Installation of MIDAS on UNIX Systems

Data Management Division,
European Southern Observatory,
Karl-Schwarzschild-Straße 2,
D-85748, Garching, FRG.

July 1998

Abstract

The Munich Image Data Analysis System, MIDAS, developed and maintained by ESO, provides general tools for image processing and data reductions with emphasis on astronomical applications. MIDAS is at present distributed once a year.

The official name, ESO-MIDAS, is a registrated trademark. ESO-MIDAS is available under the GNU General Public License (GPL), and can be implemented on VMS, OpenVMS and UNIX systems.

This document describes the installation procedure for MIDAS on UNIX systems.

Chapter 1

Introduction

The Munich Image Data Analysis System, MIDAS, is developed and maintained by the European Southern Observatory. Its main goal is to provide the astronomical community with software tools for both general image processing and analysis, and for the reduction of data acquired at ESO telescopes. Although originally developed on a VAX/VMS system, since 1988 MIDAS is available for systems with a VAX/VMS operating systems, or with one of the main flavors of the UNIX systems.

For UNIX systems MIDAS is distributed by ESO on the following media:

```
- 8mm Exabyte tape, TAR format.
- 4mm Cartridge tape, DDS/DAT, TAR format.
- CD-ROM ISO9660 with RockRidge extensions.  Compressed-TAR format.
- anonymous ftp from ftphost.hq.eso.org, Compressed-TAR format.
```

This document describes the installation of MIDAS on computer systems running under the UNIX operating system. It contains all information needed to ensure a smooth installation of the MIDAS software. In Chapter 2 the actual installation of MIDAS is described, as well as how MIDAS should be set up properly. In addition this chapter describes a few items related with file initializations. In Chapter 3, the installation procedures of graphic devices can be found. Chapter 4 gives an overview of how to install, modify and test local tape drives. Chapter 5 discusses the optional installation of the tapeserver. Chapter 6 describes other optional stand-alone utilities. Chapter 7 contains the address of the MIDAS Hot-Line service in case of problems.

Throughout this document a number of typographic conventions are used. These conventions are given below.

- *Italic type style* is used to emphasize a term or statement in the body of a paragraph:
- `Typewriter type style` is used:

  - to show, within the body of the text, command lines or options that should be typed verbatim on the screen;
  - to show, within examples, computer-generated input/output or the contents of files.

- **Bold type style** only used to emphasize a very important note or remark for the installation of MIDAS.

In addition the following symbols are used:

- # for the standard prompt when executing as superuser.
- % for the standard prompt when executing as a normal user.

Chapter 2

Main Installation

2.1 Create a directory for MIDAS

First create a directory for MIDAS into which you will read the distribution tape (usually named something like `/midas`, which also can be referred as the **MIDASHOME** directory). You will need roughly 30 Mbytes to hold sources of MIDAS (no documentation) and, depending on your selection of packages, up to 130 Mbytes for the installed version on SunOS 4.1.3.

2.2 Retrieve MIDAS sources from the distribution tape

The tape contains several UNIX tar files. As indicated in the label of the tape, the first tar file contains the sources of MIDAS. To unload them, execute:

```
% cd /midas
% tar xvf tapedevice
```

All unloaded files will come under the release directory `98NOV`, also referred to as **MIDVERS**.

2.3 Retrieve MIDAS sources from the *midas* ftp account

To retrieve MIDAS sources:

```
% cd /midas
% ftp ftphost.hq.eso.org      (or 134.171.6.9)
Name:  ftp
```

```
Password: ¡your_email_address¿
ftp> cd /midaspub/98NOV/sources
ftp> get README.sources
ftp> binary                          (necessary for binary files)
ftp> get 98NOV.tar.Z
ftp> get  any_other_file
ftp> quit
```

To decompress and un-tar the MIDAS sources:

```
% cd /midas
% zcat 98NOV.tar.Z | tar xvf -
```
(and after a backup of 98NOV.tar.Z)
```
% rm -f 98NOV.tar.Z
```

All unloaded files will come under the release directory `98NOV`, also referred to as **MIDVERS**[1].

User support is offered to users affiliated with registered ESO-MIDAS sites. For site registration please use the ESO-MIDAS Registration Form in *http://http.hq.eso.org/midas-info/midas-registerform.html*. We can provide help for problems, questions during installation or usage of the MIDAS system. If answers to questions cannot be obtained locally (e.g. through the manuals) or after consulting our FAQ, please feel free to contact us for support.

2.4 The Config Script

Located in the directory `/midas/98NOV/install/unix` is the front-end of the installation of MIDAS to the operator. It offers a menu of other options that are executed when selected by the option number. To execute the `config` script type:

```
% cd /midas/98NOV/install/unix
% sh config
```

The `config` script will generate the following output on the screen:

```
MIDAS CONFIG MENU:
==================
1 - list of MIDAS packages available
```

[1]If you do not use the standard MIDASHOME (`/midas`) and MIDVERS (`98NOV`) settings, you have to update the file /midas/98NOV/local/default.mk, because MIDVERS and MIDASHOME are hard coded in there.

```
 2 - select only MIDAS core
 3 - select all MIDAS packages
 4 - select MIDAS core and packages of your choice
 5 - preinstall MIDAS
 6 - install MIDAS
 7 - update MIDAS
 8 - setup MIDAS
 9 - clean MIDAS
10 - help
 q - quit

 Select: (option number or q to quit)
```

2.5 Preinstallation of MIDAS

From the `config` menu, select the option `5 - preinstall MIDAS`. If your system matches one of the system names located in the directory `/midas/98NOV/install/unix/systems`, the procedure will use it automatically to complete the preinstallation of MIDAS, otherwise it will show you a list of system names for a selection. If your machine is not included in the list type the option `unknown`.

NOTE: For some systems, the preinstallation will print out, on your screen, important notes for the installation of MIDAS on your system. Please, read these notes carefully because they could indicate a specific environment that you should check before you continue with the installation of MIDAS. The file containing these notes is located in `/midas/98NOV/local/README`.

2.6 Selection of optional packages

MIDAS sources are divided in the *core* of MIDAS (indispensable and therefore always included and installed) and several *optional* packages, grouped in three categories: applications *(applic)*, standard reduction packages *(stdred)* and contribution packages *(contrib)*. The selection is done in the `config` script using the following options:

```
MIDAS CONFIG MENU:
==================
1 - list of MIDAS packages available
2 - select only MIDAS core
3 - select all MIDAS packages
4 - select MIDAS core and packages of your choice
```

The first option gives you the complete list of optional packages available in the MIDAS release and an indication whether they have been selected. A brief description of each package is displayed when the name of the package is entered.

By default, and without selecting any of the above options, only the *core* of MIDAS will be installed.

2.7 Installation of MIDAS

If you are satisfied with the selection of packages, you can start with the installation of MIDAS.

From the `config` menu, select option `6 - install MIDAS`. The procedure will also check for the existence of the NAG mathematical library, the X11 library and the Motif library. For the NAG library, the procedure will create a dummy NAG library if you do not have one. Without the X11 library the installation will give some errors in `/midas/98NOV/system/idiserv/src`. MIDAS will be installed anyway but without graphic and display capabilities. And without the Motif library MIDAS cannot be installed with Graphical User Interfaces (GUIs).

The installation of MIDAS runs in the background. A log file is written in the temporary directory `/midas/98NOV/tmp` which can be checked for errors when finished. If you want to trace the output of the installation try the following command:

```
% tail -f /midas/98NOV/tmp/install.here_is_a_number
```

2.8 Update of MIDAS

If, for whatever reason, the installation of MIDAS is stopped, and you want to finish it, you can continue from the point where it was stopped by selecting the option `7 - update MIDAS`, from the `config` menu.

This option is useful to avoid re-compilation of files. For instance, when selecting other packages of MIDAS that were not included in a first selection.

The update procedure also runs in the background and the output is written in the file `/midas/98NOV/tmp/update.`*here_is_a_number*.

2.9 Setup of MIDAS

While the installation or update of MIDAS is still running, you can execute this step from the `config` menu by selecting option `8 - setup MIDAS`. This script will ask you for five definitions that will be updated in your file `/midas/98NOV/monit/syskeys.unix` (default values are shown in parenthesis):

`Name of the computer/system`: Max. 20 characters. The string will be appended to the message '`MIDAS version 98NOV on` ' and will welcome you when starting a MIDAS session.

`Debugger command`: Max. 8 characters. Name of the debugger in your system (adb, dbx, sdb, xdb, etc) that will be invoked when running a MIDAS command with the DEBUG option.

`Command to print a Postscript file`: Max. 20 characters. Command to send a Postscript file to a Postscript printer.

`Command to print an ASCII file`: Max. 20 characters. Name of Command to convert an ASCII file into Postscript and then send it to a Postscript printer.

`Email of the MIDAS administrator at your site (problem-report service` Max. 40 characters. Problem reports generated by the `REPORT/PROBLEM` and the `Feedback...` option in the Graphical User Interface HELP will be sent to this address. If your site does not a have a person responsible for such a service, the default is the MIDAS problem-report service supported at ESO.

The `setup MIDAS` will now configure some files needed for the start-up of MIDAS:

```
/midas/98NOV/system/unix/inmidas
/midas/98NOV/system/unix/gomidas
/midas/98NOV/system/unix/helpmidas
```

`inmidas` and `gomidas` are the front-end of MIDAS for users. `helpmidas` is a script to start a stand-alone GUI help for MIDAS, equivalent to execute `crea/gui help` within MIDAS. These files should be copied by the system administrator to a public directory (included in the *PATH* list) where users of MIDAS can access them by name (e.g. `/usr/local/bin`).

The command `inmidas` starts a MIDAS session. Type `inmidas -help` for a description of which options are available. Options can also be passed to MIDAS through their equivalent environment variables, but options passed by argument take preference over those defined in the environment.

Next is a list of nroff files for the UNIX **man** command. describing the syntax for the previous scripts and should be copied to a local *man* directory (e.g */usr/local/manl*):

```
/midas/98NOV/system/unix/inmidas.l
/midas/98NOV/system/unix/gomidas.l
/midas/98NOV/system/unix/helpmidas.l
/midas/98NOV/system/unix/readline.l
```

2.10 Running MIDAS for the first time

Once `option 6 - install MIDAS` is successfully completed you still need to initialize some MIDAS files by executing the following sequence:

```
% mkdir /midas/tmp
% cd /midas/tmp
% inmidas
Midas 001> @ compile.all
(this will produce about 40 lines of output and it will exit to the Shell)
% gomidas
Midas 002 > @ ascii_bin
(this will create the LUTs and ITTs files)
Midas 003 > bye
```

Now you are ready to run MIDAS. Type *inmidas* for your first MIDAS session or *gomidas* to resume a previous interrupted MIDAS session.

2.11 Verifying MIDAS

To verify the correct installation of MIDAS you can now execute a set of verification procedures:

```
% cd /midas/tmp
% inmidas
Midas 001> @ vericopy
```

The `vericopy` procedure will copy the verification procedures and some data files to your working directory. Some of these procedures use, by default, the display facilities but they can be disabled with a given argument. To run a particular verification procedure:

```
Midas 002> @@ verify XX
```

where *XX* can be *1,3,4,5,6,7,t1,t2 or dio*, or if you want to run all of them (it stops if something fails), just type:

```
Midas 003> @@ veriall
```

The verification procedures are designed to exercise MIDAS commands and to compare results with internal and correct values. They should stop and report to the operator if the tests detect problems with MIDAS commands.

NOTE: Distributed with the official tape and under ftp a verification form is now included. We kindly ask you to fill this form once you have finished the installation and verification of MIDAS and return it to the address indicated or via email by using the ASCII format file provided in `/midas/98NOV/install/unix/verification.doc`

2.12 Cleaning MIDAS

If you are satisfied with the installation, you could clean the MIDAS directory and get some disk space by removing installation files or even packages of MIDAS that are not needed. A backup of the whole MIDAS directory is recommended before executing this step.

From the `config` menu, select the option `9 - clean MIDAS`. You will get a menu of choices, and for any option you select, confirmation will be requested before it starts removing files:

```
MIDAS DELETE MENU:
==================
1 - Delete only object files.
2 - Delete object and source files.
3 - Delete object, source files and libraries.
4 - Delete executable files.
```

Option 1 deletes all object files (.o) under the MIDAS release directory (upto 20% of MIDAS size) but they do not compromise MIDAS performance. It's the safest option.

Option 2 deletes all object and source files (.o .c and .for) under the MIDAS release directory (upto 45% of MIDAS size). They are not needed for MIDAS performance, however you will not be able to install new patches without sources. You still can compile and link your own MIDAS code.
NOTE: DO NOT CHOSE this option on IBM/AIX systems, it will make MIDAS unusable because libmidas.a is a shared library and is needed by all executables.

Option 3, like Option 2, removes also all static library files (.a) (upto 55% of MIDAS size). Since MIDAS uses shared libraries the static ones are not required for development with MIDAS.

Option 4, removes only executables files (.exe) for the optional package categories: applic, stdred and contrib. The core of MIDAS remains, you only remove packages you do not need anymore. Source files remain so executables could be regenerated if necessary.

If what you really want is to reduce your MIDAS installation to the minimun but still operational expresion, you could also use our *cleanmidas* script used to generate the only binaries releases of MIDAS (it is like Option 3 plus other unnecessary files). You use the script in this way:

```
% cd /midas/98NOV
% ./system/unix/cleanmidas
```

2.13 Access to tape devices in MIDAS

A more complete description of how MIDAS accesses tape devices can be found in the chapters *Tape Devices in MIDAS* and *Tapeserver Installation.*

In MIDAS, tape devices are accessed via the `INTAPE/OUTTAPE` commands. They accept logical and system devicenames for tape devices.

- **Access to tape devices via logical names**
  Logical names for tape devices start with the string `tape` (case insensitive) and must be defined in the environment. Example:

  ```
  Midas 003> $echo $tape0
  ws7:/dev/nrst1
  (or just /dev/nrst1 if the tape device is located in localhost)
  Midas 003> INTAPE/FITS 1-10 aaa tape0
  ```

  When first started, MIDAS appends to the environment all definitions found in the configuration file `/midas/98NOV/monit/devices.sh`. You will have to edit this file to assign the system names for your tape devices *hostname:devicename* to the logical names `tapename`, and must follow the next convention:

  `set` *tapename*`=`*hostname:devicename* `; export` *tapename* `# comments`

  where:

  tapename is the logical name used by `INTAPE/OUTTAPE` commands. The name must start with the prefix `tape` (lower case) to be considered a tape device.

  hostname is the official name of the host to which the device tape is attached to.
  NOTE: alias hostnames are not recognized. Official name is printed by the command `hostname`.

  devicename is the system device name for the tape in its `not_rewind_mode`. On many Unix systems, the `not_rewind_mode` is selected for those tape devices names with the prefix character: `n`.
  NOTE: The MIDAS tape interface will not work properly with the `rewind_mode` name.

  The character : is the separator between *hostname* and *devicename*. An example for the file `devices.sh` could look like:

```
set tape0=ns0:/dev/nrst0 ; export tape0  # here_some_comments
set tape8mm=ns2:/dev/nrst2 ; export tape8mm # here_some_comments
```

Also the comments for each logical tape definition (anything after the `#` character) can be used later for the generation of the help file `/midas/98NOV/prim/help/tapede.alq` (used by the MIDAS command `HELP [TAPEDEVICES]`) if they follow the next convention:

```
#:tape_type:capacity_density:location
```

where:

tape_type stands for the type of the tape (maximum of 9 characters). E.g.: DAT/DDS, 8mm, QIC-24.

capacity_density stands for the capacity or density of the tape (maximum 9 characters). E.g.: 1.2Gb, 8Gb, 6250 bpi.

location stands for the physical location of the tape device (maximum 16 characters). E.g.: User room 2nd floor, room num. 443.

The MIDAS help file `/midas/98NOV/prim/help/tapede.alq` is generated by executing the following:

```
% cd /midas/98NOV/prim/help
% make tapede.alq
```

An entry in the file `devices.sh` like:

```
set tape0=ns0:/dev/nrst0 ; export tape0  #:DAT/DDS:1.2Gb:Tape room
```

will generate an entry in the help file `tapede.alq` like:

Name	Host	Device	Tape	Cap/Den	Location
tape0	ns0	/dev/nrst0	DAT/DDS	1.2Gb	Tape room

- **Access to tape devices via system names**
  System devicenames for tape devices can also be used directly in the `INTAPE/OUTTAPE` command line. The syntax must be *devicename* if the tape device is located in localhost, or *hostname:devicename* if the tape device is located in a remote host As described above *devicename* is the system device name for the tape in its *not_rewind_mode*. Example:

```
Midas 003> INTAPE/FITS 1-10 aaa ws7:/dev/nrst1
```
(or if the tape device is located in localhost)
```
Midas 003> INTAPE/FITS 1-10 aaa /dev/nrst1
```

2.14 FITS Data Decompression on the Fly

Compressed FITS files on disk can also be accessed with the MIDAS command INTAPE/FITS. Compressed FITS files are recognized by extra extensions(e.g *.z* or *.Z*) to their names. If INTAPE/FITS matches a filename with one of its expected compressed extensions, the file will be decompressed with the command indicated for such extension (e.g. *.Z* files will be decompressed with the command *zcat*). The decompression of the FITS file is done on-the-fly, that means that the decompress command is spawned in a piped mode, sending the decompressed data directly to the input of the INTAPE command. This improves speed access and reduces disk space because it does not create temporary files.

The decompression code uses, by default, a *built-in* table of compressed extensions and decompress commands as indicated in the following table:

Built-in table:	
Extension	**Command**
.Z	zcat
.z	gunzip -c

The algorithm to apply the decompression is:

- If the file exists an the filename matches on the compressed extensions the file will be decompressed.
- If the file does not exist, the first file with a filename+extension found in the same directory will be decompressed.
- In any other case the file will be opened if it exists or an error will be returned if it does not exist.
- Whenever a decompress command is executed it will also be displayed in the standard output.

Example: In a directory with no `myfits.mt` file but where a `myfits.mt.Z` file exists, MIDAS will execute something like "`zcat myfits.mt.Z | INTAPE/FITS`"

```
Midas 001> $ls myfits*
myfits.mt.Z
```

```
Midas 002> INTAPE/FITS 1 xxx myfits.mt
zcat myfits.mt.Z
Image xxx0001 ...
Midas 003>
```

The *built-in* decompression table is ignored and overwritten by another user-defined table in a file pointed by the environment variable `DECOMPRESS_TABLE` if it exists, otherwise in a MIDAS-defined table located in the file `/midas/98NOV/incl/decompress.dat` if it exists. The syntax of these two files is described in the header of the `/midas/98NOV/incl/decompress_dat` file, which could also be used as a template.

2.15 Access to printer-devices in MIDAS

Printer-devices in MIDAS are used to print out text, graphics (plots) and images loaded onto the display window of MIDAS. In order to ease the installation a number of generic definitions of printer-devices used by MIDAS are defined in the file `/midas/98NOV/monit/devices.sh`, like:

`LPRINT` for the main line printer (text output only)
`LASER` for Postscript b+w laser printer
`PLASER` has to be the same as LASER !!
`COLOUR` for Postscript color laser printer
`SLIDE` for film recorder (slides)
`PENPLOT` for a pen plotter (HP Graphics Language format)

You have to edit this file and assign the printer system names to the logical names above (if existing at your site).

Apart from these logical names also system names for printers can be used. However, the use of printer-devices for graphics output (plots) involves some extra installation work. A separate chapter, Chapter 3 *Installation of Graphics Devices* describes in detail how extra devices for graphics can be added.

2.16 Line editing in MIDAS

MIDAS is provided with two different line editors `TermWindows` and `GNU readline`. `TermWindows` was developed at ESO and has been until 94MAY the line editor used by the MIDAS monitor. However, starting with 94NOV the line editor by default is now the `GNU readline` but the `TermWindows` is still available the user environment variable `TERMWIN` is set to `yes`.

The `GNU readline` is a public domain library widely supported on many UNIX platforms and enhances the line-editor capabilities of MIDAS like a history stack of commands, Emacs or Vi editing mode, command and filename completion functions and a communication channel with the MIDAS Graphical User Interface HELP for `help-on-line` functions.

The editing mode by default with `GNU readline` is `Emacs` but it can be switched to mode `Vi` by setting the variable `editing-mode` to `vi` in the initialization file `$INPUTRC` if exists otherwise in the file *$HOME/.inputrc*. As a brief introduction to `GNU readline` here is list of most common key-sequences:

```
Self-insert                 CTR-L  -> clear-screen
CTR-E -> end-of-line        CTR-A  -> beginning-of-line
CTR-U -> unix-line-discard  CTR-K  -> kill-line
TAB   -> complete           TABs   -> possible-completions
CTR-X -> (also F1) on-line-help through the GUI HELP
```

More detailed information about editing commands with `GNU readline` in the man-page */midas/98NOV/system/unix/readline.l*

2.17 Stand-alone MIDAS utilities

Chapter 5 `Tapeserver Installation` and Chapter 6 `Other MIDAS Utilities` describe the stand-alone MIDAS utilities, that is, programs that run, outside a MIDAS session, at the system level. These are optional programs located in the */midas/98NOV/util* directory and are not included in the standard installation of MIDAS.

2.18 Calibration and Demo Data for MIDAS

The distribution file containing the sources of MIDAS does not include any calibration data required by some packages or the demo data required with some tutorials. The reason for this is to reduce the distribution size of MIDAS via Internet to the essentially minimum required, leaving the retrieval of this bulky data to the discretion of the operator deciding which optional packages are installed.

In our *anonymous* and *midas* ftp accounts the calibration and demo data is located under the `./calib` and `./demo` subdirectories respectively. For those receiving MIDAS via tapes, the calibration and demo data is located as the second file on the distribution tape, just after the sources of MIDAS.

Data is distributed in compressed-tar files. The whole calibration data is located in one single file called `calib.tar.Z` while demo data is offered in different files, each of them representing the tutorial for the package with the same name, e.g. `wavelet.tar.Z`

contains the demo data for the tutorial of the *wavelet* package and should only be retrieved and installed if this package has been selected and installed. (See section 2.6). The file `core.tar.Z` contains the demo data for all tutorials available in the *core* of MIDAS, e.g. *TUTORIAL/EXTRACT, TUTORIAL/FILTER, TUTORIAL/GRAPHICS, TUTORIAL/ITT, TUTORIAL/LUT, TUTORIAL/SPLIT, TUTORIAL/TABLE, TUTORIAL/ALIGN and TUTORIAL/FIT.*

Calibration and demo data is release-independent therefore it must be installed, not under the */midas/98NOV* directory, but under `/midas/calib` and */midas/demo* or equivalent. Tutorial and calibration procedures expect to find their data under these two directories.

For a proper installation of the demo data for all tutorials in the *core* of MIDAS (file *core.tar.Z*) type the following:

```
% cd /midas
(from tape)
% mt -f tapedevice rewind
% mt -f tapedevice fsf 1
% tar xvf tapedevice core.tar.Z  (from anonymous ftp)
% ftp ftphost.hq.eso.org       (or 134.171.6.9)
Name:  ftp
Password: ¡your_email_address¿
ftp> cd /midaspub/demo
ftp> get README.demo
ftp> binary                     (necessary for binary files)
ftp> get core.tar.Z
ftp> get  any_other_file
ftp> quit
```

Then to decompress, un-tar and install:

```
% zcat core.tar.Z | tar xvf -
% cd /midas/demo/install
% cat README.core
% inmidas
Midas 001> @@ core
Midas 002> bye
```

This procedure will populate the directory `/midas/demo/data` which is the only one required to run the *core* tutorials, so afterwards you can remove the directories `/midas/demo/install` and `/midas/demo/raw` and the compressed-tar file:

```
% cd /midas
% rm -rf core.tar.Z ./demo/install ./demo/raw
```

2.19 MIDAS documentation

The file containing the sources of MIDAS does not include the MIDAS documentation. It is located as the third file in the distribution tape or in our *anonymous* and *midas* ftp accounts under the subdirectory `./98NOV/doc`.

All MIDAS documentation is classified in four volumes and distributed in compressed Postscript files:

`vol1.ps.z` - *ESO-MIDAS Volume A: System*
`vol2.ps.Z` - *ESO-MIDAS Volume B: Data Reduction*
`vol3.tar.Z`- *ESO-MIDAS Volume C: Detailed Command Description*
`env.ps.Z` - *ESO-MIDAS Environment*

The file `vol3.tar.Z` is a compressed-tar file which contains the frame for volume C. The individual contexts must be printed separately. For further information see the DOCREADME file.

You will also find smaller compressed Postscript files for the individual chapters of the MIDAS volumes A and B and for the figures to be included in the various chapters.

Chapter 3

Installation of Graphic Devices

Below you will find a description of how to implement (new) graphic devices or how to make adjustments according to the local situation. Section 1 gives an overview of how the MIDAS plot system works. Section 2 describes the various files which play a role in the system. Finally, section 3 gives an example of how and where to make modifications in case a new device has to be added to the system.

3.1 The MIDAS Graphics System

All information about the status of the MIDAS plot package is stored in MIDAS keywords. This information describes the layout of the plot(s) to be produced, the last plot file created and the graphic device to which the result will be sent. Once a plot file (with extension *.plt*) is created the user can send this file to another device than the default one. These two possibilities (default device name stored in keyword or device name from user input) make life somewhat more complicated, since the user may use MIDAS device names rather than the system names.

If the setup of the system is correctly done MIDAS knows three sets of device names. The first set contains the names of the graphics terminal, the graphic windows, and the display windows. The second one contains the system device names of the hardcopy devices, or a subset of that. Finally, a third set of names contains logical names (MIDAS device names) for a few of the devices in the second set.

To solve this complication the names for the hardcopy devices stored in the MIDAS keywords are identical to the system names. If a MIDAS device (logical) name is used, MIDAS will first convert this name into the system name. If no corresponding system name is found a fatal error will be given. Once the system name is established the plot system knows which of the drivers has to be used in order to produce the plot. If a window or terminal name is used, or an abbreviation of that, MIDAS will expand that name to its full name known by the plot system.

The assignment of the hardcopy devices as the output device is done in the procedure *plassign.prg*. Copying the plot-file onto a hardcopy device is done by the procedure *sendplot.prg*. Both procedures make use of the separate procedure *checkdev.prg* that checks device names and, if required, converts MIDAS logical names into system names. The system names of all devices should be included in the file *agldevs.dat* which can be found in the directory */midas/98NOV/systab/ascii/plot*. For each device the file also gives the driver that will be used for that device.

3.2 Modification or Installation of New Devices

Within the MIDAS directory structure you will find a directory */midas/98NOV/libsrc/agl* that contains all source code for building the lower level AGL library which the MIDAS plot system needs. The code can roughly be divided into device independent routines and the device drivers. Information about the first part can be found in the AGL References manual; the latter is described in the set of small device documents. However, this graphics directory is not relevant for changes in the graphics devices. For that purpose only minor modifications are needed elsewhere. They are in different subdirectories of the MIDAS directory tree.

Below, in a few small sections you will find what to do in order to install a new hardcopy graphics device.

3.2.1 The file *agldevs.dat*

MIDAS can handle a number of devices. The file *agldevs.dat* in the directory */midas/98NOV/systab/ascii/plot* defines which drivers have to be used for the various devices. In the system environment variable `AGL3CONFIG` points to this directory. The variable is defined in the file */midas/98NOV/monit/midlogs.sh*.

The format in the *agldevs.dat* file is simple: each system name is followed by a colon and then followed by a driver name. Comments are preceded by a hash (#) character. In this file you will first find a number of lines describing graphic terminals, graphics windows and display windows. Hereafter, you will find a number of lines describing the available hardcopy devices at ESO and their associated drivers. At the end in the Appendix A an example is given.

In case one wants to use a device that is not included in the *agldevs.dat* file the device can be defined by the command:

```
% setenv AGL3DEV dev_driver # (C-Shell)
% AGL3DEV=dev_driver ; export AGL3DEV # (Bourne-Shell)
```

where `dev_driver` stands for the name of the device driver which AGL should use (provided it supports that device) when it is made to plot-file. A useful application of this possibility

is if one wants to use a graphics terminal that is not of the default MIDAS type (VT640). The file *agldevs.dat* contains a listing of all available terminal drivers.

3.2.2 The file *devices.sh*

This file can be found in */midas/98NOV/monit* and contains the logical MIDAS names for a limited number of the available hardcopy devices. The format is described in the file. Make sure that the names in the file are unique. In addition, if names are added or modified also the procedure *checkdev.prg* may need to be modified (see below).

3.2.3 The procedure *checkdev.prg*

This procedure in the directory */midas/98NOV/prim/proc* converts MIDAS logical names into system names and checks these on their existence in the *agldevs.dat* table. Hence it relies on the complete and correct information in this file. In case you want to install (a) new MIDAS logical name(s) the procedure only needs minor modifications. Comments indicate where the changes have to be made.

3.3 Examples

3.3.1 Adding a hardcopy device (system name only)

Suppose one wants to add a hardcopy device with a system name `laserprinter` and suppose this device is a postscript device. The only thing that is needed is to include the system name in the *agldevs.dat* file:

```
laserprinter:pscript              # new device name
```

3.3.2 Adding a hardcopy device (MIDAS name and system name)

Suppose one wants to add the MIDAS device name `mdevice` to the system, which should point to the system name `laserprinter`.

The first step is the same as in the previous example: Add an extra line in the file *agldevs.dat*:

```
laserprinter:pscript              # new device name
```

The second step would be to include the MIDAS name and system name in the file *device.sh*:

```
mdevice=laserprinter; export mdevice
```

Finally, in the procedure *checkdev.prg* a few lines should be included in order to recognize the new logical name. The addition should be done in the section "MIDAS Logical names". For this particular example these line should look like:

```
ELSEIF P1(1:3) .EQ. "MDE" THEN    ! here for mdevice
   P1 = M$SYMBOL("MDEVICE")
```

In case of problems you could use the MIDAS command `DEBUG/PROCEDURE`. This enables you to execute the `ASSIGN/GRAPHICS` and the `COPY/GRAPHICS` command step by step and to find out where a device name is missing or misinterpreted.

Finally, whenever you modify your system please document this. E.g. add or modify some help documentation in the *assigngrap.hlq*, *copy_grap.hlq*, and the *printe.alq* files in */midas/98NOV/prim/help*.

Chapter 4

Tape Devices in MIDAS (Optional)

4.1 Introduction

This chapter is intended to give an overview of the process required to install, modify, and test the interface used by MIDAS to access local tape drives. This interface, used by the MIDAS `INTAPE/OUTTAPE` commands, will allow MIDAS to access data from your own local tape devices.

This chapter also complements the document *Access to Devices* issued together with these installation notes. It provides more detailed information and code about tape interfaces in MIDAS and related files. It is mandatory to read this chapter if you have to write or modify these interfaces.

Most UNIX workstations do not have their own tape unit but share a common network tape station. A remote tapeserver is now available for UNIX systems using TCP/IP protocols. When installed, it enables the MIDAS `INTAPE/OUTTAPE` commands to access a remote drive as if it were a local device. More information about how to install a MIDAS tapeserver can be found in the next chapter.

4.2 Tape interface in MIDAS

MIDAS users read and write their data by using two well-known MIDAS commands called `INTAPE/OUTTAPE` that read and write user data in FITS and IHAP formats. The new version of these two commands used in MIDAS are able to access data files both on disk and on tape. The argument for the input/output device (prefix) can be the system device-name (e.g /dev/nrst0) or a environment variable starting with the string `tape` (case insensitive) defining a system device-name, or filenames on disk.

By default, the access to files on disk is included in all releases of MIDAS. It uses the standard file interface common to all Unix systems.

Access to tape devices can differ from one system to another, and, within the same system, not all tape devices are accessed the same way. Drivers for tape devices are depending on the manufacturer implementation (and sometimes they also change with different versions of the OS), however more and more manufacturers are adopting a semi-standard interface called `mtio`. The standard MIDAS installation provides a generic tape driver based in this interface and therefore is able to handle most tape devices. If this driver is not sufficient for your some of your tape devices you will have to implement your own driver following the procedures indicated in the next sections.

4.3 The generic driver

Starting with 94MAY, MIDAS releases are delivered with a generic driver called `iodevg.c` in the directory `/midas/98NOV/lib/os/unix`. This is based in the semi-standard `mtio` (general magnetic tape) interface. The MIDAS tape interface will select this driver, by default, whenever your device-name is not found in the `devcap.dat` file (pointed by environment variable `DEVCAPFILE`). The generic driver can support most of the tape devices (tested on SunOS-4.1.3 and HP-UX) without any further modification, just try to access your tape by giving the device-name, in its no-rewind mode, to the `INTAPE/OUTTAPE` commands (check `HELP/INTAPE` for more details).

4.4 Before modifying the tape interface

If you are installing MIDAS for the first time you might not need to modify the tape interface. The `5 - preinstall` option in the `config` MIDAS procedure could have done the work for you already .

This procedure is executed in the first steps of the installation of MIDAS. It asks the operator to choose one system name from a list of names available. The names are in fact subdirectories under `/midas/98NOV/install/unix/systems` and they contain system-dependent files of known systems. When selection is made, the procedure will copy all files from the subdirectory identified by that name to the local directory `/midas/98NOV/local`.

Tape interfaces for some known systems are provided in the MIDAS distribution tape. The module `iodev.c` containing the tape interface for your system, together with a `makefile` to compile it, can be found in the directory
`/midas/98NOV/local/libsrc/os/unix` after `5 - preinstall` has been executed. If this is your case, your module will be automatically compiled and included in the libraries, so your `INTAPE/OUTTAPE` commands are ready to access your tape unit.

NOTE: The `iodev.c` modules provided for some known systems should be always tested. In same cases they might need some changes since there are many things that could modify the behavior of the tape interface (e.g. new releases of the O.S., new devices, modification in the drivers, etc)

If you have already written your own `iodev.c` module just copy it to your `/midas/98NOV/local/libsrc/os/unix` if you have this directory, otherwise copy it to `/midas/98NOV/libsrc/os/unix`.

4.5 Using a template file

If after all you have to write your own `iodev.c` module, it would be easier if you take the module `iodevg.c` in/ the directory `/midas/98NOV/local/libsrc/os/unix` which contains the generic driver, and use it as your template file. The template should then be copied to `/midas/98NOV/local/libsrc/os/unix/iodev.c` if you have this directory already, otherwise to `/midas/98NOV/libsrc/os/unix/iodev.c`

4.6 Unix system calls for tape interfaces.

Most of the tapes in Unix are controlled basically in the same way:

- The interface between the program and the tape driver program is the same as for normal files, when using system calls like `open()`, `close()`, `read()` and `write()`.
- For special tape commands like `rewind`, `skip back a file` etc... a different interface is used. The system call is standard Unix and is called `ioctl()`. It accepts as parameter the tape command to be executed and can return a status of the tape. The differences between one machine and another are the names and number of commands supported by the device (usually defined in an system include file called `mtio.h`), and the status codes (usually undefined).

4.7 Writing and checking your tape interface

Below, you will find the procedure steps to follow if you have to change the tape interface.

1. For the programmer the most difficult thing is to guess the status codes from `ioctl` system calls. And this is done easily uncommenting the `printf` lines of the `iostat` routine so the status of all commands will be printed in the console. The most important codes to be detected are:

```
#define M_ST_SUCC  0??     /* Command Successed */
#define M_ST_DATA  0??     /* Data Error */
#define M_ST_BOT   0??     /* Beginning of Tape */
#define M_ST_TAPEM 0??     /* Tape Mark Encountered */
```

If you are lucky enough to have an include file containing those definitions then most of the work is done for you. (e.g. some mVAX/ULTRIX define those values in </usr/sys/vax/tmscp.h>). If you are not so lucky, you will have some fun in the next steps!

The modifications for `iodev.c` are updated in the generic MIDAS library `midaslib.a` through the following sequence:

```
% vi iodev.c # or any other editor
% make SYS=-DDEBUG
% cd /midas/98NOV/lib
% make
```

2. After compiling the `iodev.c` and its object being included in the `midaslib.a` MIDAS library, the next step is to link the first of the tape tests:

   ```
   % cd /midas/98NOV/test/os
   % make testtape.exe
   ```

   This test is interactive and self explanatory. It will print on your terminal a menu containing options for exercising the `iodev` routines. Because you uncommented the printf lines of the `iostat` routine, you will get the tape status for each command executed.

   Put a tape in the device for testing the "iodev", then type:

   ```
   % testtape.exe
   ```

   Using the different options of the `testtape.exe` menu you can exercise all the routines of `iodev`. Also you can force some known situations for getting the different codes. (e.g. trying to read some data where no data was written, or trying to read data where a File Mark was written, you will get the definitions (in error register `mt.mt_erreg`):

   ```
   #define M_ST_DATA  0??      /* Data Error */
   #define M_ST_TAPEM 0??      /* Tape Mark Encountered */
   ```

   Playing with the test and modifying the `iodev` according to the behavior of your tape unit, you can tune your tape interface, ready for the next steps.

3. Once you think `iodev` is ready and in accordance with the functionality defined, you could check the upper level "osu".

   But before, you have to set-up the variable of the environment `DEVCAPFILE` which defines the location of the file `devcap.dat`. The contents of this file should define the name and hosts of your tapes (see the document *Access to Devices*, Section 2.4, or read the template file `/midas/98NOV/incl/devcap.dat`). You can do so by:

```
% setenv DEVCAPFILE /midas/98NOV/incl/devcap.dat # (C-Shell)
% DEVCAPFILE=/midas/98NOV/incl/devcap.dat #(Bourne-Shell)
% export DEVCAPFILE (Bourne-Shell)
```

Thereafter, proceed with:

```
% cd /midas/98NOV/test/os
% make testosu.exe
% testosu.exe
```

The script `testosu.exe` will display a menu for selection that will allow you to check the upper tape interface `osu`. It works with the same philosophy as the script `testape.exe`. Following the options in the menu you can:

open/close tape devices
write/close file on tape
skip over files backward/forward
etc...

When getting errors, you will have to go back to the previous step, and find out what was wrong or missed.

4. There is still one more exhaustive `osu` test:

```
% cd /midas/98NOV/test/os
% make atestosu.exe
% atestosu.exe
```

This test will try to exercise most possible combinations of `osu` calling sequences, checking the return codes and writing and reading data in all possible modes. The test will exit if an error is detected, so you can then reproduce the error by running by hand the last commands before the fault with the `testosu.exe` test.

NOTE: Only when this last test `atestosu.exe` has been executed to the end with no error, can you be sure that all tape interfaces `osu` and `iodev` are ready to be used by MIDAS commands like `INTAPE/OUTTAPE`.

5. Finally, you will need to compile the `iodev.c` file one more time but without the conditional flag DEBUG:

```
% vi iodev.c # or any other editor.
% make
% cd /midas/98NOV/lib
% make
% cd /midas/98NOV/prim/dio/src
% make
```

Chapter 5

Tapeserver Installation (Optional)

The MIDAS tapeserver is an optional package that allows MIDAS commands like `INTAPE/OUTTAPE` to access, tape drives located in other workstations via the network,

In this chapter, the term `tapehost` is used to refer to machines with tape drives, and the term `client` to any other machine in the network accessing the remote tape drives in `tapehosts`. Any host in the network can be `tapehost`, `client`, or both at the same time.

The connection between the `client` and `tapehost` is based on the TCP/IP socket interprocess communication. It is of type `STREAMS` which provides sequenced, reliable, two-way connection based byte streams. `client` and `tapehost` talk to each other via a new INET service called `mtape`. This new service must be available to all hosts through the file `/etc/services` or its equivalent in the NIS (Network Information System).

The MIDAS tapeserver is spawned by the Internet services daemon `inetd`. This daemon listens for connections on the Internet addresses of the services that its configuration file `/etc/inetd.conf` specifies. When a connection on the service `mtape` is found, it invokes the MIDAS tapeserver daemon `/usr/local/etc/in.mtaped`.

NOTES: The installation of MIDAS (at least the core) is assumed to be done already. This notes applies to SUN/OS workstations. Slight modifications may apply to other machines.

5.1 Compiling the MIDAS tapeserver sources

```
% cd /midas/98NOV/util/tapeserv
% make
```

The compilation will create an executable called `/midas/98NOV/util/exec/inmtaped.exe` that will be used later as the tapeserver daemon.

5.2 A new service for the MIDAS tapeserver

For all clients and tapehosts add the following line in the file `/etc/services`:

```
mtape service_number/tcp # MIDAS tapeserver
```

where `service_numer` is any number in the range 1024-2048 not used by any other service. If you use NIS (Network Information Service) this modification might be done in the master host for NIS.

5.3 The $DEVCAPFILE, a tape configuration file

This file defines the configuration of tape devices which behaviour deviates from the default definitions: `cl=generic` for the driver, `tm=1` one Tape Mark at End of Tape, `am=1` Append Mode is available, `bw=0` Backward Movement is allowed, and accepts Variable Block Size. The file is addressed through the MIDAS environment variable DEVCAPFILE, by default defined as `/midas/98NOV/incl/devcap.dat`, which can be used as template for your own configuration.

5.4 The MIDAS Tapeserver Daemon

The installation of the MIDAS tapeserver is NOT included in the default installation of MIDAS, so the first thing you will have to do is to compile and link:

```
% cd /midas/98NOV/util/tapserv
% make
```

The MIDAS tapeserver daemon `in.mtaped` does not use any variable `DEVCAPFILE`, but it addresses the file `/usr/local/lib/devcap.dat` if it exists, otherwise the tapeserver addresses tape devices with the *generic* driver.

If you have configured your own `devcap.dat` file for client hosts you will have to copy it to tt /usr/local/lib/devcap.dat in the tapeserver host. As superuser:

```
# cd /usr/local/lib
# cp /midas/98NOV/incl/devcap.dat devcap.dat
or
# ln -s /midas/98NOV/incl/devcap.dat devcap.dat
```

The second option (soft-link) has the advantage of using the same file used by MIDAS users, so any modification in the original file will be also detected by the MIDAS tapeserver.

Copy the tapeserver daemon `inmtaped.exe` to `/usr/local/etc`

```
# cd /usr/local/etc
# cp /midas/98NOV/util/exec/inmtaped.exe in.mtaped
```

5.5 The file `inetd.conf`

This is the configuration file used by the `inetd` daemon to invoke, only when they are needed, other server daemons (like the MIDAS tapeserver). Edit the file `/etc/inetd.conf` and add the following line:

```
mtape stream tcp nowait nobody /usr/local/etc/in.mtaped in.mtaped
```

5.6 SIGHUP Signal

Send the signal SIGHUP to the `inetd` proccess to force it to reread its configuration file, and to listen for the new MIDAS tapeserver service. This can be done by:

```
# ps -ax | grep inetd | grep -v grep
```
(this will tell you the PID number for the inetd proccess)
```
# kill -HUP inetdPID
```

5.7 Access a Remote Tape

To access a remote tape try now from client host the MIDAS commands `INTAPE/OUTTAPE`.

5.8 Problems

If problems arise, use the following test programs: `testosu.exe`, `osutest.exe`, and `atestosu.exe`, after first having them compiled and linked. This can be done via:

```
cd /midas/98NOV/test/os
make atestosu.exe
make testosu.exe
make osutest.exe
```

and if you have configured your own `devcap.dat` file:

```
setenv DEVCAPFILE /midas/98NOV/incl/devcap.dat # (C-Shell)
DEVCAPFILE=/midas/98NOV/incl/devcap.dat # (Bourne-Shell)
export DEVCAPFILE # (Bourne-Shell)
```

`testosu.exe` and `osutest.exe` are similar tests, and offer you a menu of options. `atestosu.exe` will execute a complete verification test on the selected tape.

5.9 Tracing the Server

The server can be traced by allowing debug messages into a log-file. Edit the file `/etc/syslog.conf` and be sure that `debug` messages are sent to some log-file (`e.g. /var/log/mtaped.log`. create an empty file with this name first). Add an entry in the `/etc/syslog.conf` as follow:

```
local0.debug /var/log/mtaped.log
```
or
```
user.debug /var/log/mtaped.log
```
if your system does not support the facility `local0` (e.g. Ultrix)

Send the signal SIGHUP to the `syslogd` daemon to force it to reread its configuration file, therefore sending `debug` messages and others of higher priority from `local0` daemon (`in.mtaped`) to the log-file.

```
# ps -ax | grep syslogd | grep -v grep
```
(this will tell you the PID number for the syslogd proccess)
```
# kill -HUP syslogdPID
```

You can read the messages coming in the log-file by:

```
# tail -f /var/log/mtaped.log
```

5.10 NOTES

For those machines not supporting the `syslog()` routine (e.g. pure UNIX SYSTEM_V) the tapeserver can be recompiled with the flag `-DNO_SYSLOG` and, optionally, with the flag `-DDEBUG` (it will fill the log-file with debug information):

```
# cd /midas/98NOV/util/tapeserv
# make 'SYS="-DNO_SYLOG -DDEBUG"'
```

The log-file is then fixed to `/var/log/inmtape.log` but can easily be changed in the source.

Chapter 6

Other MIDAS Utilities (Optional)

6.1 bdf2ps: convert MIDAS-bdf files to Postscript format

- **Description:** The command bdf2ps reads a single MIDAS-bdf file or a specification file with a set of images and creates a Postscript file depending on the options.
- **Synopsis:**
  bdf2ps [-wbneavfN] [-i bdf-file] [-o ps-file] [-l spec-file] [-p page-format] [-r resolution] [-c lut] [-s xscale,yscale] [-z lcut,hcut]
- **Installation:**

  ```
  % cd /midas/98NOV/util/bdf2ps
  % make
  ```
  And, as superuser, copy the binary and man-page files to standard system directories (e.g.: */usr/local/bin* and */usr/man/manl*):
  ```
  # cp /midas/98NOV/util/exec/bdf2ps.exe /usr/local/bin/bdf2ps
  # cp /midas/98NOV/util/bdf2ps/bdf2ps.l /usr/man/manl/bdf2ps.l
  ```

6.2 ltape: list headers of FITS and IHAP format files on tape or disk

- **Description:** The command ltape reads files in either FITS and IHAP format from disk or tape and list their headers. Both a one line summary and a full listing are available. A number of comment fields can be provided and will be printed in the list header.
- **Synopsis:**
  ltape [-hnv] [-i input] [-o output-file] [-l list-file]

- **Installation:**

  ```
  % cd /midas/98NOV/util/ltape
  % make
  ```
  And, as superuser, copy the binary and man-page files to standard system directories (e.g.: */usr/local/bin* and */usr/man/manl*):
  ```
  # cp /midas/98NOV/util/exec/ltape.exe /usr/local/bin/ltape
  # cp /midas/98NOV/util/ltape/ltape.l /usr/man/manl/ltape.l
  ```

Chapter 7

Panic Section and other services

- **Panic Section**
  In case of problems with the MIDAS installation use the electronic email service:

  midas@eso.org

  or have a look to our Web Page:

  http://www.eso.org/esomidas

- **ESO-MIDAS latest news with finger**
  You can obtain the latest news about releases and patches of MIDAS through the Internet with the command `finger`:

  ```
  % finger -s midas@ftphost.hq.eso.org
  ```

- **ESO-MIDAS Courier via ftp**
  All issues of the ESO-MIDAS *Courier*, the Newsletter of the MIDAS User's Community, are available in Postscript format via the *anonymous* and *midas* ftp account on our *ftphost.hq.eso.org* (or IP 134.171.6.9) under the subdirectory `./courier` or *./midaspub/courier*.

- **ESO-MIDAS on the World Wide Web**
  The ESO-MIDAS WWW pages can be accessed via the ESO home page *http://http.eso.org/eso-homepage.html* or directly reached using the ESO-MIDAS page *http://http.eso.org/midas-info/midas.html*. The information provided contains an overview of the MIDAS hard- and software requirements, distribution policy (including the ESO-MIDAS Request Form), documentation and support, and User Guides. Also, hyper-text links enable the viewing of previous issues of the ESO-MIDAS *Courier* using Ghostview.

Appendix A

Graphic Devices

```
# @(#)agldevs.dat       6.5 (ESO-IPG) 11/30/93 11:09:37
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#.Identification:
#  /$MIDASHOME/$MIDVERS/systab/ascii/plot/agldevs.dat
#
#.Purpose:
#  Defines the graphics devices used by MIDAS.
#
#.Description:
#  Every line not starting with a hash (#) contains two entries, separated by
#  a colon and a comment. The first entry is the system name for the device,
#  the second one is the AGL driver name.
#  In principle all graphics units, except for terminals, should be put into
#  this file. This file is read in the plot package in order to check if a
#  valid device is used. For those who need to install devices not used at ESO,
#  a number of examples are given at the end.
#
#.Syntax:
#  The syntax for each line is simple:
#       <device>:<driver>:<node>:=<command>        # comment
#       where:
#             <device>  Is  the  identification  of  the  physical  device
#                       according to the rules of the operating system.
#                       The  special  device  TT stands  for  the  current
#                       user's  terminal and it is meaningful only in  the
#                       user's local configuration file.
#             <driver>  Is  the identification of the corresponding driver
#                       as  specified in the driver configuration  module.
```

```
#                       Some drivers require further specifications to
#                       select specific characteristics of the device
#                       (this is device dependent) (e.g.: tkg.vt640).
#
#             <node>    Is the optional specification of the remote node
#                       (provided for future support of device in a
#                       distributed environment).
#
#             <command> Is the optional command to be executed when the
#                       device is closed. The = sign is required and is
#                       not part of the command. The % sign may be used
#                       within the command to indicate the place where
#                       the driver generated file name must be put into
#                       the command line.
#
#.Note:
# AT ESO WE ONLY USE THE <device> AND <driver> ENTRIES ***
# For further information please consult the documentation in the file
# /$MIDASHOME/$MIDVERS/doc/agl/aglinsta.doc
#------------------------------------------------------------------------
#
# *** Here for the general terminal. Other terminal devices supported by AGL
#     and MIDAS are below.
graph_term=tkg.vt640     # default DEC VT100 terminal with retrographics
#graph_term=tkg.cit101   # CIT101 with CIG 201 card
#graph_term=tkg.hds22    # HDS 2200
#graph_term=tkg.4010     # tektronix 4010
#graph_term=tkg.4014     # tektronix 4014
#graph_term=tkg.cit101   # CIT101 with CIG 201 car
#graph_term=vt125        # DEC VT125 Terminal
#graph_term=vt125        # DEC VT240 Terminal
#
# *** Here for the default image display (e.g. Deanza)
image_displ:idi.sxw0     # default image display
#
# *** Here for workstations
graph_wnd0:idi.sxw0g     # IDI driver for workstation graphics window 0
graph_wnd1:idi.sxw1g     # IDI driver for workstation graphics window 1
graph_wnd2:idi.sxw2g     # IDI driver for workstation graphics window 2
graph_wnd3:idi.sxw3g     # IDI driver for workstation graphics window 3
image_wnd0:idi.sxw0      # IDI driver for workstation image window 0
image_wnd1:idi.sxw1      # IDI driver for workstation image window 1
```

```
image_wnd2:idi.sxw2      # IDI driver for workstation image window 2
image_wnd3:idi.sxw3      # IDI driver for workstation image window 3
#
# *** Here for the hardcopy systems device names
ps0mmf0:pscript          # APPLE Laserwriter II
ps0mmf0.l:pscript.l      # APPLE Laserwriter II
ps0mmf0.p:pscript.p      # APPLE Laserwriter II
ps2usr0:pscript          # AGFA P400
ps2usr0.l:pscript.l      # AGFA P400
ps2usr0.p:pscript.p      # AGFA P400
ps2usr1:pscript          # HP LaserJet III
ps2usr1.l:pscript.l      # HP LaserJet III
ps2usr1.p:pscript.p      # HP LaserJet III
ps2vlt1:pscript          # HP LaserJet IIID
ps2vlt1.l:pscript.l      # HP LaserJet IIID
ps2vlt1.p:pscript.p      # HP LaserJet IIID
ps3vlt0:pscript          # AGFA P3400
ps3vlt0.l:pscript.l      # AGFA P3400
ps3vlt0.p:pscript.p      # AGFA P3400
ps3vlt3:pscript          # HP LaserJet III
ps3vlt3.l:pscript.l      # HP LaserJet III
ps3vlt3.p:pscript.p      # HP LaserJet III
ps4ast0:pscript          # HP LaserJet IIISi
ps4ast0.l:pscript.l      # HP LaserJet IIISi
ps4ast0.p:pscript.p      # HP LaserJet IIISi
ps4ipg0:pscript          # Apple Laserwriter
ps4ipg0.l:pscript.l      # Apple Laserwriter
ps4ipg0.p:pscript.p      # Apple Laserwriter
ps4ipg1:pscript          # HP LaserJet III
ps4ipg1.l:pscript.l      # HP LaserJet III
ps4ipg1.p:pscript.p      # HP LaserJet III
ps5ast0:pscript          # HP LaserJet IIISi
ps5ast0.l:pscript.l      # HP LaserJet IIISi
ps5ast0.p:pscript.p      # HP LaserJet IIISi
ps5ecf0:pscript          # HP LaserJet III
ps5ecf0.l:pscript.l      # HP LaserJet III
ps5ecf0.p:pscript.p      # HP LaserJet III
pc2usr0:pscript.c        # Tektronix Phaser, paper
pc2usr1:pscript.c        # Tektronix Phaser, overhead
sl2usr0:pscript.c        # Agfa Chromascript
hp2vlt2:hpgl.a           # Kyocera F-500
hp2usr0:hpgl.a           # HP 7550A pen-plotter, A4
```

```
hp2usr1:hpgl.c           # HP 7550A pen-plotter, A3
hp2usr3:hpgl.c           # HP 7550A pen-plotter, A3
versa:raster.200         # MIDAS default name for the Versatec plotter
#
# *** Below a number of examples for HARDCOPY device not available at ESO.
#your_ln03:tkg.ln03      # your local LN03 plus laser printer (name your_ln03)
#exam_qms:tkg.qms        # your local qms laser printer (system name exam_qms)
#my_apple:pscript        # your Apple Laser Writer (system name my_apple)
#
# *** Here for the null device
null:null                # null device
unknown:null             # end of device file
```