Command RESEGDEF is used to redefine Segments, either as a normal modification or as part of an error recovery attempt. The redefinition can be accomplished using the same file space as the original definition if the amount of storage for saving the parameter, time series, Operations Table and carryover information remains unchanged. This is known as an 'in-place' update. New file space is needed for Segment redefinition if the new information cannot fit in the space taken by the current definition. This is a 'displacing' update. In this situation, the old definition must be obsoleted and a new definition, in the same process as adding a Segment, must be added to the files.

The rest of this Sections describes:

- file modification and messages printed for an in-place update
- file modification and messages printed for a displacing update
- potential error recovery from command RESEGDEF
- messages printed during an error recovery from an abnormal program termination that happened during execution of a previous SEGDEF or RESEGDEF command

In-place Update

The following files are modified in the order listed:

File Name	Action Taken
FCPARAM	Write redefined parameter values to file
FCSEGSTS	Update Segment definition
FCCARRY	Replace carryover values with default values input with operations (for undated existing carryover), or adjust existing carryover with new parameter values and replace on file (for dated existing carryover)

The messages printed by the RESEGDEF command for an in-place update are for the parameter updates and the carryover updates. The parameter updates are those that change files FCPARAM and FCSEGSTS. The carryover updates are the changes made to file FCCARRY.

The first message printed before any file updates for the parameter section is:

```
*** BEGINNING FILE UPDATE FOR SEGMENT xxxxxxxx.
```

Next, the following messages are printed, just before and immediately after file updates for the 2 files:

```
*** BEGINNING UPDATE OF PARAMETER FILE.
*** SUCCESSFUL UPDATE OF PARAMETER FILE.
```

```
*** BEGINNING UPDATE OF SEGMENT STATUS FILE
*** SUCCESSFUL UPDATE OF SEGMENT STATUS FILE.
```

When this portion of the file updates is complete, the following is printed:

```
*** SUCCESSFUL FILE UPDATE FOR (RE)DEFINITION OF SEGMENT xxxxxxxx.
```

The first message printed in the carryover update section is:

```
*** BEGINNING CARRYOVER FILE UPDATE FOR SEGMENT xxxxxxxx.
```

Different update messages are printed for writing to the carryover file, depending on whether the existing carryover is dated or undated.

For undated carryover, the following pair of messages is printed:

```
*** DEFAULT CARRYOVER BEING PLACED ON SLOT nn WITH DATE mm/dd/yyyy-hrtzc.
*** DEFAULT CARRYOVER SUCCESSFULLY PLACED ON SLOT nn OF CARRYOVER FILE.
```

For dated carryover, the following four messages are printed:

```
*** SEGMENT REDEFINITION. CARRYOVER TRANSFER WILL BE DONE FOR SLOT nn
    WITH DATE mm/dd/yyyy-hrtzc.
*** CARRYOVER TRANSFER SUCCESSFULLY DONE.

*** TRANSFERRED CARRYOVER WILL BE WRITTEN ON SLOT nn FOR DATE
    mm/dd/yyyy-hrtzc.
*** CARRYOVER FOR SLOT nn SUCCESSFULLY WRITTEN.
```

When all carryover values have been successfully written to the file, the message printed is:

```
*** END OF CARRYOVER FILE UPDATE FOR SEGMENT xxxxxxxx.
```

Displacing Update

The following files are modified in the order listed:

File Name	Action Taken
FCSEGSTS	Set Segment name to 'OBSOLETE' in old location
FCPARAM	Set Segment name to 'OBSOLETE' in old location
FCCARRY	Set Segment name to 'OBSOLETE' in old location
FCPARAM	Write redefined parameters to file
FCCOGDEF	Update carryover file usage information
FCSEGPTR	Update Segment pointer information
FCSEGSTS	Write redefined Segment to file
FCCARRY	Write carryover input with Segment redefinition to file

(for undated existing carryover) or
Perform carryover transfer and write adjusted carryover to file (for dated existing carryover).

The messages printed by the RESEGDEF command for a displacing update are for the old definition deletion, the parameter updates and the carryover updates. The old definition deletion change the first 3 files listed above, the parameter updates change the next 4 files and the carryover update changes file FCCARRY.

The old definition deletion produces 3 messages, 1 before the update takes place and 2 after the old definition is set for deletion:

```
*** NOT AN IN-PLACE UPDATE.  DELETING OLD DEFINITION.
*** SUCCESSFUL DELETION OF OLD SEGMENT DEFINITION.
*** PROGRAM TERMINATION AT THIS POINT MAY DAMAGE FILE CONTENTS.
```

The first message printed before message printed before any file updates for the parameter section is:

```
*** BEGINNING FILE UPDATE IN FOR SEGMENT xxxxxxxx.
```

The following messages are printed just before and immediately after file updates of the 4 different files:

```
*** BEGINNING UPDATE OF PARAMETER FILE.
*** SUCCESSFUL UPDATE OF PARAMETER FILE.

*** BEGINNING UPDATE OF CARRYOVER GROUP DEFINITION FILE.
*** SUCCESSFUL UPDATE OF CARRYOVER GROUP DEFINITION FILE.

*** BEGINNING UPDATE OF SEGMENT POINTER FILE.
*** SUCCESSFUL UPDATE OF SEGMENT POINTER FILE.

*** BEGINNING UPDATE OF SEGMENT STATUS FILE.
*** SUCCESSFUL UPDATE OF SEGMENT STATUS FILE.
```

When this portion of the file updates is complete, the following is printed:

```
*** SUCCESSFUL FILE UPDATE FOR (RE)DEFINITION OF SEGMENT xxxxxxxx.
```

The first message printed in the carryover update section is:

```
*** BEGINNING CARRYOVER FILE UPDATE FOR SEGMENT xxxxxxxx.
```

For each slot on the carryover file, the following pairs of messages are printed; the first in each pair indicating updating to take place and the second indicating success of the update:

```
*** HEADER INFORMATION WILL BE WRITTEN ON SLOT nn OF CARRYOVER FILE.
*** HEADER INFORMATION NOW PLACED ON SLOT nn OF CARRYOVER FILE.

*** DEFAULT CARRYOVER BEING PLACED ON SLOT nn WITH DATE mm/dd/yyyy-hrtzc.
*** DEFAULT CARRYOVER SUCCESSFULLY PLACED ON SLOT nn OF CARRYOVER FILE.
```

For undated existing carryover, no other messages are printed.

For dated carryover, the following 4 messages are printed:

```
*** SEGMENT REDEFINITION. CARRYOVER TRANSFER WILL BE DONE FOR SLOT nn
    WITH DATE mm/dd/yyyy-hrtzc.
*** CARRYOVER TRANSFER SUCCESSFULLY DONE.

*** TRANSFERRED CARRYOVER WILL BE WRITTEN ON SLOT nn FOR DATE
    mm/dd/yyyy-hrtzc.
*** CARRYOVER FOR SLOT nn SUCCESSFULLY WRITTEN.
```

When all carryover values have been successfully written to the file, the message printed is:

```
*** END OF CARRYOVER FILE UPDATE FOR SEGMENT xxxxxxxx.
```

Error Recovery

The files can be restored after program failure in 2 cases:

1. When an abnormal program termination occurs after the old Segment definition has been deleted during a displacing redefinition. Since a Segment can be redefined while still part of either a Forecast Group or Carryover Group, different recovery methods are needed for the depending on if the Segment is referenced:

   - If the Segment does not belong to either a Forecast Group or a Carryover Group, the Segment can be defined (using SEGDEF since it no longer exists) after the cause of the failure has been resolved. If the carryover for the Segment had been dated, the dates are lost and the Segment must be re-dated (using the SEGDATE command) to restore the carryover dates.

   - If the Segment belongs to a Forecast Group, the Forecast Group must be deleted and redefined using the defined Segment definition. The Forecast Group must be deleted and the Segment must exist before the Forecast Group can be redefined using the redefined Segment. If the carryover for the Segment had been dated, the dates are lost and the Segment must be re-dated (using the SEGDATE command) to restore the carryover dates.

   - If the Segment belongs to a Forecast Group and a Carryover Group, both the Carryover Group and Forecast Group must be deleted and redefined after the Segment has been defined. After the Segment has been defined and the Carryover Group and the Forecast Group have been deleted, the Forecast Group can be redefined and the Carryover Group can be redefined. Since carryover dates must be assigned during a Carryover Group definition, no information for the Segment will be lost as long as the existing carryover dates are used to redefine the Carryover Group.

2. When an abnormal program termination after the parameter updates and before or during the carryover update. If processing has

not gone beyond the parameter update section, the Segment is unusable and the files must be restored using the most recent backup copy.

During carryover updating, the slots are set to incomplete until the updating is successful. Once a slot is marked incomplete, it must be restored to complete status before it can be dated (using either the CGDEF or SEGDATE command).

The first step to file restoration is determination of the cause of the program failure. Once this is done and the problem is resolved, the Segment redefinition deck should be resubmitted, again using the RESEGDEF command.

If the carryover that existed for the Segment on file before to the abnormal program termination is undated, no information is lost because of the first message printed before. If the Segment belongs to a Carryover Group, the carryover values are lost but the dates of carryover are retained as they can be found in the Carryover Group definition. If the Segment does not belong to a Carryover Group but is dated, both the carryover values and dates are lost since the dates of carryover in this case are held on the carryover file only and any information on the carryover file for that slot is ignored.

An alternative for file recovery is restoring the files from the most recent backup copy.

Error Recovery Output

No special output is produced for restoring the Segment in the first type of recoverable error (i.e. - when the Segment has been deleted).

If an abnormal program termination occurs during carryover updating for either SEGDEF or RESEGDEF command, the RESEGDEF command can be used to restore the files. The messages printed for an error recovery RESEGDEF are identical to a normal RESEGDEF submission, up to the carryover updates. The same output is produced through the deletion section (if the user changes the Segment definition during error recovery) and parameter update section.

During carryover updates, any incomplete slot encountered must be renewed as all information found in that slot is ignored. The first message printed in the carryover update section is:

```
*** BEGINNING CARRYOVER FILE UPDATE FOR SEGMENT xxxxxxxx.
```

For an in-place update, the following messages will be printed for each incomplete slot found:

```
**WARNING** SLOT NUMBER n FOR SEGMENT xxxxxxxx IS INCOMPLETE.
            NO CARRYOVER TRANSFER TO OCCUR.
            DEFAULT CARRYOVER WILL BE PLACED IN THIS SLOT WITH DATE
            mm/dd/yyyy-hrtzc.

*** HEADER INFORMATION WILL BE WRITTEN ON SLOT nn OF CARRYOVER FILE.
```

```
*** HEADER INFORMATION NOW PLACED ON SLOT nn OF CARRYOVER FILE.

*** DEFAULT CARRYOVER WILL BE WRITTEN ON SLOT nn FOR DATE
   mm/dd/yyyy-hrtzc.
*** CARRYOVER FOR SLOT nn SUCCESSFULLY WRITTEN.
```

For a displacing update, the following messages are printed for each incomplete carryover slot found:

```
*** HEADER INFORMATION WILL BE WRITTEN ON SLOT nn OF CARRYOVER FILE.
*** HEADER INFORMATION NOW PLACED ON SLOT nn OF CARRYOVER FILE.

*** DEFAULT CARRYOVER BEING PLACED ON CARRYOVER FILE ON SLOT nn WITH
    DATE mm/dd/yyyy-hrtzc.
*** DEFAULT CARRYOVER SUCCESSFULLY PLACED ON SLOT nn OF CARRYOVER FILE.

**WARNING** SLOT NUMBER n IS INCOMPLETE FOR SEGMENT xxxxxxxx.
            NO CARRYOVER TRANSFER WILL OCCUR.
```

With the successful completion of the resubmitted RESEGDEF command, the files are restored.