VLBA Correlator Memo No. 99

Correlator Software Status, September 1990*

Don Wells (for the Correlator Group)
National Radio Astronomy Observatory, Charlottesville, Virginia

September 20, 1990

Abstract

A status report for the real-time software project for the VLBA Correlator for the period mid-July to mid-September 1990 is given. During this period the model-change and crossbar logic was developed, and first versions of the tape tasks began to function, including initial uses of the screens package as operator interface. An error logging facility has been developed. The station task now contains nearly all the final logic for controlling the FFT cards. The HCB driver code has been re-furbished in order to integrate it with the tasks and maintain it under NSE. NSE worked as intended during this period: it permitted the Group to work concurrently on shared code, with each Group member having a private development environment.

Contents

*as distributed for the Correlator teleconference held on Wed19Sep90.

1 Tasks

The Group continued to integrate tasks along an axis connecting the job script files which are produced by the DBMS and the HCB [Hardware Control Bus] which connects to the FFT cards, plus the `modelTask` and `tapeTask`. This axis is the vertical line of arrows in Figure 1 below, which is an updated version (an "as built" schematic) of Fig. 4 on p. 36 in VLBA Correlator Memo No. 95 (Sept. 29, 1989). It differs slightly from the Figure 1 shown in the July 1990 status memo; in particular, it now includes `ctskPerror` (error logging). The goal of this effort is to deliver a first version of the integrated task structure which will run jobs, commanding the hardware as it becomes available, but also able to run independent of the hardware for software debugging purposes.

The tasks `tickTask`, `tickTest`, `stnFftTask`, `modlTask`, `stnXbrTask`, `stnMchgTask` and `ctskPerror` are marked in the figure as "permanent unique"; the latter attribute refers to the fact that there is only one instance of each of these tasks, the former attribute refers to the fact that these tasks are spawned at boot time and run forever.

1.1 `stnTask`

Code has been added to `stnTask` to download control codes to FFT cards when the crossbar signal is received; this is the actual mechanism of setting the Correlator's FFT hardware to process signals from the PBDs. Code has also been added to evaluate delay models and download them into the FFT cards. The `stnTask` now monitors job time and commands model changes and mode changes as the stations change sources (`jobTasks` also command model changes every two minutes of observe time). We have declared `stnTask` to be "Done" even though much further development[1] of special cases and features remains.

1.2 `stnMchgTask` and `stnXbrTask`

These two coordinator tasks were completed during this period, and testing was in progress as the period ended. Testing of the actual model change and crossbar setting code in `stnTask` was done separately using kluge stimuli and appeared to be successful.

[1]For example, pulsar mode, Mark III Compatibility mode, rapid source chopping.

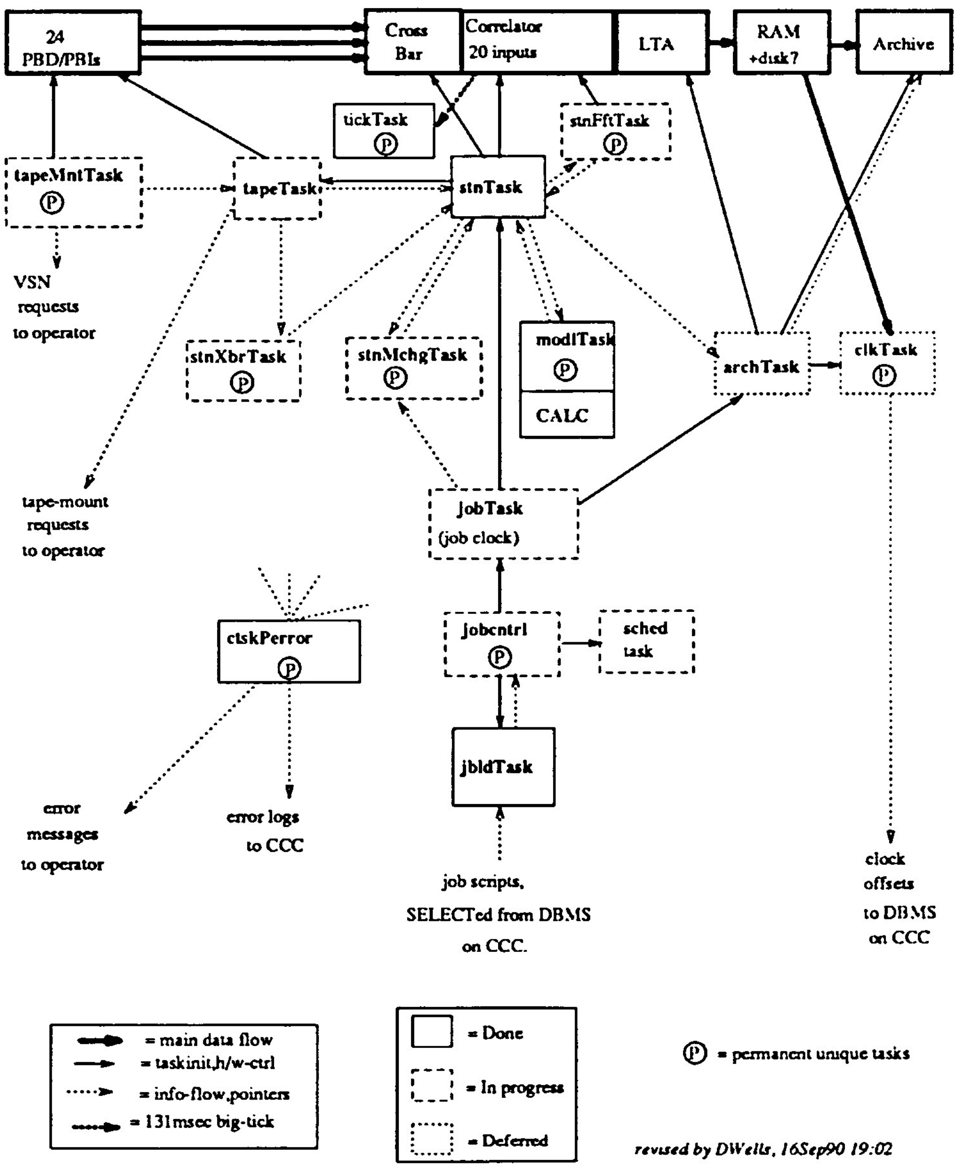

Figure 1: Real-Time Tasks

1.3 tapeTask and tapeMntTask

The experimental standalone tape manipulation code developed during 2Q has been integrated into the tasking structure. The code is now able to cause the motion of a PBD to follow the tape log in a job script as the job time increments. A test version of the code late in the period ran the two PBDs simultaneously, following the scripts for two stations. Further elaboration and testing of this code is proceeding. The code utilizes screens for its operator interface.

1.4 ctskPerror (Error Logging)

The first version of the error logging mechanism was completed near the end of the period. Tasks call function `ctskPerror` with arguments of error-level, error-number, message-format, and one or more variable arguments for the format statement. The *values* of these arguments (*not* their "names") are written into a vxWorks pipe device in binary form, along with a precision time stamp and certain task-specific information. An error logging task reads the pipe and uses the format statement and arguments to produce the formatted error message. The message is written to a file on CCC which is mounted with NFS (in the future log files will be written to a local disk when CCC is not reachable). The `ctskPerror` mechanism includes provision for renaming previously existing log files at the start of a new RT session, so that the programmer can keep old logs. The precision time stamps in the logs are expected to greatly aid the debugging of fully concurrent multiple job and multiple station operations during 3Q and 4Q90.

2 Use of NSE

No news is good news: NSE did its job during this period. Disciplined concurrent development of shared code is a reality.

Early in the period we changed our Makefiles so that we keep individual object files instead of combining them into libraries. Previously we had to recompile all object modules after an NSE "resync" operation because `make` could not properly utilize the timestamp information inside the libraries when doing its dependency analysis.

2.1 Component Structure

Table 1 shows the component structure as it stood on 11-September-90; this table was adapted from output produced by "`nsecomp list -r :`". The `:vxWorks` component contains components (directories) that execute under vxWorks, the `:sun` component is under SunOS, and the `:op-indep` component contains items that execute under both vxWorks and SunOS. The directory structure is essentially identical to the component structure.

3 RT Hardware News

A second PBD was delivered late in August. This forced a re-arrangement of the Correlator laboratory to conserve floor space.

The second HCB controller was completed near the end of this period, but has not been fully checked out. This controller has some new registers for front-panel oscilloscope test points, with LED indicators, plus pushbuttons and toggle switches ("sense switches" to you old-timers). The oscilloscope test points will facilitate precision analysis of the timing of software events. The buttons and switches will give flexible control of software modes when testing the system.

An additional Ethernet transceiver has been procured, which will enable all four of our RT CPUs to be re-booted independently (currently one of our CPUs lacks a transceiver and talks to the network through another CPU via the VME backplane; re-booting the second CPU forces re-boot of the first).

Component	*Nature*	*Status*
`:vxWorks:include`	*all* RT includes	
`:vxWorks:jobloader`	`jbldTask`	Done
`:vxWorks:hcb`	Device driver	Done
`:vxWorks:calc`	GSFC library	Done
`:vxWorks:model`	`modlTask`	Done
`:vxWorks:tick`	`tickTask`	Done
`:vxWorks:station`	`stnTask`, etc.	Done
`:vxWorks:job`	`jobTask`, etc.	Prototype
`:vxWorks:tape`	`tapeTask`, etc.	In progress
`:vxWorks:sched`		In progress
`:vxWorks:jobcontrol`		In progress
`:vxWorks:util:efc`	Event flag library	Done
`:vxWorks:util:tables`	table library	Done
`:vxWorks:util:ctsk`	tasking library	Done
`:vxWorks:util:ctsk:test`		
`:vxWorks:util:scripts:vsh`	Boot script	Prototype
`:vxWorks:util:scripts:tx`	CX script	`job1000.tx`
`:vxWorks:util:service`		
`:vxWorks:archive`		Deferred
`:vxWorks:clock`		Deferred
`:sun:include`		
`:sun:hcb`	Compute tables	Done
`:sun:hcb:angles`		
`:sun:hcb:fringe`		
`:sun:lib`		
`:sun:bin`		
`:sun:etc`		
`:sun:dbms`	schema, batch jobs	Suspended
`:op-indep:include`		
`:init`	Initialize Delivery	Done
`:doc:memo95`		History
`:doc:include`		
`:doc:misc_memos`		This memo

Table 1: Component Structure (as of 11Sept90)