An Observer's Guide to Taking CCD Data with ICE

P. Massey T. E. Armandroff, J. Barnes, B. Bohannan,
T. Boroson, G. Jacoby, S. Rooke, R. Seaman D. Silva,
D. Tody

8 Janurary 2000

Contents

1 Introduction

ICE is a system which allows the astronomer to acquire CCD data using IRAF on a SUN workstation. ICE (which stands for "IRAF Control Environment") replaces the FORTH data acquisition systems that have been in use with PDP 11 computers since the earliest days of CCDs on Kitt Peak (circa 1980). The astronomer uses ICE to initiate an exposure using basically a single command, **observe**. At the end of the integration, the data are written to disk in IRAF format, where they can be taped in FITS format. Furthermore, the astronomer has immediate and full access to the many powerful diagnostic tools contained within IRAF, as well as routines for quick-look and data reduction.

The IRAF package for taking CCD data is actually called **ccdacq**; it was originally written by Skip Schaller at Steward Observatory, and subsequently adapted to run the Kitt Peak controllers with various improvements to the user interface as we gained experience with the system.

Additional resources you may wish to review are:

- *A Beginner's Guide to Using IRAF* by Jeannette Barnes
- *Notes for using X Windows on Kitt Peak* by Rob Seaman
- *A User's Guide to CCD Reductions with IRAF* by Phil Massey

These documents are all available via anonymous *ftp* to "ftp.noao.edu", and can be found as "/iraf/pub/beguide.ps.Z", "/kpno/manuals/ice.xnotes.ps.Z", and "iraf/iraf/docs/ccduser2.ps.Z". (For more on anonymous ftp and retrieving manuals, see Sec. P.) These manuals can also all be accessed via the Kitt Peak home page: *http://www.noao.edu/kpno/kpno.html*.

2 Observing Overview: Taking the Data

2.1 The Only Command You Really Need: *observe*

With ICE, one takes data with a single command: **observe**. The astronomer will be prompted for the information necessary for controlling the exposure. This includes the following:

- **Image type:** Can be "object", "zero", "dark", "flat", "comp", "focus", or "ramp". (Note that IRAF refers to a "bias" as a "zero". "Ramp" is a new feature that will be described in Sec. M.)
- **Exposure time:** If the object type is anything other than "zero" (which always has a zero-second exposure time), then the user is prompted for the integration time.

- **Object title:** This is the title put in the image header.

In addition, the user may choose to be prompted for the required filter position, and the user may also choose to be be prompted for the required telescope focus setting. (How to do this will be explained when we discuss the **obspars** parameter file in Section 3.3.) If so, the filter wheel will be moved to the requested position and/or the telescope set to the requested focus setting before continuing.

An example of taking a 300 second "object" exposure is shown in Figure 1. Note that in each query the user is also presented with a default value that s/he could invoke simply by hitting [CR]; these default values are just the previous entries. Once the title is entered, the user is informed what the name of the image will be; this name consists of a root ("n1" in this example), plus a number which is automatically incremented on each exposure until reset by the user.

If all goes well, the user will be given a one-line "status" report, then told that the chip is "preparing", followed by "exposing". If the exposure is long enough, the astronomer is also shown both the time left to expose and the amount already exposed; these update at ≈ 5 sec intervals, as do the temperature of the CCD (Tcam) and the temperature of the dewar (Tdew). (The dewar temperature will begin to increase if the liquid nitrogen is used up; keep an eye on this number and call for help if it goes up by two degrees.) Finally, the user will be informed when the chip is reading out. Read-out time for our chips varies, but can be several minutes for the largest, so be patient. Finally the user is told that the image has been written to disk.

NOTE: All observing commands must be issued from the "DATA ACQUISITION window" and **NOT** *from the "DATA REDUCTION" window. We will describe these windows in Section A.2.*

The actual steps that occur during an "observe" sequence are listed in Sec. N

2.2 Doing More with *mores*

Did you like that last exposure and want to do some more just like it? There is a command to help you out: *mores*. To do two more exposures with the same parameters (image type, filter setting, telescope setting, exposure length, title) of the previous **observe** command, do a

mores 2

and you will be told what images are being written to disk.

2.3 Changing Things On the Fly

ICE allows you to pause ("p"), resume ("r"), stop ("S") and abort ("A") an exposure in the midst of an integration. Furthermore, you can change the integration time ("x"), or the title

```
cl>observe
Image type (object|zero|dark|flat|comp|focus|ramp) (comp): object
Exposure time (seconds) (0.:16200.) (60.): 2000
Object title (comp 685): M33f8a
image 'n10027' will be written to disk
T2KB  [1:2048,500:1500] bin=[1,1] gain 3.2 dewtemp -162 tel-link=up
preparing...
exposing...

    Time left  Exposed  Tcam Tdew

exposure = 2000.0 seconds (0:33:20.0)
darktime = 2000.0 seconds (0:33:20.0)
reading out...
image 'n10027' written to disk
```

Figure 1: Making a 2000 second exposure of type "object".

("t") during an exposure. Since each command is a single key-stroke command (no carriage return is needed), accidentally typing in the data acquisition window during an exposure could have serious consequences. (For instance, typing a **flpr** with the mouse positioned in the acquisition window would **pause** ("p") and then **resume** ("r") an exposure!) Thus we require an exposure to be **paused** before doing anything else, and the commands that terminate an exposure, stop and abort, require capital letters ("S" and "A").

2.3.1 A Momentary Flutter: How to Pause and Resume an Exposure

Consider the unlikely situation that clouds have drifted overhead during an exposure. You may wish to close the instrument shutter until that single cloud moves by. To **pause** the integration simply type **p** in the acquisition window. (The exposure must be long enough that you are seeing the updating "Time left" message, i.e., > 5 secs.) You will see a "PAUSED" message printed next to the dewar temperature.

To **resume** the exposure, simply type an **r**. The shutter will open and the time will begin updating again.

2.3.2 Terminating an Exposure Early: Stop and Abort

You may decide to **stop** an exposure early. First you must **pause** the exposure using **p**; you can then type a **S** to stop. (Note that the "S" must be upper-case.)

Or you could decide you don't want the current exposure at all. In order to **abort** the exposure, first **pause** the exposure with a **p** and then **abort** the exposure with an **A**. The image will be lost, and the exposure counter will not be updated. (Note that the "A" must be upper-case.)

2.3.3 Changing the Exposure Time

It is also possible to change the exposure time in ICE. You must first **pause** the exposure using a **p**. Once paused, you may change the exposure time by striking the **x** key. You will be queried for the new exposure time. Don't forget to follow this with an **r** to **resume** the exposure.

Note that while you can always *shorten* the length of an exposure, you can only *increase* the length of an exposure if the initial exposure length was longer than 5 mins (by default). The user should be warned that an exposure time that has been *changed* to 30 secs, say, will not be as accurately timed as an exposure that began and ended as 30 secs. This is due to the fact the Unix system, rather than the CCD controller, keeps track of the time if the exposure time is altered. We are uncertain, currently, what sort of error this may introduce; the most pesimistic estimates are 1 or 2 seconds regardless of exposure time. Thus this method should be fine for long exposures, but may be dicey if 1% photometry is needed for short exposures! We recommend not altering the exposure time if 1-2 sec errors in the subsequent timing would affect your results.

2.3.4 Changing the Title

Once you are **paused**, you can even change the title of the exposure. (Of course, you *are* losing valuable observing time while you are exercising this feature!) First type **p** to pause, and then type **t**. You will be asked for the new title. Don't forget to **resume** the exposure with an **r**.

2.4 The Amazing Power of *flpr*

If you've gotten an error message, or got out of some parameter file in a funny way, or typed a **CNTL-c** or done anything at all out of the ordinary, you may well find that things are just a little bit funny. The safest thing to do is to type **flpr**. (This will "flush the process cache.") It doesn't hurt to type this twice, and sometimes it helps. If **flpr** itself produces some strange message that doesn't clear itself with another **flpr**, you may need to log out of ICE/IRAF completely (see Section A.3).

2.5 How to Get Help

IRAF comes with on-line documentation that gives the nitty-gritty of each command. Simply type **help mores** to see the help page for the **mores** command. You can get a hard-copy of this help page by directing the output to the laser writer:

help mores | lprint

2.6 Other Useful Observing Words

Although **observe** is the only data-collection word you need to learn, there are a few others that you may find useful.

2.6.1 Doing a Sequence of Observations

We saw above that we can use **mores** n to do n more observations just like the last one. But what if you know exactly what you want to do in the first place and would just like to start a series of 20 bias frames running? There are specialized tasks that will do a series of observations of each image type, namely:

- **zeros**
- **objects**
- **flats**
- **comps**
- **darks**
- **ramps**

Each of these commands will query for the number of exposures, exposure time, and title. Note that **mores** will repeat more exposures of sequences begun using any of the above commands.

Also note that if you change the exposure time during a series of integrations, this change will propagate to subsequent exposures of the series; this is not true of changing the title, at present, however. **Stopping** or **Aborting** an exposure will terminate the series.

2.6.2 Running a Test Exposure Using *test*

In addition, it is possible to do a **test** exposure that will always overwrite an image named **test** by typing **test**.

3 The Parameters Files

ICE has made it easy for you to observe by providing a single word for data-collection: **observe**. However, this ease has not come at the cost of flexibility; instead, all the options have been buried in the parameter files. There are four of these parameter files which must be properly set, but which you will likely leave alone throughout the course of your run:

detpars This parameter file controls the fundamentals of how the CCD **det**ector is read. All items in this parameter file are crucial for correct operation of the CCD.

instrpars This is the **instr**ument parameter file, and at most telescopes is currently used by ICE only to add information to the picture headers. If you wish to use the scan-table at the 4-m PFCCD, however, **instrpars** must be modified by setting "instrument" to "pfccd".

obspars This is the parameter file that is used by the astronomer to tailor the observe command to his/her liking, and to convey to the software fundamental aspects of how the observations, particularly focus frames, are to be executed.

telpars This parameter file is used only to specify the protocol needed to communicate with the telescope computer in order to allow proper transfer of header information (time, telescope position and particulars) to ICE.

These parameter files can each be listed using the **lpar** command, e.g.,

lpar detpars

and may be edited using the parameter editor **epar**, e.g.,

epar detpars

(In fact, you can enter the parameter editor for **detpars** by simply typing **detpars**.) To change a value in the parameter editor move the cursor up and down with the arrow keys until you are on the correct line, and then simply type the new value, followed by [CR]. When done editing a parameter file, type **CNTL-z** (or **CNTL-d**, depending upon what you chose for an eof character when you ran *obsinit*; see Section B).

3.1 *detpars*

The detector parameter file **detpars** allows the astronomer to specify how the chip is to be formatted, how much preflash is to be used, a number specifying the gain, and the chip identification so that the correct microcode is loaded. An example is shown in Figure 2.

- **detname**: The most important entry in **detpars** is the chip identification. This should have been set automatically when you ran *obsinit* at the beginning of your run and specified which chip you were going to use. (See B.) However, it is a good idea to check this, as the chip designation controls what microcode is sent to the CCD controller, and you will be quite surprised at the results if the wrong microcode is loaded. ("detcap" specifies the pathname of where this name is used to find the microcode.) In principle, you will be warned when you take an observation if the chip

```
cl> lpar detpars
    (firstcol = 1)              First column of data (device coordinates)
     (lastcol = 2048)           Last column of data (device coordinates)
    (firstrow = 500)            First row of data (device coordinates)
     (lastrow = 1500)           Last row of data (device coordinates)
      (colbin = 1)              Column binning factor
      (rowbin = 1)              Row binning factor
    (preflash = 0)              Preflash time in seconds
        (gain = 4)              Instrumental gain setting (0 for default)
     (detinfo = "")             Optional image header info about detector
      (detcap = "runlib$detcap") Detector capabilities file
     (detname = "t2kb")         Detector name
        (mode = "ql")
```

Figure 2: Parameter file **detpars**.

name does not agree with the name returned from the chip itself. The current chip designations are basically the lower-case version of what is shown on the observing schedule, e.g., "t5hc", "t1ka", "t2ka", "t2kb", "s2ka", "f3kb", "gcam", "cryo", or "ti5".

- **firstcol, lastcol, firstrow, lastrow, colbin, rowbin**: These parameters describe the "active" area of the chip that you wish to use, and whether adjacent pixels should be binned or not. For most applications, the chip will be formatted to full size, and no binning used, but in the example shown the astronomer is using the 2048×2048 T2KB chip for spectroscopy, and only needed to read out a thousand rows near the center. Note that all the Kitt Peak chips get an extra 32 columns of overscan in order to monitor the electronic pedestal level, and hence in the example shown the images will be ($2048+32 = 2080$) by 1001 pixels. In the case that you do choose to bin the data, the first and last columns and rows refer to that of the *unbinned* pixels.

- **to preflash or not to preflash**: You will only ever have to preflash if you are using a TI chip, in which case you will need to specify the number of seconds of preflash you need—typically 3 to 5. This should be adjusted (along with the "intensity" digit-switches on the CCD electronics box) until you get 50-100 electrons per unbinned pixel for the TI chip (200-400 electrons if you are binned 2×2).

- **gain**: All our CCDs other than the TI chips have adjustable gain settings. The gain parameter in **detpars** is an ordinal number that allows you to change the number of electrons per ADU (Analog to Digital Unit, the numbers you see on the screen). The correspondence between these ordinal settings that you enter in detpars, and the corresponding number of electrons per ADU, can be found typing **ccdinfo** as long as the proper CCD identification is in **detpars**. This can also be found by using **dcapinfo** (for instance, **dcapinfo t1ka**).

Many of the parameters that the user is likely to change are displayed on the "status" line that appears at the beginning of each integration: the chip name, the format and binning factors, and the gain.

3.1.1 Digression: How To Choose A Gain

"How do I choose a gain setting?" is one of the most asked questions. All of our chips (other than ti5) have gains which are adjustable by the software. Why would you want to adjust the gain? The Tektronix chips all have a full-well capacity of >100,000 e^-, and some as high as 250,000 e^-, before there is any detectable deviation from linearity. However, the A/D converters are limited to 16-bits, and cannot output data that is greater than 65,535. Thus to make full use of the dynamic range you would like the gain factor to be about 4. Why not simply set it to 4? The reason is that most of our chips also have very low read-noise, and thus if the gain is greater than the read-noise, you will be undersampling the read-noise—in effect, increasing the read-noise to the level of the gain simply because of digitization noise. (You can't very well recover a read-noise of $3.5e^-$ if each data unit is equivalent to $4e^-$.) So like most things in life, there are trade-offs. Up-to-date values of the read-noise and linearity limits for all of our chips can be found in the CCD Characteristics Manual in each dome.

This was more of a problem before the new 16-bit A/D converters were implemented; now, most users will be happy by just accepting the default gain settings (see Sec. Q).

If you are attempting to do 1% stellar photometry of stars in a cluster, you are probably interested in covering as large a magnitude range as possible, and furthermore, your noise is going to be primarily photon-noise, not read-noise. Go for the largest value of e^- per ADU as you can without exceeding the linearity of the particular chip. Generally, this will be the default gain, which is also the gain that will give you the least amount of horizontal bleeding from very saturated stars.

If you are doing surface brightness studies of objects through narrow-band filters, and the read-noise is significant but the dynamic range of your objects is limited, you may wish to stay with the largest gain number (smallest number of e^- per ADU). Similarly in some very low-signal spectroscopic applications you are limited by the read-noise.

Note: Some of our CCDs show some very low-level "streaking" to the right of the most heavily exposed stars. This problem is significnt with S2KA and T2KB. The electronics has been adjusted to minimize this problem for the default gain setting. If you are concerned about the effects of very saturated stars on your direct imaging data, you would do well to stay with the default gain settings.

```
cl> ccdinfo
    (firstcol = 1)             First column of data (device coordinates)
     (lastcol = 2048)          Last column of data (device coordinates)
    (firstrow = 500)           First row of data (device coordinates)
     (lastrow = 1500)          Last row of data (device coordinates)
      (colbin = 1)             Column binning factor
      (rowbin = 1)             Row binning factor
    (preflash = 0)             Preflash time in seconds
        (gain = 4)             Instrumental gain setting (0 for default)
     (detinfo = "")            Optional image header info about detector
      (detcap = "runlib$detcap") Detector capabilities file
     (detname = "t2kb")        Detector name
        (mode = "ql")

T2KB  --   2048 columns,  2048 rows, pixel type = short

old dwell value                    (dw)    15
read noise in electrons/adu        (rn)   4.0
gain instrumental settings (dwell)   (gi)     1     2     3     5     8    15
gain values in electrons/adu       (gv)  10.7   8.2   5.4   3.2   2.0   1.1
default gain index if detpars.gain=0 (gd)                             5
SELECT AN INDEX FOR DETPARS.GAIN FROM:       1     2     3     4     5     6

camtemp     = -85
dewtemp     = -162
```

Figure 3: The command **ccdinfo** produces a listing of the current parameters in effect on the chip, and shows what ordinal number to use in **detpars** to set the gain to a particular number of e^{-}/ADU.

3.1.2 Digression: The Power of *ccdinfo*

Once your **detpars** is setup, you can verify all of these important parameters by simply typing the command

ccdinfo

You will get a response that resembles that of Figure 3.

If you ever wish to get information about all the (major) chips available, one can do a **dcapinfo show=list**. Information about a particular chip can be found by typing **dcapinfo t1ka**, for example.

```
cl>lpar instrpars
(instrfilters = "")             filter bolt positions
    (aperture = "300um slit")   aperture
      (tvfilt = "BG-39")        tv filter
    (complamp = "")             comparison lamp
    (probepos = "")             probe position file
   (disperser = "KPC-022")      disperser
     (tiltpos = "4435.5")       tilt position
      (decker = "")             decker
  (instrfocus = "")             instrument focus
    (posangle = "")             position angle
    (dispaxis = "1")            dispersion axis
         (fts = "")             filter translation
  (scanoffset = "")             scan table offset (tenth microns)
    (scanstep = "")             scan step size (tenth microns)
   (instrinfo = "")             Optional image header info about instrument
    (instrcap = "runlib$instrcap") Instrument capabilities file
   (instrname = "test")         Instrument name
        (mode = "ql")
```

Figure 4: The rather uninteresting parameter file **instrpars**. If you are using the scan-table at the 4-m, set *instrname* to *pfccd*; otherwise, set this to *test*.

3.2 *instrpars*

This parameter file currently has two uses: (a) to allow the observer to easily enter some items automatically into the header, and (b) to enable the "scanning" with the PFCCD scan-table at the 4-m. If you are using the scan table the paramater **instrname** should be set to "pfccd"; otherwise, it should be set to "test" (Figure 4). You may enter whatever you wish into the other parameters; these will make their way into your headers. Spectroscopists might find it particularly useful to specify if the dispersion axis is along rows (dispaxis=1) or along columns (dispaxis=2) to ease the reduction process.

3.3 *obspars*

The parameter file *obspars* allows the astronomer to specify details of how the images will be named, whether or not s/he wishes to set the filter and/or telescope focus for each exposure, details of how focus frames should be taken for direct imaging, and additional comments for the header. A sample **obspars** can be seen in Figure 5. An inspection of Figure 5 can be a little misleading—it is *not* necessary to change any of the first nine items in the parameter file, as the user will be prompted for these whenever appropriate. The

```
cl> lpar obspars
exposuretime = 1.               Exposure time (seconds)
    imagetype = "object"         Image type
  objecttitle = "m31ob48a"     Object title
       nfexpo = 7               Number of focus exposures
       shtype = "detector"      Shift type
      focmode = "manual"        Focus mode
       fstart =                 Starting focus value
       fdelta = 0               Focus increment
      nrvrows = 30              Number of rows to reverse shift
    (rootname = "n3")           Image root name
    (sequence = 1)             Sequence number
  (setfilters = no)             Query and set filters?
    (setfocus = no)             Query and set focus?
 (setscanrows = no)             Query and set nscanrows?
   (nscanrows = 1)              Number of scan rows (1 disables scanning)
     (filtype = "telescope")    Type of filters to use
     (foctype = "telescope")    Type of focus to use
     (pixtype = "s")            Data type of IRAF pixels
   (observers = "Massey")       Observers
      (propid = "1440")         Observing proposal ID
    (comments = "")             Comments
     (comfile = "")             Observer header comments file
     (obsinfo = "")             Optional observing information for image header
 (observatory = "KPNO")         Observatory name
     (command = "postproc %s &") Postprocessing command
 (preallocate = 1)              Preallocate image (0=no 1=yes N=if exptime > N)
   (preprefix = ".")            Preallocate image prefix
    (longexpo = 300.)           Long exposure time (seconds)
     (verbose = yes)            Type out image name?
       (debug = no)
        (mode = "ql")
```

Figure 5: An example of **obspars** is shown.

values listed there at any time are simply the most recent entries, and is what ICE uses to offer as default values when you hit [CR], as you may remember from the previous example of using **observe** (refer back to Fig. 1). Parameters that you may wish to alter:

- **rootname** and **sequence**: These control what the next picture will be called, e.g., n30001 in this example. You may wish to change the rootname from night-to-night (see Section C below on "Keeping Your Data Organized"), but remember to keep this short—you will have to type this every time you refer to the image. The **sequence** number allows you to reset the picture counter to whatever number you want, although you must first explicitly get rid of any images with the same name using **imdelete**.

- **setfilters** and **setfocus**: These two parameters control whether the user will be prompted for the correct setting of the filter and/or focus value at the beginning of each exposure. The filter parameter has proven to be invaluable for direct imaging; it both reminds the astronomer what filter is currently in place, and offers an opportunity to change it at that time. Since different focus values are sometimes required for each filter, it is also sometimes useful to set the focus prompt-and-set as well; otherwise you (or the telescope operator) may set the telescope focus manually.

- **setscanrows**: This parameter controlls the "short-scan" option with the scan-table at the 4m P/F. A new scanning table, capable of contending with the 2048 chips, is now in use, and ICE makes using this option fun and easy to use. Set **setscanrows** to "yes" in order to be queried for the number of rows to scan ("1" is equivalent to not scanning). Note that the instrument name must be set to "pfccd" in **instrpars** for scanning to work.

- **nscanrows**: This parameter determines the number of rows you are scanning if you are using the scan table at the 4-m. If this parameter is "1" the scan table is not used. Note that the instrument name must be set to "pfccd" in **instrpars** for this parameter to actually do anything.

- **observers** and **propid**: These are put in the image headers and are used by the Save-the-Bits data archive for record keeping. (See Sec. 4.3.)

- **comments** and **comfile**: These two parameters allow you to add even more information to your header. The parameter **comments** may be edited to contain a single line of additional comment. The parameter **comfile** may contain a file name that contains multiple lines of comments. Each line of this file is automatically formatted into a FITS COMMENT record. You should not include the COMMENT keyword in this file. Since this file is read at the beginning of the read-out, you may edit this file any time during the exposure. The **comfile** is automatically set if you are using the automatic logging software (Sec. L).

- **command**: This is a very powerful (and potentially dangerous) feature: any IRAF task placed here will be executed immediately after the chip is read-out. The default is to execute the script *postproc.cl* in background mode. This file will causes the terminal to "bleep" at the end of the integration, and will automatically display the new image in the ximtool window and activate the automatic logging, if in use (see Sec. L. You can edit the *postproc.cl* file to execute any command you so choose, but be careful to add new commands only at the end of the file. One *could*, in principle, automatically tape each new exposure and then, say, run the data through **ccdproc**, but we are not recommending this.

- **preallocate**: This switch controls whether or not diskspace is "reserved" for your image. If there is insufficient space for the new image at the beginning of the integration, a warning will be issued and the exposure won't begin. If there is sufficient space at the beginning of the exposure, **preallocate** guarantees that space will be there at the end of the exposure; it accomplishes this by writing a dummy image of the same size but with a "." stuck in front of its name in order to make this an "invisible" file. Upon read-down, the new data replaces the old, and the image is renamed. There is a slight time penalty incurred in preallocated disk space. Thus, rather than insisting that **preallocate** always be on or off, the parameter can be set to a minimal integration time for preallocation. The default value is 60, meaning that preallocation will only be in effect for integration times greater than 60 seconds. Setting this to 1 will result in preallocation happening for *all* images; setting this to 0 turns off any disk space checking.

- **longexposuretime**: This parameter determines whether the Unix system is controlling the length of the exposure, or whether it is controlled by the CCD controller. Historically the exposure times have always been controlled by the CCD controller; this time should be extremely accurate. However, in order to implement the ability to *change* the exposure time, it was necessary to let the Unix system take over the timing. The default value of 300 seconds means that for exposures longer than 300 seconds, the Unix system is doing the timing; this is believed to be good to ± 1 or 2 seconds (regardless of exposure time) but we don't know this for a fact, yet. Exposure times shorter than this value cannot have their exposure times lengthened. If you are never going to lengthen your exposure times "on-the-fly" (and why would you?) you might want to set this to a length greater than your longest exposure time; this will assure accurate exposure lengths.

3.4 *telpars*

With the exception of the Burrell Schmidt, all the Kitt Peak telescopes are run by computers that can communicate with the ICE/IRAF computer to pass along header information giving the time, telescope position, and telescope particulars at the beginning of the integration. However, for this to work, ICE must know the name of the telescope you are using. This is automatically set for you at the beginning of your run by running *obsinit*, and is based upon which SUN workstation you are using. These names are, at present, "kp09m", "kp21m", "kpcdf" (note the middle "d"), "kp4m", "kpmm" (for McMath), and "kpschmidt". Setting the telescope name to "test" allows you to take data when the telescope computer is down or otherwise incommunicado; of course, you won't get any telescope header information in this case. In the case of the Schmidt the only header information you will get is the fitler position and the local siderial time. (You could, if you felt so inclined, **epar telpars** before each exposure at the Schmidt and enter whatever of the header information you would like to preserve.) An example **telpars** is shown in Figure 6.

4 Writing Your Data To Tape

4.1 Making a FITS tape

So now that you have some data, you probably want to save it to tape. At each telescope there are both Exabyte and DAT tape drives; there are also a few 9-track dinosaurs still roaming some of the domes. Since all of our ICE systems subscribe to IRAF networking, and you can in fact use any tape drive in any dome, although you need to check with the appropriate astronomer if you want to use someone else's tape drive—be nice!

The first step is to allocate the appropriate tape drive. The 1600/6250 bpi drives are known as **mta** to IRAF; the Exabytes are known as **mtb**; and the DAT drives are known as **mtc**. (You can find these names printed on each drive unit; you can also run the command *devices* to list both the IRAF and Unix names of all the I/O devices on the mountain.) Thus you need to do an

allocate mta

to allocate the 1600/6250 bpi tape drive at your telescope; to allocate a drive elsewhere you might do something like a

allocate lapis!mta

The parameters for **wfits** are shown in Figure 7. The crucial parameters are:

- **iraf_files** : This refers to the image name(s) that you wish to save, e.g., "a0002,a0005" if you wish to only save those two files. Instead, you can also substitute an "at" file here (such as shown in the example). An "at" file is simply a text file that contains

```
cl> lpar telpars
     (dateobs = "")             date (dd/mm/yy) of observation
          (ut = "")             universal time (hh:mm:ss)
          (st = "")             sidereal time (hh:mm:ss)
          (jd = "")             Julian date
          (ra = "")             right ascension (hh:mm:ss)
         (dec = "")             declination (dd:mm:ss)
       (epoch = "")             epoch of ra and dec
          (ha = "")             hour angle (hh:mm:ss)
          (zd = "")             zenith distance (dd:mm:ss)
     (airmass = "")             airmass
    (telfocus = "")             telescope focus
  (telfilters = "")             filter bolt positions
    (rotangle = "")             rotation angle
    (pressure = "")             barometer
     (teltemp = "")             telescope temperature
   (windspeed = "")             wind speed
(winddirectio = "")             wind direction
    (humidity = "")             humidity
      (seeing = "")             seeing
     (telinfo = "")             Optional image header info about telescope
      (telcap = "runlib$telcap") Telescope capabilities file
     (telname = "kp4m")         Telescope name
        (mode = "ql")
```

Figure 6: The rather uninteresting parameter file **telpars** contains only one crucial entry, the telescope name.

```
cl> lpar wfits
   iraf_files = "@savethisstuff" IRAF images
   fits_files = "mta6250"       FITS filename
      newtape = yes             Blank tape?
       bscale = 1.              FITS bscale
        bzero = 0.              FITS bzero
  (make_image = yes)            Create a FITS image?
 (long_header = no)             Print FITS header cards?
(short_header = yes)            Print short header?
      (bitpix = 0 )             FITS bits per pixel
(blocking_fac = 0)              FITS tape blocking factor
       (scale = yes)            Scale data?
   (autoscale = yes)            Auto_scaling?
        (mode = "ql")
```

Figure 7: Parameters for **wfits**.

the names (one per line) of each image. You can easily create such a file by **files *.imh > savethisstuff**.

- **fits_files** This refers to the tape drive you wish to use. In the case of the Exabytes or DATS this can simply be the same as the device name that you used in **allocate**, but for the 1600/6250 bpi tape drives, you should explicitly specify the density you want by referring to it as either **mta1600** or **mta6250**.
- **newtape** This is, as you may well imagine, a critical parameter to get right. If you want to start at the beginning of the tape, writing over any data that might be present, then answer **yes** to this; otherwise it will search for the end of the data on the tape before writing.
- **blocking_fac** : The default, 0, results in "maximum" blocking (10) for both Exabyte and 9-track tapes. If you are planning to read the data with a non-IRAF FITS reader, you may want to reset this to 1; otherwise, leave it at 0 for maximum tape writing efficiency.

In order to check to see what is on the tape, you can list the titles quite easily. Simply do a

rfits mtb 1-999 make– short+ old+

to see what's there. To direct this output into a file, you can add a > **tapelist** to the end, and then you can print that list on the lineprinter by a simple **lprint tapelist**. Alternatively, you could choose to read a single file back onto disk to examine:

rfits mtb 15 junk make+ old+

will create an image with the original name. (Note that even if you specify **old+** you must still give the name of a legal temporary file in the output file position, i.e. *junk* in this example.) If the same image name already exists on disk, you should instead use **old-** to create an image named *junk*, or whatever you wish.

As you write your tape, you will get a file-by-file account of what is going on the tape. When you come to the end of the tape, you will find that there is a message saying that the file was NOT completely written to tape. An easy way to continue at this point is to edit your "at" file to remove the names of the files that were successfully written to tape. Rewind the tape with **rew mtb**, mount the new tape, and then re-run **wfits**, remembering to set **newtape=yes**.

The new "x" IRAF tool "xtapemon" will allow you to see how much space is left on your tape, and to keep tabs on where on the tape you are; see Sec. A.2.

NOTE: If you do write additional files to an "old tape" (one containing useful data but which had previously been removed from the drive), make certain that the software (IRAF and Unix) is aware that the tape has been rewound before starting to write to the tape—-or your old data may be overwritten! To safegaurd against this possibility we suggest that you *ALWAYS* swap tapes by first:

- **deallocate mta** (or mtb or ...)
- Physically swap tapes
- **allocate mta** (or mtb or ...)

4.2 Safe Taping

NOTE: Prior to Version 2.10.1 of IRAF arriving on the mountain, we experienced a number of unpleasant difficulties with tape writing. With the current version of IRAF and ICE these problems appear to have been solved. However, these troubled times reminded us all of the importance of "safe taping" practices. We remind the observer that s/he, and s/he alone, is responsible for keeping one's bits safe. We recommend the following before deleting any data from disk:

1. Each night write data to whatever media you like using **wfits**.
2. Read the tape using **rfits mta make- old+** to substantiate everything is there.
3. Deallocate the drive, remove the tape, and *stick it under your pillow.*
4. Make a second copy of your tape. (This tape could be an accumulative copy of the data throughout your run.) Check this tape with **rfits**!
5. Only now delete the data from disk if necessary.

You may also wish to use alterative means of getting your data home. See Section D.

4.3 Save-the-Bits!

Starting in the Fall 1993 semester all CCD images have been "archived" onto Exabyte tapes. (This typically amounts to 1-2 Gb *per night.*) This program should not encourage lax taping procedures, and we strongly emphasize the need for the "safe taping" procedures above. But if you ever do need to recover a night's worth of data, take heart! You can contact Ed Carder if you need this service (*ecarder@noao.edu*).

```
cl> observe
Image type (object|zero|dark|flat|comp|focus) (zero): focus
Number of focus exposures (0:4096) (7):
Shift type (detector|telescope) (detector):
Focus mode (auto|manual) (manual):
Number of rows to reverse shift (0:4096) (30):
Starting focus value: 1200
Focus increment (0): 20
Exposure time (seconds) (0.:86400.) (1000.): 10.
Filter bolt positions: 3
Object title (biases, 2nd afternoon): focus V filter
image 'a0021' will be written to disk
T2KA  [400:1650,1:2048] bin=[1,1] gain 5.6 dewtemp -158 tel-link=up
preparing...
Telescope focus (1200):
focus = '1200' exposing...
Telescope focus (1220):
focus = '1220' exposing...
Telescope focus (1240):
focus = '1240' exposing...
Telescope focus (1260):
focus = '1260' exposing...
Telescope focus (1280):
focus = '1280' exposing...
Telescope focus (1300):
focus = '1300' exposing...
Telescope focus (1320):
focus = '1320' exposing...
reading out...
image 'a0021' written to disk
    "REMINDER: The double space marks the beginning of the sequence!"
```

Figure 8: A focus sequence of seven 10-sec exposures, centered at a focus of 1260, with a double space between the focus=1200 and focus=1220.

5 Focusing the Telescope for Direct Imaging: Don't Touch That Hand-Paddle!

In the old days, we used to take a focus plate by taking a multiple exposure in which the telescope focus and position were incremented about 5-10 times. With CCDs, we can do the same thing, but we don't have to run downstairs to the darkroom. Furthermore, with ICE, we can eliminate another process in the chain. We no longer need to increment the telescope position: the software commands the detector to "reverse vertical shift" its charge between successive exposures, effectively simulating a stepping of the telescope. **The FIRST exposure in the sequence is given a double shift for easy identification.**

You can initiate a focus sequence the same as any other observing command, simply by using **observe** and selecting "focus" as the image type. An example is shown in Figure 8. In this example, the astronomer has chosen to make seven exposures, shifting the charge on the CCD between each exposure ("Shift type=detector"). S/he has chosen to *manually* set the focus between each exposure, rather than relying on the telescope computer to automatically set the focus to the next appropriate value. The shifts were each of 30 rows, except that a double shift (60 rows) will be found before the last exposure. The focus was stepped by 20 units from a focus reading of 1200 to 1320. (These numbers are telescope depedent; see the table below.) Before each exposure the astronomer was prompted with the focus number and the routine waited until the astronomer hit [CR]. Note that in order to carry out this focus sequence, it was not necessary to alter a single parameter in any of the parameter files; all the necessary items were requested. The **setfilter** flag in **obspars** was set to yes, and for that reason it was possible to change the filter to position 3 before the exposure began.

After the multiple exposure series is complete, look at the images on the display to select the optimum focus setting of the telescope. There is a newly implemented routine **kpnofocus** specifically tailored for analyzing focus frames obtained with ICE; simply run **kpnofocus** and use the "g" key to mark the top-most image for a few well-exposed stars in the focus sequence (see Section J.1). In addition, the general-purpose image examination routine **imexamine** is described in Section G.

Typical Focus Steps

Telescope	V Focus	Step
4m P/F	2300	+5
2.1m	22125	+25
0.9m f/7.5	4600	−5
0.9m f/13	5320	+5
Schmidt	−200	+25

6 Summary of Useful IRAF Commands for Quick Look

This section has no pretensions of being a tutorial to IRAF, but it is our intention to call your attention to or remind you of some of the more useful IRAF commands for examining your data. Use the IRAF **help** command and/or poke around to become familiar with these.

diskspace	check remaining space on disk
dir	lists the files in current directory
imcopy	copies an image, or subraster of an image, to another image
imrename	renames an image
imdelete	delete images
imheader	list image headers
hedit	edit the header parameters of an image
hselect	list specific items from the image headers
display	displays an image on the ximtool window
imexamine	general image analysis utility (row and column plots, radial profiles, surface plots, contour plots, image statistics)
imhist	histogram the pixel values in an image
imstat	detailed statistics on an image
implot	plot rows, columns, and averages of these
splot	plot a row from an image with spectral analysis tools
imarith	arithmetic operations between images $(+, -, *, /)$
imcombine	combines images (such as averaging or median)
allocate	allocates a tape drive
deallocate	deallocates a tape drive (necessary in order to logout)
devstatus	reveals what IRAF thinks about the tape
devices	lists all the I/O devices and Unix names
wfits	write a FITS format image to tape or disk
rfits	reads a FITS format image from tape or disk
mtexamine	examines a magnetic tape and divulges interesting facts
files	generate a list of files (useful with wfits)

A Logging In, What It Looks Like, and How to Get Out

A.1 Logging in and Firing up ICE and IRAF

The relevant observer login name and the password are posted on each terminal. Once you are logged in, the windows system will be automatically started, and IRAF loaded.

A.2 What Windows Go Where: What Your Desktop Looks Like

Once past this hurdle, the astronomer will find several windows have been opened. On the left is a grey window labeled "Data Acquisition" and on the right a white window labeled "Data Reduction". IRAF is running in both of these windows, but different packages have been automatically loaded. (Note also that the parameter files are kept *separately* for the two windows; this has been known to cause some confusion.) You must run ICE/CCD commands from the "Data Acquisition" window, and this window alone. An ximtool picture window will appear. Finally, you will find a number of icons appearing at the bottom: the console window, a clock, a cpu performance meter, "xtapemon", "Mosaic", and a "Desktop". The console window will be open, and it is a good idea to check it from time-to-time; this is where any error messages (such as tape drive problems) will appear. Note, though, that you can not type into this window.

Beginning in the summer of 1994 we switched our window system from "sunview" to an "openwindow" ("X"-ish) system that differs only slightly from sunview. Most changes will be transparent. The new display window is now "ximtool" and is even nicer than the old one: for more on using "ximtool" see Sec. F. There is a guide called "Notes for using X Windows on Kitt Peak" found in every dome; this will walk you through manipulating windows if you find you don't guess right. The rudimentary manipulations remain unchanged: To uncover a window move the mouse to the general area and press the L5 key. You can "close" a window (not the same as quitting) by moving the mouse onto the window and hitting the L7 key. To open a window, place the mouse on the icon and hit L7. "Double-clicking" on the closed icon will accomplish the same thing.

One very useful feature is the ability to change your Data Acquisition window or your Data Reduction window to a *scrolling* window, allowing you to pan up and see what it is you did previously. Simply place the mouse in the window, and hold down the CNTL key while pressing the right-most mouse button. Select "Enable Scrollbar".

If you need to create a second Data Reduction window (because the first one is tied up doing something else) you can do this by moving the mouse to one of the blank areas, holding down the right-most mouse button, and selecting "Data Reduction".

(You could also start IRAF from a "Unix xgterm" window. However, because the parameter files are kept separately for the Data Acquisition task and the Data Reduction tasks, you would first need to do a **cd reduce**, followed by a **cl** followed by a **cd** to return to the main directory. If you ever start up IRAF and see the message "Warning: no login.cl found in login directory", you're in trouble. Log out and start over.)

A.3 Logging Out

The cleanest and safest way of exiting is the following:

1. Type **logout** in each of your IRAF windows.
2. Move the mouse to a blank area of the screen and hold down the right mouse button; select "Quit Windows".

We recommend that you log out at least once a day. Unix memory management is such that allocated blocks of memory are never truely freed for reuse until a process dies. IRAF, on the other hand, will arrange to keep a process around indefinitely (even after a task exits) to avoid the rather stiff startup penalities that sometimes apply. This can become a problem if an observer runs a memory-hungry task, such as **imcombine**, over and over for several days.

A.4 Two Monitors?

At the 4-m, 0.9m, and Schmidt there are actually *two* monitors that can be used; by default the ximtool comes up on the second window to save valuable display real estate.

B At The Very Beginning of Your Run: *obsinit*

At the very beginning of your observing run, you should clean off all of the previous observer's images and files, and reinitialize all of ICE and IRAF to the default values. You or the instrument assistant can accomplish this by the following:

- Log in using the observer login and password posted on the machine.
- In each of the two IRAF windows ("Data Acquisition" and "Data Reduction") type **logout**. The windows should vanish.
- Select "Unix Xgterm" from the root menu. (Move the mouse to any blank area of the screen, and hold down the right mouse button. Slide the mouse down to "Unix Xgterm".)
- In the new window, type **obsinit** and answer the questions appropriately (Fig. 9).

The reason for the above procedure is that you cannot have IRAF running during an *obsinit* or parameters will not be correctly reset. Make sure that you have logged out of each IRAF window before running *obsinit*.

The *obsinit* command will also allow you to specify which CCD you are using, and will automatically set the correct telescope identification; both of these are crucial for correct operation. An example of using *obsinit* is given in Figure 9. Note that the user has the option of selecting whether **CNTL-z** or **CNTL-d** will be the default for an end-of-file command; the former is the standard at NOAO, but the latter is the standard at many other places. Finally, note that it is possible to run *obsinit* WITHOUT deleting any files or images! If you simply wish to set all of the ICE parameters back to their default values, you may run this in the middle of your observing run without losing any files.

C Keeping Your Data Organized: Directories, Subdirectories, and Pixel Files

Those of us who have been at one time or another labeled "organizationally impaired" find that it is useful to keep our images separated by different nights. There are a couple ways of doing this:

1. You could use a separate root-name for each night, i.e., "a" for the first night, "b" for the second night, and so on.

2. You could create a separate subdirectory for each night. To do this,

   (a) Use **mkdir** *subdirectory_name* to create a subdirectory that is named *subdirectory_name*.

   (b) Move to that directory by doing a **cd** *subdirectory name*. You will need to do this in each of your IRAF windows (both "Data Acquisition" and "Data Reduction").

You can always find what directory you are in by doing a **path**. To return to your "home" directory, do a **cd** (no argument).

When you take data, the image headers are actually all that wind up in the current directory. The pixel files (i.e., the actual data) wind up in a special "image directory" called **imdir**. However, by setting the image directory to the special designation "HDR$pixels/" when you ran **obsinit** (Sec. B), you have assured that your pixels will wind up on the same disk as your headers, stored in a subdirectory called "pixels" of whatever directory you happen to be in when you take an image.

The fact that the pixels and the headers are stored on the same disk makes it very easy to see whether or not you are in danger of running out of disk-space. If you type

```
OBSINIT deletes all images and files, and replaces the startup files.
  There is a prompt later if you want to only replace the files.

  Initialize the '4meter' account? (yes):

OBSINIT thinks node 'khaki' means telescope 'kp4m'.

The names of the observers and the KPNO/NSO proposal IDs are needed
  by the NOAO archive to preserve the paper trail.  Please enter the
  names of the ACTUAL OBSERVERS for this run, separated by commas.
  SPELLING & CAPITALIZATION count.  First names & initials may be used.

  Observer(s): Massey

PLEASE CHECK THE PROPOSAL ID CAREFULLY (see the posted copy of the
  observing schedule).  Note that special projects (e.g., synoptic
  or queue observing) may have non-numeric proposal IDs.

  Proposal ID: 1440

What observing setup? (arcon fire ice) ice

Enter the imdir directory name (HDR$pixels/):

Default interrupt characters are ^Z(eof), ^X(susp) (not UNIX standard).
  Change these to the standard values ^D(eof), ^Z(susp) instead? (no):

OBSINIT replaces a large number of the observer's unix and iraf
  configuration files while defining the complete appearance of
  all of the acquisition and reduction windows.

  Do you also want to DELETE ALL IMAGES AND FILES? (yes):
...deleting images and files, be patient
    ...restoring default login and startup files
    ...configuring 4meter account for ice
    ...setting telpars.telname = kp4m

Primary detector names from /usr/iraf/extern/ice/ccdacq/lib/runlib/detcap:
t5ha  Tek   512x512  - 27u pixels (lower amp)
t1ka  Tek  1024x1024 - 24u pixels (A amp)
t2ka  Tek  2048x2048 - 24u pixels (D amp)
t2kb  Tek  2048x2048 - 24u pixels (D amp)
s2ka  Tek  2048x2048 - 21u pixels (D amp)     <- STIS
f3kb  Ford 3072x1024 - 15u pixels (lower amp)
gcam  Ford 3072x512  - 15u pixels (lower amp) <- GCAM
cryo  Ford 1200x800  - 15u pixels (upper amp) <- CRYO
 ti4   TI   800x800  - 15u pixels (lower amp) <- McMath SSG
 ti5   TI   800x800  - 15u pixels (lower amp)

  Select a detector name from this list: t2kb

Logout and login again for all of these changes to take effect.
4meter@khaki%
```

Figure 9: Running *obsinit* will reset all ICE and IRAF parameters back to their default values.

diskspace you will see a listing of the percentage used on each disk area; the one to pay attention to is the one labeled **data1**.

By executive fiat, the disk-names have been made sensible and consistent across the mountain-top. When you log onto the SUN computer in your dome and do a **path** you will see that your home directory is called

machine_name!/data1/account_name/

I.e., if you are logged onto "taupe" as "36inch" the home directory is

taupe!/data1/36inch/

At each telescope, the data-acquisition disk is **/data1** is physically attached to the data-taking computer. There may also be a smaller disk, **/data2**, which is "cross-mounted" from a second (data-reduction) computer in some of the domes (cocoa, royal, rust). Given the limitations of cross-mounted disks, we suggest that you use **/data2** only as a reduction area, and that you access it primarily from the second computer. If you attempt to take data using **/data2** (by changing directories to this disk), you may find that things are incredibly slow. (A normal 3^m10^s read-down of T2KB took 5^m30^s on **/data2**.) It takes less time to keep your taping up to date!

To make a duplicate copy of your images on **/data2** you can do the following:

imcopy *.imh /data2/4meter/

Note that this has to be done while in the directory where the images you wish to copy currently sit.

D Taking the Bus: Alternative Schemes of Getting Your Data Home

Although we recommend all observers make two FITS tapes of everything, there are additional ways of sending your data home.

Observers with lots of time on their hands, or who are using very small chips, may wish to transfer their data home over the internet using *ftp*. An example is given in Figure 10. First the data are written as *disk* FITS images by specifying a non-tape drive name. The *ftp* facility is then used in non-interactive, *binary* mode to transfer all the images.

It is also possible to use the UNIX tar facility to tape your data, but because IRAF keeps the header and pixel files separately, you must first write your IRAF images to FITS images on disk, and then *tar* the FITS files. Thus you might:

- **files e*.imh > tonight**
- **wfits @tonight sendem make+**

Next, allocate the drive in IRAF (**all mta**, say), and then do a

!tar -cvf /dev/nrmt8 sendem*

```
cl> files e*.imh > tonight
cl> wfits @tonight sendem make+
WARNING: FITS tape blocking factor is 10
File 1: e0011.imh -> sendem001 flats 6707 start of   Size = 38 x 1022
        2 Header  54  Data logical (2880 byte) records written
File 2: e0012.imh -> sendem002 flats 6707 start of   Size = 38 x 1022
        2 Header  54  Data logical (2880 byte) records written
cl> ! ftp tofu.tuc.noao.edu
Connected to tofu.tuc.noao.edu.
220 tofu FTP server (SunOS 4.1) ready.
Name (tofu.tuc.noao.edu:feed): massey
331 Password required for massey.
Password:
230 User massey logged in.
ftp> cd /data1/massey
250 CWD command successful.
ftp> prompt
Interactive mode off.
ftp> binary
200 Type set to I.
ftp> mput sendem*
200 PORT command successful.
150 Binary data connection for sendem001 (140.252.53.6,1824).
226 Binary Transfer complete.
local: sendem001 remote: sendem001
161280 bytes sent in 1.6 seconds (97 Kbytes/s)
200 PORT command successful.
150 Binary data connection for sendem002 (140.252.53.6,1825).
226 Binary Transfer complete.
local: sendem002 remote: sendem002
161280 bytes sent in 2.3 seconds (67 Kbytes/s)
ftp>bye
```

Figure 10: Sending images home electronically.

```
cl> files e*.imh > nite5 [on the mountain computer create a file]
cl> ! telnet tofu [log on downtown]
Trying 140.252.3.55 ...
Connected to tofu.tuc.noao.edu.

login: massey
Password:
Last login: Sun Nov 24 22:37:12 from indigo.kpno.noao
SunOS Release 4.1.1 (NOAOSS) #9: Thu Oct 10 14:10:16 MST 1991
tofu% cl [start up IRAF]
cl>set dum="indigo!/data1/feed/"
cl> files dum$//@dum$nite5 > nite5
cl> imcopy @nite5 @dum$nite5
dum$e0011.imh -> e0011.imh
dum$e0012.imh -> e0012.imh
```

Figure 11: Copying images from the mountain to downtown.

where the "nrmt8" should be whatever the Unix device name is (*nrmt8*, *nrst17*, *nrst0*, etc.), which you will find marked on each drive. (The **!** is just a way of telling IRAF that you are executing a Unix command; if you are doing this from a Unix window, forget the "!".) Additional tar files can be added with the same command. However, if you remove the tape from the drive, and append to it at a later time, you must first position the tape to the end of the last file using

!mt -f /dev/nrmt8 fsf 3

where 3 is the number of tar files already on the tape in this example. The contents of a tar file may be listed using

!tar -tvf /dev/nrmt8

but the tape must first be rewound with

!mt -f /dev/nrmt8 rewind

and if necessary positioned to the proper file again with **mt** as shown above. Files can be recovered from a tar file using

!tar -xvf /dev/nrmt8

after positioning the tape with **mt**'s. Use **rfits sendem* make+ old+** to then restore the FITS files to IRAF files. Note that for the time being one should not use the IRAF **wtar** and **rtar** commands.

For those lucky few of us that have accounts on both the mountain and downtown machines, it is relatively easy to simply use IRAF networking to **imcopy** a set of files directly down onto our disks. Log onto your home machine remotely by using **telnet**, log into IRAF on your home machine, and then do a **set dum="indigo!/data1/feed/"** (note the final "/"!). One can then copy a single image by doing a **imcopy dum$e0011 e0011**. To copy a slew of images is somewhat trickier; follow the example given in Figure 11.

E IRAF Networking

To network with IRAF it is not necessary to "do lunch." Instead, the astronomer may address images on one machine from another by specifying their full path names. For instance, to display images stored on the "schmidt" account on **loden** while running IRAF logged on to **cocoa** as "4meter", one would merely say **set hum="loden!/data1/schmidt/"** (note the final "/") and then do a **display hum$e0011.imh 1 fill+**, for instance. Similarly, one could copy a file from **loden** to **cocoa** by typing **imcopy hum$e0011.imh newe0011.imh**. A further example may be inferred from that of Fig. 11.

F Using ximtool

The IRAF image display window is "ximtool", and combines the best features of Sunview/IMTOOL with SAOIMAGE. To load an image into the display, simply type **display image 1**, where the "1" denotes display frame number 1. (Up to 4 images can be loaded in frames 1-4.) This command uses many defaults that sometimes need to be changed, depending on the data and your needs.

Since "ximtool" contains a "panner box", how we recommend using the display is a little different than what we described in previous versions of this manual for Sunview/IMTOOL.

Since the largest image that physically fits on the screen contains only about 1000 pixels on a side, and since many of our CCDs are considerably bigger than this, there is some trickiness to how you might want to display your data. Basically you have two options:

1. Set the buffer size (*set stdimage=imtname*) to the full size of the image, and display to it using *display imagename 1 fill−*. The **fill−** will assure that each pixel in the image gets mapped into a single pixel in the display. For our large chips, however, only a subsection of the image will fit on the displayed area. Fortunately there is a panner box in the upper right hand corner of the window which will show you the entire image with a bright green box indicating which subregion is displayed on the main part of the window. The advantage of this is that when you see a pixel, it is really a pixel, and you can easily evaluate things like bad pixels and columns. The disadvantage is that you cannot see the entire image at once, but must move the panner box in order to change the region being shown.

2. Set the buffer size (*set stdimage=imtname*) to something small enough to fit on the screen but display to it using *display imagename 1 fill+*. The **fill+** means that every pixel in the image will contribute to the intensity of each pixel in the display, but not 1:1. In other words, if you were displaying a 1024 x 1024 image into a frame buffer that was 512 x 512 large, using fill+ would be the same as averaging ajacent pixels. The advantage of this is that you will see the entire frame without having to move

the panner box; the disadvantage will be that even if you "zoom", you will not be seeing actual individual pixels.

Note: In the past we recommended using the "fitframe" option. However, if you do this with an frame buffer larger than the frame size, you are in danger of "losing" your window—you may not be able to reach the control bar to have any effect on the window!

The first author on this document prefers option (1) for spectroscopic applications but (2) for direct imaging. One of the other authors prefers to do both, using (2) [see the full frame] in the "Data Acquisition" window, so that images will be automatically displayed showing the full field, but then setting the defaults in the "Data Reduction" window using (1) so that each pixel can be cherished by redisplaying an image from there.

By default, *obsinit* sets the frame buffer to a small size (832×820) for t2ka, t2kb, and s2ka, but uses the full size for chips used solely for spectroscopic applications (gcam, cryo).

A complete list of frame buffers is available by typing the command **gdevices**.

Once the image is loaded, you can adjust the brightness and contrast parameters by moving the mouse in the window while holding down the right-hand button. Negative contrast (black stars on white background) is realized by moving the mouse to the upper half of the ximtool window; positive contrast is obtained by moving the mouse to the lower half of the window. Higher contrast is achieved by moving the mouse vertically away from the center. The image is made darker (blacker) by moving the mouse to the upper left (negative contrast) or lower right (positive contrast). The most extreme displays are usually obtained when the mouse is in one of these two corners.

You can learn the approximate pixel location of the mouse by reading the value in the box at lower right. The intensity is also displayed there. If the intensity read-out has a "+" or "-" next to it, the pixel is either brighter or fainter than the display command has scaled the data. To change the scaling from the default, use the display command as follows:

display *image* **1 zscale- zrange- z1=0 z2=2000**

to set the scaling is now set to 0-2000 rather than the automatic scaling printed out previously when display was executed.

You may also zoom the picture using the middle mouse button. Press the button once and the image will move in an attempt to center up on the mouse cursor. Press it again without moving the cursor and the image will zoom up a factor of 2. Pressing the middle button 3 more times will zoom the picture up to a factor of 4, 8, and back to 1 (normal). If you move the cursor while zoomed and then press the middle button, the picture will pan to center up on the cursor.

You can also pan by simply grabbing the green outline in the panner box using the left mouse button, and moving it to outline the part of the image you would like.

The controls on the upper right allow one to flip the image left and right, up or down, change frames (the frame number is displayed between the two fat arrows), and, best of all, open a control pannel (left most icon). The control pannel allows you to turn the coordinates box and panners on and off, flip, blink, and much, much more.

If you find that you have somehow mysteriously added little green boxes on your image, you can get rid of these by placing the cursor in such an extraneous box, holding down the right-most mouse button, and selecting "Destroy".

G Using *imexamine*

The **imexamine** task provides some of the most powerful diagnostic quick-look tools within IRAF. If an image that you wish to examine is already displayed in the ximtool window, simply type **imexamine**. Wait patiently *without moving the mouse* and a blinking, round cursor will appear on the display. (If you lose patience, try pressing the "L5" key without moving the cursor.) Place the cursor over a star, and strike the **r** key, and you will be presented with a radial plot of the star, along with the values of a fit to the stellar profile. The last number displayed is the FWHM in pixels, quite useful for determining the best focus. Other very useful commands include **l** for making a line plot at the position of the cursor, and **c** for making a column plot at the position of the cursor. Other useful cursor strokes are shown below.

r - Make a radial profile of the star near the cursor. The FWHM will be the last number shown in the plot.

a - Print FWHM without showing the radial plot.

c - Plot the column nearest the image cursor

l - Plot the line nearest the image cursor

j - Fit a 1-d Gaussian in the x direction, centered near the cursor

k - Fit a 1-d Gaussian in the y direction, centered near the cursor

m - Print the statistics in a box around the image cursor

e - Make a contour plot of a region around the image cursor

h - Plot the histogram of a region around the image cursor

s - Make a surface plot of a region around the image cursor

:c N - Plot column N when in graphics mode

:l N - Plot line N when in graphics mode

:naverage M - ave M columns (or lines) during plots

x - Print the x, y, z values of the pixel nearest the image cursor

z - Print a 10 by 10 grid of pixels around the image cursor

o - Overplot

g - Go to the graphics window from the ximtool window

i - Return to the ximtool window from the graphics window

? - Print help

q - Quit **imexamine**

:epar r - Edit the radial profile plot parameters

:epar c - Edit the column plot parameters

:epar e - Edit the contour plot parameters

:epar h - Edit the histogram plot parameters

:epar l - Edit the line plot parameters

:epar s - Edit the surface plot parameters

H Running ICE from a VT640

If for some reason you need to run ICE from a VT640 terminal (which is a VT100 terminal with "RETRO-GRAPHICS" printed below it), you can do just about everything that doesn't require the ximtool window. Log on to the correct machine using the same account and password. If you want to run ICE/CCD commands, start IRAF by typing "cl". However, if you want the same parameters as you have been using for reductions, first do a **cd reduce**, then a **cl**, followed by a **cd**.) When IRAF starts itself, you may get the message "setting terminal type to xgterm...", followed by a pause, and the statement "timeout—terminal type set wrong? ('stty termtype' to reset)" So do exactly that by typing

stty vt640

You will find that all the ICE commands work fine, including the **implot** plotting commands, but if you attempt to address the ximtool window you will either get an error message or (if someone else is using the main console) write over what is in the ximtool window of the main console.)

I Speed Reading

Changes made in the microcode for the 3 Feb 1992 release have made a substantial improvement in how quickly the large-format CCDs can be read. It now takes about half as much time to read each pixel, the preclear at the start of the exposure is ten times faster than it was. Changes to the A/D card and the microcodes in August 1996 have sped this up even more; see Q.

Some users are scheduled with a large-format device but wish to use only a small section of the chip. The way these devices work, you will get the maximum gain in speed if the region to pick is at low row numbers. (The column location is irrelevant.) If you wanted a 512 $\times$ 512 section of one of the 2048 chips (t2ka, t2kb, s2ka), you may want to choose a region at the bottom of the chip: **firstcol=768 lastcol=1279 firstrow=1 lastrow=512** in **detpars** would be a region at the bottom center of the chip. Using the large chip in this fashion should add little or no overhead compared to using the equivalent 512 $\times$ 512 device, such as "t5ha".

Remember too that the read time is dependent upon the gain value you have chosen (Sec. 3.1.1). The default value (lowest number of electrons per ADU) yields the lowest possible read-noise, but will take substantially longer to read the chip.

J IRAF Miscellanea

There are several routines that have been provided recently by the IRAF group that we have chosen to highlight here. These are either in their preliminary stages, or are considered "prototypes", and hence do not come with the same seal-of-approval attached to more mature software. Nevertheless, we have used these with good benefit to our own programs, and feel you may too.

J.1 Analyzing Direct Focus Frames with *kpnofocus*

Analyzing direct imaging focus frames has now been made fun and easy thanks to the new *kpnofocus* routine. Simply run *kpnofocus*, giving as input the name of the focus frame. Next, mark the top image in a multiple focus exposure using the "g" key. You will have to wait a few seconds, but you will then be presented with a plot of FWHM vs. focus value. Do this for a few stars and then exit using "q". A This routine must be loaded from the *nmisc* package.

```
nl> lpar specfocus
       images = "d*.imh"        List of images
       (focus = "1430x10")      Focus values
    (corwidth = 20)             Correlation width
       (level = 0.5)            Percent or fraction of peak for width measureme
      (shifts = yes)            Compute shifts across the dispersion?\n
    (dispaxis = 2)              Dispersion axis (long slit only)
    (nspectra = 1)              Number of spectral samples (long slit only)
       (ndisp = 1)              Number of dispersion samples
       (slit1 = 5)              Lower slit edge
       (slit2 = 50)             Upper slit edge\n
     (logfile = "logfile")      Logfile
        (mode = "ql")

SPECFOCUS: NOAO/IRAF V2.10EXPORT feed@indigo Sat 19:37:35 19-Sep-92
  Best average focus at 1465. with average width of 1.41 at 50% of peak

  -- Average Over All Samples

                                        Image  Focus  Width
                                    d0001.imh  1430.   2.86
                                    d0002.imh  1440.   2.23
                                    d0003.imh  1450.   1.66
                                    d0004.imh  1460.   1.44
                                    d0005.imh  1470.   1.44
                                    d0006.imh  1480.   1.89
                                    d0007.imh  1490.   2.54

  -- Image d0004.imh at Focus 1460. --
```

Figure 12: A run of specfocus on 7 focus exposures, in which the camera focus began with a value of 1430 and was stepped by 10 units for each exposure.

J.2 Focusing a Spectrograph with *specfocus*

Frank Valdes has provided a very useful routine to aid us in determining the best focus of a spectrograph using a series of exposures of comparison lines. The assumption is that the collimator or camera focus has been changed between each of the exposures. The routine works by cross-correlation, and provides an easy measure of the full-width-at-half-maximum (fwhm) of the images. In addition, one can specify that the dispersion axis be broken into multiple samples to determine if the best focus is wavelength dependent, and/or the spatial axis can be broken into multiple samples to determine if the best focus varies across the spatial axis. We show in Fig. 12 the simplest use of the program. The dispersion axis goes along columns (**dispaxis=2**), and the slit covers columns 5 through 50. In addition, there is a very sophisticated graphical display that helps in interpreting the results in the case of multiple sampling along the dispersion/spatial axis; read the **help** page for **specfocus**. Currently **specfocus** may be loaded from the **nlocal** package.

J.3 Determining the gain and readnoise using *findgain*

At CTIO it is common to measure the CCD gain and read-noise whenever a CCD is installed on the telescope. Rob Seaman has provided a useful script called **findgain** that will provide good estimates of the gain and read-noise using two flat-field exposures and two bias frames. The frames should not have been processed or combined in any manner. The astronomer should specify a section of the image over which the flat-field is relatively constant. Additional details can be found by reading the help page for **findgain**. The routine currently lives in the **nproto** package.

J.4 Determining the shutter correction time with *findshutcor*

The actual exposure time of your image may differ by some fraction of a second from the requested exposure time due to the fact that no shutter can open or close instantenously. In practice, our "big" shutters (used for direct imaging with the T2KA and T2KB chips at the 4m and 0.9m) have a shutter correction time of about -0.12 sec, uniform across the field. Our brand new big shutter used at the Schmidt with S2KA seems to have a negligible shutter correction time. The "medium" shutter (used for direct imaging with the T1KA chip has a correction time of roughly +0.026 sec at the center and +0.004 at the edge. Our "small" shutters (used for direct imaging with the T5HA and TI chips) are said to have a nearly negligible shutter correction time. By contract, the shutters in our spectrographs are quite slow, with a measured correction time for GoldCam of -0.25 sec, for example.

If you are attempting absolute photometry or spectrophotometry, and plan short enough exposure times that you are worried about the shutter correction, there is a simple routine in IRAF called *findshutcor* which will help you evaluate the correction. Written by Rob Seaman, the routine determines the shutter correction from a series of flat-field exposures of varying lengths (1 sec - 20 sec, say). You must be careful not to have saturated on the longest exposures, of course, and the frames must have been processed for removing the overscan level, but not flat-fielded. In taking the data one must also carefully take into account the possibility that the flat-field lamps intensity may drift slightly with time; i.e., a good exposure sequence might be 1sec, 20 sec, 2 sec, 2 sec, 20 sec, 1 sec. The *findshutcor* routine currently lives in the **nlocal** package.

J.5 Checking the weather without leaving the dome: *wdisplay* and picking up the latest forecast.

Are you wondering how likely it is going to remain clear, or are you considering packing it in for the night? The **wdisplay** task in the **nlocal** package may help you decide what to do. Three times an hour, 24 hours a day, a new GOES weather satellite picture in one of three wavelength bands (visible, IR, and a water-vapor narrow-IR band) is downloaded.

It is fun, easy, and informative to display the last four images, and quickly blink between to see what's coming your way, or leaving you alone. Simply run

wdisplay ir four+

to load the last four images into your ximtool window. (You may have to select the menu item "FitFrame".) Hit CNTL-F repeatedly to make a little movie. Further information can be gleaned from the help page.

If you would like to compare your guessimate with that of an expert, you can also easily retrieve the most recent weather prediction. Put the mouse in a blank area of the screen and select "Weather Forecast" under "Astronomer Tools". You'll be presented with a map of the US with little dots on it; place the cursor near Tucson, and click. You can check the weather back home the same way, of course.

K Being Environmentally Conscious: Scripts

We have so far used ICE only to help you take and store your data in a simple, pleasant fashion and examine the results. However, the "E" in ICE stands for "environment", and many observers may benefit from the powerful possibilities inherent in unleashing the "full power" of IRAF. Using ICE, the observer can construct simple observing scripts that may make his/her life much easier.

Let us take the hypothetical situation that an observer has an observing program that consists of repeated short exposures on a single field through two filters, each with different exposure times. S/he *could* spend the next three hours typing **observe** over and over, each time changing the filter, focus, and exposure time. Or s/he could write a short script that would repeat this sequence, freeing the observer to pay attention to results and enjoy a cup of coffee. How would we go about this?

The sequence that we want to emulate is something like the one shown in Fig. 13. The script we write for doing this in an easy and relatively painless manner is shown in Fig. 14.

This has been a relatively simple example, but perhaps illustrates how a little cleverness can save you an immense amount of work at the telescope. Dave Silva has put together a set of such scripts of varying complexity, intended as simple observing aids, and we list them below:

- **reformat.cl:** Reformats the chip to a central square region.
- **photstds.cl:** Once you have **display**ed an image containing standard stars, the script **photstds** can be used to get aperture magnitudes. (It simply runs IMEXAMINE having set the aperture photometry values to "sensible" values.)
- **obslooper.cl:** Repeats some sequence of observing tasks, similar to what is shown in Fig. 14.

```
cc> observe
Image type (object|zero|dark|flat|comp|focus) (object):
Exposure time (seconds) (0.:16200.) (10.): 20.
Filter bolt positions (4): 5
Telescope focus (1260): 1240
Object title (WR7 b): WR7 u
image `mj0051' will be written to disk
T2KA  [1:512,1:512] bin=[1,1] gain 1.6 dewtemp -169 tel-link=up fil=5
preparing...
exposing...

    Time left  Exposed  Tcam Tdew

exposure = 20.0 seconds (0:00:20.0)
darktime = 20.0 seconds (0:00:20.0)
reading out...
image `mj0051' written to disk
cc> observe
Image type (object|zero|dark|flat|comp|focus) (object):
Exposure time (seconds) (0.:16200.) (20.): 10.
Filter bolt positions (5): 4
Telescope focus (1240): 1260
Object title (WR7 u): WR b
image `mj0052' will be written to disk
T1KA [1:512,1:512] bin=[1,1] gain 1.6 dewtemp -169 tel-link=up fil=4
preparing...
exposing...

    Time left  Exposed  Tcam Tdew

exposure = 10.0 seconds (0:00:10.0)
darktime = 10.0 seconds (0:00:10.0)
reading out...
image `mj0052' written to disk
cc> observe
Image type (object|zero|dark|flat|comp|focus) (object):
Exposure time (seconds) (0.:16200.) (10.): 20.
Filter bolt positions (4): 5
Telescope focus (1260): 1240
Object title (WR7 b ): WR7 u
image `mj0051' will be written to disk
T2KA  [1:512,1:512] bin=[1,1] gain 1.6 dewtemp -169 tel-link=up fil=5
preparing...
exposing...

    Time left  Exposed  Tcam Tdew

exposure = 20.0 seconds (0:00:20.0)
darktime = 20.0 seconds (0:00:20.0)
reading out...
image `mj0053' written to disk
(and so on)
```

Figure 13: The observer is seen endlessly repeating the sequence of a 20 sec exposure in filter 5 with title "WR7 u" (focus set to 1240) followed by a 10 sec exposure in filter 4 with title "WR7 b" (focus set to 1260). The observer eventually goes insane and turns to writing manuals for pleasure.

```
#
# OBSLOOPER - do some repeatative observing task N times
#
# template version: dave silva (drs), noao/kpno, 9 aug 92
#
# To define under IRAF:
# task obslooper = "pathname/obslooper.cl"
#
# We suggest you copy this to a different file and edit as desired.
#
#
procedure obslooper(numloops)
int numloops {prompt="Number of loops"}
begin
int reps
# query of input if necessary
reps = numloops
# loop over some tasks
for(i = 0; i < reps; i = i + 1){

# TASKS TO BE REPEATED START HERE:
# Note: "qpars.aborted" lines allows observe to abort entire loop

    telescope(telfilter=5)
    telescope(telfocus=1240)
    observe(object="WR7 u",image="object",exposure=20.,
setfilter-, setfocus-)

    if( qpars.aborted )
break

    telescope(telfilter=4)
    telescope(telfocus=1260)
    observe(object="WR7 b",image="object",exposure=10.,
setfilter-, setfocus-)

    if( qpars.aborted )
break

# ...AND END HERE

}

end
```

Figure 14: A simple script that repeats the observing sequence discussed in the previous figure as many times as you'd like. The observer is free to go have some coffee and think deep thoughts.

- **dostds.cl:** This is a utility designed to take a series of exposures (without moving the telescope) through different filters, with different exposure times and (optionally) different focus settings. Could be used to step through UBV exposures of a Landolt standard field, say.

- **domeflats.cl:** This command is a script that will acquire a complete set of $UBVRI$ dome flats at the 0.9m without user intervention.

- **boaa_constrictor.cl:** This task steps the telescope through a grid of exposures. This works only at the 0.9m.

- **tcpcom:** This is a C routine provided by Lindsey Davis and D'Anne Thompson that allows one to communicate directly with the telescope control computer from IRAF, i.e., **tcpcom "5 robject preset"**. This works only at the 1.3m and 0.9m telescopes.

- **contrib.cl:** Allows the user to contribute a script for possible future inclusion in this package.

While it is possible to get yourself into trouble using these scripts, we believe the potential benefits to one's observing program outweigh the potential for loss of time. These examples are meant to be illustrative of what can be done; they have all been demonstrated to work at the 0.9m, and extensions to other applications will doubtless occur to you. You can readily modify these scripts as you wish; understand, though, that their availability comes with the disclaimer that you use these scripts at your own peril.

To use the unmodified version of these scripts, you simply need to type the command as inferred from the above list, i.e., **reformat**. To modify a script you need only edit it; they are located in a subdirectory of your home directory called **scripts**, i.e.,

- **cd scripts**

- **edit obslooper.cl** (Edit obslooper to do what you want, specifying the exposure lengths, titles, and filters.)

- **cd**

- **obslooper** (The modified script is now run.)

You can also copy one of these scripts to a new file and run it; in that case you will need to define it as an external task:

- For a script without parameters (i.e., a simple list of commands, like **domeflats**), do a:

  task $newscript=/data1/36inch/nite1/newscript.cl

- For a script with parameters (such as **obslooper** or **boaa_constrictor** do a:

  task newscript=/data1/36inch/nite1/newscript.cl

Note: Because the **observe** *task is patiently waiting you to interrupt it with a keystroke once you begin an exposure, it is no longer possible to run simple lists of commands by doing a* **cl** < **scriptname.cl**.

Additional information can be gleaned from the help pages for these scripts (i.e., **help obslooper**) and the README file in the *scripts* subdirectory. We recommend that one always keep a copy of the **An Introductory User's Guide to IRAF Scripts** by Anderson and Seaman handy; a copy is located in every dome in the Big White Binder ("KPNO IRAF Documentation")

L Other Nifty Things

There is a prototype automatic logging routine that constructs a TeX version of the observing log. Instructions may be obtained via anonymous ftp to "ftp.noao.edu", cd kpno/manuals, binary, get ccdlogs2.ps.Z. Comments should be directed to tlauer@noao.edu.

Under "Astronomer tools" in the root menu, there is an ephemeris program ("xephem"), provided by Elwood Downey, that will tell you the Julian date and times of sunrise, sunset, and twilight, and many more wonderful things. (Do a Unix level **"man xephem"** for more information.)

M Ramps

Are you suspicious that your A/D converter may have a stuck bit? This problem is described in detail in the Massey & Jacoby *CCD Data: The Good, the Bad, and the Ugly* reprint, found in many domes. The ramp exposure gives us the opporunity for sampling many data values in a single exposure: the shutter is opened, and the chip is read down while still exposing to light. We are still in the process of experimenting with this new feature, and future versions of this manual will provide some diagnostic examples.

N What ICE Really Does During An Exposure

The exact sequence of events during a CCD camera exposure is laid out below.

- Prompt the observer for task parameters.

- Update the filter and focus parameters to be passed to the (telescope) server.

- Prompt the observer for the image title and retrieve hidden parameters.
- Construct the name for the output image.
  - abort if it already exists
  - print a message otherwise
- Open a connection to the detector server.
  - pass info back and forth to the controller
  - retrieve the detcap information
  - complain about the head ID if it does not match
  - check the error status
- Update the scan table parameters to be passed to the instrument server (if appropriate).
- Open a connection to the instrument server (usually "test").
  - pass info back and forth,
  - check the error status
- Open a connection to the telescope server.
  - pass info back and forth (e.g. image header info)
  - check the error status, but ignore any error if this is not an object exposure
- If no errors have been reported by the servers:
  - do whatever the instrument does before the exposure, e.g., position the scan table if appropriate
  - check the error status
- If no errors have been reported by the servers (including the telescope):
  - do whatever the telescope does before the exposure, e.g., position the focus and filter if appropriate
  - check the error status
- If no errors have been reported by the servers:
  - assign a unique RECID to the exposure

- preallocate disk space if appropriate; the observe task aborts if the disk fills up
- write out the exposure status line
- prepare the chip
- if no new detector error, start the exposure; the count down loop is entered if the exposure is longer than a few seconds
- if exposure is not aborted, read out the chip:
  * open the output image (or reopen if preallocated)
  * write the header. Most keywords were filled in previously get a few (e.g., filter) from the servers now that were set after the bulk of the header info was retrieved
  * print "reading out..." message
  * initiate ccd data transfer
  * complain if there is a problem
  * close the output image
  * rename the hidden image if preallocated
  * complain if the disk would fill up

- Close the detector, instrument and telescope servers.
- Null out the motor control parameters for the filter, focus, and scan table.
- If no errors were reported by the servers and the exposure was not aborted:
  - If not a test image, pass the image name to the archive queue.
  - Execute the postprocessing command:
    * sync the disks
    * beep the terminal
    * display the image
    * log the exposure (if loginit was run)

That's all folks!

O Problems?

Please email **pmassey@noao.edu** any discrepancies you find between this manual and reality; please report any problems by writing a **service** request (see instructions at your telescope); suggestions for ICE improvements may be made on your run evaluation form. Some of the items that we feel may cause some confusion or difficulties at present include:

1. If you have **display**ed an image and then reduced the size of the image by running **ccdproc** on it, you will get crazy results if you run **imexamine** on this image without first explicitly redisplaying it.

2. If you inadvertently use the wrong "eof" character (**CNTL-z** when you have defined **CNTL-d** as the eof or vice-versa) you may put IRAF into suspended animation. You can recover quickly by simply typing **fg**.

3. If the SUN is rebooted, or the CCD controller has been turned off and back on, the following exposure may be very strange indeed. Do several "zero" exposures to flush it out entirely

4. Observers have occasional problems with Exabyte taping. We have found that in *nearly all* instances this has come from using non-computer grade tapes. If you have bought your Exabytes during a blue-light special at K-Mart, you may wish to consider purchasing real computer-grade tapes on the mountain.

5. If you change a title during a sequence of exposures, the change may not propagate to subsequent exposures.

6. If you are using the scan-table at the 4-m, the value of "nrowscans" will be reset to "1" (no scan) if you select a bias, dark, or focus frame. It will not be reset to the previous value when you next choose object. We recommend running the scan-table with "query and set scan parameters?" to yes if you want to use the scan-table.

P How To Get the Latest Copy of This Manual

The most recent copy of this manual may be obtained over the net from either the mountain or your home institution. To obtain the most recent copy, use the "anonymous ftp" utility to **ftp.noao.edu** as shown in Figure 15. The file is "compressed" to save space, and to make the transfer faster; you will have to **uncompress** it as shown, before printing it out. Once this is done, you can print it from Unix by using **lpr -Plw(n) ice.ps**, where (n) is the local printer number. In the interest of saving paper, you may wish to first do an "ls -l" (as shown in the example) to substantiate that the date of the manual is different than the date on your current version.

```
cl> ! ftp ftp.noao.edu
Connected to pandora.tuc.noao.edu.
220 pandora FTP server (SunOS, DCT 16May93) ready.
Name (ftp.noao.edu:4meter): anonymous
331 Guest login ok, send ident as password.
Password: [log in with your email address]
230 Guest login ok, access restrictions apply.
ftp> cd kpno/manuals
250 CWD command successful.
ftp> ls -l [check the date to see if the manual has been updated]
200 PORT command successful.
150 ASCII data connection for /bin/ls (140.252.52.2,1705).
total 3741
-rw-r--r--  1 223        593821 Jun 13 02:11 0.9m.ps.Z
...
-rw-rw-r--  1 166        124327 Jan 18  1994 ice.ps.Z
-rw-r--r--  1 354         28993 Nov  3 20:52 ice.xnotes.ps.Z
...
226 ASCII Transfer complete.
remote: -l
958 bytes received in 0.29 seconds (3.2 Kbytes/s)
> binary [set transfer for binary mode]
200 Type set to I.
ftp> get ice.ps.Z
200 PORT command successful.
150 Binary data connection for ice.ps.Z (140.252.53.6,2928) (123539 bytes).
226 Binary Transfer complete.
local: ice.ps.Z remote: ice.ps.Z
123539 bytes received in 1.7 seconds (70 Kbytes/s)
ftp> bye
221 Goodbye.
cl> ! uncompress ice.ps.Z
cl> ! lpr -Plw1 ice.ps
```

Figure 15: How to get the most up-to-date copy of this manual.

Q What's New?

Q.1 Faster Read-outs and 16-bits!: Major Changes for Fall 1996

New 16-bit A-to-D converters have now been installed, and fully implemented, in all of our CCD controllers. This means that "digital saturation" now occurs at 65,535 rather than at 32,767, and we've gained a significant improvement in dynamic range and/or read-noise sampling. IRAF handles these 16-bit words as "unsigned integers" (type "u"); once you have processed the images they will be 32-bit "real's." As long as you don't change the defaults, IRAF tape writing (WFITS) and processing will handle this data-type fine, and this change should be transparent—other than the fact that you will have twice the dynamic range for a given read-noise sampling than before this change.

The default gains have been revised to reflect this improvement (see Sec. 3.1.1); sticking with the default gains will generally make you happy.

CCD	Default Gain (electrons/ADU)	electrons @65K ADU	0.1% linearity (electrons)	1.0% linearity (electrons)
Cryocam	0.8	52,000	150,000	160,000
F3KB	2.3	149,500	170,000	190,000
GoldCam	1.4	91,000	80,000	80,000
S2KA	2.5	162,500	145,000	160,000
S2KB	2.8	182,000	210,000	230,000
T5HC	3.2	208,000	420,000	450,000
T2KA	3.6	234,000	180,000	190,000
T2KB	3.1	201,500	220,000	250,000
T2KC	1.7	110,500	190,000	200,000
T1KA	3.0	195,500	250,000	250,000

The electronics has also been readjusted to minimize the effects of overshooting for these new default gain settings for the chips used for direct imaging; users who are concerned with the effects of very saturated stars on their imaging data should stick with these default gains to minimize the effect. Spectroscopists should be unaffected by this problem.

Because these A-to-D converters are also faster than the old ones, the read-down times of all of our chips are now shorter. For every 2048×2048 image you take, you will realize a savings of some 16 seconds. While this may not sound like much, it translates to an hour or more per night spread among the various Kitt Peak telescopes; over the course of a year, we estimate that this should save about 5-10 nights worth of telescope time in read-out time alone!

Q.2 Changes for the Millenium!

The first significant update of ICE is being made in some years; included are the following tweaks:

- **FOCUS GAP MOVED TO START**. The "double gap" in a focus sequence will now appear between the first and second exposures. We have done this for compatibility with Mosaic, Mini-Mosaic, and the CTIO Arcon software. In addition the following two steps have been made to keep things sane:
  - A reminder message will be printed at the end of the focus sequence: "REMINDER: The double space marks the beginning of the sequence!"
  - The "kpnofocus" routine has been modified to expect the double gap at the beginning.
- **Header words modified:**
  - For Y2K complaince the DATE-OBS format has been changed; this took place quietly earlier this year; we are now modifying the description of the format to match what is actually being written namely date AND time (UT).
  - Better precision is being reported for the RA and DEC position of the telescope (this was actually done last year); we are now deleting the rather confusing duplication of these numbers at the old precision with the header words "RA_OLD" and so on.
  - "OBSID" now replaces"RECID" as the save-the-bits tracker.