ULT

O/S Revision 200 New Features

The Ultimate Corp.
East Hanover, NJ

Version 2

The Ultimate Operating System
Release 10 - Revision 200
New Features, Version 2

Publication Information

Document Number 15112-2

Contents

Notes

Preface

This document describes the new features of the Ultimate Operating System, Release 10, and is intended as a reference guide for all system users. The new features are available on all product lines except Tandem and ULT/ix.

This release includes the following revision numbers:

Revision 200	all implementations except IBM and 1400
Revision 204	IBM implementation
Revision 205	1400 implementation

This document contains information that supplements the Ultimate system documentation set. Readers of this document are assumed to be familiar with the Ultimate operating system and documentation. For complete information on existing features, please refer to the following Ultimate manuals:

- Assembly Language Reference Guide
- BASIC Language Reference Guide
- Recall and Update User Guide
- System Commands Reference Guide
- System Management and Support User's Guide
- UltiKit User Guide

Readers who are new to Ultimate can get acquainted with the system by reading the Beginner's Guide to Ultimate tutorial manual before using the features described in this guide.

Upgrade Considerations

Revision 200 of the Ultimate operating system requires the following:

- existing revision of operating system must be 180 or higher
- UltiCalc must be upgraded to UltiCalc 3.2
- Rev 200 systems using UltiNet require that all other systems on the network be at revision 190V or higher
- all user assembly programs must be reassembled after upgrade
- all user BASIC programs must be recompiled after upgrade

For complete information on upgrading to Revision 200, see the Ultimate Upgrade document.

How the Manual is Organized

Section 1 is an introduction and lists areas in which there are new features. Section 2 describes the changes to the operating system; Section 3 describes the changes to system commands. Section 4 describes the changes to the Ultimate implementation of the BASIC programming language. Section 5 describes the changes to UltiKit for multi-lingual capabilities. Sections 6 through 10 describe implementation specific changes. Appendix A contains an ASCII chart for reference; Appendix B contains a list of features that were introduced in Revision 190.

Conventions

The following conventions are used to describe the new features:

Convention	Description
UPPER CASE	Characters in syntax definitions printed in upper case are required and must appear exactly as shown.
lower case	Characters in syntax definitions printed in lower case are parameters to be supplied by the user.
{ }	Braces surrounding a parameter indicate that the parameter is optional and may be included or omitted at the user's option.
EXECUTE	The word EXECUTE in all upper case letters refers to the BASIC EXECUTE statement.
execute	The word execute in lower case letters refers to the execution of programs, statements, or commands
RETURN	The RETURN symbol indicates a physical carriage return pressed at the keyboard. A RETURN is required to complete a command line, and signals the system to begin processing the command.
<key>	Angle brackets are used to indicate a key other than letters or numbers; for example <ESC>.
enter	The word enter is used to mean "type in the required text, then press RETURN."
X'nn'	This form is used to define a hexadecimal number where 'nn' is the hex value; for example, X'0B', X'41', X'FF'.

1 Introduction

Revision 200 of the Ultimate Operating System contains the following enhancements:

- O/S enhancements
  - TERM-VIEW
  - character translation
  - LOGOFF process changes
  - system file changes
  - UPDATE changes
  - UltiKit changes
  - workspace linking change
  - tape subsystem changes
  - file indexing (B-trees)
  - assembler and system debugger changes
  - UltiMation changes
  - UltiWord changes
- New and enhanced system commands
- BASIC enhancements
- IBM-specific enhancements (Rev 204)
  - performance and scheduler enhancements
  - TERM-VIEW not supported
  - Phase II HIFAS controller enhancements
  - Series 1 terminal support subsystem
  - security based on system serial number
  - block-size option
- LSI11-specific enhancements
  - ECP diagnostics option
  - EECP diagnostics option
- Ultimate 6000/7000 series-specific enhancements
  - disk shadowing
  - faster file save capability
  - GCR and Pertec tape drive enhancements
- VAX-specific enhancements
  - changes to installation and startup procedures
  - enhanced error logging and recording
  - enhancements for tape attachment
  - on-line help for Ultimate command qualifiers
  - shutdown enhancements
  - software disk cache
  - spooler enhancements
  - 16 parallel printers
  - 124 interactive processes for ECP
- 1400-specific enhancements (Rev 205)
  - 2K data frames
  - modem control

Notes

2 O/S Enhancements

The O/S enhancements are for all product lines except the Tandem and ULT/ix, unless otherwise noted.

- TERM-VIEW
- character translation
- LOGOFF process changes
- system file changes
- UPDATE changes
- UltiKit changes
- workspace linking change
- tape subsystem changes
- file indexing (B-trees)
- assembler and system debugger changes
- UltiMation changes
- UltiWord changes

TERM-VIEW

One terminal can be logically connected to a second terminal; information entered at either terminal is accepted as input to the process on the second terminal and is echoed on both terminals. Output from the process on the second terminal is displayed on both terminals.

The two ports remain logically connected until the connection is broken by pressing <ESC>, then the exit character.

Revision 200 provides several new system (TCL) commands designed for terminal viewing process:

TERM-VIEW	sets up connection
TERM-VIEW-OFF	locks port against TERM-VIEW
TERM-VIEW-ON	unlocks port

These commands are described in Section 3, System Commands.

Character Translation

Translation tables can be created and used to translate characters input to or output by the system. Any character or sequence of characters can be translated to any other character or sequence of characters.

Revision 200 provides several new system (TCL) commands designed for the translation process:

TRANSLATE-INPUT	defines a translation table
TRANSLATE-LOAD	makes the table available to a port.
TRANSLATE-ON	enables the actual translation to start
TRANSLATE-OFF	stops the translation.

These commands are described in Section 3, System Commands.

LOGOFF Process Changes

When DSR (Data Set Ready) drops on a line, the system now drops the DTR (Data Terminal Ready) signal and sets NOCRT for the process. This ensures that the line to the computer will be logged off and that no one can log onto the line until logoff is completed. Once logoff is completed, the system raises DTR and clears NOCRT.

System File Changes

ERRMSG and PROCLIB

The system files ERRMSG and PROCLIB now have three levels, with multiple data levels. The multiple data levels are provided to allow a variety of languages to be used. The default data level for ERRMSG is ERRMSG,US and for PROCLIB, it is PROCLIB,US (US indicates United States; that is, English as used in the United States).

The default data levels are actually Q-Pointers to the dictionary of the ERRMSG file and PROCLIB file. This provides compatibility with previous revisions of the operating system.

A new code has been added to the ERRMSG process: AM. This inserts an attribute mark in the message.

LOGON and NOLOG Items

The items LOGON and NOLOG that used to be in the SYSTEM file have been moved to the dictionary of ERRMSG file (DICT ERRMSG). The LOGON item contains the welcome message and logon prompt for users; the NOLOG item contains the message to be used when users cannot log on.

NEWAC

The SET-BAUD and PASSTHRU commands are now part of the NEWAC file and are included in all user accounts.

USERMSG File

A new system file, the USERMSG file, has been added. The USERMSG file is designed as a multiple data level file with a level for each language translation on the system. The USERMSG file enables system users to create custom messages for their applications, which can then be translated by the UltiKit multi-lingual process.

The USERMSG file can be used with the BASIC USERTEXT function, which is described in Section 4, BASIC Enhancements. Items can also be printed using the system command PRINT-ERR. The format of item IDs in the USERMSG file is up to the user.

Each line in a USERMSG item must conform to a general format:

```
code{text}
```

The valid codes are as follows:

Code	Meaning
A	inserts the next parameter from the list of parameters passed by USERTEXT
A(n)	inserts the next parameter as above, but left justified in a field of 'n' blanks
AM	inserts attribute mark
D	inserts the current date
E	inserts the item.id enclosed in brackets
H	inserts the text following the H; does not include a CR/LF
H+	used at the end of the USERMSG item only to suppress final CR/LF that is normally output
L	inserts CR/LF
L(n)	inserts n-1 blank lines
R(n)	inserts the next parameter as A (above), but right justified in a field of 'n' blanks
S(n)	Sets the output buffer pointer to location 'n'
T	inserts the current time
X	Skips a parameter in the list of parameters passed by USERTEXT

```
Sample USERMSG item

   item.id  Welcome
      001   E
      002   H,
      003   A
      004   L
      005   HIt is
      006   D
```

Update Changes

The Update processor has been modified as follows:

- integration of multi-lingual concepts when printing error and help messages
- two additional values are available to the common subroutine variable C.CLEARSCREEN that allow refreshing data without clearing entire screen
- a question mark (?) is displayed in the lower left corner of a screen when additional help is available
- UPD-DEF has been revised

UltiKit Changes

The UltiKit processor has been modified as follows:

- subroutines and functions now obtain messages from ERRMSG file to provide multi-lingual capabilities
- menus have been revised; information on the menus is being incorporated into the UltiKit manual.

Note: *For information on the multi-lingual capabilities, see Section 5, Multi-Lingual Features.*

Workspace Linking Change

The time when workspace is linked has been modified. If a user attempts to log on to the system after a restore while the spooler is linking workspaces, that user's workspace is now linked immediately and the log on completed.

Tape Subsystem Changes

The tape subsystem has been changed to provide more meaningful messages and to allow the operator more options when an error is encountered.

Specifying Options

The messages produced by the tape subsystem now require that RETURN be pressed before the system will continue. Previously, the system started immediately after any key was pressed.

Mount Next Reel

If the operator enters C to continue with next reel before the tape is on-line, the following message is displayed:

```
Tape unit off-line
```

The continue/quit prompt is redisplayed.

Block Size

If an attempt is made to transfer a block that is less than the minimum size, the following message is displayed:

```
Invalid size - block skipped
```

The minimum block size is 19 bytes, unless the ANSI inhibit option in the T-READ verb is specified, in which case the minimum block size is 1.

Write Errors

The continue/quit message displayed when a write error is encountered has been replaced. On unbuffered devices where a retry is possible, the following message is displayed:

```
(R)etry/(Q)uit
```

On all other devices, the following message is displayed:

```
(A)ccept/(Q)uit
```

On buffered devices where retries are possible, the A option executes the retry logic.

Read Errors

The tape subsystem has been modified to provide more information and to allow the tape movement to be manually controlled when a read error is encountered on the tape.

The following information is displayed after a read error has occurred:

```
Tape read error : Status1 = xxxx  Status2 = yyyy  Bytes = zzzz
```

where

xxxx	status word A
yyyy	status word B
zzzz	number of bytes read in from current block

Note: *The system cannot determine if a tape error is non-recoverable. If a nonrecoverable tape error does occur, these procedures will have no effect.*

When a tape read error is first encountered, the following options are offered:

```
(A)ccept/(R)etry/(Q)uit/(F)orward/(B)ackward
```

A	accept error and continue to read block
R	retry; if system cannot retry, for example, because drive is buffered, prompt is redisplayed
Q	exit the tape read
F	forward space
B	backspace

The F (forward space) and B backspace options allow the user to manually control the tape movement to determine where the error was and to resume reading the tape at a specific location.

Manual Control

If either F or B is entered, the following prompt is displayed:

```
Enter number of blocks or (S)can to EOF :
```

A number in the range 0 to 99 can be entered; 0 specifies no tape motion. S specifies to scan to end of file (EOF). In no case will the system scan past an EOF mark.

When the specified number of blocks has been read, or when an EOF, end of tape (EOT) or beginning of tape (BOT) has been reached, the system displays the block on which the tape has stopped in a T-DUMP hexadecimal format. The block number and block size are displayed, similar to the following:

```
Block# = bbb     Size = sss
```

where

bbb	number of blocks forward (positive number) or backward (negative number) from the block that caused the error
sss	size of block in number of bytes

To terminate the display, press <CTRL-X>. The following prompt is displayed:

```
(R)esume at this block/(D)isplay next/(F)orward/(B)ackward
```

R resume tape read activity; depending on the condition of the tape at that point, additional error messages may be displayed; for example

```
invalid size - block skipped
tape format error - segment skipped
```

It is the user's responsibility to recover the missing data.

D read and display the next tape block

F forward space; the number of blocks prompt is displayed as described above

B backspace; the number of blocks prompt is displayed as described above

File Indexes

As of Revision 200, file indexing is available. A file index is a sorted set of data based on attributes in the file, which can be used as an alternative means of accessing the data in the file. Once an index is created, the system automatically updates all indexes associated with a file whenever the file itself is updated.

Revision 200 provides several new system (TCL) commands designed to handle file indexes:

CLEAR-INDEX-LOCKS	clear index locks
CREATE-INDEX	create a new index
DELETE-INDEX	delete an index
LIST-INDEX-LOCKS	list the status of index locks
LIST-INDEXES	list the indexed attribute values

These commands are described in Section 3, System Commands.

Overview of File Indexing

File indexes provide a permanent, presorted, up-to-date means of accessing the items in that file and are an efficient way to retrieve data from large files when frequent sorting or selection criteria are used.

Before Release 200, you could create and save lists of item-ids in order to cut down future selecting and sorting time. However, the list was not kept current; if the file changed after the list was created, you either had to create a new list or edit the existing list. Indexes, on the other hand, are kept up to date by the system itself. Once an index is created, the system automatically updates all indexes associated with a file whenever the file itself is updated.

The use of indexes in day-to-day file reporting operations is transparent to the user. You never need to request that an index be used. The system automatically uses them to access a file when a Recall command that specifies one BY and/or WITH clause using an indexed attribute is entered.

An index can be on any attribute or combination of attributes, such as vendor in the parts file or employee name and address in an employee file. The index is created from a single Update definition item (dictionary item created by the UPDATE processor); however, the Update definition item can combine several attributes (see the section, Multiple Attribute Criteria).

A file may have as many indexes as it has Update definition items. Each index is sorted in ascending sequence in the order of the attribute used to create the index.

An index may be created at any time using the new TCL command CREATE-INDEX. Creating an index takes about the same amount of time as does sorting the file.

Creating Indexes

To select candidates for indexing, review the purpose and usage of each of your files. If you have large files that are often sorted by one or more specific attributes, those attributes should be considered for indexing.

Each attribute to be indexed must be defined by an Update definition item, which is created using the system command UPD-DEF. If you indicate to UPD-DEF that the attribute being defined is an Indexed Field, UPD-DEF will automatically create the index. You can use this feature for both new and existing Update definition items.

You can also create an index from an existing Update definition item by using the system command CREATE-INDEX. To list existing indexes, use the command LIST-INDEXES. For more information on these and other system commands to use with indexes, see Section 3, System Commands. (For information on creating Update definition items, see the *Ultimate Recall and Update User Guide*).

In theory, any Update definition item can be used to create an index. However, the use of certain values in an Update definition item prevents it from being used to create an index. The following attribute numbers and associated values are not valid for indexes (the attribute number refers to the attribute in the Update definition item):

Attribute Number	Values	Description
4	D	Dependent status
8	B	BASIC subroutine
	LPV	Load previous value
	NI	Current item counter
	ND	Number of detail lines
	NV	Multi-value counter
	NS	Multi-subvalue counter
	NB	Current BREAK level counter
	P	Pattern match
	R	Range match
	S	Substitute value
	T	File translation
	U	User exit

Also, if attribute 8 has an A, F, or C processing code, attribute 2 must have an AMC of 0; otherwise, the index cannot be created from that attribute.

Multiple Attribute Criteria

An index is created from just one Update definition item. However, if you need an index based on several attributes, you can create an Update definition item that uses the required attributes, and use that Update definition item to create an index. That is, the way to take advantage of an index when sorting by multiple attributes is to build the index key from the multiple attributes.

For example, the following statement produces a report in vendor order and date sub-order (date within vendor):

```
SORT PO BY VENDOR BY DATE VENDOR DATE
```

The same result can be returned by creating a single Update definition item, then sorting on that attribute:

```
IVD  item-id
001  U
002  0
...
008  A;N(VENDOR)(ML#15):N(INTERNAL.DATE)
...
```

The above A processing code uses a mask to format the VENDOR attribute within a fixed length field so that the internal date field aligns properly for sorting. Also note that 0 (zero) is specified as the AMC; this is a requirement if an A processing code is used as a correlative.

With this index, the following command produces the same report as the SORT command above:

```
SORT PO BY IVD VENDOR DATE
```

Note: *When using a mask, be aware that the mask character becomes part of the "key" in the index. Therefore, be as conservative as possible; if the desired results can be obtained with the mask ML#15, do not waste space by using ML#40.*

Single- and Multi-Valued Attributes

Data files typically contain both single-valued and multi-valued attributes. The Update definition item for each attribute indicates whether the data in that attribute is single- or multi-valued. Indexes for single-valued attributes are created differently from indexes for multi-valued attributes.

If the Update definition item does not define the attribute as multi-valued, the create index process assumes the attribute is single-valued and one index entry is created for each attribute, even if the attribute is multi-valued.

If an Update definition item defines the attribute as multi-valued, one index entry is created for each value in the attribute. If the attribute is not defined in the Update definition item as multi-valued, only the first value is indexed.

A Recall statement that uses the index to generate a report would not produce the expected report if the attribute was not defined as multi-valued in the Update definition item.

For example, assume that a file called FRUITS contains one item called APPLE with one multi-valued attribute called VARIETY (the] indicates a value mark):

```
item-id  APPLE
    001  GRANNY]DELICIOUS]ROME]JONATHAN
```

If the Update definition item for VARIETY does not specify that VARIETY is multi-valued, the index would have one entry, GRANNY. The command

```
SORT FRUITS WITH VARIETY "ROME" VARIETY
```

results in the message

```
[401]  No items present.
```

If VARIETY was specified as being multi-valued, the index would be created with four entries and the command

```
SORT FRUITS WITH VARIETY "ROME" VARIETY
```

results in the report

```
FRUITS...  VARIETY...........

APPLE      GRANNY
           DELICIOUS
           ROME
           JONATHAN
```

How Recall Uses Indexes

Indexes are used by Recall for two phases of Recall command processing: selection (a WITH clause) and sorting (a BY clause).

Without indexes, Recall steps through the file one item at a time to produce the requested report. On large files, this can be very time-consuming. However, whenever an attribute on which an index is based is specified for a sort or selection, the index associated with that attribute is used to process the Recall command. Because the index is in sorted order, the time required to process the command can be reduced dramatically.

Once an index is created, Recall can access that index using any attribute definition item that is for the same attribute number and uses the same correlatives and justification code as the Update definition item.

For an index to be used to process a Recall command, the specified attribute must have an index and the Recall sentence must meet one of the following conditions:

- selection using one WITH clause
- sorting using one BY clause
- selection using one WITH clause and sorting using one BY clause

If one indexed attribute is specified for selection, and a second indexed attribute is specified for sorting, the index associated with item selection (WITH clause) is used; the index for the sort attribute is not used.

The Recall command may specify a sort of an index-attribute in either ascending or descending order. The BY modifier used in the command for the index attribute must match the index (that is, single or multiple valued index-attribute). An index built for single valued attributes (one index entry per item) must use BY or BY-DSND; an index built for multiple valued attributes (multiple index entries per item, if applicable) must use BY-EXP or BY-EXP-DSND. If the BY modifier does not match the index, the index is not used by the Recall command.

The BY modifiers are:

BY and BY-EXP	ascending sorts
BY-DSND and BY-EXP-DSND	descending sorts

Below are examples of Recall commands that use an index.

Selecting

To use an index to select items, the Recall command can contain only one WITH modifier, but may have multiple criteria for the WITH index-attribute. The index is used to match and select items that meet the criteria. For example, the following statement uses an index if one is present for STATE:

```
LIST NAMES WITH STATE = "CA" "AZ" STATE
```

Sorting

To use an index to sort items, the Recall command may contain only one BY modifier. The following statement uses an index if one is present for STATE:

```
SORT NAMES BY STATE
```

If a multi-valued attribute with an index is specified in a Recall command, the index is used only if the BY-EXP modifier is also present. For example, if the Update definition item specifies that COMPANY is multi-valued, the following statement uses an index if one is present:

```
SSELECT NAMES BY-EXP COMPANY
```

Selecting and Sorting

To select and sort items using an index for both operations, the Recall command must contain one WITH and one BY clause for the same indexed attribute. This command uses an index if one is present for STATE:

```
SORT NAMES BY STATE WITH STATE "CA" "AZ"
```

To select and sort multi-valued index items using an index for both operations, the Recall command must contain one WITH and one BY-EXP clause for the same indexed attribute. For example, the following command uses an index if one is present for COMPANY:

```
SORT NAMES BY-EXP COMPANY WITH COMPANY "ABC"
"PQRS"
```

The index is used to match and select items that meet the criteria (ABC or PQRS company). The selected items are already in COMPANY-sorted order; the report is in COMPANY major order (all ABC before all PQRS) and item-id minor order (Jones before Smith).

If the Recall statement specifies a WITH clause using one indexed attribute and a BY clause using a different indexed attribute, selection uses the index, but sorting does not. For example, the following statement uses the index to select the companies, but not to sort them:

```
SORT NAMES BY STATE WITH COMPANY "ABC" "PQRS" STATE
COMPANY
```

Differences between Reports using Indexed and Non-Indexed Multi-Values

When you specify both selection criteria and sort specifications for a multi-valued attribute, the resulting report differs, depending on whether or not the attribute is indexed. If the attribute is indexed, only the specified multi-value is listed; however, if the attribute is not indexed, the report lists each multi-value in the attribute.

For example, if VARIETY is indexed, the command

```
SORT FRUITS BY-EXP VARIETY WITH VARIETY "ROME"
VARIETY
```

results in the report

```
FRUITS...  VARIETY...........

APPLE      ROME
```

However, if VARIETY is not indexed, the command

```
SORT FRUITS BY-EXP VARIETY WITH VARIETY "ROME"
VARIETY
```

results in the report

```
FRUITS...  VARIETY...........

APPLE      DELICIOUS
APPLE      GRANNY
APPLE      JONATHAN
APPLE      ROME
```

Without indexing, Recall requires special codes to suppress the listing of the extra values.

Limitations to Use of Indexes

Recall uses an index whenever possible for item selection and sorting, as determined by the parameters used in the Recall command. However, there are a few circumstances where Recall cannot use an index, even if one exists.

Currently, indexes are not used in the following cases:

- more than one WITH modifier is specified
- more than one BY modifier is specified
- explicit item-ids are listed
- a select list is active
- the Recall command is REFORMAT or SREFORMAT
- an attribute is selected rather than item-ids
- the tape option is specified

Assembler Changes

There have been several changes to the Assembler for Revision 200.

Reassembly

All existing assembly language programs **must be reassembled** before they can be used with Revision 200. The MLOAD command will not load any programs that are not in Revision 200 format.

Return Stack Changes

The following return stack enhancements have been made:

- the depth of the return stack has been extended from 11 to 125
- the first two return stack entries are now reserved for the system to handle return stack full and return stack empty processes
- the system can mark the return stack, which allows entries to be grouped logically into multiple stacks

New instructions have been added to access the return stack; these instructions replace the specified existing instruction:

INITRTN	initializes return stack; replaces MOV X'184',RSCWA instruction
POPRTN	pops one entry off the return stack; replaces DEC RSCWA,4 instruction

Any existing DEC RSCWA,4 instructions **must** be changed to POPRTN; any existing MOV X'184',RSCWA instructions **must** be changed to INITRTN.

I/O Flag Bits

The I/O flag bits have been moved out of the PCB; to access these bits, new subroutines have been added.

The following I/O flag bits are in T0 (T zero); currently, all other bits in T0 are ignored:

PSYM Name	**Previous Name**
B15	CCDEL
B14	TITFLG
B13	FRMTFLG

The new subroutines that access the flag bits are

ANDIOFLGS	sets one or more I/O flag bits to false
Input Interface:	T0: bits B15, B14, B13. A zero (0) value sets the corresponding bit to false; a one (1) value maintains the current setting
Output interface	T0
Element use:	D0, R14, R15

Maintaining Integrity of Indexes

The system automatically maintains the integrity of indexes. For example, only one process at a time can update an index. If an index is being updated by one process, the system locks it and no other process can access that index, either to update it or to read from it.

Also, if a process is reading from an index, that index cannot be updated by another process. An index can be read by several processes at the same time, but it is locked against updates and can be changed only when no other process is using it.

If a process tries to access a locked index, the process is paused until the index is unlocked.

File Saves and Restores and Indexes

Indexes are saved and restored along with the rest of a file unless the following options are used in the SAVE command:

Option	Meaning
B	Backward compatible. This option is used when the file save will be restored to a pre-190 system. No Rev 190 and above file information is saved to tape, and therefore, no indexes are saved.
E	No indexes saved to tape.

GETIOFLGS	gets the current I/O flag settings
Input Interface:	none
Output interface	T0
Element use:	T0, R14

ORIOFLGS	sets one or more I/O flag bits to true
Input Interface:	T0: bits B15, B14, B13. A one (1) value sets the corresponding bit to true; a zero (0) value maintains the current setting
Output interface	T0
Element use:	D0, R14, R15

Input/Output Routines

The opcodes that read and write single characters have been rewritten as subroutines because of requirements for the new system features TERM-VIEW and character translation. The old opcode names are no longer valid and must be changed.

READX@IB	input one character at the current position of R10 and do not echo character; replaces READXR opcode
Input interface:	R10 at the character location
Output interface	R10 unchanged; ~~R14~~
Element use:	R14, FP0
Usage:	this subroutine can be used to replace the READXR opcode

READ@IB	input one character at the current position of R10 and echo data character, if echo is on; replaces READR macro
Input interface:	R10 at the character location
Output interface	R10 unchanged; ~~R14~~
Element use:	R14, FP0
Usage:	this subroutine can be used to replace the READR opcode

WRITEX@OB	output one character at the current position of R11, unless current terminal is IBM 3270 type, in which case, output nothing; replaces WRITEX opcode
Input interface:	none
Output interface	R11 unchanged; R14
Element use:	R14

WRITE@OB	output one character at the current position of R11; replaces WRITE opcode
Input interface:	none
Output interface	R11 unchanged; R14
Element use:	R14

Note: *The old opcodes were saved and renamed. You have the option to use the new name of the old opcode; however, if you choose to do this, your programs cannot be used with the TERM-VIEW and character translation features. The following list shows the standard and optional new names and the old names:*

Standard New Name	Old Name	Optional New Name
READ@IB	READR	INP1B
READX@IB	READXR	INP1BX
WRITE@OB	WRITE	OUT1B
WRITEX@OB	WRITEX	OUT1BX

Standard I/O Routines

The standard input and output routines that read and write lines have been changed internally to use the standard new names and require no user changes. These instructions include

GETBUF
READIB
READLIN
READLINX
WRITLIN
WRITOB

The number of bytes in the input buffer used by READLIN is now IBSIZE plus three. The three bytes are reserved at the end of the input buffer for BACKSPACE and <CTRL-W> processing.

Notes

UltiMation Changes

Several changes have been made to UltiMation, the Ultimate office automation tool.

General Changes

Responses may now be entered in upper or lower case.

When a user on one of the following terminals receives a message or letter, the terminal bell sounds:

Terminal name	Terminal type
Wyse50	W
Wyse60	S
Wyse85	Y

Main Menu Changes

The exit routine from the Main Menu now executes a PROC in the account master dictionary called OA-QUIT. You can modify this PROC as desired; it can be used for example, to return users to an application menu. The default PROC returns to TCL.

The default OA-QUIT PROC is automatically set up by the upgrade procedure.

Note Pad Changes

The TO DO list can now be sorted by priority and action date as well as by priority and status. When option 3, TO DO list, is selected, the following options are displayed:

```
TO DO List Print Selection

1. By Priority By Status
2. By Priority by Action Date
```

Select the desired order for the report.

Telephone Messages Changes

Help is now available at the Telephone Message Selector menu by entering a question mark (?) at the selection prompt.

To acknowledge the receipt of a message, you may enter either A or ACK.

The Acknowledged message retrieval report now displays the time the message was received.

The Enter telephone messages function now displays a message similar to the following to confirm that the message was sent:

Message has been sent to '*username*'

Utility Menu Changes

The telephone extension entry in the User entry and maintenance option no longer has a minimum of three digits. One or two digits may be entered if desired.

The pager number entry in the User entry and maintenance option can now have up to ten digits.

Slave printers can now be specified as the default output device. To specify this, select option 5, Port Entry and Maintenance. At the default output device prompt, enter the letter S. This causes all output that is directed to a printer to go to the slave printer, rather than the system printer. To redirect output to another printer, change the letter S to the number of the desired form queue.

UltiWord Changes

Several changes have been made to UltiWord, the word processing package.

Prompts

Prompts now use dots to indicate the maximum number of characters that can be entered at a prompt. The dots are displayed when the cursor is on the prompt line. For example, the main menu prompt to enter your user name now displays the default name, followed by dots to fill the prompt to twenty characters, similar to the following

```
YOUR USER NAME:WP..................
```

Error Messages

The error messages displayed by the Copy and Utility options now provide more information when an error occurs. For example, in the Copy option, if the specified user to copy from does not exist, a message similar to the following is displayed:

```
"user" not found!
```

3 System Commands

The following new commands have been added to the system (TCL) level of operation for Revision 200:

CLEAR-INDEX-LOCKS
CREATE-INDEX
DELETE-INDEX
LIST-INDEX-LOCKS
LIST-INDEXES
LIST-NAMED-COMMON
LIST-VSAVE-STATS
SET-LANGUAGE
TERM-VIEW
TERM-VIEW-OFF
TERM-VIEW-ON
TRANSLATE-INPUT
TRANSLATE-LOAD
TRANSLATE-OFF
TRANSLATE-ON
VERIFY-SAVE

The following commands have been enhanced:

LOGOFF
LOGON
READ-STATUS
SAVE
SET-LOGOFF
T-ATT
T-STATUS
WHERE

These commands are detailed on the following pages.

The following command has been deleted from the system and is no longer available:

:RESTARTLINE

CLEAR-INDEX-LOCKS - New

The CLEAR-INDEX-LOCKS command unlocks all indexes for a specified file.

Syntax

```
CLEAR-INDEX-LOCKS file.name
```

file.name	file in which the index is located

Description

The CLEAR-INDEX-LOCKS command is used to unlock all index locks currently set for the file. Its purpose is to allow manual unlocking of locks.

This command is intended for customer support personnel. The system normally unlocks index locks; however, if a process that uses indexes comes to some abnormal ending, it is possible that an index will not be unlocked. In that case, this command can be used to unlock the index. (Index locks can be listed using the system command LIST-INDEX-LOCKS, which is described in this section.)

The account must have SYS2 privileges to use this command.

There are no messages. Any locks in any index in the file are automatically unlocked. The system returns to TCL.

Provided On

Any User Account

See Also

LIST-INDEX-LOCKS

CREATE-INDEX - New

The CREATE-INDEX command builds an index for a file, using a specified Update definition item to build the key structure on which the index is based.

Syntax

CREATE-INDEX {file.name {attrib.name}}

file.name	name of file for which index is to be created; if file.name is not specified, it is prompted for
attrib.name	name of Update definition item on which to base index; if attrib.name is not specified, it is prompted for

Description

The account must have SYS1 or higher privileges to use this command.

CREATE-INDEX takes information from the specified file and Update definition item to build a key structure, which it then uses to create the actual index keys. An index key structure contains the following information:

- attribute number
- attribute correlative
- attribute justification
- item-id justification (from file definition item)

This key structure becomes a permanent part of the index.

After the key structure is built, CREATE-INDEX uses it to build the index and the system uses it to maintain the index. The system does not rely on the Update definition item itself.

The benefits of this approach are:

1. Integrity of the index is maintained. If the Update definition item is subsequently changed or deleted, the index is still updated correctly, according to the original attribute definition.
2. Since Recall matches the criteria in the Recall command to the key structure, any number of synonym attribute definition items that match the key structure can use the corresponding index.

Note: *Although the index is not hurt by changing or deleting the Update definition item, Recall cannot use the index if it does not find an Update definition item that matches the key structure. Also, Recall cannot use the index if the justification of the item-id is changed, since it it also part of the key structure.*

Any attribute that is defined with the same elements as were used to create the index is considered to be an indexed attribute, whether or not it is the original Update definition item.

Justification of the item-id is part of the key structure because Recall always sorts by item-id as a secondary sort after any sort criteria specified in the command; the effective sort is

```
SORT BY attribute (BY item-id)
```

If the Update definition item defines the attribute as multi-valued, one index entry is created for each value in the attribute. If the attribute is not defined as multi-valued, one index entry is created for each attribute.

Provided On

Any User Account

See Also

DELETE-INDEX
LIST-INDEXES

DELETE-INDEX - New

The DELETE-INDEX command deletes the index for the specified attribute in the specified file.

Syntax

DELETE-INDEX {file.name {attrib.name}}

file.name	name of file from which index is to be deleted; if file.name is not specified, it is prompted for
attrib.name	name of Update definition item on which to base index; if attrib.name is not specified, it is prompted for

Description

The DELETE-INDEX command is used to delete an existing index.

The account must have SYS1 or higher privileges to use this command.

If the specified index name is not in the file, an error message is displayed and the system returns to TCL.

'attrib' not on file.

When an index is successfully deleted, no message is displayed and the system returns to TCL.

Provided On

Any User Account

See Also

CREATE-INDEX
LIST-INDEXES

LIST-INDEX-LOCKS - New

The LIST-INDEX-LOCKS command lists the lock statuses for each index associated with a specified file.

Syntax

LIST-INDEX-LOCKS file.name

file.name	name of file whose index locks are to be listed

Description

The system locks an index as follows:

- if the index is being read by one user, it is locked against being updated; however, it can be read by other users
- if the index is being updated by one user, it is locked against being read or updated by all other users

The LIST-INDEX-LOCKS command displays the lock status on the screen, then returns to TCL. The report is similar to the following:

```
Attr#   Index-L    Write   Read-ctr.
------  ---------- ------- -----------
    2       -        37         0
-0001       -         -         0
    1       -         -         4
```

The Attr# column shows the attribute number. A negative attr# indicates that the index is a generated value based on a correlative. In the example, the second index is based on a generated value, as indicated by an attr# of -0001.

The Index-L and Write columns display the port number of a user who is currently setting the overall Index-Lock for that attribute (Index-L) or updating the index (Write). In the example, the user on port 37 is currently updating information for the index on attribute 2.

The Read-ctr. column displays the current number of users who are accessing (reading) the index. In the example, four users are currently accessing information from the index based on attribute 1.

Provided On

Any User Account

See Also

CLEAR-INDEX-LOCKS

LIST-INDEXES - New

The LIST-INDEXES command lists all the indexes for a specified file.

Syntax

LIST-INDEXES file.name

file.name	name of file for which the indexes are to be listed

Description

The LIST-INDEXES command displays the report on the screen, and returns to TCL. The information displayed includes the information used in the key structure plus the name of the Update definition item specified in the CREATE-INDEX command.

The report is similar to the following:

```
                                           Attr   Id's   Multi-
Attr#  Attr.name    Correlative            Just.  Just.  Value
------ ------------ ---------------------- -----  ------ ------
     2 MANUFR                              L      R
     0 CS           A;(2:",":3)            L      R
     1 COLORS                              L      R      Y
```

Provided On

Any User Account

See Also

CREATE-INDEX
DELETE-INDEX

LIST-NAMED-COMMON - New

The LIST-NAMED-COMMON command lists the BASIC COMMON areas that are currently named.

Syntax

LIST-NAMED-COMMON

Description

This command lists the named COMMON areas for the current port.

For information on named COMMON areas, see Section 4, Extended BASIC.

```
   LIST-NAMED-COMMON

Result:
   /CTR/
```

The current port has one named common area called CTR.

LIST-VSAVE-STATS - NEW

The LIST-VSAVE-STATS command generates a report based on the VSAVE-STATS file created by the last VERIFY-SAVE.

Syntax

LIST-VSAVE-STATS {LPTR} {132}

LPTR	Directs output to printer
132	Puts terminal in 132-column mode

Description

The LIST-VSAVE-STATS report contains a summary of the information on the file save tape, similar to the STAT-FILE report produced by the save process. In addition, the command compares the information in the STAT-FILE with the information in the VSAVE-STATS file and indicates on the report any differences.

The information that is displayed includes

reel number
file number
accountname{*dictname{{*dataname}}
number of items in file on tape, as determined by VERIFY-SAVE
number of bytes in file on tape, as determined by VERIFY-SAVE
number of items in file on disk, as determined from STAT-FILE
number of bytes in file on disk, as determined from STAT-FILE
number of item-size errors
number of tape format errors
number of object item errors

If the number of items or bytes on tape differs from the number on disk, an asterisk is appended to the display of the number on disk.

LOGOFF - Enhanced

The LOGOFF command has been enhanced and now gives you the following options:

- to specify a range of lines to log off at the same time
- to indicate the number of seconds to wait for completion of the logoff
- to suppress messages
- to log off trapped processes

Syntax

LOGOFF {line.nos} {(options}

line.nos the lines to be logged off; can be one of following

n	logs off line n
n-m	logs off lines in range n through m, inclusive

options the following options are available

t	number of seconds to wait for the logoff to complete; the default value is 20 seconds. A value of 0 causes the system to return to the user issuing the command immediately after posting a logoff condition to the specified lines; the maximum value of t is 60.
S	suppresses all terminal output on the lines to be logged off until the LOGON message is displayed
U	untraps a trapped process and then logs off the line; if not specified, a trapped line is not logged off and a message is displayed on the terminal of the user issuing the LOGOFF command

Description

A SYS2 privilege level is required to execute LOGOFF.

Caution! The LOGOFF command can cause data to be lost, depending on the activity on the specified line.

The LOGOFF command is equivalent to pressing the BREAK key, then entering OFF in the debugger and should be used only under drastic conditions. If the BREAK-KEY-OFF command has been executed, the line cannot be logged off until a BREAK-KEY-ON command has been executed.

LOGOFF waits until it is safe for the user to be logged off; this means that the user can be logged off without causing GFEs or corrupting the overflow table.

If no parameters are specified, the following prompt is displayed:

```
Enter processes and (options) :
```

Enter the line number of the processes to log off. This prompt is provided to maintain compatibility with the existing LOGOFF command.

When the logoff is completed, a message similar to the following is displayed:

```
[534] Successful logoff of process : n
```

A new message has been added to the LOGOFF process. If the number of seconds to wait for logoff is exceeded, a message similar to the following is displayed:

```
[577] Logoff posted for process : n
```

The line is logged off as soon as possible. For example, if the line is in the middle of updating the file, the LOGOFF command must wait until the update is complete.

LOGON - Enhanced

The LOGON command has been enhanced and now gives you the following options:

- to specify a range of lines to log on at the same time
- to indicate the number of seconds to wait for completion of the logon
- to suppress messages

Syntax

LOGON {line.nos,acct.name{,password}} {(options}

line.nos	the lines to be logged on; can be one of following

n	logs on line n
n-m	logs on lines in range n through m, inclusive

acct.name	name of account to log specified lines to
password	password for specified account
options	the following options are available

t	number of seconds to wait for the logon to complete; the default value is 20 seconds. A value of 0 causes the system to return to the user issuing the command immediately after posting a logon condition to the specified lines; the maximum value of t is 60.
S	suppresses all terminal output on the lines to be logged on, until the line is logged on
U	untraps any trapped process and then logs on the line; if not specified, a trapped line is not logged on and a message is displayed on the terminal of the user issuing the LOGON command

Description

A SYS2 privilege level is required to execute LOGON.

If no parameters are specified, the following prompt is displayed:

```
Enter processes, account, password and (options) :
```

Enter the line numbers, account name, and password, if any, to log on. This prompt is provided to maintain compatibility with the existing LOGON command.

When the logon is completed, a message similar to the following is displayed:

[533] Successful logon of process : *n*

A new message has been added to the LOGON process. If the number of seconds to wait for logon is exceeded, a message similar to the following is displayed:

[574] Logon posted for process : *n*

The line is logged on as soon as possible.

READ-STATUS - Enhanced

The READ-STATUS command now returns the status of DTR (Data Terminal Ready), as well as DSR, CTS, and CD.

Syntax

READ-STATUS {line.number}

Description

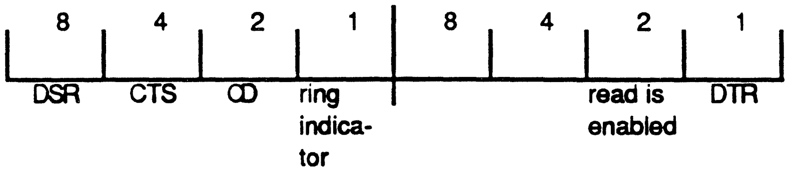
The status codes are determined by the settings of bits that indicate the current state of the RS232 control signals. The bits are interpreted as two hexadecimal values and displayed by READ-STATUS. Previously, the following status codes were returned:

E0	DSR, CTS, and CD are high
C0	DSR and CTS are high
80	DSR is high

where

DSR	Data Set Ready
CTS	Clear To Send
CD	Carrier Detect

Additional bits are now defined in the status byte. The following chart shows the meaning of each bit, and its hexadecimal value:



DTR indicates Data Terminal Ready.

This chart should be used to interpret the status codes. For example, if the status that is returned is 62, then CTS and CD (4 + 2) are high and read is enabled. All other bits are off, indicating the state of those control signals is low.

This command is available on Ultimate 6000/7000 series on the SYSPROG and SECURITY accounts.

Example	Description
READ-STATUS	reads status on current line
E1	status that is returned; indicates DSR,CTS, CD, and DTR are all high (8 + 4 + 2 = E).

SAVE - Enhanced

A new option, E, has been added to the SAVE verb. This option specifies that file indexes are not to be saved.

Syntax

SAVE (options

E new option; specifies that no indexes are to be saved to tape

Description

Indexes are saved and restored along with the rest of a file unless the E option is specified with the SAVE command.

Note: *The indexes are also not saved if the* B *option (save in pre-Rev 190 mode) is specified, since indexes are not compatible with revisions earlier than 200.*

SET-LANGUAGE - New

The SET-LANGUAGE command is used to specify the language setting for the current port, or to display the current setting for a specified port.

Syntax

```
SET-LANGUAGE
SET-LANGUAGE lang.code
SET-LANGUAGE port.no
SET-LANGUAGE ?
```

(null)	if no parameters are specified, displays setting for current port
lang.code	two-digit language code for desired language
port.no	displays current language setting for specified port
?	displays current language setting for all ports on the system

Description

The SET-LANGUAGE command stores the current language setting in the dictionary of the ACC file in the item for the current port.

The pointers to the ERRMSG file and PROCLIB file for the current user are changed to the data level corresponding to the current language code.

The default language code is US (United States, that is, English as used in the United States).

The following information is displayed by SET-LANGUAGE:

```
Line/Port #     Language Code     Language Name

  nnn                xx           description
```

The language codes are defined using the system application UltiKit; for more information, see Section 5, Multi-Lingual Capabilities.

Example	Description
SET-LANGUAGE FR	Sets language code for current port to French.

result:

```
Line/Port #     Language Code     Language Name

  031                FR           Français
```

T-ATT - Enhanced

Two new keywords have been added to T-ATT: UNBUFFERED and SPEED. In addition, the keyword DENSITY is now available for GCR and Pertec tape drives. These enhancements are available on Ultimate 6000 and 7000 series systems.

Syntax

T-ATT {UNBUFFERED} {SPEED = s} {DENSITY = d}

UNBUFFERED	sets the tape drive to unbuffered mode to allow greater tape error handling capability; affects only the GCR tape drive on Ultimate 6000 and 7000 series systems. The drive is reset to buffered mode when the tape is detached.
SPEED	sets the speed; affects only the Pertec FS1000 tape drive on Ultimate 6000 and 7000 series systems; valid SPEED settings are 25 and 100
DENSITY	sets the density; valid DENSITY settings for the GCR tape drive on Ultimate 6000 and 7000 series systems are 1600 and 6250; valid settings for the Pertec FS1000 are 1600 and 3200

Description

In order for these features to be set by T-ATT, the appropriate options need to be set on the front panel of the tape drive; for more information, see Section 8, Ultimate 6000/7000 Series Enhancements.

T-STATUS - Enhanced

The messages produced by the T-STATUS command have been enhanced to return more complete information based on the settings in T-ATT.

- If the tape is attached to the line issuing the command and the DENSITY keyword was used, the T-STATUS message is as follows:

  Tape n attached to your line, *status*, write density is *xxx*, *write*

  status is either 'off-line' or 'on-line'
  xxx is the density specified with the DENSITY keyword
  write is either 'write-protected' or 'write-permit'

- if tape is attached to another line, the message is as follows:

  Tape n attached to line m

- if tape is not attached, the message is as follows:

  Tape n not attached

TERM-VIEW - New

The TERM-VIEW command logically connects one terminal to a second terminal; information entered at either terminal is accepted as input to the process on the second terminal and is echoed on both terminals. Output from the process on the second terminal is displayed on both terminals.

Syntax

TERM-VIEW port.no {({exit.char}{,I})}

port.no	the port to which the current terminal is to be connected
exit.char	ASCII value in decimal of character to be used to exit the TERM-VIEW process; the default character is the upper case letter X. To exit the process, press <ESC> followed by the exit character. The process may be exited from either terminal. (For a list of ASCII codes, see Appendix A.)
I	inhibit synchronization between terminals of output display; the default of the TERM-VIEW process is to synchronize the rate of output between the two terminals. This option causes output to be displayed at the rate set on the second terminal; if this rate is higher than the rate on the terminal that initiated the TERM-VIEW verb, the I option could cause characters sent to the initiating terminal to be lost.

Description

TERM-VIEW can be used whenever a user needs to interact directly with another user at a second terminal. For example, in a training situation, the instructor can use TERM-VIEW to quickly and easily connect to a student's terminal to help with a lesson. In a support situation, a technician can connect to a user terminal (perhaps through a modem) and has access directly to the user's process, including register settings and other characteristics.

Terminals that are to be connected by the TERM-VIEW verb should be of the same term-type to ensure that the screen displays retain their integrity. TERM-VIEW uses the term settings of the second terminal to format the screens.

TERM-VIEW synchronizes the display of output between the two terminals. If the two terminals have the same baud rate, the display is virtually identical. However, if the terminals have different baud rates, there might be a slight discrepancy between the screens. For example, if the terminal that initiates the command is connected via a modem that is running at 1200 baud and the second terminal is running at 9600 baud, the display at the second terminal is slowed down so that the information is displayed at each terminal at approximately the same rate; however, because of the buffering capabilities of the terminals, the display on the two terminals is not identical. All characters are displayed on both terminals.

If the I option is specified, TERM-VIEW sends data at the speed determined by the second terminal. If the initiating terminal is operating at a speed slower than the second terminal, characters could be lost at the initiating terminal when the I option is specified. If the initiating terminal is operating at a speed equal to or faster than the second terminal, the I option has virtually no effect.

The two ports remain connected until you break the connection by pressing <ESC>, then the exit character. The default exit character is the upper case letter X.

Note: *On VT220 terminals, which do not have <ESC> keys, you can break the connection by pressing <CTRL-3> followed by the exit character.*

Any port can be the target of TERM-VIEW; to prevent a port from being set up as a TERM-VIEW port, use the TERM-VIEW-OFF verb. The default setting is TERM-VIEW-OFF.

TERM-VIEW 31 (65)	Connects the port executing the verb to port 31; sets the exit character to upper case A

Provided On

Any account with SYS2 privileges

See Also

TERM-VIEW-OFF
TERM-VIEW-ON

TERM-VIEW-OFF - New

TERM-VIEW-OFF locks the specified port against being the target of the TERM-VIEW verb. This is the default setting.

Syntax

TERM-VIEW-OFF {port.no}

port.no	port number to lock against TERM-VIEW; if not specified, the current port is locked
	port.no can only be specified from the SYSPROG or SECURITY accounts

Description

TERM-VIEW-OFF can be used whenever you want to protect a port from being connected to another port.

Provided On

Any user account with SYS2 privileges

See Also

TERM-VIEW
TERM-VIEW-ON

TERM-VIEW-ON - New

TERM-VIEW-ON enables the specified port to be the target of the TERM-VIEW verb.

Syntax

TERM-VIEW-ON {port.no}

port.no	port number to enable for TERM-VIEW; if not specified, the current port is enabled port.no can only be specified from the SYSPROG or SECURITY accounts

Description

TERM-VIEW-ON is used to enable the use of TERM-VIEW by a port that had previously been locked by TERM-VIEW-OFF.

Provided On

Any account with SYS2 privileges

See Also

TERM-VIEW
TERM-VIEW-OFF

TRANSLATE-INPUT - New

The TRANSLATE-INPUT command creates a translation table that may be used to translate characters input to or output by the system. Any character or sequence of characters can be translated to any other character or sequence of characters.

Syntax

TRANSLATE-INPUT {file.name {item.name}}

file.name	name of file in which to place translation table; if not specified, TRANSLATE-INPUT prompts for file name
item.name	name of item in which to save translation table; if not specified, TRANSLATE-INPUT prompts for item name

Description

The TRANSLATE-INPUT command is used whenever characters that are input or output need to be translated to other characters. For example, TRANSLATE-INPUT can be used to translate lower case letters to upper case or to specify characters for foreign keyboards. TRANSLATE-INPUT can also be used to translate keys for terminal emulations or for serial printers (must be enabled for output only).

TRANSLATE-INPUT defines a translation table. Another verb, TRANSLATE-LOAD, makes the table available to a port. A third verb, TRANSLATE-ON, enables the actual translation to start; a fourth verb, TRANSLATE-OFF, stops the translation.

Each translation sequence is made up of three and possibly four parts:

- character ID
- additional characters, if any, associated with the character ID
- translation characters
- flag settings, which are used to set display options

The decimal value of the character ID is used as an index that points to the attribute in which the translation for this character ID is stored.

There may be multiple additional characters for each character ID. Each additional characters entry has a corresponding translation characters entry and flag settings entry. All translation sequences for one character ID are stored as values in the attribute pointed to by the character ID.

Character ID

The first character in each translation sequence is called the character ID (char id). The character ID is entered in hexadecimal. You may define up to 256 unique character IDs.

When TRANSLATE-INPUT is invoked, a screen similar to the following is displayed:

```
File name: file name      Item name: item name      status.msg

  Char id :    ..




Enter char in hex (ex: 'A'='41',' '='20'), or <CR> to exit
```

If the item is new, the status.msg displays 'new item'; if the item currently exists, the status.msg displays 'item exists'.

The char id is the ASCII value in hexadecimal of the character to be translated; if a sequence of characters is to be translated, char id is the first character. For example, to translate the sequence <ESC>a0, enter a char id of 1B, which is the hexadecimal code for ESC. The decimal equivalent of the hexadecimal value is displayed; if the character itself is displayable, it is displayed. Any existing translation sequences for this char id are displayed.

Additional Characters

TRANSLATE-INPUT then prompts for additional characters:

```
File name: file name      Item name: item name      status.msg

  Char id : 1B      Decimal value = 27         ASCII char =

     ..........



Enter any additional characters in hex or CTL-X to exit
```

Each character ID may be associated with any number of additional character sequences with up to ten hexadecimal characters in each sequence. Each sequence must be unique and no complete sequence may be identical to the beginning characters of another sequence. For example, if you have a sequence defined for 123456, you cannot have a sequence 1234. However, you can have sequences 123456 and 1235 or sequences 123456 and 123457.

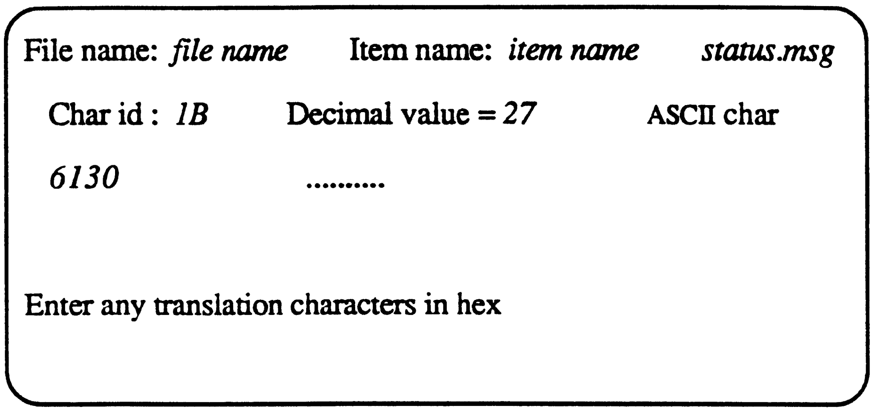
To translate a sequence of characters, enter the additional characters as a single string of hexadecimal characters. For example, to translate the additional characters in the sequence <ESC>a0, enter

6130, which is the hexadecimal representation of a0; press RETURN. If there is no sequence following the character ID, press just RETURN.

To return to the char id prompt, press <CTRL-X>.

Translation Characters

TRANSLATE-INPUT then prompts for the translation characters:

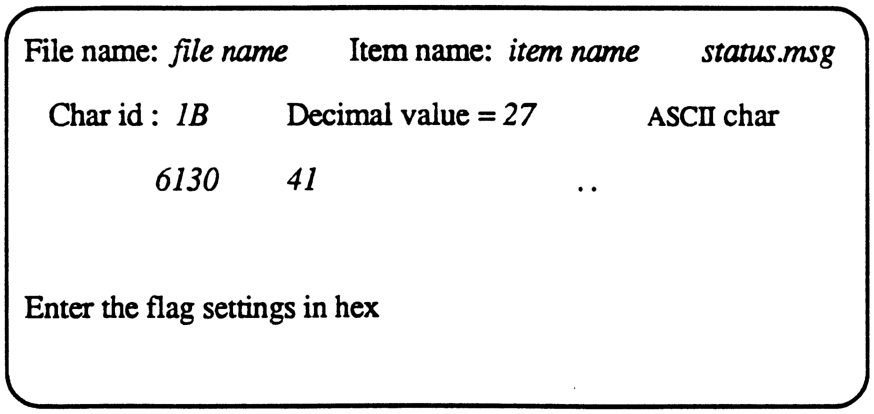


```
File name: file name     Item name: item name     status.msg

  Char id : 1B      Decimal value = 27         ASCII char

  6130              ..........


Enter any translation characters in hex
```

Enter the character or sequence of characters the specified characters are to be translated to. For example, if <ESC>a0 is to be translated to the character A, enter 41. Press RETURN.

Flag Settings

TRANSLATE-INPUT then prompts for flag settings, which determine the display options:



```
File name: file name     Item name: item name     status.msg

  Char id : 1B      Decimal value = 27         ASCII char

          6130      41                    ..


Enter the flag settings in hex
```

Currently, the only flag that is defined determines the characters to be echoed to the screen, as follows:

00	echo both source and translation characters
40	no echo on translation characters
80	no echo on source characters
C0	no echo on source or translation characters

An entry must be made for the flag setting. To list the currently available flag settings, press the question mark key (?). A message similar to the following is displayed:

```
The first digit is user selectable; the second is set by the program.
Valid user codes are :
  00 = echo both source and translation characters

Press <CR> for more
```

After the flag setting has been entered, the prompt to enter another sequence of additional characters is displayed. Enter as many sequences as desired.

Saving the Translation Table

After all sequences for the current char id have been entered, return to the char id prompt by pressing <CTRL-X> at the additional characters prompt. The system then checks the additional character sequences to make sure they are all valid. If any invalid entries are found, the first invalid one is displayed. For example, if a sequence is entered that is identical to the beginning of a second sequence, a message similar to the following is displayed:

```
File name: file name      Item name: item name        status.msg

  char id  1B        decimal value 27          ASCII char

     6130            41                        80
     613031          4131                      80


 6130       41
 Ambiguous translate string
```

Press RETURN. You can then change any of the information. Use the terminal arrow keys to move from field to field.

When the information is correct, press <CTRL-X> at the additional characters prompt.

TRANSLATE-INPUT returns to the char id prompt. Continue entering translation information. When all translation information has been entered, press just RETURN at the char id prompt. A prompt similar to the following is displayed:

```
FI to file, <F2> or X to void, <CR> to continue:
```

To file the table, enter the characters FI. To exit without changing the table, press function key <F2> or type X and press RETURN. To enter additional translation characters, press just RETURN.

Provided On Any account

See Also

TRANSLATE-LOAD
TRANSLATE -OFF
TRANSLATE-ON

TRANSLATE-LOAD - New

The TRANSLATE-LOAD command makes the specified translation table available to a port.

Syntax

TRANSLATE-LOAD {port.no} {file.name {item.name {INP|OUT}}}

port.no	port on which to make translation table available; if not specified, the port from which the command is issued is used
file.name	name of file in which to place translation table; if not specified, TRANSLATE-LOAD prompts for file name
item.name	name of item in which to place translation table; if not specified, TRANSLATE-LOAD prompts for item name
INP|OUT	specifies if table is to be used for input (INP) or output (OUT) translation; if not specified, TRANSLATE-LOAD prompts for direction. Either INP or OUT must be specified, but not both.

Description

Each port may have one input and one output table assigned at the same time.

Assigning a table does not enable it; for information on enabling the table, see the command TRANSLATE-ON.

Provided On

Any account

See Also

TRANSLATE-INPUT
TRANSLATE-OFF
TRANSLATE-ON

TRANSLATE-OFF - New

TRANSLATE-OFF is used to turn off character translation for a specified port.

Syntax

TRANSLATE-OFF {port.no} (INP|OUT

port.no	port number on which to turn off translation; if not specified, translation is turned off for the current port
INP|OUT	INP indicates to turn off table used for input translation OUT indicates to turn off table used for output translation Either INP or OUT must be entered, but not both.

Description

Translation must have previously been turned on and in the direction specified by the TRANSLATE-OFF command.

In order to turn off the translation table for a port other than your own, you must have SYS2 privilege level.

Provided On

Any account

See Also

TRANSLATE-INPUT
TRANSLATE-LOAD
TRANSLATE-ON

TRANSLATE-ON - New

TRANSLATE-ON is used to turn on character translation for a specified port.

Syntax

TRANSLATE-ON {port.no} (INP|OUT

port.no	port number on which to turn on translation; if not specified, translation is turned on for the current port
INP|OUT	INP indicates table is to be used for input translation; characters entered at the keyboard are translated before being seen by the system
	OUT indicates the table is to be used for output translation; characters generated by the system for output are translated before they are output
	Either INP or OUT must be entered, but not both.

Description

A translation table must have been loaded previously with translation in the direction specified by the option in TRANSLATE-ON.

In order to turn on the translation table for a port other than your own, you must have SYS2 privilege level.

Provided On

Any account

See Also

TRANSLATE-INPUT
TRANSLATE-LOAD
TRANSLATE -OFF

VERIFY-SAVE - New

The VERIFY-SAVE command is used to verify the integrity of a file-save or account-save tape.

Syntax

VERIFY-SAVE {(options}

options the following options are available:

A do not rewind tape; this option is intended to be used when verifying tapes that contain multiple account saves; if not used, VERIFY-SAVE always rewinds tape before beginning the verify

D debugging option; when specified, VERIFY-SAVE prompts the user with a continue or quit message whenever an item-size error, a tape format error, or an object item error occurs

N do not create VSAVE-STATS file; it is recommended that this option be used only when there is not enough disk space for the statistics

Description

The VERIFY-SAVE command checks the tape on the attached unit for the following types of errors:

- item size errors
- object item errors
- tape format errors

The statistics for the report are saved on a file called VSAVE-STATS; this file is provided on the SYSPROG account. For information on using the file, see the LIST-VSAVE-STATS command.

WHERE - Enhanced

A new column has been added to the information displayed by WHERE. The new column displays the character d if the information came from the Debugger Control Block (DCB); it contains the character m if the return address is delimited by a stack marker.

The number of entries in the return stack has been increased from 11 to 125. WHERE displays all the entries in the return stack; if there are more entries than fit on one line, a second line is used.

```
011 0210 000AC0 T0   1141.032 1142.05C  14.1EC  13.057
012 0210 000B00      1141.032  685.02F   5.0A0
023 0290 000DC2   d  1153.035  132.063
         000DC0         1.1A6 1141.032 685.02F   5.0A0
028 0210 000F00   m  1141.032 1142.05C 247.0D4
031 8010 07FE85      1125.091
         000FC0         1.155 1141.032 685.02F   5.0A0
```

4 Extended BASIC

Revision 200 of the Ultimate Operating System, Release 10, includes new and enhanced features for the Ultimate BASIC language. Because of these changes, **all BASIC programs from previous revisions of the operating system must be recompiled** after upgrading to Revision 200. For information on upgrading, see the *Ultimate Revision 200 Upgrade Document.*

The following changes have been made to BASIC:

- new reserved words
- addition of predefined variables and system variables
- change to file variables
- extension of arithmetic operators to dynamic arrays
- extension of comments to CALL, COMMON, DIM, and EQUATE statements
- new compiler directives:

  $COMPATIBILITY
  INCLUDE
  $INSERT
- extensions to assignment statements:

  new operators: += -= :=
  overlaying a substring with a specified expression
  modifying a specified number of delimited substrings
- new and enhanced BASIC statements and functions:

BREAK{KEY} statement	Enhanced
CLEARDATA statement	New
CLEARSELECT statement	New
COMMON statement	Enhanced
CONVERT statement	New
CRT statement	New
DIM statement	Enhanced
ECHO statement	Enhanced
EQU{ATE} statement	Enhanced
ERRTEXT function	New
EXECUTE statement	Enhanced
FIELD function	Enhanced
FMT function	New
FOR/NEXT statement	Enhanced
INMAT function	New
INPUT statement	Enhanced
MATCH{ES} operator	Enhanced
MATCHFIELD function	New
MATPARSE statement	New
MATREAD statement	Enhanced
OPEN statement	Enhanced
PAGING statement	New

READT statement	Enhanced
RELEASE statement	Enhanced
REMOVE statement	New
REUSE function	New
RQM statement	Enhanced
SLEEP statement	New
SORT function	New
SOUNDEX function	New
SUM function	New
SYSTEM function	Enhanced
TRAP ON THEN CALL statement	New
TRIM function	Enhanced
USERTEXT function	New
WRITE statement	Enhanced
WRITET statement	Enhanced

- BASIC compiler changes

These new and enhanced features are discussed in detail in the following pages.

Reserved Words

The following is a list of BASIC reserved words. These words cannot be used as simple variable names, array variable names, or labels. The new reserved words for Revision 200 are *italicized.*

AND	GOTO	_OUT._ *
_ARG._*	GO	_PASSLIST_
CAPTURING	GT	REPEAT
CASE	_IN._ *	_RETURNING_
CAT	LE	_RTNLIST_
DO	LOCKED	_SELECT._ *
ELSE	LT	_STACKING_
END	MATCH	STEP
EQ	MATCHES	THEN
_FROM_*	NE	_TO_ *
GE	NEXT	UNTIL
GLE	ON	WHILE
GOSUB	OR	

*not new to BASIC, only to the reserved word list

The following is a list of BASIC functions; the names of these functions cannot be used as array or matrix variable names. Ultimate strongly advises against using these names as simple variables and labels, although the compiler allows such use. The new reserved words for Revision 200 are *italicized.*

@	FIELD	RND
ABS	_FMT_	SADD
ALPHA	FMUL	SCMP
ASCII	FSUB	SDIV
CHAR	ICONV	SEQ
COL1	INDEX	SIN
COL2	_INDEXINFO_*	SMUL
COS	_INMAT_	_SORT_
COUNT	INSERT	_SOUNDEX_
DATE	INT	SPACE
DCOUNT	LEN	SQRT
DELETE	LN	SSUB
EBCDIC	_MATCHFIELD_	STR
EOF	_MAXIMUM_*	_SUM_
ERROR	_MINIMUM_*	SYSTEM
EXP	MOD	TAN
EXTRACT	NOT	TIME
ERRTEXT	NUM	TIMEDATE
FADD	OCONV	TRIM
FCMP	PWR	_TRIMB_
FDIV	REM	_TRIMF_
FFIX	REPLACE	_USERTEXT_
FFLT	_REUSE_	

*Reserved for future use

Predefined Symbols

The following predefined symbols and their values are now available for all BASIC programs:

@FM	field mark; this has the value CHAR(254)
@VM	value mark; this has the value CHAR(253)
@SM	sub-value mark; this has the value CHAR(252)

System Variables

The following symbols return information based on the current status of the system:

@EXECLEVEL	returns current EXECUTE level; equivalent to SYSTEM(21)
@HOLDFILE	number of last hold file created by PRINT statement in current BASIC program; if no hold file has been assigned, returns zero; equivalent to SYSTEM(22)
@LANGUAGE	returns two character language code of the language assigned to current port; equivalent to SYSTEM(27)
@PRIVILEGE	returns 0, 1, or 2 to indicate system privilege level of current user; equivalent to SYSTEM(23)
@SELECT	returns 1 if external select list is active; equivalent to SYSTEM(25)
@SENTENCE	returns TCL statement that invoked current program; statement is formatted as dynamic array; equivalent to SYSTEM(18). Elements in the statement are separated by attribute marks. If an element is enclosed in delimiters, the delimiters are removed. For example, if a program is invoked using the following comand:

```
RUN BP PGM1 A 'B,C' (D)
```

the following is returned in @SENTENCE within PGM1:

```
RUN~BP~PGM1~A~B,C~(D)
```

@SPOOLOPTS	returns current spooler assignment status; equivalent to SYSTEM(24)
@USERNO	returns current port number; equivalent to SYSTEM(19)
@WHO	name of current user; does not return name of any CHARGE-TO account; equivalent to SYSTEM(26)

File Variable Changes

A file variable, that is, the variable to which a file is opened, can now be used in a BASIC PRINT statement or BASIC debugger / (list) instruction to display the base frame NUMBER (FID) of the file.

In addition, the file variable can be tested to see if any file has been opened to it. A non-zero value indicates the file is opened.

If the file variable is changed in any way by the BASIC program, it is no longer considered a file variable.

```
   OPEN 'TEST.FILE' TO TF ELSE STOP
     .
     .
   PRINT 'Base frame ID: ':TF

result:
   Base frame ID:  120898
```

```
   OPEN 'TEST.FILE' TO TF ELSE TF = 0
     .
     .
   IF TF THEN
    READ ITEM FROM TF,'T1' ELSE ITEM = ""
   END ELSE
    ITEM = 'No test file available'
   END
   PRINT ITEM<1>
```

Arithmetic Operators and Dynamic Arrays

Arithmetic operators can now be used with dynamic arrays. The specified operation is automatically performed on corresponding array element. The operators are:

+	Add
-	Subtract
*	Multiply
/	Divide

If the arrays do not have the same number of elements, the system assumes a value of zero (0) for the missing elements for addition, subtraction, multiplication, and dividends in division. It assumes a value of one (1) for missing divisors in division.

Note: *A new function, REUSE, allows you to use the previous value instead of zero when the number of elements differ. For more information on REUSE, see the section, New and Enhanced BASIC Statements and Functions.*

```
   ARRAY1 =  1:AM: 2:VM:  2:VM:  2:AM: 3
   ARRAY2 = 10:AM:20:VM:20:VM:20:AM:30
   ARRAY3 = ARRAY1 + ARRAY2

result:
   ARRAY3 = 11:AM:22:VM:22:VM:22:AM:33
```

The elements of ARRAY3 are composed of the sums of the five elements in ARRAY1 and ARRAY2.

```
   ARRAY1 = 1:VM:1:AM:2:VM:2:AM:3
   ARRAY2 = 1:       AM:2:       AM:3
   ARRAY3 = ARRAY1 + ARRAY2
```

result:

```
   ARRAY3 = 2:VM:1:AM:4:VM:2:AM:6
```

ARRAY3 is built as follows:

1. The first two values in attribute 1 are added.
2. ARRAY2 does not have a second value in attribute 1, so 0 is added to the second value in ARRAY1.
3. The first value in attribute 1 of ARRAY1 is added to the first value in attribute 2 of ARRAY2.
4. ARRAY2 does not have a second value in attribute 2, so 0 is added to the second value in ARRAY1.
5. The values in attribute 3 are added.

Extension of Comments

Comments can now be included with the following statements on lines that end in a comma and continue on to the next line:

Syntax

```
CALL sub(a,b,..., ; *
        ...,x)

COM{MON} a,b,..., ; *
        ...,x

DIM{ENSION} a(n),b(m),..., ; *
        ...,x(z)

EQU{ATE} a TO b, c TO d,..., ; *
        ...,x TO y
```

Description

A semicolon must follow the comma and the comments must start with an asterisk (*).

```
010 COMMON FIRST,        ; *This is the first comment
011      CTR,            ; *This is another comment
012      LAST            ; *This is the last comment
```

result:
FIRST, CTR, and LAST are all compiled as COMMON variables.

Compiler Directives

A new compiler directive, $COMPATIBILITY, has been added. The compiler directives INCLUDE and $INSERT, which are alternative forms of $INCLUDE, have been added.

$COMPATIBILITY - New

A new directive to the compiler, $COMPATIBILITY, has been added. $COMPATIBILITY allows you to specify the compiler implementation to be used when compiling a BASIC program. This directive alters certain instructions to work according to the specified standard instead of the Ultimate standard if there is a conflict.

Syntax

$COMPATIBILITY compiler.imp

compiler.imp	implementation standards to use when compiling the program; currently, only R83PC, which generates instructions according to PICK R83 PC standards, can be specified

Description

The $COMPATIBILITY R83PC directive can be used in cases where an Ultimate release is not compatible with the SMA standard. Currently, $COMPATIBILITY R83PC affects the EXECUTE statement.

If the program has the $COMPATIBILITY R83PC directive set, then when the DATA stack is active, and there is no IN. or STACKING clause in the EXECUTE command, the DATA stack is transferred to the EXECUTE statement. The DATA stack is also cleared on return from the EXECUTE statement.

Note: *This use of and clearing of the DATA stack does not occur using current Ultimate standards.*

INCLUDE - New
$INSERT - New

The compiler directives INCLUDE and $INSERT can be used in place of $INCLUDE:

```
INCLUDE {file.name} program.name
$INSERT {file.name} program.name
```

Extension to Assignment Statements

The assignment statement has been enhanced with several new features.

New Operators

The assignment statement has three new assignment operators:

+= plus-equals; adds an expression to a variable and returns the results to the variable; this is equivalent to

```
var1 = var1 + expression
```

–= minus-equals; subtracts an expression from a variable and returns the results to the variable; this is equivalent to

```
var1 = var1 - expression
```

:= concatenate-equals; concatenates an expression with a variable and returns the results to the variable; this is equivalent to

```
var1 = var1 : expression
```

The value of the variable does not change until the entire right side of the statement has been evaluated.

Example	Description
I += 1	increments the value of variable I by 1; this is equivalent to I = I + 1
J –= 2	decrements the value of variable J by 2; this is equivalent to J = J - 2
A := B	concatenates the value of A with B and assigns the new value to A; this is equivalent to the statement A = A : B

Overlaying a Substring

A substring can now be overlaid by a string by using an assignment statement.

Syntax

variable[start.char,no.chars] = expression

variable	string to be overlaid
start.char	starting character position in the variable; if start.char evaluates to 0 or less, 1 is used as the value. If

	start.char evaluates to greater than the number of characters in the string, no characters are overlaid.
no.chars	number of characters to be overlaid; if no.chars evaluates to 0 or less, no characters are overlaid.
expression	replacement string; value used to replace the specified substring characters

All parameters can be literals or expressions.

Description

This form of the assignment statement does not change the length of the string variable.

If the number of characters in the replacement expression is less than the number of characters specified in no.char, blanks are added to the end of expression. If the number of characters in the replacement expression is greater than no.char, the excess characters in expression are truncated, not overlaid.

If no.chars is greater than the number of characters remaining in the string, only the number of characters remaining are overlaid. Any extra characters are ignored.

```
   A = 'ABCDEFGHI'
   A[4,3] = '***'

result:
   A = 'ABC***GHI'
```

The substring starting at the fourth character position and containing three characters (DEF) is replaced by the specified three characters (***).

```
   M = 'ABCDEFGHI'
   M[2,10] = 'XXXXX'

result:
   M = 'AXXXXX   '
```

The substring 'BCDEFGHI', which starts at the second character and extends to the end of the string (since there are fewer than ten characters left in the string), is replaced by the specified 5-character substring plus 3 spaces.

Assigning Values to Delimited Substrings

One or more delimited substrings can now be added, deleted, or replaced in a string. Unlike the form just described, this form of the assignment statement can change the length of the string.

Syntax

variable[delimiter,start.sub,no.subs] = expression

variable	contains original string
delimiter	substring delimiting character; if the delimiter evaluates to more than one character, only the first character is used as the delimiter
start.sub	first substring to be changed; if start.sub is 0 or less, 1 is used as the value. If start.sub is greater than the number of delimited substrings in the original string, the required number of null delimited substrings are appended to the string.
no.subs	number of substrings to be changed; the actual change is determined as follows: • if no.subs is greater than 0, this number of delimited substrings are replaced • if no.subs is 0, expression is inserted at the location specified by start.sub • if no.subs is less than 0, then starting at the substring specified by start.sub, no.subs substrings are deleted from the existing string, then expression is inserted
expression	string to be inserted in place of the delimited substrings; should contain substrings delimited by the same value as the original string

All parameters can be literals or expressions.

Description

The first substring has no initial delimiter; the last substring has no final delimiter.

If the delimiter is a system delimiter (attribute mark, value mark, or sub-value mark), the substring is terminated by a second delimiter of the same level and ignores any higher level delimiter. For example, if sub-value mark is the delimiter, the substring does not stop at a value mark or attribute mark, only at the next sub-value mark.

If no.subs is greater than the number of substrings in expression, the required number of null delimited substrings are added to expression.

If no.subs is less than the number of substrings in expression, the extra characters in expression are ignored.

If no.subs is greater than the number of substrings remaining in the original string, the required number of null delimited substrings are appended to the original string.

```
  A = '1*2*3*4'
  A['*',2,2] = 'A*B'

result:
  A = '1*A*B*4'
```

The substring delimiter is an asterisk (*); the substring to replace starts at the second delimited substring and contains two substrings (2*3); it is replaced by the specified two substrings and their delimiter (A*B).

```
  A = '1':VM:'2':AM:'3':VM:'4':VM:'5'
  A[VM,2,2] = 'A':VM:'B'

result:
  A = '1':VM:'A':VM:'B':VM:'5'
```

The substring delimiter is a value mark; the substring to replace starts at the second delimited substring and contains two substrings ('2':AM:'3':VM:'4':); it is replaced by the specified two substrings and their delimiter ('A':VM:'B'). The attribute mark that separates the value marks is ignored by this form of the assignment statement.

```
  A = '1*2*3*4'
  A['*',-3,5] = 'A*B'

result:
  A = 'A*B***'
```

The assignment starts at the first substring (-3 defaults to 1). String A contains fewer than five substrings, the number of substrings specified, so one null substring is appended to A. The replacement expression also contains fewer than five substrings, so three substrings are appended to it. Finally, the expression overlays the specified substrings in A.

```
A = '1*2*3*4'
A['*',6,2] = 'A*B'
```

result:

```
A = '1*2*3*4**A*B'
```

The assignment starts at the sixth substring. However, string A contains only four substrings, so two null substrings are appended to it. The expression is then appended to A, starting at the sixth substring.

```
A = '1*2*3*4'
A['*',3,0] = 'A*B'
```

result:

```
A = '1*2*A*B*3*4'
```

Since the number of substrings is zero, the two substrings in expression are inserted at the third substring position and no substrings are deleted.

```
A = '1*2*3*4'
A['*',3,-2] = 'Z'
```

result:

```
A = '1*2*Z'
```

Two delimited substrings are deleted starting at the third delimited substring and the new substring is inserted.

New and Enhanced BASIC Statements and Functions

The following pages describe each of the new and enhanced BASIC statements and functions.

The new statements are completely documented here; the enhanced statements have only the enhancements documented here. For more information on BASIC, see the *Ultimate BASIC Language Reference Guide*.

BREAK KEY Statement - Enhanced

The BREAK statement, which enables or disables the BREAK key at the user's terminal, can now be set with an expression. The word KEY can now be used as part of the statement.

Syntax

BREAK {KEY} expression

expression	determines setting; must evaluate to a numeric value; a value of zero (0) is equivalent to OFF, and all other values are equivalent to ON.

Description

The expression form of BREAK increments or decrements the BREAK inhibit counter by one, as appropriate. (Setting the BREAK key is cumulative. That is, each time a BREAK statement is encountered, the system increments or decrements by one, as appropriate, a counter called the BREAK inhibit counter. For example, if three BREAK OFF statements are encountered, three BREAK ON statements must be encountered before the BREAK key is enabled.)

Example	Description
F = 0 BREAK F	Disables BREAK key
F = 1 BREAK KEY F	Enables BREAK key

CLEARDATA Statement - New

The CLEARDATA statement removes all data that has been pushed onto the stack with the DATA statement.

Syntax

CLEARDATA

Description

The CLEARDATA statement is useful to ensure that the stack is empty so that terminal input can be requested.

All data previously placed on the stack and not yet used is removed, even when that data has been established by a PROC.

After CLEARDATA has been executed, subsequent INPUT statements will request terminal input unless another DATA statement is executed between the CLEARDATA and the INPUT statements.

```
   DATA '123'
   DATA '456'
   DATA '789'
   INPUT A
   INPUT B
   CLEARDATA
   INPUT C
```

result:

```
   A = '123'
   B = '456'
   C = (waiting for input from the terminal)
```

The first two stacked DATA elements are used to satisfy the first two INPUT statements. The third stacked DATA element ('789') is removed from the stack, allowing input to be requested from the terminal to use as the value of variable C.

CLEARSELECT Statement - New

The CLEARSELECT statement cancels any outstanding select list.

Syntax

CLEARSELECT {select.variable}

select.variable	select list to clear; if omitted, the program's internal default select variable is used

Description

Once the list is cleared, any subsequent READNEXT executes its ELSE statements.

```
    SELECT MASTER.FILE TO MASTER.LIST
      .
      .
100 READNEXT CUST FROM MASTER.LIST ELSE GOTO 999
      .
      .
    CLEARSELECT MASTER.LIST
```

A select list is created in the variable MASTER.LIST. The list is later cleared. If the READNEXT statement is executed after the CLEARSELECT statement, the program transfers to the routine at 999.

COMMON Statement - Enhanced

A COMMON area can now be named; the variables within named COMMON areas are accessible to all subsequent programs, including EXECUTEd programs.

Syntax

COM{MON} /name/ var1,var2,...

name	name to be given to common area; must be enclosed within slashes (/)
var1,var2,...	name of variables to be included in named COMMON area

Description

Two different programs can be executed with the same named COMMON area; the value of the variables are passed from one program to the other. You can nest EXECUTE statements within programs and a named COMMON established at one level is available to all other levels.

The areas for named COMMON statements are assigned at runtime. If the named COMMON statements are not executed, the named COMMON areas are not assigned.

The values in the named COMMON area can be examined and modified within a program as necessary.

More than one named COMMON area can be used in a program. Each named COMMON area counts as one variable. Each named COMMON area can have up to 3223 variables in it.

Named COMMON areas are associated with a port; each port on the system has its own set of named COMMON areas. The TCL statement LIST-NAMED-COMMON can be used to list the set of named COMMON areas for the current port. (LIST-NAMED-COMMON is described in Section 3, System Commands.)

The variables in the named COMMON area retain their values until the port is logged off or until the areas are canceled by a RELEASE /name/ statement.

```
In program 1:
   COMMON /PEOPLE/ NAME,ADDRESS,STATE,TOTAL
     .
     .
   IF STATE = 'CA' THEN EXECUTE CA.CALC CAPTURING TAX

In program CA.CALC
   COMMON /PEOPLE/ NAME,ADDRESS,STATE,TOTAL
     .
     .
   END
```

```
   COMMON /ITERATIONS/ CTR
   CTR += 1
   PRINT 'This program has been executed ':CTR:' times.'
```

result:

The first time the program is executed, the message says

```
   This program has been executed 1 times.
```

Each time this program is executed, until the user logs off, one is added to a counter that is defined in a named COMMON area. Each time the program is entered, CTR contains the value set the last time the program was executed. The third time it is executed during the current log on session, the message says

```
   This program has been executed 3 times.
```

CONVERT Statement - New

The CONVERT statement converts all occurrences of a character in a string to another character.

Syntax

CONVERT search.expr TO replace.expr IN var

search.expr	expression evaluating to a list of one or more characters to be converted
replace.expr	expression evaluating to a corresponding list of replacement characters
var	specifies the string variable in which characters are to be converted

Description

The search expression and replace expression correspond on a 1-for-1 basis. If the search expression contains more characters than the replace expression, any of the excess search characters found in var are deleted.

If the replace expression contains more characters than the search expression, the excess replacement characters are ignored.

```
   UPPER = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
   LOWER = 'abcdefghijklmnopqrstuvwxyz'
   STRING = 'the quick brown fox'

   CONVERT LOWER TO UPPER IN STRING

result:
   STRING = 'THE QUICK BROWN FOX'
```

All lower case letters were converted to their upper case form. This example is equivalent to using the output conversion function OCONV(STRING,'MCU') (mask characters upper case).

```
   CONVERT "O" TO "*" IN STRING

result:
   STRING = 'THE QUICK BR*WN F*X'
```

The letter O was converted to an asterisk in STRING.

CRT Statement - New

The CRT statement displays specified output at the terminal and is a synonym for the DISPLAY statement.

Syntax

CRT expression

expression	may contain any valid BASIC expression, formatting operators, and @ functions.

Description

The CRT statement can be used interchangeably with DISPLAY to print data at the terminal.

Output from the CRT statement is always directed to the terminal, regardless of PRINTER ON statements or the P option on the RUN verb.

Output of the CRT statement cannot be captured using the OUT. or CAPTURING clause in an EXECUTE statement. Any output specified by CRT continues to be directed to the terminal.

Output is not counted for purposes of determining line spacing for HEADING, FOOTING, or PAGE statements.

DIM Statement - Enhanced

Variables can now be used to define the size of an array at runtime. The array can also be dimensioned to zero, which in effect gives it a variable dimension.

Syntax

DIM array.name(var)

Description

If the value of the variable is not determined before the DIM statement is executed, the array is dimensioned to zero.

If an array is dimensioned with a variable or to the value zero, then is subsequently used as the destination for MATPARSE or MATREAD, the array is automatically redimensioned using the number of attributes in the item as the new dimension for the array. This automatic redimensioning occurs each time a new item is read into the array.

The array can also be redimensioned using another DIM statement.

If the array is redimensioned to a size with fewer elements than previously, the elements beyond the new size are cleared.

The INMAT function can be used to determine the current size of the array.

The maximum size to which an array can be dimensioned is 3223.

```
   SZ = 0
   DIM A(SZ)
   OPEN 'TEST' TO TEST ELSE ABORT '201', 'TEST'
   MATREAD A FROM TEST,'ITEM.1' ELSE STOP 202,'ITEM.1'
   SZ = INMAT()
```

result:
Array is initially dimensioned to 0; MATREAD redimensions it to size of ITEM.1.

ECHO Statement - Enhanced

The ECHO statement, which enables or disables the echoing of characters at the user's terminal, can now be set with an expression.

Syntax

ECHO expression

expression	determines setting; must evaluate to a numeric value; a value of zero (0) is equivalent to OFF, and all other values are equivalent to ON.

Example	Description
DISPL = 0 ECHO DISP	Input will not be displayed.
DISP = 1 ECHO DISP	Input will be displayed.

EQUATE Statement - Enhanced

The EQUATE statement now allows negative numbers to be equated.

```
   EQU minus1 TO -1

result:
   This equates the variable "minus1" to the value "-1".
```

ERRTEXT Function - New

The ERRTEXT function returns the text associated with a specified ERRMSG file item.

Syntax

ERRTEXT(errmsg.id{,errmsg.param{,...}})

errmsg.id	item.id of item in ERRMSG file item to be displayed
errmsg.param	parameters used by ERRMSG item

Description

The ERRMSG file can contain multi-level data files, where each data file is in a different language. The system command SET-LANGUAGE is used to specify the particular data level that is used. The ERRTEXT function formats the message appropriately for each language.

```
   PRINTERR ERRTEXT('202',CUSTID)
```

result:
   'custid' not on file.

This inserts the value of the variable CUSTID into error message 202, then prints the message on the last line of the terminal.

```
   IF X = "" THEN
       Y = ERRTEXT(204,FILE.NAME)
       GOSUB PRINT.SUB
   END
```

result:
   If X is null, the text of ERRMSG 204, with file.name inserted, is assigned to the variable Y, and the program branches to subroutine; Y = [204] File definition *'file.name'* is missing.

EXECUTE Statement - Enhanced

The EXECUTE statement now supports five new parameters in redirection clauses. In addition, the comma and double slashes preceding a redirection clause are now optional.

Syntax

EXECUTE expression {{,{//}} redirection.clause {{,{//}} redirection.clause}...}

redirection.clause	specifies source or destination for data in executed statement; in addition to previous parameters, may be any of the following:

CAPTURING variable	output of executed statement is redirected to variable; equivalent to OUT. > or OUT. = variable
PASSLIST expression	expression contains select list to be redirected to statement being executed; equivalent to SELECT. < expression
RETURNING variable	returns all ERRMSG message numbers and parameters generated by executed statement; ERRMSG numbers are separated by attribute marks; parameters within a message are separated by value marks; equivalent to values returned by GET(MSG.)
RTNLIST variable	select list generated by executed command is redirected to variable; equivalent to SELECT. > variable
STACKING expression	data in expression is to be used as input for the executed statement; equivalent to IN. < or IN. = expression

Description

The equal sign (=) is a new symbol in EXECUTE; it is synonymous with < in an IN. clause, and > in an OUT. clause.

When formatting the EXECUTE statement in your BASIC program, you may begin a new line after any comma.

The double slashes (//) do not compile if they precede the new parameters. In general, Ultimate advises using a comma (,) as a separator between clauses if using the old forms.

The $COMPATIBILITY directive may have some effect on EXECUTE; for information, see the $COMPATIBILITY directive for details.

```
EXECUTE 'SORT BP = "AR]"' CAPTURING LST RETURNING MSGS
```

result:
The list of sorted items is returned in LST; the message number and number of items is returned in MSGS.

FIELD Function - Enhanced

The FIELD function can now return multiple delimited fields.

Syntax

FIELD(string.expr,delimiter,start.field{,no.fields})

string.expr	string to be used
delimiter	delimiting character of a field; if delimiter evaluates to more than one character, only the first character is used
start.field	starting occurrence of field to return; if start.field is 1 and the delimiter is null ("") or is not in the string, the entire string is returned. If start.field is greater than 1 and the delimiter is null ("") or is not in the string after the specified starting field, a null value is returned. If start.field is less than zero, 1 is used as the value.
no.fields	number of fields to return; if omitted or less than 1, 1 is assumed. If no.fields is greater than the number of fields remaining in the string, the remainder of the string is returned.

All parameters can be literals or expressions.

Description

The boundary delimiter characters of the substring are not returned.

The first delimited substring has only an end delimiter character. For example, if the string is A*B*C and the delimiter is *, the first field is A; if the string is *A*B*C and the delimiter is *, the first field is null ("").

This form of the FIELD function is equivalent to a G processing code in an ICONV or OCONV function, where start.field is one greater than the field to skip used in the G processing code.

```
   X = 'A*B*C*D*E*F'
   Y = FIELD(X,'*',2,3)

result:
   Y = 'B*C*D'
```

The multiple FIELD function extracts three fields from X starting with the second field and returns the result to Y; the field delimiter is an asterisk (*). This is equivalent to the following OCONV function:

```
   Z = OCONV(X,'G1*3')

result:
   Z = 'B*C*D'
```

FMT Function - New

The FMT function formats a string to a pattern; it can be used as an alternative to the format operator in an assignment statement.

Syntax

FMT(expression,format.mask)

expression	variable to be formatted
format.mask	format mask; may contain any valid set for mask codes

Description

Format masks are described in the *Ultimate BASIC Language Reference Guide.*

```
   Y = 12000
   X = FMT(Y,'R2#10')
```

result:

```
   X = 12000.00
```

This function yields the same results as the following assignment statement using a format operator:

```
   X = Y 'R2#10'
```

FOR/NEXT Statement - Enhanced

The variable specified with the FOR statement no longer need be specified with the NEXT statement.

Syntax

```
FOR var = expr1 TO expr2 {STEP expr3} {WHILE/UNTIL expr4}
     .
     .
NEXT {var}
```

```
FOR I = 1 TO 10
  PRINT I
NEXT
```

INMAT Function - New

The INMAT function is used to return information about elements in dimensioned arrays and modulos of files that were opened.

Syntax

INMAT()

Description

INMAT() can be used to return values generated by the following statements:

DIM
MATPARSE
MATREAD
OPEN

INMAT() returns the number of elements in a dimensioned array if the array is redimensioned by another DIM statement

INMAT() returns the actual number of elements in a dimensioned array after a MATREAD or MATPARSE in the following cases:

- when the array is initially dimensioned as zero
- when the array is dimensioned with a variable
- when the array is dimensioned with a literal that is greater than the actual number of attributes read or parsed into it

INMAT() returns zero as the number of elements after the following:

- a DIM statement
- a MATREAD or MATPARSE statement is used to assign data to an array that is dimensioned with a non-zero literal, and the number of elements being read or parsed is greater than the dimensioned size of the array

INMAT() returns the modulo of the file after an OPEN statement.

Caution! The value of INMAT() is volatile; if the value is needed, it should be retrieved immediately after the statement that generates it has completed.

```
DIM A(0)
MATREAD A FROM TESTFILE,'ITEM.1' ELSE STOP
SZ = INMAT()
```

result:
The number of elements in the array A are returned in SZ.

INPUT Statement - Enhanced

A new form of INPUT is available that accepts a single character.

Syntax

INPUT var,0

Description

The character that is input is returned in var with no editing. This allows non-printable characters such as <ESC> to be input.

A literal 0 (zero) must be specified.

Note: The X-ON/X-OFF characters, represented by <CTRL-Q> and <CTRL-S>, are not returned by INPUT var,0 if X-ON/X-OFF protocol is enabled.

No prompt character is displayed.

```
   PRINT 'Press <ESC> to return to main menu, any other key to
      continue: ':
   INPUT A,0
   IF SEQ(A) = 27 THEN GOTO BEGIN
      .
      .
result:
   If <ESC> is pressed, program goes to routine at label BEGIN.
```

MATCHES Operator - Enhanced

The MATCHES (Pattern Matching) operator can now be used to find the negation of an nA (n alphabetic characters) or nN (n numeric characters) pattern; the pattern nX (n characters of any type) cannot be negated, nor can a literal text pattern be negated.

Syntax

```
MATCH{ES} {~} nN
MATCH{ES} {~} nA
```

~ indicates negation of pattern that follows
n number of characters in pattern

Description

The number of characters specified by n must match the number of characters in the string to be compared.

The tilde (~) negates the pattern match. The negation is true only if no characters in the expression match the type (N or A). For example, ~3N matches a string of exactly three non-numeric characters. It does not match a string of three characters containing a number.

```
   A = 'ABC123'
   IF A MATCHES '3A3N' THEN .....

result:
   The test is true.
______
   IF A MATCHES '~3N~3A' THEN .....

result:
   The test is true.
______
IF A MATCHES '~6N' THEN .....

result:
   The test is false because there are not six non-numeric characters
   in the string.
```

MATCHFIELD Function - New

The MATCHFIELD function performs a MATCH operation, and if the string matches the specified pattern, MATCHFIELD returns a portion of the string.

Syntax

MATCHFIELD(string.expr,pattern,return.field)

string.expr	string to be used
pattern	valid pattern match; the valid match patterns (where 'n' is an integer) are:

0X	zero or more characters of any type
nX	exactly n characters of any type
{~}0A	zero or more alphabetic characters
{~}nA	exactly n alphabetic characters
{~}0N	zero or more numbers
{~}nN	exactly n numbers
"text"	literal string enclosed in single or double quotes

These patterns may be combined, as desired.

return.field	number of the field in the string to return

All parameters can be literals or expressions.

Description

The MATCHFIELD function returns the portion (field) in the string that matches the pattern element number specified in the return.field parameter. Each element in the pattern defines a field in the string. For example, the pattern "3N-2N-4N" contains five fields as follows:

1	3N
2	-
3	2N
4	-
5	4N

The tilde (~) negates the pattern match. The negation is true only if no characters in the expression match the type (N or A). For example, ~3N matches a string of exactly three non-numeric characters. It does not match a string of three characters containing a number.

A null string matches the following patterns:

0A, 0N, 0X, '', and ""

The 0X pattern matches the entire string; if any pattern match is appended to 0X, the match is false and MATCHFIELD returns a null value.

If either the 0A or 0N pattern is specified, matching continues until a character is encountered that does not match the pattern (either alphabetic or numeric). If another pattern follows a 0A or 0N pattern, that pattern is used in the match as soon as a character is encountered that does not match 0A or 0N pattern.

Alternative patterns may be specified within a single match expression by concatenating the alternatives, separated by a value mark. For example, 3N:VM:"" can be used to match a field that contains either three numbers or a null string.

To result in a match, the value of the string expression must match the entire value of the pattern expression. If the string does not match the pattern or pattern alternatives, MATCHFIELD returns a null value.

The return.field expression specifies an element in the match pattern and the string.expr. Because the match pattern may be made up of several distinct pattern elements, the return.field is used to return only the part of the string that matches the return.field pattern element.

```
   S = 'ABC-123-XYZ'
   A = MATCHFIELD(S,'3A-3N-3A',1)

result:
   A = 'ABC'
```

The specified five-field pattern is matched and the first field of the matched string is returned in A.

```
   S = 'ABC-123-XYZ'
   A = MATCHFIELD(S,'3A-3N-3A',2)

result:
   A = '-'
```

The second field of the matched string is returned.

```
   A = 'ABCDEFGHIJKLMNOP123'
   M = MATCHFIELD(A,'0A3N',2)

result:
   M = '123'
```

The string is compared to the pattern 0A until a nonalphabetic character is encountered; the 3N pattern is then used.

MATPARSE Statement - New

The MATPARSE statement is used to copy a dynamic array or other delimited string into a dimensioned array.

Syntax

MATPARSE dim.array FROM string,delimiter

dim.array	previously defined dimensioned array
string	string whose elements are to be copied
delimiter	character that delimits elements in string

Description

If the dimensioned array is defined with a fixed size, MATPARSE places each delimited value in the string into successive elements of the dimensioned array, up to the size of the array. If there are more delimited values in the string than there are elements in the dimensioned array, the extra delimited values are all placed in the last element of the dimensioned array. If there are fewer delimited values in the string than there are elements in the array, the remaining elements are set to null.

If the dimensioned array is defined with a size of zero or with a variable size, MATPARSE places each delimited value in the string into successive elements of the array and redimensions the array to the number of elements in the string. The INMAT function can be used to determine the size of the dimensioned array. If the number of delimited values in the string is greater than the maximum size of a dimensioned array (currently 3223), the remaining values are placed in the last element in the array. In this case, INMAT returns a value of zero.

```
   SZ = 0
   DIM A(SZ)
   MATPARSE A FROM DY.ARRAY,@FM
   SZ = INMAT()
```

result:
   Array is initially dimensioned to 0; MATPARSE redimensions it to size of DY.ARRAY.

MATREAD Statement - Enhanced

MATREAD can now be used to read into variably dimensioned arrays. The syntax of the MATREAD statement has not changed.

Description

If the dimensioned array is defined with a size of zero or with a variable size, MATREAD reads each attribute from the item into one element of the array. The INMAT function can be used to determine the size of the array. If the number of attributes in the dynamic array is greater than the maximum size of a dimensioned array (currently 3223), the remaining attributes are placed in the last element in the array. In this case, INMAT returns a value of zero.

If the dimensioned array is defined with a fixed size, MATREAD functions as previously.

OPEN Statement - Enhanced

The OPEN statement can now be used to open subroutines that are to be indirectly called using the CALL @ statement.

Syntax

OPEN 'SUB{ROUTINE}', '{file.name }sub.name' TO var THEN/ELSE statements

file.name	file that contains the subroutine; if file.name is not specified, the system looks for a cataloged subroutine in the current user's master dictionary
sub.name	subroutine name
var	variable to be used in CALL @ statement

Description

This statement increases the speed of using indirect calls by approximately six times because the system saves the location of the subroutine and does not have to recalculate it each time.

The variable can be declared in a COMMON statement and, if a subroutine is opened to it in one program, the information can be passed to subsequent programs.

```
OPEN 'SUB', 'SUBR.ADD' TO S ELSE STOP 'B25', "",'SUBR.ADD'
 .
 .
CALL @S(X,Y,Z)
```

PAGING Statement - New

The PAGING statement turns off heading and footing statements. The output is then produced as one continuous page.

Syntax

PAGING OFF

Description

The PAGING OFF statement affects both HEADING and FOOTING statements. A new HEADING or FOOTING statement must be executed in order for the program to resume automatic paging.

The PAGING OFF statement resets the page counter to zero.

```
HEADING "Delivery Summary  'CDLPL'"
  .
  .
PAGING OFF
  .
  .
HEADING "Scheduling Summary  'CDLPL'"
```

READT Statement - Enhanced

An option has been added that allows data read from tape to be converted to ASCII hexadecimal.

Syntax

READT{X} var THEN/ELSE statements

X	converts data to hexadecimal format

Description

This feature is intended to allow binary data to be read by BASIC programs. For example, if the data contains segments marks (x'FF'), this option can be used to read the segment mark and any data that follows.

For information on writing data as hexadecimal characters, see the description of WRITET.

READTX A
B = FIELD(A,'F',3)
B = ICONV(B,'MX')
	Reads data as hexadecimal characters, extracts any data past a hexadecimal FF (segment mark); converts the string to ASCII

RELEASE Statement - Enhanced

The RELEASE statement can be used to release named COMMON areas.

Syntax

RELEASE /name/

/name/ name of COMMON area to be released; the slashes (/) are required

Description

The COMMON block can be released by any program that uses it.

When a named COMMON block is released, its name is removed from the list of named COMMON blocks and all space used by the variables is released to free (overflow) space.

```
    COMMON /PEOPLE/ NAME,ADDRESS,STATE,TOTAL
      .
      .
    RELEASE /PEOPLE/
```

result:
The named common area is released and the information in the variables is no longer accessible.

REMOVE Statement - New

The REMOVE statement places successive elements of a dynamic array into a specified variable and returns a value that corresponds to the delimiter encountered. The REMOVE statement does not change the dynamic array.

Syntax

REMOVE var FROM array.var SETTING delimiter.var

var	variable to which the substring is assigned
array.var	dynamic array being used
delimiter.var	variable whose value indicates the delimiter character for the substring last assigned

Description

The REMOVE statement is useful for sequential processing of dynamic arrays because it avoids repeated scanning.

As successive substrings are assigned from the dynamic array to the variable, an internal pointer is left pointing to the end of the substring last assigned. In addition, a value is assigned to the delimiter.var that indicates the type of delimiter located at the end of that substring.

Each subsequent execution of REMOVE moves the pointer to the next delimiter and assigns this substring to the variable. The statement can be executed repeatedly until all substrings have been processed.

You may reset the pointer to the beginning of the array at any time by assigning the array.var to itself. The pointer is also reset if you attempt to REMOVE past the end of the array.

The possible values returned in the delimiter.var are:

Value	Delimiter Encountered
0	segment mark (SM) for end-of-string
2	attribute mark (AM)
3	value mark (VM)
4	subvalue mark (SVM)

```
   A = 'ABC':AM:'123':VM:'456':AM:'XYZ'
   FOR I =1 TO 4
    REMOVE STRING FROM A SETTING DELIMITER
   NEXT

result:
   STRING = 'ABC' ;      DELIMITER = 2
   STRING = '123' ;      DELIMITER = 3
   STRING = '456' ;      DELIMITER = 2
   STRING = 'XYZ' ;      DELIMITER = 0
```

The first pass extracts an attribute ('ABC'). The second pass extracts a value ('123'). The third pass extracts an attribute ('456'). The fourth pass extracts the last attribute ('XYZ'), which is delimited by a segment mark.

REUSE Function - New

The REUSE function can be used when performing arithmetic operations on two dynamic arrays that may have unequal numbers of attributes, values, or subvalues. In these cases, REUSE reuses the value of the last attribute, value, or subvalue within the same level until the next higher delimiter is encountered.

Syntax

REUSE(array.expr)

array.expr	expression that evaluates to the dynamic array to be used; it may also be a single value

Description

The REUSE function can be used to repetitively process a single value against a dynamic array.

With REUSE, the value of the last attribute, value, or subvalue in the same level is reused until the next higher delimiter is encountered.

Without REUSE, a zero is used as the value when a corresponding attribute, value, or subvalue is not available. The substitution stops when the next higher delimiter is encountered.

REUSE can be used only within an expression; it cannot be assigned to a variable.

```
   ARRAY1 =  1:AM: 2:VM: 2:VM: 2:AM: 3:VM:10
   ARRAY2 = 10:AM:20:VM:20:VM:20:AM:30
   ARRAY3 = ARRAY1 + REUSE(ARRAY2)

result:
   ARRAY3 = 11:AM:22:VM:22:VM:22:AM:33:VM:40
```

The elements of ARRAY3 are composed of the sums of 6 elements in ARRAY1 and 5 elements in ARRAY2, with the last ARRAY2 element (30) being used twice.

```
   ARRAY1 = 110:VM:100:AM:120:AM:140
   ARRAY2 = ARRAY1 * REUSE(20)

result:
   ARRAY2 = 2200:VM:2000:AM:2400:AM:2800
```

20 is multiplied with each element of ARRAY1, assigning the resulting value to ARRAY2.

```
   ARRAY1 = 1:AM:  2:VM:4:AM:  6:VM: 7:AM: 8
   ARRAY2 = 5:AM:10        :AM:20:VM:30
   ARRAY3 = ARRAY1 + ARRAY2
   ARRAY4 = ARRAY1 + REUSE(ARRAY2)

result:
   ARRAY3 = 6:AM:12:VM:  4:AM:26:VM:37:AM: 8
   ARRAY4 = 6:AM:12:VM:14:AM:26:VM:37:AM:38
```

ARRAY3 has four attributes; the second and third attributes are multi-valued, and 0 was used twice because of missing elements in ARRAY2.

For ARRAY4, elements in the second and fourth attributes in ARRAY2 were each REUSEd to account for the missing elements.

RQM Statement - Enhanced

The amount of time RQM is to pause a program can now be specified.

Syntax

RQM {expression}

expression	if expression is a numeric value, defines number of seconds RQM is to pause program; if expression is a string value in the form hh:mm, defines time of day program is to end pause

Description

If no expression is used, RQM pauses for one second.

If the value of expression is less than or equal to zero, a command is sent to the kernel to perform a kernel RQM, which deactivates the process until its next timeslice.

```
RQM 5
```

result:
Pauses program for five seconds.

```
A = 13:05
RQM A
```

result:
Pauses program until 1:05 p.m.

SLEEP Statement - New

The SLEEP statement pauses a program for a specified amount of time, which can be expressed either as number of seconds or a specific time of day.

Syntax

SLEEP {expression}

expression	if expression is a numeric value, defines number of seconds SLEEP is to pause program; if expression is a string value in the form hh:mm, defines time of day program is to end pause

Description

SLEEP is a synonym for RQM.

If no expression is used, SLEEP pauses for one second.

If the value of expression is less than or equal to zero, a command is sent to the kernel to perform a kernel RQM, which deactivates the process until its next timeslice.

```
   SLEEP 5

result:
   Pauses program for five seconds.
```

```
   A = '13:05'
   SLEEP A

result:
   Pauses program until 1:05 p.m.
```

SORT Function - New

The SORT function sorts elements in a dynamic array. The sort is in ASCII sequence.

Syntax

SORT(var)

var evaluates to dynamic array

Description

The elements are sorted in ascending order, left justified.

The dynamic array is sorted by the highest system delimiter in the array. That is, if the dynamic array contains any attribute marks, the sort is by attributes; values and subvalues are unaffected and remain with the original attribute. If the dynamic array contains value marks and no attribute marks, the sort is by value; subvalues are unaffected and remain with the original value. If the dynamic array contains subvalue marks and neither attribute nor value marks, the sort is by subvalue.

```
   A = 'Cherries':AM:'Apples':VM:'Granny Smith':VM:'Delicious'
   S = SORT(A)

result:
   S = 'Apples':VM:'Granny Smith':VM:'Delicious':AM: 'Cherries'
```

There is an attribute mark, so the sort is according to attributes; the first value of each attribute determines the order.

```
   A = 'Rome':VM:'Granny Smith':VM:'Delicious'
   S = SORT(A)

result:
   S = 'Delicious':VM:'Granny Smith':VM:'Rome'
```

There are no attribute marks, but there are value marks, so the sort is by values.

SOUNDEX Function - New

The SOUNDEX function returns the soundex code for a specified string. The code consists of the first letter of the word, plus values for the next three consonants or combinations of consonants.

Syntax

SOUNDEX(expression)

expression	string to be used

Description

These codes can be useful for cross references based on words or names that sound alike. Soundex codes also overcome problems with upper and lower case, typographical errors, misspellings, etc.

The first value in the soundex code is the first alphabetic character in the string. Subsequent values in the soundex codes are numeric values given to consonants. If two or more characters with the same numeric value are adjacent, only one value is returned. Characters that are not consonants, other than the first character, are ignored. The following numeric values are returned by the function:

Letters	Code
a,e,i,o,u,h,w,y	null
b,f,p,v	1
c,g,j,k,q,s,x,z	2
d,t	3
l	4
m,n	5
r	6

Words with a similar arrangement of consonants have similar soundex codes, regardless of the actual spelling; similar sounding consonants may have the same soundex code. If two adjacent consonents have the same soundex code, only one instance of the code is returned. For example, the following words all have the same soundex code (L6):

Laura	Lora	Laurie	lorry
Lorrie	Lori	LARRY	

Example	Description
A = SOUNDEX('SMITH')	A = 'S53'
B = SOUNDEX('SMITHS')	B = 'S532'
C = SOUNDEX('SMITHSON')	C = 'S532'

SUM Function - New

The SUM function sums the lowest level elements in a dynamic array.

Syntax

SUM(expression)

expression	evaluates to dynamic array to be summed

Description

The elements at the lowest level are summed until the next higher level is encountered. For example, if the lowest delimited level is a subvalue, then all subvalues are summed within each value; when the value mark is encountered, the result is stored in that value.

The SUM function is useful to eliminate the lowest level of element present in a numeric dynamic array by combining at the next highest level. For example, if a dynamic array containing subvalues is summed, the lowest level in the returned array is value.

The SUM function can be used repeatedly until only one element remains.

```
  A = 1:AM:21:VM:22:AM:311:SVM:312:VM:321:SVM:322
  FOR I = 1 TO 3
   A=SUM(A)
  NEXT

result:
  A = 1:AM:21:VM:22:AM:623:VM:643
  A = 1:AM:43:AM:1266
  A = 1310
```

The first loop sums the subvalues to the next value delimiter (or end-of-string). The second loop sums the values to the next attribute delimiter (or end-of-string). The third loop sums the attributes.

SYSTEM Function - Enhanced

New values that list additional system information are now available for the SYSTEM function as follows:

Value	Information Returned	
13	reserved	
14	returns item in ERRMSG file called INPUT@; item contains messages used by system command UPDATE	
15	reserved	
16	cause of abort; valid only in subroutine called by TRAP ON THEN CALL statement; the following may be returned:	
	0	program termination - ABORT, END, or STOP statement in program
	1	BREAK key pressed
	2	END command entered in BASIC or system debugger
	3	OFF command entered in BASIC or system debugger
	4	SET-LOGOFF has been invoked and DSR has dropped, or the process has been logged off by another process
	5	CHAIN 'OFF' or EXECUTE 'OFF' statement in program
	Bnnn	BASIC ERRMSG item.id of error that caused trap
17	returns file and program name of current program	
18	returns TCL statement that invoked current program; statement is formatted as dynamic array. Elements in the statement are separated by attribute marks. If an element is enclosed in delimiters, the delimiters are removed. This function is equivalent to @SENTENCE	
19	port number of current process; equivalent to @USERNO	
20	date format:	
	0	USA (mm/dd/yy)
	1	European (dd/mm/yy)
	2	Swedish (yy/mm/dd)
21	level of nested EXECUTE; the highest level is 0; equivalent to @EXECLEVEL	
22	number of last hold file created by PRINT statement in current BASIC program; equivalent to @HOLDFILE	
23	privilege level of current process; equivalent to @PRIVILEGE	
24	returns current spooler assignment status; equivalent to @SPOOLOPTS	
25	returns 1 if external select list is active; equivalent to @SELECT	
26	current account name; equivalent to @WHO	
27	two character language code of the language assigned to current port; equivalent to @LANGUAGE	

TRAP ON THEN CALL Statement - New

The TRAP ON THEN CALL statement provides program control for certain unusual conditions.

Syntax

TRAP ON THEN CALL trap.subr

trap.subr	name of subroutine to be called if unusual condition occurs; no parameters can be passed

Description

The TRAP ON THEN CALL statement specifies the name of a subroutine to be called if one of the following conditions occurs:

- execution of ABORT, END, STOP, CHAIN 'OFF', or EXECUTE 'OFF' statement
- BREAK key pressed; trapped only if BREAK OFF is active
- execution of END or OFF command in BASIC or system debugger
- DSR drops or line is logged off by another process
- any runtime error that would normally cause the program to enter the BASIC debugger

The trap subroutine is **not** called when the TRAP ON statement is encountered; it is called only if one of the above unusual conditions occurs.

Although no parameters can be passed between the trap subroutine and the program in which the trap occurred, values can be passed through COMMON variables. The subroutine called by the trap may decide on further action; this could be, for example, terminating, logging off, or capturing the reason for the trap in a COMMON variable and RETURNing to the main program. Within the subroutine, SYSTEM(16) can be used to determine the reason for the trap and SYSTEM(17) used to determine the name of the program that called the subroutine.

If the trap subroutine executes its RETURN statement, the path the system follows depends on the reason for the trap. Table 4-1 lists the possible values for SYSTEM(16) and where the subroutine returns.

SYSTEM(17) returns the file and program name of the program that was executing at the time of the trap; the information is in the following format:

```
file.name prog.name
```

Once a trap is set up in a program, it remains set unless a subsequent TRAP ON is encountered. The last trap that was encountered at the current EXECUTE level is the active trap. A trap is passed to any subroutine that is called after the trap is set up; a trap that is set up in a subroutine is passed back to the calling program. For example, if one trap is set up in the main program, and a second is set up in a subroutine, the trap set up in the main program is in effect until the TRAP ON statement in the subroutine is encountered. The trap set up by the subroutine remains in effect, even after the subroutine returns to the main program.

BASIC programs that are CHAINed to after a trap is set up inherit the trap if the CHAIN statement includes the I option with the verb. PROCs and system commands that are chained to do not inherit traps.

Programs that are EXECUTEd do not inherit traps. However, if a TRAP ON is set up at one level and a condition that would produce a SYSTEM(16) value of 2, 3, 4, or 5 occurs in a lower EXECUTE level where no TRAP ON is set, the system passes control back to the trap subroutine in the EXECUTE level with the TRAP ON. The SYSTEM(16) value is preserved if it is 2, 3, or 4. If the condition would produce a value of 5 if the TRAP ON were in the same level, SYSTEM(16) returns a value of 3. (For an explanation of SYSTEM(16) values, see Table 4-1.)

During the execution of the trap subroutine, the trapping mechanism is disabled; this is to prevent an error within the subroutine from causing the system to get into an unbreakable loop. However, if DSR drops or the process is logged off by another process while the trap subroutine is executing, the condition is saved and the trap subroutine is reentered with the saved condition as soon as the trap subroutine RETURNs out of the current condition.

Table 4-1. SYSTEM(16) Values

SYSTEM(16) Value and Description		Normal Path After RETURN
0	Program termination - ABORT, END, or STOP statement in program	returns to program in which trap occurred and follows through on statement
1	BREAK key pressed when BREAK OFF is in effect	returns to program in which trap occurred
2	END command entered in BASIC or system debugger	returns to program in which trap occurred
3	OFF command entered in BASIC or system debugger	returns to program in which trap occurred
4	SET-LOGOFF has been invoked and DSR has dropped, or the process has been logged off by another process	returns to program in which trap occurred
5	CHAIN 'OFF' or EXECUTE 'OFF' statement in program	logs user off unless statement causing trap is in an EXECUTEd program **and** a TRAP ON has been set in a previous level. In this case, the RETURN enters the trap subroutine at that level and reports SYSTEM(16) = 3
Bnnn	BASIC ERRMSG item.id of error that caused trap	returns to TCL or, if in an EXECUTEd program, to previous level

The system determines the statement to return to as follows:

- to beginning of statement that caused trap if statement not completely executed
- to statement following the location of trap if trap occurred after statement was executed
- to INPUT prompt if trap occurred while waiting for input; if it is desired to force the program to return to the statement following the INPUT, the DATA statement can be used in the trap subroutine to stack data for the INPUT statement

Example	Description
TRAP ON THEN CALL TRAP.SUBR	
.	
.	
*	Debugger entered
*OFF	OFF typed in BASIC debugger
result: TRAP.SUBR is called when OFF is typed with SYSTEM(16) = 3. If RETURN statement in TRAP.SUBR is executed, program returns to statement that was being executed when debugger entered.	
TRAP ON THEN CALL TRAP.SUBR	
.	
.	
CHAIN 'OFF'	OFF is encountered in program.
result: Trap subroutine is entered with SYSTEM(16) = 5. If RETURN statement in TRAP.SUBR is executed, user is logged off.	
PROG1:	
TRAP ON THEN CALL TRAP.SUBR	
EXECUTE "RUN BP PROG2"	
	.
	.
PROG2:	No trap is specified in program
EXECUTE "RUN BP PROG3"	
	.
	.
PROG3:	No trap is specified in program
	.
	.
*	Debugger entered
*OFF	OFF typed in BASIC debugger
result: System returns to last level in which trap has been set and calls trap subroutine. In this example, this is the subroutine TRAP.SUBR specified in PROG1; SYSTEM(16) = 3. If the RETURN statement in TRAP.SUBR is executed, program returns to statement following EXECUTE statement in PROG1.	

TRIM Function - Enhanced

The TRIM function has two new forms: TRIMB, which removes only the trailing (back) spaces from a string, and TRIMF, which removes only leading (front) spaces from a string.

Syntax

TRIM{B|F}(string.expr)

string.expr	string being trimmed.

```
   A = '  A  B C   '
   A = TRIMB(A)

result:
   A = '  A  B C'

The back spaces are trimmed.
```

```
   A = '  A  B C   '
   A = TRIMF(A)

result:
   A = 'A  B C   '

The front spaces are trimmed.
```

USERTEXT Function - New

The USERTEXT function returns the text of a specified item in the USERMSG file.

Syntax

USERTEXT(item.id{,param1{, ... }})

item.id	the item.id of the USERMSG file item to be returned.
param	parameters to be passed to the USERMSG item.

Description

The USERMSG file follows the same the format as the ERRMSG file, but the items are created and maintained by the users. The USERMSG file is described in Section 2, O/S Enhancements.

The USERMSG file can contain multi-level data files, where each data file is in a different language. The system command SET-LANGUAGE is used to specify the particular data level that is used. The USERTEXT function formats the message appropriately for each language.

```
USERMSG item Welcome:

   001  E
   002  H,
   003  A
   004  L
   005  HIt is
   006  D

   X = USERTEXT('Welcome',CUSTNAME)
```

result:
This returns the text of "Welcome" after inserting the value of CUSTNAME into the message:

[Welcome], *custname*
It is 21 Jun 1989

WRITE Statement - Enhanced

The WRITE statement now supports the keyword TO as well as the standard ON. These two words are equivalent.

Syntax

WRITE var TO {file.var,}item.id {ON ERROR stmts}

WRITET Statement - Enhanced

An option has been added to WRITET that allows data in ASCII hexadecimal format to be written to tape as binary data.

Syntax

WRITET{X} var THEN/ELSE statements

X	indicates to convert data from hexadecimal to binary format

Description

This feature is intended to allow binary data to be written by BASIC programs.

WRITETX assumes all characters in the variable are hexadecimal. If any non-hexadecimal characters are encountered, they are ignored. Data integrity is the responsibility of the programmer.

For information on reading binary data from tape and converting it to hexadecimal, see the description of READT.

```
   WRITETX '31323334FF414243' ELSE STOP

result:
   123_ABC, where _ represents a segment mark, is written to
   tape.
```

BASIC Compiler Changes

The following changes have been made to the BASIC compiler:

- Compilation is faster and more efficient; for example, large programs generally compile at least twice as fast as previously
- Error messages produced by the BASIC compiler have been changed and now give more information when an error is encountered; the compiler also indicates where on the line it was scanning when it noted the error. For example, if the THEN/ELSE clause is missing in an OPEN statement, the compiler displays an error message similar to the following:

```
004 OPEN 'BP' TO BP
***                    ^ THEN or ELSE clause missing
```

- BASIC object code size has been increased to 57,534 bytes.
- The BASIC and COMPILE verbs, which are used to compile BASIC programs, have been changed as follows:
  - The A option, which provided a listing of assembled code, is no longer available
  - The E option, which provided an errors only listing, has been removed; the errors only listing is the default
  - The F option can be used with the M option to list internal variables and labels, including those created by IF/THEN and FOR/ NEXT loops; internal variables and labels are displayed preceded by an asterisk.
  - An I option has been added, which, if the L option is also specified, displays lines from $INCLUDEd programs as part of the listing
  - The format and information displayed by the M option has changed; the following is now displayed:

```
Symbol table is 2% full
Last variable is at 210
                 -------- V A R I A B L E S --------
  30  REPLY      40  FEXISTS       50  FTYPE        60  TIME
  70  HH         80  MM           100  N           150  MODE
                 ----------- L A B E L S -----------
  55  PRINTID    59  ERRORPR
                 ---------- E Q U A T E S ----------
BELL=CHAR(7)         CR=CHAR(13)         ESC=CHAR(27)
```

The variable locations are given as offsets from the beginning of the symbol table. The gaps in the table are either because a variable is a dimensioned array or because there are CALLs to subroutines between two definitions. The location of the last variable shown above the variables may be greater than the last location shown in the table for the same reasons. In addition, offsets 10 and 20 are never displayed; offset 10 is used for the internal default file variable and offset 20 is used for the internal default select variable. (The descriptors used for subroutines and for internal variables will be displayed if the F option is also specified.)

The number preceding each label is the line number where the label is defined. If the program is compiled with the C option, the line number is always 1; if there is a $NODEBUG directive in the program itself, the line number is always the line number of the $NODEBUG statement.

The following shows the display with both the F and the M options specified:

```
Symbol table is 2% full
Last variable is at 210
                ---------- V A R I A B L E S ----------
  10  *FV        20  *SV          30  REPLY        40  FEXISTS
  50  FTYPE      60  TIME         70  HH           80  MM
  90  N         150  MODE
                ------------- L A B E L S -------------
   5  *10001      5  *20002        6  *10003        8  *20003
  55  PRINTID    59  ERRORPR
                ------------ E Q U A T E S ------------
BELL=CHAR(7)         CR=CHAR(13)                 ESC=CHAR(27)
```

Notes

5 Multi-Lingual Capabilities

The multi-lingual capabilities of the Ultimate operating system allow each port on the system to receive and display information in any language that has been translated for the system.

The language translation process can extract phrases from menus, screen displays, and HELP messages used by UltiKit processes, phrases from ERRMSG and USERMSG files, and item.ids from the master dictionary. Once these phrases have been extracted, the language process is used to translate them into any desired language.

The language translation process includes report functions that can be used to display the extracted phrases, the translated phrases, and even phrases that are longer than the original phrase.

The default language is US (English as spoken in the United States).

The following system features support multi-lingual processing:

UltiKit

Languages are translated using the system application UltiKit. A new option, Language Menu, which provides access to the language translation process, is included in the UltiKit main menu for Revision 200. These changes are described in this section.

System Commands

A new system command, SET-LANGUAGE, changes the language setting for the port on which it is issued. This command is described in Section 3, System Commands.

BASIC Commands

Two new BASIC functions, ERRTEXT and USERTEXT, print messages based on the current language setting. Two new features, BASIC function SYSTEM(27) and system variable @LANGUAGE, return the current language setting. These functions are described in Section 4, Extended BASIC.

It is recommended that all messages generated by BASIC programs be placed in the USERMSG file, in order to facilitate translations.

System Files

The following system files contain separate data levels for each language that is defined for the system:

- ERRMSG - contains system messages
- PROCLIB - contains system programs and PROCs
- USERMSG - contains user-defined messages

The following UltiKit files also have separate data levels for each language that is defined for the system:

- UPD.CHECK.SUM - contains checksums for system and UltiKit files; these checksums vary for each language implementation
- UPD.DEV.DOC - contains help messages and documentation for UltiKit
- UPD.DOC - contains help messages generated for UltiKit menus
- UPD.MENUS - contains the data displayed by UltiKit menus
- UPD.PAINT.TABLES - contains the data displayed by the data entry screens created by UltiKit
- UPD.PHRASES - contains the phrases accumulated by the language process
- UPD.PROCS - contains the PROCs created by UltiKit
- UPD.PXREF - contains soundex-based cross-references for the phrases accumulated by the language process
- UPD.STATEMENTS - contains information used to process UltiKit data screens, reports, and forms

The language translation process in UltiKit is used to translate the information in the following files:

Area	**Files**
Menus	UPD.MENUS
Procs	UPD.PAINT.TABLES UPD.STATEMENTS
Help Messages	HELP-MSGS item in dictionary of UltiKit files
User Messages	USERMSG
Error Messages	ERRMSG
Master Dictionary	all item.ids from master dictionary of current account

Information in other files must be extracted and translated individually; the multi-lingual process cannot be used.

Requirements

The multi-lingual process requires Revision 200 or later of the Ultimate Operating System.

UltiKit applications written for previous revisions of the operating system function with the multi-lingual process and require no changes.

The multi-lingual process has no impact on any pre-existing software.

Creating Translations

The following procedure lists the steps needed to create a new language code:

1. Log on to SYSPROG or SECURITY account.
2. Invoke UltiKit. Select Language menu.
3. Create language code. See page 5-8 for description.

After the language code is created, the information in the files can be translated on account basis, including the files on the SYSPROG and SECURITY accounts, as follows:

1. Log on to account to be set to translated language.
2. Invoke UltiKit. Select Language menu.
3. Create local language code. See page 5-11 for description.
4. Extract phrases - see page 5-15 for description.
5. Translate phrases - see page 5-16 for description.
6. Replace phrases - see page 5-20 for description.
7. Invoke SET-LANGUAGE verb for each port that is to use the language. See Section 3 for description.

Repeat these steps for each account that is to use multi-lingual features.

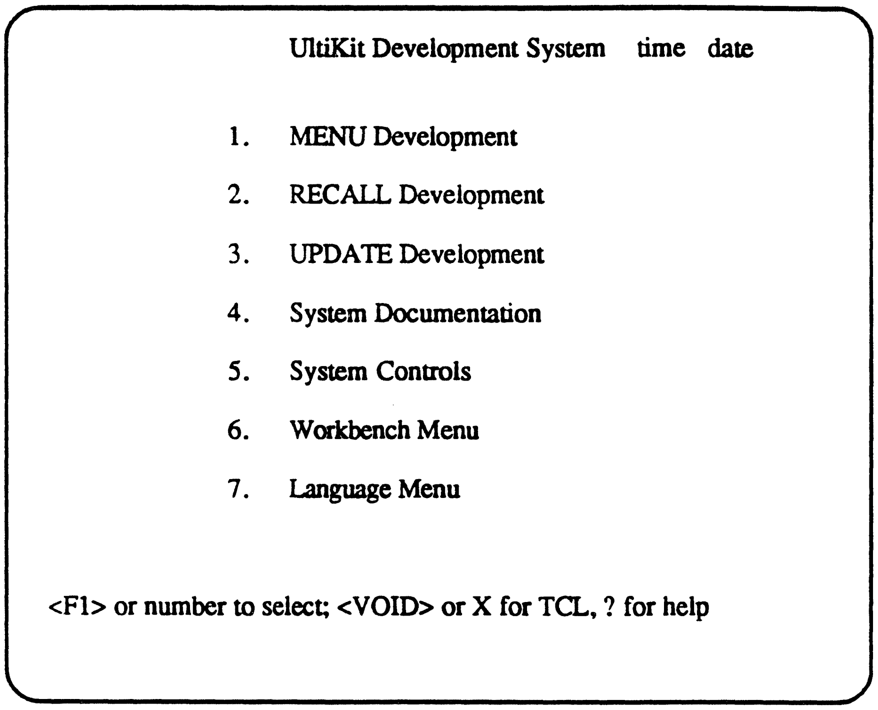
The following pages describe all the features of the UltiKit Language Translation process.

Note: *The language codes are defined by account, but the languages are enabled by port.*

Using the Language Translation Process

The language translation process is accessed from the UltiKit Main Menu. All the standard UltiKit features are available; for example, help can be invoked whenever a question mark (?) is displayed in the lower right corner. Cursor movement is standard. For more information on using UltiKit, see the *UltiKit User Guide.*

At the TCL prompt, enter **ULTIKIT**. A menu similar to the following is displayed:



```
                         UltiKit Development System    time   date

                    1.   MENU Development

                    2.   RECALL Development

                    3.   UPDATE Development

                    4.   System Documentation

                    5.   System Controls

                    6.   Workbench Menu

                    7.   Language Menu


  <F1> or number to select; <VOID> or X for TCL, ? for help
```

To use the language translation process, select option 7.

The Language System Menu menu is displayed as follows:

```
                    Language System Menu              time   date

               1.   Language Control Menu

               2.   Translation Menu

               3.   Listings Menu


   <F1> or number to select; <VOID> or X for TCL, ? for help
```

The language control menu contains options used to create new language codes, and to maintain the multi-level data files for the system and UltiKit files used by the language process. New language codes can be created only from the SYSPROG or SECURITY accounts; copying existing data to multi-level data files can be done from all accounts.

The translation menu contains options used to accumulate and translate phrases.

The listing menus contains options used to produce reports based on the language process.

Language Control Menu

The language control features are accessed by selecting option 1 from the Language Menu. In order to create new language codes, you must be logged on to the SYSPROG or SECURITY account. In order to set up an account to be multi-lingual, you must be logged on to that account. The language control menu that is displayed depends on the account that you are logged on to.

SYSPROG and SECURITY Account

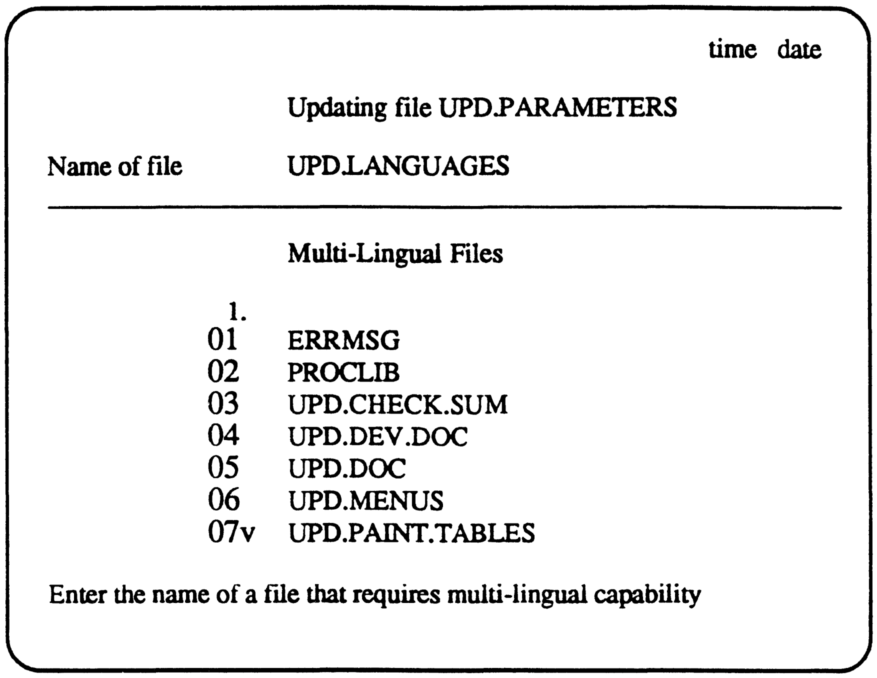
If you select the Language Control Menu option while logged on to the SYSPROG or SECURITY account, the following menu is displayed:

```
                    Language Security Menu              time   date

              1.    Define Multiple Data-Level Files

              2.    Create Language Codes

              3.    Copy Items from Existing Files

  <F1> or number to select; <VOID> or X for previous Menu, ? for help
```

These options are described on the following pages.

Define Multiple Data-Level Files

Option 1 on the Language Security Menu, Define Multiple Data-Level Files, is used to update the list of files that have multiple data levels because of the language process. The following screen is displayed:



```
                                                          time  date

                        Updating file UPD.PARAMETERS

Name of file            UPD.LANGUAGES
____________________________________________________________________

                        Multi-Lingual Files

                  1.
                 01     ERRMSG
                 02     PROCLIB
                 03     UPD.CHECK.SUM
                 04     UPD.DEV.DOC
                 05     UPD.DOC
                 06     UPD.MENUS
                 07v    UPD.PAINT.TABLES

Enter the name of a file that requires multi-lingual capability
```

This list defines the files for which UltiKit will build multi-levels. The v in the listing indicates that the list continues.

The system is initially in overtype mode when this screen is displayed. To insert a new file name, press the <INSERT> key (<F3> on most terminals); to delete a name, press <DEL>.

If additional files are added to this list, UltiKit must be modified to be able to use them. See the UltiKit User Guide for information on modifying

Create Language Codes

Option 2 on the Language Security Menu, Create Language Codes, is used to define or delete language codes. This option creates the multi-level data files and copies information from an existing set of files. It also creates a help message item in the dictionary of every UltiKit file on the system.

When option 2 is selected, the following screen is displayed:

```
                                                            time   date

                        Enter and Maintain Language Codes

          Language Code

     1.   Language Name

  ------------------------------------------------------------------------

     Cross Translation Files

          Code      Name
     2.

  Enter 2-character code for desired language or type '?' for list        ?
```

To list existing codes, enter a question mark (?) or press <HELP> (Shift-F1 on most systems). To define a new code or edit an existing code, enter the language code at the prompt.

If the language code that is entered already exists, and if phrases from it have been translated, the languages used as sources are listed under Cross Translation Files. This information is for display only and cannot be edited. The description in Language Name can be changed.

If the language code does not currently exist, enter a description of the language at the Language Name prompt. Press RETURN to file the item. A data level is created for each file specified in the list of files in Option 1 and information is copied from the default data level of each file. Additionally, an item called HELP-MSGS,lang.code is created in the dictionary of each file that already has a HELP-MSGS,US item.

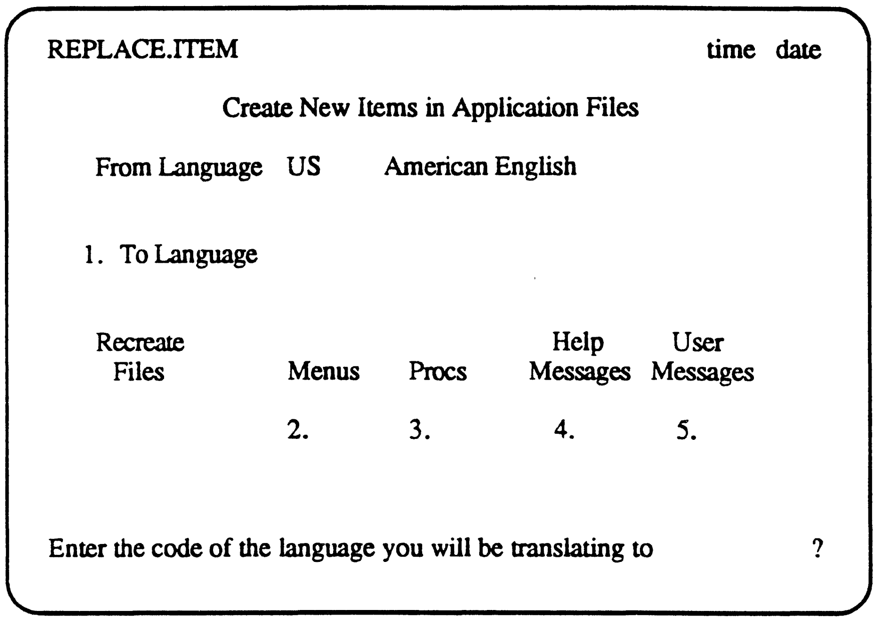
To delete an existing code, enter the code, then press the <DEL> key. The process asks for verification. *(Note: Once translations have been created for a code and cross translation files exist, the language code cannot be deleted.)*

Copy Items from Existing Files

Option 3 on the Language Security Menu, Copy Items from Existing Files, is used to update the multi-level data files. The following prompt is displayed:

From Language

Enter the language code of the source files. A screen similar to the following is displayed:



```
REPLACE.ITEM                                              time  date

                  Create New Items in Application Files

   From Language  US       American English


 1. To Language


   Recreate                                  Help      User
    Files          Menus      Procs       Messages  Messages

                   2.         3.            4.         5.


Enter the code of the language you will be translating to            ?
```

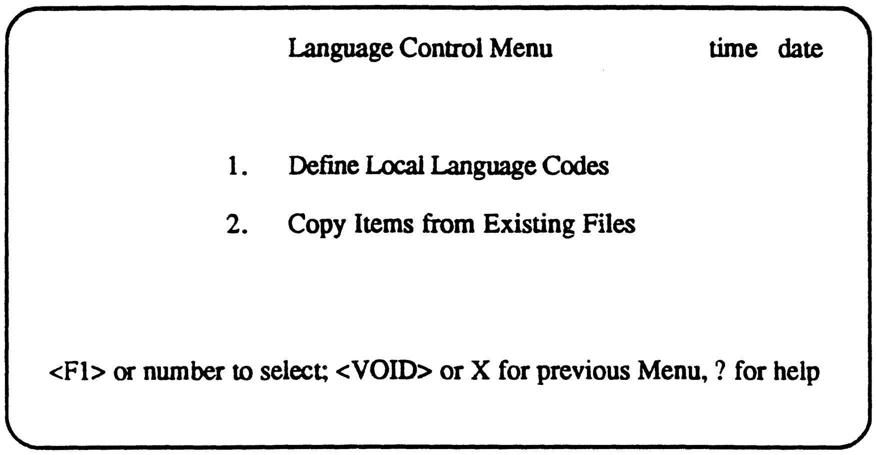
Enter the language code. Select the areas to be copied by entering the number of the field, then entering the letter Y. After the areas have been selected, enter the letters FI to start the copy; the language process then copies items from the specified areas to the To Language data level.

The copy overwrites existing items, including any translated phrases that have been replaced in the specified files. After the copy, you can use the replace option again to put the translated phrases back into the files.

Note: *The files are copied from the default files when the language code is first defined. The copy option is intended to be used if a default file changes after the code has been defined. The copy option can also be used to remove translated phrases from the file.*

User Accounts

If you select the Language Control Menu option while logged on to any account other than SYSPROG or SECURITY, the following menu is displayed:

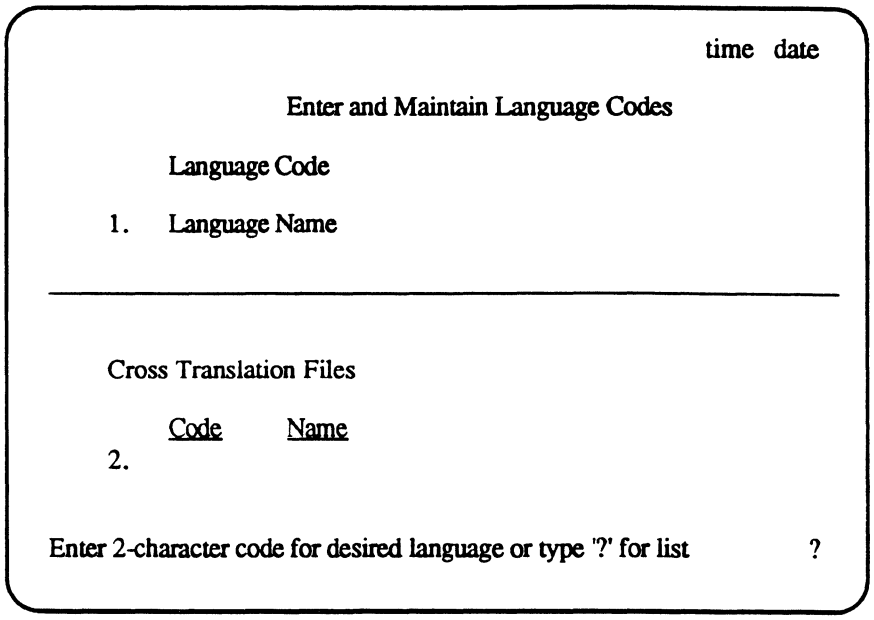


```
                    Language Control Menu              time  date


               1.   Define Local Language Codes

               2.   Copy Items from Existing Files



 <F1> or number to select; <VOID> or X for previous Menu, ? for help
```

This menu is used to set up translation codes for individual accounts. The language codes must have been created previously.

Define Local Language Codes

Option 1 on the Language Control Menu, Define Local Language Codes, is used to define language codes that can be used for the current account. This option creates the multi-level data files and copies information from an existing set of files. It also creates a help message item in the dictionary of every UltiKit file on the account. The following screen is displayed:



```
                                                         time   date

                        Enter and Maintain Language Codes

               Language Code

          1.   Language Name

   ____________________________________________________________________

          Cross Translation Files

               Code      Name
          2.

   Enter 2-character code for desired language or type '?' for list      ?
```

To list the available codes, enter a question mark (?) or press <HELP> (Shift-F1 on most systems). To specify a code, enter the language code at the prompt. The language name is displayed and if phrases from the specified language code have been translated, the language codes used as sources are listed under Cross Translation Files. This information is for display only and cannot be edited.

Press RETURN to file the item. A data level is created in the current account for each file specified in the list of files defined on the SYSPROG account and information is copied from the ERRMSG,US (default) file. Additionally, an item called HELP-MSGS,lang.code is created in the dictionary of each file that already has a HELP-MSGS,US item.

To delete an existing code, enter the code, then press the <DEL> key. The process asks for verification of any deletions. Local language codes can be deleted whether or not cross translation files exist.

New codes cannot be created from any account other than SYSPROG and SECURITY. If a code that does not exist is entered, a message similar to the following is displayed:

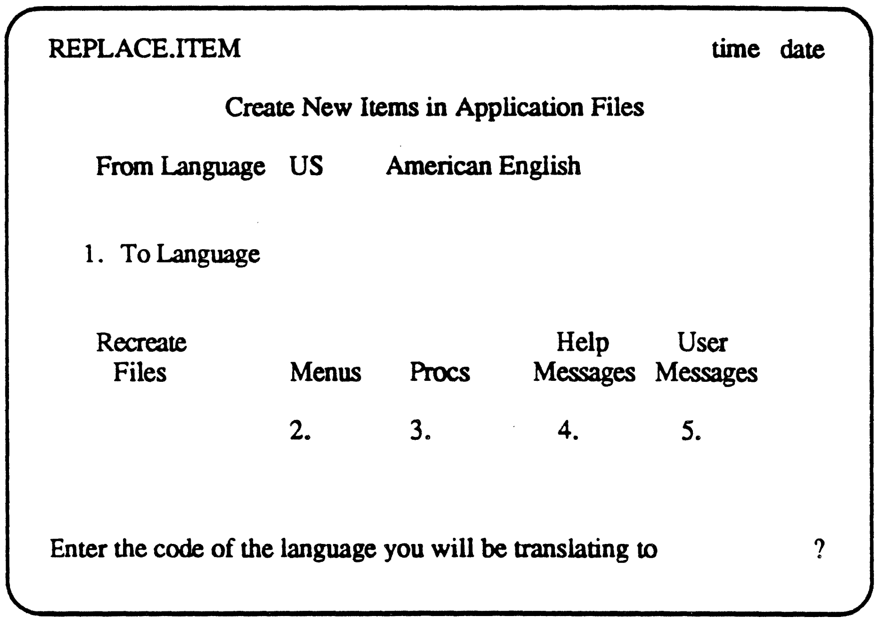
> You may not create a new language code from this account.

Copy Items from Existing Files

Option 2 on the Language Control Menu, Copy Items from Existing Files, is used to update the multi-level data files. The following prompt is displayed:

From Language

Enter the language code of the source files. A screen similar to the following is displayed:



```
REPLACE.ITEM                                                  time  date

                  Create New Items in Application Files

   From Language  US       American English


  1. To Language


   Recreate                                   Help       User
    Files          Menus       Procs        Messages   Messages

                   2.          3.           4.         5.


Enter the code of the language you will be translating to              ?
```

Enter the language code, then select the areas to be copied by entering the number of the field, then entering the letter Y. After the areas have been selected, enter the letters FI; the language process then copies items from the specified areas to the To Language data level.

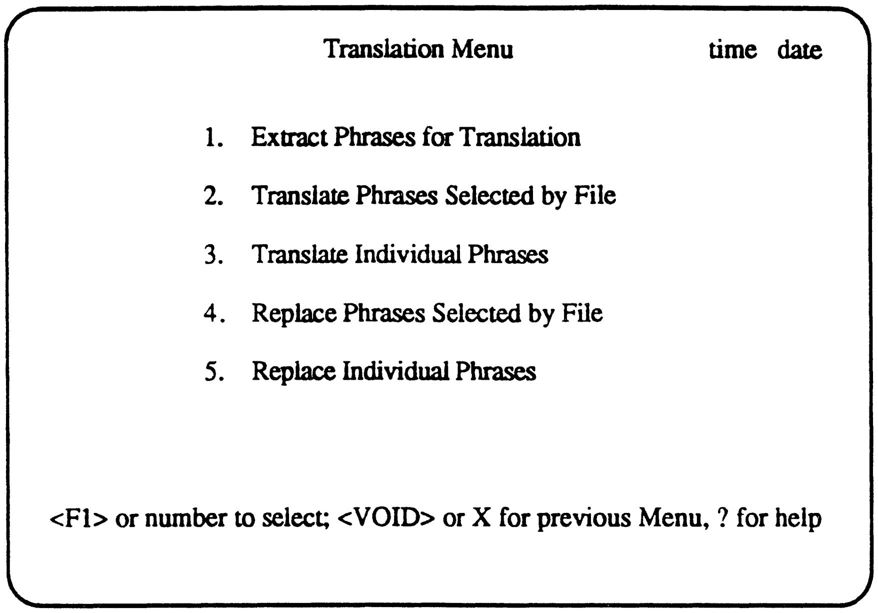
The copy overwrites existing items, including any translated phrases that have been replaced in the specified files. After the copy, you can use the replace option again to put the translated phrases back into the files.

Note: *The files are copied from the default files when the language code is first defined. The copy option is intended to be used if a default file changes after the code has been defined. The copy option can also be used to remove translated phrases from the file.*

Translation Menu

The Translation Menu is accessed by selecting option 2 on the Language System Menu. The Translation Menu contains the options to extract phrases, to translate phrases, and to replace the original phrase with the translated phrase.

When the Translation Menu option is selected, the following screen is displayed:



```
                         Translation Menu                    time   date


              1.  Extract Phrases for Translation

              2.  Translate Phrases Selected by File

              3.  Translate Individual Phrases

              4.  Replace Phrases Selected by File

              5.  Replace Individual Phrases



<F1> or number to select; <VOID> or X for previous Menu, ? for help
```

The language system extracts phrases from the following areas:

Area	**Files**
Menus	UPD.MENUS
Procs	UPD.PAINT.TABLES UPD.STATEMENTS
Help Messages	HELP-MSGS item in dictionary of UltiKit files
User Messages	USERMSG
Error Messages	ERRMSG
Master Dictionary	all item.ids from master dictionary of current account

A phrase is any text that is delimited by a system delimiter or that, within delimiters, is separated by at least two spaces. Each phrase is saved as an item in the UPD.PHRASES file for the specified From Language and appears only once in the file. A sequential number is used as the item.id.

The language process notes the file and item from which the phrase was extracted. If there is more than one occurrence of a phrase, each occurrence is noted in the item for the phrase.

A phrase is translated only once, and all occurrences of the phrase are replaced by that translation.

The following pages describe the Translation options.

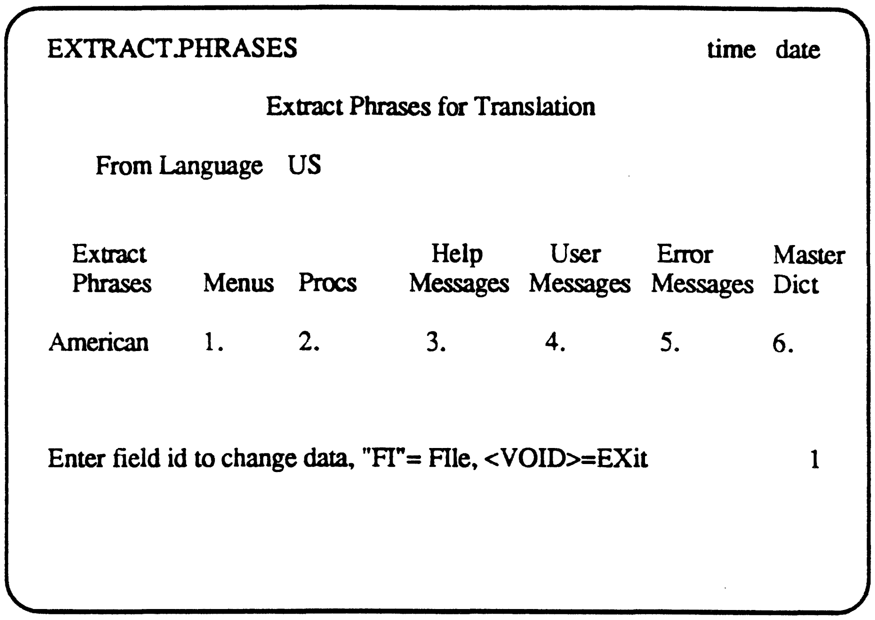
Extract Phrases for Translation

Option 1 on the Translation Menu, Extract Phrases for Translation, copies phrases from specified files in the current account to the UPD.PHRASES file. Each account has a Q-Pointer to this file, which is in the SYSLIB account.

The following prompt is displayed:

From Language

Enter the language code of the source files. A screen similar to the following is displayed:



```
EXTRACT.PHRASES                                          time  date

                       Extract Phrases for Translation

  From Language   US


 Extract                             Help      User      Error     Master
 Phrases     Menus    Procs      Messages  Messages  Messages  Dict

American     1.        2.           3.         4.         5.        6.



Enter field id to change data, "FI"= FIle, <VOID>=EXit                1
```

Phrases are extracted from each area using the From Language data level of the appropriate files.

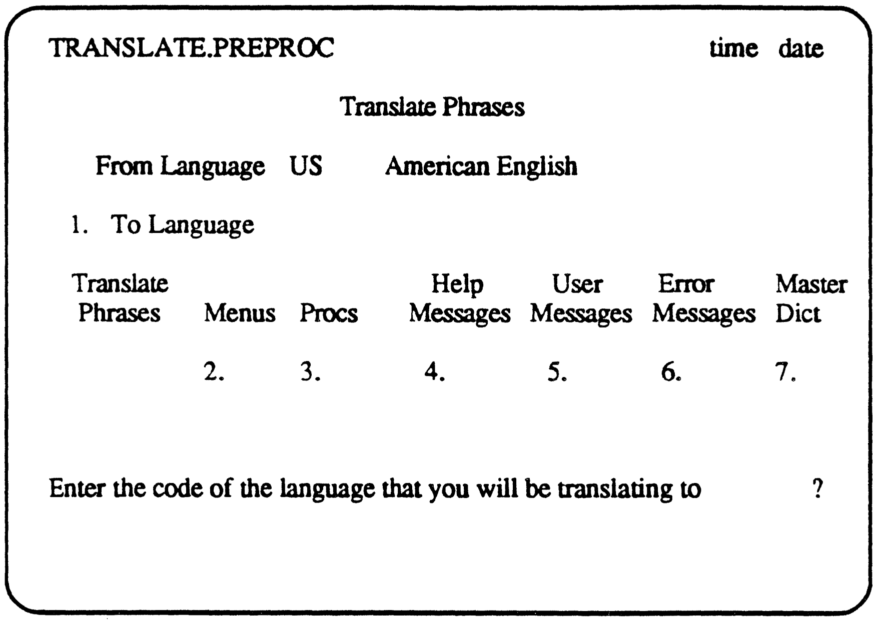
Select the areas for extraction by entering the number of the field, then entering the letter Y. After the areas have been selected, enter the letters FI; the language process then extracts all phrases from the specified areas.

Translate Phrases Selected by File

Option 2 on the Translation Menu, Translate Phrases Selected by File, presents phrases to be translated for a specific area. Only those phrases that have not been translated before are presented. The following prompt is displayed:

From Language

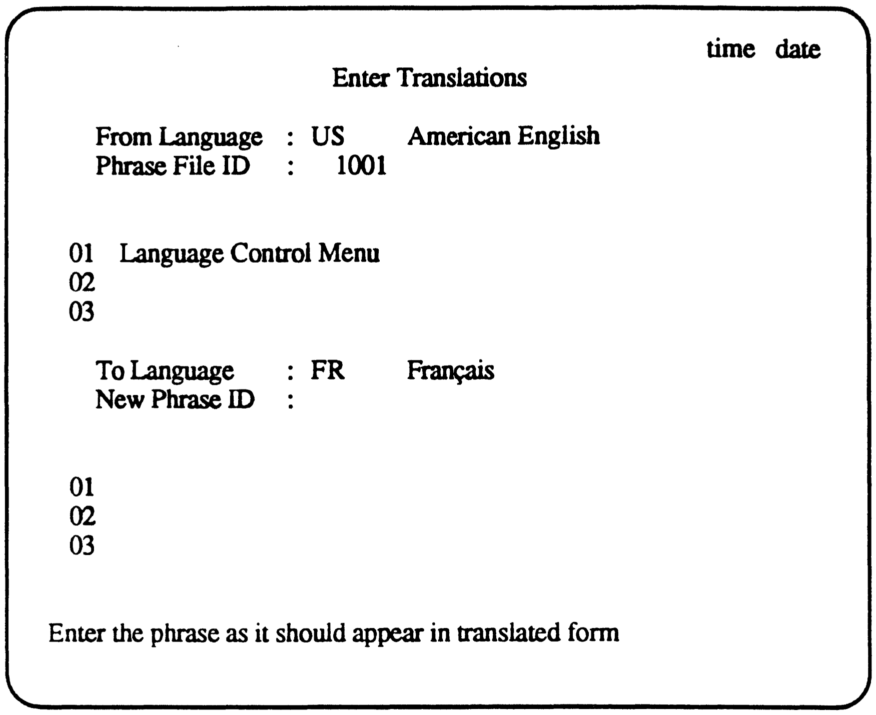
Enter the language code of the source files; it should be the same as the language code specified in the extraction process. A screen similar to the following is displayed:


```
TRANSLATE.PREPROC                                          time  date

                            Translate Phrases

  From Language  US      American English

 1.  To Language

 Translate                     Help      User      Error     Master
  Phrases    Menus  Procs    Messages  Messages  Messages  Dict

              2.     3.        4.        5.        6.        7.


Enter the code of the language that you will be translating to        ?
```

The To Language code must have been defined previously.

Enter the language code, then select the areas to be translated by entering the number of the field, then entering the letter Y. After the areas have been selected, enter the letters FI.

The language process then offers each phrase from the specified area for translation. Only phrases that have not been translated previously are offered. The screen is similar to the following:



```
                                                         time  date
                           Enter Translations

   From Language  :  US      American English
   Phrase File ID :    1001


01  Language Control Menu
02
03

   To Language    :  FR      Français
   New Phrase ID  :


01
02
03


Enter the phrase as it should appear in translated form
```

Enter the translation. If the translation is longer than the line, press <CR> to continue the translation on the next line.

After a phrase is translated, the translation is written into the data level of the UPD.PHRASES file for the To Language. A phrase needs to be translated only once, regardless of the number of times it appears in the system. The language translation process notes each occurrence of the phrase. The process also notes the number of characters in the original phrase and in the translated phrase. Those translated phrases that are longer than the original phrases can be listed using the Collisions Report on the Listings Menu.

To change the translation of a specific phrase, use the option Translate Individual Phrases.

Translate Individual Phrases

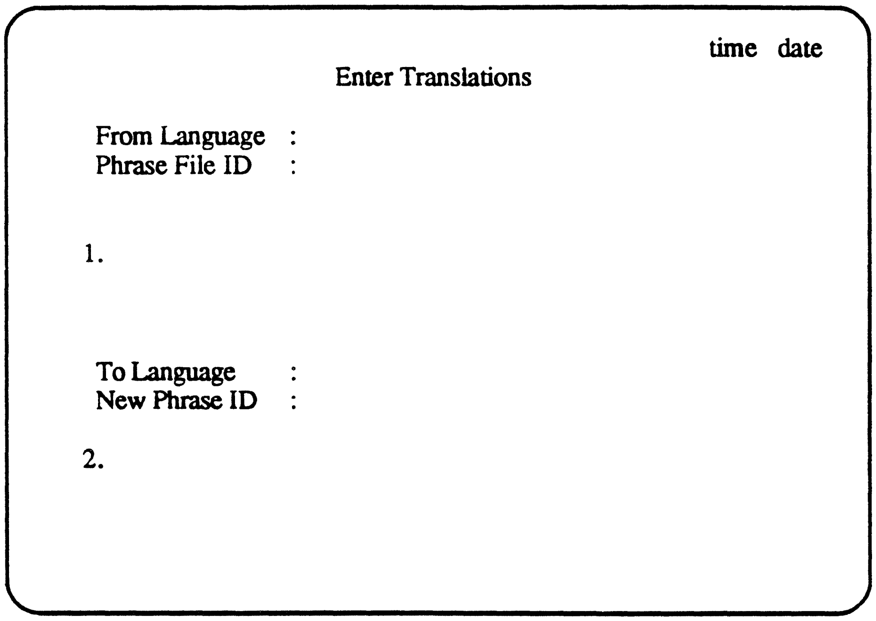
Option 3 on the Translation Menu, Translate Individual Phrases, is used to translate specified phrases. It can be used to change previous translations as well as translate phrases that have not been previously translated.

The following prompts are displayed:

From Language

1. To Language

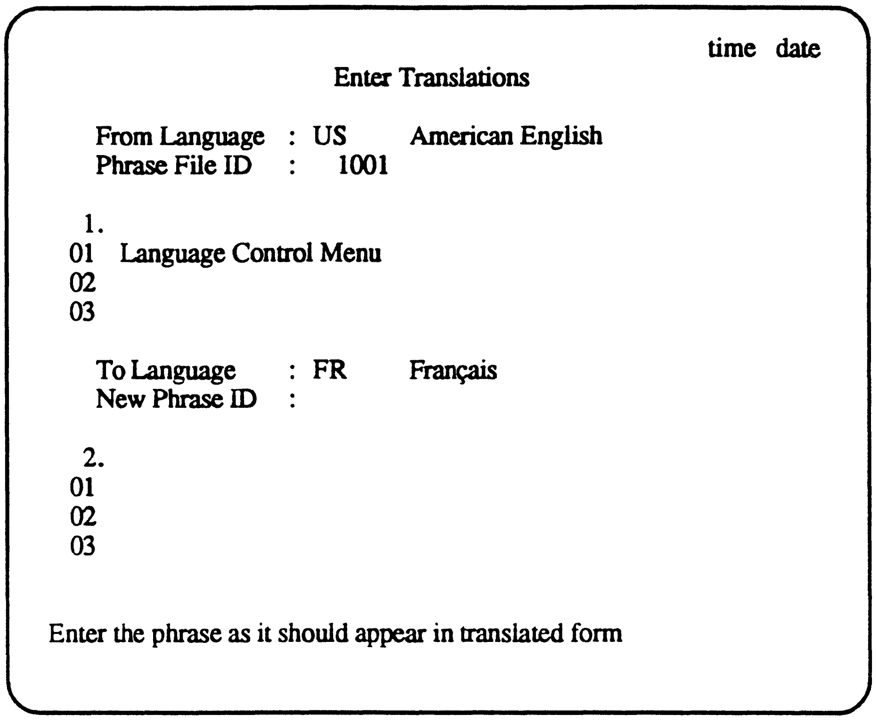
The language code of the From Language should be the same as the language code specified in the extraction process. After both codes are entered, a screen similar to the following is displayed:



```
                                                        time   date
                                  Enter Translations

      From Language   :
      Phrase File ID  :


    1.



      To Language     :
      New Phrase ID   :

    2.
```

The phrase file id is the item.id of the phrase in the data level of the UPD.PHRASES file for the From Language. To list the item.ids, enter ? or press <HELP>. The help screen can be used to list the phrases by item.id or specific text.

After the phrase id is entered, the phrase and associated translation, if any, are displayed, similar to the following:



```
                                                        time  date
                         Enter Translations

    From Language  :  US       American English
    Phrase File ID :    1001

  1.
 01  Language Control Menu
 02
 03

    To Language    :  FR       Français
    New Phrase ID  :

  2.
 01
 02
 03


 Enter the phrase as it should appear in translated form
```

Enter the translation. If the translation is longer than the line, press <CR> to continue the translation on the next line.

After a phrase is translated, the translation is written into the data level of the UPD.PHRASES file for the To Language. A phrase needs to be translated only once, regardless of the number of times it appears in the system. The language translation process notes each occurrence of the phrase. The process also notes the number of characters in the original phrase and in the translated phrase. Those translated phrases that are longer than the original phrases can be listed using the Collisions Report on the Listings Menu.

The system then prompts for the phrase file id of the next phrase to translate.

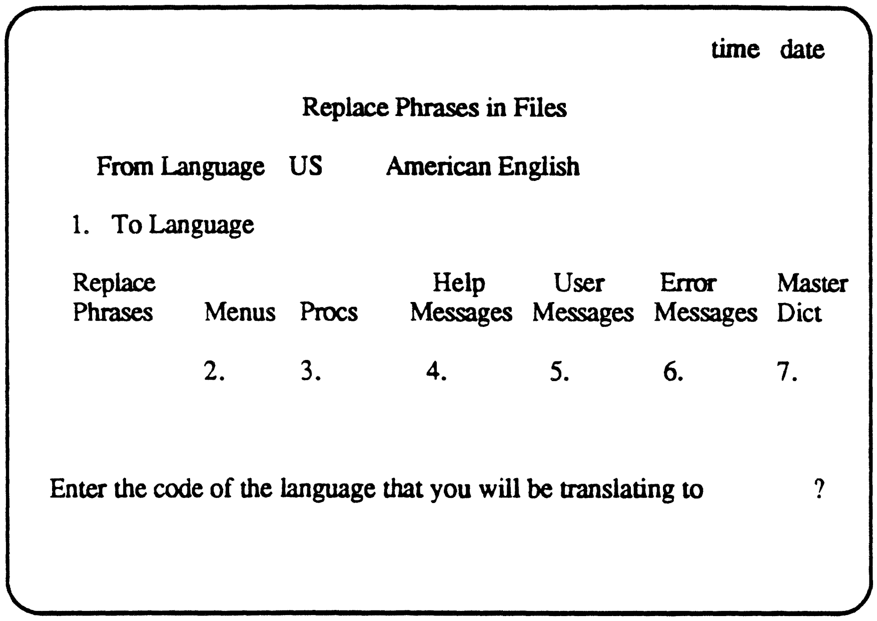
Once translations have been used to replace phrases, any changes that are made should be made using the translation language as both the From and To Language.

Replace Phrases Selected by File

Option 4 on the Translation Menu, Replace Phrases Selected by File, replaces all the phrases in the data levels of the To Language files specified in this option. The following prompt is displayed:

From Language

Enter the language code of the source files; it should be the same as the language code specified in the extraction and translation processes. A screen similar to the following is displayed:



```
                                                         time  date

                         Replace Phrases in Files

     From Language   US      American English

   1.  To Language

   Replace                      Help      User      Error     Master
   Phrases    Menus  Procs    Messages  Messages  Messages  Dict

               2.     3.        4.        5.        6.       7.


 Enter the code of the language that you will be translating to        ?
```

The To Language code must have been defined previously.

Phrases are replaced from each area using the From Language data level of the appropriate files.

Select the areas for replacement by entering the number of the field, then entering the letter Y. After the areas have been selected, enter the letters FI. The phrases are replaced as follows:

1. The system selects a phrase from the data level of the From Language UPD.PHRASES file. This item points to an item in the To Language UPD.PHRASES file.

2. The system gets the translated phrase from the data level of the To Language UPD.PHRASES file.

3. The system locates all occurrences of the untranslated phrase in the files indicated in the To Language item and replaces them with the translated phrase.

To use the translated files, use the SET-LANGUAGE verb, which is described in Section 3, System Commands.

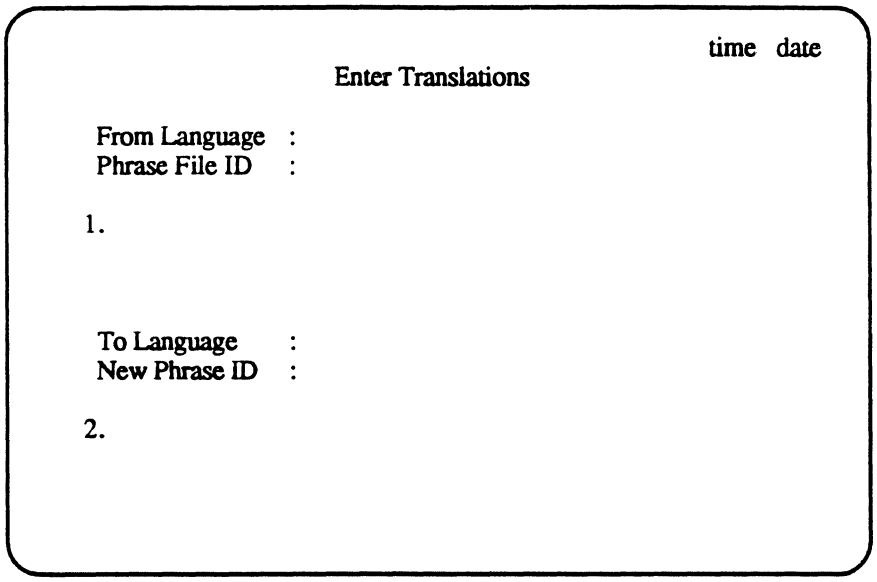
Replace Individual Phrases

Option 5 on the Translation Menu, Replace Individual Phrases, is used to replace phrases one at a time. The following prompts are displayed:

From Language

1. To Language

The language code of the From Language should be the same as the language code specified in the extraction and translation processes. After both codes are entered, a screen similar to the following is displayed:



```
                                                        time   date
                              Enter Translations

    From Language  :
    Phrase File ID :

  1.


    To Language    :
    New Phrase ID  :

  2.
```

The phrase file id is the item.id of the phrase in the data level of the UPD.PHRASES file for the From Language. To the list item.ids, enter ? or press <HELP>.

After the phrase id is entered, the associated phrase and translation, if any, are displayed, similar to the following:

```
                                                          time  date
                               Enter Translations

    From Language   :  US       American English
    Phrase File ID  :    1001

  1.
 01  Language Control Menu
 02
 03

    To Language     :  FR       Français
    New Phrase ID   :

  2.
 01  (translation)
 02
 03

Enter field id to change data, <CR> = FIle, <VOID> = EXit          2
```

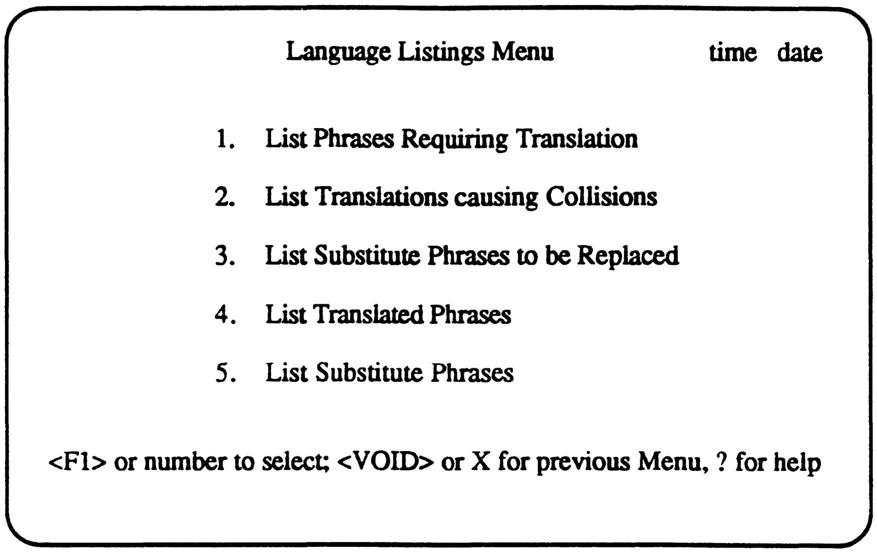
To replace the phrase, press <CR>; to exit press the <VOID> key (<F2> on most terminals).

The translated phrase is written into all the data levels of the files for the To Language that contain the original phrase.

The system then prompts for the Phrase File ID of the next phrase to translate.

Listings Menu

The Language Listings Menu is used to list the results of various options of the language translation process. It is accessed by selecting option 3 from the Language Menu.



```
                    Language Listings Menu              time  date

               1.  List Phrases Requiring Translation

               2.  List Translations causing Collisions

               3.  List Substitute Phrases to be Replaced

               4.  List Translated Phrases

               5.  List Substitute Phrases

  <F1> or number to select; <VOID> or X for previous Menu, ? for help
```

The language listings use the standard UltiKit printer selection screen. For information on using this screen, see the *UltiKit User Guide.*

List Phrases Requiring Translation

Option 1 on the Language Listings Menu, List Phrases Requiring Translation, lists the phrases that have been extracted, but not yet translated. The following prompts are displayed:

From Language

To Language

The report is in alphabetical order by phrase and is similar to the following:

```
PHRASES.LIST                        Time    Date              Page n
                               *** SYSLIB ***
                          List of phrases to be Translated
UPD.PHRASES.......Phrase...................................Source............


1287              Average no. of characters in data...  H,UPD.PARAME
                                                        TERS
1103              Check Sums                            M,UPD.SYSDOC
1052              Component                             M,UPD.DEV
1053              Create                                M,UPD.DEV
  .
  .
```

The Source column lists the files and item names in which the phrase was found. A one-digit character indicates either the file name or, if the code is the letter H, indicates that the phrase came from the HELP-MSGS item. The following codes are used:

Code	File/Item name
D	Master Dictionary
E	ERRMSG file
H	HELP-MSGS item
M	UPD.MENUS file
P	UPD.PAINT.TABLES file
S	UPD.STATEMENTS file
U	USERMSG file

List Translations causing Collisions

Option 2 on the Language Listings Menu, List Translations causing Collisions, lists the phrases with translations that are longer than the original phrase. The following prompts are displayed:

From Language

To Language

The report is in alphabetical order by original phrase and is similar to the following:

```
COLLISIONS.LIST                         Time    Date              Page n
                                  *** SYSLIB ***
                          List of phrases that create collisions
UPD.PHRASES ...... Collision with ................................ ........Collision


1003               Copy Items from Existing Files                    -8
                    (phrase that is too long would go here)
1002               Define Local Language Codes                       -13
                    (phrase that is too long would go here)
```

The phrase that is too long is truncated to 47 characters, if necessary.

The Collision column indicates the difference in the number of characters between the original and the translation.

List Substitute Phrases to be Replaced

Option 3 on the Language Listings Menu, List Substitute Phrases to be Replaced, lists those phrases that have been translated into the same language and have not yet been placed in the files from which the phrases were extracted. The following prompt is displayed:

From which Language

The report is in alphabetical order by original phrase and is similar to the following:

```
REPLACE.LIST                           Time    Date              Page n
                                  *** SYSLIB ***
                            List of phrases to be Replaced
UPD.PHRASES  Substituted with.................................Substituted
                                                               by ID#

1004          Listing of subroutines by system code           1005
              (substitute phrase would go here)
```

Substitute phrases are intended to update translations that have already replaced phrases from the original language.

The substitute phrase is truncated to 47 characters, if necessary.

List Translated Phrases

Option 4 on the Language Listings Menu, List Translated Phrases, lists all the phrases that have been translated to a specified language. The following prompts are displayed:

From Language

To Language

The report is in alphabetical order by original phrase and is similar to the following:

```
TRANSLATIONS.LIST                      Time    Date            Page n
                                 *** SYSLIB ***
                             List of Translated Phrases

UPD.PHRASES  Translations ........................... Translation ID

        1287  Average no. of characters in data...          1002
                (translation would go here)

        1103  Check Sums                                    1032
                (translation would go here)

        1052  Component                                     1222
                (translation would go here)

        1053  Create                                        1007
                (translation would go here)
```

The Translation ID is the item.id of the translated phrase in the UPD.PHRASES file for the translated language.

The translated phrases are truncated to 47 characters, if necessary.

List Substitute Phrases

Option 5 on the Language Listings Menu, List Substitute Phrases, lists all the phrases that have been translated back into the same language. The following prompts are displayed:

From which Language

The report is in alphabetical order by original phrase and is similar to the following:

```
SUBSTITUTES.LIST                Time    Date              Page n
                              *** SYSLIB ***
                          List of Substitute Phrases
UPD.PHRASES  Substituted with..............................................

1004          Listing of subroutines by system code
               1005 (substitute phrase would go here)
```

The substitute phrase is truncated to 47 characters, if necessary.

6 IBM Enhancements

Enhancements for the IBM-specific implementation include the following:

- performance and scheduler enhancements
- TERM-VIEW not supported
- Phase II HIFAS controller enhancements
- Series/1 terminal support enhancements
- security by system serial number
- block-size option for RP-READ command

Performance and scheduler enhancements have been added to increase the overall efficiency of the system. These changes are internal.

The new system command TERM-VIEW described in Section 3 is not supported on the IBM implementation.

The other enhancements are described on the following pages.

Note: *The revision number for the IBM implementation is 204.*

HIFAS Enhancements

The HIFAS (HIgh Function ASCII Subsystem) system supports ASCII terminals in both 3270 block mode and ASCII mode; it is available for 9370 IBM models. The HIFAS Phase II enhancements provide improved performance in both throughput and response time, as well as improved reliability and serviceability. The HIFAS now includes support for full flow control using X-ON/X-OFF and provides improved high-speed device support. The standard Ultimate system command SET-BAUD can be used to specify baud rates and line characteristics.

The HIFAS II system consists of one to eight input/output processor (IOP) units. Each IOP unit can have up to four slave boards called IOAs connected to it and each IOA can have up to eight terminals connected to it. This gives a total possible configuration of 256 interactive ports. See Figure 6-1. For information on setting up the HIFAS unit, see the *Ultimate IBM 9370 Operation and Maintenance Manual.*

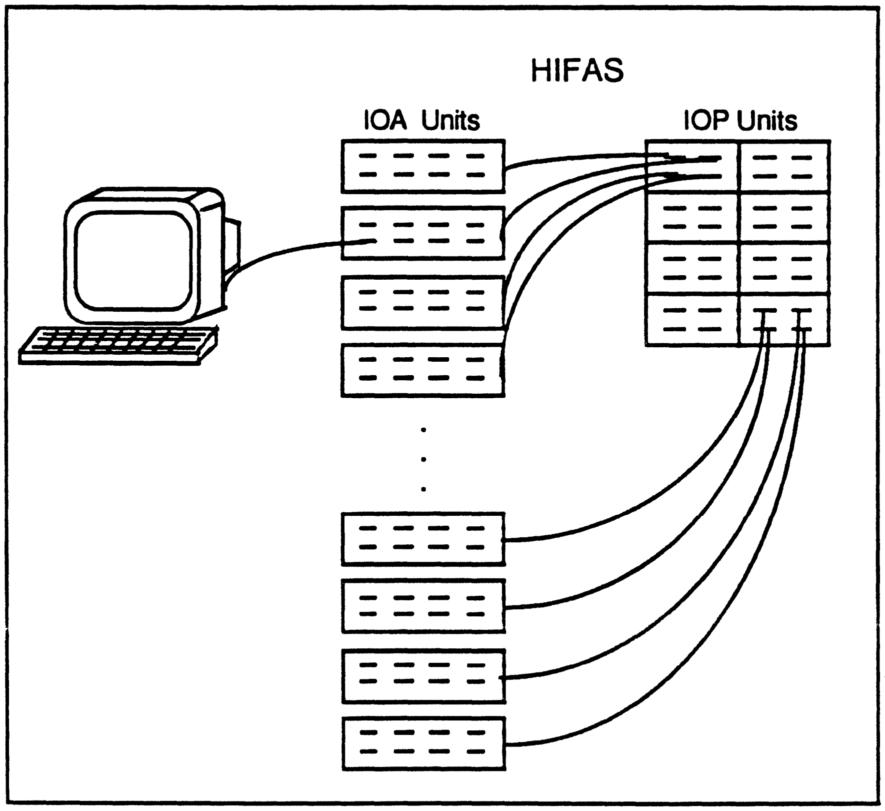


Figure 6-1. HIFAS II System

Terminal Groups

HIFAS II implements a concept called 'terminal group'. Terminals connected to the HIFAS can be grouped and each group can be associated with a different virtual machine. Each virtual machine may be running under the Ultimate Operating System or another operating system. Up to eight groups of terminals can be configured on each IOP unit. For information on configuring the terminal groups, see the *Ultimate IBM 9370 Operation and Maintenance Manual.*

Terminal Modes

The mode in which each terminal is to operate is specified using the Ultimate system command TERM with the MODE keyword. The parameters 3270 and ASCII are now supported for HIFAS II. The particular characteristics of the mode depend on the parameters that were specified when the terminal was installed. For information on using TERM, see the *Ultimate System Commands Reference Guide.*

Flow Control Requirements

The HIFAS includes a 128-byte buffer for each port, which is used for high-speed bursts of data. The HIFAS sends an X-OFF character to the sending device when the buffer is within 25 characters of being filled. If characters are sent after the buffer is filled, the extra characters are lost; HIFAS does not notify the sending application. An X-ON character is transmitted when the buffer is emptied.

The HIFAS also includes a 1920-byte buffer for each port, which is used for data transmitted at a slower rate of speed.

On output, any device that will send an X-OFF character to the HIFAS must be able to buffer at least two characters after the X-OFF character is sent. The size of the buffer may need to be larger, depending on the number and speed of devices between the HIFAS port and the device.

The HIFAS does not support DTR/RTS protocols.

Baud Rates

The Ultimate system command SET-BAUD can be used to set the baud rate for each port connected to the HIFAS. However, SET-BAUD cannot be used to set the auto-baud feature of 9370 system; this feature must be set using the correct Programmer's Function screen.

The following baud rates are supported for the HIFAS controller:

19200	1200
9600	600
7200	300
4800	150
3600	110
2400	75
1800	50

Note: *The baud rate cannot be set when the terminal mode is set to 3270; when the mode is changed from ASCII to 3270, the last baud rate in effect is maintained.*

Series/1 Terminal Enhancements

The Series/1 is used to connect ASCII terminals to IBM System/370 systems. The Ultimate implementation that supports this controller has been rewritten and provides improved reliability, availability, and serviceability.

The new version includes support for full flow control using X-ON/X-OFF and provides improved support for high-speed devices such as wand readers, PCs, and file transfer interfaces.

Users of CPU model 4955, please note: These enhancements require a clock; therefore, if you have not already done so, you must add a timer device, Model Number 7840 to your system, before upgrading to Revision 200. All other CPU models already have a clock.

No other changes are necessary at the user level.

Supported Configurations

The following devices and features are supported:

Feature	Model Number
S/370 Attachment	1200 (required)
Channel Repower	1565 (required if device 4959 is installed)
Communications Indicator Panel	2000
8-Line FPMLCC Controller	2095
4-Line RS-232 Attachment	2096
8-Line RS-422 Attachment	RPQ DO2350
I/O Expansion Unit	4959
Diskette Drive (IBM CE use)	4964
S/370 Channel Termination	4993 (required)
Programmer Console	5655 (required)
Timer	7840 (required on CPU model 4955)

Number of Users

The number of users that can be supported has been increased.

Note: *The preliminary estimates indicate that up to a maximum of 248 can be supported; however these figures are not confirmed. The following lists these preliminary maximum configurations by CPU model number:*

Model	Number of Devices
4954	64
4955	64
4956B	128
4956E	160
4956K	248

Using high-speed devices may reduce the total number of devices that can be supported, depending on the aggregate data rates.

Line Characteristics

The following line characteristics are supported:

data bits	7 or 8	
stop bits	1 or 2	
parity	odd, even, or none	
baud rates	19200	2000
	9600	1800
	4800	1200
	3600	600
	2400	300

The Ultimate system command SET-BAUD is used to specify the line characteristics for each terminal.

Flow Control Requirements

The new Series/1 software includes full support for flow control using X-ON/X-OFF. The buffer that receives input has been increased to between 126 and 510 bytes, depending on the number of terminals and the amount of available memory.

The Series/1 controller sends an X-OFF character to the sending device when the buffer that is receiving data has been half filled. It continues to send an X-OFF character each time it receives another input character. (If the controller is transmitting a message, it does not send the X-OFF character.)

When the final character that fits in the buffer is received, the controller sends an ASCII bell character to the sending terminal. If characters are sent after the buffer is filled, the extra characters are lost and ASCII bell characters are sent to the sending terminal.

On output, the Series/1 sends at most four additional characters after it receives an X-OFF character. This means that any device that sends an X-OFF signal must be able to buffer at least four characters. However, Ultimate recommends that buffers for devices that are not directly connected to the controller (for example, communications devices) have at least 256 bytes each.

Security Enhancements

A new security procedure for IBM systems is used with Revision 200 of the Ultimate Operating System. This new security procedure requires that the system administrator of a new system obtain an Access Code from Ultimate Technical Support, and then enter this code into the system.

A temporary access code is included with each new system or upgrade. This code must be entered into the system the first time it is coldstarted; a message is displayed on port 0 outlining this procedure. Until the code is entered, only port 0 can function.

A permanent access code must be requested from Ultimate Technical Support within 45 days of first entering the temporary code.

The permanent code is entered into the system in the same manner as the temporary code.

The following procedure describes the process:

1. Upgrade or install your system according the installation procedures.
2. Coldstart the system. Only port 0 is active.
3. Have the temporary access code available.
4. Log on to the SECURITY account. The Security menu is displayed, similar to the following:

```
*****************************************************
*              The ULTIMATE Accounts Manager        *
*              Version 1.5          10/06/88        *
*****************************************************
1.  System Security Specifications
2.  Terminal Security Specifications
3.  Create/Update an Account
4.  Rename an Account
5.  Account Save to Tape
6.  Account Restore from Tape
7.  Delete an Account
8.  Display Terminal Logon Failures
9.  Security System Documentation
10. Delete Q-pointer Account
11. Enter Access Code for this machine
Enter Selection, 'TCL', or 'OFF'
```

To invoke the process to enter the access code, select option 11, Enter Access Code for this machine. Alternatively, you can enter the following command at TCL:

```
ACCESS-CODE
```

5. The following is displayed:

```
While this program does check the general format of an access code entered,
it does NOT verify that the code given is valid. Please make sure that the
code is entered exactly as received from Ultimate support.
* Note: the date displayed with the access code is the date the code was
entered. It is NOT an expiration date!
Cpu id: nnnnnnnnn

Enter new access code or <CR>:
```

   Enter the access code exactly as it is given.

6. Shutdown and re-IPL the system.

7. Call Ultimate Technical Support at (201) 887-2721 for your permanent access code. To enter the permanent access code, follow steps 4 through 6.

This process works only from the SECURITY account on port 0.

Block-Size Option for RP-READ Command

The block size option for the RP-READ command can now be used to specify the physical record and block size of records read from the VM spool. The record size must be a multiple of the block size. Physical blocks are combined, if necessary, to form the specified record size. For example, if the physical block size is 80 and the record size is 160, two physical blocks are used for one record.

The default record and block size is 80.

7 LSI11 Enhancements

The LSI11 enhancements for Release 200 of the Ultimate Operating System consist of new diagnostics for the Enhanced Co-Processor boards. There are two diagnostics, one for ECP board and one for the EECP board.

ECP Diagnostics Description

The Enhanced Co-Processor (ECP) Diagnostics test the ECP board itself, the cable connecting the board to the LSI memory, the ECP instruction set, and for 3040 systems, the 2- or 4-Mb memory.

The diagnostics perform the following functions:

- test data patterns are loaded into ECP Writable Control Storage (WCS) RAM, read back, and verified
- the diagnostic functional code is loaded in WCS RAM and the loaded code is verified
- commands are sent between the LSI host and the ECP to test the hardware interface
- a menu is loaded that allows the user to specify the tests and other functions to be performed on the ECP

The ECP Diagnostics are available from disk or from a standalone bootable tape.

Using the Diagnostics

The following procedure outlines the steps to loading and using the ECP diagnostics.

1. Boot the system from either disk or the standalone diagnostics tape.

   If you booted the system from disk, the following System Startup Options menu is displayed:

```
This is the Ultimate Operating System

     Bootloader revision #nn

     System Startup Options:
     (B)oot
     (D)iagnostics Monitor
     (O)ff-line Monitor
     (U)tilities Monitor
     (W)armstart

     Enter Option(s) or ? for help:
```

   If you booted the system from a standalone tape, the following System Startup Options Menu is displayed:

```
This is the Ultimate Operating System

     Bootloader revision #nn

     System Startup Options:
     (B)oot
     (C)oldstart
     (D)iagnostics Monitor
     (F)ile Restore
     (O)ff-line Monitor
     (U)tilities Monitor
     (W)armstart

     Enter Option(s) or ? for help:
```

2. Select the (D)iagnostics Monitor option. The system displays the welcome message for the Diagnostics Monitor.

```
This is the Ultimate Diagnostic Monitor
                    (date)

    (D)isc diagnostic
    (E)CP diagnostic


    option:
```

3. To select the ECP diagnostic tests, enter E. The disk label is displayed. The test patterns to check the board are loaded and verified. Each pattern is displayed on the terminal as it is executed, along with the current status. The display is similar to the following:

```
L2000# Time Date ABS ULTIMATE RELEASE nn*xxx COLD LOAD
FRAMES

                    ECP Diagnostic Monitor
                          Release xx

Testing ECP WCS RAM

Pattern:123456    load  ...verify  ...Pass
Pattern:125252    load  ...verify  ...Pass
Pattern:052525    load  ...verify  ...Pass
Pattern:177777    load  ...verify  ...Pass
Pattern:000000    load  ...verify  ...Pass
```

If a failure is detected, the expected pattern and the actual pattern are displayed on the user's terminal along with the failing location in WCS Ram; the loading sequence is then terminated.

4. If no errors are detected by the test patterns, the Diagnostic Functional Code is loaded into WCS RAM and verified. A test is then conducted to establish that the coprocessor can acknowledge data transmitted by the host. If the tests are successful, messages similar to the following are displayed:

```
Download ECP RAM Diagnostic  ...verify  ...Pass

Testing ECP interface  ...Pass
```

If a failure is detected, the expected test pattern and the actual test pattern read are displayed on the user's terminal along with the failing location in WCS Ram; the testing is then terminated.

If no errors are detected, the following prompt is displayed:

```
Press <RETURN> to continue
```

5. Press RETURN. The ECP Diagnostic Menu is displayed. The menu contains all the diagnostic tests that can be executed; it also displays the diagnostic firmware revision being executed. The menu is similar to the following:

```
                         ECP DIAGNOSTIC MENU
                          Firmware rev: nnn

 0 Execute All Diag Tests (1 thru 19)   1 Test Temporary Flags
 2 Test RAD Counting                    3 Test Branch on Temp Flags
 4 Test RALU Shifting                   5 Test D-BUS Literal Const. Gen.
 6 Test Z-BUS and ARAM                  7 Test Hex Decoder
 8 Test RALU Scratch Pad Reg's          9 Test Shift Register
10 Test Page Overflow                  11 Test Stop Code RAM & Ind. Bit
12 Test ADRA/ADRB/ADRP/PCTR Regs       13 Test WCS Memory Access
14 Test System Memory                  15 Test 256-way Branch
16 Test BR-Arith Branch                17 Test Procedure Buffer
18 Test Proc. Buff. PTAKE Operation    19 Test Mem. & Proc. Buff. Oper...
20 Test ECP Interrupt Vectors          21 Run PROM QLT Test
22 Set 50/60 Hz line frequency *060*   23 Test WCS RAM & Reload Firm...
24 Dump & Display ECP Registers        25 Exit ECP Diags (Reboot Syste...

                          Enter Selection(s) :
```

For an explanation of each of the options, see the next section, Menu Selections.

6. Selections 1 through 19 can be executed in combination with each other; if more than one test is desired, enter the test numbers separated by commas. Up to twenty tests can be specified.

Selections 20 through 25 can only be executed one at a time.

After the selections are entered, the number of times the tests are to be performed is requested:

```
Enter number of Passes (1 to 9999 or 0 for continuous) -
```

The following message is displayed as the tests begin:

```
(Press <ESC> to terminate test at any time)
Running ECP Test(s)
```

While the tests are executing, the system displays the number of loops and the time in seconds; the numbers are updated at the end of every loop or, if the loop takes more than five seconds to run, every five seconds:

```
Loop No.  nnnnn       Run Time xxxxx
```

To interrupt a test, press <ESC>.

Note: *It may take the system several seconds to react to the <ESC> key; it checks for an <ESC> only after completing the tests in the loop.*

7. When the tests finish or are interrupted because <ESC> was pressed, the total number of loops and number of seconds run time are displayed, then the following prompt is displayed:

   ```
   Press <RETURN> to continue
   ```

   To return to the Enter Selections prompt of the Diagnostics menu, press RETURN.

8. If an error is detected by one of the tests, testing halts and a message that shows the menu selection, status word, and error message is displayed, similar to the following:

```
**ECP error on Menu selection # 006**
   Status = x'0018' (ARAM Test error)

(L)oop on error, (C)ontinue or e(X)it?
```

In the example, the # 006 is the menu selection in which the test failed; the 0018 is the error message number; and ARAM Test error is the error message. For a complete list of all error messages, see the section, Diagnostic Error Messages.

9. To repeat just the test that failed, enter L. To continue with all the selected tests, enter C. To return to the Enter Selections prompt of the Diagnostics menu, enter X.

10. If the L option is selected, the following prompt is displayed:

    Inhibit error message display <Y/N>?

    If Y is entered, no more error or run time messages are displayed.

Menu Selections

A summary description of the menu selections follows below.

0 - Execute All Diag Tests (1 thru 19)

The diagnostics tests 1 through 19 as displayed in the Diagnostic Menu are executed in succession. These options test the entire firmware instruction set.

1 - Test Temporary Flags

Each Temporary Flag firmware instruction is tested in the set and reset condition. The flags are left in the reset condition when the test is finished.

2 - Test RAD Counting

The RAD register is tested using the INC-RAD and IF-RAD:F firmware instructions. After each increment, the RAD register is verified.

3 - Test Branch on Temp Flags

The Branch on Temporary Flags firmware instruction is tested by performing a 16-way branch on all 16 possible combinations of branches.

4 - Test RALU Shifting

The RALU shifting is tested by floating a test pattern consisting of 1's and 0's through the Q and ALU Reg0 in the right and left direction.

5 - Test D-BUS Literal Const. Gen.

The D-BUS Literal Constant Generator is tested by generating and verifying literal constant patterns.

6 - Test Z-BUS and ARAM

The Z-BUS and ARAM locations are tested by floating a test pattern of 010101 through the 24-bit bus for each ARAM location. Once written, the ARAM locations are read back and verified.

7 - Test Hex Decoder

The Hex decoder is tested by using the RAD register to address it. Patterns 8000, 4000, 2000,... ,0001 are generated and verified.

8 - Test RALU Scratch Pad Reg's

The RALU scratch pad registers are tested by floating and verifying test patterns of 'F', 'A', and '5' through each of the ALU registers.

9 - Test Shift Register

The Shifter and the Shift Register are tested by shifting a basic test pattern of '0FA578' left and right, then verifying the shifter value after each shift operation.

10 - Test Page Overflow

The Page Overflow condition is tested by comparing patterns for each stage of the comparator. The bit in the indicator register is verified after each test pattern.

11 - Test Stop Code RAM & Ind. Bit

The Stop Code RAM and Indicator Bit is tested by filling the even addresses of RAM with value 0 and odd addresses with value 1. Each address is read and the stop code bit verified. Next, the even addresses of RAM are filled with 1 and the odd addresses are filled with 0 and each address is read and stop code bit verified.

12 - Test ADRA/ADRB/ADRP/PCTR Regs

The address registers ADRA, ADRB, ADRP, and the PCTR are tested by floating a 1 through each register and verifying the 24 bits of ADRA, ADRB and ADRP and the 9 bits of the PCTR.

13 - Test WCS Memory Access

WCS memory access is tested by writing the WCS memory address in each 16 bits of the firmware word, reading it back, and verifying it.

14 - Test System Memory

The System Memory is tested by executing the write-read firmware instructions available to the coprocessor. The 2/4 Mbyte memory is first tested by the word write-read mode. Each memory location is written to, read back, and the test pattern verified before the next memory location is tested. Next, the memory locations are read back in the read-double-word mode and verified. Last, the memory is tested in the byte write-read mode.

The CIB memory space is tested in the word write-read mode. The memory location is again written to, read back, and verified before the next memory location is tested. Then, the CIB memory space is tested in the byte write-read mode.

15 - Test 256-way Branch

The BR 256-way firmware instruction is tested by verifying the branching to all 256 possible locations. An incrementing byte pattern starting at value 0 is loaded into the memory test buffer. The values are pulled into the Procedure buffer and the 256-way branch action verified.

16 - Test BR-Arith Branch

The Branch Arithmetic firmware instruction is tested by setting arithmetic conditions and verifying the proper branching action.

17 - Test Procedure Buffer

The Procedure buffer test verifies hardware functions associated with the operation of the procedure buffer: the procedure buffer condition status, the RAB, and diagnostics firmware instruction branches. First, an incrementing byte test pattern is loaded into the memory test buffer, then the test patterns are pulled into the procedure buffer and verified via the procedure buffer operations.

18 - Test Proc. Buff. PTAKE Operation

The procedure buffer PTAKE operation test emulates a functional firmware series of firmware steps by exercising read and write memory operations along with procedure buffer operations. A memory buffer is loaded with an incrementing byte data pattern. Incrementing buffer locations are read and then the data pattern written back to the same location. PTAKE operations are executed in the same loop.

19 - Test Mem. & Proc. Buff. Operation

This test emulates a series of functional firmware steps by exercising the execution of a set of virtual instructions. The hexadecimal codes for the virtual instruction are used to fill a memory buffer. The instruction is then executed out of the buffer repeatedly.

20 - Test ECP Interrupt Vectors

This test sends the command to change the interrupt vector address at which the coprocessor interrupts the host. The command and associated index value, indicating which one of the eight interrupt vector addresses to select, is sent to the coprocessor. The coprocessor uses the index value to interrupt the host at the indicated address.

21 - Run PROM QLT Test

The PROM QLT test initiates execution of the PROM resident diagnostics each time it is selected.

22 - Set 50/60 Hz line frequency *060*

The set 50/60 Hz line frequency option can be used to set the watch-dog timer internal to the host-based software to the particular AC line frequency that the host system is connected to. The value enclosed in asterisks is the current setting. When the option is selected, the following prompt is displayed:

```
Select (50) or (60) Hz:
```

After the setting is entered, press RETURN at the prompt to redisplay the menu. The new line frequency takes effect as soon as the menu option is displayed.

23 - Test WCS RAM & Reload Firmware

This test writes a test pattern to all of WCS memory, reads it back and verifies it. Altogether, 5 different test patterns are used. Upon a detected failure, the expected and actual data patterns are displayed along with the failing WCS memory location.

24 - Dump & Display ECP Registers

This menu selection displays the coprocessor's current register contents to the user's screen. This selection can be useful, for example, in determining the status of the hardware at the time of a failure.

25 - Exit ECP Diags (Reboot System)

This selection exits the Diagnostic Menu and gets the user back to the system level menu to reboot the system.

Diagnostic Error Messages

The following is a complete listing of the possible error messages by the ECP Diagnostics.

Error Number	Message
1	Permanent Flag Test error
2	Permanent Flag Test error
3	RALU Test error
4	No Real Time Clock detected
5	Return Address Test error
6	Return Address Test error
7	Temporary Flag Test error
8	Temporary Flags Branch Test error
9	Literal Generator Test error
10	RAD Test error
11	RAD Test error
12	RALU Shift Left Test error
13	RALU Shift Right Test error
14	Shifter error
15	Shift Register Test error
16	RALU Register Test error
18	ARAM Test error
19	Address Page Comparator Test error
20	Stop Code RAM Test error
24	BR-256 Way Branch Test error
25	Address Register A Test error
26	Address Register B Test error
27	Address Register P Test error
28	PCTR Test error
29	Word Write/Read Memory Test error
30	Double Word Read Memory Test error
31	Procedure Buffer - Branch Test error
32	Procedure Buffer - PTAKE Test error
33	Procedure Buffer - INT Test error
34	Hex Decoder Test error
35	WCS Write/Read Memory Test error
36	BR-ARITH Branch Test error
38	Procedure Buffer Operation Test error
39	Procedure Buffer Status Test error
41	Instruction Decode Test error
42	CIB Memory Test error
48	System Memory Test error

EECP Diagnostics Description

The Extended Enhanced Co-Processor (EECP) Diagnostics test the EECP board itself, the cable connecting the board to the LSI memory, the EECP instruction set, and for 3050 systems, the 4-Mb memory.

The diagnostics perform the following functions:

- test data patterns are loaded into EECP Writable Control Storage (WCS) RAM, read back, and verified
- the diagnostic functional code is loaded in WCS RAM and the loaded code is verified
- commands are sent between the LSI host and the EECP to test the hardware interface
- a menu is loaded that allows the user to specify the tests and other functions to be performed on the EECP

The EECP Diagnostics are available from disk or from a standalone bootable tape.

Using the Diagnostics

The following procedure outlines the steps to loading and using the EECP diagnostics system.

1. Boot the system from either disk or the standalone diagnostics tape.

   If you booted the system from disk, the following System Startup Options menu is displayed:

```
This is the Ultimate Operating System

     Bootloader revision #nn

     System Startup Options:
     (B)oot
     (D)iagnostics Monitor
     (O)ff-line Monitor
     (U)tilities Monitor
     (W)armstart

     Enter Option(s) or ? for help:
```

   If you booted the system from a standalone tape, the following System Startup Options menu menu is displayed:

```
This is the Ultimate Operating System

     Bootloader revision #nn

     System Startup Options:
     (B)oot
     (C)oldstart
     (D)iagnostics Monitor
     (F)ile Restore
     (O)ff-line Monitor
     (U)tilities Monitor
     (W)armstart

     Enter Option(s) or ? for help:
```

2. Select the (D)iagnostics Monitor option. The system displays the welcome message for the Diagnostics Monitor.

```
This is the Ultimate Diagnostic Monitor
                    (date)

    (D)isc diagnostic
    (E)ECP diagnostic


    option:
```

3. To select the EECP diagnostic tests, enter E. The disk label is displayed. The test patterns to check the board are loaded and verified. Each pattern is displayed on the terminal as it is executed, along with the current status. The display is similar to the following:

```
L2000# Time Date ABS ULTIMATE RELEASE nn*xxx COLD LOAD
FRAMES

                    EECP Diagnostic Monitor
                         Release xx

Testing EECP WCS RAM

Pattern:123456   load ...verify ...Pass
Pattern:125252   load ...verify ...Pass
Pattern:052525   load ...verify ...Pass
Pattern:177777   load ...verify ...Pass
Pattern:000000   load ...verify ...Pass
```

If a failure is detected, the expected test pattern and the actual test pattern read are displayed on the user's terminal along with the failing location in WCS RAM; the loading sequence is then terminated.

4. If no errors are detected by the test patterns, the Diagnostic Functional Code is loaded into WCS RAM and verified. A test is then conducted to establish that the coprocessor can acknowledge data transmitted by the host. If the tests are successful, messages similar to the following are displayed:

```
Download EECP RAM Diagnostic ...verify ...Pass

Testing EECP interface ...Pass
```

If a failure is detected, the expected pattern and the actual pattern are displayed on the user's terminal along with the failing location in WCS RAM; the testing is then terminated.

If no errors are detected, the following prompt is displayed:

```
Press <RETURN> to continue
```

5. Press RETURN. The EECP Diagnostics Menus is displayed. The menu contains all the diagnostic tests that can be executed; it also displays the diagnostic firmware revision being executed. The menu is similar to the following:

```
                        EECP DIAGNOSTIC MENU
                          Firmware rev: nnn

 0 Execute All Diag Tests (1 thru 24)    1 Test Temporary Flags
 2 Test RAD Counting                     3 Test Branch on Temp Flags
 4 Test RALU Shifting                    5 Test D-BUS Literal Const. Gen.
 6 Test Z-BUS and ARAM                   7 Test Hex Decoder
 8 Test RALU Scratch Pad Reg's           9 Test Shift Register
10 Test Page Overflow                   11 Test Stop Code RAM & Ind. Bit
12 Test ADRA/ADRB/ADRP/PCTR Regs        13 Test WCS Memory Access
14 Test System Memory                   15 Test 256-way Branch
16 Test BR-Arith Branch                 17 Test Procedure Buffer
18 Test Proc. Buff. PTAKE Operation     19 Test Mem. & Proc. Buff. Oper...
20 Test EECP Interrupt Vectors          21 Test PTAKE and LD-RAB'C
22 Test Read Byte and PTAKE             23 Test Memory and ADRA Opera...
24 Test Increment ADRA                  25 Run PROM QLT Test
26 Set 50/60 Hz line frequency *060*    27 Test WCS RAM & Reload Firm...
28 Dump & Display EECP Registers        29 Exit EECP Diags (Reboot Syst...

                        Enter Selection(s) :
```

For an explanation of each of the options, see the next section, Menu Selections.

6. Selections 1 through 24 can be executed in combination with each other; if more than one test is desired, enter the test

numbers separated by commas. Up to twenty-four tests can be specified.

Selections 25 through 29 can only be executed one at a time.

After the selections are entered, the number of times the tests are to be performed is requested:

```
Enter number of Passes (1 to 9999 or 0 for continuous) -
```

The following message is displayed as the tests begin:

```
(Press <ESC> to terminate test at any time)
Running EECP Test(s)
```

While the tests are executing, the system displays the number of loops and the time in seconds; the numbers are updated at the end of every loop or, if the loop takes more than five seconds to run, every five seconds:

```
Loop No.  nnnnn      Run Time xxxxx
```

To interrupt a test, press <ESC>.

Note: *It may take the system several seconds to react to the <ESC> key; it checks for an <ESC> only after completing the tests in the loop.*

7. When the tests finish or are terminated because <ESC> was pressed, the total number of loops and number of seconds run time are displayed, then the following prompt is displayed:

   ```
   Press <RETURN> to continue
   ```

   To return to the Enter Selections prompt of the Diagnostic menu, press RETURN.

8. If an error is detected by one of the tests, testing halts and a message that shows the menu selection, status word, and error message is displayed, similar to the following:

```
**EECP error on Menu selection # 006**
   Status = x'0018' (ARAM Test error)

(L)oop on error, (C)ontinue or e(X)it?
```

In the example, the # 006 is the menu selection in which the test failed; the 0018 is the error message number; and ARAM Test error is the error message. For a complete list of all error messages, see the section, Diagnostic Error Messages.

9. To repeat just the test that failed, enter L. To continue with all the selected tests, enter C. To return to the Enter Selections prompt of the Diagnostics menu, enter X.

10. If the L option is selected, the following prompt is displayed:

    Inhibit error message display <Y/N>?

    If Y is entered, no more error or run time messages are displayed.

Menu Selections

A summary description of the menu selections follows below.

0 - Execute All Diag Tests (1 thru 19)

The diagnostics tests 1 through 19 as displayed in the Diagnostic Menu are executed in succession. These options test the entire firmware instruction set.

1 - Test Temporary Flags

Each Temporary Flag firmware instruction is tested in the set and reset condition. The flags are left in the reset condition when the test is finished.

2 - Test RAD Counting

The RAD register is tested using the INC-RAD and IF-RAD:F firmware instructions. After each increment, the RAD register is verified.

3 - Test Branch on Temp Flags

The Branch on Temporary Flags firmware instruction is tested by performing a 16-way branch on all 16 possible combinations of branches.

4 - Test RALU Shifting

The RALU shifting is tested by floating a test pattern consisting of 1's and 0's through the Q and ALU Reg0 in the right and left direction.

5 - Test D-BUS Literal Const. Gen.

The D-BUS Literal Constant Generator is tested by generating and verifying literal constant patterns.

6 - Test Z-BUS and ARAM

The Z-BUS and ARAM locations are tested by floating a test pattern of 010101 through the 24-bit bus for each ARAM location. Once written, the ARAM locations are read back and verified.

7 - Test Hex Decoder

The Hex decoder is tested by using the RAD register to address it. Patterns 8000, 4000, 2000,... ,0001 are generated and verified.

8 - Test RALU Scratch Pad Reg's

The RALU scratch pad registers are tested by floating and verifying test patterns of 'F', 'A', and '5' through each of the ALU registers.

9 - Test Shift Register

The Shifter and the Shift Register are tested by shifting a basic test pattern of '0FA578' left and right, then verifying the shifter value after each shift operation.

10 - Test Page Overflow

The Page Overflow condition is tested by comparing patterns for each stage of the comparator. The bit in the indicator register is verified after each test pattern.

11 - Test Stop Code RAM & Ind. Bit

The Stop Code Ram and Indicator Bit is tested by filling the even addresses of RAM with value 0 and odd addresses with value 1. Each address is read and stop code bit verified. Next, the even addresses of RAM are filled with 1 and the odd addresses are filled with 0 and each address is read and stop code bit verified.

12 - Test ADRA/ADRB/ADRP/PCTR Regs

The address registers ADRA, ADRB, ADRP, and the PCTR are tested by floating a 1 through each register and verifying the 24 bits of ADRA, ADRB and ADRP and the 9 bits of the PCTR.

13 - Test WCS Memory Access

WCS memory access is tested by writing the WCS memory address in each 16 bits of the firmware word, reading it back, and verifying it.

14 - Test System Memory

The System Memory is tested by executing the write-read firmware instructions available to the coprocessor. The 2/4 Mbyte memory is first tested by the word write-read mode. Each memory location is written to, read back, and the test pattern verified before the next memory location is tested. Next, the memory locations are read back in the read-double-word mode and verified. Last, the memory is tested in the byte write-read mode.

The CIB memory space is tested in the word write-read mode. The memory location is again written to, read back, and verified before the next memory location is tested. Then, the CIB memory space is tested in the byte write-read mode.

15 - Test 256-way Branch

The BR 256-way firmware instruction is tested by verifying the branching to all 256 possible locations. An incrementing byte pattern starting at value 0 is loaded into the memory test buffer. The values are pulled into the Procedure buffer and the 256-way branch action verified.

16 - Test BR-Arith Branch

The Branch Arithmetic firmware instruction is tested by setting arithmetic conditions and verifying the proper branching action.

17 - Test Procedure Buffer

The Procedure buffer test verifies hardware functions associated with the operation of the procedure buffer: the procedure buffer condition status, the RAB, and diagnostics firmware instruction splatter branches. First, an incrementing byte test pattern is loaded into the memory test buffer, then the test patterns are pulled into the procedure buffer and verified via the procedure buffer operations.

18 - Test Proc. Buff. PTAKE Operation

The procedure buffer PTAKE operation test emulates a functional firmware series of firmware steps by exercising read and write memory operations along with procedure buffer operations. A memory buffer is loaded with an incrementing byte data pattern. Incrementing buffer locations are read and then the data pattern written back to the same location. PTAKE operations are executed in the same loop.

19 - Test Mem. & Proc. Buff. Operation

This test emulates a series of functional firmware steps by exercising the execution of a set of virtual instructions. The hexadecimal codes for the virtual instruction are used to fill a memory buffer. The instruction is then executed out of the buffer repeatedly.

20 - Test EECP Interrupt Vectors

This test sends the command to change the interrupt vector address at which the coprocessor interrupts the host. The command and associated index value, indicating which one of the eight interrupt vector addresses to select, is sent to the coprocessor. The coprocessor uses the index value to interrupt the host at the indicated address.

21 - Test PTAKE and LD-RAB'C

This test verifies the execution of the firmware instruction sequence of back-to-back PTAKE and LD-RAB'C operation.

22 - Test Read Byte and PTAKE

This test verifies the execution of the firmware instruction sequence of back-to-back byte memory read and PTAKE operations.

23 - Test Memory and ADRA Operation

This test verifies the execution of the firmware instruction sequence of memory operations and address register operations.

24 - Test Increment ADRA

This test verifies the hardware incrementing operation of the ADRA address register. The firmware instruction is tested with all possible memory access operations.

25 - Run PROM QLT Test

The PROM QLT test initiates execution of the PROM resident diagnostics each time it is selected.

26 - Set 50/60 Hz line frequency *060*

The set 50/60 Hz line frequency option can be used to set the watch-dog timer internal to the host-based software to the particular AC line frequency that the host system is connected to. The value enclosed in asterisks is the current setting. When the option is selected, the following prompt is displayed:

    Select (50) or (60) Hz:

After the setting is entered, press RETURN at the prompt to redisplay the menu. The new line frequency takes effect as soon as the menu option is displayed.

27 - Test WCS RAM & Reload Firmware

This test writes a test pattern to all of WCS memory, reads it back and verifies it. Altogether, 5 different test patterns are used. Upon a detected failure, the expected and actual data patterns are displayed along with the failing WCS memory location.

28 - Dump & Display EECP Registers

This menu selection displays the coprocessor's current register contents to the user's screen. This selection can be useful, for example, in determining the status of the hardware at the time of a failure.

29 - Exit EECP Diags (Reboot System)

This selection exits the Diagnostic Menu and gets the user back to the system level menu to reboot the system.

Diagnostic Error Messages

The following is a complete listing of the possible error messages by the EECP Diagnostics.

Error Number	Message
1	Permanent Flag Test error
2	Permanent Flag Test error
3	RALU Test error
4	No Real Time Clock detected
5	Return Address Test error
6	Return Address Test error
7	Temporary Flag Test error
8	Temporary Flags Branch Test error
9	Literal Generator Test error
10	RAD Test error
11	RAD Test error
12	RALU Shift Left Test error
13	RALU Shift Right Test error
14	Shifter error
15	Shift Register Test error
16	RALU Register Test error
18	ARAM Test error
19	Address Page Comparator Test error
20	Stop Code RAM Test error
24	BR-256 Way Branch Test error
25	Address Register A Test error
26	Address Register B Test error
27	Address Register P Test error
28	PCTR Test error
29	Word Write/Read Memory Test error
30	Double Word Read Memory Test error
31	Procedure Buffer - Branch Test error
32	Procedure Buffer - PTAKE Test error
33	Procedure Buffer - INT Test error
34	Hex Decoder Test error
35	WCS Write/Read Memory Test error
36	BR-ARITH Branch Test error
38	Procedure Buffer Operation Test error
39	Procedure Buffer Status Test error
41	Instruction Decode Test error
42	CIB Memory Test error
43	Op-Code Test error

44	Op-Code Test error
45	Memory Read - ADRA Test error
46	Increment ADRA Test error
48	System Memory Test error

Notes

8 Ultimate 6000/7000 Enhancements

Enhancements for the Ultimate 6000 and 7000 series of computers include the following:

- disk shadowing
- faster file save capability
- GCR and Pertec tape drive enhancements

Disk Shadowing

Disk shadowing is the concept of maintaining duplicate copies of the Ultimate database on two sets of disks. With disk shadowing, you have two complete, parallel disk subsystems; whenever your database is changed, both subsystems are updated. Even if a disk or an entire controller fails, your data is safe and the Ultimate system continues to run.

In Ultimate's disk shadowing facility, the two sets of disks consist of shadowed pairs of disks that have the same properties and contain identical data. The two disks in a shadowed pair are called sibs.

If an unrecoverable error occurs in one sib, the system automatically decouples the pair and stops the shadowing for that pair. System operations are continued as unshadowed on the undamaged disk in the pair and as shadowed on all other pairs. Decoupled disks can resume shadowing via a subsequent boot after the damaged disk has been repaired and the unimpaired disk has been copied to it.

Before you can use disk shadowing, you must set up the following:

- install two identical disk subsystems
- label the disks
- put the same data on both sets of disks.

Disk shadowing is handled automatically by the system when it senses the duplicate configuration.

Equipment Needed for Disk Shadowing

For Release 200, disk shadowing requires the following equipment:

- Ultimate 6000 or 7000 series systems
- FSD or EMD disk drives
- at least two controllers
- enough disks to configure disk shadowing; that is, at least twice the number of disks needed for the Ultimate database on the system

Configuring the Hardware

Note: *This section is intended for the field engineer responsible for setting up the controllers and disks.*

To configure the hardware for a disk shadowing system, you should set up two separate and identical disk subsystems. Each subsystem requires its own controllers and disks. If subsystem 1 has two controllers with the disks set up as 0 and 1 on each controller, then subsystem 2 needs to have two controllers with disks set up as 0 and 1 on each controller. The cabling for the subsystems should be identical.

The channel addresses are the only items that are unique. Set up the channel addresses for one disk subsystem. Then set up the channel addresses for the second subsystem in a similar manner, using different channel addresses. For example, if disk subsystem 1 uses channels F800, F880, F900, and F980, then disk subsystem 2 could be set up on channels F000, F080, F100, and F180.

Setting up the disk subsystems in this way has the following advantages:

- provides optimum resiliency of the shadowed system
- provides another level of redundancy; an entire controller can go down and the system will continue to operate
- allows the drive numbering to be simple and consistent
- on fixed surface drives (FSD), if the configuration is small enough, the numbered address plugs correspond to the logical volume sequence number

Setting Up Disk Shadowing

Disk shadowing can be set up on new and existing Ultimate systems using Release 200 or later versions of the operating system. In both new and existing systems, you need to have two identically configured disk subsystems. Each disk in a disk pair must have the same label as its sib and it must have the same data on it. If your system is set up this way, the system software recognizes the configuration as a disk shadow setup and automatically provides disk shadowing.

After the hardware is installed, identical labels must be placed on both sets of disks. The labels may be written either by the format process or by the D-WTLB command. The following procedure describes the writing of the labels.

Labeling Disks

1. If necessary, install the disks and associated hardware that you need to create a shadowed disk set made up of shadowed pairs.

2. Mount the SYSGEN tape and boot the system. The Startup Options Menu is displayed; see Figure 8-1.

```
This is the Ultimate Operating System

        Bootloader revision #18

        System Startup Options:
        (B)oot
        (C)oldstart
        (D)iagnostics Monitor
        (F)ile Restore
        (O)ff-line Monitor
        (U)tilities Monitor
        (W)armstart

        Enter Option(s) or ? for help:  D
```

Figure 8-1. Start-Up Menu, Tape Boot

3. Select the (D)iagnostics Monitor option. The system automatically displays the tape label, loads the boot and diagnostics sections, then displays the welcome message for the Diagnostics Monitor. The :> is the Diagnostic Monitor prompt.

```
L2000# Time Date ABS ULTIMATE RELEASE 10*200 COLD LOAD
FRAMES

This is the Ultimate Diagnostics Monitor
Revision ###          (date)

 :>
```

4. The format and labeling processes require the channel addresses of all your disks. These addresses are set up at system installation. To determine the addresses, you can use the CONFIG command:

```
CONFIG <CR>
```

The system configuration is displayed in the following format.

```
Chan      Dev-id   FW-REV   .....Device  Description.....
 .
F000      2305     12          413mb  FSD disc
F080      2305                 413mb  FSD disc
F800      2305                 413mb  FSD disc
F880      2305                 413mb  FSD disc
 .
 .
```

The report shows all hardware configured on the system: CPU, terminals, printers, as well as the disks (FSD disks or other disk description). For details about the CONFIG command, please refer to the *Ultimate System Management and Support Guide.*

Keep the list of channel addresses handy.

5. You also need to know the disk set name used for the existing set (if any). To check the set name, you can use the D-RDLB command. Type:

```
D-RDLB channel# <CR>
```

where channel# is the channel address of any disk in the existing disk set. The second line of the displayed report shows the set name. For example:

```
D-RDLB F800
```

```
Device type is x'2305'  413mb FSD disc
Name of set: TERRY
.
.
```

6. Once you have the channel addresses of the disks in both sets and the set name of an existing disk set (if any), you can start labeling the new disks.

   If the disks have already been formatted, you only need to write the disk label. Enter

   D-WTLB channel# <CR>

   Skip to step 8.

7. To format the disk, as well as write the labels, enter

   FORMAT channel# <CR>

   where channel# is the channel of the disk you wish to format and label.

   The device type is displayed for the specified channel and the system type is requested, similar to the following:

   ```
   Device type is x'2305' 413mb FSD disc

   Format for (U)ltimate or (T)VOS/GCOS?
   ```

   Enter U.

8. The following prompts request information that is used to build the disk label and are the same for D-WTLB and FORMAT:

   ```
   Enter set name (12 char max)
   ```

   Type the name of the disk set (the same name as used for any existing set); press RETURN. All disks in both sets must have the same set name.

   The next prompt is:

   ```
   Enter # of discs in the set :
   ```

Enter the total number of disks in the set and press RETURN. Each set of disks in the shadowed system may have up to 16 disks.

If you enter a number other than 1, the next prompt is:

```
Sequence # of this pack :
```

Enter the sequence number of this pack within the set; press RETURN.

In any disk set, the disk packs must be labeled consecutively from '1 of n' to 'n of n'.

10. The disk label is written. After writing the label, D-WTLB returns to the Diagnostics Monitor prompt (:>); continue with step 12. FORMAT proceeds to format the disk.

11. The FORMAT command asks:

```
Save old defective space table (<CR>=Y/N):
```

If the disk has been formatted previously, enter **Y** or press just RETURN to save the table of tracks that were marked defective the last time the disk was formatted. This ensures that any bad spots in the disk are always marked defective. If the disk is new, that is, it has not been formatted previously, **N** should be entered. An **N** cancels any previous user-designated markings.

The formatter proceeds through four phases: Format, Verify, Data Pattern write, and Data Pattern verify. As each phase executes, an '*' is displayed when each set of 16 cylinders is completed. If any tracks are found to contain errors, these are reported on the terminal and marked as defective.

Note: *For more information on D-WTLB and FORMAT, see the Ultimate System Management and Support Guide.*

12. Repeat the formatting and/or labeling for each disk in the set, so that each data disk in one disk set has a label identical to its sib in the other set.

13. After all the disks have been labeled, check each label. Enter

```
D-RDLB channel# <CR>
```

This verb displays the label information for the disk at the specified channel address; see Figure 8-2.

```
Device type is x'2305' 413mb FSD disc
Name of set:     TERRY
Drives/ set:     0002
Sequence #:      0001
Revision #:      0029
Option Wd 1:     C000
Option Wd 2:     0000
Option Wd 3:     0000
Interlace #:     0001
Frm Layout:      0000
Cyl/Volume:      02C5
Trk/Cylindr:     0018
Sectors/Trk:     005F
Sector Size:     0100
Max-Def Trk:     0000
Kernel Bias:     02F8
Failure ctr:     0000
Consistent:      0000
Shadow flag:     0000
```

Figure 8-2. Disk Label

The first four and last three lines relate to disk shadowing.

In the example, channel F000 contains a 413 FSD disk in a disk set called TERRY; F000 is Sequence # 1 of a Set of 2.

The Failure Counter is 0000; this means that the system has not been rebooted (with disk shadowing in effect) since the label was written. This counter is initially cleared to zero (0) when the disk label is first written. When disk shadowing is in effect, the Failure Counter is incremented periodically by the system; for example, it is incremented when the system is rebooted after a system problem or failure.

The Consistent and Shadow flags are cleared; this means that the disk has not yet been used as part of a shadowed pair.

14. After the report is displayed, the system returns to the Diagnostics Monitor prompt.

15. Check each label and make sure that the D-RDLB report for each shadowed pair shows the same definition. The Consistent and Shadow flags are the only items that may not be identical on both sibs.

16. To exit the Diagnostics Monitor after the disks have been formatted, labeled, and checked, reboot the system.

Copying the Data

You are now ready to copy the data from each disk in the primary set onto its sib in the secondary set. This creates a shadowed set ready for booting and a warmstart.

Before proceeding, make sure that:

- you have assigned each disk in the disk set the same set name, and a sequential sequence number
- the labels are all correct and that you have two complete sets of matching labels
- your existing set, if any, contains your entire Ultimate system database

New Systems

If you are starting up a new system that is configured for disk shadowing, the restore process automatically restores data to both pairs of disks at the same time. The following procedure describes the steps.

1. Ensure that both sets of disks and associated hardware are installed.
2. Format the sets of disks, if necessary. When you format the second set of disks, specify the same label information on each disk as was specified in the first set. In particular, the set name and number of disks in the set must be identical.

   For more information on labeling disks, see the section in this document entitled Labeling Disks.
3. Install the software using the procedure described in the Upgrade Procedures document for Release 200.

Existing Systems

If you are upgrading an existing system that is configured for disk shadowing and are using a full restore, the restore process automatically restores data to both pairs of disks at the same time. No special procedures are required.

If you are installing disk shadowing on an existing Release 200 system, you can use the D-COPY utility in the Utility Monitor to make an exact copy of your existing data.

Note: *After shadowing is in place and working, D-COPY should be used whenever a disk needs to be copied to its sib. This can occur, for example, if a disk has had an error and was out of service for a period of time but is now ready to resume use as a sib.*

The following procedure describes the D-COPY utility.

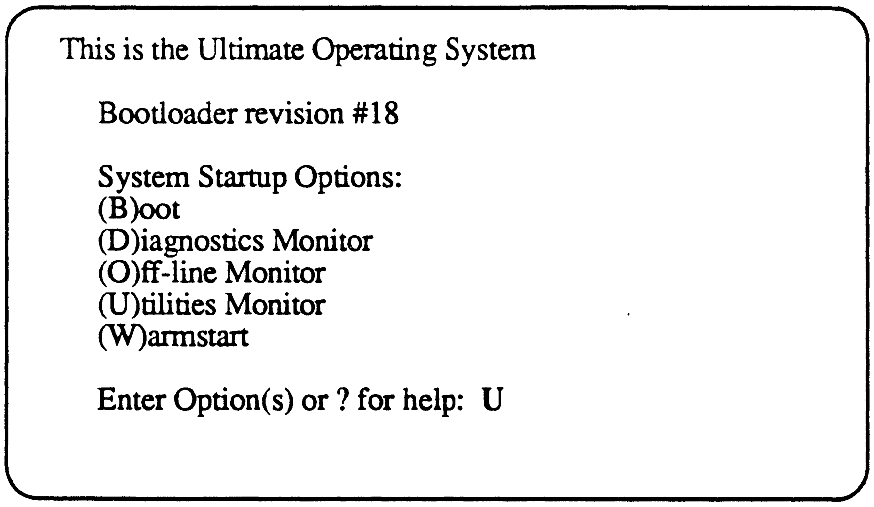
1. If necessary, upgrade your system to Release 200.

2. Backup your system.

3. Make sure that all users are logged off. Execute a :WARMSTOP from the SYSPROG account.

4. If necessary, install the hardware and write the disk labels on the new set to match the labels on the existing set.

5. Boot the system from disk, or if you prefer, mount the SYSGEN tape and boot from tape. For the Start-Up Options Menu on disk, see Figure 8-3; for the Start-Up Options Menu on tape, see Figure 8-1.



```
This is the Ultimate Operating System

   Bootloader revision #18

   System Startup Options:
   (B)oot
   (D)iagnostics Monitor
   (O)ff-line Monitor
   (U)tilities Monitor
   (W)armstart

   Enter Option(s) or ? for help:  U
```

Figure 8-3. Start-Up Menu, Disk Boot

6. To select the (U)tilities Monitor from the Startup menu, enter U. While the Utilities Monitor is running, only line 0 is active.

7. When the Utilities Monitor is loaded, the system displays the current status and the Utilities Monitor welcome message. The greater-than (>) sign is the prompt character.

```
   Disc configuration(s)
<<<< Set #1 >>>>
Chan Set Name # of .Status... Chan Set Name # of .Status...
F000 TERRY   1  2 F=0     C  F800 TERRY   1  2 F=0
F080 TERRY   2  2 F=0     C  F880 TERRY   2  2 F=0

   This is the Ultimate Utility Sub-system

   Rev #nn        Rev date (date)

>
```

8. From the Utilities Monitor, use the D-COPY verb to duplicate the first disk that needs to be copied to its sib.

   > D-COPY <CR>

9. D-COPY asks for the channel addresses of the copy-from drive:

   Enter channel address of copy-from device:

   Enter the channel address of the first disk in the existing disk set; for example, F000.

10. D-COPY then uses the labels of the second set of disks to determine the sib of the copy-from device. The disk set information on the label of each sib is displayed and the copy process starts. This is a sample of the screen display:

```
   History string: xxxx xxxx xxxx xxxx xxxx ...
 Set name : TERRY
 Sequence # : 1 of 2
 Failure Counter : 0
 Not Shadowed and Consistent
 Copy-to device: is on channel F800
    History string: xxxx xxxx xxxx xxxx xxxx ...
  Set name: TERRY
  Sequence#: 1 of 2
  Failure Counter: 0
  Not Shadowed and not Consistent
  Copy started at 00:03:15 - copying 21280 frames
       nnnnn
```

The history string contains information used by system support.

nnnnn represents the number of frames copied; the number is updated after every 1000 frames are copied.

Note: *The number displayed for the Failure Counter may or may not be 0, depending on previous shadowing.*

11. When the copy is complete, a message similar to the following is displayed. The system then returns to the Utilities Monitor.

```
Disc copy complete - 21280 frames processed
Elapse time: 00:00:31
>
```

The elapse time is the time it took to complete the copy. This is normally about 5-30 minutes on a typical fully operational system.

13. Repeat steps 3 through 12 for each disk that needs to be copied to its sib.

14. When all disks are ready to start shadowing, use the LOADER command to return to the Startup Menu in order to warmstart the system. At the Utilities monitor prompt enter

```
>LOADER
```

15. When prompted to continue, press RETURN. The Startup Options Menu is displayed. Enter W to execute the Warmstart option.

14. For information on D-COPY errors, see the next section.

D-COPY Errors

If D-COPY is not able to complete normally (for example, a power-fail occurs during the copy), the copy-to disk is marked 'incomplete' and cannot be used. Reformat the copy-to disk and retry the D-COPY.

If the disks are not configured as shadowed pairs, the D-COPY cannot proceed.

If the copy-to device has a higher Failure Counter number than the copy-from device, the copy-to is more current than the copy-from. In this case, the copy does not start. D-COPY displays:

```
Copy-to device more up-to-date
---(C)ontinue, (Q)uit or e(X)change from<-->to?
```

To exchange the specified copy-from and copy-to devices, enter X; to go ahead with the copy, enter C; to exit to the Utilities Monitor prompt, enter Q. If either X or C is entered, a disk copy is performed.

Initiating Disk Shadowing

After the disks are configured and the data copied, disk shadowing is automatically initiated by a boot and warmstart. When the system is warmstarted, the duplicate labels signal the bootloader to display the shadowed disk set.

```
4096 Kb main memory
 Disc configuration(s)
<<<< Set #1 >>>>
Chan  Set Name  # of .Status...  Chan  Set Name  # of  .Status...
F000  TERRY     1 2  F=0  S  C   F800  TERRY     1 2  F=0  S  C
F080  TERRY     2 2  F=0  S  C   F880  TERRY     2 2  F=0  S  C

<<< This is the Ultimate Operating System      >>>
```

There are three Status indicators:

F=n	Failure Counter; a zero indicates the disk has not been changed since the disk label was written
S	Shadow Flag; an S indicates the disks have been set up for shadowing. That is, the bootloader has found at least two complete and identical disk sets in the system.
C	Consistent Flag; a C indicates the same data is on both shadowed pairs of disks. The bootloader clears the C flag when it starts, and :WARMSTOP sets it, so that if the system was previously brought down by a :WARMSTOP, a C is displayed.

If the system crashed, the C is missing and the disks cannot be guaranteed to be consistent; therefore the operating system forces one of the disks to fail. In that case, you should use the DISK-STATUS command to determine the disk that failed and use D-COPY to restore the information to the failed disk.

Warmstart Errors

If any pair of disks are not set up as shadowed pairs, the non-shadowed pair is flagged and a message is displayed. The system displays the disk status and attempts to set up disk shadowing.

If at least one complete disk set is available, the system selects it and displays it as the working disk configuration. After the working configuration is displayed and if the system was booted from the front panel, a message similar to the following may be displayed:

```
Do you wish to continue the bootload?
```

To continue with the displayed working configuration, press Y followed by RETURN. To stop the load, enter N. The load is stopped and the system returns to the Startup Menu.

If there are no complete disk sets available, the system cannot be loaded and the following message is displayed:

```
No usable disk set found

Because of the above error(s), Ultimate cannot be loaded.
     Press <CR> to continue.
```

The system returns to the Startup Menu.

The following examples list some of the possible errors at WARMSTART time. Other errors may be detected by the boot process; handle them as shown in the examples.

- If the failure counter of the second disk is greater than the failure counter of the first disk, the system notes that the two disks are out of order; the system then makes the more up-to-date disk the first disk. Disk shadowing is still in effect and the booting of the system continues. An F-restore of the system will reset the failure counters so that the two disk sets are the same.

```
4096 Kb main memory
 Disc configuration(s)
<<<< Set #1 >>>>
Chan  Set Name  # of .Status... Chan  Set Name  # of  .Status...
F000  TERRY     1 2  F=0  S  C  F800  TERRY     1  2  F=1  S  C
F080  TERRY     2 2  F=1  S  C  F880  TERRY     2  2  F=1  S  C

*** WARNING *** Shadow device(s) out of date!

 Working disc configuration(s)
<<<< Set #1 >>>>
Chan  Set Name  # of .Status... Chan  Set Name  # of  .Status...
F800  TERRY     1 2  F=2  S     F000  TERRY     1  2 Out of date
F880  TERRY     2 2  F=2  S     F080  TERRY     2  2  F=2  S
```

- If Status indicators S and C are not set for a disk, that disk is decoupled and the boot continues. This is the normal situation whenever you are changing disk packs or reformatting. An F-restore of the system will reset the status indicators so that the two disk sets are the same.

```
4096 Kb main memory
 Disc configuration(s)
<<<< Set #1 >>>>
Chan  Set Name  # of .Status... Chan  Set Name  # of  .Status...
F000  TERRY     1 2  F=1  S  C  F800  TERRY     1  2  F=1
F080  TERRY     2 2  F=1  S  C  F880  TERRY     2  2  F=1  S  C

*** WARNING *** Shadow device(s) missing/decoupled

 Working disc configuration(s)
<<<< Set #1 >>>>
Chan  Set Name  # of .Status... Chan  Set Name  # of  .Status...
F000  TERRY     1 2  F=2  S     F800  TERRY     1  2 Decoupled
F880  TERRY     2 2  F=2  S     F080  TERRY     2  2  F=2  S
```

- If one of the disks in the set has been damaged and is no longer available, the system notes that it is missing. If the other disk set is complete, the boot continues. Only those paris that are still valid are shadowed.

  When a disk is missing or damaged, an F-restore of the system will not take care of the problem.

```
4096 Kb main memory
 Disc configuration(s)
<<<< Set #1 >>>>
Chan  Set Name  # of .Status...  Chan  Set Name  # of  .Status...
F000  TERRY     1 2  F=1  S  C   F800  TERRY     1 2  F=1  S  C
F080  TERRY     2 2  F=1  S  C      ** Missing **

*** WARNING ***  Shadow device(s) missing/decoupled

 Working disc configuration(s)
<<<< Set #1 >>>>
Chan  Set Name  # of .Status...  Chan  Set Name  # of  .Status...
F000  TERRY     1 2  F=2  S      F800  TERRY     1 2  F=2  S
F080  TERRY     2 2  F=2  S         ** Missing **
```

Saving the Initial Disk Status

After the operating system is started, disk shadowing of the system is in effect and continues as long as no unrecoverable disk errors are detected.

As soon as possible, make a hard copy of the disk shadowing system configuration. Log onto the SYSPROG account. From SYSPROG, type

```
DISK-STATUS (P
```

Press RETURN. This sends a disk status report to the printer. If shadowing is running successfully, the report is similar to the following:

```
All disks are shadowed.

 Volume   Channel(s)       Status

   1    F000 / F800    Both running shadowed.
   2    F880 / F080    Both running shadowed.
```

The first line of the report gives the current status of disk shadowing. This message is followed by a summary of the disk configuration and the status of each sib pair. If your report contains messages other than the ones shown above, read the next section to determine how to interpret an error status report.

Handling Disk Errors

Errors detected on all online disks are logged to the SYSTEM-ERRORS file as always. In addition, the status of the disk is noted and can be displayed by using the DISK-STATUS command.

When a disk error is detected with disk shadowing in effect, the normal retries are first attempted. If the error is transient, there is no effect or data loss in the shadowing operation. However, if the disk error cannot be cleared by retries, the bad disk is disabled via software. The system then sends a message with the time and date out on Port 0; for example:

> 00:15:18 22 JUL 1989 From Kernel: System is running without backup disks.

To determine which drive is no longer in operation, first use DISK-STATUS to determine if there is a problem with disk shadowing, then use LIST-SYSTEM-ERRORS to determine more information about the error.

Note: *All unimpaired disks continue to be used by the system as usual, including the disk shadowing on unimpaired sibs.*

Checking the Disk Status

To verify the current status of shadowing, log onto the SYSPROG account. Enter

```
DISK-STATUS {(P}
```

Press RETURN. The (P option outputs the report on the printer; otherwise it is displayed on the terminal. If there has been an error, the shadowed pairs that have been decoupled are identified.

The first line of the report gives the current status of disk shadowing. The possible status messages are:

> All disks are shadowed.
>
> n shadowed disks have failed.
>
> The disk set is not shadowed.

This message is followed by a summary of the disk configuration and the status of each sib pair.

The three sample reports below show the possible statuses:

```
All disks are shadowed.

 Volume  Channel(s)       Status

   1     0600 / 1800      Both running shadowed
```

```
2 shadowed disks have failed.

 Volume  Channel(s)       Status

   1     0600 / 1800      Both running shadowed.

   2     1880 / 0680      0680 has failed and is decoupled.

   3     0700 / Offline   Offline disk is decoupled.
```

```
The disk set is not shadowed.

 Volume  Channel(s)       Status

   1     E080             Running
```

Checking SYSTEM-ERRORS

If a disk has failed, you can use the SYSTEM-ERRORS file to help you determine the cause of the error. List the file using the LIST-SYSTEM-ERRORS command.

From SYSPROG, enter

LIST-SYSTEM-ERRORS

Press RETURN. The following prompts are displayed; respond as indicated:

```
To the Printer (Y=<CR>/N)? <CR>

System Error listing explanation (Y/N=<CR>)? <CR>

Would you like the list sorted (Y/N)? Y <CR>
```

After the report is printed, find the Disk Errors page. This page of the report shows all current disk errors in the system. Find the error code F000, which is the code for

Error caused decoupling of sib disks

The detail line on the report shows the channel address associated with the Decoupled error. The channel address is a 2-byte number in the column "CHAN #"; for example:

0680

Refer to your system configuration to match the channel address to the bad disk via its controller and drive number. You may wish to bootload to display the disk configuration for final verification before proceeding to the next step.

Call Ultimate Support for recommended recovery actions. This may include reformatting the disk, assigning an alternate track, or physical repair of the disk.

Restarting Disk Shadowing

Once a disk is ready to resume operation, there are two methods of restarting the disk shadowing:

- D-COPY
- File save and restore

Using D-COPY

1. :WARMSTOP the system.
2. Boot the system and select the Utilities Monitor from the Startup Options.
3. From the Utilities Monitor, use the D-COPY verb to perform a disk copy. The time involved should range from approximately 5-30 minutes. Please refer to the previous discussion of D-COPY.
4. When the repaired disk has been copied and is identical in data to its sib, use the LOADER command to return to the Startup Options menu. Select the Warmstart option.
5. Disk shadowing automatically resumes without any loss of data between the sibs.

Using File Save and Restore

1. Do a regular file-save from the remaining disks
2. Restore to the two sets of disks. The information is automatically restored to both sets of disks.

At the end of the restore, disk shadowing resumes automatically.

Error Codes Associated with Disk Shadowing

Code	Description
E800	Write failed despite 10 retries - status from original error
E900	Retry got a NAK - status is from original error
EA00	Recalibrate failed - status is from recal
EBxx	Status read during recovery got NAK - D_STATE in xx - status from original error
EC00	Read retry failed on shadowed disk - status from original error, retry will be on sib
ED00	Successful retry - status from original error
EE00	Retry limit reached - status from original error
EF00	Recalibrate got a NAK - status from original error
F000	Error caused decoupling - status from original error
F100	Not ready timeout - no status
FFDx	DIRAM decoupled - unit number in error is in x

Faster File Save on Ultimate 6000 Systems

The file save process has been improved and its speed significantly increased. Currently, this process is used on Ultimate 6000 systems with the following configuration:

- at least 4 Mb of available memory
- an HPP co-processor that supports memory addressing to 8 Mb and that has rev 82 or higher firmware
- 7XE processor

The actual amount of time saved depends on the organization of the data on the disk. The maximum saved time is realized on systems whose files are mostly in primary file space.

If the new process seems to have little effect, it is probably because of the way the files are stored. In this case, it is recommended that a file restore be performed.

GCR and Pertec Tape Drive Enhancements

The enhancements for GCR tape drives include the addition of an option that allows the tape drive to run in an unbuffered mode and the ability to specify tape density. The enhancement for Pertec tape drives is support for the FS1000, and includes the ability to specify density and speed.

GCR Tape Drive Enhancements

Two new T-ATT command keywords are available for use with the GCR tape drive: UNBUFFERED and DENSITY.

These keywords should be issued only when the tape is mounted, on-line, and at beginning of tape (BOT) mark. If either of these keywords are entered without the tape meeting these conditions, they have no actual effect.

The GCR tape drive uses a local input/output buffer to improve performance when accessing tape. However, use of this buffer prevents accurate tape error handling.

The keyword UNBUFFERED forces the device to run in unbuffered mode. When a tape error occurs while the system is in unbuffered mode, an option to mount a new reel has been added to the retry/quit options. This option sets a software end of reel condition, which preserves the data already written to the tape. The tape error prompt is similar to the following:

```
Tape write error : Status1 = xxxx  Status2 = yyyy  Bytes = zzzz
(S)wap reels/(R)etry/(Q)uit
```

To mount a new reel, enter S. The system writes an end of file (EOF) mark, an end of reel label, and another EOF mark. If the system cannot write the information, the retry/quit option without the swap is displayed.

When the tape is detached using T-DET, the buffer mode is reset to on; that is, the UNBUFFERED setting is canceled.

Note: *When the tape drive is running in UNBUFFERED mode, a status of 40 is displayed on the drive; this is normal and does not indicate an error.*

The DENSITY keyword can now be used with the GCR tape drive; however, the appropriate option bits must be set on the drive. These option bits can be set by running diagnostic test 84 from the front panel. The following procedure outlines the method for running this test:

1. If necessary, remove the tape from the drive.

2. Hold down the CE switch and press the TEST switch. The DIAGNOSTIC monitor lights up and displays 00.

3. Press STEP until the number displayed is 8.

4. Press TEST; this shifts the 8 to the left.

5. Press STEP until the number displayed is 84.

6. Press EXECUTE and hold until 01 is displayed. This indicates byte 1. Release EXECUTE. The current value in byte 1 is displayed. Use the procedure described in steps 3-5 to change the value to 12. Then press EXECUTE again to get the next byte. Repeat this procedure to set bytes 1-8. After the bytes have been set, the RESET light is on.

   The option bit settings are

Byte	Value
1	12
2	80
3	80
4	80
5	02
6	80
7	80
8	80

7. **After all the bytes have been set, press RESET, then power off and on the drive. This is necessary in order to update the settings in the drive's EPROM.**

8. The following DENSITY settings for the GCR may be specified:

   1600
   6250

Pertec Tape Drive Enhancements

The Pertec FS1000 tape drive is now supported on the Ultimate 6000/7000 series. Two options, SPEED and DENSITY, are available on the T-ATT command; SPEED allows the tape speed to be specified and DENSITY allows the number of bits per inch (bpi) to be specified.

In order to use the DENSITY and SPEED options, the function codes on the drive must be set as follows:

1. If necessary, remove the tape from the drive.

2. Press the FUNCTION SELECT switch and hold for at least three seconds, until 00 is displayed. This indicates function 0. Press the FUNCTION SELECT switch again and hold it; the next function is selected. Continue pressing the FUNCTION SELECT switch until function 10 is displayed.

3. The SET/CLEAR LED indicates the current state of the displayed function code: off or on. Press SET/CLEAR switch to toggle the setting.

4. Functions 10, 20, and 21 should be set as follows:

Function	LED Setting
10	off
20	on
21	off

5. After the function codes have been set, press RESET.

6. The following DENSITY settings may be specified for the FS1000:

   1600
   3200

7. The following SPEED settings may be specified for the FS1000:

   25
   100

9 VAX Enhancements

VAX enhancements include the following:

- changes to installation and startup procedures
- enhanced error logging and recording
- enhancements for tape attachment, including new defaults and keywords
- on-line help for Ultimate command qualifiers
- shutdown enhancements
- software disk cache
- spooler enhancements
- 16 parallel printers
- 124 interactive processes for ECP

For more information on the commands and features described in this section, see the *VAX/VMS Installation and Operation Manual* for version 200 of the Ultimate Operating System.

Changes to Installation and Startup Procedures

Startup Command Procedure

An Ultimate startup command procedure is now created during VMSINSTAL and copied into the manager's directory. This procedure contains information needed to define system logical names and to load the Ultimate device driver. The file specification for the procedure is

```
SYS$MANAGER:ULTIMATE_STARTUP.COM
```

The file is created based on answers to questions presented during the installation procedure.

You must change the Ultimate startup commands in the site-specific startup command procedure, either SYSTARTUP.COM or SYSTARTUP_V5.COM, to the following:

```
$@SYS$MANAGER:ULTIMATE_STARTUP.COM
```

Print File Directory

A new system logical name, ULTI$PRINT, has been added. ULTI$PRINT is used to define a directory that will contain print files before they are printed. ULTI$PRINT points, by default, to the same directory as SYS$ULTIMATE. To change the directory, edit the SYS$MANAGER:ULTIMATE_STARTUP.COM procedure.

Enhanced Error Logging and Recording

Print Spooling Errors

All informational and unsuccessful messages occurring during print spooling are logged to OPCOM as well as in the spooler log file in SYS$ULTIMATE. Included with each message is a date and time stamp.

System Errors

VMS Ultimate executable images now log all informational and unsuccessful messages to OPCOM. Included with each message is a date and time stamp.

Tape Errors

All informational and unsuccessful messages occurring during VMS tape processing are logged to OPCOM. Included with each message is a date and time stamp.

Enhancements for Tape Attachment

New Keywords

Two new keywords are available for T-ATT: ASSIST and DENSITY.

All tape processing requests from the Ultimate Operating System now use the VMS MOUNT/NOASSIST qualifier as the default. The new keyword for T-ATT, ASSIST, can be used to change the default.

The ASSIST keyword allows users to request VMS operator assistance in mounting a tape. If a tape is not on-line on the specified tape drive and the ASSIST keyword is included in the T-ATT statement, a message is logged to OPCOM requesting the operator to place the specified tape on-line. T-ATT waits for the tape to come on-line before continuing. The syntax for ASSIST is

```
T-ATT ASSIST
```

The DENSITY keyword allows the tape density to be specified when attaching magnetic tapes. The syntax for DENSITY is

```
T-ATT DENSITY = d
```

where d is the density. The following densities are supported:

800
1600
6250

T-ATT Errors

During a T-ATT, if an error occurs on the VMS mount, the VMS error message is now returned to the Ultimate process issuing the attach command.

```
:T-ATT 1

[180] Error detected by VMS:
%SYSTEM-F-DEVALLOC, device already allocated to another user
```

For a complete list of mount error messages, please see the *VMS Mount Reference Manual.*

On-Line Help for Ultimate Command Qualifiers

The VMS on-line help for the Ultimate Operating System has been enhanced to provide information on each of the Ultimate command qualifiers.

The following is a list of command qualifiers that can be used with the ULTIMATE command:

/ACCOUNT
/BREAK
/PASSWORD

The following is a list of command qualifiers that can be used with utilities available from VMS:

/ABORT
/CHANGE
/COLDSTART
/DELETE
/INITIALIZE
/RETURNTERM
/SHOWTERM
/SHUTDOWN
/TERMINIT

Note: */ABORT and /TERMINIT are qualifiers for the /SHUTDOWN qualifier.*

Software Disk Caching

An optional software disk cache sub-system has been added to the Ultimate kernel for ABS release 200. The cache system has the following advantages:

- minimizes the number of VMS disk I/O requests
- increases the average VAX transfer size
- increases effective throughput rate for disk I/O

The cache memory is allocated from VMS non-paged pool memory. When the Ultimate system faults on a frame (that is, the requested frame is not in memory), the system checks the cache memory for the frame; if the frame is in the cache, it is copied from the cache directly into Ultimate memory. If the frame is not in the cache, it is accessed from disk. When a disk access is necessary, the cache software reads a cluster of frames into the cache memory, then copies the requested frame into Ultimate memory. The number of frames in a cluster is determined by the user.

When writing data back to disk, the cache is not used, due to problems of maintaining cache consistency. However, the write process attempts to write logically contiguous frames back to the disk in a cluster.

Creating the Disk Cache

The disk cache is set up by the ULTI$INIT program at the time the Ultimate database is initially allocated. Two parameters are required to initialize the disk cache: Cache_size and Cluster_size. These parameters can be changed by the ULTI$CHANGE program.

Cache_size	determines the number of frames (pages) of VMS memory to allocate for the disk cache. The value specified for Cache_size must be a multiple of 512 pages. The default value for Cache_size is 2048 pages (one megabyte). If no cache is desired, specify a Cache_size of 0.
Cluster_size	determines the number of frames that are read into the cache by one disk I/O. The value specified for Cluster_size must be a power of 2, and must be between 1 and 16. The default value for Cluster_size is 4 frames.

Allocating VMS Memory

The cache memory is allocated from non-paged pool memory. Depending on the size of the cache and the size of the non-paged pool memory, the amount of memory reserved for the non-paged pool may need to be increased.

The pool size may be increased by adding a line similar to the following to the file SYS$SYSTEM:MODPARAMS.DAT:

```
add_npagedyn=xxx
```

where *xxx* is the number of bytes of non-paged memory allocated for the cache. The number of bytes can be determined by multiplying the number of frames specified for Cache_size by 512.

After modifying the SYS$SYSTEM:MODPARAMS.DAT file, you can put the size into effect by issuing the following DCL command:

```
@SYS$UPDATE:AUTOGEN GETDATA REBOOT
```

This command reboots the system and puts the new parameters into effect.

Disk Caching Performance

Disk caching is provided as a means of improving system performance. However, depending on the processes on the system, the cache may not increase, and in some instances may actually decrease, system performance. By changing the size of the cluster and cache, you should be able to improve the system performance.

To determine how your system is performing, monitor the disk activity. The following chart can be used to help decide what values to use for cluster and cache sizes:

If disk activity is	...and cluster size is	then try
high	small	increasing cluster size
high	large	increasing cache size
low	----	leave as is

High disk activity is considered to be more than 45 I/Os per second. Low disk activity is fewer than 10 I/Os per second.

If it is necessary to increase the cache size, it should be done in increments of 512 frames; after each increment, monitor the disk activity to determine if there is any effect.

It is possible to increase system throughput significantly during file saves and restores by increasing the cluster size. This is true of any other activity that reads information from the disk sequentially.

Spooler Enhancements

The following spooler enhancements have been made for the parallel printer interface.

New Sub-System

The VMS Ultimate print spooling sub-system has been rewritten. The new version allows multiple data transfers, up to 16 at a time, to be processed. It also supports up to 16 parallel printers. Previously, the spooler sub-system spooled print jobs to VMS one at a time and supported only four parallel printers.

VMS Print File Names

The VMS file that is created when information is transferred from Ultimate to VMS is now given a name made up of the Ultimate account name followed by an underscore and the line number of the Ultimate process, followed by an extension of .LIS. The file is located in the directory pointed to by the VMS logical name ULTI$PRINT.

For example, if a print job is initiated by the account SYSPROG on line 1, the VMS file will have the name

```
SYSPROG_1.LIS
```

This information is also included on the header page (first page) of the printed job.

If the Ultimate account name contains any character that is invalid in VMS file names, the character is changed to a dash. The following characters are not valid in VMS file names:

```
` ~ ! @ # $ % ^ & * ( ) + { [ } ] ; : ' " \ | < > , . ? /
```

Error Recovery

If the Ultimate VMS print spooling process encounters a fatal, non-recoverable error, the process logs the error in OPCOM and the spooler log file, then terminates.

After correcting the problem that caused the error, you can restart the Ultimate VMS spooler process by executing the following command procedure from a privileged account:

```
SYS$ULTIMATE:ULTI$SPOOLER.COM
```

Terminating Spooler Processes

The Ultimate print spooling system on the VAX now supports the TCL command SP-KILL for use with parallel printers. This command terminates the data transfer portion of the Ultimate print spooling process and deletes the print file created in VMS. The syntax for the command is

```
SP-KILL n
```

For more information on using SP-KILL, see the *Ultimate System Commands Guide.*

To terminate a print job after the job has started printing, issue the following VMS DCL command from a process logged into VMS:

```
DELETE/ENTRY = nnnn vms_queue_name
```

The print file remains in the directory pointed to by ULTI$PRINT; to delete the file, issue a VMS DCL DELETE command.

Shutdown Enhancements

The Ultimate Shutdown program has been enhanced to reduce the amount of time required to flush Ultimate memory to disk and shut down the Ultimate Operating System.

A new DCL qualifier, ABORT, has been added to the Ultimate Shutdown program that allows the system to be shut down while Ultimate users are still logged on. The syntax for the qualifier is

```
$ ULTIMATE/SHUTDOWN/ABORT
```

WARNING! *The use of the ABORT qualifier can cause GFEs and loss of data. Extreme care should be used when specifying this qualifier during the Shutdown process.*

Notes

10 1400 Enhancements

Enhancements for the 1400-specific implementation include the following:

- all data frames now contain 2K bytes; this feature requires reallocation of all files. For more information, please see the *Upgrade Procedures for Revision 200 for 1400 Systems*
- support for the following system commands, which provide modem control, has been added:

  COFF

  DROP-DTR

  DROP-RTS

  RAISE-DTR

  RAISE-RTS

  RESET-LOGOFF

  SET-LOGOFF

Note: The revision number for the 1400 implementation is 205.

Modem Control

The modem control features are available to lines that are connected to the system by an SP1 or later communications controller. The features do not affect lines that are connected to an SP0 controller.

The system commands work as documented for the Ultimate 6000/7000 systems in the *Ultimate System Commands Guide* , except that **both** DTR (Data Terminal Ready) and RTS (Request To Send) are effected by all the commands. The following is a brief description of each feature. For more information, please refer to the *Ultimate System Commands Guide.*

COFF

COFF logs off the current line and drops both DTR and RTS.

Syntax

COFF

DROP-DTR

DROP-DTR drops both DTR and RTS on a specified line. If the line is not specified, DROP-DTR drops DTR and RTS on the current line.

Syntax

DROP-DTR {line.number}

DROP-RTS

DROP-RTS drops both DTR and RTS on a specified line. If the line is not specified, DROP-RTS drops DTR and RTS on the current line.

Syntax

DROP-RTS {line.number}

RAISE-DTR

RAISE-DTR raises both DTR and RTS on a specified line. If the line is not specified, RAISE-DTR raises DTR and RTS on the current line.

Syntax

RAISE-DTR {line.number}

RAISE-RTS

RAISE-RTS raises both DTR and RTS on a specified line. If the line is not specified, RAISE-RTS raises DTR and RTS on current line.

Syntax

RAISE-RTS {line.number}

RESET-LOGOFF

RESET-LOGOFF disables the automatic logoff function set up by SET-LOGOFF on a specified line. If the line is not specified, RESET-LOGOFF disables the automatic logoff function on current line.

Syntax

RESET-LOGOFF {line.number}

SET-LOGOFF

SET-LOGOFF sets up a specified line to automatically log off if either CD (Carrier Detect) or CTS (Clear To Send) is detected; it also causes the system to drop both DTR and RTS. If the line is not specified, SET-LOGOFF sets up the automatic logoff function on current line.

Syntax

SET-LOGOFF {line.number}

Notes

A List of ASCII Codes

This appendix presents a list of ASCII codes for decimal number values from 0 through 255. The hexadecimal equivalent value and ASCII character generated are also given.

Decimal values 0-31 are assigned as non-printable functions; these codes may be specified by control key sequences as input to a BASIC program. In the listing, the control key is indicated by a caret (^) in the first position in the Key column.

Decimal values greater than 127 (x'7F') are not defined in the ASCII character set. The functions or characters assigned to these values are dependent on the terminal being used. However, special file structure functions and control key sequences have been assigned to decimal values 251 through 255 (x'FB' through x'FF').

Decimal	Key	Hexadecimal	Name
0	^@	00	NUL
1	^A	01	SOH
2	^B	02	STX
3	^C	03	ETX
4	^D	04	EOT
5	^E	05	ENQ
6	^F	06	ACK
7	^G	07	BEL
8	^H	08	BS
9	^I	09	HT
10	^J	0A	LF
11	^K	0B	VT
12	^L	0C	FF
13	^M	0D	CR
14	^N	0E	SO
15	^O	0F	SI
16	^P	10	DLE
17	^Q	11	DC1
18	^R	12	DC2
19	^S	13	DC3
20	^T	14	DC4
21	^U	15	NAK
22	^V	16	SYN
23	^W	17	ETB
24	^X	18	CAN
25	^Y	19	EM
26	^Z	1A	SUB
27	^[	1B	ESC
28	^\	1C	FS
29	^]	1D	GS
30	^^	1E	RS
31	^_	1F	US

Decimal	Key	Hexa-decimal
32		20
33	!	21
34	"	22
35	#	23
36	$	24
37	%	25
38	&	26
39	'	27
40	(	28
41	)	29
42	*	2A
43	+	2B
44	,	2C
45	-	2D
46	.	2E
47	/	2F
48	0	30
49	1	31
50	2	32
51	3	33
52	4	34
53	5	35
54	6	36
55	7	37
56	8	38
57	9	39
58	:	3A
59	;	3B
60	<	3C
61	=	3D
62	>	3E
63	?	3F
64	@	40
65	A	41
66	B	42
67	C	43
68	D	44
69	E	45
70	F	46
71	G	47
72	H	48
73	I	49
74	J	4A
75	K	4B
76	L	4C
77	M	4D
78	N	4E
79	O	4F
80	P	50
81	Q	51
82	R	52
83	S	53
84	T	54
85	U	55
86	V	56
87	W	57
88	X	58
89	Y	59
90	Z	5A
91	[	5B
92	\	5C
93	]	5D
94	^	5E
95	_	5F
96	`	60
97	a	61
98	b	62
99	c	63
100	d	64
101	e	65
102	f	66
103	g	67
104	h	68
105	i	69
106	j	6A
107	k	6B
108	l	6C
109	m	6D
110	n	6E
111	o	6F
112	p	70
113	q	71
114	r	72
115	s	73
116	t	74
117	u	75
118	v	76
119	w	77
120	x	78
121	y	79
122	z	7A
123	{	7B
124	|	7C
125	}	7D
126	~	7E
127	DEL	7F

Decimal	Hexadecimal	Symbol	Name
128 (x'80') thru 250 (x'FA')		not used	
251	FB	SB	Start buffer
252	FC	SVM	Subvalue Mark
253	FD	VM	Value Mark
254	FE	AM	Attribute Mark
255	FF	SM	Segment Mark

B Summary of Revision 190

The following features were added to Revision 190 of the Ultimate Operating System, Release 10:

- TCL stacker
- new TCL prompt
- use of keywords in TCL commands
- item sizes larger than 32k
- conversion code that provides the ability to call BASIC subroutines from Recall
- relational operator that provides soundex capability in Recall
- INPUTCONTROL statement in BASIC
- selective transaction logging
- Update enhancements
- UltiKit enhancements

For a complete description of the new features in Revision 190, please refer to the *Revision 190 Guide to New Features*.

Notes

Index