APPENDIX I

Instructions for Use of FIRNU001 and FIRNU002

With the invention of magnetic storage media (tapes and disk drives) data records were no longer restricted to the 80 bytes of the punched card. As a result, today virtually all data fields which contain a useful amount of data within a single record use record lengths exceeding 80 bytes. These files cannot be moved via NJE/NJI/RJE facilities without first converting the records to card images. After the card image data has been moved to the destination site the card images can be used to recreate the file in its original format.

Two programs are available for these purposes. FIRNU001 will perform both functions. Depending upon parameters specified on a control card FIRNU001 will either:

1) read a sequential data file with either fixed or variable length records (up to 6400 bytes), segment the input records into card images (80 byte records), and produce an output file consisting of the card images and delimiter records, or

2) read a file produced by a previous execution of FIRNU001, use the information on the delimiter records to build records from card image data, and produce an undefined format output file consisting of 6400 byte blocks.

A utility program such as IEBGENER can read the undefined format file and produce an output file with the same characteristics as the original file.

FIRNU002 is an extremely short COBOL program which is intended to be executed as a "compile-and-go" run; this allows the programmer to explicitly define the characteristics of the output file. After the program has compiled it will read a file produced by FIRNU001, build records from card images, and produce an output file with the format defined in the program.

FIRNU001 and FIRNU002 are both Version 4 COBOL programs. COBOL was chosen to maximize program portability. The source code for both resides in DPS.DISTRICT.SOURCE. FIRNU001 is compiled resident at NWRDC in DPS.DISTRICT.LINKLIB.

<u>Use of FIRNU001</u>

FIRNU001 will run in "transmit" (create card images) or "receive" (read card images) mode depending on parameters specified on the SYSIN control card. The syntax of this card is:

```
          111111111
1234567890123456789
m rcf 9999
```

m = mode: "T" for transmit, "R" for receive
rcf = file format: "FIX" for FB, "VAR" for VB
9999 = Blocksize for FB files: 0081 to 6400

Mode is always required. RCF is required for "T" runs. Blocksize is required for "FIX" runs. The following are examples of valid syntax:

```
T FIX 0200
T,FIX,0095
R VAR
R
```

When executing in "T" (Transmit) mode the input to FIRNU001 must be either a VB (Variable record length, Blocked) file on tape or a FB (Fixed record length, Blocked) file on disk or tape. The FB file must contain only one record per block; a utility program such as IEBGENER can be used to reblock the file (see following example). The blocksize of the input file cannot exceed 6400 bytes for both VB and FB files. Output from FIRNU001 is a card image file which uses "blockheader" records to separate the card image groups from one another (each group of card images corresponds to a single record in the input file). After the card image file has been created it can be moved using IEBGENER (or similar method) as described in the previous section.

The following is an example of a FIRNU001 job which creates a card image file. Note that STEP1 is an IEBGENER step which reads the input file and creates a temporary, one-record- per-block data set which is passed to the FIRNU001 step, STEP2. The original data file is a catalogued data set on disk with the name ORIGIN.DATA, with record lengths of 175 bytes, and which occupies ten tracks on USER01, the diskpack upon which reside all of the example's data sets. The job executes at an OS/MVS/JES2 site, FH1. FIRNU001 resides in a library named USER.LINKLIB. The output from FIRNU001 is a file with the name CARD.IMAGE.DATA.

```
//jobname JOB                          (valid account number and syntax)
/*ROUTE PRINT FH1.RO                          (route to FH1 local)
/*PASSWORD PPPP                               (valid password and syntax)
//STEP1 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSIN DD DUMMY
//SYSUT1 DD DSN=ORIGIN.DATA,DISP=SHR  (cataloged disk file)
//SYSUT2 DD DSN=&&TEMP,DISP=(NEW,PASS),UNIT=SYSDA,
//      SPACE=(TRK,(10,1)),DCB=(RECFM=FB,LRECL=175,BLKSIZE=175)
//STEP2 EXEC PGM=FIRNU001

//STEPLIB DD DSN=USER.LINKLIB,DISP=SHR
//SYSIN DD *
T FIX 0175
//JCLIN DD DATA,DLM='##'
$$$$$$$$$$                                                  (required by program)
##
//DATAIN DD DSN=&&TEMP,DISP=(OLD,DELETE)
//TRANOUT DD DSN=CARD.IMAGE.DATA,DISP=(NEW,CATLG),UNIT=SYSDA,
```

```
//      VOL=SER=packname,SPACE=(TRK,(10,1)),
//      DCB=(RECFM=FB,LRECL=80,BLKSIZE=80)
//TRANIN DD DUMMY
//DATAOUT DD DUMMY
```

When running in "R" (Receive) mode input to FIRNU001 must be a card image file created by a previous execution of FIRNU001. Output is an undefined format file with one record per block and a blocksize of 6400 bytes. It is strongly recommended that when running FIRNU001 in receive mode that the output file be passed to a utility program which can read the DATAOUT file records and write output records using only the part of the DATAOUT record which contains data (see following example).

The following is an example of a FIRNU001 job running in receive mode. STEP1 is a FIRNU001 step which reads an input file, CARD.IMAGE.DATA, and generates a temporary one-record-per-6400-byte-block data set which is passed to an IEBGENER step, STEP2. CARD.IMAGE.DATA was originally created by a previous execution of FIRNU001 at another site and was copied to its current location using one of the methods described in section 2.2. The SYSIN DD file for STEP1 is control information for IEBGENER and instructs the program to perform the following actions: when reading an input record the 175 bytes beginning in byte 1 are to be treated as a single field. This field is to be written to the output record with the first byte of the field aligned with the first byte of the output record. The job executes at an OS/MVS/JES2 site, FH2. FIRNU001 resides in a library called FH2.UTILITY.LINKLIB. The output from the IEBGENER step is a file named DEST.DATA.

```
//jobname JOB                                   (valid account number and syntax)
/*ROUTE PRINT FH2.RO                            (route to local)
/*PASSWORD PPPP                                 (valid password and syntax)
//STEP1 EXEC PGM=FIRNU001
//STEPLIB DD DSN=FH2.UTILITY.LINKLIB,DISP=SHR
//SYSIN DD *
R
//JCLIN DD DUMMY
//DATAIN DD DUMMY
//TRANOUT DD DUMMY
//TRANIN DD DSN=CARD.IMAGE.DATA,DISP=SHR            (cataloged disk file)
//DATAOUT DD DSN=&&TEMP,DISP=(NEW,PASS),UNIT=SYSDA,

//          DCB=(RECFM=U,LRECL=6400,BLKSIZE=6400),
//          SPACE=(TRK,(90,10))
//STEP2 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
      GENERATE MAXFLDS=1
      RECORD  FIELD=(175,1,,1)
```

```
//SYSUT1 DD DSN=&&TEMP,DISP=(OLD,DELETE)
//SYSUT2 DD DSN=DEST.DATA,DISP=(NEW,CATLG),UNIT=SYSDA,
//          SPACE=(TRK,(10,1)),DCB=(RECFM=FB,LRECL=175,BLKSIZE=175),
//          VOL=SER=PACK001
```

Use of optional JCLIN file

If the destination of the data is a NODE to which jobs can be routed, the user may wish to use the JCLIN file option. The JCLIN allows the data to be moved to its new location with a single user operation. This is accomplished in the following way:

1. An "R" mode jobstream is merged into the "T" jobstream between the //JCLIN DD DATA,DLM=## statement and a ## statement (the ## is a delimiter).

2. The TRANOUT DD statement for the "T" job is altered to route the FIRNU001 output to the internal reader:

```
//TRANOUT DD SYSOUT=(,INTRDR)
```

3. A /*ROUTE XEQ Nn must be added to the "R" job.

4. The TRANIN DD for the "R" job is altered and two delimiter records follow it:

```
//TRANIN DD DATA,DLM=@@
$$$$$$$$$$
@@
```

The $$$$$$$$$$ record is used by FIRNU001 to determine where the card image data are to be merged.

The following example shows how the operations performed in the two previous examples can be performed with one operation:

```
(note that both FH1 and FH2 are JES2 sites)
//jobname JOB                    (valid account number and syntax for FH1)
/*ROUTE PRINT FH1.RO
/*PASSWORD PPPP                  (valid password and syntax for FH1)
//STEP1 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSIN DD DUMMY
//SYSUT1 DD DSN=ORIGIN.DATA,DISP=SHR                (cataloged disk file)
//SYSUT2 DD DSN=&&TEMP,DISP=(NEW,PASS),UNIT=SYSDA,
//          SPACE=(TRK,(10,1)),DCB=(RECFM=FB,LRECL=175,BLKSIZE=175)
//STEP2 EXEC PGM=FIRNU001
//STEPLIB DD DSN=USER.LINKLIB,DISP=SHR
```

```
//SYSIN DD *
T FIX 0175
//JCLIN DD DATA,DLM='##'
//jobname JOB                    (valid account number and syntax for FH2)
/*ROUTE XEQ FH2
/*ROUTE PRINT FH1.RO             (print output routed back to FH1 local)
/*PASSWORD                       (valid password and syntax for FH2)
//STEP1 EXEC PGM=FIRNU001
//STEPLIB DD DSN=N2.UTILITY.LINKLIB,DISP=SHR
//SYSIN DD *
R
//JCLIN DD DUMMY
//DATAIN DD DUMMY
//TRANOUT DD DUMMY
//TRANIN DD DATA,DLM=@@
$$$$$$$$$$
@@
//DATOUT DD DSN=&&TEMP,DISP=(NEW,PASS),UNIT=SYSDA,
//          DCB=(RECFM=U,LRECL=6400,BLKSIZE=6400),
//          SPACE=(TRK,(90,10))
//STEP2 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
          GENERATE MAXFLDS=1
          RECORD FIELD=(175,1,,1)
//SYSUT1 DD DSN=&&TEMP,DISP=(OLD,DELETE)
//SYSUT2 DD DSN=DEST.DATA,DISP=(NEW,CATLG),UNIT=SYSDA,
//          SPACE=(TRK,(10,1)),DCB=(RECFM=FB,LRECL=175,BLKSIZE=175),
//          VOL=SER= PACK001
##
//DATAIN DD DSN=&&TEMP,DISP=(OLD,DELETE)
//TRANOUT DD SYSOUT=(,INTRDR)
//TRANIN DD DUMMY
//DATAOUT DD DUMMY
```

Use of FIRNU002

FIRNU002 is a modified version of FIRNU001 which reads a card image file generated by FIRNU001 and produces an output file with record lengths determined by the user prior to compilation of FIRNU002. FIRNU002's output file contains one record-per-block.

When used as a utility FIRNU002 must be executed as a "compile-and-go" run (this is not as bad as it sounds; FIRNU002 is very short and compiles in .4 CPU seconds on an IBM 3081). The user must utilize an interactive editor to insert a numerical value on the PIC X(_) clause for the DATAOUT record definition and must save the modified copy of the program or merge it into a jobstream.

The following is an example of a utility execution of FIRNU002 at FH2. The COBOL compile and go proc is locally defined as COBUCG.

```
//jobname JOB                                  (valid account number and syntax)
/*ROUTE PRINT FH2.RO
/*PASSWORD PPPP                                (valid password and syntax)
//   EXEC COBUCG
//COB.SYSIN DD DSN=FIRNU002.TEMP.SOURCE,DISP=SHR (altered program)
//GO.TRANIN DD DSN=CARD.IMAGE.DATA,DISP=SHR  (cataloged disk file)
//GO.DATAOUT DD DSN=NEW.DATA,DISP=(NEW,CATLG),UNIT=SYSDA,
//          DCB=(RECFM=FB,LRECL=175,BLKSIZE=175),VOL=SER=PACK001,
//          SPACE=(TRK,(10,1))
```

FIRNU002 can also be used in production operations wherein the output file always has the same record size. In these situations the program is modified, compiled and stored in a program library using a name <u>other</u> than FIRNU002.