

MailForm™ for Ink-Jet

Reference Manual



SoftView Systems, Inc.

Revised Edition - January, 1999

Notice

SOFTVIEW SYSTEMS MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

SoftView Systems shall not be liable for errors contained herein or for incidental consequential damages in connection with the furnishing, performance or use of this material.

All rights reserved. No part of this document may be used or reproduced in any form or by any means, or stored in a database or retrieval system, without prior written permission from SoftView Systems, Inc. Making copies of this book for any purpose is a violation of United States copyright laws.

The information contained in this document is subject to change without notice.

Copyright © 1997, 1998, 1999 by SoftView Systems, Inc.

For information contact:

SoftView Systems, Inc.,
9411 Meadow Woods Lane, Dayton OH 45458 USA
(937) 436-1189

Printing History

First Edition	May, 1997
Revised	November, 1997
Revised	February, 1998
Revised	January, 1999

Current MailForm™ Version - 1.8

Trademark Credits

DijiComp™ is a trademark of Scitex Digital Printing Systems.
Microsoft® is a U.S. registered trademark of Microsoft Corporation.
Windows™ is a trademark of Microsoft Corporation. TrueType is a registered trademark of Apple Computer Inc. MailView™, FontView™, MailForm™ and MailProof™ are trademarks of SoftView Systems, Inc.

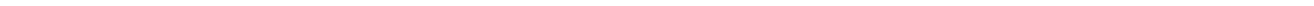
Contents

Introduction	1
MailForm™ Capabilities	3
Page Formatting	3
Variable Data Manipulation	4
Operational Features	5
Printer Systems Supported	6
MailForm™ Operation	7
The MailForm Job Queue	7
MailForm Remote Controller	8
Submitting a Job	8
Queue Status	10
Preferences	12
MailForm Server	13
Hold/Release the Job Queue	13
The Queue Display	14
The Job Display	15
Starting a Job from the Server	16
Filepath Preferences	17
Job Queue Preferences	19
Log File Output	20
The MailForm™ Control File	21
Statement Order	24
Control File Statement Descriptions	25
MAILFORM Statement	25
PRINTER Statement	26
INFILE Statement	28
OUTFILE Statement	31
OPTIONS Statement	32
RUN Statement	35
ERRORS Statement	38
INPUT Statement	40
RIP Statement	42
FONT Statement	44
DEFINITIONS Statement	46
TRANSLATE Statement	52
GROUPTRANS Statement	54

LOOKUP Statement	55
LIST Statement	57
EXTERNAL Statement	58
VARIABLE Statement	60
CONDITION CLAUSES	62
VARIABLE Content Clause	64
Data Descriptors	64
Constant Descriptor	64
Type Descriptors	65
List Descriptor	66
Document Number Descriptor	67
Input Record Number Descriptor	68
Input File Number Descriptor	69
Compute Descriptor	70
Input Descriptor	71
Reference Descriptor	72
DATA MODIFIER GROUPS	73
Variable Statement Example:	75
CONDITION Statement	77
ACTION Statement	79
DOCUMENT Statement	81
PAGE Statement	83
BUFFER Statement	85
BOUNDS Statement	86
FRAME Statement	88
TEXT Statement	92
Text Commands	92
Text Command Descriptions	94
Text Strings	101
Graphic Drawing Statements	103
BOX Statement	104
RULE Statement	105
IMAGE Statement	107
END Statement	108
Error Messages	109
MailForm Log File	109
Critical Resource Errors	109
Severe Errors	109
Job Start-up or Format Errors	112
Run-Time Errors	134
MailForm Server Messages	139
MailForm Remote Messages	141

APPENDIX A

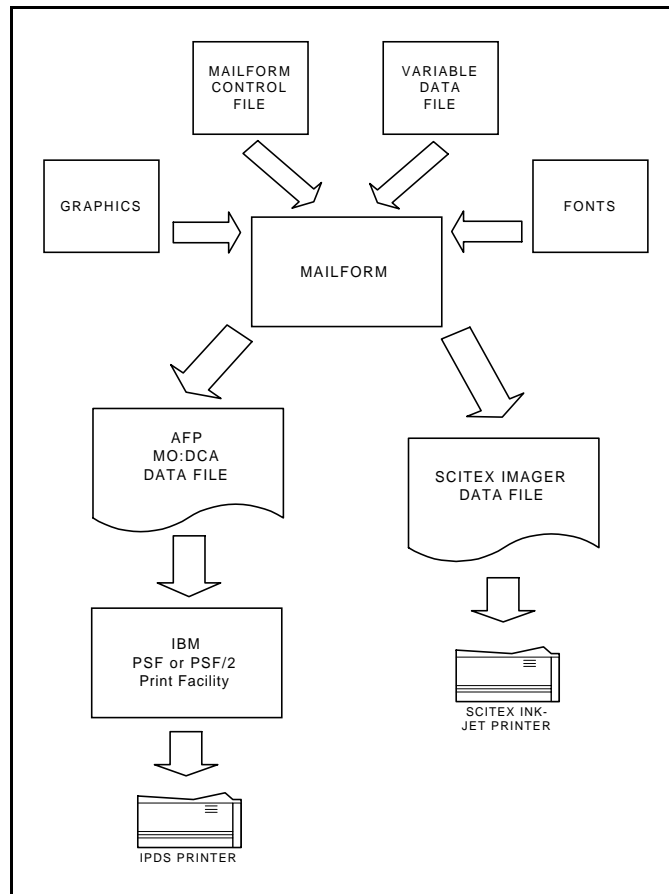
Supported Printer Systems	144
APPENDIX B	
MailForm Built-In Procedures	145
Appendix C	
Developing External Procedures for MailForm	151
Communication Word	152
Externals 0 through 31	154
External 32	155
External 33	155
External 34	156
External 35	157
External 36	158
External 37	158
External 38	159
External 39	159
External 40	159
Return Codes in the Communication Word	160
Restarting	160



1

Introduction

MailForm™ is a text composition software system used for preparing printer files for high-speed variable printing systems. Essentially, MailForm merges variable data from a Variable Data File with a document description supplied in the MailForm Control File following merge parameters and formatting instructions also supplied in the latter file. The output is a printer file formatted to drive Scitex IJPDS-compatible printers (or, in future, IPDS-compatible printers).



The diagram above illustrates the data flow in a MailForm system processing output for IPDS and Ink-Jet printers. The MailForm Control File is normally generated as an output from the MailView™ WYSIWYG front-end system

(although it is possible to generate it with any ASCII text editor). The Control File specifies the page size and orientation, the fonts and graphic elements to be used, and how the text and graphics are positioned. It also defines how variable information is extracted from the Variable Data File and combined with the text to form personalized output documents. The Control File may specify complex conditional processing of the variable data input and reformatting of the data via built-in and external routines.

The MailForm software operates under Windows/NT as a client/server multi-threaded task, allowing it to operate extremely efficiently on single or multi-processor computing systems. MailForm output files may be generated on disk or can be directed to 9-track tape (1600/6250) or to 3480/3490 data cartridge.

Although the two software systems are very different, MailForm provides complete functional compatibility with the Scitex DijiComp™ mainframe system. Existing DijiComp program files may be converted to MailForm format via the SoftView MailView™ system.

2

MailForm™ Capabilities

Page Formatting

- Typesets text using dot-matrix fonts within a variable line length. Automatically cascades excess text to succeeding lines. Does not provide automatic word hyphenation.
- Sets text lines flush left, flush right, centered or justified to the current line width.
- Allows font changes anywhere in a line and adjusts text vertical positioning to ensure that all character baselines line up. Up to 255 fonts of 256 characters each may be used in a given job. Where multiple-RIP systems are used, each RIP may use 255 fonts.
- Text blocks may be positioned anywhere on a page with the text oriented in any of four orientations - upright, inverted, rotated left or right.
- Indenting in text blocks (or frames) may be specified at left and/or right ends of the line and also for just the first line of a paragraph or for all lines after the first (hanging indent).
- Spacing may be inserted horizontally and vertically either from an absolute position or from the current (and possibly variable) position. Line spacing may be specified as additional to the basic font height or line-to-line for greater flexibility.
- Can draw rules and boxes with varying line thicknesses or solid rectangles.
- Can logically combine text and drawing elements allowing white-on-black text and other effects to be generated.
- Can include variable bitmap graphic elements.
- For Scitex printers, automatically optimizes text generation on the printer tape so that lines that will be identical in all copies of a document are stored in the printer's memory and not repeated on the printer file.
- Can generate multiple pages in one document.
- Allows up to 256 different, selectable document formats in one job.

-
- Can select groups of text lines for one instance of a document and optionally exclude them from another (or substitute different text).
 - Allows, by use of special fonts, incremental rotation of text frames in units of 1/10 degree. This requires only the provision of the special font - no other controls are necessary.
 - Underline commands allowing underscoring of fixed and variable text and underscores that cascade to following lines without requiring special fonts.
 - Vertical Justification of text blocks to top, bottom or centered in the frame area.

Variable Data Manipulation

- Selects text character groups from variable input records for inclusion at any point in the document. Variable text may be subjected to extensive editing (such as capitalization) using built-in functions. Over thirty different editing procedures are available and as many of these as required may be applied in any order to achieve the desired result.
- Can compare variable data and use the results to control document selection, text group selection and/or variable text selection, formatting and placement. The controls provided form a high-level programming language which is extremely flexible and powerful.
- Can perform character replacements on any part of the variable data.
- Can look up variable data elements in tables defined in the control input and substitute other text when and if a match is detected.
- Can test variable data elements against lists defined in the control input and perform actions and/or generate text based on whether a list item is matched.
- Can perform arithmetic computations on variable data elements to derive values to be printed and/or to control positioning, formatting and placement.
- Can call user-written external procedures at various

points in the processing sequence to perform any special formatting or data manipulation required.

- Provides test capabilities for generating printer files that display each possible conditional document format, text substitution, variable data selection etc only once thus allowing the programmer to check all possibilities, or to know which ones have not been exercised.

Operational Features

- Generates printer files to control single printers, Scitex Multi-RIP systems and optionally up to ten interconnected Scitex printers via up to ten simultaneously-written data files.
- Printer files may be formatted for disk or sequential tape devices. Tape output uses IBM Standard Label formatting.
- Variable Input files may be input on disk or via tape devices. IBM Standard Label format and Undefined tape formats are supported. Fixed Blocked, Unblocked and IBM Format V variable formats are supported. For both input and output files, automatic switching between primary and alternate tape drives is supported.
- MailForm can read from up to 10 variable input files simultaneously and output to up to 10 printer files for one job. Input files may be read forward or backward.
- MailForm can create a Checkpoint data file to which it writes periodically. The contents of this file may be used to restart the job after a deliberate or accidental stop.
- An Executive procedure allows network clients to submit and queue jobs for execution on the MailForm server. A client-based display element allows users to check job status and progress and to retrieve results of completed runs.
- The server-based Executive allows up to six simultaneous job executions and provides a continuous status readout for the operator.

**Printer Systems
Supported**

- Scitex 3500/3600 Ink-Jet Imagers
- Scitex 3500/3600 Multi-RIP Ink-Jet Imagers
- Scitex 3000 (120dpi) Ink-Jet Imagers
- Scitex 6240 Imager Systems (with IJPDS input capabilities)
- Scitex 5240/5120 Imager Systems (with IJPDS input capabilities)

3

MailForm™ Operation

MailForm is implemented as a Client/Server application. MailForm Server is installed on a server system running WindowsNT (3.51 or 4.0). MailForm Remote Controller is installed on any Windows95 or WindowsNT workstation on the network from which MailForm jobs will be submitted. Jobs submitted from the remote workstations are placed in a Job Queue from which they are selected for execution when resources are available.

The MailForm Job Queue

MailForm Server maintains a Job Queue into which all jobs submitted by workstations running MailForm Remote Controller are initially placed. A new job enters the queue with 'QUEUED' status. MailForm Server scans the queue for 'queued' jobs every 5 seconds when it has an available 'processing slot'. Jobs having a status other than 'queued' are not considered for execution. MailForm Server can potentially run six concurrent copies of MailForm in six processing slots. When all are filled, MailForm Server does not scan the queue until one of the six jobs completes. MailForm will run concurrently only the number of jobs for which the attached security key (or keys) have licenses.

When selecting a job from the 'queued' entries, MailForm Server chooses the job having highest priority level (lowest number). Should there be more than one job with the highest priority, MailForm Server selects the job that was submitted first.

The Job Queue file may be shared by more than one copy of MailForm Server, perhaps running on different physical server systems. In this case, the queued job will be executed by the server that has the next available processing slot. To enable this type of organization, the Job Queue file must be placed in a shared directory with read/write access to all servers. This may be specified via "Filepath Preferences".

Entries in the Job Queue that represent completed jobs are not removed until explicitly deleted by the owning user via MailForm Remote Controller.

MailForm Remote Controller

The MailForm Remote Controller is a utility program that allows users to submit MailForm jobs from a network workstation for execution by the MailForm Server. Jobs are placed in a job queue which may be interrogated and controlled by MailForm Remote Controller.

Submit MailForm Job

Control File Path: Browse

Priority Level:

MailForm Server

A B Either One

Notify me when Job Completes

OK Cancel

Submitting a Job

Select "File | Submit" to add a new job to the MailForm job queue. The dialog box that appears has the following data items:

1. Control File Path - this is the filepath to the MailForm Control File. The filepath must be in a directory that is shared on the network. Also, the filepath must be specified using the sharename of that directory so that the server can access it. For instance, if you are submitting a Control File whose full filepath is:

E:\MAILFORM\DATA\CFILE.DAT

and the E:\MAILFORM directory is set up for sharing as CONTROL, you would enter:

`\CONTROL\DATA\CFILE.DAT`

If you always submit files from the same directory, the sharepath can be set up via the Preferences menu entry, so that all you have to enter on the Submit Dialog is the filename. Mailform Remote also will add a default extension of .FIL if none is specified. You can use the 'Browse' button to get an 'Open File' dialog which allows you to search for the file.

2. Priority Level - this is a number that controls the order in which jobs are selected from the MailForm job queue for execution. A low number is a high priority and increases the chances of early execution. If you leave the Priority box blank, the Remote Controller will assign a priority of 25 as the default. The lowest priority value that can be requested by a non-privileged user is 5.
3. The 'MailForm Server' group of radio buttons is used when more than one MailForm Server is operating from the same Job Queue and you want to select which server is to process the job. The default is 'Either One' which means the job will be processed by the first available server. Server selection is valuable if your Scitex printers are directly connected to a specific MailForm Server's output disk resource and you do not want them reading the output file over the main network.
4. The 'Notify me when Job Completes' checkbox causes the Remote to pop up a message when the Server completes processing on the submitted job. The Remote must remain running for this to work - you can minimize it into the system tray, but do not exit the program or you will not be notified.



Queue Status

This display allows you to view the current status of the top six jobs that you submitted to the MailForm queue. The display does not show any jobs submitted by other users on the network. The display shows the 8-character job ID, the current status of that job and, if the job is running or finished, the current or final document number.

Below the status display area are four buttons - OK, Update, Display Next and Display Prevs. OK removes the display and allows you to do something else. Update is used to refresh the status being shown. MailForm Remote does not automatically continue to get job status information from the Server while the Queue Status display is on-screen. If you want the display updated, click on the Update button and new job status information will be obtained from the Server and redisplayed. The Display Next button is used when you have more than 6 jobs in the queue and you want to see the next group of six - Display Prevs shows you the previous group of six.

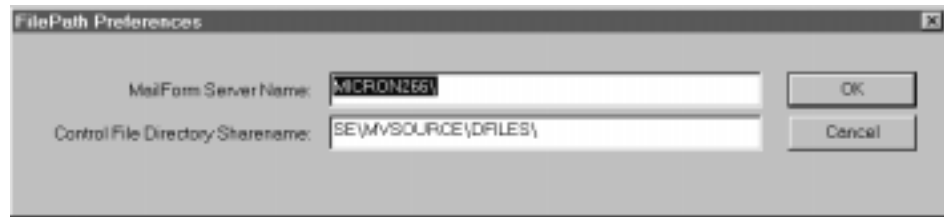
To the right of each line of job status information is a radio button (a small circular item that goes black when you click on it). This column of radio buttons is labeled CHG and you must click the appropriate button in the Change column if you want to change an item in the job queue with the buttons on the far right of the display. The rightmost

group of buttons, in the 'Modify Entry' group area, perform the following actions on the selected job entry:

Higher/Lower Priority	Change the priority of the queued job up or down
Hold/Release	Hold the queued item or release an item previously held (only items that have Queued status will respond to this button).
Pause/Run	Pause a running job or continue running a job previously paused.
Stop	End a running job (allows for a subsequent restart).
Cancel	Cancels a queued job or aborts a currently-running job.
Swap Servers	Changes the job to the alternate server (job must have Queued status).
Any Server	Changes a job from a specified server to any available one (job must be Queued to allow change).
Restart	Restarts a previously-stopped job from the last checkpoint.
Delete	Delete the selected entry from the job queue (running jobs must be Canceled or Stopped before they may be deleted).

After selecting the job with the radio button, click on the desired Action Button in the rightmost column. The action will be transmitted to the Server and the result will be immediately redisplayed on your Queue Status display.

The two other radio buttons at bottom right control what jobs are shown in the Queue Status display. 'Show Wait/Run' causes only those jobs that are queued or running to be displayed - completed jobs are not shown. 'Show All' displays all jobs in the queue.



Preferences

The only other significant item that you can perform via MailForm Remote Control is to set filepath preferences. Select the Preferences menu item and a dialog box appears that allows you to set the Server Name and the Control File Directory path defaults.

The Server Name is the computer name of the computer on which MailForm Server is running or will be running. If more than one Server is running MailForm, the Server Name is the one with which your copy of MailForm Remote is to communicate. No backslashes are necessary before or after this name. The Server Name should be set once when you set up the MailForm system or when you change servers.

The Control File Directory Sharepath is the path to the directory from which MailForm Control Files will normally be submitted to the queue. This directory (or a higher-level directory) must be set up for shared access in at least read mode - thus allowing the Server to read the file submitted for execution. The default path in this entry should specify the sharename of the directory (or its parent directory) so that the server can use the path to access the file. If the sharepath includes your computer's network name, it must start with two backslashes:

E:\MAILFORM\DATA on computer FRED where
E:\MAILFORM is shared as CONTROL, would have a
sharepath of

`\CONTROL\DATA` or `\\FRED\CONTROL\DATA`

If you do not include the computer name, MailForm will automatically add it to the default path.

MailForm Server

MailForm Server runs multiple copies of the MailForm program. MailForm merges text and control commands from a Control File with variable data from a Variable Input File to form an IJPDS Output File to drive Scitex Ink-Jet Printer Systems.

MailForm Server can run up to six copies of the MailForm program simultaneously if licenses are available. A MailForm license allows one copy of MailForm to execute through the Server at one time. To run six simultaneous copies of MailForm on one Server, six licenses (or an unlimited license) must be available through the Security Keyplug.

The MailForm program is not multi-threaded within one instance, but each copy executed by the Server is a separate thread and may therefore be executed by a different CPU on multi-processor systems under WindowsNT.

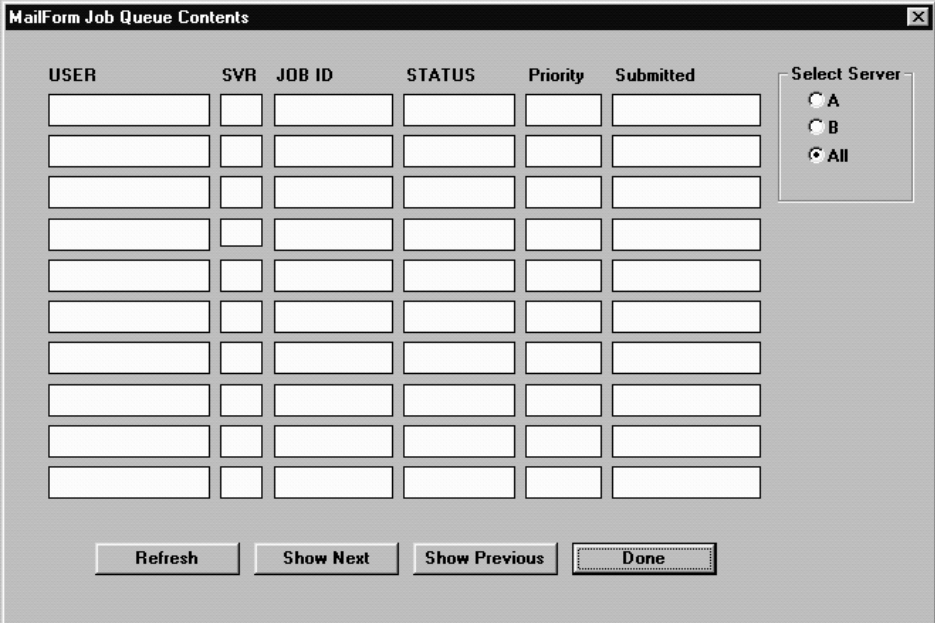
Hold/Release the Job Queue

Select the menu item "Queue | Hold" to prevent MailForm Server from initiating any new jobs from the queue. All currently-executing jobs will continue without interruption. This function may be used to cause an orderly system shutdown.

To restart the queue, select "Queue | Restart" from the menu. When MailForm Server is started, the default state is to immediately attempt to initiate new jobs, so a restart is not necessary in this circumstance.

Purge the Job Queue

As explained later under 'Preferences', aborted or completed entries remain in the job queue until the owner deletes them. Under 'Preferences' you can set a limit of the number of days these entries are held before being automatically deleted. Normally this automatic deletion of old entries only happens when the MailForm Server is started up. If you want at any time to clear out the queue without restarting the Server, select 'Queue | Purge'.



The screenshot shows a window titled "MailForm Job Queue Contents". It contains a table with the following columns: USER, SVR, JOB ID, STATUS, Priority, and Submitted. There are 10 rows of empty input fields for each column. To the right of the table is a "Select Server" section with three radio buttons: A, B, and All. The "All" radio button is selected. At the bottom of the window are four buttons: Refresh, Show Next, Show Previous, and Done.

USER	SVR	JOB ID	STATUS	Priority	Submitted

The Queue Display

Select the menu item "Queue | Display" to cause the contents of the Job Queue to be displayed in the Server window. The first ten job entries in the queue are displayed initially. The display shows all jobs submitted by all users. Use the "Show Next" and/or "Show Previous" buttons to advance or back up the display respectively. The queue entries cannot be edited from this display. The queue display is provided to help the server operator decide

when to use the Queue Hold/Restart function. Queue entries may currently only be changed by the user that submitted them.

The 'Refresh' button redisplay the queue from the current pointer with updated data. 'Done' indicates that you no longer wish to see this display.

By default, the Queue Display shows jobs specified for all servers. You can click on the 'A' or 'B' radio button in the 'Select Server' group and then click the 'Refresh' button to display jobs specified for a particular server (also shows jobs that are specified for 'Any Server'). The 'SVR' column of the display shows which server the jobs are specified for.

The screenshot shows a window titled "MailForm Job Status" with a table of job information. The table has six columns and several rows of labels. Below the table are two rows of buttons: "Pause" and "Abort", each with one button per column.

	SVR	SVR	SVR	SVR	SVR
Job ID:					
Status:					
Document Number:					
Input Unit:					
Alternate Unit:					
First Output Unit:					
Alternate Unit:					
Second Output Unit:					
Alternate Unit:					
Checkpoints ?					
	Pause	Pause	Pause	Pause	Pause
	Abort	Abort	Abort	Abort	Abort

The Job Display

MailForm Server automatically shows a window which displays the current status of the last six jobs that were started. If less than six jobs are currently executing, the remaining display slots will show the final status of jobs that recently completed. The display is automatically updated every second with the current document number and thus provides a good indication of how the work is

progressing.

When a job is using tape input or output devices, the display shows the tape unit currently being used and the unit on which MailForm expects to see the next reel when it finishes the current one.

In line with each job on the display are two buttons - "Pause" and "Abort" - which the operator may use to temporarily or permanently stop the associated job. To restart a paused job, click a second time on the "Pause" button. When the "Abort" button is used, the job is actually stopped at the end of the next document and the output file is completed so that the job can be run on the ink-jet printer up to that point.

The display also shows the operator whether Checkpoints are being taken or not. This information may affect the decision as to whether to stop the job or let it continue to completion.

Starting a Job from the Server

The operator at the MailForm Server terminal may start a MailForm job that is not in the queue. There must be an available "processing slot" in order to do this successfully. Select "Job | Start" from the menu. An "open file" dialog box will appear allowing you to select a MailForm Control File to be executed.

Following this a new window will open for the new copy of MailForm and another dialog box may appear requesting Variable Input File and IJPDS Output File paths if these have not been provided via INFILE and OUTFILE statements in the Control File. (This dialog box never appears when running a queued job and disk files are being used - if INFILE and OUTFILE statements are not present or are incorrect, the job is aborted). This dialog box does appear when tape input/output is used so that

the operator can enter tape drive unit numbers.

When the Variable Input and IJPDS Output data has been entered, the job should run like a queued item.

To restart a directly-submitted job from a checkpoint after it has been interrupted, select "Job | Restart" from the menu, select the same MailForm Control File in the "open file" dialog and the job should restart from the most recent checkpoint data. When the Variable Input is from tape, the most recently-used reel should be mounted. For tape output, a new tape should be mounted for the first output reel.

FilePath Preferences

MaiView 240dpi Fonts Directory: F:\FONTS240\

MaiView 120dpi Fonts Directory: F:\FONTS120\

Job Queue Filepath: MAILFORM.JQU

Bitmap Image Files Directory:

Log and Checkpoint Files Directory: E:\

Variable Input Files Directory: E:\MVSOURCE\TESTS\

External Procedures Directory:

IJPDS Output Files Directory: D:\

Control Files Directory: e:\mvsource\files\

Tape Output Type: Cartridge 9-Track

OK

Cancel

Filepath Preferences

Select "Preferences | Filepaths" from the menu to set up the directories MailForm is to use to find the various data items it requires. It is necessary to provide at least one of these filepaths before MailForm can be successfully executed following installation - the MailView Font Directory paths.

The dialog box that appears has eight data entry boxes. The first two are the paths to the MailView 120dpi and

240dpi font directories. If you are not running any 120dpi jobs, that line may be left blank.

The Variable Input File Directory may be entered if you expect your Variable Input files to appear mostly in one directory. This can always be overridden by a complete path specification in the INFILE statement in the Control File.

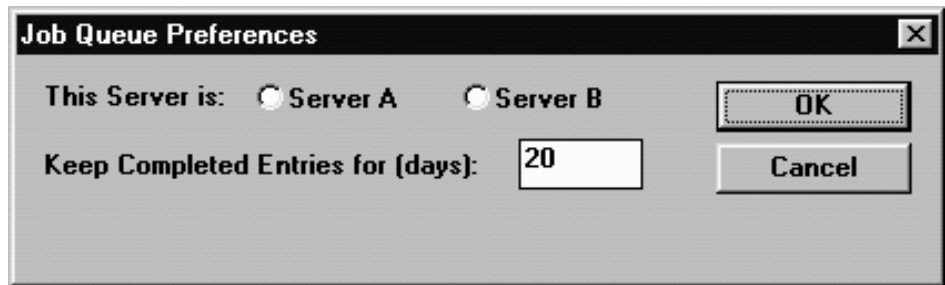
Similarly, the IJPDS Output File Directory may be entered if you expect those files will be generated mostly in one directory. It can also be overridden by a full path in the OUTFILE statement.

The Bitmap Image Files Directory should be entered when using IMG statements in the Control File, and the External Procedures Directory should be entered when External Procedures are expected to be called.

The Log and Checkpoint Files Directory is optional. If omitted, MailForm will attempt to generate the Log file in the same directory from which it read the Control File (which may be local to the workstation submitting the job) and the Checkpoint files will be generated in the current working directory on the server.

The Job Queue path is necessary if the job queue is to be accessed by more than one MailForm Server. If not specified, the queue is generated in the current working directory on the server.

At the bottom of the Filepath Preferences dialog box is an item reading "Tape Output Type:" with two radio buttons to the right of it - one reads "Cartridge", the other "9-track". If using tape output, select the type of output drive being used. This option selects the type of filemarks written. If the wrong option is selected, an error will occur during output to the tape.



Job Queue Preferences

Select “Preferences | Job Queue” to bring up the Job Queue Preferences dialog box.

The top line on this dialog allows you to specify the identity of this Server in a multi-server setup. Click either the ‘Server A’ or ‘Server B’ button. Of course, when setting the preferences on the other MailForm Server, the opposite button should be selected. In a single-server operation, ‘Server A’ should be selected.

The other item on this control determines the ‘automatic aging’ of Job Queue Entries. Normally, only the owner of a Job Queue entry can modify or delete it. However, most people will not clear out old entries regularly, so MailForm has a parameter that allows automatic deletion of entries that have been in the queue for a long time.

By default, this automatic aging of queue entries is inactive. By specifying a number of days in the Job Queue Preferences dialog, the limit becomes active. Each time that MailForm Server is executed (started from the Start Menu or via the Start-Up folder), the Job Queue is scanned for unused entries and entries that:

1. Have either Aborted or Completed status and
2. Have been in the queue for more than the number of days specified as the limit.

All these entries are deleted and the Job Queue file is condensed and shortened to minimum length.

Log File Output

All feedback from MailForm to the user is provided in the Log File. The Log File is generated in the directory indicated by the “Log and Checkpoint Files Directory” entry on the Filepath Preferences dialog box in MailForm Server. If this is blank, Log Files will be generated in the same directory as the Control File for the job - if using this option, you must ensure that the server has write privileges in that directory. Log files have the same name as the Control File with an extension of .LOG.

The Log File lists the complete Control File, with embedded error messages when errors occur during the run. If any error is detected in the Control File, MailForm aborts the job and generates no output file(s) other than the Log File. If the job is started, MailForm first shows the amounts of Font and Fixed File storage used by the job (and the remaining free space). Following this are listed all the variable input volumes read and the number of records processed from each. This is followed by a listing of the output volumes generated, the number of documents on each file and the starting document number on each volume. Finally totals for input records and output documents are listed and the elapsed time is noted.

4

The MailForm™ Control File

The MailForm Control File specifies how records of the Variable Data File are to be processed to create an output file for the specified printer device.

The file consists of 80-character records in ASCII, with or without carrier return and linefeed characters after each record. The character set used is from X'20' through X'7F' which allows the file to be printed and transferred via modem without alteration.

The following statements are identified by a name beginning in the first character of the line. Continuations, if required, must have a blank as the first character. The statements should be assembled in the file generally in the sequence listed below:

MAILFORM	Identifies the file
PRINTER	Specifies the printer type and model and the output mode
INFILE	Indicates the path to the Variable Input File
OUTFILE	Indicates the path(s) to the Imager Output File(s)
OPTIONS	Selects options for this job
RUN	Parameters specific to this execution of the job
ERRORS	Selects error reporting options
INPUT	Supplies data about the input file
RIP	Parameters specific to Scitex RIPs
FONT	Specifies the fonts to be used
DEFINITIONS	Defines special character type groupings
TRANSLATE	Defines character translation tables
GROUPTRANS	Defines character-group substitution tables
LOOKUP	Defines Lookup Tables for conversions
LIST	Defines List Tables for condition testing
EXTERNAL	Specifies External procedures to be

VARIABLE	linked for special processing Specifies variables, conditions and procedures to develop their contents
CONDITION	Specifies conditions that control variable contents and the appearance of text areas
ACTION	Defines an action to be taken
DOCUMENT	Starts the text of a document
PAGE	Starts a new page and specifies dimensions
BUFFER	Starts a new page buffer
BOUNDS	Sets an area check within a page
FRAME	Specifies position, size and characteristics of a frame
TEXT	Specifies the text content of a frame
BOX	Specifies a graphic rectangle
RULE	Specifies a graphic line
IMAGE	Specifies a bitmap picture image
END	Ends the file

Many of the above statements can appear more than once in the Control File. The sequence and rules for appearance are detailed below under the description of each statement. The identifying statement name must appear starting in the first character of the line and one blank space must follow the name. The remainder of the line (with continuations as necessary) may have one of two formats:

1. A series of parameters, each separated from the next by a comma, with no comma after the final parameter. In this format, the parameters **MUST** be specified in the order listed in the descriptions and the comma following any omitted parameter must be included.
2. A series of parameter specifications in the form *'keyword=parameter'* each (except the last one) followed by a comma. The appropriate keywords are spelled out for each type of statement in the descriptions. When using this format, parameters

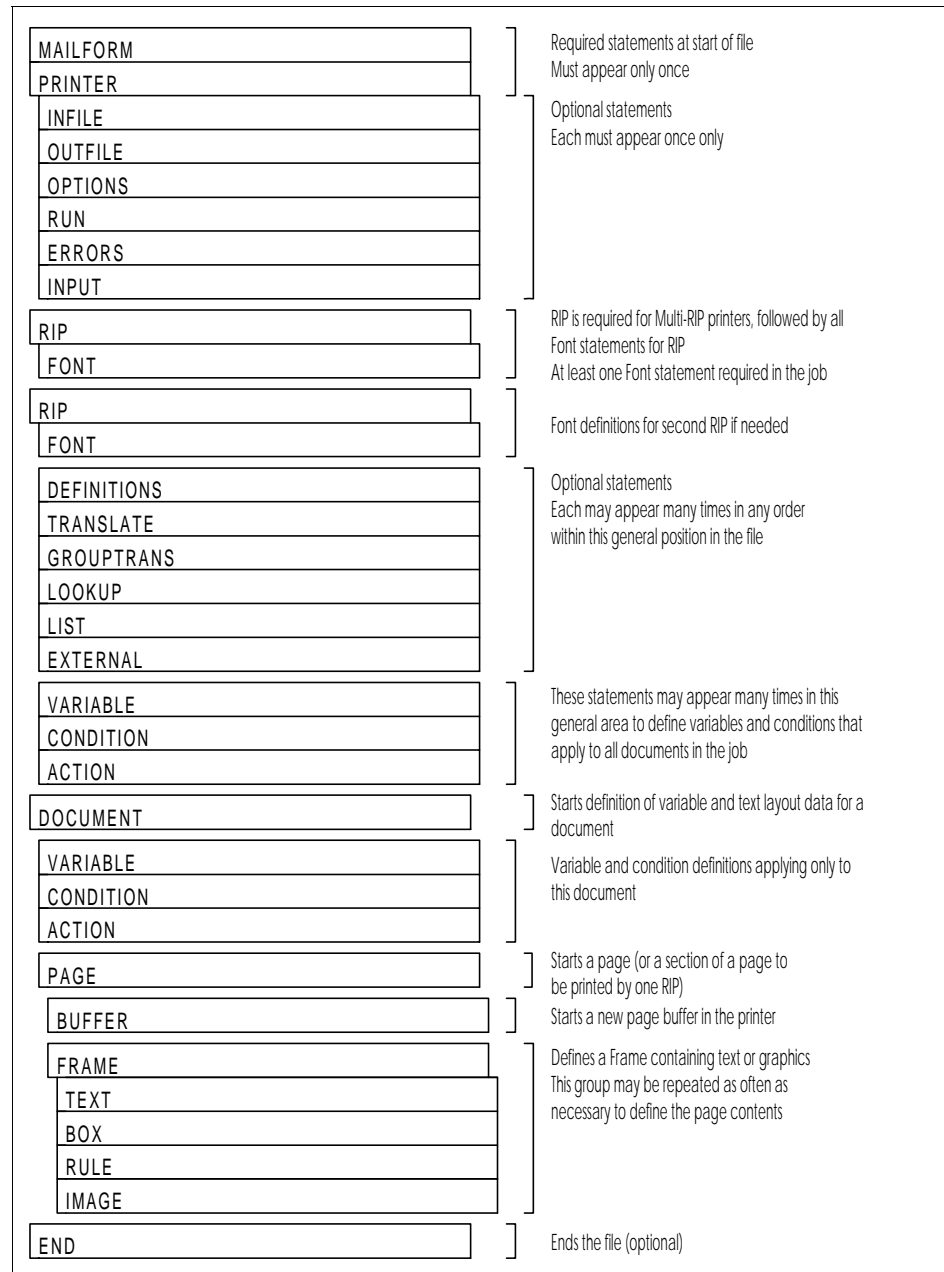
may be entered in any order and any may be omitted.

It is not allowable to mix the two formats in one statement, but separate statements may use either format. Continuation lines may begin after any comma that separates two parameters and must have at least one blank character at the start. Some statements (such as the TEXT statement) do not follow these rules - exceptions will be noted below.

Comment lines may be inserted at any point (even between statement continuation lines) and are identified by an asterisk (*) as the first character. The remainder of the comment line is ignored and may contain any desired characters.

Statement Order

The previous list of statement types generally defines the required order of MailForm statements in the Control File. The following diagram illustrates statement ordering:



5

Control File Statement Descriptions

MAILFORM Statement

The MAILFORM statement must appear as the first statement of the file, and there may only be one in each file. It is used to identify the file as a MailForm Control File and to identify the job during MailForm processing.

Parameters following the identifier are:

1. Job ID up to 8 alphanumeric characters, the first of which must be alphabetic, which is used to identify the job in the production process
2. Jobname a variable-length alphanumeric string identifying the job.
3. Date an 8-character group in the format MM/DD/YY or DD/MM/YY

Example:

```
MAILFORM J000123,Direct Mail Job,07/29/96
```

PRINTER Statement

The Printer statement must immediately follow the MAILFORM statement and there may be only one statement of this type in the file. The Printer statement serves to identify the type(s) of output printer device for which the job will be generated and, if multiple printer systems are to be used, how they are organized.

Parameters following the identifier are:

1. **Manufacturer** name, in capital letters, of the printer manufacturer selected from the list in Appendix A.
2. **Model** alphanumeric model designation of the target printer also from the list in Appendix A.

For Scitex Ink-Jet printers, if the target printer system has a Multi-RIP organization, the model designation should be replaced with the parameter:

$RIPS=n$

where n is the number of RIPs. The specific printer types for each RIP are specified in the RIP statement.

- 3-n **Printer-dependent parameters** in the form *keyword=value*.

Scitex Printers:

$SYSTEMS=n$

for multiple synchronized printer set-ups, the number of linked printers. This will correspond with the number of output printer files to be written.

Example:

```
PRINTER SCITEX,3600,SYSTEMS=2
```

INFILE Statement

The INFILE statement is optional. If used, it must appear after the MAILFORM and PRINTER statements and before the RIP and/or FONT statements. If Variable Input file(s) are on disk, it supplies the path to those file(s). If the Variable Input is from tape, the INFILE statement supplies the Data Set Name of the file(s) - the actual tape unit numbers are provided by the system operator. Up to 10 Variable Input files may be read from simultaneously on one job. With disk files, MailForm can also read the file backwards if required.

Parameters following the identifier are (one set for each file):

1. File Type Either D or T indicating the file is from *Disk* or *Tape* respectively. Default is Disk.
2. File Path/
Data Set
Name Full file path to a disk-resident file or the Data Set Name from the tape header label
3. Starting
Record
Number (optional) a number indicating the offset (in records) to the first record to be read from the file. Default is zero. This offset may be positive or negative.
4. Read Direction (optional) F or B meaning read *Forward* or *Backward* respectively. Default is F.

Each of the above parameters must be followed by a comma, except the last parameter for the last file specifier. Items 1 and 2 must be supplied for each file. Items 3 and/or 4 may be omitted and the commas do *not* have to be supplied for the missing operands. If the INFILE statement is not supplied, the system operator will be asked to supply the information.

Note The Starting Record Number parameter may be used to perform various set-ups that otherwise would require preprocessing of the Variable Input File:

“North/South Split”

This is a page format where one half of the file is printed in the left column and the second half in the right column, for example (with MailForm, up to 10 columns are possible). To achieve this, use an arrangement like:

```
INFILE D,filename,0,F,D,filename,5000,F
* where filename is the same file path for both
* and 5000 is half the number of records in the
* file
.
* variables for the left side
VARIABLE ...
VARIABLE ...
READ:2          * reads from the second 'file'
* variables for the right side
VARIABLE ...
VARIABLE ...
```

MailForm always reads from the first INFILE ‘file’ at the start of a document. The READ statement causes it to read again from the second INFILE ‘file’ - in this case an offset pointer in the same disk file.

Front and Back Offsetting

Where you want to print front and back from two Scitex stations that are offset up and down the web, whether these are different RIPs of a multi-RIP system or multiple Systems linked via a Synchronizer, you must offset the front and back data by the number of documents covering the distance between print stations so that the correct back side data prints together with its front side. We do this in MailForm by setting up the INFILE to read from two pointers in the same file thus:

```
INFILE D,filename,0,F,D,filename,-10,F
* where filename is the same filepath for both
* and 10 is the number of documents between
* Scitex print stations along the web
.
* variables for the front side
VARIABLE ...
VARIABLE ...
READ:2          * reads again for the back
* variables for the back side
VARIABLE ...
VARIABLE ...
```

When the INFILE Start Record Number is negative, MailForm does not actually read - it creates a dummy blank record and counts it. When the counter goes to zero, it does a real read from the file. We therefore generate the back side 10 records offset from the front, which causes the station printing the back to print document 1 just as the front document 1 gets to it.

The MailForm INFILE Statement is a much more convenient way to achieve these two effects, because no preformatting of the input file is necessary, and it is easier to change the setup when necessary.

OUTFILE Statement

The OUTFILE statement is optional. If used, it must appear after the MAILFORM and PRINTER statements and before the RIP and/or FONT statements. If one or more of the Output Imager files is to be generated on disk, it supplies the path to those files. If the Imager Output is to be written to tape, the OUTFILE statement supplies the Data Set Names of the files - the actual tape unit numbers are provided by the system operator.

Parameters following the identifier are:

- | | |
|-----------------------------------|---|
| 1. File Type | Either D or T indicating the file is from <i>Disk</i> or <i>Tape</i> respectively. Default is Disk. |
| 2. File Path/
Data Set
Name | Full file path to a disk-resident file or the Data Set Name from the tape header label |
| 3. Rewrite
Indicator | (optional) the letter R if the user wants MailForm to overwrite any existing file with the same filename. |

The above two (or three) parameters should be repeated as many times as necessary to identify all the output files for this job - i.e. if the PRINTER statement specifies 'SYSTEMS=3', three sets of File Type/Path/DataSetName should be included on the OUTFILE statement. (The statement may be continued on as many lines as necessary - start a new line after the comma between parameters and ensure that the first character of the continuation line is blank).

If the OUTFILE statement is not supplied, the system operator will be asked to supply the information.

OPTIONS Statement

The Options Statement allows the user to control and vary some optional parameters about this job. The Options statement is not required in the Control File - it may be included when one or more parameters need to be changed from the default values.

Parameters following the identifier are:

1. Measuring Units
Specifies the measuring units to be used for the job. All dimensional values will be converted from these units to dots of the printer resolution. Possible values are:
INCH inches
CM centimeters

POINT points (1/72")
DOT printer dots (default)

Keyword: UNIT=*parameter*

2. Cueing Option
Indicates how pages will be cued (registered with the web by use of an optical sensor). Possible values are:
FIRST only the first page of a document is cued (default)
EVERY every page of a document is individually cued
NONE do not cue any pages

Keyword: CUE=*parameter*

3. Variable Inputting
Indicates whether MailForm should read a new record automatically at the start of a document or if the user will include explicit READ commands where required.
Values are:

YES read record for each document (default)
NO only read at READ command

Keyword: *READ=parameter*

4. Document Skipping

When documents are skipped where a SKIP command is entered in the logic, normally any remaining variables for the skipped document are generated if they are "Inheritable" (see the Variable Statement for more information) or if an Exit Routine is used in their generation. This option can force MailView to skip generation of these variables to improve processing speed.

Possible values are:

NO do not skip (default)

ALL skip all remaining variables

Keyword: *SKIPVG=parameter*

5. Justify Spacing

a number indicating the percentage of the standard space size that the smallest space size may be reduced to when in Justify mode (where all lines are spaced to fit the width of the frame). The default is 50%.

Keyword: *MINSP=parameter*

6. Generated Spaces

Maximum size of generated spaces. MailForm creates space characters into unused character positions in each font to help position text in the line. The default is 64 dots. If the job contains very long lines and flush

right or large tab excursions, a larger value may reduce the amount of output data.

Keyword: GENSP=*parameter*

RUN Statement

The RUN Statement specifies parameters specific to this particular execution of this job.

Parameters following the identifier are:

1. Starting Document a number (max 999999) indicating the document number with which this run is to start. All documents up to this number are processed, but not written to the output file. If omitted, the default is 1.

Keyword: *START=number*

2. Documents per Reel a number representing the maximum number of documents to write to each output tape reel. If omitted, the tape reels are filled to their capacity.

Keyword: *REEL=number*

3. Stop Count a number (max 999999) indicating that the system is to generate a Stop Code causing the printer to stop after every group of this number of documents.

Keyword: *STOP=number*

4. Number of Documents a number (max 999999) giving the number of documents to be printed on this run. MailForm will stop generating documents on the output file after this many documents are processed.

Keyword: *COUNT=number*

5. Start a number (max 999999) giving the

Document Counter	number of the starting document printed on this run. If omitted, the default is to start at document 1. Keyword: DOCNUM= <i>number</i>
6. Restart	the word "RESTART" will cause MailForm to attempt to restart the job from a Checkpoint File written by a previous run of this job. This parameter is not required for a restart - the computer operator on the server system can use a menu item to restart a job without altering the Control File.
7. Test Run	either the word "SELECTED" or "ALL" indicating whether selected documents or all documents should be output. If omitted, the run is assumed to be a normal production run, not a test. If included, parameter #4 (Number of Documents - COUNT) must also be specified. Keyword: TEST= <i>parameter</i>
8. Split Job Output	a number representing the maximum number of documents to be written to output files generated in this run. When this option is used, the Filename(s) specified on the OUTFILE statement (or the Data Set Name(s) for tape output) should end in a series of numeric digits - for example: AMFP0001.MPC or Jones.Inc.2.25.0001.MPC

MailForm generates the first output to the filename (or Data Set Name) specified, then adds one to the numeric digits at the end of the name to create the second filename, etc. Thus AMFP0001.MPC continues with AMFP0002.MPC. Should the name not include numeric digits at the end, alpha characters are incremented to the next alphabetic value, with a carry to the next digit after incrementing 'Z' to 'A'. Other ASCII characters increment to the next legitimate value.

Each output file is a self-contained job, complete with fonts and fixed-file data at the start and an End Job command at the end. IJPDS requires that document numbers start at one at the beginning of a job, therefore the document number *written to the output* is reset at the start of each file, but the internal Document Number is not reset.

You can use the ACTION FEOV statement to force the end of a disk file when this option is in effect (otherwise FEOV has no effect with disk output).

Keyword: JOBSPLIT=*parameter*

ERRORS Statement

The Errors Statement allows the user to change actions taken when or before errors are detected.

Parameters following the identifier are:

1. Checkpoint
Frequency

This option controls how often MailForm saves checkpoint data, expressed in number of documents. If omitted, MailForm does not take checkpoints.

Keyword: CHECKPT=*number*
2. Maximum
Error Count

This option controls the number of errors listed in the MailForm Log file. The default is 20.

Keyword: MAXERRS=*number*
3. Message
Control

This option controls the type of messages included in the MailForm Log file. Possible values are:

ALL	All messages included (default)
WARN	Error and Warning messages
ERROR	Only Error messages

Keyword: MSGS=*parameter*
4. Duplicate
Error
Control

This option controls whether MailForm shows duplicate error messages. Possible values are:

SUPPRESS	Show errors once only
SHOW	Show all errors (default)

Keyword: DUPERR=*parameter*
5. Error Action

This option controls how MailForm

Control

proceeds after an Error is encountered.

Possible values are:

STOP Stop after MAXERRS messages (default)

NO No action

CHECK Stop the job at the next Checkpoint

Keyword: *ERRACT=parameter*

INPUT Statement

The INPUT statement defines and describes the variable input file from which the personalization data is to be extracted to form each individual document.

Parameters following the identifier are:

1. File Type F or V meaning Fixed or Variable-length blocks respectively. This parameter may also have the value NONE, meaning no variable input file is to be used. In this latter case, no other parameters are needed on the INPUT statement. If omitted, the default is Fixed.

Keyword: TYPE=

2. Record Size a number giving the size of the record in bytes. No default, must be supplied.

Keyword: RECSIZ=

3. Block Size a number giving the size of tape blocks in bytes - should be a multiple of the record size. Default is same as the record size.

Keyword: BLKSIZ=

4. Start Column 0 or 1 - the first byte of the record is counted as zero or one. Default is zero.

Keyword: STARTC=

5-n Field Specs for each field in the record, the following group of items may be entered, each separated from the next by a colon (:). These items are only

necessary if using field names instead of location and length values in subsequent Variable definitions:

- Field Starting Position
 - Field Length in bytes
 - Field Name - up to 30 alphanumeric characters
 - Field Type - a letter indicating the expected content format
- C = Character data
B = Binary data
P = Packed Decimal data
Z = Zoned Decimal data

Any of the above items may be omitted from each Field Specification entry, but the colon character must be included if a later item is supplied. The default values that will be used when items are omitted are:

1:1:FIELD:C

The Input Statement may be continued on as many lines as necessary by ending a line after any comma and starting continuation lines with at least one blank character.

Example:

```
INPUT F,112,1120,1,
      1:30:NAME,
      31:30:ADDRESS LINE 1,
      61:30:ADDRESS LINE 2,
      91:15:CITY,
      106:2:STATE,
      108:5:ZIP
```

RIP Statement

The RIP Statement is used to introduce the fonts required for each of the RIPs used in the current job. In a multi-RIP job, a RIP statement must immediately precede the first FONT statement for each RIP. In a single-RIP job, the RIP statements are not required and may be omitted.

Parameters following the identifier are:

1. **RIP Number** a number indicating which RIP is being specified. For 3000-series systems this is 1 or 2. For 6000-series systems this may be 1 - 8.

Keyword: RIPNUM=

2. **Printer System Type** the ID number of the printer system (either 3000, 3600 or 6240) operated by this RIP.

Keyword: TYPE=

The following parameters are required only for 3600 and 6240 RIPs:

3. **Total number of jets** The maximum number of jets controlled by this RIP. Default values are:

3000:	1600
3600:	4096
6240:	1024

Keyword: JETS=

4. **Number of heads** Number of heads actually used in this job in this RIP. It is assumed that heads are assigned left to right when looking in the web direction. (Default=1)

Keyword: HEADS=

5. Printhead Offsets A series of numbers (one for each head counted in parameter 4) that indicates the position in dots of the start of each printhead measured from the left edge of the web (or from the left edge of the leftmost printhead). These values allow you to place frames in actual positions in the document text and have MailForm reposition them to the necessary buffer positions to address the appropriate printheads. Default values are:

3000:	0
3600:	0,1024,2048,3072
6240:	0

Keyword: OFFSETS=

6. Printhead Sizes A series of numbers (one for each head counted in parameter 4) that indicate the number of jets in each printhead. Default values are:

3000:	1600
3600:	1024,1024,1024,1024
6240:	1024

Keyword: SIZES=

If parameter 6 (SIZES) values are to be specified, parameter 5 (OFFSETS) values should also be supplied.

Each RIP statement must be immediately followed by all the FONT statements for fonts to be used on this RIP in this job.

FONT Statement

A Font Statement must be included for each font to be used in each RIP in the current job. The first two parameters are required on every Font Statement. It is not necessary with MailForm to include a Font statement for each rotation of each font. If the rotation is not specified, MailForm will scan the text of all documents and will define the rotations required for all frames that use each font. Parameters following the identifier are:

1. Font ID a number, 0-255 by which this font will be identified in this job.

Keyword: ID=*number*

2. Font Name the eight-character font name.

Keyword: NAME=*name*

The following parameters are optional and may be omitted:

3. Orientation a letter or number representing the orientation of the page thus:
U (or 0) - Upright
R (or 1) - Head to the right
I (or 2) - Inverted
L (or 3) - Head to the left
If omitted, MailForm will scan the document text for FRAME statements and Set Font commands and will determine what orientations of each font are required.

Keyword: ROT=*parameter*

4. Default InterLine Space a number representing the additional dot rows of Inter-Line Space to be added below this font if no explicit InterLine Space (or Line Spacing) command or parameter is given in the document text.

5. Font Adjustments	any or all of:
	T= <i>nn</i> add or subtract dot rows at top of matrix
	B= <i>nn</i> add or subtract dot rows at bottom
	L= <i>nn</i> add or subtract dots at the left side of all characters in the font
	R= <i>nn</i> add or subtract dots at the right side of all characters in the font

the value *nn* may be positive or negative (starting with -) to add or subtract dots respectively. Between each adjustment parameter and the next should be a slash (/).

Keyword: SIZE=*parameter*

NOTE: Font Adjustments are not yet implemented.

As many font statements as are necessary should be included as a group, with no other type of statement between them.

Example:

```
FONT 3,AUNT4000,U,2,T-2/B3
```

The above example removes two dot rows at the top and adds three rows at the bottom of all characters.

DEFINITIONS

Statement

There may be several Definitions statements in the file. Each defines additions or subtractions to a character group or processing sequences for various built-in procedures. There should be a separate Definition Statement for each parameter type.

The possible Definition parameters are:

ALPHA	These define additions and/or subtractions to/from the standard default sets. ALPHA, for example, commonly consists of uppercase and lowercase alphabetic characters only. For European and other non-English language use, the ALPHA set may be augmented with accented characters. (Since ALPHA and ALNUM are actually combinations of UPR, LWR and NUM, if additions and/or subtractions are made to either ALPHA or ALNUM, the changes will actually be made to the UPR set. If you wish to use UPR and LWR independently, make changes to UPR and LWR which will automatically also change ALPHA and ALNUM also.)
NUM	
ALNUM	
UPR	
LWR	
PUNCT	
MATH	
SPC	

Standard definitions of these classifications are:

UPR	Uppercase: A-Z
LWR	Lowercase: a-z
ALPHA	UPR plus NUM
NUM	Numeric: 0-9
ALNUM	ALPHA plus NUM
PNCT	Punctuation: .,:;-'"?/()[]{}_*&%\$#@!
MATH	+ = / * -
SPC	Any whitespace character
XDGT	Hexadecimal digit: 0-9 or A-F
PRNT	Printable Character

(ALPHA+NUM+PNCT+MATH)

Example:

ALPHA , xx , xx , xx , xx , xx , -xx , -xx

where - xx is a hexadecimal character value of a character to be added to the group and -xx is a hexadecimal character value of a character to be subtracted from the group.

NAME This defines the processing sequence for the "Name Defaults" built-in procedure (P15). This procedure determines the length of a name having multiple parts and, if the length exceeds a supplied limit value, proceeds to shorten the name by dropping various elements until the length is reduced below the limit. Several sequences may be defined and referenced by number for different results in different areas of the same job. The element codes are:

TM	Title if 'Mrs'
TS	Title if 'Ms' or 'Miss'
TR	Title if 'Mr'
TD	Title if 'Dr'
TT	General Title
TB	Title if Blank
FN	First Name
FI	First Initial
FB	First Name Blank
MN	Middle Name
MI	Middle Initial
MB	Middle Name Blank
LN	Last Name
LT	Last Name Truncated
SA	Academic Suffix

SP	Professional Suffix
SM	Maturity Suffix
SF	General Suffix
SB	Suffix Blank

The data is supplied to the procedure in a single field with colons (:) between each element - i.e. Title:First:Middle>Last:Suffix.

The Definitions statement is used to control how the procedure operates when dropping or shortening elements to fit the width value. The Definition sequence takes the form:

NAME,nn,(ifclause)code:code:...code,
(ifclause)code:...code

where:

NAME	identifies this definition statement as defining a 'Name Default' procedure
nn	is the number of this NAME definition table

the data after each subsequent comma is a set of codes describing the arrangement of name elements to be tested for length. The (ifclause) in parentheses is optional. Using the same two-letter codes listed above, various elements may be tested and if all are true, this arrangement of name elements will be tested for length - otherwise the system will skip to the next one in sequence. Within the parentheses, each two-letter code is separated from the next by either & or | meaning logical AND and logical OR respectively. Logical AND is grouped within logical OR so:

(TB&SB | FB)

would be true if both the Title and Suffix were blank OR the first name was blank.

After any (ifclause) is a series of two-letter name element codes separated by colons and ending with the next comma or the end of the statement. Should there be more name element arrangement groups than will fit on one statement line, the statement can be continued on a following line by ending after the comma and starting the next line with at least one blank.

Each sequence of codes between commas defines a group of name elements to be tested for length. If the total width exceeds the limits, a name consisting of the elements indicated by the next sequence of codes is to be constructed and tested for width.

Example: NAME ,nn ,TT : FN : MN : LN : SF ,
 TT : FN : MN : LN , TT : FN : MI : LN ,
 TM : FN : LN , FN : MI : LN ,
 FN : LN , FI : LN , LT

Under the above definition, first the suffix is dropped, then the middle name is changed to an initial, then the title is kept if 'Mrs' and the middle initial dropped, then the title is dropped but the middle initial retained, then the middle initial dropped, then the first name initialized and finally the last name truncated. Note that not all elements need be used and the sequence is up to the user except that generally the groups of coded elements should tend toward shorter and shorter names. The second parameter 'nn' is a number, 0-31, identifying the defined sequence, and allowing selection of this sequence by the call to the processing routine.

When the two-letter element codes are used in the (ifclause) section they set a TRUE result under the following conditions:

TM	Title element = 'MRS'
TS	Title = 'MISS' or 'MS'
TR	Title = 'MR'
TD	Title = 'DR'
TT	Title is non-blank
TB	Title is blank
FN	First Name element is non-blank
FI	First Name is one letter
FB	First Name is blank
MN	Middle Name element is non-blank
MI	Middle Name is one letter
MB	Middle Name is blank
LN	Last Name element is non-blank
SA	Suffix = 'BA', 'MA', 'PHD', 'JD', 'LLD', 'BS', 'DD', 'MBA', 'MS', 'MSC', 'BSC'
SP	Suffix = 'MD', 'DDS', 'DVM', 'ESQ', 'DC'
SM	Suffix = 'SR', 'JR', 'III', 'IV'
SF	Suffix is non-blank
SB	Suffix is blank

When comparisons are done to determine if an item fits a category, casing (upper/lowercase) and punctuation are ignored.

The procedure tests each name element arrangement in sequence, first testing the (ifclause) for applicability and then testing the result for overall width in the font specified in the procedure call. When an arrangement is found that is applicable and also fits, the formatted name is returned in place of the data sent to the procedure. This procedure will ensure one blank between each name element

and the next and one period following any initial letter.

NOTE: The Name Defaults Procedure is currently not implemented.

FONT This defines the 'Font Defaults' processing sequence for the built-in (P13 and P14) procedures.

Example: `FONT,nn,rr,fontid,...,fontid`

where the procedure determines the width of the field contents in each font in turn until one is found such that the total width is less than the limit value. The second parameter '*nn*' is a number, 0-31, identifying the defined sequence, and allowing selection of this sequence by the call to the processing routine. The third parameter, '*rr*' is the number of the RIP in which this sequence will be used.

TRANSLATE Statement

The Translate statement defines a table used to convert characters to other characters - to force them to address different font positions than they would normally access. All translate tables begin with a background setup that converts all characters without changing them. The definition of each table should therefore include only those areas where characters need to be changed from the original value to a different one. Parameters following the identifier are:

1. Translate ID a number, 0 or 5-31, by which this table will be identified in this job.
2. Specifications a series of translate specifications which define the table contents.

These may take the forms:

ASCII When ASCII is specified, the table is set to convert ASCII to EBCDIC. If this specification is used, it should be the first in the set.

XX=YY where XX and YY are hexadecimal values and indicate that any character XX must be converted to YY when this table is in use.

XX=YY-ZZ where XX,YY and ZZ are hexadecimal values and this construction indicates that XX becomes YY, XX+1 becomes YY+1 ... XX+n becomes ZZ.

'a'='b' where a and b are character values and indicate that any character a must be converted to b when the table is in use.

Example:

```
TRANSLATE 5,81=C1-C9,91=D1-D9,A2=D2-D9
```

MailForm provides four standard translate tables

normally used to convert all uppercase text input into normal and bold upper and lowercase. The tables are:

Table 1	No change - capitals to capitals, lowercase to lowercase.
Table 2	Upper normal to lowercase normal - no change for any other characters.
Table 3	Normal to bold - normal upper, lowercase and punctuation characters to the equivalent bold font positions.
Table 4	Same as table 3, but also changes normal capitals to bold lowercase.

Tables 1 and 3 are used with fonts that contain both roman and bold character sets. Tables 2 and 4 may also be used when text in the MailForm Control File (or in the variable input) needs to be converted from capitals to lowercase.

Table 0 is a special table used to translate the entire contents of each input record before any further processing takes place. This only happens when the user defines Table 0. This can be useful when the variable input file is in some character set other than EBCDIC (ASCII for example) or when foreign language characters are present in the file and need to be mapped to appropriate font positions.

GROUPTRANS Statement

The Grouptrans Statement is used to define tables that convert groups of one or more characters into other groups of zero or more characters. When this table is in effect, MailForm compares each group of characters in the input text to items in the table and, if they match, replaces them with the defined replacement character string. Note that this search continues character-by-character throughout the input text, not just at the start of a variable character group as the Lookup Statement does. Once a character group is matched and replaced, none of the replacement characters are re-searched for new matches - the search continues immediately following the final replacement character. Parameters following the identifier are:

1. GroupTrans a number, 0-31, by which this table
ID will be identified in this job.
2. Specifications a series of translate specifications
 which define the table contents.

These take the form:

'*ab*' = '*cde*' where *ab* and *cde* are character values and indicate that any instance of the character group '*ab*' will be converted to '*cde*' whenever this table is applied to a text string. The conversion may be applied more than once in any one text string if the character group being matched appears more than once in the string.

Example:

```
GROUPTRANS 8, 'Mister' = 'Mr.'
```

LOOKUP Statement

The Lookup Statement defines a lookup substitution table. When this table is in effect, MailForm compares the entire contents of a variable with each search string and, if it matches, replaces the entire variable contents with the replacement string. There is also an optional DEFAULT value which will replace the variable contents of any variable that does not match any search string in the table. Note that this differs from the Grouptrans table in that Lookup must match the whole variable contents whereas Grouptrans matches any character group within the variable. Parameters following the identifier are:

1. Table ID a number, 0-31, by which this lookup table will be identified in this job.
2. Specifications a series of replacement specifications which define the table contents.

These are in the form:

'abc' = 'defg', where *abc* and *defg* are character values, both enclosed in single quotes. When this table is used, if the entire character string being processed matches *'abc'* then it is replaced by *'defg'*. As many specifications as desired may be included in one table. If the single-quote character is required as part of the search or replacement argument, two quotes (' ') together will be used as a single quote.

One special replacement specifier that may be present once in any one table reads
DEFAULT=*'xyz'*, where *xyz* are character values. This specifier indicates that, if the entire character string being processed does not match any other specification in this table, then it should be replaced with *'xyz'*. The DEFAULT specifier may appear at any point in the list of specifiers.

Example:

```
LOOKUP 4, 'NY'='New York',  
        'LA'='Los Angeles',  
        'DC'='Washington', DEFAULT='Chicago'
```

LIST Statement

This statement may be used to define and name a list of character strings. The list may be used in comparisons to test whether the contents of a variable matches any item in the list. This provides an alternative to a whole series of explicit conditional tests. Parameters following the identifier are:

1. Table ID a number, 0-31, by which this list table will be identified in this job.
2. Specifications a series of character strings enclosed in single quotes ('abc'). When this list table is used in a comparison, the result is set TRUE if the item being compared matches any of the character strings in the list (to match, the lengths must also be equal). If there is no match, the result is set FALSE. As many character strings as desired may be included in one table. If the single-quote character is required as part of the search or replacement argument, two quotes (' ') together will be used as a single quote. A character string may also be specified in hexadecimal using X' . . . '

Example:

```
LIST 3, 'Mr. ', 'Mrs. ', 'Ms '
```

EXTERNAL Statement

The External statement specifies the linking of a user-supplied program segment which MailForm will call when specified to perform special additional processing of the variable input data. There are two basic types of Externals:

- A. Routines called to perform additional processing on the contents of a Variable. These routines are included by specifying Process IDs from 0 to 31.
- B. Routines called at specific points in the MailForm processing sequence to allow custom processing options. These routines are included by specifying Process IDs from 32 to 40. The points in the MailForm process that these routines gain access are listed below and more information is provided in Appendix C.

Parameters following the identifier are:

1. Process ID a number, 0-40, by which this external process will be identified throughout this job. This parameter may not be omitted.

Keyword: ID=*number*

2. Process Name the filename of the executable file for this external procedure. An extension of .EXE will be assumed if none is given. This parameter may not be omitted.

Keyword: NAME=*name*

3. Buffer Size this optional parameter may be used to ensure that MailForm allocates an adequate size common buffer for data transfer between the main program and the External

procedure. Normally MailForm ensures an area twice the length of the longest variable or, when Externals 33, 34 or 35 are used, twice the input record size. If a larger area is needed, it may be specified (in bytes) in this parameter. MailForm only uses this number if it is larger than the other criteria listed above.

Keyword: BUFFER=*number*

External procedures must be written to conform with specific rules to allow proper linking and execution with MailForm software. See Appendix C for information on how to develop exit routines for use with MailForm.

Example:

```
EXTERNAL 3, BARCODE.EXE
```

MailForm processing points at which type B externals are called are as follows:

Process ID	Processing point
32	When this EXTERNAL statement is processed
33	When a variable input record is required (user input opportunity)
34	After a variable input record is read
35	Before processing of variables starts
36	After all variables have been generated
37	At End Volume on the Variable Input file
38	At End Volume on the Printer Output file
39	At End of Job
40	Before each Checkpoint Record is written

VARIABLE Statement

The Variable statement is used to define variable items and their contents. This statement may occupy a number of lines and does not strictly conform to the parameter format used by other statements. Parameters following the identifier are:

1. ID Number a number, 0-9999 by which this variable will be identified within this job.
2. Inheritable if this Variable is to be 'inheritable' (i.e. its contents are not to be erased at the start of each document and its value can be passed from one document to the next) then this parameter should consist of the letter 'I'. Otherwise the parameter may be blank or omitted.
3. Inhibit Float if, when this variable is used on a line by itself (with no other text characters) and the variable is empty, the user does *not* wish the line to be omitted and following lines to 'Float' up into the area normally occupied by this variable, then this parameter should be the letter 'N'. Otherwise the parameter may be blank or omitted.

Following the last of the above parameters *must* be one or more spaces followed by the specification of the contents of the variable. The content specification may be complete on the same line as the start of the Variable statement or may be continued on several lines. Each statement element must be complete on one line, however, (including the comma between elements) and continuation lines must begin with at least one blank character.

The Variable content specification takes the general

form:

```
[ if clause ] content clause  
  [ else if clause content clause ]  
  [ else clause content clause ]
```

Items above in square brackets are optional. Thus a simple variable content may consist only of:

content clause

or may expand to the more complex:

```
IF( condition clause ) content clause  
  ELSEIF( condition clause ) content clause  
  ELSEIF( condition clause ) content clause  
  ELSE content clause
```

The above example defines a variable which has four possible contents:

1. If the condition clause on the first line is true, the contents are defined by the content clause on that line.
2. If the condition clause on the first line is false and the one on the second line is true, the contents are defined on the second line.
3. The third line defines the contents if the condition clauses on the first and second line are both false and that on the third line is true.
4. If none of the previous condition clauses are true, the contents of the variable are defined by the content clause on the last (ELSE) line.

The IF, ELSEIF and ELSE clauses must begin on a new line (with at least one blank before them to indicate a continuation of the Variable Statement).

CONDITION CLAUSES

The Condition clause in a Variable Statement (and in the Condition Statement and Action Statement described later) occurs immediately after the word(s) IF and ELSEIF and is enclosed in parentheses (). The clause may consist of one or more Condition Numbers or Condition Expressions, each separated from the next by Logical AND (&) or Logical OR (|) symbols, or the clause may consist of a single Status Word. In evaluating the clause, all Logical ANDs are processed before any Logical ORs. Inner sets of parentheses may be used before or after Logical AND and/or OR symbols to force a specific evaluation order.

Condition Numbers, if used, must refer to a Condition defined in a previous Condition Statement, and are numbers from 0 to 1025. They may be preceded by an exclamation or backslash (! or \) to negate the condition.

Condition Expressions consist of two Data Descriptors separated by a Comparison Operator. Data Descriptors are defined on page xx. Valid Comparison Operators are as follows:

=	Equal To
\	Not Equal To
<	Less Than
>	Greater Than
<=	Less Than or Equal To
>=	Greater Than or Equal To

The only Status Word currently defined is:

START	true only when the process is at the start of a job (i.e. on the first page).
-------	---

Examples:

```
( 23&241 | !25 )  
( 16&I123 : 5 = '00000 ' )  
( I99 : 4 \ '      ' &I99 : 4 \ '0000 ' )  
( START )
```

In Condition Clauses having several Condition

Expressions, if the first or second Data Descriptor is identical in two successive Condition Expressions, it may be omitted. Thus:

(I99:4\ ' '&I99:4\ '0000')

may also be coded:

(I99:4\ ' '&\ '0000')

Similarly:

(I241=' '|I242=' '|I243=' ')

may also be coded:

(I241=' '|I242=|I243=)

VARIABLE Content Clause

The Variable Content clause consists of one or more Data Descriptors which define an item of character data. Data Descriptors are also used in Condition Statements and Condition Clauses of Variable Statements. Several Data Descriptors may be grouped together with enclosing parentheses () in order to apply a series of Data Modifiers to the whole group at one time.

Data Descriptors

A Data Descriptor defines a character or group of characters, or a procedure or series of procedures that will result in obtaining a group of zero or more characters. Normally, in Variable and Condition statements, Data Descriptors are used individually and may be separated by commas. Several Data Descriptors in a Variable Content Clause produce output that is concatenated together to form the Variable Content.

The types of Data Descriptors are as follows:

Constant Descriptor

The Constant Descriptor consists of one or more characters enclosed in single-quotes ('). These characters are copied to form part of the variable contents. The Constant Descriptor may also describe characters in hexadecimal format by preceding the first quote with a capital X.

Examples:

```
'Dear Mr. '  
X'23BFC0'
```

Certain Text Commands (see the LINE statement) may also be entered as part of a Constant Descriptor by enclosing the command characters in double-quotes (") and putting a comma between commands.

Examples:

"F03,H240"
"A480"

Type Descriptors

These descriptors are used only as the objects of a comparison in a **CONDITION** statement. They are used to determine the classification of input characters. The available Type Descriptors are:

ALPHA	Alphabetic: A-Z or a-z
NUM	Numeric: 0-9
ALNUM	ALPHA plus NUM
UPR	Uppercase: A-Z
LWR	Lowercase: a-z
PNCT	Punctuation: .,:;"'"/()[]*_&%\$#@!
MATH	+ = / * -
SPC	Any whitespace character
XDGT	Hexadecimal digit: 0-9 or A-F
PRNT	Printable Character(s) (ALPHA+NUM+PNCT+MATH)

Many of the above classifications may be altered from the default definitions shown above by using a **DEFINITION** statement. The default sets may also be changed at installation time.

List Descriptor

This descriptor is used only as the object of a comparison. The only legal operators in this comparison are = and \ (equal and not equal). The descriptor is coded as the letter 'L' followed by the number of the LIST table defined in a previous LIST statement. If the operator is =, the condition is TRUE if one of the list items exactly matches the other comparand, otherwise the condition is false. If the operator is \, the condition is TRUE when no match is found. When the comparand is the contents of a Variable, the entire contents must be identical to one of the strings in the list.

Example:

```
LIST 3, 'ME', 'NH', 'VT', 'MA', 'RI', 'CT'  
VAR 21 IF(STATE=L3) 'New England'
```

Document Number Descriptor

The Document Number Descriptor retrieves the number of the current document into the current character group and has the form:

Dn

where: D is the letter 'D'
n is the number of digits to be generated. The output contains leading zeros or will be truncated at the left side if the significant digits are fewer or more than nn respectively.

Example:

D6

When using the JOBSPLIT option (see the RUN Statement), the Document Number is continuous from the first document on the first file to the last document on the final file (it does not get reset to 1 at the beginning of each file). To retrieve the Document Number of the current document on the current JOBSPLIT file, use the following:

DSn

Where: DS indicates the split file document number
n is the number of digits.

Input Record Number Descriptor

The Input Record Number Descriptor retrieves the number of the current record in the Variable Input File into the current character group. The descriptor has the format:

Rn

where: R is the letter 'R'
n is the number of digits to be generated. As with the document number, the result has leading zeros or is truncated according to length.

Example:

R8

Where multiple variable input files are in use, the number retrieved is the number of the record in the file from which the last record was read.

Input File Number Descriptor

The Input File Number Descriptor retrieves the number of the output file currently being written to when using the JOBSPLIT option (see the RUN Statement) into the current character group. The descriptor has the format:

F_n

where: F is the letter 'F'
 n is the number of digits to be generated. As with the document number, the result has leading zeros or is truncated according to length.

Example:

F8

When not using the JOBSPLIT option, this Descriptor will always return a value of 1.

Compute Descriptor

The Compute Descriptor defines an arithmetic expression which MailForm evaluates to produce a decimal result value. The descriptor has the form:

Cn(expression)

where: **C** is the letter 'C'
 n is a number which indicates the number of decimal digits in the resulting value
 (and) are parentheses enclosing the arithmetic expression.
 expression - contains any of the following elements and arithmetic operators:

 nnn - decimal numbers

 Vnn - the numeric contents of variable nn

 Lnn - the number of characters in variable nn

 Arithmetic operators:

 + plus

 - minus

 * multiply

 / divide

 % modulo

Multiply, divide and modulo operations are evaluated before plus and minus. Parentheses may be used to force other orders of evaluation - inner parenthesized expressions are evaluated before outer ones.

Example:

C4((V32+19)/12*100)

Input Descriptor

The Input descriptor allows the user to select a group of characters from the variable input record for inclusion in the variable contents at the point the descriptor is entered. The Input descriptor has the form:

`Iss:nn`

where: I is the letter 'I'
 ss is a number which defines an offset into the current record of the variable input file.
 : is the colon character.
 nn is a number of characters to be extracted from the referenced position.

Both ss and nn may be up to five digits long.

Example:

`I123:5` specifies that five characters be extracted starting at position 123 of the current record.

The Input Descriptor may also be encoded by name (implying both the offset into the record and the number of characters to be extracted) as defined in a previous INPUT statement.

Example:

```
INPUT F,386,386,0
  123:5:ZIP
.
.
VARIABLE ZIP
```

Reference Descriptor

This descriptor recalls some or all of the contents of a Variable into the current data group. The Descriptor has the form:

Vxx:ss:nn

where: V is the letter 'V'
xx is the number of the (previously-defined) Variable whose contents are to be recalled.

The remaining items are optional:

: is the colon character.
ss is a number which defines an offset into the contents of the referenced Variable.
: is the colon character
nn is a number of characters to be extracted from the referenced position.

Both ss and nn may be up to 5 digits long. If only xx is specified (ss and nn are omitted), the entire contents of the Variable are inserted.

Examples:

V24 specifies that the current contents of Variable 24 will be added to the end of the current Variable Contents.

V15:2:8 specifies that eight characters starting with the second character of the contents of Variable 15 will be added to the end of the current Variable Contents.

DATA MODIFIER GROUPS

Following either the Input Descriptor or the Reference Descriptor, before the comma that separates the next Data Descriptor from the current one, may be a Data Modifier Group enclosed in square brackets. The elements within this Data Modifier Group specify actions to be performed on the character group described by the previous Data Descriptor, before the result is added to the current Variable Contents. This allows the user to specify internal or external procedures that will modify the data to form the desired output. Any number of Data Modifiers may be specified in one group and the same type of Modifier may be used more than once if necessary to obtain the desired result. If a Data Modifier Group is not specified, the default is to perform procedures L and T with a blank character (thus removing all leading and trailing blanks from the character group).

The Data Modifier Group may contain any of the following items separated by commas. The order in which the items are entered in the Modifier Group is the order in which the procedures will be performed on the character group. Each invoked procedure is passed the character group resulting from processing done by all previously-invoked procedures in the Modifier Group.

- *An* (or just *n*) (where *n* is a number) - discard all characters after the first *n*. If the character group is shorter than *n* characters, there is no effect.
- *Zn* (where *n* is a number) - discard all characters before the last *n*. If the character group is shorter than *n* characters, there is no effect.
- *Bc* (where *c* is a character) - discard all characters up to and including the first occurrence of *c*. If *c* is not found, there is no effect.
- *Nc* (where *c* is a character) - discard all characters including and after the next occurrence of *c*. If *c* is not found, there is no effect.
- *Qc* (where *c* is a character) - discard all characters

-
- including and after the last occurrence of *c*. If *c* is not found, there is no effect.
- **L*c*** (where *c* is a character) - discard all *c* characters at the start of the character group. If the group does not begin with a *c* character, there is no effect.
 - **T*c*** (where *c* is a character) - discard all *c* characters at the end of the character group. If the group does not end with a *c* character, there is no effect.
 - **F*n*** (or **+*n***) (where *n* is a number) - discard the first *n* characters of the character group. If the character group is shorter than *n* characters, the result will be an empty group.
 - **W*n*** (where *n* is a number) - discard the last *n* characters of the character group. If the character group is shorter than *n* characters, the result will be an empty group.
 - **E*n*** (where *n* is the previously-defined number of an External Procedure) - invoke the External Procedure and pass the current contents of the character group as the argument.
 - **X*n*** (where *n* is the previously-defined number of a Translate Table) - translate the current contents of the character group by the specified table.
 - **G*n*** (where *n* is the previously-defined number of a GROUPTRANS Table) - translate the current contents of the character group substituting groups of characters as the table specifies.
 - **Y*n*** (where *n* is the previously-defined number of a Lookup Table) - look up the current character group in the specified table and substitute as the table specifies.
 - **P*n*[:*xx*][:*yy*]** (where *n* is the number of a MailForm internal procedure) - invoke the specified Internal Procedure and pass the current contents of the character group as the argument. Some Internal Procedures require additional parameters as indicated by *xx* and *yy*

which, if present, must be preceded by a colon. Available Internal Procedures are listed in Appendix B.

In the above items which use a numeric parameter (*n*), the value of *n* may vary from 0-999 (or 0 through the maximum table number or procedure number).

Variable Statement Example:

```
VARIABLE 10
  IF(I121:1='M' | I170:1='0') 'CAR-RT SORT **'
  ELSE 'A', (I171:5, I177:4, I191:2)[P10], 'A'
```

In the above example, Variable 10 is defined by the following 2 lines. The IF clause tests position 121 for the letter 'M' OR position 170 for a zero. If either condition is TRUE, then the constant value 'CAR-RT SORT **' is used as the value of the variable. If both of the tests in the IF clause are FALSE, the value of the variable is defined by the ELSE clause - i.e. the letter 'A' followed by a character group returned by Internal Procedure 10 after it has processed an input string consisting of five characters from position 171, four characters from position 177 and two characters from position 191. This is then followed by a second letter 'A'. This is the procedure one would use to generate a Postal Bar Code - the 'A' character would be the start and stop bar, position 171 would be the zip, position 177 the zip+4 and position 191 the delivery point digits. The IF clause is used to determine if a bar-coded zip is required.

Each Variable statement is processed by MailForm in the sequence that it occurs in the Control File. Therefore Variables that are referred to by a Reference Descriptor *must* occur before they are referenced. Variables that are not defined as Reusable should

have only one definition in the Control File - a second definition will cause an error.

CONDITION Statement

IF clauses within the Variable definition may contain the specific tests to be made or alternatively separate, numbered Conditions may be set and then referred to in the IF clause of the Variable. It is simpler to use a numbered Condition if the test is used to condition several Variables. Since the test is done where the Condition Statement occurs in the processing sequence, if the data were to be changed by a later statement, the result of the Condition test could still be referred to by number following the change. Numbered Conditions may also be used to control sections of text in the document.

The Condition Statement has a similar organization to the Variable statement. Parameters following the identifier in the Condition line are:

1. Identifier a number, 0-1025, by which this Condition will be identified within this job.

Following the above parameter may be zero or more spaces followed by a condition clause (as defined above for the Variable Statement - see page 14). The condition clause may be complete on the same line as the start of the Condition statement or may be continued on several lines. Each statement element must be complete on one line, however, (including the comma between elements) and continuation lines must begin with at least one blank character.

Example:

```
CONDITION 16 IF(I334:3='MR.' | I334:4='MRS.' |  
I334:4='MISS')
```

The above example tests positions 334-7 of the current record for a content of 'MR.' or 'MRS.' or 'MISS'. If the record contains any one of those groups

of characters at that position, Condition 16 is set TRUE and may be tested later in a Variable definition or in the text of the document. Otherwise Condition 16 is FALSE.

This example could also be expressed as:

```
CONDITION 13 IF(I334:3='MR. ' )  
CONDITION 14 IF(I334:4='MRS. ' )  
CONDITION 15 IF(I334:4='MISS' )  
CONDITION 16 IF(13|14|15)
```

In the above example the keyword 'IF' is used at the start of each condition clause. In the Condition Statement only, the 'IF' and the parentheses may be omitted.

ACTION Statement

The Action Statement is used to cause MailForm to perform an action or to output a control to cause the printer to perform an action. It has the form:

```
ACTION [ if clause ] type [ : parm ]  
      [ elseif clause type [ : parm ] ]  
      [ else clause type [ : parm ] ]
```

where:

- ACTION is the statement identifier (which, in the case of the Action Statement only, may be omitted)
- items in brackets ([]) are optional
- *if clause*, *elseif clause* and *else clause* are as defined previously for the Variable Statement
- *type* may be one of the following:
 - READ reads the next record from the Variable Input file - one optional parameter indicates the file number to read from.
 - STOP adds an output code to cause the printer to stop
 - SKIP skips the current document
 - END end the job without printing the current document
 - FEOV force end of volume on the output file
 - LIST list the text of the current document to the Log File
 - INPLIST list the input record to the MailForm Log File
 - REPEAT use the current record for the next document also
 - SIGNAL adds a code causing the printer to activate an external signal

-
- **ERROR** log an error to the MailForm Log File
 - **MESSAGE** adds output coding to cause the printer to output a message
 - **LOG** log a message to the MailForm Log File
- *parm* is an optional parameter necessary with some actions (**SIGNAL**, **LOG**, **REPEAT**, **ERROR** and **MESSAGE**). **LOG**, **ERROR** and **MESSAGE** parameters may be a text string enclosed in double-quotes (“”) or the number of a variable whose contents will be used as the message.

Examples:

```
READ:2  
ACTION IF(121) REPEAT:2  
      ELSE REPEAT:5  
IF(I42:1=' ') SKIP
```

DOCUMENT Statement

The Document Statement normally begins the section defining the text of a document. Since MailForm can conditionally generate different documents, more than one Document Statement can appear in a job. All Document Statements except one should have a condition parameter and those with condition parameters should precede the one that does not. Following each Document statement may optionally be Variable and Condition statements specific to this document only. After these (if any) should be all the Page, Buffer, Frame, Text and Graphic statements necessary to describe the contents of the document. Parameters that may follow the identifier are:

1. Condition String this takes the form:

IF(*condition*)

where: condition may be either the number of a previously-defined condition or a condition clause as previously described under Variable Condition Clauses.

2. Break Flag the word 'BREAK'. This causes MailForm to output the text of this Document but *not* to read additional variable input records before the next document is output. Variables currently built will be used for the next document, (and this one, if called out). This special document is *not* counted in the Document counter. After this Break Document is output, MailForm will select the next document from all *other* document definitions, not including this one - this ensures the break document is output only once when the break condition is true. This option is used where you want to

generate a special printed flag page in the output every so many documents (perhaps for box or bundle breaks).
NOTE: This option is currently not yet implemented

Example:

```
DOCUMENT IF(D6\'000001\'&D3=\'001\'),BREAK
```

PAGE Statement

The Page Statement begins a page in the Document. There should always be a Page Statement even if the document has only one page. Parameters following the Identifier are:

1. Vertical Size a number giving the vertical height of the page in page positioning units. The maximum value may be controlled by a printer limit.

Keyword: HEIGHT=

2. Horizontal Size a number giving the width of the page in page positioning units. This may also be limited by the printer.

Keyword: WIDTH=

3. Orientation a letter or number representing the orientation of the page thus:

U (or 0) - Upright

R (or 1) - Head to the right

I (or 2) - Inverted

L (or 3) - Head to the left

If omitted, Upright is assumed.

Keyword: ROT=

NOTE: The Orientation option is currently not yet implemented. All pages are processed as upright.

4. Printer/RIP a number indicating the printer (or RIP) that the following page should be sent to. If omitted, the first printer/RIP is assumed.

Keywords: RIP= or PTR=

Example:

PAGE 2540,2040,U,1

or

PAGE HEIGHT=2540,WIDTH=2040,RIP=1

**BUFFER
Statement**

This statement tells MailForm to start a new page buffer in the current RIP. If the current page is deeper than the printers' page buffer (normally 4096 dots or 17"), a BUFFER statement is required. Generally, the BUFFER statement should be placed before the current buffer position reaches the end of the buffer, and so that the page is split into approximately equal sections.

The BUFFER statement has one optional numeric parameter. If provided, it should indicate the bottom-most point in the current buffer, from which the next buffer contents will be measured. MailForm will ensure that the imager outputs the entire previous buffer up to the dot row indicated. If this parameter is missing, MailForm assumes that the next buffer is measured from the bottom of the bottom-most frame in the previous buffer - the next buffer positioning should then be measured from this point.

Example:

```
BUFFER 3942
```

BOUNDS Statement

The BOUNDS statement sets up a rectangular area within a page within which all subsequent text is supposed to fall. If any text is placed outside the rectangle, an error message will be logged. Bounds statements may appear anywhere within a page to reset the bounding rectangle to a new area. Parameters following the identifier are:

1. **Left Horizontal Limit** the left side of the bounding rectangle in page positioning units. This value may also be entered as HEADx (where x may be 1 through 4) representing the position of the left side of the indicated printhead.

2. **Right Horizontal Limit** the right side of the bounding rectangle in page positioning units. This value may also be entered as HEADx (where x may be 1 through 4) representing the position of the right side of the indicated printhead.

3. **Top Vertical Limit** the top edge of the bounding rectangle in page positioning units.

4. **Bottom Vertical Limit** the bottom edge of the bounding rectangle in page positioning units.

The bounds rectangle is initially set to the limits of the page buffer and defaults to those limits whenever a new BOUNDS statement is encountered. Thus a BOUNDS statement with no operands resets the bounds to the full page buffer - effectively setting the check off. A BOUNDS statement with just the first two operands sets left and right limits while keeping vertical limits at the page buffer height.

Example: `BOUNDS HEAD2,HEAD3`

The above example sets a bounding rectangle that will cause an error message if any text is placed outside of the area between the left edge of printhead 2 and the right edge of printhead 3. The top and bottom limits of the rectangle remain at the limit of the page buffer. An error message will also be logged in this case if printheads 2 and 3 are not stitched (i.e. when the RIP statement for the current RIP contains printhead offsets that indicate that there is a gap or overlap between the two printheads on the printed product).

**FRAME
Statement**

The frame statement defines an area within the page in which text or graphics will be displayed. Parameters following the identifier are:

1. Horizontal Position distance from the left page margin to the leftmost edge of the frame in page positioning units

Keyword: X=

2. Vertical Position distance from the top margin of the page to the topmost edge of the frame in page positioning units

Keyword: Y=

3. Width width of the frame along the line in page positioning units

Keyword: W=

4. Depth depth of the frame, down the frame as read, in page positioning units

Keyword: D=

5. Orientation a letter or number representing the orientation of text and graphics in the frame relative to the page. The letter/number system is the same as for parameter 3 of the PAGE Statement

Keyword: ROT=

-
- | | |
|----------------------|--|
| 6. Font ID | the number of the font to be used at the start of text in the frame

Keyword: FNT= |
| 7. Line Spacing | a number giving the spacing in page positioning units from one line to the next in this frame.

Keyword: LSP= |
| 8. Line Justify Mode | one of L,R,C or J (Left, Right, Center or Full Justify) indicating the justification mode to be used at the start of this frame - until changed by a Text Command.

Keyword: JST= |
| 9. Vertical Justify | one of T,B,C (Top, Bottom or Center Justify) indicating the position of the text lines in this frame. The default is T. If B, the text will be spaced down so that the last line is at the bottom of the frame. If C, the text will be equally spaced from the beginning and end of the frame.

Keyword: VJST= |
| 10. Condition Number | If this parameter is included, and the referenced condition is set FALSE, the entire frame and contents are omitted |

from the document.

Keyword: CND=

11. Writing Mode

one of REP, OR, XOR, or AND (or 0, 1, 2 or 3 respectively) indicating how the text is to be combined with printable dots previously written to the page buffer of the printer.

Keyword: MODE=

In all of the above parameters, except ROT, JST, VJST and MODE, the value can be a literal number, or it can be expressed in the form:

Vnn+xx

where:

V indicates that the value is to be taken from the contents of a Variable

nn is the number of the Variable

+ may be a plus sign (+) or minus sign (-) indicating that the contents of the Variable is to be modified by adding or subtracting xx from it

xx is an amount by which the Variable contents should be modified

The (+xx) or (-xx) is optional and may be omitted if inappropriate.

Note The Line Spacing parameter (LSP) sets the system into line-to-line spacing mode at the start of each frame. If you wish to use ILS mode, you should omit the LSP parameter and also issue an ILS command before the first text if you need

a different ILS than the default value specified on the FONT Statement. If the LSP value is set to zero, MailView will space lines at the height of the specified font.

Examples:

```
FRAME 240,360,960,720,0,1,24,L,T
```

```
FRAME 1.0,1.5,4,3,0,1,.1
```

```
FRAME X=240, Y=360, W=960, D=720, ROT=UP,  
FNT=1, LSP=24, JST=L, VJST=T
```

All of the above statements are equivalent. The frame is four inches wide by three inches high, with the top left corner at one inch from the left of the page and 1.5 inches from the top of the page. It is an upright frame using Font 1. Text will be justified left and aligned at the top of the frame. The second example assumes that UNITS were previously defined as INCH. Note that decimal points may be entered in operands and that extra spaces may be added after any comma.

TEXT Statement

The Text statement begins text within a Frame. The text may be broken into Text statements as is convenient to the user. Obviously there may be many Text statements within one frame. Text statements may be continued on subsequent lines by leaving the first position of the line blank. The parameters to a Text statement are commands and/or text strings.

Text Commands

Commands begin with a single character that identifies the command followed by a single operand and a comma which separates one command from the next. A comma is allowed, but not required, between a command and a following text string. The list of commands is as follows:

Code	Command	Starts Line before Code	Use in Variable Data	Operand from Variable
Add	Vertical Tab	Yes	Yes	Yes
Lddd	Left Indent	Yes		Yes
Rddd	Right Indent	Yes		Yes
Pddd	Paragraph Indent	Yes		Yes
Gddd	Hanging Indent	Yes		Yes
Sddd	Vertical Space	Yes	Yes	Yes
Jc	Line Justification	Yes		
Tddd	Horizontal Tab		Yes	Yes
Fnnn	Font Change		Yes	Yes
Ennn	End Line		Yes	
Znnn	Translate by Table		Yes	

Code	Command	Starts Line before Code	Use in Variable Data	Operand from Variable
Vnnn	Insert Variable			
Cnnn	Start Conditional Text			
Nnnn	End Conditional Text			
Uc	Underscore start/end			
Wnnn	Set Space Width			Yes
Hddd	Horizontal Space		Yes	Yes
Oddd	Set Line Spacing (Offset)	if not after F code		Yes
Iddd	Set Interline Space	if not after F code		Yes
Kcnnn	Save current position			
-	Discretionary hyphen		Yes	

Command operand types are indicated in the table as follows:

nnn	a numeric value
ddd	a decimal value with optional decimal point
c	a character value

All nnn or ddd operands may be of variable length with or without leading zeros. Operands of commands that allow decimal values are assumed to be specified in the units indicated by the UNIT parameter of any OPTIONS statement in the file, and are converted to equivalent dot values.

The first seven commands on this list can only occur at the start of an output line. When these commands are encountered, if not at the start of a line, a new line is

begun. Other commands may occur at the line start or within a line (between text strings).

Those commands indicated as “Use in Variable Data” may be included in a Constant Descriptor and may thus become part of the contents of the Variable.

In those commands that are listed as “Operand from Variable”, the parameter may be a literal number or may be of the form 'Vnnn' indicating that the contents of Variable nnn should be used as the operand. The item 'Vnnn' may also optionally be followed by a plus or minus sign and a decimal value indicating that the decimal value should be added to or subtracted from the contents of Variable nnn to form the operand of the command (example: AV32+10 - vertical tab to position given by the contents of Variable 32 plus 10 dots). When the “Operand from Variable” option is used, the computed value is assumed to be specified in dots, not in UNITS.

Text Command Descriptions

Vertical Tab	Sets the starting point for the next line to a point measured from the start of the current frame perpendicular to the direction the text lines run. The distance is specified in the current UNITS, with decimal point if necessary. If the resulting position is nearer the start of the frame than the previous vertical position, an error message will be issued and the command ignored. If this code is encountered when not at the start of a line, a new line is started.
--------------	--

Vertical Space	Sets the starting point for the next line to a point measured from the current
----------------	--

position perpendicular to the text line axis - thus adding extra space between the previous line and the current one. The distance is specified in the current UNITS, with decimal point if necessary. If this code is encountered when not at the start of a line, a new line is started.

- Horizontal Tab** Sets the position in the line that the next character will be imaged to a distance along the line axis measured from the start of the line (after any left indents have adjusted the start point). The distance is specified in the current UNITS, with decimal point if necessary. If the resulting position is nearer the start of the line than the previous character position, an error message will be issued and the command will be ignored. The Horizontal Tab command cannot be used in full-justify mode.
- Horizontal Space** Sets the position in the line that the next character will be imaged to a distance along the line axis measured from the current character position - thus creating extra space between the previous character and the next one. The distance is specified in the current UNITS, with decimal point if necessary.
- Left Indent** Sets the starting point for this and all subsequent lines in the frame along the line axis by the value specified (until changed by another command). The distance is specified in the current UNITS, with decimal point if necessary. If the resulting effective line length is zero or less, an error message will be issued and

the command is ignored. If this code is encountered when not at the start of a line, a new line is started.

- Right Indent** Sets the ending point for this and all subsequent lines in the frame along the line axis by the value specified (until changed by another command). The distance is specified in the current UNITS, with decimal point if necessary. If the resulting effective line length is zero or less, an error message will be issued and the command is ignored. If this code is encountered when not at the start of a line, a new line is started.
- Hanging Indent** Sets the starting point for all subsequent lines in the frame along the line axis by the value specified (until changed by another command). Note that this command does not affect the length of the current line. The distance is specified in the current UNITS, with decimal point if necessary. If the resulting effective line length is zero or less, an error message will be issued and the command is ignored. If this code is encountered when not at the start of a line, a new line is started.
- Paragraph Indent** Sets the starting point for this line only along the line axis by the value specified. This command does not affect subsequent lines. The distance is specified in the current UNITS, with decimal point if necessary. If the resulting effective line length is zero or less, an error message will be issued and the command is ignored. If this code is encountered when

	not at the start of a line, a new line is started.
Set Line Spacing	Sets the spacing perpendicular to the line axis between the top of the current line and the top of the next line. The distance is specified in the current UNITS, with decimal point if necessary, and may be smaller than the font height. If this code is encountered when not at the start of a line and not immediately following a Change Font command, a new line is started. This command overrides any existing Interline Space setting.
Set Interline Space	Sets the spacing perpendicular to the line axis between the bottom of the current line (including any white dot rows built into the current font matrix) and the top of the next line. The distance is specified in the current UNITS, with decimal point if necessary, and must not exceed 255 dots. If this code is encountered when not at the start of a line and not immediately following a Change Font command, a new line is started. This command overrides any existing Line Spacing setting.
Change Font	Sets a new font for characters following this command in the current line. The operand of this command is the numeric font ID specified in a Font Statement for the current RIP, and which must be between 1 and 255. This command may appear at the start or within the line. If the new font is a different size than the previous font and there are already some characters of the previous font in the line, the position of the text is adjusted in the

direction perpendicular to the line axis so that the "baselines" (the imaginary line on which most of the letters sit) of all the characters in the line are lined up. (See diagram below). If this command is not followed by an Interline Space command or by a Line Spacing command, the default interline space from the Font Statement is applied. Following this command the word space width is reset to the standard value of the new font. The new font must have the same orientation as the font specified in the previous Frame Statement.

Set Space Width	Sets the width of the normal word space to the value specified. The width is always specified in dots even if the current UNITS are different. All subsequent word spaces will be set to this width until the next change of font, the next Space Width command or the end of the current frame. The Space Width command cannot be used in full-justify mode.
End Line	Causes the current line to end and subsequent text to form a new line. If this command code is followed by a numeric operand, this value is taken to indicate the number of additional blank lines to be added after the line that just ended.
Insert Variable	The contents of the Variable specified by the value of the operand is inserted into the text at the current position in the line. Any Text Commands included in the contents of the Variable will be executed

where they are encountered. The effect of Change Font and Translate commands included in the Variable contents are limited to the contents of the Variable - the font and translation table in effect before the Variable was inserted are restored following the Variable.

Line
Justification

Sets the justification mode for this and following lines to the new value. The single-character operand of this command may be L,R,C or J meaning Left Justified, Right Justified, Centered or Full Justify respectively. In Full Justify mode, spaces between words are expanded to make all lines the full effective line width. The final line of each Full Justify paragraph (ended by an End Line code or the end of the frame) is set Left Justified. If this code is encountered when not at the start of a line, a new line is started.

Translate by
Table

Causes all following characters to be translated by the table indicated by the numeric operand. All text characters are translated by one of the tables. Unless changed by this command, each frame starts with text being translated by Table 1 which does not change the characters. The effect of this command continues until another Translate command or until the end of the current frame.

Underscore
start/end

This command starts or ends underscoring of characters. The operand may be 'S' for start or 'E' for end. All characters between these two commands will be underlined. Underlining will end at the start of a new frame even if the

'End' command is missing.

Start
Conditional
Text

Specifies a group of text lines that will be imaged only when the condition number indicated by the numeric operand of the command is set TRUE. The group of text lines is ended by an End Conditional Text command or by the end of the current frame. If this code is encountered when not at the start of a line, a new line is started.

End
Conditional
Text

Ends a group of conditional text lines. This command has a numeric operand which should match the operand of the Start Conditional Text command that started the group that is to be ended. This allows for nesting of multiple groups of conditional text lines. If this code is encountered when not at the start of a line, a new line is started.

Save Current
Position

Allows the current vertical and/or horizontal position to be saved for use in a subsequent command. This command has the following format:

KTnnn

where K is the command code, T may be 'H' or 'V' or may not be present and nnn is a numeric value. If T is 'H' or 'V', the vertical or horizontal position respectively is saved in the Variable indicated by the numeric operand. If T is not specified, both horizontal and vertical position values are saved in Variable 'nnn' and 'nnn+1' respectively.

Discretionary
Hyphen

This command consists of the hyphen character and has no operand. It may be

entered at suitable hyphenation points in words that may need breaking. If the word needs to be hyphenated, the indicated hyphen point will be used. All points that are not used are invisible in the printed output.

Text Strings

A text character string is encoded as a series of character codes enclosed in single quotes. Both upper and lowercase letters (and any other keyboard characters) may be used. There is no need for 'shift' codes. Should the user need to print the single-quote character, two single quotes (' ') should be used in place of one at the appropriate point in the text string. Hexadecimal text is encoded as a series of hexadecimal digits preceded by a capital X and optionally enclosed in single quotes. A text string may optionally be separated from following commands or other text strings with a comma.

Examples:

```
'This is a text string'X'C8C5E7'  
'This is also'Xclc2c3,E
```

Frame Example:

```
FRAME 240,480,720,480,0,1,24  
TEXT QL'This is Fred's first line of text'E  
TEXT T120'Tabbed at 1/2 inch from left'E  
TEXT L480'Left indented text'  
TEXT 'continued on the'  
TEXT 'following lines as'  
TEXT 'the text fits.'E
```

In the above example (on a 240dpi printer), a frame is started at 1" from the left and 2" from the top of the page. The frame is 3" wide, 2" high and the text is upright. It uses Font 1 with a line spacing of 10 lines

per inch. The first line sets the justification to left-justified and outputs a text string followed by a forced line end. The second line starts with a tab of 1/2" followed by a text string and another forced line end. A 2" left indent is set and the next four lines are set using the remaining line width. Lines 3,4 and 5 do not have forced line ends and would be re-cascaded to fit the line width.

```
OPTIONS UNITS=INCH
.
.
FRAME 1,2,3,2,0,1,.1
TEXT QL'This is Fred''s first line of text'E
T.5'Tabbed at 1/2 inch from left'E,L2.0
'Left indented text'
'continued on the'
'following lines as'
'the text fits.'E
```

The above shows the same example assuming a previous Options Statement had specified 'UNITS=INCH' as the measurement unit for the job. The printed result would be the same as the earlier example. Note that values may include a decimal point and as many decimal places as necessary (positioning will be done to the nearest dot).

Graphic Drawing Statements

Each of the following Graphic Drawing and Image statements must appear within a Frame and must be the only statement in the frame (i.e. it must be preceded and followed by a Frame Statement). All parameters may be literal numbers or letters as shown or may have the form:

$Vnn+xx$ (or $Vnn-xx$)

which allows them to be defined by the contents of a variable, plus or minus some constant value.

BOX Statement

Box draws a rectangular outline within the perimeter of the frame, with square or round corners and filled with a shading pattern if specified. Parameters following the identifier are:

1. Line Thickness a number indicating the dot width of the line with which the rectangle will be drawn.

The following parameters are optional:

2. Shading Fill a number, 0-100, indicating the percent of solid black to fill the interior of the figure. Zero is white and 100 is solid black. Zero is assumed if omitted.

Note: Currently only solid black or white interior fills are implemented - 0 or 100.

Examples:

```
BOX 12,0  
BOX 0,100
```

RULE Statement

The Rule Statement draws vertical or horizontal rules starting at the top or left edge of the frame, respectively. Parameters following the identifier are:

- | | |
|-------------------|--|
| 1. Line Thickness | a number indicating the dot width of the rule. |
| 2. Direction | letter 'H' for horizontal, or 'V' for vertical |

The following additional parameters are optional:

- | | |
|--------------|--|
| 3. Line Type | a number (0-xxxx) indicating the type of rule (solid, dotted etc). Default is 0 (solid). |
|--------------|--|

Note: Currently only the solid line type is available.

- | | |
|-----------------|--|
| 4. Number | a number specifying the number of lines to generate. The first rule is generated at the top or left edge of the frame. Succeeding rules are drawn at positions down (or across) the frame at spacing controlled by the next parameter. |
| 5. Rule Spacing | distance from the left (or top) edge of the first rule to the left (or top) edge of the next rule. |

Example:

```
RULE 12,H,0,8,60
```

The above example draws 8 horizontal solid rules at

quarter-inch intervals (on a 3600 system) each rule being 12 dots thick.

**IMAGE
Statement**

The Image Statement retrieves a bitmap image file and prints it aligned with the top left corner of the frame. Parameters following the identifier are:

- | | |
|--------------|--|
| 1. File Name | name of the file containing the bitmap image |
|--------------|--|

Image files may only be either PaintBrush type (.PCX) or Windows Bitmap type (.BMP) at present.

The size of the image is controlled by the bitmap data itself. If the image is larger in either dimension than the frame in which it is called out, MailForm will log an error, but the image will be output to the printer.

END Statement

Identifies the end of the MailForm Control File. This statement is optional and has no parameters.

6

Error Messages

The following section includes error messages displayed by MailForm Server and MailForm Remote Controller in addition to messages generated by in the MailForm Log File.

MailForm Log File

Log File messages are identified by a four-character code in the form:

Ann

where:

A is a letter which may be:

M - Critical Resource Error

E - Severe Error

F - Error in Job Start-up

R - Run-time Error

nnn is a message reference number

Critical Resource Errors

M001 Insufficient Application Memory

MailForm attempted to allocate memory for some storage requirement and the operating system denied the request. The job is terminated immediately.

Severe Errors

Severe errors cause the job to be terminated without saving any results. The job must be resubmitted after the problem is corrected.

E001 Could not open MailForm Control File

The job control file could not be opened.

E002 Unrecoverable I/O Error writing to Output File

This message could be caused by a full disk, a media error or a number of other causes. An I/O error message (see below) may indicate the cause.

E003 Could not Open Variable Input File

Error attempting to open variable input file. An I/O error message (see below) may indicate the problem.

E004 Read Error on Variable Input File

Error reading from the variable input. An I/O error message at the same point may indicate the cause.

E005 Incorrect Control File Format

The first line in the control file is not a 'MAILFORM' statement.

E006 Read Error on Control File

Error reading from the control file. An I/O error message at the same point may indicate the cause.

E007 Error Opening IJPDS Output File

This can be caused by a full disk, the file already exists with no Rewrite option specified, or various other possibilities. An I/O error may be indicated at the same point and may explain the problem.

E008 Actual Tape Blocksize greater than specified

When the variable input is being read from tape and the block actually read is larger than specified in the INPUT Statement, this error will result.

E009 Job Cancelled by Operator

When the MailForm Server operator or the user via the MailForm Remote Controller cancels the job, this message will record the event.

E010 Error Opening Output File - Job Aborted

When generating header labels on tape, any error causes this message. An I/O error message (see below) immediately ahead of this message should identify the cause.

E011 Input Data Block smaller than Record Size

On tape (or at the end of a disk file) the input was smaller than a full block. The block is padded out with EBCDIC space characters and processing continues.

E012 Job Aborted

Any fatal error while running the job will cause this message in addition to the one that indicates the reason for the abort.

E013 Output Filepath(s) not Specified - Job Aborted

When the job is started by the Server from the job queue, the output filepaths must be given via an OUTFILE Statement.

E014 Input Filepath(s) not Specified - Job Aborted

As for message E013, a queued job must include an INFILE Statement specifying the variable input files.

Job Start-up or Format Errors

The following errors indicate problems with the control file statements. MailForm will output the error message to the Log File *before* the line on which the error was detected. The message will be generated in the form:

```
---ERROR message  
-----V
```

where *message* indicates the error number and text and the following line of dashes normally appears, with the 'V' indicating the position in the following statement line that the error was found. If more than one error is detected in the line, several message and position pointer lines may appear before the statement line. You can search for errors in the Log File using a search string of '---ERROR'.

Usually, MailForm will attempt to continue processing the rest of the control file after detecting and logging errors of this category, but any one or more of this type of error will cause the job to be rejected after processing the control file. No output data file(s) will be generated when that happens.

F001 Format Error on Font Statement

There is no Fontname parameter following the ID number on the first line of the Font Statement.

F002 Font not found in Library

The *Fontname.FNT* file is missing from the

MailView/MailForm library. Make sure the font is for the correct resolution.

F003 Error in Library Font File

The file size of the font file is illegal. Try reloading the font into the library.

F004 More than one Input Statement

Only one Input Statement is allowed per job. When using multiple input files, all must have the same format.

F005 Format Error on Input Statement

The File Type parameter of the Input statement is not either F or V, or the Field Type on one of the field specifications is not one of C, B, P or Z.

F006 Format Error on Xlate Statement

The Translate statement is attempting to define a table with ID greater than 31. Only 32 tables are allowed.

F007 Format Error on GroupTrans Statement

Similarly to message F006, only 32 GroupTrans tables are allowed with IDs 0-31.

F008 Format Error on LookUp Statement

Only 32 Lookup tables are allowed with IDs 0-31. This statement has a different ID.

F009 Invalid Operand on Bounds Statement

The Bounds statement must have either a space

following the word BOUNDS before the first operand, or nothing after the word BOUNDS. The Bounds operands are not enclosed in parentheses.

F010 Descriptor type not recognized

The Data Descriptor type in a Variable or Condition statement is not legal. Check the statement format.

F011 Illegal Comparison Operator Pair

The legal comparison operators in a condition clause are: =, \, <, >, \=, <=, >=.

F012 Logical Syntax Error in Expression

In a conditional expression, several errors can cause this message - two operands together with no comparison between, two comparison operators together, no second operand after the comparison operator, etc. Check the format.

F013 Nested Brackets not Legal

Data Modifier groups, enclosed in brackets [], follow the operand they modify. You cannot use brackets within brackets, or the bracket character within a Data Modifier group.

F014 Open Bracket Missing

MailForm found the end of a Data Modifier group without a start. Check the format.

F015 Unbalanced Quotes in Expression

In a Variable statement, constant data must be delimited at each end by either a single quote

character (') or a double quote character ("). The same character must appear at both ends of the constant data. MailForm has found a statement containing an odd number of one or other type of quote character.

F016 Unbalanced Parentheses in Expression

All opening parentheses (other than those within constant data) must be matched by a closing parenthesis later in the expression. MailForm has found an expression with an unmatched set.

F017 Unbalanced Brackets in Expression

All opening brackets must have a matching closing bracket. MailForm has found an unmatched set.

F018 Illegal Character in Expression

In a Compute Descriptor parenthesized expression, the only legal characters are alpha, numeric and +, -, *, /, and %. Otherwise, characters legal in expressions are alpha, numeric and the following:
() ' " [= < > ! \ | & ~ @ \$ % * _ { } ; : ? . ^ `

F019 Illegal Hex Digit in Expression

The only legal characters in a hex expression are 0-9 and A-F. Lowercase alpha is also accepted.

F020 Odd Number of Hex Digits in Expression

All hex expressions longer than one character must have an even number of digits.

F021 Backslash not Legal in Variable Definition

The backslash character (for logical NOT) can only

be used in Condition definitions.

F022 Data Modifier Group in Illegal Position

Two Data Modifier groups immediately follow each other. A Data Modifier group must follow a Data Descriptor or parenthesized group of Data Descriptors.

F023 Vertical Justify Code not T,B,C

In the Frame statement, these are the only valid options for the Vertical Justify parameter.

F024 Non-numeric in Length Specifier

The 'A' specifier indicating truncation of characters after a desired number, must have a numeric operand.

F025 Format Error on External Statement

The NAME field is missing from the External Statement.

F026 Non-numeric in Offset Specifier

The 'F' or '+' specifier indicating truncation of a number of characters at the start of the field, must have a numeric operand.

F027 Non-numeric in External Procedure ID

External Procedure IDs may range from 0-40.

F029 Non-numeric in Internal Procedure ID

All Internal Procedures are referred to by number. See Appendix B.

-
- F030** **Non-numeric in Translate ID**
- Translate Table IDs range from 0-31.
- F031** **Non-numeric in GroupTrans ID**
- GroupTrans Table IDs range from 0-31.
- F032** **Non-numeric in W Modifier**
- The 'W' modifier discards a number of characters from the end of the field. It must have a numeric operand.
- F033** **Illegal Internal Procedure Code**
- Following the 'P' modifier code, an Internal Procedure has been referenced which is not one listed as currently available in Appendix B.
- F034** **Non-numeric in Lookup ID**
- Lookup Table IDs range from 0-31.
- F035** **Illegal Data Modifier Code**
- MailForm found a Data Modifier function code that is not one of those listed in the 'Data Modifier Groups' section of this manual. Check the format.
- F036** **Statement Unidentified or Out of Sequence**
- MailForm found a line in the control file that did not begin with a space, and the Statement Name at the start of the line either was not one of the supported statement types, or that statement was in the wrong place in the file. Check spelling and Statement Order in Section 4.

F037 Non-numeric in Z Modifier

The 'Z' data modifier function discards all characters in the field before some number at the end. It must have a numeric operand.

F038 Null Text Character String - Error?

In a Text Statement, MailForm found a 'null' character string — i.e. a text string that has a starting quote, an ending quote, but no printable characters between them. This is flagged as an error, because it is a questionable construction that MailForm doesn't allow.

F039 Illegal Hexadecimal Value

The only legal characters in a hex text string are 0-9 and A-F. Lowercase alpha is also accepted.

F040 Conditional Expression missing on Condition Statement

Following the condition identifier in a Condition statement, the condition clause appears to be missing. Check the format.

F041 Command Code not recognized

In a Text statement, the command code character at the start of the line or following a comma or a text string was not one of those defined in the table under Text Statement in Section 5 of this manual.

F042 Action Keyword not recognized

In an Action statement, a keyword was used which is not one of the types listed under Action Statement in Section 5 of this manual.

F043 Non-numeric Action Parameter

READ, REPEAT and SIGNAL action parameters must have numeric parameters, if any.

F044 Format Error in Action Statement

A LOG, ERROR or MESSAGE action code has a text parameter string that has no ending double-quote (“).

F045 Non-numeric/Illegal Value in Font Statement

The Font ID must be a numeric value from 1 to 255.

F046 Input Record Size is zero bytes

The input record must be at least one byte long. Note that the use of no input file is signaled using an Input Statement with a parameter of ‘NONE’.

F047 Non-numeric or Comma Missing in Statement

This message is used on a lot of statements where a numeric parameter is required and none is found. If it is allowable to omit the parameter, the comma is usually required.

F048 Input Block Size is zero bytes

A zero-length block size is illegal, but if you omit the parameter but include the comma, MailForm will assume a block size equal to the record size.

F049 Block Size must be multiple of Record Size

A block should contain several complete records, therefore the block size should be a multiple.

F050 Record Start Column must be zero or one

You can count the first character of the record as position zero or as position one. This parameter has a different value.

F051 Start position not in record

A field start position in the Input statement is less than the starting column value or beyond the end of the record.

F052 Field length exceeds Record end point

The specified length for the field goes over the end of the record.

F053 Field Name must not exceed 20 characters

This is a current MailForm limitation.

F054 Variable Number exceeds maximum

Variables may be numbered up to 9999.

F055 Condition Number exceeds maximum

Conditions may be numbered up to 1025.

F056 Non-numeric or Colon Missing in Statement

In an Input Descriptor or Internal Procedure modifier, MailForm expected a numeric value or a colon character (:).

F057 Orientation Value Incorrect

Orientation parameters on FONT, PAGE, FRAME statements must be U, I, L or R or 0-3.

F058 **Justification Code not L,R,C or J**

In the Frame statement, the line justification parameter must be one of those listed in the message.

F059 **Data Type not recognized**

In building a Variable or Condition, the input did not match a Data Descriptor type, any of the Type Descriptors (listed under the Variable Statement subsection) or any of the Action keywords.

F060 **Format Error on Printer Statement**

Either the manufacturer and/or model name is not as listed in Appendix A, the model parameter is not 'RIPS=n' (it must be this form for 6240/5240 systems), or the third parameter is not 'SYSTEMS=n'.

F061 **More than 255 fonts needed per RIP**

If individual font rotations are not defined with separate Font Statements, MailForm generates the necessary additional font identifiers by scanning the document text for rotational usages other than those specified. If those additional identifiers would cause the number of font IDs to exceed 255 in any one RIP, this error will appear.

F062 **Duplicate Font Definition - First Ignored**

If two Font Statements specify the same ID in the same RIP, MailForm ignores the earlier of the two statements. The job will, of course, not run to completion with this problem.

F063 Font Selected has Wrong Rotation

In scanning the document text for font rotation usages, MailForm found an instance where the Font Statement specified one orientation and it was used in a frame having a different rotation.

F064 Translate Table Undefined

A 'Translate by Table' (Z) command in a Text Statement (or constant data within a Variable or Condition statement) specified a table number that was not previously defined by a Translate Statement.

F065 Text Position Off Page Buffer to Left

F066 Text Position Off Page Buffer to Right

F067 Text Position Off Page Buffer at Top

F068 Text Position Off Page Buffer at Bottom

The starting point of a text line is beyond the extent of the current page buffer in the direction indicated by the message.

F069 Format Error in Compute Descriptor

If the 'number of digits in the result' parameter following the initial 'C' is zero or non-numeric or the open parenthesis enclosing the compute expression is missing, this message will result.

F070 Frame Width exceeds Bounds limit - truncated

The specified frame width puts the edge of the frame outside the current Bounds limit (either default or set by a Bounds Statement). MailForm truncates the width to the limit in order to continue processing the control file. The job will not, of course, run to completion.

F071 Invalid RIP Number

On the RIP statement, the RIP number must be 1 or 2 for 3000-series RIPs or 1-8 for 6000-series RIPs.

F072 Illegal System Type on RIP Statement

Parameter 2 of the RIP statement may only be one of the Model ID numbers listed in Appendix A.

F073 Illegal Value in Total Jets on RIP Statement

If the Total Jets parameter is less than 240 or more than the maximum for the selected printer type, this message will result.

F074 Number of Heads Invalid on RIP Statement

The number of heads must be from 1 to the maximum for the selected printer type.

F075 Illogical Head Position on RIP Statement

If the head position offset for any head is less than zero or greater than 9999, this message will result.

F076 Less Head Positions than Heads Used - RIP Statement

If head position offsets are specified on the RIP statement, offsets must be given for all heads used.

F077 Invalid Head Size on RIP Statement

A specified printhead size is not a width used on the selected printer type.

F078 **Less Head Sizes than Heads Used on RIP Statement**

If printhead sizes are specified on the RIP statement, they must be given for all heads used.

F079 **Incorrect Number of Parameters on RIP Statement**

MailForm found more parameters than it expected on the RIP statement based on the values given.

F080 **Invalid/Undefined Font ID**

On the Frame statement or Font Change command on the Text statement, the specified font ID was not previously defined.

F081 **Logical Combine Keyword Incorrect**

The Writing Mode parameter on the Frame Statement (number 11) must be REP, OR, AND or XOR or a number from 0-3.

F082 **Bad format in Translate Statement**

The table specifications on the Translate Statement generally take the form *xx=yy* followed by a comma before the next one. The entire specification item between commas must exist on the same line of the file. It is permissible to start a new line (with a blank in the first position) after any comma but nowhere else. Also if single character values are specified using single quotes before and after them, if the second quote is missing, this error will occur.

F083 **More than one Unconditional Document Statement**

If more than one Document Statement is supplied in one job file, all except one must have a

conditional expression parameter.

F084 No Unconditional Document

See message F083. There must be one Document Statement that has no conditional parameter.

F085 End Text Condition with No Condition in Effect

An End Conditional Text command with no operand was encountered on a Text Statement and no previous Start Conditional Text is in effect in this frame, or if a condition number operand was supplied for the End Condition, there is no condition with that number in effect at this point.

F086 Redefining Non-Inheritable Variable

A Variable statement using the same variable number as a previously-defined Variable was encountered and the previous definition did not indicate the variable was Inheritable (and therefore may be redefined).

F087 Condition not Previously Defined

In a Variable, Condition, Action or Document statement, a condition clause references a condition number not previously defined in another Condition Statement.

F088 Referencing Illegal Variable Number

A Reference Descriptor specifies a variable number greater than 9999.

-
- F089 Referenced Variable not Previously Defined**
- Variables to be recalled by a Reference Descriptor must be defined earlier in the file than their reference point.
- F090 Comparison Operator Missing or Incorrect**
- In a conditional expression, the legal comparison operators are listed under Condition Clauses in Section 5 of this manual.
- F091 Multi-RIPs Cannot be used with Multiple Systems**
- If using multiple printer systems linked via a “Synchronizer” or similar device, multiple RIPs cannot be used in the same job.
- F092 Non-Numeric Value in Graphic Statement**
- In a Box or Rule statement, one of the parameters is non-numeric.
- F093 Box Fill must be 0 or 100 (percent)**
- Currently MailForm for Ink-Jet can only generate boxes with solid white or black fills.
- F094 Rule Direction must be H or V**
- Only vertical or horizontal rules can be generated at present.
- F095 Rule Thickness must be at least 1 dot**
- The thickness parameter cannot be zero.

F096 Repeated Rule with zero spacing

Spacing between repeat rules cannot be zero.

F097 Repeated Rules exceed Frame Area

All repeated rules must be within the area of the frame. Increase the frame or reduce the spacing or number of rules.

F098 Rule Thickness exceeds Frame Area

The complete area of a rule must be within the current frame.

F099 Variable Undefined

A Variable command in a Text Statement references a variable number that has no been defined.

F100 Font Statements must be before Document Statement

No Font Statements were found before the Document Statement in the job file.

F101 Document Statement must be before Page Statement

A Page Statement was encountered before any Document Statement.

F102 Page Statement must be before Frame Statement

A Frame Statement was encountered before any Page Statement.

F103 Illegal Keyword in Statement Parameter

This message will result on any statement that allows Keyword Parameters when the keyword on the statement does not match any of the list of keywords valid for that statement.

F104 Undefined Definition Type Name

On a Definitions Statement, the parameter type for a character group or processing sequence table does not match any of the valid keywords.

F105 Cannot change Font Angle in Angled Frame

When using an angled font in a frame, all fonts used in that frame must be created for the same angle.

F106 Must Specify Document Count when no Variable Input

If the Input Statement specifies 'NONE' or if no Input Statement is included, MailForm runs the job with no variable input. In this case a count of the number of documents to be generated must be given on the Run Statement.

F107 Format Error on Definitions Statement

On the Definitions Statement a comma must immediately follow the name of the character group or table being defined.

F108 Font ID Undefined in FONT Defaults Definition

On the Definitions Statement, when defining a FONT Defaults table, one of the Font IDs has not been previously defined.

F109 Printer Statement missing before RIP/FONT Statements

The Printer Statement should immediately follow the MailForm Statement at the start of the job file.

F110 No Recognized Descriptors on Variable Statement

MailForm reached a comment at the end of a variable definition without encountering any Data Descriptors.

F111 Multiple RIPs not specified on Printer Statement

A RIP Statement was supplied and the Printer Statement did not specify that multiple RIPs would be used.

F112 Illegal File Type Parameter

The File Type parameter on the INFILE and OUTFILE statements must be either D for disk files or T for tape files. (The Windows95 version of MailForm does not support tape files, so the only legal parameter is D). If omitted, the default for this parameter is D.

F113 Format Error on INFILE Statement

If there is no comma after the File Type parameter or there is no comma in place of a missing File Type parameter, this message will result.

F114 Cannot access specified Input File

The file specified on the INFILE statement is not accessible by MailForm. Check the path and ensure that the MailForm Server has access privileges to the directory.

F115 Format Error on OUTFILE Statement

If there is no comma after the File Type parameter or there is no comma in place of a missing File Type parameter, this message will result.

F116 Statement out of sequence - must be after PRINTER

INFILE and OUTFILE statements must follow the Printer Statement in the Control File.

F117 Output Files do not match number of Image Systems

The Printer Statement included a SYSTEMS= parameter indicating multiple output files should be generated and the OUTFILE Statement has a different number of output filepaths specified.

F118 End Underscore without Start

An End Underscore command was encountered in a Text Statement when there was no previous matching Start Underscore command.

F119 End Underscore command missing

A Start Underscore command was encountered, but no matching End Underscore command was found.

F120 Duplicate External routine definition

There should be no more than one External Statement defining any External ID number.

F121 Failed to Find/Load External Routine

The executable file for the named external routine is missing or not accessible.

F122 Unbreakable Text Exceeds Line Length

A fixed text string specified in a Text statement is too wide for the line width and contains no characters that would allow a line break.

F123 Format Error on List Statement

The List ID number specified is greater than 31 or one of the list item specifications does not begin with either X' or a single quote.

F124 Illegal External Procedure ID

The External Procedure number must be 0-40.

F125 List Table Undefined

A List Data Descriptor references a List Table number that has not been defined.

F126 Closing parenthesis missing in Name Default

On a Definitions Statement, defining a Name Defaults table, an *ifclause* has been specified with an open parenthesis but the matching close parenthesis is missing.

F127 Name Default element code not recognized

On a Definitions Statement defining a Name Defaults table, a two-letter element code does not match any legal value.

F129 Already past position for Horizontal Tab

In processing fixed text from a Text Statement, the text in the line was already wider than the amount of the Horizontal Tab when the command was encountered.

F130 External did not Initialize Correctly

In attempting to start up External 32 (the start-of-job external procedure), the initialization timed out and did not proceed correctly.

F131 Text Position Exceeds Left Bounds

F132 Text Position Exceeds Right Bounds

F133 Text Position Exceeds Top Bounds

F134 Text Position Exceeds Bottom Bounds

One edge of the frame defined by the current Frame Statement is outside the current Bounds limits (either default or set by a previous Bounds Statement).

F135 Printheads not Stitched Across Bounds Area

The Bounds Statement defines an area which is not covered by one printhead or by several 'stitched' printheads.

F136 Frame Depth exceeds Bounds depth - truncated

The defined frame depth exceeds the limit of the current Bounds area.

F137 No Valid Font Definitions

In a single-RIP job, there are no valid font definitions. MailForm cannot continue to process the control file in these circumstances.

F138 No Valid Font Definitions for RIP

One RIP has no valid font definitions. MailForm cannot continue to process the file.

F139 Reverse-Reading not supported on Tape Input Files

The backward-reading input file option is only available for disk-resident files.

F140 Reading from Undefined Input File

An Action READ statement has a file number parameter for which no input file was defined.

F141 Translate Table Undefined

A type X Data Modifier was specified with a table number that has not been defined.

F142 Lookup Table Undefined

A type Y Data Modifier was specified with a table number that is undefined.

F143 External Procedure Undefined

A type E Data Modifier was specified with an External number that has not been defined.

F144 GroupTrans Table Undefined

A type G Data Modifier was specified with a table number that is undefined.

F145 No RIP Statements in File

The PRINTER Statement contains a RIPS= parameter, but no RIP Statements appear in the file. If the PRINTER Statement specifies RIPS=1, you must include a RIP Statement.

F146 **Cannot Offset Negative on Tape Files**

If the INFILE Statement specifies tape input, the offset record number parameter must be positive or omitted.

F147 **Number of Rules less than One**

In the RULE Statement, if the Number of Rules parameter (4) is included, it must be greater than zero.

F148 **Font has no Hex 40 Space Character and no room**

If fonts have no Hex 40 Blank character, MailForm tries to generate one in an unused location. This message indicates the font uses all 256 character locations.

Run-Time Errors

The following errors are generated in the Log File while MailForm is running the job. The format of the messages is different from those generated while processing the Control File. The Run-Time messages generally have the following format:

```
---ERROR message  
-- Document nnn
```

None of these messages cause MailForm to stop processing the job.

R001 **Overset Line**

The text contents of a line exceed the line width and there is no possible line break point within the line. MailForm prints the line so that it runs over

the specified line width.

R002 Already past position for Horizontal Tab

In the current line, the text is already wider than the point to which the Horizontal Tab code specifies movement. MailForm ignores the command and continues.

R003 Already past position for Vertical Tab

In the current frame, the lines have already passed the vertical point to which the Vertical Tab code specifies movement. MailForm ignores the command and continues.

R004 Word in Variable *nnn* too Long for Line

A single 'word' with no possible line break point in the contents of the variable whose number is reported in the message, is too wide for the current line length. MailForm prints the text so that it exceeds the line width and continues.

R005 Word in Document too Long for Line

Similarly to message R004, there is a word, this time in fixed document text, which is longer than the line width with no possible break point. MailForm oversets the line width and continues.

R006 Text Position Off Page Buffer to Left

R007 Text Position Off Page Buffer to Right

R008 Text Position Off Page Buffer at Top

R009 Text Position Off Page Buffer at Bottom

The end of the current line runs off the page buffer at the edge indicated in the message. MailForm

continues with the job.

R010 Input Record too small - skipped

If an input block is of such a size that some input records end up smaller than the defined record size, MailForm skips these records. This situation can only occur on tape input files when the input block size varies.

R011 Process 12 Font undefined

Internal process 12 sums the width of the contents of a variable using a specified font. If the font is undefined, this error results.

R012 Process 13 Table not Defined

This error should never occur because this problem should have been caught during processing of the job control file.

R013 Illegal Frame Orientation Variable Value

When the orientation parameter of a Frame is specified as being obtained from the contents of a variable, if the resulting value is not 0-3, this error results and MailForm defaults to upright orientation.

R014 Illegal Frame Logical Combine Variable Value

When the Writing Mode parameter of a Frame is specified as being obtained from a variable, if the value is not 0-3 this error results and MailForm defaults to REP (replace).

R015 Illegal Frame Line Justification Variable Value

When the Line Justify parameter of a Frame is specified as being obtained from a variable, if the value is not 0-3 this error results and MailForm defaults to Left Justify mode.

R016 Illegal Frame Initial FontID Variable Value

When the Initial Font ID parameter of a Frame is specified as being obtained from a variable, if the resulting ID number does not specify a legal font, MailForm sets a default font.

R017 Illegal FontID Variable Value

When a Font Change command specifies that the new ID value is to be obtained from a variable, if the resulting ID number does not point to a legal font, MailForm defaults to a legitimate substitute font.

R018 Illegal Line Justification Variable Value

When the operand of a Line Justify command specifies that the value is to be obtained from a variable, if the resulting value is not 0-3, MailForm defaults to Left Justify mode.

R019 Process 14 - Text too wide using smallest font

When internal process 14 is used to select a font that will allow a text string to fit within a certain width. If the smallest font in the table of fonts is still too large, this error results but MailForm uses the smallest font.

R020 Illegal Font ID - Font Code in Variable

When a Font Change command specifies that the new ID value is to be obtained from a variable, if the resulting ID number does not point to a legal font, MailForm defaults to a legitimate substitute font.

R021 Font Code in Variable changes Font Angle - ignored

In an angled frame, when a Font Change command specifies that the new ID value is to be obtained from a variable, if the resulting ID number addresses a font with a different angle, the change command is ignored.

R022 Frame Variable position off Page Buffer to Left
R023 Frame Variable position off Page Buffer at Top

When the X and/or Y position of a frame is specified to be obtained from a variable, if the resulting position is off the page buffer, one of these messages results and the position is set to zero.

R024 Undeclared External Called

This error message should not appear since this error should have been detected when processing the control file.

R025 External did not return from processing variable

MailForm called an external routine to process a variable and then timed out waiting for the external to complete. Processing continues.

R026 External Returned Illegal Message Pointer

MailForm ignores the message.

-
- R027 Text Position Exceeds Left Bounds
 - R028 Text Position Exceeds Right Bounds
 - R029 Text Position Exceeds Top Bounds
 - R030 Text Position Exceeds Bottom Bounds

The end of the current text line exceeded the Bounds limit in the direction indicated in the message.

- R031 Unsupported Command in Variable *nn*

MailForm allows the possibility of creating both text and commands in the contents of a variable. This message occurs when a command is not recognized. It is usually due to incorrect data in the variable.

MailForm Server Messages

MailForm Server messages are displayed in a standard Windows message box. The following are the messages that can be displayed by the Server:

Failed to Get Registry Variables

This message should never occur if MailForm is correctly installed. MailForm did not find the expected items in the Windows Registry. Reinstall the MailForm Server.

Error Opening JobQueue File - Err #*nnn*

This message can occur only if the Server does not have access privileges in the directory in which the Job Queue is to be maintained. Check the Preferences setting for the Job Queue directory. If all appears OK, report the error number in the message to SoftView.

Problem Opening Job Queue

This can be caused by I/O errors when starting up the Job Queue. Restore or delete the file (MAILFORM.JQU) and retry.

CreatePipeHandler Error

This error indicates some operating system incompatibility. Call SoftView Systems.

Create Pipe Error=*nnn*

This indicates the Server was unable to start up the Named Pipe by which it communicates with the MailForm Remotes. Call SoftView to help with the problem.

CreateThread Error

Again, this is an operating system incompatibility problem. Call SoftView Systems.

Maximum Tasks currently running

If six jobs are currently running on this Server and you try to start another via the Server menu, this message will result. The Menu command is ignored.

Security License Unavailable

The number of jobs currently running equals the number of licenses stored in the MailForm keyplug. When you try to start another job via the Server menu, this message will then appear. Wait for a job to complete and then try starting the job again (or buy more licenses).

Security License Expired

The temporary license stored in the MailForm keyplug has expired. Call SoftView for renewal.

Failed to Allocate INFO data

A problem with mapping shared memory occurred when starting a MailForm Task. Report this problem to SoftView. Try exiting and restarting MailForm Server - make sure all jobs are completed or can be restarted when you do this.

Error *nn* Opening File Map

Another problem setting up the shared memory, this time from the MailForm Task perspective. The task will not start. Try restarting the MailForm Server when convenient.

Could not Map File

Another shared memory problem. The task will fail to start. If persistent, a restart of the MailForm Server should be done. Please report any message of this type to SoftView.

MailForm Remote Messages

MailForm Remote Controller messages are displayed in standard Windows message boxes.

Unable to Connect to Server Connect Error = *nnn*

This message can occur if the MailForm Server that is managing the Job Queue is not running.

Cannot Access *filepath*

This message occurs if, when submitting a job, the *filepath* entered in the Submit Job dialog box is incorrect.

Job Submission Rejected by Server

If an error occurred while attempting to write the job entry to the Job Queue, this message will result. Try submitting the job again.

Server could not access Control File *filepath*

A job was submitted but the Server cannot read the specified MailForm Control File. Check the sharing privileges the server has in respect to the directory where the file resides.

Output File *filepath* Already Exists OK to Delete?

This message is equipped with YES and NO buttons. Click YES to delete the old file. NO will cause the job to be rejected.

Cannot Delete File - Submit Failed

When you click YES on the previous message, if the Server is unable to delete the old file (perhaps it is open), this message will result.

Job Submission is Cancelled

When you click NO on the File Delete message, this one then makes sure you know what happened to your job.

A

APPENDIX A Supported Printer Systems

The following printer systems are supported:

Printer	Manufacturer ID	Model ID
Scitex 3600 System	SCITEX	3600
Scitex 3500 System	SCITEX	3500
Scitex 3000 New Data System Units	SCITEX	3000
Scitex 6240 System	SCITEX	6240

B

APPENDIX B MailForm Built-In Procedures

The currently-defined MailForm procedures that can be invoked with the P Modifier are as follows:

- 1 A simple Capitalization procedure that expects uppercase alphanumeric input and converts all characters following a letter or number to lowercase. All other characters are unaffected. The first character is always uppercase.
- 2 A more complex Capitalization procedure also expecting uppercase alphanumeric input. It converts all characters not at the start or not following either a blank, period, comma, hyphen, ampersand or slash to lowercase with the following exceptions:
 - The following words are capitalized only if they occur at the start of the character group: an, and, for, of, on, the.
 - Any word containing numeric characters is not lowercased except numbers followed by: st, nd, rd, th.
 - Words beginning with MC or O' are converted with both first and third letters remaining uppercase.
 - The following numbers and abbreviations are left in all uppercase: II, III, IV, VI, VII, IX, NE, NW, SE, SW, RR, PO, MD, PHD, DDS, DVM, RFD.
- 3 A decimal incrementing procedure. The input to this procedure should be a character group that ends with one or more numeric digits. This routine increments the numeric part of the character group at the end of the group only. When all numeric digits are 9s, the next increment converts the value to zero without affecting any preceding alphabetic characters.
- 4 A Modulus 10 check digit procedure. This procedure

expects a character group containing only numeric digits. If alphabetic characters are present, the character group is unchanged. The procedure computes a check digit from the character group and appends the computed digit to the end of the group.

- 5 A Modulus 11 check digit procedure. This procedure is similar to Procedure 5, but the resulting check digit is computed based on a modulus 11 method.
- 6 A decimal decrementing procedure. This procedure operates like Procedure 3 but decrements instead of incrementing. When all numeric digits of the decrementing part are zero, the next decrement converts the value to all 9s without affecting preceding alphabetic characters.
- 7 An unpack procedure for IBM packed decimal fields. This procedure expects a valid packed decimal character group as input. If invalid digits are found in the input, the character group is returned unchanged. The sign of the packed field is discarded - only the digits are returned as numeric characters.
- 8 A procedure that removes leading and trailing spaces on the character group, reduces all contiguous spaces within the group to one space at each instance and then, if the group is not empty, attaches one space to the end of the group.
- 10 A Postal Check Digit procedure. This routine expects a character group of 5, 9, or 11 numeric characters only, and computes a check digit according to US Postal rules for Postal Bar Codes. The computed check digit is appended to the character group. If a non-zero parameter is supplied (P10:1), the routine will check the Delivery Point, Zip+4 and Zip5 digits to ensure they are numeric and each item is not all zeros. If either problem is found in any item, the

routine will discard those digits from the character group and fall back to the next shorter code length - the check digit is then computed with the remaining characters. If the Zip5 item is invalid, the routine returns a zero-length result.

- 11 A procedure which counts the characters in the character group and returns the decimal count in place of the character group. The count is six EBCDIC decimal digits with leading zeros.
- 12 A procedure which computes the printed width of the character group and returns the result as a decimal number in dots. This procedure requires one parameter - the font ID of the font to be used. Thus the Modifier is coded 'P12:xx' where 'xx' is the font ID. The width value is six EBCDIC decimal digits with leading zeros. If the font is undefined, the returned value is zero.
- 13 A Font Defaults procedure. The Modifier is coded 'P13:xx:www' where 'xx' is the 'font defaults' definition ID (see the DEFINITIONS statement) and 'www' is the maximum width for the character group. The procedure uses the list of fonts indicated in defaults list 'xx' and sums the width of the character group using each font in turn until it finds one that gives a total width smaller than 'www'. It then adds a Font Change command at the start of the character group in this Variable. If no font is found that gives a total width less than 'www', the procedure sends an error message to the Log File and sets the last font in the defaults list.
- 14 Another Font Defaults procedure, identical to Procedure 13, but which returns the selected font ID as a three-digit EBCDIC number in place of the original contents of the Variable. Again, the Modifier must be coded as 'P14:xx:www'.

-
- 15 A 'Name Defaults' procedure. The Modifier is coded as 'P15:xx:yy:www' where 'xx' is the 'name defaults' definition ID (the number entered on the DEFINITIONS statement), 'yy' is the font ID of the font to be used for typesetting the name and 'www' is the maximum width for the character group. As was detailed earlier in the description of the DEFINITIONS statement, the character group supplied as input to this procedure must be in the format 'Title:First:Middle:Last:Suffix' where there should be four colon characters in the character group. If more or less than four colons are present, the routine will generate an error message and will assume:

- 3 colons - Title:First:Middle:Last

- 2 colons - First:Middle:Last

- 1 colon - First:Last

- >4 colons - Title:First:Middle:Last:Suffix:additional data ignored

If there are no colons, the text will be returned unmodified. The returned data from the procedure is a formatted name, in the same casing as the input, with one blank between name elements and one period after any initial letter (if used). If no arrangement of name elements is found that will fit the required width in the specified font, an error message is sent to the log file and the shortest arrangement of elements is returned. If the 'Truncate Last Name' directive is used (and the procedure returns a truncated last name) an error message is sent to the log file.

NOTE: Procedure 15 is currently not implemented.

- 16 A procedure to convert a binary field into its decimal equivalent. This procedure requires one parameter - the number of digits to be generated. Thus the Modifier is coded "P16:xx" where xx is the number of digits requested. If the number of digits is greater than the number of significant digits in the input

value, zeros will be added to the high-order of the output character group. If the number of digits is fewer than the number of significant digits in the value, high-order digits will be truncated and a warning error will be generated.

- 17 A procedure to convert a character group containing hexadecimal characters into the decimal equivalent of its value. Again, this procedure needs one parameter - the number of digits to be generated. Thus the Modifier is coded "P17:xx" where xx is the number of digits requested. If the number of digits is greater than the number of significant digits in the input value, zeros will be added to the high-order of the output character group. If the number of digits is fewer than the number of significant digits in the value, high-order digits will be truncated and a warning error will be generated. An error message will be generated if any digits in the input character group are not legitimate hexadecimal values.

The following procedures are currently not implemented:

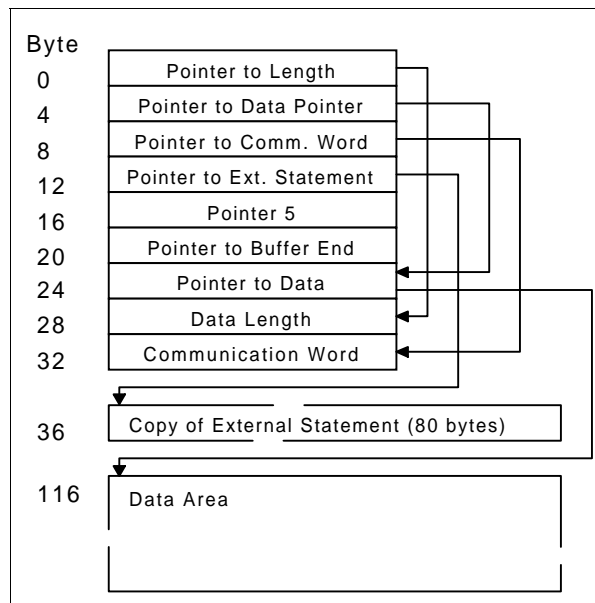
- 18 A procedure which converts the final word of the character group into the possessive form (fred's, jones', etc).
- 19 A procedure which converts the final word of the character group into its plural form. The word is assumed to be a name. The procedure adds an 'S' (in the appropriate case) except when the name ends in X, S, Z, J or CH.

C

Appendix C Developing External Procedures for MailForm

In the design of MailForm, an attempt was made to keep the interface to External Procedures very similar to those used in MailView™ so that the same External code could be used with both systems and so that the transition would not be difficult. MailForm passes two parameters to the External when calling the procedure:

1. The number of the External procedure. This can be used by procedures that may be called at several different MailForm processing points to determine under what circumstances the call is being made.
2. A pointer to a list of pointers to data being passed in the common buffer. The pointer list structure is illustrated in the following diagram:



MailForm External Procedure Parameter Structure

Parameter 2 passed at the call of the External routine points to Byte 0 of the structure in the diagram. Although the position of elements starting at Byte 24 is shown here, the user should not rely on the absolute location of these elements but should use the pointers to them thus isolating the External code from future possible changes. The pointer at Byte 20 points to the final byte in the common buffer. Memory between the end of the Data Area and this location may be used to pass back messages or for any other purpose. Memory locations beyond this address are not in the External's address scope - referencing addresses outside the common buffer will cause a program exception. Pointer 5 is used on some External calls to point to additional parameters - see the descriptions below.

Communication Word

The Communication Word is used as a flag to indicate status on some External calls. Normally it is set to zero. On the return, the External may set the Communication Word in one of two ways:

1. To pass a message to MailForm for logging to the Log File, the high-order byte of the Communication Word is set to the length of the message (max 255) and the low-order is set to the address *in the common buffer* of the first message byte. As spelled out in more detail below, the message text must be copied to the buffer or MailForm cannot access it. When the External returns with a message indicated, the message is logged and MailForm then immediately calls the External again. If the first character of the message is used as a carriage control character (as on the mainframe system), the character will be written to the log file - MailForm does not recognize carriage controls. The External Development Kit provided with MailForm provides a routine (Save_Communication_Code) that takes care of combining the length and

address elements of the Communication Word in various environments. The user should call this procedure to isolate the External code from the type of processor and the environment in which the system is running.

2. After any messages have been passed back, the External returns the results of its processing by updating the appropriate elements in the parameter list and by setting the high-order byte of the Communication Word to zero. The low-order three bytes of the Communication word may contain a value other than zero in some cases as a status. The External procedure may modify any of the pointers and/or the contents of the words and data areas that they point to. However, in Windows NT (and in Windows/95), the task calling an external procedure can only pass data back and forth in a buffer which is mapped into the address space of both processes. This means MailForm cannot address constants or other locations in the External procedure and vice-versa. The effect of this change is that when you pass messages or any other data back to MailForm, the data itself must be moved to the common buffer and all pointers in the pointer list must point to locations in the common buffer. MailForm automatically provides additional buffer space to allow for extending the data being processed and a pointer to the end of the available buffer area is included in the list. Should even larger buffer work space be required, the BUFFER parameter on the External Statement should be used to ensure that MailForm allocates adequate memory space.

Another major difference from the mainframe environment is that, as with MailView, MailForm exits receive all their text data in ASCII representation, even if the data is extracted from an EBCDIC variable input file. All non-

mainframe compilers assume ASCII representation for character constants and this is not easy to work around. With ASCII input, External routines can use character constants without problem. If the user absolutely wants the input data in the same representation as on the input file (without translation), the INPUT statement allows any or all fields to be identified as binary, thus inhibiting conversion.

The following sections identify each type of External procedure, the circumstances under which it may be called, the data passed to it and the data and information expected back from it.

Externals 0 through 31

Up to 32 different procedures may be linked in any one job and called to perform special work on the contents of a Variable or part of a Variable. The External is identified in the Data Modifier Group in square brackets following an Input Descriptor or Reference Descriptor in the Variable definition. The letter 'E' signals an External call and the parameter value is the number of the procedure to be called. The data area contains the current contents of the item being modified (after any previous modifiers have been applied) and the length reflects the number of characters in the group. The External procedure may make whatever changes it wishes to the data and may change both the data address and data length. Note that MailForm uses the returned length value, therefore if the number of characters in the data changes, the length value must be updated. The size of the data sent to and received back from the External is limited only by the size of the common buffer.

Return Codes in the Communication Word

- 0 Normal processing return.
- 4 Skip this entire input record and go on to the next. All variables already built are discarded. No

-
- document is produced. (Same effect as an ACTION SKIP)
 - 8 Process the data returned on this call, but do not call this procedure again. The remainder of the job is processed without calls to this External.
 - 12 Process the data returned on this call, but also insert a Stop Code into the output tape before the current document. (Same effect as an ACTION STOP)
 - 16 Process the data returned on this call, but also start a new output tape reel with the current document. (Same effect as an ACTION FEOV)
 - 20 End the job. This variable and all other variables already built are discarded. No document is produced. (Same effect as an ACTION END)

External 32

This external is called when the External Statement for it is processed. A copy of the External statement is passed, but no other data (the data length is zero). The communication code is zero and should be returned as zero, but log messages may be sent if desired. There is only one call to this routine, but because it occurs very near the start of processing, it may be used as a start-of-job call. There is no problem with one External Procedure being called from several different processing points in MailForm (just specify the same External on more than one External Statement), and this call may be used for initialization. MailForm does not expect any returned data or flags (other than log messages).

External 33

This external allows the provision of a custom procedure for reading the variable input file. MailForm calls this routine whenever a new input record is required instead of calling its own input routine. The Communication Word is zero on the call and no data is passed in the Data Area (the length is zero). The External procedure must handle all I/O functionality using Win32 Functions and all necessary tape label processing, multi-volume handling etc. External

33 passes the input record, after any required editing of the data, back to MailForm in the common buffer and must change the data length and data address values (or their pointers) to reflect the data being returned.

Return Codes in the Communication Word

- 0 Normal processing
- 4 Skip this record. Return immediately to this External.
- 8 Process this record and then end the job.
- 12 Insert a Stop Code at the start of the document produced from this record. (Same as an ACTION STOP)
- 16 Start a new output tape reel before the document produced from this record. (Same as an ACTION FEOV)
- 20 End the job immediately without processing this record. (Same as an ACTION END)

External 34

This access point provides the External Procedure an opportunity to examine and/or modify the variable input records before they are processed by MailForm. If an External 33 is also specified, the call to External 34 is made right after the return from 33 (except where the return code from External 34 is a 4 or 20). The parameter list points to the record that has just been read. Prior to sending the call to External 34, MailForm normally translates all character fields in the record to ASCII to allow easier programming of the external. As was described previously, this can be inhibited by specifications in the INPUT statement. All character fields are consequently also translated back to EBCDIC when the external returns to MailForm - if a Translate 0 is defined for overall translation of the input record, this occurs after the conversion back to EBCDIC.

External 34 may change the record in any way it needs to. The length and starting address may be changed and the

data updated and rearranged. New fields may be added if desired. The converted record that is returned to MailForm must be placed in the common buffer. When an External 34 is defined, MailForm automatically provides a buffer that is twice the original record size. If more space is needed, the BUFFER parameter on the External Statement may be used.

Return Codes in the Communication Word

- 0 Normal processing of the returned data record.
- 4 Skip this record and go on to the next.
- 8 Process this record and then end the job.
- 12 Insert a Stop Code at the start of the document produced from this record. (Same as an ACTION STOP)
- 16 Start a new output tape reel before the document produced from this record. (Same as an ACTION FEOV)
- 20 End the job immediately without processing this record. (Same as an ACTION END)
- 24 Process the returned data record normally and then return to this External without reading another input record (Same as an ACTION REPEAT). The record passed on this second call is the same data returned by the last call, but if a Translate 0 was also specified, the data will have been translated by this table.

External 35

This access point also provides an opportunity to examine and/or modify the input record. The call point is later than with External 34 - just before the process of generation of variables begins. The major difference is that 34 occurs before any Translate 0 modifies the whole record and 35 occurs after the translate. Also 34 will be called when an ACTION READ is encountered, while 35 will not. External 35 will be called at the start of variable processing when there is no variable input file and also when the READ= parameter on the Options Statement is NO. The data

passed on this call is the latest record read. External 35 can modify the record in any desired manner.

Return Codes in the Communication Word

- 0 Normal processing of the returned data record.
- 4 Skip this record and go on to the next.
- 8 Process this record and then end the job.
- 12 Insert a Stop Code at the start of the document produced from this record. (Same as an ACTION STOP)
- 16 Start a new output tape reel before the document produced from this record. (Same as an ACTION FEOV)
- 20 End the job immediately without processing this record. (Same as an ACTION END)

External 36

This access point allows processing after all variables have been generated and before the document is built. When this call is made, MailForm has determined that a document will be output for this input record. The data passed at this call is a 32-bit word containing the document number. MailForm does not expect any returned data. The Communication Word is zero on the call and may not be changed except to send back log messages.

External 37

This External access point occurs when MailForm reaches End Volume on input tape reels. The data passed is the log message that MailForm will be outputting to the Log file (up to 120 bytes).

Return Codes in the Communication Word

- 0 Log the MailForm EOV message
- 4 Do not print the EOV message
- 8 Print the message and return immediately to this

External 38**External**

This External access point occurs when MailForm reaches End Volume on printer output tape reels. The data passed is the log message that MailForm will be outputting to the Log file (up to 120 bytes).

Return Codes in the Communication Word

- 0 Log the MailForm EOV message
- 4 Do not print the EOV message
- 8 Print the message and return immediately to this External

External 39

This External access point occurs at the end of the job and provides access to the End Job message and the final checkpoint record. The Pointer 5 address in the Parameter List at the call points to a copy of the Checkpoint Record. The Communication Word contains one of the following values when the External is called:

- 0 Job ended normally with no errors
- 1 Job was completed, but some errors were encountered
- 2 Job was terminated before completion because of errors
- 3 Job was terminated due to errors before writing any documents
- 4 Abnormal termination

Return Codes in the Communication Word

- 0 Log the MailForm EOJ message
- 4 Do not print the EOJ message
- 8 Print the message and return immediately to this External

External 40

This External access point provides access to the Checkpoint Record before it is written to the file. The data

length and pointer reflect a copy of the Checkpoint Record in the common buffer. The external procedure can add to the checkpoint record, but should not modify the data provided in the buffer, because this could affect the ability to restart MailForm successfully. The Communication Word contains zero.

MailForm checkpoint records are variable-length, because they must contain copies of the contents of all inheritable variables. User checkpoint data should be appended to the end of the record, and may be retrieved from there on a restart.

Return Codes in the Communication Word

- 0 Write the checkpoint record (as modified)
- 4 Skip this checkpoint
- 8 Write the checkpoint record, but do not call this routine for further checkpoints

Restarting

External 40 is called when MailForm is restarting the job from a Checkpoint. It is only called once, after the Checkpoint record has been read. The Communication Word on this call contains the value 8. The MailForm Checkpoint Record has the following format:

Byte		
0	Document Number	Last Document output before checkpoint
4	Input Record Number	Last record read before checkpoint
8	Records on Prev Vols	Number records on previous input volumes
12	Complete Input Vols	Number input tapes completely read
16	Complete Output Vols	Number output tapes completely written
20	Documents Skipped	Initial Skip before first document produced
24	Last Doc from Prevs Input Vol	Last output document before start of current input volume
28	Last Doc on Prevs Output Vol	Last complete output document on last full tape reel
32	Number Bytes Var Data	Number of bytes of inheritable variable data
36	(unused)	
	Inheritable Variable Data	

Word 9 (at byte 32) can be used to index to the start of the user data when restarting.