EDIT

GENERAL DESCRIPTION AND HOW TO USE THE PROGRAM

G. Kellner
CERN, Geneva, Switzerland

TABLE OF CONTENTS

1. INTRODUCTION
   1.1 What is EDIT
   1.2 Why use EDIT
   1.3 Why use FQX/FQT format
   1.4 How does EDIT work

2. HOW TO USE PROGRAM EDIT
   2.1 Selection of program version
   2.2 I/O units and control of I/O files
   2.3 FQX/FQT format

3. SET-UP OF PRODUCTION VERSION
   3.1 Definition of user header information
   3.2 Definition of structural relation of HYDRA banks
   3.3 Definition of valid action codes
   3.4 Subroutines to be supplied by the user
   3.5 Title DS
   3.6 Title AC
   3.7 Title MC
   3.8 Title RQTR

4. ACTION REQUESTS TO CONTROL FLOW OF EDIT
   4.1 Action card input
   4.2 User supplied input routines

5. SHORT COMMENTS ON PROGRAM
   5.1 General remarks
   5.2 Format of action requests at various stages of processing
   5.3 Details of information for action requests

CERN/EP/DHR 78-3
(Rev. version of 77-3)
12 September 1978
1. INTRODUCTION

1.1 What is EDIT

EDIT is a general program to manipulate HYDRA data structures written
in FQX/FQT record format onto tape or disk files. Information in the HYDRA
banks or in the user header vector can be added, deleted or updated from
external control information provided by the user from cards, terminal in-
put or other sources.

Any HYDRA data structure may be treated by EDIT, the structural
relations of banks are specified in a title block. A set of actions has
been coded and is provided on the PAM files, but any additional procedures
needed by the user can easily be integrated. EDIT is written with HYDRA
conventions. A global view of the organization of the program is given
in Appendix 1.

1.2 Why use EDIT

As a practical show-case for the use of EDIT we will consider the
data structure as produced by the HYDRA geometry applications program.
All examples in the text will refer to this structure, shown in Appendix 2.

Depending on the sophistication of the measuring system for the bubble
chamber pictures, the output from the geometry program will be more or less
complete. Additional interventions may be necessary to:
- remeasure part of an event or add additional measurements into the
  existing structure,
- resolve ambiguities and discard the unwanted solution(s),
- add information on classification of the event after inspection,
- add extra information not available from/or at the measurement stage,
- add statistics and bookkeeping information together with the measurements.

The important fact is that the edited information for any set of
films will be the most complete information available up-to-date and that
this information is available on a unique file and not dispersed onto
various tape and disk files or printer output. EDIT can of course inter-
vene at many different phases of the analysis.
1.3 Why use FOX/FQT format

The most important feature of this format is that automatic conversion to/from IBM 370 32-bit VBS format will be done on non-IBM computers whenever FQT is substituted for FOX input/output. This certainly solves the problems of tape interchange on computers with different word lengths and different data representation on tape/disk. The small penalties one has to pay are:

- **NO BCD information** (HYDRA bank names are correctly converted, however),
- **integers** must be 24 bits maximum (first 8-bit byte non-zero means conversion as floating point number).

The general lay-out of FOX/FQT records is shown in Appendix 3. One notes that the record consists of a precursor record and one to several physical records with HYDRA bank information.

The precursor record contains control information for the reading routines and the user header vector of up to 400 words. The precursor record can be read separately and a decision can be taken if the remainder of the information should be read or skipped. It is possible to have precursor records without HYDRA banks. This is very handy if one wants to keep special information on the file.

The user header vector can contain general information to identify the records without the need to find this information from HYDRA banks. It can contain bookkeeping and statistics information in standard words or bitted information. This can be quite powerful to take quick decisions if records should be selected or skipped in any subsequent analysis programs.

1.4 How does EDIT work

Details on how to set up the program, select available options, prepare program decks etc. will be discussed explicitly in the following sections. Details on the program and on control of input/output streams are given in the last section. Here we will only give a short outline to understand the basic operation:
- Action requests (in the form of action cards or user supplied information) in any number and any order will be sorted in increasing reference numbers (roll, frame or others) and stored on an intermediate file.

- The input consists of one or several tape or disk files of the Master Tape (MT) with FOX/FQT records. Processing order is the order on MT, the intermediate file with action requests is searched forward/backward if needed. A certain order of reference numbers on MT is recommended (i.e. order in increasing frame numbers within a roll, order of rolls may be random).

- New records (frames or rolls) will be inserted at the proper place.

- Information to be merged with the input MT can be given on a second MT (single file only). The merging of partial or complete data structures (replace completely an existing frame or insert a new frame) is entirely controlled through the action requests.

- Records will be selected, deleted, copied, updated or converted according to the action requests.

- The output MT will be written under user control, in addition precursor records can be split-off for statistics analysis.

- Extensive log information and error messages on actions performed (or failed) is printed.

2. HOW TO USE PROGRAM EDIT

2.1 Selection of program version

The program has to be assembled via PATCHY from the following PAM files:

- **EDITPAM** - complete PAM for program EDIT.
- **GENUTYPAM** - to load several user routines.
- **HYDRACDEPAM** - to load HYDRA CDE decks and machine-dependent patches.
- **others** - to load special user routines or parts of other application PAM files.
EDITPAM has been written in a very modular way and it should be fairly obvious how to select the wanted options or delete parts not used for a given application.

One of the following pilot patches should be selected:

P=*USEDIT  - assembles complete program EDIT with input of action requests and update of MT,

or P=*USEDIT1  - selects only that program part which inputs action requests,

or P=*USEDIT2  - selects update phase of EDIT, sorted action requests must be provided on unit ITB1.

P=CDC  - version for CDC 7000 (should also work for CDC 6000),

or P=PDP  - version for PDP10 (FOROTS),

or P=IBM  - version for IBM 360/370,

or P=UNI  - version for UNIVAC 1100 (not tested).

For these computers the FQX/FQT packages have been installed.

Normally, the FQX input/output routines will be loaded. If conversion to/from IBM 370 is wanted (on non-IBM computers) one of the following patches should be selected:

P=INFQX  - loads FQTXIN instead of FQXIN,

P=OUTFQX  - loads FQTOUT instead of FQXOUT,

P=FQXFQX  - loads FQX and FQT packages, wasteful if rarely used.

Whenever conversion to/from IBM 370 is required, this has to be signalled by an action request */FQTOUT */FQTXIN, even if the proper FQT routines have been loaded.

Debug prints have been provided for most of the processors and can be useful if problems are encountered or simply to make some check

P=DUSEDIT  - will select all debug options, this produces a lot of printout!

P=D... D=D...  - selects debug prints for the requested patch or deck.
A standard size of 15 000 words has been reserved for the HYDRA working space. If you want to readjust this size define in the cradle:

```
+DEF, Z=SPACE.
+, SPACE (n).
```

A listing of run decks as used for tests on CDC, IBM and PDP has been added in Appendix 7.

2.2 I/O units and control of I/O files

I/O units are set in the selected computer patch and can easily be re-defined via sequences.

Printed output is distributed onto different files since it is accumulated concurrently. At termination of a run all non-empty printer files are copied to output automatically. Files used are:

- IQPRNT - standard output and error messages,
- IQLOG - for run statistics accumulated during execution,
- IQPR2 - for special user printout (*PRINT,...),
- ITB2 (ITSTA) - for special statistics printout of all actions up to now (*STAT).

2.2.1 MT input

EDIT accepts several input MT. The number used for a given run has to be indicated via action request *TAPE, which specifies the number of input "volumes", 1 is assumed if request is absent. FQX and FQT cannot be mixed.

The definition of "volume" depends on computer/operating system used:

- **CDC** - A "volume" on tape or disk is terminated by a trailer label (tape only) or by two consecutive EOF, any number of files separated by single EOF are accepted. A new volume has to be mounted with logical unit number n+1 (example: standard
assignment for input MT is unit 11, second volume to be
mounted on unit 12, etc.). This allows to use different file
specifications and the use of common I/O system buffers
(EQUBUF). A maximum of 10 input volumes can be used.

- **PDP**
  - A "volume" on tape or disk is terminated by two consecutive
  EOF, on tape any number of files separated by single EOF are
  accepted. Assignment of logical units and file specifications is
  made via teletype. At end of a volume the input file is closed
  and unloaded. New assignments are requested via TTY. Any
  number of volumes is accepted.

- **IBM**
  - A "volume" on tape or disk is terminated by a single EOF.
  Several files on a single tape or different tapes are treated
  in the same way. All file specifications should be done via
  JCL cards. Any number of volumes is accepted.

- **UNI**
  - A "volume" on tape or disk is terminated by two consecutive
  EOF, on tape any number of files separated by single EOF are
  accepted. Any number of volumes is accepted.

2.2.2 **MT output**

If output is written on tape, one should try to avoid multivolume
files, i.e. a single file continued after EOT mark on another tape. Multi-
volume files are always clumsy to use and cause problems with some operating
systems.

We have added some coding to the output routine which allows to
terminate a tape and request a new output tape under program control.
FQX/FQT provides after each call the number of records and words written
so far. A sequence +SEQ, Z=IHOLD, DATA IHOLD/n/ can be set to indicate
the maximum number of words that can be written onto a tape. This depends
on recording density, word length and tape length but is usually not
wildly different in a given application. For example, 4.4 Mwords can be
written onto a full 1600 bpi tape in CDC on records of 512 60-bit words.
At the begin of each new unit (e.g. roll) the program will calculate the
average number of words per roll and the number of words still available. If not enough space is available it will automatically ask for a new tape. One should therefore always provide an additional output tape if one gets close to tape capacity.

The following differences should be noted:

CDC  - Logical unit number for new output tape is n+1, standard assignment for MT output is unit 21. Up to 10 output MT are possible.

FDP  - Output MT file is closed and unloaded, new specifications are requested via TTY.

IBM  - New file specifications should be provided via JCL card.

UNI  - Logical unit number for new output tape is n+1, standard assignment for MT output is unit 21. Must be tape (NTRAN).

2.2.3 MT input for merge

A single input "volume" in FQX or FQT format is allowed, all other specifications as for sect. 2.2.1).

2.2.4 STRIP output for precursor records

A single output "volume" in FQX format is allowed. Note that the information is very compact since only precursor records are written. All other specifications as for sect. 2.2.2).

2.3 FQX/FQT format

Input/output on the home computer will generally be done with FQX to avoid repeated conversions. FQT is used only when information on tape should be exchanged with other computers. On IBM 370 FQT does not exist - it is FQX of course.

No problems were encountered when reading properly written FQT tapes. When writing FQT one should keep the following points in mind:
CDC 7000 - Write FQT output to disk with default format, (normally RT=W, BT=blank), copy disk to output tape with file specification (RT=F, BT=E, FL=4800, MBL=4800). Writing directly to tape may cause problems with writing of EOF. This probably depends on the CDC 7000 record manager used.

CDC 6000 - You can write directly onto the tape with file specification (RT=F, BT=E, FL=4800, MBL=4800).

IBM 370 - When a tape should be exchanged do not forget to use DCB=(RECFM=VBS, BLKSIZE=3600), otherwise your tape may be unreadable on other than IBM machines.

PDP 10 - When you write FQT output to disk and later copy with PIP to an output tape do not forget: SET BLOCKSIZE MTAn: 800.

UNIVAC - FQT output is via NTRAN and all relevant parameters are set.

Always use *FQTIN/*FQTOUT to request conversion from/to IBM 370 format, even if the proper FQT package is loaded.

3. **SET-UP OF PRODUCTION VERSION**

3.1 **Definition of user header information**

The user header is part of the precursor record of FQX/FQT records and is automatically transferred to/from a local array specified in the subroutine call. This array is called NHEAD in EDIT.

Normally, the standard data information in NHEAD will be stored at fixed addresses. An option has been provided in EDIT to handle also packed bitted information stored dynamically. This is very convenient to store status bits of any kind in a very compact form. Standard quantities can be addressed directly, dynamic information is handled with routines ISETH and IGETH which do all the necessary operations to store and fetch bit information and associated data (if any). These routines can also be used outside EDIT. Dynamic information will be stored after the standard information. Refer to Appendix 4 for more details.
The content of the header vector has to be defined by the user for his particular application. A few obligatory quantities have to be defined to ensure a correct execution of EDIT (i.e. ROLL, FRAME, TYPE). Special features of EDIT can be used if some optional quantities are defined. The corresponding option is ignored if the address of the optional quantity is zero. Any other information stored in the fixed or dynamic parts of the header vector is simply transferred.

The fixed addresses for obligatory and optional quantities in the standard data words in NHEAD have to be defined in routine INITIT by the user with sequences Z=EDM04 and Z=EDHEAD. Standard values are given in P=CUSEEDIT. The quantities will then be found in NHEAD (IH...):

IHROLL - **Obligatory**, defines quantity "ROLL" as a set of logically related elements (e.g. roll of film which contains several frames on MT).

IHFRAM - **Obligatory**, defines quantity "FRAME" as an element of the "ROLL" set (e.g. a given frame on a roll of film).

IHTYPE - **Obligatory**, defines "TYPE" of the user header. This quantity allows to define a hierarchy of user headers. **Normal records on MT should contain NHEAD (IHTYPE)=5** (if 0 on input then 5 will automatically be provided). The exact definition of TYPE is:

1 = "run" header  
2 = "experiment" header  
3 = "roll" header  
4 = "special frame" header  
5 = "frame" header, standard value on MT  
> 5 = special applications.

IHMEAS - **Optional**, to define a further sub-classification of FRAME (e.g. part of a frame = one record/event if several events on frame or 1 event split into pieces). If IHMEAS is defined also IHNUMB must be set to the number of digits (in powers of 10) expected from the quantity in NHEAD (IHMEAS).
EDIT will define a new FRAME = NHEAD(IHFRAM)*
*IHNUMB+NHEAD (IHMEAS). This is the FRAME number to be
given on the action requests (see sect. 4.1).

IHNWB
- Optional, to signal to EDIT that user header contains
dynamic information. Note, that no standard header words
may be stored after NHEAD (IHNWB).

IHRUN, etc.
- Optional, to signal additional standard header words
which may be printed on run statistics, 0 will be printed
if addresses set to zero.

IHEDIT, etc.
- Optional, these are bit addresses in the dynamic part of
the user header where EDIT itself can store status informa-
tion for the run. If IHNWB or any of the bits defined
in COMMON /HEADST/ is not set this information is simply
ignored.

Standard header words are assumed to be integer (maximum 24 bits).
A practical example of what kind of information may be contained in the user
header, is given in Appendices 4 and 5.

3.2 Definition of structural relation of HYDRA banks

It has been mentioned before that any HYDRA bank structure can be
treated by EDIT. This is possible because the structural relation of
banks is given in a title DS. The detailed format and content of DS is
given in sect. 3.5.

Only those banks which are addressed by an action request have to be
indicated explicitly. Please refer to the data structure shown in
Appendix 2 for the following explanations:

Let us assume that banks PF, TF, MF and US are explicitly addressed by
action requests. The vertical structures KT-PF-TF-MF and KT-PF-TF-US have
to be given in title DS. PF and TF are already contained in these chains
and need not be specified again. The program also has to be told which
vertical links have to be followed to get to the next lower level. For
our example, these link addresses are 0-2-2-2 and 0-2-2-4, respectively.
Link address 0 is used to indicate the entry bank to the structure. The links to the next lower level will be found in IQ(LKT-2), IQ(LPF-2), IQ(LTF-2) and IQ(LTF-4).

At each level there may exist several banks of the same type connected into a horizontal structure through link 1. If one does not want to follow each bank of the horizontal structure vertically to the next lower level, one has to use identifiers to find the wanted bank in the linear structure and then follow the vertical structure for this bank only.

Identifiers are the contents of suitably chosen data words in each bank. These may be counters, labels, code words, etc. in a simple representation. External representations may be used on the action requests to facilitate the use of more complicated numbers. Conversion routines will transfer this external representation to the true value contained in the data word of the bank. Labels of PF and TF banks are used as BCD characters externally and stored as floating numbers in the bank; other examples are mass values stored in MF banks and represented as particle names externally.

Some banks may have no identifiers defined or identifiers may have been skipped on the action request. In this case all banks in the horizontal structure will be followed vertically to the next level(s). One must make sure, however, that the wanted bank can be addressed uniquely. If no identifier is given for the last addressed vertical level the address of the entry bank of the linear structure will be returned.

The names of the banks quoted in titles DS and AC must be the same; they need not be identical to the actual bank names of the data structure. But it might be simpler just to use the real bank names to avoid confusion.

3.3 Definition of valid action codes

For each application a valid set of action codes has to be defined in title AC (see details in sect. 3.6). This is needed to check for validity of input. Action codes are composed of a first letter which
indicates the action and 1-3 letters which specify the bank name for which the request should be executed (same bank names as in title DS). For action H no bank name is given since the request refers to the user header information (see details in sect. 4.1.2).

Once production has been started one should not change the order of action codes in title AC, since this information is also used for statistics purposes. If additional action codes are needed they should be added at the end of the list, up to 48 different action codes are allowed. Action codes no longer needed should not be removed, however.

3.4 Subroutines to be supplied by the user

Several entries for additional user routines have been supplied in EDIT. The coding of these routines has to be provided by the user because they have to be adapted to his special application.

DSTO - Routine to prepare DST information, this can be present data structure modified or non-HYDRA format - indeed anything the user wishes to code. If output is produced in non-HYDRA format, writing should by controlled by routine DSTO itself and IDELET in COMMON /FLAGS/ should be set non-zero to skip output of MT.

PRIN - Routine to print information contained in HYDRA banks and user header. Printing routines generally exist already for other HYDRA applications programs and simply have to be interfaced in PRIN.

USER - A special user subroutine which can be defined via sequences.

VERI - Verify modified HYDRA banks and user header for correct and complete information. Note, that EDIT will insert correct structural links but cannot insert reference links or verify for unique identifiers, since it has no a prior knowledge about the detailed information in the banks treated.
X, Y, Z - Processors for manipulation of data structure by user code defined via sequences.

XLABEL PARDAT - Routines for conversion of external to internal representation of labels or mass codes (unless standard versions from GENUTY are used).

RDUIN1 RDUIN2 RDGST - User input routines for action requests, anything may be coded into these routines, they should transfer requests to the internal format as shown in sect. 5.

3.5 Title DS

This title is **obligatory**.

**Title DS** specifies the structural relation of banks, the identifier address in the data word area of the bank and the mode of the identifier.

Only those banks need be specified explicitly which are directly addressed by any action request. The whole chain of banks, from the lowest addressed bank, up to the entry bank to the data structure must be given.

**Example:** (see also corresponding data structure in Appendix 2)

```
*  DS  93****
   1,1   (I2,3X,15)
  0  23
     23,13   (I2,3X,13(A4,1X))
     1  MF  TF  PF  KT  SH  IN  PF  KT  EH  KT  PLS  PF  KT
     2  TLS  TF  PF  KT  US  TF  PF  KT  PHY  KT
     3  2  2  2  0  2  4  2  0  4  0  3  2  0
     4  3  2  2  0  4  2  2  0  5  0
     5  23 13   (I2,3X,13I5)
     6  1  1  1  0  1  3  0  1  0  0  0  0  1  0
     7  9  1  1  0  1  1  1  0  0  0
     8  23 13   (I2,3X,13I5)
```

The following information is given in the title (standard format of title as described in HYDRA System Manual):

```
line 0 Number of entries in title.
lines 1,2 Chain of connected banks, starting with the lowest addressed
bank up to the entry bank to the structure, stored without gaps.

lines 3,4 Link address of bank to next highest bank, 0 means this is entry bank to structure. If banks with same name are attached to more than 1 link, the link addresses are packed (base 10) and flagged by "-" sign (e.g. -23).

lines 5,6 Data word address for identifier of bank, 0 means that no identifier to be expected for this bank.

lines 7,8 Mode of identifier (0 = no identifier given, 1 = floating, 2 = integer, 3 = label [integer part of floating] 4 = mass code).

3.6 Title AC

This title is obligatory.

It gives a list of valid action codes for the given application. Any bank names directly addressed must be given with at least one of the letters defining the type of action.

IMPORTANT: DO NOT CHANGE order of action codes in title AC once you have started production, add additional codes to the end of the list and do not remove unwanted codes. A maximum of 48 action codes is allowed.

Example:

```
*  AC  43   (I5/(I4(A4,1X)))
42
H  CKT MKT APF CPF DPF LPP MPF RPF UPF ATF CTF DTF LTF
MTF FTF UTF AMP CMP DNF RNF UMF AMS CUS DUS RUS UUS AIN
DIN KIN ASH DSH RSH USH DHE DPHY UPHY CPLS DPLS CTLS DTLS H
```

3.7 Title MC

This title is optional.

It gives a list of mnemonic codes for given bank names (blank for header), the corresponding numerical values and the type of information. The first four letters of a code word are significant. Mnemonic codes
are only used for frequently addressed quantities.

Example:

```
  MC  112****
  6   (I2,3X,2I5,4(A4,1X),5I5)
  35,13 (I2,3X,13(A4,1X))
  1   NH EXP RUN BEM.LAB SC2 SC3 PART REM EVl EV2 EV3 EVN
  2   FLUX CL1 CL2 CL3 CL4 CL5 CL6 CL7 CL8 CL9 GAM KNK XLAB
  3   VERT FIXM STP XLAB TEND PUM MASS PL DP1
  35,13 (I2,3X,13I5)
  4   1  3  12  13  14  2  3  12  14  26  27  28  29
  5   64 97 103 109 115 121 127 133 139 145 13 14  1
  6   12  7  8  1  7  4  1  2  5
  7   35,13 (I2,3X,13I5)
  8   3  3  3  3  3  1  1  1  1  1  1  1
  9   1  1  1  1  1  1  1  1  1  1  2
  9   2  1  1  2  2  1  2  2

The following information is given in the title (standard title format as described in HYDRA System Manual):

line 0  Number of banks with codes, total number of codes, bank names (blank for header), number of codes for each bank specified, zero has to be added always.

lines 1-3  Mnemonic codes for bits, data words, etc.

lines 4-6  Absolute address associated with code.

lines 7-9  Type of information:

1  = bit number
2  = data word address
3  = standard data word for header
4  = reason for decision (action R)
5  = link address (action L)
6  = data word contents

3.8 Title RQTR

This title is obligatory.

Two trap classes have been initialized:

class 10  Trap to terminate program if some fatal condition has been reached (error in I/O handling, obligatory information missing).
class 20 Trap if end of some input information is encountered (end of intermediate file, end of merge input, etc.). Processing will continue from beginning of EVLOOP.

Note, if program version *USEEDIT1 (input processing) has been selected, the RQ package is not loaded and only the standard system condition ID's (< 100) will be handled.

A list of presently defined condition ID's is given in Appendix 6.

4. ACTION REQUESTS TO CONTROL FLOW OF EDIT

All action requests are treated in routine RDEDIT and can be intermixed under user control. The same internal format is created by the various input routines and any further processing is therefore independent of the source of action requests.

The standard input unit is ITI1, but unit ITI2 can be used as alternative in the user supplied input routines.

The default input format for action requests are action cards, all general requests use this format. If any other input format should be used one simply switches mode by indicating the selected option via an action card. If one wants to switch to yet another input mode, one should give a new action card (which must be understood by user input routine) and so on until all input is finished.

4.1 Action card input

Action cards are free format, 80 column card images. Continuation cards are read in if a $ (dollar sign) is encountered, a maximum of 300 characters are accepted for a single input "card". Blanks are ignored. Any number of action cards can be read in in random numerical order.

The general format of action cards is:

ROLL, FRAME, ACTION CODE [,identifiers][,specifications]

The first three items must always be present.
ROLL, FRAME are given as integers or * (asterisk). They refer to the contents of the user header words NHEAD (IHROLL) and NHEAD (IHFRAM). If IHMEAS > 0 was set in routine INITIT (i.e. event/measurement number should be used also to select a subset) then FRAME corresponds to NHEAD (IHFRAM)*THNUMB+NHEAD(IHMEAS). This means, the extra digits should be included in FRAME on the action cards. All references to the user vector NHEAD are explained in sect. 3.1:

ROLL: *

All roll numbers on MT should be treated.

0 Only records for roll 0 should be treated, this refers to run or experiment header records.

n < 0 Only special header records for roll n should be treated.

n > 0 Records for roll n should be treated.

n₁/n₂ Records for rolls n₁ to n₂ (inclusive) should be treated.

FRAME: *

All frame numbers for a selected roll (or all rolls) should be treated.

0 Only special header records with frame = 0 should be treated. Refers to roll, experiment or run header records.

m < 0 Special header record for frame m should be treated.

m > 0 Record for frame m should be treated.

m₁/m₂ Records for frames m₁ to m₂ (inclusive) should be treated.

"Number range" (n₁/n₂) for rolls and/or frames can be useful if the same actions apply for a subset of the MT to be updated. Different ranges, overlapping or not, may be used. Note the following limitations, however:

- "number range" creates 1 request for every element of the indicated range, you should therefore keep the number of such elements reasonably small. Rather use "*", if applicable.

- "number range" cannot be used to create new frames on MT, you have to use roll, frame explicitly.

The list of valid combinations of ROLL, FRAME and the records on MT actually treated then looks as follows: ("THROLL" stands for NHEAD(IHROLL), etc.):
<table>
<thead>
<tr>
<th>ROLL</th>
<th>FRAME</th>
<th>&quot;IHROLL&quot;</th>
<th>&quot;IHFRAM&quot;</th>
<th>&quot;IHTYPE&quot;</th>
<th>to be treated</th>
<th>flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>*</td>
<td>any</td>
<td>any</td>
<td>any</td>
<td>all rolls, all frames, including all special header records</td>
<td>0</td>
</tr>
<tr>
<td>*</td>
<td>0</td>
<td>any</td>
<td>0</td>
<td>1,2,3</td>
<td>all roll headers, incl. all run and exp headers</td>
<td>1</td>
</tr>
<tr>
<td>*</td>
<td>-m</td>
<td>any</td>
<td>m</td>
<td>4</td>
<td>all rolls, special header record for frame m only</td>
<td>2</td>
</tr>
<tr>
<td>*</td>
<td>m</td>
<td>any</td>
<td>m</td>
<td>5</td>
<td>all rolls, frame m only</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>*,0</td>
<td>0</td>
<td>0</td>
<td>1,2</td>
<td>all run and experiment header records only</td>
<td>3</td>
</tr>
<tr>
<td>-n</td>
<td>-m</td>
<td>n</td>
<td>m</td>
<td>4</td>
<td>special header record for roll n frame m only</td>
<td>6</td>
</tr>
<tr>
<td>n</td>
<td>*</td>
<td>n</td>
<td>any</td>
<td>any</td>
<td>roll n, all frames</td>
<td>4</td>
</tr>
<tr>
<td>n</td>
<td>0</td>
<td>n</td>
<td>0</td>
<td>3</td>
<td>roll header record for roll n only</td>
<td>5</td>
</tr>
<tr>
<td>n</td>
<td>m</td>
<td>n</td>
<td>m</td>
<td>5</td>
<td>roll n frame m only</td>
<td>7</td>
</tr>
</tbody>
</table>

Combinations not indicated will be flagged as error by the input routine.

ACTION CODES indicate which action should be performed on the selected record(s) for given ROLL, FRAME. Two types of action requests are accepted:

- **General actions** which can refer to a given roll-frame or to some part of the MT globally. They are specified by **fixed** keywords of the form * keyword (see sect. 4.1.1 for details).

- **Specific actions** on HYDRA banks and user header vector. Action codes are composed of a first letter which indicates the type of action requested, followed by 1-3 letters which specify the bank name to be addressed. The general form of the action codes is: abbb (see sect. 4.1.2 for details).

Identifiers and specifications are normally used only for specific actions and will be discussed in sect. 4.1.2.
4.1.1 General actions

All action codes start with * (asterisk). Only the first 4 characters of the keyword are significant, but you may use more if you want. The general definitions for ROLL and FRAME may be used as explained in the previous section. This allows selection of individual frames, "number ranges", complete rolls or the complete information on the MT for global options. For I/O options only the form *,*,* keyword is meaningful.

Global options:

*SELECT [,x]  - Select rolls/frames from MT input. Note, that all rolls/frames not selected will be removed from output, but will still appear on STRIP output.

*DELETE [,x]  - Delete rolls/frames from MT input, all rolls/frames not quoted will be transferred to MT output. Deleted records will still appear on STRIP output.

*SELECT, *DELETE can both be used for same MT input. [,x] denotes optional information, see end of this section.

*STRIP   - Strip-off precursor records onto separate file.

*USER   - User option, code can be defined in cradle via sequences.

*DISCARD   - Drop all discarded banks (i.e. status bit 1 has been set).

*VERIFY   - Verify data structure and user header information for consistency and completeness. Subroutine VERI has to be supplied by the user. Note, that VERI will automatically be called for some specific actions where banks are inserted or relinked (actions C,L,M) to verify correct linking, especially for reference links.

*DSTOUT   - Create DST information for output on MT. Subroutine DSTO has to be provided by the user.

*DUMP   - Get dump of user header and data structure (DQSNAP). Three options are included at present:

mode = 0 - memory map
 = 1 - full dump of data structure
 = 2 - full dump of memory

user header vector is always printed.

*PRINT   - Print information in banks and user header vector. Subroutine PRIN and other print routines have to be provided by the user.

*HIST   - Histogram of bits set in user header vector up to now.

*STAT   - Get statistics on all EDIT operations done up to now.
The last two options can only be used if dynamic part of header is used, see sect. 3.1 for details.

I/O options:

*ACTIN
- Switch to input of action cards (default option).

*GSTIN
- Switch to input of GST cards, subroutine supplied by user.

*UIN1 (2)
- Switch to input of user format 1 (2), subroutine(s) supplied by user.

*FQTMERGE
- Merge input is in FQT format (default is FOX format).

*FQTIN
- Read MT with FQT (default is FOX).

*FQTOUT
- Write MT with FQT (default is FOX).

*TAPE
- Several MT input "volumes" to be processed (default is 1 if request absent or number not specified).

*IBMIN
- Same as *FQTIN (kept for compatibility).

*IBMOUT
- Same as *FQTOUT (kept for compatibility).

*TAPIN
- Same as *TAPE (kept for compatibility).

*NOLIST
- Suppresses all statistics printout produced by routine EVSTAT, error messages are still printed.

*SORT
- Sort MT input in ascending roll/frame/type order.
  Input can be FOX/FQT records, several input "volumes" can be used. The number of records that can be sorted depends on the size of blank common and input buffer, adjust sequences Z = SPACE and Z = USSORT as required.

*COMBINE
- Combine parts of the same frame stored consecutively on MT. The detailed code has to be provided by the user via sequences.

  Note, that SORT and COMB need extra intermediate files, to be defined in the main program.

*MTMERGE
- Merge input is on file MT (default is from LUNMRG). Only one record can be accepted and the practical usability is fairly restricted.

For actions *SELECT and *DELETE additional information can be provided for conditional selection. This information relates to standard data words and/or bits in the user header vector. The format is identical to the one described for action H in sect. 4.1.2.

For all other general actions one extra floating word is accepted following the action keyword. This word is transferred as argument to
the subroutine and can be used to select various options (e.g. number of
tapes on input, mode for dumps, mode of printout, etc.).

Additional general actions may be included by the user, a total of
24 different keywords are accepted for each of global options and I/O
options.

4.1.2 Specific actions on HYDRA banks and user header vector

Action codes are composed of a first letter which indicates the
type of action requested, followed by 1-3 letters which specify the bank
name to be treated. Note, that the bank names must be identical to the
bank names given in title DS (sect. 3.5), they need not correspond to the
true bank names in the HYDRA data structure. It is nevertheless recom-
mended to use the same names to make life easier. The list of valid
action codes for a particular application is given in title AC (sect. 3.6).

The following actions have been coded:

A = activate discarded banks (reset status bit 1 to 0)
C = create new banks
D = drop banks and structure supported
H = update/create user header information (only H used, no bank name)
L = relink banks in same structure
M = merge banks (or complete structures) from two different data struc-
tures
R = remove banks (set discard bit = status bit 1 to 1)
U = update existing banks, status bits and data words
X = user option, action card input as for A,D
Y = User option, action card input as for L
Z = User option, action card input as for U.

Identifiers are needed in order to address different banks in a
possibly complicated structural relation. They are defined as the contents
of suitably chosen data words of banks. One identifier per bank is allowed,
there may be no identifier and different data words may be used for
different banks. Identifiers can be integer, floating, external label
or particle name representation, or indeed any other form that can be
uniquely converted between external (i.e. printout) and internal (i.e.
HYDRA bank) representation. Note, that data words in HYDRA banks should
preferentially contain floating information, but **never** hollerith charac-
ters.

Title DS (see sect. 3.5) gives the relation between banks, the data
word address and the mode of the identifiers. Several identifiers may be
needed to address the required HYDRA bank, they have to be given starting
from the highest bank (after entry bank) downwards. One can skip a re-
quired identifier by inserting "-" (minus) into the sequence of identifiers.
In this case all banks with the same name will be followed vertically. This
can be done for several levels but may soon become time consuming and can
result in errors if not unique.

The word "specifications" has been used to denote any additional
information that may be needed to execute the action request. Explicit
information is given for each action further on in this section.

A few conventions for identifiers and specifications are used:

- Floating point numbers must be written with the decimal point, other-
  wise they are stored as integer.

- Octal numbers (i.e. bitted information) can be used but should not
  exceed the 24 rightmost bits in the word, the number has to be written
  with prefix 0 (i.e. 0777). Hexadecimal numbers have to be written with
  prefix Z (i.e. Z1F). Octal and hexadecimal numbers are converted to
  integer on input to give the correct bit pattern.

- Hollerith characters are **not** allowed as information to be stored.

- Items are separated by comma (,).

- Consecutive elements of an item are separated by slash (/). That is,
  several identifiers are separated by (/) or a series of consecutive
  bits or data words, etc.
Addresses for bits or data words are given by their absolute address in integer form or by a mnemonic code (CODE). In this case the exact relation between mnemonic code, absolute address and type of information is given in title MC (see sect. 3.7).

- Standard data words for header are denoted by

  \[ *n = \text{value} \quad \text{or} \quad *\text{CODE} = \text{value} \]

- Status bits in banks or bits in the user header are denoted by

  \[ n = \text{ON(OFF)} \quad \text{or} \quad \text{CODE} = \text{ON(OFF)} \]

1(0) may be used instead of ON(OFF) to set bit to 1(0).

- Data words for banks or extra data information for bits in user header are denoted by

  \[ +n = \text{value} \quad \text{or} \quad +\text{CODE} = \text{value}. \]

- The order of specifications should be:

  - standard data words for header (H only)
  - bits
  - data words.

- Roll, frame numbers are integer (or *).

- Bits and data word addresses are integer or mnemonic codes.

- Identifiers and data word contents have to be given in the form they are stored in the HYDRA banks, this is floating in general (see note on identifiers). In addition, data word contents can be given as mnemonic codes or with prefixes L (external label, i.e. "L"A1), M (mass code, i.e. "M"PI), O (octal number, i.e. "O"77) or Z (hexadecimal number, i.e. "Z"1F). The user has to make sure that no conflicts arise between prefixed items and mnemonic codes.

- All other quantities are floating.

The following examples show a typical deck of action cards used for some application:
Status bit 18 will be set automatically for all banks modified by EDIT for actions A, C, L, M, R and U. The action request VERI will automatically be generated for actions C, L and M to verify bank chaining and reference links.

(a) **Action A** - activate bank

ROLL, FRAME, Abbb, II/.../I_b

set status bit 1 to zero for bank bbb with identifier I_b. I_b must be given.

Example: 125, 1325, AMF, A/5/PI
reset π mass fit for track 5 from vertex A.

(b) **Action C** - create new bank

ROLL, FRAME, Cbbb, II/.../I_b, \( n_1 = \text{ON}, n_2 = \text{ON/ON}, + d_1 = \alpha, + d_2 = \beta/\gamma/... \)

bank bbb will be linked into the data structure and the identifier of the created bank set to I_b, bits \( n_1, n_2, ... \) and data words \( d_1, \ldots \)
\[ d_2 \ldots \text{will be set, quantities not explicitly defined will be set to zero.} \]

**Example:** 125, 1325, CMF, A/5/PI, \(4=\text{ON/ON/ON}, \, \, +P=1./0.3/0.3\)

create MF bank for \( \pi \), set status bits 4, 5, 6 and set data words \( P, \, P+1, \, P+2 \).

Note, that this assumes that at least one bank \( bbb \) exists already so that number of links (\( NL \)), number of structural links (\( NS \)) and number of data words (\( ND \)) can be extracted to lift the new bank. If a completely new bank has to be inserted the appropriate quantities have to be provided. Only in this special case the action request would take the form:

\[ \text{ROLL, FRAME, C} bbb, \, I_l/\ldots/I_b, \, *_l=NL/NS/ND, \, n_1=\text{ON}, \, \ldots \]

\( NL, NS, ND \) should be inserted as integers.

(c) **Action D** - drop bank

\[ \text{ROLL, FRAME, D} bbb, \, I_l/\ldots/I_b \]

bank \( bbb \) with identifier \( I_b \) will be dropped and also the data structure supported vertically, horizontal banks will be bridged.

**Example:** 125, 1325, DEH or 125,*,DEH

drop bank EH and the whole data structure for roll 125, frame 1325 or for all frames of roll 125.

(d) **Action H** - create or update FOX/FQT user header

\[ \text{ROLL, FRAME, } *m=\text{value, } n_1=\text{ON, } n_2=\text{ON/OFF/ON or} \]

\[ \text{ROLL, FRAME, } *m=\text{value, } n_1=\text{ON, } +d_1=\alpha/\beta/\gamma/\ldots \]

an existing user header vector will be updated or a new one will be created (precursor only, no data structure associated).

Please note the following restrictions:

- Standard header data words (\( *m= \)) need in general only be given if a new header is created. In this case word 1 (length) and the obligatory words NHEAD (IHROLL, IHFRAM, IHTYPE) must be set as the minimum requirement. Standard data words can also be replaced in or inserted into existing user header information.
- Any number of bits can be set ON/OFF (or 1/0 if you prefer) on the same action card.

- If extra data words should be stored in the dynamic part of the user header vector, only one bit should be addressed on the action card. All or only a selection of extra data words can be replaced, the complete set must be given if new data words should be created. One word counting the total number of extra data words for the bit will be inserted before the user information.

- If bits are set OFF the corresponding data (if any) will also be removed, if bits are set ON any already existing data will be overwritten completely or just the data words to be replaced. No checks are made.

**Example:** 157, 500 H, *1 = 14/16/81/157/500, *15=3
create new header for roll 157, frame 500 set standard data words 1-5 and 15
157, 329, H, 77=ON, +1 = 3./10./1.25
157, 330, H, 78=ON, +5 = 6.2/8.9, +8 = 16.32
set bit 78 and insert (replace) data words.

(e) **Action L** - relink bank

\begin{verbatim}
ROLL, FRAME, Lbbb, I1/.../I_b, Lxxx, J1/.../J_x, n.
\end{verbatim}

this will relink bank bbb with identifier I_b at the bank xxx with identifier J_x at link address n. This option should be used with caution since no good checks for correct linking are possible! The use of "null" identifier (I_b = 0 for instance) is possible and means the whole linear structure with all dependent structures will be linked at the requested place.

**Example:** 125, 3125, LEH, LTF, A/5, 4.
bank EH and the whole structure will be linked at LTF-4 for bank TF with identifier 5 (and PF identifier A). Note, that LEH does not have identifiers as will be found from title DS, this will be correctly treated by the input routine.
Action M - merge bank

ROLL, FRAME, M; In/.../I; M; In/.../J

This will merge bank bbb with given identifier I; from second data structure read from LUNMRG at bank bbb with identifier Jj in memory, read from input MT. Note, only banks with same name can be merged and both M; have always to be given.

The actual type of merging depends on the identifiers given and on the values for ROLL, FRAME. No automatic merging is done, frames (or rolls) have to be explicitly requested:

- If bbb represents the entry bank to the structure (e.g. KT in our examples) the whole structure and the user header will be replaced. This allows to replace or insert complete frames or rolls (e.g. ROLL;MKT, MKT).

- If bbb represents any bank other than the entry bank to the structure the frame from MT will be updated with banks from LUNMRG, the user header from LUNMRG will simply be ignored:

(i) if \( I; > 0, J; > 0 \): bank with identifier \( I; \) will be inserted, bank with identifier \( J; \) will be dropped

(ii) if \( I; = "-", J; > 0 \): linear data structure of banks bbb will be inserted, bank with identifier \( J; \) will be dropped

(iii) if \( I; > 0, J; = "-" \): bank with identifier \( I; \) will be inserted, no banks will be dropped.

(iv) if \( I; = "-", J; = "-" \): linear data structure of banks bbb will be added at end of existing bank, no banks will be dropped.

Note, that banks will not be replaced automatically.

A special option is available to replace or insert precursor records only (stripped-off header). In this case one specifies ROLL, FRAME, M, M and the correct precursor record will be selected from LUNMRG and transferred to MT output, any HYDRA banks that might be present will be completely ignored. Note, that identical output will be produced on MT
output and strip output (if requested). Action code M has to be included in title AC (sect. 3.6).

**Example:** 125, 1325, MKT, MKT
merge complete frame 1325, roll 125

125, 1325, MPF, A, MPF, -
PF bank A (and whole structure associated) will be added to the already existing PF banks

125, 1325, MTF, A/5, MTF, C/9
TF bank 5 (from PF bank A) will be inserted and TF bank 9 from PF bank C in the present structure will be dropped.

(g) **Action R** - remove bank

```
ROLL, FRAME Rbhb, I1,.../Ib, option/i/j [_,Ib1, option/k,/,...]
```

set discard bit (status bit 1 set to 1) for bank bbb with identifier $I_b$ ($I_{b1}, I_{b2},...$). Several banks in the same linear structure can be addressed on the same action card or on separate action cards if preferred. **Option signals what should be done with the other banks in the same linear structure:**

+, A or ON signals that bank with identifier $I_b$ should be kept (discard bit = 0) all other banks should be discarded

-, E or OFF signals that bank with identifier $I_b$ should be discarded, all other banks are not touched.

The three alternatives for each option are entirely equivalent, users may use one or the other form as preferred. Combinations of both options (accept, exclude) are treated correctly.

**Reasons** for exclusion of certain banks may be given, optionally, together with the flag. They must be integers in the range 1-24 or mnemonic codes defined in title MC (sect. 3.7).

These "reasons" are transferred to a user bank if defined for the application. Reasons for accepting a certain bank with identifier $I_b$ are taken as reasons for excluding the other banks.
Example: 125, 3125, RSH, A/14, A/5/8, 16, A/5/6
accept SH banks with identifier 14 and 16 and reject others
for reasons 5 and 8

125, 3126, RMF, A/5/MU, OFF, K, OFF
switch-off MF banks for μ and K for track 5 from vertex A.

(h) **Action U** - update existing banks

\[ \text{ROLL, FRAME, Ubba, \ I_b/} \ldots /I_b, \ n_1 = \text{ON, } n_2 = \text{ON/ON/OFF, } +d_1 = \alpha_1, \ldots \]

update bank bbb with identifier I_b, replace status bits and data
word contents. I_b must be defined

Example: 125, 1325, UMIF, A/5/PI, PUM=ON/ON/OFF, +P1=1. 0.5
reset bits PUM, PUM+1, PUM+2 and data words P1, P1+1
for MF bank for U. Absolute values for PUM, P1 are
found from title MC (sect. 3.7).

(i) **Action X** - user option

\[ \text{ROLL, FRAME, Xbb, I_l/} \ldots /I_b \]

(j) **Action Y** - user option

\[ \text{ROLL, FRAME, Ybb, I_l/} \ldots /I_b, \ Yxxx, J_l/ \ldots /J_x, n. \]

(k) **Action Z** - user option

\[ \text{ROLL, FRAME, Zbb, I_l/} \ldots /I_b, \ n_1 = \text{ON, } +d_1 = \alpha_1, \ldots \]

4.2 User supplied input routines

Two user input routines for action requests can be called via action
cards "UIN1 or "UIN2 (see sect. 4.1.1). The sequences Z=RDUIN1 and
Z=RDUIN2 should contain the actual calls to the subroutines (see routine
REDIT) which have to be provided by the user.

At exit from the user routine the actual action request(s) for a
given frame should be returned in a well-defined format given in section 5
of the write-up. Input from various sources can be mixed in this way and
any further processing of the requests is independent of the source of
input.
5. SHORT COMMENTS ON PROGRAM

5.1 General remarks

A diagram of the program organisation can be found in Appendix 1.

Comments inside subroutines follow more or less WRUP conventions
(see HYDRA Applications Manual). A WRUP listing of the PAM file should
provide sufficient information to follow details. Optional material
has been provided via sequences. This was at a time when the PAM was
to be kept compatible with PATCHY3. This procedure has been abandoned
for other PAM files since then. The present PAM is no longer compatible
with PATCHY3, it does not use all the nice PATCHY4 facilities, however,
since there was not enough time for it.

It would be convenient to use random file organisation for sorting
of action requests and organisation of input/output files. Unfortunately,
handling of random files is very different on computers of different
manufacturers and one soon hits various restrictions. This has therefore
only been used in SORT to order MT input before the execution phase of
EDIT starts. Separate programs should be used if a large number of re-
cords have to be sorted.

5.2 Format of action requests at various stages of processing

5.2.1 At exit of input routines

At exit of routine RDACT (or user routines RDGST, RDUIN1, RDUIN2)
one or several requests are stacked without gaps in array ACT of sequence.
Z = RDEEDIT1. Information for one request is terminated by "next roll"
or by a negative number in place.

<table>
<thead>
<tr>
<th>Word</th>
<th>Format</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACT (1)</td>
<td>F</td>
<td>roll</td>
</tr>
<tr>
<td>(2)</td>
<td>F</td>
<td>frame-flag (FFFFf, see definition of flag in sect. 4.1)</td>
</tr>
<tr>
<td>(3)</td>
<td>F</td>
<td>length of information</td>
</tr>
<tr>
<td>(4)</td>
<td>F</td>
<td>keyword of action (=position in title AC)</td>
</tr>
<tr>
<td>(5)</td>
<td>F, I,...</td>
<td>data for this action request, if any</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>(see details in section 5.3)</td>
</tr>
<tr>
<td>n</td>
<td></td>
<td>roll (or -1. to signal end)</td>
</tr>
<tr>
<td>n+1</td>
<td></td>
<td>frame flag</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Any number of requests can be stored for same call, as long as they fit into array ACT. Length of information always includes this word in the count.

5.2.2 Internal to REDIT

Small HYDRA structures are built up and sorted in roll-frame order at end of input or if store-full condition occurs.

for rolls: \( +1 = \text{roll number} \)

for frames: \( +1 = \text{frame - flag} \)
\[ +2 = \{ \text{action keywords for this frame} \} \]
\[ +3 = \{ \text{packed in 2*24 bits} \} \]
\[ +4 = \text{keywords for global actions} \]
\[ (24 \text{ bits}) \]
\[ +5 = \text{keywords for I/O options} \]
\[ (24 \text{ bits}) \]

for actions: \( +1 = \text{length of information} \)
\[ +2 = \text{keyword} \]
\[ +3 = \{ \text{data for this action} \} \]
\[ ... \}
\[ \text{request (details in sect. 5.3)} \]

5.2.3 Format of intermediate file AR

An intermediate file is written by REDIT whenever the store is full or the end of information has been reached. The file contains one logical record for each roll-frame-flag, sorted in increasing order. This is the file read by AREC to assemble all action requests for a given frame from MT input.
<table>
<thead>
<tr>
<th>Word</th>
<th>Format</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>F</td>
<td>Length of record</td>
</tr>
<tr>
<td>2</td>
<td>F</td>
<td>roll</td>
</tr>
<tr>
<td>3</td>
<td>F</td>
<td>frame-flag</td>
</tr>
<tr>
<td>4</td>
<td>I</td>
<td>keywords for this frame (2 * 24 bits)</td>
</tr>
<tr>
<td>5</td>
<td>I</td>
<td>keywords for global actions (24 bits)</td>
</tr>
<tr>
<td>6</td>
<td>I</td>
<td>keywords for I/O options (24 bits)</td>
</tr>
<tr>
<td>7</td>
<td>F</td>
<td>length of information</td>
</tr>
<tr>
<td>8</td>
<td>F</td>
<td>keyword</td>
</tr>
<tr>
<td>9</td>
<td>F, I</td>
<td>data for this action request</td>
</tr>
<tr>
<td>n</td>
<td></td>
<td>first action request</td>
</tr>
<tr>
<td>n+1</td>
<td>F</td>
<td>length</td>
</tr>
<tr>
<td>n+2</td>
<td>F</td>
<td>keyword</td>
</tr>
<tr>
<td>m</td>
<td></td>
<td>data for this action request</td>
</tr>
<tr>
<td></td>
<td></td>
<td>second action request</td>
</tr>
</tbody>
</table>

5.2.4 **During execution phase of EDIT**

Processor AREC has the task of collecting all requested actions for a given roll-frame from MT input, to be inserted from MERGE input or directly created from the action request itself (i.e. special header record). There may be no requests, global requests or requests specific for the frame. This is handled by keeping in memory the previous, and possibly the next record, from file AR to cope with jumps in roll-frame sequence. Banks RFC are created which are just a copy of the record read from AR. Banks FRC and ARC contain all accumulated requests for the currently treated roll-frame. The detailed layout is shown in Appendix 8. Banks ARC are sorted in increasing keyword value.

5.3 Details of information for action requests.

In the following description word 1 always refers to the total length of information, word 2 to the keyword for the given action (see above).

"Identifiers" denote the contents of the data word of a bank specified in title DS, (see sect. 3.5). "Data word addresses" denote packed 24
bit words which indicate the addresses of the information to be stored. New words are automatically inserted if addresses go beyond word boundaries (24, 48, ...). Bit 25 will then be set to indicate that another word with packed data word addresses follows. Bits equal to 0 in the data word addresses indicate that no new information is present, any existing information in the data words in the banks will be left untouched. Immediately following the data word addresses are the contents, in increasing addresses, stored without gaps.

All information is floating, except bitted words, standard words for user header and any data word contents stored different from floating at input of action request.

5.3.1 **Action A** - activate banks

word 3
\[
\begin{array}{l}
\text{NI} = \text{no. of identifier} \\
\text{NI identifiers (last is for bank to be activated)}
\end{array}
\]

5.3.2 **Action C** - create new bank

word 3
\[
\begin{array}{l}
\text{NI} = \text{no. of identifiers} \\
\text{NI identifiers} \\
\text{NI+4} = 0. \\
\text{NI+5} = \text{NBITS} \times \text{1000} + \text{NDATA} \\
\text{NI+6} = \text{NBITS words with bit no.} \times 10 + 0 \text{ off + 1 on} \\
\text{NI+NBITS+5} = \text{data word addresses (24 bits, additional words if needed)} \\
\text{+6:} \\
\text{NDATA data word contents to be filled in.}
\end{array}
\]

This form implies that at least one bank with fixed specifications for NL/NS/ND is present.

An alternative form for action C is coded which allows to create banks without pre-existing banks of the same name or with bank specifications different from the standard values. Note, NL/NS/ND have all to
be given in this case (on action card as *1 = NL/NS/ND).

word 3   NI = no. of identifiers
        4  }   NI identifiers
        ..
NI+4     NL = total number of links
NI+5     NS = number of structural links
NI+6     ND = number of data words
NI+8     NBITS * 1000 + NDATA
...     ...  same information as above
...     ...  ...

5.3.3 Action D - drop bank

word 3   NI = no. of identifiers
        4  }   NI identifiers
        ..

5.3.4 Action H - update/create user header in PQX/FQT precursor

word 3   NDSTD = number of standard data words given
        4  }   data word addresses (24 bits, additional words if needed)
        5  }   NDSTD standard data words contents
        ..
NDSTD+5[4] NBITS * 1000 + NDATA (if NDATA > 0 then NBITS = 1 for unique association)
+6[5]
     ..  }   NBITS words with : bit no. * 10 + 0 off
     ..  }   + 1 on
NDSTD+NBITS+  data word addresses (24 bits, additional words if needed)
+6[5]  }  NDATA data word contents associated to bit
     ..

if NDSTD = 0 the next word following is "NBITS*1000+NDATA"

if NDATA ≠ 0 then NBITS must be 1 to ensure correct association of extra
data words with given bit number.
5.3.5 **Action L** - links banks

word 3  \( NI = \) no. of identifiers (bank to be linked)  
\( \vdots \)  
\( NI \) identifiers  
\( \vdots \)  
\( NI + 4 \)  \( NJ = \) no. of identifiers (bank where linked to)  
\( \vdots \)  
\( NJ \) identifiers  
\( \vdots \)  
\( NI + NJ + 5 \) link address

5.3.6 **Action M** - merge banks

word 3  \( NI = \) no. of identifiers (bank to be merged)  
\( \vdots \)  
\( NI \) identifiers  
\( \vdots \)  
\( NI + 4 \)  \( NJ = \) no. of identifiers (bank to be replaced or added at)  
\( \vdots \)  
\( NJ \) identifiers

5.3.7 **Action R** - remove / reset banks

word 3  \( NI = \) no. of identifiers  
\( \vdots \)  
\( NI \) identifiers  
\( \vdots \)  
\( NI + 4 \) last identifiers (for bank to be reset)  
\( NI + 5 \) specifications for \( \alpha \)  
\( NI + 6 \) specifications for \( \beta \)  
\( \vdots \)

\( \alpha, \beta, \ldots \) are identifiers in the same **linear** structure.

"specifications" contain bitted information on reasons for reset (see description of R) and in bit 25:

- 0 = keep this bank, remove all others (A, +, ON)
- 1 = remove this bank, keep all others (E, -, OFF)
5.3.8 **Action U** - update existing bank

word 3  NI = no. of identifiers
  4  }
  :  }  NI identifiers
  :  }
NI+4  NBITS * 1000 + NDATA
NI+5  }
  :  }  NBITS words with bit no. * 10
      :  +0 off
      :  +1 on
NI+NBITS+5  data word addresses (24 bits, additional words if needed)
+6  }
  :  }  NDATA data word contents to be updated.

Bits or data words not explicitly indicated are left untouched.

5.3.9 **Action X** - user option

Same format as for actions A or D.

5.3.10 **Action Y** - user option

Same format as for action L.

5.3.11 **Action Z** - user option

Same format as for action U.

5.3.12 **Global actions** *SELECT, DELETE* - select, delete frames

Same format as for action H, except that NDATA will always be zero.
<table>
<thead>
<tr>
<th>Appendix</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Organisation of program EDIT.</td>
</tr>
<tr>
<td>2</td>
<td>Lay-out of data structure used for examples.</td>
</tr>
<tr>
<td>3</td>
<td>FQX/FQT record format.</td>
</tr>
<tr>
<td>4</td>
<td>Structure of user header information in FQX/FQT precursor records.</td>
</tr>
<tr>
<td>5</td>
<td>Example for definition of bits in user header information.</td>
</tr>
<tr>
<td>6</td>
<td>List of condition ID's used by EDIT.</td>
</tr>
<tr>
<td>7</td>
<td>Examples for program run decks on CDC 7000, IBM 370/168, PDP10.</td>
</tr>
<tr>
<td>8</td>
<td>Handling of action requests in AREC, ACT.</td>
</tr>
</tbody>
</table>
EDIT: ORGANISATION OF PROGRAM

EDMAIN

EDINIT

INITIT

initialize program

REDIT

main steering for phase 1

RDGST

read/decode action requests

RDUI

RDUI

RDUIN2

read/decode action requests

EVINIT

initialise banks and I/O flags

called once only

SORT

COMB

order/combine MT records

AREC

dress list of action requests

EVSEL

select/delete

EVSTAT

run statistics

EVOT

output MT

STRIP

strip-off headers

ACT

main steering for execution of action requests

A

B

specific action requests

Z

USER

general action requests

STAT

APPENDIX 1
LAY-OUT OF DATA STRUCTURE

linear structure
with bank
KS, EMI, ...
FOX/FQT RECORD FORMAT

Logical record:

precursor  banks  banks  banks

Precursor:

- control words

- user header (max. 400 words)

- table entries for re-location of banks

Banks:

- HYDRA banks, only banks marked are output by FOXOUT.

- Records are automatically split into optimum physical record size for each machine, for FQT blocks of 3600 8-bit bytes will be used (see HYDRA System Manual B/D FOX and FQT).

Information on tape:

1 frame 1 frame 1 frame 1 frame

+ roll header banks banks precursor only special header + roll header
STRUCTURE OF USER HEADER INFORMATION IN FOX/FQT
PRECURSOR RECORDS

On input of the precursor record the user header is transferred to array NHEAD.

<table>
<thead>
<tr>
<th>Word</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>total number of words following (&gt;19)</td>
</tr>
<tr>
<td>2</td>
<td>number of words of standard information following (=16)</td>
</tr>
<tr>
<td>3</td>
<td>experiment number</td>
</tr>
<tr>
<td>4</td>
<td>roll number</td>
</tr>
<tr>
<td>5</td>
<td>frame number</td>
</tr>
<tr>
<td>6</td>
<td>measurement number (or event number or combination of both)</td>
</tr>
<tr>
<td>7</td>
<td>pass number</td>
</tr>
<tr>
<td>8</td>
<td>update flag</td>
</tr>
<tr>
<td>9</td>
<td>date of last action by operator</td>
</tr>
<tr>
<td>10</td>
<td>program version</td>
</tr>
<tr>
<td>11</td>
<td>title version</td>
</tr>
<tr>
<td>12</td>
<td>run number</td>
</tr>
<tr>
<td>13</td>
<td>beam type</td>
</tr>
<tr>
<td>14</td>
<td>laboratory</td>
</tr>
<tr>
<td>15</td>
<td>record type (1 = run header, 2 = experiment header, 3 = roll header, 4 = special header, 5 = frame header, this is the standard value, &gt;5 = special application)</td>
</tr>
<tr>
<td>16</td>
<td>free for special information needed by laboratories</td>
</tr>
<tr>
<td>17</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td></td>
</tr>
</tbody>
</table>

19 \( \text{NWB} = \) number of words with bit information

20 \( \text{NWE} = \) number of words with addresses for extra information for bits

21 \( \text{dynamic part} \)

\[
\begin{align*}
\text{NWB words with 24 bits/word (numbered from right to left, starting at bit 1)} \\
\text{NWE words with bit number + address relative to start of array NHEAD (bit } \alpha_i \text{ *10000} + \text{ adress } n_i) \\
(\text{additional information for bit } \alpha_i) \\
\text{information for other bits, stored without gaps.}
\end{align*}
\]

All words in the user header (standard information and \( \text{NWB} + \text{NWE} \) words) are integer with 24 bits maximum. Additional information for bits is floating point, including the first word which gives the total number of words of information for the bit.
DEFINITION OF BITS IN USER HEADER INFORMATION

[obligatory] = add. information must be given if bit is set to 1

<table>
<thead>
<tr>
<th>bit</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>scan 1 done</td>
</tr>
<tr>
<td>2</td>
<td>scan 2 done</td>
</tr>
<tr>
<td>3</td>
<td>scan 3 done</td>
</tr>
<tr>
<td>4</td>
<td>measurement done</td>
</tr>
<tr>
<td>5</td>
<td>measurement incomplete</td>
</tr>
<tr>
<td>6</td>
<td>unmeasurable [obligatory]</td>
</tr>
<tr>
<td>7</td>
<td>re-measurement to be done</td>
</tr>
<tr>
<td>8</td>
<td>geometry passed</td>
</tr>
<tr>
<td>9</td>
<td>geometry incomplete</td>
</tr>
<tr>
<td>10</td>
<td>geometry decisions added</td>
</tr>
<tr>
<td>11</td>
<td>kinematics passed</td>
</tr>
<tr>
<td>12</td>
<td>kinematics incomplete</td>
</tr>
<tr>
<td>13</td>
<td>kinematics decisions added</td>
</tr>
<tr>
<td>14</td>
<td>frame to be ignored [obligatory]</td>
</tr>
<tr>
<td>15</td>
<td>1 event on frame</td>
</tr>
<tr>
<td>16</td>
<td>2 events on frame</td>
</tr>
<tr>
<td>17</td>
<td>3 events on frame</td>
</tr>
<tr>
<td>18</td>
<td>&gt; 3 events on frame</td>
</tr>
<tr>
<td>19</td>
<td>event 1 to be ignored [obligatory]</td>
</tr>
<tr>
<td>20</td>
<td>&quot; outside fiducial volume 1</td>
</tr>
<tr>
<td>21</td>
<td>&quot; &quot; &quot; &quot; 2</td>
</tr>
<tr>
<td>22</td>
<td>&quot; &quot; &quot; &quot; 3</td>
</tr>
<tr>
<td>23</td>
<td>event 2 to be ignored [obligatory]</td>
</tr>
<tr>
<td>24</td>
<td>&quot; outside fiducial volume 1</td>
</tr>
<tr>
<td>25</td>
<td>&quot; &quot; &quot; &quot; 2</td>
</tr>
<tr>
<td>26</td>
<td>&quot; &quot; &quot; &quot; 3</td>
</tr>
<tr>
<td>27</td>
<td>event 3 to be ignored [obligatory]</td>
</tr>
<tr>
<td>28</td>
<td>&quot; outside fiducial volume 1</td>
</tr>
<tr>
<td>29</td>
<td>&quot; &quot; &quot; &quot; 2</td>
</tr>
<tr>
<td>30</td>
<td>&quot; &quot; &quot; &quot; 3</td>
</tr>
</tbody>
</table>

associated wall interaction unmeasured
associated wall interaction measured [obligatory]
wall interaction unmeasured
wall interaction measured [obligatory].
<table>
<thead>
<tr>
<th>bit</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>49</td>
<td>EMI information added</td>
</tr>
<tr>
<td>50</td>
<td>EMI not available</td>
</tr>
<tr>
<td>51</td>
<td>EMI incomplete</td>
</tr>
<tr>
<td>52</td>
<td>picket fence inform. added</td>
</tr>
<tr>
<td>53</td>
<td>&quot; &quot; not available</td>
</tr>
<tr>
<td>54</td>
<td>&quot; &quot; incomplete</td>
</tr>
<tr>
<td>55</td>
<td>veto counter</td>
</tr>
<tr>
<td>56</td>
<td></td>
</tr>
<tr>
<td>57</td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>calorimeter</td>
</tr>
<tr>
<td>59</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>other</td>
</tr>
<tr>
<td>62</td>
<td></td>
</tr>
<tr>
<td>63</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>flux information [obligatory]</td>
</tr>
<tr>
<td>65</td>
<td></td>
</tr>
<tr>
<td>66</td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>v-spectrum [obligatory]</td>
</tr>
<tr>
<td>68</td>
<td></td>
</tr>
<tr>
<td>69</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>other beam/chamber inform.</td>
</tr>
<tr>
<td>71</td>
<td></td>
</tr>
<tr>
<td>72</td>
<td></td>
</tr>
</tbody>
</table>

| 73  | EDIT pass done [obligatory]                                               |
| 74  | last EDIT pass incorrect                                                  |
| 75  | header information verified                                                |
| 76  | header information incorrect                                              |
| 77  | data structure verified                                                   |
| 78  | data structure incorrect                                                  |
| 79  | data structure relinked                                                   |
| 80  | discarded banks dropped                                                   |
| 81  | part of data structure dropped                                            |
| 82  | structure transferred to DST format (HYDRA format)                        |
| 83  | structure transferred to DST format (non-HYDRA)                           |
| 84  | information has been selected                                             |
| 85  | information has been deleted                                              |
| 86  | this is stripped-off header information                                    |
| 87  | information converted to IBM (FQTOU)                                     |
| 88  | information converted from IBM (FQTOIN)                                   |
| 89  | information merged                                                        |

status of equipment

status of editing
LIST OF CONDITION ID's

100,0 (EVIN)  single EOF detected on MT file
101,0 (EVIN)  two consecutive EOF detected on MT file
103,0 (EVIN)  bad input data
104,0 (EVIN)  no more input tapes for MT
105,0 (M)     single EOF for merge input file
110,0 (EVOT)  data structure empty
111,0 (EVOT)  not enough memory
112,0 (EVOT)  no pending data structure or bad data
113,0 (EVOT)  unexpected EOF
114,0 (EVOT, NEWTOT)  too many output tapes requested
120,0 (NEWIN) .new input tape loaded
121,0 (NEWIN)  too many input tapes requested
131,1 (STRIP) unit LUNSTR not defined
132,1 (M)      unit LUNMRG not defined
141,1 (AREC,M) end of input information
142,0 (VAREC)  illegal request for inserted frame
161,1 (PARDAT) wrong code
162,0 (PARADAT) mass index 0
200,0 (RDACT) error on action cards
201,0 (NEXTAC) input buffer for action cards overflows
202,0 (ACTINC) array ACT overflows (buffer for requests)
203,0 (KEYSET) illegal action generated from GST cards
204,0 (PCODE)  wrong mass code from GST cards
205,0 (ENDT)   wrong end-type from GST cards
206,0 (RGDST)  format error on GST card
207,1 (INITIT) obligatory header words not defined
208,0 (LMCTIT) requested code not found in title MC
209,0 (IGETH, ISETH) requested code found in MC, but wrong type
211,0 (LDSTIT) requested bank not found in structure
212,0 (LDSTIT) requested bank not found in title DS

trap class 10:  serious error occurred and program is terminated
                included are: 114, 131, 132, 207

trap class 20:  return to EVLOOP and continue program
                included are: 141, 161.
EXAMPLES OF PROGRAM RUN DECKS

For CDC 7000 the complete deck is listed, including job control cards, PATCHY cradle, title blocks and action requests. Title and action requests were identical for tests on IBM 370/168 and PDP10 and are not repeated.

You can get an up-to-date run deck for CDC 7000 from K. Gieselmann, also PAM files GADGET and EMIANA with additional user routines as example.

X3GOK.T23. TEST OF EDIT PROGRAM ON CDC7600
ACCOUNT,KELLNER,EP,T120ED.
FIND,HYCDE,HYDRACDEPAM,ID=PHLIB.
FIND,HYPAM,HYDRAPAM,ID=PHLIB.
FIND,HYDUM,HYDRADUMMYLG0,ID=PHLIB.
FIND,HYLIB,HYDRAHIL,ID=PHLIB.
FIND,ULIB,ULIB,ID=PHLIB.
FIND,GSLIB,GENSECILB,ID=PHLIB.
FIND,LIB,7600LIBRARY,ID=PROGLIB.
FIND,GENUTY,HAGENUMYPAM,ID=PHLIB.
LIBRARY,ULIB,LIB.
FILE,OLD,RT=2,PL=90.
ATTACH,OLD,EDIT1R,ID=EP120KELL,ST=CCQ.
YTOBIN,OLD,PAM.
YPATCHY,PL=10000.
FTN,I=ASM,T,TS.
RETURN,HYCDE,GENUTY,GSLIB,ULIB,PAM,ASM,OLD.
LIBRARY,HYLIB,LIB.
FIND,TAPE8,EDITIN158,RP=EP120KELL,ST=CCQ. ROLL 158
FIND,TAPE11,EDITIN157,RP=EP120KELL,ST=CCQ. ROLL 157
FIND,TAPE12,EDITIN159,RP=EP120KELL,ST=CCQ. ROLL 159
MAP,PART.
SLOAD,HYDUM,2QDUMMY,2QMULTD.
LGO,INPUT,PL=10000.
CATALOG,TAPE21,EDITOUTCDC,ID=EP120KELL,RP=999,ST=CCQ.
CATALOG,TAPE7,EDITSTRIP,ID=EP120KELL,RP=999,ST=CCQ.
*EOR
+EXE.
+USE,P=QCDE.
+USE,P=USEEDIT,P=CDC,P=FQXFQT.
+ADD,P=ACT,D=ACT,C=31.....CLEAR Q FOR TESTS TO REDUCE DQSNAP PRINTS
CALL MQGARB
+DEF, Z= SPACE. REDEFINE SIZE OF Q-VECTOR
+ SPACE(14000)
+DEF, Z= EDHEAD. REDEFINE ADDRESSES OF HEADER WORDS AND BITS
DIMENSION IHLOC(10), IHBITS(17)
DATA NHLOC/10/, NhBITS/17/
DATA IHLOC/2,3,4,0,0,9,5,6,7,8/
DATA IHBITS/0,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89/
+PAM,F=HYCDE.
+PAM,F=PAM.
+PAM,F=GENUTY.
+PAM,F=HYPAM.
+QUIT.
*----- TITLE FOR EDIT PROGRAM - DETAILS IN WRITEUP
* AC 43 (IS/(14(A4,1X)))
H CKT  MKT  APF  CPF  DPF  LPF  MPF  RPF  UPF  ATF  CTF  DTF  LTF
KTF UTF  UTF  AMF  CMF  DHR  RMF  UMF  AUS  CUS  DUS  RUS  UUS  AIN
DIN RIN  ASH  DSH  RSH  USH  DEH  DPHY  UPHY  CPLS  DPLS  CTLS  DTLN
* DS 93****
  1,1  (I2,3X,15)
  23  (I2,3X,13(A4,1X))
  23,13  (I2,3X,13(A4,1X))
  3  2  2  0  2  4  2  0  0  4  0  3  0
  4  3  2  0  4  2  0  2  0  0  5  0
  5  1  2  1  1  0  3  0  1  0  0  0  1  0
  6  0  1  1  1  0  1  1  1  0  0  0
  7  23,13  (I2,3X,13(A4,1X))
  8  0  3  3  0  1  3  3  0  0  0  0
  9  0  112****
MC
  11,11 (I2,3X,215,4(A4,1X),515)
  0  4  35  PF  TF  MF  23  4  4  0
  35,13  (I2,3X,13(A4,1X))
  35,13  (I2,3X,13(A4,1X))
  1  NH  EXP  RUN  BEAM  LAB  SC2  SC3  PART  REM  EV1  EV2  EV3  EVN
  2  FLUX  CLI  CL2  CL3  CL4  CL5  CL6  CL7  CL8  CL9  GAM  KKN  XLAB
  3  VEFP  FINX  STP  XLAB  TEMD  PUM  MAss  PL  DPL
  35,13  (I2,3X,13(A4,1X))
  4  1  3  12  13  14  2  3  12  14  26  27  28  29
  5  64  97  163  109  115  121  127  133  139  145  13  14  1
  6  12  7  0  1  7  4  1  2  5
  35,13  (I2,3X,13(A4,1X))
  7  3  3  3  3  3  1  1  1  1  1  1  1  1
  8  1  1  1  1  1  1  1  1  1  1  2
  9  2  2  2  2  2  2
  10  RCTR  6(8F5.0)
  11  U.  0.  10.  141.  161.
  12  RCTR  8(3F5.0)
  13  U.  0.  11.  114.  131.  132.  207.
*FINISH
157,50,CFP,51./52.*1=6/4/9,2=ON,2=10/.3/.0,3/.7+1=18.
157,50,CFP,51./53.*1=6/4/9,2=ON,2=10/.3/.0,3/.0+1=18.
157,50,CFP,51.*1=6/4/12,+2=111111.1
157,50,H,*1=3/157/50/5/1=ON,11=ON
157,50,CFP,*1=1/50/5,2=ON/ON,2=12/ON/ON/OFF/OFF/ON
157,46,CFP,YA/GB,2=ON,2=ON,2=ON,2=12/1/11111.123,+7=18.
157,46,CFP,-/GB/.0,1=3/157/2/ON/ON/OFF/ON,2=-10/1/0.3/-0.3
157,144,CFP,YA/G5,2=1/4/2/4/1/1/1/1/1,1=ON,2=12/1/0.3/1234.567,+7=18.
157,46,CUS,YA/ZA/1.,*1=11/3/12,2=1/1/0/1/1,1,1=1.1/1.1/20.2/236.3,+10=111
157,46,CUS,-/ZA/2.,*1=3/4/2,1/1,1/1/1/1/1/1/1,1=2.19/1/0.3/0.3/0.3/5
157,46,DEH
157,46,H,11=ON,15=ON,27=ON/ON,45=ON,69=ON/ON
157,46,!!,*1=5/157/2/1/3,64=ON,2=1.3/157/16.
157,46,UPF,YA,16=ON/ON/ON,16=10.5,+1=1.25
157,46,UPF,YA,10=ON/ON/ON,16=1.25,1=1.25
157,46,UPF,YA,10=ON/ON/ON,16=0.777/22/6.99,7=16.9,+6=2.5
157,46,DPF/Z
157,46,DPS, YA/3.
158.*"IC":MKT
157,50,*,*1=7/157/50/5/12/1/1/1/1/3,51=ON
157,292,DELETE,1=ON
*"*TAPN,2.
*"*HIST
*"*STAT
*"*STRIP
For IBM 370/168 the following deck was used for tests of the program:

```plaintext
//GOKEDIT JOB GOK$XQ, MSGCLASS=U, TIME=1
  EXEC YPATCHY,
  PAMDSN='PH.LIB.HYCEPAM',
  PAM2DSN='XQ.GOK.EDITPAM',
  PAM3DSN='PH.LIB.HYPAH'
//Y.SYSIN DD *
+EXE.
+LIST,P=*CRA.
+USE,P=*IBM.
+USE,P=*USEDIT,P=IBM.
+ADD,P=ACT,D=ACT,C=31......CLEAR Q FOR TESTS TO REDUCE DQSNAP PRINTS
  CALL MQGARB
+DEF, Z=EDHEAD
  DIMENSION IHLOC(10), IHBITS(17)
  DATA NHLOC/10/, NHBITS/17/
  DATA IHLOC/23, 4, 8, 0, 9, 5, 6, 7, 8/
  DATA IHBITS/73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 33, 84, 85, 86, 87, 88, 89/
+PAM,LUN=11.
+PAM,LUN=12.
+USE,P=IBS,M,T=INHIBIT.
+QUIT.
/*
  EXEC XFORTCLG,
  CPRD=DUMMY,CPRM='SYSOUT=*',
  LLB5='PH.LIB.HYLIB',
  LPRM='LET,LIST,MAP',
  GRON=512K
/C.SYSIN DD DSN=8ASG, DISP=SHR
/G.FT06F001 DD SYSOUT=*, DCB=(RECFM=FBA, LRECL=133, BLKSIZE=133),
  SPACE=(CYL,(1,1),RLSE)
/G.FT10F001 DD DUMMY
/G.FT06F001 DD DSN=XQ.GOK.FILE2, DISP=SHR, UNIT=SYSDA, VOL=REF=XQ,
  DCB=(RECFM=VBS, LRECL=3600, BLKSIZE=3600)
/G.FT11F001 DD DSN=XQ.GOK.FILE1, DISP=SHR, UNIT=SYSDA, VOL=REF=XQ,
  DCB=(RECFM=VBS, LRECL=3600, BLKSIZE=3600)
/G.FT11F002 DD DSN=XQ.GOK.FILE3, DISP=SHR, UNIT=SYSDA, VOL=REF=XQ,
  DCB=(RECFM=VBS, LRECL=3600, BLKSIZE=3600)
/G.FT21F001 DD DUMMY
/G.FT31F001 DD DSN=66TEMP3, UNIT=SCR, DCB=(RECFM=FBA, LRECL=133,
  BLKSIZE=1995), SPACE=(CYL,(1,1),RLSE)
/G.FT32F001 DD DSN=66TEMP4, UNIT=SCR, DCB=(RECFM=FBA, LRECL=133,
  BLKSIZE=1995), SPACE=(CYL,(1,1),RLSE)
/G.FT33F001 DD DSN=66TEMP5, UNIT=SCR, DCB=(RECFM=FBA, LRECL=133,
  BLKSIZE=1995), SPACE=(CYL,(1,1),RLSE)
/G.FT51F001 DD DSN=66TEMP1, UNIT=SCR,
  DCB=(RECFM=VBS, BLKSIZE=6160),
  SPACE=(6160,(10,5)), DISP=(NEW,DELETE)
/G.FT52F001 DD DSN=66TEMP2, UNIT=SCR,
  DCB=(RECFM=VBS, BLKSIZE=6160),
  SPACE=(6160,(10,5)), DISP=(NEW,DELETE)
/G.SYSIN DD *
---- TITLE FOR EDIT PROGRAM - DETAILS IN WRITEUP
... title and data cards as used for test on CDC7600
```
Tests on the PDP10 were done with the following example:

TY EDIT.CRA
+EXE.
+USE: P=HYGRA.
+USE: P=GDE.
+USE: P=USEDIT,P=PDP,P=FOT.
+ADD: P=ACT,D=ACT,C=31,... CLEAR 0 FOR TESTS TO REDUCE DOSNAP PRINTS
CALL MOGAR!
+DEF, 2= EDHEAD
DIMENSION IHLDC(10), IHBITS(17)
DATA NHLD C/10/, NHBITS/17/
DATA IHLDC/2,3,4,0,0,9,5,6,7,8/
DATA IHBITS/73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89/
+PAM: LUN=11,T=H.YCDE
+PAM: LUN=12,T=EDIT4
+QUIT.

TY EDIT.CMD
%- MAXCD=50P %NEWPAGE %COMMH,COMM=115360 %COMMH,END=1 [200+16
0]HYDCOM,REL
ERASYS,REL[10,101]% INC: (PAGED)/
[200:160] MOMEM,REL% INC: (INMEM)/
I-------------------------------------------- INITIATION / TERMINATION / DUMP --
ERASYS,REL[10,101]% INC: (RENT)/
HYLIB,REL[300,302]% INC: (FATAL, GABEND, GOSUM, GTRAP, JOINT, ROINIT, ROEND
JOINIT,MOINIT)/
HYLIB,REL[300,302]% INC: (OSNAP, DDBank, DDATA, DTYP, DOCRIT, DOJTR)/
HYLIB,REL[300,302]% INC: (OSORT, DNOP, KLAST, KNAME, NODRL, MOGAR)/
HYLIB,REL[300,302]% INC: (MOWORK, LOFIND, QTITLE, MOZERO, POTELL, MOLIFT)/
HYLIB,REL[300,302]% INC: (JOJUMP, GTOUCH, JOBACK, NOFIND, OBLow)/
GENLIB,REL[300,302]% INC: (JBUCH, OBLow, ICUOMP, SBIT, SBID, SBID, SBIT)/
GENLIB,REL[300,302]% INC: (JSYDOR, JRSYD, JUCOPY, JUCOPY2, JFRONT, JUPKBY)/
GENLIB,REL[300,302]% INC: (JZERO, JBYD)/
EDIT,REL
QBANG,REL
HYLIB,REL[300,302]% INC: (JWRITE, ROACC, ROEDIT, MOWIF, OPAGE, OTCHID)/
FO,REL
I----------------------------------------- EVERYTHING ELSE ------------------
/SEARCH [300,302]HYLIB,REL, HYLIB,REL, GENLIB,REL

RU EDIT.
MOINIT EXECUTED.

DECLARATION OF I/O UNITS AND FILES
<ITBU=31 AND ITB2=32 ARE AUTOMATICALLY ASSIGNED>

GIVE LUN AND FILE NAME FOR THE FOLLOWING STREAMS
IN FORMAT 12,1X,24S
IF LUN=0, THIS STREAM WILL NOT BE OPENED

ACTION REQUEST MATERIAL, STREAM 1:
+20,AR1.DAT
ACTION REQUEST MATERIAL, STREAM 2:
0
MASTER TAPE 1 (INPUT):
+21,E157.IN
MASTER TAPE 2 (MERGE INPUT):
0
NEW MASTER TAPE (OUTPUT):
+22,ED.OUT
STRIP OUTPUT (HEADERS ONLY, OPTIONAL):
0
LOG OUTPUT:
0
ALTERNATE BCD OUTPUT (++PRINT):
0

END OF EXECUTION
CPU TIME: 4.11  ELAPSED TIME: 1:51.26
EXIT
HANDLING OF ACTION REQUESTS IN AREC, ACT

This small structure is built up by AREC from all accumulated action requests and used by ACT to branch to the different subroutines to handle the requests. Banks FRC and ARC are modified to signal correct execution or failure.

Each record of thegregory contains exactly the information described in sect. 5.2.3 for the format of the records on file AR.

Contents of banks ARC, FRC, RFC can be found in bank descriptions at the end of EDIT PAM.