SG/6.3

Starlink Project
Starlink Guide 6.3

R.F. Warren-Smith

4th February 1994

ADAM
Programmer's
Facilities & Documentation
Guide

Abstract

This document contains a directory of the facilities available to the ADAM programmer. All documented aspects of ADAM programming are covered, ranging from introductory documentation, through applications programming to real-time and system programming. Each facility is briefly described along with the type of work for which it might be used. Most importantly, this guide shows at a glance where to obtain the relevant documentation.

It is hoped to update this document relatively frequently, so that it serves as a guide to the latest ADAM developments. Its format, which includes an overall index, is designed to allow its use as a table of contents for a personal file of programming documentation.

Contents

Documentation Index

1 Introductory Documentation

1-1	INTRO	*Introduction to ADAM Programming*	SUN/101
1-2*	ADAM	*The Starlink Software Environment*	SG/4

2 General Applications Programming

2-1*	PAR	*ADAM Parameter System Routines*	SUN/114
2-2	NDF	*Extensible N-Dimensional Data Format*	SUN/33
2-3*	MSG & ERR	*Message and Error Reporting*	SUN/104
2-4	CHR	*Character Handling*	SUN/40
2-5*	PGPLOT	*Graphics Library*	SUN/15
2-6	SGS	*Simple Graphics System*	SUN/85
2-7*	FIO & RIO	*File Input/Output*	SUN/143
2-8	MAG	*Magnetic Tape Handling*	SUN/171
2-9	PRIMDAT	*Primitive Numerical Data Processing*	SUN/39
2-10	ICL	*ADAM Command Language*	SG/5
2-11	IFL	*ADAM Interface Module Reference Manual*	SUN/115

3 More-specialised Facilities

3-1	SLALIB	*Positional Astronomy and Time*	SUN/67
3-2	NCAR	*Graphics Utilities*	SUN/88
3-3	SNX	*Starlink Extensions to NCAR*	SUN/90
3-4*	AGI	*Applications Graphics Interface*	SUN/48
3-5	PSX	*POSIX Interface Routines*	SUN/121
3-6*	REF	*References to HDS Objects*	SUN/31
3-7	TRANSFORM	*Coordinate Transformation Facility*	SUN/61
3-8*	HELP	*Help Text Retrieval System*	SUN/124

[1] The documentation for these facilities has changed since the previous version of this guide (dated 7th August 1992). Please ensure that you have a copy of the most recent documents.

[2] New facility.

1 Introductory Documentation

1-1 **INTRO** **SUN/101.2**
Introduction to ADAM Programming

A tutorial introduction to writing ADAM programs. This document is intended for newcomers to ADAM who already have a basic understanding of Fortran programming but are unfamiliar with ADAM conventions. The use of the most important facilities listed in this guide is illustrated, and a set of example programs is provided.

1-2 **ADAM** **SG/4.2**
The Starlink Software Environment

This document provides the most comprehensive overview of ADAM from the perspective of a typical Starlink user, covering both the running of existing applications and the writing of new ones. It contains essential introductory information for those new to ADAM, as well as reference material for the more experienced. Since it is a large document, it cannot be updated very frequently, so reference should be made to the other more specialised documents in this guide if up-to-the-minute information is needed on a particular topic.

2 General Applications Programming

2-1 **PAR** **SUN/114.1**
ADAM Parameter System Routines

Provides access to "*parameters*" from within applications. These are the values which provide external control over an application (broadly similar to command line switches in UNIX, or command qualifiers in VMS). The PAR library allows these values to be obtained from the command line, by prompting the user, or by a variety of other mechanisms. **The latest version of PAR includes a complete revision of the documentation and the addition of a new set of routines which permit parameter values to be obtained subject to various constraints.**

2-2 **NDF** **SUN/33.3**
Extensible N-Dimensional Data Format

The Extensible N-Dimensional Data Format (NDF) is the Starlink format for storing bulk data in the form of *N*-dimensional arrays of numbers. It is typically used for storing spectra, images and similar datasets with higher dimensionality. The NDF format is based on the *Hierarchical Data System*, HDS (see SUN/92), and is extensible; not only does it provide a comprehensive set of standard ancillary items to describe the data, it can also be extended indefinitely to handle additional user-defined information of any type. SUN/33 describes the routines provided for accessing NDF data objects. It also discusses all the important NDF concepts and includes a selection of simple example applications.

2-3 **MSG & ERR** **SUN/104.5**
Message and Error Reporting

These provide a means for sending simple messages to the user of an application, together with a mechanism for reporting errors. They address many of the common problems encountered when formatting and displaying text through their use of "*message tokens*", a concept which is also exploited by a number of other ADAM libraries. Most importantly, SUN/104 explains the ideas underlying the error handling scheme and "*inherited status checking*" used by the bulk of Starlink software.

2-4 **CHR** **SUN/40.3**
Character Handling

This library provides a simple way to perform many commonly required operations on Fortran character variables. These include: formatting and decoding numerical values, case conversion, analysing lists, string searching and character testing. This library is currently being upgraded to include a substantial number of new routines.

2-5 **PGPLOT** ***PGPLOT Manual***
Graphics Library **SUN/15.5**
SUN113.2

If you are unsure which high-level graphics library to use, then this is probably the one for you. It caters for most simple graph-plotting needs in a straightforward way (although it may not satisfy the more demanding user, for whom NCAR may be preferable – see SUN/88). The PGPLOT manual, entitled "*PGPLOT – Graphics Subroutine Library*", is available from Starlink site managers in the *Miscellaneous User Documents* (MUD) series. SUN/15 describes the PGPLOT implementation on Starlink and shows how it integrates with SGS (see SUN/85). Note that the *Applications Graphics Interface* AGI also provides special routines for ease of use with PGPLOT (see SUN/48).

2-6 SGS **SUN/85.5**

Simple Graphics System **SUN/113.2**

SGS provides an easy-to-use interface to the *Graphical Kernel System*, GKS (see SUN/83). GKS is the underlying system which provides graphics "*device-independence*" but it is too low-level for general use, so calls to SGS routines may be used instead to perform simple plotting operations which do not amount to complete graph drawing. SGS is particularly useful for managing the layout of plotting surfaces which are to be filled with graphs drawn by higher-level routines. SUN/85 describes the "*stand-alone*" version of SGS, while SUN/113 describes additional routines which connect SGS to ADAM. The *Applications Graphics Interface* AGI also provides special routines for ease of use with SGS (see SUN/48).

2-7 FIO & RIO **SUN/143**

File Input/Output

Provides a set of routines to facilitate the reading and writing of Fortran files. These routines are not intended to replace the portable features of standard Fortran 77, but to help integrate these features with ADAM. They also hide some of the unavoidable, but non-portable, features of Fortran input/output. **The latest version of FIO & RIO includes a complete revision of the documentation and new facilities to assist with error reporting and the portable classification of I/O error status values. Some rationalisation of routine naming has also occurred.**

2-8 MAG **SUN/171**

Magnetic Tape Handling

Provides routines for accessing magnetic tapes and controlling tape drive operations *via* the ADAM parameter system. A feature is incorporated to minimise tape movement between applications by keeping track of the absolute tape position.

2-9 PRIMDAT **SUN/39.1**

Primitive Numerical Data Processing

Provides arithmetic and mathematical functions for processing primitive numerical data (real, integer & double precision values, *etc.*) including the non-standard data types (*e.g.* unsigned byte) which are used in ADAM but are not directly supported by Fortran 77. Routines are provided for both scalar values and arrays, with optional recognition of "*undefined*" data and methods for handling numerical errors such as overflow. A set of include files is also provided to define machine-dependent constants, including the important "*bad*" values used to flag undefined data. The systematic naming of files and routines in this library is designed to integrate with the GENERIC pre-processor (see SUN/7).

2-10 **ICL** **SG/5.1**
ADAM Command Language

Although ICL is really a command language from which applications are run, the writer of ADAM applications will still find it useful to refer to this document. It describes the ICL language, which includes a number of special features for controlling and communicating with ADAM tasks.

2-11 **IFL** **SUN/115.1**
ADAM Interface Module Reference Manual

An "*interface module*" (or "*interface file*") acts as an interface between an application and its user. It describes to the ADAM parameter system (see SUN/114) how to obtain values and other information (*e.g.* help text) for an application's parameters, and hence influences, in broad terms, how the application behaves. This document provides reference information describing in detail how to write an interface module.

3 More-specialised Facilities

3-1 SLALIB SUN/67.17

Positional Astronomy and Time

SLALIB (the name just stands for "*Subprogram Library A*") contains a large number of routines mainly concerned with positional astronomy and time. Some also have wider trigonometrical, numerical or general significance, while others are essentially miscellaneous. Its facilities include: string decoding; sexagesimal conversions; handling of angles, vectors and rotation matrices; calendar and timescale calculations; precession, nutation and proper-motion calculations; celestial coordinate conversions and astrometric transformations.

3-2 NCAR NCAR Manual SUN/88.4

Graphics Utilities

NCAR is a large and sophisticated set of high-level plotting routines developed at the National Centre for Atmospheric Research at Boulder, Colorado. It can do more things than most other plotting packages and is reasonably straightforward to use. The "*NCAR Manual*" is a large document in the *Miscellaneous User Documents* (MUD) series, available from Starlink site managers. SUN/88 provides a concise summary of the NCAR facilities and describes the Starlink release of this system.

3-3 SNX SUN/90.8

Starlink Extensions to NCAR

Since NCAR uses GKS, calls to its routines may be interspersed with calls to other GKS-based graphics libraries. To do this effectively, however, it is usually necessary to know about such things as the NCAR coordinate system, or to arrange for an NCAR plot to appear within a specified region of the display surface. The SNX routines provide a bridge between NCAR and SGS which allows this type of interaction.

3-4 **AGI** **SUN/48.5**

Applications Graphics Interface

AGI is a Graphics Database system which allows graphics applications to record information about the plots they produce so that subsequent applications can make use of it. For example, information about the coordinate system of a graph might be recorded so that a subsequent application could use a cursor to read off positions. AGI allows a considerable amount of information to be stored, ranging from simple comments to non-linear coordinate transformations and references to HDS objects. It also allows graphics devices to be divided into separate "*pictures*" so that multiple plots may appear, each with their own graphics information. AGI has special interfaces for ease of use with the PGPLOT, SGS and IDI graphics libraries.

3-5 **PSX** **SUN/121.1**

POSIX Interface Routines

POSIX is an IEEE standard defining an interface to a "*virtual operating system*" which applications may use to interact with their host machine in a portable manner. For example, POSIX could be used to determine the current "*user-name*" in a manner which does not vary between machines. Unfortunately, the POSIX standard currently only provides for calls from programs written in C (the Fortran interface definition is not yet complete), so the PSX library provides a solution by making some of the more useful POSIX functions available in Fortran-callable form. It provides an ADAM-style interface to POSIX, including inherited status checking and error reporting.

3-6 **REF** **SUN/31.4**

References to HDS Objects

It is sometimes useful to use HDS to store pointers to other HDS objects. For instance, the *Applications Graphics Interface*, AGI, depends on this ability in order to associate HDS data objects with graphical displays (see SUN/48), without having to make a separate copy of the data object. The REF library is provided to facilitate this data-object "*referencing*" process and the subsequent accessing of objects which have been referenced in this way.

3-7 **TRANSFORM** **SUN/61.2**

Coordinate Transformation Facility

A specialised facility which allows mappings (or coordinate transformations) to be formulated and stored as HDS objects. These transformations may later be recovered and evaluated. This facility is used by the *Applications Graphics Interface*, AGI, to implement world-to-data coordinate transformations (see SUN/48). TRANSFORM also incorporates a facility for parsing and evaluating Fortran-like arithmetic expressions, a feature which can sometimes be used to good effect in applications software.

3-8 **HELP** **SUN/124.6**
Help Text Retrieval System

A subroutine library designed for implementing interactive hierarchical help systems. It is modelled on the VAX/VMS help system but has a number of additional features, including the ability for "*help files*" to refer to other help files. This library will mainly be of interest to those writing ADAM system software, but this does not preclude its use in applications which wish to handle their own help information explicitly.

4 Lower-level Facilities

4-1 **ARY** **SUN/11.2**
Access to Array Data Structures

The HDS-based ARRAY data structure is one of the building blocks from which the Extensible N-Dimensional Data Format, or NDF (see SUN/33), is constructed. The ARY library provides access to these structures and is used extensively by the NDF access routines. ARRAY structures can also be useful in their own right as a more sophisticated alternative to primitive HDS arrays for storing data. They can be particularly valuable when used in NDF "*extensions*" to extend the NDF definition by adding arrays of additional information.

4-2 **HDS** **SUN/92.8**
Hierarchical Data System **SUN/224.2**

HDS is the low-level data system upon which other important data structures, such as the NDF (see SUN/33), are built. This document describes routines for accessing these structures at the lowest level. Calls to HDS routines may be intermixed with those to other higher-level data access routines, and are also invaluable for creating and accessing data structures of your own design (such as NDF extensions). All data structures created by HDS are portable between the machines on which HDS is implemented. SUN/92 describes the "*stand-alone*" version of HDS, while SUN/227 describes additional routines which connect HDS to ADAM.

4-3 **IDI** ***Astron. Astrophys. paper***
Image Display Interface **SUN/65.7**

IDI is a standard for the display and manipulation of data on image displays in astronomy. Unlike the other graphics packages, IDI is not based on GKS, so calls to IDI cannot be intermixed with other graphics calls. However, IDI has the advantage of providing facilities for interactive manipulation of image displays (*e.g.* panning, zooming and blinking) which GKS lacks. The definitive document on IDI is "*An image display interface for astronomical image processing*" by Terrett *et al.* (1988), Astron. Astrophys. Suppl. Ser., **76,** 263-304. This paper describes the IDI routines in detail but without reference to a specific implementation. SUN/65 describes the Starlink implementation of IDI, which includes some additional routines to connect it with ADAM. The *Applications Graphics Interface* AGI also contains special routines for ease of use with IDI (see SUN/48).

4-4 **GKS** — *RAL GKS Manual*, **SUN/83.12**, **SUN/113.2**

Graphical Kernel System

This is the low-level graphics system which underlies all the other ADAM graphics libraries (except IDI) and which provides "*device independence*", allowing these libraries to access a large range of graphics devices. GKS is an international standard, but its concepts are rather abstract and inconveniently low-level for most simple graphical work. SGS (see SUN/85) provides a simpler interface to GKS, but direct GKS calls are often useful and can, with care, be intermixed with calls to other graphics routines. The primary document on GKS is the "*RAL GKS Manual*", available in the *Miscellaneous User Documents* (MUD) series from Starlink site managers. In addition, SUN/83 gives details of the Starlink GKS release and hardware implementations, while SUN/113 describes additional routines which connect GKS (and SGS) with ADAM.

4-5 **GRP** — **SUN/150.3**

Routines for Managing Groups of Objects

GRP provides facilities for managing groups of character strings which are the "names" of objects such as data files, galaxies, or even sets of coordinates. A typical use might be within a package of co-operating applications which are to process multiple datasets specified by a list of file names. GRP allows such applications to access these names very flexibly – by accepting explicitly typed values, by reading them from a file (indirection) or by editing an existing list (modification). It also supports the passing of groups of objects between separate applications. GRP is a configurable low-level facility and it is expected that higher-level interfaces specialised to particular types of object will be provided in future.

4-6 **GWM** — **SUN/130.2**

X Graphics Window Manager

GWM provides facilities for creating and controlling "*persistent*" graphics windows in an X windows environment. The windows it creates are intended for use as "*virtual displays*" for GKS- and IDI-based graphics; unlike other X windows, they do not disappear when an application terminates, so the plot remains visible and may be accessed by subsequent applications. GWM allows control over graphics windows both from the command line and *via* a Fortran programming interface, so that applications may create and destroy graphics windows for use both by themselves and by other applications.

4-7 **GNS** **SUN/57.7**
Graphics Workstation Name Service

An important feature of ADAM graphics is its "*device independence*", which means that graphical applications can generally be used with a wide range of graphics devices. GNS provides the services needed for managing the names which refer to these devices. It allows a local database of easy-to-remember names to be set up, and then provides applications with the information they require about each named graphics device. GNS is used by all the high-level graphics libraries, so direct calls to GNS routines will seldom be necessary.

5 Programming Tools

5-1 **STARLSE** **SUN/105.5**
Starlink Language Sensitive Editor

STARLSE is based on the VAX Language Sensitive Editor LSE. It exploits the extensibility of LSE to provide a Language Sensitive Editor with special facilities for the ADAM programmer. These include definitions of the calling sequences for most of the libraries which appear in this guide, together with on-line help information. Standard prologue templates are also provided, with information being filled in automatically by the editor in many cases. These prologues may be processed into various forms of documentation using applications from the SST package (see SUN/110).

5-2 **SST** **SUN/110.1**
Simple Software Tools

Simple tools for performing common operations on software, implemented as a package of ADAM applications. The most important feature of this package is the ability to process prologue information formatted using STARLSE (see SUN/105) into documentation in the form of LaTeX documents, help libraries, or LSE subroutine-library definitions. The resulting subroutine-library definitions may then be used with STARLSE.

5-3 **GENERIC** **SUN/7.4**
Utility for Compiling Generic Fortran

GENERIC is a utility which pre-processes "*generic*" Fortran code into multiple versions which are capable of processing different data types (all the primitive numeric types are supported together with character and logical values). It works by detecting special tokens embedded in the input file. GENERIC integrates with the PRIMDAT package (see SUN/39) which provides consistently named routines for performing operations on primitive data.

5-4 **SPT** **SUN/111.2**
Software Porting Tools

A package of tools to assist when moving software between different hardware platforms. At present there are two SPT tools available; a utility for moving Fortran 77 software between VAX/VMS and UNIX systems (it performs those unavoidable conversions which are needed in this situation, including the translation of include file names) and a utility for defining "soft links" to standard Starlink include files on UNIX systems.

6 Instrumentation Programming

6-1 **TASK** **SUN/134.2**
Guide to Writing ADAM Instrumentation Tasks

An introductory guide to writing ADAM "*real-time*" instrumentation tasks. This document includes a description of the ADAM tasking model and develops real-time concepts through a series of examples. It also describes the library of ADAM TASK routines.

6-2 **NBS** **SUN/77.2**
Noticeboard System

NBS provides a mechanism for the rapid transfer of large quantities of data between processes running on the same processor. It utilises shared memory in which a hierarchical data structure (a "*noticeboard*") may be built and subsequently written and read by co-operating processes. Noticeboards are typically used in data-acquisition environments, but this does not preclude their use for other purposes. The data-structuring facilities of NBS are broadly similar to those of HDS (see SUN/92), although noticeboards have a static structure and are volatile.

6-3 **MSP** **SSN/2.1**

Message System Primitive Routines

These routines provide the low-level message-passing services upon which inter-task communication within ADAM depends. They will normally only be of use to ADAM system programmers or those developing real-time ADAM software.

7 System Programming

7-1 **EMS** **SSN/4.4**

Error Message Service

EMS is a set of low-level routines which implement the facilities required by the ERR and MSG routines (see SUN/104). This separate interface to the lower levels is provided for those writing system software which does not require the ability to interact with a user, but simply needs to report and manage error messages. In particular, EMS should be used in preference to ERR/MSG by those developing ADAM system software residing in, or beneath, the parameter system.

7-2 **CNF & F77** **SGP/5.1**

Mixed Language Programming

Provides advice, routines and macros to help you pass information between code written in Fortran 77 and C (in both directions) in a machine-independent way.

8 Standards and Conventions

8-1 **FORTRAN** **SGP/16.10**
Application Programming Standard

Describes the Starlink Fortran programming standard, which exists to promote portability and maintainability in Fortran 77 software. It specifies use of the Fortran 77 language standard plus a small number of approved extensions. It also contains a wealth of advice on good programming style and how to avoid common pitfalls.

8-2 **C** **SGP/4.1**
Starlink C Programming Standard

Written in a similar style to the Fortran standard (SGP/16), this programming standard for the C language also exists to promote portability and maintainability. Although C is not recommended as an applications language, it is being used to an increasing extent for real-time and system software. C has a far greater potential for danger than Fortran, especially given the variety of compilers available. This document guides you through the C minefield to help you produce good software which will run on a wide range of computer systems.

8-3 **DATA** **SGP/38.2**
Starlink Standard Data Structures

An extensive document describing the philosophy behind the use of HDS for building general-purpose data structures. It also gives a full (and gory) description of the *Extensible N-Dimensional Data Format*, or NDF. The detail in this document is only likely to be of interest to those developing related data-access software. However, the general sections covering the use and design of data structures would be valuable reading for anyone planning extensive use of HDS.

8-4 **PACKAGES** **SSN/64.2**
Organisation of ADAM Applications Packages

Describes how separate ADAM applications packages are organised for release. It covers such things as the provision of standard "*startup*" files, the structure of help libraries, introductory help information, *etc.* This is essential reading for anyone planning to release a new package of ADAM applications *via* Starlink.

8-5 **LIBRARIES** **SSN/8.1**
Conventions for Accessing Starlink Libraries (VMS)

This document will mainly be of interest to those developing new subroutine libraries. It covers the facilities which a library should provide, the use of shareable libraries on VMS, and the naming scheme through which callers of the library should be able to access it. This document also describes the distinction between the ADAM and "*stand-alone*" versions of a library.

8-6 **UNIX** **SSN/66.2**
Starlink Software Organisation on UNIX

Describes how Starlink software is distributed and installed on UNIX machines. This also covers access to subroutine libraries and their "*include files*" on UNIX systems. The final part of the document covers "*make files*" and the methods used for the building of UNIX executables.