The ADC prism separation is controlled by a Galil DMC-2200 motion controller. It is controlled by the ADC "dispatcher" software, which provides all higher-level control logic, including DCS monitoring and the ADC keyword service.

For the detailed design phase, we are implementing a nearly complete prototype ADC dispatcher in Tcl, rather than the C-language dispatcher called for in the conceptual design document. Tcl has the advantage of being a higher-level language than C, so we can develop code quickly and uncover any design problems before writing in C.

The ADC client-side KTL library is complete. As part of the development effort, we have automated the creation of the client library for any UCO/Lick "dispatcher2"-style service, starting from the keyword descriptions entered in the UCO/Lick "memes" database. The current set of keywords in the client library is visible at
    http://spg.ucolick.org/cgi-bin/Tcl/runRpt.cgi?target=k1adc&type=MemeByCon

Since we have written the complete prototype dispatcher in Tcl, we'd like to revisit the earlier decision to code the final dispatcher in C. The prototype ADC dispatcher is a solid design that can become the final ADC dispatcher with only a small amount of work. This would save one month of work compared to writing the final dispatcher in C. The use of Tcl in Keck code was addressed at the Keck SCC meeting of December 2003, in discussions that included Hilton Lewis and Al Conrad. Recognizing that Tcl is presently being used at Keck for a wide variety of critical uses, including serving KTL keywords for DEIMOS via the "infopatcher" program, it was agreed that Tcl should be treated as first-class supported language. Since Tcl already serves an essential role in many existing Keck applications, we suggest that the ease of development and maintenance of the package makes it a good choice for the final ADC dispatcher as well as the prototype.

The prototype dispatcher will be posted to the ADC web site later this week.

Compared to the ready-to-ship system, the prototype is missing these features:

(a) a "testbed" motor is being used, rather than the actual hardware that will be delivered.

(b) no testing has been done with DCS, since DCS has not yet been modified to support the ADC.

(c) there is no real mapping function to convert between prism separation and corrected zenith angle.

The design of the prototype differs in one significant way from the compared to the dispatcher design proposed in the conceptual design document: the conceptual design called for a threaded dispatcher, but the prototype is event driven. That is because the prototype is in Tcl, whereas the conceptual design anticipated a C application, and Tcl's built-in event loop makes it natural to do event-driven programming.

The prototype does retain a separate thread (actually, a co-process) for monitoring the DCS-demanded ADC mode and position. That's done because of the known problems with "DCS dropouts", in which a KTL client silently loses some of its keyword connections to DCS. We use a co-process to isolate all direct DCS interactions, making it simple to ensure that we can close and restart DCS communications without affecting the rest of the dispatcher.

The prototype itself was efficient to build, because it uses three supplementary packages to do the hard work:

(a) the KTcl package is used to communicate with DCS;
(b) the Traffic.tcl package is used for MUSIC messages, which is used for the k1adc service;
(c) the Galil package is used for Galil communications.

The last of these is a new package, and will be posted along with the prototype dispatcher later this week. All are in the CVS repository.

The conceptual design document noted that a standalone ethernet-based Galil, such as the DMC-2200, must be able to operate safely when network communications fail. This is handled via a Galil thread that provides a deadman switch. It periodically decrements a counter; if that counter ever reaches zero, the Galil halts the motion of the motor. The dispatcher is responsible for resetting the counter at a rate at least 1 Hz. This does not affect manual (pushbutton) mode, because in manual mode the Galil is electrically disconnected from motor control. The conceptual design document also called for a temperature-monitoring thread to ensure that the electronics enclosure does not overheat, even if the dispatcher is not operating. This thread has been descoped in the detailed design, because it assumes that the ADC electronics will be placed in the HIRES enclosure, and thus temperature control will be handled by the HIRES dispatchers.