National
Optical
Astronomy
Observatories

950 N. Cherry Ave.
P.O. Box 26732
Tucson, Arizona 85726-6732

(602) 327-5511 FAX: (602) 325-9360
Telex: 0666-484 Aura Noao Tuc

Internet: noao@noao.edu

Kitt Peak National Observatory • Cerro Tololo Inter-American Observatory • National Solar Observatory

November 1, 1990

Dear Domain/IRAF Installer,

This is a supplement to the *BSD-UNIX/IRAF Installation Guide* (IG) and the accompanying *UNIX/IRAF Site Manager's Guide* (SMG). You will be using those other documents both for installation and maintenance, but there are some minor differences for the Apollo covered here. **Please read these notes first.**

1. Acknowledgments

Apollo Corporation provided a grant of an Apollo DN-3500 to NOAO for the purpose of porting IRAF in May, 1989 around the time Apollo was purchased by Hewlett-Packard. Pierre Bouchard and Frank Prince of Apollo were instrumental in obtaining the grant. Various persons at the Apollo headquarters in Chelmsford MA have been helpful during the port in solving or working around bugs in Domain/OS and its compilers and debuggers, including Mike Burati, Rick Conti, and Steve Rehrauer.

HP/Apollo made a demo DN-10000 available in Phoenix, AZ for the mini-port to the PRISM architecture. Chris DeVita and Robert Johnson were helpful in providing access to the DN-10000.

Åke Nordlund of the Copenhagen University Observatory pioneered an IRAF port to Domain/OS in 1988 and 1989 before we received our machine, and provided much helpful information. He also acquired a Domain assembler long before we were able to (it is not a released Apollo product), and wrote the lone IRAF assembler routine for the Motorola architecture. Åke and the Denmark Apollo staff also figured out a clever way to avoid using an assembler on the DN-10000 entirely (exploiting an unusual feature of the Domain Fortran libraries), which we subsequently used ourselves as we were unable to acquire an assembler for the PRISM architecture.

2. Domain/OS Operating System and Environment

V2.9 Domain/IRAF was developed on a DN-3500 running Domain/OS SR10.2 for both the Motorola 68882 and Apollo Floating Point Accelerator architectures. It is not known whether this version of IRAF will run under SR10.1 or SR10.0, but quite possibly it will not. Presumably this release of IRAF will continue to run without relinking under SR10.3 when it arrives, and rumor has it that that may be the last release of Domain/OS prior to the arrival of OSF/1 or OSF/2, which will require a brand new port.

Note: It is not known at this time whether NOAO will upgrade to SR10.3. At the time of writing, our DN-3500 is off both hardware and software maintenance and we are seeking funds for their continuation.

IRAF was originally developed in the Berkeley UNIX environment, and that is what we used in Domain; it is assumed that anyone installing IRAF in Domain/OS has installed BSD. With one exception, it is not necessary to have the System V SYSTYPE installed, though of course it cannot hurt. The exception is in creating the named pipes, or *fifo*'s, used for image communication between IRAF and the SAOimage display server. Although this is a one-shot thing, the pipes apparently cannot be created from any operating environment under Domain other than System V, at least at SR10.2. The presence or absence of the Aegis environment is immaterial to the installation and running of IRAF.

3. Domain/OS BSD

In setting up our Apollo for a default BSD environment, a couple of points were noted. The disk had to be `INVOL`'d with Berkeley ACL's prior to installing Domain/OS. Further things appeared to be necessary in order to get a reasonably Berkeley-looking file system:

- `chacl -B` had to be run on *each root directory* intended to contain Berkeley files *before* creating any files in the directory.
- `umask` had to be set individually in each user account; our local value being `022`.

It is therefore important to give a new IRAF root directory Berkeley ACL's prior to loading the tape, otherwise the file protections will not be right (see § 2.1.2 of the IG, "If installing IRAF for the first time..."), e.g.:

```
% cd /usr
% mkdir iraf
% chacl -B iraf
```

4. Loading IRAF Distribution

Distribution tapes for Domain/IRAF are compressed `tar` archives, just as in the BSD-UNIX/IRAF Installation Guide. The only differences between the examples listed in the IG and what you will use to load IRAF are the following substitutions (see, e.g., § 2.2.1). The sizes given are for the uncompressed files.

0	Table of Contents	
1	AS.DOM2.GEN.Z	43.9Mb IRAF, NOAO packages and Domain/SR10.2 sources
2	IB.DOM2.A88.Z	31.0Mb Prism binaries for IRAF system - SR10.2
3	IB.DOM2.F68.Z	25.1Mb m68k/mc68882 binaries for IRAF system - SR10.2
4	IB.DOM2.FPA.Z	25.2Mb m68k/FPA1 binaries for IRAF system - SR10.2
5	NB.DOM2.A88.Z	29.3Mb Prism binaries for NOAO packages - SR10.2
6	NB.DOM2.F68.Z	23.4Mb m68k/mc68882 binaries for NOAO packages - SR10.2
7	NB.DOM2.FPA.Z	23.5Mb m68k/FPA1 binaries for NOAO packages - SR10.2

5. Multiple Architectures

5.1. Installation

Domain/IRAF is configured for multiple architecture support within the Apollo Motorola 680x0 and PRISM series. That is, there is one copy of the IRAF directory tree, source and help files, etc., and three copies of the executables both for the core system and the NOAO packages.

The following examples are from §2.2.3.1 of the IG, with the Apollo "a88k" binary architecture substituted, and assuming the archive consists of compressed tar files on a cassette mounted at /dev/rct12, the no-rewind interface.

```
% cd $iraf
% rm bin.a88k                    # remove old symbolic link
% mkdir bin.a88k                 # create real bin directory
% cd bin.a88k
% mt -f /dev/rct12 fsf 1         # position tape to IB.DOM2.A88
% dd if=/dev/rct12 ibs=126b | uncompress | tar -xpbf 20 -  # read binaries
```

Alternatively we could have rewound the tape and done an `fsf 2` to get to tape file 2. Now read in the NOAO binaries.

```
% cd $iraf/noao
% rm bin.a88k                    # remove old symbolic link
% mkdir bin.a88k                 # create real bin directory
% cd bin.a88k
% mt -f /dev/rct12 fsf 1         # position tape to NB.DOM2.A88
% dd if=/dev/rct12 ibs=126b | uncompress | tar -xpbf 20 -  # read binaries
```

5.2. Runtime Environment

On a Domain/IRAF system with multiple architectures, the system will automatically select the appropriate binaries for running IRAF. On Motorola machines, an unsupported feature of Domain/OS is currently being used to determine whether a floating point accelerator is present. It is possible this could break in the future; if so, simply defining `IRAFARCH` as described below will solve the problem.

5.3. Software Development

User application software development within IRAF, either in IMFORT or SPP, can be done for any of the supported Apollo architectures. All the Motorola machines (e.g. DN-2500, 35x0, and 45x0) support the 68882 floating point coprocessor, while some also have an optional floating point accelerator. Software developers on machines containing more than one Domain architecture should set this environment variable prior to compiling:

PRISM architecture (DN-10000):

```
% setenv IRAFARCH a88k
% setenv MACH a88k
```

Motorola architectures:

```
% setenv MACH m68k                # and
% setenv IRAFARCH m68k_fpa        # FPA or
% setenv IRAFARCH m68k_f68        # 68882
```

Developers in Domain/IRAF must also watch carefully their XC command line arguments. It is best to use the arguments defined in `hlib$mkpkg.inc` for the appropriate architecture. If you compile and link exclusively using MKPKG you do not need to worry as long as `IRAFARCH` and `MACH` are set correctly. It is not recommended to make code changes to the core system or NOAO package, and the system as distributed is configured *generic*. But if it becomes necessary, one could reconfigure the system libraries and bin links as follows:

```
% cd $iraf
% mkpkg showfloat       # assume it is not fpa
% mkpkg m68k_fpa
% cd noao
% mkpkg showfloat
% mkpkg m68k_fpa
```

6. Display Interfaces

IRAF does not support the Domain Display Manager at all. Do not attempt to run the IRAF CL within a Domain Pad, as terminal I/O will be completely mangled. It is best to run entirely in the X11 environment, from an XTERM window, and tell the CL about it (e.g. `cl> stty xtermjh nl=54` for a 54-line xterm window using the xtermjh graphcap entry). If this is not possible for some reason, then it is at least necessary to run the `vt100` server, which tries to convert a Pad into a vt100 emulator. The emulator still has some problems in SR10.2, and can be expected to hang up occasionally, especially if networked into another computer. If this happens, it is necessary to kill the vtserver and start a new one.

6.1. SAOimage

The image display is **SAOimage** from the Center for Astrophysics at Harvard, modified at NOAO to work around Domain bugs. Domain/IRAF comes with executable versions of SAOimage for each target Apollo architecture loaded in the `$hbin` directories. Since this is not part of the standard IRAF system, and we do not yet have a mechanism in place for handling multiple binary architectures at the host level, *you will need to install* `saoimage` *in UNIX by hand after the IRAF installation*. Let's assume you chose `/usr/local/bin` as the localbin directory when you ran the IRAF `hlib$install` script, and that `$hbin` is defined as when logged in via the IRAF account:

```
% su
% ln -s $hbin/saoimage.m68k_f68 /usr/local/bin/saoimage     # 68882 or
% ln -s $hbin/saoimage.m68k_fpa /usr/local/bin/saoimage     # FPA1 or
% ln -s $hbin/saoimage.a88k /usr/local/bin/saoimage         # Prism
```

Now install the auxiliary files necessary for communication with IRAF; this requires that the System V environment be loaded in Domain/OS (all that is really necessary is the sys5.3 version of `/etc/mknod`):

```
% make install_imtoolrc     # (still as root)
% make install_fifo         # -- requires sys5.3 /etc/mknod
% make install_saoman       # optional
```

NOTE: Domain/BSD has some bugs in its implementation of named pipes. This required modifying SAOimage with the following result: you should always fire up SAOimage before you attempt to display an image (e.g. with the DISPLAY task in the images.tv package). However, you will not see the SAOimage window until you actually attempt to display an image:

```
% saoimage -imtool &   # from a UNIX shell
```

Nothing happens; don't be alarmed. Now, from an IRAF window:

```
cl> images
cl> tv
cl> display dev$pix 1
```

Finally, the SAOimage window becomes visible and the image is displayed.

Should it be desired to recompile SAOimage for some reason, e.g. a local source modification, use one of the following from its root directory `$iraf/unix/x11/saoimage`:

```
% make apollo_a88k >& spool &          # or
% make apollo_m68k_f68 >& spool &      # or
% make apollo_m68k_fpa >& spool &
```

7. Possible NFS Problem

There is an unresolved bug, which may be related to the Domain implementation of NFS. In testing, sometimes if many new images were created in a single application in an NFS-mounted directory, the task would abort after creating the 60th file. This did not happen on a native Apollo disk and may represent an underlying NFS bug.

Good luck on your installation! Please let us hear from you if you have any problems or suggestions as you use the system.

Sincerely,

Steve Rooke

IRAF HOTLINE

telephone	`(602) 323-4160`	
internet	`iraf@noao.edu`	
span/hepnet	`noao::iraf`	(noao = 5355)
uucp	`{arizona,decvax,ncar}!noao!iraf` or	
uucp	`uunet!noao.edu!iraf`	
bitnet	`iraf@noao.edu` (through a gateway)	
FAX	`(602) 325-9360`	