SADIST Products (Version 600)

Paul Bailey[1]

Space Science Department
Rutherford Appleton Laboratory

25 May 1994 release

---

[1] Telephone (0235) 445705; Fax (0235) 445848; e-mail (IP) pb@atsr.rl.ac.uk, (DECnet) 19590::pb

Contents

List of Figures

List of Tables

References

[1] *Using SADIST: The One-hundred and Twenty Days of ATSR Data-processing*. Paul Bailey, Space Science Department, Rutherford Appleton Laboratory, 25 May 1994 release.

[2] *SADIST Products (Version 500)*. Paul Bailey, Space Science Department, Rutherford Appleton Laboratory, 24 May 1993.

[3] *Building SADIST (Version 600) for VAX/VMS and Alpha AXP*. Paul Bailey, Space Science Department, Rutherford Appleton Laboratory, 25 May 1994 release.

[4] *Converting SADIST v500 Archive Files to v600 Format*. Paul Bailey, Space Science Department, Rutherford Appleton Laboratory, 25 May 1994 release.

[5] *ERS-1 radiomètre hyperfréquence ATSR-M X-band telemetry*. A. Le Cornec, Y. De Javel, CRPE, December 1988. ER-TN-CRP-AT-0042.

[6] *ATSR S-band telemetry words*, Issue 5.2, R. Browning, RAL, 11 February 1991. ER-IS-RAL-AT-0019.

1 Introduction

SADIST (Synthesis of ATSR Data Into Sea-surface Temperatures) is the Rutherford Appleton Laboratory's (RAL's) ATSR data-processing scheme. This document describes the format and contents of product files which are the result of SADIST processing. Note that the processes by which products are derived are discussed only very briefly; the products themselves are the subject of this document.

The contents of this document can be assumed to be valid for products of SADIST version 600 only.

1.1 Overview

Input to SADIST comes in the form of magnetic tapes containing ATSR raw data (known to ESA as RATSR data), as telemetered by the ERS-1 satellite to an appropriate ground-station (Kiruna, Sweden; Maspalomas, Canary Islands; Gatineau & Prince Albert, Canada).

The task of deriving sea-surface temperatures (SSTs) from such data may conveniently be decomposed into processing *levels*. The levels are sequential; each represents a significant step in the transformation of raw data into sea-surface temperatures. As processing may be divided into levels, SADIST products may be seen to *belong* to a particular level; products are the output from one level and, consequently, the input to the next. Thus, the products are stepping stones, and the processing levels the steps which join them.

Figure 1 is a summary of the definable levels of ATSR data-processing within SADIST, and of the characteristics of products of each level.

Level 0 products are the ATSR raw data themselves. No processing of such data has taken place; they are products of the ATSR instrument, therefore, rather than of SADIST.

Level 1.0 processing involves: decoding and unpacking of the ATSR raw data source packets (which are telemetered in a compressed form); extraction of key calibration and instrument engineering parameters.

Level 1.0 products are: decoded infra-red detector count image products; extracted ATSR/M microwave instrument raw data; extracted ATSR instrument engineering parameters.

Level 1.5 processing involves: conversion of infra-red detector counts to calibrated brightness temperatures; geometric correction of ATSR nadir- and forward-view scans; geolocation and collocation of nadir- and forward-view scans.

Level 1.5 products are: geolocated and collocated nadir/forward brightness temperature image products; sub-sampled browse image products.

Level 2.0 processing involves: land-flagging and cloud-identification of brightness temperature images; derivation of sea-surface temperatures, at both high and spatially-averaged resolutions; derivation of spatially-averaged land and cloud products.

Level 2.0 products are: land-flagging/cloud-identification results products; derived ATSR/M microwave instrument cloud products; high-resolution sea-surface temperature image products; spatially-averaged sea-surface temperature products; spatially-averaged land-surface temperature products; spatially-averaged cloud temperature/coverage products.

Level 3.0 processing (and above) involves: derivation of global maps of sea-surface, land and cloud temperatures; temporal averaging of sea-surface temperatures; mosaicking of high-resolution images; combination of products from ATSR and other instrument/in situ measurements; and so on. Such processing is *not* implemented within SADIST; it is best

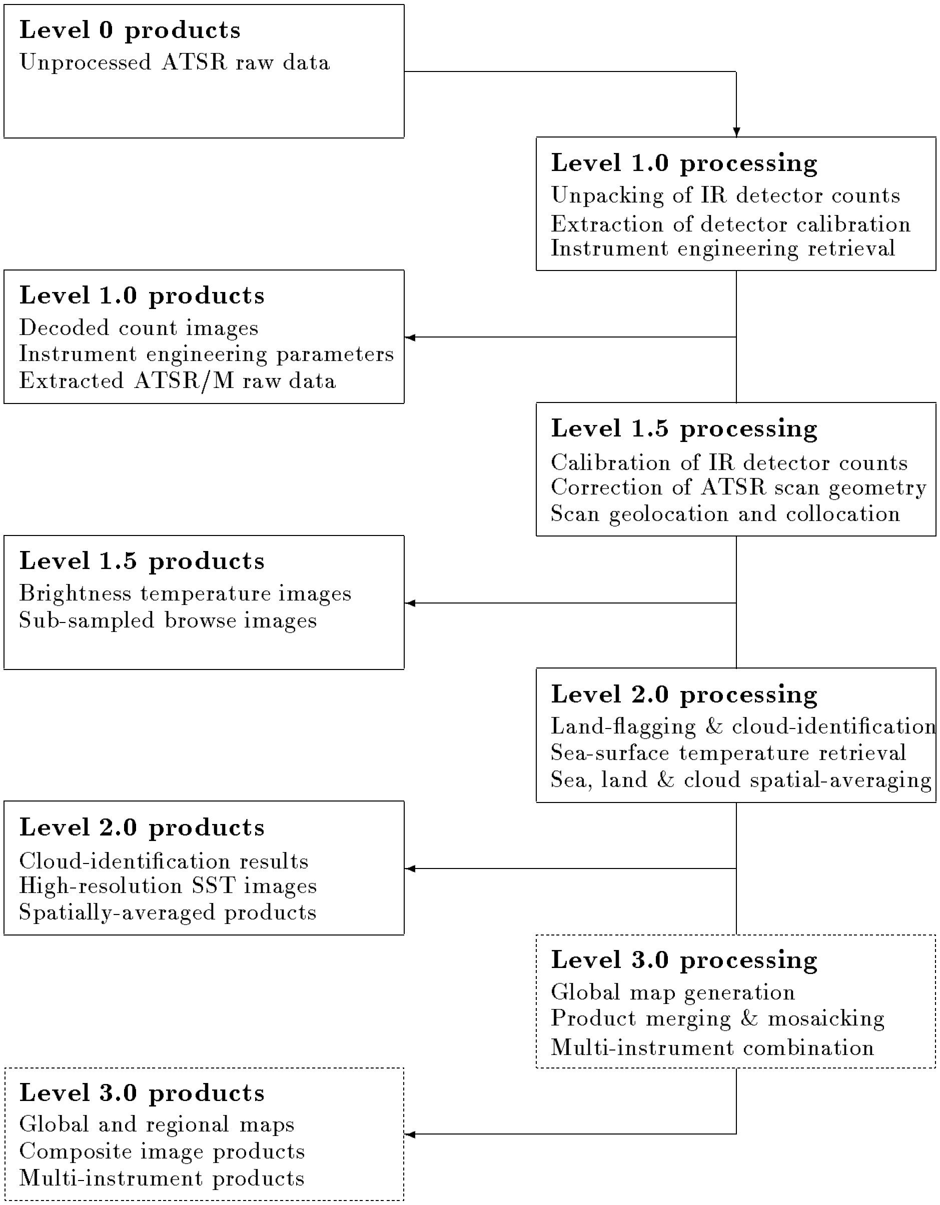

Figure 1: ATSR/SADIST product and processing levels

viewed as offline post-processing of standard ATSR products, and is subject to considerable research effort.

1.2 Product types

SADIST version 600 has twelve product types. Each type is recognised by a short name, which is provided as the extension within product file-names (see appendix A for a full description of the product file nomenclature). The short name of each product type is parenthesised within the description of the product provided below.

The product types may be divided into three logical groups:

Primary products are exactly that; they are the most important ATSR/SADIST products for scientific exploitation, and represent a direct route between ATSR raw data and derived sea-surface temperatures. Such products would be provided by any ATSR data-processing scheme. There are four primary SADIST products:

- Decoded infra-red count image products (COUNTS) are the primary result of SADIST level 1.0 processing. Contained are uncollocated, ungeolocated infra-red detector counts in both nadir and forward views, with supplementary calibration information.
- Brightness temperature image products (BT) are the primary result of SADIST level 1.5 processing. Contained are 512 * 512km images of brightness temperature/reflectance in nadir and forward views from 12.0μm, 11.0μm and $3.7/1.6\mu$m channels. The product contents are geolocated and collocated.
- Sea-surface temperature image products (SST) are the primary result of high-resolution SADIST level 2.0 processing. Contained are 512 * 512km images of derived 1km sea-surface temperature, with associated processing and confidence information.
- Spatially-averaged sea-surface temperature products (ASST) are the primary result of spatially-averaged SADIST level 2.0 processing. Contained are sets of derived half-degree sea-surface temperatures, with associated processing and confidence information.

Secondary products have reduced value on their own, but provide useful supplementary information when supplied with corresponding primary products. Such products are unique to SADIST; they should not be assumed to comply with products from any other ATSR data-processing scheme, in format or content. There are five secondary SADIST products:

- Brightness temperature browse image products (BROWSE) are a secondary result of SADIST level 1.5 processing. Contained are sub-sampled 128 * 128 pixel versions of 512 * 512km images of brightness temperature/reflectance in nadir and forward views from 12.0μm, 11.0μm and $3.7/1.6\mu$m channels. Browse images provide a convenient method of searching and cataloguing the voluminous brightness temperature image products (or indeed sea-surface temperature products thence derived) when the full image resolution is not necessary.
- Land-flagging/cloud-identification results products (CLOUD) are a secondary result of SADIST level 2.0 processing. Contained are images whose values are composite flags indicating the detailed results of the land-flagging/cloud-identification process applied to brightness temperature image products (BT). Such information is supplementary to that found within the confidence word of sea-surface temperature image products (SST) and is of value when diagnosing problems within this process, or undertaking cloud studies.

- Nadir-only sea-surface temperature image products (NSST) are a secondary result of SADIST level 2.0 processing. Such products have a format identical to primary sea-surface temperature image products (SST). Only nadir-view information is used within sea-surface temperature derivation, however. Existence of this product type is largely historical: whilst nadir/forward-view collocation remained poor in early versions of SADIST, it was necessary to derive sea-surface temperatures using nadir-view data only; now that collocation within SADIST is acceptable for dual-view SST derivation, this product type has a purpose (in conjunction with the primary sea-surface temperature image product) in supplying information about differences between SSTs derived using nadir-only and dual-view retrieval algorithms.
- Spatially-averaged land-surface temperature products (ALST) are a secondary result of spatially-averaged SADIST level 2.0 processing. They may be seen to be complementary to spatially-averaged sea-surface temperature products, whose derivation, by definition, does not include data acquired over land. Note that no attempt is made to derive a true land-surface temperature; this is beyond the scope of ATSR data-processing. What is provided within land-surface temperature products is a set of spatially-averaged brightness temperatures, from all available channels and views.
- Spatially-averaged cloud temperature/coverage products (ACLOUD) are a secondary result of spatially-averaged SADIST level 2.0 processing. Input to these products comes from those ATSR pixels within which cloud has been identified, and which therefore won't contribute to the spatially-averaged sea-surface or land-surface temperature products. These products provide derived cloud-top temperatures, in addition to measurements of the abundance of cloud within the ATSR swath.

Auxiliary products may be considered by-products of SADIST processing. Generation of each product type has been implemented within SADIST in response to a specific requirement, to which SADIST provides the best solution. Generally, generation of auxiliary products occurs in parallel with generation of primary or secondary products, and makes parasitic use of the results from a particular level of ATSR/SADIST processing. There are three auxiliary SADIST products:

- ATSR instrument engineering products (ENG) contain calibrated ATSR instrument engineering parameters extracted from ATSR raw data during SADIST level 1.0 processing. Generation of such products enables maintenance of a long-term instrument engineering status archive at RAL, and is therefore expected to continue indefinitely.
- ATSR/M microwave instrument raw data products (MWR) contain raw data from the ATSR/M microwave instrument (21 bytes per source packet) extracted from ATSR raw data during SADIST level 1.0 processing. No processing of such data is performed within SADIST; the products may therefore be regarded as level 0. Generation of this product type was originally effected to enable provision of ATSR/M raw data by RAL to CRPE (Centre de Recherches en Physique de l'Environnement Terrestre et Planétaire—the French research institute undertaking scientific exploitation of the ATSR/M instrument) during the early part of the ERS-1 mission, when distribution of ATSR raw data to RAL by ESA was significantly better than to CRPE. Even if—as expected—improved supply of ATSR raw data by ESA to CRPE obviates the need for generation of this product type, the capability to generate it will remain.
- ATSR/M microwave cloud products (MCLOUD) contain cloud information derived during SADIST level 2.0 processing which has been reprocessed to provide identification of cloud within the two elliptical ATSR/M footprints. The algorithm for this reprocessing

was designed by CRPE; since presence of the results of ATSR cloud-identification is required for generation of this product type, however, it must execute within SADIST. For this reason, generation of this auxiliary product is expected to continue for the forseeable future.

1.3 Product format/content changes within version 500

In addition to software changes whose effect is a refinement of the ATSR processing science (pixel geolocation; cloud-identification), and of the user interface to SADIST, version 500 embodies the results of a complete redesign of SADIST product formats and (to a lesser degree) product contents.

Pre-500 product formats had changed very little since the original product design work was carried out before the launch of ERS-1. In the light of the operational context within which SADIST must now operate, many original design decisions can be seen to have been poor ones.

At that time, the extent to which the ATSR science team, instrument validators and other users would be reliant on SADIST for processed data was underestimated, and was certainly not a factor within the product design process. Similarly, the variety of operating systems and programming languages which would be used was not fully considered.

The product formats chosen by RAL were heavily dependent upon VMS-specific variable-length record structures. Such structures complicated the definition of product formats, and obstructed the passage of data from RAL to many unix-based users. Even at RAL, the naively-chosen formats created difficulties when attempts were made to read data using a number of different languages.

Also, in a number of cases, product formats were such that data from the four ATSR detectors and two instrument views were held within interleaved records. This situation implicitly assumed that the entire contents of a product were desired by a user: if all of the data had to be read, then the order in which it was read was irrelevant. If, however, a user sought to use only a small section of the product contents (the prime example was a case where a user sought only one nadir-view channel from a brightness temperature product), he was forced either to read data in which he wasn't interested, or to tiptoe carefully through the product, picking up the data he needed, whilst studiously avoiding the rest. Neither solution was believed to be acceptable.

It had been clear to RAL for some time that change in product design would be beneficial. Such considerations had always given way to the recognition that significant changes would impact a large user community, and (possibly worse) would create discontinuity between formats of products from different versions of SADIST. However, in the knowledge that, twenty months after the launch of ERS-1, SADIST remained the only fully-functional ATSR data-processing system, and with the possibility looming of SADIST being made available to users external to RAL, the bullet was bit.

Therefore, with the intention of making SADIST product formats and contents more appropriate to the wide and varied use to which they are—and will be—put:

- All products now use fixed-length records; this facilitates easy transfer between VMS and unix-based machines, and allows easy file-reading from a variety of programming languages. RAL has the ability to create unix tar tapes.

- Where possible, channel information within products is not interleaved. All data from each channel and view are contiguous within product files. In combination with the use of fixed-length records, this allows easy random-access to product contents.

- The brightness temperature image product has been wholly redesigned. Rather than always containing data from all detectors and both views, the product may contain any number of such channels and views. This redefinition results from discussion at RAL about a so-called "quick-look" product, containing brightness temperatures from only one channel and view. Rather than retaining the previous brightness temperature image product (with all channels present), and defining a new "quick-look" product, the consensus at RAL was that the format of the brightness temperature product should be generalised such that the amount of data contained is wholly flexible. The contents required are requestable by the user, and are itemised within the header.

- The 25km reference grid and ten-arcminute pixel position information (supplied with each pixel in both nadir and forward views) has been removed. In its place precise geolocation (degrees/1000) is provided for each pixel. Grid interpolation is therefore no longer necessary. The pixel x/y offsets supplied with these data have been retained, since they enable precise location of true pixel centres.
- In addition, secondary header space has been added to high-resolution image products to enable storage of auxiliary information not available when the product is created, for example: information enabling attractive rendering of the data, like colour tables and contrast stretching parameters; user-defined message areas; etc. At the time of writing there is no convention for the use of the secondary header.

It should be remembered, though, that SADIST is a VMS application, and its products unavoidably acquire the characteristics of VMS files:

- Byte ordering within integers in product files follows the DEC/VMS standard (bytes ordered in increasing significance), and swapping of bytes into the user's local architecture may be necessary.

1.4 Product format/content changes within version 600

The great majority of the changes contained within SADIST version 600 are operational, rather than scientific, and are described in the document *Using SADIST v600*[1]. Those changes which affect product formats and contents are:

- The spatially-averaged cloud temperature/coverage product (ACLOUD) has been introduced.
- The ATSR instrument engineering product (ENG) has been redesigned. The principal change is the replacement of floating-point engineering parameters with appropriately-scaled integer mantissa and exponent components; the intention is to minimise the difficulty of transfer of this product from VMS systems to non-VMS systems. Note that this product is fully documented here for the first time.
- The ATSR/M microwave instrument raw data product (MWR) has been slightly redesigned, to comply with the UK.ATS.0.P100 product generated by the UK-PAF. This has necessitated using a stream format for this file; it is fully documented here for the first time.
- The brightness temperature browse image product (BROWSE) has been significantly extended. Full-resolution brightness temperatures from all available channels and views are now provided. As with the brightness temperature image product (BT) itself, the contents of the BROWSE product are now individually requestable by the user.

All other products are unchanged in version 600.

1.5 Interpretation of latitudes within SADIST v600 products

The Earth is not a sphere. When one is attempting to specify the locations of points on the Earth's surface as observed by an orbiting satellite, it is crucial to clearly define the assumptions one is making about the Earth's oblateness, and the latitudes one derives.

A **geocentric** latitude is the angle enclosed by the Equatorial plane, and the line connecting the centre of the Earth to the point on the Earth's surface seen by the satellite.

A **geodetic** latitude is the angle enclosed by the Equatorial plane, and the local normal at the point on the Earth's surface seen by the satellite. The local normal intersects the centre of the Earth only at latitudes of zero degrees and ninety degrees; these are the only points where the geodetic and geocentric latitudes are equal.

The business of pixel geolocation within SADIST data-processing is based on a *geocentric* coordinate system, which is most appropriate for the vector algebra by which pixel locations are derived from knowledge about: the position of the ERS-1 satellite; the ATSR scan geometry; the dimensions of the Earth. Pixel latitudes provided within image products are *geodetic*; they are derived from the geocentric latitudes by applying a simple function:

$$geodetic = tan^{-1}(1.0067451 \times tan(geocentric))$$

The latitudes of half-degree cells provided in spatially-averaged products, however, remain geocentric. If latitudes of the contents of image and spatially-averaged products are to be compared directly, conversions must be performed to ensure consistency. Geocentric latitudes may be derived from geodetic latitudes by applying:

$$geocentric = tan^{-1}(0.9933001 \times tan(geodetic))$$

The geocentric/geodetic inconsistency is present within products from SADIST v500 *and* SADIST v600. Rather than introducing latitude consistency within SADIST v600, at the same time introducing a systematic difference between spatially-averaged products from SADIST v500 and SADIST v600, the inconsistency has consciously been perpetuated in SADIST v600.

It is considered that the very minor changes to the scientific processing introduced in SADIST v600[1] do not justify a large-scale reprocessing of ATSR data. Maintaining the latitude inconsistency supports this view-point.

The latitudes present within SADIST v600 products are summarised in table 1.

Product type	**Latitude type**
COUNTS (header)	geodetic
BT	geodetic
SST	geodetic
ASST	geocentric
BROWSE (header)	geodetic
NSST	geodetic
CLOUD	n/a
ALST	geocentric
ACLOUD	geocentric
ENG	n/a
MWR	n/a
MCLOUD	n/a

Table 1: Latitude types within SADIST products

The differences between geocentric and geodetic latitudes (based on the formulae shown above) are characterised in table 2.

---

[1] Some small bug fixes and cloud-identification tweaks.

Geocentric	**Geodetic**
0	0
10	10.066
20	20.124
30	30.167
40	40.190
45	45.193
50	50.190
60	60.167
70	70.124
80	80.066
90	90

Table 2: Geocentric/geodetic latitude differences

2 Decoded infra-red count image product (COUNTS)

2.1 General description

The decoded infra-red count image product contains detector counts and (a limited amount of) engineering data from up to 560 consecutive ATSR scans.

The product contains:

- Primary header;
- Secondary header;
- A maximum of 2240 records containing decoded counts from the four ATSR/IR detectors in both nadir and forward views, from up to 560 nadir/forward-view scans.

The data are uncollocated and ungeolocated; that is, the nadir/forward views are contemporaneous (and therefore are separated along-track by approximately 900km), and retain the ATSR scan geometry.

Also contained are ancillary values which provide: detector counts obtained from views of the ATSR hot and cold black-body calibration targets; measured temperatures of the black bodies; coefficients for use during calibration of the detector signals (detector counts to brightness temperatures); and signal channel processor (SCP) gains and offsets for each of the detector signals.

This product has a fixed-length 2048-byte record format; it has a maximum size of 2242 records, containing 4591616 bytes.

2.2 Primary header

The primary header for this product is a single 2048-byte record, whose contents are wholly ASCII text. See table 3.

Note that the ascending node whose longitude is provided in the header is that which occurs during the course of the ATSR raw data file from which the product was derived; this may therefore occur either before or after the product itself. (Conversely, the ascending node whose time is provided in the product file-name is that which occurs *before* the product start; see appendix A.)

Byte range	**Parameter description**	**Type**	**Unit**
0 – 45	Product file-name (as described by appendix A)	Character	None
46 – 56	Latitude (geodetic) of sub-satellite point at first scan in product	Real	Degrees
57 – 67	Longitude of sub-satellite point at first scan in product	Real	Degrees
68 – 78	Longitude of ascending node crossed during course of orbit	Real	Degrees
79 – 98	Universal time of first scan in product	Character	None
99 – 2047	Unused	None	None

Table 3: Decoded infra-red count image product 2048-byte primary header

2.3 Secondary header

The secondary header for this product is a single 2048-byte record. No convention exists for the use of the secondary header.

2.4 Product format

The product consists of values obtained from an integer multiple of 80 scans; this represents the scan grouping which is present during level 1.0 processing within SADIST. A maximum file size of 560 scans has been selected since it is the first integer multiple of 80 which lies above 512, the size of all other SADIST high-resolution image products; an approximate correspondence between level 1.0 products and derived level 1.5 and 2.0 image products is maintained. Please note, though, that this correspondence remains approximate, and *direct* comparison between products is not possible, since level 1.0 products are ungeolocated, and therefore retain the ATSR scan geometry.

Data from each ATSR scan are split into four consecutive product records; there is one record per instrument detector. The contents of each record are itemised in table 4.

Byte range	**Parameter description**	**Type**	**Unit**
0 – 3	Time of scan (days since January 1st, 1950)	Integer	Days
4 – 7	Time of scan (milliseconds within current day)	Integer	msecs
8 – 1117	555 two-byte nadir-view pixel detector counts	Integer	None
1118 – 1859	371 two-byte forward-view pixel detector counts	Integer	None
1860 – 1891	16 two-byte plus black body (+bb) pixel detector counts	Integer	None
1892 – 1923	16 two-byte minus black body (-bb) pixel detector counts	Integer	None
1924 – 1951	7 four-byte measured plus black body (+bb) temperatures	Integer	K/1000
1952 – 1979	7 four-byte measured minus black body (-bb) temperatures	Integer	K/1000
1980 – 1983	Four-byte calibration bias for even pixels	Integer	1/1000000
1984 – 1987	Four-byte calibration bias for odd pixels	Integer	1/1000000
1988 – 1991	Four-byte calibration slope for even pixels	Integer	1/1000000
1992 – 1995	Four-byte calibration slope for odd pixels	Integer	1/1000000
1996 – 1997	Gain used by signal channel processor (SCP)	Integer	None
1998 – 1999	Offset used by signal channel processor (SCP)	Integer	None
2000 – 2001	IDF scan count when SCP gain/offset last changed	Integer	None
2002 – 2005	Four-byte average counts of plus black body (+bb)	Integer	None
2006 – 2009	Four-byte average counts of minus black body (-bb)	Integer	None
2010 – 2013	Four-byte cooler cold-tip temperature	Integer	K/1000
2014 – 2017	Four-byte 12.0μm detector temperature	Integer	K/1000
2018 – 2021	Four-byte 11.0μm detector temperature	Integer	K/1000
2022 – 2025	Four-byte 3.7μm detector temperature	Integer	K/1000
2026 – 2029	Four-byte 1.6μm detector temperature	Integer	K/1000
2030 – 2047	Unused	None	None

Table 4: Decoded infra-red count image product record

The four records are provided in the order: 12.0μm, 11.0μm, 3.7μm, 1.6μm.

2.5 Notes

- This product was redesigned for SADIST version 500, and is unchanged for version 600.
- Since a group of four consecutive records is derived from the same ATSR scan, it can be seen that the scan times, measured black body temperatures, and cooler/detector temperatures will be identical for each.
- Note that, though it is normal for the +bb black-body to be hot, and the -bb black-body to be cold, this should not be assumed to be the case, since both bodies may be operated in either

mode.

- 11.0μm count values are negated to show the presence of the blanking pulse which, when set, indicates operation of one of the active ERS-1 instruments (SAR, Altimeter, Scatterometer); when referencing the count value itself, use the absolute value. (Note that this situation changed in version 500; previously, the 12.0μm channel was negated to show the presence of the blanking pulse.)
- The calibration slopes and biases provided within the product have been scaled, in order to allow their representation as integers. To retrieve the actual slopes and biases, divide the product representation by 1000000.
- Exceptional values within the decoded infra-red count image product are described in table 5.

Parameter	Value	Reason
Count	-1	Channel not present within telemetered data (e.g. under normal circumstances, the 3.7μm channel during the day, and the 1.6μm channel during the night)
Count	0	No data available, due to break in continuity of downlinked data (in which case the entire scan will contain zeroes), or absence of signal in this channel
Count	1	(in 11.0μm channel) Channel not present within telemetered data, and blanking pulse present (i.e. -1 * -1)

Table 5: Decoded infra-red count image product exceptional values

2.6 Sample code

The following sample VAX C code demonstrates how the decoded infra-red count image product might be read. It assumes that the **short** type is a two-byte integer, and that the **long** type is a four-byte integer.

```
/*---------------------------------------------------------------*/

#include <file.h>
#include <unixio.h>

/*---------------------------------------------------------------*/

struct{
        char file_name[46];
        char latitude[11];
        char longitude[11];
        char node_longitude[11];
        char universal_time[20];
        char unused[1949];
     } primary_header;

struct{
        long days_since_1950;
        long msecs_in_day;
        short nadir_counts[555]; short frwrd_counts[371];
        short plus_bb_counts[16]; short minus_bb_counts[16];
```

```
         long plus_bb_temps[7]; long minus_bb_temps[7];
         long even_calibration_bias; long odd_calibration_bias;
         long even_calibration_slope; long odd_calibration_slope;
         short channel_gain;
         short channel_offset;
         short scans_since_update;
         long plus_bb_average; long minus_bb_average;
         long cooler_tip_temp;
         long detector_temps[4];
         unsigned char unused[18];
      } chan_12_rec, chan_11_rec, chan_37_rec, chan_16_rec;

/*-----------------------------------------------------------------*/

unsigned char secondary_header[2048];

short nadir_chan_1[560][555], nadir_chan_2[560][555], nadir_chan_3[560][555], nadir_chan_4[560][555],
      frwrd_chan_1[560][371], frwrd_chan_2[560][371], frwrd_chan_3[560][371], frwrd_chan_4[560][371];

/*-----------------------------------------------------------------*/

main()
{

   int counts_file, scan_loop, pixel_loop;

   counts_file = open("banks$303251725_25000_30325_x600.counts", O_RDONLY, 0, "rfm = fix", "mrs = 2048");

   read(counts_file, (void *) &primary_header, 2048);
   read(counts_file, (void *) secondary_header, 2048);

   for (scan_loop = 0; scan_loop < 560; scan_loop++)
   {

      read(counts_file, (void *) &chan_12_rec, 2048);
      read(counts_file, (void *) &chan_11_rec, 2048);
      read(counts_file, (void *) &chan_37_rec, 2048);
      read(counts_file, (void *) &chan_16_rec, 2048);

      for (pixel_loop = 0; pixel_loop < 555; pixel_loop++)
      {
         nadir_chan_1[scan_loop][pixel_loop] = chan_12_rec.nadir_counts[pixel_loop];
         nadir_chan_2[scan_loop][pixel_loop] = chan_11_rec.nadir_counts[pixel_loop];
         nadir_chan_3[scan_loop][pixel_loop] = chan_37_rec.nadir_counts[pixel_loop];
         nadir_chan_4[scan_loop][pixel_loop] = chan_16_rec.nadir_counts[pixel_loop];
      }

      for (pixel_loop = 0; pixel_loop < 371; pixel_loop++)
      {
         frwrd_chan_1[scan_loop][pixel_loop] = chan_12_rec.frwrd_counts[pixel_loop];
         frwrd_chan_2[scan_loop][pixel_loop] = chan_11_rec.frwrd_counts[pixel_loop];
         frwrd_chan_3[scan_loop][pixel_loop] = chan_37_rec.frwrd_counts[pixel_loop];
         frwrd_chan_4[scan_loop][pixel_loop] = chan_16_rec.frwrd_counts[pixel_loop];
      }
   }

   close(counts_file);

}

/*-----------------------------------------------------------------*/
```

The following sample IDL[2] code demonstrates how the decoded infra-red count image product might be read.

```
;/*----------------------------------------------------------*/

primary_head = { file_name: bytarr(46), latitude: bytarr(11), $
                 longitude: bytarr(11), node_longitude: bytarr(11), $
                 universal_time: bytarr(20), unused: bytarr(1949) }

secondary_head = bytarr(2048)

scan_times = lonarr(2, 560)

nad_chan_1 = intarr(555, 560)
nad_chan_2 = intarr(555, 560)
nad_chan_3 = intarr(555, 560)
nad_chan_4 = intarr(555, 560)

for_chan_1 = intarr(371, 560)
for_chan_2 = intarr(371, 560)
for_chan_3 = intarr(371, 560)
for_chan_4 = intarr(371, 560)

counts_record = { crec, time: lonarr(2), nadir: intarr(555), frwrd: intarr(371), $
                        plusbb: intarr(16), minusbb: intarr(16), $
                        plusbbtemp: lonarr(7), minusbbtemp: lonarr(7), $
                        calib: lonarr(4), scp: intarr(3), avrad: lonarr(2), $
                        detector_temps: lonarr(5), unused: bytarr(18) }

;/*----------------------------------------------------------*/

openr, 1, 'banks$303251725_25000_30325_x600.counts', /fixed, 2048

readu, 1, primary_head
readu, 1, secondary_head

for scan_loop = 0, 559 do begin

   readu, 1, counts_record

   scan_times(*, scan_loop) = counts_record.time
   nad_chan_1(*, scan_loop) = counts_record.nadir
   for_chan_1(*, scan_loop) = counts_record.frwrd

   readu, 1, counts_record

   nad_chan_2(*, scan_loop) = counts_record.nadir
   for_chan_2(*, scan_loop) = counts_record.frwrd

   readu, 1, counts_record

   nad_chan_3(*, scan_loop) = counts_record.nadir
   for_chan_3(*, scan_loop) = counts_record.frwrd

   readu, 1, counts_record

   nad_chan_4(*, scan_loop) = counts_record.nadir
   for_chan_4(*, scan_loop) = counts_record.frwrd

endfor
```

---

[2] IDL is a registered trademark of Research Systems, Inc.

```
end

;/*---------------------------------------------------------*/
```

3 Brightness temperature image product (BT)

3.1 General description

The brightness temperature image product consists of 512 * 512km geolocated, collocated nadir and forward brightness temperature images, at a 1km resolution, from some or all available ATSR channels.

The product contains:

- Primary header;
- Secondary header;
- 2048 (optional) records containing precise pixel locations;
- 512 (optional) records containing x/y offsets of true pixel centres from their implied image positions;
- 3072 (optional) records containing 512 * 512km images of calibrated brightness temperature/reflectance from the ATSR/IR detectors in both nadir and forward views.

The data are collocated and geolocated.

The image x-coordinate is the across-track direction, and the y-coordinate is the along-track direction. The along-track direction at any point around the orbit is defined to be parallel to the velocity vector of the sub-satellite point relative to a fixed Earth. The across-track coordinate is then measured along the Great Circle orthogonal to the sub-satellite track. From these definitions, it is clear that the along- and across-track directions vary considerably around the orbit, and do not conserve any particular alignment with parallels or meridians.

This product has a fixed-length 1024-byte record format; it has a maximum size of 5634 records, containing 5769216 bytes.

3.2 Primary header

The primary header for this product is a single 1024-byte record, whose contents are wholly ASCII text. See table 6 for a full description of this header.

Product file-name. Note that the extension to the file-name may be used to determine the product contents.

Ascending node time/state-vector/longitude. The ascending node whose time/state-vector/longitude are provided in the header is that which occurs during the course of the ATSR raw data file from which the product was derived; this may therefore occur either before or after the product itself. (Conversely, the ascending node whose time is provided in the product file-name is that which occurs *before* the product start; see appendix A.)

Image acquisition time. This Universal time has a format *dd-mmm-yyyy hh:mm:ss*. It is the acquisition time of the ATSR scan which supplied the pixel at the centre of the first line of the nadir-view image.

Universal time at ascending node. This Universal time is the same as that provided (as days since 1950) within bytes 46 to 60; it has a format *dd-mmm-yyyy hh:mm:ss*.

Latitude/longitude of sub-satellite point at image start. This geodetic latitude/longitude pair provides the position of the sub-satellite point at the along-track distance of the start of the nadir image.

Along-track distance of first line of image. This is the along-track distance from the last ascending node crossing; it is the same as that provided within the product file-name.

Source of ERS-1 state vector. The parameter within the header which indicates the source of the ERS-1 state vector used within orbit propagation has three possible values:

1. `ground predicted`: the state vector is a pre-orbit prediction; it was inserted into ATSR raw data headers at the ERS-1 ground-station to which the data were downlinked.
2. `esrin predicted`: the state vector is a pre-orbit prediction; it was distributed directly to the SADIST processing scheme by ESRIN.
3. `esrin restituted`: the state vector is a post-orbit restituted location of ERS-1; it was distributed directly to the SADIST processing scheme by ESRIN.

Known problems during transmission from ESRIN to ERS-1 ground-stations have resulted in errors within some of the state vectors inserted into ATSR data headers (the first source listed above). Geolocation/collocation errors present within products processed using these state vectors may be largely attributable to such errors. Differences between orbit propagation using ESRIN predicted and restituted state vectors are trivial at the resolution which ATSR requires; results using these state vectors can be assumed to be identical.

Solar angles The solar elevation, elevation and azimuth differences are provided at eleven equally-spaced points along the nadir and forward scans: the first point is at the first scan-pixel; the sixth point is at the middle scan-pixel; the eleventh point is at the last scan-pixel. For both views, the points at which angles are provided traverse the scan from left to right; that is, even though ATSR scans clockwise, and the forward- and nadir-view scans are acquired in different directions, the angles are consistent in both views.

The scan used in each case can be assumed to have been acquired within 6 seconds of the time at which the sub-satellite point crossed the image (forward or nadir) centre.

Note that if a brightness temperature image product is incompletely filled (data are missing at the beginning or at the end of the product), the solar angles may be missing; values of -999 degrees will be present within the primary header.

3.3 Secondary header

The secondary header for this product is a single 1024-byte record. No convention exists for the use of the secondary header.

3.4 Product format

Apart from the primary and secondary headers, the contents of the brightness temperature image product are wholly optional. The extension to the product file-name indicates which data are contained, as do bytes 753 to 766 within the primary header. The ordering of data within the product remains the same, however. If and when product contents have been requested, their order is:

1. Pixel geolocation & x/y offsets (2560 records);
2. Nadir-view 12.0μm brightness temperature image (512 records);
3. Nadir-view 11.0μm brightness temperature image (512 records);
4. Nadir-view 3.7/1.6μm brightness temperature/reflectance image (512 records);

Byte range	**Parameter description**	**Type**	**Unit**
0 – 45	Product file-name (as described by appendix A)	Character	None
46 – 60	Ascending node time (days since January 1st, 1950)	Real	Days
61 – 72	Ascending node state vector x-coordinate	Real	Km
73 – 84	Ascending node state vector y-coordinate	Real	Km
85 – 96	Ascending node state vector z-coordinate	Real	Km
97 – 106	Ascending node state vector x-velocity	Real	Km/sec
107 – 116	Ascending node state vector y-velocity	Real	Km/sec
117 – 126	Ascending node state vector z-velocity	Real	Km/sec
127 – 147	Image acquisition time	Character	None
148 – 168	Universal time at ascending node	Character	None
169 – 178	Latitude (geodetic) of sub-satellite point at time of image start	Real	Degrees
179 – 188	Longitude of sub-satellite point at time of image start	Real	Degrees
189 – 198	Longitude of the ascending node	Real	Degrees
199 – 204	Along-track distance of first line of image	Integer	Km
205 – 224	Source of ERS-1 state vector used by orbit propagation	Character	None
225 – 312	Solar elevations at 11 points along central nadir scan	11 * Real	Degrees
313 – 400	Pixel-to-sun/pixel-to-ERS1 elevation differences at 11 points along central nadir scan	11 * Real	Degrees
401 – 488	ERS1-to-sun/ERS1-to-pixel azimuth differences at 11 points along central nadir scan	11 * Real	Degrees
489 – 576	Solar elevations at 11 points along central forward scan	11 * Real	Degrees
577 – 664	Pixel-to-sun/pixel-to-ERS1 elevation differences at 11 points along central forward scan	11 * Real	Degrees
665 – 752	ERS1-to-sun/ERS1-to-pixel azimuth differences at 11 points along central forward scan	11 * Real	Degrees
753 – 754	Pixel geolocation information present (1 is true)	Integer	None
755 – 756	Nadir 12.0μm channel image present (1 is true)	Integer	None
757 – 758	Nadir 11.0μm channel image present (1 is true)	Integer	None
759 – 760	Nadir 3.7/1.6μm channel image present (1 is true)	Integer	None
761 – 762	Forward 12.0μm channel image present (1 is true)	Integer	None
763 – 764	Forward 11.0μm channel image present (1 is true)	Integer	None
765 – 766	Forward 3.7/1.6μm channel image present (1 is true)	Integer	None
767 – 774	Cooler cold-tip temperature	Real	Kelvin
775 – 782	12.0μm detector temperature	Real	Kelvin
783 – 790	11.0μm detector temperature	Real	Kelvin
791 – 798	3.7μm detector temperature	Real	Kelvin
799 – 806	1.6μm detector temperature	Real	Kelvin
807 – 1023	Unused	None	None

Table 6: Brightness temperature image 1024-byte primary header

5. Forward-view 12.0μm brightness temperature image (512 records);
6. Forward-view 11.0μm brightness temperature image (512 records);
7. Forward-view 3.7/1.6μm brightness temperature/reflectance image (512 records).

3.4.1 Pixel geolocation & x/y offsets

Pixel geolocation and x/y offset information is present within a brightness temperature image product if:

1. The file-name extension is **bt** (the default);
2. The file-name extension contains the letter **g** (indicating that the product is incomplete, but that the geolocation information is present);
3. Byte 753 within the primary header contains the character **1**.

If the pixel geolocation information is present within the product, it occupies 2560 records. The first 1024 records contain precise pixel latitude (geodetic) values; the next 1024 records contain precise pixel longitude values; the final 512 records contain composite pixel x/y offsets. Provision of x and y offsets of pixels from the centres of the image pixels to which they have been assigned increases the precision with which pixel across- and along-track coordinates may be known,

The pixel latitudes for each image across-track scan—starting at scan 0, the first scan—are split between two consecutive records, each containing precise latitudes for 256 pixels. The two records are:

- 256 four-byte integer geodetic latitudes (degrees/1000) for pixels 0 to 255;
- 256 four-byte integer geodetic latitudes (degrees/1000) for pixels 256 to 511.

The pixel longitudes for each image scan—starting at scan 0, the first scan—are split between two consecutive records, each containing precise longitudes for 256 pixels. The two records are:

- 256 four-byte integer longitudes (degrees/1000) for pixels 0 to 255;
- 256 four-byte integer longitudes (degrees/1000) for pixels 256 to 511.

Unlike the pixel geolocation latitude/longitude information, pixel x/y offsets are provided separately for nadir and forward views; differences of scan geometry within the two ATSR views makes this necessary. The first 256 records contain offsets for nadir view pixels; the following 256 records contain offsets for forward view pixels. Each pixel x/y offset record contains offsets for two consecutive image scans. Each record is:

- 512 one-byte pixel x/y offsets for all pixels within scan number n; and
- 512 one-byte pixel x/y offsets for all pixels within scan number $n + 1$.

Each one-byte pixel x/y offset contains an x-offset (that is, an offset in the across-track direction) within the least significant four bits, and a y-offset (that is, an offset in the along-track direction) within the most significant four bits. The range of possible values of each offset (0000 to 1111), represents offsets from the centre of the image pixel of -0.46875km to 0.46875km, in steps of 0.0625km (1/16km). Negative offsets represent movement towards the left-hand swath edge (for x-offsets) or against the direction of satellite motion (for y-offsets); positive offsets represent movement towards the right-hand swath-edge (for x-offsets) or with the direction of satellite motion (for y-offsets).

3.4.2 Brightness temperature/reflectance images

The nadir-view 12.0μm channel image is present if:

1. The file-name extension is **bt** (the default);
2. The file-name extension includes the characters **n1**, or **na** (indicating that the product is incomplete, but that the nadir-view 12.0μm channel has been requested);
3. Byte 755 within the primary header contains the character **1**.

The nadir-view 12.0μm channel image is contained within 512 records, each holding the brightness temperatures for one image scan, starting with the first scan. Each record is:

- 512 two-byte nadir-view 12.0μm integer brightness temperatures (Kelvin/100).

The nadir-view 11.0μm channel image is present if:

1. The file-name extension is **bt** (the default);
2. The file-name extension includes the characters **n2**, or **na** (indicating that the product is incomplete, but that the nadir-view 11.0μm channel has been requested);
3. Byte 757 within the primary header contains the character **1**.

The nadir-view 11.0μm channel image is contained within 512 records, each holding the brightness temperatures for one image scan, starting with the first scan. Each record is:

- 512 two-byte nadir-view 11.0μm integer brightness temperatures (Kelvin/100).

The nadir-view 3.7/1.6μm channel image is present if:

1. The file-name extension is **bt** (the default);
2. The file-name extension includes the characters **n3**, or **na** (indicating that the product is incomplete, but that the nadir-view 3.7/1.6μm channel has been requested);
3. Byte 759 within the primary header contains the character **1**.

The nadir-view 3.7/1.6μm channel image is contained within 512 records, each holding the brightness temperatures/reflectances for one image scan, starting with the first scan. Each record is:

- 512 two-byte nadir-view 11.0μm integer brightness temperatures/reflectances (Kelvin/100 for 3.7μm, % reflectance/100 for 1.6μm).

Forward-view channel images follow, and are formatted identically. The forward-view 12.0μm, 11.0μm, and 3.7/1.6μm channel images are present if the file-name extension is **bt**, or if the extension includes the characters **fa** (all forward-view channels present), **f1** (12.0μm channel present), **f2** (11.0μm channel present), and **f3** (3.7/1.6μm channel present). Alternatively, bytes 761, 763 and 765 may be used, as above.

3.4.3 Interpretation of the 3.7μm/1.6μm image contents

In order to minimise brightness temperature image product file sizes, 3.7μm brightness temperatures and 1.6μm reflectances are stored in the same records within product files. This makes good sense: operation of the instrument is such that data are received from only one of these detectors at any one time. It will always be possible to distinguish 3.7μm from 1.6μm data by the following method:

- 3.7μm brightness temperature values are in the range 19720 to 31882 (units of Kelvin/100; range 197K to 318K);
- 1.6μm reflectance values are in the range 0 to 10000 (where 10000 represents 100% reflectance).

There are three cases:

1. Only the 3.7μm channel is present. This is the principal mode of operation during the night-time.
2. Only the 1.6μm channel is present. This is the principal mode of operation during the day.
3. The 3.7μm and 1.6μm channels are both present, either in interleaved mode (3.7μm in odd pixels; 1.6μm in even pixels) or in threshold mode (1.6μm present if reflectance threshold is exceeded; 3.7μm otherwise). This mode will mostly be used at or near the terminator crossing.

3.5 Notes

- This product was redesigned for SADIST version 500, and is unchanged for version 600.
- Within SADIST processing, successive brightness temperature image products have a 12km overlap. The along-track distance between one image and the next is therefore 500km.
- 11.0μm brightness temperature values are negated to show the presence of the blanking pulse which, when set, indicates operation of one of the active ERS-1 instruments (SAR, Altimeter, Scatterometer); when referencing the brightness temperature itself, use the absolute value. (Note that this situation changed in version 500; previously, the 12.0μm channel was negated to show the presence of the blanking pulse.)
- Due to the undersampled nature of the ATSR forward view (371 pixels to fill 512 image pixels across-track), a significant number of pixels remain unfilled after image geolocation. (Though there are 555 nadir-view pixels per scan, a small number of nadir-image pixels also remain unfilled after geolocation.) Any such pixels are "cosmetically" filled by copying from the spatially-nearest of their eight neighbours.

  The 12.0μm and 11.0μm channel images are always filled in this way. Unfilled pixels within the 3.7μm/1.6μm image are filled using the channel of the nearest neighbouring pixel.

  12.0μm brightness temperature values are negated to indicate which pixels have been "cosmetically" filled. When the brightness temperature is negative, cosmetic filling has been employed. When referencing the brightness temperature, use the absolute value.
- Exceptional values within the brightness temperature image product are described in table 7.

Parameter	Value	Reason
Solar angles (header)	-999	Image incomplete, and solar angles at centre of image could not be derived
Brightness temperature/ reflectance	-1	Channel not present within telemetered data (since the 3.7μm and 1.6μm channels are now merged within the same images, this is expected to be a rare occurrence), or pixel could not be "cosmetically" filled due to absence of appropriate neighbouring pixel
Brightness temperature/ reflectance	0	No data available, due to break in continuity of downlinked data or absence of signal in this channel
Brightness temperature	1	(in 11.0μm channel) Channel not present within telemetered data, and blanking pulse present (i.e. -1 * -1)
Brightness temperature	1	(in 12.0μm channel) Channel not present within telemetered data, and pixel cosmetically filled (i.e. -1 * -1)

Table 7: Brightness temperature image product exceptional values

3.6 Sample code

The following sample VAX C code demonstrates how the brightness temperature image product might be read. It assumes that the **short** type is a two-byte integer, and that the **long** type is a four-byte integer.

```
/*---------------------------------------------------------------*/

#include <file.h>
#include <string.h>
#include <unixio.h>

/*---------------------------------------------------------------*/

struct{
        char file_name[46];
        char node_time[15];
        char node_x_coord[12]; char node_y_coord[12]; char node_z_coord[12];
        char node_x_velocity[10]; char node_y_velocity[10]; char node_z_velocity[10];
        char image_time[21];
        char node_time_text[21];
        char ssp_lat[10]; char ssp_lon[10]; char asc_node_lon[10];
        char along_track_distance[6];
        char state_vector_source[20];
        char nadir_elevations[11][8]; char nadir_elev_diff[11][8]; char nadir_azim_diff[11][8];
        char frwrd_elevations[11][8]; char frwrd_elev_diff[11][8]; char frwrd_azim_diff[11][8];
        char geoloc_present[2];
        char nad_1_present[2]; char nad_2_present[2]; char nad_3_present[2];
        char for_1_present[2]; char for_2_present[2]; char for_3_present[2];
        char cooler_tip_temp[8];
        char detector_temps[4][8];
        char unused[217];
      } primary_header;


/*---------------------------------------------------------------*/

unsigned char secondary_header[1024];
```

```
unsigned char offsets_record[1024], nadir_xy_offsets[512][512], frwrd_xy_offsets[512][512];

short nadir_chan_1[512][512], nadir_chan_2[512][512], nadir_chan_3[512][512];
      frwrd_chan_1[512][512], frwrd_chan_2[512][512], frwrd_chan_3[512][512];

float latitude[512][512], longitude[512][512];

long geoloc_record[256];

/*---------------------------------------------------------------*/

main()
{

   int bt_file, scan_loop, pixel_loop;

   bt_file = open("cohen$303251725_25000_30325_x600.bt", O_RDONLY, 0, "rfm = fix", "mrs = 1024");

   read(bt_file, (void *) &primary_header, 1024);
   read(bt_file, (void *) secondary_header, 1024);

   if (strncmp(geoloc_present, "1", 1) == 0)
   {

      for (scan_loop = 0; scan_loop < 512; scan_loop++)
      {
         read(bt_file, (void *) geoloc_record, 1024);

         for (pixel_loop = 0; pixel_loop < 256; pixel_loop++)
            latitude[scan_loop][pixel_loop] = (float) geoloc_record[pixel_loop] / 1000.0;

         read(bt_file, (void *) geoloc_record, 1024);

         for (pixel_loop = 0; pixel_loop < 256; pixel_loop++)
            latitude[scan_loop][pixel_loop + 256] = (float) geoloc_record[pixel_loop] / 1000.0;
      }

      for (scan_loop = 0; scan_loop < 512; scan_loop++)
      {
         read(bt_file, (void *) geoloc_record, 1024);

         for (pixel_loop = 0; pixel_loop < 256; pixel_loop++)
            longitude[scan_loop][pixel_loop] = (float) geoloc_record[pixel_loop] / 1000.0;

         read(bt_file, (void *) geoloc_record, 1024);

         for (pixel_loop = 0; pixel_loop < 256; pixel_loop++)
            longitude[scan_loop][pixel_loop + 256] = (float) geoloc_record[pixel_loop] / 1000.0;
      }

      for (scan_loop = 0; scan_loop < 512; scan_loop += 2)
      {
         read(bt_file, (void *) offsets_record, 1024);

         for (pixel_loop = 0; pixel_loop < 512; pixel_loop++)
         {
            nadir_xy_offsets[scan_loop][pixel_loop] = offsets_record[pixel_loop];
            nadir_xy_offsets[scan_loop + 1][pixel_loop] = offsets_record[pixel_loop + 512];
         }
      }

      for (scan_loop = 0; scan_loop < 512; scan_loop += 2)
```

```
      {
         read(bt_file, (void *) offsets_record, 1024);

         for (pixel_loop = 0; pixel_loop < 512; pixel_loop++)
         {
            frwrd_xy_offsets[scan_loop][pixel_loop] = offsets_record[pixel_loop];
            frwrd_xy_offsets[scan_loop + 1][pixel_loop] = offsets_record[pixel_loop + 512];
         }
      }
   }

   if (strncmp(nad_1_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 512; scan_loop++)
         read(bt_file, (void *) nadir_chan_1[scan_loop], 1024);
   }

   if (strncmp(nad_2_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 512; scan_loop++)
         read(bt_file, (void *) nadir_chan_2[scan_loop], 1024);
   }

   if (strncmp(nad_3_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 512; scan_loop++)
         read(bt_file, (void *) nadir_chan_3[scan_loop], 1024);
   }

   if (strncmp(for_1_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 512; scan_loop++)
         read(bt_file, (void *) frwrd_chan_1[scan_loop], 1024);
   }

   if (strncmp(for_2_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 512; scan_loop++)
         read(bt_file, (void *) frwrd_chan_2[scan_loop], 1024);
   }

   if (strncmp(for_3_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 512; scan_loop++)
         read(bt_file, (void *) frwrd_chan_3[scan_loop], 1024);
   }

   close(bt_file);

}

/*--------------------------------------------------------------*/
```

The following sample IDL code demonstrates how the brightness temperature image product might be read.

```
;/*--------------------------------------------------------------*/

primary_head = { file_name: bytarr(46), node_time: bytarr(15), $
                 node_x_coord: bytarr(12), node_y_coord: bytarr(12), node_z_coord: bytarr(12), $
                 node_x_velocity: bytarr(10), node_y_velocity: bytarr(10), node_z_velocity: bytarr(10), $
```

```
                 image_time: bytarr(21), node_time_text: bytarr(21), $
                 ssp_lat: bytarr(10), ssp_lon: bytarr(10), asc_node_lon: bytarr(10), $
                 along_track_distance: bytarr(6), state_vector_source: bytarr(20), $
                 nad_elevs: bytarr(8, 11), nad_elev_diff: bytarr(8, 11), nad_azim_diff: bytarr(8, 11), $
                 frw_elevs: bytarr(8, 11), frw_elev_diff: bytarr(8, 11), frw_azim_diff: bytarr(8, 11), $
                 geoloc_present: bytarr(2), $
                 nad_1_present: bytarr(2), nad_2_present: bytarr(2), nad_3_present: bytarr(2), $
                 for_1_present: bytarr(2), for_2_present: bytarr(2), for_3_present: bytarr(2), $
                 cooler_tip_temp: bytarr(8), detector_temps: bytarr(8, 4), unused: bytarr(217) }

secondary_head = bytarr(1024)

nad_chan_1 = intarr(512, 512)
nad_chan_2 = intarr(512, 512)
nad_chan_3 = intarr(512, 512)
for_chan_1 = intarr(512, 512)
for_chan_2 = intarr(512, 512)
for_chan_3 = intarr(512, 512)

grid_line = lonarr(256)
offsets_line = bytarr(1024)
temp_line = intarr(512)

lat_grid = fltarr(512, 512)
lon_grid = fltarr(512, 512)

;/*-----------------------------------------------------------------*/

openr, 1, 'cohen$303251725_25000_30325_x600.bt', /fixed, 1024

readu, 1, primary_head
readu, 1, secondary_head

if (fix(string(primary_head.geoloc_present)) eq 1) then begin

   for lat_loop = 0, 511 do begin

      readu, 1, grid_line
      lat_grid(0, lat_loop) = float(grid_line) / 1000.0

      readu, 1, grid_line
      lat_grid(256, lat_loop) = float(grid_line) / 1000.0

   endfor

   for lon_loop = 0, 511 do begin

      readu, 1, grid_line
      lon_grid(0, lon_loop) = float(grid_line) / 1000.0

      readu, 1, grid_line
      lon_grid(256, lon_loop) = float(grid_line) / 1000.0

   endfor

   for offsets_loop = 0, 511 do begin

      readu, 1, offsets_line

   endfor

endif
```

```
if (fix(string(primary_head.nad_1_present)) eq 1) then begin

   for temp_loop = 0, 511 do begin

      readu, 1, temp_line
      nad_chan_1(0, temp_loop) = temp_line

   endfor

endif

if (fix(string(primary_head.nad_2_present)) eq 1) then begin

   for temp_loop = 0, 511 do begin

      readu, 1, temp_line
      nad_chan_2(0, temp_loop) = temp_line

   endfor

endif

if (fix(string(primary_head.nad_3_present)) eq 1) then begin

   for temp_loop = 0, 511 do begin

      readu, 1, temp_line
      nad_chan_3(0, temp_loop) = temp_line

   endfor

endif

if (fix(string(primary_head.for_1_present)) eq 1) then begin

   for temp_loop = 0, 511 do begin

      readu, 1, temp_line
      for_chan_1(0, temp_loop) = temp_line

   endfor

 endif

if (fix(string(primary_head.for_2_present)) eq 1) then begin

   for temp_loop = 0, 511 do begin

      readu, 1, temp_line
      for_chan_2(0, temp_loop) = temp_line

   endfor

endif

if (fix(string(primary_head.for_3_present)) eq 1) then begin

   for temp_loop = 0, 511 do begin

      readu, 1, temp_line
      for_chan_3(0, temp_loop) = temp_line

   endfor
```

```
endif

end

;/*-----------------------------------------------------------------*/
```

4 Sea-surface temperature image product (SST)

4.1 General description

The sea-surface temperature image product consists of a 512 * 512km geolocated, collocated sea-surface temperature image, at a 1km resolution, with precise pixel latitudes/longitudes and associated processing and confidence information.

The product contains:

- Primary header;
- Secondary header;
- 2048 records containing precise pixel locations;
- 512 records containing x/y offsets of true pixel centres from their implied image positions;
- 512 records containing a 512 * 512km image of derived sea-surface temperature;
- 512 records containing processing and confidence information.

The data are collocated and geolocated.

The image x-coordinate is the across-track direction, and the y-coordinate is the along-track direction. The along-track direction at any point around the orbit is defined to be parallel to the velocity vector of the sub-satellite point relative to a fixed Earth. The across-track coordinate is then measured along the Great Circle orthogonal to the sub-satellite track. From these definitions, it is clear that the along- and across-track directions vary considerably around the orbit, and do not conserve any particular alignment with parallels or meridians.

This product has a fixed-length 1024-byte record format; it has 3586 records, containing 3672064 bytes.

4.2 Primary header

The primary header for this product is a single 1024-byte record, whose contents are wholly ASCII text. See table 8, and section 3.2, for a full description of this header.

4.3 Secondary header

The secondary header for this product is a single 1024-byte record. No convention exists for the use of the secondary header.

4.4 Product format

4.4.1 Pixel geolocation & x/y offsets

Pixel geolocation occupies 2560 records. The first 1024 records contain precise pixel latitude (geodetic) values; the next 1024 records contain precise pixel longitude values; the final 512 records contain composite pixel x/y offsets. Provision of x and y offsets of pixels from the centres of the image pixels to which they have been assigned increases the precision with which pixel across- and along-track coordinates may be known,

The pixel latitudes for each image scan—starting at scan 0, the first scan—are split between two consecutive records, each containing precise latitudes for 256 pixels. The two records are:

- 256 four-byte integer geodetic latitudes (degrees/1000) for pixels 0 to 255;

Byte range	**Parameter description**	**Type**	**Unit**
0 – 45	Product file-name (as described by appendix A)	Character	None
46 – 60	Ascending node time (days since January 1st, 1950)	Real	Days
61 – 72	Ascending node state vector x-coordinate	Real	Km
73 – 84	Ascending node state vector y-coordinate	Real	Km
85 – 96	Ascending node state vector z-coordinate	Real	Km
97 – 106	Ascending node state vector x-velocity	Real	Km/sec
107 – 116	Ascending node state vector y-velocity	Real	Km/sec
117 – 126	Ascending node state vector z-velocity	Real	Km/sec
127 – 147	Image acquisition time	Character	None
148 – 168	Universal time at ascending node	Character	None
169 – 178	Latitude (geodetic) of sub-satellite point at time of image start	Real	Degrees
179 – 188	Longitude of sub-satellite point at time of image start	Real	Degrees
189 – 198	Longitude of the ascending node	Real	Degrees
199 – 204	Along-track distance of first line of image	Integer	Km
205 – 224	Source of ERS-1 state vector used by orbit propagation	Character	None
225 – 312	Solar elevations at 11 points along central nadir scan	11 * Real	Degrees
313 – 400	Pixel-to-sun/pixel-to-ERS1 elevation differences at 11 points along central nadir scan	11 * Real	Degrees
401 – 488	ERS1-to-sun/ERS1-to-pixel azimuth differences at 11 points along central nadir scan	11 * Real	Degrees
489 – 576	Solar elevations at 11 points along central forward scan	11 * Real	Degrees
577 – 664	Pixel-to-sun/pixel-to-ERS1 elevation differences at 11 points along central forward scan	11 * Real	Degrees
665 – 752	ERS1-to-sun/ERS1-to-pixel azimuth differences at 11 points along central forward scan	11 * Real	Degrees
753 – 766	Unused	None	None
767 – 774	Cooler cold-tip temperature	Real	Kelvin
775 – 782	12.0μm detector temperature	Real	Kelvin
783 – 790	11.0μm detector temperature	Real	Kelvin
791 – 798	3.7μm detector temperature	Real	Kelvin
799 – 806	1.6μm detector temperature	Real	Kelvin
807 – 1023	Unused	None	None

Table 8: Sea-surface temperature image 1024-byte primary header

- 256 four-byte integer geodetic latitudes (degrees/1000) for pixels 256 to 511.

The pixel longitudes for each image scan—starting at scan 0, the first scan—are split between two consecutive records, each containing precise longitudes for 256 pixels. The two records are:

- 256 four-byte integer longitudes (degrees/1000) for pixels 0 to 255;
- 256 four-byte integer longitudes (degrees/1000) for pixels 256 to 511.

Unlike the pixel geolocation latitude/longitude information, pixel x/y offsets are provided separately for nadir and forward views; differences of scan geometry within the two ATSR views makes this necessary. The first 256 records contain offsets for nadir view pixels; the following 256 records contain offsets for forward view pixels. Each pixel x/y offset record contains offsets for two consecutive image scans. Each record is:

- 512 one-byte pixel x/y offsets for all pixels within scan number n; and
- 512 one-byte pixel x/y offsets for all pixels within scan number $n + 1$.

Each one-byte pixel x/y offset contains an x-offset (that is, an offset in the across-track direction) within the least significant four bits, and a y-offset (that is, an offset in the along-track direction) within the most significant four bits. The range of possible values of each offset (0000 to 1111), represents offsets from the centre of the image pixel of -0.46875km to 0.46875km, in steps of 0.0625km (1/16km). Negative offsets represent movement towards the left-hand swath edge (for x-offsets) or against the direction of satellite motion (for y-offsets); positive offsets represent movement towards the right-hand swath-edge (for x-offsets) or with the direction of satellite motion (for y-offsets).

4.4.2 Sea-surface temperature image

The following 512 records within the product contain the sea-surface temperature information. Each record contains values for the 512 pixels within a single across-track scan. Each record is:

- 512 two-byte integer derived sea-surface temperatures (Kelvin/100).

Every pixel within the sea-surface temperature image contains a value. There are three cases:

The pixel is cloud-free. In this case, a dual-view sea-surface temperature retrieval algorithm is used. The 3.7μm channel brightness temperature is used if and when it is present.

The pixel is cloudy. If only the forward view is cloudy, a nadir-only sea-surface temperature retrieval algorithm is used. If the nadir view is cloudy, or if both views are cloudy, a dual-view sea-surface temperature retrieval is used. This scheme assumes that:

- If the pixel is *truly* cloudy, any sea-surface temperature derivation will be equally invalid;
- If, however, the pixel is actually cloud-free (and has been incorrectly identified as cloud), the algorithm used should attempt to retrieve the best possible sea-surface temperature.

The pixel is over land. Since derivation of a sea-surface temperature over land is nonsense, and the retrieval algorithm is valid only over clear sea, no derivation takes place. Such pixels contain simple 11.0μm brightness temperature values.

Correct interpretation of the value contained within each pixel requires use of the processing and confidence word. The cloud and land flags (bits 0 and 2) may be used to retrieve the state of each pixel; the forward-view-used and 3.7μm-used flags (bits 8 and 11) indicate the sea-surface temperature retrieval algorithm used in each case.

Note that derivation of sea-surface temperatures requires the presence of the nadir-view 12.0μm and 11.0μm brightness temperatures. If either or both are missing, retrieval cannot take place, and sea-surface temperature values of -1 are returned.

4.4.3 Processing and confidence information

The following 512 records within the product contain the image processing and confidence information. Each record contains values for the 512 pixels within a single across-track scan. Table 9 describes the contents of the image processing and confidence word. Each record is:

- 512 two-byte integer sea-surface temperature image processing and confidence words.

Bit number	Meaning if set
0 (lsb)	Pixel is cloudy in nadir-view
1	Pixel is cloudy in forward-view
2	Pixel is over land
3	Unused
4	Unused
5	1.6μm channel present in source data
6	3.7μm channel present in source data
7	12.0μm channel present in source data
8	Forward view used in sea-surface temperature derivation
9	1.6μm reflectance histogram cloud test used dynamic threshold
10	1.6μm reflectance histogram cloud test performed
11	3.7μm channel used in sea-surface temperature derivation
12	1.6μm reflectance histogram cloud test detected sunglint
13	Unused
14	Blanking pulse occurred during this pixel
15 (msb)	Sea-surface temperature used cosmetically filled pixel in either nadir or forward view

Table 9: Contents of the sea-surface temperature image processing and confidence word

4.5 Notes

- Precise latitude/longitude for each pixel was new to version 500; the primary header was revamped. This product is unchanged for version 600.

- Successive sea-surface temperature image products have a 12km overlap. The along-track distance between one image and the next is therefore 500km.

- The 3.7μm channel is used within sea-surface temperature derivation only during periods of full darkness (solar elevation less than -5 degrees). Use of the 3.7μm channel during the daytime is precluded because of the large contribution to the signal by solar radiation scattered into the line of sight. This effect is not, of course, present in the 11.0μm and 12.0μm channels, at which wavelengths the solar spectrum is very much weaker.

- Note that the nadir and forward views are combined by selecting pixels from the nadir and forward images which have the same x and y image coordinates. This does *not* guarantee that the selected collocated pixels are the spatially-nearest match. In the worst case, a collocation error of 1.4km will exist for a forward/nadir pixel pair. Although this is less than the 3.0km collocation error which would result from a worst-case pitch/yaw pointing error of 0.13 degrees, it is still significant and should be understood when making use of the sea-surface temperature image product.

- Exceptional values within the sea-surface temperature image product are described in table 10.

Parameter	Value	Reason
Solar angles (header)	-999	Image incomplete, and solar angles at centre of image could not be derived
Sea-surface temperature	-1	12.0μm or 11.0μm brightness temperature unavailable

Table 10: Sea-surface temperature image product exceptional values

4.6 Sample code

The following sample VAX C code demonstrates how the sea-surface temperature image product might be read. It assumes that the **short** type is a two-byte integer, and that the **long** type is a four-byte integer.

```
/*---------------------------------------------------------------*/

#include <file.h>
#include <unixio.h>

/*---------------------------------------------------------------*/

struct sst_confidence{
                        int nadir_cloudy: 1;
                        int frwrd_cloudy: 1;
                        int land: 1;
                        int unused1: 1;
                        int unused2: 1;
                        int chan_4_present: 1;
                        int chan_3_present: 1;
                        int chan_1_present: 1;
                        int forward_used: 1;
                        int dynamic_thresh: 1;
                        int cc_used_chan_4: 1;
                        int sst_used_chan_3: 1;
                        int sunglint: 1;
                        int unused3: 1;
                        int blanking_pulse: 1;
                        int cosmetic: 1;
                     };

struct{
         char file_name[46];
         char node_time[15];
         char node_x_coord[12]; char node_y_coord[12]; char node_z_coord[12];
         char node_x_velocity[10]; char node_y_velocity[10]; char node_z_velocity[10];
         char image_time[21];
         char node_time_text[21];
         char ssp_lat[10]; char ssp_lon[10]; char asc_node_lon[10];
         char along_track_distance[6];
         char state_vector_source[20];
         char nadir_elevations[11][8]; char nadir_elev_diff[11][8]; char nadir_azim_diff[11][8];
         char frwrd_elevations[11][8]; char frwrd_elev_diff[11][8]; char frwrd_azim_diff[11][8];
         char unused1[14];
         char cooler_tip_temp[8];
         char detector_temps[4][8];
```

```
         char unused2[217];
      } primary_header;


/*---------------------------------------------------------------*/

unsigned char secondary_header[1024];

unsigned char offsets_record[1024], nadir_xy_offsets[512][512], frwrd_xy_offsets[512][512];

short sst_image[512][512];

float latitude[512][512], longitude[512][512];

long geoloc_record[256];

struct sst_confidence confidence[512][512];

/*---------------------------------------------------------------*/

main()
{

   int sst_file, scan_loop, pixel_loop;

   sst_file = open("wilson$303251725_25000_30325_x600.sst", O_RDONLY, 0, "rfm = fix", "mrs = 1024");

   read(counts_file, (void *) &primary_header, 1024);
   read(counts_file, (void *) secondary_header, 1024);

   for (scan_loop = 0; scan_loop < 512; scan_loop++)
   {
      read(sst_file, (void *) geoloc_record, 1024);

      for (pixel_loop = 0; pixel_loop < 256; pixel_loop++)
         latitude[scan_loop][pixel_loop] = (float) geoloc_record[pixel_loop] / 1000.0;

      read(sst_file, (void *) geoloc_record, 1024);

      for (pixel_loop = 0; pixel_loop < 256; pixel_loop++)
         latitude[scan_loop][pixel_loop + 256] = (float) geoloc_record[pixel_loop] / 1000.0;
   }

   for (scan_loop = 0; scan_loop < 512; scan_loop++)
   {
      read(sst_file, (void *) geoloc_record, 1024);

      for (pixel_loop = 0; pixel_loop < 256; pixel_loop++)
         longitude[scan_loop][pixel_loop] = (float) geoloc_record[pixel_loop] / 1000.0;

      read(sst_file, (void *) geoloc_record, 1024);

      for (pixel_loop = 0; pixel_loop < 256; pixel_loop++)
         longitude[scan_loop][pixel_loop + 256] = (float) geoloc_record[pixel_loop] / 1000.0;
   }

   for (scan_loop = 0; scan_loop < 512; scan_loop += 2)
   {
      read(sst_file, (void *) offsets_record, 1024);

      for (pixel_loop = 0; pixel_loop < 512; pixel_loop++)
      {
         nadir_xy_offsets[scan_loop][pixel_loop] = offsets_record[pixel_loop];
```

```
         nadir_xy_offsets[scan_loop + 1][pixel_loop] = offsets_record[pixel_loop + 512];
      }
   }

   for (scan_loop = 0; scan_loop < 512; scan_loop += 2)
   {
      read(sst_file, (void *) offsets_record, 1024);

      for (pixel_loop = 0; pixel_loop < 512; pixel_loop++)
      {
         frwrd_xy_offsets[scan_loop][pixel_loop] = offsets_record[pixel_loop];
         frwrd_xy_offsets[scan_loop + 1][pixel_loop] = offsets_record[pixel_loop + 512];
      }
   }

   for (scan_loop = 0; scan_loop < 512; scan_loop++)
      read(sst_file, (void *) sst_image[scan_loop], 1024);

   for (scan_loop = 0; scan_loop < 512; scan_loop++)
      read(sst_file, (void *) confidence[scan_loop], 1024);

   close(sst_file);

}

/*--------------------------------------------------------------*/
```

The following sample IDL code demonstrates how the sea-surface temperature image product might be read.

```
;/*--------------------------------------------------------------*/

primary_head = { file_name: bytarr(46), node_time: bytarr(15), $
                 node_x_coord: bytarr(12), node_y_coord: bytarr(12), node_z_coord: bytarr(12), $
                 node_x_velocity: bytarr(10), node_y_velocity: bytarr(10), node_z_velocity: bytarr(10), $
                 image_time: bytarr(21), node_time_text: bytarr(21), $
                 ssp_lat: bytarr(10), ssp_lon: bytarr(10), asc_node_lon: bytarr(10), $
                 along_track_distance: bytarr(6), state_vector_source: bytarr(20), $
                 nad_elevs: bytarr(8, 11), nad_elev_diff: bytarr(8, 11), nad_azim_diff: bytarr(8, 11), $
                 frw_elevs: bytarr(8, 11), frw_elev_diff: bytarr(8, 11), frw_azim_diff: bytarr(8, 11), $
                 unused1: bytarr(14),
                 cooler_tip_temp: bytarr(8), detector_temps: bytarr(8, 4), unused2: bytarr(217) }

secondary_head = bytarr(1024)

sst_image = intarr(512, 512)
flags = intarr(512, 512)

grid_line = lonarr(256)
offsets_line = bytarr(1024)
temp_line = intarr(512)

lat_grid = fltarr(512, 512)
lon_grid = fltarr(512, 512)

;/*--------------------------------------------------------------*/

openr, 1, 'wilson$303251725_25000_30325_x600.sst', /fixed, 1024

readu, 1, primary_head
readu, 1, secondary_head
```

```
for lat_loop = 0, 511 do begin

   readu, 1, grid_line
   lat_grid(0, lat_loop) = float(grid_line) / 1000.0

   readu, 1, grid_line
   lat_grid(256, lat_loop) = float(grid_line) / 1000.0

endfor

for lon_loop = 0, 511 do begin

   readu, 1, grid_line
   lon_grid(0, lon_loop) = float(grid_line) / 1000.0

   readu, 1, grid_line
   lon_grid(256, lon_loop) = float(grid_line) / 1000.0

endfor

for offsets_loop = 0, 511 do begin

   readu, 1, offsets_line

endfor

for temp_loop = 0, 511 do begin

   readu, 1, temp_line
   sst_image(0, temp_loop) = temp_line

endfor

for temp_loop = 0, 511 do begin

   readu, 1, temp_line
   flags(0, temp_loop) = temp_line

endfor

end

;/*-------------------------------------------------------------*/
```

5 Spatially-averaged sea-surface temperature product (ASST)

5.1 General description

The spatially-averaged sea-surface temperature product contains half-degree spatially-averaged sea-surface temperatures, with associated positional and confidence information.

The product has neither primary nor secondary headers; it contains spatially-averaged sea-surface temperatures derived from up to a complete file of ATSR raw data (which may in most circumstances be considered to be equivalent to one ERS-1 orbit).

This product has a fixed-length 32-byte record format; it has a variable length, though the largest volume of ATSR raw data which can contribute to a single spatially-averaged sea-surface temperature product (approximately one orbit) places an upper limit on the product size.

5.2 Spatially-averaged sea-surface temperature derivation

Derivation of half-degree spatially-averaged sea-surface temperatures is a two-stage process. The first stage is carried out at a ten-arcminute resolution: there are nine ten-arcminute cells within each half-degree cell. Single-pixel brightness temperatures are combined into ten-arcminute totals, whose means are derived. The derived ten-arcminute mean brightness temperatures in all available channels and views are then used within the sea-surface retrieval algorithm to obtain derived ten-arcminute sea-surface temperatures.

The second stage involves combination of the available (up to nine) ten-arcminute sea-surface temperatures to obtain a half-degree (thirty-arcminute) sea-surface temperature.

Now, depending upon the distribution of cloud within the forward and nadir views, and the availability of the 12.0μm, 11.0μm, and 3.7μm brightness temperatures (for example, the 3.7μm channel is normally not available during the day-time), the derived ten-arcminute sea-surface temperatures which contribute to a half-degree sea-surface temperature may have different characteristics. One of the problems to be addressed in the evaluation of this product is that of which combination of available ten-arcminute sea-surface temperatures gives the most accurate representation of the true temperature. Should only those ten-arcminute sea-surface temperatures which were derived using dual-view retrieval be included? Given that the spatial coverage will be improved, should we also include ten-arcminute sea-surface temperatures which used only nadir-view data?

With the aim of enabling better analysis to be performed, the spatially-averaged sea-surface temperature product contains three derived sea-surface temperatures for each half-degree cell:

1. A half-degree sea-surface temperature using those ten-arcminute cells whose sea-surface temperatures were derived using only nadir-view data;
2. A half-degree sea-surface temperature using only those ten-arcminute cells whose sea-surface temperatures were derived using nadir- *and* forward-view data;
3. A half-degree sea-surface temperature using the best available ten-arcminute cell sea-surface temperature in each case; that is, using nadir/forward-view derived sea-surface temperatures if available, and nadir-only sea-surface temperatures otherwise.

Note that the first and second sea-surface temperatures were new to SADIST version 500. The third sea-surface temperature is equivalent to that provided with pre-500 versions: if it was derived using no dual-view ten-arcminute cell sea-surface temperatures, it will be the same as the first (nadir-only) sea-surface temperature; if it was derived using only dual-view ten-arcminute cell sea-surface temperatures, it will be the same as the second (dual-only) sea-surface temperature.

Associated with each of the three sea-surface temperatures within the product is a standard deviation. This is the root-mean-square deviation of the differences of contributing ten-arcminute sea-surface temperatures from the mean of such temperatures.

Also provided is a single value containing the mean difference between derived nadir-only and nadir/forward-view derived sea-surface temperatures for the ten-arcminute cells within the half-degree cell.

5.3 Product format

The contents of each 32-byte product record are shown in table 11.

Byte range	**Parameter description**	**Type**	**Unit**
0 – 3	Time of data (days since January 1st, 1950)	Integer	Days
4 – 7	Time of data (seconds within current day)	Integer	Seconds
8 – 9	Latitude (geocentric) of half-degree cell	Integer	Cell
10 – 11	Longitude of half-degree cell	Integer	Cell
12 – 13	Mean across-track band number	Integer	None
14 – 15	Nadir-only spatially-averaged sea-surface temperature	Integer	K/100
16 – 17	Standard deviation of ten-arcminute sea-surface temperatures contributing to nadir-only spatially-averaged sea-surface temperature	Integer	K/100
18 – 19	Dual-only spatially-averaged sea-surface temperature	Integer	K/100
20 – 21	Standard deviation of ten-arcminute sea-surface temperatures contributing to dual-only spatially-averaged sea-surface temperature	Integer	K/100
22 – 23	Mixed nadir/dual spatially-averaged sea-surface temperature	Integer	K/100
24 – 25	Standard deviation of ten-arcminute sea-surface temperatures contributing to mixed nadir/dual spatially-averaged sea-surface temperature	Integer	K/100
26 – 27	Mean view-difference of nadir/dual ten-arcminute sea-surface temperatures	Integer	K/100
28 – 31	Confidence word associated with spatially-averaged sea-surface temperature derivation	None	None

Table 11: Spatially-averaged sea-surface temperature product record

Time of data. The integer value containing the number of days since January 1st, 1950, does not include the current, incomplete day. Note that the time used within each record is the time of the first ATSR nadir-view scan within the orbit to contribute to the spatially-averaged sea-surface temperature derivation. The variable nature of cloud-cover makes it impossible to predict the position of this scan relative to the centre of the half-degree cell. Under any circumstances, this time cannot be more than approximately six seconds from the time at which the centre of the cell is scanned by the nadir view.

Latitude. The latitude is provided as a cell number. The edges of half-degree cells are sections of parallels and meridians. The latitude cells are numbered from the South Pole to the North Pole, in the range 0 to 359. Latitude cell number 0 extends from 90 degrees South to 89.5 degrees South; latitude cell number 359 extends from 89.5 degrees North to 90 degrees North. The latitude of the cell centre may be derived by:

$$latitude = ((lat\_cell\_num - 180.0)/2.0) + 0.25.$$

In common with the other spatially-averaged products (ALST, ACLOUD), the latitudes which are implied by the product cell numbers are *geocentric*. The latitudes provided with image products, and within product headers, are *geodetic*. This is discussed in section 1.5.

Longitude. The longitude is provided as a cell number. The edges of half-degree cells are sections of parallels and meridians. The longitude cells are numbered from 180 degrees West to 180 degrees East, in the range 0 to 719. Longitude cell number 0 extends from 180 degrees West to 179.5 degrees West; longitude cell number 719 extends from 179.5 degrees East to 180 degrees East. The longitude of the cell centre may be derived by:

$$longitude = ((lon\_cell\_num - 360.0)/2.0) + 0.25.$$

Mean across-track band number. The five across-track bands (numbered 0 to 4) are symmetric about the ground-track. Each band is 50km wide (except the fifth, which is 62km wide, and extends to the edge of the swath).

Nadir-only SST and SD. These are the mean and root-mean-square deviation of ten-arcminute cell sea-surface temperatures derived using only nadir-view data. Note that the standard deviation is -1 if fewer than 3 ten-arcminute cells contributed.

Dual-only SST and SD. These are the mean and root-mean-square deviation of ten-arcminute cell sea-surface temperatures derived using nadir- and forward-view data. Note that the sea-surface temperature is -1 if dual-view retrieval was not possible for any ten-arcminute cell, and that the standard deviation is -1 if fewer than 3 ten-arcminute cells contributed.

Mixed SST and SD. These are the mean and root-mean-square deviation of ten-arcminute cell sea-surface temperatures derived using the best available combination of nadir- and forward-view data. Note that the standard deviation is -1 if fewer than 3 ten-arcminute cells contributed. If no ten-arcminute cells used dual-view retrieval, these values will be identical to the nadir-only sea-surface temperature and standard deviation.

Mean view-difference. This is the mean difference between the ten-arcminute cell sea-surface temperatures derived using dual-view and nadir-only retrieval algorithms. This value is -1 if no dual-view retrieval was possible.

Confidence word. Table 12 describes the contents of the spatially-averaged sea-surface temperature product confidence word.

Bits 0 to 8 are set if, within all of the contributing ten-arcminute cells, at least 90% of the contributing pixels have the appropriate characteristic.

Bits 9 to 12 contain the number of ten-arcminute cell sea-surface temperatures contributing to the nadir-only half-degree sea-surface temperature, and (the number is necessarily the same) to the mixed nadir-only/dual-view half-degree sea-surface temperature.

Bits 13 to 16 contain the number of ten-arcminute cell sea-surface temperatures contributing to the dual-only half-degree sea-surface temperature. This number may be zero.

Bit number(s)	Meaning if set
0 (lsb)	12.0μm channel present in source data
1	11.0μm channel present in source data
2	3.7μm channel present in source data
3	1.6μm channel present in source data
4	Cloud-identification used 1.6μm histogram reflection cloud test
5	1.6μm histogram reflectance cloud test used a dynamic threshold
6	Sunglint detected by 1.6μm reflectance reflectance cloud test
7	3.7μm channel used in sea-surface temperature retrieval
8	Sea-surface temperature derivation used day-time data (night-time if zero)
9 – 12	Number of ten-arcminute cell SSTs derived using nadir-only retrieval (1 – 9)
13 – 16	Number of ten-arcminute cell SSTs derived using dual-only retrieval (0 – 9)
17 – 31 (msb)	Unused

Table 12: Spatially-averaged sea-surface temperature product confidence word

5.4 Notes

- The latitude and longitude cell numbers were split for version 500; previously they were combined into a single "half-degree cell number". The nadir-only and dual-only sea-surface temperatures were new to version 500. The mixed sea-surface temperature is equivalent to the (only) sea-surface temperature value provided with pre-500 versions.
- This product is unchanged for version 600.
- No explicit or implicit ordering of product values may be assumed. The Earth-location must be derived using the latitude and longitude information provided with each sea-surface temperature record.
- Exceptional values within the spatially-averaged sea-surface temperature product are described in table 13.

Parameter	Value	Reason
Dual-only sea-surface temperature	-1	No ten-arcminute cells used a dual-view sea-surface temperature retrieval
Standard deviations	-1	Fewer than 3 ten-arcminute cells contributed to the half-degree spatially-averaged sea-surface temperature
Mean view-difference	-1	No ten-arcminute cells used a dual-view sea-surface temperature retrieval

Table 13: Spatially-averaged sea-surface temperature image product exceptional values

5.5 Sample code

The following sample VAX C code demonstrates how the spatially-averaged sea-surface temperature product might be read. It assumes that the **short** type is a two-byte integer, and that the **long** type is a four-byte integer.

```
/*------------------------------------------------------------*/

#include <file.h>
#include <unixio.h>

/*------------------------------------------------------------*/

struct asst_confidence{
                         int chan_1_present: 1;
                         int chan_2_present: 1;
                         int chan_3_present: 1;
                         int chan_4_present: 1;
                         int histogram_done: 1;
                         int dynamic_thresh_used: 1;
                         int sunglint_found: 1;
                         int chan_3_used: 1;
                         int day_night: 1;
                         int nadir_only_x_arcmin_cells: 4;
                         int dual_only_x_arcmin_cells: 4;
                         int unused: 15;
                      };

struct half_degree_cell{
                          long day_number;
                          long secs_in_day;
                          short latitude;
                          short longitude;
                          short across_track_band;
                          short nadir_only_sst;
                          short nadir_only_deviation;
                          short dual_only_sst;
                          short dual_only_deviation;
                          short mixed_sst;
                          short mixed_deviation;
                          short mean_view_diff;
                          struct asst_confidence confidence;
                       };

/*------------------------------------------------------------*/

struct half_degree_cell cell_array[20000];

/*------------------------------------------------------------*/

main()
{

   int asst_file, cell_loop = 0;

   asst_file = open("charltonj$303251725_25000_30325_x600.asst", O_RDONLY, 0, "rfm = fix", "mrs = 32");

   while (read(asst_file, (void *) &cell_array[cell_loop++], 32) == 32);

   close(asst_file);

}

/*------------------------------------------------------------*/
```

The following sample IDL code demonstrates how the spatially-averaged sea-surface temperature product might be read.

```
;/*-----------------------------------------------------*/

on_ioerror, end_of_file

nadir_map = intarr(720, 360)
dual_map = intarr(720, 360)
mixed_map = intarr(720, 360)

asst_rec = { file_record, time: lonarr(2), position: intarr(3), $
             nadir_asst: intarr(2), dual_asst: intarr(2), mixed_asst: intarr(2), $
             view_diff: intarr(1), confidence: lonarr(1) }

asst_stuff = replicate(asst_rec, 20000)

;/*-----------------------------------------------------*/

openr, 1, 'charltonj$303251725_25000_30325_x600.asst', /fixed, 32

asst_loop = 0

read_from_sadist_file:

readu, 1, asst_rec

nadir_map(asst_rec.position(1), asst_rec.position(0)) = asst_rec.nadir_asst(0)
dual_map(asst_rec.position(1), asst_rec.position(0)) = asst_rec.dual_asst(0)
mixed_map(asst_rec.position(1), asst_rec.position(0)) = asst_rec.mixed_asst(0)

asst_stuff(asst_loop) = asst_rec

asst_loop = asst_loop + 1

goto, read_from_sadist_file

end_of_file:

close, 1

end

;/*-----------------------------------------------------*/
```

6 Brightness temperature browse image product (BROWSE)

6.1 General description

The brightness temperature browse image product contains sub-sampled copies of some or all of the 512 * 512km geolocated, collocated nadir and forward brightness temperature images within a brightness temperature image product, at a 4 km resolution, with approximate positional information.

The product contains:

- Primary header;
- Secondary header;
- 768 (optional) records containing sub-sampled 512 * 512km images of calibrated brightness temperature/reflectance from the ATSR/IR detectors in both nadir and forward views.

The data are collocated and geolocated, although only a small amount of pixel positional information is provided within the product.

The image x-coordinate is the across-track direction, and the y-coordinate is the along-track direction. The along-track direction at any point around the orbit is defined to be parallel to the velocity vector of the sub-satellite point relative to a fixed Earth. The across-track coordinate is then measured along the Great Circle orthogonal to the sub-satellite track. From these definitions, it is clear that the along- and across-track directions vary considerably around the orbit, and do not conserve any particular alignment with parallels or meridians.

This product has a fixed-length 256-byte record format; it has a maximum size of 770 records, containing 197120 bytes.

6.2 Primary header

The primary header for this product is a single 256-byte record, whose contents are wholly ASCII text. See table 14 for a full description of this header.

The latitude and longitude values provided in the primary header can be seen to define the positions of the image corner-points. Note that the extension to the file-name may be used to determine the product contents.

6.3 Secondary header

The secondary header for this product is a single 256-byte record. No convention exists for the use of the secondary header.

6.4 Product format

Apart from the primary and secondary headers, the contents of the brightness temperature browse image product are wholly optional. The extension to the product file-name indicates which data are contained, as do bytes 106 to 117 within the primary header. The ordering of data within the product remains the same, however. If and when product contents have been requested, their order is:

1. Sub-sampled nadir-view 12.0μm brightness temperature image (128 records);
2. Sub-sampled nadir-view 11.0μm brightness temperature image (128 records);
3. Sub-sampled nadir-view 3.7/1.6μm brightness temperature/reflectance image (128 records);
4. Sub-sampled forward-view 12.0μm brightness temperature image (128 records);

Byte range	**Parameter description**	**Type**	**Unit**
0 – 45	Product file-name (as described by appendix A)	Character	None
46 – 60	Ascending node time (days since January 1st, 1950)	Real	Days
61 – 72	Ascending node state vector x-coordinate	Real	Km
73 – 84	Ascending node state vector y-coordinate	Real	Km
85 – 96	Ascending node state vector z-coordinate	Real	Km
97 – 106	Ascending node state vector x-velocity	Real	Km/sec
107 – 116	Ascending node state vector y-velocity	Real	Km/sec
117 – 126	Ascending node state vector z-velocity	Real	Km/sec
127 – 133	Latitude (geodetic) of left-most pixel in first image scan	Integer	Degrees/1000
134 – 140	Latitude (geodetic) of right-most pixel in first image scan	Integer	Degrees/1000
141 – 147	Latitude (geodetic) of left-most pixel in last image scan	Integer	Degrees/1000
148 – 154	Latitude (geodetic) of right-most pixel in last image scan	Integer	Degrees/1000
155 – 162	Longitude of left-most pixel in first image scan	Integer	Degrees/1000
163 – 170	Longitude of right-most pixel in first image scan	Integer	Degrees/1000
171 – 178	Longitude of left-most pixel in last image scan	Integer	Degrees/1000
179 – 186	Longitude of right-most pixel in last image scan	Integer	Degrees/1000
187 – 188	Nadir 12.0μm channel image present (1 is true)	Integer	None
189 – 190	Nadir 11.0μm channel image present (1 is true)	Integer	None
191 – 192	Nadir 3.7/1.6μm channel image present (1 is true)	Integer	None
193 – 194	Forward 12.0μm channel image present (1 is true)	Integer	None
195 – 196	Forward 11.0μm channel image present (1 is true)	Integer	None
197 – 198	Forward 3.7/1.6μm channel image present (1 is true)	Integer	None
199 – 206	Cooler cold-tip temperature	Real	Kelvin
207 – 214	12.0μm detector temperature	Real	Kelvin
215 – 222	11.0μm detector temperature	Real	Kelvin
223 – 230	3.7μm detector temperature	Real	Kelvin
231 – 238	1.6μm detector temperature	Real	Kelvin
239 – 255	Unused	None	None

Table 14: Brightness temperature browse image product 256-byte primary header

5. Sub-sampled forward-view 11.0μm brightness temperature image (128 records);
6. Sub-sampled forward-view 3.7/1.6μm brightness temperature/reflectance image (128 records).

6.4.1 Sub-sampled brightness temperature/reflectance images

The sub-sampled nadir-view 12.0μm channel image is present if:

1. The file-name extension is **browse** (the default);
2. The file-name extension includes the characters **n1**, or **na** (indicating that the product is incomplete, but that the sub-sampled nadir-view 12.0μm channel has been requested);
3. Byte 187 within the primary header contains the character **1**.

The sub-sampled nadir-view 12.0μm channel image is contained within 128 records, each holding the brightness temperatures for one image scan, starting with the first scan. Each record is:

- 128 two-byte nadir-view 12.0μm integer brightness temperatures (Kelvin/100).

The sub-sampled nadir-view 11.0μm channel image is present if:

1. The file-name extension is **browse** (the default);
2. The file-name extension includes the characters **n2**, or **na** (indicating that the product is incomplete, but that the sub-sampled nadir-view 11.0μm channel has been requested);
3. Byte 189 within the primary header contains the character **1**.

The sub-sampled nadir-view 11.0μm channel image is contained within 128 records, each holding the brightness temperatures for one image scan, starting with the first scan. Each record is:

- 128 two-byte nadir-view 11.0μm integer brightness temperatures (Kelvin/100).

The nadir-view 3.7/1.6μm channel image is present if:

1. The file-name extension is **browse** (the default);
2. The file-name extension includes the characters **n3**, or **na** (indicating that the product is incomplete, but that the sub-sampled nadir-view 3.7/1.6μm channel has been requested);
3. Byte 191 within the primary header contains the character **1**.

The sub-sampled nadir-view 3.7/1.6μm channel image is contained within 128 records, each holding the brightness temperatures/reflectances for one image scan, starting with the first scan. Each record is:

- 128 two-byte nadir-view 11.0μm integer brightness temperatures/reflectances (Kelvin/100 for 3.7μm, % reflectance/100 for 1.6μm).

Sub-sampled forward-view channel images follow, and are formatted identically. The sub-sampled forward-view 12.0μm, 11.0μm, and 3.7/1.6μm channel images are present if the file-name extension is **browse**, or if the extension includes the characters **fa** (all forward-view channels present), **f1** (12.0μm channel present), **f2** (11.0μm channel present), and **f3** (3.7/1.6μm channel present). Alternatively, bytes 193, 195 and 197 may be used, as above.

6.4.2 Interpretation of the 3.7μm/1.6μm image contents

See section 3.4.3, which describes the interpretation of the 3.7μm/1.6μm brightness temperature image contents.

6.5 Notes

- This product has been redesigned for SADIST version 600. The product may now contain any and all available brightness temperature/reflectance images, as determined by the user request. No image histogram-equalisation is now performed; full-precision brightness temperatures are retained.
- Within SADIST processing, successive brightness temperature browse image products have a 12km overlap. The along-track distance between one image and the next is therefore 500km.
- Unlike the brightness temperature image product, the brightness temperature browse image product does not use negation of 12.0μm and 11.0μm channel brightness temperatures. Blanking pulse and cosmetic filling information is not available within this product.
- Exceptional values within the brightness temperature browse image product are described in table 15.

Parameter	Value	Reason
Brightness temperature/ reflectance	-1	Channel not present within telemetered data (since the 3.7μm and 1.6μm channels are now merged within the same images, this is expected to be a rare occurrence), or pixel could not be "cosmetically" filled due to absence of appropriate neighbouring pixel
Brightness temperature/ reflectance	0	No data available, due to break in continuity of downlinked data or absence of signal in this channel
Brightness temperature	1	(in 11.0μm channel) Channel not present within telemetered data, and blanking pulse present (i.e. -1 * -1)
Brightness temperature	1	(in 12.0μm channel) Channel not present within telemetered data, and pixel cosmetically filled (i.e. -1 * -1)

Table 15: Brightness temperature browse image product exceptional values

6.6 Sample code

The following sample VAX C code demonstrates how the brightness temperature browse image product might be read. It assumes that the **short** type is a two-byte integer, and that the **long** type is a four-byte integer.

```
/*---------------------------------------------------------------*/

#include <file.h>
#include <string.h>
#include <unixio.h>

/*---------------------------------------------------------------*/

struct{
```

```
         char file_name[46];
         char node_time[15];
         char node_x_coord[12]; char node_y_coord[12]; char node_z_coord[12];
         char node_x_velocity[10]; char node_y_velocity[10]; char node_z_velocity[10];
         char lower_left_lat[7]; char lower_right_lat[7];
         char upper_left_lat[7]; char upper_right_lat[7];
         char lower_left_lon[8]; char lower_right_lon[8];
         char upper_left_lon[8]; char upper_right_lon[8];
         char nad_1_present[2]; char nad_2_present[2]; char nad_3_present[2];
         char for_1_present[2]; char for_2_present[2]; char for_3_present[2];
         char cooler_tip_temp[8];
         char detector_temps[4][8];
         char unused[17];
      } primary_header;


/*-----------------------------------------------------------------*/

unsigned char secondary_header[256];

short nadir_chan_1[128][128], nadir_chan_2[128][128], nadir_chan_3[128][128];
      frwrd_chan_1[128][128], frwrd_chan_2[128][128], frwrd_chan_3[128][128];

/*-----------------------------------------------------------------*/

main()
{

   int browse_file, scan_loop, pixel_loop;

   browse_file = open("moore$303251725_25000_30325_x600.browse", O_RDONLY, 0, "rfm = fix", "mrs = 256");

   read(browse_file, (void *) &primary_header, 256);
   read(browse_file, (void *) secondary_header, 256);

   if (strncmp(nad_1_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 128; scan_loop++)
         read(browse_file, (void *) nadir_chan_1[scan_loop], 256);
   }

   if (strncmp(nad_2_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 128; scan_loop++)
         read(browse_file, (void *) nadir_chan_2[scan_loop], 256);
   }

   if (strncmp(nad_3_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 128; scan_loop++)
         read(browse_file, (void *) nadir_chan_3[scan_loop], 256);
   }

   if (strncmp(for_1_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 128; scan_loop++)
         read(browse_file, (void *) frwrd_chan_1[scan_loop], 256);
   }

   if (strncmp(for_2_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 128; scan_loop++)
         read(browse_file, (void *) frwrd_chan_2[scan_loop], 256);
```

```
   }

   if (strncmp(for_3_present, "1", 1) == 0)
   {
      for (scan_loop = 0; scan_loop < 128; scan_loop++)
         read(browse_file, (void *) frwrd_chan_3[scan_loop], 256);
   }

   close(browse_file);

}

/*--------------------------------------------------------------*/
```

The following sample IDL code demonstrates how the brightness temperature browse image product might be read.

```
;/*--------------------------------------------------------------*/

primary_head = { file_name: bytarr(46), node_time: bytarr(15), $
                 node_x_coord: bytarr(12), node_y_coord: bytarr(12), node_z_coord: bytarr(12), $
                 node_x_velocity: bytarr(10), node_y_velocity: bytarr(10), node_z_velocity: bytarr(10), $
                 lat1: bytarr(7), lat2: bytarr(7), lat3: bytarr(7), lat4: bytarr(7), $
                 lon1: bytarr(8), lon2: bytarr(8), lon3: bytarr(8), lon4: bytarr(8), $
                 nad_1_present: bytarr(2), nad_2_present: bytarr(2), $
                 nad_3_present: bytarr(2), for_1_present: bytarr(2), $
                 for_2_present: bytarr(2), for_3_present: bytarr(2), $
                 cooler_tip_temp: bytarr(8), detector_temps: bytarr(8, 4), unused: bytarr(17) }

secondary_head = bytarr(256)

nad_chan_1 = intarr(128, 128)
nad_chan_2 = intarr(128, 128)
nad_chan_3 = intarr(128, 128)
for_chan_1 = intarr(128, 128)
for_chan_2 = intarr(128, 128)
for_chan_3 = intarr(128, 128)

temp_line = intarr(128)

;/*--------------------------------------------------------------*/

openr, 1, 'moore$303251725_25000_30325_x600.browse', /fixed, 256

readu, 1, primary_head
readu, 1, secondary_head

if (fix(string(primary_head.nad_1_present)) eq 1) then begin

   for temp_loop = 0, 127 do begin

      readu, 1, temp_line
      nad_chan_1(0, temp_loop) = temp_line

   endfor

endif

if (fix(string(primary_head.nad_2_present)) eq 1) then begin

   for temp_loop = 0, 127 do begin
```

```
      readu, 1, temp_line
      nad_chan_2(0, temp_loop) = temp_line

   endfor

endif

if (fix(string(primary_head.nad_3_present)) eq 1) then begin

   for temp_loop = 0, 127 do begin

      readu, 1, temp_line
      nad_chan_3(0, temp_loop) = temp_line

   endfor

endif

if (fix(string(primary_head.for_1_present)) eq 1) then begin

   for temp_loop = 0, 127 do begin

      readu, 1, temp_line
      for_chan_1(0, temp_loop) = temp_line

   endfor

 endif

if (fix(string(primary_head.for_2_present)) eq 1) then begin

   for temp_loop = 0, 127 do begin

      readu, 1, temp_line
      for_chan_2(0, temp_loop) = temp_line

   endfor

endif

if (fix(string(primary_head.for_3_present)) eq 1) then begin

   for temp_loop = 0, 127 do begin

      readu, 1, temp_line
      for_chan_3(0, temp_loop) = temp_line

   endfor

endif

end

;/*-----------------------------------------------------------------*/
```

7 Land-flagging/cloud-identification results product (CLOUD)

7.1 General description

The land-flagging/cloud-identification results product contains detailed results from the land-flagging and cloud-identification applied within SADIST level 2.0 processing to a brightness temperature image product. Some of this information is provided within the confidence word supplied with sea-surface temperature image products (during the derivation of which the land-flagging/cloud-identification process is performed).

The product has neither primary nor secondary headers; it contains 1024 records, which supply composite flags summarising the results of land-flagging/cloud-identification of both nadir and forward views.

This product has a fixed-length 1024-byte record format; it has 1024 records, containing 1048576 bytes.

7.2 Product format

The first 512 records contain the results of land-flagging/cloud-identification of the nadir-view images. Each record contains 512 values, each of which is a composite result flag for a pixel within the brightness temperature image product. Table 16 describes the contents of each composite result flag. Each record is:

- 512 two-byte nadir-view composite result flags, as described in table 16.

The following 512 records contain the results of land-flagging/cloud-identification of the forward-view images. Each record contains 512 values, each of which is a composite result flag for a pixel within the brightness temperature image product. Table 16 describes the contents of each composite result flag. Each record is:

- 512 two-byte forward-view composite result flags, as described in table 16.

7.3 Notes

- Introduced with SADIST version 500 was a fixed-length record format. Previously, record lengths were all 1024 bytes, but the record format remained "variable". This product is unchanged for version 600.
- The SADIST cloud-identification algorithm suite remains in a state of flux. The contents of this product may therefore be subject to significant change between SADIST versions.
- Bit 1 is unused. Previously, this bit stored results of cloud-identification tests which attempted to identify pixels not wholly cloud-filled. Such tests are no longer used within SADIST.

7.4 Sample code

The following sample VAX C code demonstrates how the land-flagging/cloud-identification results product might be read.

```
/*---------------------------------------------------------------*/

#include <file.h>
```

Bit number	Meaning if set
0 (lsb)	Pixel is cloudy (summary of cloud-identification tests)
1	Unused
2	Pixel is over land
3	Cloud identified by 3.7μm/11.0μm view-difference test
4	Cloud identified by 11.0μm/12.0μm view-difference test
5	1.6μm reflectance histogram test performed
6	1.6μm reflectance histogram test used dynamic threshold
7	1.6μm reflectance histogram test detected sunglint contamination
8	Cloud identified by 1.6μm reflectance histogram test
9	Cloud identified by 11.0μm spatial coherence test
10	Cloud identified by 11.0μm/12.0μm thin-cirrus test
11	Cloud identified by 12.0μm gross cloud test
12	Cloud identified by 11.0μm/3.7μm fog/low-stratus test
13	Cloud identified by 3.7μm/12.0μm medium/high-level test
14	Cloud identified by 11.0μm/12.0μm infra-red histogram test
15 (msb)	Unused

Table 16: Land-flagging/cloud-identification composite result word

```
#include <unixio.h>

/*---------------------------------------------------------------*/

struct cloud_flag{
                    int cloudy: 1;
                    int unused: 1;
                    int land: 1;
                    int view_diff_37_11_test: 1;
                    int view_diff_11_12_test: 1;
                    int histogram_done: 1;
                    int dynamic_thresh_used: 1;
                    int sunglint_found: 1;
                    int histogram_test: 1;
                    int coherence_test: 1;
                    int thin_cirrus_test: 1;
                    int gross_cloud_test: 1;
                    int fog_low_stratus_test: 1;
                    int med_high_level_test: 1;
                    int diff_11_12_hist_test: 1;
                    int unused: 1;
                 };


/*---------------------------------------------------------------*/

struct cloud_flag nadir_cloud_image[512][512], frwrd_cloud_image[512][512];

/*---------------------------------------------------------------*/

main()
{

   int cloud_file, scan_loop;
```

```
   cloud_file = open("peters$303251725_25000_30325_x600.cloud", O_RDONLY, 0, "rfm = fix", "mrs = 1024");

   for (scan_loop = 0; scan_loop < 512; scan_loop++)
      read(cloud_file, (void *) nadir_cloud_image[scan_loop], 1024);

   for (scan_loop = 0; scan_loop < 512; scan_loop++)
      read(cloud__file, (void *) frwrd_cloud_image[scan_loop], 1024);

   close(cloud_file);

}

/*--------------------------------------------------------------*/
```

The following sample IDL code demonstrates how the land-flagging/cloud-identification results product might be read.

```
;/*------------------------------------------------------------------*/

nadir_cloud_flags = intarr(512, 512)
frwrd_cloud_flags = intarr(512, 512)

line_of_flags = intarr(512)

;/*------------------------------------------------------------------*/

openr, 1, 'peters$303251725_25000_30325_x600.cloud', /fixed, 1024

for line_loop = 0, 511 do begin

   readu, 1, line_of_flags
   nadir_cloud_flags(0, line_loop) = line_of_flags

endfor

for line_loop = 0, 511 do begin

   readu, 1, line_of_flags
   frwrd_cloud_flags(0, line_loop) = line_of_flags

endfor

close, 1

end

;/*------------------------------------------------------------------*/
```

8 Nadir-only sea-surface temperature image product (NSST)

8.1 General description

The nadir-only sea-surface temperature image product is identical in format and content to the primary sea-surface temperature image product (SST). This product contains sea-surface temperatures derived using a nadir-only retrieval algorithm, however, even when forward-view information is available and cloud-free.

See section 4 for a description of product headers, format, and contents.

8.2 Notes

- Note that where derivation of the primary sea-surface temperature image product uses a nadir-only retrieval algorithm (where forward-view channels are cloud-contaminated, but nadir-view channels are not), the contents of the nadir-only sea-surface temperature image product would be identical.

9 Spatially-averaged land-surface temperature product (ALST)

9.1 General description

The spatially-averaged land-surface temperature product contains half-degree spatially-averaged mean brightness temperatures/reflectances, with associated positional and confidence information.

The product has neither primary nor secondary headers; it contains spatially-averaged land-surface temperatures derived from up to a complete file of ATSR raw data (which may in most circumstances be considered to be equivalent to one ERS-1 orbit).

This product has a fixed-length 34-byte record format; it has a variable length, though the largest volume of ATSR raw data which can contribute to a single spatially-averaged land-surface temperature product (approximately one orbit) places an upper limit on the product size.

9.2 Product format

The contents of each 34-byte product record are shown in table 17.

Byte range	Parameter description	Type	Unit
0 – 3	Time of data (days since January 1st, 1950)	Integer	Days
4 – 7	Time of data (seconds within current day)	Integer	Seconds
8 – 9	Latitude (geocentric) of half-degree cell	Integer	Cell
10 – 11	Longitude of half-degree cell	Integer	Cell
12 – 13	Mean across-track band number	Integer	None
14 – 15	Spatially-averaged nadir 12.0μm brightness temperature	Integer	K/100
16 – 17	Spatially-averaged nadir 11.0μm brightness temperature	Integer	K/100
18 – 19	Spatially-averaged nadir 3.7μm brightness temperature	Integer	K/100
20 – 21	Spatially-averaged nadir 1.6μm reflectance	Integer	%ref/100
22 – 23	Spatially-averaged forward 12.0μm brightness temperature	Integer	K/100
24 – 25	Spatially-averaged forward 11.0μm brightness temperature	Integer	K/100
26 – 27	Spatially-averaged forward 3.7μm brightness temperature	Integer	K/100
28 – 29	Spatially-averaged forward 1.6μm reflectance	Integer	%ref/100
30 – 33	Confidence word summarising spatially-averaged land-surface temperature derivation	None	None

Table 17: Spatially-averaged land-surface temperature product record

Time of data. The integer value containing the number of days since January 1st, 1950, does not include the current, incomplete day. Note that the time used within each record is the time of the first ATSR nadir-view scan within the orbit to contribute to the spatially-averaged land-surface temperature derivation.[3] The variable nature of cloud-cover makes it impossible to predict the position of this scan relative to the centre of the half-degree cell. Under any

---

[3] Though note that, if and when no nadir data contributed to the temperature derivation, the time of the first contributing forward-view scan is used.

circumstances, this time cannot be more than approximately six seconds from the time at which the centre of the cell is scanned by the nadir view.

Latitude. The latitude is provided as a cell number. The edges of half-degree cells are sections of parallels and meridians. The latitude cells are numbered from the South Pole to the North Pole, in the range 0 to 359. Latitude cell number 0 extends from 90 degrees South to 89.5 degrees South; latitude cell number 359 extends from 89.5 degrees North to 90 degrees North. The latitude of the cell centre may be derived by:

$$latitude = ((lat\_cell\_num - 180.0)/2.0) + 0.25.$$

In common with the other spatially-averaged products (ASST, ACLOUD), the latitudes which are implied by the product cell numbers are *geocentric.* The latitudes provided with image products, and within product headers, are *geodetic.* This is discussed in section 1.5.

Longitude. The longitude is provided as a cell number. The edges of half-degree cells are sections of parallels and meridians. The longitude cells are numbered from 180 degrees West to 180 degrees East, in the range 0 to 719. Longitude cell number 0 extends from 180 degrees West to 179.5 degrees West; longitude cell number 719 extends from 179.5 degrees East to 180 degrees East. The longitude of the cell centre may be derived by:

$$longitude = ((lon\_cell\_num - 360.0)/2.0) + 0.25.$$

Mean across-track band number. The five across-track bands (numbered 0 to 4) are symmetric about the ground-track. Each band is 50km wide (except the fifth, which is 62km wide, and extends to the edge of the swath).

Confidence word. Table 18 describes the contents of the spatially-averaged land-surface temperature product confidence word. The three-bit values which indicate how many pixels have contributed to each spatially-averaged land-surface temperature value are coded. The codes are described in table 19.

Spatially-averaged brightness temperatures. Derivation of land-temperatures using multi-channel and multi-view retrieval schemes is beyond the scope of SADIST processing of ATSR data. The spatially-averaged brightness temperatures and reflectances supplied within the product are simple means of the available cloud-free data over land.

9.3 Notes

- This product type was new to SADIST version 500. It is unchanged for version 600.
- No explicit or implicit ordering of product values may be assumed. The Earth-location must be derived using the latitude and longitude information provided with each land-surface temperature record.

9.4 Sample code

The following sample VAX C code demonstrates how the spatially-averaged land-surface temperature product might be read. It assumes that the **short** type is a two-byte integer, and that the **long** type is a four-byte integer.

Bit number(s)	**Meaning if set**
0 (lsb)	Nadir-view land-surface temperature derivation used day-time data (night-time data if zero)
1	Forward-view land-surface temperature derivation used day-time data (night-time data if zero)
2 – 4	Number of pixels contributing to nadir-view 12.0μm land-surface temperature (0 – 7)
5 – 7	Number of pixels contributing to nadir-view 11.0μm land-surface temperature (0 – 7)
8 – 10	Number of pixels contributing to nadir-view 3.7μm land-surface temperature (0 – 7)
11 – 13	Number of pixels contributing to nadir-view 1.6μm land-surface reflectance (0 – 7)
14 – 16	Number of pixels contributing to forward-view 12.0μm land-surface temperature (0 – 7)
17 – 19	Number of pixels contributing to forward-view 11.0μm land-surface temperature (0 – 7)
20 – 22	Number of pixels contributing to forward-view 3.7μm land-surface temperature (0 – 7)
23 – 25	Number of pixels contributing to forward-view 1.6μm land-surface reflectance (0 – 7)
26 – 31 (msb)	Unused

Table 18: Spatially-averaged land-surface temperature product confidence word

Code	**Number of pixels**
0	Fewer than 400
1	400 to 799
2	800 to 1199
3	1200 to 1599
4	1600 to 1999
5	2000 to 2399
6	2400 to 2799
7	More than 2799

Table 19: Land-surface temperature three-bit number-of-pixels encoding

```
/*---------------------------------------------------------------*/

#include <file.h>
#include <unixio.h>

/*---------------------------------------------------------------*/

struct alst_confidence{
                         int day_night: 1;
                         int nadir_chan_1_pixels: 3;
                         int nadir_chan_2_pixels: 3;
                         int nadir_chan_3_pixels: 3;
                         int nadir_chan_4_pixels: 3;
                         int forward_chan_1_pixels: 3;
                         int forward_chan_2_pixels: 3;
                         int forward_chan_3_pixels: 3;
                         int forward_chan_4_pixels: 3;
                         int unused: 7;
                      };

struct half_degree_cell{
                          long day_number;
                          long secs_in_day;
                          short latitude;
                          short longitude;
                          short across_track_band;
                          short nadir_chan_1_mean;
                          short nadir_chan_2_mean;
                          short nadir_chan_3_mean;
                          short nadir_chan_4_mean;
                          short frwrd_chan_1_mean;
                          short frwrd_chan_2_mean;
                          short frwrd_chan_3_mean;
                          short frwrd_chan_4_mean;
                          struct alst_confidence confidence;
                       };

/*---------------------------------------------------------------*/

struct half_degree_cell cell_array[20000];

/*---------------------------------------------------------------*/

main()
{

   int alst_file, cell_loop = 0;

   alst_file = open("charltonb$303251725_25000_30325_x600.alst", O_RDONLY, 0, "rfm = fix", "mrs = 34");

   while (read(alst_file, (void *) &cell_array[cell_loop++], 34) == 34);

   close(alst_file);

}

/*---------------------------------------------------------------*/
```

The following sample IDL code demonstrates how the spatially-averaged land-surface temperature product might be read.

```
;/*-----------------------------------------------------*/

on_ioerror, end_of_file

twelve_map = intarr(720, 360)
eleven_map = intarr(720, 360)

file_rec = { file_record, time: lonarr(2), position: intarr(3), $
             nadir_alst: intarr(4), frwrd_alst: intarr(4), confidence: lonarr(1) }

alst = { alst_record, time: lonarr(2), position: intarr(3), $
         nadir_alst: intarr(4), frwrd_alst: intarr(4), $
         daynight: bytarr(1), nadir_pixels: bytarr(4), frwrd_pixels: bytarr(4) }

alst_stuff = replicate(alst, 20000)

;/*-----------------------------------------------------*/

openr, 1, 'charltonb$303251725_25000_30325_x600.alst', /fixed, 34

alst_loop = 0

read_from_sadist_file:

readu, 1, file_rec

twelve_map(file_rec.position(1), file_rec.position(0)) = file_rec.nadir_alst(0)
eleven_map(file_rec.position(1), file_rec.position(0)) = file_rec.nadir_alst(1)

alst_stuff(alst_loop).time = file_rec.time
alst_stuff(alst_loop).position = file_rec.position
alst_stuff(alst_loop).nadir_alst = file_rec.nadir_alst
alst_stuff(alst_loop).frwrd_alst = file_rec.frwrd_alst
alst_stuff(alst_loop).daynight = file_rec.confidence(0) and 1

alst_stuff(alst_loop).nadir_pixels(0) = (file_rec.confidence(0) and 14) / 2
alst_stuff(alst_loop).nadir_pixels(1) = (file_rec.confidence(0) and 112) / 16
alst_stuff(alst_loop).nadir_pixels(2) = (file_rec.confidence(0) and 896) / 128
alst_stuff(alst_loop).nadir_pixels(3) = (file_rec.confidence(0) and 7168) / 1024

alst_stuff(alst_loop).frwrd_pixels(0) = (file_rec.confidence(0) and 57344) / 8192
alst_stuff(alst_loop).frwrd_pixels(1) = (file_rec.confidence(0) and 458752) / 65536
alst_stuff(alst_loop).frwrd_pixels(2) = (file_rec.confidence(0) and 3670016) / 524288
alst_stuff(alst_loop).frwrd_pixels(3) = (file_rec.confidence(0) and 29360128) / 4194304

alst_loop = alst_loop + 1

goto, read_from_sadist_file

end_of_file:

close, 1

end

;/*-----------------------------------------------------*/
```

10 Spatially-averaged cloud temperature/coverage product (ACLOUD)

10.1 General description

The spatially-averaged cloud temperature/coverage product contains information concerning the temperature and abundance of cloud within the ATSR nadir and forward views, at a half-degree spatial resolution, with associated positional and confidence information.

The product has neither primary nor secondary headers; it contains spatially-averaged cloud temperature/coverage information derived from up to a complete file of ATSR raw data (which may in most circumstances be considered to be equivalent to one ERS-1 orbit).

This product has a fixed-length 244-byte record format; it has a variable length, though the largest volume of ATSR raw data which can contribute to a single spatially-averaged cloud temperature/coverage product (approximately one orbit) places an upper limit on the product size.

10.2 Cloud temperature/coverage derivation

It should be emphasised that *cloudy pixels*, in the context of the cloud temperature/coverage product, are those pixels which have been identified as cloudy by SADIST's cloud-identification tests. No assurance can be made that all true cloud will be detected by such tests; nor that all detected pixels will be truly cloudy.

Cloud temperature and coverage results are derived independently for the nadir and forward views; no attempt is made to combine the two views within a cloud temperature retrieval algorithm. Similarly, no attempt is made to combine information from ATSR's multiple detectors. All cloud temperature information within this product is based on brightness temperatures from the 11.0μm channel.

For reasons of processing efficiency, derivation of cloud temperatures (i.e. calculation of brightness temperature means) proceeds via the construction of a histogram of 11.0μm brightness temperatures within each half-degree cell. Each histogram records the distribution of brightness temperatures of cloudy pixels from 190.0 Kelvin to 290.0 Kelvin, at a 0.1 Kelvin resolution, within 1000 boxes; the first box records the number of cloudy pixels with 11.0μm brightness temperatures between 190.0 Kelvin and 190.1 Kelvin; the last box records the number of cloudy pixels with 11.0μm brightness temperatures between 289.9 Kelvin and 290.0 Kelvin. It can be seen that construction of the 0.1 Kelvin histogram involves a loss of the precision with which brightness temperatures are known.

No cloud temperature derivation is performed if fewer than 20 cloudy pixels have been identified, in either view. If sufficient pixels *have* been identified, two cloud temperatures are calculated, via the 0.1 Kelvin histogram.

The first is a simple mean of the 11.0μm brightness temperatures of all cloudy pixels. The second is an attempt to derive a cloud-top temperature; the 11.0μm brightness temperatures of only the coldest 25% of the cloudy pixels contribute to this derivation.

The product also contains the numbers of cloudy and cloud-free pixels which have been located. From these numbers, the percentage cloud-cover is derived, and is provided.

10.3 Product format

The contents of each 244-byte product record are shown in table 20.

Time of data. The integer value containing the number of days since January 1st, 1950, does not include the current, incomplete day. Note that the time used within each record is the time of the first ATSR nadir-view scan within the orbit to contribute to the spatially-averaged

Byte range	**Parameter description**	**Type**	**Unit**
0 – 3	Time of data (days since January 1st, 1950)	Integer	Days
4 – 7	Time of data (seconds within current day)	Integer	Seconds
8 – 9	Latitude (geocentric) of half-degree cell	Integer	Cell
10 – 11	Longitude of half-degree cell	Integer	Cell
12 – 13	Mean across-track band number	Integer	None
14 – 15	(Nadir) Number of cloudy pixels	Integer	None
16 – 17	(Nadir) Number of cloud-free pixels	Integer	None
18 – 19	(Nadir) Spatially-averaged 11.0μm brightness temperature of all cloudy pixels	Integer	K/100
20 – 21	(Nadir) Standard deviation of 11.0μm brightness temperatures of all cloudy pixels	Integer	K/100
22 – 23	(Nadir) Lowest 11.0μm brightness temperature of cloudy pixels	Integer	K/100
24 – 25	(Nadir) Cloud-top temperature: spatially-averaged 11.0μm brightness temperature of coldest 25% of cloudy pixels	Integer	K/100
26 – 27	(Nadir) Percentage cloud-cover	Integer	%/100
28 – 127	(Nadir) One Kelvin histogram of 11.0μm brightness temperatures of all cloudy pixels	Integer	None
128 – 129	(Frwrd) Number of cloudy pixels	Integer	None
130 – 131	(Frwrd) Number of cloud-free pixels	Integer	None
132 – 133	(Frwrd) Spatially-averaged 11.0μm brightness temperature of all cloudy pixels	Integer	K/100
134 – 135	(Frwrd) Standard deviation of 11.0μm brightness temperatures of all cloudy pixels	Integer	K/100
136 – 137	(Frwrd) Lowest 11.0μm brightness temperature of cloudy pixels	Integer	K/100
138 – 139	(Frwrd) Cloud-top temperature: spatially-averaged 11.0μm brightness temperature of coldest 25% of cloudy pixels	Integer	K/100
140 – 141	(Frwrd) Percentage cloud-cover	Integer	%/100
142 – 241	(Frwrd) One Kelvin histogram of 11.0μm brightness temperatures of all cloudy pixels	Integer	None
242 – 243	Confidence word associated with spatially-averaged cloud temperature/coverage derivation	None	None

Table 20: Spatially-averaged cloud temperature/coverage product record

sea-surface temperature derivation. The variable nature of cloud-cover makes it impossible to predict the position of this scan relative to the centre of the half-degree cell. Under any circumstances, this time cannot be more than approximately six seconds from the time at which the centre of the cell is scanned by the nadir view.

Latitude. The latitude is provided as a cell number. The edges of half-degree cells are sections of parallels and meridians. The latitude cells are numbered from the South Pole to the North Pole, in the range 0 to 359. Latitude cell number 0 extends from 90 degrees South to 89.5 degrees South; latitude cell number 359 extends from 89.5 degrees North to 90 degrees North. The latitude of the cell centre may be derived by:

$$latitude = ((lat\_cell\_num - 180.0)/2.0) + 0.25.$$

In common with the other spatially-averaged products (ASST, ALST), the latitudes which are implied by the product cell numbers are *geocentric*. The latitudes provided with image products, and within product headers, are *geodetic*. This is discussed in section 1.5.

Longitude. The longitude is provided as a cell number. The edges of half-degree cells are sections of parallels and meridians. The longitude cells are numbered from 180 degrees West to 180 degrees East, in the range 0 to 719. Longitude cell number 0 extends from 180 degrees West to 179.5 degrees West; longitude cell number 719 extends from 179.5 degrees East to 180 degrees East. The longitude of the cell centre may be derived by:

$$longitude = ((lon\_cell\_num - 360.0)/2.0) + 0.25.$$

Mean across-track band number. The five across-track bands (numbered 0 to 4) are symmetric about the ground-track. Each band is 50km wide (except the fifth, which is 62km wide, and extends to the edge of the swath).

Number of cloudy pixels. This is the number of pixels within the half-degree cell which were identified as cloudy by the SADIST cloud-identification tests. Note that, since the surface area covered by a half-degree cell decreases towards the poles, the maximum value this parameter may have will also decrease (to a limit of zero at the poles themselves). Note also that, since no cloud temperature/coverage information may be derived using fewer than 20 cloudy pixels, 20 is the practical minimum for this parameter.

Number of cloud-free pixels. This is the number of pixels within the half-degree cell which were identified as cloud-free by the SADIST cloud-identification tests. Note that, since the surface area covered by a half-degree cell decreases towards the poles, the maximum value this parameter may have will also decrease (to a limit of zero at the poles themselves). This parameter may be zero.

Spatially-averaged brightness temperature of all cloudy pixels. This mean is calculated from the 0.1 Kelvin histogram described in Section 10.2. Therefore, although it is supplied at a precision of 0.01 Kelvin, its accuracy (with respect to the *true* mean), can only be assumed to be ±0.05 Kelvin. The derivation is:

$$mean\_of\_cloudy = 19000 + 10 \times \frac{\sum_{i=0}^{999}((i+0.5) \times histogram[i])}{\sum_{i=0}^{999} histogram[i]}$$

Standard deviation of brightness temperatures of cloudy pixels. Again, this is calculated from the 0.1 Kelvin histogram, described above, so its accuracy (with respect to the *true* mean), can only be assumed to be ±0.05 Kelvin. The derivation is:

$$sd\_of\_cloudy = 10 \times \sqrt{\frac{\sum_{i=0}^{999}(((i+0.5)-((mean\_of\_cloudy-19000)/10))\times histogram[i])^2}{(\sum_{i=0}^{999} histogram[i])-1}}$$

Lowest 11.0μm brightness temperature of cloudy pixels. This is simply the lowest 11.0μm brightness temperature of the pixels identified as cloudy within the half-degree cell.

Cloud-top temperature. This value, which is known as the cloud-top temperature (though which makes no great claims to represent exactly that physical parameter), is calculated from the 0.1 Kelvin histogram, described above. Only the coldest 25% of the histogram is used; that is, only those histogram boxes up to and including that which contains the twenty-fifth percentile are included within the mean derivation. Its accuracy (with respect to the *true* mean of the coldest 25% of the cloudy pixels) can only be assumed to be ±0.05 Kelvin. The derivation is:

$$ctt = 19000 + 10 \times \frac{\sum_{i=0}^{n}((i+0.5)\times histogram[i])}{\sum_{i=0}^{n} histogram[i]}$$

where n is the histogram box containing the twenty-fifth percentile.

Percentage cloud-cover. This is the percentage of all pixels within the half-degree cell which were identified as cloudy by the SADIST cloud-identification tests. The derivation is:

$$percentage\_cover = 10000 \times \frac{number\_of\_cloudy\_pixels}{number\_of\_cloudy\_pixels+number\_of\_cloud\_free\_pixels}$$

since the percentage is provided in units of %/100. Values of percentage cloud-cover for half-degree cells within across-track band 4 of the ATSR swath should be used with caution; such cells may not be wholly covered by the swath.

One Kelvin histogram of 11.0μm brightness temperatures of all cloudy pixels. This histogram, which describes the temperature distribution of the pixels identified as cloudy, is derived from the 0.1 Kelvin histogram constructed during calculation of the spatially-averaged cloud temperatures shown above. The derivation involves two steps:

1. The 0.1 Kelvin histogram is reduced to a 1.0 Kelvin histogram, by merging histogram boxes in groups of ten.
2. The 1.0 histogram is normalised, so that each histogram value may be represented by a single byte. The normalisation is:

   $$normalised\_value[i] = 255 \times \frac{merged\_histogram[i]}{merged\_histogram[max]}$$

   where i is each of the boxes of the (merged) 1.0 Kelvin histogram (from 0 to 99), and max is the histogram box containing the largest number of cloudy pixels.

The resulting 1.0 Kelvin histogram is a one-hundred element array of one-byte normalised values, scaled so that a value of 255 represents the most populous histogram box. The first histogram box represents brightness temperatures between 190.0 Kelvin and 191.0 Kelvin; the last histogram box represents brightness temperatures between 289.0 Kelvin and 290.0 Kelvin. Note that the values within the normalised histogram remain in correct proportion, so the values of the number of cloudy and cloud-free pixels may be used to (approximately) reconstitute the original (non-normalised) 1.0 Kelvin histogram.

Confidence word. Table 21 describes the contents of the spatially-averaged cloud temperature/coverage product confidence word.

Bit 0 is set if the contributing nadir-view data were acquired by ATSR during day-time; that is, when the local solar elevation was positive.

Bit 1 is set if the contributing forward-view data were acquired by ATSR during day-time; that is, when the local solar elevation was positive.

Bit 2 is set if the half-degree cell contains land (whether cloudy or otherwise); bit 3 is set if the half-degree cell contains sea (whether cloudy or otherwise). Note that bits 2 *and* 3 will be set if a half-degree cell is a mixture of land and sea; that is, when it contains a coast-line.

Bits 4 to 15 are unused.

Bit number(s)	**Meaning if set**
0 (lsb)	Contributing nadir-view data acquired during day-time
1	Contributing forward-view data acquired during day-time
2	Half-degree cell contains land
3	Half-degree cell contains sea
4 – 15 (msb)	Unused

Table 21: Spatially-averaged cloud temperature/coverage product confidence word

10.4 Notes

- This product is new to version 600.
- No explicit or implicit ordering of product values may be assumed. The Earth-location must be derived using the latitude and longitude information provided with each cloud temperature/coverage record.
- Note that, apart from the time/position information, and the confidence word, all product contents are duplicated for nadir and forward views. There is no attempt to combine nadir and forward views within a cloud temperature retrieval algorithm.
- In common with the other spatially-averaged products, the derivation of the cloud temperature/coverage product does not involve the precise geolocation of ATSR pixels, and the remapping of nadir- and forward-view pixels onto a 1km grid whose axes are the along-track and across-track directions. The pixels which contribute to this product are those within the ATSR scan; each ATSR scan contains 555 nadir-view pixels, and 371 forward-view pixels. When considering the numbers of cloudy and cloud-free pixels supplied within this product, it should be understood that a nadir:forward ratio of 555 : 371 is to be expected.
- Exceptional values within the spatially-averaged cloud temperature/coverage product are described in table 22.

10.5 Sample code

The following sample VAX C code demonstrates how the spatially-averaged cloud temperature/coverage product might be read. It assumes that the **short** type is a two-byte integer, and that the **long** type is a four-byte integer.

```
/*-------------------------------------------------------------*/

#include <file.h>
```

Parameter	Value	Reason
Number of cloudy pixels	-999	Fewer than 20 cloudy pixels identified
Number of cloud-free pixels	-999	Fewer than 20 cloudy pixels identified
Spatially-averaged brightness temperature	-999	Fewer than 20 cloudy pixels identified
Standard deviation of brightness temperatures	-999	Fewer than 20 cloudy pixels identified
Lowest brightness temperature	-999	Fewer than 20 cloudy pixels identified
Cloud-top temperature	-999	Fewer than 20 cloudy pixels identified
Percentage cloud-cover	-999	Fewer than 20 cloudy pixels identified

Table 22: Spatially-averaged cloud temperature/coverage product exceptional values

```
#include <unixio.h>

/*-----------------------------------------------------------------*/

struct acloud_confidence{
                           int nadir_day_night: 1;
                           int frwrd_day_night: 1;
                           int land: 1;
                           int sea: 1;
                           int unused: 12;
                        };

struct view_info{
                   short number_of_cloudy_pixels;
                   short number_of_cloud_free_pixels;
                   short mean_cloudy_bt;
                   short cloudy_bt_deviation;
                   short lowest_cloudy_bt;
                   short cloud_top_temperature;
                   short percentage_cloudy;
                   unsigned char histogram[100];
                };

struct half_degree_cell{
                          long day_number;
                          long secs_in_day;
                          short latitude;
                          short longitude;
                          short across_track_band;
                          struct view_info nadir;
                          struct view_info frwrd;
                          struct acloud_confidence confidence;
                       };

/*-----------------------------------------------------------------*/

struct half_degree_cell cell_array[20000];

/*-----------------------------------------------------------------*/

main()
{

   int acloud_file, cell_loop = 0;
```

```
   acloud_file = open("ball$303251725_25000_30325_x600.acloud", O_RDONLY, 0, "rfm = fix", "mrs = 244");

   while (read(acloud_file, (void *) &cell_array[cell_loop++], 244) == 244);

   close(acloud_file);

}

/*--------------------------------------------------------------*/
```

The following sample IDL code demonstrates how the spatially-averaged cloud temperature/coverage product might be read.

```
;/*------------------------------------------------------*/

on_ioerror, end_of_file

nadir_percent_map = intarr(720, 360) & frwrd_percent_map = intarr(720, 360)
nadir_ctt_map = intarr(720, 360) & frwrd_ctt_map = intarr(720, 360)

acloud_rec = { file_record, time: lonarr(2), position: intarr(3), $
               nadir_stuff: intarr(7), nadir_histogram: bytarr(100), $
               frwrd_stuff: intarr(7), frwrd_histogram: bytarr(100), $
               confidence: intarr(1) }

acloud_stuff = replicate(acloud_rec, 20000)

;/*------------------------------------------------------*/

openr, 1, 'ball$303251725_25000_30325_x600.acloud', /fixed, 244

acloud_loop = 0

read_from_sadist_file:

readu, 1, acloud_rec

nadir_percent_map(acloud_rec.position(1), acloud_rec.position(0)) = acloud_rec.nadir_stuff(6)
frwrd_percent_map(acloud_rec.position(1), acloud_rec.position(0)) = acloud_rec.frwrd_stuff(6)

nadir_ctt_map(acloud_rec.position(1), acloud_rec.position(0)) = acloud_rec.nadir_stuff(5)
frwrd_ctt_map(acloud_rec.position(1), acloud_rec.position(0)) = acloud_rec.frwrd_stuff(5)

acloud_stuff(acloud_loop) = acloud_rec

acloud_loop = acloud_loop + 1

goto, read_from_sadist_file

end_of_file:

close, 1

end

;/*------------------------------------------------------*/
```

11 ATSR instrument engineering product (ENG)

11.1 General description

The ATSR instrument engineering product contains temporally-averaged analogue engineering parameters from ATSR instrument telemetry. The averaging period is one minute; times of data-acquisition are included within the product.

The product has neither primary nor secondary headers; it contains ATSR instrument engineering parameters derived from up to a complete file of ATSR raw data (which may in most circumstances be considered to be equivalent to one ERS-1 orbit).

This product has a fixed-length 560-byte record format; it has a variable length, though the largest volume of ATSR raw data which can contribute to a single ATSR instrument engineering product (approximately one orbit) places an upper limit on the product size.

11.2 Product format

The contents of each 560-byte product record are shown in table 23. The ATSR engineering parameters contained within each record are listed in tables 24 and 25. This telemetry information has been taken from the RAL document ER-IS-RAL-AT-0019 [6], which should be the source for further details.

In order to facilitate representation of engineering parameters in integer form, to increase the portability of this product, each engineering parameter is stored within the product as mantissa and exponent components. The mantissa is a four-byte integer, which maintains nine significant figures; the exponent is a one-byte integer, which indicates the power of ten by which the mantissa must be multiplied, to reconstitute the floating-point representation. Therefore:

$$parameter = mantissa * 10^{exponent}$$

It is believed that the loss of precision which is caused by the maintenance within this product of only nine significant figures for each engineering parameter is not relevant.

Byte range	**Parameter description**	**Type**	**Unit**
0 – 3	Time of data (days since January 1st, 1950)	Integer	Days
4 – 7	Time of data (seconds within current day)	Integer	Seconds
8 – 375	Mantissa components (each four-bytes) of one-minute temporal averages of ninety-two instrument engineering parameters (see tables 24 and 25)	Integer	Various
376 – 467	Exponent components (each one-byte) of one-minute temporal averages of ninety-two instrument engineering parameters (see tables 24 and 25)	Integer	Various
468 – 559	Number of samples (each one-byte) in each of the ninety-two one-minute averages	Integer	None

Table 23: ATSR instrument engineering product record

11.3 Notes

- This product has been revised for SADIST version 600, and is documented here for the first time.
- The integer value containing the number of days since January 1st, 1950, does not include the current, incomplete day.

	TM.Z	**Parameter description**	**Unit**
1	401	Pixel phase monitor	Degrees
2	501	Scan mechanism encoder temperature	Kelvin
3	502	Focal plane assembly vacuum jacket	Kelvin
4	503	Focal plane assembly wall plate adapter	Kelvin
5	504	Optical bench diagonal midpoint -Z	Kelvin
6	505	Optical bench base beam midpoint -Z	Kelvin
7	506	IR enclosure adjacent to cooler +Z	Kelvin
8	507	Temperature adjacent to optical bench near +XBB	Kelvin
9	508	Temperature adjacent to optical bench near -XBB	Kelvin
10	509	IR chassis under optical bench	Kelvin
11	510	ATSR baseplate adjacent to base of cooler electronics unit	Kelvin
12	511	IR enclosure above scan mirror on +Z	Kelvin
13	512	IR enclosure inside +Y face	Kelvin
14	513	IR enclosure outside -Y face	Kelvin
15	514	IR enclosure -Z face at -X rim of nadir baffle	Kelvin
16	515	IR enclosure -Z face at -Y position on baffle	Kelvin
17	516	Signal channel electronics housing inside surface	Kelvin
18	517	IEU box base temperature	Kelvin
19	518	Optical bench base beam adjacent to +XBB	Kelvin
20	519	IR enclosure -Z face at +X rim of nadir baffle	Kelvin
21	520	IR enclosure -Z face at -X rim of nadir baffle	Kelvin
22	521	Along-track baffle +Z face facing MWR instrument	Kelvin
23	522	Along-track baffle -Z temperature	Kelvin
24	523	Along-track baffle +X temperature	Kelvin
25	524	Along-track baffle -X temperature	Kelvin
26	525	Along-track baffle inner rim +X temperature	Kelvin
27	526	Along-track baffle inner rim +Z temperature	Kelvin
28	527	Along-track baffle inner rim -Z temperature	Kelvin
29	528	Along-track baffle inner rim -X temperature	Kelvin
30	529	Nadir baffle inner rim +X temperature	Kelvin
31	530	Nadir baffle inner rim +Y temperature	Kelvin
32	531	Nadir baffle inner rim -Y temperature	Kelvin
33	532	Nadir baffle inner rim -X temperature	Kelvin
34	551	Fixed mirror field stop temperature 1	Kelvin
35	552	Fixed mirror field stop temperature 2	Kelvin
36	553	Fixed mirror field stop temperature 3	Kelvin
37	554	Fixed mirror mount temperature	Kelvin
38	555	Focal plane assembly baffle temperature	Kelvin
39	556	SCC cold tip temperature	Kelvin
40	557	SCC cooling block temperature	Kelvin
41	558	SCC precision resistance thermometer reference voltage	Counts
42	562	1.6μm detector temperature	Kelvin
43	563	3.7μm detector temperature	Kelvin
44	564	11.0μm detetctor temperature	Kelvin
45	565	12.0μm detector temperature	Kelvin
46	566	Focal plane assembly base plate temperature (not functioning)	Kelvin

Table 24: ATSR instrument engineering parameters (1 to 46)

	TM.Z	**Parameter description**	**Unit**
47	602	+XBB temperature 1	Kelvin
48	604	-XBB temperature 1	Kelvin
49	606	+XBB temperature 5	Kelvin
50	608	BB electronics unit temperature	Kelvin
51	610	-XBB temperature 5	Kelvin
52	612	+XBB temperature 2	Kelvin
53	614	+XBB temperature 3	Kelvin
54	616	+XBB temperature 4	Kelvin
55	618	+XBB temperature 6	Kelvin
56	620	+XBB temperature 7	Kelvin
57	622	-XBB temperature 2	Kelvin
58	624	-XBB temperature 3	Kelvin
59	626	-XBB temperature 4	Kelvin
60	628	-XBB temperature 6	Kelvin
61	630	-XBB temperature 7	Kelvin
62	701	SCC operating frequency	Hertz
63	702	SCC compressor A drive amplitude	mVolts (rms)
64	703	SCC compressor A/B amplitude balance	Ratio
65	704	SCC compressor A drive offset	mVolts (dc)
66	705	SCC compressor B drive offset	mVolts (dc)
67	706	SCC displacer drive amplitude	mVolts (rms)
68	707	SCC displacer drive offset	mVolts (dc)
69	708	SCC displacer phase	Degrees
70	709	SCC control temperature	Kelvin
71	710	SCC mean pressure (not operational)	Psi Abs
72	711	SCC +10V reference monitor	mVolts
73	712	SCC -10V reference monitor	mVolts
74	713	SCC +12V supply monitor	mVolts
75	714	SCC -12V supply monitor	mVolts
76	715	SCC block heater monitor	mWatts
77	716	SCC compressor A current (measured as voltage)	mVolts
78	717	SCC compressor B current (measured as voltage)	mVolts
79	718	SCC displacer loop error	mVolts
80	719	SCC compressor A position pickoff amplitude	mVolts (rms)
81	720	SCC compressor B position pickoff amplitude	mVolts (rms)
82	721	SCC displacer position pickoff amplitude	mVolts (rms)
83	722	SCC compressor A position pickoff temperature	Kelvin
84	723	SCC compressor B position pickoff temperature	Kelvin
85	724	SCC displacer position pickoff temperature	Kelvin
86	725	SCC displacer flange temperature	Kelvin
87	726	SCC compressor head temperature	Kelvin
88	727	SCC compressor amplifier temperature	Kelvin
89	900	IRR converter 1 input power monitor	Watts
90	901	ICU converter 2 input power monitor	Watts
91	902	DEU temperature monitor 2 (ICU)	Kelvin
92	903	DEU temperature monitor 1 (PCSU)	Kelvin

Table 25: ATSR instrument engineering parameters (47 to 92)

- One minute of ATSR raw data comprises 400 scans/packets. Within SADIST processing, this corresponds to five blocks, each of eighty scans/packets.
- Each of the engineering parameters within this product is telemetered only once in each frame of eight packets. During the full 400 scan/packet averaging period, there are therefore only fifty occurrences of each parameter.
- If the ATSR raw data available to SADIST does not comprise an integer multiple of 400 packets, the averaging period of the last set will be less than one minute; this may be inferred from the sample count supplied with each parameter. If at least three eighty-scan blocks contribute, the time which is supplied with each set of one-minute temporal averages is that of the fortieth ATSR scan within the third block. Otherwise (i.e. fewer than three blocks contribute) the time is that of the fortieth scan within the last block.

11.4 Sample code

The following sample VAX C code demonstrates how the ATSR instrument engineering product might be read. It assumes that the **short** type is a two-byte integer, and that the **long** type is a four-byte integer.

```
/*---------------------------------------------------------------*/

#include <file.h>
#include <math.h>
#include <unixio.h>

/*---------------------------------------------------------------*/

struct{
         long day_number;
         long secs_in_day;
         long mantissa[92];
         char exponent[92];
         unsigned char sample_count[92];
      } eng_record;

struct eng_converted{
                       double eng_time;
                       double eng_value[92];
                       unsigned char eng_count[92];
                    };

/*---------------------------------------------------------------*/

struct eng_converted eng_stuff[120];

/*---------------------------------------------------------------*/

main()
{

   int eng_file, record_loop = 0, parameter_loop, bytes_read;

   eng_file = open("hunt$402091245_05220_40209_x600.eng", O_RDONLY, 0, "rfm = fix", "mrs = 560");

/* Read the derived one-minute averages from the product file. */

   while ((bytes_read = read(eng_file, (void *) &eng_record, 560)) == 560)
```

```
     {
        eng_stuff[record_loop].eng_time = (double) eng_record.day_number +
                                         ((double) eng_record.secs_in_day / 86400.0);

        for (parameter_loop = 0; parameter_loop < 92; parameter_loop++)
        {
           eng_stuff[record_loop].eng_value[parameter_loop] =
                                       (double) eng_record.mantissa[parameter_loop] *
                               pow(10, (double) eng_record.exponent[parameter_loop]);

           eng_stuff[record_loop].eng_count[parameter_loop] = eng_record.sample_count[parameter_loop];
        }

        record_loop++;

     }

     close(eng_file);

  }

  /*--------------------------------------------------------------*/
```

The following sample IDL code demonstrates how the ATSR instrument engineering product might be read.

```
  ;/*------------------------------------------------------*/

  on_ioerror, end_of_file

  eng_rec = { file_record, time: lonarr(2), mantissa: lonarr(92), $
                           exponent: bytarr(92), sample_count: bytarr(92) }

  eng_converted = { converted_record, eng_time: dblarr(1), eng_value: dblarr(92), eng_count: bytarr(92) }

  eng_stuff = replicate(eng_converted, 120)

  ;/*------------------------------------------------------*/

  openr, 1, 'hunt$402111050_12345_40211_x600.eng', /fixed, 560

  eng_loop = 0

  read_from_eng_file:

  readu, 1, eng_rec

  eng_stuff(eng_loop).eng_time = double(eng_rec.time(0)) + (double(eng_rec.time(1)) / 86400.0)

  for parameter_loop = 0, 91 do begin

;/* This exponent derivation is necessary since IDL does not have a signed-byte data-type. */

     exponent = fix(eng_rec.exponent(parameter_loop))
     if (exponent gt 127) then exponent = exponent - 256

     eng_stuff(eng_loop).eng_value(parameter_loop) = double(eng_rec.mantissa(parameter_loop)) * $
                                             (10.0 ^ double(exponent))

     eng_stuff(eng_loop).eng_count(parameter_loop) = eng_rec.sample_count(parameter_loop)
```

```
endfor

eng_loop = eng_loop + 1

goto, read_from_eng_file

end_of_file:

close, 1

end

;/*-------------------------------------------------------*/
```

12 ATSR/M microwave instrument raw data product (MWR)

12.1 General description

The ATSR/M microwave instrument raw data product contains the raw data from the ATSR/M microwave instrument (natch), with auxiliary information describing source packet times, the ATSR/IR and ATSR/M sub-commutation frames and instrument status, and the active instrument (SAR, SCATT, RA) blanking pulses.

The product has neither primary nor secondary headers; it contains records derived from up to a complete file of ATSR raw data (which may in most circumstances be considered to be equivalent to one ERS-1 orbit).

This product has a stream format, with implied 21-byte records; it has a variable length, though the largest volume of ATSR raw data which can contribute to a single ATSR/M microwave instrument raw data product (approximately one orbit) places an upper limit on the product size.

12.2 Product format

The contents of each 21-byte product record are shown in table 26. The record contents are transcribed from a single ATSR source packet, and source packet byte-ordering is maintained. This means that byte-ordering does not in all cases follow the DEC standard, and that some decoding is necessary.

Byte(s)	Source	Bits (0 is lsb)	TM #	Parameter description	Type	Unit
0 – 3	6 – 9	0 – 31	TM.Z105	Satellite clock at time of packet	Integer	sec/256
4	11	5 – 7	TM.Z107	ATSR/IR frame counter	Integer	None
4	11	0 – 4	TM.Z106	ATSR/M frame counter	Integer	None
5	48	0 – 1	TM.Z219	ATSR/M status word	Integer	None
6	49	6 – 7	TM.Z220	ATSR/IR status word	Integer	None
7 – 8	136 – 137	0 – 10	TM.Z952	SAR blanking pulse pixel	Integer	Count
7 – 8	136 – 137	11 – 15	TM.Z951	SAR blanking pulse sub-pixel offset	Integer	Count
9 – 10	138 – 139	0 – 10	TM.Z958	SCATT blanking pulse pixel	Integer	Count
9 – 10	138 – 139	11 – 15	TM.Z957	SCATT blanking pulse sub-pixel offset	Integer	Count
11 – 12	140 – 141	0 – 10	TM.Z964	RA blanking pulse pixel	Integer	Count
11 – 12	140 – 141	11 – 15	TM.Z963	RA blanking pulse sub-pixel offset	Integer	Count
13 – 20	3992 – 3999	All	–	ATSR/M microwave raw data	Composite	Various

Table 26: ATSR/M microwave instrument raw data product record

Satellite clock (TM.Z105)

Bytes 0 to 3 within each product record contain bytes 6 to 9 from the ATSR source packet. These bytes hold the value of the satellite clock at the time of the source packet.

The bytes are held in decreasing significance; that is, the most significant byte is first, and the least significant byte is last. This does not conform to the DEC standard for integer representation (which is to hold bytes in increasing significance). On a DEC system, therefore, the bytes must be swapped if they are to be interpreted as a 32-bit word.

A mechanism for doing this using VAX C is shown in the sample code.

Note that this product does not contain calibration parameters for the satellite clock. The clock has a resolution of approximately 4 milliseconds; given an ATSR scan time of 150 milliseconds, the clock increments by an average of 38.4 per scan/packet. The clock has a range of approximately six months, so a number of resets are expected within the mission. Users who require calibration parameters for the satellite clock should contact RAL.

ATSR/IR (TM.Z107) and ATSR/M (TM.Z106) frame counters

Byte 4 within each product record contains byte 11 from the ATSR source packet. This byte holds the ATSR/IR and ATSR/M instrument frame counters, which specify packet numbers within the data sub-commutation period for the instruments.

The most significant three bits contain the ATSR/IR frame counter. This cycles from 0 to 7, defining packets within the ATSR/IR frame of eight scans/packets.

The least significant five bits contain the ATSR/M frame counter. This cycles from 0 to 31, defining packets within the ATSR/M frame of thirty-two scans/packets.

ATSR/IR (TM.Z220) and ATSR/M (TM.Z219) status words

Byte 5 within each product record contains byte 48 from the ATSR source packet. This byte holds the ATSR/M instrument status word. Note that this is a two-bit word, which is stored within the least significant two bits of the byte, and that the status word is valid only for ATSR/IR frame number 5. In other frames, the meaning is undefined.

Byte 6 within each product record contains byte 49 from the ATSR source packet. This byte holds the ATSR/IR instrument status word. Note that this is a two-bit word, which is stored within the most significant two bits of the byte, and that the status word is valid only for ATSR/IR frame number 5. In other frames, the meaning is undefined.

Table 27 defines the possible values for each of these parameters.

Value	Meaning
0	Instrument unpowered
1	Instrument unpowered
2	Instrument in run-up
3	Instrument nominal

Table 27: ATSR/M and ATSR/IR instrument status words

SAR (TM.Z952 & TM.Z951), SCATT (TM.Z958 & TM.Z957) and RA (TM.Z964 & TM.Z963) blanking pulses

Bytes 7 and 8 within each product record contain bytes 136 and 137 from the ATSR source packet. These bytes hold the position within the ATSR scan of the first blanking pulse from the Synthetic Aperture Radar instrument. Bytes 9 and 10 within each product record contain bytes 138 and 139 from the ATSR source packet. These bytes hold the position within the ATSR scan of the first blanking pulse from the Wind Scatterometer instrument. Bytes 11 and 12 within each product record contain bytes 140 and 141 from the ATSR source packet. These bytes hold the position within the ATSR scan of the first blanking pulse from the Radar Altimeter instrument.

Note that, in each case, the byte-ordering is the same as that within the source packet: the most significant byte is first. The sample code shows how the original two-byte words might be reconstructed on a DEC system, using VAX C.

Within each two-byte word, the least significant eleven bits contain a count value describing the number of the ATSR pixel during which the first active instrument blanking pulse occurred. The calibration is:

$$pixel\_number = count + 2$$

where *pixel_number* has the range 1 to 2000. This calibration includes a consideration of the fact that the count value does not allow for the one-pixel delay between integration and transmission.

The most significant five bits contain the delay between the pixel synchronisation pulse and the start of the blanking pulse; this may be used to locate the position of the blanking pulse at a sub-pixel resolution. The calibration of this sub-pixel delay is given in table 28.

Count	Sub-pixel delay/μs
0	0.0 – 2.0
1	1.2 – 5.2
2	4.4 – 8.4
3	7.6 – 11.6
4	10.8 – 14.8
5	14.0 – 18.0
6	17.2 – 21.2
7	20.4 – 24.4
8	23.6 – 27.6
9	26.8 – 30.8
10	30.0 – 34.0
11	33.2 – 37.2
12	36.4 – 40.4
13	39.6 – 43.6
14	42.8 – 46.8
15	46.0 – 50.0
16	49.2 – 53.2
17	52.4 – 56.4
18	55.6 – 59.6
19	58.8 – 62.8
20	62.0 – 66.0
21	65.2 – 69.2
22	68.4 – 72.4
23	71.6 – 75.6
24	74.8 – 78.8
25 – 31	Not valid

Table 28: Calibration of blanking pulse sub-pixel offsets

12.2.1 ATSR/M microwave raw data

Since SADIST performs no processing of the ATSR/M microwave raw data, it is not within the scope of this document to describe its format and content. The document ER-TN-CRP-AT-0042 [5] provides a general description of the ATSR/M instrument and its telemetry.

12.3 Notes

- This product has been revised for SADIST version 600, and is documented here for the first time.
- Note that records within this product may not be continuous. Where a source packet is not present within the the telemetered raw data, no record will be produced. The satellite clock should be the basis for detection of such discontinuity.

12.4 Sample code

The following sample VAX C code demonstrates how the ATSR/M microwave instrument raw data product might be read. It assumes that the **short** type is a two-byte integer, and that the **long** type is a four-byte integer.

```
/*--------------------------------------------------------------------*/

#include <file.h>
#include <unixio.h>

/*--------------------------------------------------------------------*/

struct interpreted_mwr{
                         unsigned long clock;
                         unsigned char irr_frame; unsigned char mwr_frame;
                         char irr_status; char mwr_status;
                         short sar_blank_pixel; short sar_blank_offset;
                         short scatt_blank_pixel; short scatt_blank_offset;
                         short ra_blank_pixel; short ra_blank_offset;
                         unsigned char mwr_raw_data[8];
                      };

/*--------------------------------------------------------------------*/

struct interpreted_mwr mwr_stuff[50000];

/*--------------------------------------------------------------------*/

void interpret_mwr_record(unsigned char mwr_record[21], long record_count)
{

   int byte_loop;

   unsigned short sar_blank, scatt_blank, ra_blank;

/* Reconstitute four-byte satellite binary clock. Note that the bytes are ordered in
      decreasing significance. */

   mwr_stuff[record_count].clock = ((unsigned long) mwr_record[0] << 24) |
                                   ((unsigned long) mwr_record[1] << 16) |
                                   ((unsigned long) mwr_record[2] << 8) |
                                   ((unsigned long) mwr_record[3]);

/* Extract the ATSR/IR and ATSR/M frame counters. */

   mwr_stuff[record_count].irr_frame = (mwr_record[4] & 0xe0) >> 5;
   mwr_stuff[record_count].mwr_frame = (mwr_record[4] & 0x1f);

/* Extract the ATSR/IR and ATSR/M status words. Note that these words are valid only
      within ATSR/IR frame number five. */

   if (mwr_stuff[record_count].irr_frame == 5)
   {
      mwr_stuff[record_count].irr_status = (mwr_record[6] & 0xc0) >> 6;
      mwr_stuff[record_count].mwr_status = (mwr_record[5] & 0x03);
   }
   else
   {
      mwr_stuff[record_count].irr_status = -1;
      mwr_stuff[record_count].mwr_status = -1;
   }
```

```
/* Decode the SAR, SCATT, and RA blanking pulse summaries. First, the original two-byte
      words must be reconstructed; then the pixel numbers and sub-pixel offsets can be
      extracted. */

   sar_blank = (unsigned short) mwr_record[7] << 8 | (unsigned short) mwr_record[8];
   scatt_blank = (unsigned short) mwr_record[9] << 8 | (unsigned short) mwr_record[10];
   ra_blank = (unsigned short) mwr_record[11] << 8 | (unsigned short) mwr_record[12];

   mwr_stuff[record_count].sar_blank_pixel = (sar_blank & 0x07ff);
   mwr_stuff[record_count].sar_blank_offset = (sar_blank & 0xf800) >> 11;

   mwr_stuff[record_count].scatt_blank_pixel = (scatt_blank & 0x07ff);
   mwr_stuff[record_count].scatt_blank_offset = (scatt_blank & 0xf800) >> 11;

   mwr_stuff[record_count].ra_blank_pixel = (ra_blank & 0x07ff);
   mwr_stuff[record_count].ra_blank_offset = (ra_blank & 0xf800) >> 11;

/* Store the ATSR/M raw data. */

   for (byte_loop = 0; byte_loop < 8; byte_loop++)
      mwr_stuff[record_count].mwr_raw_data[byte_loop] = mwr_record[byte_loop + 13];

}

/*--------------------------------------------------------------------*/

main()
{

   unsigned char mwr_record[21];

   long mwr_file, bytes_read, mwr_loop = 0;

   mwr_file = open("hurst$402041110_25000_40204_x600.mwr", O_RDONLY, 0);

   while ((bytes_read = read(mwr_file, mwr_record, 21)) == 21)
   {
      interpret_mwr_record(mwr_record, mwr_loop);
      mwr_loop++;
   }

   close(mwr_file);

}

/*--------------------------------------------------------------------*/
```

A SADIST product file nomenclature

Each SADIST product has a file-name which contains information specifying:

- Time of data acquisition (in the form of the time of the previous ascending node crossing);
- Along-track distance, or relative scan number (from ascending node) of product start;
- Date of product generation;
- Version of SADIST used for product generation;
- Product type.

The format of a SADIST product file-name is:

$$requestor\$yr_n mo_n dy_n hr_n mi_n\_dist\_yr_g mo_g dy_g\_vers.type$$

where:

requestor is a string of up to twelve characters which defines the requestor of the product. Note that this information has particular relevance at RAL, in which place products requested by many individuals and/or organisations may reside.

yr_n is the single-digit year number of the time of the last ascending node crossing before the start of the product; 1 is 1991, 2 is 1992 ....

mo_n is the two-digit month number of the time of the last ascending node crossing before the start of the product; 1 is January, 2 is February ....

dy_n is the two-digit number of the day within the month specified by mo_n.

hr_n is the two-digit number of the hour within the day specified by dy_n.

mi_n is the two-digit number of the minute within the hour specified by hr_n.

dist is the distance (in km) along the sub-satellite track from the last ascending node crossing to the start of the product, or the number of the first ATSR scan to contribute to the product, relative to the last ascending node crossing.

yr_g is the single-digit year number of the time of product generation; 1 is 1991, 2 is 1992 ....

mo_g is the two-digit month number of the time of product generation; 1 is January, 2 is February ....

dy_g is the two-digit number of the day within the month specified by mo_g.

vers is the number of the SADIST version which was used to derive the product. The three-digit version number is preceded by a letter **t** in pre-operational versions of SADIST; by a letter **x** in operational VAX versions of SADIST; and by a letter **a** in operational Alpha AXP versions of SADIST.

type is the product type:

acloud spatially-averaged cloud temperature/coverage product.

alst spatially-averaged land-surface temperature product.

asst spatially-averaged sea-surface temperature product.

browse brightness temperature browse image product.

bt... brightness temperature image product.

cloud land-flagging/cloud-identification results image product.

counts decoded detector count image product.

eng ATSR instrument engineering product.

mcloud ATSR/M microwave cloud product.

mwr ATSR/M microwave instrument raw data product.

nsst nadir-only sea-surface temperature image product.

sst sea-surface temperature image product.

For example, the SADIST product:

stiles$109041400_15000_10905_x600.sst

is a sea-surface temperature image product, which was requested by **stiles**. The data used were acquired during the orbit whose ascending node crossing was on September 4, 1991, at 14:00, and the product was generated on September 5, 1991. The product begins at an along-track distance of 15000km; it was generated using SADIST version 600, running on a VAX system.

A.1 Brightness temperature image product file-name extension

Apart from the primary and secondary headers, the contents of brightness temperature image products are wholly selectable by the the SADIST user at the time of generation. In addition to values within the primary header, the product contents which are actually present may be determined using the extension to the product file-name:

- If the extension is **bt**, the product is complete; all optional contents are present. Otherwise, the product is not complete. The extension will be **bt-**, followed by a series of characters indicating which product contents *are* present.
- If the extension includes **g**, the pixel geolocation information (pixel latitude/longitude values and x/y offsets) is present.
- If the extension includes **na**, all nadir-view channel images (12.0μm, 11.0μm, 3.7/1.6μm) are present.
- If the extension includes **n1**, the nadir-view 12.0μm image is present.
- If the extension includes **n2**, the nadir-view 11.0μm image is present.
- If the extension includes **n3**, the nadir-view 3.7/1.6μm image is present.
- If the extension includes **fa**, all forward-view channel images (12.0μm, 11.0μm, 3.7/1.6μm) are present.
- If the extension includes **f1**, the forward-view 12.0μm image is present.
- If the extension includes **f2**, the forward-view 11.0μm image is present.
- If the extension includes **f3**, the forward-view 3.7/1.6μm image is present.

A.2 Brightness temperature browse image product file-name extension

Apart from the primary header, the contents of brightness temperature browse image products are wholly selectable by the the SADIST user at the time of generation. In addition to values within the primary header, the product contents which are actually present may be determined using the extension to the product file-name:

- If the extension is **browse**, the product is complete; all optional contents are present. Otherwise, the product is not complete. The extension will be **browse-**, followed by a series of characters indicating which product contents *are* present.
- If the extension includes **na**, all sub-sampled nadir-view channel images (12.0μm, 11.0μm, 3.7/1.6μm) are present.
- If the extension includes **n1**, the sub-sampled nadir-view 12.0μm image is present.
- If the extension includes **n2**, the sub-sampled nadir-view 11.0μm image is present.
- If the extension includes **n3**, the sub-sampled nadir-view 3.7/1.6μm image is present.
- If the extension includes **fa**, all sub-sampled forward-view channel images (12.0μm, 11.0μm, 3.7/1.6μm) are present.
- If the extension includes **f1**, the sub-sampled forward-view 12.0μm image is present.
- If the extension includes **f2**, the sub-sampled forward-view 11.0μm image is present.
- If the extension includes **f3**, the sub-sampled forward-view 3.7/1.6μm image is present.

A.3 Notes

- Note that, within the file-names of products which are not precisely-geolocated images (COUNTS, ASST, ALST, ACLOUD, MWR, MCLOUD, ENG), the along-track distance is strictly speaking a *relative scan number*; that is, since precise geolocation has not been performed, the value used in such cases is the number of the first ATSR scan, relative to the previous ascending node, to contribute to the product. Though this approximates to the along-track distance, it is not the same thing.

B Product version history

Following are descriptions of changes made within SADIST since its inception which have affected product format or contents.

17th July 1991: Launch of ERS-1; release of SADIST version 100

1st November 1991: Release of SADIST version 110

Format changes introduced in version 110

- Added 512-byte ASCII header to the decoded infra-red count image product.
- Changed format of the 25km geolocation reference grid supplied with level 1.5 and level 2.0 image products. Grid previously stored as 529 8-byte records (each record containing a single latitude/longitude pair); grid now stored as two 2116-byte records (the first containing all 529 latitude values; the second containing all 529 longitude values).

Content changes introduced in version 110

- ATSR scan cone angle value modified; most along-track nadir/forward-view collocation error removed.
- 25km reference grid interpolation error within land-flagging corrected. Error was causing along-track error in land-flagging of up to 25km.
- Visible steps in 12.0μm, 11.0μm and 3.7μm channel images due to automatic detector gain/offset changes have been removed. Previously, calibration coefficients were derived using averaged black-body values; now, single scans are used.
- Incorrect ordering of look-up tables containing black-body brightness temperature to radiance and sea-surface temperature retrieval coefficients has been fixed.
- Incorrect determination of longitude of products occuring before the Equator has been fixed. Such products previously assigned longitude with approximate 25 degree error.
- Removed pitch and roll steering of ATSR scan cone relative to ERS-1 platform; steering of ERS-1 already achieves local vertical pointing of satellite and ATSR z-axes.
- Blanking pulse information now feeds through to level 1.5 and level 2.0 image products.
- Sea-surface temperature image product now contains nadir-only sea-surface temperature values; this will continue whilst nadir/forward-view collocation errors remain.
- Interpolation of sea-surface temperature from derived regional values corrected for areas north of the Equator; fixed incorrect calculation of interpolation weights within high-resolution and spatially-averaged sea-surface temperature derivation.
- Gross cloud check in spatially-averaged sea-surface temperature derivation now uses correct thresholds. This test is latitude-dependent. Incorrect latitudes, resulting from incorrect decoding of the latitude/longitude composite word, was causing lower thresholds than expected to be used; and, hence, less cloud to be detected.

13th November 1991: Release of SADIST version 120

Content changes introduced in version 120

- Approximate 2 K error within derived sea-surface temperatures traced to incorrect ordering of look-up tables containing black-body brightness temperature to radiance and sea-surface temperature retrieval coefficients; this problem was only partially solved within version 110 (see above).

20th December 1991: Release of SADIST version 200

Content changes introduced in version 200

- Extra flag added to the confidence word of the spatially-averaged sea-surface temperature product; this indicates whether each sea-surface temperature value was acquired during the day- or night-time.
- A pitch correction of approximately 0.1 degrees has been applied to the ATSR scan cone. Nadir/forward-view collocation errors are now very small.
- Spherical trigonometry within pixel geolocation now uses geocentric, rather than (incorrect) geodetic latitudes. All latitudes are converted to geodetic before being written to SADIST image products.
- New cloud test introduced. This compares nadir 11.0μm/12.0μm and nadir/forward 11.0μm differences. If the difference between the differences is large (initial threshold is 0.6 K), both nadir and forward view pixels are cloudy. This test is also performed within spatially-averaged sea-surface temperature derivation; in this case, the test uses ten-arcminute averages of brightness temperature.
- High-resolution images which cross the Equator are now permitted. Previously, this exceptional case caused many problems within geolocation.

22nd January 1992: Release of SADIST version 210

Content changes introduced in version 210

- Since geolocation errors still exist, and land-flagging has a low resolution, sea-surface temperature values are derived for all land/sea pixels, in the interests of not discarding valid data. Contents of confidence word can still be used to retrieve results of land-flagging/cloud-identification.
- Fixed incorrect use of cloud-identification thresholds due to erroneous array subscripting.

23rd January 1992: Release of SADIST version 220

Content changes introduced in version 220

- Geolocation error of approximately 0.2 degrees (20km) traced to persistent incorrect conversion of the 25km geolocation reference grid to geodetic latitudes; interpolation uses the grid in a geocentric form; latitudes are converted to geodetic before being written to image products.

17th February 1992: Release of SADIST version 230

Content changes introduced in version 230

- Changes made to 1.6μm reflectance histogram cloud test. 100% reflectance reference value modified; detection of sunglint improved; new histogram parameters and relectance thresholds provided.

4th March 1992: Release of SADIST version 300

Format changes introduced in version 300

- Larger (1024-byte) headers provided with level 1.5 and level 2.0 high-resolution image products.
- 25km geolocation reference grid now stored as scaled integers. Each value has units of degrees/1000.

Content changes introduced in version 300

- To reduce "record grooving" (visible change from scan to scan in brightness temperature and sea-surface temperature images), black-body temperature measurements now averaged over a period of scans. To avoid previous visible calibration changes, detector gains and offsets still extracted from each scan.

6th March 1992: Release of SADIST version 310

Content changes introduced in version 310

- Corrected interpolation errors within pixel geolocation at edges of nadir and forward-view scans leading to spatially-averaged sea-surface temperature derivation. Note that only spatially-averaged sea-surface temperature products affected.

24th July 1992: Release of SADIST version 400

Content changes introduced in version 400

- New high-resolution land-flag introduced. Land-flag derived at RAL from CIA's World Data Bank II. Callable at three resolutions: one degree; six arcminutes (1/10 degree); and thirty-six arcseconds (1/100 degree). Highest resolution used for high-resolution land-flagging; medium resolution used for spatially-averaged land-flagging.
- Land-flagging error of approximately 0.19 degrees traced to use of geocentric—rather than geodetic—latitudes (again). 25km reference grid values now converted to geodetic before use within land-flagging.
- Large geolocation errors detected when using ESRIN-distributed restituted orbital elements within orbit propagation have been fixed; error was caused by SADIST allowing ERSORB to perform correction of position and velocity vectors, but keeping the original state vector time. A mismatch between time, position and velocity resulted, which was anything up to 200km. Note that such errors occurred only when restituted elements were used.
- Filling of 3.7μm/1.6μm image within level 1.5 brightness temperature product improved; missing pixels now filled with the majority channel, whichever that is (previously, filling took place only if one channel was present). Such pixels are flagged (as before) as cosmetically filled.

- Headers of level 1.5 brightness temperature and level 2.0 sea-surface temperature products now contain entry which indicates the type of orbital elements used during orbit propagation (ground-station; ESRIN-predicted; ESRIN-restituted).
- Dual-view, multi-channel sea-surface temperature values now calculated for all cloudy pixels. This makes sense: if the pixel is actually cloud-free, we have the best available sea-surface temperature; if the pixel really is cloudy, the value is invalid anyway. The results of cloud-identification still appear within the confidence word.
- Within sea-surface temperature image products, 11.0μm brightness temperatures are supplied for land pixels; derived sea-surface temperatures would make no sense over land. The good quality of geolocation and land-flagging make this viable.
- Coast/lake flags have been removed from sea-surface temperature image product confidence word; they have never been used by SADIST.
- The partly-cloudy pixel cloud test has been removed; the flag which stored the result of this test within the sea-surface temperature image product confidence word has also been removed. Its place is now taken by a flag which stores the results of cloud-identification on the forward view channels (i.e. bit zero is the nadir cloud flag; bit one is the forward cloud flag).
- Cloud-identification has been much improved: the 1.6μm reflectance histogram test has better sunglint detection, and refined reflectance thresholds; the 11.0μm spatial coherence test performs post-processing of $3 * 3$ pixel box results to identify and restore sharp fronts wrongly identified as cloud; the nadir/forward view-difference test now uses the 3.7μm and 11.0μm channels at night (and the 11.0μm and 12.0μm channels during the day, as before); a new thermal histogram test attempts to identify cloud which has passed all other cloud tests.

26th January 1993: Release of SADIST version 410

Content changes introduced in version 410

- Previously, the "calibration" of the 1.6μm channel used the average of the black-body radiances to derive a calibration bias. This scheme failed if and when the data compression mode was such that the appropriate black-body average was not present. The calibration now uses the engineering parameter (TMZ277) which contains the average black-body radiance in the 1.6μm channel. This is always present.

18th May 1993: Release of SADIST version 500

Format changes introduced in version 500

- As per Section 1.3, all products now have fixed-length record formats.
- The decoded infra-red counts image product (COUNTS) now contains more instrument engineering and channel calibration parameters. Packet times are provided for each image scan.
- The primary high-resolution image products (COUNTS, BT, SST) have redesigned primary headers, whose contents are all ASCII text, and secondary headers, which are provided for post-processing storage by users of SADIST products.

- The geolocated primary high-resolution image products (BT, SST) now have precise geolocation (latitude/longitude and pixel x/y offsets) for each pixel. Interpolation within the 25km reference grid is no longer necessary. The ten-arcminute cell numbers have been removed from the brightness temperature image product; they were obsolete.
- The brightness temperature image product has undergone radical redesign. The channel images are no longer interleaved; this, in conjunction with the new fixed-length record format, enables random access to the product contents to be performed simply. Also, product contents are now requestable by the SADIST user. Geolocation information and channel images may be omitted if they are not required. Values within the primary header and the file-name extension indicate which product contents are actually present.
- The spatially-averaged land-surface temperature product is new to SADIST version 500.

Content changes introduced in version 500

- Calibration of the infra-red channels now extends to lower temperatures. The 12.0μm and 11.0μm channels are calibrated to 150 Kelvin (-123C); the 3.7μm channel is calibrated to 197 Kelvin (-76C). Such calibration is of course not necessary for sea-studies, and is intended to be of assistance to those studying high (and therefore cold) cloud using ATSR data.
- The cosmetic filling of unfilled 3.7/1.6μm pixels within brightness temperature image products now attempts to fill from the nearest pixel (whichever channel that contains). Previously, pixels may only have been used for filling others if they belonged to the majority channel within the image. This change will have an effect only when both 3.7μm and 1.6μm pixels exist within the same image; the result will be that all pixels will be filled, rather than only those neighboured by a pixel from the majority channel.
- Pixel geolocation has been wholly redesigned. The previous scheme, which used spherical trigonometry to perform pixel geolocation, has been abandoned in favour of a scheme which performs geolocation using vector algebra. The current geolocation and collocation accuracies are TBD.
- The thresholds determining maximum and minimum permissible high-resolution sea-surface temperatures within the sea-surface temperature image product (263K and 313K) have been removed. Since the sea-surface temperature product is liable to contain some brightness temperature values (where land has been detected), imposition of thresholds was a pointless exercise, and was leading to possibly-valuable information being removed from the product.
- A bug within 25km reference grid interpolation during land-flagging of brightness temperature image products crossing the Greenwich Meridian has been corrected. The bug resulted in the appearance of circular artefacts within the image land flag.
- The spatially-averaged sea-surface temperature product now contains three sea-surface temperature values per half-degree cell: one derived using nadir-view data only; another using dual-view data only; and a third using the best available data (dual-view if possible; nadir-only otherwise). Examination of which provides the best representation of ground truth will follow.
- A bug in the gross-cloud test within cloud-identification of high-resolution image data has been fixed. The bug related to the derivation of the latitude of an image pixel from the pixel ten-arcminute cell number. Derivation was incorrect, and was resulting in the use of a gross-cloud threshold which was significantly too high, over much of the Earth's surface. This test was effectively disabled, and cloud-identification was forced to rely on other tests.

- The 11.0/12.0μm thermal histogram cloud-identification test now includes a far more rigorous search for the best available histogram peak. The 1.6μm reflectance histogram cloud-identification test now performs a spatial coherence test if a dynamic threshold could not be derived (i.e. if the histogram peak is too poorly defined).
- Values within decoded infra-red detector counts products now use 4095 within all channels to indicate channel saturation.
- Negation of 11.0μm values (rather than 12.0μm) is now used within decoded infra-red detector counts and brightness temperature image products to identify blanking pulses.
- The cosmetic fill flag (previously identified by the negation of ten-arcminute cell numbers in the brightness temperature image product, which are now defunct) is now flagged by the negation of 12.0μm brightness temperature values.
- A bug whose effect was a mis-interpolation of the longitudes of pixels close to the -180/180 degree meridian during spatially-averaged product (ASST, ALST) derivation has been fixed. A small number of pixels were very badly geolocated, and did not contribute to spatially-averaged products.
- Post-processing smoothing is now performed on sea-surface temperature image products. The smoothing involves recalculation of the atmospheric correction for each pixel, based on the mean of the previously-derived atmospheric correction for the **3 * 3** pixel box centred on the pixel in question (the mean correction used is the mean difference between 12.0μm brightness temperature and derived sea-surface temperature values).
- Incorrect calculation of solar azimuth difference angles (within the headers of brightness temperature and sea-surface temperature image products) has been corrected. Elevation and azimuth difference angles are now signed, where appropriate.

25th May 1994: Release of SADIST version 600

Format changes introduced in version 600

- The brightness temperature browse image product (BROWSE) has been redesigned. Sub-sampled brightness temperatures are now provided at the full two-byte temperature resolution. The product may contain sub-sampled versions of each of the nadir- and forward-view images. The product header has been enlarged, and includes the orbit state vector, and instrument detector temperatures.
- The ATSR/M microwave instrument raw data product (MWR) has been redesigned, to conform with the UK-PAF product UK.ATS.0.P100. It is described in this document for the first time.
- The ATSR instrument engineering product (ENG) has been redesigned, It is described in this document for the first time.

Content changes introduced in version 600

- The spatially-averaged cloud temperature/coverage product (ACLOUD) is new to SADIST version 600.
- SADIST v600 supports VAX and Alpha AXP platforms. Though source code is identical, there exist differences between products; these differences are quantified in appendix C. Note that product file-names now contain a letter which identifies the source system.

- Incorrect use of SADIST's 25km pixel geolocation grid was resulting in the generation of a number of image products containing longitudes greater than 180 degrees west; SADIST was failing to adjust such longitudes to account for the crossing of the -180/180 degree meridian. The anomalous image longitudes were causing artefacts within land-flagging, and, hence, cloud-clearing. Affected images were derived from ATSR ascending passes over the western Pacific, and descending passes over northern Asia; these areas are shown in figure 2.

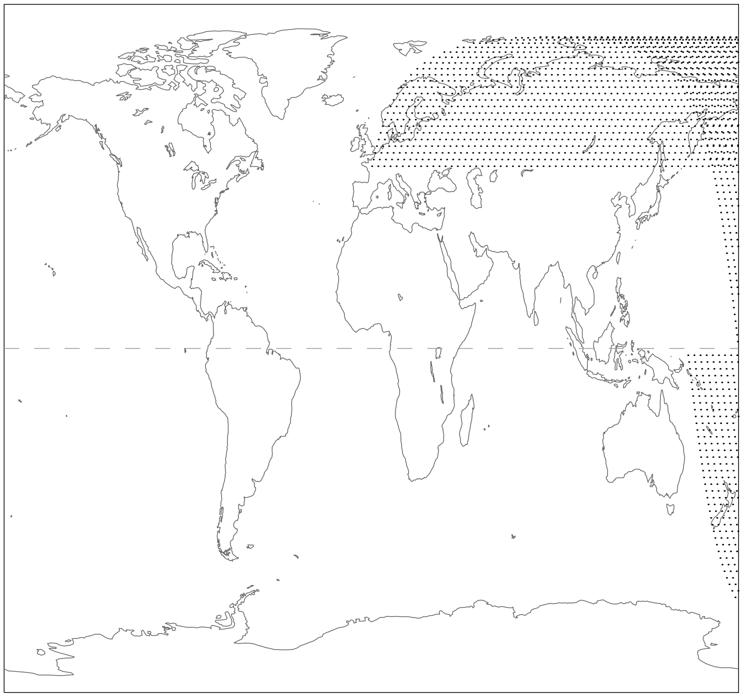

Figure 2: Locations of longitude anomalies due to incorrect geolocation grid adjustment

- The approximate image acquisition time within the headers of brightness temperature image and sea-surface temperature image products has been modified. Previously, there was no precise definition of which time was used; it was possible to assume only that the time was that of an ATSR scan which contributed to the nadir-view image. The time is now that of the scan which supplies the central pixel within the first nadir-view image line. Note that, since ATSR's scan is conical, the other pixels within the first image line are supplied by different (later) scans, and therefore have different (later) times.
- Post-processing smoothing of sea-surface temperature image products now uses the 11.0μm channel, in preference to the 12.0μm channel. ATSR's 11.0μm detector displays a greater signal to noise ratio than its 12.0μm detector, so is a better candidate for the smoothing algorithm.
- The attempt in the development of version 500 to correct a bug whose effect was a mis-interpolation of the longitudes of pixels close to the -180/180 degree meridian during

spatially-averaged product (ASST, ALST) derivation was faulty. One geolocation error was substituted for another; the effect of the new error was to map a number of pixels onto each other, reducing slightly the spatial coverage, and producing a small number of anomalous sea-surface temperatures.

- Calibration of very low 1.6μm reflectances has been tidied up. Prior to version 600, the calibration of very low signals from the 1.6μm detector was resulting in a number of apparently negative reflectances. This is patent nonsense. If the derived reflectance is negative, it is now fixed to -1, which is the recognised value for missing data.
- In exceptional cases, the derivation of spatially-averaged products was failing to gather together all pixels within a half-degree cell. The effect of this was that there appeared within a small number of products to be duplicated values for a small number of cells. Each value was correct, but only for the limited number of pixels which contributed to it. The period over which the spatial-averaging is performed has been increased, and this anomaly no longer exists.
- Incorrect handling of multiple requests for COUNTS products was resulting in the last product of each request (each strip of contiguous products) containing data from different parts of an orbit. Note that this anomaly was present only when multiple requests for COUNTS and higher-level products were made. This is now fixed.
- The 1.6μm histogram cloud-identification test now includes better handling of very narrow histograms, which occur when the 1.6μm signal is very uniform. Location of histogram peaks, and derivation of histogram thresholds, now handle very narrow histograms explicitly.

C Residual differences between SADIST for VAX and Alpha AXP

SADIST v600 for VAX/VMS and SADIST v600 for Alpha AXP originate from identical source code. Their products, however, exhibit small differences, which this section will attempt to quantify, for each of the four primary SADIST products. These differences arise from the radically different architectures present within VAX and Alpha AXP systems; in particular, the two systems use different standards (with different precisions) for double-precision floating-point arithmetic. Geolocation of ATSR pixels within SADIST processing makes free use of double-precision floating-point arithmetic; it is believed that the great majority of product differences are introduced at this stage.

Decoded infra-red count image product (COUNT) differences

Four decoded infra-red count image products were generated by SADIST v600 on VAX and Alpha AXP systems, then compared. The products were:

***$401030950_15978*.counts**
***$401030950_16538*.counts**
***$401030950_17098*.counts**
***$401030950_17658*.counts**

VAX and Alpha AXP versions of these products contained no differences. This is consistent with the assumption that most product differences are introduced by the process of pixel geolocation.

Brightness temperature image product (BT) differences

Ten brightness temperature image products, evenly spaced around a single orbit, were generated by SADIST v600 on VAX and Alpha AXP systems, then compared. The products were:

***$401030950_16000*.bt**
***$401030950_20000*.bt**
***$401030950_24000*.bt**
***$401030950_28000*.bt**
***$401030950_32000*.bt**
***$401030950_36000*.bt**
***$401031130_00001*.bt**
***$401031130_04000*.bt**
***$401031130_08000*.bt**
***$401031130_12000*.bt**

Tables 29 to 36 contain the results of such comparisons. Each table describes the differences present within a single parameter, and provides three values for each product: the number of pixels which are different in the VAX/Alpha images; the mean difference between the VAX/Alpha images; and the standard deviation of the differences between the VAX/Alpha images. When using the tables, be aware that:

- The means and standard deviations of the image differences are provided in the same units as the image contents; that is, degrees/1000 for latitude and longitude images, Kelvin/100 for temperature images, and images.
- Each image contains $512 \times 512 = 262144$ pixels.

- Differences in latitudes and longitudes between VAX and Alpha images were no greater than one-thousandth of a degree; this is the smallest difference which can be recognised by the precision of the latitude/longitude values, but is still large enough to result in ATSR pixels being placed in different image pixels, potentially causing large difference in brightness temperatures between VAX and Alpha images. The spatial variability of temperature within an image determines the extent to which very small geolocation differences result in large temperature differences.

Product name	# pixels	Mean diff.	St. dev. of diffs
$401030950_16000.bt	345	-6.14e-4	0.0363
$401030950_20000.bt	0	0	0
$401030950_24000.bt	0	0	0
$401030950_28000.bt	547	-1.29e-3	0.0456
$401030950_32000.bt	265	7.25e-5	0.0320
$401030950_36000.bt	0	0	0
$401031130_00001.bt	0	0	0
$401031130_04000.bt	0	0	0
$401031130_08000.bt	0	0	0
$401031130_12000.bt	391	-4.96e-5	0.0386

Table 29: VAX/Alpha differences between latitudes in BT products

Product name	# pixels	Mean diff.	St. dev. of diffs
$401030950_16000.bt	2695	0.0102	0.101
$401030950_20000.bt	2518	9.61e-3	0.0975
$401030950_24000.bt	2598	9.91e-3	0.0990
$401030950_28000.bt	3139	0.0111	0.109
$401030950_32000.bt	562	2.03e-3	0.0462
$401030950_36000.bt	707	2.70e-3	0.0518
$401031130_00001.bt	8	2.29e-5	0.00552
$401031130_04000.bt	95	3.62e-4	0.0190
$401031130_08000.bt	339	1.29e-3	0.0359
$401031130_12000.bt	1512	-1.70e-3	0.0759

Table 30: VAX/Alpha differences between longitudes in BT products

Sea-surface temperature image product (SST) differences

Ten sea-surface temperature image products, evenly spaced around a single orbit, were generated by SADIST v600 on VAX and Alpha AXP systems, then compared. The products were:

***$401030950_16000*.sst**
***$401030950_20000*.sst**
***$401030950_24000*.sst**
***$401030950_28000*.sst**
***$401030950_32000*.sst**

Product name	**# pixels**	**Mean diff.**	**St. dev. of diffs**
$401030950_16000.bt	125	1.29e-3	1.14
$401030950_20000.bt	30	-2.30e-3	2.37
$401030950_24000.bt	52	-0.107	55.5
$401030950_28000.bt	197	5.48e-3	2.61
$401030950_32000.bt	55	7.67e-4	0.430
$401030950_36000.bt	43	-5.95e-4	2.05
$401031130_00001.bt	53	3.06e-3	2.86
$401031130_04000.bt	50	-3.51e-3	1.87
$401031130_08000.bt	65	1.19e-3	1.46
$401031130_12000.bt	142	1.48e-2	103.0

Table 31: VAX/Alpha differences between nadir-view 12.0μm temperatures in BT products

Product name	**# pixels**	**Mean diff.**	**St. dev. of diffs**
$401030950_16000.bt	109	5.84e-4	1.18
$401030950_20000.bt	31	-1.90e-3	2.56
$401030950_24000.bt	38	-0.108	55.8
$401030950_28000.bt	192	4.23e-3	2.48
$401030950_32000.bt	32	1.18e-3	0.559
$401030950_36000.bt	33	7.06e-4	2.07
$401031130_00001.bt	47	3.61e-3	2.72
$401031130_04000.bt	51	-3.18e-3	2.14
$401031130_08000.bt	67	3.18e-3	1.59
$401031130_12000.bt	132	4.98e-3	8.13

Table 32: VAX/Alpha differences between nadir-view 11.0μm temperatures in BT products

Product name	**# pixels**	**Mean diff.**	**St. dev. of diffs**
$401030950_16000.bt	105	-2.40e-3	3.35
$401030950_20000.bt	9	2.25e-4	0.837
$401030950_24000.bt	27	-4.28e-3	3.92
$401030950_28000.bt	167	-9.64e-3	6.11
$401030950_32000.bt	28	-1.44e-3	0.496
$401030950_36000.bt	17	-6.48e-5	0.0188
$401031130_00001.bt	31	0	0.0236
$401031130_04000.bt	38	-2.29e-5	0.0130
$401031130_08000.bt	40	-4.58e-5	0.0170
$401031130_12000.bt	93	-1.79e-4	0.0859

Table 33: VAX/Alpha differences between nadir-view 1.6μm reflectances in BT products

Product name	# pixels	Mean diff.	St. dev. of diffs
$401030950_16000.bt	n/a	n/a	n/a
$401030950_20000.bt	23	-8.42e-3	3.30
$401030950_24000.bt	27	-1.37e-4	0.191
$401030950_28000.bt	110	-1.56e-4	1.32
$401030950_32000.bt	59	-7.25e-5	0.642
$401030950_36000.bt	59	-4.16e-3	2.10
$401031130_00001.bt	66	3.71e-3	13.0
$401031130_04000.bt	61	8.53e-2	46.1
$401031130_08000.bt	53	3.85e-3	0.669
$401031130_12000.bt	102	-1.00	193.0

Table 34: VAX/Alpha differences between forward-view 12.0μm temperatures in BT products

Product name	# pixels	Mean diff.	St. dev. of diffs
$401030950_16000.bt	n/a	n/a	n/a
$401030950_20000.bt	15	-9.70e-3	3.25
$401030950_24000.bt	17	3.97e-4	0.198
$401030950_28000.bt	110	-2.40e-4	1.32
$401030950_32000.bt	50	2.21e-4	0.679
$401030950_36000.bt	44	-4.99e-3	1.83
$401031130_00001.bt	58	-5.04e-3	3.57
$401031130_04000.bt	47	8.53e-2	46.2
$401031130_08000.bt	49	2.55e-3	0.669
$401031130_12000.bt	112	-0.0129	5.55

Table 35: VAX/Alpha differences between forward-view 11.0μm temperatures in BT products

Product name	# pixels	Mean diff.	St. dev. of diffs
$401030950_16000.bt	n/a	n/a	n/a
$401030950_20000.bt	10	7.14e-3	1.90
$401030950_24000.bt	18	1.24e-3	2.95
$401030950_28000.bt	83	8.56e-3	4.22
$401030950_32000.bt	33	3.13e-4	0.167
$401030950_36000.bt	14	-5.72	0.0210
$401031130_00001.bt	17	-1.49e-4	0.0220
$401031130_04000.bt	40	3.43e-5	0.0235
$401031130_08000.bt	27	-1.14e-5	0.0147
$401031130_12000.bt	52	1.14e-5	0.133

Table 36: VAX/Alpha differences between forward-view 1.6μm reflectances in BT products

***$401030950_36000*.sst**
***$401031130_00001*.sst**
***$401031130_04000*.sst**
***$401031130_08000*.sst**
***$401031130_12000*.sst**

Table 37 contains the results of such comparisons. The table provides four values for each product: the number of pixels in the product which contain clear (uncloudy) sea; the number of clear sea pixels which are different in the VAX/Alpha images; the mean difference between clear sea pixels in the VAX/Alpha images; and the standard deviation of the differences between clear sea pixels in the VAX/Alpha images. When using the table, be aware that:

- The means and standard deviations of the sea-surface temperature differences are provided in the same units as the image contents; that is, Kelvin/100;
- Each image contains 512 × 512 = 262144 pixels.

Product name	**# clear sea**	**# diff**	**Mean diff.**	**St. dev. of diffs**
$401030950_16000.sst	0	n/a	n/a	n/a
$401030950_20000.sst	15021	76	-2.99e-3	0.0725
$401030950_24000.sst	69676	383	3.01e-3	0.558
$401030950_28000.sst	0	n/a	n/a	n/a
$401030950_32000.sst	5435	31	0.0350	1.12
$401030950_36000.sst	96248	720	6.96e-4	1.38
$401031130_00001.sst	100985	817	-4.75e-4	4.20
$401031130_04000.sst	371	2	0	0.0735
$401031130_08000.sst	388	5	0.0180	2.45
$401031130_12000.sst	16059	132	-4.30e-3	4.48

Table 37: VAX/Alpha differences between sea-surface temperatures in SST products

Spatially-averaged sea-surface temperature product (ASST) differences

Three spatially-averaged sea-surface temperature products were generated by SADIST v600 on VAX and Alpha AXP systems, then compared. The products were:

***$401030950_15658*.asst**
***$401031130_15019*.asst**
***$401031310_13811*.asst**

Table 38 contains the results of such comparisons. The table provides four values for each product: the number of cells in the product which contain clear (uncloudy) sea; the number of clear sea cells which are different in the VAX/Alpha images; the mean difference between clear sea cells in the VAX/Alpha images; and the standard deviation of the differences between clear sea cells in the VAX/Alpha images. When using the table, be aware that:

- The means and standard deviations of the sea-surface temperature differences are provided in the same units as the product contents; that is, Kelvin/100;

- The comparisons used only those cells containing values in both VAX and Alpha products. The values parenthesised after the number of cells shows the number of cells containing a sea-surface temperature value in one product, but not the other; these cells were not included in the derivation of means and standard deviations.

Product name	**# cells**	**# diff**	**Mean diff.**	**St. dev. of diffs**
$401030950_15658.asst (nadir)	2218 (12)	19	-2.71e-3	0.112
(mixed)	2218 (12)	112	-0.112	2.59
(dual)	2218 (12)	99	2.99e-2	1.29
$401031130_15019.asst (nadir)	2043 (44)	16	-4.89e-4	0.0965
(mixed)	2043 (44)	207	-0.449	6.14
(dual)	2043 (44)	166	-0.0410	2.96
$401031310_13811.asst (nadir)	862 (9)	9	-0.113	3.21
(mixed)	862 (9)	48	-0.328	3.27
(dual)	862 (9)	35	-0.0211	1.16

Table 38: VAX/Alpha differences between spatially-averaged sea-surface temperatures in ASST products

D Changes to this document

Following is a description of changes made to this document since the last release (*SADIST Products (Version 500)*, 24 May 1993).

- Errors in the sample code supplied with the definitions of the brightness temperature image product and the sea-surface temperature image product have been corrected. The variables **node_x_velocity**, **node_y_velocity** and **node_z_velocity**, within the primary headers declared by both the C and IDL code, were previously shown as 12 bytes long. The correct length is 10 bytes, as shown within tables 6 and 8.
- Within the descriptions of pixel geolocation & x/y offsets supplied with brightness temperature and sea-surface temperature image products, the positions of pixel x-offsets and y-offsets have been corrected. Previously, this document stated that

  > Each one-byte pixel x/y offset contains an x-offset (that is, an offset in the across-track direction) within the most significant four bits, and a y-offset (that is, an offset in the along-track direction) within the least significant four bits.

  This was incorrect; the positions of the offsets are reversed. The correct description, which appears in sections 3.4.1 and 4.4.1 of this document, is:

  > Each one-byte pixel x/y offset contains an x-offset (that is, an offset in the across-track direction) within the least significant four bits, and a y-offset (that is, an offset in the along-track direction) within the most significant four bits.

- Description of the new spatially-averaged cloud temperature/coverage product has been added, with appropriate references within the introduction and appendices.
- Description of the redesigned brightness temperature browse image product has been added, with appropriate references within the introduction and appendices.
- Description of the redesigned ATSR instrument engineering product has been added, with appropriate references within the introduction and appendices.
- Description of the redesigned ATSR/M microwave instrument raw data product has been added, with appropriate references within the introduction and appendices.
- The file unit used within the sample C code for the brightness temperature image product was inconsistent; it is now *bt_file* in all cases.
- The use of variables of type **byte** in sample C code has been discontinued. This is a VAX C extension. The standard variable type **unsigned char** is now used.
- References to documents describing the ERS-1 ATSR/M microwave instrument telemetry, and the ATSR S-band telemetry, have been added.
- Description of the use of the "sign bit" of 12.0μm and 11.0μm detector count and brightness temperature values to store blanking pulses and cosmetic fill flags was extremely sloppy, and has been removed. The correct description is that such values are *negated*. Negation is not the same as setting the sign bit, unless integers are represented using a sign-magnitude standard. VAXes, and Alphas, and Suns, and most other machines, use twos-compliment integer representation, so it's important that the description is clear.
- A new appendix quantifies the residual differences between products from SADIST v600 for VAX/VMS and SADIST v600 for Alpha AXP.

E Glossary and abbreviations

Along-track distance	Distance to a point on the ground-track from the previous ascending node crossing (cf. relative scan number)
Ascending node	Point of northward Equator crossing by satellite (cf. descending node)
Black-body	An idealised body which absorbs all radiation incident upon it, with no reflection, and which emits with perfect efficiency
Brightness temperature	Temperature of perfectly-emitting black-body which would produce equivalent emitted radiation at a particular wavelength
Cloud-identification	Determination of cloud-content of ATSR data using threshold/uniformity tests applied to combinations of available brightness temperatures
Collocation	Spatial matching of data within ATSR forward and nadir views
Confidence word	Collection of summaries of processing results, which comprise an indication of the confidence with which such results may be used
Count	Result of analogue-to-digital conversion of the signal from an ATSR detector
Descending node	Point of southward Equator crossing by satellite (cf. ascending node)
Geolocation	Determination of the Earth location of acquired ATSR data
Ground-track	The locus of the sub-satellite point over a period of time
High-resolution	In the context of ATSR imagery, product contents which exist at the full resolving power of the instrument. (more specifically, for ATSR, 1 km)
Housekeeping	Those sections of ATSR raw data which contain information concerning the status of the satellite/instrument hardware
Land-flagging	Determination of the surface characteristic (land or sea) of a point on the Earth's surface, by reference to a database of such information
Preprocessor	The user interface, data preprocessor and task scheduler within the SADIST system
Reflectance	The extent to which incident radiation is reflected (rather than absorbed, to be emitted later as heat) at a surface
Relative scan number	Number of an ATSR scan relative to that which occurred at

	the previous ascending node crossing (cf. along-track distance)
Repeat-cycle	Uninterrupted sequence of orbits at the beginning and the end of which, the position of a satellite is (within appropriate tolerances) the same. The ERS-1 mission includes 3-day, 35-day, and 168-day repeat-cycles
Sea-surface temperature	Approximation to the actual temperature at the sea-surface derived using a combination of brightness temperatures at the wavelengths and viewing angles provided by ATSR
Slave	A batch process to which actual ATSR data-processing may be delegated by a SADIST preprocessor
Source packet	Encoded ATSR scan, as telemetered to an ERS-1 ground-station
Spatially-averaged	In the context of ATSR imagery, product contents which have been derived via combination of a number of adjacent input values (more specifically, for ATSR, 30 arcminutes)
State vector	Precise position and velocity of ERS-1 in three dimensions at specified time (commonly ascending node crossing)
Sub-satellite point	Point on the Earth's surface at which the local normal intersects the satellite
Transcribed products	SADIST products which are the result of filtering of appropriate data by SADIST, rather than processing of data
Universal time	A measure of time that conforms, within a close approximation, to the mean diurnal motion of the sun, and which serves as the basis of all civil timekeeping. For the purposes of this document, universal time may be considered to be equivalent to Greenwich time

ASCII	American Standard Code for Information Interchange
ASST	Average Sea-Surface Temperature
ATSR	Along-Track Scanning Radiometer
ATSR/M	ATSR Microwave instrument
BPI	Bits Per Inch
BT	Brightness Temperature
CCT	Computer Compatible Tape

CLI	Command-Line Interpreter
CPU	Central Processing Unit
DCL	Digital Command Language
DEC	Digital Equipment Corporation
EECF	Earthnet ERS-1 Central Facility (at ESRIN)
ERS-1	First European Remote-sensing Satellite
ERSORB	ERS-1 ORBit propagator software
ESA	European Space Agency
ESRIN	European Space Research INstitute (Frascati, Italy)
IR	Infra-Red
LRDTF	Low-Rate Data Transcription Facility (Fucino ground-station, Italy)
PRT	Platinum Resistance Thermometer
RAL	Rutherford Appleton Laboratory
RATSR	ATSR raw data
SADIST	Synthesis of ATSR Data Into Sea-surface Temperatures
SST	Sea-Surface Temperature
UK-PAF	United Kingdom ERS-1 Processing and Archiving Facility (Farnborough, UK)
UT	Universal Time
VAX/VMS	Virtual Address eXtension/Virtual Memory System