



CSI

INTERNATIONAL

Original Printing01/24/2003
Last Revised01/24/2003

TABLE OF CONTENTS

| | |
|--|-----------|
| GETTING STARTED | 1 |
| Introduction | 3 |
| Benefits | 4 |
| Compression Modes | 5 |
| Basic Compression (mode 1) | 5 |
| Basic+Numeric Compression (mode 3) | 5 |
| Basic+Alpha Compression (mode 5) | 5 |
| Full Compression (mode 7) | 6 |
| Compression Range | 7 |
| Component Description | 8 |
| Installation | 10 |
| GETMAIN Usage | 13 |
| | |
| USING M-Pact-PACK | 15 |
| File Analysis | 17 |
| VSAM File Considerations | 20 |
| VSAM KSDS File Definition Considerations | 20 |
| VSAM ESDS File Definition Considerations | 21 |
| Defining a M-Pact-PACK File | 22 |
| FSIPACTB Compression Table Generation | 23 |
| User Compression and Expansion | 25 |
| User Compression Sample Program | 27 |
| Questions and Answers | 29 |
| | |
| MESSAGES | 33 |

GETTING STARTED

Introduction

M-Pact-PACK addresses one of the major problems of today's computer installations - disk storage. With the ever increasing demands to keep more and more data on-line, computer installations are faced with the need to continually install more disk storage devices with even greater storage density and capacity. This situation leads to the inevitable problems of the cost of more disk drives, floor space for the new disk drives and controllers, air-conditioning upgrades, and the scheduling of long running backup jobs with their ever increasing number of magnetic tape reels. If you use VSAM and are not using M-Pact-PACK you are probably wasting valuable disk space. Many installations have files that contain records where only a small amount of the record is used, giving rise to large areas of unused space, and records not being efficiently blocked into VSAM Control Intervals and accordingly Control Areas.

M-Pact-PACK addresses these problems for Entry- and Key-Sequenced VSAM files by allowing the user to compress the data on selected files, and thus reduce the amount of space required to hold such files.

The amount which can be saved by compression depends on the contents and layout of the user's records, and the degree of compression chosen by the user. Compression of more than 70% has been achieved by M-Pact-PACK's sophisticated algorithms.

M-Pact-PACK includes a file analysis program which can be run against selected files to establish the potential compression ratio, and thus the value of using M-Pact-PACK for the file, and the site generally. The compression ratio for each of the four compression modes mentioned below is calculated for each file being analyzed.

M-Pact-PACK is fully transparent to programmers, operators, and application programs. Selected files are re-DEFINED and REPRO'd, after which the compression process is automatic and invisible.

Please note that RRDS files are not able to be compressed.

Programs are included for user application program special use, for example to compress sequential disk or tape files.

Benefits

As a consequence of installing M-Pact-PACK, the user should experience:

- Dramatic reductions in disk space requirements.
- Reduced disk arm movement due to high data compression.
- Substantially faster retrieval of data records when reading files sequentially.
- For KSDS files a reduction in index levels and index records.
- Reduction of CI/CA splits given the same percentage of freespace per CI/CA.
- More productive use of data transfer time, and possible reductions in channel contention.
- Reduced backup times for VSAM files and catalogs with a corresponding reduction in the number of tapes held in each cycle.

ALL of the above give rise to QUICKER on-line response times, FASTER batch throughput and BETTER utilization of virtual storage.

Compression Modes

Four degrees of compression are provided, to cater to the varying record contents of the user's files. Each file can be given a different compression mode depending on the contents of its records.

The M-Pact-PACK file analysis program FSIPACAN (see later) reports the differing percentage savings that may be achieved for each of the four modes, and can help the user to decide the optimum mode for each file, to achieve the best "mix" between disk saving and CPU overhead.

Very brief details of each of the four compression modes follow:

Basic Compression (mode 1)

This mode performs, amongst other things, compression of Space and Null characters, and repetitive characters. This mode is suitable for most files, and has the least CPU overhead.

Basic+Numeric Compression (mode 3)

This mode performs basic compression as described above and also applies compression algorithms to numeric data. The CPU overhead is greater than for mode 1, depending on the record content, and it may not be advisable to use this mode unless FSIPACAN shows an increased saving over mode 1 of over 5%.

Basic+Alpha Compression (mode 5)

This mode performs basic compression as described above and also applies compression algorithms to alpha data. The CPU overhead is similar to mode 3, depending on the record content, and it may not be advisable to use this mode unless FSIPACAN shows an increased saving over mode 1 of over 5%.

Full Compression (mode 7)

This mode performs all the above compression's, and therefore the CPU overhead is the greatest. Depending on the usage of the file, it may not be advisable to use this mode unless FSIPACAN shows an increased saving over mode 1 of over 10%.

The CPU overhead in compression/expansion is virtually impossible to predict, since it depends solely upon the contents of the record. However, tests have shown that basic compression (mode 1) gives rise to only a negligible overhead. Full compression could give rise to an overhead of up to 15 percent. The file analysis program (see later) will tell you whether the extra compression achieved by choosing an option other than "basic" (mode 1) will be worthwhile. Also, bear in mind that the I/O time saved by having records in a compressed form may well counteract the CPU overhead.

The compression mode is established when the file is DEFINE'd to VSAM, and is expressed by the letter "n" throughout this manual

Refer to section "Defining a M-Pact-PACK File" on page 22 for information on how to activate M-Pact-PACK for a particular VSAM dataset.

Compression Range

It is not usually possible to compress the entire record. For instance, on a KSDS, the key must not be compressed, since VSAM must see it as it really is, and in its correct position. Also, for a file with an alternate index, that portion of the record up to and including the alternate index key must not be compressed for the same reason.

In the case of an ESDS, the entire record may be compressed UNLESS the record will be updated. This is because the update of a byte in the compressed portion of a record may cause a record length change. This is not permitted by VSAM for an ESDS.

For "straight" KSDS with no alternate index, M-Pact-PACK will automatically calculate the end position of the key, and compress the record from that point up to the end of the record. For KSDS with alternate indexes, and updatable ESDS files, M-Pact-PACK has no way of knowing where to start compressing, and must therefore be told. Refer to section "Defining a M-Pact-PACK File" on page 22 for more information.

Component Description

The M-Pact-PACK system consists of the following components:

- FSIPACAN** This module is supplied to enable the user to analyze each problem VSAM file and to establish the compression ratios possible on the file. It reports whether compression was possible, and if so, reports the compression ratios achieved and the average compressed record length. For large files, FSIPACAN can be run against a user-specified number of records rather than the whole file. Just by running FSIPACAN against a few of your major files you will see at a glance the savings that can be made with M-Pact-PACK.
- FSIPACST** This module starts the M-Pact-PACK system. the user must ensure that FSIPACST is executed immediately after IPL. FSIPACST checks that the M-Pact-PACK components are present in the system, and triggers the VSAM intercept mechanism. After this phase has been executed, all VSAM calls are intercepted to see whether the call is related to a M-Pact-PACK file. If so, the appropriate compression or expansion is performed automatically.
- FSIPACCM** This module does the actual compression. It is loaded automatically by various M-Pact-PACK components. It is also used to perform user-controlled compression (see later).
- For VSE, this phase must be placed in the SVA.
- FSIPACEX** This module does the actual expansion, It is loaded automatically by various M-Pact-PACK components. It is also used to perform user-controlled expansion (see later).
- For VSE, this phase must be placed in the SVA.
- FSIPACCP** This module receives control for all calls to VSAM for record management from whatever source. If the file concerned is running under the control of M-Pact-PACK, the record is compressed on an output-type call, or expanded on an input-type call.

- FSIPACOP** This module receives control for all VSAM open requests. It determines if the VSAM dataset is compressed, and if so, alters the VSAM ACB to call FSIPACCP for every VSAM I/O request from the application.
- FSIPACCL** This module receives control for all VSAM close requests. It releases all storage and resources acquired by M-Pact-PACK for the VSAM dataset being closed..
- FSIPACK** This relocatable module is supplied to allow user-controlled compression and expansion. It performs the same functions as the automatic compression, however it allows the user to compress and expand files not currently supported by M-Pact-PACK. See section "User Compression and Expansion" on page 25 for more information.

Installation

M-Pact-PACK is distributed on a tape containing two datasets created by IEBCOPY. The first dataset contains pre-generated load modules to be transferred to your Load Library. The second dataset contains source modules and sample JCL to be transferred to your Source Library.

Load Library Modules:

| | |
|-----------------|-----------------------------------|
| FSIPACCM | Compression program. |
| FSIPACEX | Expansion program. |
| FSIPACK | User compression interface phase. |
| FSIPACAN | File analysis program. |
| FSIPACST | System startup program. |
| FSIPACCP | VSAM I/O interface program. |
| FSIPACOP | VSAM open interface program. |
| FSIPACCL | VSAM close interface program. |
| FSIPACTB | Default compression table. |

Source Library Modules:

| | |
|-----------------|-------------------------------------|
| FSIPACAS | Sample JCL to generate FSIPACTB |
| FSIPACTM | Macro required to generate FSIPACTB |
| FSIPACTB | Source for dummy FSIPACTB |

Installation Step 1

Create the following JCL which will load the distribution tape to your Source and Load Libraries.

```
//JOBID      JOB (accounting), 'LOAD M-Pact-PACK',  
//              CLASS=A,MSGCLASS=*,MSGLEVEL=(1,1)  
//*  
//*  LOAD M-Pact-PACK DISTRIBUTION TAPE  
//*  
//IEBCOPY EXEC PGM=IEBCOPY  
//SYSPRINT DD SYSOUT=*  
//TAPEF1    DD DSN='MPACT.PACK.LOADLIB',UNIT=TAPE,DISP=OLD,  
//              LABEL=(1,SL),VOL=(,RETAIN,SER=FSISYS)  
//TAPEF2    DD DSN='MPACT.PACK.SOURCE',UNIT=TAPE,DISP=OLD,  
//              LABEL=(2,SL),VOL=REF=*.TAPEF1  
//LOADLIB   DD DSN='authorized.linklist.library',DISP=SHR  
//SRCELIB   DD DSN='your.source.library',DISP=SHR  
//SYSIN     DD *  
            COPY I=((TAPEF1,R)),O=LOADLIB  
            COPY I=((TAPEF2,R)),O=SRCELIB  
/*  
//
```

Installation Step 2

The programs for M-Pact-PACK must be executed from an **APF Authorized Load Library**. To prevent the need for changes to all of your Batch JCL, the M-Pact-PACK Load Library should also be placed in the Linklist. This means that the library must be listed in both members, **IEAAPFxx** and **LNKLSTxx**, in **SYS1.PARMLIB**.

Installation Step 3

M-Pact-PACK uses an SVC Routing Facility to intercept VSAM OPEN and CLOSE processing. This facility allows one or more FSI products to intercept OPEN and/or CLOSE processing in a prescribed order. The Routing Facility is implemented transparently in M-Pact-PACK by means of the STRT and STOP parm of program FSIPACST.

The SVC Routing Facility is shipped separately on a tape labeled: **FSIVSR/MVS**. A cover letter should have been shipped with the tape that will explain how to install it.

Do NOT proceed to Installation Step 4 until you confirm that the SVC Routing Facility has been installed.

Installation Step 4

```
//FSIPACK JOB          activate M-Pact-PACK
//STEP1 EXEC PGM=FSIPACST,PARM='STRT'
//FSILIB DD DSN=authorized.linklist.library,DISP=SHR
//FSILPA DD DSN=SYS1.LPALIB(IFG0192A),DISP=SHR
//SYSPRINT DD SYSOUT=*
//
```

The DD statement "FSILIB" must point to the authorized library containing the M-Pact-PACK programs.

The JOB doing the STRT must be authorized for read access to 'SYS1.LPALIB' to verify the DFP and VSAM release levels.

The activation is accomplished by the following procedure:

- The Routing programs of FSIVSR/MVS are loaded into CSA storage and queued onto the CDE chain.
- The unloaded load module IDA0192A is read from SYS1.LPALIB. The CESD and RLD records are scanned for the VCONS for IDA019A and IDA0200T. The addresses of the routers are then inserted into these VCONS.
- The addresses of IDA0192A and IDA0200T are stored in a control section of the router.

Since this process does change internal control blocks within MVS, it is important that you test this in your environment during a non-production time period.

If you experience problems with your system after activating M-Pact-PACK, first attempt to deactivate it using the same JCL used to start M-Pact-PACK, except with a PARM='STOP' on the EXEC for FSIPACST.

After M-Pact-PACK has been activated, the compression table (FSIPACTB) will determine which VSAM datasets are to be compressed.

WARNING:

M-Pact-PACK should be started directly after IPL. The reason for this is that if M-Pact-PACK fails to startup for any reason, the operator can prevent any other jobs from starting, perhaps with dire consequences, in a M-Pact-PACK-less system, until the problem with the startup has been corrected (see also description of message FSIPACK-99).

GETMAIN Usage

M-Pact-PACK acquires region GETMAIN to use as a work area during the compression and expansion of records.

In VSE, M-Pact-PACK attempts to acquire a large area that will be used for the duration of the job. M-Pact-PACK first tries to acquire 33K. If this amount is not available, M-Pact-PACK reduces its request by steps of 1K, down to 9K. If this amount is not available, then M-Pact-PACK issues an error message, and returns an error code to the program.

No matter what size is actually obtained by M-Pact-PACK, data integrity is always assured. Also, most applications will not need anything near the maximum size to run without performance problems, and extra GETMAIN allocation will almost never be necessary.

In MVS, M-Pact-PACK acquires a storage area for each VSAM dataset at open. The size is based on the maximum record size of the dataset and the maximum number of strings specified in the ACB.

USING M-Pact-PACK

File Analysis

Run program FSIPACAN to establish the compression factor of your files:

```
//step1 EXEC PGM=FSIPACAN
//fname1 DD DSN='dsn.fname.1',DISP=SHR
//fname2 DD DSN='dsn.fname.2',DISP=SHR
.
.
//fnamen DD DSN='dsn.fname.n',DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
FILE=fname1,COUNT=count,HIKEY=kkkk,CTYPE=n
FILE=fname2,COUNT=count,HIKEY=offset,KEYLEN=11
.
.
FILE=fnamen
/*
```

where:

| | |
|-------|--|
| fname | is the file name of the file you wish to analyze, and relates to the corresponding DLBL/DD card. |
| count | is (optionally)the number of records you wish to analyze. If "count" is not specified, the entire file is analyzed. |
| CTYPE | is (optionally)the specific compression type to be analyzed. If omitted, all compression types will be analyzed, which will result in a longer run time. |
| kkkk | is the offset within the record that you want the compression routine to start compressing, as fully described above. If you don't specify the HIKEY=kkkk parameter, the default of zero is used, which is what you want for "straight" KSDS/ESDS. |

You can optionally specify the offset of the highest alternate index key for the HIKEY operand, and specify the length of the alternate index key in the KEYLEN operand.

Example control cards:

```
FILE=MASTER COUNT=1000
```

analyzes the first 1000 records of file "MASTER"

FILE=TRANS

analyzes the entire file "TRANS"

Control cards are completely free format. There is no limit to the number of files which can be analyzed in a single run.

It is important to analyze a representative sample of each file to accurately determine the compression factor.

The following information is reported for each file analyzed, and is repeated for each compression mode:

- Number of records analyzed, and number and percent which could be compressed
- Total byte count and average record length of the expanded (i.e., uncompressed) file
- Total byte count and average record length of the file should you decide to compress it
- Total, average, and percent saving that compressing the file would achieve

Sample Report Output:

| FILE NAME | COMP TYPE | RECORDS READ | RECORDS COMPRESSED | # OF BYTES BEFORE | # OF BYTES SAVED | # OF BYTES AFTER | AVG. LRECL BEFORE | AVG. LRECL AFTER | AVG. SAVINGS | PERCENTAGE SAVED |
|-----------|-----------|--------------|--------------------|-------------------|------------------|------------------|-------------------|------------------|--------------|------------------|
| OUTDD1 | 1 | 5480 | 5358 | 3233946 | 1319702 | 1914244 | 590 | 349 | 240 | 40 |
| OUTDD1 | 3 | 5480 | 5358 | 3233946 | 1335264 | 1898682 | 590 | 346 | 243 | 41 |
| OUTDD1 | 5 | 5480 | 5360 | 3233946 | 1413781 | 1820165 | 590 | 332 | 257 | 43 |
| OUTDD1 | 7 | 5480 | 5360 | 3233946 | 1410729 | 1823217 | 590 | 332 | 257 | 43 |

After a file is loaded under M-Pact-PACK, you may notice that the disk space used (High RBA) is more or less than FSIPACAN determined, even for the exact same data and a full file analysis. This is because the CISIZE may be a better or worse choice for the new average (compressed) record size. However, it should be fairly close to the projected savings, and the CISIZE can, of course, be adjusted.

VSAM File Considerations

When planning to install M-Pact-PACK it is important that each VSAM file to be controlled by M-Pact-PACK is reviewed. For each different file type there is a separate section following. Read the section carefully before conversion.

VSAM KSDS File Definition Considerations

VSAM KSDS files are fully supported by M-Pact-PACK. The section "Defining a M-Pact-PACK File" on page 22 details how to define the file, however, this section gives more details.

A VSAM KSDS is normally defined with only a DATA and INDEX component. In this case M-Pact-PACK will automatically compress the records from the end of the prime key to the end of the record. For example, if you specify that the key is 8 bytes long and start at position 1 of the record, the record length is 500, then the record will be compressed from position 9 up to position 500. The compression will be transparent to any user wishing to access the file at record level.

In the event of a program accessing the file by CI, there are two situations that can arise:

- a) The program is a utility (e.g. VSAM BACKUP/RESTORE) and it does not want to interrogate each individual record. The program will have each CI returned with COMPRESSED records in it, because M-Pact-PACK will not attempt to expand the records. It therefore follows that if the utility is backing a file up to tape that a corresponding reduction in tape usage will also be realized. When the file is restored again it will read COMPRESSED CI records from the tape and build the corresponding VSAM file in COMPRESSED format. This particular method gives maximum gain from M-Pact-PACK file compression.

- b) The program is user written and for reasons of performance, reads the file by CI, deciphers each CI and looks at each individual record. M-Pact-PACK will again pass back a CI that contains COMPRESSED records, so in this case the user program needs to be amended to call M-Pact-PACK to expand the compressed records, and if it changes records call M-Pact-PACK again to compress the records. As M-Pact-PACK compression can lead to a record length change if the data is changed, the program, if it rewrites at CI level, must be able to handle change of record length situations and all the subsequent consequences, for example CI splits. This type of user program is very rare.

If a KSDS file has an alternate index, then the definition to M-Pact-PACK is slightly different and so is the resulting compression percentage.

If the file is being defined using the FSIPACTB table, the entry for the above example of KEYS(10 50) would be defined with the FSIPACTB operands HIKEY=10 and KEYLEN=50.

The only exception to this rule is when the alternate index key is at a lower position than the prime key. In this case specify HIKEY=0 and KEYLEN=0 in the FSIPACTB table, as M-Pact-PACK will automatically calculate the end of the prime key.

IMPORTANT:

It must be remembered that if you wish to add or change an alternate index to an existing file then you must backup and DEFINE the file again with the correct FSIPACTB table HIKEY and KEYLEN operands before attempting to build the new alternate index.

VSAM ESDS File Definition Considerations

VSAM ESDS files are supported by M-Pact-PACK. See section "Defining a M-Pact-PACK File" on page 22.

If the ESDS file is never updated in place then specify HIKEY=0 and KEYLEN=0 for the FSIPACTB table definition for the ESDS file.

If any record on the file can potentially be updated by a program, then because of M-Pact-PACK's compression it could change the length of the record. A change of record length for this type of file is not allowed and would generate a VSAM error. To enable M-Pact-PACK to compress the file, specify the HIKEY and KEYLEN operands to define the portion of the record that can be updated. See earlier sections "Compression Modes" on page 5 and "Compression Range" on page 7 for detailed description of "n" and "kkkk".

Defining a M-Pact-PACK File

Once the analysis program FSIPACAN has been run and you are satisfied that M-Pact-PACK will reduce your disk space requirements you must convert the file to a M-Pact-PACK compressed dataset.

STEP 1

Backup or copy your selected VSAM file either to tape or to another disk area. You may not at this time use a backup program that reads VSAM files by Control Intervals, such as IBM's BACKUP and RESTORE. The suggested method would be to use VSAM's IDCAMS to REPRO the file to tape so that the file gets reorganized on the subsequent reloading. However you backup the file, it must be in a form that will allow for a restore of the file by logical record, not by CI.

STEP 2

- a) DELETE/DEFINE your file on the designated VSAM catalog.
- b) Add an entry to your sites M-Pact-PACK compression table FSIPACTB as described in the next section.
- c) Refresh your FSIPACTB table as described in the next section.

STEP 3

Reload your file from your VSAM backup. Remember, you must restore the file by logical record, not by CI. As the records are written to the newly defined file they will be compressed.

All access to the file is now being intercepted by M-Pact-PACK and records will be expanded automatically before control is returned to the user program.

FSIPACTB Compression Table Generation

M-Pact-PACK decides whether to compress a VSAM file or not depending on whether or not the cluster has been defined in the compression table FSIPACTB.

Sample FSIPACTB

```

1...5...10...15...20...25...30          ...72
FSIPACTB CSECT
          FSIPACTM DSNM=VSAM.NAME1
          FSIPACTM DSNM=VSAM.CLUSTER.NAME,          X
              DSNML=44,CTYPE=1,                    X
              HIKEY=4,KEYLEN=8
          FSIPACTM PGM=GVRESTOR
FSIEOT   DC    X'FF'          MARKS THE END OF THE TABLE
          END

```

Explanation

| | |
|--------|--|
| DSNM | is a dsname mask. This can be a complete dataset name, or the leading characters of a group of dataset names. |
| DSNML | is the length of the dsname mask specified in DSNM. If omitted, the actual length of DSNM is used. To ensure that a complete dataset name is not used as a mask, specify DSNML=44. |
| CTYPE | is the compression type to be used for the matching dataset(s). If omitted, CTYPE=1 is assumed. |
| HIKEY | is the highest relative key position of any alternate index associated with this base cluster. If omitted, HIKEY=0 is assumed. |
| KEYLEN | is the length of the alternate index key specified in HIKEY, or for an updatable ESDS, the last updatable byte position. |
| PGM | is the name of a PGM on an EXEC statement, that will have compression/decompression automatically suppressed by M-Pact-PACK. |

Activating a new table

After you have modified your FSIPACTB table, you must inform M-Pact-PACK that a new version is to be used. The following JCL can be used:

The table must be linked into the library containing the other M-Pact-PACK modules.

```
//STEP01 EXEC PGM=FSIPACST,PARM='REFR'
```

User Compression and Expansion

It is also possible to compress and expand records under user application program control, using the supplied interface. This facility may be particularly useful when creating non-VSAM files, such as sequential tape and disk.

The calling sequence is as follows:

BAL:

```
CALL FSIPACK, (function, exp-record, comp-record, length, max-  
length, offset), VL
```

COBOL:

```
CALL 'FSIPACK' USING function, exp-record,  
                      comp-record, length,  
                      max-length, offset.
```

where:

function is "E" for Expand, "A" "B" "C" or "D" for Compress with compression mode 1, 3, 5, or 7 respectively.

exp-record is the expanded record.

comp-record is the compressed record.

length is the length of the input (i.e. length of the compressed record for the Expand function, length of the expanded record for a Compress function). This field **MUST** be a two-byte binary field. The resulting length is returned in this field. For example, before a Compress operation, set this field to the actual length of the record. After the Compress, this field is set to the compressed length.

max-length is the maximum length of this record when NOT compressed. This value must be specified identically for all functions, that is, the same number must be used for Compress and Expand calls. This field **MUST** be a two-byte binary field. Normally, the maximum length would be a constant value for all compress and expand calls for a file, but technically it is only imperative that the same value be used for all calls for the same record.

`offset` is the offset, relative to zero, into the record that the compression/decompression is to start. This field MUST be a two-byte binary field.

The 'offset' operand is optional. If omitted, zero is assumed. If you are calling FSIPACK from a BAL program, and you omit the 'offset' operand, you must specify x'80' in the upper byte of the address pointer to the 'max-length' operand. The VL operand of the CALL macro automatically sets the x'80' bit in the last operand address pointer.

On return from the FSIPACK module, the function byte will be set as follows:

| | |
|-----|--|
| "0" | Call successful |
| "1" | Invalid function |
| "2" | GETMAIN macro failed |
| "3" | LOAD macro (of either FSIPACCM or FSIPACCM) failed |
| "4" | FREMAIN macro failed |
| "5" | 'Length' greater than 'Max-Length' |

Also, the length of the record after expansion or compression will be returned in the 'length' parameter.

User Compression Sample Program

The COBOL source coding below contains a part of a program which builds a (blocked) variable-length sequential file of records in M-Pact-PACK compressed format, using the FSIPACK called subroutine. Following that is replacement logic which could be used to expand the file for processing.

```
FILE SECTION.
FD  SQFILE
      LABEL RECORDS ARE STANDARD
      RECORDING MODE IS V
      BLOCK CONTAINS 16 RECORDS.
01  SQFILE-RECORD.
      05  SQFILE-RECLEN                PIC S9(3) COMP.
      05  SQFILE-RECDATA.
            10  FILLER                OCCURS 1 TO 400 TIMES
            DEPENDENT ON SQFILE-RECLEN PIC X.
WORKING-STORAGE SECTION.
77  PACK-FUNCTION                      PIC X.
77  PACK-LENGTH                       PIC S9(3) COMP.
77  PACK-MAXLENGTH                    PIC S9(3) COMP.
77  PACK-OFFSET                       PIC S9(3) COMP.
01  PACK-RECORD                       PIC X(400).
PROCEDURE DIVISION.
OPEN-FILES.
      OPEN OUTPUT SQFILE.
MAIN-LOOP.
      MOVE 'A' TO PACK-FUNCTION.
      MOVE 400 TO PACK-MAXLENGTH.
      MOVE 400 TO PACK-LENGTH.
      MOVE 0 TO PACK-OFFSET.
      COMPRESSION MODE 1 OF A 400-BYTE RECORD.
*
*
*   'PACK-RECORD' WOULD BE LOADED HERE WITH 400 BYTE
*   COMPRESSED DATA RECORD.
*   LOOP UNTIL INPUT SOURCE EXHAUSTED.
*
      CALL 'FSIPACK' USING PACK-FUNCTION, PACK-RECORD,
                        SQFILE-RECDATA, PACK-LENGTH,
                        PACK-MAXLENGTH, PACK-OFFSET.
      IF PACK-FUNCTION NOT EQUAL '0',
        DISPLAY 'PACK-ERROR, FUNCTION = ', PACK-FUNCTION,
        GO TO END-RUN.
      MOVE PACK-LENGTH TO SQFILE-RECLEN.
      WRITE SQFILE-RECORD.
      GO TO MAIN-LOOP.
```

The statements below could be used to expand the file compressed above:

```
OPEN INPUT SQFILE.
MAIN-LOOP.
  READ SQFILE AT END GO TO END-RUN.
  MOVE SQFILE-RECLLEN TO PACK-LENGTH.
  MOVE 'E' TO PACK-FUNCTION.
  MOVE 400 TO PACK-MAXLENGTH.
  MOVE 0 TO PACK-OFFSET.
* EXPAND PREVIOUSLY COMPRESSED FILE OF 400-BYTE RECORDS.
* EXPANDED RECORD PUT INTO 'PACK-RECORD' AREA.
  CALL 'FSIPACK' USING PACK-FUNCTION, PACK-RECORD,
    SQFILE-RECDATA, PACK-LENGTH,
    PACK-MAXLENGTH, PACK-OFFSET.
  IF PACK-FUNCTION NOT EQUAL '0',
    DISPLAY 'PACK-ERROR, FUNCTION = ', PACK-FUNCTION,
    GO TO END-RUN.
  GO TO MAIN-LOOP.
```

Questions and Answers

A selection of common questions and answers posed by "M-Pact-PACK" users.

How will I see my records in a dump?

The data in your program's I/O areas will never be in COMPRESSED format. Compression and expansion takes place in areas acquired by M-Pact-PACK. However, if you delve into the depths of VSAM, you may see a compressed control interval.

How can I send data to another installation?

To de-compress a file to tape, simply REPRO it. The de-compression will take place automatically. Similarly, should you wish to de-compress a file onto disk for any reason, again use REPRO. If the output file has not been defined with an EEXT in M-Pact-PACK format, de-compression will take place.

How can I tell if a file is compressed?

This is a tricky one! The system was designed to be COMPLETELY transparent to the user. One way is to LISTCAT the file to see whether it has an EEXT of M-Pact-PACK format. You could ALTER the file to have an EEXT of non-M-Pact-PACK format, and then use a VSAM PRINT command. The file will then be printed in its "raw" state.

I read a file in a SORT exit. Will it work on a file compressed by M-Pact-PACK?

Yes. You will be returned an expanded record.

How can I gauge the anticipated CPU overhead of M-Pact-PACK?

You can't. The overhead is dependent on the compression mode chosen, and the contents of the record. In our experience, Basic compression will involve the least overhead. Full compression could involve an overhead of as much as 15% under extreme circumstances. However, the expansion process is much, much, faster than compression, so if a file is read substantially more than it is written, the overhead will be diminished. In many circumstances, the CPU overhead involved will be more than compensated for by the decrease in I/O time.

Do I need to allocate extra storage to run M-Pact-PACK?

As previously stated, M-Pact-PACK expands and compresses records into its dynamic work areas. These work areas are acquired by "GETVIS" as necessary. The program attempts first to acquire an area of length 33K. This is more than adequate in normal circumstances for all M-Pact-PACK requirements. If this amount of (contiguous) storage is not available in a partition, M-Pact-PACK dynamically lowers its request, until storage CAN be obtained. This lower allocation may be at the expense of performance. Should M-Pact-PACK not be able to acquire any GETVIS, a message is displayed on the console, and the VSAM return code indicating "Insufficient GETVIS" is returned to the user, exactly as if VSAM itself had detected the error. Data integrity is assured.

How do I add an alternate index to a M-Pact-PACK file?

If the new alternate index key finishes at a lower position than the prime key, and any existing alternate indexes, just build the new index as normal. If the new index is outside this range, REPRO the file to another disk or tape, re-DEFINE the file with a different EEXT parameter based on the new AIX position, REPRO the file back in, and build the new index. Read section "Compression Range" on page 7 for full explanation.

What if M-Pact-PACK won't start up?

This is a MOST unlikely event. In a testing environment, or one that has few, unimportant, compressed files, this may not be a problem. However, in a production environment whose files are heavily compressed, it could be pretty disastrous to let the system continue, since records read will not be expanded etc. In the former case, the operator should reply "YES", to allow the system to continue. In the latter case, however, he or she should ascertain the correct course of action from their technical support people.

What if I need to DITTO print a M-Pact-PACK file?

No problem. Use DITTO functions VDP and VPR as normal and DITTO will print the record in expanded format. If however, you use DITTO to print from a physical disk location using functions DP, DD, DPD, or DDD then the resulting print will show the records in compressed format.

Why did I get less (more) compression than FSIPACAN projected?

This is because the CISIZE may be a better or worse choice for the new average (compressed) record size. However, it should be fairly close to the projected savings, and the CISIZE can, of course, be adjusted.

MESSAGES

- FSIPACK-1** **RELEASE x.xx ACTIVATED [EXPIRES IN nn DAYS]**
Program FSIPACST has successfully started the M-Pact-PACK system. The expiration note will appear if the product is within 30 days of expiration. Contact support representative.
- FSIPACK-98** **M-Pact-PACK HAS EXPIRED**
Program FSIPACST cannot startup due to the product having expired. (Verify system date is correct.) Contact support representative immediately.
- FSIPACK-99** **RUN TERMINATING - ENTER "YES" TO CONTINUE**
Program FSIPACST has detected an error condition and is now terminating. See previous message for reason for termination. In a testing environment, or one that has few, unimportant, compressed files, this may not be a problem. However, in a production environment whose files are heavily compressed, it could be pretty disastrous to let the system continue, since records read will not be expanded etc. In the former case, the operator should reply "YES", to allow the system to continue. In the latter case, however, he or she should ascertain the correct course of action from their technical support people.
- FSIPACK-101** **ANALYSING FILE xxxxxxxx, RECORDS READ nnnnnn**
Program FSIPACAN has issued this message in response to an interrupt to the partition (e.g., MSG pp). xxxxxxxx is the name of the file currently being processed and nnnnnn is the number of records read so far.
- FSIPACK-102** **ENTER OPTION :- C=CONTINUE,S=SKIP TO NEXT FILE,E=END**
Program FSIPACAN has issued this message in response to an interrupt to the partition (MSG pp). To continue processing enter "C", if you wish to go on to the next file for analysis enter "S". To stop the run enter "E". In either of the last two cases the analysis so far will be printed on SYSLST.
- FSIPACK-110** **MODULE xxxxxxxx NOT FOUND**
Program FSIPACAN cannot load phase xxxxxxxx, because it is not in an available library. Correct the problem and rerun.
- FSIPACK-111** **VSAM ERROR, PREVIOUS LINE EXPLAINS THE ERROR**
A VSAM error such during a GENCB, SHOWCB, or OPEN has occurred. The exact error is displayed in the previous text line. Request technical assistance for this error.
- FSIPACK-112** **SYNVSAM ERROR**
A VSAM physical error has occurred. Request technical assistance for this error.

| | |
|--|---|
| FSIPACK-113 | LERVSAM ERROR A VSAM logical error has occurred. Request technical assistance for this error. |
| FSIPACK-114 | SYSIN ERROR, xxxxxxxxxxxxxxxxxxxxxxxxx A parameter error was detected in the input. The text of the message will describe the exact error detected. Correct the problem and rerun. |
| FSIPACK-208 | VSAM CONTROL BLOCKS NOT SETUP Program BIMPACSV has found an error in the VSAM control blocks. Ensure the file was opened correctly, before a VSAM request was issued. If problem persists dump the problem program after the file has been opened and contact your support representative. |
| FSIPACK-211 | UNABLE TO FREEVIS I/O AREAS Program BIMPACSV has received an unexpected return code when trying to release its I/O areas. If problem persists dump partition and contact your support representative. |
| FSIPACK-212 FSIPACK-213 FSIPACK-214 | UNABLE TO FREE WORK AREA UNABLE TO FREE BUFFER AREA UNABLE TO FREE WORK AREA Program BIMPACSV has received an unexpected return code when trying to release area one of its control areas. If problem persists dump partition and contact your support representative. |
| FSIPACK-215 | UNSUPPORTED CALL VIA IIP Program BIMPACSV has received a "PUT LOCATE" type call from the user program. These types of calls are generated by the ISAM Interface Program (IIP), and are not supported by M-Pact-PACK. The request is aborted, and a physical data-write error code is passed back to the caller. |
| FSIPACK-300 | FSIPACTB TABLE NOT FOUND Table FSIPACTB cannot be loaded, because it is not in an available library. Correct the problem and rerun. |
| FSIPACK-301 | TABLE NOT FOUND ON JPQ Request technical assistance for this message. |
| FSIPACK-302 | FREEMAIN ERROR, (description of error) Request technical assistance for this message. |
| FSIPACK-303 | GETMAIN ERROR, (description of error) Request technical assistance for this message. |

| | | | | |
|--------------------|---|----------------|---------------|------------------|
| FSIPACK-304 | NO BUFFER ASSOCIATED WITH RPL Request technical assistance for this message. | | | |
| FSIPACK-305 | COMPRESS TYPE NOT VALID Valid compression types are:1, 3, 5, and 7. Correct the problem and rerun. | | | |
| FSIPACK-307 | MODULE xxxxxxx NOT FOUND Module xxxxxxx cannot be loaded because it is not in an available library. Correct the problem and rerun. | | | |
| FSIPACK-308 | SHOWCB ERROR, (description of error) Request technical assistance for this message. | | | |
| FSIPACK-309 | NO PARM SPECIFIED ON EXECUTE CARD FSIPACST requires a PARM to be specified. Correct the problem and rerun. | | | |
| FSIPACK-310 | INVALID PARM SPECIFIED FSIPACST requires a PARM to be specified. Correct the problem and rerun. | | | |
| FSIPACK-311 | TABLE CANNOT REFRESH, NOT LOADED Attempting to use PARM='REFR' with FSIPACST while M-Pact-PACK is not started. Use PARM='STRT' to start M-Pact-PACK instead. | | | |
| FSIPACK-312 | TABLE ALREADY LOADED Attempting to use PARM='STRT' with FSIPACST while M-Pact-PACK is already started. Use PARM='REFR' to refresh the FSIPACTB table. | | | |
| FSIPACK-314 | ERROR RETURNED FROM ROUTER, (description of error) Request technical assistance for this message. | | | |
| FSIPACK-315 | SUBPOOL NOT FOUND Request technical assistance for this message. | | | |
| FSIPACK-316 | M-Pact-PACK RELEASE x.xx HAS BEEN <table border="1"><tr><td>STARTED</td></tr><tr><td>STOPED</td></tr><tr><td>REFRESHED</td></tr></table> This is an informational message in response to a FSIPACST execution to confirm that the requested function has been performed. | STARTED | STOPED | REFRESHED |
| STARTED | | | | |
| STOPED | | | | |
| REFRESHED | | | | |
| FSIPACK-400 | INVALID FUNCTION SPECIFIED An invalid function has been provided to the FSIPACK user program interface. Correct the problem and rerun. | | | |

FSIPACK-401**LENGTH GREATER THAN MAX LENGTH**

A record length greater than the maximum define to VSAM for the dataset has been provided to the FSIPACK user program interface. Correct the problem and rerun.

FSIPACK-402**MODULE xxxxxxx NOT FOUND**

Module xxxxxxx cannot be loaded because it is not in an available library. Correct the problem and rerun.