマンドは、現在の省略時の設定の装置およびディレクトリにある PRINT_FILE.TXT ファイルを、OpenVMS 以外のノード KUDOS の PRINT_FILE.TXT ファイルに転送します。/TRANSFER_MODE 修飾子に CONVERT オプションが指定されているため、入力ファイルは逐次アクセスで読み込まれ、その後の CONVERT オプションで指定されたように変更されて、ブロック単位で出力ファイルに書き込まれます。出力ブロックとして、収容できるだけの数のレコードがパックされます。

CONVERT オプション CARRIAGE_CONTROL が指定されているので、キャリッジ制御情報は ASCII 文字へ変換されます。また、接頭辞制御と接尾辞制御の使用状況に応じて、キャリッジ制御情報をレコードの前に挿入するか、レコードの後に追加するかを指定します。

CONVERT オプション FIXED_CONTROL は、任意の固定長制御部情報を、ASCII 文字に変換してレコードの先頭に挿入するように指定します。

CONVERT オプション RECORD_SEPARATOR=CRLF は、指定した 2 文字のキャリッジ・リターンおよび行送りをレコードの終端に追加します。

CONVERT オプション COUNTED は、レコードの全体の長さ (前の CONVERT オプションのすべての実行結果が追加された後の長さ) をカウントしたり、結果をレコードの先頭の 2 バイトに挿入したりすることを指定します。

EXIT

コマンド・プロシージャ，またはサブルーチンの処理を終了し，呼び出したコマンド・レベル(コマンド・プロシージャまたは会話型 DCL) に制御を戻します。また，ユーザが Ctrl/Y を入力した後，イメージを正常終了させます。

フォーマット

EXIT [状態コード]

パラメータ

状態コード

すでに用意されているグローバル・シンボル \$STATUS に与える数値を定義します。状態コードには整数，または整数値が求められる式を指定できます。この値は，外側のコマンド・レベルからテストできます。ロングワード整数値の下位 3 ビットは，予約されているグローバル・シンボル \$SEVERITY の値を変更します。

状態コードを指定する場合，DCL はコンディション・コードとして解釈します。値が偶数の場合には，警告メッセージ，エラー・メッセージおよび重大なエラー・メッセージが作成されます。値が奇数の場合には，正常終了メッセージまたは情報メッセージが作成されます。

状態コードを指定しない場合には，\$STATUS の現在の値が保存されます。制御が外側のコマンド・レベルに渡される場合には，\$STATUS には最後に実行されたコマンドまたはプログラムの状態が設定されます。

説明

EXIT および STOP コマンドは，ともにプロシージャの実行を終了させる方法を提供します。EXIT コマンドは，現在のコマンド・プロシージャの実行を終了させ，呼び出し側のコマンド・レベルに制御を戻します。EXIT コマンドをコマンド・レベル 0 で非会話型プロセス (バッチ・ジョブなど) から入力すると，プロセスは終了します。

STOP コマンドは，現在のコマンド・レベルにかかわらず，制御をコマンド・レベル 0 に戻します。STOP コマンドをコマンド・プロシージャまたは非会話型プロセス (バッチ・ジョブなど) から実行すると，プロセスは終了します。

DCL コマンド、ユーザ・プログラム、またはコマンド・プロシージャが終了する場合と、コマンド・インタプリタは条件コード値をグローバル・シンボル\$STATUS に保存します。EXIT コマンドで明示的に\$STATUS の値を設定しないと、コマンド・インタプリタは\$STATUS の現在の値を使用してエラー状態を決定します。

\$STATUS に含まれる状態値の下位 3 ビットが、条件の重大度を表します。予約されているグローバル・シンボル\$SEVERITY には、条件コードのこの部分が入ります。重大度の値の範囲は、次のように 0 ~ 4 です。

値	重大度
0	警告
1	正常終了
2	エラー
3	情報
4	重大な (致命的な) エラー

正常終了および情報のコードは奇数の値、警告およびエラーのコードは偶数の値であることに注意してください。

コマンド・プロシージャが終了し、制御が別のレベルに戻る場合には、コマンド・インタプリタが\$STATUS の現在の値をテストします。\$STATUS の値が偶数で、その高位ビットが 0 の場合は、コマンド・インタプリタは、(存在する場合は) その状態コードに対応するシステム・メッセージを表示します。メッセージが存在しない場合は、NOMSG メッセージが表示されます。高位ビットが 1 の場合、メッセージは表示されません。

DCL コマンドですでに表示された警告またはエラー状態の後でコマンド・プロシージャが終了すると、コマンド・インタプリタは、\$STATUS の高位ビットを 1 にセットします。残りのビットは変更されません。これによって、エラーの原因になったコマンドも、コマンド・プロシージャも、エラー・メッセージを表示しなくなります。

Ctrl/Y でイメージを中断してから EXIT コマンドを使用すると、現在実行中のイメージが正常終了します。イメージに終了処理ルーチンが宣言されていた場合、終了処理ルーチンに制御が移ります。これに対し、STOP コマンドは、終了処理ルーチンを実行しません。このため、通常は STOP コマンドではなく EXIT コマンドを使用してください。

例

1. \$ EXIT 1

\$STATUS と\$SEVERITY に 1 を設定して、1 つ外側のプロシージャ・レベルに抜けます。

EXIT

2.

```
$ ON WARNING THEN EXIT
$ FORTRAN 'P1'
$ LINK 'P1'
$ RUN 'P1'
```

EXIT コマンドは、ON コマンドの条件成立時のコマンドとして使用されています。このステートメントは、警告またはエラーがプロシージャ内のコマンドから出されたときに、コマンド・プロシージャが必ず終了するようにします。

プロシージャは終了する前に、終了する原因となったコマンドまたはプログラムの状態コードを設定します。

3.

```
$ START:
$      IF (P1 .EQS. "TAPE") .OR. (P1 .EQS. "DISK") THEN GOTO 'P1'
$      INQUIRE P1 "Enter device (TAPE or DISK)"
$      GOTO START
$ TAPE: ! Process tape files
.
.
.
$      EXIT
$ DISK: ! Process disk files
.
.
.
$      EXIT
```

この例では、コマンド・プロシージャ内で、異なったコマンド・パスを終了させる方法を示しています。このコマンド・プロシージャの実行時には、パラメータに TAPE または DISK を指定します。IF コマンドは論理 OR を使用して、これらの文字列が入力されたかテストします。結果が真であれば GO TO コマンドは対応するラベルに分岐します。P1 が TAPE でも DISK でもない場合は、正しいパラメータを入力するよう INQUIRE コマンドが要求します。

ラベル TAPE および DISK に続くコマンドは、パスが異なっています。ラベル DISK の前の EXIT コマンドは、プロシージャが明示的に DISK に分岐した時のみ、ラベル DISK 以降のコマンドが実行されることを保証しています。

プロシージャの終わりでは、自動適に EXIT コマンドが実行されるので、最後の EXIT コマンドは必要ありません。しかし、このように明示的に使用することをおすすめします。

4.

```
$ IF P1 .EQS. "" THEN -
    INQUIRE P1 "Enter filespec (null to exit)"
$ IF P1 .EQS. "" THEN EXIT
$ PRINT 'P1'/AFTER=20:00/COPIES=50/FORMS=6
```

このコマンド・プロシージャは、パラメータがそのプロシージャに渡されたかどうかをテストします。パラメータが渡されていない場合には、プロシージャは必要なパラメータを要求するプロンプトを表示します。そのあと、パラメータ P1 を再びテストします。データを含まない行を指定するために、キャリッジ・リター

ンを入力し、パラメータとして空文字列が与えられた場合には、プロセスは終了します。それ以外の場合には、入力パラメータとして P1 の現在の値を使用して、PRINT コマンドを実行します。

5.

```
$ IF P1 .EQS. "" THEN INQUIRE P1 "Code"
$ CODE = %X'P1'
$ EXIT CODE
```

この例は、システム・ステータス・コードからシステム・メッセージを得る 1 つの方法を示しています。パラメータが 1 つ必要で、入力しないと入力するよう要求します。基数演算子%X を付けてシンボル CODE に代入しています。そして、EXIT コマンドを実行します。たとえば次のように使用します。

```
$ @E 1C
%SYSTEM-F-EXQUOTA, exceeded quota
```

プロセス終了時には、\$STATUS には%X1C が代入され、この値は EXQUOTA メッセージに評価されます。または、レキシカル関数 F\$MESSAGE を使用しても同様のことができます。

6.

```
$ RUN MYPROG
Ctrl/Y
$ EXIT
```

この RUN コマンドは、MYPROG.EXE というイメージの実行を開始します。そのあと、CTRL/Y を使用して実行に割り込みがかけられています。EXIT コマンドは MYPROG.EXE を終了する前に、イメージによって宣言された終了ハンドラ・ルーチンを呼び出します。

FONT

Alpha システムの場合は、ASCII ビットマップ分散フォーマット (BDF) をバイナリ・ポータブル・コンパイル・フォーマット (PCF) に変換します。また VAX システムの場合は、ASCII ビットマップ分散フォーマット (BDF) をバイナリ・サーバ・ナチュラル・フォーマット (SNF) に変換します。DECwindows サーバは、PCF または SNF ファイルを使用してフォントを表示します。BDF ファイルをバイナリ・フォームに変換する際に、フォント・コンパイラは、フォントとコンパイル・プロセスに関する統計情報を提供します。

フォント・コンパイラについての詳細は、OpenVMS DECwindows のプログラミングに関するマニュアルまたはオンライン・ヘルプを参照してください。

フォーマット

FONT ファイル指定

GOSUB

コマンド・プロシージャ・レベルを変更せずに、制御をコマンド・プロシージャ内のラベルが付けられたサブルーチンに渡します。

フォーマット

GOSUB ラベル

パラメータ

ラベル

コマンド行の最初の項目として、1文字から255文字までの英数字のラベルを指定します。ラベルの中にブランクを含むことはできません。GOSUB コマンドの実行後、制御は指定されたラベルのあとのコマンドに渡されます。

ラベルは、現在のコマンド・プロシージャの中で、GOSUB ステートメントの前でもあとでもかまいません。コマンド・プロシージャ内でラベルを使用する場合には、最後にコロンを指定しなければなりません。ラベルが重複している場合、最も最近読まれたラベルへ飛びます。

説明

ラベルで指定されたサブルーチンへ制御を移すには、コマンド・プロシージャで GOSUB コマンドを使用します。コマンド・ストリームをランダム・アクセス装置（つまりディスク装置）から読み取っていないと、GOSUB コマンドは動作しません。

RETURN コマンドは、GOSUB サブルーチン・プロシージャを終了させ、GOSUB 文の呼び出しの次のコマンドへ制御を移します。RETURN コマンドは、省略可能な状態値を受け付けます。

GOSUB コマンドでは、新しいプロシージャ・レベルは生成されません。したがって、これを“ローカルの”サブルーチン呼び出しと呼びます。現在のコマンド・プロシージャ・レベルで定義されたラベルとローカル・シンボルは、GOSUB コマンドで呼び出されたサブルーチンで使用できます。GOSUB コマンドは、プロシージャ・レベル当り、最高 16 レベルまでネストできます。

コマンド・インタプリタは、ラベルを検出すると、ラベル・テーブルにラベルを入れます。このテーブルは、ローカル・シンボル・テーブルで利用できる領域から割り当てられます。コマンド・インタプリタがすでにテーブルに存在しているラベルを検出

すると、既存の定義が新しい定義で置き換えられます。したがって、重複ラベルを使用すると、制御は常に DCL が最後に読み取ったラベルに移ります。次の規則が適用されます。

- GOSUB コマンドの前と後に重複ラベルがある場合、制御はコマンドの前にあるラベルに移ります。
- すべての重複ラベルが GOSUB コマンドより前にある場合、制御は、最新のラベル、つまり GOSUB コマンドに最も近いラベルに移ります。
- すべての重複ラベルが GOSUB コマンドより後にある場合、制御は、GOSUB コマンドに最も近いラベルに移ります。

現在のコマンド・プロシージャにラベルが存在しない場合、プロシージャは続行できないため、強制終了します。

ラベルに使用できる領域のサイズには、制限があることに注意してください。コマンド・プロシージャが多くのシンボルを使用し、多くのラベルがある場合、コマンド・インタプリタのテーブル領域が不足し、エラー・メッセージが出ることがあります。

例

```
1. $!
   $! GOSUB.COM
   $!
   $ SHOW TIME
   $ GOSUB TEST1
   $ WRITE SYS$OUTPUT "success completion"
   $ EXIT
   $!
   $! TEST1 GOSUB definition
   $!
   $ TEST1:
   $     WRITE SYS$OUTPUT "This is GOSUB level 1."
   $     GOSUB TEST2
   $     RETURN %X1
   $!
   $! TEST2 GOSUB definition
   $!
   $ TEST2:
   $     WRITE SYS$OUTPUT "This is GOSUB level 2."
   $     GOSUB TEST3
   $     RETURN
   $!
   $! TEST3 GOSUB definition
   $!
   $ TEST3:
   $     WRITE SYS$OUTPUT "This is GOSUB level 3."
   $     RETURN
```

このコマンド・プロシージャは、ラベルを付けられたサブルーチンへ制御を移すための GOSUB コマンドの使い方を示します。GOSUB コマンドは TEST1 にラベルを付けたサブルーチンに制御を移動します。プロシージャは、サブルーチン TEST1 でコマンドを実行し、サブルーチン TEST2 へ分岐します。その後、プロシージャは、サブルーチン TEST2 でコマンドを実行し、サブルーチン TEST3 へ分岐します。それぞれのサブルーチンは RETURN コマンドによって終了します。TEST3 が実行された後、RETURN コマンドはそれぞれの呼出し GOSUB ステートメントに後続するコマンドラインへ制御を返します。この時点で、プロシージャはうまく実行されています。

GOTO

制御をコマンド・プロシージャ内のラベルが付けられたステートメントに渡します。

フォーマット

GOTO ラベル

パラメータ

ラベル

コマンド行の最初の項目として、1文字から255文字までの英数字のラベルを指定します。ラベルの中に空白を含むことはできません。GOTO コマンドの実行後、制御は指定されたラベルのあとのコマンドに渡されます。

ラベルは、現在のコマンド・プロシージャの中で、GOTO ステートメントの前でもあとでもかまいません。コマンド・プロシージャ内でラベルを使用する場合には、最後にコロンを指定しなければなりません。ラベルが重複している場合、最も最近読まれたラベルへ飛びます。

説明

プロシージャ内の次の行ではない行に制御を移すには、コマンド・プロシージャで GOTO コマンドを使用します。ラベルは、現在のコマンド・プロシージャの GOTO 文の前でも後でも使用できます。コマンド・ストリームがランダム・アクセス装置（つまりディスク装置）から読み取られていない場合、GOTO コマンドは動作しません。

GOTO コマンドのターゲット・ラベルが別の IF-THEN-ELSE 構造内にある場合は、エラー・メッセージ (DCL-W-USGOTO) が返されます。

コマンド・インタプリタは、ラベルを検出すると、ラベル・テーブルにラベルを入れます。このテーブルは、ローカル・シンボル・テーブルで利用できる領域から割り当てられます。コマンド・インタプリタがすでにテーブルに存在しているラベルを検出すると、既存の定義が新しい定義で置き換えられます。したがって、重複ラベルを使用すると、制御は常に DCL が最後に読み取ったラベルに移ります。次の規則が適用されます。

- GOTO コマンドの前と後に重複ラベルがある場合、制御はコマンドの前にあるラベルに移ります。

- すべての重複ラベルが GOTO コマンドより前にある場合、制御は、最新のラベル、つまり GOTO コマンドに最も近いラベルに移ります。
- すべての重複ラベルが GOTO コマンドより後にある場合、制御は、GOTO コマンドに最も近いラベルに移ります。

現在のコマンド・プロシージャにラベルが存在しない場合、プロシージャは続行できないので、強制終了します。

ラベルに使用できる領域のサイズには、制限があることに注意してください。コマンド・プロシージャが多くのシンボルを使用し、多くのラベルがある場合、コマンド・インタプリタのテーブル領域が不足し、エラー・メッセージが出ることがあります。

例

```
1. $ IF P1 .EQS. "HELP" THEN GOTO TELL
   $ IF P1 .EQS. "" THEN GOTO TELL
   .
   .
   .
   $ EXIT
   $ TELL:
   $ TYPE SYS$INPUT
   To use this procedure, you must enter a value for P1.
   .
   .
   .
   $ EXIT
```

この例では、IF コマンドはプロシージャに渡された最初のパラメータを調べます。このパラメータが HELP という文字列の場合、あるいはパラメータが指定されていない場合には、GOTO コマンドが実行され、制御は TELL というラベルの行に移ります。それ以外の場合には、プロシージャは EXIT コマンドが検出されるまで実行を継続します。TELL というラベルでは、TYPE コマンドがプロシージャの使用方法を示す入力ストリームのデータを表示します。

```
2. $ ON ERROR THEN GOTO CHECK
   .
   .
   .
   $ EXIT
   $ CHECK: ! Error handling routine
   .
   .
   .
   $ END:
   $ EXIT
```

ON コマンドは、エラー処理ルーチンを設定します。そのあと、コマンド・プロシ

GOTO

ージャ内で実行されるコマンドあるいはプロシージャが、エラーまたは重大なエラーを報告した場合には、GOTO コマンドは、制御を CHECK というラベルに移します。

HELP

HELP コマンドを実行すると、HELP 機能が起動します。HELP 機能は、コマンドの形式や説明、パラメータ、修飾子、およびシステム・メッセージを含め、システムの使用に関する情報を表示します。Topic? というプロンプトに対して、次の指示ができます。

- 知りたいコマンドまたはトピックの名前をタイプします。
- HELP の使用方法についてより詳細な情報が欲しいときは、INSTRUCTIONS とタイプします。
- 知りたいコマンドまたはトピックの名前がはっきり分からない場合は、HINTS とタイプします。
- HELP/MESSAGE ユーティリティについての知りたい場合は、HELP /MESSAGE とタイプします。
- 一番最後に要求したテキストをもう一度見たい場合は、疑問符 (?) をタイプします。
- HELP 機能を終了させたい場合は、Return を 1 回または数回押してください。

すべてのトピック名は短縮することができます。一意にトピックを選択できない場合は、あてはまるすべてのトピックが表示されます。

フォーマット

HELP [トピック[サブトピック...]]

パラメータ

トピック[サブトピック...]

ヘルプ・ライブラリの情報が必要なトピック、またはトピックとサブトピックを指定します。

説明

ヘルプ・ライブラリ内の情報は、階層構造になっています。階層構造の各レベルは、次のとおりです。

1. なし — キーワードを指定しない場合には、HELP コマンドの説明、ルート・ライブラリに含まれているトピックのリストが表示されます。リストに含まれる各項目は、階層構造の第 1 レベルのキーワードです。
2. トピック名 — キーワードとしてトピック名を指定した場合は、ルート・ライブラリ、または他の使用可能な省略時のライブラリに含まれているトピックの説明が表示されます。また、このトピックに関する追加情報を要求するためのキーワードも表示されます。
3. トピック名とサブトピック — トピックのあとにサブトピックを指定した場合は、指定したサブトピックの説明が表示されます。
4. 上記のいずれかの後の@ファイル指定 — 現在のルート・ライブラリと置き換えるために HELP ライブラリを指定した場合は、指定されたトピックまたはサブトピックの説明を、指定されたライブラリから検索します。ファイル指定の形式は、/LIBRARY 修飾子に指定されるファイル指定と同じ形式です。しかし、指定されたライブラリが使用可能な省略時のユーザ定義ライブラリである場合には、ファイル指定は、省略時のライブラリの論理名変換の中で、それぞれ一意に識別できる文字列まで省略できます。

OpenVMS 上で最も簡単な形式で HELP 機能を使用する方法は、ターミナルから HELP コマンドを入力することです。HELP コマンドを入力すると、HELP 機能はユーザのターミナルにトピックの一覧と Topic? というプロンプトを表示します。あるトピックに関する情報を表示したい場合は、Topic? プロンプトに対してトピック名を入力します。システムは、そのトピックに関する情報を表示します。

トピックにサブトピックがある場合は、サブトピックの一覧と Subtopic? というプロンプトが表示されます。あるサブトピックに関する情報を表示したい場合は、Subtopic? プロンプトに対してサブトピック名を入力します。他のトピックの情報を見たい場合は、Return を押します。Topic? プロンプトが表示されている時は、他のトピックの情報を表示させることができます。HELP 機能を終了し DCL レベルに戻るには、Return を押します。

キーワードの代わりにアスタリスク(*)を使用すると、HELP コマンドはアスタリスクを指定したレベルで使用可能な、すべての情報を表示します。たとえば、HELP COPY *と入力すると、COPY 以下のすべてのサブトピックを表示します。

主キーワードのすぐ後に反復記号(...)を使用すると、指定したトピックとそのトピックのすべてのサブトピックに関する、すべての情報が表示されます。たとえば HELP COPY... と指定すると、COPY というトピックに関する情報と、COPY の下のすべてのサブトピックに関する情報が表示されます。

アスタリスク(*)とパーセント記号(%)ワイルドカード文字を、キーワードに指定することはできません。

修飾子

/EXACT

正確に一致する文字列を検索する場合、/PAGE=SAVE および/SEARCH 修飾子とともに使用します。この時、検索文字列は引用符(“ ”)で囲まなければなりません。

/SEARCH 修飾子を指定せずに/EXACT 修飾子を指定した場合は、Find キー (E1) を押して検索文字列を設定すると正確に一致する文字列を検索できます。

/HIGHLIGHT[=キーワード]

/PAGE=SAVE および/SEARCH 修飾子とともに使用して、検索文字列が見つかった時に強調表示する方法を指定します。文字列が見つかったら、その行全体が強調表示されます。キーワードには、BOLD、BLINK、REVERSE、UNDERLINE があり、省略時の設定は BOLD です。

/INSTRUCTIONS (省略時の設定)

/NOINSTRUCTIONS

(トピックが指定されていない場合) トピックの一覧とともに、HELP コマンドの説明を表示します。省略時の設定では、HELP 機能の説明と形式、およびトピックの一覧が表示されます。/NOINSTRUCTIONS 修飾子を指定すると、トピックの一覧だけが表示されます。

/LIBLIST (省略時の設定)

/NOLIBLIST

使用できるすべてのヘルプ・ライブラリを表示します。

/LIBRARY=ファイル指定

/NOLIBRARY

省略時の設定のライブラリ SYS\$HELP:HELPLIB.HLB の代わりに、代替のヘルプ・ライブラリを使用することを指定します。ここで指定したヘルプ・ライブラリはメイン (ルート)・ヘルプ・ライブラリとして使用され、省略時のユーザ定義のヘルプ・ライブラリを検索する前に、このライブラリから HELP 情報を検索します。

装置とディレクトリの指定を省略すると、省略時の値として SYS\$HELP が使用されます。SYS\$HELP は、システム・ヘルプ・ライブラリの位置を示す論理名です。省略時のファイル・タイプは.HLB です。

省略時のヘルプ・ライブラリを検索しないようにするには、/NOLIBRARY 修飾子を指定します。

/MESSAGE

システム・メッセージの説明を表示します。詳細は、HELP/MESSAGE コマンドの説明を参照してください。

/OUTPUT[=ファイル指定]
/NOOUTPUT

コマンドの出力を送る場所を制御します。省略時の設定では、出力は SYS\$OUTPUT に送られます。SYS\$OUTPUT は、現在のプロセスの省略時の設定の出力ストリームまたは装置です。

/OUTPUT 修飾子で部分ファイル指定を行う (たとえば /OUTPUT=[JONES]) と、省略時のファイル名は HELP、省略時のファイル・タイプは .LIS になります。アスタリスク(*)およびパーセント記号(%)ワイルドカード文字を使用することはできません。

/NOOUTPUT 修飾子を指定すると、どこにも出力されません。

/PAGE[=キーワード]
/NOPAGE (省略時の設定)

画面での情報の表示を制御します。

/PAGE 修飾子とともに次のキーワードを指定することができます。

CLEAR_SCREEN	各ページを表示する前に画面をクリアします。
SCROLL	一行ずつ情報を表示します。
SAVE[=n]	情報の表示を制御することができます。nは履歴を保持するページ数です。

/PAGE=SAVE 修飾子を指定すると、最大 5 画面 (最大 255 カラムまで) 分の履歴を保存できます。/PAGE=SAVE 修飾子を使用すると、次のキーを使用して画面を移動することができます。

キー	説明
Up arrow key, Ctrl/B	1 行ずつスクロールアップ
Down arrow key	1 行ずつスクロールダウン
Left arrow key	1 カラム左シフト
Right arrow key	1 カラム右シフト
Find (E1)	文字列検索を起動
Insert Here (E2)	半画面右シフト
Remove (E3)	半画面左シフト
Select (E4)	80/132 カラム切り替え
Prev Screen (E5)	前ページに移動
Next Screen (E6), Return, Enter, Space	次ページに移動
F10, Ctrl/Z	終了 (ユーティリティにより異なる場合があります)
Help (F15)	ユーティリティ・ヘルプ・テキストを表示
Do (F16)	最新 (現在) 画面と (履歴内で) 最古画面の入れ替え
Ctrl/W	再表示

/PAGE 修飾子と/OUTPUT 修飾子を同時に指定することはできません。

/PROMPT (省略時の設定)
/NOPROMPT

最初の HELP 要求が処理された後、HELP が会話型モードのセッションを開始するかどうかを制御します。/NOPROMPT 修飾子を指定すると、要求した情報が表示された後、DCL コマンド・レベルに戻ります。

/PROMPT 修飾子が有効な場合は、特定のヘルプ・トピックまたはサブトピックを指定するように要求する、プロンプト (4 つのプロンプトのうちいずれか 1 つ) が表示されます。ヘルプ情報の階層構造のレベルにより、異なるプロンプトが表示されます。次のその 4 つのプロンプトを示します。

- 1. Topic? — ルート・ライブラリはメイン・ライブラリであり、現在、特定のトピックに関する情報を表示するために HELP を調べていないことを示します。
- 2. [ライブラリ指定] Topic? — ルート・ライブラリは、メイン・ライブラリ以外のライブラリです。現在、特定のトピックに関する情報を表示するために HELP を調べていないことを示します。
- 3. [キーワード] Subtopic? — ルート・ライブラリがメイン・ライブラリであり、現在、特定のトピック (または) サブトピックに関する情報を表示するために、HELP を調べていることを示します。
- 4. 上記の 2 と 3 の組み合わせ。

これらのプロンプトのいずれか 1 つが表示されたら、次の表のいずれか 1 つの応答を入力します。

応答	現在のプロンプト	動作
キーワード[...]	1,2	使用可能なすべてのライブラリから、指定したキーワードを検索する。
	3,4	現在のトピックおよびサブトピックの両方、またはどちらかに関する追加のヘルプ・ライブラリから、指定したキーワードを検索する。
@ファイル指定 キーワード[...]	1,2	上記と同じ。ただし@ファイル指定で指定したライブラリが、ルート・ライブラリである点が異なる。指定したライブラリが存在しない場合は、HELP は@ファイル指定を普通のキーワードとして扱う。 ルート・ライブラリで利用できるトピックの一覧を表示する。
	3,4	上記と同じ。@ファイル指定を普通のキーワードとして扱う。 ヘルプが存在する現在のトピック (またはサブトピック) のサブトピックのリストを表示する。
<div>Return</div>	1	HELP を終了する。
	2	ルート・ライブラリをメイン・ライブラリに変更する。
	3,4	1 つ上のレベルで、トピックまたはサブトピックを要求するプロンプトを表示する。
<div>Ctrl/Z</div>	1,2,3,4	HELP を終了する。

/SEARCH="文字列 "

/PAGE=SAVE 修飾子とともに使用し、表示される情報内で検索したい文字列を指定します。文字列中にスペース文字を入りたい場合は、検索文字列を二重引用符で囲まなければなりません。

情報が表示されている時に、Find キー (E1) を押して検索文字列を動的に変更することができます。この場合、二重引用符は必要ありません。

/USERLIBRARY=(レベル[,...])

/NOUSERLIBRARY

使用可能なライブラリで情報を検索するレベルを指定します。次のレベルを指定できます。

PROCESS	プロセス・レベルで定義したライブラリ
GROUP	グループ・レベルで定義したライブラリ
SYSTEM	システム・レベルで定義したライブラリ
ALL	すべてのライブラリ (省略時の設定)
NONE	ライブラリなし (/NOUSERLIBRARY 修飾子を指定した場合と同じ)

使用可能なヘルプ・ライブラリは、論理名 HLP\$LIBRARY, HLP\$LIBRARY_1, HLP\$LIBRARY_2... で定義されたライブラリです。各ライブラリは、この順序で検索されます。つまりルート (現在の) ライブラリ, メイン・ライブラリ (ルート・ライブラリと異なる場合), プロセス・レベルで定義されたライブラリ, グループ・レベルで定義されたライブラリ, システム・レベルで定義されたライブラリ, ルート・ライブラリの順序です。検索した結果、指定したトピックを見つけられなかった場合には、ルート・ライブラリがもう一度検索され、コンテキストは、検索が開始されたルート・ライブラリに戻されます。省略時の設定は、/USERLIBRARY=ALL です。検索のためのレベルを 1 つしか指定しない場合は、括弧は省略できます。

/WRAP

/NOWRAP (省略時の設定)

/PAGE=SAVE 修飾子とともに使用し、画面のカラム数を制限し、画面の幅を超える行を改行することを指定します。

/NOWRAP 修飾子を指定すると、画面の幅以上に行を続けます。/PAGE=SAVE 修飾子により提供されるスクロール機能 (右および左) を使用すれば、行すべてを読むことができます。

例

```
1. $ HELP
HELP
.
. (HELP message text and list of topics)
.
Topic?
```

この例では修飾子やパラメータを指定せずに HELP コマンドを入力しています。ここではルート・ヘルプ・ライブラリ SYS\$HELPLIB.HLB で使用できるヘルプ・トピックが表示されます。

Topic? というプロンプトに対して、いずれか 1 つのトピックを入力すると、HELP はそのトピックに関する情報と、サブトピックが存在する場合にはそのリストを表示します。1 つまたは複数のサブトピックが存在する場合は、サブトピックを要求するプロンプトが表示されます。

```
Topic? ASSIGN
ASSIGN
.
. (HELP message text and subtopics)
.
ASSIGN Subtopic?
```

サブトピック名をタイプすると、HELP はそのサブトピックに関する情報を表示します。次の例を参照してください。

```
ASSIGN Subtopic? Name
ASSIGN
Name
.
. (HELP message text and subtopics, if any)
.
ASSIGN Subtopic?
```

1 つまたは複数のサブ・サブトピックが存在する場合は、HELP はサブ・サブトピックを要求するプロンプトを表示します。この例のようにサブ・サブトピックが存在しない場合は、現在調べているトピックの、別のサブトピックの入力を要求するプロンプトが表示されます。

疑問符(?)タイプすると、現在のレベルの HELP メッセージとオプションが再表示されます。Return を押すと、次のいずれかの操作が行われます。

- サブトピック・レベルの場合は、前のヘルプ・レベルに戻る。
- 第 1 レベルの場合は、HELP を終了する。

Ctrl/Z を押すと、どのレベルでも HELP は終了します。

2. \$ HELP COPY...

この HELP コマンドを入力すると、COPY コマンド、および COPY コマンドのパラメータや修飾子の説明が表示されます。反復記号(...)は、トピック・レベルでしか使用できない点に注意してください。サブトピック・レベルで使用することはできません。

3. \$ HELP/NOPROMPT ASSIGN/GROUP

```
.
. (ASSIGN/GROUP HELP message)
.
$
$ HELP/NOPROMPT/PAGE EDIT *
.
. (HELP messages on all first-level EDIT subtopics)
.
$
```

2つの HELP コマンドは、特定のトピックに関する情報を要求しています。どちらの場合も、HELP コマンドはユーザが要求したヘルプ・メッセージを表示し、その後、DCL コマンド・レベルに制御を戻し、ドル記号プロンプト(\$)が表示されます。

最初のコマンドは、ASSIGN コマンドの/GROUP 修飾子に関する情報を要求しています。2番目の例のアスタリスク(*)は、ワイルドカード文字です。これは、すべての EDIT サブトピックに関する情報を要求しており、これらの情報は、アルファベット順に表示されます。/NOPROMPT 修飾子を指定しているため、どちらの場合もプロンプトは表示されません。2つめの HELP コマンドでは/PAGE 修飾子を指定しているため、画面に情報が表示されると、そこで表示が一時止まります。

4. \$ HELP FILL

```
Sorry, no documentation on FILL
Additional information available:
```

```
.
. (list of first-level topics )
.
Topic? @EDTHELP FILL
FILL
.
. (FILL HELP message)
.
@EDTHELP Topic?
```

省略時のヘルプ・ライブラリに含まれていないトピックに関するヘルプ情報を要求する場合は、そのトピックを他のヘルプ・ライブラリから検索するよう、HELP に指示を与えることができます。この例では、@EDTHELP FILL というコマンドを入力することにより、EDT エディタ・コマンドである FILL に関する情報を、SYS\$HELP:EDTHELP.HLB というヘルプ・ライブラリから検索するよう HELP に指示しています。HELP はメッセージを表示し、他の EDT エディタ・トピックを指定するよう要求するプロンプトを表示します。


```

5. $ SET DEFAULT SYS$HELP
   $ DEFINE HLP$LIBRARY EDTHELP
   $ DEFINE HLP$LIBRARY_1 MAILHELP
   $ DEFINE HLP$LIBRARY_2 BASIC
   $ DEFINE HLP$LIBRARY_3 DISK2:[MALCOLM]FLIP
   $ HELP REM

```

論理名を使用してライブラリを定義しておく、指定したトピックを OpenVMS ルート・ヘルプ・ライブラリで見つけれない場合、HELP は自動的にこれらのライブラリを検索します。この例では、省略時のルート・ライブラリ SYS\$HELP:HELPLIB.HLB に加え、これらのライブラリも検索するよう HELP に指示しています。

DEFINE 文でユーザ定義のヘルプ・ライブラリに論理名を割り当てています。HELP は、ルート・ライブラリを検索した後、これらのユーザ定義のヘルプ・ライブラリを検索します。最初の 3 つのエントリは、現在の省略時のディレクトリにあるヘルプ・ライブラリです。省略時の設定では、HELP は、論理名 SYS\$HELP で定義されるディレクトリ内のヘルプ・ライブラリを検索します。4 つ目のエントリは、ディレクトリ DISK2:[MALCOLM]内のヘルプ・ライブラリ FLIP.HLB です。ヘルプ・ライブラリを定義するために使用する論理名は、連続的に指定しなければならない点に注意してください。つまり、論理名の数字をスキップさせることはできません。

この例では HELP は、まずルート・ライブラリで REM を検索します。次に、HLP\$LIBRARY、HLP\$LIBRARY_1、HLP\$LIBRARY_2 という順に検索します。これは REM が見つかるまで、またはすべてのライブラリで見つけれなかったと分かるまで、検索を続けます。BASIC.HLB ライブラリで REM が見つかった場合 HELP は、REM に関する情報を表示し、BASIC.HLB ライブラリ内のサブトピックを指定するよう要求します。BASIC.HLB ライブラリにないトピックを指定すると、HELP は、定義したヘルプ・ライブラリでそのトピックを検索します。

HELP/MESSAGE

システム・メッセージの説明を表示します。

フォーマット

HELP/MESSAGE [/修飾子[...]] [検索文字列]

パラメータ

検索文字列

メッセージ識別子、またはメッセージ・テキスト中の 1 つまたは複数の単語を指定します。省略時の設定では、HELP/MESSAGE は最後に実行したコマンドにより生成されたメッセージ (つまり、現在 CLI シンボル \$STATUS に格納されている値に関連するメッセージ) の説明を表示します。

Help Message ユーティリティ (MSGHLP) は、次の規則に従って検索文字列を扱います。

- 2 文字以下の英数字文字は無視する。
- 単語は任意の順序で指定できる。

最初に一般的ではない単語を指定すると、検索にかかる時間を短縮することができます。

- 非英数字文字は、検索時に無視される (ただしメッセージの前に付くパーセント記号 (%) およびハイフン (-) は例外)。そのためこれらの特殊文字を含み変数 (たとえばファイル名) を削除したメッセージを、検索文字列に指定することができる。

指定したメッセージを Help Message がデータベースで見つけられなかった場合は、先行の特殊文字、ファシリティ・コード、および重大度を削除し、もう 1 回コマンドを実行します。いくつかのメッセージは、ファシリティ固有のメッセージではなく、"共有"メッセージと記されています。

- Help Message は、検索文字列に指定した文字で始まるすべての単語を検索します。単語全体が一致する単語を検索する場合は、/WORD_MATCH=WHOLE_WORD 修飾子を指定します。

説明

Help Message ユーティリティは、テキスト・ファイル中のメッセージの説明にアクセスします。このテキスト・ファイルは、最新の『OpenVMS system messages documentation』から派生したファイルです。またオプションで(ユーザが指定したメッセージ・ドキュメントを含む)他のソース・ファイルから派生させることもできます。省略時の設定では Help Message は、最後に終了したコマンドに関する情報を提供します。

1 つまたは複数の指定したファシリティにより生成される、すべてのメッセージを抽出することができます。この出力をファイルに書き込むと、ユーザ独自のメッセージ・ドキュメントを作成し印刷することができます。

Help Message データベースにコメントまたはメッセージを追加する方法についての詳細は、『OpenVMS System Messages: Companion Guide for Help Message Users』を参照してください。

修飾子

/BRIEF

メッセージ・テキストだけを表示します。

/DELETE=ファイル名.MSGHLP

以下のファイルの 1 つから指定した最初に見つかった.MSGHLP ファイル内のすべてのメッセージを削除します。

- /LIBRARY 修飾子で指定された.MSGHLP\$DATA ファイル
- /LIBRARY 修飾子で指定された検索順序内の、最初の.MSGHLP\$DATA ファイル
- 省略時の検索パスで最初の.MSGHLP\$DATA ファイル (論理名 MSGHLP\$LIBRARY で定義されたファイル)
- SYS\$HELP:MSGHLP\$LIBRARY.MSGHLP\$DATA (省略時の.MSGHLP\$DATA ファイル)

弊社が提供したデータベースからメッセージを削除するには、弊社が提供した.MSGHLP\$DATA ファイルに対する書き込みアクセス権が必要です。

注意

検索文字列を指定して.MSGHLP ファイルを作成する場合は、出力される.MSGHLP ファイルをチェックして、データベースから削除したくない意図しないメッセージが、検索されなかったことを確認してください。このようなメッセージは、削除操作を行う前に.MSGHLP ファイルの外で編集してください。

/EXTRACT=ファイル名.MSGHLP

データベースからメッセージを取り出し、編集可能な.MSGHLP ファイルを作成します。このファイルは、/INSERT および/DELETE 操作の入力としても使用できます。/EXTRACT 修飾子は、.MSGHLP\$DATA ファイル、/LIBRARY 修飾子で指定される論理検索順序、または論理名 MSGHLP\$LIBRARY で定義される省略時の検索順序から、データを取り出します。/EXTRACT 修飾子を指定しない場合は、Help Message は省略時の標準的な形式で出力します (/OUTPUT を参照してください)。

/FACILITY=?

/FACILITY=(ファシリティ名[,...])

/FACILITY=ALL

データベース内で一致するかどうかを検索されるファシリティを指定します。

省略時のデータベースまたは/LIBRARY で指定されたデータベース内のすべてのファシリティのリストを出力するには、/FACILITY=? と入力します。

検索範囲を狭めるために、1 つまたは複数のファシリティ名を/FACILITY に指定します (複数のファシリティを指定する場合には、コンマで区切り括弧で囲みます)。Help Message は、指定されたファシリティで生成されるメッセージの中で一致するものだけを出力します。

データベース内のすべてのファシリティのメッセージを出力するには、/FACILITY=ALL を指定します。他のファシリティが暗黙的に指定されない限り、/FACILITY=ALL が省略時の設定になります。これは、たとえば/STATUS を指定したり、\$STATUS という CLI シンボルの値を使用する場合などで、この場合には特定のファシリティが自動的に指定されます。同様に、ファシリティ名を含んだメッセージのカット・アンド・ペーストは、/FACILITY 修飾子の使用を無効にします。

/FACILITY 修飾子の使用についての詳細は、『OpenVMS System Messages: Companion Guide for Help Message Users』を参照してください。

/FULL (省略時の設定)

完全なメッセージの説明を出力します。メッセージ・テキスト、ファシリティ名、説明、ユーザ処置、ユーザのコメントが表示されます。

/INSERT=ファイル名.MSGHLP

/INSERT=TT:

指定された.MSGHLP ファイルで最初に見つかったファイル内の新しいまたは変更された情報で、以下のファイルのうち 1 つを更新します。または、/INSERT=TT: が指定された場合端末から入力されたデータを直接変更します。

- /LIBRARY 修飾子で指定された.MSGHLP\$DATA ファイル
- /LIBRARY 修飾子で指定された検索パス内の、最初の.MSGHLP\$DATA ファイル
- 省略時の検索パス内の最初の.MSGHLP\$DATA ファイル (論理名 MSGHLP\$LIBRARY で定義されたファイル)

- SYS\$HELP:MSGHLP\$LIBRARY.MSGHLP\$DATA (省略時の.MSGHLP\$DATA ファイル)

弊社が提供する.MSGHLP\$DATA ファイルにデータを挿入するには、このファイルに対する書き込みアクセス権が必要です。ユーザが指定したデータは、Help Message の出力中で、変更バーが付けられます。

```
/LIBRARY=ディスク:[ディレクトリ]ファイル名.MSGHLP$DATA
/LIBRARY=ディスク:[ディレクトリ]
/LIBRARY=論理名
```

現在のコマンドに対するメッセージ・データベースに、特定の.MSGHLP\$DATA ファイル、指定ディレクトリ内の全.MSGHLP\$DATA ファイル、または論理名による特定の検索パスを定義します。

大部分の操作では、省略時のデータベースは SYS\$HELP:MSGHLP\$LIBRARY.MSGHLP\$DATA であるか、論理名 MSGHLP\$LIBRARY で定義された.MSGHLP\$DATA ファイルの検索パスです。

/DELETE および/INSERT 操作では、省略時のデータベースは、SYS\$HELP:MSGHLP\$LIBRARY.MSGHLP\$DATA であるか、論理名 MSGHLP\$LIBRARY で定義された検索パスの最初のファイルです。

/OUTPUT=ファイル指定

指定したファイルに出力を書き込みます。省略時の設定では、Help Message は SYS\$OUTPUT に出力します。これは通常、ターミナルです。/OUTPUT=ファイル指定と/PAGE は同時に指定することはできません。

/PAGE (スクリーン表示の際の省略時の設定)
/NOPAGE

1 度に 1 画面分のターミナル出力を表示します。ページ長は、SET TERMINAL /PAGE で指定した値よりも 1 行だけ短い値に、自動的に設定されます。/PAGE と/OUTPUT=ファイル指定は、同時に指定することはできません。

/SECTION_FILE=*
/SECTION_FILE=ファイル指定

Help Message が、ファイル中のメッセージの\$STATUS 値を解釈できるよう、メッセージ・セクション・ファイルを指定します。省略時のファイル指定は SYS\$MESSAGE:.EXE です。/SECTION_FILE=*を指定すると、OpenVMS が提供するすべてのメッセージ・セクション・ファイルも自動的に含まれます。詳細は『OpenVMS System Messages: Companion Guide for Help Message Users』を参照してください。

注意

この修飾子を使用した結果は、SET MESSAGE コマンドによる結果とは関係ありません。Help Message ユーティリティと Message ユーティリティは、互いに影響を与えることはありません。結果を得るためには、各ユーティリティをそれぞれコーディングしなければなりません。

/SORT

/NOSORT (省略時の設定)

出力をアルファベット順にソートします。ソートに失敗した場合は、/WORK_FILES 修飾子を使用して再試行してください。

/STATUS=状態コード

/STATUS='シンボル'

/STATUS='\$STATUS' (省略時の設定)

指定した状態コードに相当するメッセージを出力します。状態コードは 10 進数、16 進数、またはアポストロフィで囲んだシンボルで指定します。先行する 0 は省略できますが、16 進数の前には "%X" を付けなければなりません。

HELP/MESSAGE コマンドに検索文字列が含まれない場合、Help Message は \$STATUS という CLI シンボルに相当するメッセージを、省略時の値として出力します。つまり、Help Message は、最後に実行したコマンドがどのように終了したかを示す情報を表示します。

/STATUS 修飾子を指定した場合は、検索文字列や/FACILITY 修飾子は指定できません。検索文字列を省略し、/STATUS='\$STATUS' という省略時設定を使用する場合は、/FACILITY を指定することはできません。

/WORD_MATCH=INITIAL_SUBSTRING (省略時の設定)

/WORD_MATCH=WHOLE_WORD

/WORD_MATCH=INITIAL_SUBSTRING は、検索文字列に指定した単語で始まるすべての単語を検索します。検索文字列には、複数の単語を指定することができます。すべての単語が一致するメッセージだけが出力されます。この時、単語の順序には関係がありません。

/WORD_MATCH=WHOLE_WORD は、単語全体が一致するものを検索し、指定した単語を細かに検索します。たとえば、ACC に関する精密な検索は、ACC という文字で始まる単語を持つ、多数の他のメッセージを表示します。

/WORK_FILES=nn

/WORK_FILES=0 (修飾子が省略された場合の省略時の設定)

/WORK_FILES=2 (値なしで修飾子が指定された場合の省略時の設定)

/SORT 修飾子を指定した時、使用する作業ファイルを指定します。nn には、0 から 10 の値を指定します。/SORT 修飾子が指定されていない場合は、この修飾子は無視されます。

例

1. `$ SHOW DEVICE KUDOS`
`%SYSTEM-W-NOSUCHDEV, no such device available`
`$ HELP/MESSAGE`

最初のコマンドはエラーになります。修飾子のない省略時の HELP/MESSAGE コマンドは、NOSUCHDEV という SYSTEM ファシリティ・メッセージの説明を表示します。

2. `$ HELP/MESSAGE ACCVIO`
`$ HELP/MESSAGE/BRIEF ACCVIO`
`$ HELP/MESSAGE/FACILITY=SYSTEM ACCVIO`
`$ HELP/MESSAGE VIRTUAL ACCESS`
`$ HELP/MESSAGE/STATUS=12`
`$ HELP/MESSAGE/STATUS=%XC`

これらのコマンドは、ACCVIO メッセージにアクセスし表示するためのいろいろな修飾子を、異なる形式で使用方法を示しています。

3. `$ HELP/MESSAGE/BRIEF ACC`
`$ HELP/MESSAGE/BRIEF/WORD_MATCH=WHOLE_WORD ACC`

最初のコマンドでは、Help Message は省略時の設定により“ACC”という文字列で始まる多くの単語を検索します。/WORD_MATCH=WHOLE_WORD 修飾子を指定すると、単語だけに厳密に一致するよう検索処理を改良することができます。

4. `$ HELP/MESSAGE/FACILITY=(BACKUP,SHARED)/SORT/OUTPUT=MESSAGES.TXT`

このコマンドは、BACKUP ファシリティから発行されるすべてのメッセージと、“Shared by several facilities,”と記載されているメッセージを選択し、アルファベット順に並べ、印刷可能な MESSAGES.TXT というファイルを出力します。

メッセージを選択し、それらをファイルに出力することによって、ユーザ独自の変更したメッセージ・ドキュメントを作成、および印刷することができます。

5. `$ HELP/MESSAGE/EXTRACT=BADMESSAGE.MSGHLP BADMESSAGE`
`$ HELP/MESSAGE/DELETE=BADMESSAGE.MSGHLP-`
`_ $ /LIBRARY=SYS$LOGIN:MYMESSAGES.MSGHLP$DATA`
`$ CONVERT SYS$LOGIN:MYMESSAGES.MSGHLP$DATA-`
`_ $ SYS$LOGIN:MYMESSAGES.MSGHLP$DATA`
`$ PURGE SYS$LOGIN:MYMESSAGES.MSGHLP$DATA`
`$ HELP/MESSAGE/INSERT=BADMESSAGE.MSGHLP`

最初のコマンドは、BADMESSAGE という仮のメッセージを省略時のデータベースから取り出し、それを BADMESSAGE.MSGHLP というファイルに出力します。

2 番目のコマンドは、/LIBRARY 修飾子に指定した MYMESSAGES.MSGHLP\$DATA というファイルから、BADMESSAGE の

説明を削除するために、この BADMESSAGE.MSGHLP というファイルを使用しています。

次の2つのコマンドは、削除後にディスク領域を節約するために、ファイル MYMESSAGES.MSGHLP\$DATA ファイルを圧縮しています。

最後のコマンドは、省略時の.MSGHLP\$DATA ファイルに BADMESSAGE メッセージを挿入するために、ファイル BADMESSAGE.MSGHLP を使用しています。ファイル BADMESSAGE>MSGHLP は、編集されている可能性があります。

```
6. $ HELP/MESSAGE/EXTRACT=NOSNO.MSGHLP NOSNO
$ EDIT/EDT NOSNO.MSGHLP
1NOSNO, can't ski; no snow
2XCSKI, XCSKI Program
3Your attempt to ski failed because there is no snow.
4Wait until there is snow and attempt the operation again.
5If you don't want to wait, go to a location where there is
5snow and ski there.
5
5Or, try ice skating instead!
[EXIT]
$ HELP/MESSAGE/INSERT=NOSNO.MSGHLP
```

このコマンドは、弊社が提供する.MSGHLP\$DATA ファイルに書き込みアクセス権を持つユーザが、弊社の提供しているメッセージにコメントを追加の様子を示しています。

最初のコマンドは、NOSNO という仮のメッセージをファイル NOSNO.MSGHLP に出力しています。2番目のコマンドで、その.MSGHLP ファイルを編集し、メッセージの最後にコメントを追加しています。各コメント行には、それが空白行であっても、“5”という文字が最初に付けられています。次のコマンドで NOSNO.MSGHLP を使用してデータベースを更新し、省略時の.MSGHLP\$DATA ファイルに変更したメッセージを挿入しています。

IF

式の値を評価し、次のように指定された構文に基づいて、コマンドを実行します。

- 評価が真の場合、THEN キーワードに続く 1 コマンド
- 条件が真の場合、\$THEN ステートメントに続く複数のコマンド
- 条件が偽の場合、\$ELSE ステートメントに続く複数のコマンド

フォーマット

```
$ IF 式 THEN [$]コマンド
```

または

```
$ IF 式
```

```
$ THEN [コマンド]
```

```
コマンド
```

```
·  
·  
·
```

```
$ [ELSE] [コマンド]
```

```
コマンド
```

```
·  
·  
·
```

```
$ ENDIF
```

注意

DCL コマンド名としてすでに使用されているシンボル名を割り当てないでください。コマンド・プロシージャの実行を妨げる可能性がありますので、IF, THEN, ELSE および GOTO のようなシンボルの割り当ては行わないでください。

パラメータ

式

実行されるテストを定義します。式は、1 つまたは複数の数値定数、文字列リテラル、シンボル名、またはレキシカル関数から構成でき、各要素は論理演算子、算術演算子、あるいは文字列演算子で区切ることができます。

IF コマンドに指定する式は、そのコマンドが実行されるときに、自動的に評価されます。英字から始まる文字列で、引用符で囲まれていない文字列は、すべてシンボル名またはレキシカル関数であると解釈され、IF コマンドは、その文字列を現在の値と置き換えます。

IF コマンドに指定する式のシンボル置換は、反復置換ではありません。つまり、各シンボルは1回だけしか置き換えられません。しかし、反復置換が必要な場合には、シンボル名の前に引用符、またはアンパサンドを指定します。

コマンド・インタプリタは、IF コマンドに未定義シンボルが含まれている場合、そのコマンドを実行しません。この場合、コマンド・インタプリタは、警告メッセージを表示し、プロシージャの次のコマンドを実行します。

演算子のまとめと式の指定方法についての詳細は、『OpenVMS ユーザーズ・マニュアル』を参照してください。

コマンド

式の結果が真または偽の場合、実行する1つまたは複数のコマンドを構文に従って指定します。

説明

IF コマンドは、式の値をテストして式の結果が真の場合、指定されたコマンドを実行します。結果が奇数整数値、文字 Y, y, T, または t で始まる文字列値、または奇数の数値文字列値の場合、式は真になります。

結果が偶数整数値、文字 Y, y, T, または t 以外で始まる文字列値、または偶数の数値文字列値の場合、式は偽になります。

例

```
1. $ COUNT = 0
   $ LOOP:
   $ COUNT = COUNT + 1
   .
   .
   .
   $ IF COUNT .LE. 10 THEN GOTO LOOP
   $ EXIT
```

この例は、COUNT というシンボル名と IF ステートメントを使用して、コマンド・プロシージャ内でループを設定する方法を示しています。IF ステートメントは COUNT の値を調べ、COUNT の値が 10 より大きくなったときに、EXIT コマンドを実行します。

```

2. $ IF P1 .EQS. "" THEN GOTO DEFAULT
   $ IF (P1 .EQS. "A") .OR. (P1 .EQS. "B") THEN GOTO 'P1'
   $ WRITE SYS$OUTPUT "Unrecognized parameter option 'P1' "
   $ EXIT
   $ A:      ! Process option a
   .
   .
   .
   $ EXIT
   $ B:      ! Process option b
   .
   .
   .
   $ EXIT
   $ DEFAULT: ! Default processing
   .
   .
   .
   $ EXIT

```

この例は、パラメータが渡されたどうかを調べるコマンド・プロシージャです。GOTO コマンドは、パラメータで指定されたラベルへ制御を移します。

プロシージャがパラメータで実行されると、プロシージャはそのパラメータを使用して分岐先のラベルを決めます。次に例を示します。

```
@TESTCOM A
```

上記のプロシージャを実行した場合、ラベル A へ分岐します。EEXIT コマンドは、ラベル B の手前でプロシージャを終了することに注意してください。

```

3. $ SET NOON
   .
   .
   .
   $ LINK CYGNUS,DRACO,SERVICE/LIBRARY
   $ IF $STATUS
   $ THEN
   $ RUN CYGNUS
   $ ELSE
   $ WRITE SYS$OUTPUT "LINK FAILED"
   $ ENDIF
   $ EXIT

```

このコマンド・プロシージャは、SET NOON コマンドでエラー・チェックを無効にしています。LINK コマンド実行後、IF コマンドは予約済みのグローバル・シンボル\$STATUS の値をテストします。\$STATUS の値がLINK コマンドの正常終了を示す場合は、次にプログラム CYGNUS を実行します。LINK コマンドがエラー状態値を返した場合は、コマンド・プロシージャはメッセージを出力して終了します。

IF

```
4. $ if 1 .eq. 1
$ then
$   if 2 .eq. 2
$   then
$     write sys$output "Hello!"
$   endif
$ endif
```

この例は、ネストした IF 構造体を使用する方法を示しています。

INITIALIZE

ディスクまたは磁気テープ・ボリュームをフォーマットしたり、ボリュームにラベルを書き込んだりします。また、構造情報を持つシステム・ファイルを除き、ディスクを空にします。ディスクの以前の内容はすべて失われます。

INITIALIZE コマンドの大部分の操作には、VOLPRO (ボリューム保護) 特権が必要です。

フォーマット

INITIALIZE 装置名[:] ボリューム・ラベル

パラメータ

装置名[:]

初期化するボリュームを物理的にマウントする装置の名前を指定します。

この時点で装置を割り当てる必要はありません。ただし、初期化の前には割り当てるようにしてください。

ボリューム・ラベル

ボリュームでエンコードする ID を指定します。ディスク・ボリュームの場合は、ANSI 文字を 12 文字まで指定できます。磁気テープ・ボリュームの場合は、英数字を 6 つまで指定できます。文字は自動的に大文字に変換されます。ディスク・ボリューム・ラベルは、英数字、ドル記号 (\$)、アンダスコア (_)、およびハイフン (-) だけで構成することをおすすめします。

磁気テープのボリューム・ラベルに ANSI a 文字を使用するには、ボリューム名を二重引用符(")で囲まなければなりません。ANSI a 文字についての詳細は、/LABEL 修飾子の説明を参照してください。

説明

OpenVMS オペレーティング・システムでのディスク・ボリュームの省略時の設定の形式は、Files-11 オンディスク構造レベル 2 と呼ばれています。磁気テープ・ボリュームの省略時の設定の形式は、磁気テープ・ラベルとファイル構造に関する情報交換用の ANSI 規格 Level 3 (ANSI X3.27-1978) に基づいています。

INITIALIZE コマンドを使用して、Files-11 オンディスク構造レベル 1 形式でディスク・ボリュームを初期化することもできます。

INITIALIZE

ボリュームを初期化するには VOLPRO 特権が必要です。ただし次のボリュームを初期化する場合は、VOLPRO 特権は必要ありません。

- 空のディスクまたは磁気テープ・ボリューム (つまり、何も書き込まれていないボリューム)。
- 現在の利用者識別コード (UIC)、または UIC[0,0]によって所有されるディスク・ボリューム。
- 初期化時に現在の UIC から保護しなかったために、現在の UIC による書き込み (W) アクセスが許可されている磁気テープ・ボリューム。

ボリュームを初期化してマウントした後に、SET SECURITY コマンドを使用してセキュリティ・プロファイルを変更できます。ディスク・ボリュームを初期化すると、そのルート・ディレクトリ (000000.DIR;1) のキャッシング属性はライトスルーに設定されます。つまり、省略時の設定では、ボリューム中に作成するすべてのファイルとディレクトリが、ライトスルーのキャッシング属性を継承します。キャッシング属性を変更するには、SET FILE コマンドを/CACHING_ATTRIBUTE 修飾子を付けて使用します。

磁気テープ・ボリュームを初期化する場合は常に、INITIALIZE コマンドはボリュームに対して読み込みを行います。磁気テープが空の場合は、次のような回復不可能なエラーを検出することがあります。

- ボリューム番号の誤りを示すエラー・メッセージ。

```
%INIT-F-VOLINV, volume is invalid
```

- 磁気テープの暴走。これは、前に書き込んだことがない、またはマシンの動作確認に使用されていた新しい磁気テープで頻繁に発生します。磁気テープの暴走を停止させる唯一の方法は、磁気テープ・ドライブをいったんオフラインにしてからオンラインに戻すことです。

上記のような回復不可能なエラーが発生した場合に、磁気テープを正常に初期化するためには、VOLPRO (ボリューム保護) 特権を持つアカウントから、次の修飾子を指定して INITIALIZE コマンドを各磁気テープに対して繰り返し実行します。

```
/OVERRIDE=(ACCESSIBILITY,EXPIRATION)
```

この修飾子を指定すると INITIALIZE コマンドは、磁気テープのラベルをチェックしなくなります。

VOLPRO 特権を持っている場合、INITIALIZE コマンドは所有権情報を読み込まずにディスクを初期化します。VOLPRO 特権を持っていない場合は、ディスク・ボリュームを初期化する前にその所有権を調べます。ディスクが空の場合、または形式が誤っている場合は、回復不可能なドライブ・エラーが発生することがあります。空のディスクまたは誤った形式のディスクによって回復不可能なドライブ・エラーが発生した場合は、VOLPRO 特権を持つアカウントから、/DENSITY 修飾子を指定して

INITIALIZE コマンドを各磁気テープに対して実行すると、ディスクを正常に初期化できます。

ほとんどの INITIALIZE コマンド修飾子には、入出力 (I/O) 効率を最大にするパラメータを指定できます。

修飾子

/ACCESSED=ディレクトリ数

Files-11 オンディスク構造レベル 1 (ODS-1) ディスクにだけ有効です。

ディスク・ボリュームに対して、システム空間で作成できるディレクトリ数を 0 ~ 255 の範囲で指定します。省略時の設定の値は 3 です。

/BADBLOCKS=(領域[,...])

ディスク・ボリュームに対して、ボリュームの不良領域を指定します。INITIALIZE コマンドは、この領域を割り当て済みとしてマークして、データを書き込めないようにします。

領域の指定に使用できる形式は、次のとおりです。

lbn[: 数]	割り当て済みとしてマークする、最初のブロックの論理ブロック番号 (LBN)、または、最初のブロックから数えたブロック数 (省略可能)。
sec.trk.cyl[: 数]	割り当て済みとしてマークする、最初のブロックのセクタ、トラック、シリンダ、または、最初のブロックから数えたブロック数 (省略可能)。

弊社が提供し、OpenVMS オペレーティング・システムでサポートしている媒体は、ディスクと TU58 カートリッジを除いて、すべて工場出荷時にフォーマットされていますが、不良ブロック・データが含まれています。Bad Block Locator ユーティリティ (BAD) または診断フォーマッタ EVRAC を使用して、不良ブロック・データをリフレッシュしたり、不良ブロック・データをマークしたりできます (ただし、ディスクと TU58 カートリッジを除きます)。/BADBLOCKS 修飾子は、ボリュームの不良ブロック・データとして識別できない不良ブロックを指定する場合にだけ必要です。

DIGITAL Storage Architecture (DSA) ディスク (たとえば、UDA-50 や HSC50 制御装置に接続されているディスク) では、不良ブロックが制御装置によって処理されるため、ファイル・システムからは、論理的には不良ブロックは存在しないように見えます。

BAD を実行する方法については、『OpenVMS Bad Block Locator Utility Manual』(ドキュメンテーション CD-ROM に用意されています) を参照してください。

/CLUSTER_SIZE=ブロック数

ディスク・ボリュームに対して、最小割り当て単位をブロック数で定義します。1つのボリュームに指定できる最大のサイズは、16382 ブロック、またはそのボリューム・サイズの 1/50 のどちらか小さい方の値です。

Files-11 オンディスク構造レベル 5 ディスク (ODS-5) の場合、省略時のクラスタ・サイズは 16 です。この場合最小の値は、次の式から計算できます。

$(\text{ブロック数で表わしたディスク・サイズ}) / (65535 \times 4096)$

端数は整数値に切り上げなければならず、省略時には、最も近い 16 の倍数に切り上げられます。

Files-11 オンディスク構造レベル 2 ディスク (ODS-2) の場合、クラスタ・サイズの省略時の設定は、ディスク容量によって異なります。ディスク容量が 50,000 ブロックより小さいディスクでは、省略時の設定は 1 です。ディスク容量が 50,000 ブロックより大きいディスクでは、省略時の設定は 16 あるいは次の式によって求められる値以上です。

$(\text{ブロック数で表わしたディスク・サイズ}) / (255 \times 4096)$

端数は整数値に切り上げなければならず、省略時には、最も近い 16 の倍数に切り上げられます。

注意

V7.2 以降では、ODS-2 の計算式で求められる値よりも小さな値を ODS-2 ボリュームのクラスタ・サイズに指定することができます。しかし、このボリュームを V7.2 よりも前のバージョンを稼働しているシステムでマウントしようとした場合、次のエラーが表示されてマウントが中断します。

```
%MOUNT-F-FILESTRUCT, unsupported file structure level
```

ODS-2 ディスクの初期化時に省略時の値を選択した場合、そのディスクは OpenVMS の前のバージョンでマウントすることができます。

ODS-1 ディスクの場合、クラスタ・サイズは常に 1 にしなければなりません。

注意

/LIMIT を指定し、/CLUSTER_SIZE の値を指定しない場合、/CLUSTER_SIZE=8 の値が使用されます。

/DATA_CHECK[=(オプション[,...])]

ディスクでの読み込みと書き込みの操作をすべてチェックします。省略時の設定では、データ・チェックは行われません。次のオプションのいずれか、または両方を指定します。

READ 読み込み操作をすべてチェックします。
WRITE 書き込み操作をすべてチェックします。/DATA_CHECK 修飾子だけを指定した場合の省略時の設定です。

ディスクの初期化時に指定したチェック方法に従いたくない場合は、MOUNT コマンドで/DATA_CHECK 修飾子を指定してボリュームをマウントします。

/DENSITY=記録密度

特定のテープとディスクのフォーマット密度の値を指定することができます。

磁気テープ・ボリュームでは、磁気テープに書き込む際の密度をビット/インチ (bpi) で指定します。指定できる密度値は 800 bpi, 1600 bpi, または 6250 bpi です (ただし磁気テープ・ドライブがサポートしている必要があります)。

空の磁気テープに対して密度値を指定しなかった場合、システムはそのテープ・ドライブで使用可能な最大値を省略時の密度として使用します。6250 bpi, 1600 bpi, および 800 bpi で動作するドライブでは、省略時の密度は 6250 bpi となります。

以前に書き込みが行われた磁気テープに対して密度値を指定しなかった場合、システムはボリューム上の最初のレコードの密度を使用します。そのレコードが異常なほど短ければ、密度値の省略時の値は設定されません。

/DENSITY 修飾子は、テープ装置 TF には適用できません。

指定できるテープ記録密度の値です。

キーワード	意味
DEFAULT	省略時の密度
800	NRZI 800 BPI
1600	PE 1600 BPI
6250	GRC 6250 BPI
3480	IBM 3480 HPC 39872 BPI
3490E	IBM 3480 圧縮
833	DLT TK50: 833 BPI
TK50	DLT TK50: 833 BPI
TK70	DLT TK70: 1250 BPI
6250	RV80 6250 BPI EQUIVALENT
注意: 上記のキーワードは OpenVMS V7.2 よりも前の TMSCP/TUDRIVER コードでのみ有効です。この他のキーワードは Alpha のみでサポートされています。	
TK85	DLT Tx85: 10625 BPI - Cmpt III - Alpha/I64 のみ

INITIALIZE

キーワード	意味
TK86	DLT Tx86: 10626 BPI - Cmpt III - Alpha/I64 のみ
TK87	DLT Tx87: 62500 BPI - Cmpt III - Alpha/I64 のみ
TK88	DLT Tx88: (Quantum 4000) - Cmpt IV - Alpha/I64 のみ
TK89	DLT Tx89: (Quantum 7000) - Cmpt IV - Alpha/I64 のみ
QIC	すべての QIC 装置は装置設定のみ - Alpha/I64 のみ
8200	Exa-Byte 8200 - Alpha/I64 のみ
8500	Exa-Byte 8500 - Alpha/I64 のみ
DDS1	Digital Data Storage 1 - 2G - Alpha/I64 のみ
DDS2	Digital Data Storage 2 - 4G - Alpha/I64 のみ
DDS3	Digital Data Storage 3 - 8-10G - Alpha/I64 のみ
DDS4	Digital Data Storage 4 - Alpha/I64 のみ
AIT1	Sony Advanced Intelligent Tape 1 - Alpha/I64 のみ
AIT2	Sony Advanced Intelligent Tape 2 - Alpha/I64 のみ
AIT3	Sony Advanced Intelligent Tape 3 - Alpha/I64 のみ
AIT4	Sony Advanced Intelligent Tape 4 - Alpha/I64 のみ
DLT8000	DLT 8000 - Alpha/I64 のみ
8900	Exabyte 8900 - Alpha/I64 のみ
SDLT	SuperDLT - Alpha/I64 のみ
SDLT320	SuperDLT320 - Alpha/I64 のみ

テープ密度のキーワードは短縮することはできません。

RXnn ディスク・ドライブでディスクをフォーマットするには、INITIALIZE /DENSITY コマンドを使用します。次のように記録密度を指定して、ディスクをフォーマットします。

キーワード	意味
single	RX01 - 8 インチ
double	RX02 - 8 インチ
dd	倍密度: 720K - 3 1/2 インチ
hd	高密度: 1.44MB - 3 1/2 インチ
ed	拡張高密度: 2.88MB - 3 1/2 インチ

ドライブで初期化するディスクに記録密度を指定しない場合は、そのボリュームが最後にフォーマットされたときの密度のままになります。

注意

倍密度でフォーマットしたディスクは、単密度で再フォーマットしない限り、VAX-11/780 のコンソール・ブロック記憶装置 (RX01 ドライブ) では読み込みまたは書き込みができません。

RX33 ディスクは、RX50 ディスク・ドライブでは読み込みまたは書き込みができません。RX50 ディスクは、RX33 ディスク・ドライブで読み込みと書

き込みができます。しかし、RX33 ディスク・ドライブでフォーマットはできません。

/DIRECTORIES=エントリ数

この修飾子の効果は、ディスク構造によって異なります。

- ODS-1 については、/DIRECTORIES は指定されたディレクトリ・エントリ数用のスペースを 000000.DIR (MFD) に確保します。
- ODS-2 および ODS-5 については、/DIRECTORIES は MFD の初期サイズを設定します。指定された数を 16 で除算して事前割り当てブロック数を生成します。続いて完全なクラスタ数に切り上げられます。

エントリ数は、16 ~ 16000 の整数でなければなりません。省略時の設定の値は 16 です。

/ERASE[=キーワード]

/NOERASE (省略時の設定)

セキュリティのためのデータの消去 (DSE) を行うかどうかを指定します。ディスク・ボリュームの場合には、ボリューム属性に ERASE_ON_DELETE を設定するかどうかも指定します。

/ERASE 修飾子は、Files-11 オンディスク構造レベル 2 (ODS-2) およびレベル 5 (ODS-5) ディスク・ボリュームおよび ANSI 磁気テープ・ボリュームに使用でき、ハードウェア消去機能をサポートする TU78 や MSCP 磁気テープなどに有効です。

テープ・ボリュームでは、/ERASE を指定すると、削除対象のデータに上書きすることで物理的に破壊します。

ディスク・ボリュームでは、キーワードなしで /ERASE を指定すると、このコマンドは SET VOLUME/ERASE_ON_DELETE と機能的に等しくなり、以下のことを実行します。

- ブロックを初期化する前に、ボリューム上の全ブロックに対して、システムで指定された消去パターンを書き込むことで、セキュリティのためのデータの消去 (DSE) を行います。DSE 操作にかかる時間はボリューム・サイズに依存します。
- ボリューム属性に ERASE_ON_DELETE を設定し、ボリューム上の各ファイルが削除される際に、DSE によって消去されるようにします。

ディスク・ボリュームでは、2 つのオプション・キーワードを使用して、上記の動作のどちらかだけを個別に指定することができます。

- /ERASE=INIT

ボリュームを初期化する前に、ボリューム上でセキュリティのためのデータの消去 (DSE) 操作を行います。ボリューム属性に ERASE_ON_DELETE を設定しません。この操作は、/ERASE=DELETE を指定した場合よりも長い時間がかかり、SET VOLUME/NOERASE_ON_DELETE を実行するのと同じこととなります。

- /ERASE=DELETE

ボリューム属性 ERASE_ON_DELETE を設定しますが、ディスクに対して DSE 操作は行いません。

どちらのキーワードも指定しないか、両方のキーワードを指定した場合は、両方の動作が実行されます。すなわち、/ERASE は /ERASE=(INIT,DELETE) と同じこととなります。

/EXTENSION=ブロック数

ディスク・ボリュームに対して、ボリューム上のすべてのファイル用の省略時の設定の拡張サイズとして使用するブロック数を指定します。更新時に、ファイルのサイズがはじめの省略時の設定の割り当てより大きくなると、拡張サイズの省略時の設定が使用されます。Files-11 オンディスク構造レベル 2 およびレベル 5 ディスクの場合、ブロック数パラメータの値には、0 ~ 65,535 の値を指定します。省略時の設定の値は 5 です。Files-11 オンディスク構造レベル 1 ディスクの場合、この値には 0 ~ 255 の値を指定します。

ファイルに対して異なる拡張サイズが設定されてなく、プロセスに対して SET RMS_DEFAULT コマンドを使用して省略時の設定の拡張サイズが設定されていない場合に限り、OpenVMS オペレーティング・システムは、省略時の設定のボリューム拡張サイズを使用します。

/FILE_PROTECTION=コード

Files-11 オンディスク構造レベル 1 (ODS-1) ディスクにのみ有効です。

ディスク・ボリュームに対して、ボリューム内のすべてのファイルに適用されるファイル保護の省略時の設定を定義します。

『OpenVMS システム・セキュリティ・ガイド』に説明されている標準の構文規則に従って、コードを指定します。属性を何も指定しない場合は、ファイル保護の現在の省略時の設定が適用されます。

OpenVMS システムでボリュームを使用している場合、この属性は使用されませんが、RSX-11M システムでは、属性を使用してプロセスによるボリュームの使用を制御できます。OpenVMS システムは常に、省略時の設定のファイルの保護を使用します。省略時の設定のファイル保護を変更するには、SET PROTECTION/DEFAULT コマンドを使用します。

/GPT (I64 での省略時の設定)

/NOGPT (Alpha での省略時の設定)

Files-11 オンディスク構造レベル 2 (ODS-2) およびレベル 5 (ODS-5) ディスクのみに適用されます。

注意

/GPT を指定すると、古いバージョンの OpenVMS が動作しているシステムでは、ディスクをマウントできない場合があります。

/GPT を指定すると、システム・ファイル[000000]GPT.SYS が作成されます。GPT.SYS には、I64 コンソール・ソフトウェアが必要とするパーティション/ブート情報が格納されています (GPT は GUID Partition Table の略で、GUID は Global Unique Identifier の略です)。

BACKUP ユーティリティは GPT.SYS を認識し、保存/復元操作の際に、その内容を保持します。

/NOGPT を指定すると、バージョン 8.2 より前の VBN レイアウトの[000000]INDEXF.SYS が使用されます。VBN レイアウトの説明は、『Guide to OpenVMS File Applications』および Kirby McCoy 著の『VMS File System Internals』(ISBN 1-55558-056-4, 1990) を参照してください。

/GROUP

/NOSHARE 修飾子とともに使用して、グループ・ボリュームを作成します。グループ・ボリュームは、システム(S)、オーナー(O)、およびグループ(G)の各ユーザがアクセスできます。保護は、(S:RWCD,O:RWCD,G:RWCD,W) です。

ボリューム所有者の利用者識別コード (UIC) は、ユーザのグループ番号とメンバ番号 0 を省略時の設定にします。

/HEADERS=ヘッダ数

ディスク・ボリュームに対して、索引ファイル用に割り当てるファイル・ヘッダ数を指定します。指定できる最小値および省略時の設定の値は 16 です。最大値は、/MAXIMUM_FILES 修飾子を使用して設定する値です。ただし、/LIMIT を指定し、/HEADERS や /MAXIMUM_FILES には値を指定しなかった場合、以下の省略時の設定が適用されます。

- /MAXIMUM_FILES: 16711679 ファイル
- /HEADERS: 現在の装置の MAXBLOCK (F\$GETDVI 項目コード) のサイズの 0.5 パーセント

たとえば 33GB のディスクでは、省略時に事前に割り当てられるヘッダ・ブロック数は、約 355000 になります。

/HEADERS は、多くのファイルを作成した際、ファイル・ヘッダの領域の割り当ての効率を上げるのに有用です。この修飾子を指定しない場合、ファイル・システムは、ボリュームの新しいヘッダに必要なスペースを動的に割り当てます。

注意

/HEADER 修飾子の省略時の設定の値は通常、ODS-2 および ODS-5 ディスクには不十分です。性能を向上させて SYSTEM-F-HEADERFULL エラーを回避するには、ディスクにあるおおよそのファイル数を予想して、その値に設定してください。ただし、この値を極端に多く見積もると、ディスク領域を浪費する結果になります。

/HEADER 修飾子は、INDEXF.SYS に最初に割り当てられるヘッダの領域の量を制御します。ディスク上の各ファイルにはファイル・ヘッダが少なくとも 1 つ必要で、ヘッダはそれぞれ INDEXF.SYS 内の 1 つのブロックを占有します。多数のアクセス制御リスト・エントリ (ACE) を持つファイル、または非常に細かく断片化されているファイルには、2 つ以上のヘッダを使用することもあります。

省略時の設定の値の 16 は、INDEXF.SYS が拡張される前に、作成される 10 未満のファイル用の空間を確保するためのものです。それを考慮した上で、ディスクに作成するファイルの合計数を見積もって、この修飾子に指定してください。これによってディスク・アクセス性能が向上します。この値を多く見積もると、ディスク領域を浪費する結果になります。この値は、ボリュームを再初期化しないと変更できません。

INDEXF.SYS を拡張できる回数は制限されています。ヘッダのマップ領域 (検索ポインタが格納されている場所) がいっぱいになると、ファイルの作成は失敗し、"SYSTEM-W-HEADERFULL" というメッセージが表示されます。

/HIGHWATER (省略時の設定)

/NOHIGHWATER

Files-11 オンディスク構造レベル 2 (ODS-2) ディスクおよびレベル 5 (ODS-5) ディスクにのみ適用されます。

ファイルのハイウォーター・マーク (FHM) のボリューム属性を設定します。これによって、まだ何も書き込まれていないデータについては、読み込みができないことを保証します。磁気テープに対して、/NOHIGHWATER 修飾子は指定できません。

/NOHIGHWATER 修飾子は、ディスク・ボリュームの FHM を無効にします。

/HOMEBLOCKS=オプション

Files-11 オンディスク構造レベル 2 (ODS-2) ディスクおよびレベル 5 (ODS-5) ディスクにのみ適用されます。

ボリュームのホームブロック、およびホームブロックのコピーを、ディスク上のどこに置くかを指定します。オプションには、次のいずれか 1 つを指定します。

- GEOMETRY

ディスク・ブロックの失敗に備えて、ホームブロックをディスク上の複数の位置に置きます。位置は、ディスクの物理構造により異なります。

- FIXED (省略時の設定)

ホームブロックを、ディスク上の複数の特定の位置に置きます。その位置は、ディスクの物理構造とは関係ありません。そのため、接続しているコントローラのタイプにより物理構造がさまざまに異なるディスクにも対応できます。

- CONTIGUOUS

ホームブロックを、ディスクの開始点から連続的に置きます。

/INDEX=BEGINNING 修飾子と同時に指定すると、コンテナ・ファイル・システムがディスク上の連続領域を最大にできるようになるため、たとえばデータベースのようなサイズの大きいファイルをそのまま 1 つ保存できます。

/INDEX=位置

ボリュームのディレクトリ構造に関する索引ファイルの記憶位置を指定します。指定できる位置は、次のとおりです。

BEGINNING	ボリュームの先頭。
MIDDLE	ボリュームの中間 (省略時の設定)。
END	ボリュームの終わり。
BLOCK:n	論理ブロックの先頭をnで指定します。

/INTERCHANGE

磁気テープを、他ベンダーの環境でも使用することを指定します。/INTERCHANGE 修飾子は、ANSI VOL2 ラベルを省略します。OpenVMS 環境では、ANSI VOL2 ラベルには OpenVMS 固有のセキュリティ属性が含まれています。

/INTERCHANGE 修飾子、磁気テープのラベリング、およびテープのインターチェンジについての詳細は、『OpenVMS システム管理者マニュアル(上巻)』を参照してください。

/LABEL=オプション

オプションで指定したように、磁気テープ・ボリューム・ラベルの特性を定義します。指定できるオプションを次に示します。

- OWNER_IDENTIFIER:“(14 の ANSI 文字)”

ボリューム・ラベルの所有者識別子フィールドを指定できます。指定されるフィールドには、14 文字までの ANSI 文字を使用できます。

- VOLUME_ACCESSIBILITY:“文字”

ANSI 磁気テープ上にある OpenVMS ANSI ボリューム・ラベル VOL 1 の、ボリューム・アクセス可能フィールドに書き込まれる文字を指定します。任意の有効な ANSI a 文字を指定できます。数字、大文字、および次の英数字以外の文字も指定できます。

! " % ' () * + , - . / : ; < = > ?

省略時の設定では、OpenVMS オペレーティング・システムは、次の方法でこのフィールドをチェックするルーチンを提供します。

- 磁気テープが ANSI 規格の Version 3 に対応するバージョンの OpenVMS オペレーティング・システムで作成されている場合は、ASCII スペース以外の文字をすべて上書きする必要があります。
- 保護が指定されていて、磁気テープが Version 3 以降の ANSI 規格に対応している場合は、ASCII 1 以外の文字をすべて上書きする必要があります。

省略時の設定以外の任意の文字を指定する場合、磁気テープにアクセスするためには、INITIALIZE および MOUNT コマンドに/OVERRIDE=ACCESSIBILITY 修飾子を指定しなければなりません。

/LIMIT[=n]

Files-11 オンディスク構造レベル 2 (ODS-2) およびレベル 5 (ODS-5) ディスクのみに適用されます。

ボリュームの拡張に伴い、ボリュームを初期化する必要があることを指定します。nは、ボリュームの最大拡張可能数をブロック単位で定義します。値を指定しない場合は、最大拡張可能数が設定されます。

ボリュームの最大値は、/CLUSTER_SIZE に指定される値によって異なります。

/CLUSTER_SIZE に 8 以上を指定 1TB の拡張が設定されます。

/CLUSTER_SIZE に 8 未満を指定 拡張可能な上限として 65535*4096*クラスタ値が設定されます。これは、ビットマップの最大サイズが 65535 ブロックであるためです。

ボリュームの拡張についての詳細は、『Volume Shadowing for OpenVMS 説明書』を参照してください。

拡張可能な最小値は、次の値のうち最も大きい値です。

- /LIMIT で指定された値
- 物理ディスクのサイズ
- 256 ブロックの BITMAP.SYS ファイルから得られるサイズ (1 ブロックあたり 256 × 4096 ビットにディスクのクラスタ値を乗じた値)

可能な最小値より小さい値が入力されると、その値は最小値まで増加されます。この値は、SHOW DEVICE/FULL コマンドの出力の中で「Expansion Size Limit (拡張下限サイズ)」としてブロック単位で表示されます。

注意

/LIMIT を指定し、以下のパラメータに対して明示的に値を設定しない場合、これらのパラメータに対する省略時の設定は、以下のとおりです。

- /CLUSTER_SIZE: 8
- /MAXIMUM_FILES: 16711679 ファイル

- /HEADERS: 現在の装置の MAXBLOCK (F\$GETDVI 項目コード) のサイズの 0.5 パーセント

たとえば 33GB のディスクでは、省略時の指定で事前に割り当てられるヘッダ・ブロック数は、約 355000 になります。

/MAXIMUM_FILES=n

ボリュームが持つことのできるファイルの最大数を設定します。/MAXIMUM_FILES 修飾子は、省略時の設定の値を上書きします。省略時の設定の値は、次の式から計算できます。

$$(\text{ボリュームのブロック数}) / ((\text{クラスタの要素の数} + 1) * 2)$$

注意

/LIMIT を指定し、/MAXIMUM_FILES に対して値を設定しないと、省略時の設定は 16711679 ファイルになります。

ボリュームに指定できる最大サイズは、次の式から計算できます。

$$(\text{ボリュームのブロック数}) / (\text{クラスタの要素の数} + 1)$$

最小値は 0 です。最大値を変更するためには、ボリュームを再初期化する必要があります。

注意

/MAXIMUM_FILES 修飾子は、ボリューム上に新しいファイル・ヘッダ用のスペースを予約したり、作成したりしません。ファイル・システムは、新しいヘッダに必要なスペースを動的に割り当てます。

/MEDIA_FORMAT=[NO]COMPACTION

データの圧縮をサポートする装置で、データ・レコードの自動圧縮と自動ブロック化を行うかどうかを制御します。データの圧縮とレコードのブロック化を行うと、1 つのテープ・カートリッジに格納できるデータ量が増えます。

あるカートリッジでデータの圧縮や非圧縮を選択すると、カートリッジ全体にその指定が適用されます。

VAX での制限

SCSI テープでは、圧縮を行えるのは、テープの接続の際にローカル SCSI バスを使用している場合に限られます。VAX が TMSCP クライアントである場合、またはテープが HSJ コントローラに常駐している場合には、圧縮を行えません。

/OVERRIDE=(オプション[,...])

重ね書きされないように保護された磁気テープ・ボリュームのデータを無視するように INITIALIZE コマンドに要求します。次のオプションの 1 つまたは複数を指定できます。

ACCESSIBILITY	磁気テープ専用。初期化しようとしているボリュームの場合に、このオプションは、ボリュームの Accessibility フィールドにある任意の文字を上書きします。このオプションの使用は、インストールによって定義されます。つまり、各インストールには、磁気テープ・ファイル・システムがこのフィールドの処理に使うルーチンを指定するオプションがあります。省略時の設定では、OpenVMS は、次の方法でこのフィールドをチェックするルーチンを提供します。磁気テープが ANSI 規格の Version 3 に対応するバージョンの OpenVMS で作成されている場合は、このオプションを使用して、ASCII スペース以外の文字をすべて上書きする必要があります。保護が指定されていて、磁気テープが Version 3 以降の ANSI 規格に対応している場合は、このオプションを使用して、ASCII 1 以外の文字をすべて上書きする必要があります。ACCESSIBILITY オプションを使用するには、VOLPRO 特権を持っているか、またはそのボリュームの所有者でなければなりません。
EXPIRATION	磁気テープ専用。まだ満了日に達していないテープに書き込むことができます。このオプションを実行する必要があるのは、VAX/VMSバージョン 4.0 より前の弊社のオペレーティング・システムで作成され、ボリュームの所有者識別子フィールドに D%形式を指定している磁気テープです。ボリューム保護を上書きするには、VOLPRO 特権を持っているか、または INITIALIZE コマンドを実行している UIC が、そのボリュームに書き込まれている UIC と一致している必要があります。
OWNER_IDENTIFIER	ボリューム・ラベルの所有者識別子フィールドの処理を上書きできます。

オプションを 1 つだけ指定する場合は、括弧を省略できます。

/PROTECTION 修飾子を指定して初期化されたボリュームを再初期化するためには、INITIALIZE コマンドを実行している UIC が、そのボリュームに書き込まれている UIC と一致しているか、または VOLPRO 特権を持っている必要があります。

制御アクセス権を持っている場合は、/PROTECTION を指定して、初期化されたボリュームを再初期化できます。

/OWNER_UIC=uic

ボリュームの所有者の利用者識別コード (UIC) を指定します。省略時の設定は、使用しているユーザの省略時の設定の UIC です。『OpenVMS システム・セキュリティ・ガイド』に説明されている標準の UIC 形式に従って、UIC を指定します。

磁気テープの場合は、磁気テープに保護が指定されないと、UIC が書き込まれません。保護が指定されて所有者 UIC が指定されない場合は、現在実行中の UIC にボリュームの所有権が割り当てられます。

/PROTECTION=(所有権[: アクセス][,...])

ボリュームに指定した保護を適用します。

- 所有権パラメータには、システム(S)、オーナ(O)、グループ(G)、ワールド(W)のいずれか1つを指定します。
- アクセスパラメータには、読み込み(R)、書き込み(W)、作成(C)、削除(D)のいずれかを指定します。

省略時の設定は、ユーザの省略時の設定の保護です。/GROUP、/SHARE、および/SYSTEM 修飾子を使用して、ディスク・ボリュームの保護を設定することもできます。

磁気テープの場合は、ボリューム・ラベルの OpenVMS 固有の部分に保護コードが書き込まれます。システムは、読み込み(R)と書き込み(W)のアクセス制約だけを適用します。作成(C)および削除(D)のアクセスを指定しても無効です。さらに、システムとオーナには、指定した保護コードにかかわらず、磁気テープに対する読み込み(R)と書き込み(W)のアクセスが与えられます。

保護コードの指定についての詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。属性を指定しない場合は、現在の省略時の設定の保護が適用されます。

ディスク・ボリューム全体に対する保護コードでは、E(実行)は作成を意味します。

/SHADOW=(device_name_1, device_name_2, device_name_3) label (Alpha/I64 のみ)

将来のシャドウ・セットの複数のメンバを初期化します。この方法で複数のメンバを初期化しておくと、後でシャドウ・セットを作成するときにフル・コピーを行う必要がなくなります。

INITIALIZE コマンドで/SHADOW および/ERASE 修飾子を指定すると、以下の操作が実行されます。

- 1回のコマンドで6つまでの装置をフォーマットできます。フォーマット終了時に、任意の3つの装置を新しいホスト・ベースのシャドウ・セットのメンバとしてマウントすることができます。
- 各ボリュームにラベルを書き込みます。
- システム・ファイルを除いたデバイスからのすべての情報を削除し、各デバイスと同一のファイル構造情報は残します。ディスクの以前の内容はすべて失われます。

/ERASE 修飾子の使用を強くお勧めします。/ERASE 修飾子を使用することで、マージ操作がかなり削減されます。ただし、/ERASE の使用には2つの副作用があり、ボリューム・シャドウイングを行う際には十分な検討が必要です。具体的には、ERASE ボリューム属性の設定と、/ERASE を使用してボリュームを初期化するための所要時間です。

/ERASE とともに /SHADOW を指定すると、ディスクが順次に消去されるため、コマンドの実行にかかる時間が 2 倍または 3 倍程度かかります。ディスクのサイズが大きい場合には、複数のディスクに対して同時に INITIALIZE/ERASE コマンドを (/SHADOW を使用せずに) 実行してディスクを消去するようにしてください。これらのコマンドがすべて実行されたら、INITIALIZE/SHADOW コマンドを (/ERASE を使用せずに) 実行してください。

一度 /ERASE および /SHADOW を使用して装置を初期化すると、これらの装置を最大で 3 つ、新しいホスト・ベースのシャドウ・セットのメンバとしてマウントできるようになります。

既存のシャドウ・セットに追加するディスクを初期化するときは、INITIALIZE /SHADOW コマンドを使用しないでください。これを使用しても何の利点もないからです。

ボリューム・シャドウイングの詳細については、『Volume Shadowing for OpenVMS 説明書』を参照してください。

/SHARE (省略時の設定)

/NOSHARE

すべての複数のユーザ(システム、オーナー、グループ、ワールド)によって、すべての種類のアクセスを許可します。/NOSHARE 修飾子は、(/GROUP 修飾子がともに指定されていない限り) グループ、およびワールドのユーザのプロセスへのアクセスを拒否します。

/SIZE=*n*

磁気ディスクに /SIZE=*n* を指定した場合、*n* は、論理ボリュームのサイズ、つまりファイル・システムのために使用可能な領域をブロック単位で指定します。このため、ファイル・システムのサイズを物理ボリューム・サイズよりも小さくして、ディスクを初期化することができます。これは、このディスクとサイズがより小さい物理ディスクとを使用してシャドウ・セットを作成する場合に有用です。*n* の値は、SHOW DEVICE/FULL コマンドの出力の中で「Expansion Size Limit (拡張下限サイズ)」としてブロック単位で表示されます。

DECram ディスクの場合、/SIZE は、使用可能なメモリから割り当てられるディスク(装置タイプ DT\$_RAM_DISK) のサイズを指定します。装置のサイズは、ディスクの初期化の際に設定されます。

領域の割り当てを解除するには、/SIZE=0 を指定します。DECram ディスクのために割り当てられていたすべてのリソースは、システムに戻されます。

n は、VAX システムあるいは V2.3 よりも前のバージョンの DECram のいずれかでは、524,280 ブロックを超えることができないことに注意してください。Alpha システムで動作している DECram V2.3 では、67,108,864 ブロック (32GB) までをサポートします。

/STRUCTURE=レベル

ボリュームをフォーマットする場合に、Files-11オンディスク構造レベル1、レベル2 (省略時の設定)、レベル5のいずれの形式を使用するかを指定します。

構造レベル1では、修飾子/DATA_CHECK および/CLUSTER_SIZE について互換性がありません。構造レベル1のディスクの省略時の設定の保護では、システム、オーナー、およびグループにはフル・アクセス、他のすべてのユーザには読み込み(R)アクセスを許可します。

Alpha システムは ODS-1 のディスクをサポートしていないため、Alpha システムで1を指定すると、エラーが発生します。VAX システムは ODS-5 のディスクをサポートしていないため、VAX システムで5を指定すると、エラーが発生します。

ODS-5 のディスクについての詳細は、『OpenVMS システム管理者マニュアル(上巻)』を参照してください。

/SYSTEM

システム UIC または SYSPRV(システムに関する特権) 特権が必要です。

システム・ボリュームを定義します。所有者 UIC の省略時の設定は、[1,1]です。省略時の設定の保護では、システム・プロセスだけが最上位ディレクトリを作成できる点を除いて、すべての所有権カテゴリにフル・アクセスを許可します。

/USER_NAME=名前

ボリュームに対応するユーザ名を指定します。ここで指定するユーザ名は1～12文字の英数字でなければなりません。省略時の設定は、使用しているユーザのユーザ名です。

/VERIFIED

/NOVERIFIED

ディスクに不良ブロック・データがあるかどうかを指定します。/NOVERIFIED 修飾子を使用すると、ディスクの不良ブロック・データは無視されます。省略時の設定は、4096 以上のブロックを持つディスクには/VERIFIED 修飾子、4096 未満のブロックを持つディスクには/NOVERIFIED 修飾子です。

/VOLUME_CHARACTERISTICS=([NO]HARDLINKS, [NO]ACCESS_DATES[=デルタ時間])

Files-11オンディスク構造レベル5 (ODS-5) ディスクのみに適用されます。

ODS-5 ボリューム上でのハードリンクおよびアクセス日の自動更新を、可能または不可能にします。

デルタ時間の省略時の設定は1秒で、POSIX のst_atimeで必要とされる「EPOCH からの経過秒数」時間インタフェースに準拠するよう選択された値です。サイトでは、より値の大きいデルタ時間を選択して、1秒単位の粒度が不要になった場合にもオーバヘッドを削減できます。

NOACCESS_DATES オプションが影響を与えるのは、コマンドを実行したノードだけです。その他のノードは、次にボリュームがマウントされるまでは、この変化の影響を受けません。

詳細は、『Guide to OpenVMS File Applications』を参照してください。

/WINDOWS=n

ファイル・ウィンドウに割り当てられるマッピング・ポインタ(ファイル内のデータへのアクセスに使用)の数を指定します。7 ~ 80 の整数の値を指定します。省略時の設定の値は7です。

例

1. \$ INITIALIZE/USER_NAME=CPA \$FLOPPY1 ACCOUNTS

\$FLOPPY1 のボリュームを初期化して、そのボリュームに ACCOUNT というラベルをつけ、さらに CPA というユーザ名を指定します。

2. \$ ALLOCATE DMA2: TEMP
 _DMA2: ALLOCATED
 \$ INITIALIZE TEMP: BACK_UP_FILE
 \$ MOUNT TEMP: BACK_UP_FILE
 %MOUNT-I-MOUNTED, BACK_UP_FILE mounted on _DMA2:
 \$ CREATE/DIRECTORY TEMP:[GOLDSTEIN]

この一連のコマンドは、RK06/RK07ボリュームを初期化する方法を示します。まず装置が割り当てられて、他のユーザはアクセスできないようになります。次に、ボリュームが装置に物理的にマウントされ、INITIALIZE コマンドによって初期化されます。ボリュームが初期化された後、MOUNT コマンドによりファイル構造が使用できるようになります。ボリュームにファイルを配置するには、その前に CREATE/DIRECTORY コマンドを使用して、ディレクトリを作成しなければなりません。

3. \$ ALLOCATE MT:
 _MTB1: ALLOCATED
 \$ INITIALIZE MTB1: SOURCE
 \$ MOUNT MTB1: SOURCE
 %MOUNT-I-MOUNTED, SOURCE mounted on _MTB1:
 \$ COPY *.FOR MTB1:
 \$ DIRECTORY MTB1:
 .
 .
 .
 \$ DISMOUNT MTB1:

上記のコマンドは、磁気テープを初期化するのに必要な手順を示しています。ドライブを割り当てたら、そのドライブに磁気テープを挿入し、INITIALIZE コマンドを使用してそこにラベル SOURCE を書き込みます。次に、MOUNT コマンドで磁気テープをマウントして、ファイルの書き込みができますようにします。

4. \$ BACKUP filespec MUA0: ... /MEDIA_FORMAT=NOCMPACTION-
_\$/REWIND

この例では、圧縮とレコードのブロック化を禁止して、BACKUP テープを作成します。

5. \$ INITIALIZE/ERASE/SHADOW=(\$4\$DKA1300, \$4\$DKA1301) NONVOLATILE

```
$MOUN/SYS DSA42 /SHAD=( $4$DKA1300 , $4$DKA1301 ) NONVOLATILE
%MOUNT-I-MOUNTED, NONVOLATILE MOUNTED ON _DSA42:
%MOUNT-I-SHDWMEMSUCC, _$4$DKA1300: (WILD3) IS NOW A VALID MEMBER OF THE SHADOW SET
%MOUNT-I-SHDWMEMSUCC, _$4$DKA1301: (WILD4) IS NOW A VALID MEMBER OF THE SHADOW SET
$SHO DEV DSA42:
```

DEVICE NAME	DEVICE STATUS	ERROR COUNT	VOLUME LABEL	FREE BLOCKS	TRANS COUNT	MNT CNT
DSA42:	MOUNTED	0	NONVOLATILE	5799600	1	1
\$4\$DKA1300: (WILD3)	SHADOWSETMEMBER	0	(MEMBER OF DSA42:)			
\$4\$DKA1301: (WILD4)	SHADOWSETMEMBER	0	(MEMBER OF DSA42:)			

この例は、INITIALIZE/ERASE/SHADOW コマンドの正しい使い方を示しています。同じ行で複数の装置を指定している点に注意してください。

INITIALIZE/QUEUE

キューの作成または初期化を行います。このコマンドを使用してキューを作成し、名前とオプションを割り当てます。バッチ・キューの作成には、/BATCH 修飾子が必要です。

キューの作成には OPER (オペレータ) 特権、キューの変更には管理 (M) アクセス権が必要です。

フォーマット

INITIALIZE/QUEUE キュー名[:]

パラメータ

キュー名[:]

実行キューまたは汎用キューの名前を指定します。キュー名には、1 ~ 31 の文字列を使用します。文字列には、大文字と小文字の英文字、数字、ドル記号(\$)、アンダスコア(_)を使用できます。ただし、英文字を少なくとも1つ含めなければなりません。

説明

キューを作成するか、または終了している既存のキューのオプションを変更するには、INITIALIZE/QUEUE コマンドを使用します。

システムまたは OpenVMS Cluster システムをセットアップするときに、必要な INITIALIZE/QUEUE コマンドを実行して、出力キューとバッチ・キューを作成します。後から、必要に応じて INITIALIZE/QUEUE コマンドを使用して、キューを作成して追加することができます。INITIALIZE/QUEUE コマンドでキューを作成すると、キューについての情報がキュー・データベースに格納されます。

キューの作成と起動を同時に行う場合は、INITIALIZE/QUEUE/START コマンドを使用します。キューの作成だけを行い、別の時にそれを起動する場合は、INITIALIZE/QUEUE コマンドだけを実行します。後から START/QUEUE コマンドを入力すると、キュー起動できます。

INITIALIZE/QUEUE、START/QUEUE、および SET QUEUE コマンドによってキュー・オプションを変更できます。キュー・オプションを変更すると、キュー・データベース内のキューに関する情報が更新されます。

INITIALIZE および START コマンドは、終了したキューに対してのみ使用できます。動作中のキューのオプションを変更するには、SET QUEUE コマンドを使用します。SET QUEUE コマンドで変更できないキュー・オプションを変更するには、次の手順に従ってください。

1. STOP/QUEUE/NEXT コマンドでキューを終了します。
2. START/QUEUE または INITIALIZE/QUEUE/START コマンドを使用してキューを再起動し、必要なオプションに合わせて適切な修飾子を指定します。
指定しない修飾子はすべて、以前にキューの初期化、起動、または設定を行ったときのままです。

既存のキューを初期化しても、そのキューに登録されている現在処理中のジョブは削除されません。新しい INITIALIZE/QUEUE コマンドによって設定された新しいキューの設定はすべて、キューで待機中のすべてのジョブ、またはこれからキューに登録されるジョブに作用します。キューの終了時に、キューで実行中のジョブはすべて、古い設定で実行されます。

次の修飾子は、汎用キューと実行キューで使用できます。

```
/OWNER_UIC
/PROTECTION
/[NO]RETAIN
/[NO]START
/NAME_OF_MANAGER
```

次の修飾子は、すべての実行キューで使用できます。

```
/AUTOSTART_ON
/BASE_PRIORITY
/[NO]CHARACTERISTICS
/[NO]ENABLE_GENERIC
/[NO]NO_INITIAL_FF
/ON
/WSDEFAULT
/WSEXTENT
/WSQUOTA
```

次の修飾子は、バッチ実行キューにのみ使用できます。

```
/CPUDEFAULT
/CPUMAXIMUM
/[NO]DISABLE_SWAPPING
/JOB_LIMIT
```

次の修飾子は、プリンタ、端末、またはサーバの各実行キューにのみ指定できます。

```
/[NO]BLOCK_LIMIT
```

```

/[NO]DEFAULT
/[FORM_MOUNTED
/[NO]LIBRARY
/[NO]PROCESSOR
/[NO]RECORD_BLOCKING
/[NO]SEPARATE

```

キューのタイプ

キューには、いくつかのタイプがあります。キューは、汎用キューと実行キューの2つに分類できます。実行キューにジョブが送られると、ジョブはそのキューで実行され、汎用キューでは実行されません。汎用キューは、実行キューで実行されるジョブを保留します。

汎用キュー 次に、汎用キューのいくつかのタイプを示します。

- 汎用バッチ・キュー：バッチ実行キューで実行するバッチ・ジョブを保留します。
- 汎用出力キュー：出力キューで実行するジョブを保留します。汎用出力キューには、次の3つのタイプがあります。
 - 汎用プリント・キュー：出力実行キューでプリントするプリント・ジョブを保留します。
 - 汎用サーバ・キュー：出力実行キューで処理するジョブを保留します。
 - 汎用端末キュー：出力実行キューでプリントするプリント・ジョブを保留します。

/GENERIC 修飾子は、キューを汎用キューとして指定します。次のいずれかの方法で、汎用キューがジョブを登録する実行キューを設定します。

- /GENERIC 修飾子を使用してキューのリストを指定すると、汎用キューに割り当てた実行キューに明示的に名前を付けることができます。
- 実行キューの作成時に/ENABLE=GENERIC 修飾子を指定すると、明示的なターゲット・リストを指定しない汎用キューからジョブを受け取る実行キューを指定できます。

実行キューと異なり、汎用キューはシステムがシャットダウンしたりキュー・マネージャが終了しても、自動的に終了しません。したがって、通常は、システムをリブートするたびに汎用キューを再起動する必要はありません。

論理キュー

キューのもう1つのタイプは、論理キューです。論理キューは特別なタイプの汎用キューで、ASSIGN/QUEUE コマンドで指定した実行キューでだけ、ジョブを実行できます。実行キューと論理キューの関係は、DEASSIGN/QUEUE コマンドで割り当てを取り消すまで有効です。

実行キュー 次に、実行キューのいくつかのタイプを示します。

- バッチ実行キュー — バッチ・ジョブを実行します。
- 出力実行キュー — プリント出力ジョブを処理します。出力実行キューには、次の3タイプがあります。
 - プリンタ実行キュー — シンビオントを起動して、プリンタのプリント・ジョブを処理します。
 - サーバ実行キュー — ユーザが作成したシンビオントを起動して、ジョブを処理します。
 - 端末実行キュー — シンビオントを起動して、端末プリンタのプリント・ジョブを処理します。

バッチ実行キューは、バッチ・ジョブを実行します。バッチ・ジョブは、バッチ・プロセスで1つまたは複数のコマンド・プロシージャの実行を要求します。

出力実行キューは、プリント・ジョブを処理します。プリント・ジョブは、シンビオント・プロセスで実行する1つのシンビオントによって、1つまたは複数のファイルの処理を要求します。省略時の設定のシステム・シンビオントは、ハードコピー装置(プリンタまたは端末)でファイルをプリントするように設計されています。ユーザが作成するシンビオントは、プリントまたは他のファイル処理を実行するように設計できます。サーバ・キューは、/PROCESSOR 修飾子で指定したサーバ・シンビオントを使用してジョブを処理します。サーバ・キュー・シンビオントは、ユーザが作成します。

/AUTOSTART_ON 修飾子または/ON 修飾子は、キューを実行キューとして指定して、キューを実行する場所を指定します。

/ON 修飾子を使用すると、キューを起動できる1つのノード(バッチ・キューの場合)、またはノードと装置(出力キューの場合)を指定できます。/ON 修飾子で初期化したキューは、明示的にキューを指定したコマンドで起動する必要があります。

/AUTOSTART_ON 修飾子を使用すると、キューを起動できる1つまたは複数のノード(あるいはノードと装置)を指定できます。キュー・マネージャによってキューのノードが自動起動を許可されている場合は、そのキュー・マネージャによって、/AUTOSTART_ON 修飾子を使用して初期化したキューが自動的に起動します。

キューの自動起動 実行キュー(バッチまたは出力のいずれか)は、自動起動キューとして指定できます。ノードにあるキュー・マネージャの自動起動キューのすべてが1つのコマンドによって起動できるため、自動起動キューでは、長いキューのスタートアップ・プロシージャは必要ありません。

OpenVMS Cluster では、自動起動キューをいくつかのノードのうちの1つのノードで実行するように設定できます。この方法で設定したキューを実行しているノードがクラスタからはずれると、そのキューは、別のノードにフェイルオーバーし、クラスタ内でそのまま使用できます。

/AUTOSTART_ON 修飾子は、実行キューを自動起動キューとして指定します。

修飾子

/AUTOSTART_ON=(ノード::[装置][,...])

キューを自動起動実行キューとして指定し、キューを置くことができるノードまたはノードと装置を指定します。バッチ・キューの場合は、ノードだけ指定できます。

クラスタでは、ノードがキューを要求する順序で、キューが実行できる2つ以上のノード(またはノードと装置)を指定できます。これによって、キューを実行しているノードがクラスタから離れると、別のノードにそのキューをフェイルオーバーできます。

/AUTOSTART_ON 修飾子を使用して INITIALIZE/QUEUE コマンドを実行する場合には、INITIALIZE/QUEUE コマンドで/START 修飾子を指定するか、または START/QUEUE コマンドを実行して、最初にキューの自動起動を有効にしなければなりません。ただし、キューを実行するノードに対して ENABLE AUTOSTART/QUEUES コマンドが実行されるまで、キューはジョブの処理を開始しません。

この修飾子は、/ON または/GENERIC 修飾子と同時に使用できません。ただし、既存のキューを再初期化する場合は、/ON 修飾子を指定して作成または起動してあったキューに対しては、/AUTOSTART_ON 修飾子を指定できます。これを実行すると、/ON 修飾子を無効になり、キューは自動起動キューになります。

自動起動キューについての詳細は、『OpenVMS システム管理者マニュアル(上巻)』のキューについての章を参照してください。

/BASE_PRIORITY=n

バッチ実行キューからジョブが開始されるプロセスの基本優先順位を指定します。省略時の設定では、修飾子を省略すると、ジョブはシステム生成時に DEFPRI によって設定された基本優先順位と同じ優先順位で開始されます(通常は4)。基本優先順位指定子には、10進数で0～15の値を指定します。

この修飾子は、出力実行キューにも指定できます。この場合は、シンビオント・プロセスが作成されるときに、/BASE_PRIORITY 修飾子がシンビオント・プロセスの基本優先順位を設定します。

/BATCH

/NOBATCH (省略時の設定)

バッチ・キューとしての初期化を指定します。既存のキューを再初期化する場合に、/BATCH 修飾子を使用できるのは、キューを最初にバッチ・キューとして作成した場合だけです。

バッチ・キューは、バッチ実行キューまたはバッチ汎用キューのいずれかに分類されます。省略時の設定では、/BATCH 修飾子はバッチ実行キューを初期化します。汎用

バッチ・キューを指定するには、/GENERIC 修飾子を/BATCH 修飾子と同時に使用します。

/BATCH 修飾子と/DEVICE 修飾子は、同時に使用できません。/NOBATCH 修飾子と/NODEVICE 修飾子も、同時に使用できません。

/BLOCK_LIMIT=([下限],上限)
/NOBLOCK_LIMIT (省略時の設定)

出力実行キューで処理できるプリント・ジョブ・サイズを制限します。この修飾子を使用すると、特定のプリンタに対して、決まったサイズのジョブを予約できます。上限パラメータは必ず指定しなければなりません。

下限パラメータは、プリント・ジョブに対してキューが受け付ける最小のブロック数(10 進数)です。下限値より少ないブロック数のプリント・ジョブがキューに登録されると、キューのブロック制限値の下限が変更されるまでジョブは待ち状態になります。ジョブは、キューのブロックの下限値がジョブのブロック数以下に減少された後に処理されます。

上限パラメータは、プリント・ジョブに対してキューが受け付ける最大のブロック数(10 進数)です。この値を超えるプリント・ジョブがキューに登録されている場合、キューのブロック制限値が変更されるまでジョブは待ち状態になります。ジョブは、キューのブロックの上限値がジョブのブロック数以上に増加された後に処理されます。

ジョブの上限値だけを指定する場合は、括弧を省略できます。たとえば、/BLOCK_LIMIT=1000 は、キューの中で 1000 以下のブロックを持つジョブだけが処理されます。ジョブの下限値だけを指定するには、空文字列("")を使用して、上限の指定子を示さなければなりません。たとえば、/BLOCK_LIMIT=(500,"") は、キューの中で 500 以上のブロックを持つジョブはすべて処理されます。下限および上限の両方を指定することもできます。たとえば、/BLOCK_LIMIT= (200,2000) は、キューの中で 200 未満のブロックを持つジョブと、2001 以上のブロックを持つジョブは処理されないことを意味します。

/NOBLOCK_LIMIT 修飾子は、/BLOCK_LIMIT 修飾子を指定してこのキューに設定した前の設定を取り消します。

/CHARACTERISTICS=(属性[,...])
/NOCHARACTERISTICS (省略時の設定)

実行キューでジョブを処理するための属性を、1 つまたは複数指定します。属性を 1 つだけ指定する場合は、括弧を省略できます。ジョブに指定した属性のすべてをキューが備えていない場合、ジョブは待ち状態のままになります。/CHARACTERISTICS 修飾子を指定するたびに、その前に設定された属性はすべて取り消されます。この修飾子で指定した属性だけが、キューに設定されます。

キューの属性は、それぞれのシステムによって異なります。属性パラメータは、0 ~ 127 の値、または DEFINE/CHARACTERISTIC コマンドを使用して定義した属性名になります。

/NOCHARACTERISTICS 修飾子は、/CHARACTERISTICS 修飾子を指定してこのキューに設定していた前の設定をすべて取り消します。

/CLOSE

PRINT または SUBMIT コマンドによって、あるいは REQUEUE 操作の結果として、ジョブがキューに登録されないようにします。ジョブに登録可能するには、/OPEN 修飾子を使用します。キューが新しいジョブ・エントリを受け付けるかどうかは、キューの状態（一時停止、終了、または止められている）とは関係ありません。キューがクローズ状態の場合、実行中のジョブは継続されます。キューですでに待ち状態のジョブは、引き続き実行待ちになります。

/CPUDEFAULT=時間

このバッチ実行キューにあるすべてのジョブに対して、省略時の設定の CPU 時間制限値を定義します。時間には、デルタ時間、0、INFINITE、または NONE（省略時の設定）を指定できます。デルタ時間には、最高 497 日まで指定できます。

キューに時間制限値 CPUMAXIMUM が指定されていない場合や、利用者登録ファイル (UAF) に設定された値が CPU 時間制限値に NONE を指定している場合は、値 0 またはキーワード INFINITE を使用すると、CPU 時間の制限をなくすことができます。NONE を指定すると、UAF または SUBMIT コマンドで指定した値（指定している場合）が CPU 時間値の省略時の設定になります。CPU 時間値は、システム・パラメータ PQL_MCPULM で指定した値以上でなければなりません。この時間は、/CPUMAXIMUM 修飾子を使用して設定した CPU 時間制限を超えることができません。デルタ時間を指定する方法についての詳細は、『OpenVMS ユーザーズ・マニュアル』またはオンライン・ヘルプのトピック Date を参照してください。CPU 時間制限の指定についての詳細は、表 DCLI-1 を参照してください。

/CPUMAXIMUM=時間

バッチ実行キューにあるすべてのジョブの最大 CPU 時間制限値を定義します。時間には、デルタ時間、0、INFINITE、または NONE（省略時の設定）を指定できます。デルタ時間には、最高 497 日まで指定できます。

/CPUMAXIMUM 修飾子は、キューにジョブに登録するすべてのユーザに対して、利用者登録ファイル (UAF) で指定した時間制限を変更します。値 0 またはキーワード INFINITE を使用すると、CPU 時間の制限をなくすことができます。NONE を指定すると、UAF または SUBMIT コマンドで指定した値（指定している場合）が CPU 時間値の省略時の設定になります。CPU 時間値は、システム・パラメータ PQL_MCPULM で指定した値以上でなければなりません。

デルタ時間を指定する方法についての詳細は、『OpenVMS ユーザーズ・マニュアル』またはオンライン・ヘルプのトピック Date を参照してください。CPU 時間制限値を指定する方法についての詳細は、表 DCLI-1 を参照してください。

プロセスの CPU 時間制限値は、システムの UAF でのユーザ・レコードごとに指定します。また、省略時の設定の CPU 時間制限値、指定したキューに存在するすべてのジョブに対する最大 CPU 時間制限値、またはキューの各ジョブに対する省略時の

CPU 時間制限値も指定できます。指定した各値に従って実行される処理と、指定の組み合わせを表 DCLI-1 に示します。

表 DCLI-1 CPU 時間制限値の指定と処理

SUBMIT コマンドで CPU 時間制限値が指定 されている	キューで省略時 の設定の CPU 時 間制限値が指定 されている	キューで CPU 時間制限 値が指定されている	処理
No	No	No	UAF の値を使用します。
Yes	No	No	SUBMIT コマンドと UAF 値のうち、小さい方を使用します。
Yes	Yes	No	SUBMIT コマンドと UAF 値のうち、小さい方を使用します。
Yes	No	Yes	SUBMIT コマンドとキューの最大値のうち、小さい方を使用します。
Yes	Yes	Yes	SUBMIT コマンドとキューの最大値のうち、小さい方を使用します。
No	Yes	Yes	キューの省略時の値と最大値のうち、小さい方を使用します。
No	No	Yes	最大値を使用します。
No	Yes	No	UAF とキューの省略時の値のうち、小さい方を使用します。

/DEFAULT=(オプション[,...])

/NODEFAULT

PRINT コマンドの特定のオプションの省略時の値を設定します。省略時の設定の値は、オプションのリストで指定します。オプションを 1 つだけ指定する場合は、括弧を省略できます。/DEFAULT 修飾子を使用してオプションをキューに設定した後は、PRINT コマンドにそのオプションを指定する必要はありません。PRINT コマンドでこれらのオプションを指定すると、PRINT コマンドを使用して指定した値は、/DEFAULT 修飾子を使用してキューに設定した値を上書きします。

/DEFAULT および/GENERIC 修飾子は、同時に指定できません。

使用可能なオプションは、次のとおりです。

[NO]BURST[=キーワード]

バースト・ページのバーを出力の前にプリントするかどうかを制御します。値 ALL (省略時の値) を指定すると、バースト・ページは、ジョブの各ファイルの前にプリントされます。値 ONE を指定すると、バースト・ページは、ジョブの最初のファイルの前に 1 度プリントされます。

[NO]FEED

ページの最後で、自動的に改ページを行うかどうかを指定します。

[NO]FLAG[=キーワード]	ファイル・フラグ・ページを、出力の前にプリントするかどうかを制御します。値 ALL (省略時の設定の値) を指定すると、フラグ・ページは、ジョブの各ファイルの前にプリントされます。値 ONE を指定すると、フラグ・ページは、ジョブで最初のファイルの前に 1 度プリントされます。
FORM=タイプ	出力実行キューの省略時のプリント形式を指定します。プリント形式を明示的に定義しないでジョブをキューに登録すると、このプリント形式がジョブの処理に使用されます。FORM キーワードによるプリント形式タイプが明示的に指定されていない場合、システムは、プリント形式 DEFAULT をキューに割り当てます。/FORM_MOUNTED=タイプ修飾子の説明も参照してください。
[NO]TRAILER[=キーワード]	出力に続けてファイル・トレーラ・ページをプリントするかどうかを制御します。値 ALL (省略時の設定の値) を指定すると、ジョブの各ファイルの後にファイル・トレーラ・ページがプリントされます。値 ONE を指定すると、トレーラ・ページは、ジョブのファイルの最後に 1 度プリントされます。

ファイルに BURST オプションを指定すると、[NO]FLAG オプションは、ファイルの前にプリントされる 2 つのフラグ・ページに対して、フラグ・ページの追加または削除を行いません。

必須のキュー・オプションの設定についての詳細は、/SEPARATE 修飾子の説明を参照してください。省略時のキュー・オプションの指定についての詳細は、『OpenVMS システム管理者マニュアル』のキューの章を参照してください。

/DESCRIPTION=文字列

/NODESCRIPTION (省略時の設定)

オペレータからのキューについての情報を提供するための、最大 255 文字の文字列を指定します。

文字列に英小文字、ブランク、またはその他の英数字以外 (スペースを含む) の文字列を含める場合は、文字列全体を二重引用符(“ ”)で囲みます。

/NODESCRIPTION 修飾子は、そのキューに関する説明をすべて削除します。

/DEVICE[=オプション]

/NODEVICE

特定のタイプの出力キューを初期化するように指定します。既存のキューを再初期化する場合、出力キューとして作成したキューに対してのみ、/DEVICE 修飾子を使用できます。使用可能なオプションは、次のとおりです。

PRINTER	プリント・キューを示します。
SERVER	サーバ・キューを示します。サーバ・キューは、/PROCESSOR 修飾子に指定した、ユーザが変更したシンビオント、またはユーザが作成したシンビオントによって制御されます。
TERMINAL	端末キューを示します。

キュー・タイプなしで/DEVICE 修飾子を指定すると、省略時の設定により /DEVICE=PRINTER 修飾子を使用されます。

出力キューは、出力実行キューまたは汎用出力キューに分類されます。省略時の設定では、/DEVICE 修飾子によって、出力実行キューが初期化されます。汎用出力キューを指定するには、/DEVICE 修飾子と/GENERIC 修飾子を同時に使用します。

情報用には、/DEVICE 修飾子にキュー・タイプを指定します。出力実行キューが起動すると、キューに対応したシンビオントによって、実際のキュー・タイプが決まります。標準のシンビオントは、装置の特性をチェックして、キューがプリンタ・キューまたは端末キューのいずれであるかマークをつけるよう設定します。通常は、ユーザ変更シンビオントおよびユーザ作成シンビオントは、サーバ・キューとしてキューをマークします。汎用出力キューの装置タイプは、出力実行キューの装置タイプと一致する必要はありません。

/DEVICE 修飾子と/BATCH 修飾子は、同時に使用できません。また、/NODEVICE 修飾子と/NOBATCH 修飾子も、同時に使用できません。

/DISABLE_SWAPPING

/NODISABLE_SWAPPING (省略時の設定)

キューから実行するバッチ・ジョブが、メモリヘスワップ・イン、およびメモリからスワップ・アウトできるかどうかを指定します。

/ENABLE_GENERIC (省略時の設定)

/NOENABLE_GENERIC

/GENERIC 修飾子にキューの名前を指定していない汎用キューに登録されたファイルを、このキューに移してよいかどうかを指定します。詳細は、/GENERIC 修飾子の説明を参照してください。

/FORM_MOUNTED=タイプ

出力実行キュー用にマウントされたプリント形式を指定します。

プリント形式タイプを明示的に指定しない場合、システムは、DEFAULT のプリント形式をキューに割り当てます。

プリント形式を明示的に指定してジョブがキューに登録され、明示的なプリント形式のストックがマウントされたプリント形式のストックと一致しない場合は、このジョブは待ち状態に入ります。そしてその状態は、このキューのマウントされたプリント形式のストックが、ジョブに対応したプリント形式のストックと一致するまで続きます。

プリント形式のタイプを指定するには、DEFINE/FORM コマンドで定義した数値またはプリント形式名を使用します。プリント形式のタイプは、それぞれのシステムによって異なります。/FORM_MOUNTED 修飾子と/GENERIC 修飾子は、同時に使用できません。

/GENERIC[=(キュー名[...])]

/NOGENERIC (省略時の設定)

汎用キューを指定します。このキューに置かれたジョブを互換性のある実行キューへ移動して処理できることも指定します。/GENERIC 修飾子には、ターゲット・キュー

ーとして、既にある実行キューの名前(複数可)を指定することができます。汎用バッチ・キューの場合は、これらのターゲット・キューは、バッチ実行キューでなければなりません。汎用出力キューの場合は、これらのターゲット・キューは、出力実行キューでなければなりませんが、どのようなタイプ(プリンタ、サーバ、または端末)でもかまいません。たとえば、汎用出力キューのターゲットにプリント・キューと端末キューといった種類の違う出力キューを指定したとしても、両方にジョブを転送できます。

/GENERIC 修飾子を使用してターゲット実行キューを指定しない場合、ジョブは、/ENABLE_GENERIC 修飾子で初期化されていて、かつ、汎用キューと同じタイプ(バッチまたは出力)の任意の実行キューに移動できます。

キューを汎用バッチ・キューまたは出力キューとして定義するには、/GENERIC 修飾子と同時に、/BATCH 修飾子または/DEVICE 修飾子を指定します。汎用キューの作成に、/BATCH 修飾子および/DEVICE 修飾子のいずれも指定しないと、省略時の設定により、キューは汎用出力キューになります。

/SEPARATE 修飾子は、/GENERIC 修飾子と同時に使用できません。

/JOB_LIMIT=n

キューから同時に実行できるバッチ・ジョブの数を指定します。1 ~ 255 の値を指定します。省略時の設定の値は 1 です。

/LIBRARY=ファイル名

/NOLIBRARY

装置制御ライブラリのファイル名を指定します。出力実行キューを初期化する場合は、/LIBRARY 修飾子を使用すると、他の装置制御ライブラリを指定できます。省略時の設定のライブラリは、SYS\$LIBRARY:SYSDEVCTL.TLB です。/LIBRARY 修飾子のパラメータには、ファイル名だけを指定することができます。システムは常に、そのファイルが SYS\$LIBRARY に置かれていて、ファイル・タイプが.TLB であると仮定します。

/NAME_OF_MANAGER=名前

キューを制御するキュー・マネージャの名前を示します。いったんキューが作成されると、キュー・マネージャの割り当ては変更できません。

/NAME_OF_MANAGER 修飾子を省略した場合は、省略時の設定の名前の SYS\$QUEUE_MANAGER が使用されます。

INITIALIZE/QUEUE コマンドを使用してキューの変更を行う場合や、そのキューが省略時の設定のキュー・マネージャで制御されていない場合は、/NAME_OF_MANAGER 修飾子に制御キュー・マネージャの名前を指定する必要があります。または、論理名 SYS\$QUEUE_MANAGER を正しいキュー・マネージャとするように定義して、そのキュー・マネージャを、現在処理中のプロセスの省略時の値にできます。

/NO_INITIAL_FF
/NONO_INITIAL_FF (省略時の設定)

キューを起動する時に、プリンタ・デバイスにフォーム・フィードを送るかどうか、指定することができます。初期フォーム・フィードを抑止するには、/NO_INITIAL_FF 修飾子を指定します。

/NONO_INITIAL_FF 修飾子を指定すると、出力装置にフォーム・フィードが送られ、紙の一番上から印刷が開始されます。

/ON=[ノード::]装置[:] (プリンタ、端末、サーバ・キュー)
/ON=ノード:: (バッチ・キュー)

この実行キューを置くノードまたは装置、あるいはその両方を指定します。バッチ実行キューの場合は、ノード名だけを指定できます。出力実行キューの場合は、ノード名と装置名の両方を指定できます。省略時の設定では、キューは、起動されたノードと同じノードで実行されます。省略時の設定の装置パラメータは、キュー名と同じです。

装置に対する IP アドレスとポート番号を、引用符の中に指定することができます。IP アドレスの指定についての詳細は、TCP/IP Services for OpenVMS のマニュアルを参照してください。

このノード名の指定は、OpenVMS Cluster システムで使われます。このノード名の指定は、キューが実行される OpenVMS システムのシステム・パラメータ SCSNODE によって指定されたノード名と一致しなければなりません。

/ON 修飾子は、/AUTOSTART_ON 修飾子または/GENERIC 修飾子と同時に使用できません。ただし、既存のキューを再初期化する場合、/AUTOSTART_ON 修飾子を使用して前に作成または起動していたキューに対しては、/ON 修飾子を指定できます。これを行うと、/AUTOSTART_ON オプションが変更され、キューが非自動起動キューになります。

/OPEN (省略時の設定)

PRINT または SUBMIT コマンドによって、あるいは REQUEUE 操作の結果として、ジョブをキューに登録できます。ジョブがキューに登録されないようにするには、/CLOSE 修飾子を使用します。キューが新しいジョブ・エントリを受け付けるかどうかは、キューの状態（一時停止、終了、または止められている）とは関係ありません。

/OWNER_UIC=uic

キューの利用者識別コード (UIC) を変更します。『OpenVMS システム・セキュリティ・ガイド』の説明に従って、標準形式を使用して UIC を指定します。省略時の設定の UIC は、[1,4]です。

/PROCESSOR=ファイル名
/NOPROCESSOR

出力実行キューに標準以外のプリント・シンピオントを指定できます。

/PROCESSOR 修飾子のパラメータには、任意の有効なファイル名を指定するこ

とができます。システムは、装置およびディレクトリ名の SYS\$SYSTEM とファイル・タイプ.EXE を追加します。出力キューにこの修飾子を使用すると、実行するシンビオント・イメージは SYS\$SYSTEM:filename.EXE になります。

省略時の設定では、SYS\$SYSTEM:PRTSMB.EXE が、出力実行キューに対応したシンビオント・イメージです。

/NOPROCESSOR 修飾子を指定すると、/PROCESSOR 修飾子を使用して設定した前の設定はすべて取り消され、SYS\$SYSTEM:PRTSMB.EXE が使用されます。

/PROTECTION=(所有権[: アクセス], ...)

キューの保護を指定します。

- 所有権パラメータには、システム(S)、オーナ(O)、グループ(G)、ワールド(W)のいずれか1つを指定します。
- アクセス・パラメータには、読み込み(R)、キューに登録(S)、管理(M)、または削除(D)を指定します。

アクセス・パラメータの指定がないときは、アクセスできないことを意味します。省略時の設定の保護は、(SYSTEM:M, OWNER:D, GROUP:R, WORLD:S)です。保護コードを1つだけ指定する場合は、括弧を省略できます。保護コードの指定についての詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。UIC に基づいた保護によるキュー操作の制御についての詳細は、『OpenVMS システム管理者マニュアル』のキューの章を参照してください。

/RAD=n (Alpha/I64 のみ)

キューに割り当てられたバッチ・ジョブを実行する RAD 番号を指定します。RAD 値の有効な値は、0 から \$GETSYI 項目コード、SYI\$_RAD_MAX_RADS によって返される値までの正の整数です。

AlphaServer GS シリーズ・システム上でのみサポートされます。

/RECORD_BLOCKING (省略時の設定)

/NORECORD_BLOCKING

シンビオントが、出力装置へ転送する出力レコード連結(ブロック化)できるかどうかを決めます。/NORECORD_BLOCKING 修飾子を指定すると、シンビオントは、入出力要求ごとに各レコードを出力装置に送信します。標準の OpenVMS プリント・シンビオントの場合は、レコードをブロック化すれば、シングル・レコード・モードよりも性能を大幅に向上させることができます。

/RETAIN[=オプション]

/NORETAIN (省略時の設定)

実行後のジョブを保持状態でキューに保留します。/NORETAIN 修飾子を使用すると、そのキューを省略時の設定に再設定できます。使用可能なオプションは、次のとおりです。

ALL (省略時の設定)	実行後、キューのジョブをすべて保留します。
ERROR	正常に終了できなかったジョブだけを保留します。

PRINT, SUBMIT, または SET ENTRY コマンドに/RETAIN 修飾子を指定すると、ジョブに対するジョブ保持オプションを要求できます。ただし、キューに指定したジョブ保持オプションは、そのキューに要求されたすべてのジョブの保持オプションを無効にします。

/SCHEDULE=SIZE (省略時の設定)
/SCHEDULE=NOSIZE

出力キューで待ち状態にあるジョブが、ジョブの大きさに基づいてスケジュールされてプリントされるかどうかを指定します。省略時の設定の修飾子/SCHEDULE=SIZE が有効な場合は、小さいジョブから大きいジョブの順にプリントされます。

/SCHEDULE=NOSIZE 修飾子が有効な場合には、サイズに従ったジョブのスケジュール化は行われません。

キューのいずれかに待ち状態のジョブがあるときにこのコマンドを実行すると、その後のジョブの順序は予測できなくなります。

/SEPARATE=(オプション[,...])
/NOSEPARATE (省略時の設定)

出力実行キューに必須のキュー・オプション、またはジョブ区切りオプションを指定します。ジョブ区切りオプションは、PRINT コマンドでは変更できません。

/SEPARATE 修飾子と/GENERIC 修飾子は、同時に使用できません。

ジョブの区切りに関するオプションは次のとおりです。

[NO]BURST	バースト・ページのバーを各ジョブの先頭にプリントするかどうかを指定します。
[NO]FLAG	ジョブのフラグ・ページを各ジョブの先頭にプリントするかどうかを指定します。
[NO]TRAILER	ジョブのトレーラ・ページを各ジョブの末尾にプリントするかどうかを指定します。
[NO]RESET=(モジュール[,...])	このキューで使用するジョブ・リセット・シーケンスを装置制御ライブラリ (SYS\$LIBRARY:SYSDEVCTL) の中から指定します。指定されたモジュールが装置制御ライブラリから取り出されて、各ジョブの終わりに装置をリセットします。リセット動作はファイル・トレーラの後とジョブ・コントローラの前に行われます。したがって、ジョブの区切りのためのページはすべて、装置がリセット状態においてプリントされることになります。

/SEPARATE=BURST 修飾子を指定すると、[NO]FLAG オプションは、ジョブの前にプリントされる 2 つのフラグ・ページに対して、影響を及ぼしません。

変更できるキュー・オプションの設定についての詳細は、/DEFAULT 修飾子の説明を参照してください。

必須のキュー・オプションの指定についての詳細は、『OpenVMS システム管理者マニュアル』のキューの章を参照してください。

/START

/NOSTART (省略時の設定)

INITIALIZE/QUEUE コマンドでキューを初期化し、そのキューを起動します。

自動起動キューの場合、この修飾子は、キューを自動起動として動かします。キューを動作できるいずれかのノードで、ENABLE AUTOSTART/QUEUES コマンドを使用して自動起動を有効にすると、キューはジョブの処理を開始します。

/WSDEFAULT=n

バッチ・ジョブに対して、ワーキング・セットの省略時の設定の値、つまりジョブが使用できる物理ページ数の省略時の設定の値を定義します。

この修飾子を指定すると、UAF に指定された値は使用されず、修飾子の値が使用されます。

Alpha システムの場合はページレット単位 (1 ページレットは 512 バイト) で、VAX の場合はページ単位 (1 ページは 512 バイト) で、n の値を指定します。この値は、内部でページ単位に切り上げられるため (1 ページの大きさは、CPU によって異なることがあります)、Alpha システムでは、実際に使用できる物理メモリの量は指定したものより大きくなる場合があります。詳細は、『OpenVMS システム管理者マニュアル』を参照してください。

0 または NONE を指定すると、UAF または SUBMIT コマンドで指定した値 (指定している場合) がワーキング・セットの省略時の設定の値になります。

この修飾子は、出力実行キューにも指定できます。その場合、シンビオント・プロセスが作成されるときに、/WSDEFAULT 修飾子は、実行キューのシンビオント・プロセスに対するワーキング・セットの省略時の値を設定します。

ワーキング・セットの省略時の設定の値による、バッチ・ジョブへの影響についての詳細は、表 DCLI-2 を参照してください。

/WSEXTENT=n

バッチ・ジョブに対して、ワーキング・セット超過値、つまりジョブが使用できる物理メモリの最大値を定義します。システムにフリー・ページが十分にある場合にのみ、ジョブは最大量の物理メモリを使用します。この修飾子を指定すると、UAF に指定された値は使用されず、修飾子の値が使用されます。

Alpha システムの場合はページレット単位 (1 ページレットは 512 バイト) で、VAX システムの場合はページ単位 (1 ページは 512 バイト) で、n の値を指定します。この値は、内部でページ単位に切り上げられるため (1 ページの大きさは、CPU によって異なることがあります)、Alpha システムでは、実際に使用できる物理メモリの量は指定したものより大きくなる場合があります。

0 または NONE を指定すると、UAF または SUBMIT コマンドで指定した値がワーキング・セットの省略時の設定になります (WSEXTENT 値が含まれている場合)。

この修飾子は、出力実行キューにも指定できます。その場合、シンビオント・プロセスが作成されるときに、/WSEXTENT 修飾子は出力実行キューのシンビオント・プロセスに対するワーキング・セットの省略時の値を設定します。

ワーキング・セット超過値によるバッチ・ジョブへの影響についての詳細は、表 DCLI-2 を参照してください。

WSQUOTA=n
バッチ・ジョブに対して、ワーキング・セット制限値、つまりジョブに対して保証される物理メモリの量を定義します。

この修飾子を指定すると、UAF に指定された値は使用されず、修飾子の値が使用されます。

OpenVMS Alpha の場合はページレット単位 (1 ページレットは 512 バイト) で、OpenVMS VAX の場合はページ単位 (1 ページは 512 バイト) で、n の値を指定します。この値は、内部でページ単位に切り上げられるため (1 ページの大きさは、CPU によって異なることがあります)、Alpha システムでは、実際に使用できる物理メモリ量は指定したものより大きくなる可能性があります。詳細は、『OpenVMS システム管理者マニュアル』を参照してください。

0 または NONE を指定すると、UAF または SUBMIT コマンドで指定した値がワーキング・セットの省略時の設定の値になります (WSQUOTA 値が含まれている場合)。

この修飾子は、出力実行キューにも指定できます。その場合、シンビオント・プロセスが作成されるときに、/WSQUOTA 修飾子は出力実行キューのシンビオント・プロセスに対するワーキング・セットの省略時の値を設定します。

ワーキング・セットの省略時の設定の値、ワーキング・セット制限値、およびワーキング・セット超過値は、システム UAF の各ユーザ・レコードに含まれています。ワーキング・セット値は、与えられたキューの各ジョブまたはすべてのジョブに指定できます。ワーキング・セット値を含むさまざまな組み合わせの指定と、その処理を表 DCLI-2 に示します。

表 DCLI-2 ワーキング・セット省略時の値、超過値、および制限値の決定

SUBMIT コマンドによって値が指定されている	キューに値が指定されている	処理
No	No	UAF 値を使用します。

(次ページに続く)

表 DCLI-2 (続き) ワーキング・セット省略時の値，超過値，および制限値の決定

SUBMIT コマンドによって値が指定されている	キューに値が指定されている	処理
No	Yes	キューの値を使用します。
Yes	Yes	2つの値のうち小さい方を使用します。
Yes	No	指定した値と UAF を比較して，小さい方を使用します。

例

1. `$ INITIALIZE/QUEUE/PROCESSOR=TELNETSYM -`
`_ $ /ON="192.168.1.101:9100" SYS$PRINT`

この例では，プリント・シンピオント TELNETSYM を指定し，IP アドレス 192.168.1.101，TCP ポート 9100 でプリント・キュー SYS\$PRINT を初期化します。TELNETSYM についての詳細は，『HP TCP/IP Services for OpenVMS Management』を参照してください。

2. `$ INITIALIZE/QUEUE/BATCH/START -`
`_ $ /AUTOSTART_ON=(DATA::, WARF::, DEANNA::) BATCH_1`

この例で INITIALIZE/QUEUE コマンドは，バッチ・キュー BATCH_1 を作成し，それをノード DATA，WARF，または DEANNA で実行できる自動起動キューとして指定します。/START 修飾子を指定すると，自動起動キューが有効になります。このキューは，ENABLE AUTOSTART/QUEUES コマンドが実行される最初のノード（指定されたノードのリストにある）で実行を開始します。

BATCH_1 を実行しているノードが OpenVMS Cluster からはずれると，キューはそのノードで終了し，キュー・マネージャ SYS\$QUEUE_MANAGER の自動起動が有効になっているノードのリスト中の，使用できる最初のノードにフェイルオーバーします。

リストにあるノードの 1 つで自動起動が有効になっている限り，このキューを起動してバッチ・ジョブを実行できます。例にある 3 つのノードがすべてシャットダウンされている場合，または自動起動が無効な場合は，ノード・リストにある 3 つのノードの 1 つが，クラスタに参加して ENABLE AUTOSTART/QUEUES コマンドを実行するまで，キューは終了したままになります。

/NAME_OF_MANAGER 修飾子が指定されていないために，ENABLE AUTOSTART/QUEUES コマンドと INITIALIZE/QUEUE コマンドは，省略時の設定のキュー・マネージャ SYS\$QUEUE_MANAGER が管理するキューにだけ作用します。

3. \$ INITIALIZE/QUEUE/START/BATCH/JOB_LIMIT=3 SYS\$BATCH
\$ INITIALIZE/QUEUE/START/BATCH/JOB_LIMIT=1/WSEXTENT=2000 BIG_BATCH

この例で最初の INITIALIZE/QUEUE コマンドは、すべてのバッチ・ジョブに使用できる SYS\$BATCH というバッチ・キューを作成します。/JOB_LIMIT 修飾子を使用して、3 つのジョブを同時に実行できるよう指定します。2 番目の INITIALIZE/QUEUE コマンドは、大きなジョブに使用する BIG_BATCH という 2 番目のバッチ・キューを作成します。一度に 1 つのジョブだけを実行できます。ワーキング・セット超過値は、OpenVMS Alpha (8KB ページを備えたシステム) では 125 ページまで、OpenVMS VAX では 2000 ページまでになることがあります。

4. \$ INITIALIZE/QUEUE/START/DEFAULT=(FLAG,TRAILER=ONE)-
_ \$ /ON=LPA0: LPA0_PRINT
\$ INITIALIZE/QUEUE/START/DEFAULT=(FLAG,TRAILER=ONE)-
_ \$ /BLOCK_LIMIT=(1000,"")/ON=LPB0: LPB0_PRINT
\$ INITIALIZE/QUEUE/START/GENERIC=(LPA0_PRINT,LPB0_PRINT) SYS\$PRINT
\$ INITIALIZE/QUEUE/START/FORM_MOUNTED=LETTER-
_ \$ /BLOCK_LIMIT=50/ON=TXA5: LQP

この例で最初の 3 つの INITIALIZE/QUEUE コマンドは、プリント・キューを設定します。キュー LPA0_PRINT と LPB0_PRINT は、ジョブ内の各ファイルの前にフラグ・ページを、ジョブの最後のページの後にトレーラ・ページを配置するように設定されています。さらに、LPB0_PRINT には最小ブロック・サイズの 1000 が指定されています。したがって、そのキューでは、1000 ブロック以上のプリント・ジョブだけを実行できます。SYS\$PRINT は、LPA0_PRINT または LPB0_PRINT のいずれかにジョブを転送できる汎用キューとして設定されます。小さすぎて LPB0_PRINT で実行できないジョブは、SYS\$PRINT から LPA0_PRINT に登録されます。

最後の INITIALIZE/QUEUE コマンドは、TXA5 で端末キューを設定します。プリント形式 LETTER のストック・タイプ以外のストック・タイプのプリント形式でキューに登録されたジョブは、同じストック・タイプの形式がキューにマウントされるか、またはエントリがキューから削除されるか、別のキューに移動されるまで、キューで待ち状態となります。LETTER は、専用のレターヘッド紙を示すよう、このサイトで設定されています。ブロック・サイズの上限は 50 で、このキューが 51 ブロックより小さいジョブのために用意されることを示します。

5. \$ INITIALIZE/QUEUE/ON=QUEBID::/BATCH/RAD=0 BATCHQ1

```
$ SHOW QUEUE/FULL BATCHQ1
Batch queue BATCHQ1, stopped, QUEBID::
  /BASE_PRIORITY=4 /JOB_LIMIT=1 /OWNER=[SYSTEM]
  /PROTECTION=(S:M,0:D,G:R,W:S) /RAD=0
```

この例は、ノード QUEBID 上で実行されるバッチ・キュー BATCHQ1 を作成または再初期化します。このキューに割り当てられたジョブはすべて、RAD 0 上で実行されます。

INQUIRE

値を SYS\$COMMAND (通常は、会話型モードのターミナル、または主コマンド・プロシージャの次の行) から読み込んで、シンボルに割り当てます。

フォーマット

INQUIRE シンボル名 [プロンプト文字列]

パラメータ

シンボル名

値を割り当てる、1 ~ 255 文字までの英数字シンボルを指定します。

プロンプト文字列

INQUIRE コマンドを実行するときに表示されるプロンプトを指定します。文字列は、自動的に大文字に変換されます。また、先行スペースと先行タブおよび後続のスペースと後続のタブは削除され、文字の間の複数のスペースおよびタブは1つのスペースに圧縮されます。

プロンプトに、小文字や句読点、複数のスペースとタブ、アット・マーク (@) が含まれる場合には、文字列を二重引用符 (" ") で囲みます。プロンプト文字列に二重引用符を使用する場合には、文字列全体を二重引用符で囲み、文字列の中で二重引用符が存在する場所に連続する2つの二重引用符 ("") を指定します。

システムがプロンプト文字列をターミナルに表示する場合、通常は文字列の最後にコロンの(:)とスペースを表示します(/PUNCTUATION 修飾子を参照してください)。

プロンプト文字列を指定しない場合には、コマンド・インタプリタは、値を要求するプロンプトとしてシンボル名を使用します。

説明

INQUIRE コマンドは、プロセスの作成時に確立された入力ストリームにプロンプト・メッセージを表示し、入力ストリームからの応答を読み取ります。つまり、INQUIRE コマンドは、会話型で実行されているコマンド・プロシージャで実行される場合、コマンド・プロシージャのネスト・レベルにかかわらず、プロンプト・メッセージは常に端末に表示されます。コマンド・プロシージャの INQUIRE コマンドの入力は、RECALL バッファに入ることに注意してください。

プロンプト文字列に対する応答を入力した場合、その値は、指定されたシンボルに文字列として割り当てられます。小文字は自動的に大文字に変換され、先行および後続のスペースとタブが削除され、文字の間の複数のスペースとタブが単一のスペースに圧縮されます。大文字への変換を禁止し、スペースとタブをそのまま残す場合は、文字列を二重引用符で囲みます。

プロンプト文字列に対する応答を入力する場合、シンボルまたはレキシカル関数を使用するためには、単一引用符(')を使用してシンボルの置換を要求します。

端末から会話型でデータを取得するには、READ コマンドも使用できることに注意してください。READ コマンドは、ユーザが入力したとおりにデータを受け付けます。文字は、自動的に大文字に変換されず、スペースは圧縮されません。ただし、一重引用符を使用してシンボルの置換を要求しても、シンボルとレキシカル関数は変換されません。

バッチ・ジョブに INQUIRE コマンドを入力すると、コマンドは、コマンド・プロシージャの次の行から応答を読み取ります。プロシージャがネストしている場合、第1レベルのコマンド・プロシージャから応答が読み取られます。バッチ・ジョブのコマンド・プロシージャの次の行がドル記号(\$)で始まる場合、その行は INQUIRE に対する応答でなく、コマンドとして解釈されます。次に INQUIRE コマンドは、指定されたシンボルに空文字列を割り当て、バッチ・ジョブは INQUIRE コマンドの次の行のコマンドから処理を続けます。

修飾子

/GLOBAL

シンボルが、グローバル・シンボル・テーブルに登録されることを指定します。

/GLOBAL 修飾子を指定しない場合には、シンボルはローカル・シンボル・テーブルに登録されます。

/LOCAL (省略時の設定)

シンボルが、現在のコマンド・プロシージャのローカル・シンボル・テーブルに登録されることを指定します。

/PUNCTUATION (省略時の設定)

/NOPUNCTUATION

ターミナルにプロンプトが表示されるときに、そのプロンプトのあとにコロン(:)とスペースが出力されるかどうかを制御します。コロンとスペースを出力しない場合には、/NOPUNCTUATION を指定します。

例

1.

```
$ INQUIRE CHECK "Enter Y[ES] to continue"
$ IF .NOT. CHECK THEN EXIT
```

この INQUIRE コマンドは、ターミナルに次のプロンプト・メッセージを表示します。

```
Enter Y[ES] to continue:
```

INQUIRE コマンドは、上記のようなプロンプトを表示し、CHECK というシンボルに割り当てられる値を要求します。IF コマンドは、シンボル CHECK に割り当てられた値を評価します。CHECK に割り当てられた値が真（つまり、奇数の数値、T、t、Y、y から始まる文字列、あるいは奇数の数字列）の場合には、プロセスは実行を継続します。

CHECK に割り当てられた値が偽（つまり、偶数の数値、T、t、Y、y 以外の文字から始まる文字列、あるいは偶数の数字列）の場合には、プロセスの実行は終了します。

2.

```
$ INQUIRE COUNT
$ IF COUNT .GT. 10 THEN GOTO SKIP
.
.
.
$ SKIP:
```

この INQUIRE コマンドは、次のメッセージを表示し、COUNT に割り当てられる値を要求します。

```
COUNT:
```

このあと、このコマンド・プロセスは、COUNT というシンボルに割り当てられた値を評価し、次に続くコマンドを実行するのか、あるいは制御を SKIP というラベルの行に渡すのかを判断します。

3.

```
$ IF P1 .EQS. "" THEN INQUIRE P1 "FILE NAME"
$ FORTRAN 'P1'
```

この IF コマンドは、P1 というシンボルが空文字列であるかどうかを調べることで、パラメータがコマンド・プロセスに渡されたかどうかを調べます。このシンボルの値が空文字列の場合には、パラメータが指定されていないので、INQUIRE コマンドを実行してパラメータを要求します。P1 が指定されていた場合には、INQUIRE コマンドは実行されず、Fortran コマンドがパラメータに指定された名前のファイルをコンパイルします。

INSTALL

Install ユーティリティを起動します。Install ユーティリティは選択した実行可能イメージや共有イメージをシステムに“知らせ”，適切な属性を割り当てることにより，それらのイメージの性能を向上させます。

Install ユーティリティについての詳細は，『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』またはオンライン・ヘルプを参照してください。

フォーマット

INSTALL [サブコマンド]/[ファイル指定]

JAVA

JAVA コマンドは、Java™アプリケーションを起動します。これは、JAVAC などの Java コンパイラで作成された Java クラス・ファイルを実行します。

JAVA コマンドは、Java Software Development Kit (SDK) または Run-Time Environment (RTE) が OpenVMS システムにインストールされている場合に限り使用可能です。

Java SDK インストール・キットは、OpenVMS メディア・キットの OpenVMS e-Business Infrastructure CD-ROM に含まれています。また、次の Web サイトからダウンロードすることもできます。

<http://www.hp.com/software/java/alpha>

Java SDK または RTE をインストールした後、次のコマンドを入力することによって、オンライン・ヘルプにアクセスすることができます。

```
$ JAVA -help
```

OpenVMS システムに JDK ドキュメントがインストールされている場合は、ブラウザで次の場所を参照することによって JDK ツール(コマンド)についてのドキュメントおよびその他 Java 参照情報を表示することができます。たとえば、Java SDK v 1.4.0 の場合には、ブラウザで次の位置に移動します。

```
SYS$COMMON:[JAVA$140.DOCS]INDEX.HTML
```

JOB

カード・リーダからキューに登録されたバッチ・ジョブの先頭を示します。システム・カード・リーダから登録されるバッチ・ジョブの先頭には、必ず JOB カードを付けなければなりません。

JOB コマンドを短縮形にすることはできません。

フォーマット

JOB ユーザ名

パラメータ

ユーザ名

ジョブが実行される時のユーザ名を指定します。ログインの時に入力する名前と同じユーザ名を指定します。

説明

JOB カードは、ジョブに登録するユーザを識別し、その後にパスワードを与える PASSWORD カードが続きます。PASSWORD カードは必須ですが、アカウントのパスワードが空の場合、カードでパスワードを使用する必要はありません。

ユーザ名とパスワードは、ログイン・プロシージャと同じ方法で、システム登録ファイルによって確認されます。バッチ・ジョブを実行するプロセスに、省略時のディスクとディレクトリ、およびユーザ・アカウントに付属する特権が割り当てられます。指定されたユーザ名の LOGIN.COM ファイルが存在すると、ジョブの最初に実行されます。

バッチジョブの最後は、EOJ コマンド、EOF カード (12-11-0-1-6-7-8-9overpunch)、または別のジョブ・カードで示されます。

修飾子

/AFTER=時刻

指定された時刻まで、ジョブを実行しないことを要求します。指定された時刻がすでに過ぎている場合には、ジョブは直ちに処理するために、キューに登録されます。

時刻は、絶対時刻または絶対時刻とデルタ時間の組み合わせとして指定できます。時刻の指定方法についての詳細は、『OpenVMS ユーザーズ・マニュアル』またはオンライン・ヘルプのトピック Date を参照してください。

/CHARACTERISTICS=(属性[,...])

ジョブの実行のために必要な、1 つ以上の属性を指定します。属性を 1 つしか指定しない場合には、括弧を省略できます。属性のコードはシステムごとのインストール時に定義されます。システムでどの属性を使用できるかを調べるには、SHOW QUEUE/CHARACTERISTICS コマンドを使用します。

ユーザが/CHARACTERISTICS 修飾子を指定する場合には、その修飾子に対して指定する属性はすべて、ジョブを実行するキューに対しても指定されていなければなりません。この修飾子に指定された属性がキューに対して指定されていない場合には、キュー属性が変更されるまで、または DELETE/ENTRY コマンドを使用してエントリが削除されるまで、そのジョブはキューの中で待ち状態となり実行されません。キューに対してすでに指定されている属性を指定する場合には、JOB カードにすべての属性を指定する必要はありません。属性が全く指定されていない場合も、ジョブは実行されます。

/CLI=ファイル名

この修飾子を使用すると、ジョブを処理するとき、他のコマンド言語インタプリタ (CLI) を使用するように要求できます。ファイル名を入力するとコマンド言語インタプリタ (CLI) は SYS\$SYSTEM: ファイル名.EXE となります。省略時には、ジョブは利用者登録ファイル (UAF) に指定されている CLI を使って実行されます。

/CPUTIME=n

バッチ・ジョブが使用できる CPU 時間の上限を定義します。n に対して指定できる値は、デルタ時間、0 という値、NONE あるいは INFINITE というキーワードです (時刻値の指定については、『OpenVMS ユーザーズ・マニュアル』またはオンライン・ヘルプのトピック Date を参照してください)。

許可されている CPU 時間より小さな値を指定する場合には、/CPUTIME 修飾子を使用して、システム管理者が設定したキューの基本値、あるいは利用者登録ファイルに登録されている値を変更します。無限の時間を要求する場合には、0 または INFINITE を指定します。CPU 時間を、利用者登録ファイルの値に設定する場合や、キューに対して指定されている上限に設定する場合には、NONE を指定します。基本のキューの上限、あるいは利用者登録ファイル (UAF) に指定されている値より大きな値は要求できません。

/DELETE (省略時の設定)

/NODELETE

ジョブが処理されたあと、バッチ入力ファイルが保存されるかどうかを制御します。/NODELETE 修飾子を指定する場合には、ファイルは、INPBATCH.COM という省略時の名前を使用して、ユーザの省略時のディレクトリに保存されます。/NAME 修飾子を使用すると、バッチ入力ファイルのファイル名は、/NAME 修飾子に指定したジョブ名と同じになります。

/HOLD

/NOHOLD (省略時の設定)

ジョブを直ちに処理するかどうかを制御します。

/HOLD 修飾子を指定する場合には、SET QUEUE/ENTRY コマンドに/NOHOLD 修飾子または/RELEASE 修飾子を使用して明確に解放するまで、ジョブは処理のために解放されることはありません。

/KEEP

/NOKEEP (省略時の設定)

印刷後にログ・ファイルを削除するかどうかを制御します。/NOPRINTER 修飾子が省略されている場合には、/NOKEEP 修飾子が省略時の設定として使用されます。

/LOG_FILE=ファイル指定

/NOLOG_FILE

指定した名前のログ・ファイルがジョブ実行の際に作成されるかどうか、またはログ・ファイルが作成されるかどうかを制御します。

/LOG_FILE 修飾子を指定した場合には、システムは指定したファイルにログ情報を書き込みます。/NOLOG_FILE を指定した場合には、ログ・ファイルは作成されません。どちらの修飾子も指定しない場合には、ログ情報は、省略時のディレクトリに含まれるファイルの中で、ジョブの中の最初のコマンド・ファイルと同じファイル名を持ち、ファイル・タイプが.LOG であるファイルに書き込まれます。省略時の設定は、/LOG_FILE 修飾子も/NOLOG_FILE 修飾子も指定されていない状態です。

/LOG_FILE 修飾子を使用すれば、ログ・ファイルを別の装置に書き込むように指定できます。ファイル指定に含まれる論理名は、ジョブをキューに登録する時に変換されます。バッチ・ジョブを実行しているプロセスは、ログ・ファイルが存在する装置に対してアクセスできなければなりません。

/LOG_FILE 修飾子を省略し/NAME 修飾子を指定する場合には、ログ・ファイルは、/NAME 修飾子によって指定されたものと同じファイル名を持ち、ファイル・タイプが.LOG であるファイルに書き込まれます。

/NAME=ジョブ名

ジョブ名、およびバッチ・ジョブ・ログ・ファイルとコマンド・ファイルのファイル名として使用される文字列を指定します。ジョブ名は、1 ~ 39 文字までの英数字文字列であり、ファイル名として正しくなければなりません。省略時のログ・ファイル名はINPBATCH.LOG であり、省略時のコマンド・ファイル名はINPBATCH.COM です。

/NOTIFY

/NONOTIFY (省略時の設定)

ジョブが終了または強制終了したことを示すメッセージが、ログ・インしたターミナルに表示されるかどうかを制御します。

/PARAMETERS=(パラメータ[,...])

オプションとしてコマンド・プロシージャに渡すことのできる、1 ~ 8 個のパラメータを指定します。このパラメータは、バッチ・ジョブにおいて、P1 ~ P8 までのシンボル名に割り当てられる値を定義します。これらのシンボルは、指定コマンド・プロシージャに対してのみ有効です。

パラメータを 1 つしか指定しない場合には、括弧を省略できます。

各パラメータは、コンマ(,)で区切ります。スペースや特殊文字、区切り文字、小文字を含むパラメータを指定する場合には、パラメータ全体を二重引用符(" ")で囲まなければなりません。各パラメータは、最大 255 文字の長さです。

/PRINTER=キュー名

/NOPRINTER

ジョブ終了時に、ジョブ・ログ・ファイルを印刷のために指定キューに登録するかどうかを制御します。ログ・ファイルの省略時のプリント・キューは、SYS\$PRINT です。

/NOPRINTER 修飾子を指定する場合には、/KEEP 修飾子が指定されていると解釈されます。

/PRIORITY=n

優先順位の値をシステムパラメータ MAXQUEPRI の値より高くするには、OPER (オペレータ) または ALTPRI (優先順位変更) 特権が必要です。

指定したジョブに対して、ジョブ・スケジューリング優先順位を指定します。優先順位の値は 0 ~ 255 までの範囲です。0 がもっとも低い優先順位であり、255 が最高の優先順位です。

/PRIORITY 修飾子の省略時の値は、システムパラメータ DEFQUEPRI の値です。MAXQUEPRI の値より低い優先順位を設定する場合には、特別な特権は必要ありません。

/PRIORITY 修飾子がプロセスの優先順位に影響を与えることはありません。プロセス優先順位は、キューで設定されます。

/QUEUE=キュー名[:]

ジョブが登録される、バッチ・キューの名前を指定します。/QUEUE 修飾子を指定しない場合には、ジョブの省略時のシステム・バッチ・ジョブ・キューである SYS\$BATCH に登録されます。

/RESTART

/NORESTART (省略時の設定)

システム障害が発生した後や STOP/QUEUE/REQUEUE コマンドが実行された後に、ジョブを再スタートするかどうかを指定します。

/TRAILING_BLANKS (省略時の設定)
/NOTRAILING_BLANKS

カード・デックの入力カードをカード・イメージ形式で読み込むのか、あるいは入力レコードを空白ではない最後の文字で切り捨てるのかを制御します。省略時には、システムはカード・リーダから読み取られたレコードから、後続の空白を削除しません。入力レコードから後続する空白を削除する場合には、/NOTRAILING_BLANKS 修飾子を使用します。

/WSDEFAULT=n

バッチ・ジョブのワーキング・セットの省略時の大きさの値を定義します。

/WSDEFAULT 修飾子は、利用者登録ファイル(UAF)に指定されているワーキング・セット・サイズに優先します。

n の値は、OpenVMS Alpha では 512 バイトのページ・レットの数で、OpenVMS VAX では 512 バイトのページの数で指定します。OpenVMS Alpha は、この値を CPU 固有のページに最も近い数に切り上げ、実際に許可される物理メモリの量が指定量よりも大きくなるようにします。n に対して指定できる値は、1 ~ 65535 までの整数、0、またはキーワード NONE です。詳細は『OpenVMS システム管理者マニュアル』を参照してください。

この修飾子を使用すれば、システム管理者が設定したキューの基本値、あるいは利用者登録ファイル(UAF)に登録されている値よりも小さい値にすることができます。ワーキング・セットの値を、利用者登録ファイルに指定されている省略時の値や、キューに対して指定されている省略時の値に設定する場合には、0 または NONE を指定します。省略時の値より大きな値は要求できません。

/WSEXTENT=n

バッチ・ジョブのワーキング・セットの超過値を定義します。/WSEXTENT 修飾子は、利用者登録ファイル(UAF)に指定されているワーキング・セット超過値に優先します。

n の値は、OpenVMS Alpha では 512 バイトのページ・レットの数で、OpenVMS VAX では 512 バイトのページの数で指定します。OpenVMS Alpha は、この値を CPU 固有のページに最も近い数に切り上げ、実際に許可される物理メモリの量が指定量よりも大きくなるようにします。n に対して指定できる値は、1 ~ 65535 までの整数、0、またはキーワード NONE です。詳細は『OpenVMS システム管理者マニュアル』を参照してください。

ワーキング・セット超過値を小さくするには、この修飾子を使用して、利用者登録ファイル(UAF)に指定されている値ではなく、システム管理者が設定したキューの基本値よりも小さい値にします。ワーキング・セット超過値を、利用者登録ファイルに指定されている値や、キューに対して指定されている値に設定する場合には、0 または NONE を指定します。省略時の値より大きな値は要求できません。

/WSQUOTA=n

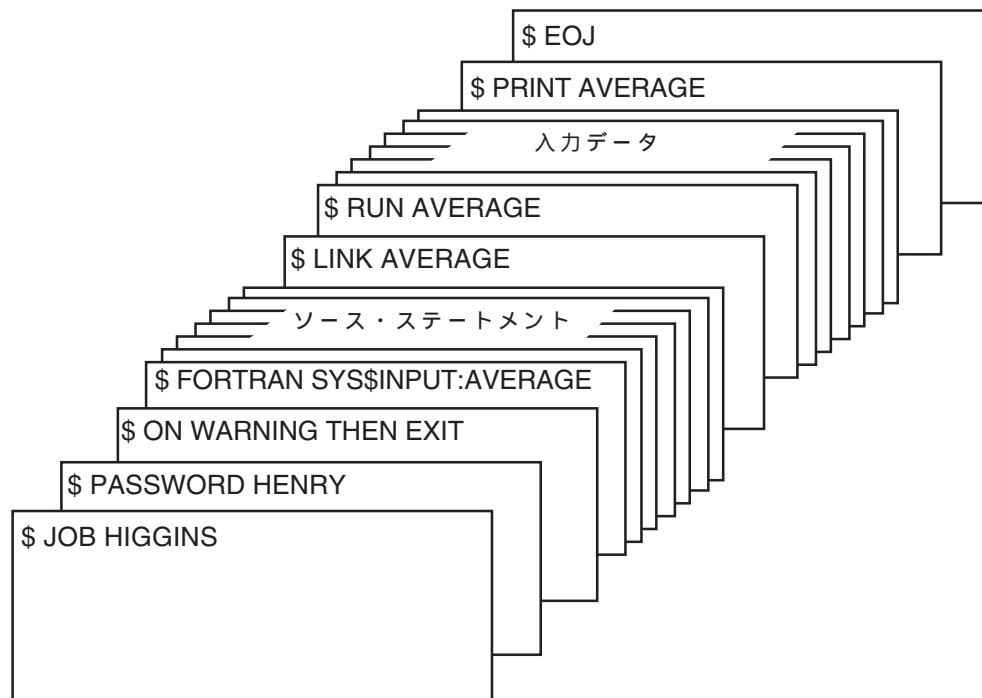
バッチ・ジョブの最大のワーキング・セット・サイズ(ワーキング・セット・クォータ)を定義します。/WSQUOTA 修飾子は、利用者登録ファイル(UAF)に指定されている値に優先します。

n の値は、OpenVMS Alpha では 512 バイトのページ・レットの数で、OpenVMS VAX では 512 バイトのページの数で指定します。OpenVMS Alpha は、この値を CPU 固有のページに最も近い数に切り上げ、実際に許可される物理メモリの量が指定量よりも大きくなるようにします。n に対して指定できる値は、1 ~ 65535 までの整数、0、またはキーワード NONE です。詳細は『OpenVMS システム管理者マニュアル』を参照してください。

この修飾子を使用すれば、システム管理者が設定したキューの基本値、あるいは利用者登録ファイル(UAF)に登録されている値よりも小さい値にすることができます。ワーキング・セット・クォータの値を、利用者登録ファイルに指定されている省略時の値や、キューに対して指定されている省略時の値に設定する場合には、0 または NONE を指定します。省略時の値より大きな値は要求できません。

例

1.

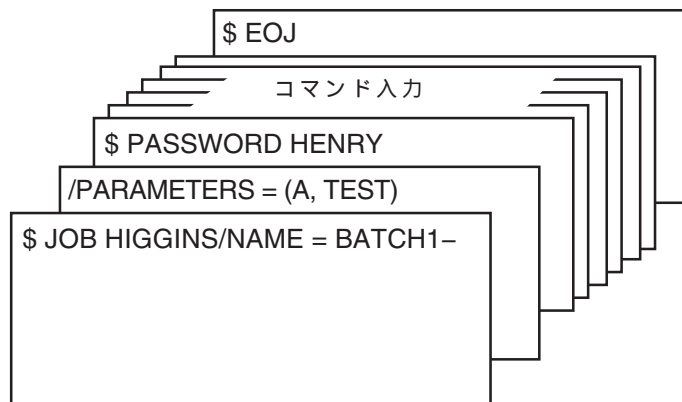


JRD-0787-GE

JOB および PASSWORD カードは、ユーザ HIGGINS がバッチ・ジョブを登録するのを識別し認可します。コマンド・ストリームは、Fortran コマンドとコンパイル前の Fortran ソース・ステートメントから構成されます。装置名 SYS\$INPUT に続くファイル名 AVERAGE は、オブジェクト・ファイルとリスティング・ファイルのファイル名と持つコンパイラを提供します。出力ファイルは、ユーザ HIGGINS の省略時のディレクトリに置かれます。

コンパイルが正常終了すると、LINK コマンドは実行可能イメージを作成し、RUN コマンドでそのイメージを実行します。コマンド・ストリームの RUN コマンドの次に、プログラムの入力を行います。ジョブの最後のコマンドは、プログラム・リスティングを印刷します。デッキの最後のカードには、EOJ (end-of-job) コマンドが含まれています。

2.



JRD-0788-GE

JOB カードの/NAME 修飾子は、バッチ・ジョブ名を指定します。ジョブが終了すると、印刷されたログ・ファイルは BATCH1.LOG として識別されます。JOB コマンドは、継続文字(-)がある 2 番目のカードまで続きます。/PARAMETERS 修飾子は、P1 を A、P2 を TEST と定義します。デッキの最後のカードには、EOJ (end-of-job) コマンドが含まれています。

レキシカル関数

文字列および現在のプロセスに関する属性の情報を戻す関数です。

説明

コマンド言語は、レキシカル関数と呼ばれる機能を含んでいます。レキシカル関数は、現在のプロセスや算術式および文字列に関する情報を返します。コマンド・インタプリタがコマンド処理の入力文字列の解析処理(レキシカル処理)フェーズを実行する間に関数を評価するため、これらの関数はレキシカル関数と呼ばれます。

通常、シンボルまたは式を使用するコンテキストであれば、任意コンテキストでレキシカル関数を使用できます。コマンド・プロシージャでは、レキシカル関数を使用して論理名の変換、文字列操作の実行、およびプロシージャを処理しているノードの判別を行うことができます。

次に標準的な形式を示します。

F\$function-name([args,...])

F\$	レキシカル関数が後に続くことを示します。
function-name	関数を評価することを指定するキーワード。一意に認識できれば、関数名は省略できます。
()	必要であれば、引数を囲みます。引数をとらない関数を含め、すべての関数で括弧は必要です。
args,...	必要であれば、整数または文字列式を使用して引数を指定します。

式の指定についての詳細は、『OpenVMS ユーザーズ・マニュアル』を参照してください。

表 DCLI-3 では、レキシカル関数とそれぞれが戻す情報について簡単に説明しています。これ以降のページで、各関数ごとに例を挙げて詳しく説明します。

表 DCLI-3 レキシカル関数の要約

関数	説明
F\$CONTEXT	F\$PID 関数とともに使用し、その選択基準を指定します。

(次ページに続く)

表 DCLI-3 (続き) レキシカル関数の要約

関数	説明
F\$CSID	OpenVMS Cluster 識別番号を戻します。また、コンテキスト・シンボルが、システムのノード・リストの中で、現在の位置を示すように変更します。
F\$CVSI	文字列データからビット・フィールドを取り出し、その結果を符号付き値として整数に変換します。
F\$CVTIME	絶対時刻、デルタ時間またはそれらの組み合わせの文字列に関する情報を戻します。
F\$CVUI	文字データからビット・フィールドを取り出し、その結果を符号なし値として整数に変換します。
F\$DELTA_TIME	指定した開始時刻と終了時刻の差を返します。
F\$DEVICE	選択基準に合う装置名をすべて返します。
F\$DIRECTORY	現在の省略時のディレクトリ名文字列を戻します。
F\$EDIT	編集リストに指定された編集をもとに、文字列式を編集します。
F\$ELEMENT	指定した区切り文字で区切られた要素で構成される文字列から、1つの要素を取り出します。
F\$ENVIRONMENT	DCL コマンド環境に関する情報を戻します。
F\$EXTRACT	文字列式から部分文字列を取り出します。
F\$FAO	\$FAO システム・サービスを呼び出し、指定された制御文字列を ASCII 形式の出力文字列に変換します。
F\$FID_TO_NAME (Alpha/I64 のみ)	ファイル識別をファイル指定に変換します。
F\$FILE_ATTRIBUTES	指定されたファイルの属性に関する情報を戻します。
F\$GETDVI	\$GETDVI システム・サービスを呼び出し、指定された装置に関する指定された情報を戻します。
F\$GETENV (Alpha のみ)	\$GETENV システム・サービスを呼び出し、指定されたコンソール環境変数の値を戻します。
F\$GETJPI	\$GETJPI システム・サービスを呼び出し、指定されたプロセスに関する会計情報、状態情報、および識別情報を戻します。
F\$GETQUI	\$GETQUI システム・サービスを呼び出し、キューに関する情報を戻します。この情報には、バッチ・キュー、プリント・キューに登録されるジョブ、ファーム定義やシステム・バッチ・キュー・ファイルに定義された属性などが含まれます。 †VAX システムでは、キュー・マネージャについての情報も戻します。
F\$GETSYI	\$GETSYI システム・サービスを呼び出し、利用者のシステムに関する状態情報と識別情報を戻します (システムが、VMScluster の一部に含まれている場合には、ローカルなノードに関する状態情報と識別情報を戻します)。
F\$IDENTIFIER	指定した形式の識別子を、それに相当する整数に変換します。あるいは、その逆の操作を実行します。
F\$INTEGER	指定した式の結果を、整数で戻します。
†VAX のみ	

(次ページに続く)

表 DCLI-3 (続き) レキシカル関数の要約

関数	説明
F\$LENGTH	指定した文字列の長さを戻します。
F\$LICENSE (Alpha/I64 のみ)	指定されたライセンスがシステムにロードされているかどうかを確認します。
F\$LOCATE	文字列に含まれる 1 文字または部分文字列を探し、文字列中でのそのオフセットを戻します。
F\$MESSAGE	指定したシステム状態コードに対応するメッセージ文を戻します。
F\$MODE	プロセスが実行されているモードを示す文字列を戻します。
F\$MULTIPATH (Alpha/I64 のみ)	特定のマルチパス対応装置に対する、指定された情報項目を戻します。
F\$PARSE	\$PARSE RMS サービスを起動してファイル指定を解析し、指定した特定のファイル指定、または拡張されたファイル指定を戻します。
F\$PID	起動するたびに、次のプロセス識別番号を戻します。
F\$PRIVILEGE	現在のプロセス特権が、引数に指定されている特権と一致するかどうかに応じて、TRUE または FALSE という値を戻します。
F\$PROCESS	現在のプロセス名を文字列で戻します。
F\$SEARCH	\$SEARCH RMS サービスを呼び出し、ディレクトリ・ファイルを検索し、指定したファイルの完全な形のファイル指定を戻します。
F\$SETPRV	指定された特権をセットし、変更される前の特権の状態を示しているキーワード・リストを戻します。
F\$STRING	指定した式の結果に相当する文字列を戻します。
F\$TIME	現在の日付/時刻を示す文字列を戻します。形式は dd-mmm-yyyy hh:mm:ss.cc です。
F\$TRNLNM	論理名を変換し、等価名文字列、または要求されている論理名の属性を戻します。
F\$TYPE	シンボルのデータ・タイプを判断します。
F\$UNIQUE (Alpha/I64 のみ)	ファイル名として適切で、クラスタ全体で一意的な文字列を生成します。
F\$USER	現在の利用者識別コード (UIC) を戻します。
F\$VERIFY	プロシージャ・チェックの設定がオンの場合には整数の 1、プロシージャ・チェックの設定がオフの場合には整数の 0 が戻されます。また、新しいチェック状態をセットすることもできます。

F\$CONTEXT

F\$PID 関数とともに使用し、その選択基準を指定します。F\$CONTEXT 関数を使用すると、F\$PID 関数は OpenVMS Cluster 内の任意のノードからプロセス情報を取得できるようになります。

フォーマット

F\$CONTEXT(コンテキスト・タイプ, コンテキスト・シンボル, 選択項目, 選択値, 値修飾子)

戻り値

空文字列("").

引数

コンテキスト・タイプ

コンテキスト・タイプを指定します。

現在使用できるコンテキスト・タイプは PROCESS だけです。これは、F\$PID に対する選択基準を決めるために使用します。同一 UIC のプロセスを見る場合は、特権は必要ありません。同一 UIC グループ内の別の UIC のプロセスを見る場合は、GROUP 特権が必要です。システム全体の UIC のプロセスを見る場合は、WORLD 特権が必要です。

コンテキスト・シンボル

F\$CONTEXT で作成されたコンテキスト・メモリを参照するために、DCL が使用するシンボル名を指定します。F\$PID は、このコンテキスト・シンボルを使用して PID リストを処理します。

シンボルを使用してコンテキスト・シンボルを指定してください。コマンド・プロシージャ内で最初に F\$CONTEXT を使用するときには、未定義または空文字列が割り当てられたシンボルを指定してください。シンボルは“PROCESS_CONTEXT”タイプでローカル・シンボルをして作成されます。コンテキストが無効になった場合（つまり F\$PID ですべての PID を検索したか、または、その実行時にエラーが発生した場合）は、シンボルのタイプは“PROCESS_CONTEXT”ではなくなっています。コンテキストをキャンセルする場合は、コマンド・プロシージャ内で F\$TYPE 関数を使用して検索できます。

選択基準を設定した後で F\$PID を呼び出す場合は、このコンテキスト・シンボルを使用します。

選択項目

選択項目は、F\$CONTEXT が使用する選択基準を指定するキーワードです。一度の呼出しに対して 1 つのキーワードしか指定できません。

注意

項目のリストを使用する場合は、NEQ 選択値を使用しないでください。これは、結果として得られる状態値は、つねに "真" だからです。

次に例を示します。

```
$ EXAMPLE=f$context("PROCESS",CTX,"USERNAME","A*",B*,"NEQ")
```

この式は、“ユーザ名が A* に等しくないまたはユーザ名が B* に等しくない場合は、その規則に合致するユーザのプロセスを返します”と解釈されます。このオペランドは論理和であるので、状態値は常に“真”になります (たとえば ALFRED は B* に等しくないとか、BOB は A* に等しくないというように、どんなユーザ名でも A* あるいは B* に等しくない値となります)。

コンテキスト・タイプ PROCESS に対して使用できる選択項目キーワードを次の表に示します。

選択項目	選択値	値修飾子	説明
ACCOUNT	文字列	EQL, NEQ	有効なアカウント名 (またはそのリスト)。アスタリスク (*) およびパーセント記号 (%) ワイルドカード文字を使用できる。
AUTHPRI	整数値	GEQ, GTR, LEQ, LSS, EQL, NEQ	許可されたベース優先度。Alpha では (0-63)。VAX では (0-31) をとる。
CANCEL			該当コンテキストに対する選択基準を取り消す。
CURPRIV	キーワード	ALL, ANY, EQL, NEQ	有効な特権名キーワード (またはそのリスト)。詳細は『OpenVMS システム・セキュリティ・ガイド』参照。
GRP	整数	GEQ, GTR, LEQ, LSS, EQL, NEQ	UIC グループ番号。
HW_MODEL	整数	EQL, NEQ	有効なハードウェア型番号
HW_NAME	文字列	EQL, NEQ	有効なハードウェア名またはキーワードのリスト。アスタリスク (*) およびパーセント記号 (%) ワイルドカード文字を使用できる。
JOBPRCNT	整数	GEQ, GTR, LEQ, LSS, EQL, NEQ	ジョブのサブプロセス数

選択項目	選択値	値修飾子	説明
JOBTYPE	キーワード	EQL, NEQ	有効なジョブ・タイプ・キーワード (DETACHED, NETWORK, BATCH, LOCAL, DIALUP, REMOTE)。詳細は『OpenVMS ユーザーズ・マニュアル』参照。
MASTER_PID	文字列	EQL, NEQ	主プロセス ID
MEM	整数	GEQ, GTR, LEQ, LSS, EQL, NEQ	UIC メンバ番号
MODE	キーワード	EQL, NEQ	プロセス・モード・キーワード (OTHER, NETWORK, BATCH, および INTERACTIVE)。詳細は『OpenVMS ユーザーズ・マニュアル』参照
NODE_CSID	整数	EQL, NEQ	ノードのクラスタ ID 番号
NODENAME	文字列	EQL, NEQ	ノード名 (またはそのリスト)。アスタリスク (*) およびパーセント記号 (%) ワイルドカードを使用できる。省略時の設定はユーザのローカル・ノード。すべてのノードを指定する場合は、値“*”を使用する。
OWNER	文字列	EQL, NEQ	直親 PID
PRCNT	整数	GEQ, GTR, LEQ, LSS, EQL, NEQ	プロセスのサブプロセス数
PRCNAM	文字列	EQL, NEQ	プロセス名 (またはそのリスト)。アスタリスク (*) およびパーセント記号 (%) ワイルドカード文字を使用できる。
PRI	整数	GEQ, GTR, LEQ, LSS, EQL, NEQ	プロセス優先度。Alpha では (0-63), VAX では (0-31) をとる。
PRIB	整数	GEQ, GTR, LEQ, LSS, EQL, NEQ	ベース・プロセス優先度。Alpha では (0-63), VAX では (0-31) をとる。
STATE	キーワード	EQL, NEQ	有効なプロセス状態キーワード。詳細は、『OpenVMS System Services Reference Manual』の \$GETJPI サービスに関する説明を参照。
STS	キーワード	EQL, NEQ	有効なプロセス状態キーワード。詳細は、『OpenVMS System Services Reference Manual』の \$GETJPI サービスに関する説明を参照。
TERMINAL	文字列	EQL, NEQ	端末名 (またはそのリスト)。アスタリスク (*) およびパーセント記号 (%) ワイルドカード文字を使用できる。
UIC	文字列	EQL, NEQ	ユーザ識別子 (UIC)。“[group,member]”の形式。
USERNAME	文字列	EQL, NEQ	ユーザ名 (またはそのリスト)。アスタリスク (*) およびパーセント記号 (%) ワイルドカード文字を使用できる。

選択値

選択基準の値を指定します。たとえば次の例に示すように、ノード MYVAX 上の全プロセスを処理するには、“NODENAME”キーワードに対して“MYVAX”を指定します。

```
$ X = F$CONTEXT("PROCESS", ctx, "NODENAME", "MYVAX", "EQL")
```

いくつかの選択項目では、値をリストにすることもできます。この場合、各値はコンマ(.)で区切らなければなりません。たとえば、ノード名に MYVAX, HERVAX, および HISVAX を指定するには、次のように入力します。

```
$ X=F$CONTEXT("PROCESS",ctx,"NODENAME","MYVAX,HERVAX,HISVAX","EQL")
```

ファイルを指定する場合と同様に、いくつかの値に対して、ワイルドカード文字(*や%)を使用できます。

値修飾子

選択値に対する修飾子を指定します。選択値は修飾しなければなりません。

たとえば、次に示すプロセス値のいずれか 1 つに基づいて選択するよう要求することにより、値を修飾できます。

- LSS — F\$PID への呼び出しで指定した値より小さい
- LEQ — F\$PID への呼び出しで指定した値より小さいかまたは等しい
- GTR — F\$PID への呼び出しで指定した値より大きい
- GEQ — F\$PID への呼び出しで指定した値より大きいまたは等しい
- EQL — F\$PID への呼び出しで指定した値と等しい
- NEQ — F\$PID への呼び出しで指定した値と等しくない

リストの中には、キーワード ALL, ANY, EQL および NEQ で修飾されるものもあります。このようなリストは通常、複数の有効な特権から構成される、プロセス特権マスクのようなマスクです。各キーワードとその意味を次に示します。

- ALL — リストのすべての項目がプロセスで真である。
- ANY — リストの任意の項目がプロセスの属性の一部である。
- EQL — 値が正確に一致する(つまり、指定されていない値は、プロセスで真であってはならない)。
- NEQ — 値が一致しない。

複数の選択値を特定の選択修飾子と使用すると、(OR オペランドが指定されているかのように) 選択基準と一致するものを値とみなします。(AND オペランドが指定されているかのように) 選択値は累積基準ではありません。

ALL は指定した値は存在しなければなりません、指定しない他の値も存在しても構いません。EQL は指定した値は存在しなければなりません、指定しない他の値は存在してはなりません。これが、ALL と EQL の相違点です。たとえば、現在のプロセスが TMPMBX と OPER 特権を持っていて、他の特権を持っても構わない場合は ALL を、他の特権を持っていない場合は EQL をそれぞれ指定します。

説明

F\$CONTEXT 関数を使用して F\$PID 関数の選択基準を設定します。

F\$CONTEXT 関数は、基準を作るときは何度でも呼び出せます。ただし、1 回の呼び出しにつき 1 つの項目しか指定できません。項目値リストが適切な位置に与えられれば、複数のコンテキストを同時に操作することができます。

適切な呼出しで選択基準を設定したら、F\$PID を繰り返し呼出して、F\$CONTEXT 関数で指定した基準と合う全てのプロセス識別番号 (PID) を戻します。基準と合うプロセスが無くなると、空文字列が戻されます。

F\$PID 関数が呼び出されると、コンテキスト・シンボルは“frozen”とみなされます。つまり、対応する選択基準が削除されるまで、同じコンテキスト・シンボルでは F\$CONTEXT を再び呼び出すことができません。同じコンテキスト・シンボルに追加の選択基準を設定しようとすると、エラー・メッセージが表示されますが、コンテキストと選択基準には影響しないため F\$PID 関数の呼出しは続けることができます。

F\$CONTEXT は、プロセス・メモリを使用して選択基準を格納します。ある 2 つの状況では、このメモリが削除されます。F\$PID 関数を呼出して空文字(“)が戻されたとき、つまり選択基準に合う全てのプロセスが戻された場合と、呼出しの際に CANCEL というキーワードをコンテキストとともに使用した場合です。このタイプの呼出しは Ctrl/Y 操作や他の条件処理ルーチンに適しています。

例

```
1. $!Establish an error and CTRL/Y handler

$ ON ERROR THEN GOTO error
$ ON CONTROL_Y THEN GOTO error
$!
$ ctx = ""
$ temp = F$CONTEXT ("PROCESS", ctx, "NODENAME", "", "EQL")
$ temp = F$CONTEXT ("PROCESS", ctx, "USERNAME", "M*,SYSTEM", "EQL")
$ temp = F$CONTEXT ("PROCESS", ctx, "CURPRIV", "SYSPRV,OPER", "ALL")
$!Loop over all processes that meet the selection criteria.
$!Print the PID and name of the image for each such process.
```

```
$loop:
$ pid = F$PID(ctx)
$ IF pid .EQS. ""
$ THEN
$     GOTO endloop
$ ELSE
$     image = F$GETJPI(pid,"IMAGENAME")
$     SHOW SYMBOL pid
$     WRITE SYS$OUTPUT image
$     GOTO loop
$ ENDIF
$!The loop over the processes has ended.

$endloop:
$ EXIT
$!
$!Error handler. Clean up the context's memory with the CANCEL
$!selection item keyword.
$!
$error:
$ IF F$TYPE(ctx) .eqs. "PROCESS_CONTEXT" THEN -
    temp = F$CONTEXT ("PROCESS", ctx, "CANCEL")
$ EXIT
```

この例では、選択基準を設置するために F\$CONTEXT が 3 回呼び出されています。最初の呼び出しは、クラスタ内の全ノードを探索するよう指定しています。次の呼出しでは、ユーザ名が“M”で始まるか“SYSTEM”であるプロセスのみを対象とすることを指定しています。最後の呼出しでは、SYSPRV と OPER 特権の両方を持っているプロセスを選択することを指定しています。

ラベル“loop”と“endloop”で囲まれたコマンド行は、F\$CONTEXT で設定した選択基準に合うプロセスを処理するため連続して F\$PID を呼んでいます。各 PID を検索した後、そのプロセスが実行しているイメージ名を得るため F\$GETJPI を呼び出しています。最後に、このプロシージャはイメージの名前を表示します。

エラーの場合、または CTRL/Y が押された場合は、制御はerrorに移され、(必要であれば)コンテキストがクローズされます。ここで、シンボル・タイプが PROCESS_CONTEXT であるか検査している点に注意してください。シンボル・タイプが PROCESS_CONTEXT であれば、選択基準は F\$CONTEXT を使用して取り消さなければなりません。シンボル・タイプが PROCESS_CINTEXT ではない場合は、選択基準が設定されなかったか、または、F\$PID 実行中にエラーが起きたかプロセス・リストが全て使用されたか、のいずれかです。

2.

```
f$context("process",ctx,"prcnam  ", "symbiont*",mcote*", "eq1")
f$context("process",ctx,"prcnam  ", "symbiont*",mcote*  ", "neq")
f$context("process",ctx,"prcnam  ", "mcote*  ", "neq")
f$context("process",ctx,"prcnam  ", "symbiont*", "neq")
```

この例では EQL と NEQ 選択値の違いを表す構文が 3 つ示してあります。最初の関数 (EQL を指定) はプロセス名に symbiont と mcote を含むすべてのプロセスを

戻します。2 番目と 3 番目の関数 (NEQ を指定) は全てのプロセス (プロセス名に symbiont がない , または mcote がないプロセス) を戻すことに相当します。

F\$CSID

OpenVMS Cluster システムから識別番号を戻し、システムのノード・リストの中で現在の位置を示すように、コンテキスト・シンボルを変更します。

フォーマット

F\$CSID(コンテキスト・シンボル)

戻り値

システムのノード・リストの中の、1つのクラスタ識別番号を示す文字列。クラスタ・システムでない場合は、空文字列。最後のシステム・クラスタ識別番が戻されると、F\$CSID 関数は空文字列("")を戻します。

引数

コンテキスト・シンボル

システムのノード・リストを示すポインタを格納するために、DCL が使用するシンボルを指定します。F\$CSID 関数は、このポインタを使用してクラスタ識別番号を戻します。

シンボルを使用してコンテキスト・シンボル引数を指定します。最初に F\$CSID を呼び出す時は、指定したシンボルは未定義または空文字列でなければなりません。

コンテキスト・シンボル引数が未定義または空文字列の場合、F\$CSID 関数は、システムのノード・リストの最初のシステムの識別番号を返します。これ以降の呼出しでは、残りのクラスタ内のノードの識別番号が順次返されます。

説明

F\$CSID 関数は、クラスタ識別番号を示す文字列を返し、システムのノード・リストで現在の位置を示すコンテキスト・シンボルを更新します。

現在使用しているシステムがクラスタのメンバではない場合は、最初の戻り値は空文字です。

F\$CSID 関数を使用すると、システム上のすべてのクラスタ識別番号を取り出せます。F\$GETSYI 関数を使用すると、特定のノードの情報を取り出せます。

最初の F\$CSID 関数呼出しで、コンテキスト・シンボル引数が初期化され、後続の F\$CSID 関数呼出しはクラスタ内の他のノードのクラスタ識別番号を戻します(クラスタ識別番号は、ランダムな順序で戻されます)。リスト内の最後のクラスタ識別番号を戻した後は、空文字列が戻されます。

例

```
1. $ IF F$GETSYI("CLUSTER_MEMBER") .EQS. "FALSE" THEN GOTO NOT_CLUSTER
   $ CONTEXT = ""
   $START:
   $   id = F$CSID (CONTEXT)
   $   IF id .EQS. "" THEN EXIT
   $   nodename = F$GETSYI ("NODENAME",,id)
   $   WRITE SYS$OUTPUT nodename
   $   GOTO start
   $NOT_CLUSTER:
   $ WRITE SYS$OUTPUT "Not a member of a cluster."
   $ EXIT
```

このコマンド・プロシージャは、F\$CSID 関数を使用してクラスタ内のシステム名を表示しています。代入式で、シンボル CONTEXT を宣言します。シンボル CONTEXT は、F\$CSID 関数のコンテキスト・シンボル引数として使用します。CONTEXT には空文字列が割り当てられているので、F\$CSID 関数は、クラスタ・ノード・リストの最初のクラスタ識別番号を戻します。

F\$CSID 関数が空文字列を返すのは、コマンド・プロシージャがリストの終わりであるか、またはクラスタではないノードでこの操作を行おうとしているか、のどちらかの場合です。F\$GETSYI 呼出しは、現在使用しているノードがクラスタのメンバか否かを確認します。コマンド・プロシージャは、この状態値で終了します。

F\$CSID 関数が空文字列を返さなかった場合、このコマンド・プロシージャは F\$GETSYI 関数の 3 番目の引数に識別番号を指定して、システム名を得ます。WRITE コマンドを使用すると、このシステム名が表示されます。

F\$CVSI

文字列データからビット・フィールドを取り出し、その結果を符号付き値として整数に変換します。

フォーマット

F\$CVSI(開始ビット, ビット数, 文字列)

戻り値

取り出したビット・フィールドを符号付き値として変換した整数

引数

開始ビット

取り出される最初のビットのオフセットを指定します。オフセットは、文字列の下位(右端)ビットを位置番号 0 として判断します。オフセットは整数式として指定します。

負の値を持つ式を指定した場合や、文字列のビット数を越える値を持つ式を指定した場合には、DCL は INVRANGE というエラー・メッセージを表示します。

ビット数

整数値に変換するために取り出されるビットの長さを指定します文字列のビット数以下でなければなりません。

負の値を持つ式を指定した場合や、ビット位置オフセットに加算したときに、値が無効になる式を指定した場合には、DCL は INVRANGE というエラー・メッセージを表示します。

文字列

ビットが取り出される文字列を指定します。文字列は、文字列式として指定します。

例

```
1. $ A[0,32] = %X2B
$ SHOW SYMBOL A
A = "+..."
$ X = F$CVSI(0,4,A)
$ SHOW SYMBOL X
X = -5   Hex = FFFFFFFB   Octal = 177773
```

この例では、算術オーバーレイを使用して、2B という 16 進数を、A というシンボルの 32 ビットすべてに割り当てています。算術オーバーレイについての詳細は、割り当て文(=)の説明を参照してください。

シンボル A は未定義状態だったため、オーバーレイが実行されたあと、このシンボルに文字列値が割り当てられます。シンボルが未定義の場合、算術オーバーレイを実行すると、その結果として文字列値が割り当てられます。シンボルがすでに定義されている場合には、オーバーレイを実行したあと、そのシンボルは、同じデータ・タイプのままです。2B という 16 進数は、ASCII のプラス記号(+)に対応します。

次に F\$CVSI 関数は、シンボル A から下位 4 ビットを取り出します。下位 4 ビットには、B という 16 進数の 2 進表現が含まれています。これらのビットが、符号付き値として整数に変換されます。変換後の値である -5 が、シンボル X に割り当てられます。

```
2. $ SYM[0,32] = %X2A
$ SHOW SYMBOL SYM
SYM = "*..."
$ Y = F$CVSI(0,33,SYM)
%DCL-W-INVRANGE, field specification is out of bounds - check sign and size
$ SHOW SYMBOL Y
%DCL-W-UNDSYM, undefined symbol - check spelling
```

この例では F\$CVSI 関数に指定された幅引数が大きすぎたため、DCL がエラー・メッセージを実行しシンボル Y に値を割り当てていません。

F\$CVTIME

絶対時刻，または複合時刻文字列を *yyyy-mm-dd hh:mm:ss.cc* という形式の文字列に変換します。また F\$CVTIME 関数は，絶対時刻，デルタ時間，またはそれらの組み合わせの文字列に関する情報も戻します。

フォーマット

F\$CVTIME([入力時刻][, 出力時刻形式[, フィールド])

戻り値

要求された情報を含む文字列。

引数

入力時刻

絶対時刻，デルタ時間，または複合時刻を含む文字列，あるいは TODAY，TOMORROW，YESTERDAY のいずれかを指定します。入力時刻文字列は，文字列式として指定します。

入力時刻引数を省略した場合，または空文字列("") が指定された場合は，絶対時刻形式での現在のシステムの日付が使用されます。日付けフィールドの一部が省略された場合は，省略された値は現在の日付の対応する値になります。時刻フィールドの一部が省略された場合は，省略された値は 0 になります。

時刻値の指定についての詳細は，『OpenVMS ユーザーズ・マニュアル』，またはオンライン・ヘルプのトピック Date を参照してください。

入力時刻引数がデルタ時間の場合は，出力時刻形式引数として DELTA を指定しなければなりません。

出力時刻形式

戻される情報の時刻形式を指定します。出力時刻形式引数は，次のいずれかの文字列を指定します。

ABSOLUTE	要求された情報が，絶対時刻形式 (<i>dd-mmm-yyyy hh:mm:ss.cc</i>) で戻されるように指定します。 <i>dd</i> が 1 桁の場合，先行の 0 や空白文字は戻されません。
----------	--

COMPARISON (省略時の設定)	要求された情報が， <i>yyyy-mm-dd hh:mm:ss.cc</i> という形式で戻されるように指定します。
------------------------	---

DELTA 要求された情報が、デルタ形式`dddd-hh:mm:ss:cc`で戻されるように指定します。出力時刻形式引数として DELTA を指定する場合には、入力時刻引数に対してもデルタ時間を指定しなければなりません。

出力フィールド

次のいずれか 1 つ (短縮できません) を含む文字列式を指定します。DATE, MONTH, DATETIME (省略時の設定), SECOND, DAY, TIME, HOUR, WEEKDAY, HUNDREDTH, YEAR, MINUTE, DAYOFYEAR, HOUROFYEAR, MINUTEOFYEAR, SECONDOFYEAR。

情報は、出力時刻形式引数に指定した時刻形式で戻されます。

入力時刻引数がデルタ時間で、出力時刻形式引数が DELTA の場合は、MONTH, WEEKDAY, YEAR, DAYOFYEAR, HOUROFYEAR, MINUTEOFYEAR, または SECONDOFYEAR は指定できません。

曜日が戻される場合は、最初の文字は大文字でそれ以降の文字は小文字です。

説明

F\$CVTIME 関数を使用する場合、(最後の引数の右に指定する) オプションの引数は省略できますが、最後に指定する引数の左の引数を省略する場合は、コンマ(,)は省略できません。

絶対時刻、または複合時刻形式で入力時刻引数を指定した場合は、出力時刻形式引数として ABSOLUTE や COMPARISON は指定できますが、DELTA は指定できません。

デルタ時間形式で入力時刻引数を指定した場合は、出力時刻形式引数として DELTA を指定しなければなりません。

例

```
1. $ TIME = F$TIME()
   $ SHOW SYMBOL TIME
      TIME = "14-DEC-2002 10:56:23.10"
   $ TIME = F$CVTIME(TIME)
   $ SHOW SYMBOL TIME
      TIME = "2002-12-14 10:56:23.10"
```

この例では F\$TIME 関数を使用して、文字列としてシステム時刻を戻し、その時刻を TIME というシンボルに割り当てています。次に、F\$CVTIME 関数を使用して、システム時刻を別の時刻形式に変換しています。TIME はシンボルなので、この引数を引用符(" ")で囲む必要はありません。レキシカル関数に対する引数としてシンボルが使用された時、シンボルは自動的に評価されます。

この結果、求められた文字列を使って、2つの日付を比較できます(.LTS. 演算子と.GTS. 演算子を使用します)。たとえば、F\$CVTIMEを使用すれば、2つの時刻文字列を変換し、その結果をTIME_1とTIME_2というシンボルに格納できます。そのあと、2つの値を比較し、次に示されているように結果にもとづいてラベルに分岐させることができます。

```
$ IF TIME_1 .LTS. TIME_2 THEN GOTO FIRST
```

2.

```
$ NEXT = F$CVTIME("TOMORROW", "WEEKDAY")  
$ SHOW SYMBOL NEXT  
NEXT = "Tuesday"
```

この例では、F\$CVTIME関数は“TOMORROW”という絶対時刻キーワードに対応する曜日を戻します。“TOMORROW”と“WEEKDAY”という引数は、引用符で囲まなければなりません。これらの引数は、文字列式だからです。また、省略する出力時刻形式引数に対しては、省略したことを示すために、プレースホルダとしてコンマを指定しなければなりません。

3.

```
$ SHOW TIME  
27-MAR-2002 09:50:31  
$ WRITE SYS$OUTPUT F$CVTIME(, "DAYOFYEAR")  
86  
$ WRITE SYS$OUTPUT F$CVTIME(, "HOUROFYEAR")  
2049  
$ WRITE SYS$OUTPUT F$CVTIME(, "MINUTEOFYEAR")  
122991  
$ WRITE SYS$OUTPUT F$CVTIME(, "SECONDOFYEAR")  
7379476
```

この例では、F\$CVTIMEは次のようなキーワードの値を戻します。
DAYOFYEAR, HOUROFYEAR, MINUTEOFYEAR, SECONDOFYEAR。

F\$CVUI

文字データからビット・フィールドを取り出し、その結果を符号なし値として整数に変換します。

フォーマット

F\$CVUI(開始ビット, ビット数, 文字列)

戻り値

取り出したビット・フィールドを符号なし値として変換した整数

引数

開始ビット

取り出される最初のビットのオフセットを指定します。文字列の下位(右端)ビットは、オフセットでの位置番号 0 です。オフセットは整数式で指定します。

負の値を持つ式や、文字列に含まれるビット数を越える値を持つ式を指定した場合、DCL は INVRANGE というエラー・メッセージを表示します。

ビット数

整数値に変換するために取り出されるビット文字列の長さを指定します。ここで指定する値は、文字列引数のビット数以下でなければなりません。

負の値を持つ式を指定した場合や、ビット位置オフセットに加算すると誤った値になる式を指定した場合は、DCL は INVRANGE というエラー・メッセージを表示します。

文字列

編集する文字列を指定します。

例

```
1. $ A[0,32] = %X2B
   $ SHOW SYMBOL A
     A = "+"
   $ X = F$CVUI(0,4,A)
   $ SHOW SYMBOL X
     X = 11   Hex = 0000000B   Octal = 00000000013
```

この例では、算術オーバーレイを使用して、シンボル A の 32 ビットすべてに 16 進数の 2B を割り当てています。シンボル A は未定義状態のため、オーバーレイが実行されたあと、このシンボルには、文字列値が割り当てられます。(シンボルが未定義の場合には、算術オーバーレイの結果として文字列値が割り当てられます。シンボルがすでに定義されていた場合には、オーバーレイが実行されたあと、そのシンボルは同じデータ・タイプのままです。) 16 進数の 2B は、ASCII の“+”に対応します。

次に F\$CVUI 関数がシンボル A から下位 4 ビットを取り出します。下位 4 ビットには、16 進数値 B の 2 進表現が含まれています。これらのビットは、符号なし値として整数に変換されます。変換後の 11 という値が、シンボル X に割り当てられます。

F\$DELTA_TIME

指定した時刻から終了時刻までの時間の差を戻します。終了時刻は開始時刻と同じかそれより遅い時刻でなければなりません。

フォーマット

F\$DELTA_TIME(開始時刻, 終了時刻)

戻り値

開始時刻から終了時刻までの差を示す文字列。戻される文字列の形式は次のとおりです。
dddd hh:mm:ss.cc

引数

開始時刻

開始時刻を次のような形式の絶対時刻表記で指定します。

dd-mmm-yyyy hh:mm:ss.cc

終了時刻

終了時刻を次のような形式の絶対時刻表記で指定します。

dd-mmm-yyyy hh:mm:ss.cc

例

```
1. $ START=F$TIME()  
   $ END=F$TIME()  
   $ SHOW SYMBOL START  
     START = "15-JUL-2003 16:26:35.77"  
   $ SHOW SYMBOL END  
     END = "15-JUL-2003 16:26:41.39"  
   $ WRITE SYS$OUTPUT F$DELTA_TIME(START,END)  
     0 00:00:05.62
```

この例では、レキシカル関数 F\$TIME() を使用して開始時刻と終了時刻のシンボルを定義しています。その後、F\$DELTA_TIME を使用して開始時刻と終了時刻の差を表示しています。

F\$DEVICE

選択基準に合う装置名をすべて返します。

装置名はランダムな順序で返されます。

フォーマット

F\$DEVICE([検索する装置名], [装置クラス], [装置タイプ], [ストリーム id])

戻り値

システム装置リストの装置名文字列。最後の装置名文字列が返された後は、空文字列("")が返されます。

引数

検索する装置名

検索する装置名を文字列で指定します。ワイルドカード(*と%)文字を使用できません。

検索する装置名引数は、文字列式として指定します。

装置クラス

検索する装置クラスを指定します。装置クラス引数は、有効な装置クラス名に対応する文字列式として指定します。

詳細は、\$GETDVI システム・サービスの DVI\$_DEVCLASS 項目を参照してください。

装置タイプ

検索する装置タイプを指定します。装置タイプ引数は、有効なタイプ名に対応する文字列式として指定します。

詳細は、\$GETDVI システム・サービスの DVI\$_DEVTYPE 項目を参照してください。

注意

装置クラスを指定せずに装置タイプを指定すると、エラーになります。

ストリーム id

検索ストリームを示す正の整数値

F\$DEVICE 関数を 2 回以上使用し、異なる選択基準引数を指定する場合、検索ストリーム識別番号を使用して、それぞれの検索コンテキストを管理します。コマンド・プロシージャの中で、F\$DEVICE 関数を 2 回以上使用し、異なる選択基準指定引数を指定した場合には、各検索を別々に識別するために、ストリーム id 引数を指定しなければなりません。

装置名リストを最後まで読み出す前に選択基準を変更した場合は、コンテキストは再初期化され検索はまた先頭から開始されます。

ストリーム id 引数を省略すると、F\$DEVICE 関数は検索ストリームが 1 つであると解釈します。つまり、異なる選択基準引数を指定するたびに、先頭から検索を開始します。

説明

F\$DEVICE 関数を使用すると、\$DEVICE_SCAN システム・サービスを使用することにより、ある選択基準を満たす装置を検索できます。

F\$DEVICE 関数では、装置名に基づいた検索に限り、アスタリスク(*)およびパーセント記号(%)ワイルドカード文字を使用できます。装置クラス、または装置タイプを指定する場合は、有効な文字列式を指定しなければなりません。

コマンド・プロシージャのループ内で F\$DEVICE 関数を使用して、指定した選択基準に対応する装置名を戻すことができます。F\$DEVICE 関数を実行する毎に、基準に合った次の装置名を戻します。装置名はランダムな順序で戻されます。基準にあった装置をすべて戻すと、F\$DEVICE 関数は空文字列を戻します。

明示的に(ストリーム id 引数を指定して)、または暗黙に(ストリーム id 引数を省略して)、検索文字列のコンテキストを管理しなければなりません。どちらの場合も、同一のストリーム(明示的あるいは暗黙的)で F\$DEVICE システム・サービスを実行する毎に同じ選択基準を指定しなければなりません。

例

1.

レキシカル関数 F\$DEVICE

```
$ START:
$   DEVICE_NAME = F$DEVICE("*0:", "DISK", "RA60")
$   IF DEVICE_NAME .EQS. "" THEN EXIT
$   SHOW SYMBOL DEVICE_NAME
$   GOTO START
```

ユニット番号 0 の RA60 をすべて表示するコマンド・プロシージャです。

ストリーム id 引数が指定されていないので、F\$DEVICE は暗黙の検索ストリームを使用します。選択基準を変更していないので、装置名が順番に返されます。選択基準に合った最後の装置名を表示した後、F\$DEVICE 関数は空文字列を返しコマンド・プロシージャを終了します。

F\$DIRECTORY

現在の省略時のディレクトリ名文字列を戻します。F\$DIRECTORY 関数には引数はありませんが、括弧を指定しなければなりません。

フォーマット

F\$DIRECTORY ()

戻り値

現在の省略時のディレクトリ名をかぎ括弧 ([]) も含めて示す文字列。SET DEFAULT コマンドで、ディレクトリ指定に不等号 (<>) を指定した場合には、F\$DIRECTORY 関数はディレクトリ文字列の中に不等号を含めて戻します。

引数

なし

説明

F\$DIRECTORY 関数を使用すると、コマンド・プロシージャの現在の省略時のディレクトリ名を保存し、別の作業のために省略時のディレクトリを他のディレクトリに変更し、再び元の設定に戻すことができます。

例

```
1. $ SAVE_DIR = F$DIRECTORY()
   $ SET DEFAULT [CARLEN.TESTFILES]
   .
   .
   .
   $ SET DEFAULT 'SAVE_DIR'
```

この例は、現在の省略時のディレクトリ設定を保存するために、F\$DIRECTORY 関数を使用するコマンド・プロシージャの一部を示しています。割り当て文で、SAVE_DIR というシンボルに、現在のディレクトリの値を割り当てます。次に SET DEFAULT コマンドで、新しい省略時のディレクトリを設定します。このあ

と、SET DEFAULT コマンドで SAVE_DIR シンボルを使用することにより、もとの省略時のディレクトリに戻しています。

F\$ENVIRONMENT 関数で DEFAULT キーワードを使用すれば、省略時のディスクとディレクトリをもとに戻すことができます。複数のディスクを含む場合には、F\$DIRECTORY 関数ではなく F\$ENVIRONMENT 関数を使用します。

F\$EDIT

編集リスト引数に指定された編集をもとに、文字列式を編集します。

フォーマット

F\$EDIT(文字列, 編集リスト)

戻り値

指定された編集を含む文字列。

引数

文字列

編集される文字列を指定します。引用符で囲まれた部分は編集されません。

編集リスト

編集リスト文字列に対して実行される編集のタイプを指定する、1 つまたは複数のキーワードを含む文字列を指定します。

編集	操作
COLLAPSE	すべてのスペースとタブを文字列から削除します。
COMPRESS	複数のスペースとタブを 1 つのスペースに変換します。
LOWERCASE	文字列を小文字に変換します。
TRIM	先行スペースとタブ、および後続のスペースとタブを文字列から削除します。
UNCOMMENT	コメントを文字列から削除します。
UPCASE	文字列を大文字に変換します。

2 つ以上のキーワードを指定する場合は、コンマ(,)で区切ります。キーワードを短縮することはできません。

文字列の中で引用符で囲まれた部分には、編集は適用されません。したがって、文字列に引用符(" ")が含まれている場合には、引用符で囲まれている文字は編集リストに指定されている編集の影響を受けません。

注意

編集リストに LOWERCASE と同時に UPCASE を指定した場合、UPCASE が優先されます。

例

```
1. $ LINE = "  THIS  LINE  CONTAINS A " QUOTED " WORD"
$ SHOW SYMBOL LINE
   LINE = "  THIS  LINE  CONTAINS A " QUOTED " WORD"
$ NEW_LINE = F$EDIT(LINE, "COMPRESS, TRIM")
$ SHOW SYMBOL NEW_LINE
   NEW_LINE = "THIS LINE CONTAINS A " QUOTED " WORD"
```

この例では、F\$EDIT 関数を使用して、複数のブランクは 1 つのブランクに変換され、先行ブランクと後続ブランクを削除することにより、文字列が短縮されています。この LINE という文字列には、引用符が含まれており、QUOTED という単語を囲んでいます (文字列に引用符を入力するには、割り当て文で二重引用符を使用しなければなりません)。

F\$EDIT 関数は、文字列の中で引用符で囲まれた部分ではスペースを短縮しません。したがって、スペースは QUOTED という単語を囲んだままになります。

```
2. $ LOOP:
$   READ/END_OF_FILE = DONE INPUT_FILE RECORD
$   RECORD = F$EDIT(RECORD, "TRIM, UPGASE")
$   WRITE OUTPUT_FILE RECORD
$   GOTO LOOP
.
.
.
```

この例では、ファイルからレコードを読み込み、編集し、出力ファイルに書き込むという作業を繰り返すためのループが設定されています。編集後のレコードからは、先行ブランクと後続のブランクが削除されており、文字列は大文字に変換されています。

```
3. $ UNCOMMENT_LINE = F$EDIT("$ DIR ! THIS IS THE COMMENT", "UNCOMMENT")
$ SHOW SYMBOL UNCOMMENT_LINE
$ UNCOMMENT_LINE = "$ DIR"
```

この例ではコメントを削除するために F\$EDIT 関数を使用しています。

F\$ELEMENT

指定した区切り文字で区切られた要素で構成される文字列から、1つの要素を取り出します。

フォーマット

F\$ELEMENT(要素番号, 区切り文字, 文字列)

戻り値

指定された要素を含む文字列。

引数

要素番号

取り出される要素の番号を指定します(要素番号は0から始まります)。要素番号引数は、整数式として指定します。要素番号が文字列に含まれる要素の数より大きい場合には、区切り文字が戻されます。

区切り文字

文字列に含まれる要素を区切るために、使用する文字を指定します。区切り文字は、文字列式として指定します。

文字列

区切り文字で区切られた要素のリストを含む文字列を指定します。文字列は、文字列式として指定します。

例

```
1. $ DAY_LIST = "MON/TUE/WED/THU/FRI/SAT/SUN"
$ INQUIRE DAY "ENTER DAY (MON TUE WED THU FRI SAT SUN) "
$ NUM = 0
$ LOOP:
$     LABEL = F$ELEMENT(NUM,"/",DAY_LIST)
$     IF LABEL .EQS. "/" THEN GOTO END
$     IF DAY .EQS. LABEL THEN GOTO 'LABEL'
$     NUM = NUM +1
$     GOTO LOOP
$
$ MON:
$
$
$
$
```

この例では、リストに含まれる要素のそれぞれと、入力値を比較するためのループが設定されています。DAY に対する値がDAY_LISTの1要素と一致する場合には、制御は対応するラベルに移ります。F\$ELEMENT 関数から戻される値が区切り文字の場合には、DAY の値はDAY_LISTに含まれていないため、制御はEND というラベルに移ります。

```
2. $ ! INDEX.COM
$ !
$ CHAPTERS = "0,1,2,3,4,5,6,A,B,C"
$ NEXT = 0
$ LOOP:
$     NEXT = NEXT + 1
$     NUM = F$ELEMENT(NEXT,"",CHAPTERS)
$     IF (NUM .NES. ",")
$     THEN
$         RUN INDEX CHAP'NUM'
$         GOTO LOOP
$     ENDIF
$
```

この例ではファイルにCHAP1, CHAP2, ... CHAP6, CHAPA, CHAPB, CHAPC という順に名前を付けるプロセスを示しています。プロシージャ・ロジックを初期化状態にするため、ゼロはCHAPTERS という文字列に含まれます。NEXT はゼロに初期化されます。このプロシージャはループを実行します。最初の繰り返してNEXT は1に増え、その結果“1”を呼び出します。次にプロシージャはindex, chapter1 を実行します。2回めの繰り返してNEXT は2に増え、その結果“1”を呼び出します。さらにプロシージャはindex, chapter2 を実行し、指定した区切り文字を呼び出すまでプロセスは続きます。

F\$ENVIRONMENT

DCL コマンド環境に関する情報を戻します。

フォーマット

F\$ENVIRONMENT(項目)

戻り値

指定された項目に対応する情報。戻される値は、指定される項目に応じて整数または文字列になります。

引数

項目

戻される情報のタイプを指定します。次に示すキーワードのいずれか 1 つを指定します (キーワードを短縮することはできません)。

項目	データ・ タイプ	戻される情報
CAPTIVE	文字列	CAPTIVE 属性を持ったアカウントにログインしている場合は、TRUE を戻します。システム管理者は、Authorize ユーティリティ (AUTHORIZE) を使用して、利用者登録ファイルにも CAPTIVE アカウントを定義できます。
CONTROL	文字列	SET CONTROL コマンドによって現在使用可能になっている制御文字を戻します。複数の制御文字が使用可能な場合には、各文字はコンマで区切られます。制御文字が使用可能ではない場合には、空文字列 ("") が戻されます。
DEFAULT	文字列	現在の省略時の装置、およびディレクトリ名を戻します。戻される文字列は SHOW DEFAULT コマンドにより出力される文字列と同じです。
DEPTH	整数値	現在のコマンド・プロシージャのネスティングの中での深さを、整数として戻します。会話型でコマンド・プロシージャを実行する場合、およびバッチ・ジョブでコマンド・プロシージャを実行する場合、コマンド・プロシージャの深さは 0 です。ネスティングされたコマンド・プロシージャの深さは、そのネスティングされたプロシージャを実行したコマンド・プロシージャの深さより、1 だけ大きな値です。

項目	データ・ タイプ	戻される情報
DISIMAGE	文字列	イメージを直接起動することが許可されていないアカウント(たとえば RUN などは許可されていません)にログインした場合、TRUE を戻します。システム管理者は Authorize ユーティリティを使用して、利用者登録ファイル内のアカウントに DISIMAGE 属性を追加または削除できます。
INTERACTIVE	文字列	プロセスが会話型で実行されている場合は、TRUE を戻します。
KEY_STATE	文字列	現在ロックされているキーボード状態を示す文字列を戻します。キーボード状態についての詳細は、DEFINE/KEY コマンドの説明を参照してください。
MAX_DEPTH	整数値	コマンド・プロシージャの最大の深さを示す整数を戻します。
MESSAGE	文字列	SET MESSAGE コマンドによる現在の設定を示す文字列を戻します。この文字列には、修飾子名を区切るために、スラッシュ(/)が含まれています。したがって、F\$ENVIRONMENT("MESSAGE")からの出力を、SET MESSAGE コマンドのうしろに追加することにより、有効な DCL コマンド行を作成できます。
NOCONTROL	文字列	SET NOCONTROL コマンドによって、現在禁止されている制御文字を戻します。複数の文字が禁止されている場合には、各文字はコンマ(,)で区切られます。SET NOCONTROL コマンドによって制御文字が禁止されていない場合には、空文字列が戻されます。
ON_CONTROL_Y	文字列	コマンド・プロシージャから実行される場合は、ON_CONTROL_Y が設定されていれば TRUE を戻します。DCL コマンド・レベルでは、ON_CONTROL_Y は常に FALSE を戻します。
ON_SEVERITY	文字列	コマンド・プロシージャから実行される場合は、ON コマンドで指定した動作が実行される重要度を戻します。SET NOON が有効な場合、または DCL コマンド・レベルで使用した場合、ON_SEVERITY は NONE を戻します。
OUTPUT_RATE	文字列	省略時の出力速度を含むデルタ時間文字列を戻します。これは、バッチ・ジョブの実行中に、バッチ・ジョブ・ログ・ファイルにデータが書き込まれる頻度を示します。会話型で使用了した場合、OUTPUT_RATE は空文字列を戻します。
PROCEDURE	文字列	現在のコマンド・プロシージャのファイル指定を戻します。会話型で使用了場合は、端末装置名が戻されます。
PROMPT	文字列	現在の DCL プロンプトを戻します。
PROMPT_CONTROL	文字列	プロンプトの前に改行が挿入される場合は TRUE を戻します。
PROTECTION	文字列	現在の省略時のファイル保護を示す文字列を戻します。文字列は有効な DCL コマンド行となるように、SET PROTECTION/DEFAULT コマンドで使用できる形式です。

項目	データ・ タイプ	戻される情報
RESTRICTED	文字列	制限付アカウントの場合は TRUE を戻します。システム管理者は、Authorize ユーティリティを使用して、利用者登録ファイルに制限付アカウントを定義できます。
SYMBOL_SCOPE	文字列	現在のシンボルの定義範囲を示すため、文字列[NO] LOCAL, [NO] GLOBAL を戻します。
VERB_SCOPE	文字列	動詞の現在のシンボルの定義範囲を示すため、文字列[NO] LOCAL, [NO] GLOBAL を戻します (詳細は、SET SYMBOL コマンドの説明を参照してください)。
VERIFY_IMAGE	文字列	SET VERIFY=IMAGE コマンドが有効である場合は、TRUE を戻します。イメージ・チェックが有効な場合には、コマンド・プロシージャは、イメージによって読み込まれた入力データを表示します。
VERIFY_PREFIX	文字列	SET PREFIX コマンドで設定される前置制御文字列を戻します。
VERIFY_PROCEDURE	文字列	SET VERIFY=PROCEDURE コマンドが有効である場合は、TRUE を戻します。コマンド・チェックが有効な場合には、コマンド・プロシージャは、DCL コマンド行を表示します。

例

```
1. $ SAVE_MESSAGE = F$ENVIRONMENT("MESSAGE")
   $ SET MESSAGE/NOFACILITY/NOIDENTIFICATION
   .
   .
   .
   $ SET MESSAGE'SAVE_MESSAGE'
```

この例では、F\$ENVIRONMENT 関数を使用して、現在のメッセージ設定を変更する前に、その設定を保存します。コマンド・プロシージャの最後で、もとのメッセージ設定が復元されます。SAVE_MESSAGE というシンボルを囲む一重引用符(' ')は、そのシンボル名を値と置き換えるように指定しています。

```
2. $ MAX = F$ENVIRONMENT("MAX_DEPTH")
   $ SHOW SYMBOL MAX
   MAX = 32   Hex = 00000020   Octal = 00000000040
```

この例では、コマンド・プロシージャ内で可能な最大のネスティングの深さを判断するために、F\$ENVIRONMENT 関数を使用しています。

レキシカル関数 F\$ENVIRONMENT

```
3. $ SAVE_PROT = F$ENVIRONMENT("PROTECTION")
   $ SET PROTECTION = (SYSTEM:RWED, OWNER:RWED, GROUP, WORLD)/DEFAULT
   .
   .
   .
   $ SET PROTECTION = ('SAVE_PROT')/DEFAULT
```

この例では、F\$ENVIRONMENT 関数を使用して、保護を変更する前に、現在の省略時の保護を保存します。コマンド・プロシージャの最後で、もとの保護に戻されます。シンボル置換を要求するためには、SAVE_PROT というシンボルを引用符で囲まなければなりません。

F\$EXTRACT

文字列から部分文字列を取り出します。

フォーマット

F\$EXTRACT(開始, 長さ, 文字列)

戻り値

開始引数と長さ引数によって区切られた部分文字列。

引数

開始

取り出したい部分文字列の先頭のオフセットを指定します。0以上の整数式として指定します。

オフセットは文字の相対位置、または文字列の先頭の部分文字列です。オフセット位置は0から始まります。文字列は、常に左端の文字から始まります。

文字列の長さ以上のオフセットを指定した場合には、空文字列("")を戻します。

長さ

取り出す文字数を指定します。文字列サイズ以下でなければなりません。長さは、0以上の整数式を指定します。

オフセットから文字列の最後まで文字数より大きな値の長さを指定すると、オフセットから文字列の最後まで文字を戻します。

文字列

編集する文字列を指定します。文字列には、文字列式を指定します。

例

1.

```
$ NAME = "PAOLO TESTA"
$ FIRST = F$EXTRACT(0,5,NAME)
$ SHOW SYMBOL FIRST
FIRST = "PAOLO"
```

この例はコマンド・プロシージャの一部であり、シンボル NAME に割り当てられた文字列から最初の 5 文字を取り出すために、F\$EXTRACT 関数を使用しています。オフセット引数と長さ引数は整数であり、文字列引数はシンボルです。レキシカル関数に対する引数として整数やシンボルを使用する場合には、これらを引用符 (" ") で囲む必要はありません。

2.

```
$ P1 = "MYFILE.DAT"
$ FILENAME = F$EXTRACT(0,F$LOCATE(".",P1),P1)
```

この例もコマンド・プロシージャの一部です。文字列からある 1 文字を見つけ、その位置で終わる部分文字列を取り出す方法を示しています。

F\$LOCATE 関数は、P1 に対応する文字列中のピリオドのオフセット位置を示す数値を戻します (ピリオドのオフセット位置は、ピリオドの前の部分文字列の長さと同じです)。

F\$EXTRACT 関数の引数として F\$LOCATE 関数を使用して、文字列から取り出す文字数を指定しています。パラメータ MYFILE.DAT を使用してプロシージャを呼び出し、これらのステートメントを実行すると、シンボル FILENAME に MYFILE という値が割り当てられます。

上記の例で F\$LOCATE 関数は、ファイル指定にサブディレクトリ名を含むディレクトリ指定や、ノード名が含まれていないとして解釈しています。完全なファイル指定からファイル名を取り出すためには、F\$PARSE 関数を使用します。

3.

```
$ IF F$EXTRACT(12,2,F$TIME()) .GES. "12" THEN GOTO AFTERNOON
$ MORNING:
$ WRITE SYS$OUTPUT "Good morning!"
$ EXIT
$ AFTERNOON:
$ WRITE SYS$OUTPUT "Good afternoon!"
$ EXIT
```

この例では、現在の時刻が午前であるか午後であるかに応じて、異なるメッセージを表示するプロシージャを示しています。最初に、F\$TIME 関数を使用して、現在の時刻が戻されます。F\$TIME 関数は文字列を戻し、この文字列が F\$EXTRACT 関数に対する文字列引数として使用されます。F\$TIME 関数が引数として使用されている場合には、この関数は自動的に評価されるので、引用符を使用する必要はありません。

次に F\$EXTRACT 関数は、F\$TIME から戻された日付と時刻の示された文字列から時を取り出します。F\$TIME から渡される文字列には、常に先頭から 12 文字のオフセットから時フィールドが含まれています。

F\$EXTRACT 関数は、このオフセットから始まる 2 文字を文字列から取り出し、取り出した文字列値と文字列値 12 を比較します。比較結果が真の場合には、プロシージャは“Good afternoon!”を出力します。比較結果が偽の場合には、“Good morning!”を出力します。

また、時刻指定から時フィールドを取り出すために、F\$CVTIME 関数を使用できます。F\$CVTIME 関数を使用する方が、上記の例に示されている方法に比べ簡単です。

F\$FAO

\$FAO システム・サービスを呼び出し、指定された制御文字列を、ASCII 形式に整えられた出力文字列に変換します。書式を指定して、整数値を文字列に変換したり、キャリッジ・リターンやフォーム・フィードを挿入したり、テキストを挿入したりできます。

フォーマット

F\$FAO(制御文字列[, 引数[,...]])

戻り値

ASCII 形式で出力される文字列。この出力文字列は、制御文字列の固定テキストと FAO ディレクティブから作成されます。

引数

制御文字列

出力文字列の固定テキストと FAO ディレクティブを結合したものです。制御文字列の長さは任意であり、FAO ディレクティブをいくつでも含むことができます。制御文字列では、文字列式として指定します。

F\$FAO 関数は、FAO ディレクティブを使用して、ASCII 形式のデータを変更したり、制御文字列中の固定テキストに割り当てたりします。

表 DCLI-4 は制御文字中で指定する FAO ディレクティブを示しています。

引数[,...]

制御文字列中の FAO ディレクティブに対する 1 ~ 15 の引数。引数は、整数式または文字列式として指定します。表 DCLI-4 は、各 FAO ディレクティブと必要な引数タイプを示しています。

FAO ディレクティブは 1 つまたは複数の引数を必要とすることがあります。その順序は、制御文字列のものと正確に一致しなくてはなりません。そうでない場合でも、エラー・メッセージは出力されません。

対応するディレクティブに対して指定可能なタイプ (整数または文字列) ではない引数を指定すると、予測できない結果が戻されます。F\$INTEGER や F\$STRING を使用して適当な型に変換してください。

引数に不足がある場合、F\$FAO は引数リストを越えて読もうとします。そのため、制御文字列のディレクティブに一致する引数を必ず指定してください。

対応するディレクティブに対して指定可能なタイプ (整数または文字列) ではない引数を指定すると、コマンドが実行されなかったという予測できないエラーになります。このエラーは、\$FAO システム・サービスからのものそのままです。

説明

\$FAO システム・サービスを起動し、文字および数値入力を ASCII 文字列に変換します (FAO は formatted ASCII output の略です)。書式を指定して、整数値を文字列に変換したり、キャリッジ・リターンやフォームフィードをテキストを挿入したり、テキストを挿入したりできます。

以下のフォーマットを使用して FAO ディレクティブを指定します。

フォーマット	機能
!DD	1 つのディレクティブです。
!n(DD)	指定した回数を繰り返します。
!lengthDD	指定した長さのフィールドに出力します。
!n(lengthDD)	指定した回数を繰り返し、指定した長さのフィールドに出力します。

感嘆符(!)は、それ以降の文字または文字列が FAO ディレクティブとして解釈されることを示しています。DD は F\$FAO が実行するためのアクションであることを示す、1 つまたは 2 つの大文字のコードを表しています。繰り返し数を指定する場合、*n* はディレクティブが繰り返される回数を指定する 10 進数です。*length* 値は、“length” 文字の出力フィールドを表わすために F\$FAO に指示する 10 進数です。

繰り返し回数と出力の長さは、絶対数ではなく、(#) を使用して指定することもできます。(#) を使用した場合は、引数リストの相当する位置に整数式として数値を指定します。

変数出力フィールドを繰り返し回数で指定すると、長さパラメータは 1 つだけ必要です。これは、各出力文字列はそれぞれ長さを指定しているからです。

FAO ディレクティブは以下のカテゴリでグループ化されます。

- 文字列挿入
- ゼロ埋め込み数値変換
- 空白埋め込み数値変換
- 特殊編集

- パラメータ解釈

表 DCLI-4 は、FAO ディレクティブと必須の引数タイプを示しています。次の節では文字列挿入、ゼロ埋め込み数値変換、および空白埋め込み数値変換を実行するディレクティブの出力文字列を説明します。

注意

\$FAO システム・サービスではサポートされていますが、DCL F\$FAO レキシカル関数ではサポートされていない 2 つのタイプのディレクティブがあります。その 2 つのタイプを次に示します。

- クォードワード数値ディレクティブ (Q, H, および J)。すべての DCL 数値はロングワードとして格納されるので、DCL ではサポートされません。
- !AS ディレクティブ以外の文字列ディレクティブ。すべての DCL 文字列は記述子として格納されるので、DCL ではサポートされません。

\$FAO システム・サービス・ディレクティブについての詳細は『OpenVMS System Services Reference Manual』を参照してください。

表 DCLI-4 FAO ディレクティブの要約

ディレクティブ	引数の型	説明
文字列挿入：		
!AS	文字列	文字列をそのまま挿入する。
ゼロ埋め込み数値変換：		
!OB	整数	バイトから 8 進数表現に変換する。
!OW	整数	ワードから 8 進数表現に変換する。
!OL	整数	ロング・ワードから 8 進数表現に変換する。
!XB	整数	バイトから 16 進数表現に変換する。
!XW	整数	ワードから 16 進数表現に変換する。
!XL	整数	ロング・ワードから 16 進数表現に変換する。
!ZB	整数	バイトから 10 進数表現に変換する。
!ZW	整数	ワードから 10 進数表現に変換する。
!ZL	整数	ロング・ワードから 10 進数表現に変換する。
空白埋め込み数値変換：		
!UB	整数	負数調整なしにバイトから 10 進数表現に変換する。
!UW	整数	負数調整なしにワードから 10 進数表現に変換する。
!UL	整数	負数調整なしにロング・ワードから 10 進数表現に変換する。

(次ページに続く)

表 DCLI-4 (続き) FAO ディレクティブの要約

ディレクティブ	引数の型	説明
!SB	整数	負数変換してバイトから 10 進数表現に変換する。
!SW	整数	負数変換してワードから 10 進数表現に変換する。
!SL	整数	負数変換してロング・ワードから 10 進数表現に変換する。
特殊編集：		
!/	None	キャリッジ・リターンとライン・フィードを挿入する。
!_	None	タブを挿入する。
!^	None	フォーム・フィードを挿入する。
!!	None	感嘆符(!)を挿入する。
!%I	整数	ロング・ワードの整数を名前形式の UIC 形式[グループ識別子, メンバ識別子]に変換する。
!%S	None	最近変換された数値が 1 でないとき "S" を挿入する。(マルチ・リンガル製品で使用することは推奨しません。)
!%U	整数	ロング・ワードの整数を数値 UIC 形式[g,m]に変換する。ここで g はグループ番号, m はメンバ番号。 ディレクティブが, かぎかっことカンマを挿入する。
!n<...!>	None	n 文字のフィールドに, すべてのデータを左詰めし, 残りを空白で埋める。
!n*c	None	c で表現される文字を n 回繰り返す。
!n%C	文字列	最も最近評価された引数の値が n の時, 文字列を挿入する(マルチ・リンガル製品で使用する)。
!%E	文字列	最も最近評価された引数が先行するどの !n%C ディレクティブにも一致しない時, 文字列を挿入する(マルチ・リンガル製品で使用する)。
!%F	None	複数形の終わりにマークする。
!%T	整数 0	現在時刻を挿入する。
!%D	整数 0	現在日付時刻を挿入する。
引数の解釈：		
!-	None	最後の引数を再使用する。
!+	None	次の引数をスキップする。

文字列挿入による出力文字列

!AS ディレクティブは, 制御文字列に(ディレクティブの引数として指定された)文字列を挿入します。制御文字列に文字列を挿入した場合, 文字列のフィールドの長さは, 文字列の長さになります(これが省略時の設定です)。省略時の長さが明示的に指定したフィールドの長さよりも短い場合は, 文字列は左詰めされ, 残りは空白で埋められます。省略時の長さが明示的に指定したフィールドの長さよりも長い場合は, 文字列の右端は切り捨てられます。

ゼロ埋め込み数値変換による出力文字列

ゼロ埋め込み数値変換のためのディレクティブは、(ディレクティブの引数として指定された)整数を、10進数、8進数、または16進数表現に変換します。整数のASCII表現は、制御文字列に挿入されます。変換された引数の省略時の出力フィールドの長さは、以下のように決められます。

- 引数を10進数表現に変換するディレクティブは、バイト変換では3桁、ワード変換では6桁、ロングワード変換では11桁を返します。数字は右詰めされ、左端はゼロで埋められます。省略時の長さより明示的に指定したフィールドの長さが長い場合は、左端は空白で埋められます。省略時の長さより明示的に指定したフィールドの長さが短い場合は、左端は切り捨てられます。
- 引数を8進数表現に変換するディレクティブは、バイト変換では2桁、ワード変換では4桁、ロングワード変換では8桁を返します。数字は右詰めされ、左端はゼロで埋められます。省略時の長さより明示的に指定したフィールドの長さが長い場合は、左端は空白で埋められます。省略時の長さより明示的に指定したフィールドの長さが短い場合は、左端は切り捨てられます。
- 引数を16進数表現に変換するディレクティブは、16進数で要求される数の文字を返します。省略時の長さより明示的に指定したフィールドの長さが長い場合は、左端はゼロで埋められます。省略時の長さより明示的に指定したフィールドの長さが短い場合は、出力フィールドはすべてアスタリスク(*)で埋められます。

バイト変換では引数の2進表現の下位8ビットのみを、ワード変換では下位16ビットのみを使用します。ロングワード変換では、32ビットすべてを使用します。

空白埋め込み数値変換による出力文字列

空白埋め込み数値変換のディレクティブは、(ディレクティブの引数として指定された)整数を10進数表現に変換します。これらのディレクティブは、符号付きあるいは符号なしとして整数を変換できます。整数のASCII表現は、制御文字列に挿入されます。

変換された引数の出力フィールドの長さは、必要な文字数です(これが省略時の設定です)。明示的に指定したフィールドの長さよりも値が短い場合は、右詰めされ残は空白で埋められます。明示的に指定したフィールドの長さよりも値が長い場合は、フィールドはアスタリスクで埋められます。

バイト変換では引数の2進表現の下位8のみを、ワード変換では下位16ビットのみを使用します。ロングワード変換では32ビットすべてを使用します。

特殊編集による出力文字列

`!n%C` および `!%E` ディレクティブは、(最後に評価された引数の値をもとにした)ASCII文字を出力文字列に挿入します。これらのディレクティブは、不規則な複数の名詞や動詞を挿入するときに便利です。

最後に評価された引数が n と等しい場合は、2つのディレクティブ間のテキストは出力文字列に挿入されます。最後に評価された引数が n と等しくない場合は、次の`!n%C`ディレクティブが実行されます。

n が負数の場合は、引数として指定し(＃)を使用しなくてはなりません。

!n%C および !%E ディレクティブを、繰り返し回数で指定することができます。繰り返し回数を指定すると、2つのディレクティブ間のテキストは指定された回数だけ出力文字列にコピーされます。

%F ディレクティブは複数の文の終わりを示します。

例

```
1. $ COUNT = 57
   $ REPORT = F$FAO("NUMBER OF FORMS = !SL",COUNT)
   $ SHOW SYMBOL REPORT
   REPORT = "NUMBER OF FORMS = 57"
```

このコマンド・プロシージャでは、制御文字列で !SL という FAO ディレクティブを使用することにより、COUNT というシンボルに割り当てられている値が、文字列に変換されます。変換後の文字列は、制御文字列に挿入されます。

COUNT には、57 という整数値が割り当てられています。したがって F\$FAO 関数は "NUMBER OF FORMS=57" という ASCII 文字列を戻し、この文字列を REPORT というシンボルに割り当てます。

```
2. $ A = "ERR"
   $ B = "IS"
   $ C = "HUM"
   $ D = "AN"
   $ PHRASE = F$FAO("TO !3(AS)",A,B,C+D)
   $ SHOW SYMBOL PHRASE
   $ PHRASE = "TO ERRISHUMAN"
```

このコマンド・プロシージャでは、A、B、C、および D というシンボルに割り当てられている値を、制御文字列に挿入するために、!AS ディレクティブが使用されています。

!AS ディレクティブに対して、3 という繰り返し回数が指定されているため、F\$FAO は 3 つの引数を調べます。この例では、引数にシンボル A ("ERR")、シンボル B ("IS")、および C + D という式 ("HUMAN") が含まれています。これらの文字列引数の値が連結され、"ERRISHUMAN" という文字列が作成されます。

```
3. $ A = "ERR"
   $ B = "IS"
   $ C = "HUMAN"
   $ PHRASE = F$FAO("TO !#(AS)",3,6,A,B,C)
   $ SHOW SYMBOL PHRASE
   $ PHRASE = "TO ERR IS HUMAN "
```

このコマンド・プロシージャでは、文字列を書式化するために、!AS ディレクティブを指定した F\$FAO 関数が使用されています。最初の番号記号(＃)は、最初の

引数である 3 によって与えられる繰り返し回数を示しています。2 番目の番号記号 (#) は、2 番目の引数である 6 が与えられるフィールド・サイズを示しています。次の 3 つの引数 (A, B, および C) は、!AS ディレクティブが繰り返されるたびに、制御文字列に挿入される文字列を示します。

各引数文字列は、6 文字の長さのフィールドに出力されます。各文字列は 6 文字より短いため、各フィールドは左桁ぞろえされ、右側に空白が挿入されます。この結果、作成された文字列は、PHRASE というシンボルに割り当てられます。

```
4. $ OFFSPRING = 1
   $ REPORT = F$FAO-
   ("There !0UL!1%Cis!%Eare!%F !-!UL !-!0UL!1%Cchild!%Echildren!%F here",OFFSPRING)
   $ SHOW SYMBOL REPORT
   $ REPORT ="There is 1 child here"
```

このコマンド・プロシージャでは、!0UL ディレクティブは引数 OFFSPRING を評価しますが、出力文字列に値を挿入しません。!n%C ディレクティブは、その値と引数 OFFSPRING の値が一致するので、“is”という文字列を出力文字列に挿入します。出力文字列の適切な位置に正確な文字が挿入されるよう修正するため、!-!UL ディレクティブは、2 回目の引数の評価をします。!%F ディレクティブは、それぞれの複数文の終わりを示しています。F\$FAO 関数は“There is 1 child here”という ASCII 文字列を返して、それを REPORT というシンボルに割り当てます。

F\$FID_TO_NAME (Alpha/I64 のみ)

ファイル識別子をファイル指定に変換します。

フォーマット

F\$FID_TO_NAME(装置名, ファイル ID)

戻り値

ファイル指定を含む文字列。

引数

装置名

ファイルが存在する装置を指定します。装置の論理名を指定することもできます。

ファイル ID

対応するファイル指定に変換するファイル ID を指定します。

例

1. \$ WRITE SYS\$OUTPUT F\$FID_TO_NAME("SYS\$SYSDEVICE", "(2901,33,0)")
DISK\$NODE1: [VMS\$COMMON.SYSEXEC] SHOW.EXE;1

この例では、システム・ディスクにある、ID が "2901,33,0" のファイルが SHOW.EXE であることを示しています。注意: ファイル ID が二重引用符で囲まれている場合には、括弧は省略できます。

F\$FILE_ATTRIBUTES

指定されたファイルの属性に関する情報を戻します。

フォーマット

F\$FILE_ATTRIBUTES(ファイル指定, 項目)

戻り値

要求した項目に応じて、整数または文字列となります。各項目の戻された値のデータ・タイプを表 DCLI-5 に示します。

引数

- ファイル指定
情報が必要なファイルの名前。ファイル名は文字列式として指定します。
- ファイル名は、1つしか指定することができません。ファイル指定の中で、ワイルドカード文字を使用することはできません。
- 項目
ファイルのどの属性を戻してほしいか指定します。項目は、文字列式として指定します。表 DCLI-5 の RMS フィールド名から 1 項目を指定します。

説明

DCL 割り当て文および式で使用して、ファイルの属性に関する情報を戻します。
表 DCLI-5 は F\$FILE_ATTRIBUTES 関数に指定する項目、戻される情報、その情報のデータ・タイプを示しています。

表 DCLI-5 F\$FILE_ATTRIBUTES 項目

項目	データ・ タイプ	戻される情報
AI	文字列	AI ジャーナリングが設定されていれば TRUE。設定されていなければ FALSE。

(次ページに続く)

表 DCLI-5 (続き) F\$FILE_ATTRIBUTES 項目

項目	データ・タイプ	戻される情報
ALQ	整数値	割り当てサイズ
BDT	文字列	バックアップ日付
BI	文字列	BI ジャーナリングが設定されていれば TRUE。設定されていなければ FALSE。
BKS	整数値	バケット・サイズ
BLS	整数値	ブロック・サイズ
CBT	文字列	contiguous-best-try 割り当てアルゴリズムが使用されていれば TRUE。使用されていなければ FALSE。
CDT	文字列	作成日付
CTG	文字列	contiguous 割り当てアルゴリズムが使用されていれば TRUE。使用されていなければ FALSE。
DEQ	整数値	省略時の拡張サイズ
DID	文字列	ディレクトリ ID
DIRECTORY	文字列	TRUE あるいは FALSE を返す。ディレクトリである場合は TRUE を返す。
DVI	文字列	装置名
EDT	文字列	満了日付
EOF	整数値	使用ブロック数
ERASE	文字列	削除時に物理的に削除される場合は TRUE。物理的に削除されない場合は FALSE。
FFB	整数値	先頭空きバイト
FID	文字列	ファイル ID
FILE_LENGTH_HINT	文字列	レコード数およびデータのバイト数を (n,m) の形式で返す。ここで n はレコード数で、 m はデータのバイト数である。無効な数は、 n あるいは m に -1 が返される。
FSZ	整数値	固定制御領域サイズ
GBC	整数値	グローバル・バッファ数
GRP	整数値	所有者グループ番号
JOURNAL_FILE	文字列	ジャーナル・ファイルの場合は TRUE。ジャーナル・ファイルではない場合は FALSE。
KNOWN	文字列	Known ファイルがインストール・ユーティリティ (INSTALL) とともにインストールされた場合は TRUE を、そうでない場合は FALSE を返す。ただし、ファイルが存在しない場合 (たとえば、ファイルがインストールされた直後に削除された場合) は、NOSUCHFILE を返す。
LOCKED	文字列	アクセスが許可されないロック状態の場合は TRUE。そうでない場合は FALSE。
LRL	整数値	最大レコード長
MBM	整数値	所有者メンバ番号

(次ページに続く)

表 DCLI-5 (続き) F\$FILE_ATTRIBUTES 項目

項目	データ・ タイプ	戻される情報
MOVE	文字列	movefile 操作が可能な場合は TRUE。そうでない場合は FALSE。
MRN	整数値	最大レコード数
MRS	整数値	最大レコード・サイズ
NOA	整数値	エリア数
NOBACKUP	文字列	バックアップ用にファイルがマークされている場合は FALSE。NOBACKUP のマークがついている場合は TRUE。
NOK	整数値	キー数
ORG	文字列	ファイル編成法 (SEQ, REL, IDX)
PRESHELVED (Alpha/I64 のみ)	文字列	ファイルがプリシェルブドである場合 TRUE。そうでない場合、FALSE。
PRO	文字列	ファイル保護文字列
PVN	整数値	プロログ・バージョン番号
RAT	文字列	レコード属性 (CR, PRN, FTN, "")
RCK	文字列	読み込みチェックが設定されている場合は TRUE。そうでない場合は FALSE。
RDT	文字列	最新更新日付
RFM	文字列	レコード・フォーマット (VAR, FIX, VFC, UDF, STM, STMLF, STMCR)
RU	文字列	RU ジャーナリングが設定されている場合は TRUE。設定されていない場合は FALSE。
RVN	整数値	改訂番号
SHELVABLE	文字列	ファイルがシェルブド可能な場合は TRUE。不可能な場合は FALSE。
SHELVED	文字列	ファイルがシェブルドの場合は TRUE。そうでない場合は FALSE。
STORED_ SEMANTICS	文字列	セマンティックスを表す ASCII 文字列
UIC	文字列	所有者 UIC 文字列
VERLIMIT	整数値	バージョン・リミット番号。32767 はバージョン・リミットが設定されていないことを示す。
WCK	文字列	書き込みチェックが設定されている場合は TRUE。設定されていない場合は FALSE。

ファイル属性は OpenVMS RMS 制御ブロックの情報から作られたファイル・ヘッダに格納されます。OpenVMS RMS 制御ブロックについての詳細は『OpenVMS Record Management Services Reference Manual』を参照してください。

例

1.

```
$ FILE_ORG = F$FILE_ATTRIBUTES("QUEST.DAT", "ORG")
$ SHOW SYMBOL FILE_ORG
FILE_ORG = "SEQ"
```

この例では、ファイル編成タイプの値を FILE_ORG というシンボルに割り当てるために、F\$FILE_ATTRIBUTES 関数が使用されています。F\$FILE_ATTRIBUTES 関数は、"SEQ"という文字列を戻します。これは QUEST.DAT が順編成ファイルであることを示します。

F\$FILE_ATTRIBUTES 関数に対する ORG 引数および QUEST.DAT 引数は、文字列リテラルであり、式の中で使用するときは二重引用符で囲まなければなりません。

2.

```
$ RFM = F$FILE_ATTRIBUTES("KANSAS::USE$:[CARS]SALES.CMD", "RFM")
$ SHOW SYMBOL RFM
RFM = "VAR"
```

この例では、リモート・ノードのファイルに関する情報を戻すために、F\$FILE_ATTRIBUTES 関数が使用されています。この関数は、VAR というレコード形式文字列を戻します。これはレコード形式が可変長レコードであることを示します。

F\$GETDVI

指定された装置に関する指定された情報を戻します。

フォーマット

F\$GETDVI(装置名, 項目[, パス名])

戻り値

要求する項目に応じて、整数または文字列となります。各項目の戻された値を表 DCLI-6 に示します。

引数

装置名

物理装置名、または物理装置名に割り当てられている論理名を指定します。文字列式として指定します。

装置名が評価されると、F\$GETDVI 関数は名前の初めの文字を調べます。名前の最初の文字がアンダースコア () の場合は、物理装置として扱われます。そうでない場合は、論理名変換を一回行いその等価名 (またはもとの名前) が使用されます。

項目

戻される装置情報のタイプを指定します。文字列式を指定します。表 DCLI-6 に示す 1 項目を指定します。

パス名 (Alpha/I64 のみ)

マルチパス対応装置のパス名を指定します。パス名は文字列式で指定します。

パス名引数を使用するかどうかについては、表 DCLI-6 の項目コードの説明を参照してください。一般に、パスによって異なる情報を戻す項目コードはパス名引数を使用します。マルチパス装置のパスは、SHOW DEVICE /FULL コマンド、システム・サービス SYS\$DEVICE_PATH_SCAN、レキシカル関数 F\$MULTIPATH で参照できます。

パス名引数を指定すると、指定された装置上に存在するパスと比較して検証されます。パスが存在しないと、たとえ指定した項目コードでパス名引数を使用していない場合でも、NOSUCHPATH エラーが戻されます。

説明

\$GETDVI システム・サービスを起動し、指定した装置に関する指定した情報を戻します。F\$DEVICE 関数を使用すると、現在使用しているシステムの装置の一覧を得ることができます。引数項目の記述が別の方法でない限り、F\$GETDVI 関数はローカル・ノードのみの装置情報を戻します。

この関数を使用すると、プロセスにチャンネルを割り当てる必要のない装置の情報を得ることができます。

F\$GETDVI 関数は、\$GETDVI システム・サービス呼び出し時に指定できるすべての項目の情報を戻します。\$GETDVI システム・サービスが許可する項目に加えて、F\$GETDVI 関数は EXISTS の指定を許可します。

表 DCLI-6 は、F\$GETDVI 関数に指定する項目、戻される情報、およびその情報のデータ・タイプを示しています。表 DCLI-6 に示す戻される情報に加えて、F\$GETDVI 関数は\$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

\$GETDVI システム・サービスおよび指定できる項目についての詳細は『OpenVMS System Services Reference Manual』を参照してください。

表 DCLI-6 F\$GETDVI 項目

項目	データ・タイプ	戻される情報 ¹
ACCESSTIMES_RECORDED (Alpha/I64 のみ)	文字列	ボリュームがアクセス時刻の記録をサポートしている場合は TRUE。そうでない場合は FALSE。
ACPPID	文字列	補助制御プロセス (ACP) 識別
ACPTYPE	文字列	ACP タイプ・コード。F11V1, F11V2, F11V3, F11V4, F11V5, F64, HBS, JNL, MTA, NET, REM, UCX, ILLEGAL のいずれか。 次の場合は、ACPTYPE 項目には ILLEGAL が返される。 <ul style="list-style-type: none">装置がマウントされていない、あるいは/FOREIGN 修飾子を使用してマウントされている場合。ACPTYPE が現在定義されていない場合。
ALL	文字列	装置が割り当てられている場合は TRUE, そうでない場合は FALSE。
ALLDEVNAM	文字列	割り当てクラス装置名

¹戻される情報リストに加え、F\$GETDVI 関数は\$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・ タイプ	戻される情報 ¹
ALLOCCLASS	0 ~ 255 間の ロングワード整数値	ホストの割り当てクラス
ALT_HOST_AVAIL	文字列	代替パスを提供するホストが使用可能である場合は TRUE。そうでない場合は FALSE。
ALT_HOST_NAME	文字列	代替パスを提供するホスト名
ALT_HOST_TYPE	文字列	代替パスを提供するホストのハードウェア・タイプ
AVAILABLE_PATH_COUNT (Alpha/I64 のみ)	整数値	マルチパス対応装置の、使用可能で稼働中のパスの数
AVL	文字列	装置が使用可能である場合は TRUE。そうでない場合は FALSE。
CCL	文字列	キャリッジ制御装置である場合は TRUE。そうでない場合は FALSE。
CLUSTER	整数値	ボリューム・クラスタ・サイズ
CONCEALED	文字列	論理装置名が隠し装置に変換される場合は TRUE。そうでない場合は FALSE。
CYLINDERS	整数値	ボリュームのシリンダ数 (ディスクのみ)
DEVBUFSIZ	整数値	装置バッファ・サイズ
DEVCHAR	整数値	装置属性
DEVCHAR2	整数値	追加装置属性
DEVCLASS	整数値	装置クラス。システム上で戻される装置クラスの値を調べる方法は、例を参照。
DEVDEPEND	整数値	装置依存情報
DEVDEPEND2	整数値	追加装置依存情報
DEVICE_TYPE_NAME	文字列	装置タイプ名。装置が SCSI テープまたはディスクである場合、装置タイプ名は装置から直接取得される。
DEVLOCKNAM	文字列	装置の一意なロック名
DEVNAM	文字列	装置名
DEVSTS	整数値	装置依存状態情報
DEVTYPE	整数値	装置タイプ。システム上で戻される装置タイプの値を調べる方法は、例を参照。
DFS_ACCESS	文字列	装置がリモート・ノードの分散ファイル・システム (DFS) サーバに接続された仮想ディスクの場合は TRUE。そうでない場合は FALSE。
DIR	文字列	装置がディレクトリ構造である場合は TRUE。そうでない場合は FALSE。
DMT	文字列	装置にディスマウントのための印がある場合は TRUE。そうでない場合は FALSE。
DUA	文字列	汎用装置である場合は TRUE。そうでない場合は FALSE。

¹ 戻される情報リストに加え、F\$GETDVI 関数は \$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・ タイプ	戻される情報 ¹
ELG	文字列	装置でエラー・ログを有効にしている場合は TRUE。そうでない場合は FALSE。
ERASE_ON_DELETE (Alpha/I64 のみ)	文字列	ボリューム上のファイルを削除する際に、ディスク・ブロックにゼロが書き込まれる場合は TRUE。そうでない場合は FALSE。
ERRCNT	整数値	装置のエラー・カウント。SET DEVICE /RESET=ERRCNT コマンドでエラー・カウントがリセットされた場合は、エラー・カウントがリセットされた日付と時刻を SHOW DEVICE/FULL コマンドで表示することができる。 パス名パラメータを指定すると、そのパスに対するエラー・カウントだけが戻される。パス名パラメータを省略すると、マルチパス装置のすべてのパスに対するエラー・カウントの合計が戻される。
ERROR_RESET_TIME (Alpha/I64 のみ)	文字列	エラー・カウントがリセットされた時刻
EXISTS	文字列	システム上に装置がある場合は TRUE。そうでない場合は FALSE。
EXPSIZE (Alpha/I64 のみ)	整数値	ボリュームで拡張可能な現在の上限。
FOD	文字列	ファイル単位取り扱い装置である場合は TRUE。そうでない場合は FALSE。
FOR	文字列	/FOREIGN 修飾子を使用してマウントされている場合は TRUE。そうでない場合は FALSE。
FREEBLOCKS	整数値	ボリューム上のフリー・ブロック数 (ディスクのみ)
FULLDEVNAM	文字列	完全指定の装置名
GEN	文字列	汎用装置である場合は TRUE。そうでない場合は FALSE。
HARDLINKS_SUPPORTED (Alpha/I64 のみ)	文字列	ボリューム上で別名ではなくハード・リンクがサポートされる場合は TRUE。そうでない場合は FALSE。
HOST_AVAIL	文字列	1 時パスを提供するホストが使用可能である場合は TRUE。そうでない場合は FALSE。
HOST_COUNT	整数値	OpenVMS Cluster 内の他のノードでの装置使用を可能にするホスト数
HOST_NAME	文字列	1 次パスを提供するホストの名前
HOST_TYPE	文字列	1 次パスを提供するホストのハードウェア・タイプ
IDV	文字列	providing 入力が可能である場合は TRUE。そうでない場合は FALSE。
LOCKID	整数値	クラスタ単位のロック識別
LOGVOLNAM	文字列	論理ボリューム名
MAXBLOCK	整数値	ボリューム上の論理ブロック数
MAXFILES	整数値	ボリューム上のファイルの最大数 (ディスクのみ)
MBX	文字列	装置がメールボックスである場合は TRUE。そうでない場合は FALSE。

¹戻される情報リストに加え、F\$GETDVI 関数は\$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・ タイプ	戻される情報 ¹
MEDIA_ID	整数値	非デコード・メディア ID
MEDIA_NAME	文字列	ディスク・タイプ名あるいはテープ・タイプ名
MEDIA_TYPE	文字列	装置名接頭辞
MNT	文字列	装置がマウントされている場合は TRUE。そうでない場合は FALSE。
MOUNT_TIME (Alpha/I64 のみ)	文字列	ボリュームがマウントされた時刻。クラスタにマウントされたボリュームの場合は、最初のマウント時刻だけが記録され、以降のマウント時刻は記録されない。
MOUNTCNT	整数値	ボリュームがローカル・ノードにマウントされた回数。 SHOW DEVICE コマンドによって表示される MOUNTCNT の値は、クラスタの全メンバが行ったボリューム・マウントの総数である。
MOUNTVER_ELIGIBLE (Alpha/I64 のみ)	文字列	ボリュームに対してマウント・チェックを行うことができる場合には TRUE。そうでない場合は FALSE。/FOREIGN または /NOMOUNT_VERIFICATION 修飾子を指定してマウントしたボリュームは、マウント・チェックの対象にならない。
MPDEV_AUTO_PATH_SW_CNT (Alpha/I64 のみ)	整数値	マルチパス装置で、入出力エラーや、ローカル・パスが利用可能になった場合のリモート・パスからローカル・パスへの自動的な「フェール・バック」などにより、パスが自動的に切り替わった回数。
MPDEV_CURRENT_PATH (Alpha/I64 のみ)	文字列	マルチパス装置の現在のパス名。 装置がマルチパス・セットに含まれていないときに、この装置のクラス・ドライバがパス名をサポートしている場合は、このレキシカルは装置パス名を返す。SYS\$DKDRIVER, SYS\$DUDRIVER, SYS\$MKDRIVER, および SYS\$GKDRIVER はパス名をサポートする。 装置のクラス・ドライバがパス名をサポートしていない場合は空文字列を返す。
MPDEV_MAN_PATH_SW_CNT (Alpha/I64 のみ)	整数値	SET DEVICE /PATH /SWITCH コマンドやシステム・サービス \$SET_DEVICE により、マルチパス装置のパスを手動で切り替えた回数。
MT3_DENSITY	文字列	装置の現在の記録密度 (テープのみ)。
MT3_SUPPORTED	文字列	装置が MT3DEF に定義されている記録密度をサポートしている場合は TRUE。そうでない場合は FALSE (Alpha のテープのみ)。
MULTIPATH (Alpha/I64 のみ)	文字列	装置がマルチパス・セットのメンバの場合は TRUE。そうでない場合は FALSE。
MVSUPMSG (Alpha/I64 のみ)	文字列	この装置上で、マウント・チェックの OPCOM メッセージが現在抑止されている場合は TRUE。そうでない場合は FALSE。マウント・チェック・メッセージの抑止についての詳細は、システム・パラメータ MVSUPMSG_INTVL および MVSUPMSG_NUM を参照。

¹戻される情報リストに加え、F\$GETDVI 関数は \$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・ タイプ	戻される情報 ¹
NET	文字列	ネットワーク装置である場合は TRUE。そうでない場合は FALSE。
NEXTDEVNAM	文字列	ボリューム・セット中の次のボリュームの装置名 (ディスクのみ)
NOCACHE_ON_VOLUME (Alpha/I64 のみ)	文字列	ボリュームがすべてのキャッシュを無効にしてマウントされた場合には TRUE。そうでない場合には FALSE。
NOHIGHWATER (Alpha/I64 のみ)	文字列	ボリューム上でハイウォーター・マークが無効になっている場合は TRUE。そうでない場合は FALSE。
NOSHARE_MOUNTED (Alpha/I64 のみ)	文字列	ボリュームが/NOSHARE 付きでマウントされた場合には TRUE。そうでない場合は FALSE。
ODS2_SUBSET0 (Alpha/I64 のみ)	文字列	マウントされているボリュームが ODS-2 ファイル構造のサブセットしかサポートしていない場合には TRUE。そうでない場合は FALSE。
ODS5 (Alpha/I64 のみ)	文字列	マウントされているボリュームが ODS-5 の場合には TRUE。そうでない場合は FALSE。
ODV	文字列	providing 出力が可能である場合は TRUE。そうでない場合は FALSE。
OPCNT	整数値	装置の操作回数。操作回数は SET DEVICE /RESET=OPCNT コマンドでリセットされているかもしれない点に注意。 パス名パラメータを指定すると、そのパスの操作回数だけが戻される。パス名パラメータを省略すると、マルチパス装置のすべてのパスに対する操作回数の合計が戻される。
OPR	文字列	装置が演算子である場合は TRUE。そうでない場合は FALSE。
OWNUIC	文字列	所有者の利用者識別コード (UIC)
PATH_AVAILABLE (Alpha/I64 のみ)	文字列	指定したパスが使用可能な場合は TRUE。そうでない場合は FALSE。 この項目コードは、一般にパス名パラメータとともに使用する。パス名パラメータを省略すると、マルチパス装置の現在のパスに関する情報が戻される。
PATH_NOT_RESPONDING (Alpha/I64 のみ)	文字列	指定されたパスが応答なしとしてマークされている場合は TRUE。そうでない場合は FALSE。 この項目コードは、一般にパス名パラメータとともに使用する。パス名パラメータを省略すると、マルチパス装置の現在のパスに関する情報が戻される。
PATH_POLL_ENABLED (Alpha/I64 のみ)	文字列	指定されたパスでマルチパス・ポーリングが有効な場合には TRUE。そうでない場合は FALSE。 この項目コードは、一般にパス名パラメータとともに使用する。パス名パラメータを省略すると、マルチパス装置の現在のパスに関する情報が戻される。

¹戻される情報リストに加え、F\$GETDVI 関数は\$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・タイプ	戻される情報 ¹
PATH_SWITCH_FROM_TIME (Alpha/I64 のみ)	文字列	このパスが手動または自動で切り替えられてからの時間。 この項目コードは、一般にパス名パラメータとともに使用する。パス名パラメータを省略すると、マルチパス装置の現在のパスに関する情報が戻される。
PATH_SWITCH_TO_TIME (Alpha/I64 のみ)	文字列	このパスが手動または自動で切り替えられるまでの時間。 この項目コードは、一般にパス名パラメータとともに使用する。パス名パラメータを省略すると、マルチパス装置の現在のパスに関する情報が戻される。
PATH_USER_DISABLED (Alpha/I64 のみ)	文字列	指定されたパスが、SET DEVICE /PATH /NOENABLE コマンドで無効にされている場合は TRUE。そうでない場合は FALSE。 この項目コードは、一般にパス名パラメータとともに使用する。パス名パラメータを省略すると、マルチパス装置の現在のパスに関する情報が戻される。
PID	文字列	装置所有者のプロセス識別番号
PREFERRED_CPU	整数値	戻り引数は、優先 CPU を示すビットがセットされた 32 ビットの CPU ビット・マスクである。戻り引数がゼロのビット・マスクを含んでいれば、Fast Path が無効にされているか、装置が Fast Path 対応装置でないために、優先 CPU は存在しない。戻り引数は、\$PROCESS_AFFINITY システム・サービスへの CPU ビット・マスク入力引数として使用される。この引数は、アプリケーション・プロセスを最適な優先 CPU に割り当てるために使用できる。
PROT_SUBSYSTEM_ENABLED (Alpha/I64 のみ)	文字列	保護サブシステムを有効にしてボリュームがマウントされた場合には TRUE。そうでない場合は FALSE。
QLEN (Alpha/I64 のみ)	整数値	装置のキューの長さ。I/O 待ち状態のキューの深さではなく、すでにドライバ内にある I/O 要求の数。
RCK	文字列	読み込みチェックが設定されている場合は TRUE。そうでない場合は FALSE。
RCT	文字列	装置に RCT が含まれている場合は TRUE。そうでない場合は FALSE。
REC	文字列	レコード単位取り扱い装置である場合は TRUE。そうでない場合は FALSE。
RECSIZ	整数値	ブロック状態のレコード・サイズ
REFCNT	整数値	装置を使用したプロセスの参照回数
REMOTE_DEVICE	文字列	リモート・デバイスである場合は TRUE。そうでない場合は FALSE。
RND	文字列	装置がランダム・アクセスを許可する場合は TRUE。そうでない場合は FALSE。
ROOTDEVNAM	文字列	ボリューム・セット中のルート・ボリュームの装置名 (ディスクのみ)
RTM	文字列	リアルタイム装置である場合は TRUE。そうでない場合は FALSE。

¹戻される情報リストに加え、F\$GETDVI 関数は\$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・タイプ	戻される情報 ¹
SCSI_DEVICE_FIRMWARE_REV (Alpha/I64 のみ)	文字列	SCSI ディスクまたは SCSI テープのファームウェアのリビジョン番号。それ以外の装置の場合には空文字列が戻される。
SDI	文字列	単一ディレクトリ構造である場合は TRUE。そうでない場合は FALSE。
SECTORS	整数値	1トラックあたりのセクタ数 (ディスクのみ)
SERIALNUM	整数値	ボリューム・シリアル番号 (ディスクのみ)
SERVED_DEVICE	文字列	served 装置である場合は TRUE。そうでない場合は FALSE。
SET_HOST_TERMINAL	文字列	リモート・ノードから SET HOST セッションを使用するためのリモート・ターミナルである場合は TRUE。そうでない場合は FALSE。
SHDW_CATCHUP_COPYING	文字列	完全コピー操作のターゲットのメンバである場合は TRUE。そうでない場合は FALSE。
SHDW_COPIER_NODE (Alpha/I64 のみ)	文字列	コピーまたはマージ操作をアクティブに実行しているノード名。
SHDW_DEVICE_COUNT (Alpha/I64 のみ)	整数値	仮想ユニット内にある装置の総数。コピー操作のターゲットとして追加されている装置も含む。
SHDW_GENERATION (Alpha/I64 のみ)	文字列	仮想ユニットの現在の内部更新番号。この値は変わることがある。
SHDW_MASTER	文字列	仮想ユニットである場合は TRUE。そうでない場合は FALSE。
SHDW_MASTER_MBR (Alpha/I64 のみ)	文字列	マージおよびコピー修理操作、およびシャドウ・セットの復元操作に使用される、マスタ・メンバ・ユニット名。
SHDW_MASTER_NAME	文字列	指定された装置がメンバであるシャドウ・セットを表わす仮想ユニットの装置名。指定された装置がメンバでない、または装置が仮想ユニットである場合、F\$GETDVI 関数は空文字列("")を戻す。
SHDW_MBR_COPY_DONE (Alpha/I64 のみ)	文字列	このメンバ・ユニットで実行されたコピー操作の割合。
SHDW_MBR_COUNT (Alpha/I64 のみ)	文字列	仮想ユニットにある、完全なソース・メンバの数。コピー・ターゲットとして追加される装置は、完全なソース・メンバではない。
SHDW_MBR_MERGE_DONE (Alpha/I64 のみ)	文字列	このメンバ・ユニットで実行されたマージ操作の割合。
SHDW_MBR_READ_COST (Alpha/I64 のみ)	文字列	メンバ・ユニットの現在の値のセット。この値を変更することによって、ユーザが指定する値を使用することができる。
SHDW_MEMBER	文字列	シャドウ・セット・メンバである場合は TRUE。そうでない場合は FALSE。
SHDW_MERGE_COPYING (Alpha/I64 のみ)	文字列	シャドウ・セットのマージ・メンバである場合は TRUE。そうでない場合は FALSE。

¹戻される情報リストに加え、F\$GETDVI 関数は\$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・ タイプ	戻される情報 ¹
SHDW_MINIMERGE_ENABLE (Alpha/I64 のみ)	文字列	クラスタ内のいずれかのシステムがクラッシュすると、仮想ユニットが完全マージではなくミニ・マージを実行する場合は TRUE。そうでない場合は FALSE。
SHDW_NEXT_MBR_NAME	文字列	シャドウ・セットでの次のメンバの装置名。仮想ユニットを指定した場合、F\$GETDVI 関数はシャドウ・セットメンバの装置名を戻す。シャドウ・セット・メンバ・ユニット名を装置名と引数とともに指定した場合、“次”のメンバ・ユニットを戻す。それ以上メンバがいなければ空文字列を戻す。 シャドウ・セットのすべてのメンバを確認するには、まず F\$GETDVI に仮想ユニットを指定する。それ以降の呼び出しでは、空文字列を戻す (F\$GETDVI 呼び出しが終わる) まで前の F\$GETDVI が戻すメンバ名を指定する。 割り当てクラスがゼロでない場合、装置名は割り当てクラスを含む。そうでない場合はディスク制御装置の装置名を含む。
SHDW_READ_SOURCE (Alpha/I64 のみ)	文字列	今回は読み込みに使用されるメンバ・ユニット名。キューの長さを読み込みコストの合計が最も少ないユニットが使用される。動的な値である。
SHDW_SITE (Alpha/I64 のみ)	整数値	指定された装置のサイト値。この値は SET DEVICE コマンドまたは SET SHADOW コマンドによって設定される。
SHDW_TIMEOUT (Alpha/I64 のみ)	整数値	この装置に設定された、ユーザが指定するタイムアウト値。ユーザが SETSHOSHADOW ユーティリティを使用して値を設定していない場合は、メンバ・ユニットには SYSGEN パラメータ SHADOW_MBR_TMO の値が使用され、仮想ユニットには MVTIMEOUT の値が使用される。
SHR	文字列	共用可能である場合は TRUE。そうでない場合は FALSE。
SPL	文字列	スプールされた状態である場合は TRUE。そうでない場合は FALSE。
SPLDEVNAM	文字列	スプールされている装置名
SQD	文字列	順編成のブロック単位取り扱い装置である場合 (つまり磁気テープの場合) は TRUE。そうでない場合は FALSE。
STS	整数値	状態情報
SWL	文字列	ライト・ロックされたソフトウェアである場合は TRUE。そうでない場合は FALSE。
TOTAL_PATH_COUNT (Alpha/I64 のみ)	整数値	マルチパス対応装置のパスの数。
TRACKS	整数値	1 シリンダあたりのトラック数 (ディスクのみ)
TRANSCNT	整数値	ポリウム・トランザクション回数
TRM	文字列	装置がターミナルである場合は TRUE。そうでない場合は FALSE。
TT_ACCPORNAM	文字列	ターミナル・サーバ名とポート名
TT_ALTYPEAHD	文字列	ターミナルに代替先読みを可能なバッファがある場合は TRUE。そうでない場合は FALSE (ターミナルのみ)

¹戻される情報リストに加え、F\$GETDVI 関数は \$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・ タイプ	戻される情報 ¹
TT_ANSICRT	文字列	ANSI CRT ターミナルである場合は TRUE。そうでない場合は FALSE (ターミナルのみ)
TT_APP_KEYPAD	文字列	キーパッドがアプリケーション・モードである場合は TRUE。そうでない場合は FALSE (ターミナルのみ)
TT_AUTOBAUD	文字列	ターミナルにボー・レート自動検出機能がある場合 TRUE。そうでない場合は FALSE (ターミナルのみ)
TT_AVO	文字列	VT-100 ファミリのターミナル・ディスプレイがある場合は TRUE。そうでない場合は FALSE (ターミナルのみ)
TT_BLOCK	文字列	ブロック・モード機能がある場合は TRUE。そうでない場合は FALSE (ターミナルのみ)
TT_BRDCSTMBX	文字列	メールボックス・ブロードキャスト・メッセージを使用する場合は TRUE。そうでない場合は FALSE (ターミナルのみ)
TT_CHARSET	整数値	ターミナルがサポートするコード化文字セットを示すビットマップ
TT_CRFILL	文字列	キャリッジ・リターンを使用した後フィルを必要とする場合は TRUE。そうでない場合は FALSE。
TT_CS_HANGUL	文字列	DEC Korean 文字セットをサポートしている場合は TRUE。そうでない場合は FALSE。
TT_CS_HANYU	文字列	DEC Hanyu 文字セットをサポートしている場合は TRUE。そうでない場合は FALSE。
TT_CS_HANZI	文字列	DEC Hanzi 文字セットをサポートしている場合は TRUE。そうでない場合は FALSE。
TT_CS_KANA	文字列	DEC Kana 文字セットをサポートしている場合は TRUE。そうでない場合は FALSE。
TT_CS_KANJI	文字列	DEC Kanji 文字セットをサポートしている場合は TRUE。そうでない場合は FALSE。
TT_CS_THAI	文字列	DEC Thai 文字セットをサポートしている場合は TRUE。そうでない場合は FALSE。
TT_DECCRT	文字列	Digital CRT ターミナルである場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_DECCRT2	文字列	Digital CRT2 ターミナルである場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_DECCRT3	文字列	Digital CRT3 ターミナルである場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_DECCRT4	文字列	Digital CRT4 ターミナルである場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_DIALUP	文字列	ダイヤルアップ回線に接続している場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_DISCONNECT	文字列	切断可能である場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_DMA	文字列	直接メモリ・アクセス (DMA) モードがある場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。

¹戻される情報リストに加え、F\$GETDVI 関数は\$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・ タイプ	戻される情報 ¹
TT_DRCS	文字列	ロード可能な文字フォントをサポートしている場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_EDIT	文字列	編集属性が設定されている場合は TRUE。そうでない場合は FALSE。
TT_EDITING	文字列	拡張編集が可能である場合は TRUE。そうでない場合は FALSE。
TT_EIGHTBIT	文字列	8 ビット ASCII 文字セットを使用する場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_ESCAPE	文字列	エスケープ・シーケンスを生成する場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_FALLBACK	文字列	multinational fallback オプションを使用する場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_HALFDUP	文字列	ターミナルが半二重モードである場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_HANGUP	文字列	ハングアップ属性が設定されている場合は TRUE。そうでない場合は FALSE。
TT_HOSTSYNC	文字列	ホスト/ターミナル通信がある場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_INSERT	文字列	挿入モードが省略時の行編集モードになっている場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_LFFILL	文字列	行送りの後フィルを必要とする場合は TRUE。そうでない場合は FALSE。
TT_LOCALECHO	文字列	ローカル・エコー属性が設定されている場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_LOWER	文字列	小文字の文字セットがある場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_MBXDSABL	文字列	ターミナルに関連づけられているメールボックスが、要求されていない、または要求されている入力通知を受け取る場合は TRUE。そうでない場合は FALSE。
TT_MECHFORM	文字列	改ページ機能がある場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_MECHTAB	文字列	タブ機能、およびタブ拡張機能がある場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_MODEM	文字列	ターミナルがモデムに接続している場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_MODHANGUP	文字列	修正するハングアップ属性が設定されている場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_NOBRDCST	文字列	ブロードキャスト・メッセージを受け取る場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_NOECHO	文字列	入力文字がエコーされる場合は TRUE。そうでない場合は FALSE。

¹戻される情報リストに加え、F\$GETDVI 関数は\$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・ タイプ	戻される情報 ¹
TT_NOTYPEAHD	文字列	読み込み操作でデータを要求する場合は TRUE。そうでない場合は FALSE。
TT_OPER	文字列	オペレータ・ターミナルである場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_PAGE	整数値	ターミナル・ページの長さ (ターミナルのみ)
TT_PASTHRU	文字列	フロー制御が有効な PASSALL モードである場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_PHYDEVNAM	文字列	チャンネル番号または仮想ターミナルに関連づけられた物理装置名
TT_PRINTER	文字列	プリンタ・ポートが使用可能な場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_READSYNC	文字列	読み込み同期化機能がある場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_REGIS	文字列	ReGIS グラフィックがある場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_REMOTE	文字列	設定したモデム制御がある場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_SCOPE	文字列	ターミナルがビデオ・スクリーン・ディスプレイである場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_SECURE	文字列	機密サーバを認識できる場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_SETSPEED	文字列	ターミナル行でスピードを設定できる場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_SIXEL	文字列	シクセルがサポートされている場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_SYSPWD	文字列	特殊なターミナルでシステム・パスワード使用可能な場合は TRUE。そうでない場合は FALSE。
TT_TTSYNC	文字列	ターミナル/ホストの同期化が存在する場合は TRUE。そうでない場合は FALSE(ターミナルのみ)。
TT_WRAP	文字列	右マージンを越えてカーソルが移動した時、新しい行を挿入する場合は TRUE。そうでない場合は FALSE。
UNIT	整数値	ユニット番号
VOLCHAR (Alpha/I64 のみ)	文字列	マウントされた装置のボリューム特性または機能を表現する、128 ビット文字列 (16 バイト)。ビットが設定されると、ボリュームは機能を実行できるようになる。
VOLCOUNT	整数値	ボリューム・セット中のボリューム回数 (ディスクのみ)
VOLNAM	文字列	ボリューム名
VOLNUMBER	整数値	ボリューム・セット中の現在のボリュームの番号 (ディスクのみ)
VOLSETMEM	文字列	装置がボリューム・セットである場合は TRUE。そうでない場合は FALSE(ディスクのみ)。

¹戻される情報リストに加え、F\$GETDVI 関数は\$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

(次ページに続く)

表 DCLI-6 (続き) F\$GETDVI 項目

項目	データ・タイプ	戻される情報 ¹
VOLSIZE (Alpha/I64 のみ)	整数値	ボリュームの現在の論理ボリューム・サイズ。
VOLUME_EXTEND_QUANTITY (Alpha/I64 のみ)	整数値	ボリューム上のすべてのファイルに対して省略時の拡張サイズとして使用するブロック数。
VOLUME_MOUNT_GROUP (Alpha/I64 のみ)	文字列	ボリュームが/GROUP 付きでマウントされている場合は TRUE。そうでない場合は FALSE。
VOLUME_MOUNT_SYS (Alpha/I64 のみ)	文字列	ボリュームが/SYSTEM 付きでマウントされている場合は TRUE。そうでない場合は FALSE。
VPROT	文字列	ボリューム保護マスク
WCK	文字列	書き込みチェックが設定されている場合は TRUE。そうでない場合は FALSE。
WRITETHRU_CACHE_ENABLED (Alpha/I64 のみ)	文字列	ボリュームが、ライトスルー・キャッシュを有効にしてマウントされている場合は TRUE。そうでない場合は FALSE。
WWID (Alpha/I64 のみ)	文字列	Fibre Channel 装置のためのワールドワイド識別子。

¹戻される情報リストに加え、F\$GETDVI 関数は\$GETDVI システム・サービスで作成される任意のエラー・メッセージも戻します。

例

- ```
$ ERR = F$GETDVI("_DQA0", "ERRCNT")
$ SHOW SYMBOL ERR
ERR = 0 Hex = 00000000 Octal = 000000
```

この例は、DQA0 という装置のエラー回数を調べるために、F\$GETDVI 関数を使用する方法を示しています。DQA0 という装置名と ERRCNT という項目は、文字列リテラルなので引用符で囲まなければなりません。

- ```
$ LIBRARY/EXTRACT=$DCDEF/OUTPUT=$DCDEF.TXT SYS$LIBRARY:STARLET.MLB
```

この例では、STARLET ライブラリの装置タイプと装置クラスの値のリストを含んでいるファイル\$DCDEF.TXT を作成する方法を示しています。装置クラスは 'DC\$' で始まり、装置タイプは 'DT\$' で始まります。

今日の SCSI ディスクとテープの大部分は汎用の DEVTYPE コード (DT\$_GENERIC_DK または DT\$_GENERIC_MK) を戻すので、DEVICE_TYPE_NAME 項目を使用するようにしてください。

```
$ X=F$GETDVI("XDELTA$DKA0:", "DEVICE_TYPE_NAME")
$ SHOW SYMBOL X
X = "RZ29B"
```

F\$GETENV (Alpha のみ)

指定されたコンソール環境変数の値を戻します。

フォーマット

F\$GETENV(項目リスト)

戻り値

指定されたコンソール環境変数の値を戻します。システムがコンソール・モードである場合、ユーザはコンソール環境変数を修正することができます。このレキシカル関数を使用することで、システムが稼動中に変数の内容を読み取ることができます。

引数

項目リスト

定義されているコンソール環境変数名は次のとおりです。

Auto_action, Boot_dev, Bootdef_dev, Booted_dev, Boot_file, Booted_file, Boot_osflags, Booted_osflags, Boot_reset, Dump_dev, Enable_audit, License, Char_set, Language, Tty_dev

説明

指定されたコンソール環境変数の値を戻します。

例

1.

```
$ dump_device = f$getenv("dump_dev")
$ write sys$output "The dump device for this system is ", dump_device
```

この関数は、システムのダンプ装置を表示します。

F\$GETJPI

\$GETJPI システム・サービスを呼出し、指定されたプロセスに関する 情報を戻します。

同じグループ内の他のプロセスに関する情報を得るためには、GROUP 特権が必要です。システム内の他のプロセスに関する情報を得るためには、WORLD 特権が必要です。

フォーマット

F\$GETJPI(*pid* , 項目)

戻り値

要求する項目に応じて、整数または文字列となります。各項目に返される値のデータ・タイプを表 DCLI-7 に示します。

引数

pid

情報が返されるプロセスの識別番号。プロセス識別の値 (*pid*) は文字列式として指定します。 *pid* 引数を指定する場合、先行する 0 は省略できます。

空文字列を指定すると、現在のプロセスに対する操作となります。

\$GETJPI システム・サービスでは *pid* 引数にワイルドカード文字を使用できますが、F\$GETJPI 関数では使用できません。プロセスの識別番号の一覧を得るには、F\$PID 関数を使用します。

項目

返されるプロセス情報のタイプ。文字列式として指定します。表 DCLI-7 の項目から 1 項目を指定します。

説明

\$GETJPI システム・サービスを呼び出し、指定したプロセスの情報を戻します。\$GETJPI システム・サービスで指定することができる、すべての項目の情報を戻します。\$GETJPI システム・サービスについての詳細は『OpenVMS System Services Reference Manual』を参照してください。

ターゲット・プロセスが中断されている，または MWAIT (リソース待ち) 状態の場合，F\$GETJPI レキシカル関数は，ゼロまたは空文字列を戻します。また，要求された項目は，プロセスの仮想アドレス空間に格納されます。

F\$GETJPI 関数を使用すると，プロセスが自動的にファイルをアンシェルフするかどうか分かります。

STS2 項目コードを指定すると，F\$GETJPI は 32 ビットの数値を戻します。この数値を 2 進形式に変換すると，シンボル・ビット位置 PCB\$V_NOUNSHELVE の数字により，省略時の設定でアンシェルフするかどうか分かります。このビットが 1 であれば，自動アンシェルフはオフになり，このビットが 0 であれば，自動アンシェルフがオンになります。

F\$GETJPI 関数に指定する項目，戻される情報，およびその情報のデータ・タイプを表 DCLI-7 に示します。

表 DCLI-7 F\$GETJPI 項目

項目	データ・タイプ	戻される情報
ACCOUNT	文字列	アカウント名文字列（後ろの空白を含め 8 文字）
APTCNT	整数値	アクティブなページ・テーブル数
ASTACT	整数値	アクティブな非同期システム・トラップ (AST) のアクセス・モード
ASTCNT	整数値	残りの AST クォータ
ASTEN	整数値	AST が有効なアクセス・モード
ASTLM	整数値	AST クォータ
AUTHPRI	整数値	ALTPRI 特権のないプロセスが \$SETPRI システム・サービスで獲得できる最高優先順位
AUTHPRIV	文字列	プロセスでの使用を認められた特権
BIOCNT	整数値	残りのバッファード I/O クォータ
BIOLM	整数値	バッファード I/O クォータ
BUFIO	整数値	プロセス・バッファード I/O 操作回数
BYTCNT	整数値	残りのバッファード I/O バイト回数クォータ
BYTLM	整数値	バッファード I/O バイト回数クォータ
CASE_LOOKUP_IMAGE (Alpha/I64 のみ)	文字列	指定されたプロセスのファイル名検索における大文字/小文字の区別に関する情報を戻す。この値は，イメージが有効な期間にのみ設定される。値は BLIND または SENSITIVE。詳細は，『Guide to OpenVMS File Applications』を参照。

(次ページに続く)

表 DCLI-7 (続き) F\$GETJPI 項目

項目	データ・ タイプ	戻される情報
CASE_LOOKUP_PERM (Alpha/I64 のみ)	文字列	指定されたプロセスのファイル名検索における大文字/小文字の区別に関する情報を戻す。スタイルが再び設定されなければ、プロセスが有効な期間にのみ設定される。値は BLIND または SENSITIVE。 詳細は、『Guide to OpenVMS File Applications』を参照。
CLASSIFICATION (Alpha/I64 のみ)	文字列	PSB に格納されている現在の MAC 分類。20 バイトの埋め込み文字列。
CLINAME	文字列	現在のコマンド言語インタプリタ。常に DCL を戻す。
CPULIM	整数値	プロセス CPU 時間の上限
CPUTIM	整数値	CPU 時間。1/100 秒単位で示される。
CREPRC_FLAGS	整数値	プロセスを作成する\$CREPRC 呼び出しのstsflg引数により指定されるフラグ
CURPRIV	文字列	現在のプロセス特権
CURRENT_CAP_MASK (Alpha/I64 のみ)	整数値	指定されたカーネル・スレッドの現在のカーバビリティ・マスク。詳細は、SET PROCESS /CAPABILITIES コマンドを参照。
DFPFC	整数値	省略時のページ・フォルト・クラスタ・サイズ
DFWSCNT	整数値	省略時のワーキング・セット・サイズ
DIOCNT	整数値	残りの直接 I/O クォータ
DIOLM	整数値	直接 I/O クォータ
DIRIO	整数値	プロセスの直接 I/O 操作回数
EFCS	整数値	ローカル・イベント・フラグ 0-31
EFCU	整数値	ローカル・イベント・フラグ 32-63
EFWM	整数値	イベント・フラグ待ちマスク
ENQCNT	整数値	残りのロック要求クォータ
ENQLM	整数値	ロック要求クォータ
EXCVEC	整数値	例外ベクタのリストのアドレス
FAST_VP_SWITCH	整数値	ベクタ・コンテキスト・スイッチの損失なしで非アクティブ・ベクタ・プロセッサを使用可能にする、ベクタ命令を発行した回数
FILCNT	整数値	残りのオープン・ファイル・クォータ
FILLM	整数値	オープン・ファイル・クォータ
FINALEXC	整数値	最終例外ベクタのリストのアドレス
FREP0VA	整数値	プログラム・リージョン (p0 空間) の終わりにある最初のフリー・ページ (イメージが実行されていない場合はなし)
FREP1VA	整数値	コントロール・リージョン (p1 空間) の終わりにある最初のフリー・ページ

(次ページに続く)

表 DCLI-7 (続き) F\$GETJPI 項目

項目	データ・ タイプ	戻される情報
FREPTECNT	整数値	仮想メモリの拡張に使用可能なページ数
GPGCNT	整数値	ワーキング・セットでのグローバル・ページ数
GRP	整数値	利用者識別コード (UIC) のグループ番号
HOME_RAD (Alpha/I64 のみ)	整数値	ホーム・リソース・アフィニティ・ドメイン (RAD)。AlphaServer GS シリーズ・システム上でのみサポートされます。
IMAGECOUNT	整数値	プロセスでラン・ダウンされたイメージ数
IMAGE_AUTHPRIV (Alpha/I64 のみ)	文字列	インストールされたイメージの認可された特権マスク
IMAGE_PERMPRIV (Alpha/I64 のみ)	文字列	インストールされたイメージのパーマネント (省略時の) 特権マスク
IMAGE_WORKPRIV (Alpha/I64 のみ)	文字列	インストールされたイメージの実行中の (アクティブな) 特権マスク
IMAGENAME	文字列	現在のイメージのファイル名
IMAGPRIV	文字列	現在のイメージをインストールした特権
INSTALL_RIGHTS (Alpha/I64 のみ)	整数値	インストール・ライト・リストの 2 進形式の内容。この項目コードはコンマ (,) で区切られたインストール・ライトのリストを戻す。
INSTALL_RIGHTS_ SIZE (Alpha/I64 のみ)	整数値	インストール・ライトの格納に必要なバイト数
JOBPRCCNT	整数値	ジョブに所有されるサブプロセス数
JOBTYP	整数値	ジョブ・ツリーのルートにあるプロセスの実行モード
LAST_LOGIN_I	文字列	最後に会話型ログインをした時刻 (ログインしたときに報告された値)
LAST_LOGIN_N	文字列	最後に非会話型ログインをした時刻 (ログインしたときに報告された値)
LOGIN_FAILURES	整数値	現在のセッションを開始する前に起きたログイン失敗数 (ログインしたときに報告された値)
LOGIN_FLAGS	整数値	ログイン手順に関連した追加情報を含むロングワード・ビットマスク
LOGINTIM	文字列	プロセス作成時刻
MASTER_PID	文字列	現在のジョブのプロセス・ツリーの一番上のプロセスの識別番号 (PID)
MAXDETACH	整数値	プロセスの所有者であるユーザに許可された、独立プロセスの最大数
MAXJOBS	整数値	プロセスの所有者であるユーザに許可された、実行中のプロセスの最大数
MEM	整数値	利用者識別コード (UIC) のメンバ番号
MODE	文字列	現在のプロセス・モード (BATCH, INTERACTIVE, NETWORK, または OTHER)

(次ページに続く)

表 DCLI-7 (続き) F\$GETJPI 項目

項目	データ・ タイプ	戻される情報
MSGMASK	整数値	SET MESSAGE コマンドによって確立された現在のメッセージ・マスク。マスクが指定されていない場合、\$GETMSG システム・サービスに省略時のシステム・メッセージ・マスクが記述されている。詳細は、\$PUTMSG システム・サービス (メッセージ・マスク・ビットについて) および F\$ENVIRONMENT レキシカル関数の MESSAGE 項目を参照。
MULTITHREAD	整数値	(sysgen 設定により制限されている) プロセスに対する現在の設定
NODENAME	文字列	プロセスが実行中である OpenVMS Cluster の名前
NODE_CSID	整数値	プロセスが実行中である OpenVMS Cluster のクラスタ ID
NODE_VERSION	文字列	プロセスが実行中である OpenVMS Cluster のオペレーティング・システムのバージョン番号
OWNER	文字列	プロセス所有者のプロセス識別番号
PAGEFLTS	整数値	ページ・フォルトの回数
PAGFILCNT	整数値	残りのページング・ファイル・クォータ
PAGFILLOC	整数値	ページング・ファイルの記憶位置
PARSE_STYLE_PERM (Alpha/I64 のみ)	文字列	\$SET_PROCESS_PROPERTIESW によって設定された値
PARSE_STYLE_IMAGE (Alpha/I64 のみ)	文字列	\$SET_PROCESS_PROPERTIESW によって設定された値
PERMANENT_CAP_ MASK (Alpha/I64 のみ)	整数値	指定されたカーネル・スレッドのパーマネント・ケーパビリティ・マスク。詳細は、SET PROCESS/CAPABILITIES コマンドを参照。
PERSONA_AUTHPRIV (Alpha/I64 のみ)	文字列	ペルソナの認可された特権マスク
PERSONA_ID (Alpha /I64 のみ)	整数値	ペルソナ ID (ロングワード整数値)
PERSONA_PERMPRIV (Alpha/I64 のみ)	文字列	ペルソナのパーマネント (省略時) 特権マスク
PERSONA_RIGHTS (Alpha/I64 のみ)	整数値	ペルソナ・ライト・リストの 2 進形式の内容。この項目コードはコンマ (,) で区切られたペルソナ・ライトのリストを戻す。
PERSONA_RIGHTS_ SIZE (Alpha/I64 のみ)	整数値	ペルソナ・ライトの格納に必要なバイト数
PERSONA_WORKPRIV (Alpha/I64 のみ)	文字列	ペルソナの実行中の (アクティブな) 特権マスク
PGFLQUOTA	整数値	ページ・ファイル・クォータ (最大仮想ページ数)
PHDFLAGS	整数値	フラグ・ワード
PID	文字列	プロセス識別番号
PPGCNT	整数値	プロセス・ページ数

(次ページに続く)

表 DCLI-7 (続き) F\$GETJPI 項目

項目	データ・ タイプ	戻される情報
PRCCNT	整数値	プロセスが所有するサブプロセス数
PRCLM	整数値	サブプロセス・クォータ
PRCNAM	文字列	プロセス名
PRI	整数値	プロセスの現在の優先順位
PRIB	整数値	プロセスの基本優先順位
PROC_INDEX	整数値	プロセスの索引番号
PROCESS_RIGHTS	文字列	利用者識別コードを含むプロセスのローカル・ライト・リスト。この項目コードはコンマ(,)で区切られた識別子名のリストを戻す。
PROCPRIV	文字列	プロセスの省略時の特権
RIGHTSLIST	文字列	すべてのプロセス・ライト・リスト。これは PROCESS_RIGHTS と SYSTEM_RIGHTS を加えたものと同様。この項目コードはコンマ(,)で区切られた識別子名のリストを戻す。
RIGHTS_SIZE	整数値	ライト・リストをバッファリングするために必要なバイト数。このリストにはシステム・ライト・リストとプロセス・ライト・リストの両方が含まれる。
SCHED_CLASS_NAME (Alpha/I64 のみ)	文字列	プロセスがクラス・スケジュールされていた場合はスケジューリング・クラスの名前を、そうでない場合は空文字列を返す。
SHRFILLM	整数値	プロセスが属するジョブに許可された、オープンな共用ファイルの最大数
SITESPEC	整数値	1 プロセスあたりのサイト固有のロングワード
SLOW_VP_SWITCH	整数値	すべてのベクタ・コンテキスト・スイッチを使用して非アクティブ・ベクタ・プロセッサを使用可能にする、ベクタ命令を発行した回数
STATE	文字列	プロセスの状態
STS	整数値	プロセス状態フラグの最初のロングワード
STS2	整数値	プロセス状態フラグの 2 番目のロングワード
SUBSYSTEM_RIGHTS (Alpha/I64 のみ)	整数値	サブシステム・ライト・リストの 2 進形式の内容。この項目コードはコンマ(,)で区切られたサブシステム・ライトのリストを戻す。
SUBSYSTEM_RIGHTS_SIZE (Alpha/I64 のみ)	整数値	サブシステム・ライトの格納に必要なバイト数
SWPFILLOC	整数値	スワップ・ファイルの記憶位置
SYSTEM_RIGHTS	文字列	プロセスのシステム・ライト・リスト。この項目コードはコンマ(,)で区切られた識別子名のリストを戻す。
SYSTEM_RIGHTS_SIZE (Alpha/I64 のみ)	整数値	システム・ライトの格納に必要なバイト数
TABLENAME	文字列	プロセスの現在のコマンド言語インタプリタ (CLI)・テーブルのファイル指定

(次ページに続く)

表 DCLI-7 (続き) F\$GETJPI 項目

項目	データ・ タイプ	戻される情報
TERMINAL	文字列	会話型ユーザのログイン・ターミナル名 (1 ~ 7 文字)
TMBU	整数値	終了メールボックスのユニット番号
TOKEN	文字列	トークン・サイズ。 TRADITIONAL (255 バイト) または EXPANDED (4000 バイト)。
TQCNT	整数値	残りのタイマ・キュー・エントリ・クォータ
TQLM	整数値	タイマ・キュー・エントリ・クォータ
TT_ACCPORNAM	文字列	プロセスと関連するターミナルのアクセス・ポート名
TT_PHYDEVNAM	文字列	プロセスと関連するターミナルの物理装置名
UAF_FLAGS	整数値	プロセスを所有するユーザの利用者登録ファイル (UAF) 記録からのフラグ
UIC	文字列	プロセスのユーザ識別コード (UIC)
USERNAME	文字列	ユーザ名文字列 (後ろの空文字を含めて 12 文字)
VIRTPEAK	整数値	仮想アドレスの最大サイズ
VOLUMES	整数値	現在マウントされているボリュームの数
VP_CONSUMER	論理値	プロセスがベクタ消費であるかを示すフラグ
VP_CPUTIM	整数値	ベクタ・カスタマとして累算したプロセスの合計時間
WSAUTH	整数値	認可された最大ワーキング・セット・サイズ
WSAUTHEXT	整数値	認可された最大ワーキング・セット超過値
WSEXTENT	整数値	現在のワーキング・セット超過値
WSPEAK	整数値	ワーキングセット最大値
WSQUOTA	整数値	ワーキング・セット・サイズ・クォータ
WSSIZE	整数値	プロセスの現在のワーキング・セット・リミット

\$GETJPI 関数を使用して、空プロセスまたはスワップ・プロセスの情報を要求する場合は、以下の項目を除いて、表 DCLI-7 にある任意の項目を指定できます。

ACCOUNT	BYTLM	ENQCNT	ENQLM
EXCVEC	FILCNT	FILM	FINALEXC
IMAGNAME	LOGINTIM	MSGMASK	PAGFILCNT
PGFLQUOTA	PRCCNT	PRCLM	PROCPRIV
SITESPEC	TQCNT	TQLM	USERNAME
VIRTPEAK	VOLUMES	WSPEAK	

例

```
1. $ NAME = F$GETJPI("3B0018","USERNAME")
   $ SHOW SYMBOL NAME
      NAME = "JANE      "
```

この例は、F\$GETJPI 関数を使用して、3B0018 というプロセス番号の利用者名を得る方法を示しています。利用者名には、NAME というシンボルが割り当てられます。

```
2. $ X=F$ENVIRONMENT("MESSAGE")
   $ SHOW SYMBOL X
      X = "/FACILITY/SEVERITY/IDENTIFICATION/TEXT"
   $ X=F$GETJPI("0","MSGMASK")
   $ SHOW SYMBOL X
      X = 15   Hex = 0000000F   Octal = 00000000017
   $ SET MESSAGE /NOFACILITY
   $ X=F$ENVIRONMENT("MESSAGE")
   $ SHOW SYMBOL X
      X = "/NOFACILITY/SEVERITY/IDENTIFICATION/TEXT"
   $ X=F$GETJPI("0","MSGMASK")
   $ SHOW SYMBOL X
      X = 7    Hex = 00000007   Octal = 00000000007
   $ SET MESSAGE /FACILITY
   $ X=F$ENVIRONMENT("MESSAGE")
   $ SHOW SYMBOL X
      X = "/FACILITY/SEVERITY/IDENTIFICATION/TEXT"
   $ X=F$GETJPI("0","MSGMASK")
   $ SHOW SYMBOL X
      X = 15   Hex = 0000000F   Octal = 00000000017
   $
```

この例は、F\$GETJPI 関数の MSGMASK 項目の利用法を示しています。

F\$GETQUI

\$GETQUI システム・サービスを呼び出し、キューに関する情報を戻します。これらの情報にはバッチ・キューおよびプリント・キューに登録されるジョブ、フォーム定義、およびキュー・データベースに保存されている属性定義が含まれます。

キュー・マネージャについての情報も戻します。

ほとんどの操作では、読み込み (R) アクセス権が必要です。

フォーマット

F\$GETQUI(機能,[項目],[オブジェクト id],[フラグ])

戻り値

指定する項目に応じて、整数または文字列になります。論理値を戻す項目については、文字列 TRUE または FALSE が戻されます。\$GETQUI システム・サービスがエラー・コードを返した場合、F\$GETQUI は空文字列を返します。

引数

機能

F\$GETQUI レキシカル関数が実行する動作を指定します。F\$GETQUI は、\$GETQUI システム・サービスで指定できるすべての機能をサポートしています。これらの機能を次の表に示します。

関数	説明
CANCEL_OPERATION	前の F\$GETQUI 呼び出しにより開始したワイルドカード操作を終了させる。
DISPLAY_CHARACTERISTIC	特定の属性定義、またはワイルドカード操作での次の属性定義に関する情報を返す。
DISPLAY_ENTRY	ワイルドカード操作で選択基準に一致する、特定のジョブ・エントリまたは次のジョブ・エントリに関する情報を返す。返されるジョブ情報の点において、DISPLAY_ENTRY の関数コードは DISPLAY_JOB 関数コードに似ている。ただし DISPLAY_JOB は、キュー・コンテキストを確立しなければならない。DISPLAY_ENTRY は、キュー・コンテキストを確立する必要はない。現在のプロセスのユーザ名と一致するこれらのエントリだけが実行されます。

関数	説明
DISPLAY_FILE	現在のジョブ・コンテキストのために定義した次のファイルに関する情報を返す。F\$GETQUI 呼び出しを行いファイル情報を要求する前に、(DISPLAY_QUEUE および DISPLAY_JOB 関数コードを使用して) キューおよびジョブ情報を表示するために、または (DISPLAY_ENTRY 関数コードを使用して) エントリ情報を表示するために呼び出しを行う必要がある。
DISPLAY_FORM	特定のフォーム定義、またはワイルドカード操作で次のフォーム定義を返す。
DISPLAY_JOB	現在のキュー・コンテキストに対して定義された次のジョブに関する情報を返す。F\$GETQUI 呼び出しを行いジョブ情報を要求する前に、(DISPLAY_QUEUE 関数コードを使用して) キュー情報を表示するために呼び出しを行う必要がある。返されるジョブ情報の点において、DISPLAY_JOB 関数コードと DISPLAY_ENTRY 関数コードは似ている。ただし DISPLAY_JOB は、キュー・コンテキストを確立しなければならない。DISPLAY_ENTRY は、キュー・コンテキストを確立する必要はない。
DISPLAY_MANAGER	特定のキュー・マネージャ、またはワイルドカード操作で次のキュー・マネージャに関する情報を返す。
DISPLAY_QUEUE	特定のキュー定義、またはワイルドカード操作で次のキュー定義に関する情報を返す。
TRANSLATE_QUEUE	キューの論理名を変換する。

項目コード、オブジェクト id、またはフラグ引数に指定できない関数の引数もあります。各関数の引数とそれに対応するフォーマット行を、次の表に示します。また、項目コード、オブジェクト・コード、およびフラグ引数は、必須であるか、オプションであるか、あるいは適用外であるかも示します。次の表において、([]) は、オプションの引数を示します。引数がないことは、引数を指定できないことを示します。2つのコンマ(,) は、(オプション/適用外のいずれの場合でも) 省略された引数を示すプレースホルダとして使用します。

関数	フォーマット行
CANCEL_OPERATION	F\$GETQUI("CANCEL_OPERATION") または F\$GETQUI(" ")
DISPLAY_CHARACTERISTIC	F\$GETQUI("DISPLAY_CHARACTERISTIC",[item],object-id,[flags])
DISPLAY_ENTRY	F\$GETQUI("DISPLAY_ENTRY",[item],[object-id],[flags])
DISPLAY_FILE	F\$GETQUI("DISPLAY_FILE",[item],,[flags])
DISPLAY_FORM	F\$GETQUI("DISPLAY_FORM",[item],object-id,[flags])
DISPLAY_JOB	F\$GETQUI("DISPLAY_JOB",[item],,[flags])
DISPLAY_MANAGER	F\$GETQUI("DISPLAY_MANAGER",[item],object-id,[flags])
DISPLAY_QUEUE	F\$GETQUI("DISPLAY_QUEUE",[item],object-id,[flags])
TRANSLATE_QUEUE	F\$GETQUI("TRANSLATE_QUEUE",[item],object-id)

項目

\$GETQUI システム・サービスの出力項目コードに相当します。特定のキュー、ジョブ、ファイル、フォーム、または属性に関する情報の種類を指定します。VAX では、キュー・マネージャ情報も指定できます。表 DCLI-9 では戻される項目コードとデータ・タイプを示しています。

オブジェクト id
\$GETQUI システム・サービスの QUI\$SEARCH_NAME , QUI\$_SEARCH_NUMBER , および QUI\$_SEARCH_JOB_NAME 入力項目コードに相当します。
F\$GETQUI が情報を返すオブジェクト (たとえば特定のキュー名, ジョブ名, またはフォーム番号など) のオブジェクト名または番号を指定します。次に示す関数では, ワイルドカードを使用できます。

- DISPLAY_CHARACTERISTIC
- DISPLAY_ENTRY
- DISPLAY_FORM
- DISPLAY_MANAGER
- DISPLAY_QUEUE

後続の呼び出しで, オブジェクト id 引数としてワイルドカードを指定すると, 特定のキューに登録されている 1 つまたは複数ジョブ, または特定のキューに登録されているジョブのファイルの状態情報を得ることができます。名前にワイルドカード文字を使用した場合は, 各呼び出しはリストで次のオブジェクト (キュー, フォームなど) の情報を返します。リストの最後に達した場合は, 空文字列("")を返します。ワイルドカードが使用できるのはオブジェクト名だけで, オブジェクト番号には使用できません。

フラグ
\$GETQUI システム・サービスの QUI\$_SEARCH_FLAGS 入力項目コードに相当するキーワードのリストを, コンマで区切って指定します。これらのフラグは, \$GETQUI システム・サービス呼び出しで指定した, オブジェクト検索の範囲を定義します。検索フラグとして使用可能なキーワードを表 DCLI-8 に示します。

表 DCLI-8 F\$GETQUI キーワード

キーワード	有効な関数コード	説明
ALL_JOBS	DISPLAY_JOB	設定されたキュー・コンテキスト内のすべてのジョブを検索するよう F\$GETQUI に要求する。このフラグを設定しないと, F\$GETQUI は呼び出したユーザと同じユーザ名を持つジョブの情報のみ返す。
BATCH	DISPLAY_QUEUE DISPLAY_ENTRY	バッチ・キューを選択する。
EXECUTING_JOBS	DISPLAY_ENTRY DISPLAY_JOB	実行中ジョブを選択する。

(次ページに続く)

表 DCLI-8 (続き) F\$GETQUI キーワード

キーワード	有効な関数コード	説明
FREEZE_CONTEXT	DISPLAY_CHARACTERISTIC DISPLAY_ENTRY DISPLAY_FILE DISPLAY_FORM DISPLAY_JOB DISPLAY_MANAGER DISPLAY_QUEUE	ワイルド・カード操作時、操作対象が次に進めない。このフラグを設定しないと、コンテキストは次のオブジェクトに進む。
GENERIC	DISPLAY_ENTRY DISPLAY_QUEUE	総称キューを選択する。
HOLDING_JOBS	DISPLAY_ENTRY DISPLAY_JOB	保留ジョブを選択する。
PENDING_JOBS	DISPLAY_ENTRY DISPLAY_JOB	待ち状態のジョブを選択する。
PRINTER	DISPLAY_QUEUE DISPLAY_ENTRY	プリンタ・キューを選択する。
RETAINED_JOBS	DISPLAY_ENTRY DISPLAY_JOB	保持されたジョブを選択する。
SERVER	DISPLAY_QUEUE DISPLAY_ENTRY	サーバ・キューを選択する。
SYMBIONT	DISPLAY_QUEUE DISPLAY_ENTRY	すべての出力を選択する。 “PRINTER,SERVER,TERMINAL”と等価。
TERMINAL	DISPLAY_QUEUE DISPLAY_ENTRY	ターミナル・キューを選択する。
THIS_JOB	DISPLAY_ENTRY DISPLAY_FILE DISPLAY_JOB DISPLAY_QUEUE	バッチ・ジョブ(エントリ)呼び出し、実行されるコマンド・ファイル、またはバッチ・キュー呼び出しに関連するキューに関する、すべてのジョブ・ファイル情報を選択する。
TIMED_RELEASE_JOBS	DISPLAY_ENTRY DISPLAY_JOB	開始時刻の指定されたジョブを選択する。
WILDCARD	DISPLAY_CHARACTERISTIC DISPLAY_ENTRY DISPLAY_FORM DISPLAY_MANAGER DISPLAY_QUEUE	コンテキストを設定し保存する。コンテキストが保存されるため、次の操作はそのコンテキストに基づいて実行される。

説明

F\$GETQUI レキシカル関数は、\$GETQUI システム・サービスを呼び出し、キューに関する情報を戻します。戻される情報には、バッチ・キューやプリント・キューに登録されているジョブ、フォーム定義、およびシステム・ジョブ・キュー・ファイルに定義された属性が含まれます。

F\$GETQUI レキシカル関数は、ワイルドカードの操作やネストしたワイルドカード操作を含め、\$GETQUI システム・サービスのすべての機能を提供します。たとえば、ネストしたワイルドカードの操作では、他のオブジェクト内で定義された情報を

戻します。呼び出しを繰り返した後、選択したキューに含まれるジョブ、または選択したジョブに含まれるファイルを照会する場合、この機能は特に便利です。各々の呼び出しの後、システムは、後続の呼び出しに必要なキューまたはコンテキストを提供できるように、GQC(内部 GETQUI コンテキスト・ブロック)を保存します。

制限事項

ワイルドカードが使用された F\$GETQUI 呼び出しのために保存される GQC は、SHOW QUEUE または SHOW ENTRY など、DCL キュー関連のコマンドを実行すると破棄されます。この問題を避けるには、SPAWN コマンドを使用して、DCL コマンドを実行する新しいプロセスを作成してください。

詳細は、『OpenVMS System Services Reference Manual』の\$GETQUI システム・サービスの説明を参照してください。

F\$GETQUI レキシカル関数は、\$GETQUI システム・サービスで指定されたすべての項目の情報を戻します。\$GETQUI レキシカル関数に指定する項目、戻される情報、およびその情報のデータ・タイプを表 DCLI-9 に示します。

表 DCLI-9 F\$GETQUI 項目

項目	データ・タイプ	戻される情報
ACCOUNT_NAME ¹	文字列	指定したジョブの所有者のアカウント名。
AFTER_TIME	文字列	指定したジョブを実行するシステム時刻。
ASSIGNED_QUEUE_NAME ¹	文字列	F\$GETQUI 呼び出しで指定した論理キューが割り当てられる実行キュー名。
AUTOSTART_ON	文字列	キューが始まる位置を示すノード、またはノード装置の組合せリスト。
BASE_PRIORITY	整数値	バッチ実行キューでのジョブの優先順位、または出力キューを制御するシンビオント・プロセスの優先順位。
CHARACTERISTICS ¹	文字列	指定したキューまたはジョブに関する属性。
CHARACTERISTIC_NAME	文字列	指定した属性名。
CHARACTERISTIC_NUMBER	整数値	指定した属性番号。
CHECKPOINT_DATA ¹	文字列	指定したバッチ・ジョブが再スタートする時の、シンボル BATCH\$RESTART の値。
CLI ¹	文字列	指定したバッチ・ジョブを実行するコマンド言語インタプリタ (CLI) 名。ファイル指定では装置名を SYS\$SYSTEM、ファイル・タイプを EXE とみなす。
COMPLETED_BLOCKS	整数値	指定したプリント・ジョブに対して処理したシンビオントのブロック数。プリント・ジョブにのみ適用される。
CONDITION_VECTOR ¹	整数値	3 つのロングワードのベクトル。最初のロングワードは指定したジョブの終了状態を示す。第 2、第 3 のロングワードはプリント・ジョブの付加状態を示す。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は、読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・タイプ	戻される情報
CPU_DEFAULT	文字列	キューに対して指定した省略時の CPU 時間をデルタ時間で表す。バッチ実行キューにのみ適用される。
CPU_LIMIT ¹	文字列	指定したジョブまたはキューに対して指定した最大 CPU 時間の上限をデルタ時間で表す。バッチ・ジョブ、およびバッチ実行キューにのみ適用される。
DEFAULT_FORM_NAME	文字列	指定した出力キューに関連する省略時のフォーム名。
DEFAULT_FORM_STOCK	文字列	省略時のフォームがプリントされるように指定した用紙名。
DEVICE_NAME	文字列	指定した実行キューがあるノードまたは装置(あるいはその両方)。出力実行キューの場合は、装置名のみが戻される。ノード名は、複合アーキテクチャ OpenVMS Cluster システムでのみ使用される。キューを実行するプロセスのシステム・パラメータ SCSNODE により、ノード名が指定される。 バッチ実行キューの場合は、空文字列("")が戻される。バッチ・キューが実行しているノード名を得るには、SCSNODE_NAME を使用する。
ENTRY_NUMBER	整数値	指定したジョブのキュー・エントリ番号。
EXECUTING_JOB_COUNT	整数値	現在実行中のキューにあるジョブ数。
FILE_BURST	文字列	ファイルの前にバースト・ページおよびフラグ・ページがプリントされている場合は TRUE, そうでない場合は FALSE を戻す。
FILE_CHECKPOINTED ¹	文字列	指定したファイルにチェックポイントが設定されている場合は TRUE, そうでない場合は FALSE を戻す。
FILE_COPIES ¹	整数値	指定したファイルが処理される時間。出力実行キューにのみ適用される。
FILE_COPIES_DONE ¹	整数値	指定したファイルが処理されていた時間。出力実行キューにのみ適用される。
FILE_COUNT	整数値	指定したジョブ中のファイル数。
FILE_DELETE	文字列	要求が実行された後にファイルが削除される場合は TRUE, そうでない場合は FALSE を戻す。
FILE_DEVICE ¹	文字列	選択したファイルを一意に識別する内部ファイル装置値。RMS NAM ブロックの以下のフィールドを指定する。 NAM\$T_DVI (16 バイト)
FILE_DID ¹	文字列	選択したファイルを一意に識別する内部ファイル DID 値。RMS NAM ブロックの以下のフィールドを指定する。 NAM\$W_DID (6 バイト)
FILE_DOUBLE_SPACE	文字列	シンピオントがダブル・スペーシングでファイルをフォーマットしている場合は TRUE, そうでない場合は FALSE を戻す。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は、読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
FILE_EXECUTING ¹	文字列	指定したファイルが処理中である場合は TRUE, そうでない場合は FALSE を戻す。
FILE_FLAG	文字列	ファイルの前にフラグ・ページがプリントされる場合は TRUE, そうでない場合は FALSE を戻す。
FILE_FLAGS ¹	整数値	指定したファイル用に選択された処理オプション。整数値はビット・フィールドを表す。フィールドの各ビット設定を確認するためには, FILE_FLAGS のかわりに次の項目のいずれか 1 つを使用する。 FILE_BURST FILE_DELETE FILE_DOUBLE_SPACE FILE_FLAG FILE_PAGE_HEADER FILE_PAGINATE FILE_PASSALL FILE_TRAILER
FILE_IDENTIFICATION ¹	文字列	選択したファイルを一意に識別する内部ファイル識別値。 RMS NAM ブロックの以下のファイル識別フィールドを指定する。 NAM\$W_FID (6 バイト)
FILE_PAGE_HEADER	文字列	各出力ページにヘッダがプリントされる場合は TRUE, そうでない場合は FALSE を戻す。
FILE_PAGINATE	文字列	下マージンで改ページを挿入して編集する場合は TRUE, そうでない場合は FALSE を戻す。
FILE_PASSALL	文字列	PASSALL モードでプリントする場合は TRUE, そうでない場合は FALSE を戻す。
FILE_SETUP_MODULES ¹	文字列	装置制御ライブラリから抽出され, 指定したファイルがプリントされる前にコピーするテキスト・モジュール名。出力実行キューでのみ適用される。
FILE_SPECIFICATION ¹	文字列	F\$GETQUI が情報を戻すファイルの, 完全修飾 RMS ファイル指定。
FILE_STATUS ¹	整数値	ファイル状態情報。整数値はビット・フィールドを表す。フィールドの各ビット設定を検索するためには, FILE_STATUS のかわりに次の項目のいずれか 1 つを使用する。 FILE_CHECKPOINTED FILE_EXECUTING
FILE_TRAILER	文字列	ファイルに続いてトレーラ・ページがプリントされる場合は TRUE, そうでない場合は FALSE を戻す。
FIRST_PAGE ¹	整数値	指定したファイルのプリントを始めるページ番号。出力実行キューでのみ適用される。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は, 読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
FORM_DESCRIPTION	文字列	ユーザおよびオペレータに、指定したフォームを示すテキスト文字列。
FORM_FLAGS	整数値	指定フォームのために選択された処理オプション。フィールドの各ビット設定を確認するためには、FORM_FLAGS のかわりに次の項目のいずれか 1 つを使用する。 FORM_SHEET_FEED FORM_TRUNCATE FORM_WRAP
FORM_LENGTH	整数値	指定したフォームの物理的な長さを行数で表す。出力実行キューでのみ適用される。
FORM_MARGIN_BOTTOM	整数値	指定したフォームの下マージンを行数で表す。
FORM_MARGIN_LEFT	整数値	指定したフォームの左マージンを文字数で表す。
FORM_MARGIN_RIGHT	整数値	指定したフォームの右マージンを文字数で表す。
FORM_MARGIN_TOP	整数値	指定したフォームの上マージンを行数で表す。
FORM_NAME ¹	文字列	指定したジョブまたはキューに関連する、指定したフォーム名、あるいはマウントされたフォーム名。
FORM_NUMBER	整数値	指定したフォーム数。
FORM_SETUP_MODULES	文字列	装置制御ライブラリから抽出され、指定したフォームでファイルがプリントされる前にコピーするテキスト・モジュール名。出力実行キューでのみ適用される。
FORM_SHEET_FEED	文字列	他の用紙を挿入できるように、物理ページの終わりにシンピオント一時停止をする場合は TRUE、そうでない場合は FALSE を戻す。
FORM_STOCK ¹	文字列	指定したフォームがプリントされる用紙名。
FORM_TRUNCATE	文字列	プリンタが右マージンを越えている文字を捨てる場合は TRUE、そうでない場合は FALSE を戻す。
FORM_WIDTH	整数値	指定したフォームの幅。
FORM_WRAP	文字列	後続の行で右マージンを越えている文字をプリントする場合は TRUE、そうでない場合は FALSE を戻す。
GENERIC_TARGET	文字列	指定した汎用キューからの作業を受け入れられる実行キュー名。汎用キューにのみ適用される。
HOLDING_JOB_COUNT	整数値	明示的にリリースまで保留になっているキューに登録されているジョブ数。
INTERVENING_BLOCKS	整数値	現在の F\$GETQUI 呼び出しの間にスキップされた、待ち状態のジョブに関連するブロック数。これらは、F\$GETQUI 呼び出しの選択基準と一致しないため報告されない。
INTERVENING_JOBS	整数値	F\$GETQUI 呼び出しの間にスキップされた、待ち状態のジョブ数。これらは、F\$GETQUI 呼び出しの選択基準と一致しないため報告されない。
JOB_ABORTING	文字列	ジョブを強制終了させる場合は TRUE、そうでない場合は FALSE を戻す。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は、読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
JOB_COMPLETION_QUEUE ¹	文字列	指定したジョブが実行されたキュー名。
JOB_COMPLETION_TIME ¹	文字列	指定したジョブが完了した時刻。
JOB_COPIES ¹	整数値	指定したプリント・ジョブが繰り返される回数。
JOB_COPIES_DONE ¹	整数値	指定したプリント・ジョブが繰り返された回数。
JOB_CPU_LIMIT ¹	文字列	ジョブに対して CPU 時間上限が指定されている場合は TRUE, そうでない場合は FALSE を戻す。
JOB_ERROR_RETENTION ¹	文字列	ジョブが失敗して終了した場合に、そのままキューに残しておくようにユーザが要求した場合は TRUE, そうでない場合は FALSE を戻す。
JOB_EXECUTING	文字列	指定したジョブが実行中あるいはプリント中である場合は TRUE, そうでない場合は FALSE を戻す。
JOB_FILE_BURST ¹	文字列	ジョブに対してバースト・ページのオプションが明示的に指定されている場合は TRUE, そうでない場合は FALSE を戻す。
JOB_FILE_BURST_ONE ¹	文字列	ジョブの最初のファイルの前でのみ、バースト・ページおよびフラグ・ページを入れる場合は TRUE, そうでない場合は FALSE を戻す。
JOB_FILE_FLAG ¹	文字列	ジョブに登録されているそれぞれのファイルの前にフラグ・ページを入れる場合は TRUE, そうでない場合は FALSE を戻す。
JOB_FILE_FLAG_ONE ¹	文字列	ジョブの最初のファイルの前でのみ、フラグ・ページを入れる場合は TRUE, そうでない場合は FALSE を戻す。
JOB_FILE_PAGINATE ¹	文字列	ジョブに対して明示的にページング・オプションを指定している場合は TRUE, そうでない場合は FALSE を戻す。
JOB_FILE_TRAILER ¹	文字列	ジョブの各ファイルの後にトレーラ・ページが続く場合は TRUE, そうでない場合は FALSE を戻す。
JOB_FILE_TRAILER_ONE ¹	文字列	ジョブの最後のファイルのコピーのみにトレーラ・ページが続く場合は TRUE, そうでない場合は FALSE を戻す。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は、読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
JOB_FLAGS ¹	整数値	指定したジョブに対して選択した処理オプション。整数値はビット・フィールドを表す。フィールドの各ビット設定を確認検索するためには、JOB_FLAGS のかわりに次の項目のいずれか 1 つを使用します。 JOB_CPU_LIMIT JOB_ERROR_RETENTION JOB_FILE_BURST JOB_FILE_BURST_ONE JOB_FILE_FLAG JOB_FILE_FLAG_ONE JOB_FILE_PAGINATE JOB_FILE_TRAILER JOB_FILE_TRAILER_ONE JOB_LOG_DELETE JOB_LOG_NULL JOB_LOG_SPOOL JOB_LOWERCASE JOB_NOTIFY JOB_RESTART JOB_RETENTION_TIME JOB_WSDEFAULT JOB_WSEXTENT JOB_WSQUOTA
JOB_HOLDING	文字列	明示的にリリースされるまでジョブが保留される場合は TRUE、そうでない場合は FALSE を戻す。
JOB_INACCESSIBLE	文字列	特定のジョブ、およびシステム・キュー・ファイルのファイル情報への読み込みアクセス権がない場合は TRUE、そうでない場合は FALSE を戻す。FALSE の場合、DISPLAY_JOB および DISPLAY_FILE 操作は、次の出力値項目コードの情報のみ戻す。 AFTER_TIME COMPLETED_BLOCKS ENTRY_NUMBER INTERVENING_BLOCKS INTERVENING_JOBS JOB_SIZE JOB_STATUS
JOB_LIMIT	整数値	指定したキューで同時に実行できるジョブ数。バッチ実行キューにのみ適用される。
JOB_LOG_DELETE ¹	文字列	プリントした後にログ・ファイルを削除する場合は TRUE、そうでない場合は FALSE を戻す。
JOB_LOG_NULL ¹	文字列	ログ・ファイルが作成されない場合は TRUE、そうでない場合は FALSE を戻す。
JOB_LOG_SPOOL ¹	文字列	ジョブの終了時に、ログ・ファイルがプリント・キューに登録される場合は TRUE、そうでない場合は FALSE を戻す。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は、読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
JOB_LOWERCASE ¹	文字列	大文字と小文字の両方が印刷できるプリンタでプリントされる場合は TRUE, そうでない場合は FALSE を戻す。
JOB_NAME ¹	文字列	指定したジョブ名。
JOB_NOTIFY ¹	文字列	ジョブが完了または強制終了した時に, ターミナルにメッセージを送信する場合は TRUE, そうでない場合は FALSE を戻す。
JOB_PENDING	文字列	ジョブが待ち状態である場合は TRUE, そうでない場合は FALSE を戻す。
JOB_PID	文字列	バッチ・ジョブを実行するプロセス識別番号 (PID)。
JOB_REFUSED	文字列	ジョブがシンピオントによって拒否され処理を待っている状態である場合は TRUE, そうでない場合は FALSE を戻す。
JOB_RESET_MODULES	文字列	装置制御ライブラリから抽出され, 指定したキューに登録された各ジョブがプリントされる前に, プリンタにコピーされたテキスト・モジュール名。出力実行キューでのみ適用される。
JOB_RESTART ¹	文字列	ジョブがシステム障害の後に再スタートする, または実行中に再度キューに登録できる場合は TRUE, そうでない場合は FALSE を戻す。
JOB_RETAINED	文字列	終了してもまだキューにジョブが残っている場合は TRUE, そうでない場合は FALSE を戻す。
JOB_RETENTION	文字列	ジョブの終了状態にかかわらず, キューにジョブが残るようにユーザが要求した場合は TRUE, そうでない場合は FALSE を戻す。
JOB_RETENTION_TIME ¹	文字列	キューにジョブを残すようにユーザが要求するまでのシステム時間。絶対時間またはデルタ時間で表す。
JOB_SIZE	整数値	指定したプリント・ジョブのブロック総数。
JOB_SIZE_MAXIMUM	整数値	指定したキューに登録できるプリント・ジョブの最大ブロック数。出力実行キューにのみ適用される。
JOB_SIZE_MINIMUM	整数値	指定したキューに登録できるプリント・ジョブの最小ブロック数。出力実行キューにのみ適用される。
JOB_STALLED	文字列	ジョブをプリントしている物理装置が一時停止しているためにジョブが一時停止している場合は TRUE, そうでない場合は FALSE を戻す。
JOB_STARTING	文字列	ジョブ・コントローラが処理を開始し, 出力シンピオントまたは他のノードのジョブ・コントローラと通信を開始していた場合は TRUE, そうでない場合は FALSE を戻す。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は, 読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
JOB_STATUS	整数値	指定したジョブの状態フラグ。整数値はビット・フィールドを表す。フィールドの各ビット設定を確認するためには、JOB_STATUS のかわりに次の項目のいずれか 1 つを使用する。 JOB_ABORTING JOB_EXECUTING JOB_HOLDING JOB_INACCESSIBLE JOB_REFUSED JOB_REQUEUE JOB_RESTART JOB_RETAINED JOB_STARTING JOB_TIMED_RELEASE JOB_SUSPENDED JOB_PENDING
JOB_SUSPENDED	文字列	ジョブが中断されている場合は TRUE、そうでない場合は FALSE を戻す。
JOB_TIMED_RELEASE	文字列	指定した時刻までジョブが実行を待っている場合は TRUE、そうでない場合は FALSE を戻す。
JOB_WSDEFAULT ¹	文字列	ジョブに対して省略時のワーキング・セット・サイズが指定されている場合は TRUE、そうでない場合は FALSE を戻す。
JOB_WSEXTENT ¹	文字列	ジョブに対してワーキング・セット拡張が指定されている場合は TRUE、そうでない場合は FALSE を戻す。
JOB_WSQUOTA ¹	文字列	ジョブに対してワーキング・セット・クォータが指定されている場合は TRUE、そうでない場合は FALSE を戻す。
LAST_PAGE ¹	整数値	指定したファイルのプリントが終了するページ番号。出力実行キューにのみ適用される。
LIBRARY_SPECIFICATION	文字列	指定したキューの装置制御ライブラリ名。装置とディレクトリ名は SYS\$LIBRARY、ファイル・タイプは.TLB とみなす。出力実行キューにのみ適用される。
LOG_QUEUE ¹	文字列	指定したバッチ・ジョブをプリントするために、ログ・ファイルが作成されるキュー名。出力実行キューにのみ適用される。
LOG_SPECIFICATION ¹	文字列	ジョブに対して指定したログ・ファイル名。バッチ・ジョブにのみ適用される。JOB_LOG_NULL コードを使用して、ログ・ファイルを作成するか決定する。
MANAGER_NAME	文字列	キュー・マネージャ名。
MANAGER_NODES	文字列	キュー・マネージャを実行するノード名。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は、読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
MANAGER_STATUS	整数値	指定したキュー・マネージャの状態フラグ。フィールドの各ビット設定を確認するためには、MANAGER_STATUS のかわりに次の項目のいずれか 1 つを使用する。 MANAGER_FAILOVER MANAGER_RUNNING MANAGER_START_PENDING MANAGER_STARTING MANAGER_STOPPED MANAGER_STOPPING
NOTE ¹	文字列	指定したジョブの、ジョブ・フラグ・ページおよびファイル・フラグ・ページにプリントされる注意書き。出力実行キューにのみ適用される。
OPERATOR_REQUEST ¹	文字列	指定したジョブを実行する前にキュー・オペレータに送るメッセージ。出力実行キューにのみ適用される。
OWNER_UIC ¹	文字列	指定したキュー所有者の利用者識別コード。
PAGE_SETUP_MODULES	文字列	装置制御ライブラリから抽出され、フォームの各ページがプリントされる前にプリンタにコピーされたテキスト・モジュール名。
PARAMETER_1 to PARAMETER_8 ¹	文字列	シンボル P1 ~ P8 それぞれの値となる、利用者定義パラメータの値。
PENDING_JOB_BLOCK_COUNT	整数値	キューで待ち状態になっているすべてのジョブのブロック総数 (出力実行キューのみ)。
PENDING_JOB_COUNT	整数値	待ち状態になっているジョブ数。
PENDING_JOB_REASON	整数値	ジョブが待ち状態になっている理由。整数値はビット・フィールドを表す。フィールドの各ビット設定を確認するためには、PENDING_JOB_REASON のかわりに次の項目のいずれか 1 つを使用する。 PEND_CHAR_MISMATCH PEND_JOB_SIZE_MAX PEND_JOB_SIZE_MIN PEND_LOWERCASE_MISMATCH PEND_NO_ACCESS PEND_QUEUE_BUSY PEND_QUEUE_STATE PEND_STOCK_MISMATCH
PEND_CHAR_MISMATCH	文字列	実行キューで使用できない属性を要求している場合は TRUE、そうでない場合は FALSE を戻す。
PEND_JOB_SIZE_MAX	文字列	ジョブのブロック数が実行キューのブロック上限を越えている場合は TRUE、そうでない場合は FALSE を戻す。
PEND_JOB_SIZE_MIN	文字列	ジョブのブロック数が実行キューのブロック下限よりも少ない場合は TRUE、そうでない場合は FALSE を戻す。
PEND_LOWERCASE_MISMATCH	文字列	ジョブが小文字用のプリンタを必要とする場合は TRUE、そうでない場合は FALSE を戻す。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は、読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・タイプ	戻される情報
PEND_NO_ACCESS	文字列	ジョブ所有者が実行キューへのアクセス権がない場合は TRUE, そうでない場合は FALSE を戻す。
PEND_QUEUE_BUSY	文字列	キューのジョブ制限と現在実行中のジョブ数が等しいために、ジョブが待ち状態になっている場合は TRUE, そうでない場合は FALSE を戻す。
PEND_QUEUE_STATE	文字列	実行キューが実行中のオープン状態でないためにジョブが待ち状態になっている場合は TRUE, そうでない場合は FALSE を戻す。
PEND_STOCK_MISMATCH	文字列	ジョブのフォームで必要なストックの形式が、実行キューでマウントされた形式と一致しない場合は TRUE, そうでない場合は FALSE を戻す。
PRIORITY ¹	整数値	指定したジョブのスケジューリング優先順位。
PROCESSOR	文字列	指定したキューから実行するプリント・ジョブを実行するシンビオント・イメージ名。
PROTECTION ¹	文字列	指定したキューの保護マスク。
QUEUE_ACL_SPECIFIED	文字列	キューに対してアクセス制御リストが指定されている場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_ALIGNING	文字列	現在アラインメント・ページをプリントしている場合は TRUE, そうでない場合は FALSE を戻す。 START /QUEUE/ALIGN コマンドを使用して再スタートしたときに、アライメント・ページをプリントする。
QUEUE_AUTOSTART	文字列	指定したキューが AUTOSTART キューと指定されている場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_AUTOSTART_INACTIVE	文字列	キューが、自動的に始動しないよう設定した自動起動キューである場合は TRUE, そうでない場合は FALSE を戻す。 TRUE の場合は、 START/QUEUE または INIT /QUEUE/START コマンドを実行して再起動させる。
QUEUE_AVAILABLE	文字列	キューで 2 つ以上のジョブを処理していて、さらに追加のジョブを処理できるキューである場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_BATCH	文字列	キューが、バッチ・キューまたは汎用バッチ・キューである場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_BUSY	文字列	現在実行中のジョブ数が、キューのジョブ制限と等しい場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_CLOSED	文字列	キューがオープン状態になるまでクローズされ、新しいジョブを受け入れない場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_CPU_DEFAULT	文字列	キューに登録されたすべてのジョブに対して省略時の CPU 時間制限が指定されている場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_CPU_LIMIT	文字列	キューに登録されたすべてのジョブに対して省略時の最大 CPU 時間制限が指定されている場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_DESCRIPTION	文字列	INITIALIZE/QUEUE コマンドに/DESCRIPTION 修飾子を指定して定義したキューの説明。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は、読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
QUEUE_DIRECTORY	文字列	キュー・マネージャのキュー・データベース・ディレクトリの装置およびディレクトリ指定。
QUEUE_FILE_BURST	文字列	キューで実行する各ジョブの各ファイルの前に、バースト・ページおよびフラグ・ページを入れる場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_FILE_BURST_ONE	文字列	キューが実行する各ジョブの最初のファイルの前にのみ、バースト・ページおよびフラグ・ページを入れる場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_FILE_FLAG	文字列	キューが実行する各ジョブの各ファイルの前にフラグ・ページを入れる場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_FILE_FLAG_ONE	文字列	キューが実行する各ジョブの最初のファイルの前にのみ、フラグ・ページを入れる場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_FILE_PAGINATE	文字列	このキューが実行する各ジョブの出力シンピオントが出力をページングする場合は TRUE, そうでない場合は FALSE を戻す。出力シンピオントは、出力がフォームの下マージンに達するとフォーム・フィードを挿入してページングを行う。
QUEUE_FILE_TRAILER	文字列	キューで実行した各ジョブの各ファイルの後にトレーラ・ページが続く場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_FILE_TRAILER_ONE	文字列	キューで実行した各ジョブの最後のファイルの最後コピーの後にのみトレーラ・ページが続く場合は TRUE, そうでない場合は FALSE を戻す。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
QUEUE_FLAGS	整数値	<p>指定したキューに対して選択された処理オプション。整数値はビット・フィールドを表す。フィールドの各ビット設定を確認するためには、QUEUE_FLAGS のかわりに次の項目のいずれか 1 つを使用する。</p> <p> QUEUE_ACL_SPECIFIED QUEUE_AUTOSTART QUEUE_BATCH QUEUE_CPU_DEFAULT QUEUE_CPU_LIMIT QUEUE_FILE_BURST QUEUE_FILE_BURST_ONE QUEUE_FILE_FLAG QUEUE_FILE_FLAG_ONE QUEUE_FILE_PAGINATE QUEUE_FILE_TRAILER QUEUE_FILE_TRAILER_ONE QUEUE_GENERIC QUEUE_GENERIC_SELECTION QUEUE_JOB_BURST QUEUE_JOB_FLAG QUEUE_JOB_SIZE_SCHED QUEUE_JOB_TRAILER QUEUE_NO_INITIAL_FF QUEUE_PRINTER QUEUE_RECORD_BLOCKING QUEUE_RETAIN_ALL QUEUE_RETAIN_ERROR QUEUE_SWAP QUEUE_TERMINAL QUEUE_WSDEFAULT QUEUE_WSEXTENT QUEUE_WSQUOTA </p>
QUEUE_GENERIC	文字列	キューが汎用キューである場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_GENERIC_SELECTION	文字列	キューが、汎用キューから作業を受けることができる実行キューである場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_IDLE	文字列	キューが、ジョブの処理が可能であるが現在処理中でないキューである、またはキューの実行を遅らせる機能を持つ汎用キューである場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_JOB_BURST	文字列	キューで実行した各ジョブの前にバースト・ページおよびフラグ・ページを入れる場合は TRUE, そうでない場合は FALSE を戻す。
QUEUE_JOB_FLAG	文字列	キューで実行した各ジョブの前にフラグ・ページを入れる場合は TRUE, そうでない場合は FALSE を戻す。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
QUEUE_JOB_SIZE_SCHED	文字列	キューで実行したジョブが、サイズに基づいて (つまり、サイズが小さいジョブの方が処理される優先順位が高いという基準に従って) スケジューリングされた場合は TRUE、総出ない場合は FALSE を戻す。出力キューにのみ適用される。
QUEUE_JOB_TRAILER	文字列	キューで実行した各ジョブの後にトレーラ・ページが続く場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_LOWERCASE	文字列	大文字も小文字もプリントできるプリンタにキューが関連付けられている場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_NAME ¹	文字列	指定したキューの名前、または指定したジョブを含むキューの名前。
QUEUE_PAUSED	文字列	現在キューに登録されているすべてのジョブの処理が一時的に停止されている場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_PAUSING	文字列	一時的にキューが停止している場合は TRUE、そうでない場合は FALSE を戻す。一時的にキューが停止していると、現在実行中のジョブが終了しても、一時的に新しいジョブは実行できなくなる。
QUEUE_PRINTER	文字列	キューがプリンタ・キューである場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_RECORD_BLOCKING	文字列	シンピオントが連結またはブロック化して、出力レコードを出力装置に送る場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_REMOTE	文字列	キューが、ローカル・ノードに接続していない物理装置に割り当てられている場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_RESETTING	文字列	キューが再設定および停止している場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_RESUMING	文字列	キューが休止のあと再起動している場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_RETAIN_ALL	文字列	キューで実行したすべてのジョブが実行後もそのままキューにある場合は TRUE、そうでない場合は FALSE を戻す。終了したジョブは終了状態の印がつけられる。
QUEUE_RETAIN_ERROR	文字列	失敗して終了したジョブだけがキューに残っている場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_SERVER	文字列	サーバ・シンピオントに処理が指示された場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_STALLED	文字列	キューに割り当てられた物理装置が停止している (つまり、登録された最後の入出力要求が正常終了していない) 場合は TRUE、停止していない場合は FALSE を戻す。
QUEUE_STARTING	文字列	キューがスタートしている場合は TRUE、そうでない場合は FALSE を戻す。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は、読み込み (R) アクセス権が必要です。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・ タイプ	戻される情報
QUEUE_STATUS	整数値	指定したキューの状態フラグ。整数値はビット・フィールドを表す。フィールドの各ビット設定を確認するためには、QUEUE_STATUS のかわりに次の項目のいずれか 1 つを使用する。 <div> <div>QUEUE_ALIGNING</div> <div>QUEUE_AUTOSTART</div> <div>QUEUE_AUTOSTART_INACTIVE</div> <div>QUEUE_AVAILABLE</div> <div>QUEUE_BUSY</div> <div>QUEUE_CLOSED</div> <div>QUEUE_IDLE</div> <div>QUEUE_LOWERCASE</div> <div>QUEUE_PAUSED</div> <div>QUEUE_PAUSING</div> <div>QUEUE_REMOTE</div> <div>QUEUE_RESETTING</div> <div>QUEUE_RESUMING</div> <div>QUEUE_SERVER</div> <div>QUEUE_STALLED</div> <div>QUEUE_STARTING</div> <div>QUEUE_STOP_PENDING</div> <div>QUEUE_STOPPED</div> <div>QUEUE_STOPPING</div> <div>QUEUE_UNAVAILABLE</div> </div>
QUEUE_STOP_PENDING	文字列	現在実行中のジョブが終了したときにキューが停止される場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_STOPPED	文字列	キューが停止された場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_STOPPING	文字列	キューが停止しようとしている場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_SWAP	文字列	キューで実行したジョブがスワップできる場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_TERMINAL	文字列	キューがターミナル・キューである場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_UNAVAILABLE	文字列	キューに割り当てられた物理装置が使用できない場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_WSDEFAULT	文字列	キューで実行した各ジョブに対して、それぞれ省略時のワーキング・セット・サイズが指定されている場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_WSEXTENT	文字列	キューで実行した各ジョブに対して、それぞれ省略時のワーキング・セット超過値が指定されている場合は TRUE、そうでない場合は FALSE を戻す。
QUEUE_WSQUOTA	文字列	キューで実行した各ジョブに対して、それぞれ省略時のワーキング・セット・クォータが指定されている場合は TRUE、そうでない場合は FALSE を戻す。

(次ページに続く)

表 DCLI-9 (続き) F\$GETQUI 項目

項目	データ・タイプ	戻される情報
RAD (Alpha/I64 のみ)	整数値	RAD の値。値 "-1" は、キューに RAD の値が割り当てられていないことを示す。AlphaServer GS シリーズ・システム上でのみサポートされます。
REQUEUE_QUEUE_NAME ¹	文字列	指定したジョブが再割り当てされているキューの名前。
RESTART_QUEUE_NAME ¹	文字列	ジョブを再スタートした場合にジョブが置かれるキューの名前。
RETAINED_JOB_COUNT	整数値	終了してキューに残っているジョブとエラーになったままのジョブを合わせた数。
SCSNODE_NAME	文字列	指定したキューで開始するジョブが実行される OpenVMS ノードの 6 バイトの名前。ノード名は、ターゲット・ノードのシステム・パラメータ SCSNODE の値に一致する。
SECURITY_INACCESSIBLE	文字列	ユーザが指定したキューへの読み込みアクセス権を持っている場合は TRUE、そうでない場合は FALSE を戻す。
SUBMISSION_TIME ¹	文字列	指定したジョブがキューに登録された時刻。
TIMED_RELEASE_JOB_COUNT	整数値	指定した時刻まで保留になっているジョブ数。
UIC ¹	文字列	指定したジョブ所有者の利用者識別コード (UIC)。
USERNAME ¹	文字列	指定したジョブ所有者のユーザ名。
WSDEFAULT ¹	整数値	指定したジョブまたはキューに対して指定された省略時のワーキング・セット・サイズ。バッチ・ジョブ, 実行キュー, および出力キューにのみ適用される。
WSEXTENT ¹	整数値	指定したジョブまたはキューに対して指定された省略時のワーキング・セット超過値。バッチ・ジョブ, 実行キュー, および出力キューにのみ適用される。
WSQUOTA ¹	整数値	指定したジョブまたはキューに対して指定された省略時のワーキング・セット・クォータ。バッチ・ジョブ, 実行キュー, および出力キューにのみ適用される。

¹DISPLAY_ENTRY, DISPLAY_JOB, または DISPLAY_FILE を使用する場合は、読み込み (R) アクセス権が必要です。

例

1. `$ BLOCKS = F$GETQUI("DISPLAY_ENTRY", "JOB_SIZE", 1347)`

この例では、プリント・ジョブ 1347 のブロック・サイズを得るために F\$GETQUI レキシカル関数が使用されています。F\$GETQUI が戻す値は、プリント・ジョブ 1347 に関連するファイルが占有している総ブロック・サイズです。

2. `$ IF F$GETQUI("DISPLAY_QUEUE", "QUEUE_STOPPED", "VAX1_BATCH").EQS.
"TRUE" THEN GOTO 500`

この例では、F\$GETQUI レキシカル関数を使用して VAX1_BATCH キューが停止状態かどうかを調べています。VAX1_BATCH が停止状態であれば TRUE、そうでなければ FALSE が戻されます。VAX1_BATCH キューがシステムに存在しない場合は、空文字列("") が戻されます。

```

3. ! This command procedure shows all queues and the jobs in them.
$ TEMP = F$GETQUI("")
$ QLOOP:
$ QNAME = F$GETQUI("DISPLAY_QUEUE","QUEUE_NAME","*")
$ IF QNAME .EQS. "" THEN EXIT
$ WRITE SYS$OUTPUT ""
$ WRITE SYS$OUTPUT "QUEUE: ", QNAME
$ JLOOP:
$ NOACCESS = F$GETQUI("DISPLAY_JOB","JOB_INACCESSIBLE","", "ALL_JOBS")
$ IF NOACCESS .EQS. "TRUE" THEN GOTO JLOOP
$ IF NOACCESS .EQS. "" THEN GOTO QLOOP
$ JNAME = F$GETQUI("DISPLAY_JOB","JOB_NAME","", "FREEZE_CONTEXT")
$ WRITE SYS$OUTPUT "    JOB: ", JNAME
$ GOTO JLOOP

```

この例は、システムのすべてのキュー、およびユーザが READ アクセスを持つすべてのジョブを表示するコマンド・プロシージャです。外側のループでは、ワイルド・カードを使用してキューの表示が実行されます。キュー情報を得る時には、アクセス権の確認は必要ありません。これは、すべてのユーザは暗黙でキュー属性への READ アクセス権を持っているからです。キュー名にワイルドカード (“*”) が指定されているため、キューのリスト (ワイルドカード・コンテキスト) が保持されます。

内側のループでは、すべてのジョブの情報を得るために、ワイルドカード表示キュー・モードからネストしたワイルドカードを入力しています。このループでは、ジョブに対しては暗黙の READ アクセス権は与えられていないので、ジョブ情報を得る事ができるかどうかを確認しなければなりません。ジョブに対する要求の中で FREEZE_CONTEXT キーワードを使用して、ワイルドカード・コンテキストを次のオブジェクトへ進めないようにしています。ジョブ名が検索され表示されると、制御は次のジョブに移ります。この時には、FREEZE_CONTEXT キーワードは使用されていないので、コンテキストは次のジョブに進みます。このワイルドカード・キュー・コンテキストは、一致するキューがなくなるまで維持されます。一致するキューがなくなると、F\$GETQUI は、一致するオブジェクトがこれ以上ないことを示すために、空文字列(“”)を戻します。

```
4. $ THIS_NODE = F$EDIT(F$GETSYI("SCSNODE"), "COLLAPSE")
$ TEMP = F$GETQUI("CANCEL_OPERATION")
$ SET NOON
$LOOP:
$ QUEUE = F$GETQUI("DISPLAY_QUEUE", "QUEUE_NAME", "*", "WILDCARD")
$ IF QUEUE .EQS. "" THEN GOTO ENDLOOP
$ IF THIS_NODE .EQS.-
F$GETQUI("DISPLAY_QUEUE", "SCSNODE_NAME", "*", "WILDCARD, FREEZE_CONTEXT")
$ THEN
$   IF .NOT.-
F$GETQUI("DISPLAY_QUEUE", "QUEUE_AUTOSTART", "*", "WILDCARD, FREEZE_CONTEXT")-
THEN START/QUEUE 'QUEUE'
$ ENDIF
$ GOTO LOOP
$ENDLOOP:
$ SET ON
```

この例は、ローカル・クラスタ・ノードに関するすべてのキューを調べ、自動起動になっていないキューを開始させるコマンド・プロシージャです。

このコマンド・プロシージャは、ローカル・システムのノード名を得て、F\$GETQUI コンテキストをクリアにします。キューが既に起動している場合は、START QUEUE コマンドの結果としてこのコマンド・プロシージャが強制終了されることのないように、エラー処理を無効にしておきます。

ループ内では、F\$GETQUI レキシカル関数は、キュー・リストから次のキュー名を取り出します。次のキューがない場合は、ループを終了します。

次の IF は、ローカル・ノードでキューが実行するかをチェックしています。実行する場合は、次に、キューが自動起動になっているかをチェックします。ローカル・ノードでキューが実行しない場合は、START コマンドを使用してキューを起動させます。その後、ループを繰り返します。

最後のコマンドは、DCL エラー処理を以前の設定にリストアします。


```

5. $ IF p1.EQS."" THEN INQUIRE p1 "Queue name"
$   TEMP = F$GETQUI("")
$   QLOOP:
$     QNAME = F$GETQUI("DISPLAY_QUEUE","QUEUE_NAME",p1,"WILDCARD")
$     IF QNAME .EQS. "" THEN EXIT
$     WRITE SYS$OUTPUT ""
$     WRITE SYS$OUTPUT "QUEUE: ", QNAME
$     JLOOP:
$       RETAINED = F$GETQUI("DISPLAY_JOB","JOB_RETAINED",,"ALL_JOBS")
$       IF RETAINED .EQS. "" THEN GOTO QLOOP
$       Entry = F$GETQUI("DISPLAY_JOB","ENTRY_NUMBER",,"FREEZE_CONTEXT,ALL_JOBS")
$       WRITE SYS$OUTPUT "    Entry: ''Entry' Retained: ''RETAINED'"
$       IF RETAINED.EQS."TRUE" THEN DELETE/ENTRY='Entry'
$       GOTO JLOOP

```

このコマンド・プロシージャは、指定されたキューに保持されているすべてのエントリを削除します。ワイルドカードを使用することもできます。

```

6. $ WRITE SYS$OUTPUT F$GETQUI("DISPLAY_QUEUE","RAD","BATCHQ1")
   -1

```

この例は、RAD の値を返します。値 "-1" は、キューに RAD の値が割り当てられていないことを示します。

F\$GETSYI

\$GETSYI システム・サービスを呼び出し、ローカル・システム (複合アーキテクチャの OpenVMS Cluster の場合は、ローカル・ノード) に関する状態情報と識別情報を戻します。

フォーマット

F\$GETSYI(項目[, ノード名][, クラスタ id])

戻り値

要求する項目に応じて、整数または文字列になります。

引数

項目

ローカル・ノード (使用しているシステムが OpenVMS Cluster の場合は、OpenVMS Cluster 内の他のノード) に関して報告される情報の種類を指定します。項目は、文字列式として指定します。

また、『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』に示すシステム・パラメータも指定することができます。

ノード名

情報が報告される OpenVMS Cluster のノードを指定します。ノードは、文字列式として指定します。ノード名に(*)や(%)のワイルド・カード文字は使用できません。

クラスタ id

情報が報告される、クラスタ・ノード識別番号を指定します。

クラスタ内のすべてのノードの情報を取得するには、F\$CSID レキシカル関数を使用して各クラスタ・システム識別番号を取得し、F\$GETSYI のクラスタ id 引数を使用して各ノードの情報を集めます。

説明

F\$GETSYI レキシカル関数は、\$GETSYI システム・サービスを呼び出し、ローカル・システム (使用しているシステムがクラスタの一部である場合は、ローカル OpenVMS Cluster のノード) の状態および識別情報を戻します。F\$GETSYI 関数

は、\$GETSYI システム・サービスで指定できる項目の情報を戻します。\$GETSYI システム・サービスについての詳細は、『OpenVMS System Services Reference Manual』を参照してください。

ノード名またはクラスタ id 引数を指定すると、情報を得るノードを指定することができます。ただし、これらの引数を同時に指定することはできません。

F\$GETSYI レキシカル関数に指定できる項目を表 DCLI-10 に示します。

表 DCLI-10 F\$GETSYI 項目

項目	データ・ タイプ	戻される情報
‡ACTIVE_CPU_MASK	整数値	CPU インデックス付きのビットベクトルを表す値。特定のビット位置がセットされているとき、その CPU ID 値を持つプロセッサはインスタンスのアクティブ・セットのメンバであり、OpenVMS SMP スケジューリング活動に参加している。
ACTIVECPU_CNT	整数値	対称的マルチプロセッサ (SMP) システムの現在のブートでアクティブな CPU の数。
ARCHFLAG	整数値	システムのアーキテクチャ・フラグ。
ARCH_NAME	文字列	CPU アーキテクチャ名。OpenVMS Alpha の場合は Alpha。OpenVMS VAX の場合は VAX。
ARCH_TYPE	整数値	CPU アーキテクチャのタイプ。VAX の場合は 1、Alpha の場合は 2。
‡AVAIL_CPU_MASK	整数値	CPU インデックス付きのビットベクトルを表す値。特定のビット位置がセットされているとき、その CPU ID 値を持つプロセッサは、インスタンスの構成セットのメンバであり、パーティションによって所有され、発行元インスタンスによって制御される。
AVAILCPU_CNT	整数値	システムが承認した CPU 回数。
BOOTTIME	文字列	システムがブートされた時刻。
CHARACTER_EMULATED	文字列	文字列命令が CPU でエミュレートされた場合は TRUE、そうでない場合は FALSE を戻す。
CLUSTER_EVOTES	整数値	クラスタ内のポート (投票数) の総数。
CLUSTER_FSYSID	文字列	クラスタ (基本ノード) 内でブートする最初のノードのシステム識別番号。16 進数を含む文字列を戻す。
CLUSTER_FTIME	文字列	クラスタで最初のノードがブートされた時刻。
CLUSTER_MEMBER	文字列	ノードがローカル・クラスタのメンバである場合は TRUE、そうでない場合は FALSE を戻す。
CLUSTER_NODES	整数値	クラスタ内のノード総数を整数で表す。
CLUSTER_QUORUM	整数値	クラスタのクォーラム総数。

‡Alpha および I64 のみ

(次ページに続く)

表 DCLI-10 (続き) F\$GETSYI 項目

項目	データ・ タイプ	戻される情報
CLUSTER_VOTES	整数値	クラスタ内の投票総数。
‡CONSOLE_VERSION	文字列	コンソール・ファームウェア・バージョン。
CONTIG_GBLPAGES	整数値	未使用で連続したグローバル・ページの総数。
‡COMMUNITY_ID	整数値	ハード・パーティション内の発行元インスタンスのハードウェア・コミュニティ ID。パーティショニングをサポートする AlphaServer システム上でのみサポートされます。
‡CPU	整数値	VAX において、システム識別 (SID) 番号を表わすプロセッサ・タイプ。たとえば、1 は VAX-11/780、6 は VAX 8530、VAX 8550、VAX 8700、または VAX 8800 を表す。
‡CPU_AUTOSTART	整数値	コンマで区切られ、CPU ID のインデックスが付けられた 0 と 1 のリスト。値が 1 の項目は、特定の CPU が外部から現在のインスタンスに移行した場合、またはすでに所有されている状態で起動された場合に、OpenVMS アクティブ・セットになることを示す。
‡CPU_FAILOVER	整数値	コンマで区切られ、CPU ID のインデックスが付けられたパーティション ID 番号のリスト。現在のインスタンスがクラッシュした場合のプロセッサのデスティネーションを定義する。パーティショニングをサポートする AlphaServer システム上でのみサポートされます。
‡CPUCAP_MASK	文字列	コンマで区切られ、CPU ID のインデックスが付けられた 16 進値のリスト。それぞれの値は、ビットベクトルを表す。セットされているとき、対応するユーザ機能がその CPU で有効である。
‡CPUTYPE	整数値	Alpha において、ハードウェア再起動パラメータ・ブロック (HWRPB) に格納されるプロセッサ・タイプ。2 は DECchip 21064 プロセッサを表す。
CWLOGICALS	論理型	クラスタ単位の論理名データベースがその CPU で初期化されたことを示すフラグ。
DECIMAL_EMULATED	文字列	10 進数文字列の命令が CPU でエミュレートされている場合は TRUE、そうでない場合は FALSE を戻す。
DECNET_FULLNAME	文字列	DECnet Phase IV システムのノード名、または DECnet-Plus システムの完全なノード名。
‡VAX のみ		
‡Alpha および I64 のみ		

(次ページに続く)

表 DCLI-10 (続き) F\$GETSYI 項目

項目	データ・ タイプ	戻される情報
DECNET_VERSION	文字列	<p>ローカル・システムにインストールされている DECnet パッケージの特定のバージョンおよび ECO レベルに関する情報。この項目は、次の形式の 16 進数を含んだ文字列を戻す。</p> <ul style="list-style-type: none"> • バイト 0 =顧客 ECO • バイト 1 = DECnet ECO • バイト 2 = DECnet フェーズ (Phase IV では 4, DECnet-Plus for OpenVMS では 5) • バイト 3 =予約済み <p>Phase IV と DECnet-Plus for OpenVMS を区別するには、DECnet バージョンを含んでいるバイト (バイト 2) を使用する。</p> <p>バイト 0 とバイト 1 の詳細については、最新版の『HP DECnet-Plus for OpenVMS Release Notes』を参照。</p>
D_FLOAT_EMULATED	文字列	D 浮動小数点数の命令が CPU でエミュレートされている場合は TRUE, そうでない場合は FALSE を戻す。
ERLBUFFERPAGES	整数値	各 S0 エラー・ログ・バッファ用に使用されるシステム・ページ数 (VAX の場合) またはページレット数 (Alpha および I64 の場合)
‡ERLBUFFERPAG_S2	整数値	各 S2 エラー・ログ・バッファ用に使用されるシステム・ページ数 (Alpha および I64 の場合)
‡ERRORLOGBUFF_S2	整数値	S2 エラー・ログ・バッファの数
ERRORLOGBUFFERS	整数値	S0 エラー・ログ・バッファの数
F_FLOAT_EMULATED	文字列	F 浮動小数点数の命令が CPU でエミュレートされている場合は TRUE, そうでない場合は FALSE を戻す。
FREE_GBLPAGES	整数値	現在の未使用グローバル・ページ数。
FREE_GBLSECTS	整数値	現在の未使用グローバル・セクション・テーブル・エントリ数。
FREE_PAGES	整数値	未使用ページの総数。
G_FLOAT_EMULATED	文字列	G 浮動小数点数の命令が CPU でエミュレートされている場合は TRUE, そうでない場合は FALSE を戻す。
‡GALAXY_ID	整数値	128 ビットの Galaxy ID。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡GALAXY_MEMBER	整数値	Galaxy 共有コミュニティのメンバの場合は 1 を、そうでない場合は 0。AlphaServer GS シリーズ・システム上でのみサポートされます。

‡Alpha および I64 のみ

(次ページに続く)

表 DCLI-10 (続き) F\$GETSYI 項目

項目	データ・ タイプ	戻される情報
‡GALAXY_PLATFORM	整数値	Galaxy プラットフォーム上で実行している場合は 1 を、そうでない場合は 0 を戻す。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡GALAXY_SHMEMSIZE	整数値	共有メモリ・ページ数を戻す。現在のインスタンスが Galaxy のメンバでない場合、共有メモリは報告されない。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡GH_RSRVPGCNT	整数値	Alpha において、システム・スタートアップ完了後、INSTALL ユーティリティで使用するために予約する粒度ヒントがカバーするページ数。
‡GLX_FORMATION	文字列	このインスタンスが属する Galaxy 構成の作成日時を示すタイムスタンプ文字列。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡GLX_MAX_MEMBERS	整数値	現在の Galaxy 構成に参加できる最大インスタンス数。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡GLX_MBR_MEMBER	整数値	64 バイトの整数。各 8 バイトは Galaxy メンバの番号を表し、7 から 0 の並びになります。値は、インスタンスがメンバの場合は 1、メンバでない場合は 0 となります。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡GLX_MBR_NAME	文字列	Galaxy メンバシップの中の既知のメンバを示す文字列。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡GLX_TERMINATION	文字列	このインスタンスが最後に属していた Galaxy 構成の終了日時を示すタイムスタンプ文字列。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡HP_ACTIVE_CPU_CNT	整数値	ファームウェア・コンソール・モードになっていないハード・パーティション内の CPU 数。OpenVMS の場合、これは CPU がハード・パーティション内のインスタンスの 1 つのアクティブ・セットに参加しているか、参加の途中であることを意味する。パーティショニングをサポートする AlphaServer システム上でのみサポートされます。
‡HP_ACTIVE_SP_CNT	整数値	ハード・パーティション内で実行中のアクティブなオペレーティング・システム・インスタンス数。パーティショニングをサポートする AlphaServer システム上でのみサポートされます。
‡HP_CONFIG_SBB_CNT	整数値	現在のハード・パーティション内の既存のシステム・ビルディング・ブロック数。パーティショニングをサポートする AlphaServer システム上でのみサポートされます。
‡Alpha および I64 のみ		

(次ページに続く)

表 DCLI-10 (続き) F\$GETSYI 項目

項目	データ・ タイプ	戻される情報
‡HP_CONFIG_SP_CNT	整数値	現在のハード・パーティション内のソフト・パーティションの最大数。この数は、特定のソフト・パーティション内で実行中のオペレーティング・システム・インスタンスを表すわけではない。パーティショニングをサポートする AlphaServer システム上でのみサポートされます。
HW_MODEL	整数値	Alpha または VAX のモデル・タイプを示す整数値。整数値が 1023 より大きい場合は Alpha オペレーティング・システム、小さい場合は VAX オペレーティング・システムを表す。
HW_NAME	文字列	Alpha または VAX のモデル名。
‡ITB_ENTRIES	整数値	Alpha において、I ストリーム変換バッファ・エントリ数。これは、常駐コードに割り当てられる粒度ヒントをサポートする。
‡MAX_CPUS	整数値	このインスタンスによって認識される CPU の最大数。
MEMSIZE	整数値	システム構成でのメモリのページ数。
MODIFIED_PAGES	整数値	変更ページの総数。
MULTITHREAD	整数値	MULTITHREAD システム・パラメータの値。
NODENAME	文字列	ノード名 (ダブルコロンは含まない)。
NODE_AREA	整数値	ターゲット・ノードの DECnet 領域。
NODE_CSID	文字列	指定したノードの CSID を、16 進数を含む文字列で戻す。CSID はシステム識別の形式。
NODE_EVOTES	整数値	ノードに割り当てた投票数。
NODE_HWVERS	文字列	指定したノードのハードウェア・バージョン。
NODE_NUMBER	整数値	指定したノードの DECnet 番号。
NODE_QUORUM	整数値	ノードが持つクォラム。
NODE_SWINCARN	文字列	指定したノードのソフトウェア・インカーネーション番号。16 進数を含む文字列で戻す。
NODE_SWTYPE	文字列	指定したノードが使用するオペレーティング・システム・ソフトウェアのタイプ。
NODE_SWVERS	文字列	指定したノードのソフトウェア・バージョン。
NODE_SYSTEMID	文字列	指定したノードのシステム識別番号。16 進数を含む文字列で戻される。
NODE_VOTES	整数値	ノードに割り当てた投票数。
‡NPAGED_FREE	整数値	非ページング・プールの未使用バイト数
‡NPAGED_INUSE	整数値	非ページング・プール内で現在使用中のメモリの合計 (バイト数)
‡NPAGED_LARGEST	整数値	非ページング・プール内の、連続した未使用メモリの最大サイズ

‡Alpha および I64 のみ

(次ページに続く)

表 DCLI-10 (続き) F\$GETSYI 項目

項目	データ・ タイプ	戻される情報
‡NPAGED_TOTAL	整数値	非ページング・プールの合計サイズ (単位はバイト)
‡PAGED_FREE	整数値	ページング・プールの未使用バイト数
‡PAGED_INUSE	整数値	ページング・プールで内で現在使用中のメモリの合計 (バイト数)
‡PAGED_LARGEST	整数値	ページング・プール内の、連続した未使用メモリの最大サイズ
‡PAGED_TOTAL	整数値	ページング・プールの合計サイズ (単位はバイト)
PAGEFILE_FREE	整数値	現在インストールされているページング・ファイルの未使用ページ数。
PAGEFILE_PAGE	整数値	現在インストールされているページング・ファイルのページ数。
PAGE_SIZE	整数値	1 つの物理ページに含まれるバイト数。
‡PALCODE_VERSION	文字列	Alpha システムでの PALCODE (特権アーキテクチャ・ライブラリ) のバージョン。
‡PARTITION_ID	整数値	ソフト・パーティション ID。パーティショニングをサポートする AlphaServer システム上でのみサポートされます。
‡POTENTIAL_CPU_MASK	整数値	CPU インデックス付きのビットベクトルを表す値。特定のビット位置がセットされているとき、この CPU ID 値を持つプロセッサは、そのインスタンスの潜在的なセットのメンバとなる。潜在的なセット内の CPU は、このインスタンスによって所有された場合、このインスタンスの OpenVMS アクティブ・セットに積極的に参加する。このルールに従うには、CPU の特性が、そのインスタンス用に個別に定義されたハードウェアおよびソフトウェア互換性ルールに一致しなければならない。
‡POTENTIALCPU_CNT	整数値	このインスタンスの潜在的なセットのメンバであるハード・パーティション内の CPU 数。潜在的なセット内の CPU は、このインスタンスによって所有された場合、このインスタンスの OpenVMS アクティブ・セットに積極的に参加する。このルールに従うには、CPU の特性が、そのインスタンス用に個別に定義されたハードウェアおよびソフトウェア互換性ルールに一致しなければならない。
‡POWERED_CPU_MASK	整数値	CPU インデックス付きのビットベクトルを表す値。特定のビット位置がセットされているとき、この CPU ID 値を持つプロセッサは、そのインスタンスのパワード・セットのメンバとなる。これらの CPU は、ハード・パーティション内に物理的に存在し、操作のために起動される。

‡Alpha および I64 のみ

(次ページに続く)

表 DCLI-10 (続き) F\$GETSYI 項目

項目	データ・ タイプ	戻される情報
‡POWEREDCPU_CNT	整数値	物理的に起動しているハード・パーティション内の CPU 数。
‡PRESENT_CPU_MASK	整数値	CPU インデックス付きのビットベクトルを表す値。特定のビット位置がセットされているとき、この CPU ID 値を持つプロセッサは、そのインスタンスのプレゼント・セットのメンバとなる。これらの CPU は、ハード・パーティション内に物理的に存在する。プレゼント・セットのメンバであっても、パワード・セットのメンバとは限らない。
‡PRESENTCPU_CNT	整数値	ハードウェア・スロットに物理的に常駐するハード・パーティション内の CPU 数。
‡PRIMARY_CPUID	整数値	この OpenVMS インスタンスのプライマリ・プロセッサの CPU ID。
QUANTUM	整数値	他のプロセスが待ち状態の間、プロセスが受信ができる最大プロセッサ時間。
‡RAD_CPUS	整数値	RAD と CPU のペアのコンマ区切りのリスト。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡RAD_MAX_RADS	整数値	このプラットフォーム上で使用可能な RADS の数の最大値。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡RAD_MEMSIZE	整数値	RAD と PAGES のペアのコンマ区切りのリスト。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡RAD_SHMEMSIZE	整数値	RAD と PAGES のペアのコンマ区切りのリスト。AlphaServer GS シリーズ・システム上でのみサポートされます。
‡REAL_CPUYPE	整数値	ハードウェア・リスタート・パラメータ・ブロック (HWRPB) から抽出されたシステムの主 CPU の実際の CPU タイプ。
SCS_EXISTS	文字列	現在 OpenVMS ノードにシステム通信サブシステム (SCS) がロードされている場合は TRUE、そうでない場合は FALSE を戻す。
‡SCSNODE	文字列	Galaxy インスタンス名。パーティショニングをサポートする AlphaServer システム上でのみサポートされます。
SID	整数値	システム識別番号。Alpha において、常に値が 256 である CPU 型フィールドを除いた、すべてのフィールドがゼロとなる値を戻す。
SWAPFILE_FREE	整数値	現在インストールされているスワッピング・ファイルの未使用のページ数。
SWAPFILE_PAGE	整数値	現在インストールされているスワッピング・ファイルのページ数。

‡Alpha および I64 のみ

(次ページに続く)

表 DCLI-10 (続き) F\$GETSYI 項目

項目	データ・ タイプ	戻される情報
SYSTEM_RIGHTS	文字列	ローカル・システムの権利リストの目次。リモート・システムを指定している場合は、空文字列("")が戻される。この項目は、コンマ(,)で区切られた識別子名のリストを戻す。
§SYSTEM_UUID	整数値	システムの 128 ビットの UUID (Universal Unique Identifier)
‡SYSTYPE	整数値	Alpha において、ファミリまたはシステム・ハードウェア・プラットフォーム。たとえば、2 は DEC 4000、3 は DEC 7000 または DEC 10000、4 は DEC 3000 を表す。
TOTAL_PAGES	整数値	物理メモリメモリ・ページの総数。
USED_GBLPAGCNT	整数値	グローバル・ページ・テーブルで現在使用しているページ数。
USED_GBLPAGMAX	整数値	グローバル・ページ・テーブルでこれまでの最大使用ページ数。
USED_PAGES	整数値	使用ページの総数。
VECTOR_EMULATOR	論理値	VAX ベクタ命令エミュレータ機能 (VVIEF) の存在を示すフラグ。
VERSION	文字列	使用している OpenVMS のバージョン (8 文字で残りを空白で埋める)。
VP_MASK	整数値	ベクタ共同プロセッサがあるプロセッサを示すマスク。
VP_NUMBER	整数値	システムのベクタ・プロセッサ数。
§I64 のみ		
‡Alpha および I64 のみ		

例

- ```

$ SYSID = F$GETSYI("SID")
$ SHOW SYMBOL SYSID
SYSID = 19923201 Hex = 01300101 Octal = 000401

```

この例では、F\$GETSYI 関数を使用して、システム識別レジスタの情報を戻す方法を示しています。SID 引数は文字列リテラルなので、引用符(" ")で囲まれています。F\$GETSYI 関数から戻される値は、SYSID シンボルに割り当てられません。ノードは指定されていないため、現在のノードに関する情報が戻されます。

- ```

$ MEM = F$GETSYI("CLUSTER_MEMBER", "LONDON")
$ SHOW SYMBOL MEM
MEM = "TRUE"

```

この例では、F\$GETSYI 関数を使用して、LONDON ノードがローカル・クラスタのメンバであるかどうかを調べています。戻り値 TRUE は、リモート・ノード LONDON がクラスタのメンバであることを示します。

3.

```
$ LIM = F$GETSYI("IJOBLIM")
$ SHOW SYMBOL LIM
LIM = 16   Hex = 00000010   Octal = 00000000020
```

この例では、SYSGEN パラメータである IJOBLIM が、F\$GETSYI 関数に対する引数として使用されています。この引数は、現在のシステムのバッチ・ジョブ制限値を戻します。

4.

```
$ DECNETVERS = F$GETSYI("DECNET_VERSION")
$ SHOW SYMBOL DECNETVERS
DECNETVERS = "00050D01"
$ DECNETPHASE = F$INTEGER(F$EXTRACT(2,2,DECNETVERS))
$ SHOW SYMBOL DECNETPHASE
DECNETPHASE = 5   Hex = 00000005   Octal = 00000000005
```

この例では、F\$GETSYI の DECNET_VERSION 項目を使用して DECnet バージョンを戻す方法を示しています。

5.

```
$ RADCPU = F$GETSYI("RAD_CPUS")
$ SHOW SYMBOL RADCPU
0,0,0,1,1,4,1,5
```

この例では、システム・パラメータである RAD_CPUS が、F\$GETSYI 関数に対する引数として使用されています。この引数は、RAD と CPU のペアのコンマ区切りのリストを戻します。この例では、第 1 の RAD と CPU のペアは 0, 0、第 2 のペアは 0,1 などとなっています。

AlphaServer GS シリーズ・システム上でのみサポートされます。

F\$IDENTIFIER

英数字の識別子を、それに相当する整数に変換します。または、その逆の操作を実行します。識別子は、ユーザのカテゴリを識別するための名前または番号です。システムは、この識別子を使用して、リソースに対する利用者のアクセス権を判断します。

フォーマット

F\$IDENTIFIER(識別子, 変換タイプ)

戻り値

F\$IDENTIFIER 関数は、識別子を名前から整数に変換する場合には整数値を、整数から名前に変換する場合には文字列を戻します。誤った識別子を指定すると、F\$IDENTIFIER 関数は空文字列("") (整数から名前に変換する場合)、または 0 (名前から整数に変換する場合) を戻します。

引数

識別子

変換される識別子を指定します。整数から名前に変換する場合は、整数式として指定します。名前を整数に変換する場合は、文字列式として指定します。

Name Hidden 属性を持つ識別子の場合、質問に識別子がない場合、または権利データベースへの読み込みアクセス権がない場合、F\$IDENTIFIER はエラーになります。属性についての詳細は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

変換タイプ

実行される変換のタイプを指定します。識別子引数が英数字の場合は、“NAME_TO_NUMBER”を含む文字列として変換タイプを指定します。識別子引数が数値の場合は、“NUMBER_TO_NAME”を含む文字列として変換タイプを指定します。

例

```
1. $ UIC_INT= F$IDENTIFIER("SLOANE", "NAME_TO_NUMBER")
$ SHOW SYMBOL UIC_INT
   UIC_INT = 15728665   Hex = 00F00019   Octal = 00074000031
$ UIC = F$FAO("!%U", UIC_INT)
$ SHOW SYMBOL UIC
   UIC = [360,031]
```

この例では、F\$IDENTIFIER 関数を使用して、MANAGERS、SLOANE] という UIC のメンバ識別子を整数に変換しています。F\$IDENTIFIER 関数は、メンバ識別子 SLOANE が、15728665 という整数に等しいことを示しています。SLOANE という識別子を指定する場合には、大文字を使用しなければなりません。

この 8 進数を標準的な数値形式の UIC に変換するには、F\$FAO 関数と !%U ディレクティブを使用します(このディレクティブは、ロングワードを名前形式の UIC に変換します)。この例では、SLOANE というメンバ識別子は、数値形式の UIC [360 , 031] と同値です。

```
2. $ UIC_INT = (%031 + (%X10000 * %0360))
$ UIC_NAME = F$IDENTIFIER(UIC_INT, "NUMBER_TO_NAME")
$ SHOW SYMBOL UIC_NAME
   UIC_NAME = "ODONNELL"
```

この例は、数値形式の UIC [360 , 031] に対応する英数字の識別子を求めています。まず、UIC [360 , 031] に対応するロングワード整数を求めなければなりません。このためにメンバ番号を下位ワードに、グループ番号を上位ワードに挿入します。次に、F\$IDENTIFIER 関数を使用して、その整数に対応する名前形式の識別子を求めます。

F\$INTEGER

指定した式の結果と等価な整数値を戻します。

フォーマット

F\$INTEGER(式)

戻り値

指定した式と等価な整数値。

引数

式

評価される式を指定します。整数または文字列式を指定します。

整数式を指定すると、F\$INTEGER 関数は式を評価し結果を戻します。文字列式を指定すると、F\$INTEGER 関数は式を評価し、その結果の文字列を整数に変換して戻します。

文字列式を評価した後、F\$INTEGER 関数は次の方法で整数に変換します。評価した結果の文字列に、有効な整数を形成する文字が含まれている場合は、F\$INTEGER 関数は整数値を戻します。評価した結果の文字列に、有効な整数を形成する文字が含まれていない場合は、文字列が T、t、Y、または y で始まっていると F\$INTEGER 関数は 1 を戻します。文字列は他の文字で始まる場合は、F\$INTEGER は 0 を戻します。

例

- ```
$ A = "23"
$ B = F$INTEGER("-9" + A)
$ SHOW SYMBOL B
B = -923 Hex=FFFFFC65 Octal=176145
```

この例では、F\$INTEGER 関数を使用して、関数により戻された整数値をシンボルに割り当てる方法を示しています。この例では、F\$INTEGER 関数は、文字列式 (“-9” + A) と等価な整数を戻します。まず F\$INTEGER 関数は、文字列リテラル “-9” と “23” を連結させて、文字列式を評価します。シンボル A の値は、自動的

に文字列式に置換される点に注意してください。どちらの引数も文字列リテラルなので、プラス記号 (+) は文字列連結演算子である点にも注意してください。

文字列式を評価した後、F\$INTEGER 関数は、結果の文字列式 (“-923”) を整数に変換し、値 -923 を戻します。この整数値は、シンボル B に割り当てられます。

---

## F\$LENGTH

指定した文字列の長さを戻します。

---

## フォーマット

F\$LENGTH(文字列)

---

## 戻り値

文字列の長さを示す整数値

---

## 引数

文字列

長さを求める文字列を指定します。文字列式として指定します。

---

## 例

```
1. $ MESSAGE = F$MESSAGE(%X1C)
 $ SHOW SYMBOL MESSAGE
 MESSAGE = "%SYSTEM-F-EXQUOTA, exceeded quota"
 $ STRING_LENGTH = F$LENGTH(MESSAGE)
 $ SHOW SYMBOL STRING_LENGTH
 STRING_LENGTH = 33 Hex = 00000021 Octal = 000041
```

この例では、最初の割り当てステートメントでは、16 進数の 1C に対応するメッセージを戻すために、F\$MESSAGE 関数を使用しています。メッセージは、文字列式として戻され、MESSAGE というシンボルに割り当てられます。

次に、F\$LENGTH 関数を使用して、シンボル MESSAGE に割り当てられた文字列の長さが戻されています。F\$LENGTH 関数に対する引数として、シンボル MESSAGE を使用する場合、引用符 (" ") で囲む必要はありません (シンボルを囲む引用符は、文字列式では使用されません)。

F\$LENGTH 関数は、文字列の長さを戻し、その値をシンボル STRING\_LENGTH に割り当てます。この例の最後で、シンボル STRING\_LENGTH の値は、シンボル MESSAGE の値の文字数、つまり 33 になります。



---

## F\$LICENSE (Alpha/I64 のみ)

指定されたライセンスがシステムにロードされているかどうかを確認します。

---

## フォーマット

F\$LICENSE(ライセンス名)

---

## 戻り値

文字列 TRUE または FALSE。

---

## 引数

ライセンス名

状態を確認したいライセンス名を指定します。

---

## 例

```
1. $ SHOW LICENSE VMSCLUSTER*
 Active licenses on node NODE1:

----- Product ID ----- ---- Rating ----- -- Version --
Product Producer Units Avail Activ Version Release Termination
VMSCLUSTER DEC 0 0 100 0.0 (none) 14-MAY-2005
VMSCLUSTER-CLIENT DEC 0 0 100 0.0 (none) 14-MAY-2005

$ WRITE SYS$OUTPUT F$LICENSE("VMSCLUSTER")
TRUE
$ WRITE SYS$OUTPUT F$LICENSE("NONEXISTENT_PAK")
FALSE
```

この例では、F\$LICENSE 関数は TRUE を返しており、VMSCLUSTER ライセンスがシステムにロードされていることを示しています。逆に、架空のライセンス NONEXISTENT\_PAK は FALSE となっており、システムにロードされていないことを示しています。

---

## F\$LOCATE

文字列に含まれる特定の部分を探し、最初の文字のオフセットを整数値で戻します。オフセットとは、文字列の先頭を基準にした文字（1文字または部分文字列）の位置です。オフセットとは、文字列の開始位置を基準にして、文字、または部分文字列までの位置を示します。文字列の1文字目は、常に文字列の先頭からオフセット0の位置にあります。

部分文字列が見つからないと、F\$LOCATE関数は、検索文字列の長さ（文字列の最後の文字のオフセットに1を加算した値）を戻します。

---

## フォーマット

F\$LOCATE(部分文字列, 文字列)

---

## 戻り値

部分文字列引数のオフセットを示す整数値。オフセットとは、文字列の先頭を基準にした文字（1文字または部分文字列）の位置です。文字列の1文字目は、常に文字列の先頭からオフセット0の位置にあります（文字列の先頭は、常に左端の文字です）。部分文字列引数がない場合には、F\$LOCATE関数は、文字列の最後の文字のオフセットより、1だけ大きなオフセットを戻します（これは文字列の長さに相当します）。

---

## 引数

部分文字列

文字列引数に指定した文字列から検索したい文字列を指定します。

文字列

F\$LOCATE が編集する文字列を指定します。

---

## 例

1. \$ FILE\_SPEC = "MYFILE.DAT;1"  
\$ NAME\_LENGTH = F\$LOCATE(".", FILE\_SPEC)

この例では、F\$LOCATE関数は、文字列の先頭を基準にした、文字列中のピリオド(.)の位置を戻します。ピリオドはオフセット6の位置に存在するため、6という値がシンボルNAME\_LENGTHに割り当てられます。また、NAME\_

LENGTH は、MYFILE.DAT というファイル指定のファイル名の部分の長さ、つまり 6 に等しくなります。

部分文字列引数であるピリオドは、文字列リテラルとして指定されているので、引用符 ( " ) で囲まれています。しかし、FILE\_SPEC 引数はシンボルであるため、引用符で囲む必要はありません。このシンボルは、関数进行处理しているときに、自動的に現在の値に置き換えられます。

```
2. $ INQUIRE TIME "Enter time"
 $ IF F$LOCATE(":",TIME) .EQ. F$LENGTH(TIME) THEN -
 GOTO NO_COLON
```

これは、コマンド・プロシージャの一部です。この中では、F\$LOCATE 関数の結果と F\$LENGTH 関数の結果を比較し、結果が等しいかどうかを調べています。1 文字または部分文字列が、ある文字列に含まれているかどうかを判断するときに、この方法をよく使用します。

この例では、INQUIRE コマンドが、時刻の値の入力を要求するプロンプトを表示し、利用者が入力した時刻をシンボル TIME に割り当てます。IF コマンドは、プロンプトに対する応答として入力された文字列に、コロンが含まれているかどうかを調べます。F\$LOCATE 関数から戻された値が、F\$LENGTH 関数から戻された値と等しい場合には、コロンは文字列の中に含まれていません。F\$LOCATE 関数と F\$LENGTH 関数はどちらも整数値を戻すため、(.EQS. 演算子ではなく) .EQ. 演算子が使用されています。

部分文字列引数であるコロンは文字列リテラルであるため、引用符で囲まれています。しかし、シンボル TIME は、引用符で囲む必要はありません。これは、文字列式として自動的に評価されるためです。

---

## F\$MESSAGE

特定のシステム状態コードに対応するファシリティ、重要度、ID、およびテキストを、文字列として戻します。

---

### フォーマット

F\$MESSAGE(状態コード[, メッセージ構成要素リスト])

---

### 戻り値

指定した引数に対応するシステム・メッセージを含む文字列。  
各システム・メッセージ・ファイルは、数値または値の範囲に対応づけられています。ただし、必ずしもすべての数値に対応するメッセージがあるわけではありません。対応するメッセージのない数値を引数として指定すると、F\$MESSAGE 関数は、NOMSG エラー・メッセージを含む文字列を戻します。

システム・エラー・メッセージについての詳細は、『OpenVMS System Messages: Companion Guide for Help Message Users』を参照してください。

---

### 引数

#### 状態コード

エラー・メッセージ文を表示したい状態コードを指定します。状態コードは、整数式として指定します。

#### メッセージ構成要素リスト

システム・メッセージの構成要素を戻すことを指定します。パラメータが空または指定されない場合は、すべてのシステム・メッセージの構成要素が戻されます。

システム・メッセージの構成要素に有効なキーワードを表 DCLI-11 に示します。

表 DCLI-11 F\$MESSAGE キーワード

| 構成要素キーワード | 戻される情報 |
|-----------|--------|
| FACILITY  | 機能名    |
| SEVERITY  | 重大度表示  |

(次ページに続く)

表 DCLI-11 (続き) F\$MESSAGE キーワード

| 構成要素キーワード | 戻される情報        |
|-----------|---------------|
| IDENT     | メッセージ・テキスト短縮形 |
| TEXT      | メッセージの説明      |

FACILITY, SEVERITY, および IDENT を (個々に, または任意に組み合わせて) 指定すると, 戻されたメッセージにはパーセント記号(%)がつくことに注意してください。複数のキーワードを指定すると, メッセージはハイフンで区切られます。

TEXT だけを指定した場合, メッセージに文字記号は含まれません。TEXT を FACILITY, SEVERITY, または IDENT と同時に指定した場合は, メッセージがコンマと空白(,)で区切られます。

---

## 例

```
1. $ ERROR_TEXT = F$MESSAGE(%X1C)
 $ SHOW SYMBOL ERROR_TEXT
 ERROR_TEXT = "%SYSTEM-F-EXQUOTA, exceeded quota"
```

この例では, F\$MESSAGE 関数を使用して, %X1C という状態コードに対応するメッセージを判断する方法を示しています。F\$MESSAGE 関数は, メッセージ文字列を戻します。その文字列は, ERROR\_TEXT というシンボルに割り当てられます。

```
2. $ SUBMIT IMPORTANT.COM
 $ SYNCHRONIZE /entry='$ENTRY'
 $ IF $STATUS THEN EXIT
 $!
 $ JOB_STATUS = $STATUS
 $!
 $ IF "%JOBDELETE" .EQS. F$MESSAGE (JOB_STATUS, "IDENT")
 $ THEN
 .
 .
 .
 $ ELSE
 $ IF "%JOBABORT" .EQS. F$MESSAGE (JOB_STATUS, "IDENT")
 $ THEN
 .
 .
 .
 $ ELSE
 $.
 $.
 $.
 $ ENDIF
 $ ENDIF
```

・  
・  
・

この例は、バッチ・ジョブをキューに登録し、終了するのを待つコマンド・プロシージャです。ジョブが正常に終了しなかった場合は、バッチ・ジョブの終了状態に基づいてさらに処理が続きます。

最初のコマンドで、コマンド・プロシージャ `IMPORTANT.COM` を登録します。2 番目のコマンドで、`SYNCHRONIZE` を使用してジョブが終了するのを待つように指示します。3 番目のコマンドでジョブの終了を確認し、成功していればプロシージャを終了します。次のコマンドでシンボルに状態を保存します。

最初の IF 文は、`F$MESSAGE` を使用して、実行前にジョブが削除されたかを判断します。削除されている場合は、ジョブの再登録、または `MAIL` を使ってユーザに知らせる、のいずれか可能な方の処理を行います。

次の IF 文は、`F$MESSAGE` を使用して、ジョブが実行中に削除されたかを判断します。その結果、クリーンアップ処理されるか、または `THEN` ブロックでオペレータの介入要求があります。

どちらの IF 文も真ではない場合は、その他の失敗状態が戻されます。ELSE 文以降のブロックで行われる他の処理が要求されます。

---

## F\$MODE

プロセスが実行されているモードを示す文字列を戻します。F\$MODE 関数には引数はありませんが、括弧は指定しなければなりません。

---

## フォーマット

F\$MODE()

---

## 戻り値

会話型プロセスの場合は INTERACTIVE という文字列。プロセスが会話型ではない場合は、BATCH、NETWORK、または OTHER という文字列が戻されます。戻される文字列は、常に大文字です。

---

## 引数

なし

---

## 説明

F\$MODE レキシカル関数は、プロセスがじっこうされているモードを示す文字列を戻します。引き数はとりませんが、括弧は省略できません。

会話型で実行する場合と、非会話型で実行する場合とでは、別々に処理する必要がある時、コマンド・プロシージャで F\$MODE 関数を使用すると便利です。ログイン・コマンド・プロシージャに F\$MODE 関数、または F\$ENVIRONMENT 関数を入れておき、会話型ターミナル・セッションと非会話型セッションではことなるコマンドを実行するようにしておきます。

ログイン・コマンド・プロシージャに F\$MODE 関数を含めず、会話型プロセスから実行されるかどうかを調べていない時に、ログイン・コマンド・プロシージャが非会話型プロセス（たとえばバッチ・ジョブ）から実行されると、コマンド・プロシージャ内に会話型プロセスでのみ使用できるコマンドが含まれている場合は、プロセスが異常終了する可能性があります。

コマンド・プロシージャは F\$MODE 関数を使用して、会話型ターミナル・セッションでプロシージャが実行されるかどうか調べることができます。また F\$MODE 関数の戻り値に基づいて、制御を移すことができます。

## 例

```
1. $ IF F$MODE() .NES. "INTERACTIVE" THEN GOTO NON_INT_DEF
 $ INTDEF: ! Commands for interactive terminal sessions
 .
 .
 .
 $ EXIT
 $ NON_INT_DEF: !Commands for noninteractive processes
 .
 .
 .
```

この例では、ログイン・コマンド・ファイルの初めの部分が表示されています。このコマンド・プロシージャは、2つの初期化コマンドがあります。1つは会話型モードの場合のコマンドで、もう1つは(バッチ・ジョブやネットワーク・ジョブを含む)非会話型モードの場合のコマンドです。IF コマンドは、F\$MODE から戻された文字列が、INTERACTIVE という文字列と、等しいかどうかを調べます。等しくない場合には、制御はNON\_INT\_DEF というラベルに分岐します。等しい場合には、INTDEF というラベルのあとのステートメントが実行され、プロシージャはNON\_INT\_DEF の前にあるステートメントで終了します。



---

## F\$MULTIPATH (Alpha/I64 のみ)

特定のマルチパス対応装置の、指定された項目の情報を戻します。

---

### フォーマット

F\$MULTIPATH(装置名, 項目, コンテキスト・シンボル)

---

### 戻り値

要求された情報を含む文字列。

---

### 引数

#### 装置名

物理装置名または物理装置名と同等の論理名を指定します。装置名は文字列式で指定します。

装置名引数を評価した後、F\$MULTIPATH 関数は 名前の最初の文字を調べます。最初の文字がアンダースコア ( \_ ) の場合は、名前は物理装置名とみなされます。そうでない場合は、1 レベルの論理名変換が実行され、等価名があればそれが使用されます。

#### 項目

戻すべき装置情報の種類を指定します。この引数は文字列式として指定します。現時点で唯一有効な項目は MP\_PATHNAME です。これは、指定されたマルチパス対応装置のパス名を示す文字列を戻します。

#### コンテキスト・シンボル

最初に MP\_PATHNAME を指定して F\$MULTIPATH を使用する前に、コンテキスト・シンボルを 0 に初期化しておく必要があります。F\$MULTIPATH 関数は、コンテキスト・シンボルの値を維持する責任があります。

---

#### 警告

---

コンテキスト・シンボルを 0 に初期化した後で値を変更しないでください。  
値を変更すると、F\$MULTIPATH が予測できない動作をします。

---

---

## 説明

システム・サービス\$DEVICE\_PATH\_SCAN を呼び出して、特定のマルチパス対応装置の、指定された項目の情報を戻します。

またレキシカル関数 F\$MULTIPATH は、システム・サービス\$DEVICE\_PATH\_SCAN が生成したエラー・メッセージも戻します。

システム・サービス\$DEVICE\_PATH\_SCAN についての詳細は、『OpenVMS System Services Reference Manual』を参照してください。

---

## 例

```
1. $ XYZ = 0
 $
 $LOOP:
 $ PATH = F$MULTIPATH("$1$DGA12", "MP_PATHNAME", XYZ)
 $ IF PATH .EQS. "" THEN GOTO EXIT
 $ WRITE SYS$OUTPUT "PATH NAME = 'PATH'"
 $ GOTO LOOP
 $
 $EXIT:
 $ EXIT
```

この例は、項目コードとして MP\_PATHNAME を指定した F\$MULTIPATH の使用方法を示します。コンテキスト・シンボル XYZ がループの外で 0 に初期化されている点に注意してください。このコマンド・プロシージャの出力を下に示します。指定されたマルチパス装置のすべてのパスが戻された後、空白のパス名が戻されてリストの終わりが示されます。

```
path name = PGA0.5000-1FE1-0001-5782
path name = PGA0.5000-1FE1-0001-5783
path name = PGA0.5000-1FE1-0001-5781
path name = PGA0.5000-1FE1-0001-5784
path name = MSCP
```

---

## F\$PARSE

ファイル指定を解析し、拡張ファイル指定、またはユーザが指定した特定のファイル指定フィールドを戻します。

---

## フォーマット

F\$PARSE(ファイル指定[, 省略時のファイル指定][, 関連ファイル指定][, フィールド]  
[, 解析タイプ])

---

## 戻り値

他のフィールドも加えて拡張されたファイル指定、または指定したフィールドを含む文字列。完全ファイル指定を行わなかった場合は、F\$PARSE 関数は省略時の文字列を戻します。詳細は説明のセクションを参照してください。

解析中にエラーを検出した場合、F\$PARSE 関数は空文字列("")を返します。たとえば、ファイル指定が間違った構文である場合や、指定したディスクやディレクトリが存在しない(結果としてファイル指定が論理的に間違っている)場合は、空文字列が返されます。ただし、フィールド名を指定する、または解析タイプに SYNTAX\_ONLY を指定すると、適切な情報が返されます。

---

## 引数

### ファイル指定

解析するファイル指定を含む文字列を指定します。

ファイル指定にはワイルドカード文字を使用できます。この場合、F\$PARSE により戻されるファイル指定もワイルドカード文字を含んでいます。

### 省略時のファイル指定

省略時のファイル指定を含む文字列を指定します。

ファイル指定引数で省略されているフィールドがあると、省略時のファイル指定の対応するフィールドで置換されます。この結果、まだ省略されているフィールドは、関連ファイル指定引数の対応するフィールドで置換されます。

### 関連ファイル指定

関連するファイル指定を含む文字列を指定します。

ファイル指定引数と省略時のファイル指定引数の両方で、省略されているフィールドがあると、関連ファイル指定の対応するフィールドで置換されます。

#### フィールド

ファイル指定のフィールド名を含む文字列を指定します。この引数を指定すると、F\$PARSE 関数は、ファイル指定の特定の部分を戻します。

次のフィールド名のいずれか 1 つを指定できます (省略することはできません)。

|           |              |
|-----------|--------------|
| NODE      | ノード名         |
| DEVICE    | 装置名          |
| DIRECTORY | ディレクトリ名      |
| NAME      | ファイル名        |
| TYPE      | ファイル・タイプ     |
| VERSION   | ファイル・バージョン番号 |

#### 解析タイプ

実行される解析タイプを指定します。省略時の設定では、F\$PARSE 関数は、ファイル指定で指定したディレクトリが、ファイル指定で指定した装置上に存在するかどうか確認します。ただし、フィールド引数を指定した場合は、ディレクトリの存在は確認しません。装置とディレクトリは、引数の 1 つに明示的に指定できます。また、省略しても省略時の値として使用されます。

また、省略時の設定で F\$PARSE 関数は、他の引数として指定されている論理名があると、その論理名を変換します。CONCEALED 属性を持つ論理名を検出すると、F\$PARSE 関数は、反復変換を停止します。

次のキーワードを使用すると、F\$PARSE 関数がファイル指定を解析する方法を変更できます。

|             |                                                                       |
|-------------|-----------------------------------------------------------------------|
| NO_CONCEAL  | ファイル指定の一部に指定された論理名変換時に、“conceal”属性を無視する。つまり、隠し論理名が検出されても、論理名変換は終了しない。 |
| SYNTAX_ONLY | ファイル指定の構文だけをチェックし、指定したディレクトリが装置上に確認するかどうかは確認しない。                      |

---

## 説明

F\$PARSE 関数は、RMS サービスの\$PARSE を使用して、ファイル指定を解析します。\$PARSE についての詳細は、『OpenVMS Record Management Services Reference Manual』を参照してください。

F\$PARSE 関数を使用する場合、最後に指定した引数の後ろに指定するオプションの引数を省略することができます。ただし、最後に指定する引数の前 (左) に指定するオプションの引数を省略する場合は、プレースホルダーとしてコンマ(,)を入れます。

ファイル指定引数で装置名およびディレクトリ名を省略すると、F\$PARSE 関数は、まず省略時のファイル指定引数、次に関連ファイル指定引数から、省略時設定の名前を補います。どちらの引数からも名前を得られない場合は、現在の省略時の設定を使用します。

ノード名、ファイル名、ファイル・タイプ、またはバージョン番号を省略すると、F\$PARSE 関数は、まず省略時のファイル指定引数、次に関連ファイル指定引数から省略時設定を補います(ただし、関連ファイル指定引数からは、バージョン番号を補えない点に注意してください)。どちらの引数からも名前を与えられない場合は、F\$PARSE はこれらのフィールドに空指定を戻します。

## 例

```
1. $ SET DEF DISK2:[FIRST]
 $ SPEC = F$PARSE("JAMES.MAR","[ROOT]",,"SYNTAX_ONLY")
 $ SHOW SYMBOL SPEC
 SPEC = "DISK2:[ROOT]JAMES.MAR;"
```

この例では、F\$PARSE 関数は、JAMES.MAR というファイルの拡張ファイル指定を戻します。この例では、SYNTAX\_ONLY キーワードを指定し、F\$PARSE 関数は構文だけを調べ、[ROOT]ディレクトリが DISK2 に存在するかどうかは調べないよう指定しています。

省略時の装置とディレクトリは、DISK2:[FIRST]です。割り当て文の省略時のファイル指定引数に[ROOT]ディレクトリが指定されているので、[ROOT]ディレクトリは出力文字列のディレクトリ名として使用されます。出力文字列に戻る省略時の装置は DISK2 であり、このファイルの省略時のバージョン番号は空文字です。JAMES.MAR と ROOT という引数は文字列リテラルなので、引用符(“”)で囲まなければなりません。

構文だけの解析をするように指定せず、しかも [ ROOT ] が DISK2 に存在しない場合には、空文字列が戻されます。

```
2. $ SET DEFAULT DB1:[VARGO]
 $ SPEC = F$PARSE("INFO.COM",,"DIRECTORY")
 $ SHOW SYMBOL SPEC
 SPEC = "[VARGO]"
```

この例では、F\$PARSE 関数は、INFO.COM というファイルのディレクトリ名を戻します。引数リストで省略時のファイル指定引数と関連ファイル指定引数が省略されているので、それぞれの位置にコンマ(,)を指定しなければなりません。

## レキシカル関数 F\$PARSE

```
3. $ SPEC= F$PARSE("DENVER::DB1:[PROD]RUN.DAT",,, "TYPE")
$ SHOW SYMBOL SPEC
SPEC = ".DAT"
```

この例では、F\$PARSE 関数を使用して、ノード名を含むファイル指定を解析しています。F\$PARSE 関数は、DENVER というリモート・ノードの、RUN.DAT というファイルのファイル・タイプである、DAT を戻します。

---

## F\$PID

プロセス識別番号 (PID) を戻し、システムのプロセス・リストで現在の位置を指すようコンテキスト・シンボルを変更します。

---

## フォーマット

F\$PID(コンテキスト・シンボル)

---

## 戻り値

システムのプロセス・リストの中の、あるプロセスのプロセス識別番号 (PID) を示す文字列。

---

## 引数

### コンテキスト・シンボル

システムのプロセス・リストを示すポインタを格納するために、DCL が使用するシンボルを指定します。F\$PID 関数は、このポインタを使用して PID を戻します。

シンボルを使用して、コンテキスト・シンボルを指定します。コマンド・プロシージャ内で最初に F\$PID 関数を使用する時は、シンボルは未定義のシンボル、空文字列 ("" ) が割り当てられたシンボル、または F\$CONTEXT 関数で設定されたコンテキスト・シンボルを使用します。

コンテキスト・シンボルが未定義または空文字列が割り当てられた場合は、F\$PID 関数は、システムのプロセス・リスト中で、アクセス特権を持つ最初の PID を戻します。つまり、ユーザが GROUP 特権を持ち、コンテキスト・シンボルが未定義または空文字列が割り当てられた場合は、F\$PID 関数は、ユーザのグループで最初のプロセスの PID を戻します。ユーザが WORLD 特権を持っている場合は、F\$PID 関数は、リストで最初のプロセスの PID を戻します。ユーザが GROUP 特権も WORLD 特権もない場合は、F\$PID 関数は、該当ユーザが所有している最初のプロセスの PID を戻します。後続の F\$PID 呼び出しは、ユーザがアクセスしているシステム上の他のプロセスを戻します。

コンテキスト・シンボルが F\$CONTEXT 関数により設定されている場合、F\$PID 関数は、F\$CONTEXT 呼び出しで指定した選択基準に一致する、システムのプロセス・リストで最初のプロセスを戻します。後続の F\$PID 呼び出しは、F\$CONTEXT 関数により設定された選択基準に一致し、ユーザの現在の特権でアクセスできるプロセスの PID だけを戻します。

---

## 説明

F\$PID 関数は、プロセス識別番号 (PID) を戻し、システムのプロセス・リストでの現在の位置を指すようコンテキスト・シンボルを変更します。システム上のすべてのプロセスを使用することもできますし、F\$CONTEXT 関数を使用して選択基準を指定することもできます。F\$CONTEXT 関数は必須ではありません。

F\$PID が戻すプロセス識別番号 (PID) は、ユーザのプロセスが持つ特権に応じて異なります。ユーザのプロセスが GROUP 特権を持っている場合は、グループ内のプロセスの PID が戻されます。ユーザのプロセスが WORLD 特権を持っている場合は、システム上のすべてのプロセスの PID が戻されます。いずれの特権も持たない場合は、該当ユーザが所有しているプロセスの PID だけが戻されます。

F\$CONTEXT 関数は F\$PID 関数が複合アーキテクチャ OpenVMS Cluster システムのどのノードからもプロセスを検索できるようにします。

最初に F\$PID を使用するときには、未定義なシンボル、空文字列が割り当てられたシンボル、または F\$CONTEXT で作成したコンテキスト・シンボルを指定してください。これにより F\$PID 関数は、ユーザがアクセス権を没、システムのプロセス・リストで最初の PID を戻します。また、F\$PID 関数はコンテキスト・シンボル引数を初期化します。

一度コンテキスト・シンボル引数が初期化されると、後続の F\$PID は、F\$CONTEXT で設定した選択基準を使用して順々に PID を戻します。そして、コンテキスト・シンボルがあれば、これを更新します。次に取り出す PID が無くなると、空文字列が戻されます。プロセス・リストの最後の PID を戻した後は、F\$PID 関数は空文字列を戻します。

---

## 例

```
1. $ CONTEXT = ""
 $ START:
 $ PID = F$PID(CONTEXT)
 $ IF PID .EQS. "" THEN EXIT
 $ SHOW SYMBOL PID
 $ GOTO START
```

このコマンド・プロシージャは、F\$PID 関数を使用して、PID のリストを表示しています。割り当てステートメントは、CONTEXT というシンボルを宣言しています。このシンボルは、F\$PID 関数に対するコンテキスト・シンボル引数として使用されます。CONTEXT には空文字列が割り当てられているので、F\$PID 関数は、アクセス権がある、プロセス・リストで最初の PID を戻します。



このコマンド・プロシージャで表示される PID は、プロセスの持つ特権に応じて異なります。GROUP 特権を持つ場合には、グループに含まれるユーザの PID が表示されます。WORLD 特権を持つ場合には、システムのすべてのユーザの PID が表示されます。GROUP 特権も WORLD 特権も持たない場合には、該当ユーザが所有しているプロセスの PID だけが表示されます。

---

## F\$PRIVILEGE

現在のプロセス特権が、引数に指定されている特権と一致するかどうかに応じて、TRUE または FALSE という文字列値を戻します。特権の肯定形、および否定形のいずれも指定できます。

---

## フォーマット

F\$PRIVILEGE(特権)

---

## 戻り値

TRUE または FALSE という値を含む文字列。priv-states 引数にリストされた特権のうち 1 つでも偽があると、F\$PRIVILEGE 関数は文字列 FLASE を戻します。

---

## 引数

特権

特権、またはコンマ(,)で区切られた特権のリストを含む文字列を指定します。プロセス特権のリストについては、『OpenVMS システム・セキュリティ・ガイド』を参照してください。[NO]ALL を除く、任意のプロセス特権を 1 つ指定します。

---

## 説明

F\$PRIVILEGE 関数を使用して、ユーザの現在のプロセスに関する特権を識別します。

特権の前に“NO”が付いている場合は、関数が TRUE を戻すよう、その特権は無効になります。F\$PRIVILEGE 関数はリストのキーワードをそれぞれチェックし、1 つでも偽があれば FALSE を戻します。

---

例

1. 

```
$ PROCPRIV = F$PRIVILEGE("OPER, GROUP, TMPMBX, NONETMBX")
$ SHOW SYMBOL PROCPRIV
PROCPRIV = "FALSE"
```

この例では、F\$PRIVILEGE 関数を使用して、プロセスが OPER 特権、GROUP 特権、および TMPMBX 特権を持っているか、またユーザが NETMBX 特権を持っていないかを調べています。

この例に示されているプロセスは、OPER (オペレータ) 特権、GROUP 特権、TMPMBX (一時メールボックス) 特権、および NETMBX (ネットワーク・メールボックス) 特権のすべてを持っています。したがって、プロセスが NETMBX 特権を持っていますが、priv-state リストに NONETMBX が指定されているため、FALSE が戻されます。他の 3 つのキーワードの論理演算の結果は真ですが、NONETMBX の結果が偽であるため、この式全体は偽であると評価されます。

---

## F\$PROCESS

現在のプロセス名文字列を得るために使用します。F\$PROCESS 関数では引数は必要ありませんが、括弧は指定しなければなりません。

---

### フォーマット

F\$PROCESS()

---

### 戻り値

現在のプロセス名を示す文字列

---

### 引数

なし

---

### 例

1. 

```
$ NAME = F$PROCESS()
$ SHOW SYMBOL NAME
NAME = "MARTIN"
```

この例では、F\$PROCESS 関数は現在のプロセス名を戻し、その名前が、NAME というシンボルに割り当てられます。

---

## F\$SEARCH

ディレクトリ・ファイルを検索し、指定したファイルの完全ファイル指定を戻します。

---

## フォーマット

F\$SEARCH(ファイル指定[, ストリーム *id*])

---

## 戻り値

ファイル指定引数に指定したファイルの、完全な形に拡張されたファイル指定を含む文字列。F\$SEARCH 関数が、ディレクトリ内でそのファイルを見つけられない場合は、空文字列 ("") を戻します。

---

## 引数

### ファイル指定

検索するファイル指定を含む文字列を指定します。装置名またはディレクトリ名を省略する場合には、F\$SEARCH 関数は、現在の省略時のディスクおよびディレクトリを、省略時の値として使用します。しかし F\$SEARCH 関数は、ファイル名またはファイル・タイプに対して省略時の値を補いません。バージョン番号を省略する場合には、F\$SEARCH 関数は、最高のバージョン番号を持つファイルのファイル指定を戻します。ファイル指定引数でワイルドカード文字を使用すると、F\$SEARCH 関数を呼び出すたびに、ファイル指定引数に当てはまる次のファイル指定が戻されます。ファイル指定引数に当てはまる最後のファイル指定が戻された後は、空文字列が戻されます。

### ストリーム *id*

検索ストリーム識別番号を示す、正の整数を指定します。

F\$SEARCH 関数を 2 回以上使用し、それぞれに異なるファイル指定引数を指定したい場合は、検索ストリーム識別番号を使用して、それぞれの検索コンテキストを管理します。コマンド・プロシージャの中で F\$SEARCH 関数を 2 回以上使用し、異なるファイル指定引数を使用する場合には、各検索を別々に識別するために、ストリーム *id* 引数を指定します。

ストリーム *id* 引数を省略すると、F\$SEARCH 関数は、異なるファイル指定引数が指定されるたびに、ディレクトリ・ファイルの先頭から検索を開始します。

---

## 説明

F\$SEARCH レキシカル関数は、RMS サービスの\$SEARCH を呼び出し、ディレクトリ・ファイルを検索し、指定したファイルの完全ファイル指定を戻します。F\$SEARCH 関数を使用すると、\$SEARCH RMS サービスを使用して、ディレクトリ内のファイルを検索できます。\$SEARCH ルーチンについての詳細は、『OpenVMS Record Management Services Reference Manual』を参照してください。

コマンド・プロシージャ内のループで F\$SEARCH 関数を使用すると、ワイルドカード文字を含むファイル指定引数と一致するファイルすべてのファイル指定を戻すことができます。F\$SEARCH 関数が実行されるたびに、ワイルドカード文字を含むファイル指定に一致する、次のファイル指定が戻されます。次に戻すファイル指定がなくなると、空文字列が戻されます。ループ内で F\$SEARCH 関数を使用する場合は、ファイル指定引数に、アスタリスク (\*) またはパーセント記号 (%) ワイルドカード文字を使用しなければなりません。ワイルドカード文字を使用しない場合は、F\$SEARCH 関数は常に同じファイル指定を戻します。

次のいずれかの方法を使用して、検索ストリームのコンテキストを維持する必要があります。

- ストリーム id 引数を指定し、明示的に維持する。
- ストリーム id 引数を省略し、F\$SEARCH 関数を実行するたびに同じファイル指定引数を使用して、暗黙に維持する。

検索ストリームのコンテキストを維持しない場合は、異なるファイル指定引数を指定するたびに、ディレクトリ・ファイルの先頭から新しい検索を開始することになります。

---

### 注意

レキシカル関数 F\$SEARCH は、ユーザが指定した選択基準に一致したすべてのファイルを戻すことができます。また、検索開始時から検索終了時までの間の任意の時点でディレクトリに存在するすべてのファイルを戻すことができます。検索中に作成、リネーム、または削除されたファイルは、戻されることがあります。

---

## 例

```
1. $ START:
$ FILE = F$SEARCH("SYS$SYSTEM:*.EXE")
$ IF FILE .EQS. "" THEN EXIT
$ SHOW SYMBOL FILE
$ GOTO START
```

このコマンド・プロシージャは、SYS\$SYSTEM ディレクトリにある、すべての.EXE ファイルの最新バージョンのファイル指定を表示します (バージョン番号にアスタリスク(\*)が使用されていないので、最新バージョンだけが表示されます)。ファイル指定引数である SYS\$SYSTEM:\*.EXE は文字列式なので、引用符(")で囲んでいます。

stream-id 引数が指定されていないため、F\$SEARCH 関数は 1 つの検索ストリームを使用します。後続の F\$SEARCH 呼び出しは、同じファイル指定引数を使用して、SYS\$SYSTEM から .EXE ファイルの次のファイル指定を戻します。各 .EXE ファイルの最新バージョンが表示された後は、F\$SEARCH 関数は空文字列を戻し、このプロシージャは終了します。

```
2. $ START:
$ COM = F$SEARCH ("*.COM;* ",1)
$ DAT = F$SEARCH ("*.DAT;* ",2)
$ SHOW SYMBOL COM
$ SHOW SYMBOL DAT
$ IF (COM.EQS. "") .AND. (DAT.EQS. "") THEN EXIT
$ GOTO START
```

このコマンド・プロシージャは、.COM ファイルと .DAT ファイルの両方の省略時のディスクおよびディレクトリを検索します。各検索のコンテキストが維持されるように、各 F\$SEARCH 関数に対して stream-id 引数が指定されている点に注意してください。

最初の F\$SEARCH 関数は、ファイル・タイプが .COM であるファイルの検索を、ディレクトリ・ファイルの先頭から開始します。.COM ファイルが検出されると、検索コンテキストを維持するためにポインタが設定されます。F\$SEARCH 関数をもう一度実行すると、ファイル・タイプが .DAT であるファイルの検索を、ディレクトリ・ファイルの先頭から開始します。このプロシージャがループを実行して、START というラベルに戻ると、各 F\$SEARCH 関数は stream-id 引数を使用して、ディレクトリ・ファイルの正しい位置から検索を開始します。.COM ファイルと .DAT ファイルのすべてのバージョンが戻された後、このプロシージャは終了します。

## レキシカル関数 F\$SEARCH

3. 

```
$ FILESPEC = F$SEARCH("TRNTO"SMITH SALLY"::DKA1:[PROD]*.DAT")
$ SHOW SYMBOL FILESPEC
FILESPEC = "TRNTO"smith password"::DKA1:[PROD]CARS.DAT"
```

この例では、F\$SEARCH 関数を使用して、リモート・ノードのファイルのファイル指定を戻しています。アクセス制御文字列は、F\$SEARCH 関数に対する引数として文字列式の一部で使用されているため、二重引用符で囲まれています。文字列式に引用符含める場合は、二重引用符を 2 つ重ねて指定する必要があります。

F\$SEARCH 関数が、アクセス制御文字列を含むノード名を戻す場合は、実際のユーザのパスワードが“password”という単語に置き換えられます。



---

## F\$SETPRV

指定した利用者特権を許可または禁止します。F\$SETPRV 関数は、利用者特権を示すキーワードのリストを戻します。このリストには、F\$SETPRV 関数が実行される前の指定した特権に関する状態が表示されます。

特権を許可または禁止するためには、指定された特権を変更できるように設定されていなければなりません。

特権の制限事項についての詳細は、『OpenVMS System Services Reference Manual』の\$SETPRV システム・サービスの説明を参照してください。

---

## フォーマット

F\$SETPRV(特権)

---

## 戻り値

F\$SETPRV 関数によって変更される前の、プロセス特権を示すキーワードを含む文字列

---

## 引数

特権

特権、またはコンマ(,)で区切られた特権のリストを定義する、文字列式を指定します。

プロセス特権のリストについての詳細は、『OpenVMS ユーザーズ・マニュアル』を参照してください。

---

## 説明

レキシカル関数 F\$SETPRV は、\$SETPRV システム・サービスを呼び出し、指定した利用者特権を許可または禁止します。\$SETPRV 関数は、利用者特権を示すキーワードのリストを戻します。このリストには、F\$SETPRV 関数が実行される前の指定した特権の状態が表示されます。

特権引数に指定した特権を変更することが許可されているかいないかに関わらず，F\$SETPRV 特権は，ユーザの現在の特権に関するキーワードを戻します。ただし，F\$SETPRV 関数が許可または禁止するのは，変更することを許可されている特権だけです。

F\$SETPRV 関数を含むプログラムまたはプロシージャを実行する場合は，F\$SETPRV 関数が，ユーザのプロセスを適正な特権状態に復元しているかを必ず確認してください。詳細は，以下の例を参照してください。

---

## 例

1. 

```
$ OLDPRIV = F$SETPRV("OPER,NOTMPMBX")
$ SHOW SYMBOL OLDPRIV
OLDPRIV = "NOOPER,TMPMBX"
```

この例では，プロセスは OPER 特権および TMPMBX 特権を変更できる権限が与えられています。F\$SETPRV 関数は OPER 特権を許可し，TMPMBX 特権を禁止します。さらに F\$SETPRV 関数は，変更する前のこれらの特権の状態を示すために，NOOPER と TMPMBX というキーワードを戻します。

特権キーワードのリストは文字列リテラルなので，引用符(“”)で囲まなければなりません。

2. 

```
$ SHOW PROCESS/PRIVILEGE
05-JUN-2001 15:55:09.60 RTA1: User: HELRIEGEL

Process privileges:

Process rights identifiers:
INTERACTIVE
LOCAL

$ NEWPRIVS = F$SETPRV("ALL, NOOPER")
$ SHOW SYMBOL NEWPRIVS
NEWPRIVS = "NOCMKRNL,NOCMEXEC,NOSYSNAM,NOGRPNAM,NOALLSPOOL,
NOIMPERSONATE,NODIAGNOSE,NOLOG_IO,NOGROUP,NOACNT,NOPRMCEB,
NOPRMMBX,NOPSWAPM,NOALTPRI,NOSETPRV,NOTMPMBX,NOWORLD,NOMOUNT,
NOOPER,NOEXQUOTA,NONETMBX,NOVOLPRO,NOPHY_IO,NOBUGCHK,NOPRMGBL,
NOSYSGBL,NOPFNMAP,NOSHMEM,NOSYSRPRV,NOBYPASS,NOSYSLCK,NOSHARE,
NOUPGRADE,NODOWNGRADE,NOGRPPRV,NOREADALL,NOSECURITY,OPER"
$ SHOW PROCESS/PRIVILEGE
05-JUN-2001 10:21:18.32 User: INAZU Process ID: 00000F24
 Node: TOKNOW Process name: "_FTA23:"

Authorized privileges:
NETMBX SETPRV SYSPRV TMPMBX
```

Process privileges:

|             |                                                |
|-------------|------------------------------------------------|
| ACNT        | may suppress accounting messages               |
| ALLSPOOL    | may allocate spooled device                    |
| ALTPRI      | may set any priority value                     |
| AUDIT       | may direct audit to system security audit log  |
| BUGCHK      | may make bug check log entries                 |
| BYPASS      | may bypass all object access controls          |
| CMEXEC      | may change mode to exec                        |
| CMKRNL      | may change mode to kernel                      |
| DIAGNOSE    | may diagnose devices                           |
| DOWNGRADE   | may downgrade object secrecy                   |
| EXQUOTA     | may exceed disk quota                          |
| GROUP       | may affect other processes in same group       |
| GRPNAM      | may insert in group logical name table         |
| GRPPRV      | may access group objects via system protection |
| IMPERSONATE | may impersonate another user                   |
| IMPORT      | may set classification for unlabeled object    |
| LOG_IO      | may do logical i/o                             |
| MOUNT       | may execute mount acp function                 |
| NETMBX      | may create network device                      |
| OPER        | may perform operator functions                 |
| PFNMAP      | may map to specific physical pages             |
| PHY_IO      | may do physical i/o                            |
| PRMCEB      | may create permanent common event clusters     |
| PRMGBL      | may create permanent global sections           |
| PRMMBX      | may create permanent mailbox                   |
| PSWAPM      | may change process swap mode                   |
| READALL     | may read anything as the owner                 |
| SECURITY    | may perform security administration functions  |
| SETPRV      | may set any privilege bit                      |
| SHARE       | may assign channels to non-shared devices      |
| SHMEM       | may create/delete objects in shared memory     |
| SYSGBL      | may create system wide global sections         |
| SYSLCK      | may lock system wide resources                 |
| SYSNAM      | may insert in system logical name table        |
| SYSPRV      | may access objects via system protection       |
| TMPMBX      | may create temporary mailbox                   |
| UPGRADE     | may upgrade object integrity                   |
| VOLPRO      | may override volume protection                 |
| WORLD       | may affect other processes in the world        |

Process rights:

INTERACTIVE  
LOCAL

System rights:

SYS\$NODE\_TOKNOW

\$ NEWPRIVS = F\$SETPRV(NEWPRIVS)

\$ SHOW PROCESS/PRIVILEGE

05-JUN-2001 16:05:07.23 RTA1: User: JERROM

Process privileges:

OPER operator privilege

```
Process rights identifiers:
 INTERACTIVE
 LOCAL
```

この例では、DCL コマンドの SHOW PROCESS/PRIVILEGE コマンドを使用して、現在のプロセスに関する特権を表示しています。このプロセスには何も特権がありません。

次に、F\$SETPRV 関数を使用して、ALL キーワードを処理し、シンボル NEWPRIVS に記録されている各特権の以前の状態を許可します。次に F\$SETPRV 関数は NOOPER キーワードを処理し、OPER の以前の状態を NEWPRIVS に記録し、OPER (オペレータ) 特権を禁止します。戻される文字列に OPER 特権が 2 回表示される点に注意してください。最初は NOOPER、次は OPER と表示されます。

SHOW PROCESS/PRIVILEGE コマンドを入力すると、現在のプロセスでは、OPER 特権を除くすべての特権が許可されていることが表示されます。

戻される文字列を F\$SETPRV のパラメータとして使用すると、プロセスは OPER 特権が許可されます。これは NEWPRIVS シンボルに OPER コマンドが 2 度表示されたためです。その結果、F\$SETPRV は最初のキーワード NOOPER を検索し、特権を無効にします。最終的には NEWPRIVS 文字列内の他のキーワードを処理した後、OPER が表示され、OPER 特権が許可されます。

現在の特権環境を保存するために ALL や NOALL を使用する場合は、次のコマンド・プロシージャを実行して、コマンド・プロシージャのプロセスを変更することをおすすめします。

```
$ CURRENT_PRIVS = F$SETPRV("ALL")
$ TEMP = F$SETPRV("NOOPER")
```

このプロシージャを使用すると、以前の特権環境に戻すために、コマンド・プロシージャの最後に次のコマンドを指定することができます。

```
$ TEMP = F$SETPRV(CURRENT_PRIVS)
```

3. 

```
$ SAVPRIV = F$SETPRV("NOGROUP")
$ SHOW SYMBOL SAVPRIV
 SAVPRIV = "GROUP"
$ TEST = F$PRIVILEGE("GROUP")
$ SHOW SYMBOL TEST
 TEST = "TRUE"
```

この例では、プロセスには GROUP 特権を変更する権限が与えられていません。しかし、F\$SETPRV 関数は、GROUP 特権の現在の設定情報を戻します。

F\$PRIVILEGE 関数は、プロセスが GROUP 特権を持つかどうかを判断するために使用されています。この関数から戻された TRUE という文字列は、

F\$SETPRV 関数がこの特権を禁止しようとしたにもかかわらず、プロセスが GROUP 特権を持つことを示しています。

4. \$ SHOW PROCESS/PRIVILEGE

05-JUN-2001 15:55:09.60 RTA1: User: KASER

Process privileges:

|           |                                               |
|-----------|-----------------------------------------------|
| AUDIT     | may direct audit to system security audit log |
| DOWNGRADE | may downgrade object secrecy                  |
| IMPORT    | may set classification for unlabeled object   |
| UPDATE    |                                               |

これらのプロセスに関する特権は VAX 固有の特権であり、OpenVMS VAX システムの Security Enhancement Service Software (SEVMS) でのみ使用されま

す。

---

## F\$STRING

指定した式に相当する文字列を返します。

---

## フォーマット

F\$STRING(式)

---

## 戻り値

指定した式に相当する文字列

---

## 引数

式

F\$STRING 関数によって評価される整数または文字列式を指定します。

整数式を指定した場合は、F\$STRING 関数は、式を評価した後で文字列に変換し、文字列を返します。文字列式を指定した場合は、F\$STRING は式を評価した結果を返します。

整数を文字列へ変換する時、F\$STRING 関数は 10 進表現を使用し先行の 0 は省略します。負の整数を変換する場合は、F\$STRING 関数は、整数を示す文字列の先頭にマイナス記号を付加します。

---

## 例

```
1. $ A = 5
 $ B = F$STRING(-2 + A)
 $ SHOW SYMBOL B
 B = "3"
```

この例で、F\$STRING 関数は、 $(-2 + A)$  という整数式の結果を、数値文字列 3 に変換します。まず、F\$STRING 関数は、 $(-2 + A)$  という式を評価します。シンボル A の値である 5 は、整数式が評価されるときに自動的に置換されます。

整数式が評価されたあと、F\$STRING 関数は、式から求められた 3 という整数を文字列の“3”に変換します。この文字列が、シンボル B に割り当てられます。

---

## F\$TIME

現在の日付/時刻を絶対時間形式で戻します。

F\$TIME 関数には引数がありませんが、括弧は指定しなければなりません。

---

## フォーマット

F\$TIME()

---

## 戻り値

現在の日付と時刻を含む文字列。戻される文字列は、次の形式の 23 文字から構成されます。

dd-mmm-yyyy hh:mm:ss.cc

現在の日が 1 ~ 9 の場合は、戻される文字列の最初の文字は空白です。文字列のうち時刻の部分は、常に文字列位置 13、つまり文字列の先頭から 12 文字のオフセット位置です。

返される文字列の空白を保持するには、等号(=)を使用しなければなりません。文字列代入演算子(:=)を使用すると、先行する空白文字が除かれます。

---

## 引数

なし

---

## 例

```
1. $ OPEN/WRITE OUTFILE DATA.DAT
 $ TIME_STAMP = F$TIME()
 $ WRITE OUTFILE TIME_STAMP
```

この例では、F\$TIME 関数を使用して、コマンド・プロシージャから作成したファイルに、タイムスタンプを記録する方法を示しています。OUTFILE は DATA.DAT というファイルの論理名で、このファイルは書き込みのためにオープンされています。F\$TIME 関数は、現在の日付および時刻文字列を戻し、この文

レキシカル関数  
F\$TIME

字列を、TIME\_STAMP というシンボルに割り当てます。WRITE コマンドは、  
日付および時刻文字列を OUTFILE に書き込みます。



---

## F\$TRNLNM

論理名を変換し、同値名文字列、または要求されている論理名の属性を戻します。

---

## フォーマット

F\$TRNLNM(論理名[, 表[, 索引[, モード[, ケース[, 項目]])

---

## 戻り値

指定した論理名の同値名、または属性。戻される値は、F\$TRNLNM 関数に指定した引数に応じて、文字列または整数になります。F\$TRNLNM 関数は、一致するものがないと空文字列("")を戻します。

---

## 引数

### 論理名

変換される論理名を含む文字列を指定します。

### 表

論理名を変換するために、F\$TRNLNM 関数が検索する論理名テーブル(1つまたは複数)名を含む文字列を指定します。ここで指定する引数は、論理名テーブルまたは論理名テーブルのリストに変換される、論理名でなければなりません。

論理名テーブルを指す論理名は、次に示す論理名テーブルのいずれかで定義しなければなりません。

- LNM\$SYSTEM\_DIRECTORY
- LNM\$PROCESS\_DIRECTORY

---

### 注意

F\$TRNLNM 関数を使用した後、CREATE/NAME\_TABLE コマンドを使用して論理名テーブルを作成し、それを F\$TRNLNM にアクセス可能なプライベート・テーブルにしたい場合は、プライベート・テーブルを含めるよう、テーブル論理名のうちのいずれか1つを再定義しなければなりません。通常 F\$TRNLNM によって検索されるすべてのテーブルを表示するには、次のコマンドを実行してください。

```
$ SHOW LOGICAL/STRUCTURE LNM$DCL_LOGICAL
```

詳細は、CREATE/NAME\_TABLE および SHOW LOGICAL コマンドの説明を参照してください。

---

table パラメータを省略した場合、省略時の設定により LNM\$DCL\_LOGICAL が使用されます。つまり F\$TRNLNM 関数は、論理名 LNM\$DCL\_LOGICAL の指す論理名テーブルを検索します。LNM\$DCL\_LOGICAL が再定義されない限り、F\$TRNLNM 関数は、プロセス、ジョブ、グループ、システムの論理名テーブルをこの順序で検索し、最初に一致した同値名を戻します。

#### 索引

論理名が 2 回以上変換された場合、戻される同値名の数を指定します。論理名が定義されていると、インデックスは、名前がリストされた順序で同値名文字列を参照します。

インデックスは 0 で始まります。つまり、同値名リストの最初の名前はインデックス 0 によって参照されます。

索引引数を指定しない場合には、省略時の値として 0 が使用されます。

#### モード

変換のアクセス・モードを指定します。アクセス・モードは、USER (省略時の設定)、SUPERVISOR、EXECUTIVE、または KERNEL のいずれか 1 つを含む文字列で指定します。

F\$TRNLNM 関数は、モード引数で指定したアクセス・モードで作成された論理名の検索を開始します。一致する論理名が見つけれないと、F\$TRNLNM 関数は、より内側のアクセス・モードで作成された論理名の検索を行い、最初に一致したものを戻します。たとえば、同一名の論理名が 2 つ定義されていて、1 つは USER アクセス・モードで、もう 1 つは EXECUTIVE アクセス・モードで作成されたらと仮定します。モード引数に USER を指定すると、F\$TRNLNM 関数は、エグゼクティブ・モードではなくユーザ・モードの、論理名の同値文字列を戻します。

#### ケース

実行される変換のタイプを指定します。ケース引数は、変換のケースおよび変換がインターロックされるかどうかの両方を制御します。

ケース引数には、CASE\_BLIND (省略時の設定)、CASE\_SENSITIVE、NONINTERLOCKED (省略時の設定)、INTERLOCKED の任意の組み合わせを指定することができます。

CASE\_BLIND を指定すると、F\$TRNLNM 関数は、ケースに関わりなく論理名テーブルで論理名を検索し、最初に見つかった論理名を変換して戻します。一致するものが見つからない場合は、この関数は空文字列("")を戻します。

CASE\_SENSITIVE を指定すると、F\$TRNLNM 関数は、論理名引数に指定したケースの論理名だけを検索します。正確に一致するものが見つからないと、F\$TRNLNM 関数は空文字列("")を戻します。

INTERLOCKED を指定すると、F\$TRNLNM 関数は、進行中のクラスタ単位の論理名のすべての修正が完了するまで実行されません。続いて一致するものが見つかり、変換の結果が返されます。一致するものが見つからないと、F\$TRNLNM 関数は空文字列('')を返します。

NOINTERLOCKED を指定すると、F\$TRNLNM 関数は直ちに実行されます。一致するものが見つかり、変換の結果が返されます。一致するものが見つからないと、F\$TRNLNM 関数は空文字列('')を返します。

#### 項目

指定した論理名に関して、F\$TRNLNM 関数が返す情報のタイプを含む文字列を指定します。次のいずれか 1 つを指定します。

| 項目          | 戻されるタイプ | 戻される情報                                                                                                                       |
|-------------|---------|------------------------------------------------------------------------------------------------------------------------------|
| ACCESS_MODE | 文字列     | 論理名に対応するアクセス・モード。アクセス・モードは、USER、SUPERVISOR、EXECUTIVE、KERNEL のいずれか 1 つ。                                                       |
| CLUSTERWIDE | 文字列     | 論理名がクラスタ単位の論理名テーブルにある場合は TRUE、そうでない場合は FALSE。                                                                                |
| CONCEALED   | 文字列     | 論理名が作成されたときに、/TRANSLATION_ATTRIBUTES 修飾子を指定して、CONCEALED 属性を指定した場合は TRUE、そうでない場合は FALSE。CONCEALED 属性は、隠し論理名を作成するために使用される。     |
| CONFINE     | 文字列     | 論理名が制限されている場合は TRUE、そうでない場合は FALSE。論理名が制限されている (TRUE) 場合には、論理名はサブプロセスにコピーされない。論理名が制限されていない (FALSE) 場合には、その論理名はサブプロセスにコピーされる。 |
| CRELOG      | 文字列     | CRELOG 属性を使用して、\$CRELOG システム・サービスまたは\$CRELNM システム・サービスにより論理名が作成された場合は TRUE、そうでない場合は FALSE。                                   |
| LENGTH      | 整数値     | 指定された論理名に対応する、同値名の長さを返す。論理名に複数の同値名が与えられている場合には、F\$TRNLNM 関数は、インデックス引数によって指定された名前の長さを返す。                                      |
| MAX_INDEX   | 整数値     | 論理名に対して定義されている最大のインデックスを返す。インデックスは、1 つの論理名にいくつの同値名が与えられているかを示す。インデックスは 0 から始まる。つまり、インデックス 0 は、同値名リストの最初の名前を参照する。             |
| NO_ALIAS    | 文字列     | 論理名が NO_ALIAS 属性を持つ場合は TRUE、そうでない場合は FALSE。NO_ALIAS 属性は、同じ名前を持つ論理名を、より外部のアクセス・モードでは作成できないことを示す。                              |
| TABLE       | 文字列     | 論理名が、論理名テーブルを指す論理名である場合は TRUE、そうでない場合は FALSE。                                                                                |

| 項目         | 戻される<br>タイプ | 戻される情報                                                                                                                       |
|------------|-------------|------------------------------------------------------------------------------------------------------------------------------|
| TABLE_NAME | 文字列         | 論理名が検出されたテーブル名。                                                                                                              |
| TERMINAL   | 文字列         | 論理名が作成されたときに、 /TRANSLATION_ATTRIBUTES 修飾子を指定して、 TERMINAL 属性を指定した場合は TRUE、そうでない場合は FALSE。 TERMINAL 属性は、論理名が反復変換の対象にならないことを示す。 |
| VALUE      | 文字列         | 省略時の設定。指定した論理名に対応する同値名を戻す。論理名に複数の同値名が与えられている場合は、F\$TRNLNM 関数は、インデックス引数で指定した名前を戻す。                                            |

## 説明

F\$TRNLNM レキシカル関数は、\$TRNLNM システム・サービスを呼び出して論理名を変換し、同値名文字列、または指定した論理名の要求した属性を戻します。反復変換は行われません。つまり、一度変換された同値名文字列は、それが論理名であるかどうかチェックされません。

F\$TRNLNM 関数を使用する場合は、最後に指定する引数の右側で使用するオプションの引数は省略できます。ただし、最後に指定する引数の左側で使用するオプションの引数を省略する場合は、プレースホルダとしてコンマ(,)を指定しなければなりません。

コマンド・プロシージャ内で F\$TRNLNM 関数を使用すると、論理名の現在の同値名文字列を保存し、後でリストアップすることができます。また、論理名が割り当てられているかどうか確認するために使用することもできます。

## 例

```
1. $ SAVE_DIR = F$TRNLNM("SYS$DISK")+F$DIRECTORY()
 .
 .
 .
 $ SET DEFAULT 'SAVE_DIR'
```

この割り当てステートメントは、F\$DIRECTORY 関数から戻された値と F\$TRNLNM 関数から戻された値を連結し、その結果である文字列を SAVE\_DIR というシンボルに割り当てます。この場合 SAVE\_DIR シンボルは、完全な装置およびディレクトリ名文字列から構成されます。

SYS\$DISK という引数は文字列であるため、引用符(" ")で囲まれています(コマンド・インタプリタは、引数が二重引用符で囲まれていない限り、英字から始まるすべての引数を、シンボルまたはレキシカル関数として取り扱います)。省略可能な引数は指定されていないため、F\$TRNLNM 関数は省略時の値を使用しません。

コマンド・プロシージャの最後で、もとの省略時のディレクトリにリセットされます。ディレクトリを元に戻す場合には、シンボル置換を強制的に実行するために、SAVE\_DIR というシンボルを一重引用符(' ')で囲まなければなりません。

2. \$ DEFINE/TABLE=LN\$GROUP TERMINAL 'F\$TRNLNM("SYS\$OUTPUT")'

この例は、コマンド・プロシージャから抜粋した行です。ここでは、(1) F\$TRNLNM 関数を使用して、現在の出力装置の名前を判断し、(2)同値文字列に基づいて、グループ論理名テーブルにエントリを作成します。

SYS\$OUTPUT という引数は文字列であるため、二重引用符で囲まなければなりません。

また、この例では、レキシカル関数が強制的に評価されるようにするために、F\$TRNLNM 関数自身も一重引用符で囲まなければなりません。引用符で囲まないと、DEFINE コマンドはレキシカル関数を自動的に評価しません。

3. \$ RESULT= -  
\_ \$ F\$TRNLNM("INFILE","LN\$PROCESS",0,"SUPERVISOR",,"NO\_ALIAS")  
\$ SHOW SYMBOL RESULT  
RESULT = "FALSE"

この例では、F\$TRNLNM 関数は、論理名 INFILE を定義したプロセス論理名テーブルを検索します。まず、スーパーバイザ・モードで作成された論理名 INFILE を検索します。一致するものがない場合は、エグゼクティブ・モードで作成された INFILE を検索します。

一致するものがあつた場合は、INFILE という名前が NO\_ALIAS 属性で作成されたかどうかを判断します。この例の場合、NO\_ALIAS 属性は指定されていません。

4. \$ foo=f\$trnlm("FOO","LN\$SYSCLUSTER",,"INTERLOCKED",)

この例では、論理名 FOO が LN\$SYSCLUSTER テーブル内で INTERLOCKED 方式で検索されて変換が実行されます。すなわち、同一クラスタ内の現在のノード上あるいは別のノード上で進行中のクラスタ単位の論理名のすべての修正は、この変換が実行される前に完了します。これにより、変換が最新の FOO 定義に基づくものであることを保証します。

ケース変換が指定されていないので、省略時の設定の CASE\_BLIND で実行されます。

## レキシカル関数 F\$TRNLNM

5. `$ foo=f$trnlnm("FOO","LNM$SYSCLUSTER",,,"INTERLOCKED,CASE_SENSITIVE",)`

この例では、CASE\_SENSITIVE および INTERLOCKED 変換の両方を指定します。

---

F\$TYPE

シンボルのデータ・タイプを判断します。シンボルが整数と等しいと定義されている場合や、シンボルが有効な整数を構成する文字列と等しいと定義されている場合には、INTEGER という文字列が戻されます。

シンボルが有効な整数を構成しない文字列と等しいと定義されている場合には、STRING という文字列が戻されます。

シンボルが未定義の場合には、空文字列("")が戻されます。

---

フォーマット

F\$TYPE(シンボル名)

---

戻り値

シンボルが整数と等しいと定義されている場合や、シンボルが有効な整数を構成する文字列と等しいと定義されている場合には、INTEGER という文字列が戻されます。シンボルが、コンテキスト・タイプ引数に PROCESS を指定した F\$CONTEXT 呼び出し、または F\$PID 関数呼び出しにより、シンボルが作成された場合は、文字列 PROCESS\_CONTEXT が戻されます。シンボルとキーワード CANCEL を使用して F\$CONTEXT を呼び出すまで、または F\$PID で空文字列("")を戻すまで、シンボルのタイプは変わりません。

同様に、F\$CSID 関数で作成されたシンボルに対しては文字列 CLUSTER\_SYSTEM\_CONTEXT が戻されます。

シンボルがコンテキスト・シンボルの場合、表 DCLI-12 に示すタイプのいずれか 1 つが戻されます。

表 DCLI-12 コンテキスト・シンボル・タイプ

| シンボル・タイプ               | シンボルを作成するレキシカル関数                             |
|------------------------|----------------------------------------------|
| PROCESS_CONTEXT        | F\$PID または F\$CONTEXT (PROCESS コンテキスト・タイプ指定) |
| CLUSTER_SYSTEM_CONTEXT | F\$CSID                                      |

シンボルが有効な整数を構成しない文字列と等しいと定義されている場合、またはシンボルのタイプがコンテキストではない場合は、文字列 STRING が戻されます。

シンボルが未定義の場合には、空文字列が戻されます。

---

## 引数

### シンボル名

評価されるシンボル名を指定します。

---

## 例

```
1. $ NUM = "52"
 $ TYPE = F$TYPE(NUM)
 $ SHOW SYMBOL TYPE
 TYPE = "INTEGER"
```

この例では、F\$TYPE 関数を使用して、NUM というシンボルのデータ・タイプを判断しています。シンボル NUM には、文字列“52”が割り当てられています。この文字列は有効な整数を構成するため、F\$TYPE 関数は文字列 INTEGER を戻します。

```
2. $ NUM = 52
 $ TYPE = F$TYPE(NUM)
 $ SHOW SYMBOL TYPE
 TYPE = "INTEGER"
```

この例では、シンボル NUM には、整数の 52 が割り当てられています。したがって F\$TYPE 関数は、このシンボルのタイプが整数データ・タイプであることを示しています。

```
3. $ CHAR = "FIVE"
 $ TYPE = F$TYPE(CHAR)
 $ SHOW SYMBOL TYPE
 TYPE = "STRING"
```

この例では、シンボル CHAR には文字列 FIVE が割り当てられています。この文字列は有効な整数を構成しないため、F\$TYPE 関数は、シンボルが文字列値を持つことを示しています。

```
4. $ x = F$CONTEXT("PROCESS",CTX,"USERNAME","SMITH")
 $ TYPE = F$TYPE(CTX)
 $ SHOW SYMBOL TYPE
 TYPE = "PROCESS_CONTEXT"
 $ x = F$CONTEXT("PROCESS",CTX,"CANCEL")
 $ TYPE = F$TYPE(CTX)
 $ SHOW SYMBOL TYPE
 TYPE = ""
```

この例では、context-type 引数に PROCESS を指定して F\$CONTEXT 関数を呼び出してシンボルを作成したので、F\$TYPE 関数は文字列 PROCESS\_CONTEXT を戻します。シンボルとselection-item 引数に CANCEL を指定して F\$CONTEXT を呼び出すまで、シンボルが戻すタイプは変わりません。



---

## F\$UNIQUE (Alpha/I64 のみ)

ファイル名に適した、クラスタ内で必ず固有である文字列を生成します (例は CLOSE /DISPOSITION を参照してください)。

F\$UNIQUE 関数には修飾子がありませんが、後に何も含まれていない括弧を添える必要があります。

---

## フォーマット

F\$UNIQUE()

---

## 戻り値

固有の文字列が含まれている 1 つの文字列。

---

## 引数

なし

---

## 例

```
1. $ WRITE SYS$OUTPUT F$UNIQUE()
 414853555241159711D7DF797CCF573F
 $
 $ WRITE SYS$OUTPUT F$UNIQUE()
 414853555241509811D7DF797E3F2777
 $
```

この例では、後に続く WRITE コマンドによって固有の文字列が戻される仕組みを示しています。

レキシカル関数  
F\$UNIQUE (Alpha/I64 のみ)

```
2. $ OPEN/WRITE TEMP_FILE 'F$UNIQUE()
$ DIRECTORY

Directory WORK1:[TEST]

594B53554C421C9C11D75463D61F58B7.DAT;1

Total of 1 file.
$
$ CLOSE/DISPOSITION=DELETE TEMP_FILE
$ DIRECTORY
%DIRECT-W-NOFILES, no files found
$
```

最初のコマンドによって一時的ファイルが作成され、そのファイルに固有の名前が与えられ、後に続く DIRECTORY コマンドによって表示されます。後でこのファイルを閉じて削除すると、ディレクトリに表示されなくなります。

---

## F\$USER

現在の利用者識別コード (UIC) を、名前形式で戻します。F\$USER 関数では、引数は使用されませんが、括弧は指定しなければなりません。

---

## フォーマット

F\$USER()

---

## 戻り値

かぎ括弧 ([]) も含めて、現在の利用者識別 (UIC) を含む文字列。UIC は、[グループ識別コード, メンバ識別コード] という形式で戻されます。

---

## 引数

なし

---

## 例

1. \$ UIC = F\$USER()  
\$ SHOW SYMBOL UIC  
UIC = "[GROUP6,JENNIFER]"

この例では、F\$USER 関数は現在の利用者識別コードを戻し、その値を UIC というシンボルに割り当てます。

---

## F\$VERIFY

プロシージャ・チェック機能が、現在オンになっているのか、オフになっているのかを示す整数値を戻します。引数を指定する場合、F\$VERIFY 関数は、プロシージャとイメージのチェック機能をオンまたはオフにすることができます。指定する引数の有無にかかわらず、F\$VERIFY 関数のあとに括弧を指定しなければなりません。

---

## フォーマット

F\$VERIFY([プロシージャ値] [, イメージ値])

---

## 戻り値

プロシージャ・チェック機能がオフの場合には整数の 0 が、プロシージャ・チェック機能がオンの場合には整数の 1 が戻されます。

---

## 引数

### プロシージャ値

プロシージャ・チェック機能をオフに設定する場合には値が 0 の整数式を指定し、プロシージャ・チェック機能をオンに設定する場合には値が 1 の整数式を指定します。

プロシージャ・チェック機能がオンの場合、コマンド・プロシージャの各行は出力装置に出力されますので、コマンド・プロシージャの実行を確認できます。

プロシージャ値引数を指定すると、まず、現在のプロシージャ・レベルのチェック機能の設定を戻します。次に、指定した引数にしたがって、コマンド・インタプリタがプロシージャ・チェック機能をオンまたはオフにします。

### イメージ値

イメージ・チェック機能をオフに設定する場合には値が 0 の整数式を指定し、イメージ・チェック機能をオンに設定する場合には値が 1 の整数式を指定します。

イメージ・チェック機能がオンの場合、コマンド・プロシージャの各行は出力装置に出力されます。

## 説明

レキシカル関数 F\$VERIFY は、プロシージャ・チェック機能のオンまたはオフを示す整数値を戻します。引数を指定すると、プロシージャおよびイメージ・チェック機能を、オンまたはオフにできます。引数の有無に関わらず、括弧は指定しなければなりません。

コマンド・プロシージャ内で F\$VERIFY 関数を使用すると、現在のチェック機能の設定を調べることができます。たとえば、チェック機能の設定を変更する前に現在の設定を保存し、その後その保存値を回復することができます。また、呼び出し前の設定に関らず、コマンド行の表示 (または印刷) を行わないコマンド・プロシージャを書くこともできます。

F\$VERIFY 関数には、0 ~ 2 個の引数を指定できます。引数を指定しないと、どちらのチェック機能の設定も変更されません。procedure-value 引数のみを指定した場合は、プロシージャ・チェック機能およびイメージ・チェック機能の両方の設定が変更されます。

両方の引数を指定した場合は、プロシージャ・チェック機能およびイメージ・チェック機能は、別々にオンまたはオフに設定できます。イメージ値引数だけを指定すると、イメージ・チェック機能だけをオンまたはオフにできます。イメージ値引数だけを指定する場合は、引数の前にコンマ(,)を指定しなければなりません。

F\$ENVIRONMENT 関数に、VERIFY\_PROCEDURE または VERIFY\_IMAGE 引数を指定することもできます。F\$ENVIRONMENT 関数は、どちらの設定状態も返すことができますが、F\$VERIFY 関数は、プロシージャ・チェック機能の設定のみ返します。

DCL は、F\$ENVIRONMENT 関数が一重引用符(')で囲まれていれば、コメント文字の後にあっても処理します。これは、DCL がコメント中で行う唯一の例外です。

## 例

```
1. $ SAVE_PROC_VERIFY = F$ENVIRONMENT("VERIFY_PROCEDURE")
 $ SAVE_IMAGE_VERIFY = F$ENVIRONMENT("VERIFY_IMAGE")
 $ SET NOVERIFY
 .
 .
 .
 $ TEMP = F$VERIFY(SAVE_PROC_VERIFY, SAVE_IMAGE_VERIFY)
```

この例は、コマンド・プロシージャの抜粋です。最初の割り当てステートメントは、現在のプロシージャ・チェック機能の設定を、SAVE\_PROC\_VERIFY というシンボルに割り当てます。2 番目の割り当てステートメントは、現在のイメー

ジ・チェック機能の設定を、SAVE\_IMAGE\_VERIFY というシンボルに割り当てます。

次に SET NOVERIFY コマンドは、プロシージャとイメージのチェック機能を禁止します。このあとで、F\$VERIFY 関数は、もとの値(シンボル SAVE\_PROC\_VERIFY および SAVE\_IMAGE\_VERIFY に割り当てられている値)を使用して、チェック機能の設定を元の状態に戻します。シンボル TEMP には、F\$VERIFY 関数によって変更される前のプロシージャ・チェック機能の設定が含まれていません(この例では、TEMP の値は使用されていません)。

```
2. $ VERIFY = F$VERIFY(0)
 .
 .
 .
 $ IF VERIFY .EQ. 1 THEN SET VERIFY
```

この例は、コマンド・プロシージャからの抜粋です。ここでは F\$VERIFY 関数を使用して、現在のプロシージャ・チェック機能の設定を保存し、プロシージャ・チェック機能とイメージ・チェック機能の両方をオフにしています。プロシージャ・チェック機能がもともとオンに設定されていた場合には、コマンド・プロシージャの最後で、プロシージャ・チェック機能とイメージ・チェック機能の両方がオンに設定されます。

---

## LIBRARY

Librarian ユーティリティを起動します。Librarian ユーティリティはオブジェクト、マクロ、ヘルプ、テキスト、または共有イメージ・ライブラリを、作成、修正、あるいは記述します。

Librarian ユーティリティについての詳細は、『OpenVMS Command Definition, Librarian, and Message Utilities Manual』またはオンライン・ヘルプを参照してください。

---

### フォーマット

LIBRARY ライブラリ・ファイル指定 [入力ファイル指定[,...]]

---

## LICENSE

ライセンス管理ユーティリティを起動します。ライセンス管理ユーティリティは、OpenVMS オペレーティング・システム上のソフトウェアのライセンスを管理します。

ライセンス管理ユーティリティについての詳細は、『OpenVMS License Management Utility Manual』またはオンライン・ヘルプを参照してください。

---

## フォーマット

LICENSE サブコマンドパラメータ



---

## LINK

OpenVMS リンカを起動します。OpenVMS リンカは 1 つまたは複数のオブジェクト・モジュールを 1 つのプログラム・イメージとリンクさせ、そのイメージの実行属性を定義します。

LINK コマンドを含めリンカについての詳細は、『OpenVMS Linker Utility Manual』またはオンライン・ヘルプを参照してください。

---

## フォーマット

LINK ファイル指定[,...]

---

## LOGIN プロシージャ

会話型モードのターミナル・セッションを開始します。

---

### フォーマット

---

### 説明

LOGIN コマンドは存在しません。システムにアクセスする意図を知らせるには、現在使用中でない端末で、Return、Ctrl/C、または Ctrl/Y を押して知らせます。システムから、ユーザ名とパスワード（および設定されている場合は第2パスワード）を求めるプロンプトが表示され、ユーザ名とパスワードが確認されます。

ユーザ名を入力した直後に省略可能な修飾子を指定します。Return を押すと、パスワードを求めるプロンプトが表示されます。

ログイン・プロシージャは、次の機能を実行します。

- ユーザ名とパスワードを、システムの利用者登録ファイル(UAF)のエントリと比較して、システムにアクセスする権利を確認する
- UAF のユーザ名エントリに基づいて、ターミナル・セッションの省略時の属性を確立する
- (存在する場合は) コマンド・プロシージャ・ファイル SYS\$SYLOGIN.COM を実行する
- (省略時の設定のディレクトリに存在する場合は) コマンド・プロシージャ・ファイル LOGIN.COM を実行する、または、(存在する場合は) UAF に定義されたコマンド・ファイルを実行する

リモートまたはダイアルアップ回線からシステムにアクセスするユーザに対して、再試行ファシリティ付きでセットアップされているシステムがあります。これらのシステムでは、ユーザまたはパスワードを誤って入力した場合に、再入力できます。ログイン情報を再入力するには、Return を押します。ユーザ名を求めるプロンプトが、再度表示されます。ユーザ名を再入力して Return を押すと、情報がシステムへ送信されます。次に、パスワードを求めるプロンプトが表示されます。ログイン情報を入力できる回数と試行間隔には制限があります。

---

## 修飾子

/CLI=コマンド言語インタプリタ

利用者登録ファイルに登録されている省略時のコマンド言語インタプリタ (CLI) を変更するために、別のコマンド言語インタプリタの名前を指定します。ここで指定する CLI は、SYS\$SYSTEM に存在しなければならず、ファイル・タイプは EXE でなければなりません。

コマンド・インタプリタを指定せずに /CLI 修飾子を指定した場合、利用者登録ファイル (UAF) にも省略時の CLI が登録されていない場合は、システムは /CLI=DCL という省略時の修飾子を使用します。

/COMMAND[=ファイル指定] (省略時の設定)

/NOCOMMAND

ログ・インしたときに、省略時のログイン・コマンド・プロシージャを実行するかどうかを制御します。別のログイン・コマンド・プロシージャの名前を指定するには、/COMMAND 修飾子を使用します。ファイル名だけを指定し、ファイル・タイプを省略する場合には、COM という省略時のファイル・タイプが使用されます。/COMMAND 修飾子だけを指定し、ファイル指定を省略する場合には、省略時のログイン・コマンド・プロシージャが実行されます。

省略時のログイン・コマンド・プロシージャを実行しない場合には、

/NOCOMMAND 修飾子を使用します。

/CONNECT (省略時の設定)

/NOCONNECT

仮想ターミナルへ再接続するかどうかを指定します。

/DISK=装置名[:]

ターミナル・セッションで使用する SYS\$DISK という論理装置に対応する、ディスク装置の名前を指定します。この名前を指定することにより、利用者登録ファイル (UAF) に設定されている、省略時の SYS\$DISK 装置を変更できます。

/LOCAL\_PASSWORD

SYSUAF.DAT ファイルに保存されているユーザ名およびパスワード情報を使用して認証を実行することを、OpenVMS に要求します。この修飾子は、外部認証を使用できない場合に、外部認証を無効にするために使用します。

/NEW\_PASSWORD

パスワードの期限切れの場合と同様に、ログ・イン前にパスワードの変更を要求します。ログイン後にパスワードを変更したい場合や、パスワードが破れたと感じた時などに使用すれば、短時間にパスワードを変更できます。

/TABLES=(コマンド・テーブル[,...])

/TABLES=DCLTABLES (省略時の設定)

利用者登録ファイル (UAF) に登録されている省略時の値を変更するために、別の CLI テーブルの名前を指定します。このテーブル名は、ファイル指定であると解釈さ

れます。省略時の装置およびディレクトリは SYS\$SHARE で、省略時のファイル・タイプは .EXE です。

論理名が使用される場合には、テーブル名指定が、システム論理名テーブルに定義されていなければなりません。

/CLI 修飾子が DCL に設定されている場合、/TABLES 修飾子は、省略時の値としてそれに対応した正しい値を使用します。/TABLES 修飾子だけを指定し、/CLI 修飾子を省略した場合には、ユーザの UAF に指定されている CLI が使用されます。

---

## 例

1. `Ctrl/Y`  
 Username: HOFFMAN  
 Password: <PASSWORD>

Ctrl/Y を押すことで、オペレーティング・システムにアクセスし、そのオペレーティング・システムは、ユーザ名を要求するプロンプトを直ちに表示します。ユーザ名が正しいかどうかをチェックしたあと、システムは、パスワードを要求するプロンプトを表示しますが、パスワードを入力してもその入力には表示されません。

2. `Return`  
 Username: HIGGINS/DISK=USER\$  
 Password: <PASSWORD>  
 Welcome to OpenVMS Alpha(TM) Operating System, Version 7.3 on node LR3  
     Last interactive login on Tuesday, 18-DEC-2001 08:41  
     Last non-interactive login on Monday, 19-DEC-2001 15:43  
 \$ SHOW DEFAULT  
   USER\$: [HIGGINS]

この Alpha の例では、/DISK 修飾子は、このターミナル・セッションに対する省略時のディスクが USER\$ になるように指定しています。SHOW DEFAULT コマンドは、USER\$ が省略時のディスクであることを示します。

3. `Return`  
 Username: HIGGINS/DISK=USER\$  
 Password: <PASSWORD>  
     Welcome to OpenVMS VAX Version 7.3 on node CELEST  
     Last interactive login on Tuesday, 15-DEC-2001 09:16:47.08  
     Last non-interactive login on Monday, 14-DEC-2001 17:32:34.27  
 \$ SHOW DEFAULT  
   USER\$: [HIGGINS]

この VAX の例では、/DISK 修飾子は、このターミナル・セッションに対する省略時のディスクが USER\$ になるように指定しています。SHOW DEFAULT コマンドは、USER\$ が省略時のディスクであることを示します。

4. **Ctrl/C**

```
Username: LIZA/CLI=DCL/COMMAND=ALTLOGIN.COM
Password: <PASSWORD>
Welcome to OpenVMS VAX Version 7.3 on node CELEST
Last interactive login on Tuesday, 15-DEC-2001 09:16:47.08
Last non-interactive login on Monday, 14-DEC-2001 17:32:34.27
$
```

/CLI 修飾子は、DCL コマンド・インタプリタを使用することを指定しています。/COMMAND 修飾子は、省略時のログイン・コマンド・ファイルの代わりに ALTLOGIN.COM というログイン・コマンド・ファイルが実行されることを指定しています。

5. **Return**

```
Username: XENAKIS
Password: <PASSWORD>
Password: <PASSWORD>
Welcome to OpenVMS VAX Version 7.3 on node CELEST
Last interactive login on Tuesday, 15-DEC-2001 09:16:47.08
Last non-interactive login on Monday, 14-DEC-2001 17:32:34.27
$
```

2 番目の Password: というプロンプトは、ユーザが第 2 パスワードを持っており、システムをアクセスするためには、このパスワードを入力しなければならないことを示しています。

6. **Return**

```
Username: JONES
Password: <PASSWORD>
User authorization failure
Return
Username: JONES
Password: <PASSWORD>
Welcome to OpenVMS Alpha (TM) Operating System, Version 7.3 on node LSR
Last interactive login on Tuesday, 15-DEC-2001 09:16:47.08
Last non-interactive login on Monday, 14-DEC-2001 17:32:34.27
1 failure since last successful login.
$
```

"User authorization failure"というメッセージは、入力されたパスワードが誤っていることを示しています。正しくログ・インすることができた場合には、最後に正しくログインしてから発生した、ログイン失敗の回数を示すメッセージが表示されます。このメッセージは、ログインを失敗した場合にだけ表示されます。

7. **Return**

```
Username: JOYCE
Password: <PASSWORD>
Welcome to OpenVMS Alpha (TM) Operating System, Version 7.3 on node LSR
Last interactive login on Tuesday, 15-DEC-2001 09:16:47.08
Last non-interactive login on Monday, 14-DEC-2001 17:32:34.27
WARNING - Primary password has expired; update immediately.
$
```

この例では、第 1 パスワードが期限切れであることを示す WARNING メッセージが出力されています。この場合、ログ・アウト前に SET PASSWORD コマンドでパスワードを再設定しなければ、これ以降はログインできなくなります。

パスワード変更方法についての詳細は、SET PASSWORD コマンドの説明を参照してください。

8. Return

```
Username: MIHALY/NEW_PASSWORD
Password: <PASSWORD>
Password: <PASSWORD>
Welcome to OpenVMS VAX Version 7.3 on node CELEST
Last interactive login on Tuesday, 15-DEC-2001 09:16:47.08
Last non-interactive login on Monday, 14-DEC-2001 17:32:34.27
Your password has expired; you must set a new password to log in.

Old password: <PASSWORD>
New password: <PASSWORD>
Verification: <PASSWORD>
```

この例では、ユーザ名 MIHALY の後に/NEW\_PASSWORD 修飾子を指定しています。このため、システムは、ログイン直後に新パスワードの設定を強制します。この時のプロンプトは、コマンド行から SET PASSWORD コマンドを実行した場合のものと同じです。

---

# LOGOUT

会話型モードのターミナル・セッションを終了します。

---

## フォーマット

LOGOUT

---

## 説明

ターミナル・セッションを終了するには、LOGOUT コマンドを使用しなければなりません。通常的环境中、LOGOUT コマンドを使用しないで端末の電源を切る、または電話接続を切断すると、ログインしたままになります。

SET HOST コマンドを使用してリモートのプロセッサにログインする場合は、通常、LOGOUT コマンドを使用してリモートのセッションを終了する必要があります。

---

## 修飾子

/BRIEF

簡略な形式のログアウト・メッセージを出します。コマンド・インタプリタは、ログアウトする時にユーザ名と日付、および時刻を表示します。

/FULL

詳細な形式のログアウト・メッセージを出します。/FULL を指定すると、コマンド・インタプリタは、ターミナル・セッションの会計情報の要約を表示します。バッチ・ジョブの省略時の設定は、/FULL です。

/HANGUP

/NOHANGUP

公衆電話回線を使用したターミナルの場合に、ログアウトしたあと、電話を切るかどうかを指定します。省略時には、ターミナル・ポートに対する/HANGUP 修飾子の設定が、電話を切るかどうかを決定します。この修飾子を使用できるかどうかは、システム管理者によって設定されます。

---

## 例

1. \$ LOGOUT  
GILLINGS logged out at 05-JUN-2001 17:48:56.73

この LOGOUT コマンドでは、省略時の簡略メッセージ形式が使用されていません。したがって、会計情報は表示されません。

2. \$ LOGOUT/FULL  
GUZMAN logged out at 05-JUN-2001 14:23:45.30  
Accounting information:  
Buffered I/O count: 22 Peak working set size: 90  
Direct I/O count: 10 Peak virtual size: 69  
Page faults: 68 Mounted volumes: 0  
Charged CPU time: 0 00:01:30.50 Elapsed time: 0 04:59:02.63  
Charged vector CPU time: 0 00:00:21.62

この LOGOUT コマンドは、/FULL 修飾子を指定しているので、このターミナル・セッションの会計情報の要約を表示します。



---

## MACRO

OpenVMS VAX での省略時の設定では、VAX MACROアセンブラを起動して、1 つまたは複数のアセンブリ言語ソース・ファイルをアセンブルします。OpenVMS Alpha および OpenVMS I64 での省略時の設定では、MACRO compiler for OpenVMS Systems を起動して、VAX アセンブリ言語ソース・ファイルを、ネイティブの OpenVMS Alpha または OpenVMS I64 オブジェクト・コードにアセンブルします。

Alpha システム上に MACRO-64 アセンブラがインストールされている場合は、/ALPHA 修飾子を指定すると、MACRO コマンドは MACRO-64 アセンブラを起動します。

/MIGRATION 修飾子は、Alpha および I64 での省略時の指定です。これらのプラットフォームでは、MACRO という指定は MACRO/MIGRATION と指定したのと同じです。

MACRO compiler for OpenVMS Systems についての詳細は、『OpenVMS MACRO-32 Porting and User's Guide』を参照してください。

---

## フォーマット

MACRO ファイル指定[,...]

---

## MAIL

Mail ユーティリティを起動します。Mail ユーティリティを使用すると、システムの他のユーザにメッセージを送ることができます。

Mail ユーティリティについての詳細は、『OpenVMS ユーザーズ・マニュアル』またはオンライン・ヘルプを参照してください。

---

### フォーマット

MAIL [ファイル指定] [受取人名]

---

## MERGE

Sort/Merge ユーティリティを起動します。Sort/Merge ユーティリティは 2 ~ 10 の同様にソートされた入力ファイルから、1 つのファイルを出力します。マージする入力ファイルは、ソート順に指定しなければならない点に注意してください。

Sort/Merge ユーティリティについての詳細は、『OpenVMS ユーザーズ・マニュアル』またはオンライン・ヘルプを参照してください。

---

## フォーマット

MERGE 入力ファイル指定 1, 入力ファイル指定 2[,...] 出力ファイル指定

---

## MESSAGE

Message ユーティリティを起動します。Message ユーティリティは、1 つまたは複数のメッセージ定義のファイルをコンパイルします。

Message ユーティリティについての詳細は、『OpenVMS Command Definition, Librarian, and Message Utilities Manual』またはオンライン・ヘルプを参照してください。

---

### フォーマット

MESSAGE ファイル指定[,...]

---

## MONITOR

Monitor ユーティリティを起動します。Monitor ユーティリティは、特定の時間隔においてシステム全体の性能に関するデータのクラスを監視します。

Monitor ユーティリティについての詳細は、『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル』またはオンライン・ヘルプを参照してください。

---

### フォーマット

MONITOR [/修飾子[,...]] クラス名[,...] [/修飾子[,...]]

---

# MOUNT

Mount コマンド (MOUNT) は、ディスクまたは磁気テープを処理可能状態にします。

---

## フォーマット

MOUNT 装置名[:/,...]/ [ボリューム・ラベル[,...]] [論理名[:/]]

---

## パラメータ

装置名[:/,...]

ボリュームをマウントする装置の物理装置名または論理名を指定します。ボリュームが階層記憶制御装置 (HSC: hierarchical storage controllers) に接続されていないシステムでは、次の形式を使用します。

ddcu:

ddlは、物理装置の装置タイプです。たとえば、ディスク・ドライブ RA60 の装置タイプはDJ、ディスク・ドライブ RA80、RA81 の装置タイプはDUです。cはコントローラ、uは装置のユニット番号です。

HSC が搭載されているシステムでは、次のいずれかの形式を使用します。

ノード\$ddcu:

割り当てクラス\$ddcu:

HSC にデュアル・ポート接続されている装置の場合、割り当てクラス形式を使用します。たとえば、\$125\$DUA23 は、ユニット番号 23 の RA80、RA81 ディスクであり、割り当てクラスは\$125\$です。HSC ディスクの場合、c は必ず A です。TROLL\$DJA12 は、ユニット番号 12 の RA60 ディスクであり、TROLL という HSC に接続されています。命名規則については、『OpenVMS Cluster システム』を参照してください。

装置名に総称を使用すれば、コントローラやユニット番号を指定しない場合に、指定した装置名構成要素を最初に満たす装置がマウントされます。指定した装置にボリュームが物理的にマウントされていない場合、ボリュームを装置に装填することを指示するメッセージが表示されます。ドライブにボリュームを装填すると、MOUNT の動作は完了します。

複数の装置名をディスク・ボリューム・セットまたは磁気テープ・ボリューム・セットに指定する場合、装置名をコンマまたはプラス記号で区切ります。磁気テープ・ボ

リューム・セットの場合、装置名より多くのボリューム・ラベルを指定でき、またボリュームより多くの装置名を指定できます。

ボリューム・ラベル[,...]

ボリュームのラベルを指定します。

ラベルに使用できる装置タイプ別文字数は次のとおりです。

| 装置タイプ         | ラベルの文字数 |
|---------------|---------|
| 磁気テープ         | 0 ~ 6   |
| Files-11ディスク  | 1 ~ 12  |
| ISO 9660 ディスク | 1 ~ 32  |

OpenVMS では、ディスク・ボリューム・ラベルの最初の 12 文字は各ドメイン内で固有の名前でなければなりません。たとえば、/GROUP 修飾子を使用して同じグループの複数のメンバがマウントするディスクは、固有のラベルを持たなければなりません。しかし、別のドメインにマウントされるディスク（たとえば、/GROUP 修飾子を使用してマウントされるディスクとプライベートにマウントされるディスク）は、ボリューム・ラベルが同じでもかまいません。

/SYSTEM 修飾子または/CLUSTER 修飾子を使用して ISO 9660 ボリュームをマウントするときに、ボリューム・ラベルの最初の 12 文字が固有のラベルでない場合には、/OVERRIDE=IDENTIFICATION 修飾子を使用して別のボリューム・ラベルを指定しなければなりません。このオプションを選択した場合、デバイスに対するマウント・チェックは禁止されます。

さらに、ボリュームがボリューム・セットの一部であり、ボリューム・セット名の最初の 12 文字がボリューム・ラベルの最初の 12 文字と同じである場合には、ロック・マネージャ・デッドロックが発生します。この問題を回避するには、ボリューム・ラベルを無効にするか (/OVERRIDE 修飾子を使用して)、またはボリューム・セット名を無効にしなければなりません (/BIND 修飾子を使用して)。

複数のボリューム・ラベルを指定する場合、ラベルをコンマまたはプラス記号で区切ります。指定するすべてのボリュームは同一のボリューム・セットに属していなければならず、ボリューム番号の昇順で指定しなければなりません。

磁気テープ・ボリューム・セットをマウントする場合、指定した装置名の数とボリューム・ラベルの数を等しくする必要はありません。テープの終わり (EOT) マークまで来ると、次のボリュームを指定した装置のいずれかにマウントする指示が出力されます。この指示は、ユーザには出力されず、オペレータだけを対象としています。

ディスク・ボリューム・セットをマウントする場合、指定した各ボリューム・ラベルは、リスト内で同じ位置にある装置名と対応していなければなりません。

/FOREIGN または/NOLABEL の修飾子でボリュームをマウントする場合や、/OVERRIDE=IDENTIFICATION を指定した場合、ボリューム・ラベル・パラメータは不要です。これらの修飾子で論理名を指定するときは、ボリューム・ラベル・パラメータに任意の英数字を入力します。

論理名[:]

ボリュームに対応づける 1 ~ 255 文字の英数文字列論理名を定義します。

論理名を指定しない場合、個々のディスク・ドライブには、省略時の論理名 DISK\$ボリューム・ラベルを設定し、ディスク・ボリューム・セットのルート・ボリュームをマウントする装置には、省略時の論理名 DISK\$ボリューム・セット名を設定します。DISK\$ボリューム・ラベルや DISK\$ボリューム・セット名とは異なる論理名をマウント要求で指定すると、2 つの論理名が対象装置に対応づけられます。

磁気テープ・ドライブの論理名を指定しない場合、論理名 TAPE\$ボリューム・ラベルがリスト内の最初の磁気テープ装置だけに設定されます。この場合、省略時の論理ボリューム・セット名は設定されません。

/GROUP または/SYSTEM を指定しないかぎり、指定した論理名はプロセス論理名テーブルに格納されます。/GROUP を指定した場合はグループ論理名テーブルに格納され、/SYSTEM を指定した場合はシステム論理名テーブルに格納されます。

/CLUSTER 修飾子を指定すると、クラスタ内の各ノードに対して論理名が設定されます。

---

#### 注意

---

SYS\$SYSTEM の実行可能イメージのファイル名と同じ論理名を設定すると、そのイメージを起動できなくなります。

---

分散ファイル・システム (DFS) アクセス・ポイントとしてボリュームに設定されている論理名は使用しないでください。論理名と同じ名前でも DFS アクセス・ポイントを追加しようとすると、次のように DFS は異常終了します。

```
$ SHOW LOG DISK$*
(LNM$SYSTEM_TABLE)
"DISK$TIVOLI_SYS" = "TIVOLI$DUA0:"

$ RUN SYS$SYSTEM:DFS$CONTROL
DFS> ADD ACCESS DISK$TIVOLI_SYS TIVOLI$DUA0:[000000]
%DNS-W-NONSNNAME, Unknown namespace name specified
```

ボリュームの論理名がプロセス・プライベート・テーブルに格納されている場合、ボリュームをディスマウントしても名前は削除されません。



---

## 説明

Mount コマンド (MOUNT) は、ディスクまたは磁気テープを処理可能状態にします。MOUNT を使用することにより、指定した装置が別のユーザに割り当てられることを防止し、指定した装置にボリュームを物理的にロードし、指定したラベルにボリューム・ラベルを一致させることができます。磁気テープ・ボリュームの場合、VOL1 ラベルのボリューム・アクセス可能フィールドもチェックされます。

通常、コマンドを入力したユーザに装置が割り当てられます。ただし、/SHARE、/GROUP、/SYSTEM の修飾子はボリュームを共用可能とするので、これらの修飾子を使用してボリュームをマウントした場合は、装置割り当てが解除されます。

---

### 注意

---

ボリュームを装置にマウントするには、その装置に対する読み込み (R)、制御 (C) のいずれかのアクセス権が必要です。

---

プロセスの木構造に属するすべてのサブプロセスは、ジョブに使用するボリュームをマウントまたはディスマウントできます。ジョブに使用するボリュームをプライベート・ボリュームとしてサブプロセスがマウントすると、ジョブのマスタ・プロセスが装置所有者となります。この割り当ては、非常に重要です。サブプロセスが削除されても、ボリュームは、依然プライベートとしてマウントされたままだからです。ただし、サブプロセスが装置を明示的に割り当て、この装置にプライベート・ボリュームをマウントした場合は、サブプロセスが装置所有者となります。この装置にアクセスできるのは、SHARE 特権を持つサブプロセスだけです。

動作が正常終了すると、SYS\$OUTPUT にメッセージが出力されます。動作が異常終了すると、エラー・メッセージが出力されます。

MOUNT のようなファイル操作ユーティリティでは、索引ファイル・ビットマップやストレージ・ビットマップのコピーを保持するために、仮想メモリを割り当てるものがあります。OpenVMS バージョン 7.2 からビットマップのサイズがさらに大きくなったので、これらのユーティリティでの仮想メモリの必要条件もこれに対応して増加しました。大規模ビットマップのボリュームで MOUNT を使用するには、ページ・ファイル・クォータを増やす必要がある可能性があります。また、OpenVMS VAX システムでは、システム・パラメータ VIRTUALPAGECNT も増やす必要がある可能性があります。仮想メモリ・サイズは、ビットマップのブロックごとの VAX ページ (または Alpha での 512 バイトのページレット) で示されます。ブロック単位の索引ファイル・ビットマップのサイズは、ファイルを 4096 で割ったものの最大数であることに注意してください。MOUNT での仮想メモリ条件は、ボリューム・セットにおける索引ファイル・ビットマップとストレージ・ビットマップすべてのサイズの合計と同じです。この条件は、ボリュームをリビルドした場合に限って、MOUNT に適用されます。

データベース・ボリュームのように、ファイル・システムにキャッシングさせたくないディスク・ボリュームがある場合は、/NOCACHE 修飾子を使用します。この修飾子はボリュームに対するキャッシングを無効にします。

- 以下のメタデータ・キャッシュが、ローカル・ノード上のボリュームのメタデータをキャッシングしないようにします。

拡張キャッシュ  
ファイル識別キャッシュ  
制限キャッシュ

- ローカルな拡張ファイル・キャッシュまたは仮想入出力キャッシュが、ボリューム中のファイルをキャッシングしないようにします。

#### MOUNT 使用法の要約

Mount コマンド (MOUNT) は、ディスク・ボリュームまたは磁気テープ・ボリュームを処理可能状態にします。

MOUNT を起動するには、DCL の MOUNT コマンドを入力し、続けて装置名、ボリューム・ラベル、論理名を入力します。/OVERRIDE=IDENTIFICATION を指定する場合や/FOREIGN または/NOLABEL の修飾子を使用する場合を除き、装置名とボリューム・ラベルは必ず指定してください。論理名は省略可能です。

マウント動作が正常終了すると、DCL レベルに戻ります。異常終了した場合も、エラー・メッセージを出力して DCL レベルに戻ります。Ctrl/Y または Ctrl/C を押すと、動作はアボートされ、DCL プロンプトに戻ります。

MOUNT の出力先は、/COMMENT 修飾子と/MESSAGE 修飾子で指定できます。オペレータ補助を要する場合は、/COMMENT を使用して、オペレータ要求に対し必要な情報を指定します。/COMMENT テキスト文字列は、オペレータ・ログ・ファイルと SYS\$OUTPUT に送信されます。文字列は 78 文字以内です。

マウント要求メッセージを現在の SYS\$OUTPUT 装置に送るには、省略時の値である/MESSAGE 修飾子を使用します。オペレータ補助マウントで/NOMESSAGE を指定すると、メッセージは SYS\$OUTPUT に送られません。ただし、オペレータ・ターミナルでメッセージの受信が許可されている場合は、オペレータ・ターミナルに送信されます。

MOUNT の修飾子の多くは、特殊な特権を必要とします。一部の修飾子では、必要となる特権が、指定する修飾子キーワードによって異なります。詳細については、それぞれの修飾子の説明を参照してください。次の表は、特殊な特権を要する MOUNT の修飾子の一覧です。

| 修飾子           | キーワード           | 必要な特権               |
|---------------|-----------------|---------------------|
| /ACCESSED     |                 | OPER                |
| /CACHE=       | [NO]EXTENT[=n]  | OPER                |
|               | [NO]FILE_ID[=n] | OPER                |
|               | [NO]QUOTA[=n]   | OPER                |
| /FOREIGN      |                 | VOLPRO <sup>1</sup> |
| /GROUP        |                 | GRPNAM              |
| /MULTI_VOLUME |                 | VOLPRO              |
| /OVERRIDE=    | ACCESSIBILITY   | VOLPRO <sup>1</sup> |
|               | EXPIRATION      | VOLPRO <sup>1</sup> |
|               | LOCK            | VOLPRO <sup>1</sup> |
|               | SHADOW          | VOLPRO <sup>1</sup> |
| /OWNER_UIC=   | uic             | VOLPRO <sup>1</sup> |
| /PROCESSOR=   | UNIQUE          | OPER                |
|               | SAME:device     | OPER                |
|               | file-spec       | OPER と CMKRNL       |
| /PROTECTION=  | code            | VOLPRO <sup>1</sup> |
| /QUOTA        |                 | VOLPRO <sup>1</sup> |
| /SYSTEM       |                 | SYSNAM              |
| /WINDOWS=     | n               | OPER                |

<sup>1</sup>または、ユーザ UIC がボリューム UIC と同じでなければならない。

## 修飾子

/ACCESSED=n

ODS-1 ディスク・ボリュームで同時に使用する予定のディレクトリの概数を指定します。(ODS-2 ボリュームの場合は、/ACCESSED 修飾子は意味がありません。)

ボリュームの初期化で指定した省略時の値を変更するには、0 ~ 255 の値を指定してください。

/ACCESSED を使用するには、ユーザ特権 OPER が必要です。

例

以下のコマンドは、WORK というラベルのボリュームを DKA1 にマウントしています。ボリューム上のアクティブ・ディレクトリ数として、150 を指定しています。

```
$ MOUNT/ACCESSED=150 DKA1 WORK
```

/ASSIST (省略時の設定)

/NOASSIST

マウント要求でエラーが発生したときに、オペレータまたはユーザによる作業を許可します。

/ASSIST 修飾子を指定した場合、マウント動作中にエラーが発生すると、ユーザと特定のオペレータに通知されます。ユーザやオペレータは、動作をアボートしたり、エラー状態を修正して動作を続行させたりできます。

オペレータ補助メッセージは、メッセージ受信が許可されているすべてのオペレータ・ターミナルに送信されます。磁気テープ・マウント要求は TAPE と DEVICE のオペレータに送られ、ディスク・マウント要求は DISK と DEVICE のオペレータに送られます。つまり、ディスク装置をマウントするときにオペレータの補助が必要となった場合、メッセージは DISK オペレータに送信されます。オペレータ・ターミナルの許可または禁止の詳細については、REPLY コマンドの説明を参照してください。

マウント要求に対するオペレータの応答は、SYS\$OUTPUT に書き込まれてユーザのターミナルに表示されるか、またはバッチ・ジョブ・ログに書き込まれます。

マウント補助要求の受信と応答が許可されているオペレータ・ターミナルがない場合、該当するユーザにメッセージが表示されます。ボリュームを指定したドライブに装填すれば、オペレータからの応答は不要となります。マウントをバッチ・ジョブで起動し、メッセージ受信が許可されているオペレータ・ターミナルがない場合、マウントはアボートされます。エラー・メッセージとユーザが行うべき処置については、『OpenVMS System Messages: Companion Guide for Help Message Users』を参照してください。

省略時の設定では、/ASSIST です。/NOASSIST で無効にできます。

#### 例

以下のコマンドは、DOC というラベルのディスク・ボリューム HSG80 Fibre Channel をマウントし、WORK という論理名を設定しています。/NOASSIST は、オペレータの介入が不要であることを指示しています。

```
$ MOUNT/NOASSIST 1DGA0: DOC WORK
%MOUNT-I-MOUNTED, DOC mounted on _1DGA0: (NODE)
```

/AUTOMATIC (省略時の設定)

/NOAUTOMATIC

MOUNT が磁気テープまたは ISO 9660 CD-ROM に対して自動ボリューム切り替えとラベル付けを許可するか、禁止するかを指定します。

#### 磁気テープ

ボリューム・セットに複数の磁気テープ・ドライブを設定している場合、磁気テープ補助制御プロセス (MTACP) は、ボリューム・セットに設定されているドライブの中で次に使用できるものを順次選択することによって、ボリュームを切り替えます。MTACP は、ボリューム・セットの次のリールが該当するドライブにロードされていると仮定します。

MTACP がボリューム・セットに書き込む場合、ラベルを作成し、ボリューム・セット内の最初の磁気テープに設定されている保護と作成したラベルとを使用して、磁気テープを初期化します。ボリューム・セットから読み込む場合、ラベルを出力し、次の磁気テープをこのラベルでマウントします。誤った磁気テープがドライブにロードされている場合や磁気テープがロードされていない場合、MTACP は、正しい磁気テープをロードするように指示するメッセージを、オペレータ・ターミナルに送信します。

MTACP が出力するラベルは、6 文字のボリューム識別子フィールドに格納されます。最初の 4 文字は、MOUNT コマンドで指定したラベルの最初の 4 文字です。4 文字未満の場合、4 文字になるまでアンダスコアで埋められます。第 5, 6 文字は、ボリューム・セットにおける該当リールの相対ボリューム番号です。

/NOAUTOMATIC を指定すると、テープの終わりで次のドライブに切り替える作業も、ボリューム・セットに追加する各リールにラベルを指定する作業も、オペレータが行います。

#### ISO 9660 CD-ROM

ISO 9660 のもとでは、ボリューム・セットに対して入出力操作を実行するために、すべてのボリューム・セット・メンバをマウントする必要はありません。省略時の設定では、入出力操作でマウントされていないボリューム・セット・メンバにアクセスしようとしたとき、システムでマウントしたボリューム・セットの場合はすべての DISK CLASS オペレータに、また、プライベートにマウントしたボリューム・セットの場合はボリューム・セットを所有しているプロセスに、オペレータ・メッセージが送信されます。このメッセージでは、要求された入出力操作を実行するためにどのボリューム・セット・メンバをマウントしなければならないかが指定されます。/NOAUTOMATIC を指定した場合には、マウントされていないボリューム・セット・メンバに対して入出力操作を実行しようすると、SS\$\_DEVNOTMOUNT エラー・メッセージが戻されます。

#### 例

以下の例は、MOUNT コマンドで指定したラベルを第 2 ボリュームに使用することを指定しています。第 2 ボリュームにまだラベルが設定されていない場合、オペレータが REPLY/INIT を使用してラベルを指定する必要があります。

```
$ MOUNT/NOAUTOMATIC MTA0: ABCD,EFGH
```

#### /BIND=ボリューム・セット名

1 つまたは複数のディスク・ボリュームで構成されるボリューム・セットを作成します。または、1 つまたは複数のディスク・ボリュームを既存ボリューム・セットに追加します。

ボリューム・セット名パラメータは、ボリューム・セットを識別する 1 ~ 12 文字の英数字名を指定します。

ISO 9660 ボリューム・セット名は 1 ~ 128 文字の長さです。

OpenVMS では、ボリューム・セット名は、最初の 12 文字が固有の名前でなければなりません。さらに、ボリューム・セット名の最初の 12 文字がボリューム・ラベルの最初の 12 文字と同じである場合には、ロック・マネージャ・デッドロックが発生します。この問題を回避するには、(/OVERRIDE 修飾子を使用して) ボリューム・ラベルを無効にするか、または (/BIND 修飾子を使用して) ボリューム・セット名を無効にしなければなりません。

ボリューム・セットを作成する場合やボリューム・セットにボリュームを追加する場合は、/BIND 修飾子を指定してください。個々のボリュームをボリューム・セットからディスマウントするには、DISMOUNT の/UNIT 修飾子を使用する必要があります。/UNIT を指定しない場合、1 つのボリュームだけをディスマウントしても、ボリューム・セット全体がディスマウントされます。

ボリューム・セットを作成すると、ラベル・リストにおける位置に基づいて、ボリューム・ラベル・リスト内のボリュームに相対ボリューム番号が設定されます。ボリューム・セットで最初に指定したボリュームが、ルート・ボリュームとなります。

ボリューム・セットにボリュームを追加した場合、最初に指定したボリューム・ラベルがルート・ボリュームのラベルとなります。そうでない場合は、ルート・ボリュームはすでに稼動しています。

ファイルとデータをすでに格納している複数のボリュームでボリューム・セットを作成する場合、MOUNT/BIND コマンドを入力しても、ファイル・システムはエラー・メッセージを出力しません。ただし、ディレクトリ構造は正しくバインドされていないので、これらのボリュームをボリューム・セットとして使用することはできません。

/SYSTEM 修飾子または/CLUSTER 修飾子を使用して ISO 9660 ボリュームをマウントするときに、ボリューム・ラベルの最初の 12 文字が固有のラベルでない場合には、/BIND=ボリューム・セット名修飾子を使用して別の 12 文字のボリューム・ラベルを指定しなければなりません。このオプションを選択した場合には、デバイスに対するマウント・チェックは禁止されます。

---

#### 注意

---

ボリューム・セットにバインドされたボリュームは、容易にアンバインドできません。バウンド・ボリューム・セット (BVS) をアンバインドする方法は、次のとおりです。

1. BVS のイメージ・バックアップをとる。
  2. BVS のすべてのボリュームを初期化する。
  3. /NOINITIALIZE 修飾子で 1 つのボリュームにイメージ復元を実行する。または、1 つのボリュームに非イメージ復元を実行する。
-

## 例

以下のコマンドは、LIBRARY というボリューム・セットを作成します。このボリューム・セットは、装置 DMA0、DMA1、DMA2 に物理的にマウントされている BOOK1、BOOK2、BOOK3 のラベルのボリュームで構成されています。

```
$ MOUNT/BIND=LIBRARY DMA0:,DMA1:,DMA2: BOOK1,BOOK2,BOOK3
```

以下のコマンドは、TEST3 という論理名のボリューム・セットを作成します。ボリューム・セット TEST3 はシャドウ化されませんが、ボリューム・セット (TEST3011 および TEST3012) の各構成要素はシャドウセットであり、ボリューム・セット全体に冗長性を持たせます。

```
$ MOUNT/BIND=TEST3 DSA3011/SHADOW=(1DUA402:,1DUA403:),
DSA3012/SHADOW=(1DUA404:,1DUA405:) TEST3011,TEST3012 TEST3
```

```
/BLOCKSIZE=n
```

磁気テープ・ボリュームの省略時のブロック・サイズを指定します。

パラメータ *n* は、磁気テープ・ボリュームの省略時のブロック・サイズ値を指定します。VMS RMS では 20 ~ 65,532 の値、VMS RMS 以外では 18 ~ 65,534 の値を指定します。省略時の設定では、レコードは 2,048 バイトのブロック単位で磁気テープ・ボリュームに書き込まれます。フォーリン磁気テープやラベルなし磁気テープの省略時の値は 512 バイトです。

/BLOCKSIZE は、次の場合に指定する必要があります。

- HDR2 ラベルのない磁気テープをマウントする場合

この種の磁気テープには、ブロック・サイズを指定する必要があります。たとえば、RT-11 磁気テープをマウントする場合は、/BLOCKSIZE=512 と指定します。

- ブロック・サイズが省略時ブロック・サイズ (2,048 バイト) より大きい磁気テープをマウントする場合

この場合は、最も大きいブロックをブロック・サイズとして指定します。

## 例

以下の例は、1,000 バイトのブロック・サイズを指定しています。/FOREIGN 修飾子でマウントする磁気テープの省略時の値は 512 です。

```
$ MOUNT/FOREIGN/BLOCKSIZE=1000 MTA1:
```

```
/CACHE=(キーワード[,...])
```

```
/NOCACHE
```

ディスクに対し、システム作成時に設定したキャッシング制限を禁止または変更するかどうかを制御します。テープ・コントローラが書き込みキャッシングをサポートしている場合、TAPE\_DATA オプションを指定すると書き込みキャッシングが許可されます。

次の表は、この修飾子に対するキーワードの一覧です。

| キーワード                    | 説明                                                                                                                                                                                                                                                                                                                                                                                                     |
|--------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| EXTENT[=n]<br>NOEXTENT   | および<br>拡張キャッシングを許可または禁止します。拡張キャッシングを許可するには、オペレータ・ユーザ特権 (OPER) が必要です。許可する場合、拡張キャッシュのエントリ数 $n$ を指定してください。NOEXTENT は、EXTENT=0 と同じであり、拡張キャッシングを禁止します。                                                                                                                                                                                                                                                      |
| FILE_ID[=n]<br>NOFILE_ID | および<br>ファイル識別子キャッシングを許可または禁止します。ファイル識別子キャッシュを許可するには、オペレータ・ユーザ特権 (OPER) が必要です。許可する場合、1 より大きい値のエントリ数 $n$ を指定してください。NOFILE_ID は FILE_ID=1 と同じであり、ファイル識別子キャッシングを禁止します。                                                                                                                                                                                                                                     |
| LIMIT=n                  | 拡張キャッシュの最大空き空間を、ディスク上の現在の空き空間の 1,000 分の 1 の単位で指定します。                                                                                                                                                                                                                                                                                                                                                   |
| QUOTA[=n]<br>NOQUOTA     | および<br>クォータ・キャッシングを許可または禁止します。クラッシュ・キャッシングを許可するには、オペレータ・ユーザ特権 (OPER) が必要です。許可する場合、クォータ・キャッシュのエントリ数 $n$ を指定してください。通常、 $n$ には、1 つのディスクに対してクォータが許可されるアクティブ・ユーザの最大数が設定されます。NOQUOTA は QUOTA=0 と同じであり、クォータ・ファイル・キャッシングを禁止します。                                                                                                                                                                                |
| TAPE_DATA                | テープ・コントローラがキャッシングをサポートする場合、磁気テープ装置の書き込みキャッシングを許可します。省略時の設定では、/NOCACHE です。書き込みキャッシングを許可するには、TAPE_DATA を指定します。テープ・コントローラが書き込みキャッシングをサポートしない場合、キーワードは無視されます。<br><br>磁気テープをディスマウントしても、書き込みバッファは許可されたままとなります。書き込みバッファを禁止するには、/NOCACHE 修飾子でテープをマウントします。<br><br>テープが圧縮をサポートしている場合は、省略時の設定では圧縮が行われ、キャッシングが有効になります。圧縮をサポートするテープ記憶装置では、次のコマンドが有効です。<br><br>\$ MOUNT TAPE_DATA/FOREIGN/MEDIA=NOCOMPACTION/NOCACHE |
| WRITETHROUGH             | ファイル・ヘッダに対し、デフォード書き込み機能を無効にします。PATHWORKS のようなアプリケーションの性能を上げることのできるこの機能は、省略時の設定では有効になっています。デフォード書き込み機能は Files-11 ODS-1 ボリュームでは使用できません。                                                                                                                                                                                                                                                                  |

ディスク・オプションを使用して、/CACHE 修飾子は、システム作成時に設定したディスク・キャッシング制限を 1 つ以上変更します。TAPE\_DATA オプションを /CACHE 修飾子と併用すると、指定したテープ・コントローラの書き込みキャッシングが許可されます。

/CACHE 修飾子を指定せず、/MEDIA\_FORMAT=COMPACTION 修飾子によってキャッシングが暗黙に示されない場合、省略時の設定により、キャッシングは許可されます。

複数のオプションを指定する場合には、各オプションをコンマで区切り、リスト全体を括弧で囲みます。[NO]EXTENT、[NO]FILE\_ID、LIMIT、[NO]QUOTA オプションはディスク・デバイスにだけ適用されます。TAPE\_DATA オプションはテープ・デバイスにだけ適用されます。

/NOCACHE 修飾子が有効であるのは、圧縮が許可されていない場合だけです。圧縮が許可されている場合には (/MEDIA\_FORMAT=COMPACTION によって)、省略時の設定により、キャッシングは許可されます。



ディスク・デバイスに/NOCACHE を指定すると、対象ボリュームのすべてのキャッシングが禁止されます。/NOCACHE 修飾子は、/CACHE=(NOEXTENT,NOFILE\_ID,NOQUOTA,WRITETHROUGH) と同じです。

磁気テープ装置に/NOCACHE を指定すると、対象ボリュームに対し、テープ・コントローラの書き込みキャッシュが禁止されます。これは、TAPE\_DATA オプションの省略時の設定です。

例

以下のコマンドは、FILES というラベルのディスク装置 HSG80 Fibre Channel をマウントし、WORK という論理名を設定しています。/CACHE 修飾子により、60 エントリの拡張キャッシュ、60 エントリのファイル識別子キャッシュ、20 エントリのクォータ・キャッシュが許可されています。ライトバック・キャッシングは禁止しています。

```
$ MOUNT/CACHE=(EXTENT=60,FILE_ID=60,QUOTA=20,WRITETHROUGH) -
_$ 1DGA0: FILES WORK
%MOUNT-I-MOUNTED, FILES mounted on _1DGA0: (NODE)
```

以下のコマンドは、ボリューム TAPE を装置 MUA0 にマウントし、テープ・コントローラの書き込みキャッシュを MUA0 に対して許可しています。

```
$ MOUNT/CACHE=TAPE_DATA MUA0: TAPE
%MOUNT-I-MOUNTED, TAPE mounted on _NODE$MUA0:
```

/CLUSTER

ボリュームをローカル・ノードにマウントした後、またはボリュームがローカル・ノード上の/SYSTEM にすでにマウントされている場合に、既存の OpenVMS Cluster 内のすべてのノードにボリュームをマウントすることを指定します。つまり、ボリュームをクラスタ単位でマウントします。

クラスタ単位でマウントできるのは、システム・ボリュームとグループ・ボリュームだけです。/SYSTEM と/GROUP のいずれの修飾子も使用せずに/CLUSTER 修飾子を指定した場合、/SYSTEM が使用されます。クラスタ装置命名規約に従ってください。「ノード\$装置名」と「割り当てクラス\$装置名」のいずれか適した方を使用します。

クラスタ単位でマウントする場合、グループ・ボリュームではユーザ特権 GRPNAM、システム・ボリュームではユーザ特権 SYSNAM が必要です。

OpenVMS Cluster に属さないシステムの場合、/CLUSTER 修飾子の効果はありません。

## 例

以下の MOUNT/CLUSTER コマンドは、SNOWWHITE というボリュームを DOPEY\$DMA1 にマウントし、続けてボリュームをクラスタ単位でマウントしています。SHOW DEVICE/FULL コマンドは、ボリューム自体とボリュームをマウントした各ノードの情報を表示しています。

```
$ MOUNT/CLUSTER DOPEY$DMA1: SNOWWHITE DWARFDISK
%MOUNT-I-MOUNTED, SNOWWHITE mounted on _DOPEY$DMA1:
$ SHOW DEVICE/FULL DWARFDISK:
```

```
Disk 2DMA1: (DOPEY), device type RK07, is online, mounted,
file-oriented device, shareable, served to cluster via MSCP
Server, error logging is enabled.
```

|                  |          |                      |                        |
|------------------|----------|----------------------|------------------------|
| Error count      | 0        | Operations completed | 159                    |
| Owner process    | " "      | Owner UIC            | [928,49]               |
| Owner process ID | 00000000 | Dev Prot             | S:RWED,O:RWED,G:RW,W:R |
| Reference count  | 1        | Default buffer size  | 512                    |
| Total blocks     | 53790    | Sectors per track    | 22                     |
| Total cylinders  | 815      | Tracks per cylinder  | 3                      |
| Allocation class | 2        |                      |                        |

|                   |             |                                  |                            |
|-------------------|-------------|----------------------------------|----------------------------|
| Volume label      | "SNOWWHITE" | Relative volume number           | 0                          |
| Cluster size      | 3           | Transaction count                | 1                          |
| Free blocks       | 51720       | Maximum files allowed            | 6723                       |
| Extend quantity   | 5           | Mount count                      | 7                          |
| Mount status      | System      | Cache name                       | "_\$_255\$DWARF1:XQPCACHE" |
| Extent cache size | 64          | Maximum blocks in extent cache   | 5172                       |
| File ID cache siz | 64          | Blocks currently in extent cache | 0                          |
| Quota cache size  | 25          | Maximum buffers in FCP cache     | 349                        |

```
Volume status: ODS-2, subject to mount verification,
file high-water marking, write-through caching enabled.
Volume is also mounted on DOC, HAPPY, GRUMPY, SLEEPY, SNEEZY, BASHFUL.
```

## /COMMENT=文字列

マウント動作でオペレータ補助が必要な場合に、オペレータ要求に追加情報を含めることを指定します。

パラメータ文字列は、オペレータ・ログ・ファイルと現在の SYS\$OUTPUT に出力されるテキスト文字列を指定します。文字列は 78 文字以内です。

## 例

以下のコマンドは、TESTSYS というディスク・ボリュームを DYA1 という装置にマウントするよう、オペレータに指示しています。/COMMENT 修飾子によって、ボリュームの記憶位置をオペレータに通知しています。オペレータがボリュームを DYA1 に装填すると、MOUNT は動作をリトライします。動作が終了すると、オペレータ要求は取り消されます。

```
$ MOUNT DYA1: TESTSYS/COMMENT="Volume in cabinet 6."
%MOUNT-I-OPRQST, Please mount volume TESTSYS in device _DYA1:
Volume in cabinet 6.
%MOUNT-I-MOUNTED TESTSYS mounted on _DYA1:
%MOUNT-I-OPRQSTDON, operator request canceled - mount
completed successfully
```

以下のコマンドは最初の例と同じですが、要求する装置が使用中であるため、オペレータがマウントをアボートしています。

```
$ MOUNT DYA1: TESTSYS/COMMENT="Volume in cabinet 6."
%MOUNT-I-OPRQST, Please mount volume TESTSYS in device _DYA1:
Volume in cabinet 6.
%MOUNT-I-OPREPLY, This is a '/pending' response from the operator.
31-DEC-1990 10:27:38.15, request 2 pending by operator TTB6
%MOUNT-I-OPREPLY, This is a '/abort' response from the operator.
31-DEC-1990 10:29:59.34, request 2 aborted by operator TTB6
%MOUNT-F-OPRABORT, mount aborted by operator
```

以下のコマンドは、ボリューム TESTSYS を装置 DYA0 にマウントするよう、オペレータに指示しています。要求する装置が使用中であるため、オペレータは装置 DYA1 にマウント先を変更しています。

```
$ MOUNT DYA0: TESTSYS/COMMENT="Volume in cabinet 6,
once again with feeling."
%MOUNT-I-OPRQST, Please mount volume TESTSYS in device _DYA0:
Volume in cabinet 6, once again with feeling.
%MOUNT-I-OPREPLY, Substitute DYA1:
31-DEC-1990 10:43:42.30, request 3 completed by operator TTB6
%MOUNT-I-MOUNTED, TESTSYS mounted on _DYA1:
```

```
/CONFIRM 仮想ユニット名[:] /SHADOW=(物理装置名[:],[...])
/NOCONFIRM 仮想ユニット名[:] /SHADOW=(物理装置名[:],[...])
```

指定されたディスク・デバイスでコピー操作を実行する前に、MOUNT を一時停止し、確認を要求します。この修飾子を使用できるのは、ボリューム・シャドウイング・オプションが用意されている場合だけです。追加情報については、『Volume Shadowing for OpenVMS 説明書』を参照してください。

シャドウ・セットをマウントするときに、フル・コピー操作を確認する要求を出すかどうかを制御します。/CONFIRM 修飾子は/SHADOW 修飾子と組み合わせて使用しなければなりません。コピー先の物理デバイスのボリューム・ラベルとボリューム所有者を表示する場合は、/CONFIRM を使用します。この場合、コピー操作の前に一時停止し、次のプロンプトが表示されます。

```
Allow FULL shadow copy on the above member(s)? [N]:
```

Y または YES と応えると、マウント動作は自動的に続行し、フル・コピー動作が許可されます。N, NO, <RETURN>, <CTRL/Z> と応えると、コピーが不要のボリュームも含め、指定したボリュームをまったくマウントせずに、コマンドが終了します。これらの応答以外を入力すると、プロンプトが再び表示されます。

/CONFIRM は、/NOCOPY と似ています。シャドウ・セットを会話形式でマウントするには /CONFIRM、サイト別スタートアップ・コマンド・プロシージャ SYS\$MANAGER:SYSTARTUP\_VMS.COM では /NOCOPY を使用します。

#### 例

次の例はデータを消去する前に、シャドウ・セットに含める装置の状態をチェックしています。指定した装置を使用してシャドウ・セットを作成した後、フル・コピー動作を実行してよいかどうかを尋ねています。YES と応えた結果、シャドウ・セットのマウントが実行されています。

```
%MOUNT/CONFIRM DSA0:/SHADOW=(200DKA200:,200DKA300:,200DKA400:) X5OZCOPY
```

```
%MOUNT-F-SHDWCOPYREQ, shadow copy required
```

```
Virtual Unit - DSA0 Volume Label - X5OZCOPY
```

```
Member Volume Label Owner UIC
```

```
200DKA200: (VIPER1) X5OZCOPY [SYSTEM]
```

```
200DKA400: (VIPER1) X5OZCOPY [SYSTEM]
```

```
Allow FULL shadow copy on the above member(s)? [N:] Y
```

```
%MOUNT-I-MOUNTED, X5OZCOPY mounted on _DSA0:
```

```
%MOUNT-I-SHDWMEMSUCC, _200DKA300: (VIPER1) is now a valid member of
the shadow set
```

```
%MOUNT-I-SHDWMEMCOPY, _200DKA200: (VIPER1) added to the shadow set
with a copy operation
```

```
%MOUNT-I-SHDWMEMCOPY, _200DKA400: (VIPER1) added to the shadow set
with a copy operation
```

/COPY 仮想ユニット名[:] /SHADOW=(物理装置名[:],[...]) (省略時の設定)

/NOCOPY 仮想ユニット名[:] /SHADOW=(物理装置名[:],[...])

シャドウ・セットをマウントするときに指定された物理デバイスでのコピー操作を許可または禁止します。この修飾子を使用できるのは、ボリューム・シャドウイング・オプションが用意されている場合だけです。追加情報については、『Volume Shadowing for OpenVMS 説明書』を参照してください。

/COPY 修飾子はシャドウ・セット・メンバに対し、コピー動作を実行します。

/NOCOPY でシャドウ・セットをマウントすれば、フル・コピー動作の対象であるかどうかを確認できます。コピー動作の対象ボリュームが存在する場合、コピー動作が不要のボリュームも含み、指定したボリュームをまったくマウントせずに、コマンドが終了します。

/NOCOPY 修飾子は /CONFIRM に似ています。システム固有のスタートアップ・コマンド・プロシージャ SYS\$MANAGER:SYSTARTUP\_VMS.COM でシャドウ・セットをマウントする場合は /NOCOPY を使用し、会話でマウントする場合は /CONFIRM を使用してください。

## 例

次の例は/NOCOPY 修飾子を使用して、データを消去する前に、シャドウ・セットに含める装置の状態を確認しています。コピー動作を必要としない場合にかぎり、指定した装置を使用してシャドウ・セットを作成することを指示しています。装置 DUA7 は、コピーしなければシャドウ・セットのメンバとなれないので、マウントは実行されません。/COPY を指定してコマンドを再び起動すれば、必要なコピーを実行してシャドウ・セットを作成できます。

```
$ MOUNT/NOCOPY DSA2: /SHADOW=(1DUA4:,1DUA6:,1DUA7:) -
_$ SHADOWVOL DISK$SHADOWVOL
%MOUNT-F-SHDWCOPYREQ, shadow copy required
%MOUNT-I-SHDWMEMFAIL, DUA7: failed as a member of the shadow set
%MOUNT-F-SHDWCOPYREQ, shadow copy required
```

/DATA\_CHECK[(キーワード[,...])]

ボリュームを初期化したときに指定した読み込みチェック・オプションと書き込みチェック・オプションのいずれか、または両方を無効にします。

キーワード READ は、すべての読み込み動作直後に、チェックを行います。キーワード WRITE は、すべての書き込み動作直後に、チェックを行います。

2 種類のキーワードのいずれかまたは両方を指定できます。両方を指定する場合、キーワードをコンマで区切り、リスト全体を括弧で囲みます。

キーワードを指定せずに/DATA\_CHECK 修飾子を指定すると、省略時の設定で/DATA\_CHECK=WRITE が使用されます。

## 例

以下のコマンドは SAM というラベルのボリュームを CLEMENS\$DKA2 にマウントし、BOOK という論理名を設定しています。以前に指定された INITIALIZE/DATA\_CHECK=WRITE を/DATA\_CHECK=READ 修飾子が無効にしているので、BOOK に対する以降の読み込み動作はチェックされます。

```
$ MOUNT/DATA_CHECK=READ CLEMENS$DKA2: SAM BOOK
```

/DENSITY=キーワード

磁気テープに書き込む密度を指定します。この修飾子は、/FOREIGN 修飾子付きでテープをマウントするときに限って有効です。テープの密度を変更する場合、テープに対する最初の操作は書き込みである必要があります。

テープに対してサポートする密度を次の表に示します。

表 DCLI-13 テープに対するキーワード

| キーワード                                                                                                                        | 意味                                              |
|------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------|
| DEFAULT                                                                                                                      | 省略時の密度                                          |
| 800                                                                                                                          | NRZI 800 BPI (インチ当たりビット数)                       |
| 1600                                                                                                                         | PE 1600 BPI                                     |
| 6250                                                                                                                         | GRC 6250 BPI                                    |
| 3480                                                                                                                         | IBM 3480 HPC 39872 BPI                          |
| 3490E                                                                                                                        | IBM 3480 圧縮形式                                   |
| 833                                                                                                                          | DLT TK50: 833 BPI                               |
| TK50                                                                                                                         | DLT TK50: 833 BPI                               |
| TK70                                                                                                                         | DLT TK70: 1250 BPI                              |
| 6250                                                                                                                         | RV80 6250 BPI に相当するもの                           |
| 注意: OpenVMS バージョン 7.2 よりも前の TMSCP/TUDRIVER コードでは、上記のリストにあるシンボルだけが解釈されます。この表の後に示すシンボルは、OpenVMS Alpha および I64 システムでのみサポートされます。 |                                                 |
| TK85                                                                                                                         | DLT Tx85: 10625 BPI—Cmpt III - Alpha/I64 のみ     |
| TK86                                                                                                                         | DLT Tx86: 10626 BPI—Cmpt III - Alpha/I64 のみ     |
| TK87                                                                                                                         | DLT Tx87: 62500 BPI—Cmpt III - Alpha/I64 のみ     |
| TK88                                                                                                                         | DLT Tx88: (Quantum 4000)—Cmpt IV - Alpha/I64 のみ |
| TK89                                                                                                                         | DLT Tx89: (Quantum 7000)—Cmpt IV - Alpha/I64 のみ |
| QIC                                                                                                                          | QIC ドライブは、すべてドライブ設定専用 - Alpha/I64 のみ            |
| TK85                                                                                                                         | DLT Tx85: 10625 BPI—Cmpt III - Alpha/I64 のみ     |
| TK86                                                                                                                         | DLT Tx86: 10626 BPI—Cmpt III - Alpha/I64 のみ     |
| TK87                                                                                                                         | DLT Tx87: 62500 BPI—Cmpt III - Alpha/I64 のみ     |
| TK88                                                                                                                         | DLT Tx88: (Quantum 4000)—Cmpt IV - Alpha/I64 のみ |
| TK89                                                                                                                         | DLT Tx89: (Quantum 7000)—Cmpt IV - Alpha/I64 のみ |
| QIC                                                                                                                          | QIC ドライブは、すべてドライブ設定専用 - Alpha/I64 のみ            |
| 8200                                                                                                                         | Exa-Byte 8200 - Alpha/I64 のみ                    |
| 8500                                                                                                                         | Exa-Byte 8500 - Alpha/I64 のみ                    |
| DDS1                                                                                                                         | Digital Data Storage 1—2G - Alpha/I64 のみ        |
| DDS2                                                                                                                         | Digital Data Storage 2—4G - Alpha/I64 のみ        |
| DDS3                                                                                                                         | Digital Data Storage 3—8-10G - Alpha/I64 のみ     |
| DDS4                                                                                                                         | Digital Data Storage 4 - Alpha/I64 のみ           |
| AIT1                                                                                                                         | Sony Advanced Intelligent Tape 1 - Alpha/I64 のみ |
| AIT2                                                                                                                         | Sony Advanced Intelligent Tape 2 - Alpha/I64 のみ |
| AIT3                                                                                                                         | Sony Advanced Intelligent Tape 3 - Alpha/I64 のみ |
| AIT4                                                                                                                         | Sony Advanced Intelligent Tape 4 - Alpha/I64 のみ |
| DLT8000                                                                                                                      | DLT 8000 - Alpha/I64 のみ                         |

(次ページに続く)

表 DCLI-13 (続き) テープに対するキーワード

| キーワード   | 意味                          |
|---------|-----------------------------|
| 8900    | Exabyte 8900 - Alpha/I64 のみ |
| SDLT    | SuperDLT - Alpha/I64 のみ     |
| SDLT320 | SuperDLT320 - Alpha/I64 のみ  |

テープ密度のキーワードは短縮することができません。

INITIALIZE コマンドでテープを初期化し、密度を指定しなかった場合、テープは使用している媒体とドライブの省略時の密度で初期化されます (通常は利用可能な密度のうち最も高いもの)。

テープの密度は、そのテープがテープの開始 (BOT) 位置にある場合にだけ変更できます。記録済のテープの密度を変更する場合、最初の操作は書き込みにする必要があります。テープに対する最初の操作が読み込みである場合、/DENSITY 修飾子でどのような密度を指定しても、磁気テープは、テープが記録された最初のレコードにある密度に設定されます。

例

以下のコマンドは、テープを MFA0: ドライブに/FOREIGN でマウントし、論理名 TAPE を割り当てます。/DENSITY 修飾子は、このテープが TK87 で書き込まれることを指定しています。

```
$ MOUNT/FOREIGN/DENSITY=TK87 MFA0: TAPE
```

/EXTENSION=n

ディスク・ファイルを拡張するブロック数を指定します。コマンドやプログラムで別の指定を行う場合以外に適用されます。

パラメータ *n* は、0 ~ 65,535 の値を指定します。この指定により、ボリューム初期化時に指定した値は無効となります。

例

以下のコマンドは DOC というラベルのボリュームを DKA0 にマウントし、WORK という論理名を設定しています。WORK 上のファイルの省略時の拡張ブロック数 64 を指定しています。

```
$ MOUNT/EXTENSION=64 DKA0: DOC WORK
```

/FOREIGN

OpenVMS オペレーティング・システムで使用している標準形式とは異なる形式のボリュームであることを示します。

ANSI 標準形式以外の磁気テープ・ボリュームと Files-11 形式以外のディスク・ボリュームには、/FOREIGN 修飾子を使用します。

/FOREIGN 修飾子でボリュームをマウントすることにより、ボリューム読み込みプログラムは、ボリュームのラベルを処理できるようになります。OpenVMS オペレーティング・システムには、ボリュームを処理するための補助制御プロセス (ACP) が用意されています。

DOS-1 ボリュームおよび RT-11 ボリュームは、/FOREIGN 修飾子でマウントし、Exchange ユーティリティ (EXCHANGE) で処理する必要があります。『OpenVMS Exchange Utility Manual』 (ドキュメンテーション CD-ROM に用意されています) を参照してください。

フォーリン・ボリュームの省略時の保護は、システムと所有者の場合は RWLP (読み込み、書き込み、論理入出力、物理入出力)、グループと一般の場合はアクセスなしです。/GROUP も指定した場合、グループ・メンバにも RWLP アクセスが許可されます。/SYSTEM または /SHARE を指定すると、グループと一般の両方に RWLP アクセスが許可されます。/GROUP、/SYSTEM、/SHARE によって省略時の保護が変更されることはありません。

現在の形式が Files-11 であるボリュームを /FOREIGN 修飾子でマウントするには、ユーザ特権 VOLPRO を持っているか、またはボリュームの UIC と同じ UIC を持っている必要があります。

/FOREIGN 修飾子は次の修飾子と互換性がありません。

/ACCESSED, /AUTOMATIC, /BIND, /CACHE, /[NO]CONFIRM, [NO]COPY, /EXTENSION, /HDR3, /INITIALIZE, /LABEL, /PROCESSOR, /QUOTA, /REBUILD, /SHADOW, /OVERRIDE=EXPIRATION, and /WINDOWS.

例

以下のコマンドは、フォーリン磁気テープを MTA1 というドライブにマウントしています。

```
$ MOUNT/FOREIGN MTA1: ABCD TAPE
```

以下のコマンドは、RK07 という装置をフォーリン・ボリュームとして DMA2 にマウントし、DISK\$SAVEDISK という省略時の論理名を設定しています。SAVEDISK は、ファイル構造ではないボリュームとして、順編成ディスクの BACKUP セーブ処理に使用できます。

```
$ MOUNT/FOREIGN DMA2: SAVEDISK
```

/GROUP

ユーザが MOUNT コマンドを入力したときに、このユーザと同じ UIC 内のグループ番号を持つ他のユーザがボリュームを利用できるようにします。

ボリュームの論理名は、グループ論理名テーブルに格納されます。/GROUP 修飾子を使用するには、ユーザ特権 GRPNAM が必要です。



別のグループが所有者であるボリュームの場合、その保護設定値によってはアクセスが拒否されることがあります。

ISO 9660 ボリューム・セットに対して/GROUP 修飾子を使用することはできません。

/GROUP 修飾子は/OVERRIDE=IDENTIFICATION、/SHARE、/SYSTEM 修飾子とは互換性がありません。

例

以下のコマンドは、PAYVOL1、PAYVOL2、PAYVOL3 というラベルのボリュームで構成されるボリューム・セットをマウントし、グループ全体で利用できるようにしています。PAY という論理名が、このセットに設定されており、ユーザは、この論理名 PAY でこれらのボリューム上のファイルにアクセスできます。

```
$ MOUNT/GROUP DB1:, DB2:, DB3: PAYVOL1,PAYVOL2,PAYVOL3 PAY
```

以下のコマンドは、PAYVOL4 というラベルのボリュームを MASTER\_PAY という既存ボリューム・セットに追加しています。このボリューム・セットのルート・ボリュームは、コマンド入力時に稼動中でなければなりません。

```
$ MOUNT/GROUP/BIND=MASTER_PAY DB4: PAYVOL4
```

/HDR3 (省略時の設定)  
/NOHDR3

ANSI 標準ヘッダ・ラベル 3 を磁気テープ・ボリュームに書き込むかどうかを制御します。

省略時の設定では、ヘッダ・ラベル 3 が書き込まれます。/NOHDR3 修飾子を指定すれば、HDR3 ラベルを正しく処理しないシステムで磁気テープに書き込みを行えます。

例

次の例は INITIALIZE コマンドと MOUNT コマンドが、ANSI 形式の磁気テープを処理可能状態にしています。/NOHDR3 修飾子により、HDR3 ラベルを書き込まないことを指定しています。したがってこの磁気テープは、処理系に依存するラベルを正しく処理しないシステムに移植できます。

```
$ INITIALIZE MTA0: ABCD
$ MOUNT/NOHDR3 MTA0: ABCD
```

/INCLUDE 仮想ユニット名[:] /SHADOW=(物理装置名[:],[...])  
/NOINCLUDE 仮想ユニット名[:] /SHADOW=(物理装置名[:],[...]) (省略時の設定)

以前のシャドウ・セットを、シャドウ・セットが破壊される前の状態に自動的に復元します。この修飾子を使用できるのは、ボリューム・シャドウイング・オプションが用意されている場合だけです。追加情報については、『Volume Shadowing for OpenVMS 説明書』を参照してください。

/INCLUDE 修飾子は、自動的にシャドウ・セットをマウントし、システムがクラッシュする前の状態に復元します。シャドウ・セットを最初にマウントしたときに使用していた仮想ユニット名とまったく同じ名前を入力してください。仮想ユニット名の形式は、DSAnnnn: です。

オリジナル・シャドウ・セット内の 1 つ以上のディスク装置を、/SHADOW 修飾子で指定する必要もあります。標準装置命名形式\$割り当てクラス\$ddcu[:]を使用してください。1 つの装置だけを指定する場合は、括弧で囲みません。

/INCLUDE 修飾子はコマンド行のどこに指定してもかまいません。

省略時の修飾子は/NOINCLUDE です。

#### 例

次の例は、シャドウ・セットを作成しています。マウント対象のシャドウ・セット・メンバは、ソフトウェアが自動的に決定します。/SHADOW 修飾子により、2 つのシャドウ・セット・メンバが正しくコピーされます。\$1\$DUA10 の方が新しいボリュームなので、\$1\$DUA10 が\$1\$DUA11 にコピーされます。

シャドウ・セットが正しくデismountされ、まだ処理されていない書き込み要求が残されていない場合は、シャドウ・セット・デバイスに矛盾がないため、そのまま追加でき、コピー操作やマージ操作は必要ありません。未処理の書き込み要求がある場合には、Volume Shadowing for OpenVMS はコピー操作またはマージ操作を自動的に実行します。

```
$ MOUNT/INCLUDE DSA0: /SHADOW=1DUA10: SHADOWVOL
%MOUNT-I-MOUNTED, SHADOWVOL mounted on DSA0:
%MOUNT-I-SHDWMEMSUCC, _1DUA10: (MEMBER1) is now a valid member of
the shadow set
%MOUNT-I-SHDWMEMCOPY, _1DUA11: (MEMBER2) added to the shadow set
with a copy operation
```

#### /INITIALIZE=CONTINUATION

磁気テープ・ボリューム・セットにボリュームを追加する場合、ボリュームへの書き込みを行う前に初期化することを指定します。

#### 例

次の例は/INITIALIZE=CONTINUATION 修飾子が、MOUNT コマンド独自の継続ラベルを設定することを指示しています。この場合、オペレータが REPLY /BLANK=n と入力すれば、オリジナル・ラベルをもとにシステムがラベルを設定します。ABCD02, ABCD03 というように、MOUNT コマンドで指定したラベルに、適宜番号が追加されます。

```
$ MOUNT/INITIALIZE=CONTINUATION MTA0: ABCD
```

/LABEL (省略時の設定)  
/NOLABEL

OpenVMS オペレーティング・システムで使用している標準形式のボリュームであることを指示します。つまり、磁気テープ・ボリュームの場合は ANSI 標準形式、ディスク・ボリュームの場合は Files-11 形式です。

省略時の設定では、/LABEL です。

/NOLABEL は、/FOREIGN と同じで、FOREIGN フラグを設定します。

例

以下のコマンドは ANSI 磁気テープを MFA1 にマウントし、TAPE\$TAPE という省略時の論理名を設定しています。

```
$ MOUNT/LABEL MFA1: TAPE
```

/MEDIA\_FORMAT=CDROM

ISO 9660 (または High Sierra) 形式であるものとして、ボリュームをマウントします。

/MEDIA\_FORMAT=CDROM 修飾子は、ISO 9660 (または High Sierra) 形式であるものとしてボリュームをマウントすることをマウント・サブシステムに要求します。

---

#### 注意

---

この修飾子は CD-ROM のマウント (ISO 9660 または High Sierra) を指定します。ボリュームが ISO 9660 形式または High Sierra CD-ROM 形式であることがわかっている場合には、この修飾子を指定してください。

省略時の設定では、Mount コマンドは Files-11 ODS-2 形式で CD-ROM を読み込もうとします。この修飾子を指定した場合には、Mount コマンドは Files-11 ODS-2 形式のマウントを実行しません。

CD-ROM の一部を Files-11 ODS-2 形式で記録し、別の部分を ISO 9660 形式で記録することが可能であるため、この修飾子を使用すれば、CD-ROM のマウントを指定できます (ISO 9660 または High Sierra)。

---

/MEDIA\_FORMAT=[NO]COMPACTION

データ圧縮をサポートするテープ・ドライブにおけるデータ圧縮とデータ・レコード・ブロッキングを許可し、制御します。

/MEDIA\_FORMAT 修飾子を使用すれば、テープをマウントし、テープ・ドライブがデータ圧縮をサポートする場合に、データ圧縮とレコード・ブロッキングを許可することができます。データ圧縮とレコード・ブロッキングを行うと、より多くのデータをテープに格納できます。

レコードは、圧縮してブロッキングすること、圧縮をサポートしないテープ・ドライブで記録する場合と同じように記録することもできます。圧縮をサポートするテ

ブ・ドライブに圧縮または非圧縮を指定すると、1本のテープ全体にその指定が適用されます。

/MEDIA\_FORMAT=[NO]COMPACTION 修飾子は/DENSITY 修飾子と互換性がありません。

Files-11テープにデータ圧縮を許可すると、キャッシングも自動的に許可されます。

---

#### 注意

---

/MEDIA\_FORMAT=[NO]COMPACTION 修飾子が有効であるのは、フォーリン・マウントの場合だけです。

/MEDIA\_FORMAT=[NO]COMPACTION 修飾子は、Files-11テープに対しては効果がありません。Files-11テープには、テープ初期化時に設定した状態が適用されます。

---

#### 例

以下のコマンドは、テープをフォーリン・マウントしてデータ圧縮とレコード・ブロッキングを許可し、BOOKS という論理名を設定しています。

```
$ MOUNT/FOREIGN/MEDIA_FORMAT=COMPACTION MUA0: BOOKS
```

ラベル BOOKS のテープをFiles-11マウントし、データ圧縮とレコード・ブロッキングを許可しています。初期化時に圧縮が禁止されているので、/MEDIA\_FORMAT=COMPACTION の効果はありません。

```
$ INIT/MEDIA_FORMAT=NOCOMPACTION MUA0: BOOKS
$ MOUNT/MEDIA_FORMAT=COMPACTION MUA0: BOOKS
```

/MESSAGE (省略時の設定)

/NOMESSAGE

マウント要求メッセージを、現在の SYS\$OUTPUT 装置に送信します。

オペレータ補助マウントで/NOMESSAGE を指定した場合、メッセージは SYS\$OUTPUT に出力されません。ただし、オペレータ・ターミナルが許可されている場合、オペレータ・ターミナルには出力されます。

#### 例

この例では、SLIP というラベルの装置 RL02 をドライブ DLA0 にマウントし、DISC という論理名を設定しています。/NOMESSAGE 修飾子により、マウント要求メッセージをユーザ・ターミナルに同報通信することが禁止されています。

```
$ MOUNT/NOMESSAGE DLA0: SLIP DISC
```

/MOUNT\_VERIFICATION (省略時の設定)

/NOMOUNT\_VERIFICATION

装置がマウント・チェックの対象であることを指定します。

/MOUNT\_VERIFICATION 修飾子は、次のような媒体に対して有効です。

- Files-11構造レベル2または5のディスク(フォーリン・マウント・ディスクでは、マウント・チェックはサポートされない)
- ISO 9660 形式と High Sierra 形式の CD-ROM
- フォーリン・ラベルおよび ANSI ラベルの磁気テープ・ボリューム

#### 例

以下のコマンドは、FILES というラベルのディスク装置 HSG80 Fibre Channel をマウントし、WORK という論理名を設定しています。/CACHE 修飾子により、拡張キャッシング、ファイル識別子キャッシング、クォータ・キャッシング、ライトバック・キャッシングが禁止され、/NOMOUNT\_VERIFICATION 修飾子により、マウント・チェックが禁止されています。

```
$ MOUNT/CACHE=(NOEXTENT,NOFILE_ID,NOQUOTA,WRITETHROUGH) -
_$ /NOMOUNT_VERIFICATION 1DGA0: FILES WORK
%MOUNT-I-MOUNTED, FILES mounted on _1DGA0: (NODE)
```

#### /MULTI\_VOLUME

/NOMULTI\_VOLUME (省略時の設定)

フォーリン磁気テープ・ボリュームやラベルなし磁気テープ・ボリュームに対し、ボリューム・アクセス・チェックを無効とするかどうかを制御します。

/MULTI\_VOLUME は、MOUNT が解釈できるラベルがないボリュームのアクセス・チェックを無効にします。OpenVMS バージョン 5.0 より前に開発され、個々のリールのマウントとディスマウントを行わないまま、フォーリン・マウントされたマルチ・ボリュームのテープ・セットを処理するソフトウェアを使用している場合、最初のボリュームを/MULTI\_VOLUME 修飾子でマウントする必要があるかもしれません。

フォーリン・マウントされたマルチ・ボリュームの磁気テープ・セットをサポートするユーティリティで第2以降のボリュームを処理する必要があり、かつ OpenVMS の Mount コマンドが解釈できるラベルがそれらのボリュームに付いていない場合に、この修飾子を使用します。

省略時の設定では、すべてのテープ・ボリュームに対して OpenVMS の Mount コマンドのアクセス・チェックが行われます。OpenVMS バージョン 5.0 より前に開発されたサード・パーティ製のユーティリティやユーザが作成したユーティリティの中には、フォーリン・テープ・セット内の最初のテープだけをマウントするものがあります。新しいバージョンの OpenVMS との互換性をとるには、セット内の各リールについて \$MOUNT と \$DISMOU のシステム・サービスを明示的に呼び出すよう、これらのユーティリティを変更します。これらのユーティリティで使用する磁気テープ・セットを/MULTI\_VOLUME 修飾子でマウントする方法もあります。

/MULTI\_VOLUME 修飾子は/FOREIGN 修飾子と併用する必要があります。また、ユーザ特権 VOLPRO が必要です。省略時の設定では、/NOMULTI\_VOLUME です。

---

#### 注意

---

OpenVMS Backup ユーティリティ (BACKUP) は、フォーリン・マウントされた磁気テープ・セットの各リールで\$MOUNT システム・サービスと\$DISMOU システム・サービスを明示的に呼び出します。詳しくは『OpenVMS システム管理ユーティリティ・リファレンス・マニュアル(上巻)』の BACKUP および複数のボリューム・セーブ・セットに関する節を参照してください。

---

#### 例

以下のコマンドは、テープ・ボリューム・セットをマウントしています。最初のボリュームにアクセス・チェックを行います。第 2 以降のリールに対しては、チェックを行わずに処理を進めます。

```
$ MOUNT/FOREIGN/MULTI_VOLUME MUA0:
```

```
/OVERRIDE=(キーワード[,...])
```

MOUNT コマンドによる 1 つまたは複数の保護チェックを禁止します。

/FOREIGN 修飾子と/OVERRIDE=(ACCESSIBILITY, EXPIRATION) を指定するには、ユーザ特権 OPER と VOLPRO が必要です。これらの特権がない場合、磁気テープは読み込まれません。

複数のキーワードを指定する場合、キーワードをコンマで区切り、リスト全体を括弧で囲みます。

次の表は、この修飾子のためのキーワードの一覧です。

| キーワード         | 説明                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ACCESSIBILITY | <p>磁気テープ専用。ボリュームのアクセス可能フィールドに定義されている文字を無効にします。このキーワードを使用するかどうかは、各システムで決定します。つまり、このフィールドの処理に磁気テープ・ファイル・システムが使用するルーチンを、システムごとに指定できます。省略時の設定では、このフィールドを次のようにチェックするルーチンが、OpenVMS オペレーティング・システムに用意されています。</p> <ul style="list-style-type: none"> <li>ANSI のバージョン 3 に準拠する OpenVMS で磁気テープを作成した場合 ASCII 空白文字以外のすべての文字をこのキーワードで無効にしてください。</li> <li>OpenVMS 保護が指定されており、かつバージョン 3 より後の ANSI 標準に磁気テープが準拠する場合 ASCII の 1 以外のすべての文字をこのキーワードで無効にしてください。</li> </ul> |

キーワード ACCESSIBILITY を使用するには、ユーザ特権 VOLPRO を持っているか、またはボリュームの所有者でなければなりません。

| キーワード             | 説明                                                                                                                                                                                                                                                                        |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| EXPIRATION        | ボリュームとそのファイルの満了日付を無効にします。満了日付を変更したいファイルの最初のファイル・ヘッダ・ラベルに定義されているデータが、満了日付にまだ達していない場合に使用できます。ユーザ特権 VOLPRO を持っているか、またはボリュームの UIC と同じ UIC を持っている必要があります。                                                                                                                      |
| IDENTIFICATION    | ボリューム識別子の処理を無効にします。ラベルが不明であるボリュームをマウントする場合や、ラベルの最初の 12 文字が一意でない ISO 9660 ボリュームをマウントする場合 (VAX システム) に使用します。無効となるのは、ボリューム識別子フィールドだけであり、ボリューム保護は変更されません。ボリュームは、明示的にまたは省略時の設定として、/NOSHARE でマウントしてください。<br><br>/OVERRIDE=IDENTIFICATION 修飾子は/GROUP および/SYSTEM 修飾子と互換性がありません。   |
| LIMITED_SEARCH    | 予想される位置からホーム・ブロックを検索できないときに、Mount コマンドがデバイス全体からホーム・ブロックを検索することを許可します。省略時の設定では、ホーム・ブロックの検索は制限され、有効なホーム・ブロックが存在しないときに、余分な検索時間を費やさないようにしています。                                                                                                                                |
| LOCK              | マウント時にエラーが発生したため、ボリュームをライト・ロックしないことを指示します。破損したボリュームを ANALYZE/DISK_STRUCTURE コマンドで処理するためマウントする場合に使用します。キーワード LOCK を使用するには、ユーザ特権 VOLPRO を持っているか、またはボリュームの所有者でなければなりません。                                                                                                     |
| NO_FORCED_ERROR   | 装置またはコントローラが強制エラー処理をサポートしていない場合でも、シャドウイングを続行することを支持します。サポートされていない SCSI ディスクを使用すると、修正できないエラーが発生した場合にシャドウ・セットからメンバが削除されることがあります。これは、SCSI ディスクではディスクの不良ブロックを修復する READL コマンドおよび WRITEL コマンドを使用できないことがあるためです。                                                                  |
| OWNER_IDENTIFIER  | 磁気テープ専用。所有者識別子フィールドの処理を無効にします。保護された磁気テープを OpenVMS と別の HP のオペレーティング・システムとの間で交換する場合に使用します。                                                                                                                                                                                  |
| SECURITY          | ボリュームに無効の SECURITY.SYS ファイルが登録されているために、エラーが戻された場合、ボリュームのマウントを継続することを許可します。このキーワードを使用するには、VOLPRO ユーザ特権を持つか、またはボリュームを所有していなければなりません。                                                                                                                                        |
| SETID             | 磁気テープ専用。継続ボリューム上の最初のファイルの最初のファイル・ヘッダ・ラベルのファイル・セット識別子のチェックを禁止します。ANSI 継続ボリューム上の最初のファイルのファイル・セット識別子が、マウントした最初のボリュームの最初のファイルのファイル・セット識別子と異なる場合に使用します。                                                                                                                        |
| SHADOW_MEMBERSHIP | 以前のシャドウ・セット・メンバの書き込み保護を無効にすることを許可します。このパラメータを使用できるのは、ボリューム・シャドウイング・オプションが用意されている場合だけです。『Volume Shadowing for OpenVMS 説明書』を参照してください。<br><br>この修飾子でボリュームをマウントすると、ボリューム・シャドウイング作成番号が消去されます。シャドウ・セット内のボリュームを再マウントしようとする、関連しないボリュームであるとみなされ、現在のシャドウ・セット・メンバの中の 1 つがフル・コピーされます。 |

以下のコマンドはボリューム識別フィールドを無効にし、MFA0 上の磁気テープをラベル指定なしでマウントしています。

```
$ MOUNT/OVERRIDE=IDENTIFICATION MFA0:
```

/OWNER\_UIC=uic

ボリュームをマウントするときに、ボリュームの所有権を指定した UIC に対して設定し、ボリュームに記録されている所有権を無効にすることを指示します。

/FOREIGN 所有権でボリュームをマウントする場合は、現在の使用中 UIC 以外の所有者 UIC を要求します。

パラメータ、UIC(ユーザ識別コード)を次の形式で指定します。

[グループ, メンバ]

UIC 指定は、大括弧で囲みます。グループ番号は、0 ~ 37776 の 8 進値です。メンバー番号は、0 ~ 17776 の 8 進値です。

Files-11 ボリュームに対して /OWNER\_UIC 修飾子を使用するには、ユーザ特権 VOLPRO を持っているか、またはボリュームの UIC と同じ UIC を持っている必要があります。

例

以下のコマンドは WORK というラベルのディスク装置を DRA3 にマウントし、所有者 UIC [016,360] を設定しています。

```
$ MOUNT/OWNER_UIC=[016,360] DRA3: WORK
```

/POLICY=[NO]MINICOPY[=(OPTIONAL)], REQUIRE\_MEMBERS, [NO]VERIFY\_LABEL

シャドウ・セットのセットアップと使用を制御します。ボリューム・シャドウイングの詳細については、『Volume Shadowing for OpenVMS 説明書』を参照してください。

次の表に、この修飾子のキーワードを示します。



| キーワード                                          | 説明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [NO]MINICOPY<br>[=OPTIONAL] (Alpha<br>/I64 のみ) | <p>シャドウイング・ミニコピー機能のセットアップと使用を制御します。</p> <p>ビットマップを作成するには、LOG_IO (論理 I/O) 特権が必要です。</p> <p>MOUNT /POLICY 修飾子のキーワード [NO]MINICOPY[=OPTIONAL] の意味は、次のようにシャドウ・セットの状態に依存します。</p> <ol style="list-style-type: none"> <li>1. スタンドアロン・システムが任意のクラスタ・メンバ上で、シャドウ・セットがマウントされておらず、MINICOPY=OPTIONAL が指定されている場合には、シャドウ・セットがマウントされ、書き込みビットマップが作成されます。書き込みビットマップはシャドウイング・ミニコピー操作を可能にします。シャドウイング・ミニコピー操作を可能にするためには、スタンドアロン・システム上かクラスタ内で、シャドウ・セットの最初のマウントで/MOUNT /POLICY=MINICOPY[=OPTIONAL]を指定する必要があります。</li> </ol> <p>OPTIONAL キーワードを指定すると、システムが書き込みビットマップを開始できなかった場合でもマウントを続行することができます。ビットマップが開始できない理由としては、シャドウ・セットが正しくデスマウントされなかった、シャドウ・セットがマージ操作を必要としている場合や、その他のさまざまなリソース関連の問題が考えられます。OPTIONAL キーワードが省略され、システムが書き込みビットマップを開始できなかった場合には、シャドウ・セットはマウントされません。</p> <p>/POLICY=MINICOPY=OPTIONAL 修飾子を指定しており、シャドウ・セットがすでにクラスタ内の別のノード上で/POLICY=MINICOPY[=OPTIONAL]なしでマウントされていた場合、MOUNT コマンドは成功しますが、書き込みビットマップは作成されません。</p> <p>NOMINICOPY が指定された場合、シャドウ・セットはマウントされますが、書き込みビットマップは作成されません。</p> <ol style="list-style-type: none"> <li>2. シャドウ・セットの前のメンバが、ミニコピーが有効になっているシャドウ・セットに戻された場合には、フル・コピーの代わりにミニコピーが開始されます。これは省略時の動作であり、/POLICY=MINICOPY[=OPTIONAL]を省略した場合でも行われます。ミニコピーの開始が成功しても、何らかの理由で実行に失敗した場合には、フル・コピーが実行されます。</li> </ol> <p>ミニコピーが開始できなかった場合、キーワード OPTIONAL が省略されていれば、マウントは失敗します。</p> <p>NOMINICOPY が指定された場合には、ミニコピーはたとえ可能であっても実行されません。</p> |
| REQUIRE_MEMBERS                                | <p>MOUNT コマンドが実行される条件として、MOUNT コマンドが発行されたときに、/SHADOW 修飾子で指定されたすべての物理デバイスがアクセス可能でなければならないかどうかを制御します。メンバはコマンド行で指定されるか、/INCLUDE 修飾子によってディスク上で検索されます。</p> <p>この修飾子を指定しなかった場合、(接続障害など) 何らかの理由で 1 つ以上のメンバにアクセスできないときには、アクセス可能なメンバとの仮想ユニットが作成されます。</p> <p>このオプションを使用すると、イベント後に正しいメンバシップが選択されるので、耐障害クラスタでリカバリを行うときに特に有効です。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| [NO]VERIFY_LABEL                               | <p>シャドウ・セットに追加されるすべてのメンバが、'SCRATCH_DISK' というボリューム・ラベルを持たなければならないようにします。</p> <p>これにより、間違ったディスクが誤ってシャドウ・セットに追加されるのを防ぐことができます。VERIFY_LABEL を使用する場合には、セットに追加されるディスクをラベル 'SCRATCH_DISK' で初期化するか、SET VOLUME/LABEL を実行する必要があります。</p> <p>省略時の動作は NOVERIFY_LABEL で、コピー・ターゲットのボリューム・ラベルはチェックされません。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

/PROCESSOR=キーワード

磁気テープとFiles-11構造レベル1ディスクの場合に、補助制御プロセス (ACP) をボリューム処理に使用することを指定します。/PROCESSOR 修飾子により、ACP を装置と対応づける省略時の方式を無効にします。

Files-11構造レベル2および5のディスクの場合は、ブロック・キャッシュの割り当てを制御します。

次の表は、この修飾子のキーワードの一覧です。

| キーワード    | 説明                                                                                                                                                                                                      |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UNIQUE   | 磁気テープ、Files-11 ODS-1、ISO 9660、High Sierra でフォーマットされた媒体のいずれかをマウントするときに、これらをサポートする省略時の補助制御プロセス (ACP) イメージを実行するプロセスを作成します。<br>Files-11構造レベル2および5のディスクの場合は、独立したブロック・キャッシュを割り当てます。                           |
| SAME: 装置 | 磁気テープ、Files-11 ODS-1、ISO 9660、High Sierra でフォーマットされた媒体のいずれかをマウントするときに、これらをサポートする ACP イメージを実行するプロセスとして、既存プロセスを使用します。<br>Files-11構造レベル2および5のディスクの場合は、指定した装置に割り当てられているブロック・キャッシュを取り外します。                   |
| ファイル指定   | ファイル指定で指定した ACP イメージ (変更を施した ACP やユーザが作成した ACP など) を実行するプロセスを作成します。ワイルドカード文字、ノード名、ディレクトリ名は、ファイル指定に使用できません。<br>このキーワードを使用するためには、CMKRNL と OPER の特権が必要です。<br>/PROCESSOR 修飾子を使用するには、オペレータ・ユーザ特権 OPER が必要です。 |

例

以下のコマンドは、現在 MTA1 に設定されている ACP プロセスを使用して MFA0 に磁気テープをマウントしています。

```
$ MOUNT/PROCESSOR=SAME:MTA1: MFA0:
```

/PROTECTION=キーワード

ボリュームに設定する保護コードを指定します。

次の表は、この修飾子のキーワードを記述します。

| キーワード | 説明                                                                                                                                          |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------|
| 保護コード | ユーザ保護指定の標準構文規則 (システム/所有者/グループ/ワールド) に準拠した保護コードを指定します。保護カテゴリを省略すると、そのカテゴリのユーザにはすべてのアクセスが禁止されます。<br>保護コードを指定しない場合、初期化時にボリュームに設定された保護が使用されません。 |
| XAR   | 拡張レコード属性 (XAR) アクセス制御の適用を許可します。XAR についての詳しい説明は『OpenVMS Record Management Services Reference Manual』を参照してください。                               |

| キーワード | 説明                                                                                                                                      |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------|
| DSI   | DEC システム識別子 (DSI) を格納した XAR に対して XAR Owner アクセスおよび Group アクセスを許可します。詳しくは『OpenVMS Record Management Services Reference Manual』を参照してください。 |

/SYSTEM または /GROUP でボリュームをマウントするときに /PROTECTION 修飾子を指定した場合、他の修飾子によるすべてのアクセス権は無効となります。

/FOREIGN 修飾子を指定した場合、実行 (E) アクセス権または作成 (C) アクセス権と削除 (D) アクセス権は、論理入出力 (L) の場合と物理入出力 (P) の場合で同義です。しかし、物理入出力 (P) と論理入出力 (L) のどちらか一方または両方のアクセス・コードを指定すれば、複数のユーザ・カテゴリが実行できる入出力操作を制限できます。

Files-11 ボリュームに対して /PROTECTION 修飾子を使用するには、ユーザ特権 VOLPRO を持っているか、ボリュームの UIC と同じ UIC を持っている必要があります。

#### 例

以下のコマンドは、WORKDISK というラベルの装置を DKA1 にマウントし、保護コードを設定しています。このボリュームへのアクセスは、SYSTEM ユーザの場合 READ, WRITE, CREATE, OWNER の場合 READ, WRITE, CREATE, DELETE, GROUP ユーザの場合 READ, CREATE, WORLD カテゴリの場合 READ だけです。

```
$ MOUNT/PROTECTION=(SYSTEM:RWE,O:RWED,G:RE,W:R) DKA1: WORKDISK
```

#### /QUOTA (省略時の設定)

##### /NOQUOTA

指定したディスク・ボリュームにクォータを実施するかどうかを制御します。

省略時の設定では、各ユーザに対してクォータを実施する /QUOTA です。/NOQUOTA は、このチェックを禁止します。/QUOTA 修飾子を指定するには、ユーザ特権 VOLPRO を持っているか、またはボリュームの UIC と同じ UIC を持っている必要があります。

#### 例

以下のコマンドは、DRA3 にマウントされている、WORK というラベルのディスク・ボリュームに対し、所有者 UIC [016,360] を設定し、クォータを実施しないことを指定しています。

```
$ MOUNT/OWNER_UIC=[016,360]/NOQUOTA DRA3: WORK
```

#### /REBUILD (省略時の設定)

##### /NOREBUILD

ディスク・ボリュームに対し、リビルド動作を行うかどうかを制御します。

システム障害などのためディスク・ボリュームが正しくディスマウントされていない場合、ボリュームをリビルドして、ディスマウント時に許可されていたキャッシング制限を回復する必要があります。省略時の設定では、リビルドが行われます。使用可能なすべての空き空間の再利用も行う正常なリビルドを実行するには、すべてのボリューム・セット・メンバをマウントする必要があります。

ボリューム上のファイル数によっては、リビルドにかなりの時間がかかります。クォータを使用している場合のファイル所有者の数も、リビルドにかかる時間に影響します。

ボリュームをディスマウントする前に許可されていたキャッシュとしては、以下のキャッシュが考えられます。

- 割り当て済み空き空間 (EXTENT キャッシュ)
- 割り当て済みファイル番号 (FILE\_ID キャッシュ)
- ディスク・クォータ使用量キャッシング (QUOTA キャッシュ)

割り当て済み空き空間やファイル番号のキャッシングが許可されていた場合、最も多くのファイルがボリュームに存在したときのファイル数にリビルド時間が比例します。ディスク・クォータ・キャッシングが許可されていた場合、ディスク・クォータ・ファイルのエントリ数の二乗に比例する時間がかかると思われます。

いずれのキャッシングも許可されていなかった場合、リビルドは不要であるため実行されません。

/NOREBUILD 修飾子を使用すると、ただちに装置をアクティブ状態に戻せます。リビルドを改めて実行する場合は、DCL の SET VOLUME/REBUILD コマンドを使用します。

システム・ディスクのリビルド方法については、『OpenVMS システム管理者マニュアル』を参照してください。

#### 例

この例では、WORKDISK というボリュームが NODE\$DKA2 にマウントされています。このボリュームは正しくディスマウントされておらず、また/REBUILD 修飾子が指定されているので、メッセージが出力され、ボリュームがリビルドされます。

```
$ MOUNT/REBUILD NODE$DKA2: WORKDISK
%MOUNT-I-MOUNTED, WORKDISK mounted on _NODE$DKA2:
%MOUNT-I-REBUILD, volume was improperly dismounted; rebuild in
progress
```

ボリューム WORKDISK が正しくディスマウントされていませんが、/NOREBUILD 修飾子が指定されているので、リビルドは実行されません。ただし、リビルドが必要であることを示すメッセージが表示され、WORKDISK をこのままの状態で使用可能にしようとしています。DCL の SET VOLUME/REBUILD コマンドを使用すれば、リビルドを行えます。

```
$ MOUNT/NOREBUILD NODE$DKA2: WORKDISK
%MOUNT-I-MOUNTED, WORKDISK mounted on _NODE$DKA2:
%MOUNT-I-REBLDREQD, rebuild not performed; some free space
unavailable; diskquota usage stale
```

/RECORDSIZE=*n*

磁気テープ・ボリュームの各レコードに格納する文字数を指定します。

パラメータ, *n*は, OpenVMS RMS の場合は 20 ~ 65,532 バイトのブロック・サイズ, OpenVMS RMS 以外を使用している場合は 18 ~ 65,534 バイトのブロック・サイズを指定します。

この修飾子は, 通常/FOREIGN, /BLOCKSIZE 修飾子と併用し, ブロック構造装置に固定長レコードを読み込みまたは書き込みする場合に指定します。レコード・サイズは, 指定したブロック・サイズ以下または省略時のブロック・サイズ以下でなければなりません。

RT-11など HDR2 ラベルのない磁気テープをマウントする場合, /RECORDSIZE 修飾子を使用して, 省略時の最大レコード・サイズを指定します。

例

次の例は, 省略時のブロック・サイズと省略時のレコード・サイズが 512 文字の磁気テープを MTA0 にマウントしています。

```
$ MOUNT/FOREIGN/BLOCKSIZE=512/RECORDSIZE=512 MTA0:
```

/SHADOW

3 つの物理デバイスを 1 つのシャドウ・セットに結合します。シャドウ・セットは, コマンドに指定した仮想ユニット名によって指定されます。この修飾子を使用できるのは, ボリューム・シャドウイング・オプションが用意されている場合だけです。詳細情報については, 『Volume Shadowing for OpenVMS 説明書』を参照してください。

この修飾子の形式は次のとおりです。

```
(virtual-unit-name[:] /SHADOW=(physical-device-name[:][,...]))
```

この修飾子は, 物理装置とこれらの物理装置を表す仮想ユニットを含むシャドウ・セットをマウントしていることを示します。この修飾子は, 仮想ユニット名をデバイス名パラメータとしてとることを MOUNT に指示します。/SHADOW 修飾子は, 仮想ユニット名パラメータの後に入力します。

仮想ユニット名の形式は DSA $n$  です。 $n$ は 0 ~ 9999 の範囲の固有の数字です。*physical-device-name*に対しては, 標準的なデバイス名形式 \$allocation-class\$ddcu[:] を使用してください。

## 例

次の例は、シャドウ・セットを作成しています。2つのシャドウ・セット・メンバのコピー動作は、ソフトウェアが自動的に正しく決定します。\$1\$DUA10は現在のボリュームなので、\$1\$DUA10が\$1\$DUA11にコピーされます。

```
$ MOUNT DSA0: /SHADOW=(1DUA10:,1DUA11:) SHADOWVOL
%MOUNT-I-MOUNTED, SHADOWVOL mounted on DSA0:
%MOUNT-I-SHDWMEMSUC, _1DUA10: (MEMBER1) is now a valid member of
the shadow set
%MOUNT-I-SHDWMEMCOPY, _1DUA11: (MEMBER2) added to the shadow set
with a copy operation
```

以下のコマンドは、TEST3013という論理名のボリューム・セットを作成します。ボリューム・セットTEST3013はシャドウ化されませんが、ボリューム・セット(TEST3011とTEST3012)の各構成要素はシャドウセットであり、ボリューム・セット全体に冗長性を持たせます。

```
$ MOUNT/BIND=TEST3013 DSA3011/SHADOW=(1DUA402:,1DUA403:),
DSA3012/SHADOW=(1DUA404:,1DUA405:) TEST3011,TEST3012 TEST3013
```

```
/SHARE
/NOSHARE
```

ディスク・ボリュームが共用可能であることを指定します。

別のユーザがすでにマウントした共用ボリュームを/SHARE修飾子でマウントする場合、他の修飾子を指定してもすべて無視されます。

省略時の設定では、ボリュームは共用可能ではなく、マウント先の装置が割り当てられます。

以前に割り当てていた装置に/SHARE修飾子を指定した場合、この他のユーザがアクセスできるようにするため、この装置の割り当てが解除されます。

/SHARE修飾子は/GROUPおよび/SYSTEM修飾子とは互換性がありません。

## 例

以下のコマンドはラベルSLIPの装置をDLA0にマウントし、MOUNTメッセージの同報通信を禁止し、共用可能ボリュームであることを指定し、論理名DISCを指定しています。

```
$ MOUNT/NOMESSAGE/SHARE DLA0: SLIP DISC
```

```
/SUBSYSTEM
/NOSUBSYSTEM
```

保護されたサブシステムと、システムACEの処理を許可します。SECURITY特権が必要です。

省略時の設定では、ブートで使用するディスクに対して/SUBSYSTEMが許可され、他のディスクに対しては許可されません。サブシステムについての詳しい説明は、『OpenVMS システム・セキュリティ・ガイド』を参照してください。

## 例

以下のコマンドでは、マウント・メッセージを禁止した状態で DUA1 に SLIP というラベルのボリュームをマウントします。ボリューム上のサブシステムをアクセスできます。MOUNT はまた、論理名として SACH も割り当てます。

```
$ MOUNT/NOMESSAGE/SUBSYSTEM DUA1: SLIP SACH
```

## /SYSTEM

ボリュームを公用ボリュームにします。つまり、UIC に基づくボリューム保護によってアクセスが許可されているかぎり、システムのすべてのユーザがボリュームを利用できるようにします。

装置の論理名は、システム論理名テーブルに格納されます。/SYSTEM 修飾子を使用するには、ユーザ特権 SYSNAM が必要です。

/SYSTEM 修飾子でボリュームを OpenVMS Cluster にマウントする場合、クラスタ単位でマウントしない場合でも、クラスタ単位で一意であるボリューム・ラベルを使用する必要があります。

/SYSTEM 修飾子は /GROUP、/OVERRIDE=IDENTIFICATION、/SHARE 修飾子とは互換性がありません。

## 例

以下のコマンドは SLIP というラベルのボリュームを DUA1 にマウントし、マウント・メッセージを禁止しています。このボリュームは、システム単位で利用できます。SACH という論理名も設定しています。

```
$ MOUNT/NOMESSAGE/SYSTEM DUA1: SLIP SACH
```

PAYVOL1、PAYVOL2、PAYVOL3 のラベルを持つ初期化済みボリュームで構成されるボリューム・セット MASTER\_PAY を作成しています。これらのボリュームは、それぞれ DB1、DB2、DB3 という装置に物理的にマウントされています。PAYVOL1 は、このセットのルート・ボリュームです。

すべてのユーザが利用できるようにするため、システム・ボリュームとしてマウントされています。

```
$ MOUNT/SYSTEM/BIND=MASTER_PAY -
_$ DB1:,DB2:,DB3: PAYVOL1,PAYVOL2,PAYVOL3
```

## /UCS\_SEQUENCE=escape\_sequence

コード化グラフィック文字セットを選択するためのエスケープ・シーケンスを指定します。これは補助ボリューム記述子 (SVD) の 1 つで ISO 9660 ボリュームをマウントするとき必要となります。

パラメータ *escape\_sequence* は、CD-ROM を製造したベンダが定義する文字シーケンスです。これは、ベンダの文字セット変換テーブルの中で一意な文字シーケンスです。

/UCS\_SEQUENCE 修飾子は、OpenVMS で非 ASCII 文字セットを含んでいる ISO 9660 CD-ROM をマウントするときに使用します。

ISO 9660 ボリュームは、グラフィック文字セットを指定する SVD を含んでいることがあります。このグラフィック文字は、マウント時に選択された場合、ボリュームのディレクトリとファイル名を表示するときの省略時の文字として使用されます。

/UCS\_SEQUENCE 修飾子は、コード化グラフィック文字セットを選択するためのエスケープ・シーケンスを定義します。

すべての ISO 9660 ボリュームは、文字セットとして ASCII (ISO 646-IRV) を使用する主ボリューム記述子 (PVD) を含んでいます。ISO 9660 と OpenVMS のファイル名規則は、どちらも、ボリュームのディレクトリとファイル名を表示するときに、同じ ASCII 文字のサブセットを使用します。

/UNDEFINED\_FAT=レコード形式:[レコード属性]:[レコードサイズ]

レコード形式が指定されていない ISO 9660 メディアのレコードに対して使用する省略時のファイル属性を設定します。

次の表は、パラメータの説明です。

| パラメータ    | 説明                                                                                                                                                                                                                         |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| レコード形式   | ファイル内のすべてのレコードの形式を指定します。指定できる形式は FIXED, VARIABLE, STREAM, STREAM_LF, STREAM_CR, LSB_VARIABLE, MSB_VARIABLE です。これらのレコード形式についての説明は、『OpenVMS Record Management Services Reference Manual』の RMS フィールド FAB\$B_RFM の説明を参照してください。 |
| レコード属性   | ファイル内のすべてのレコードの属性を指定します。指定できる属性は NONE, CR, FTN, PRN, NOBKS です。このパラメータは STREAM 以外のレコード形式にだけ適用されます。これらのレコード属性についての説明は、『OpenVMS Record Management Services Reference Manual』の RMS フィールド FAB\$B_RAT の説明を参照してください。              |
| レコード・サイズ | ファイル内のすべてのレコードの最大レコード・サイズを指定します。指定できる値は 0 ~ 32767 です。このパラメータは FIXED または STREAM レコード形式にだけ有効です。可能な RMS レコード・サイズについての説明は、『OpenVMS Record Management Services Reference Manual』の RMS フィールド FAB\$W_MRS の説明を参照してください。            |

ISO 9660 メディアは、レコード形式が前もって定義されているファイルをサポートしないプラットフォームから作成される可能性があります。/UNDEFINED\_FAT 修飾子は、レコード形式が指定されていない ISO 9660 メディアのレコードに対して使用される省略時のファイル属性を設定します。

/UNDEFINED\_FAT 修飾子は/MEDIA\_FORMAT=CDROM 修飾子と組み合わせて使用しなければなりません。

この修飾子はすべての未定義ファイル・タイプを一時的に無効にし、次に示すように、選択可能なレコード属性と選択可能なレコード・サイズを持つ選択可能なレコード形式に変更します。



|        |                                                                                                                                                                                                                                                      |
|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| レコード形式 | $\left\{ \begin{array}{l} \text{FIXED: レコード属性[,...]: レコード・サイズ} \\ \text{VARIABLE: レコード属性[,...]} \\ \text{STREAM: レコード・サイズ} \\ \text{STREAM\_LF: レコード・サイズ} \\ \text{STREAM\_CR: レコード・サイズ} \\ \text{LSB\_VARIABLE: レコード属性[,...]} \end{array} \right\}$ |
| レコード属性 | $\left\{ \begin{array}{l} \text{NONE - None} \\ \text{CR - Carriage\_return} \\ \text{FTN - Fortran} \\ \text{PRN - Print} \\ \text{NOBKS - No-Block-Span} \end{array} \right\}$                                                                     |

レコード・サイズ{ 1 ~ 32767 }

#### 例

次の例では，OFFENS というラベルのボリュームがDKA1 にマウントされ，ボリュームのすべてのファイルは固定長，キャリッジ・リターン付き，80 バイトの長さのレコードとして定義されます。MOUNT はまた，STRAT という論理名も割り当てます。

```
$ MOUNT/MEDIA_FORMAT=CDROM/UNDEFINED_FAT=(FIXED:CR:80) DKA1: OFFENS STRAT
```

/UNLOAD (省略時の設定)

/NOUNLOAD

MOUNT コマンドで指定したディスク・ボリュームまたは磁気テープ・ボリュームを，ディスマウント時にアンロードするかどうかを制御します。

#### 例

次の例は，OFFENS というラベルのボリュームを/NOUNLOAD 修飾子を使用してDKA1 にマウントしています。この結果，このボリュームは，物理的にアンロードせずにディスマウントできます。STRAT という論理名も設定しています。

```
$ MOUNT/NOUNLOAD DKA1: OFFENS STRAT
```

/WINDOWS=n

ファイル・ウィンドウに対して割り当てるマッピング・ポインタの数を指定します。

パラメータ，*n*は，7 ~ 80 の値を指定します。この指定により，ボリューム初期化時に指定した省略時の値は無効となります。

ファイルがオープンしたときに，ファイル・システムは，マッピング・ポインタを使用してファイル内のデータにアクセスします。MOUNT/WINDOWS を使用すると，ボリューム初期化時に指定した省略時の値が無効となります。ボリューム初期化時に値を指定しなかった場合，省略時のマッピング・ポインタ数は7です。

/WINDOWS 修飾子を使用するには，オペレータ・ユーザ特権 OPER が必要です。

## 例

以下のコマンドは DKA2 上の GONWITH というラベルのボリュームをシステム単位で使用可能とし、THE\_WINDOW という論理名を設定しています。/WINDOWS 修飾子に 25 が指定されているので、省略時のマッピング・ポインタ数は無効となっています。

```
$ MOUNT/SYSTEM/WINDOWS=25 DKA2: GONWITH THE_WINDOW
```

/WRITE (省略時の設定)

/NOWRITE

ボリュームが書き込み可能であるかどうかを制御します。

省略時の設定では、マウント時に読み込みおよび書き込みアクセスが許可されます。/NOWRITE を指定すれば、読み込み専用アクセス権を設定してファイルを保護することができます。この効果は、装置をライト・ロックすることと同じです。

ホスト・ベースのボリューム・シャドウイング装置の場合は、ほかにも留意点があります。詳細は、『Volume Shadowing for OpenVMS 説明書』を参照してください。

## 例

以下のコマンドは BOOKS というラベルのボリュームを NODE\$DKA1 にマウントし、OpenVMS Cluster 内の各ノードにも次々とマウントしています。/NOWRITE 修飾子により、読み込み専用アクセス権が設定されます。

```
$ MOUNT/CLUSTER/NOWRITE NODE$DKA1: BOOKS
```

---

 例

例 1, 2 では、オペレータ補助が不要であり、ボリュームがドライブに格納されていることを仮定しています。例 3 ~ 6 は、オペレータ補助を伴うマウントです。例 7, 8 では、ISO 9660 CD-ROM ボリュームをマウントしています。例 9 では、ボリュームのサブシステムをアクセス可能にしておき、例 10 では、シャドウ・セットをマウントしています。

1. 

```
$ MOUNT MTA0: MATH06 STAT_TAPE
%MOUNT-I-MOUNTED, MATH06 mounted on _MTA0:
$ COPY ST061178.DAT STAT_TAPE:
```

ボリューム・ラベルが MATH06 の磁気テープを MOUNT コマンドが装置 MTA0 にマウントし、STAT\_TAPE という論理名を設定します。

次に、COPY コマンドが ST061178.DAT というディスク・ファイルを磁気テープにコピーします。

2. \$ ALLOCATE DM:  
 %DCL-I-ALLOC, \_DMB2: allocated  
 \$ MOUNT DMB2: TEST\_FILES  
 %MOUNT-I-MOUNTED, TEST\_FILES mounted on \_DMB2:

RK06/RK07 という使用可能な装置を ALLOCATE コマンドが要求します。ALLOCATE コマンドの結果が出力された後、割り当てられた装置に物理ボリュームを格納します。次に、MOUNT コマンドがボリュームをマウントします。

3. \$ MOUNT DM: TEST\_FILES  
 %MOUNT-I-OPRQST, Please mount volume TEST\_FILES in device \_DMB2:  
 %MOUNT-I-MOUNTED, TEST\_FILES mounted on \_DMB2:

この例の結果は、上記の例と同じです。TEST\_FILES というラベルのボリュームに使用する装置として、使用可能な RK06/RK07 を MOUNT コマンドが要求します。MOUNT が出力した装置にボリュームを物理的にマウントすると、マウント動作が終了します。装置は、MOUNT が自動的に割り当てます。

4. \$ MOUNT DYA1: TESTSYS  
 %MOUNT-I-OPRQST, Please mount volume TESTSYS in device DYA1:  
Ctrl/Y  
 \$ EXIT  
 %MOUNT-I-OPRQSTCAN, operator request canceled

TESTSYS というボリュームを装置 DYA1 にマウントすることを、MOUNT コマンドがオペレータに指示します。ここでは、Ctrl/Y を押してマウントを取り消しています。マウント要求を実際に取り消す前にイメージを終了する必要があるので、EXIT コマンドでイメージを終了しています。ただし、コマンド・インタプリタ内で実行されないすべてのコマンドは、現在のイメージを終了します。

5. \$ MOUNT DYA1: TESTSYS  
 %MOUNT-I-OPRQST, Device \_DYA1: is not available for mounting.  
 %MOUNT-I-OPRQSTCAN, operator request canceled  
 %MOUNT-I-OPRQST, Please mount volume TESTSYS in device \_DYA1:  
 %MOUNT-I-MOUNTED, TESTSYS mounted on \_DYA1:  
 %MOUNT-I-OPRQSTDON, operator request canceled - mount completed successfully

ボリューム TESTSYS を装置 DYA1 にマウントすることを、MOUNT コマンドがオペレータに指示します。DYA1 は別のユーザに割り当てられているので、マウントできません。この場合、装置が使用可能となるのを待つか、別の装置に変更するか、マウントをアボートします。ここでは、オペレータ補助マウントのまま、装置を使用しているプロセスが装置を割り当て解除することを待っています。

装置は使用可能であってもボリュームがマウントされていないので、最初のマウント要求が取り消され、TESTSYS のマウント要求を改めて起動します。オペレータがドライブにボリュームを装填した後、MOUNT はマウントをリトライします。マウントが終了すると、要求は取り消されます。

6. \$ MOUNT DYAl: TESTSYS/COMMENT="Is there an operator around?"  
 %MOUNT-I-OPRQST, Please mount volume TESTSYS in device \_DYAl:  
 Is there an operator around?  
 %MOUNT-I-NOOPR, no operator available to service request  
 .  
 .  
 .  
 %MOUNT-I-MOUNTED, TESTSYS mounted on \_DYAl:  
 %MOUNT-I-OPRQSTDON, operator request canceled - mount  
 completed successfully

ボリューム TESTSYS を装置 DYAl にマウントすることを、オペレータに指示します。ところが、オペレータがいません。この場合、ユーザは、Ctrl/Y を押してマウントをアボートするか、またはオペレータを待ちます。ここでは、オペレータを待っています。

7. \$ MOUNT/SYSTEM/MEDIA=CDROM \$1\$DKA1 USER  
 %MOUNT-I-CDROM\_ISO, USER:VMS\_ONLINE\_DOCUMENTATION (1 of 4) ,  
 mounted on \_\$1\$DKA1: (CDROM)  
  
 \$ MOUNT/SYSTEM/MEDIA=CDROM \$1\$DKA2 PROGRAMMING\_1  
 %MOUNT-I-CDROM\_ISO, PROGRAMMING\_1:VMS\_ONLINE\_DOCUMENTATION (2 of 4) ,  
 mounted on \_\$1\$DKA2: (CDROM)  
  
 \$ MOUNT/SYSTEM/MEDIA=CDROM \$1\$DKA3 PROGRAMMING\_2  
 %MOUNT-I-CDROM\_ISO, PROGRAMMING\_2:VMS\_ONLINE\_DOCUMENTATION (3 of 4) ,  
 mounted on \_\$1\$DKA3: (CDROM)  
  
 MOUNT/SYSTEM/MEDIA=CDROM \$1\$DKA4 MANAGEMENT  
 %MOUNT-I-CDROM\_ISO, MANAGEMENT:VMS\_ONLINE\_DOCUMENTATION (4 of 4) ,  
 mounted on \_\$1\$DKA4: (CDROM)

"VMS\_ONLINE\_DOCUMENTATION"という名前の ISO 9660 ボリューム・セットのメンバ 4 つをマウントします。

8. \$ MOUNT/SYSTEM/MEDIA=CDROM \$1\$DKA1,\$1\$DKA2,\$1\$DKA3,\$1\$DKA4  
 USER,PROGRAMMING\_1,PROGRAMMING\_2,MANAGEMENT  
 %MOUNT-I-CDROM\_ISO, USER:VMS\_ONLINE\_DOCUMENTATION (1 of 4) , mounted on  
 \_\$1\$DKA1: (CDROM)  
 %MOUNT-I-CDROM\_ISO, PROGRAMMING\_1:VMS\_ONLINE\_DOCUMENTATION (2 of 4) ,  
 mounted on \_\$1\$DKA2: (CDROM)  
 %MOUNT-I-CDROM\_ISO, PROGRAMMING\_2:VMS\_ONLINE\_DOCUMENTATION (3 of 4) ,  
 mounted on \_\$1\$DKA3: (CDROM)  
 %MOUNT-I-CDROM\_ISO, MANAGEMENT:VMS\_ONLINE\_DOCUMENTATION (4 of 4) ,  
 mounted on \_\$1\$DKA4: (CDROM)

"VMS\_ONLINE\_DOCUMENTATION"という名前の ISO 9660 ボリューム・セットのメンバ 4 つをマウントします。

9. \$ MOUNT/SYSTEM/SUBSYSTEM \$8\$DKA300: ATLANTIS\_WORK1  
 %MOUNT-I-MOUNTED, ATLANTIS\_WORK1 mounted on \_\$8\$DKA300: (ATLANTIS)  
 \$ SHOW DEVICE/FULL \$8\$DKA300:

Disk \$8\$DKA300: (ATLANTIS), device type RZ24, is online, mounted,  
 file-oriented device, shareable, served to cluster via MSCP Server,  
 error logging is enabled.

|                    |                  |                                  |                             |
|--------------------|------------------|----------------------------------|-----------------------------|
| Error count        | 0                | Operations completed             | 385                         |
| Owner process      | " "              | Owner UIC                        | [SYSTEM]                    |
| Owner process ID   | 00000000         | Dev Prot                         | S:RWPL,O:RWPL,G:R,W         |
| Reference count    | 1                | Default buffer size              | 512                         |
| Total blocks       | 409792           | Sectors per track                | 38                          |
| Total cylinders    | 1348             | Tracks per cylinder              | 8                           |
| Allocation class   | 8                |                                  |                             |
| Volume label       | "ATLANTIS_WORK1" | Relative volume number           | 0                           |
| Cluster size       | 3                | Transaction count                | 1                           |
| Free blocks        | 396798           | Maximum files allowed            | 51224                       |
| Extend quantity    | 5                | Mount count                      | 1                           |
| Mount status       | System           | Cache name                       | "_\$8\$DKA700:XQPCACHE"     |
| Extent cache size  | 64               | Maximum blocks in extent cache   | 39679                       |
| File ID cache size | 64               | Blocks currently in extent cache | 0                           |
| Quota cache size   | 50               | Maximum buffers in FCP cache     | 295                         |
| Volume owner UIC   | [VMS,PLATO]      | Vol Prot                         | S:RWCD,O:RWCD,G:RWCD,W:RWCD |

Volume status: ODS-2, subject to mount verification, protected  
 subsystems enabled, file high-water marking, write-through caching enabled.

ATLANTIS\_WORK1 というラベルのポリリュームがマウントされ、システム全体で  
 利用することができます。ポリリューム上のサブシステムにアクセスすることがで  
 きます。

10. \$ MOUNT DSA0: /SHADOW=(\$200\$DKA200:,\$200\$DKA300:,\$200\$DKA400:) X5OZCOPY  
 %MOUNT-I-MOUNTED, X5OZCOPY mounted on \_DSA0:  
 %MOUNT-I-SHDWMEMSUCC, \_\$200\$DKA200: (VIPER1) is now a valid member of  
 the shadow set  
 %MOUNT-I-SHDWMEMSUCC, \_\$200\$DKA300: (VIPER1) is now a valid member of  
 the shadow set  
 %MOUNT-I-SHDWMEMSUCC, \_\$200\$DKA400: (VIPER1) is now a valid member of  
 the shadow set  
 \$ DISMOUNT DSA0:  
 \$ MOUNT/INCLUDE DSA0: /SHADOW=\$200\$DKA200: X5OXCOPY  
 %MOUNT-I-MOUNTED, X5OZCOPY mounted on \_DSA0:  
 %MOUNT-I-SHDWMEMSUCC, \_\$200\$DKA200: (VIPER1) is now a valid member of  
 the shadow set  
 %MOUNT-I-AUTOMEMSUC, \_\$200\$DKA300: (VIPER1) automatically added to the  
 shadow set  
 %MOUNT-I-AUTOMEMSUC, \_\$200\$DKA400: (VIPER1) automatically added to the  
 shadow set

既存のシャドウ・セットを2とおりの方法でマウントしています。最初の  
 MOUNT コマンドでは、/SHADOW 修飾子によってシャドウ・セットの各メンバ

## MOUNT

を指定しています。次に DSA0: をディスマウントした後に、2 番目の MOUNT コマンドに /INCLUDE 修飾子を使用してシャドウ・セットのすべてのメンバを自動的にマウントしています。

## A

|                                |          |
|--------------------------------|----------|
| ACCOUNTING コマンド                |          |
| SET ACCOUNTING コマンドを参照         |          |
| ACL エディタ                       | DCLI-272 |
| ALLOCATE コマンド                  |          |
| DEASSIGN コマンドを参照               |          |
| DISMOUNT コマンドも参照               |          |
| ANALYZE/CRASH_DUMP コマンド        | DCLI-22  |
| Analyze/Disk_Structure ユーティリティ |          |
|                                | DCLI-23  |
| APPEND コマンド                    |          |
| COPY コマンドを参照                   |          |
| DECwindows 複合ドキュメントに使用         |          |
|                                | DCLI-60  |
| ASSIGN/QUEUE コマンド              |          |
| DEASSIGN/QUEUE コマンドを参照         |          |
| ASSIGN コマンド                    |          |
| DEASSIGN コマンドを参照               |          |

## C

|                         |          |
|-------------------------|----------|
| Cluster 関連 DCL コマンド     |          |
| ALLOCATE                | DCLI-17  |
| ANALYZE/AUDIT           | DCLI-21  |
| ASSIGN                  | DCLI-65  |
| ASSIGN/MERGE            | DCLI-73  |
| ASSIGN/QUEUE            | DCLI-75  |
| CREATE/NAME_TABLE       | DCLI-148 |
| DEALLOCATE              | DCLI-159 |
| DEASSIGN                | DCLI-161 |
| DEASSIGN/QUEUE          | DCLI-166 |
| DEFINE                  | DCLI-175 |
| DEFINE/CHARACTERISTIC   | DCLI-183 |
| DELETE/CHARACTERISTIC   | DCLI-201 |
| DELETE/ENTRY            | DCLI-203 |
| DELETE/INTRUSION_RECORD |          |
|                         | DCLI-208 |
| DELETE/QUEUE            | DCLI-213 |
| DELETE/QUEUE/MANAGER    | DCLI-215 |
| DISABLE AUTOSTART       | DCLI-250 |
| DISMOUNT                | DCLI-255 |
| ENABLE AUTOSTART        | DCLI-285 |
| INITIALIZE              | DCLI-339 |
| INITIALIZE/QUEUE        | DCLI-358 |
| MOUNT                   | DCLI-556 |

## Cluster 関連レキシカル関数

|                        |                    |
|------------------------|--------------------|
| F\$CONTEXT             | DCLI-391           |
| F\$CSID                | DCLI-398           |
| F\$DEVICE              | DCLI-408           |
| F\$GETDVI              | DCLI-436           |
| F\$GETJPI              | DCLI-450           |
| F\$GETQUI              | DCLI-458           |
| F\$GETSYI              | DCLI-480           |
| F\$PID                 | DCLI-509           |
| F\$TRNLNM              | DCLI-527           |
| CONVERT/DOCUMENT コマンド  |                    |
| オプション・ファイルの作成          | DCLI-110           |
| COPY コマンド              |                    |
| DECwindows 複合ドキュメントに使用 |                    |
|                        | DCLI-120           |
| ファイルの上書き               | DCLI-125           |
| CPU (中央処理装置)           |                    |
| バッチ・ジョブの最大時間制限の定義      |                    |
|                        | DCLI-364, DCLI-382 |
| バッチ・ジョブの時間制限           | DCLI-364           |

## D

|                   |          |
|-------------------|----------|
| DCL コマンド          |          |
| 実行の再開             | DCLI-101 |
| 実行の続行             | DCLI-101 |
| 入力ストリームの終わりを示す    | DCLI-289 |
| 入力ストリームの先頭を示す     | DCLI-172 |
| DDIF 出力コンバータ      | DCLI-107 |
| DDIF 入力コンバータ      | DCLI-106 |
| DEALLOCATE コマンド   |          |
| ALLOCATE コマンドも参照  |          |
| DEASSIGN コマンド     |          |
| DEFINE コマンドを参照    |          |
| DECK コマンド         |          |
| EOD を参照           |          |
| DECterm ウィンドウ     |          |
| アプリケーション・キーパッドの設定 |          |
|                   | DCLI-153 |
| DECTPU            |          |
| 起動                | DCLI-284 |
| DEC テキスト処理ユーティリティ |          |
| DECTPU を参照        |          |
| DEFINE コマンド       |          |
| DEASSIGN コマンドを参照  |          |
| DEPOSIT コマンド      |          |
| EXAMINE コマンドも参照   |          |
| 基数修飾子             | DCLI-220 |

## DEPOSIT コマンド (続き)

|                  |          |
|------------------|----------|
| 長さ修飾子            | DCLI-221 |
| DIFFERENCES コマンド | DCLI-224 |
| コメント区切り文字        | DCLI-227 |
| コメント文字           | DCLI-227 |
| 終了状態             | DCLI-225 |
| 出力形式             | DCLI-230 |
| DTIF 出力コンバータ     | DCLI-108 |
| DTIF 入力コンバータ     | DCLI-107 |

## E

|                              |                              |
|------------------------------|------------------------------|
| EDT エディタ                     | DCLI-273                     |
| ELSE キーワード                   |                              |
| IF コマンド                      | DCLI-335                     |
| EOD(end-of-deck)             |                              |
| EOD コマンドを参照                  |                              |
| EOD コマンド                     |                              |
| DECK コマンド                    | DCLI-172                     |
| EOF(end-of-file)             |                              |
| 状態                           | DCLI-289                     |
| EOF(end-of-file) 指示子         | DCLI-172                     |
| Error Log Viewer (ELV)       | DCLI-24                      |
| EVE エディタ                     |                              |
| 起動                           | DCLI-284                     |
| EXAMINE コマンド                 |                              |
| DEPOSIT コマンド                 | DCLI-219                     |
| 長さ修飾子                        | DCLI-293                     |
| EXCHANGE/NETWORK コマンド        |                              |
| 修飾子                          | DCLI-302                     |
| ファイルの作成                      | DCLI-301                     |
| ファイルの転送                      | DCLI-299, DCLI-300           |
| ファイルの保護                      | DCLI-301                     |
| ワイルドカード文字                    | DCLI-300                     |
| /EXECUTIVE_MODE 修飾子          |                              |
| ASSIGN コマンド                  | DCLI-67                      |
| Extended File Specifications |                              |
| COPY コマンド                    | DCLI-126                     |
| DEFINE コマンド                  | DCLI-177                     |
| DELETE コマンド                  | DCLI-197                     |
| DIRECTORY コマンド               | DCLI-244                     |
| DUMP コマンド                    | DCLI-263, DCLI-264, DCLI-267 |
| EXCHANGE/NETWORK コマンド        |                              |
|                              | DCLI-305                     |
| F\$FILE_ATTRIBUTES レキシカル関数   | DCLI-433, DCLI-434           |
| F\$GETDVI レキシカル関数            | DCLI-437                     |
| F\$GETJPI レキシカル関数            | DCLI-454                     |
| INITIALIZE コマンド              | DCLI-354                     |

## F

|                       |          |
|-----------------------|----------|
| F\$LOGICAL レキシカル関数    |          |
| F\$TRNLNM レキシカル関数を参照  |          |
| FHM (ファイルのハイウオータ・マーク) |          |
|                       | DCLI-348 |

## Files-11 ディスク構造

|                  |          |
|------------------|----------|
| オンディスク構造レベル 1 形式 | DCLI-339 |
| ディスクの初期化         | DCLI-339 |

## G

|                         |          |
|-------------------------|----------|
| GSD (グローバル・シンボル・ディレクトリ) |          |
| オブジェクト・ファイルの分析          | DCLI-42  |
| GST (グローバル・シンボル・テーブル)   |          |
| 分析                      | DCLI-28  |
| GST(グローバル・シンボル・テーブル)    |          |
| シンボルの削除                 | DCLI-217 |
| シンボルの入力                 | DCLI-377 |

## I

|              |          |
|--------------|----------|
| ISO 9660 標準  |          |
| DUMP ユーティリティ | DCLI-261 |

## J

|           |          |
|-----------|----------|
| JAVA コマンド | DCLI-380 |
|-----------|----------|

## L

|              |          |
|--------------|----------|
| LOGIN プロシージャ | DCLI-544 |
| LOGOUT コマンド  |          |
| 複数の          | DCLI-549 |
| メッセージ        | DCLI-549 |

## M

|                     |                    |
|---------------------|--------------------|
| Mail ユーティリティ (MAIL) | DCLI-552           |
| MOUNT コマンド          |                    |
| DEASSIGN コマンドを参照    |                    |
| ANSI ラベルの磁気テープのマウント | DCLI-575, DCLI-577 |
| UIC (ユーザ識別コード) の指定  | DCLI-582           |
| オペレータ補助による要求        | DCLI-562, DCLI-593 |
| 書き込み保護の指定           | DCLI-592           |
| 共用可能ボリュームの指定        | DCLI-588           |
| サブシステムへのアクセスを可能にする  | DCLI-589           |
| 磁気テープ・ブロック・サイズの指定   | DCLI-565           |
| 磁気テープ密度の指定          | DCLI-573           |
| 磁気テープ・レコード・サイズの指定   | DCLI-587           |
| 自動リビルドの禁止           | DCLI-586           |
| シャドウイング・ミニコピーの制御    | DCLI-582           |
| シャドウ・ボリューム・セットの作成   | DCLI-565           |
| 修飾子の特殊な特権           | DCLI-560           |
| 省略時のファイル拡張ブロック数の指定  | DCLI-573           |
| 制約                  | DCLI-560           |



## MOUNT コマンド (続き)

|                                          |                    |
|------------------------------------------|--------------------|
| ディスク・ボリュームのリビルド . . .                    | DCLI-586           |
| ディレクトリ数の指定 . . . . .                     | DCLI-561           |
| パラメータ . . . . .                          | DCLI-556           |
| ファイル・ウィンドウ・マッピング・ポインタの<br>割り当て . . . . . | DCLI-592           |
| 複数のフォーリン・テープ・ボリュー<br>ム . . . . .         | DCLI-579           |
| 保護コードの指定 . . . . .                       | DCLI-585           |
| 補助制御プロセス (ACP) の要求 . . . .               | DCLI-584           |
| ボリューム・セットの作成 . . . . .                   | DCLI-565           |
| ボリュームのクラスタ単位のマウント<br>. . . . .           | DCLI-567           |
| ボリューム保護チェックを無効にする<br>. . . . .           | DCLI-580, DCLI-581 |
| ボリュームを公用にする . . . . .                    | DCLI-589           |

## O

### OpenVMS Cluster 環境

|                                |          |
|--------------------------------|----------|
| シャドウ・セット・メンバの除外 . . .          | DCLI-257 |
| ディスクのクラスタ・サイズの指<br>定 . . . . . | DCLI-342 |
| ボリュームのデismount . . . . .       | DCLI-257 |

### OpenVMS 以外のシステム

|                       |          |
|-----------------------|----------|
| リモート・ファイル指定 . . . . . | DCLI-134 |
|-----------------------|----------|

## P

|                               |                                           |
|-------------------------------|-------------------------------------------|
| /PAGE=SAVE 修飾子と移動キー . . . . . | DCLI-231,<br>DCLI-241, DCLI-266, DCLI-322 |
| PostScript 出力コンバータ . . . . .  | DCLI-109                                  |
| 処理オプション . . . . .             | DCLI-113                                  |

## R

|                                            |                              |
|--------------------------------------------|------------------------------|
| RAD (リソース・アフィニティ・ドメイン) のサポー<br>ト . . . . . | DCLI-370, DCLI-453, DCLI-487 |
| Return キー<br>ログイン . . . . .                | DCLI-544                     |

## S

|                                              |          |
|----------------------------------------------|----------|
| \$SEVERITY グローバル・シンボル<br>変更 . . . . .        | DCLI-308 |
| SPAWN コマンド<br>ATTACH コマンドを参照                 |          |
| \$STATUS グローバル・シンボル<br>変更 . . . . .          | DCLI-308 |
| STOP/QUEUE/NEXT コマンド<br>DELETE/QUEUE コマンドを参照 |          |
| STOP コマンド<br>テープの暴走 . . . . .                | DCLI-340 |
| SUMSLP エディタ . . . . .                        | DCLI-279 |
| Supplementary Volume Descriptor<br>SVD を参照   |          |
| SVD (補助ボリューム記述子) . . . . .                   | DCLI-589 |

## SYS\$SYLOGIN ログイン名

|              |          |
|--------------|----------|
| 実行 . . . . . | DCLI-544 |
|--------------|----------|

## T

|                                 |          |
|---------------------------------|----------|
| TECO エディタ . . . . .             | DCLI-280 |
| THEN キーワード<br>IF コマンド . . . . . | DCLI-335 |
| TPU<br>DECTPU を参照               |          |

## U

|                                      |          |
|--------------------------------------|----------|
| UIC (利用者識別コード)<br>指定 . . . . .       | DCLI-582 |
| UIC(利用者識別コード)<br>ディレクトリに指定 . . . . . | DCLI-141 |
| ファイルに指定 . . . . .                    | DCLI-137 |

## V

|                                         |          |
|-----------------------------------------|----------|
| VIRTUALPAGECNT システム・パラメー<br>タ . . . . . | DCLI-559 |
|-----------------------------------------|----------|

## ア

|                                         |          |
|-----------------------------------------|----------|
| アクセス制御リスト・エディタ<br>ACL エディタを参照           |          |
| アクセス日<br>DIRECTORY/DATE コマンド . . . . .  | DCLI-238 |
| DUMP/HEADER コマンド . . . . .              | DCLI-264 |
| SET VOLUME コマンド . . . . .               | DCLI-355 |
| アンロードできない装置<br>DISMOUNT コマンドで . . . . . | DCLI-259 |

## イ

|                                  |          |
|----------------------------------|----------|
| イメージ<br>EXIT コマンドによる終了 . . . . . | DCLI-308 |
| 実行の再開 . . . . .                  | DCLI-101 |
| 実行の続行 . . . . .                  | DCLI-101 |
| イメージ・ファイル<br>エラー分析 . . . . .     | DCLI-26  |
| グローバル・シンボル・テーブルの分<br>析 . . . . . | DCLI-28  |
| パッチ・テキスト・レコードの分<br>析 . . . . .   | DCLI-30  |
| フィックスアップ・セクションの分析<br>. . . . .   | DCLI-28  |
| 分析 . . . . .                     | DCLI-25  |
| インターチェンジ環境<br>保護 . . . . .       | DCLI-349 |

## ウ

### ウェイクアップ要求

- 取り消し ..... DCLI-85
- 上書き
  - 磁気テープの重ね書き保護 ..... DCLI-352
  - 所有者識別子フィールド ..... DCLI-352

## エ

### エディタ

- EVE エディタを参照

### エラー

#### 報告

- イメージ・ファイルの ..... DCLI-26
- オブジェクト・ファイル ..... DCLI-39

### エラー・メッセージ

- オンライン・ドキュメント ..... DCLI-328

## オ

### 大文字小文字の区別

- CREATE コマンド ..... DCLI-139
- F\$GETJPI レキシカル関数 ..... DCLI-451

### オブジェクト・ファイル

#### 識別

- エラー ..... DCLI-39
- 分析 ..... DCLI-39

### オブジェクト・モジュール

- 分析 ..... DCLI-39, DCLI-40, DCLI-41
- ファイルの終端レコード ..... DCLI-42

### オブジェクト・ライブラリ

### オプション・ファイル

- CONVERT/DOCUMENT コマンド
  - ..... DCLI-110
- 作成 ..... DCLI-110
- オンライン・ヘルプ ..... DCLI-319

## カ

### 解除

- 論理名の割り当て ..... DCLI-161

### 回復

- EDT 編集画面 ..... DCLI-277
- 会話型ヘルプ ..... DCLI-323

### 書き込みチェック

- APPEND コマンドによる ..... DCLI-63
- COPY コマンドによる ..... DCLI-127
- INITIALIZE コマンドによる ..... DCLI-343

### 拡張可能多機能エディタ

- EVE エディタを参照

### 仮想装置

### 仮想端末

- 接続 ..... DCLI-97
- 切断 ..... DCLI-253

### 仮想メモリ

- 内容の置換 ..... DCLI-219
- 内容を表示 ..... DCLI-292

### カード

- バッチ・ジョブの登録 ..... DCLI-381
- カードのバッチ・ジョブの最後 ..... DCLI-291
- カード・リーダー
  - バッチ・ジョブの最後 ..... DCLI-291
- 画面指向エディタ

- EDT ..... DCLI-273

### 画面用エディタ

- EVE エディタ ..... DCLI-284

### 可用性

- キューの ..... DCLI-361, DCLI-362

## キ

### 記述子ダンブ

### 偽の式

- IF コマンド ..... DCLI-335

### キーパッド・アプリケーション

- DECTerm に定義 ..... DCLI-153

### 基本優先順位

- バッチ・ジョブに設定 ..... DCLI-362

### キャッシング属性

- 省略時の値 ..... DCLI-340
- 表示 ..... DCLI-237
- ライトスルー ..... DCLI-340

### キュー

- 異常終了の設定 ..... DCLI-362
- エントリの削除 ..... DCLI-203
- オプションをオンに変更 ..... DCLI-358
- 可用性の保証 ..... DCLI-361, DCLI-362
- 既存のキューの再初期化 ..... DCLI-358
- 作成 ..... DCLI-358
- サーバ ..... DCLI-361
- 実行 ..... DCLI-360
- 自動起動またはその無効の指

- 定 ..... DCLI-361

- 自動起動 ..... DCLI-361, DCLI-362

- 自動起動キューのスタート ..... DCLI-285

- 自動起動の禁止 ..... DCLI-250

- 自動起動またはその無効の指定 ..... DCLI-361

- 初期化 ..... DCLI-358

- ジョブの削除 ..... DCLI-73

- ジョブのマージ ..... DCLI-73

- 処理中の自動起動キュー ..... DCLI-285

- シンビオント ..... DCLI-361

- 装置への割り当て ..... DCLI-75

- タイプ ..... DCLI-360

### 停止

- シャットダウン前 ..... DCLI-250

- バッチ・ジョブの CPU 時間制限の定

- 義 ..... DCLI-364

- バッチ・ジョブの基本優先順位の設定

- ..... DCLI-362

- バッチ・ジョブの省略時のワーキング・セットの
- 定義 ..... DCLI-372, DCLI-385

## キュー (続き)

|                         |                    |
|-------------------------|--------------------|
| バッチ・ジョブのワーキング・セット・クォータ  |                    |
| 値の定義                    | DCLI-386           |
| バッチ・ジョブのワーキング・セット超過値の定義 | DCLI-372, DCLI-385 |
| バッチまたはプリント・キューの削除       | DCLI-213           |
| 汎用                      | DCLI-360           |
| フェイルオーバー                | DCLI-361, DCLI-362 |
| フェイルオーバーの設定             | DCLI-361           |
| フェイルオーバーの抑止             | DCLI-250           |
| 論理                      | DCLI-360           |
| 論理名テーブルの割り当て            | DCLI-75            |
| 割り当て                    | DCLI-358           |
| 割り当て解除                  | DCLI-166           |
| キュー・オプション               |                    |
| 変更                      | DCLI-358           |
| キューの起動                  |                    |
| 自動起動                    | DCLI-361           |
| 非自動起動                   | DCLI-361           |
| キューの作成                  | DCLI-358           |
| キューのスタート                |                    |
| 自動起動                    | DCLI-285           |
| 共用可能イメージ                |                    |
| ファイルの分析                 | DCLI-26            |
| 分析                      | DCLI-25            |
| 共用可能ボリューム               |                    |
| ディスクの初期化                | DCLI-354           |
| ディスマウント                 | DCLI-255           |
| 共用装置                    |                    |
| ディスマウント                 | DCLI-257           |

## ク

|                   |                   |
|-------------------|-------------------|
| クォータ・チェック         |                   |
| 制御                | DCLI-585          |
| グループ論理名テーブル       |                   |
| エントリの削除           | DCLI-163          |
| 論理名の登録            | DCLI-67, DCLI-177 |
| グローバル・シンボル        | DCLI-2, DCLI-6    |
| グローバル・シンボル・ディレクトリ |                   |
| GSD を参照           |                   |
| グローバル・シンボル・テーブル   |                   |
| GST を参照           |                   |

## ケ

|                       |          |
|-----------------------|----------|
| 形式化                   |          |
| DIFFERENCES コマンドによる出力 |          |
| の                     | DCLI-229 |

## コ

|              |          |
|--------------|----------|
| 構造レベル        |          |
| ディスクの定義      | DCLI-355 |
| コマンド言語インタプリタ |          |
| 別の名前前の指定     | DCLI-545 |

## コマンド・プロシージャ

|                   |                    |
|-------------------|--------------------|
| 会話型でのシンボルの割り当て    | DCLI-376           |
| 式のテスト             | DCLI-335           |
| 実行                | DCLI-10            |
| 実行の再開             | DCLI-101           |
| 実行の続行             | DCLI-101           |
| 終了                | DCLI-308           |
| 制御の移動             | DCLI-80            |
| 制御を移す             | DCLI-313, DCLI-316 |
| パラメータ             | DCLI-10            |
| プロンプトの表示          | DCLI-376           |
| ラベル               | DCLI-80, DCLI-316  |
| !(コメント区切り文字) コマンド | DCLI-1             |
| コメント区切り文字         | DCLI-227           |
| DIFFERENCES も参照   |                    |
| コメント文字            | DCLI-227           |
| DIFFERENCES も参照   |                    |
| コマンド・プロシージャ       |                    |
| ラベル               | DCLI-313           |

## サ

|                     |                   |
|---------------------|-------------------|
| サイズ的一致              |                   |
| DIFFERENCES コマンドで指定 | DCLI-229          |
| 索引ファイル              |                   |
| ディスク上の配置            | DCLI-349          |
| 削除                  |                   |
| 複数のファイル             | DCLI-194          |
| 論理名                 | DCLI-161          |
| 論理名テーブル             | DCLI-161          |
| サーチ・リスト             | DCLI-65, DCLI-175 |
| サーバ・キュー             | DCLI-361          |
| サブディレクトリ            |                   |
| 作成                  | DCLI-140          |
| サブプロセス              |                   |
| 入力ストリームの制御の切り替え     | DCLI-77           |

## シ

|                        |                    |
|------------------------|--------------------|
| シェルピング                 |                    |
| ファイルがプリシェルブドであるかどうかの判断 | DCLI-243           |
| シェルブド                  |                    |
| ファイルがシェルブドであるかどうかの判断   | DCLI-243, DCLI-244 |
| ファイルがシェルブドであるかの判断      | DCLI-434           |
| ファイルをシェルブドできるかの判断      | DCLI-434           |
| 式                      |                    |
| 値のテスト                  | DCLI-335           |
| 磁気テープ                  |                    |
| ANSI ラベルのマウント          | DCLI-575, DCLI-577 |
| 複数のフォーリン・ボリュームのマウント    | DCLI-579           |
| ブロック・サイズ指定             | DCLI-565           |

## 磁気テープ (続き)

|                     |                    |
|---------------------|--------------------|
| 保護チェックを無効にする        | DCLI-580           |
| マウント                | DCLI-560           |
| 密度指定                | DCLI-573           |
| レコード・サイズ指定          | DCLI-587           |
| 式の値のテスト             | DCLI-335           |
| 実行                  |                    |
| SYS\$LOGIN          | DCLI-544           |
| 別のログイン・コマンド・プロシージャ  |                    |
| ヤ                   | DCLI-545           |
| ログイン・プロシージャ         | DCLI-544           |
| 割り込みコマンド・プロシージャの続行  | DCLI-101           |
| 割り込みプログラムの続行        | DCLI-101           |
| 実行キュー               | DCLI-360           |
| 自動起動またはその無効の指定      | DCLI-361           |
| ノードまたはノードと装置の指定     | DCLI-361           |
| 実行の再開               |                    |
| DCL コマンド            | DCLI-101           |
| コマンド・プロシージャ         | DCLI-101           |
| プログラム               | DCLI-101           |
| システム                |                    |
| アクセス                | DCLI-544           |
| システム・パラメータ          |                    |
| VIRTUALPAGECNT      | DCLI-559           |
| システム・メッセージ          |                    |
| オンライン・ドキュメント        | DCLI-328           |
| 自動アンシエルブ            |                    |
| 確認                  | DCLI-451           |
| 判断                  | DCLI-455           |
| 自動起動キュー             | DCLI-361, DCLI-362 |
| 禁止                  | DCLI-250           |
| シャットダウン前に停止         | DCLI-250           |
| 処理中の                | DCLI-285           |
| ノード上の許可             | DCLI-285           |
| フェイルオーバー            | DCLI-250, DCLI-285 |
| フェイルオーバーの抑止         | DCLI-250           |
| 自動起動キューの起動          | DCLI-362           |
| 自動起動の許可             | DCLI-285           |
| 自動起動の禁止             |                    |
| ノード上の               | DCLI-250           |
| シャットダウン前の自動起動キューの停止 | DCLI-250           |
| 終了                  |                    |
| コマンド・プロシージャ         | DCLI-308           |
| ターミナル・セッション         | DCLI-549           |
| 16 進数でダンプ           | DCLI-264           |
| 10 進数でダンプ           | DCLI-263           |
| 出力コンバータ             |                    |
| DDIF                | DCLI-107           |
| DTIF                | DCLI-108           |
| PostScript          | DCLI-109           |
| テキスト                | DCLI-109           |
| 出力メッセージ             |                    |
| ボリューム・マウント          | DCLI-559           |
| 省略時のワーキング・セット       |                    |
| バッチ・ジョブ用            | DCLI-385           |

## 初期化

|                            |                    |
|----------------------------|--------------------|
| キュー                        | DCLI-358           |
| ボリューム                      | DCLI-339           |
| ジョブ                        |                    |
| CPU 時間制限の定義                | DCLI-364           |
| キューからの削除                   | DCLI-203, DCLI-213 |
| キュー登録からの削除                 |                    |
| ASSIGN/MERGE コマンド          | DCLI-73            |
| 他のキュー登録への変更                | DCLI-73            |
| ジョブ単位の論理名テーブル              |                    |
| 論理名の登録                     | DCLI-67            |
| ジョブ・バッチ・カード                |                    |
| 最後                         | DCLI-291           |
| ジョブ論理名テーブル                 |                    |
| エントリの削除                    | DCLI-163           |
| 論理名の登録                     | DCLI-178           |
| 所有権                        |                    |
| ボリュームに対する指定                | DCLI-352           |
| 所有者識別子フィールド                |                    |
| 文字の書き込み                    | DCLI-349           |
| 処理オプション                    |                    |
| PostScript 出力コンバータ         | DCLI-113           |
| オプション・ファイル中の               | DCLI-110           |
| テキスト出力コンバータ                | DCLI-112           |
| ドメイン・コンバータ                 | DCLI-114           |
| 分析出力コンバータ                  | DCLI-111           |
| 処理中の自動起動キュー                | DCLI-285           |
| 真の式                        |                    |
| IF コマンド                    | DCLI-335           |
| シンビオント                     |                    |
| 省略時の設定の                    | DCLI-361           |
| 特別な目的の                     | DCLI-361           |
| ユーザ作成の                     | DCLI-361           |
| シンボリック名                    |                    |
| 定義                         | DCLI-2, DCLI-6     |
| シンボル                       |                    |
| コマンド・プロシージャの会話型でのシンボルの割り当て | DCLI-376           |
| 削除                         |                    |
| グローバル・シンボル・テーブルから          | DCLI-217           |
| ローカル・シンボル・テーブルから           | DCLI-217           |
| 2 進数で置き換え                  | DCLI-3             |
| 汎用割り当て                     | DCLI-2             |
| 文字の置き換え                    | DCLI-7             |
| 文字列割り当て文                   | DCLI-6             |

## ソ

### 続行

|                 |          |
|-----------------|----------|
| 割り込み DCL コマンド   | DCLI-101 |
| 割り込みコマンド・プロシージャ | DCLI-101 |
| 割り込みプログラム       | DCLI-101 |

### 装置

|                     |          |
|---------------------|----------|
| DISMOUNT コマンドでアンロード | DCLI-259 |
| アクセス                | DCLI-17  |

## 装置 (続き)

|             |                   |
|-------------|-------------------|
| 仮想          | DCLI-296          |
| 占有          | DCLI-17           |
| 占有を解除       | DCLI-159          |
| ディスマウント     | DCLI-255          |
| 論理キュー名の割り当て | DCLI-75           |
| 論理名の割り当て    | DCLI-17           |
| 装置の占有       | DCLI-17           |
| 装置の占有を解除    | DCLI-159          |
| 装置名         |                   |
| 論理名の割り当て    | DCLI-65, DCLI-175 |

## タ

|              |                     |
|--------------|---------------------|
| ダンプの形式       |                     |
| バイト          | DCLI-263            |
| 8進数          | DCLI-265            |
| ターミナル        |                     |
| 省略時の属性       | DCLI-544            |
| ターミナル・エミュレータ |                     |
| 作成           | DCLI-152            |
| ターミナル・セッション  |                     |
| ログアウト        | DCLI-549            |
| ログイン         | DCLI-544            |
| ダンプ          |                     |
| ファイルの        | DCLI-261            |
| ボリュームの       | DCLI-261            |
| ダンプの形式       | DCLI-263 ~ DCLI-268 |
| 記述子          | DCLI-263            |
| 識別子          | DCLI-265            |
| 16進数         | DCLI-264            |
| 10進数         | DCLI-263            |
| ロングワード       | DCLI-265            |
| ワード          | DCLI-268            |
| ダンプの読み込み     | DCLI-262            |
| 端末           |                     |
| 仮想           | DCLI-97, DCLI-253   |

## チ

|                           |                    |
|---------------------------|--------------------|
| チェックサム・ユーティリティ            |                    |
| ファイルとデータのチェックサムのためのアルゴリズム | DCLI-89            |
| 中央処理装置                    |                    |
| CPUを参照                    |                    |
| 重複ラベル                     |                    |
| 関係するコマンド・インタプリタの規則        | DCLI-80            |
| コマンド割り込み規則                | DCLI-313, DCLI-316 |

## テ

|                    |          |
|--------------------|----------|
| ディスク               |          |
| DECram ディスクのサイズの指定 | DCLI-354 |
| 共用可能ボリューム指定        | DCLI-588 |
| 共用可能ボリュームの定義       | DCLI-354 |
| 記録密度の指定            | DCLI-343 |
| クラスタ・サイズの指定        | DCLI-342 |

## ディスク (続き)

|                      |                    |
|----------------------|--------------------|
| クラスタワイドのマウント         | DCLI-567           |
| 構造レベルの定義             | DCLI-355           |
| 公用ディスク・ボリュームの作成      | DCLI-589           |
| 最大ファイル数の指定           | DCLI-351           |
| 索引ファイルの配置            | DCLI-349           |
| 順編成ファイルの作成           | DCLI-136           |
| 省略時の設定のファイル拡張サイズの指定  | DCLI-346           |
| ディスマウント              | DCLI-255           |
| ディレクトリ・スペース割り当て      | DCLI-345           |
| ファイル                 |                    |
| 削除                   | DCLI-194           |
| 比較                   | DCLI-224           |
| 不良ブロック・データの指定        | DCLI-355           |
| 不良領域の指定              | DCLI-341           |
| ボリューム・シャドウイングによるマウント | DCLI-569, DCLI-570 |
| ボリューム・セットのディスマウント    | DCLI-259           |
| マウント                 | DCLI-560           |
| クラスタ単位の              | DCLI-560           |
| ボリューム・シャドウイング        | DCLI-587           |
| マッピング・ポインタの割り当て      | DCLI-356           |
| ディスク・ボリューム           |                    |
| ディスクも参照              |                    |
| 自動リビルドの禁止            | DCLI-585           |
| 初期化                  | DCLI-339           |
| 物理ロード                | DCLI-559, DCLI-591 |
| 保護チェックを無効にする         | DCLI-580           |
| リビルド                 | DCLI-585           |
| ディスマウント              |                    |
| 共用装置                 | DCLI-257           |
| クラスタ全体でのボリュームの       | DCLI-257           |
| ディスクの                | DCLI-255           |
| テープの                 | DCLI-255           |
| ディレクトリ               |                    |
| コピー                  | DCLI-122           |
| 作成                   | DCLI-140           |
| シェルドのファイル            | DCLI-244           |
| ディスクでのスペース事前割り当て     | DCLI-345           |
| 内容の表示                | DCLI-235           |
| ファイルのバージョン数の制限       |                    |
| 作成時の定義               | DCLI-142           |
| 保護                   |                    |
| 作成時の定義               | DCLI-141           |
| レディ・アクセス             | DCLI-341           |
| ディレクトリのコピー           | DCLI-122           |
| ディレクトリのファイルのリスト      | DCLI-235           |
| テキスト                 |                    |
| オブジェクト・ファイルの分析       | DCLI-47            |
| テキスト・エディタ            |                    |
| EDT エディタを参照          |                    |
| EVE エディタを参照          |                    |
| SUMSLP エディタを参照       |                    |
| TECO エディタを参照         |                    |

|                       |                   |
|-----------------------|-------------------|
| テキスト出力コンバータ           | DCLI-109          |
| 処理オプション               | DCLI-112          |
| テキスト入力コンバータ           | DCLI-108          |
| テスト                   |                   |
| 式の値                   | DCLI-335          |
| データ・ストリーム             |                   |
| 終わりを示す                | DCLI-289          |
| 先頭を示す                 | DCLI-172          |
| データ・ストリームの終わり         | DCLI-289          |
| EOD コマンドを参照           |                   |
| データの圧縮                |                   |
| ボリューム・マウント            | DCLI-577          |
| データ・レコードの圧縮           | DCLI-351          |
| デバッグ                  |                   |
| DEPOSIT コマンド          | DCLI-219          |
| オブジェクト・ファイル中のデバッグの分   |                   |
| 析                     | DCLI-42           |
| 起動                    | DCLI-49, DCLI-167 |
| 情報レコード分析              | DCLI-42           |
| 保持デバッグ                | DCLI-168          |
| テープ                   |                   |
| デイスマウント               | DCLI-255          |
| テープの暴走                |                   |
| 停止方法                  | DCLI-340          |
| テープ密度                 |                   |
| ボリューム・マウント            | DCLI-571          |
| 転送モード                 |                   |
| EXCHANGE/NETWORK コマンド |                   |
|                       | DCLI-299          |

## ト

|            |                   |
|------------|-------------------|
| 等価名        |                   |
| 論理名への割り当て  | DCLI-65, DCLI-175 |
| ドメイン・コンバータ |                   |
| 処理オプション    | DCLI-114          |
| ドル記号(\$)   |                   |
| DECK コマンド  | DCLI-172          |
| EOD コマンド   | DCLI-289          |
| EOJ コマンド   | DCLI-291          |

## ナ

|        |                   |
|--------|-------------------|
| 名前     |                   |
| シンボル定義 | DCLI-2, DCLI-6    |
| 汎用装置   | DCLI-17           |
| 論理名    |                   |
| 解除     | DCLI-94, DCLI-161 |

## ニ

|             |          |
|-------------|----------|
| 二重引用符 (")   |          |
| リモート・ファイル指定 | DCLI-134 |
| 入力コンバータ     |          |
| DDIF        | DCLI-106 |
| DTIF        | DCLI-107 |
| テキスト        | DCLI-108 |

|                 |          |
|-----------------|----------|
| 入力ストリーム         |          |
| 他のプロセスへの制御の切り替え | DCLI-77  |
| 入力データ・ストリーム     |          |
| 終わりを示す          | DCLI-289 |
| 先頭を示す           | DCLI-172 |

## ハ

|                   |                    |
|-------------------|--------------------|
| バイト単位のダンプ         | DCLI-263           |
| バージョン数の制限         |                    |
| ディレクトリのファイル       | DCLI-142           |
| バージョン番号           |                    |
| ファイルへの割り当て        | DCLI-301           |
| 8 進数でダンプ          | DCLI-265           |
| パスワード             |                    |
| ログイン時に設定          | DCLI-544           |
| バッチ型エディタ          | DCLI-279           |
| バッチ・キュー           |                    |
| キューを参照            |                    |
| バッチ・ジョブ           |                    |
| CPU 時間の制限         | DCLI-383           |
| CPU の最大時間制限の定義    | DCLI-382           |
| カードからの登録          | DCLI-381           |
| カードのジョブの最後        | DCLI-291           |
| 省略時のワーキング・セットの定   |                    |
| 義                 | DCLI-372, DCLI-385 |
| 保持                | DCLI-383           |
| ログ・ファイルの削除        | DCLI-383           |
| ログ・ファイルの保存        | DCLI-383           |
| ワーキング・セット         |                    |
| クォータの定義           | DCLI-386           |
| 省略時の定義            | DCLI-372, DCLI-385 |
| 超過値の定義            | DCLI-372, DCLI-385 |
| ワーキング・セット・クォータの定義 |                    |
|                   | DCLI-386           |
| ワーキング・セット超過値の定義   | DCLI-372, DCLI-385 |
| パッチ・テキスト・レコード     |                    |
| 分析                | DCLI-30            |
| ハード・リンク           |                    |
| SET VOLUME コマンド   | DCLI-355           |
| パラメータ             |                    |
| コマンド・プロシージャ       | DCLI-10            |
| コマンド・プロシージャへの引き渡し |                    |
|                   | DCLI-80            |
| 指定                |                    |
| コマンド・プロシージャ       | DCLI-10            |
| 汎用キュー             | DCLI-360           |
| 起動                | DCLI-360           |
| 終了                | DCLI-360           |
| 初期化               | DCLI-367           |
| 汎用装置名             | DCLI-17            |

## ヒ

### 比較

|          |          |
|----------|----------|
| ファイル     | DCLI-224 |
| レコード内の文字 | DCLI-225 |

## フ

### フラグ値

#### ファイル

|                     |                              |
|---------------------|------------------------------|
| SUMSLP エディタで更新      | DCLI-279                     |
| クローズ                | DCLI-94                      |
| コピー                 | DCLI-119, DCLI-297           |
| 削除                  | DCLI-194                     |
| 作成                  | DCLI-119, DCLI-136, DCLI-297 |
| EDT エディタ            | DCLI-273                     |
| TECO エディタ           | DCLI-280                     |
| 所有者の UIC            | DCLI-137                     |
| ダンプ                 | DCLI-261                     |
| 追加                  | DCLI-59                      |
| ディスクでの最大数           | DCLI-351                     |
| ディスクの省略時の設定の拡張サイズ   | DCLI-346                     |
| ディレクトリ内のリスト         | DCLI-235                     |
| 転送                  | DCLI-297                     |
| バージョン数の制限           |                              |
| ディレクトリ作成時の定義        | DCLI-142                     |
| 比較                  | DCLI-224                     |
| 表示                  |                              |
| 最新のバージョン            | DCLI-246                     |
| 作成時刻                | DCLI-245                     |
| 作成日                 | DCLI-238                     |
| 使用ブロック数             | DCLI-244                     |
| 所有者の UIC            | DCLI-241                     |
| バックアップ時刻            | DCLI-245                     |
| バックアップ日             | DCLI-238                     |
| ヘルプ                 | DCLI-319                     |
| 変更時刻                | DCLI-245                     |
| 変更日                 | DCLI-238                     |
| 保護                  | DCLI-242                     |
| 満了時刻                | DCLI-245                     |
| 満了日                 | DCLI-238                     |
| 割り当てブロック数           | DCLI-244                     |
| ヘッダの割り当て            | DCLI-347                     |
| 編集                  |                              |
| EDT エディタ            | DCLI-273                     |
| SUMSLP エディタ         | DCLI-279                     |
| TECO エディタ           | DCLI-280                     |
| 文字, 文字列, またはレコードの無視 | DCLI-228                     |
| 連結                  | DCLI-119                     |
| 論理名の解除              | DCLI-94                      |
| ファイル・イメージ           |                              |
| フィックスアップ・セクションの分析   | DCLI-28                      |
| 分析                  | DCLI-25                      |
| ページ・ブレイクの指定         | DCLI-30                      |

#### ファイル・ウィンドウ

|                 |                    |
|-----------------|--------------------|
| マッピング・ポインタの割り当て | DCLI-356, DCLI-591 |
|-----------------|--------------------|

#### ファイル・オブジェクト

|                        |         |
|------------------------|---------|
| 識別                     | DCLI-39 |
| エラー                    | DCLI-39 |
| 分析                     | DCLI-39 |
| グローバル・シンボル・ディレクトリ・レコード | DCLI-42 |
| テキスト                   | DCLI-47 |
| デバッグ情報レコード             | DCLI-42 |
| ページ・ブレイクの指定            | DCLI-44 |
| モジュール・トレースバック・レコード     | DCLI-44 |
| モジュール・ヘッダ・レコード         | DCLI-43 |
| リンク・オプション指定レコード        | DCLI-43 |
| レコードの再配置               | DCLI-47 |

#### ファイル共用可能イメージ

|    |                  |
|----|------------------|
| 分析 | DCLI-25, DCLI-26 |
|----|------------------|

#### ファイル転送

|                  |          |
|------------------|----------|
| ファイル転送が可能であるかの判断 | DCLI-434 |
| ファイル転送モード        | DCLI-299 |

#### ファイルのハイウォータ・マーク

FHM を参照

#### ファイルの保護

|                          |                   |
|--------------------------|-------------------|
| EXCHANGE/NETWORK コマンドの使用 | DCLI-301          |
| 省略時の設定による定義              | DCLI-346          |
| ファイルの連結                  | DCLI-59, DCLI-119 |

#### ファイル保護

|            |          |
|------------|----------|
| ファイル作成時の定義 | DCLI-137 |
|------------|----------|

#### フェイルオーバー

|          |                    |
|----------|--------------------|
| キューの自動起動 | DCLI-361, DCLI-362 |
| 自動起動     | DCLI-374           |
| 自動起動キュー  | DCLI-250, DCLI-285 |
| 抑止       | DCLI-250           |

#### フォーリン・ボリューム

|      |          |
|------|----------|
| マウント | DCLI-573 |
|------|----------|

#### 不良ブロック・データ

|      |          |
|------|----------|
| ディスク | DCLI-355 |
|------|----------|

#### プリント・キュー

キューを参照

#### ブロック・サイズ・オプション

|                 |          |
|-----------------|----------|
| ディスクのクラス・サイズの指定 | DCLI-342 |
| ファイル用           | DCLI-244 |

#### プログラム

|                |          |
|----------------|----------|
| 実行の再開          | DCLI-101 |
| 実行の続行          | DCLI-101 |
| 入力ストリームの終わりを示す | DCLI-289 |
| 入力ストリームの先頭を示す  | DCLI-172 |

#### プロセス

##### ダンプ

|    |         |
|----|---------|
| 分析 | DCLI-49 |
|----|---------|

## プロセス (続き)

|                                        |          |
|----------------------------------------|----------|
| 入力ストリームの制御の切り替え . . . .                | DCLI-77  |
| ハイバネート                                 |          |
| ATTACH コマンド . . . . .                  | DCLI-77  |
| プロンプト                                  |          |
| コマンド・プロシージャでの表示 . . .                  | DCLI-376 |
| 文書変換                                   |          |
| 出力形式 . . . . .                         | DCLI-104 |
| 分析                                     |          |
| イメージ・ファイル                              |          |
| ページ・ブレイクの指定 . . . . .                  | DCLI-30  |
| イメージ・ファイルの . . . . .                   | DCLI-25  |
| イメージ・ファイルのフィックスアップ・セクシ<br>ョン . . . . . | DCLI-28  |
| イメージ・ファイル・パッチ・テキスト・レコー<br>ド . . . . .  | DCLI-30  |
| オブジェクト・ファイル                            |          |
| /DISASSEMBLE 修飾子 . . . . .             | DCLI-42  |
| グローバル・シンボル・ディレクトリ・レコ<br>ード . . . . .   | DCLI-42  |
| テキスト . . . . .                         | DCLI-47  |
| デバッグ情報レコード . . . . .                   | DCLI-42  |
| ページ・ブレイクの指定 . . . . .                  | DCLI-44  |
| モジュール・トレースバック・レコー<br>ド . . . . .       | DCLI-44  |
| モジュールの終端レコード . . . . .                 | DCLI-42  |
| モジュール・ヘッダ・レコー<br>ド . . . . .           | DCLI-43  |
| リンク・オプション指定レコード                        |          |
| . . . . .                              | DCLI-43  |
| レコードの再配置 . . . . .                     | DCLI-47  |
| オブジェクト・ファイルの . . . . .                 | DCLI-39  |
| オブジェクト・モジュールの . . . . .                | DCLI-39  |
| 共用可能イメージ・ファイルの . . . . .               | DCLI-25  |
| グローバル・シンボル・テーブル<br>の . . . . .         | DCLI-28  |
| 出力コンバータ . . . . .                      | DCLI-106 |
| 処理オプション . . . . .                      | DCLI-111 |
| 制限事項 . . . . .                         | DCLI-50  |
| ダンプ・ファイル . . . . .                     | DCLI-49  |
| パッチ・テキスト・レコード . . . . .                | DCLI-30  |
| フラグ値 . . . . .                         | DCLI-43  |

## へ

|                                |          |
|--------------------------------|----------|
| ヘッダの割り当て                       |          |
| ディスク・ボリューム . . . . .           | DCLI-347 |
| ヘルプの表示                         |          |
| 省略時の設定のライブラリの . . . . .        | DCLI-321 |
| ヘルプ・ファイル . . . . .             | DCLI-319 |
| ヘルプ・ライブラリ                      |          |
| 作成 . . . . .                   | DCLI-319 |
| ユーザ . . . . .                  | DCLI-324 |
| 変更                             |          |
| 省略時のコマンド言語インタプリ<br>タ . . . . . | DCLI-545 |

## ホ

### 保護

|                                        |                                 |
|----------------------------------------|---------------------------------|
| MOUNT コマンド . . . . .                   | DCLI-584                        |
| インターチェンジ環境 . . . . .                   | DCLI-349                        |
| 磁気テープ・ボリューム . . . . .                  | DCLI-352                        |
| ディスク初期化時の省略時の設定 . . .                  | DCLI-346                        |
| ディスク・ボリューム . . . . .                   | DCLI-352                        |
| ディレクトリ作成時の定義 . . . . .                 | DCLI-141                        |
| ファイル作成時の定義 . . . . .                   | DCLI-137                        |
| フォーリン・ボリュームの省略時の設<br>定 . . . . .       | DCLI-574                        |
| メールボックスに対する定義 . . . . .                | DCLI-146                        |
| 保持デバッグ . . . . .                       | DCLI-168                        |
| 補助制御プロセス                               |                                 |
| ボリューム・マウント . . . . .                   | DCLI-584                        |
| ボリューム                                  |                                 |
| ディスク・ボリューム, 磁気テープも参照                   |                                 |
| ANSI 標準とFiles-11形式 . . . . .           | DCLI-577                        |
| 書き込み保護 . . . . .                       | DCLI-592                        |
| 共用 . . . . .                           | DCLI-588                        |
| 公用作成 . . . . .                         | DCLI-589                        |
| 最大ファイル数の指定 . . . . .                   | DCLI-351                        |
| シャドウイング . . . . .                      | DCLI-353, DCLI-444,<br>DCLI-587 |
| シャドウイングによるマウント . . . .                 | DCLI-569,<br>DCLI-570           |
| 初期化 . . . . .                          | DCLI-339                        |
| 所有権 . . . . .                          | DCLI-589                        |
| 所有権の指定 . . . . .                       | DCLI-352                        |
| ダンプ . . . . .                          | DCLI-261                        |
| ディスク・ファイルの削除 . . . . .                 | DCLI-194                        |
| デスマウント . . . . .                       | DCLI-255, DCLI-591              |
| 非標準形式 . . . . .                        | DCLI-573                        |
| ファイル・ウィンドウのマップ・ポインタの割り<br>当て . . . . . | DCLI-591                        |
| 保護 . . . . .                           | DCLI-352                        |
| MOUNT コマンド . . . . .                   | DCLI-580,<br>DCLI-584           |
| 回復 . . . . .                           | DCLI-585                        |
| シャドウイングによるマウン<br>ト . . . . .           | DCLI-587                        |
| ユーザ・クォータ . . . . .                     | DCLI-585                        |
| マウンティング                                |                                 |
| オペレータ補助による . . . . .                   | DCLI-561                        |
| サブプロセスから . . . . .                     | DCLI-559                        |
| マウント . . . . .                         | DCLI-560                        |
| フォーリン . . . . .                        | DCLI-573                        |
| ラベル . . . . .                          | DCLI-339                        |
| ボリューム・セット                              |                                 |
| MOUNT コマンド . . . . .                   | DCLI-557                        |
| 作成 . . . . .                           | DCLI-564                        |
| デスマウント . . . . .                       | DCLI-259                        |
| ボリュームの追加 . . . . .                     | DCLI-564                        |
| ボリュームのアクセス可能フィールド                      |                                 |
| 文字の書き込み . . . . .                      | DCLI-349                        |



## メ

|                       |          |
|-----------------------|----------|
| メッセージ                 |          |
| オンライン・ドキュメント          | DCLI-328 |
| メッセージ・ロギング            |          |
| CONVERT/DOCUMENT コマンド |          |
|                       | DCLI-110 |
| メモリ                   |          |
| 修正                    | DCLI-219 |
| 内容の置換                 | DCLI-219 |
| 内容を表示                 | DCLI-292 |
| メールボックス               |          |
| 作成                    | DCLI-145 |
| 保護の設定                 | DCLI-146 |

## モ

|                    |         |
|--------------------|---------|
| モジュール・オブジェクト       |         |
| ファイルの終端レコードの分析     | DCLI-42 |
| 分析                 | DCLI-39 |
| モジュール・トレースバック・レコード |         |
| オブジェクト・ファイルの分析     | DCLI-47 |
| モジュールの終端レコードの分析    | DCLI-42 |
| モジュール・ヘッダ・レコード     |         |
| オブジェクト・ファイルの分析     | DCLI-43 |
| 文字列                |         |
| シンボル割り当て文          | DCLI-6  |

## ユ

|              |          |
|--------------|----------|
| ユーザ名         |          |
| ログイン時に指定     | DCLI-544 |
| ユーザ・ライブラリ    |          |
| ヘルプ・ライブラリを参照 |          |

## ヨ

|                    |          |
|--------------------|----------|
| 読み込みチェック           |          |
| APPEND コマンドによる     | DCLI-63  |
| COPY コマンドによる       | DCLI-125 |
| INITIALIZE コマンドによる | DCLI-343 |

## ラ

|                    |                    |
|--------------------|--------------------|
| ライブラリ              |                    |
| オブジェクト・モジュール       | DCLI-43            |
| ラベル                |                    |
| 関係するコマンド・インタプリタの規  |                    |
| 則                  | DCLI-80            |
| コマンド・プロシージャ        | DCLI-80,           |
| DCLI-313, DCLI-316 |                    |
| コマンド割り込み規則         | DCLI-313, DCLI-316 |
| 磁気テープ              | DCLI-557           |
| 磁気テープ・ボリューム・セット    | DCLI-557           |
| ディスク               | DCLI-557           |
| ディスク・ボリューム・セット     | DCLI-557           |

## ラベル (続き)

|             |          |
|-------------|----------|
| ボリューム・ヘッダ   | DCLI-339 |
| ボリュームへの書き込み | DCLI-339 |

## リ

|                  |          |
|------------------|----------|
| リソース・アフィニティ・ドメイン |          |
| RAD を参照          |          |
| リモート・ファイル        |          |
| 指定               | DCLI-134 |
| 利用者識別コード         |          |
| UIC を参照          |          |
| リンク・オプション指定レコード  |          |
| オブジェクト・ファイルの分析   | DCLI-43  |

## レ

|                  |          |
|------------------|----------|
| レキシカル関数          | DCLI-390 |
| 概要               | DCLI-388 |
| レコード             |          |
| 比較               | DCLI-224 |
| ファイルの終端の分析       | DCLI-42  |
| 分析               |          |
| グローバル・シンボル・ディレクト |          |
| リ                | DCLI-42  |
| 再配置              | DCLI-47  |
| デバッグ情報           | DCLI-42  |
| パッチ・テキスト         | DCLI-30  |
| モジュール・トレースバック    | DCLI-44  |
| モジュール・ヘッダ        | DCLI-43  |
| リンク・オプション指定      | DCLI-43  |
| レコード・サイズ         |          |
| 磁気テープを参照         |          |
| レコードの再配置         |          |
| オブジェクト・ファイルの分析   | DCLI-47  |
| レコード・ブロッキング      |          |
| ボリューム・マウント       | DCLI-577 |
| レディ・アクセス         |          |
| ディスクのディレクトリ      | DCLI-341 |

## ロ

|                  |                |
|------------------|----------------|
| ローカル・シンボル        | DCLI-2, DCLI-6 |
| ローカル・シンボル・テーブル   |                |
| シンボルの削除          | DCLI-217       |
| シンボルの入力          | DCLI-377       |
| ログアウト            | DCLI-549       |
| ログアウトと装置アクセス     | DCLI-17        |
| ログイン             | DCLI-544       |
| ログイン・コマンド・プロシージャ |                |
| 仮想ターミナルへの再接続     | DCLI-545       |
| 実行               | DCLI-544       |
| 別の名前の指定          | DCLI-545       |
| ロングワードでダンプ       | DCLI-265       |
| 論理キュー            | DCLI-360       |
| 割り当て解除           | DCLI-166       |

## 論理名

CLOSE コマンドを使用した論理名の解

- 除 ..... DCLI-94
- MOUNT での ..... DCLI-558
- 解除 ..... DCLI-161
- 作成 ..... DCLI-65, DCLI-175
- 装置への割り当て ..... DCLI-17
- テーブルの作成 ..... DCLI-148
- 割り当て ..... DCLI-65, DCLI-175

## 論理名テーブル

- エントリの削除 ..... DCLI-163
- 削除 ..... DCLI-161
- 作成 ..... DCLI-148
- システム・エントリの削除 ..... DCLI-164
- システム論理名の登録 ... DCLI-68, DCLI-178
- 論理名の登録 ..... DCLI-68, DCLI-178

## 論理名の登録

- グループ論理名テーブル ..... DCLI-67,  
DCLI-177

システム論理名テーブル ..... DCLI-68,  
DCLI-178

ジョブ単位の論理名テーブル ..... DCLI-67

ジョブ論理名テーブル ..... DCLI-178

プロセス論理名テーブル ..... DCLI-68,  
DCLI-178

## ワ

---

### ワーキング・セット

バッチ・ジョブ

省略時の定義 ..... DCLI-372

超過値の定義 ..... DCLI-372, DCLI-385

バッチ・ジョブのクォータの定義 ... DCLI-386

ワードでダンプ ..... DCLI-268

割り当て

会話型でのシンボルの ..... DCLI-376

キュー・オプション ..... DCLI-358

キュー名 ..... DCLI-358

論理キューを実行キューに ..... DCLI-75





HP OpenVMS DCL ディクショナリ：A-M

---

2005 年 4 月 発行

日本ヒューレット・パカード株式会社

〒140-8641 東京都品川区東品川 2-2-24 天王洲セントラルタワー

電話 (03)5463-6600 (大代表)

---

AA-R1EAF-TE

