SCUBA-2 DR Pipeline Project Office
University of British Columbia
6224 Agricultural Road
Vancouver, British Columbia
CANADA
V6T 1Z1

Tel: +1-604-822-2211
Fax: +1-604-822-5324
Email: jmolnar@ubc.ca
WWW: http://scuba2.jach.hawaii.edu

Document Title: Data throughput testing for DREAM/Stare images

Document Number: SC2/ANA/S210/003

Issue: 1.8

Date: 2005/05/27

Document Prepared By:	Andy Gibb	Signature and Date:	2005-05-26
Document Approved By:	Douglas Scott	Signature and Date:	2005-05-26
Document Released By:	Janos Molnar	Signature and Date:	2005-05-26

Contents

1 Introduction

One of the unique features of SCUBA-2 is its rapid sampling rate, up to 200 Hz. Processing these data will be challenging and will probably require significant computational power. For the DREAM/Stare mode the effective data rate, as far as the data-reduction (DR) pipeline is concerned, will not exceed 1 Hz. The computational requirements are therefore more modest and may be achievable with current mid-range hardware.

SCUBA-2 will have four sub-arrays at each of the two wavelengths, with a dedicated data acquisition (DA) computer for each subarray. The DR pipeline will have a Quick-Look (QL) mode running for each wavelength which must display a processed frame before the subsequent frame is written to disk (referred to as 'near real-time'[6]). In DREAM/Stare mode, this means that the frames from the four subarrays must be combined and displayed within approximately 1–2 seconds. There will be a QL pipeline for each wavelength running on separate computers.

The online (or summit) DR pipeline must process an observation's worth of data before the end of the next observation.[6] Since this could lead to the undesireable situation where it takes 2 hours to process an observation lasting 2 hours, a modified approach is adopted whereby the pipeline processes each new file as it is written and thus the pipeline never falls behind by more than the file-closing interval. This interval is likely to be in the region of 1–5 minutes (although there is no requirement for it to be a particular value other than less than the time it takes for the file size to exceed 1 GB). For the purposes of this document, an interval of 1 minute has been adopted.

The raw data will be stored on a disk local to the DA computer for each sub-array. The data-processing machines running the respective pipelines will read the data from these disks (over NFS) and process any subsequent files locally on its own disks, displaying a co-added image for the observer and/or TSS.

In order to determine the computing requirements for the SCUBA-2 DR pipeline, it is desireable to know whether current hardware is capable of processing SCUBA-2 data quickly enough using existing tools or if new (specialized) applications and/or upgraded hardware are necessary. This report summarizes the results of tests for simulated DREAM/Stare data using two simulated pipelines. The first is simply a C-shell script while the second incorporates the same steps into an ORAC-DR recipe. This recipe does not necessarily represent a finished pipeline recipe, and includes sufficient processing steps to be a realistic method for testing the pipeline throughput.

2 Hardware

The machine used for the throughput testing is `aspersa.astro.ubc.ca`, with two AMD 64-bit Opteron CPUs, each running at 1.6 GHz. There is 2 GB of RAM installed, along with a 200 GB serial-ATA hard drive, spinning at 7200 rpm. The Fedora Core 2 Linux distribution is installed using the most recent kernel (2.6.10-1.770_FC2smp). The remote disk is attached to the UBC astronomy server, `ariel`, and accessed via NFS. Note that although `aspersa` has two CPUs, only one could be utilized for the testing.

3 Applications

The following Starlink tasks were used:

- `wcsalign`, `maths` and `display` from KAPPA[1]
- `makemos` from CCDPACK[3]
- `gaiadisp` from GAIA[2]

Starlink applications were used for each step, initially incorporated into a C-shell script and later called from within ORAC-DR as part of a recipe.

4 Procedure

The steps involved can be stated as:

1. Read raw data files and align to common coordinate system
2. Perform extinction correction
3. Mosaic four frames into a single frame
4. Co-add frame with previous data
5. Display frame

De-spiking and flat-fielding will be performed by the DA system. In practice, steps 3 and 4 can be combined using the Starlink `makemos` application provided that the coadded data and the raw frames are aligned to the same coordinate system.

The extinction correction can be done in two ways, either a full per-pixel correction ($e^{\tau_i A}$) or a simple correction for the elevation of the pointing centre ($e^{\tau_0 A}$). These two options are denoted by `FULL` and `QUICK` below.

In the simulation, the 'raw' data were stored on the (same) NFS-mounted disk on `ariel` and all subsequently-produced files were stored locally on `aspersa`. (Note that in practice, the raw data files will be read from four separate NFS-mounted disks. However, the initial reading over NFS does not appear to add significant overhead.)

To simulate a pipeline processing environment, and to average over a large number of repetitions, the shell script was run on a sample of 90 groups of files. The script was then rerun a number of times to average over minor fluctuations in the load on the system. Similarly, the ORAC-DR version was run on a sample of up to 300 groups of files.

4.1 Simulated raw data

The individual subarrays contain $40 \times 32 = 1280$ pixels. In DREAM mode, the DA computers will write frames that are larger by a few pixels in both directions since the secondary mirror will be jiggled. Data generated in DREAM mode by the simulator were used. As a comparison, a section extracted from a calibrated SCUBA 850-μm scan-map image was also used.

The reason for using the SCUBA image was to check that the various steps were being performed correctly. The processed simulated data were compared with the original full-size file and found to coincide to within machine precision.

4.2 Simulating the DA system

To simulate the DA system closing files so that ORAC-DR can begin processing, a program was used to create the flag (or `.ok`) files in the location defined by `ORAC_DATA_IN` at a rate equal to the 'file close' rate. As far as the pipeline is concerned, data then 'appears' on disk at the rate at which the flag files are written. The program (`drsim`[5]) permits the user to specify the interval at which files are closed, and as mentioned above, for the purpose of these tests this interval was set to 1 minute.

The quantity of data is equivalent to 5 minutes at a DREAM/Stare processing rate of 1 Hz, 10 minutes at 0.5 Hz and 20 minutes at 0.25 Hz. Thus the most stringent test is to reduce this volume of data in the shortest time (5 minutes). The number of images generated in DREAM/Stare mode in this time is $1 \times 60 \times 5 = 300$ per subarray for a total of 1200 images.

5 Timing results

5.1 Shell script

Display	Total elapsed time (sec)	Mean time per frame (sec)	Extinction?
`display`	41.15	0.457	N
`gaiadisp`	43.01	0.478	N
`display`	54.50	0.606	Y (Quick)
`gaiadisp`	55.60	0.618	Y (Quick)
`display`	57.20	0.636	Y (Full)
`gaiadisp`	57.60	0.640	Y (Full)

Table 1: Summary of processing times using different display engines

Table 1 shows the results. Note the 'time-per-frame' value is the processing time to combine the four sub-arrays into a single frame. The spread of measurements gives a dispersion in the mean time per file of approximately 2 ms ($\sim$0.5%). In the software requirements document,[6] pipeline requirement PR5 specifies that an operating margin of up to 30 % is desireable. The current test shown in Table 1 has a margin of at least 35 %, comfortably satisfying this requirement.

5.1.1 Time spent on each step in the shell script

As a way of estimating the impact of inserting further processing steps into the DREAM/Stare QL pipeline, the time to perform each step was estimated. The initialization step consists of `source`ing the relevant C-shell resource files for KAPPA and CCDPACK.

Table 2 shows the results, clearly demonstrating that displaying the resultant image is the largest single use of time - the mosaicking step looks significant on account of it being performed twice. However, the ORAC-DR[4] infrastructure contains a module that could be used which would reduce the time spent in this stage.

Step	Time (sec)	Fraction of total (per cent)
Initialize	0.005	0.8
Align	0.145	22.6
Extinction	0.082	12.8
Mosaic (2 steps)	0.193	30.2
Display	0.215	33.6
Total:	0.640	100.0

Table 2: Timings for each step of the test reduction shell script. The display task is GAIA.

5.2 ORAC-DR recipe

Two variants of a basic ORAC-DR recipe were used. The first was a direct copy of the steps performed in the shell script (denoted Recipe1). The second (denoted by Recipe2) is an attempt at a more realistic data reduction recipe and includes flux calibration, source extraction and flux calculation. In both cases the Full extinction correction method was used.

Recipe1

Data rate (Hz)	Processing time (sec)	Time available (sec)	Ratio
1	11/120	30/ 300	0.37/0.40
0.5	11/120	60/ 600	0.18/0.20
0.25	11/120	120/1200	0.09/0.10

Recipe2

Data rate (Hz)	Processing time (sec)	Time available (sec)	Ratio
1	20/253	30/ 300	0.67/0.84
0.5	20/253	60/ 600	0.33/0.42
0.25	20/253	120/1200	0.17/0.21

Table 3: Summary of processing times for the two ORAC-DR recipes. The data rate is the rate at which DREAM/Stare images are written to the output file, not the rate at which the files are closed. The ratio is that of processing time to available time and is equivalent to the mean processing time per image in seconds (compare with the mean time per frame in Table 1.

Table 3 shows the timing results for the ORAC-DR pipeline using the Quick and Full variants of the recipe. The total processing time is given for 120 (1200) images, equivalent to 1 (5) minute(s) at an image rate of 1 Hz. The increase in the ratio for the larger volume of data is due to the larger number of files to be combined (with a corresponding increase in the number of intercomparisons).

The co-adding step first mosaics the four subarray subscans together before coadding all of these mosaicked frames to form a single image. This method rapidly slows down the processing when many files are to be combined (since there are $N \times N$ intercomparisons). Thus it may be that for online or quick-look processing only a co-add to the current running average can be performed when images are written at 1 Hz for long observations. Alternatively, the mosaic step may be broken into smaller convenient blocks.

Compared with the simple shell-script version, the ORAC-DR recipe is 50 % faster, and therefore clearly within the near-real-time limit of 1 s or less for processing DREAM/Stare data. For the more realistic Recipe2 the margin is somewhat reduced at only ∼16 % although it increases (up to nearly 80 %) when the data rate is reduced. This margin is still within that specified in the pipeline requirements document.[6] It should be noted that, during the running of these tests, the system load never exceeded 85 % indicating that the processing is mostly not CPU-limited (although faster CPUs will undoubtedly help).

6 Conclusions

Existing (Starlink) applications on current hardware satisfy the requirement that data from the DREAM/Stare mode for SCUBA-2 be processed in 'near real-time': there is no need to write new applications for processing DREAM/Stare data. The current margin of at least ~ 20 % is, at worst, only slightly lower than that specified in the requirements document, but this is unlikely to pose a serious problem as it will be alleviated by the use of faster processors and/or a reduction in the image rate.

References

[1] Malcolm J. Currie and David S. Berry. KAPPA – Kernel Application Package. Starlink User Note 95, Starlink Project, CCLRC, 2002.

[2] Peter W. Draper, Norman Gray, and David S. Berry. GAIA - Graphical Astronomy and Image Analysis Tool. Starlink User Note 214, Starlink Project, CCLRC, 2003.

[3] Peter W. Draper, Mark Taylor, and Alasdair Allan. CCDPACK – CCD data reduction package. Starlink User Note 139, Starlink Project, CCLRC, 2002.

[4] Frossie Economou, Tim Jenness, Malcolm J. Currie, Andy Adamson, Alasdair Allan, and Brad Cavanagh. ORAC-DR – overview and general introduction. Starlink User Note 230, Starlink Project, CCLRC, 2002.

[5] Douglas Scott. SCUBA-2 SCUBA-2 Data Simulation and Pipeline Testing. SCUBA-2 Project SC2/SOF/S210/003, 2003.

[6] Douglas Scott. SCUBA-2 Data Reduction Software Requirements. SCUBA-2 Project SC2/SRE/S210/001, 2004.