Appendix A

Operating the Instrument

A.1 Hardware setup

Figure A.1 illustrates how the instrument should be mounted on the MJUO 1 m telescope. Note the orientation of the CCD relative to the offset-guider box. The cables should be cable-tied as illustrated to provide strain relief.

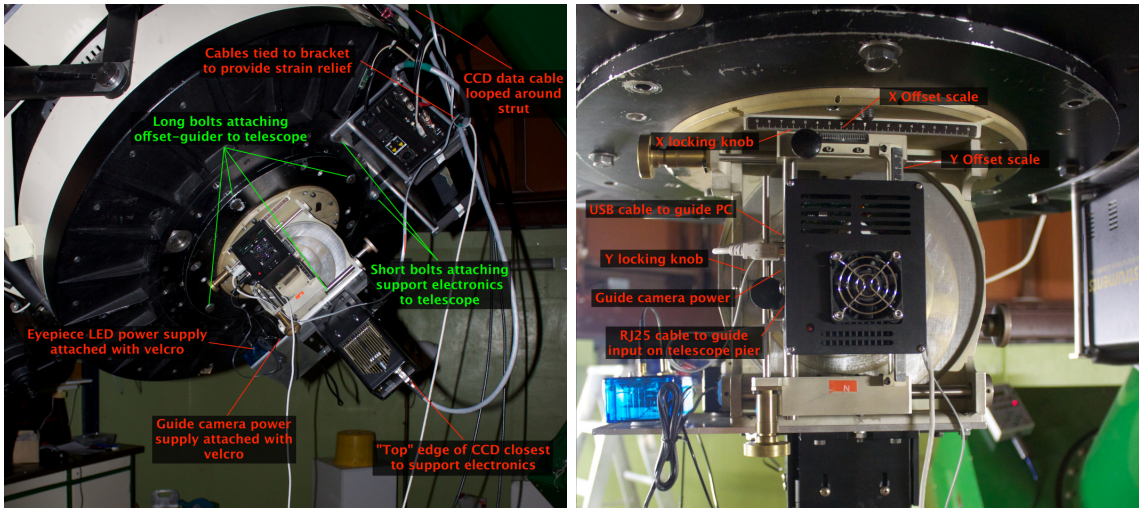

Figure A.1: Illustrated notes for mounting the instrument on the MJUO 1 m telescope.

When mounting the guide camera, the focus ring should be screwed completely in, and then screwed out by 6.5 turns. Ensure that both locking screws are tightened and that the (unused) dichroic filter is moved out of the light path.

Figure A.2 illustrates the cable connections on the camera and timer hardware.

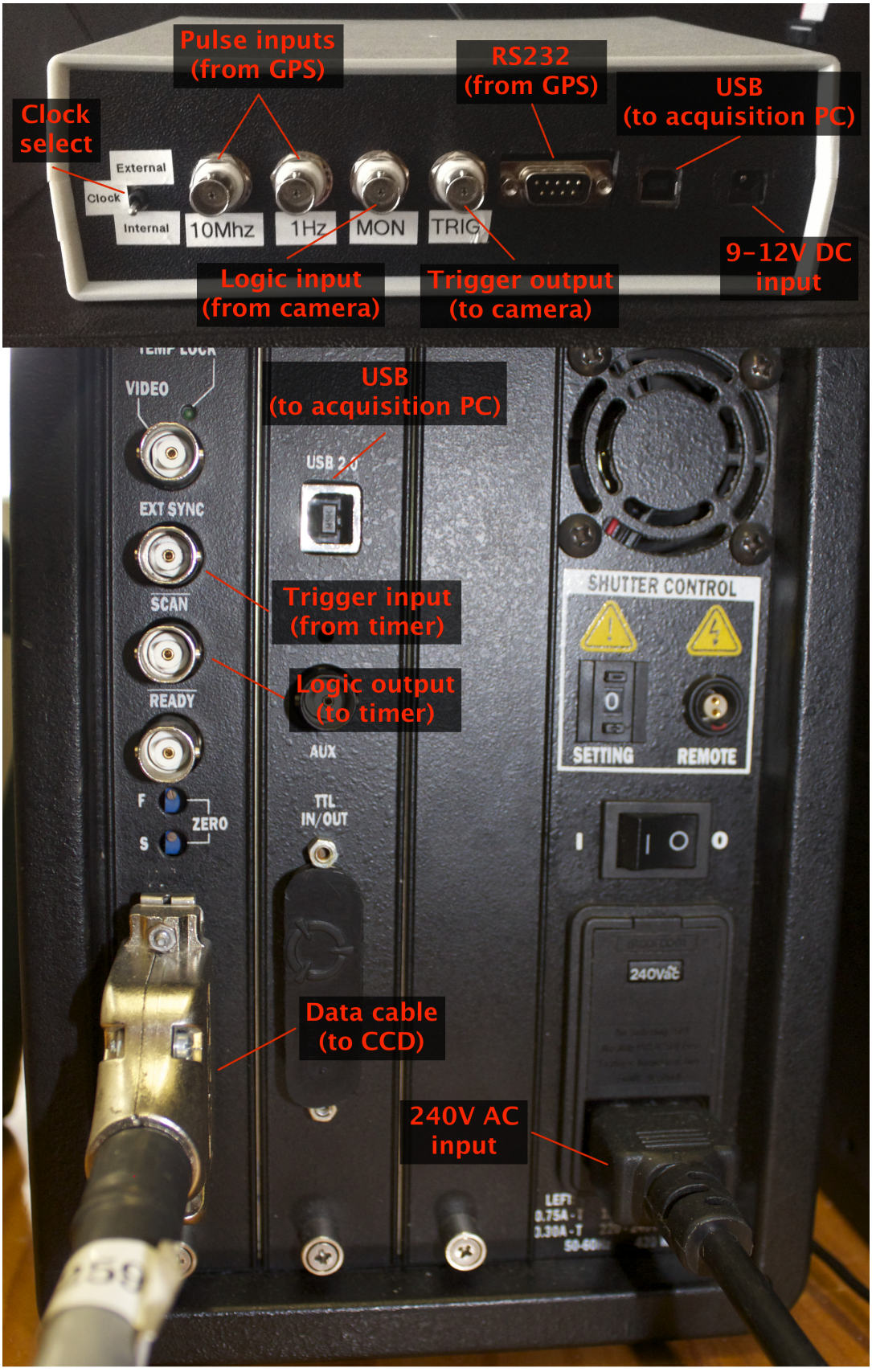

Figure A.2: Illustrated notes on the cable connections on the timer unit and CCD support electronics.

A.2 Software Setup

A.2.1 Acquisition PC

A useful first step is to enable internet access for the acquisition and guide PCs from MJUO. On the guide PC, run `ienabler.exe` and enter the standard username/password. On the acquisition PC, telnet to `ienabler.canterbury.ac.nz` port 259 and follow the prompts.

Synchronize the acquisition PC clock to NTP. The default timeserver is not accessible from VUW or MJUO, so use `ntp.vuw.ac.nz` or `1.nz.pool.ntp.org` directly:

```
$ sudo ntpdate ntp.vuw.ac.nz
```

A.2.2 Guide PC

Launch the CCDOPS program from the desktop or start menu.

In the menu, select **Camera** → **Setup...**, enable temperature regulation (the best temperature will have the cooler operating at 50–70% power; -10°C is about right) and select OK. Begin continuous exposures for alignment and focusing via **Camera** → **Focus...**.

Find a field with at least one bright star and calibrate the auto-guider by selecting **Track** → **Calibrate...**. The best x and y movement times will depend on the current RA/DEC - adjust them until the calibration graph has reasonable motion in both x and y.

When ready to start guiding (after focusing and lining up on the target field), select **Track** → **Autoguide...**.

A.3 Collecting Data

A.3.1 Starting the Acquisition

Run the acquisition software using the shortcut in the launcher (Ubuntu) or desktop (Windows). The Ubuntu launcher offers options for launching with simulated hardware via the right-click menu.

Frame metadata can be changed in the **Set Metadata** panel at any time when saving is disabled. The frame prefix is used by `tsreduce` to distinguish between

dark/flat/object frames, so ensure that a unique prefix for each acquisition sequence. The acquisition type can be changed between **Continuous** for regular observations, or **Burst** for saving a specified number of frames (set in the **Burst Count** field). The frame type can be set to **Target** allowing a target name to be entered. The other options pre-set the target name to standard values.

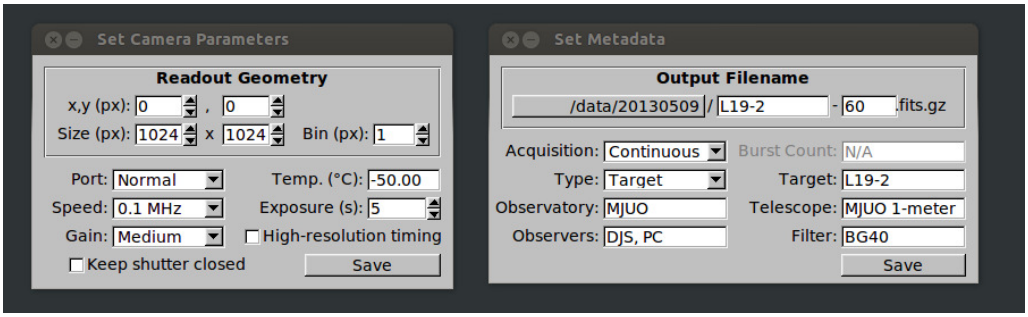

Figure A.3: The acquisition camera and metadata settings panels.

Camera properties can be changed in the **Set Camera** panel when the camera is idle. The readout geometry can be changed to read a restricted window of the CCD or to bin pixels during readout. The x and y settings specify the corner of the window closest to the readout port, and width and height its size.

The readout port, speed, and gain can also be configured to optimize the signal / noise of the observations. The ProEM camera includes a checkbox for keeping the shutter closed during acquisition of dark frames.

The timing resolution can be configured between pulse counting and high resolution modes. The high resolution mode changes the exposure time resolution to 1ms, up to a maximum of 65.535s.

Acquisition can be started and stopped with the **Acquire** button. Acquired frames will only be saved if the **Save** button is toggled. The file-name that the next acquired frame will be saved as is displayed in the in the sidebar.

Target intensity and FWHM can be monitored in the frame preview by dragging an annulus around the star, which is used by `tsreduce` to estimate the sky intensity (see Section A.3.2 for tips about selecting apertures). These statistics are recalculated for each acquired frame and do not require the frames to be saved to disk. An example is shown in Figure A.4.

Acquired frames will only be saved if the **Save** button is toggled. The file-name that the next acquired frame will be saved as is displayed in the in the sidebar.

The **Reduction** button enables the online reduction. The reduction will proceed if `<run prefix>.dat` (created using `tsreduce create` - see Section A.3.2) exists in

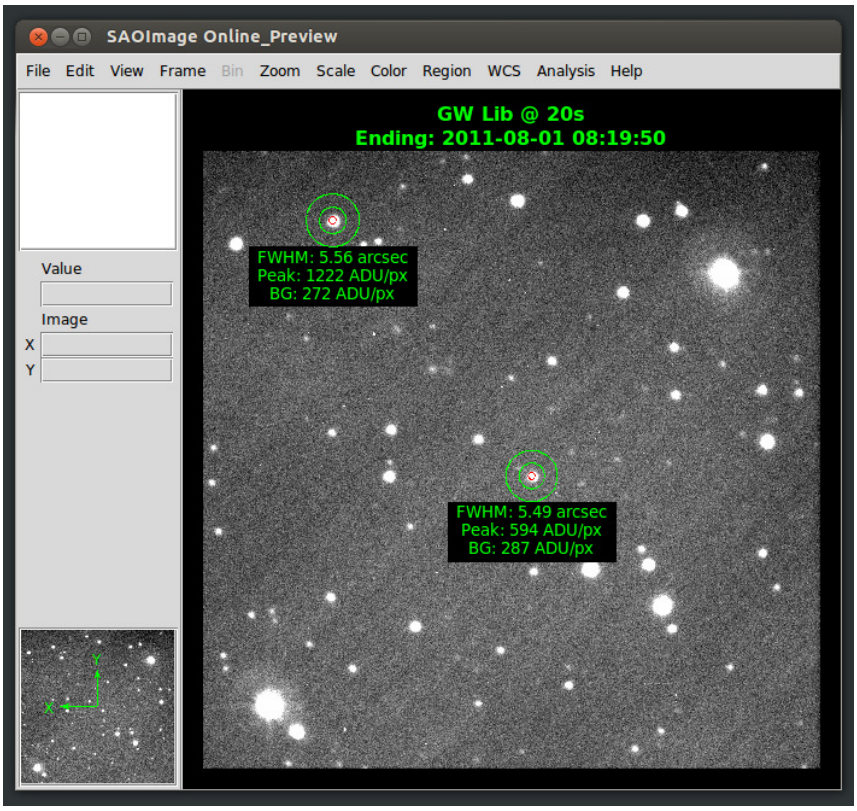

Figure A.4: The frame preview window will calculate the full-width at half-maximum (FWHM) and peak and background intensities for any circled stars each time the frame is updated.

the frame directory, and the reduction preview (opened from the launcher [Linux] or desktop [Windows]) will automatically reflect the current observations.

A.3.2 Configuring the Online Reduction

Technical Description

The interface between the acquisition software and `tsreduce` is managed by the bash script `reduction.sh`. This allows users to modify the reduction parameters (to, for example, pre-process the saved frames or use a different reduction package) without modifying the acquisition source code. It is safe to run the acquisition when the reduction time is greater than the frame exposure time - the reduction preview will simply lag behind the data, updating the preview in chucks.

The default behavior of the script is to search for `<run prefix>.dat` in the frame directory. If found, `tsreduce update` is called, followed by `tsreduce plot` which saves two images (`online_ts.gif` and `online_dft.gif`) to the acquisition software directory. The web-page `preview.html` reads the pixel data from these images at a regular interval into `<canvas>` tags for display.

There are two parameters that can be overridden by creating `config.sh` in the acquisition software directory. An example `config.sh` explaining these options is shown in Figure A.5.

```
# Override the filename controlling the reduction
# If unspecified, this will default to <run prefix>.dat
REDUCTION_FILENAME="object.dat"

# Size parameter for the online reduction plots
# If unspecified, this will default to "9"
REDUCTION_PLOTSIZE="6"
```

Figure A.5: An example `config.sh` script which overrides the reduction file-name and plot size.

Enabling Online Reduction

The reduction file can be created as soon as the first target frame has been saved (for best results, ensure that dark and flat frames have also been acquired, but this is not required).

`tsreduce` is a command-line program, so open a bash prompt from the launcher (Linux) or desktop (Windows) and run:

```
$ cd <path to frame directory>
$ tsreduce create <run prefix>.dat
```

where `<run prefix>` is the frame prefix for the target frames. If this is not an appropriate file-name, see the previous section for how to override the file-name expected by the acquisition software.

`tsreduce` will prompt for the reduction parameters and, if appropriate, create ‘‘master” dark and flat frames. It will then open a `ds9` window containing a bias/dark subtracted and flat fielded frame and ask for target stars and backgrounds to be selected. Click and drag a generous annulus around the stars - the first selected stars is assumed to be the target. `tsreduce` will center the annulus about the mean moment of the contained flux and calculate the background intensity from inside the annulus. Please note that `tsreduce` does not attempt to isolate any stellar flux inside the sky annulus, so ensure that the annulus is sufficiently far from any stars in the frame to avoid contamination. Figure A.6 shows example apertures for a sparse target field (EC 04207-4748).

The aperture size is determined using one of three techniques on the first frame in the sequence:

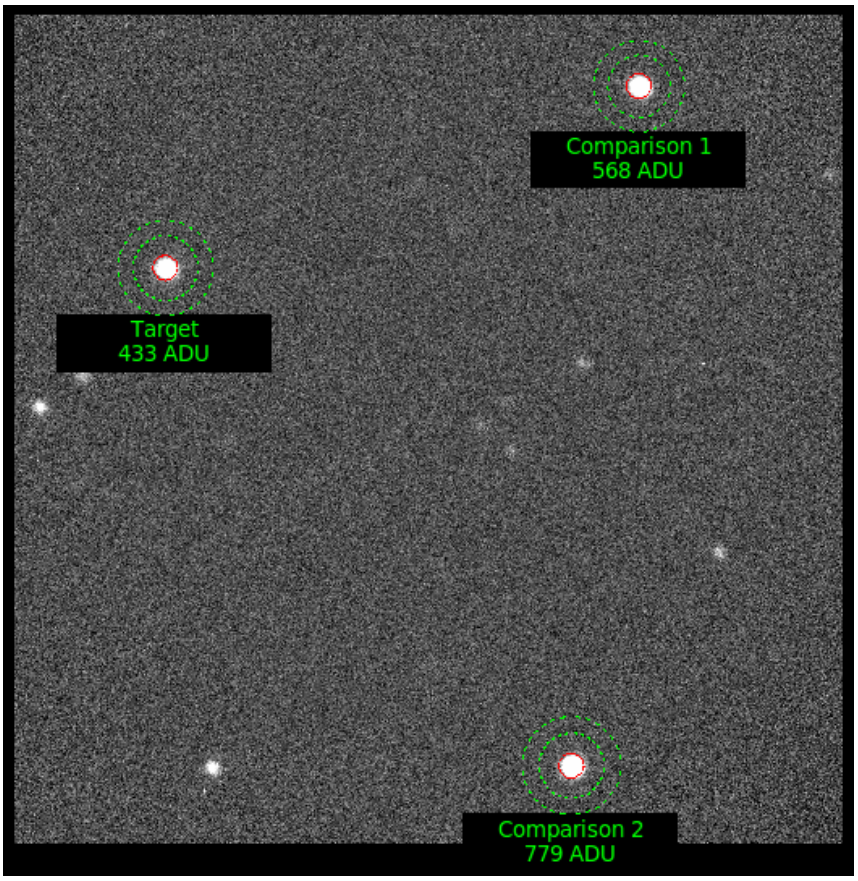

Figure A.6: Demonstration of apertures selected during `tsreduce create`. The dotted green circles define the sky annulus, and the red circle the integration apertures.

1. Estimated radius where the radial profile of the brightest star drops falls to 5σ of the background
2. Diameter of 3 times the largest estimated full-width at half-maximum.
3. Manual (prompting the user for a value)

After selecting the apertures, `tsreduce` will update the ds9 preview with the sky annuli and integration apertures. Once the user has confirmed these are correct, the reduction file is written to disk and the reduction can begin.

`tsreduce` calculates the normalized stellar intensity by dividing the integrated target brightness by the sum of the integrated comparison brightness. You will want to ensure that you pick bright comparison stars that remain within the linear CCD regime for best results.

You can now enable reduction by toggling the **Reduction** button in the acquisition software. There are several hidden options that you can configure by editing the `.dat` file with a text editor. These options are described in Table A.1.

Relay Mode

The `timerutil` program can be used to put the timer into relay mode, allowing direct communication between the acquisition PC and GPS receiver. The command syntax is :

```
$ timerutil <port> relay
```

where <port> is the OS handle of the timer (e.g. `/dev/ttyUSB0` under Linux, `/dev/tty.usbserial-00001004` under OSX, or `COM5` under Windows).

The timer can be returned to its regular mode by running the acquisition software or otherwise resetting the unit (via hardware or toggling the DTR control signal).

Note that the Trimble Studio software can corrupt the timer firmware, and should not be used. The older Trimble Monitor software is more than sufficient for checking or configuring the GPS status.

A.3.3 Additional acquisition software settings

The acquisition software has several advanced settings (see Table A.2) that can be configured by editing `preferences.dat` with a text editor. Ensure that the acquisition software is closed before editing this file to prevent the changes from being overwritten and lost by the acquisition software.

Overscan can be enabled on the MicroMax camera by setting `CameraOverscanColumns` to a non-zero value (24 is recommended). This increases the CCD width available in the geometry settings beyond the standard 1024 to include the masked post-scan pixels and the requested number of overscan columns. FITS header keys `IMAG-RGN` and `BIAS-RGN` are written into frames to indicate the image and bias regions to use during analysis.

Keyword	Default	Description
`# RatioFitDegree: <degree>`	2	Polynomial degree to fit to the ratio data.
`# PlotMaxRaw: <max>`	5000	Maximum intensity to display in the raw data plot.
`# PlotMaxDFT: <max>`	(automatic)	Maximum intensity to display in the Fourier spectrum plot.
`# PlotMinUhz: <min>`	0	Minimum frequency to display in the Fourier spectrum plot.
`# PlotMaxUhz: <max>`	10000	Maximum frequency to display in the Fourier spectrum plot.
`# PlotNumUhz: <num>`	10000	Number of points to display in the Fourier spectrum plot.
`# BlockRange: (<min>, <max>)`	–	Exclude a time range (in seconds) from ratio and Fourier calculations (heavy cloud, etc). Can be included multiple times with different ranges.
`# PlotErrorBars: <num>`	0	If non-zero, display error bars in online ratio and MMA plots.
`# MMAFilterSigma: <num>`	5	Exclude data points outside this range. Increase this if peaks are clipped.
`# PlotFwhmSmooth: <num>`	7	The number of points to average for the FWHM running estimate.
`# RA: <HH:MM:SS>`	}	Target coordinates used for BJD corrections by `tsreduce create-ts`.
`# Dec: <DD:MM:SS>`		

Table A.1: Additional tsreduce keywords which can be specified in the reduction file header.

Key		Default	Description
`Instrument`		Puoko-nui	Instrument name
`TimerMonitorLogicOut`		1	Enable monitoring of CCD logic out to when starting/stopping a run, or use fixed time delays.
`TimerSerialPort`		/dev/ttyUSB0	USB emulated serial port handle (e.g. `COM9` under Windows or `/dev/ttyusb0` under Linux)
`TimerBaudRate`		9600	Communication baud rate (requires specific timer firmware if changed).
`CameraOverscanColumns`	(Puoko-nui only)	0	Number of overscan columns to configure.
`CameraPlatescale`		0.33	Camera plate scale in arcsec/px.
`CameraFrameBufferSize`		5	Number of frames to allocate space for in the internal frame buffer used by the camera driver.
`ProEMExposureShortcut`		5	Number of milliseconds to shorten exposure times to allow the camera to complete frame transfer and prepare for subsequent triggers.
`ProEMEMGain`		1	Electron-multiplication gain.
`ProEMShiftMode`		1	Frame-transfer speed mode index.
`ValidateTimestamps`		1	Enable timestamp validation check.
`FrameFlipX`		0	Mirror image in x axis before saving.
`FrameFlipY`		0	Mirror image in x axis before saving.
`FrameTranspose`		0	Transpose image (flip x and y) before saving.
`PreviewRateLimit`		500	Minimum number of milliseconds between frame preview updates.

Table A.2: Advanced acquisition software settings, set in `preferences.dat`.

A.4 List of `tsreduce` commands by example

This section contains a list of examples of the commonly used `tsreduce` commands. Additional commands are available for offline analysis, but they are considered to be advanced functionality - refer to the `tsreduce` source code for details on these commands. Items in square brackets indicate optional parameters. A single back-slash at the end of a line indicates a line-break to fit the single-line command onto the page.

```
$ tsreduce create ec04207.dat
```
Create a reduction file `ec04207.dat` in the current directory. `tsreduce` will prompt for input regarding the reduction.

```
$ tsreduce update ec04207.dat
```
Update the reduction file `ec04207.dat` with any un-reduced data.

```
$ tsreduce plot ec04207.dat [ts.ps/cps dft.ps/cps 10]
```
Generate an online reduction plot from the reduction file `ec04207.dat`. The optional parameters define the PGPLOT devices and size for the plots - in this case, to postscript files with size 10. If unspecified, the plots will be drawn into X-windows 5 and 6 under Linux/OS X or into `ts.gif` and `dft.gif` under windows.

```
$ tsreduce playback ec04207.dat 100 5 [ts.ps/cps dft.ps/cps 10]
```
Replay the acquisition of data at an accelerated rate. `tsreduce plot` is called repeatedly with a minimum wait of 100 ms between updates. Each update advances the data by 5 observations. The optional parameters match those for `tsreduce plot`

```
$ tsreduce tracer ec04207.dat
```
Displays the latest raw observation frame in ds9, overlaid with a line indicating the movement of the target and comparison stars. This function is useful to monitor shifts in the field of view over a run.

```
$ tsreduce display ec04207.dat ec04207-1234.fits.gz
```

The frame `ec04207-1234.fits.gz` is bias/dark subtracted, flat fielded, and displayed in ds9. The apertures used for the reduction are overlaid. The requested frame must have been previously reduced via `tsreduce update`.

```
$ tsreduce reduce-range template.dat 10 15 0.5 20110703
```

Re-reduce the reduction file `template.dat` with apertures between 10 px and 15 px in 0.5 px increments. Output files will be saved as `20110703-<aperture size>.dat`.

```
$ tsreduce plot-range 20110703-[0-9]+.[0-9]+.dat
```

Display timeseries and DFT plots of the data created with `tsreduce reduce-range`. Aperture sizes can be cycled through with the `p` and `n` keys with the input window selected. This command only works under Linux and OSX.

```
$ tsreduce create-ts 2011-07-03 06:00:00 20110703.dat 20110704.dat \
 20110705.dat [...] july2011.ts
```

Combine a set of reduction files into a single timeseries containing relative $\mathrm{BJD_{TDB}}$ time, mma, and noise. The first two arguments specify the reference date and time to use as the zero of time. The remaining arguments are a list of (any number of) input file-names, followed by a single output file-name. The first input file is required to define the (J2000) target coordinates via the `RA` and `Dec` keywords.

The UTC $\rightarrow$ $\mathrm{BJD_{TDB}}$ time conversion is detailed in Appendix **??**.

```
$ tsreduce dft july2011.ts 100 10000 1 july2011.dft [july2011.freq]
```

Generate a dft of `july2011.ts` between 100 and 10000 μHz with 1 μHz increments. The output file is saved to `july2011.dft`. The optional parameter specifies a list of frequencies for prewhitening the data (see `tsreduce` source code for details).

```
$ tsreduce window july2011.ts 1000 800 1200 0.01 july2011.win
```

Generate a window function for `july2011.ts` using 1000 μHz sinusoid between 800 and 1200 μHz. Data is saved to `july2011.win`.

A.5 Troubleshooting

The online frame preview exits and restarts if the frame is updated while selecting an aperture:

This appears to be a bug in `ds9` which cannot be worked around. The best you can do is to monitor the exposure progress, and make sure that you aren't dragging an aperture when a frame readout finishes. If the exposure time is short, you may need to stop the acquisition while you select the apertures.

`tsreduce create` cannot find frames with special prefix characters:

`tsreduce` converts the frame prefix into a (POSIX compatible) regular expression in order to find the files on disk. Characters in the set `[]\^$.|?*+(){}` have special meaning in regular expressions, and must be escaped with a '\' when you enter the prefix into tsreduce.

Cannot configure online reduction if the first frame in a run is cloudy:

The reference frame queried during `tsreduce create` is used to determine the aperture positions and time-zero. You can specify a more suitable reference later in the run, then manually change the reference time if you want to avoid negative run times.

The mouse jumps around the screen when running Relay Mode:

The timer unit acts as a USB-to-Serial adapator for the attached GPS when running in relay mode. Windows may misinterperet the incoming serial data as data from a serial mouse and cause erratic cursor behavior.

You can deselect the serial enumerator in the FTDI advanced driver properties to disable this behavior.

A.6 Software updates

The three main software components of the Puoko-nui instrument are managed using the `git` version control system, and are currently hosted on the github service.

Acquisition software:	`https://github.com/pchote/Puoko-nui.git`
`tsreduce`:	`https://github.com/pchote/tsreduce.git`
Timer firmware:	`https://github.com/pchote/Karaka.git`

`git` is an extremely powerful tool for managing source code across multiple locations (originally created by Linux Torvalds to simplify development of the Linux kernel), but we will focus on just the necessary commands to fetch and reset the local code version to match that stored on github.

If you'd like more information, github provides a good introduction and tutorial at `http://learn.github.com/p/intro.html`, with more topics related to git and github at `https://help.github.com/`

The code can be acquired for the first time using the `git clone` command, e.g.:

```
$ git clone https://github.com/pchote/Puoko-nui.git puokonui
```

This will create a new directory `puokonui` in the current directory, containing the acquisition software source code.

If a local copy of the code exists, the current state of the upstream server can be queried using the `git fetch` command from the project directory:

```
$ git fetch -p
```

The `-p` flag tells fetch to *prune* any branches that have been deleted from the upstream server.

You can then *reset* the local code state to the `master` branch on the server, discarding any local changes:

```
$ git reset --hard origin/master
```

`origin/master` specifies the upstream branch to reset to. You can view a list of remote branches with the `git branch -r` command:

```
$ git branch -r
```

Once you have reset the local code to the latest version, you can `make` the updated executable:

```
$ make clean
$ make <project options>
```

The `Makefile` for each project starts with a list of options that can be configured. See each file for the options and their descriptions.

The overall process for updating the project will look something like the following

```
$ cd <project directory>
$ git fetch
$ git reset --hard origin/master
$ make clean
$ make <project options>
```

For completeness, a list of the main `make` commands are included below:

Linux

Acquisition (Puoko-nui):	`$ make CAMERA_TYPE=PVCAM`
Acquisition (ProEM):	`$ make CAMERA_TYPE=PICAM`
`tsreduce`:	`$ make`

Windows

Acquisition:	`$ make CAMERA_TYPE=PICAM`
`tsreduce`:	`$ make USE_READLINE=no`

Timer Firmware:

The timer unit contains a boot-loader which will be run at startup and enter a firmware update mode if any data is recieved within 4 seconds of connecting via USB.

The upgrade process is controlled by the `Makefile` in the source code directory. Several configuration options are defined at the top of the file; ensure that these are set correctly when `make` is invoked - incorrect values may leave the unit in a non-functional state that requires physical access to the unit to fix.

The standard firmware can be updated via USB by the following:

```
$ make clean
$ make install PORT=<port>
```

where <port> is the OS handle of the timer (e.g. `/dev/ttyUSB0` under Linux, `/dev/tty.usbserial-00001004` under OSX, or `COM5` under Windows).

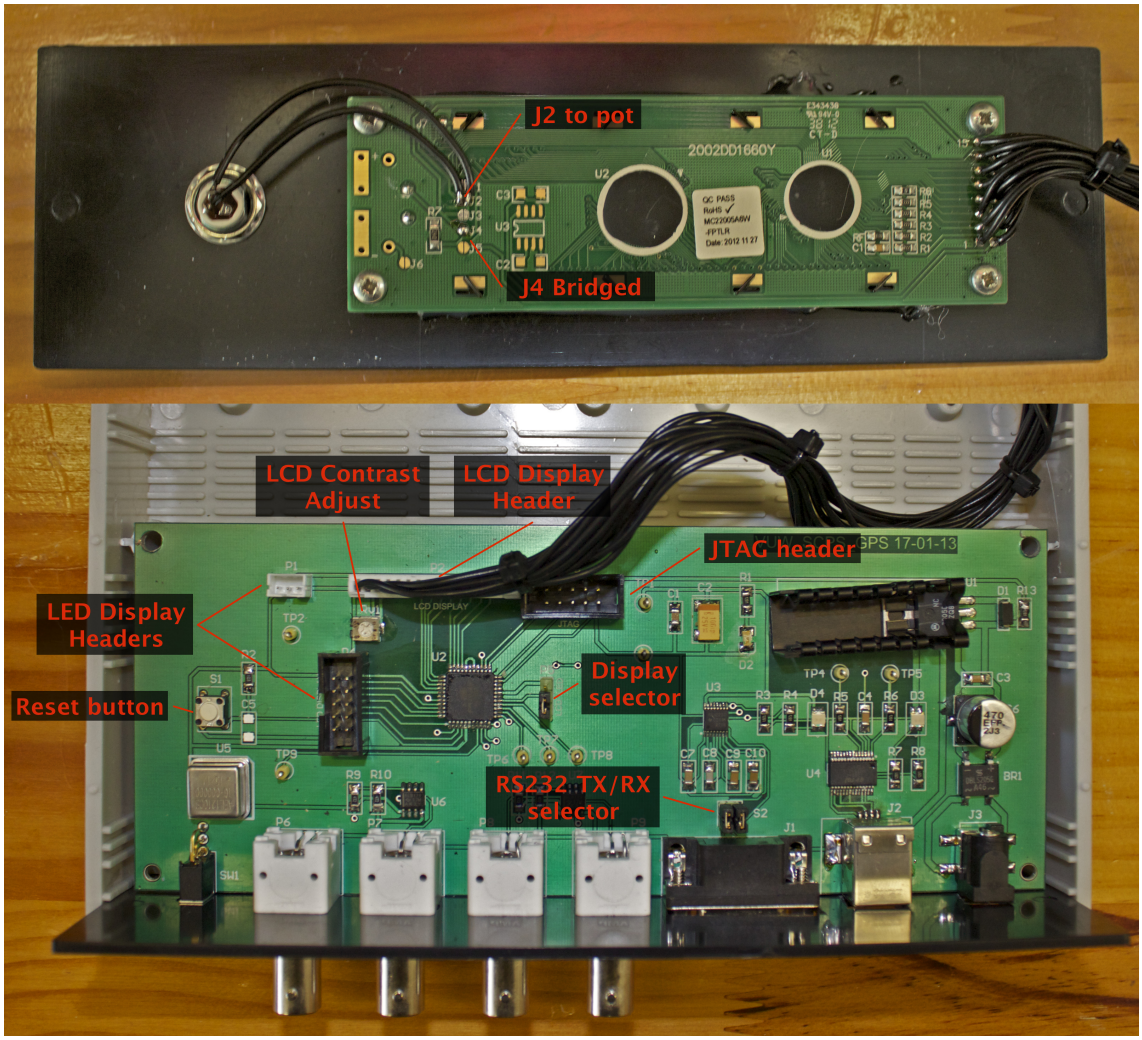

Figure A.7: Illustrated notes on the timer unit circuit board.

Timer Boot-loader:

The boot-loader cannot update itself via USB, and so boot-loader upgrades require the JTAG programmer.

After attaching the JTAG programmer, the boot-loader and firmware can be updated together by running:

```
$ make clean
$ make jtag
```