Computing
Tech. Notes & Reports
AT/25.1.1/028

# A DRAFT SPECIFICATION FOR RPFITS

Ray Norris

24 FEBRUARY 1986

## 1 INTRODUCTION

RPFITS is a dialect of FITS designed to be suitable for real-time UV data, and is the format in which the AT data will be written. This note is divided into four parts:

1. A draft specification of RPFITS
2. A justification for developing yet another 'standard' data format
3. A note about the implementation of RPFITS in the AT software.
4. A note describing various representations of REAL*4.

It should be noted that little mention is made of media in this document. This is because RPFITS (unlike FITS which is expressly designed for mag tape) is intended to be medium-independent. The working assumption is that the data will be transferred from Culgoora to Epping via optical disk, but it is not yet possible to make a firm decision on this point, and so the use of mag tape or other media cannot be ruled out.

Note also that RPFITS can easily be turned into FITS. The easiest route would be to read the RPFITS data into AIPS and then write out a FITS file. Alternatively, a special-purpose program could easily be written to perform the conversion. Note that this process is not significantly speeded up by using a block size of 2880 bytes instead of 2560 for RPFITS, and that such a change in block sze would slow the reading and writing of RPFITS, and increase the size of data storage and archive media, by about 10%, because of the padding that would have to be placed after the 2880th byte to fill up the sixth 512-byte block on the disk.

## 2 THE RPFITS SPECIFICATION

Rather than write a specification from scratch, I instead list the differences from standard FITS.

RPFITS is a dialect of FITS which differs from standard FITS in the following respects ONLY:

1. The block size is 2560 bytes, in order to maximise i/o speed on a VAX.

2. Several scans can occur on one file. Each scan starts with a FITS header. The headers are discussed further in section 2.4 below.

3. All data (other than header information) are stored as IEEE format REAL*4.

4. No extension tables are permitted. Antenna parameters may instead be entered by using keywords in the file header (as in early VLA FITS files).

There now follow brief notes on each of these changes.

## 2.1 Block Size

The block size has been changed from 2880 bytes (standard FITS) to 2560 bytes, which is 5 VAX blocks. This enables the use of fast i/o routines, such as those written by PTR for the VAX. A block length of 5 x 512-bytes rather than 1 x 512-bytes was chosen so that

1. For the header, an exact number of 80 byte card images would fit into one block.

2. The number of blocks over which a large data group would be spread is reduced, thus reducing overheads associated with reading visibilities split over two blocks.

## 2.2 Number Of Scans In A File

A scan is defined as a contiguous series of data on one source. Thus an observation consisting of half-hour runs on a source interleaved with 5 minute calibration observations would consist of a number of half-hour and 5-minute scans. It would clearly be inconvenient to have these data scattered over several files.

## 2.3 Data Type

All data are stored as IEEE REAL*4 (which is shown together with some other formats, in Section 4 below), rather than the standard FITS INTEGER format. This has the following advantages:

1. The load on the on-line AP is reduced by eliminating the need for data conversion.

2. The need for predetermined scaling factors is eliminated

3. The data will be faster to load into AIPS, as the data conversion will be a shift instead of a multiply and add (see 5.2 below).

## 2.4 Headers

A file starts with a full FITS header which contains the keyword SCANS. This keyword gives the number of scans in the file. This header (here called the file header) is followed immediately by the data from the first scan. Each subsequent scan also starts with a FITS header (here called the scan header), but the scan headers will not include the SCANS keyword. and need not duplicate any information given in the file header. If they do duplicate a keyword given in the file header. then the value associated with that keyword is overwritten by the value given in the scan header. Thus each scan 'looks' like a standard FITS file, except that unchanging header information need not be duplicated unnecessarily in each scan.

Note that RPFITS leaves open the option of repeating the header completely at the start of each scan, as opposed to giving only those parameters which have changed. Since header information generally occupies a small volume relative to data, it is recommended that the full header be reproduced on each scan, thus making the file much more robust against corruption.

## 2.5 Overwriting Of Keyword Parameters

The keywords GCOUNT and SCANS are intended to give the number of groups in a scan and the number of scans in a file respectively. In general these numbers will not be known until after the data have been taken. It is therefore suggested that these keywords be set initially to -1, and their values overwritten at the end of the scan or file. However, this latter action is not essential, and may be abandoned if the resulting head movement slows data transfer unduly. Thus, programs for reading RPFITS must be able to detect the end of a scan by recognising the start of the next header.

An alternative to setting parameters to -1 would be to enter an estimate of GCOUNT or SCANS, so that a forward search for particular data may be optimised. In this case, the value of GCOUNT or SCANS must be an <u>underestimate</u>, so that a reading program scan skip to that point and then start searching forwards for the next header.

## 2.6 Data Ordering

The data will be organised into groups, each of which corresponds to one integration on one baseline. Each measured visibility is recorded as a complex visibility plus a weight. Thus if NSTOK Stokes parameters and NFREQ channels are being measured, each group will consist of 3*NSTOK*NFREQ REAL*4 words. In addition, each group is preceded by 5 parameters:

```
U        (in light-seconds)
V        (in light-seconds)
W        (in light-seconds)
BASELINE (= (# of ant1)+256*(# of ant2))
UT       (in hhmmss.s)
```

## 2.7 Header Parameters

The header will contain most of the usual FITS parameters (DATE-OBS, OBSERVER, SOURCE, etc.). It will not, however, contain any SCALE or ZERO parameters, which are rendered redundant by the use of REAL*4 data. It will contain the following additional parameters:

SIMPLE .FALSE. (because non-standard FITS)

FORMAT RPFITS

CAL a keyword describing as a character string what calibration has been applied

TELESCOP an identifier of the array in use (e.g. ATCA, ATLBA, ATLBAN, ATCAN (N for non-standard), or DUMMY)

SCANS (in file header only) The number of scans in a file.

# 3 WHAT'S WRONG WITH THE ALTERNATIVES?

## 3.1 Standard FITS

It was originally proposed that AT data should be output and stored as standard FITS files (Greisen & Harten 1981). However, further deliberations have shown that a standard FITS format has two main disadvantages:

1. The block size of 2880 bytes prevents the use of fast i/o routines on most machines such as the Convex or VAX.

2. The requirement of only one header (and hence only one source) per file means that data which contains many short scans on calibration sources as well as on the astrophysical source has to be output as many separate small files.

## 3.2 FITS + Source Tables

To overcome the second disadvantage of FITS mentioned above, the VLBA have designed a modification of FITS which includes an additional parameter: the source number. Each uv data point is labelled with a source number, and the file is followed by a Source Table identifying each of the source numbers. This has the following advantages over RPFITS:

1. It still looks like standard FITS (within the original definition).

2. AIPS has also been modified to recognise files with this structure. Thus if we adopted this solution, no changes to AIPS would be required.

It also has the following disadvantages:

1. It still has a silly block size, causing inefficient I/O and low packing densities.

2. It is fragile to computer failures. If the Culgoora computer crashed during an observation, the source table would not have been written and the data up to that point would be lost. This problem could be circumvented when using disk by writing the source table continuously as a separate file, and overwriting when necessary, but this solution would be no good for mag tapes and might be messy for optical disks.

3. All the data depend on the completeness of both the header and the extension tables, thus requiring no media errors in these two critical areas, which will be written at the extremes of an observation (typically 12h).

## 3.3 VMS Backup Utility

The data could be written by the correlator AP in internal AIPS format, and then transferred from Culgoora to Epping using BACKUP. This has the following disadvantages, however:

1. The data will not be written by a VAX. Instead, it will be written by the correlator AP, which will be a Sky Warrior. This AP uses the IEEE floating point format used by RPFITS, as opposed to the inverted DEC floating-point format.

2. The data may not be read by a VAX. Instead, it may be read on a Convex C-1 (which uses IEEE floating point), a Cyber 205 (which uses its very own floating point format), or some other machine.

## 4 IMPLEMENTATION

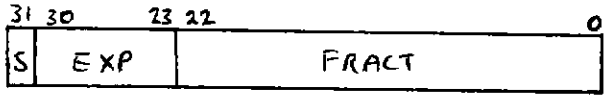
Two routines (RPFITSIN and RPFITSOUT) have been written for reading and writing the data. Corresponding AIPS routines are also being written (by MRC) to perform the same function. AIPS may then be used to translate between RPFITS and FITS. At the time of writing, RPFITSIN expects NCOUNT and NSCANS to represent the actual number: i.e. it can't cope with -1 or an estimate in these positions. This will be fixed shortly.

RPFITS is being used for the Parkes-Tidbinbilla interferometer, so that the format is getting some live testing. It has also been incorporated into ARRAY (the AT simulation program).

## 5 THE REAL*4 FORMAT

### 5.1 The IEEE Standard

An IEEE REAL*4 number has the following representation:




where:

S is the sign bit. A binary value of 0 denotes positive: a binary value of 1 denotes negative.

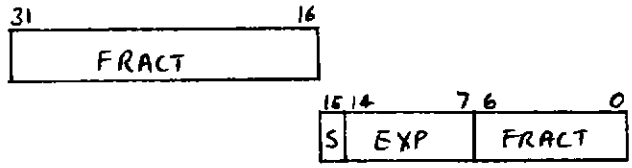
Exp is a binary-biased exponent. The decimal value of the exponent is obtained by subtracting 128 from the binary value. This exponent is then used as a power of two by which the fraction must be multiplied.

Fract is a fractional value, with an implicit 1 bit (the hidden bit) to the left of bit 22. Note that this is always positive, and is NOT expressed as a complement. Thus +n and -n are identical apart from the sign bit.

This standard is used by both the Convex C-1 and the Sky Warrior AP. Its precision is approximately 1 part in 10**23, or about seven decimal places, and its range is from 3*10**-39 to 2*10**38.

## 5.2 The DEC REAL*4 (F_floating)

This is similar to the IEEE standard, except that the two halves are swapped:



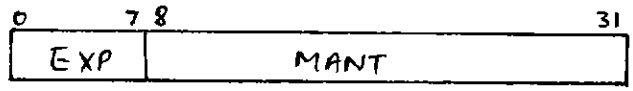

The range and precision are identical to those of the IEEE standard. Note that IEEE and F_FLOATING may easily be inter-converted:

```
REAL*4 VAX, IEEE
INTEGER*2 A(2), B(2)
EQUIVALENCE (A(1),VAX) (B(1),IEEE)

A(1)=B(2)
A(2)=B(1)
```

## 5.3 The Cyber 205 REAL*4 (HALF PRECISION)

This is non-standard:




where:

exp — is the exponent, and is a two's complement integer representing the power of two by which the mantissa must be multiplied.

mant — is the mantissa, and is a two's complement integer

The range of the Cyber 205 REAL*4 is from 8*10**-28 to 2*10**40, and it has a precision of about seven decimal places. Conversion to and from this format will be messy.