IRAM Memo 2014-?

Mrtcal Programmer Manual

S. Bardeau[1], J. Pety[1,2], A. Sievers[3]

1. IRAM (Grenoble)
2. LERMA, Observatoire de Paris
3. IRAM (Granada)

July, 22^{th} 2014
Version 0.3

Abstract

Some abstract here.
Related documents: some related documents here.

Contents

1 Chunks and chunksets

For the processing of the IMBFITS files, **MRTCAL** introduces the concept of *chunks* (see Fig 1). One dump of the IMBFITS `DATA` column is a collection of such chunks. **MRTCAL** can make no assumption on the way they are ordered (this comes from ???). The `PART` column indicates to which spectrum each chunk belongs (spectra are identified with integer numbers). Each "set of chunks belonging to the same final spectrum" is called a *chunkset* in the **MRTCAL** nomenclature. From one chunkset, **MRTCAL** will produce one spectrum[1].

The chunks in each `DATA` row can belong to several spectra (*e.g.* 8 for FTS), and by definition to as many chunksets. **MRTCAL** maps those chunksets internally as a `chunkset_1D` Fortran structure, *i.e.* an array of chunksets.

The `DATA` column has as many rows as the number of pixels of the receiver times the number of time dumps. This introduces 2 new dimensions which are used to map all[2], the `DATA` column into a `chunkset_3D` Fortran structure. The 2 new dimensions here are *pixels* and *time* (see Fig 2).

For its internal needs, **MRTCAL** has to refer to a single time dump in the 3rd dimension. It can also time-average this 3rd dimension. In those 2 use-cases, the final product is then available through a `chunkset_2D` Fortran structure.

1.1 Chunks slicing

In order to be able to make calibration at a bandwidth smaller than the hardware chunks, **MRTCAL** is able to map the raw data block read with CFITSIO into smaller *memory* chunks. The main advantage is that this approach has very well defined boundaries: when the IMBFITS `BACKEND` table is read, it can be replaced on-the-fly with its sliced version. All the rest of the code does not have to know this happened. The downside is that the more memory chunks are defined, the more overheads they carry (e.g. description of their spectroscopic axis, calls to chunk-by-chunk subroutines, stitching, etc). Experience shows that slicing the 1400 MHz of FTS chunks into 14 chunks of 100 MHz introduces a penalty of 10% in the whole computation time (reading, slicing, calibrating, writing).

In practice, user can request the bandwidth he wishes. In order to slice the `USED` channels into comparable pieces, **MRTCAL** will choose the nearest bandwidth which gives an integer number of slices:

$$N = nint\left(\frac{USED \times |SPACING|}{W}\right) \tag{1}$$

where W is the requested bandwidth. N must also be corrected so that there is at least 1 and at most `USED` channels per subchunk:

$$USED \geq N \geq 1 \tag{2}$$

While the `CHANS` value (total number of channels) is often a multiple of 2, `USED` channels have no special value. In particular, it may not be a multiple of N. Their relationship can be written as:

$$USED = q \times N + r \tag{3}$$

where q and r are respectively the integer quotient and the remainder of the division $USED/N$. Since $0 \leq r \leq N-1$, `USED` can be divided in N subchunks, r of them having one extra channel:

$$USED = (q \times (N - r)) + ((q + 1) \times r) \tag{4}$$

This gives the best channels division and a negligible difference of subchunks bandwidth.

ZZZ Say a word about the "famous" special channel with its half weight. Slicing does not have any special problem with it.

[1] For debugging purpose, **MRTCAL** can also produce one spectrum per chunk.

[2] **MRTCAL** can also map a fraction of the `DATA` column, process it, and then iterate on the next fraction. This can be useful on memory-limited machines.

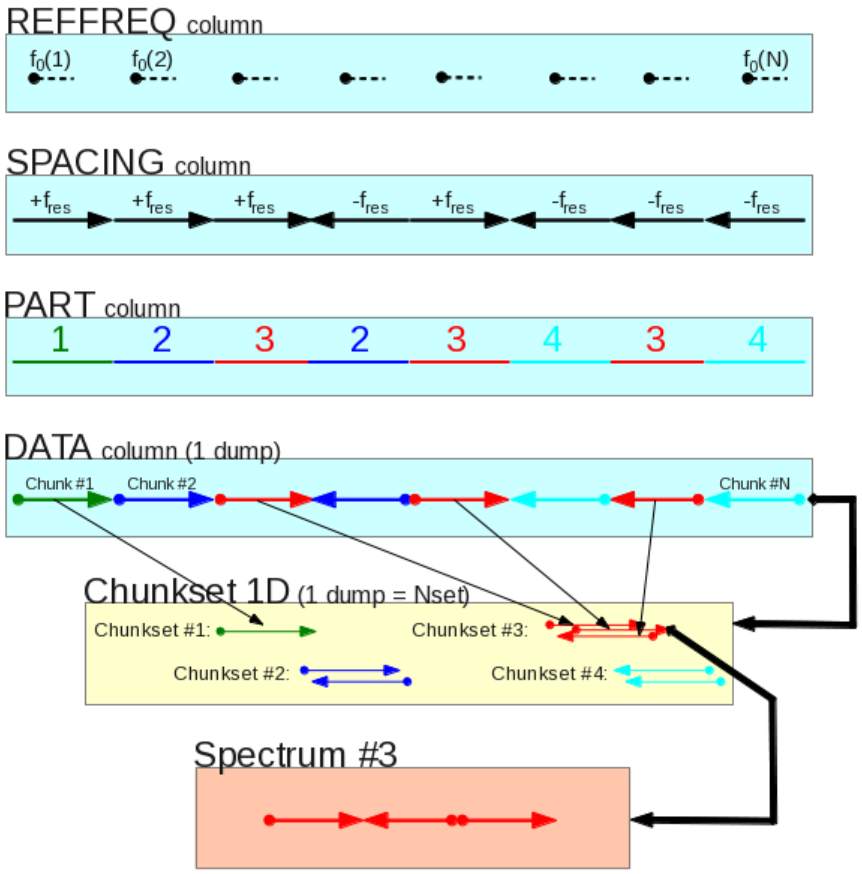

Figure 1: How one dump in the IMBFITS `DATA` column is divided into chunks, gathered by family (chunksets), and reordered to produce the spectra.

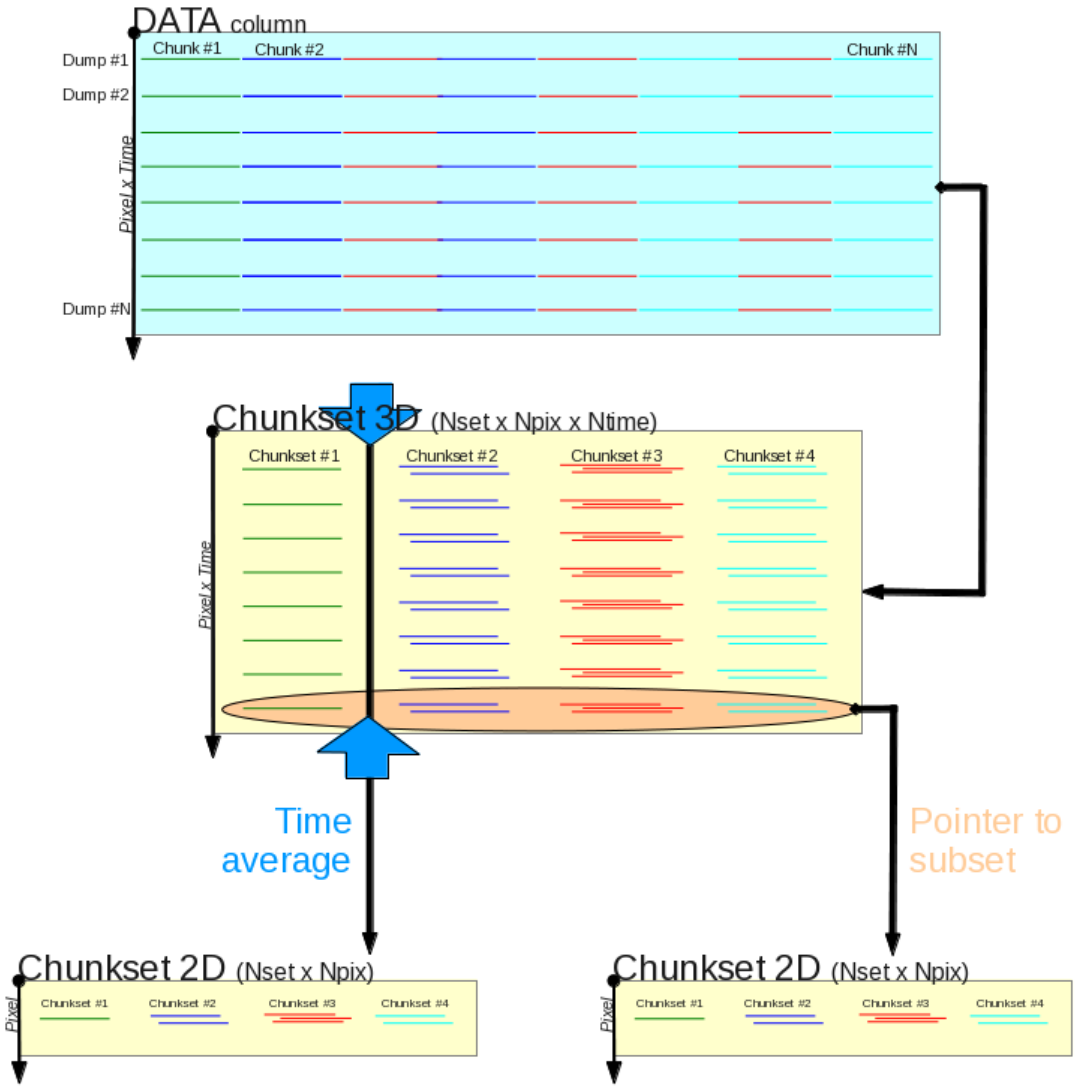

Figure 2: How all the dumps in the IMBFITS `DATA` column is divided into 3D chunksets. After this, they can be time-averaged, or can be accessed one by one.

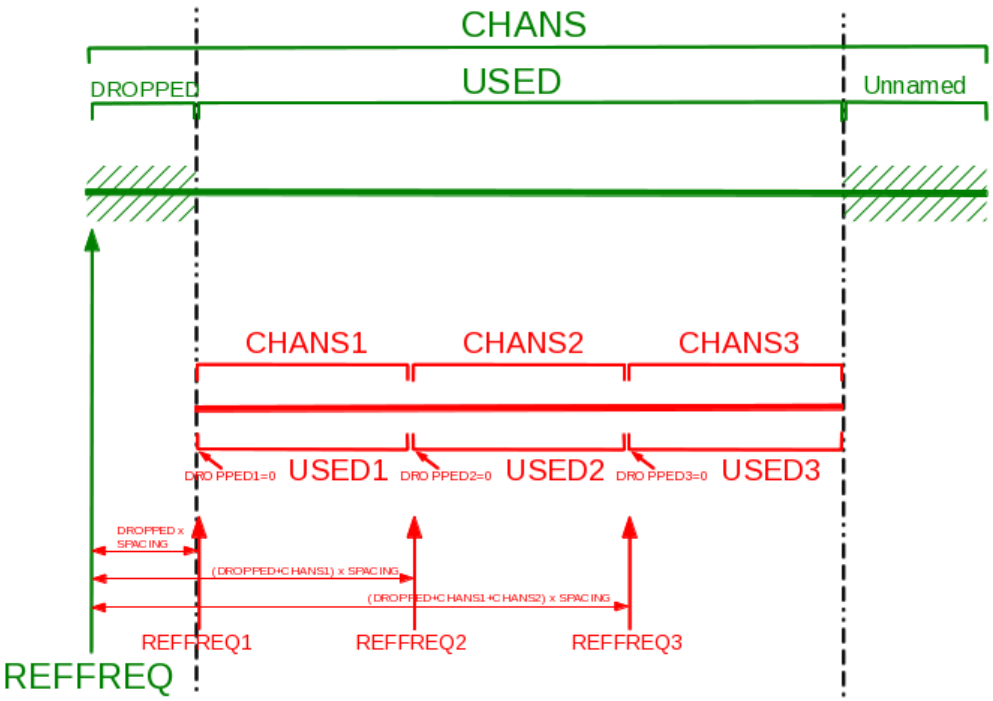

Figure 3: Hardware chunks can be virtually divided into smaller pieces just by redefining the `BACKEND` table. In this example, the table is sliced from 1 chunk to 3 subchunks (3 times more lines in the table). The same raw data block read with CFITSIO will be mapped by Mrtcal with a larger number of *memory* chunks.

1.2 Polarimetry

The presence of polarimetric measurements in the IMB-FITS file is described in the `POLAR` column of the `BACKEND` table. For a non-polarimetric scan, all the chunks in the `POLAR` column have the `NONE` value. For a polarimetric scan, values can be `NONE`, `AXIS`, `REAL` or `IMAG` (note that non-polarimetric chunks -`NONE`- can be mixed with polarimetric ones).

When processing a polarimetric scan, **MRTCAL** will offer 2 modes:

1. full polarimetry support, with calibration of both self-products (`AXIS` *i.e.* HH and VV) and cross-products (`REAL` and `IMAG` *i.e.* HV and VH), plus non-polarimetric data (`NONE`),
2. minimal support, *i.e.* calibration of self-products and non-polarimetric data (`AXIS` and `NONE`).

In order to achieve the second point, **MRTCAL** skips the `REAL` and `IMAG` chunks at read time of the `BACKEND` table.

Finally, the polarimetric status (*does this scan contains polarimetric data?*) is saved in the index file. The command `MFIND /POLARIMETRY YES|NO|*` allows to make the desired selections in this context.

2 Positions

2.1 Reference

For each spectrum, the Class Data Format expects a reference (usually source) position. In **MRTCAL**, this reference is taken from the `LONGOBJ` and `LATOBJ` (*Source longitude and latitude in basis frame*) in the Scan HDU.

From this reference, the IMB-FITS format can define an intermediate reference position, through the `SYSOFF`, `XOFFSET` and `YOFFSET` columns in the Scan table:

- if only the *Nasmyth* row is present, an implicit *projection* row is added at read time with (0,0) offsets,
- if the *projection* row is available, the associated offsets are added to the antenna offsets (see next section) and used into the **CLASS** Data Format,
- if something else than *projection* is available, *e.g. horizontalTrue*, a specific support is not yet available in **MRTCAL** and an error is raised. However, if the associated offsets are (0,0)[3], Mrtcal will tolerate this system of offsets. This is a temporary solution, waiting for a full support of all systems.

2.2 Offset interpolation

In addition to the reference position(s) set up above, the 4 following elements have to be computed:

- lambda offset from reference position,
- beta offset,
- azimuth,
- elevation

All these elements are available in the *Antenna Slow* table, at a typical sampling rate of 1 Hz. Since the spectra dumps can be produced at different time sampling (no assumption is made on the rate or its variations), each spectrum positions are interpolated from the Antenna table thanks to their respective *Modified Julian Day* values:

[3]This can happen *e.g.* if the user has commanded `OFFSETS 0.0000000E+00 0.0000000E+00 /SYSTEM "trueHorizon"` in **PAKO**

$$f = \frac{mjd_S - mjd_A(j)}{mjd_A(j+1) - mjd_A(j)} \tag{5}$$

where mjd_S is the spectrum MJD value, and $mjd_A(j)$ is the MJD value of the j^{th} trace in the Antenna Slow table. j is computed thanks to a dichotomic search in the table such as:

$$mjd_A(j) \leq mjd_S < mjd_A(j+1) \tag{6}$$

f being the interpolation fraction between the j^{th} and $j+1^{th}$ trace, the positions are interpolated by:

$$l_S = l_A(j) + f \times (l_A(j+1) - l_A(j)) \tag{7}$$

where l_S and l_A are the spectrum and antenna lambda offsets respectively. Same formula applies for the beta, azimuth, and elevation values.

If mjd_S is found beyond the Antenna Slow table limits, the boundary values are applied without extrapolation. However, this is not expected to happen since such spectra should be rejected since they are out of the on-track range.

2.3 Antenna Slow and Fast traces

The IMB-FITS files provide 2 antenna streams and associated tables: the slow (1 Hz) and the fast (128 Hz) traces. For each of those traces, some antenna position parameters are available, plus the associated time as MJD values.

As of today (IMB-FITS version 2), the 2 traces are in the same FITS extension. Both have the same number of rows, but the fast trace has 128 values per column and per row, while the slow trace has 1 value per column and per row. However, even if they have the same number of rows, they are INDEPENDENT: for example, their MJDs are misaligned by a ∼2 seconds shift (2 rows). They should not be correlated! In the IMB-FITS version 3, the 2 traces will be clearly separated in 2 FITS extensions, which will avoid any misinterpretation.

Note that the caveat exposed above explains why **MRTCAL** selects more dumps than **MIRA** in on-the-fly maps: **MIRA** looks at the antenna slow column `TRACEFLAG` to flag out MJD values in the antenna fast. Because of the ∼2 seconds shift between the 2 traces, more time dumps are discarded.

3 Tsys interpolation

As of today, **MRTCAL** provides 2 modes for the Tsys value.

In the *flat* mode, one Tsys value is computed for each chunk, at the middle of its associated bandwidth[4]. This unique value is then used for calibrating all the channels in the chunk, whatever its bandwitdth.

The second mode is *interpolation*. In this mode, the chunks are reordered by ascending frequency, and the Tsys value at the middle of their bandwidths is computed (same as the *flat* mode). Then, the Tsys are interpolated between the current chunk and its nearest neighbour, so that it is computed at the frequency of the considered channel. This has the advantage to avoid discontinuities in the Tsys along the whole bandwidth and thus to avoid calibration platforming. For the two half-chunks at the boundary of the whole bandwidth, where no neighbour chunk is available, the Tsys is extrapolated from the Tsys of the current and previous chunk.

4 Memory index

4.1 Strategy

MRTCAL indexes IMB-FITS files in the so-called *index files*. A raw index file named `index.mrt` can be produced with the command `INDEX BUILD` (see command help for more advanced uses). When one or

[4]Remember that it is possible to customize the chunk bandwidths.

more index files are then reopened for reading, the command `INDEX OPEN` builds a *memory index* named the "Input indeX" (IX). At this stage, IX gathers the summary of all entries in all opened index files. From this full set, the command `MFIND` can be used to make a selection of desired entries: this is the "Current indeX" (CX).

While the primary purpose of the index files is to reference IMB-FITS files, their other advantage is to store other *products* that can be derived from the IMB-FITS files. In particular, after the calibration, the commands `CALIBRATE` and `PIPELINE` will save new *versions* of each calibrated entry. These new versions have a modified calibration status (from *none* to *failed* or *done*), and optionally a calibration section added (storing the calibration results). These modified versions of the entries are themselves saved in an index file (either the original one, or a new one, depending on the user choice), AND are implicitly appended to IX. After the calibration process, it is then easy for the user to select from IX the calibrated entries (or failed, or pending, etc). The figure 4 shows a basic example of these steps.

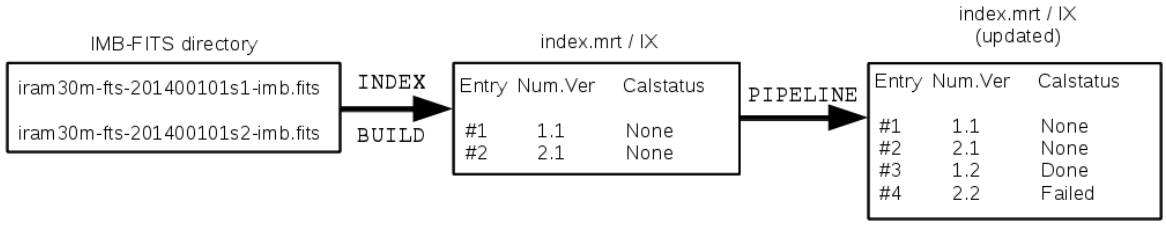

Figure 4: Basic example showing how the IMB-FITS files are first indexed, and then how the calibration command (here, `PIPELINE`) adds a new version of each (tentatively) calibrated IMB-FITS.

4.2 Contents

When the index file is loaded, its index content is duplicated in memory for all the entries. These elements are described in the Table 1.

However, these elements alone are not enough to deal with each IMB-FITS file referenced in the index. Extra numbers are associated on-the-fly, at load time. The Table 2 summarizes those numbers. In details, here is the exact meaning of those numbers:

- `num`: the *observation number* is a unique number associated to each IMB-FITS file. Uniqueness is ensured by a unique combination of the observing date, the scan number, and the backend. In a memory index, there can be several `version` of the same observation, meaning that the same IMB-FITS file is indexed several times (*e.g.* a first time when the index is built, a second time after the calibration). This number (and optionaly its version) is intended to be used as a frontend to the end-user. However, it is volatile: it can change from one session (or one INDEX OPEN) to another, depending on the index contents, and their number and order if several indexes are loaded in memory. In practice, the observation numbers are contiguous, starting from 1, and ordered by observing date, then by scan number, then by backend code.

- `mnum`: the *memory number* is the entry position in the input index (IX), and thus it provides a unique identifier for each entry indexed in memory. This means that the array `ix%mnum(:)` runs contiguously from 1. Since IX is the absolute reference for all sub-indexes (*e.g.* the current index CX), each entry of any index can easily be identified thanks to this back pointer (*e.g.* `cx%mnum(ient)`).

- `fnum`: the *file number* is the entry position in its associated index file. This number can not be implicit, because several index files can be loaded in memory, and because the entries are reordered by date, scan and backend at load time.

- `idir`: the *directory code* identifies each index file loaded in memory, so that each entry knows easily to which index file it comes from. This code also identifies the directory the IMB-FITS can be found

Table 1: Memory index elements coming directly from the index file.

Parameter	Unit	Comment
`bloc`	records	Entry position in index file
`word`	words	Entry position its this record
`version`	—	Entry version
`telescope`	code	Telescope code
`projid`	—	Project id
`source`	—	Source name
`dobs`	gag_date	Observation date
`ut`	rad	Observation time
`lst`	sec	Local Sideral Time
`az`	rad	Azimuth
`el`	rad	Elevation
`frontend`	—	Receiver names
`scan`	—	Scan number
`backend`	code	Backend code
`obstype`	code	Observation type
`switchmode`	code	Switching mode
`filstatus`	code	Can read file or not?
`calstatus`	code	Calibration status
`filename`	—	IMB-FITS file (no path)
`itime`	nanosec	Last indexing time (from 01-jan-1970)

Table 2: Memory-only index elements associated to each entry

Parameter	Unit	Comment
`num`	—	Observation number
`mnum`	—	Entry position in the Input indeX (IX)
`fnum`	—	Entry position in the index file
`idir`	code	Associated index file and IMB-FITS directory
`sort`	—	Sorting array

in (remember the index file may not be hosted in the same directory; this is controled by the user through the command `INDEX /FILE`).

- `sort`: the *sorting array* is an indirection array which would order the memory index by date, by scan, and by backend. At load time, the entries are actually ordered in memory, so this array is not needed at this time. But later on, commands like `CALIBRATE` or `PIPELINE` can add new versions of the entries. These new entries are appended at the end of the memory indexes, in order to avoid breaking the cross references between indexes (in particular, the `cx%mnum(:)` back pointers to IX). Each application which needs to access the entries sorted by date, scan, and backend, should then use the sorting array. Typically, one has to loop on `cx%mnum(cx%sort(ient))` (with `ient` 1 to `cx%next-1`) to access the entries ordered correctly.

5 Scan date vs observation date: midnight issue

When a new scan is *prepared*[5] at the telescope, the associated IMB-FITS file name is constructed with the current date. Later in this process (up to several minutes after), the scan is actually *started*[6], and the current date and time are used to build the `MJD-OBS` and `DATE-OBS` in the FITS Primary header. If those two steps are performed exactly before and after midnight respectively, the file name and the Primary header refer to 2 different dates, e.g.

```
iram30m-fts-20120809s323-imb.fits:
  MJD-OBS =     56149.0038078704 / MJD at observation start
  DATE-OBS= '2012-08-10T00:05:29.000'
```

For several purposes, Mrtcal saves the *observing date and time* of each file in the index. In order to deal correctly with the above issue, it is decided to:

1. save the date as found in the file name, so that there is no difference for the user between the file name and the date exposed by **MRTCAL**,
2. the time value saved for this file refers to the above date.

The direct consequence of those 2 rules is that, in case of the midnight issue exposed above, the UT value will be larger then 24h. Note that the UT value associated to each file is saved for bookkeeping purpose, e.g. for analysis of the calibration though date and time. It is considered as a typical value for the whole scan. Each individual spectrum produced by **MRTCAL** uses its own date and time.

In details, the observing date saved in the index is used:

- to check the entry uniqueness (date, scan and backend triplet should be unique),
- to sort the entries (by date, scan and backend),
- through the command `MFIND /DATE`,
- in the output of the command `MLIST`,
- in the Sic variables `MDX%DOBS` and `MHEAD%KEY%DOBS`

and the associated UT value is used:

- to sort the entries,
- in the output of the command `MLIST`,
- in the Sic variables `MDX%UT` and `MHEAD%KEY%UT`

The user should be ready to encounter UT values larger than 24h in the `MLIST` output and larger than 2π in the Sic variables.

[5] *scan prepared* or *scan loaded* in the NCS nomenclature.
[6] *scan started* or *subscan 1 started* in the NCS nomenclature.

6 MRTCAL Language Internal Help

6.1 Language

```
                    MRTCAL\ Command Language Summary

CALIBRATE: Calibrate a single file
MCOPY    : copy the current index to a file
MDUMP    : DUMP the content of the Mrtcal buffers
MFIND    : Select entries from index file
INDEX    : Build/read the index of the IMB-FITS files
MLIST    : List the files selected by the FIND command
MJD      : Convert an ISO date to/from a Modified Julian Date
PIPELINE : Calibrate the Current indeX
READ     : READ IMBFITS file
MSETUP   : Tune some program behaviours
VARIABLE : Add Sic variables mapping internal buffers
MWRITE   : WRITE the class data format
```

6.2 CALIBRATE

```
    [MRTCAL\]CALIBRATE Num[.Ver] [/WITH Num[.Ver]]
```

Calibrate the file indentified by its Number. The calibration scan is chosen implicitly by the command, except if explicitly specified through the option /WITH.

Boths file numbers can be added a Version number. If not given, the latest version is selected.

6.3 MCOPY

```
    [MRTCAL\]MCOPY IndexFile
```

Copy the current index (from memory) to the given file (to disk). Any pre-existing file is overwritten. This command is useful for two main purposes:

- extract a subset from an index file, by combining the commands INDEX OPEN, MFIND (with criteria), and MCOPY, or
- merge several index files, by combining INDEX OPEN, INDEX APPEND (repeated), MFIND, MCOPY.

6.4 MDUMP

```
    [MRTCAL\]MDUMP [ALL|SUMMARY|FILE|PRIMARY|SCAN|FRONTEND|BACKEND|BACK-
DATA|ANTSLOW|ANTFAST|DATA|SUBSCANS] [/OUTPUT LogFile]
```

Display the content of all or one section from the internal memory buffers. Default section is ALL sections.

SUMMARY displays a summary of the main HDUs and the subscans (as equivalence classes). SUBSCANS displays the full list of subscans and re-reads the whole file to do so.

The option /OUTPUT will redirect the display to the given file. Default is on screen (STDOUT).

6.5 MFIND

[MRTCAL\]MFIND [/DATE Date] [/SCAN List] [/BACKEND Name] [/FRONTEND Name] [/OBSTYPE List] [/SWITCHMODE List] [/SOURCE Name] [/PROJID Id] [/COMPLETENESS Key] [/CALIBRATED List] [/POLARIMETRY YES|NO]

Select IMB-FITS files from the index file and build the Current Index in memory. With no options, all files are selected. The options allow to add one or more selection criteria. Using a wildcard '*' as the option value makes it uneffective. The number of selected entries is stored in the variable MFOUND, and the index arrays are available in the structure MDX%.

- /DATE: the date list, various formats are supported (see subtopic for details),
- /SCAN: the scan list. See subtopic for details
- /BACKEND: the backend name as known in the IMB-FITS file.
- /FRONTEND: the receiver name as known in the IMB-FITS file. Note that there can be several receivers per file, at least one should match.
- /OBSTYPE: the observation type(s) as known in the IMB-FITS file.
- /SWITCHMODE: the switching mode(s) in the list PSW, WSW, FSW, BSW, and UNK(nown).
- /POLARIMETRY: select files with polarimetry data (YES|NO),
- /SOURCE: the source name. Wildcards are allowed in the name.
- /PROJID: the project identifier (character string). Wildcards are allowed in the string.
- /COMPLETENESS: one of the following:
  - YES: select only complete files,
  - NO: select unreadable, empty (0 subscan) or partial (missing subscans) files,
  - READABLE: select partial or complete files,
  - UNREADABLE: select only unreadable or empty files
- /CALIBRATED: select by one or several calibration statuses, in NONE, DONE, or FAILED.

6.5.1 MFIND /SCAN

[MRTCAL\]MFIND /SCAN [List1] [List 2] ... [ListN]

Scan is selected if its number is found in one of the given lists, where one list has the form "Val1 [TO Val2]". Typical uses are:

/SCAN Val: select this unique value,
/SCAN Val1 Val2 Val3: select those 3 scalar values,
/SCAN Val1 TO Val2: select this range of values,
/SCAN Val1 Val2 Val3 TO Val4: select these values (Val1, Val2) or range of values (Val3 to Val4).

6.5.2 MFIND /DATE

```
[MRTCAL\]MFIND /DATE [List1] [List 2] ... [ListN]
```

The file is selected if its scan date, as found in the file name, matches one of the the given lists. The dates in the lists should be strings in one of the following formats:
- DD-MMM-YYYY (MMM as letters),
- YYYYMMDD (all numbers),
- Keyword "TODAY" or "YESTERDAY".

A list has the form "Date1 [TO Date2]". Typical uses are:
/DATE Date: select this unique date,
/DATE Date1 Date2 Date3: select those 3 scalar dates,
/DATE Date1 TO Date2: select this range of dates,
/DATE Date1 Date2 Date3 TO Date4: select these dates (Date1, Date2) or range of dates (Date3 to Date4).

Note that if the scan was started just before midnight, there can be subscans on the next day, i.e. calibrated data may have a different date.

6.6 INDEX

[MRTCAL\]INDEX [BUILD|UPDATE|OPEN|APPEND|WATCH [DirName]] [/FILE IndexFile] [/RECURSIVE] [/PATTERN String] [/DATE DateString] [/TIMEOUT Value]

This commands creates, updates, or opens a file index (database) of IMB-FITS files, for later use by the commands MFIND and/or CALIBRATE.
- BUILD creates or rebuilds the index file from scratch, overwriting the previous one if any,
- UPDATE creates or updates the index file if it already exists,
- OPEN opens the index file,
- APPEND appends in memory another index file,
- OUTPUT select the index file used for writing new entries,
- WATCH waits for new files to appear in the IMB-FITS directory, and index them (INDEX UDPATE) as soon as one new is found.

BUILD and UPDATE will implicitly OPEN the index file after creating it. The default is to open the index of the current working directory.

By default, the index file created/searched as 'index.mrt' in the directory where the IMB-FITS files are (DirName). However, if user has no right (e.g. write access) to put the index file in this directory, he can override this default and give any IndexFile name (with its path). In this case, it is his responsibility to ensure a correct DirName/IndexFile match. For clarity reasons, the default should be prefered when possible.

6.6.1 INDEX UPDATE

```
[MRTCAL\]INDEX UPDATE [DirName] [/PATTERN String] [DATE DateString]
```

Build or update the index of the IMB-FITS files. The index is stored in a binary file named 'index.mrt'. The command implicitly performs an INDEX OPEN right after the UPDATE to allow direct use of the index.

The index lists all the files named 'iram30m-*-*s*-imb.fits' in the directory, including unreadable and incomplete (in terms of number of subscans) ones. Selection is made from this list with the various options and features of the commands using this list.

The command can be called several times. The index file is updated if needed, in particular if the file readability has improved (e.g. scan has been completed).

By default, the command searches for the IMB-FITS files in the current working directory. This can be overriden thanks to the option /DIRECTORY. Note that the index file is always created in the same directory of the files it lists (write access is thus needed). Directory search is not recursive.

6.6.2 INDEX OPEN

```
[MRTCAL\]INDEX OPEN [DirName]
```

Load the index in memory, for subsequent use by the commands FIND and/or CALIBRATE. Default is to open the index of the current working directory.

6.6.3 INDEX OUTPUT

```
[MRTCAL\]INDEX OUTPUT [DirName] [/FILE IndexFile]
```

Append a new index file in memory (i.e. appended to the Input indeX) and make it the default for writing new entries (i.e. entries saved after a calibration). This index file is thus in-out.

By default, the new entries are saved in the same index file they come from. If you want to keep the original index file unmodified (e.g. if you have no write access), use this command to select another output file. This output index destination is reset by commands which fully reset the Input indeX, namely INDEX OPEN and INDEX BUILD.

Note that if you mix several directories in the Input indeX, it becomes a non-sense to use a unique index file as output (you will get lost when reopening it). This command is basically protected against this situation.

6.6.4 INDEX WATCH

```
[MRTCAL\]INDEX WATCH [/TIMEOUT Value]
```

Wait for new files added in the directory associated to the input index. As soon as new data is detected, the index is updated (same as INDEX UPDATE) and the command returns. There are 3 major waiting modes:

```
  - INDEX WATCH: wait indefinitely,
  - INDEX WATCH /TIMEOUT Value>0: wait until this time (in seconds) is
    elapsed, and then return normally. Note that you can mimic an infi-
    nite loop by setting a huge value (1 year is ~3e7 seconds),
  - INDEX WATCH /TIMEOUT Value<=0: does not wait. This can be useful
    when writing generic code including time-outs.

The waiting loop can also be stopped with CTRL-C. In this case the com-
mand will return with an error status.
```

6.6.5 INDEX /PATTERN

```
    [MRTCAL\]INDEX [BUILD|UPDATE|WATCH [DirName]] [/FILE IndexFile]
[/RECURSIVE] /PATTERN String

The option /PATTERN allows to indicate which files should be indexed.
Default is 'iram30m-*-*s*-imb.fits', i.e. all the IMB-FITS files in the
directory, but this can be restricted with other patterns. Note that
file names matter in Mrtcal: basically you can not index a file which do
not match the default pattern above.

For best portability, you should use a sh-compatible syntax.
```

6.6.6 INDEX /DATE

```
    [MRTCAL\]INDEX [BUILD|UPDATE|WATCH [DirName]] [/FILE IndexFile]
[/RECURSIVE] /DATE [List1] [List 2] ... [ListN]

The option /DATE allows to restrict the files to be indexed for a date
or list of dates.

A list has the form "Date1 [TO Date2]". Typical uses are:
  /DATE Date: select this unique date,
  /DATE Date1 Date2 Date3: select those 3 scalar dates,
  /DATE Date1 TO Date2: select this range of dates,
  /DATE Date1 Date2 Date3 TO Date4: select these dates (Date1,
    Date2) or range of dates (Date3 to Date4).

Accepted formats are:
    - DD-MMM-YYYY (MMM as letters),
    - YYYYMMDD (all numbers),
    - Keyword "TODAY" or "YESTERDAY".
```

6.7 MLIST

```
    [MRTCAL\]MLIST [IN|CURRENT] [/TOC List] [/COLUMNS List] [/FILE Out-
putFile] [/VARIABLE VarName]

MLIST the content of the given index. MLIST CURRENT refers to the selec-
tion made with the command MFIND. MLIST IN lists all the files known in
the memory index. Default is MLIST CURRENT.
```

The output can be redirected to an ASCII file thanks to the option /FILE. Default is print to STDOUT.

6.7.1 MLIST /TOC

MRTCAL\MLIST [IN|CURRENT] /TOC [Attr1 Attr2 ... AttrN] [/VARIABLE VarName]

MLIST /TOC displays the Table Of Contents of the current index, or of the input (MLIST IN) index file. A summary of the different values of each attribute is made, plus a final summary of all the combinations (setups). The results are accessible in the structure MTOC%. This name can be customized thanks to the option /VARIABLE. The attributes can be:

- NUMBER: observation number
- VERSION: observation version
- TELESCOPE
- ID
- SOURCE
- OBSERVED
- SCAN
- FRONTEND
- BACKEND
- OBSTYPE
- SWITCHMODE
- POLARIMETRY: file contains polarimetry data
- CALIBRATED: calibration status
- DIRECTORY

By default, OBSERVED and SCAN are used.

6.7.2 MLIST /COLUMNS

MRTCAL\MLIST [IN|OUT] /COLUMNS [Attr1 Attr2 ... AttrN]

MLIST /COLUMNS allows to customize the columns displayed. They can be repeated as many time as desire. Valid values are:

- NUMBER
- VERSION
- DIRECTORY
- FITSFILE
- MRTFILE (the index file)
- RECORD (position in MRT index file)
- WORD (position in MRT in record)
- TELESCOPE
- PROJID
- SOURCE
- OBSERVED
- UT
- LST
- AZIMUTH
- ELEVATION
- FRONTEND
- SCAN

```
    - BACKEND
    - OBSTYPE
    - SWITCHMODE
    - POLARIMETRY: file contains polarimetry data
    - COMPLETE
    - CALIBRATED
    - ITIME (last indexation time)
```

6.8 MJD

```
    [MRTCAL\]MJD MJDVar   FROM ISODateString
    [MRTCAL\]MJD MJDValue TO   ISODateVar

Convert an ISO date of the form:
  YYYY-MM-DDTHH:MM:SS.SSSSSS
to/from a Modified Julian Date.
```

6.9 PIPELINE

```
    [MRTCAL\]PIPELINE [/SHOW [NONE|DONE|FAILED]]

Calibrate iteratively all the elements in the Current indeX. Selection
of the calibration scans is automatic.

The option /SHOW will update the Current indeX at the end of the pro-
cess, and MLIST it. The optional argument indicates to populate CX only
with the elements with the corresponding calibration status.
```

6.10 READ

```
    [MRTCAL\]READ   [FileName|FileNum|FIRST|LAST|NEXT|PREV]  [/FILENAME]
[/SUBSCAN Isub]
    [MRTCAL\]READ ZERO

Load a file in the Mrtcal buffers:
  - the 4 first HDU in the IMBFITS file (namely Primary, IMBF-scan,
    IMBF-frontend and IMBF-backend), and then read
  - the 3 HDU IMBFITS-backendXXX, IMBFITS-antenna and IMBFITS-subreflec-
    tor (this latter not implemented) for the subscan number Isub (de-
    fault 1)
  - the on-track part of the DATA column of the HDU IMBFITS-backendXXX
      is read (take care that it may be large).

Without argument, the command will try to read again the current file-
name. Else the command argument can be either
  - the file name when the /FILENAME option is present,
  - the file number (and optionally its version) in the Input index, of
      the form Num (or Num.Ver),
  - the FIRST|LAST file in Current indeX,
  - the NEXT|PREVIOUS file in Current indeX, considered from previous
      call to the command.
```

The reference file (for NEXT or PREVIOUS) in Current indeX is reset by the command MFIND. READ ZERO also reset the reference file and does not read any file.

6.11 MSETUP

[MRTCAL\]MSETUP [INPUT|BOOKKEEPING|CALIBRATION|OUTPUT|PIPELINE|DEBUG]

Tune global 6 different kinds of behaviour for the READ, CALIBRATE and PIPELINE commands:
INPUT: how the command READ reads the data
BOOKKEEPING: how the input data is handled
CALIBRATION: how the calibration is performed
OUTPUT: how the spectra are output in CLASS format
PIPELINE: how the pipeline works.
DEBUG: which debugging messages are displayed.

The MSETUP command without arguments will display the current status of all these categories. MSETUP CATEGORY without further argument will display the current status of the named category.

6.11.1 MSETUP INPUT

[MRTCAL\]MSETUP INPUT Keyword Switch

[MRTCAL\]MSETUP INPUT BANDWIDTH Value

Setup the desired reading bandwidth in MHz. The nearest value that allows the division of the good spectra channels of each chunkset in equal virtual chunk sizes will be used. Exceptions are:
- if value is 0, use the native hardware chunk size (no subdivision),
- if value is larger than the native hardware chunk size, the latter is used instead (same as 0),
- if value is lower than the channel spacing, the latter is used instead (all channels calibrated individually, use with caution).

Default is native chunk size.

[MRTCAL\]MSETUP INPUT POLARIMETRY YES|NO

Choose if Mrtcal should read or not the cross-polarimetry chunks, i.e. the REAL and IMAG ones. The others (NONE and AXIS) are always read. Default is YES.

[MRTCAL\]MSETUP INPUT DATA NONE|ONTRACK|ALL

Select how Mrtcal should read the DATA column:
- NONE: no data is read. Useful to read the other IMB-FITS tables and avoiding the DATA column which may be large (save time and/or memory).
- ONTRACK: read only the on-track part of the DATA column, according to the subscan start and stop MJDs.

- ALL: real the whole DATA column, including the off-track parts.
Default is ONTRACK.

[MRTCAL\]MSETUP INPUT TOCHUNK YES|NO

Choose if Mrtcal should map the DATA column into chunksets at read time. NO means that you won't be able to do anything but READing the raw DATA into the internal buffers, and accessing from the VARIABLEs. YES is needed for calibration. This is the default.

6.11.2 MSETUP BOOKKEEPING

[MRTCAL\]MSETUP BOOKKEPPING SPACE value

Set the amount (in MB) of RAM memory used to buffer the input backend data. This allows the user to handle huge data amount on a small computer, even though it will be slower.

6.11.3 MSETUP CALIBRATION

[MRTCAL\]MSETUP CALIBRATION Keyword Switch

[MRTCAL\]MSETUP CALIBRATION SCAN NEAREST|LINEAR

Define how to select the calibration scan used calibrate science data. In NEAREST mode, the scan nearest in time will be used. In LINEAR mode, the algorithm will linearly interpolate calibration parameters between the two nearest calibration scans. Default is NEAREST.

[MRTCAL\]MSETUP CALIBRATION OFF NEAREST|LINEAR

Define how to select the OFF data matching the current ON data. In NEAREST mode, the OFF data nearest in time will be used. In LINEAR mode, the algorithm will linearly interpolate between the two nearest OFF spectra. This tuning is only meaningfull for On-The-Fly data. Default is NEAREST.

[MRTCAL\]MSETUP CALIBRATION TSYS LINEAR|NEAREST

Define how the applied system temperature will be interpolated as a function of frequency. In NEAREST mode, the algorithm will use the system temperature computed at the nearest RF frequency. In LINEAR mode, it will linearly interpolate the system temperatures surrounding the current frequency. Default is LINEAR.

[MRTCAL\]MSETUP CALIBRATION BANDWIDTH Value

Setup the desired calibration bandwidth in MHz. The nearest value that allows the division of the good spectra channels of each chunkset in equal virtual chunk sizes will be used. Exceptions are:
- if value is 0, use the native hardware chunk size (no subdivision),
- if value is larger than the native hardware chunk size, the latter is used instead (same as 0),

- if value is lower than the channel spacing, the latter is used instead (all channels calibrated individually, use with caution).

Default is 20 MHz.

[MRTCAL\]MSETUP CALIBRATION POLARIMETRY YES|NO

Choose if Mrtcal should calibrate or not the cross-polarimetry chunks, i.e. the REAL and IMAG ones. The others (NONE and AXIS) are always calibrated. Default is NO.

[MRTCAL\]MSETUP CALIBRATION FEEDBACK PIXEL|SET|ELEMENT

Tune the granurality of the calibration results (Tsys, Tcal, Trec, ...). The median value per pixel, per chunkset, or per chunk element will be printed in the PIXEL, SET, or ELEMENT mode, respectively. One value per chunk element may result in very long lists when using virtual chunk sizes (see MSETUP CALIBRATION BANDWIDTH). Default is SET.

6.11.4 MSETUP OUTPUT

[MRTCAL\]MSETUP OUTPUT Keyword Switch

[MRTCAL\]MSETUP OUTPUT SPECTRA YES|NO

Indicate whether the calibration spectra should be written in the output CLASS file. Default is YES, i.e., write the results.

[MRTCAL\]MSETUP OUTPUT USERSECTION YES|NO

Indicate whether the MRTCAL user section should be written for each CLASS observation. Default is NO, as this is very experimental to test future use CLASS, i.e., we do not warrants its stability.

[MRTCAL\]MSETUP OUTPUT CHUNK ELEMENT|SET

Write one CLASS spectra per CHUNK ELEMENT or per CHUNKSET. Default is CHUNKSET.

[MRTCAL\]MSETUP OUTPUT INTEGRATION CYCLE|SUBSCAN|SCAN

Write one CLASS spectra per phase cycle, or per subscan, or per scan. Only tracked observations are affected. On-The-Fly observations are always written per phase cycle. The default is SCAN.

6.11.5 MSETUP PIPELINE

[MRTCAL\]MSETUP PIPELINE ONERROR CONTINUE|STOP

Set the error recovery mode of the PIPELINE command. In STOP mode, any raised error will immediately stop the pipeline. In CONTINUE mode, the pipeline will ontinue to iterate with the next index item when an error is raised on the current item. Default is CONTINUE.

6.11.6 MSETUP DEBUG

```
[MRTCAL\]MSETUP DEBUG [Topic [Subtopic]] ON|OFF
```

Turn ON or OFF debugging messages for the given topics:
- IMBFITS [ALLOCATION|OTHERS]: for the IMB-FITS reader,
- INDEX [ALLOCATION|OTHERS]: for the index.mrt reading/writing,
- SYNCHRONIZATION: time synchronization of the tables,
- CALIBRATION [ALLOCATION|BOOKKEEPING|OTHERS]: for the calibration process.

Use MSET DEBUG ON|OFF to turn ON or OFF all the debug messages.

6.12 VARIABLE

```
[MRTCAL\]VARIABLE
```

Create the Sic structures mapping the Mrtcal internal buffers, namely:
- IMBF%PRIM, IMBF%SCAN, IMBF%FRONT, IMBF%BACK map the FITS HDUs for the scan,
- SUBS%BACKDATA, SUBS%ANTSLOW, SUBS%ANTFAST map the FITS HDUs for one subscan,
- IMBDATA%TIME and IMBDATA%IMBF% describe a portion of data of the DATA table,
- MHEAD%KEY, MHEAD%PRI, MHEAD%CAL map the sections of the index entry which was read.

Note that these variables are not updated when the buffers are changed. For safety they should be updated by recalling the command.

6.13 MWRITE

```
[MRTCAL\]MWRITE
```

Convert the chunks as Class observations and WRITE them in the current Class output file. The chunks are written one by one or are stitched by chunk set before writing according to the MRTCAL\SET BYCHUNK rule.