The Keck II telescope control system

William Lupton
Hilton Lewis, Kevin Tsubota, Allan Honey, Sarah Quady

W. M. Keck Observatory
65-1120 Mamalahoa Highway, Kamuela, HI 96743, USA
email: wlupton@keck.hawaii.edu

Abstract

The Experimental Physics and Industrial Control System (EPICS) originated in the high energy physics community and has been used for several years to control accelerators. It is now in use or soon to be in use at several observatories around the world.

In 1995, it was decided that Keck II telescope would have a new EPICS-based control system rather than use a copy of the Keck I system. This decision was made because it was felt that EPICS provided a superior software infrastructure to that developed for Keck I, and that it would scale well to encompass adaptive optics and eventual use of the two telescopes for interferometry.

The new control system was developed throughout 1995 and the early part of 1996, leading to first light in January 1996, making it the first fully EPICS-controlled telescope control system in the world.

This paper describes how EPICS has been used to implement the control system, including a detailed discussion of the axes control, pointing and timing system, and of how they interact with each other.

Keywords: telescope control, Keck observatory, EPICS, VME, VxWorks

2. Introduction

The paper, in these proceedings, on "Software Engineering for the Keck II Telescope Control System[1]" discusses the reasons for writing a new control system for the Keck II telescope and gives the current status of the project. This paper therefore concentrates on technical aspects of the system.

The control system is based on the EPICS[2,3] software environment. This paper doesn't contain any tutorial information on EPICS but is nevertheless intended to be intelligible to people who don't know anything about EPICS.

3. System overview

The control system is divided into well-defined subsystems, each of which has been developed and tested independently. The data flows between the main subsystems are illustrated in Figure 1, from which it can be seen that the subsystems run asynchronously and have remarkably simple interfaces.

The pointing, axes control and time subsystems are considered in somewhat more detail.

3.1. Pointing

The pointing (PNT) subsystem implements Patrick Wallace's Keck I proposals,[4] splitting processing into "fast" (20Hz), "medium" (every 5sec) and "slow" (every 5min). Pointing data flow is illustrated in Figure 2.

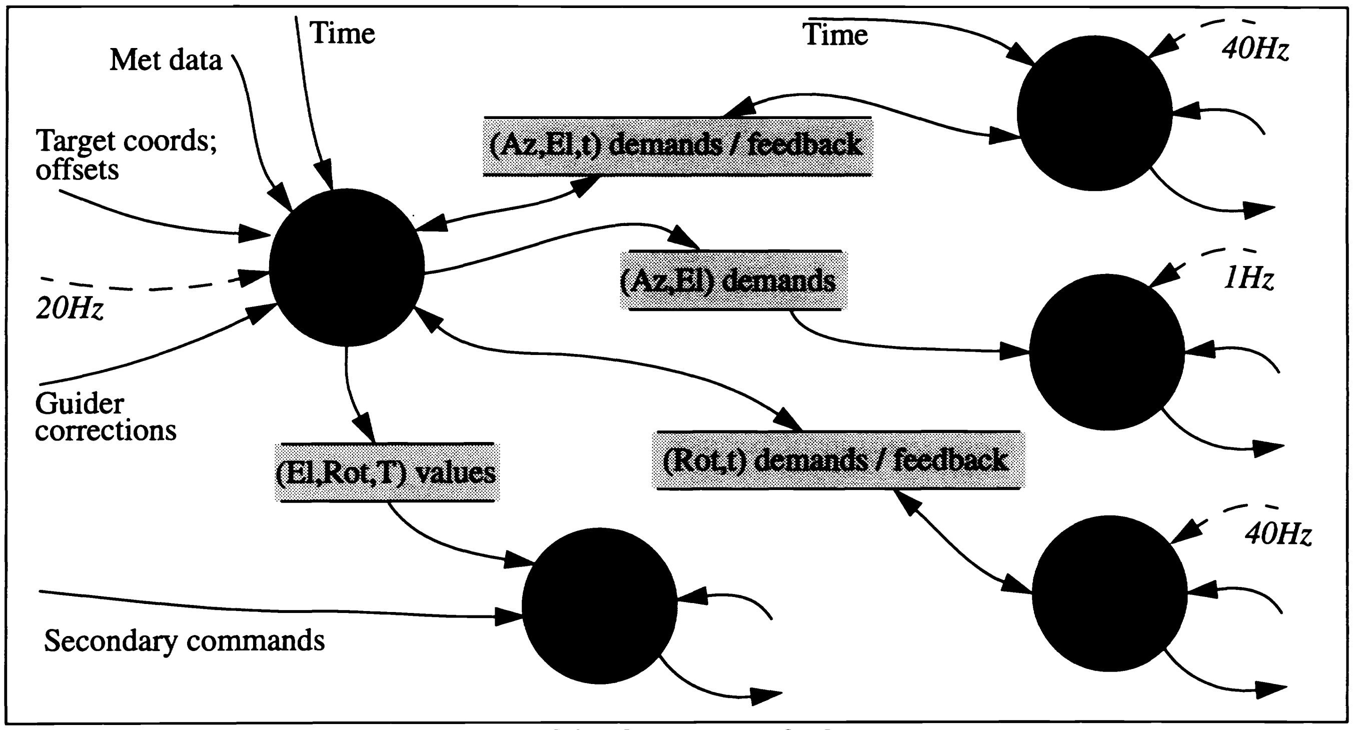


Figure 1. Main telescope control subsystems

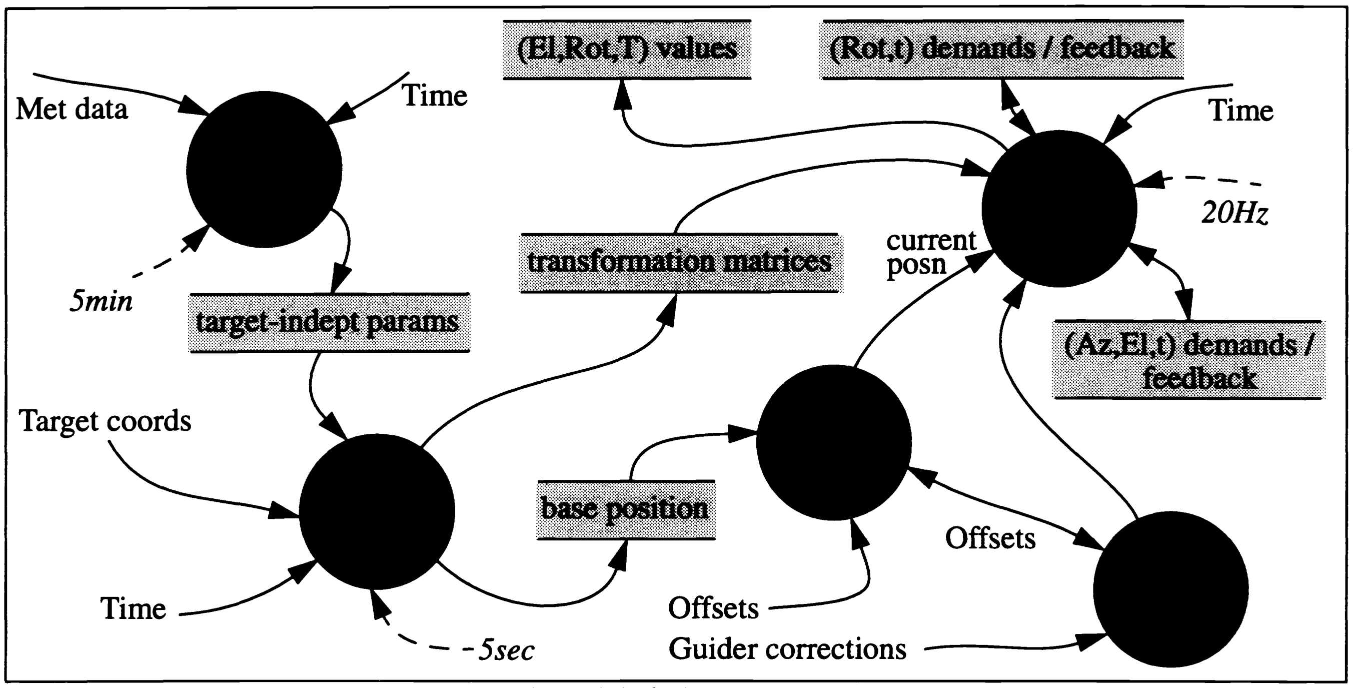


Figure 2. Pointing subsystem

The heart of the pointing code is the fast 20Hz processing, which uses pre-calculated transformation matrices to convert the current target position to a time-stamped (Az,El) demand. The transformation matrices are calculated every five seconds, and slowly-varying target-independent parameters such as atmospheric refraction are calculated every five minutes.

Timing statistics for the pointing subsystem are given in Section 6.2.

3.2. Axes control

The axes control (AXE) subsystem is responsible for closing the position loop for the telescope's azimuth and elevation axes. Data flows are illustrated in Figure 3.

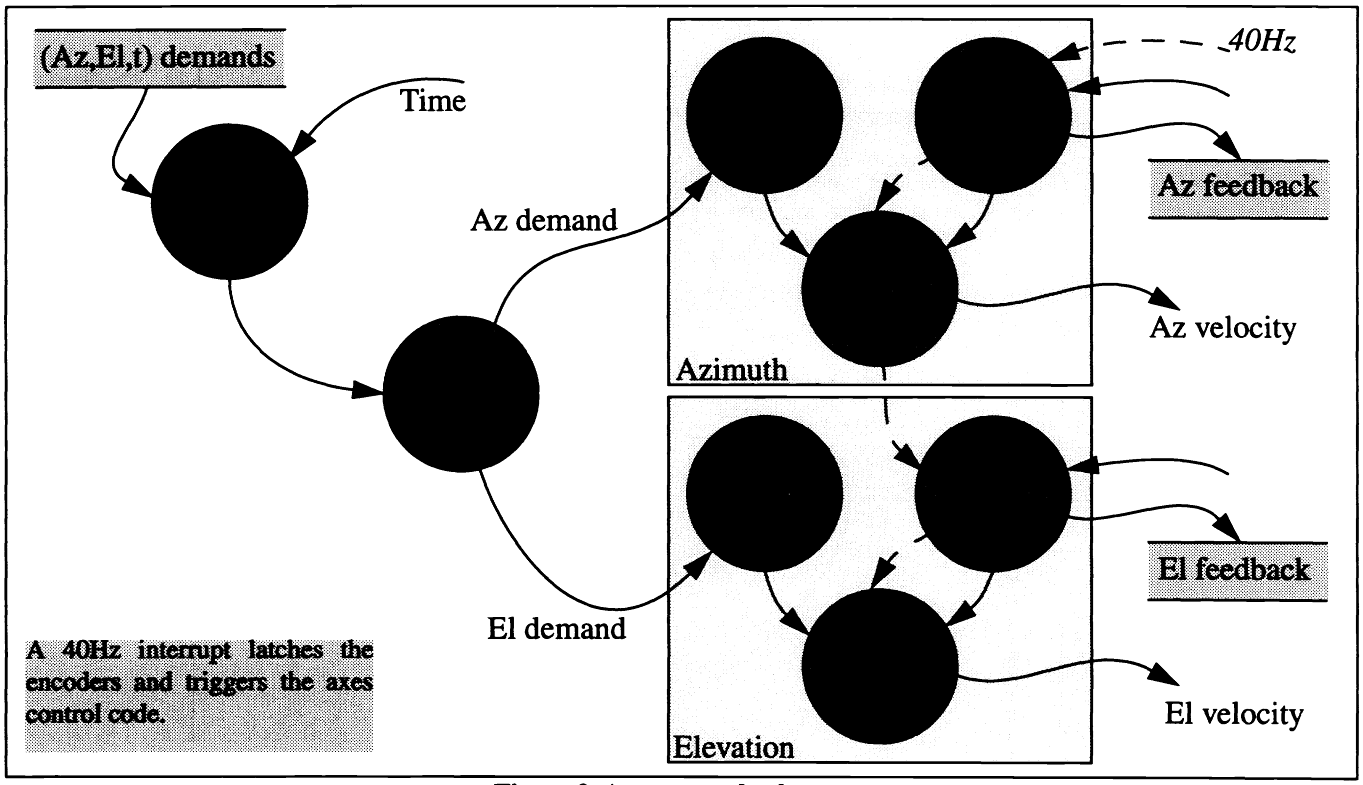


Figure 3. Axes control subsystem

Time-stamped (Az,El) demands from the pointing subsystem are interpolated to the current time, then passed through trajectory control processing (which knows about limits and forbidden regions), through servo pre-processing (command shaping) to prevent servo saturation, and finally into the servos.

The subsystem runs at 40 Hz and is asynchronous to the pointing. A single 40Hz interrupt, generated by the time system, latches the encoders and triggers the software, which thus always runs just after the encoders have been latched. Processing does not occur in the natural data flow order of the previous paragraph. This is because it is important to minimize the time spent in the "critical region" between when the encoders are latched and the last velocity demand is output. Processing order is illustrated by labels ❶, ❷ etc. in Figure 3:

❶ *Start of critical region.* Az encoder processing reads latched encoders and uses pre-calculated coefficients to generate corrected encoder position. Az encoder triggers Az servo.

❷ Az servo executes, using pre-calculated Az demand. Az velocity demand is output. Az servo triggers El encoder processing.

❸ El encoder processing reads latched encoders and uses pre-calculated coefficients to generate corrected encoder position. El encoder triggers El servo.

❹ El servo executes, using pre-calculated El demand. El velocity demand is output. *End of critical region.* El servo continues with post-processing and causes El shaper to execute.

❺ El shaper causes trajectory control to execute.

❻ Trajectory control causes interpolation to execute.

❼ Interpolation interpolates the (Az,El) demands for the time of the next execution of the 40Hz code.

❽ El servo and encoder post-processing complete, and Az servo post-processing causes Az shaper to execute. On completion of Az encoder post-processing, all coefficients for the next iteration have been calculated and 40Hz processing is complete.

Timing statistics for the axes control subsystem are given in Section 6.2.

3.3. Time system

The time (TIM) subsystem is responsible for generating the 40Hz interrupt which latches encoders and triggers axes processing, and for providing absolute time (in a variety of timescales) on demand.

The TYMSERVE 2000 GPS receiver / NTP server is used, which sends time via IRIG-B (over co-axial cable) to Bancomm bc635 VME boards (one per VME crate). EPICS support is provided by the UKIRT Bancomm driver. This arrangement is illustrated in Figure 4.

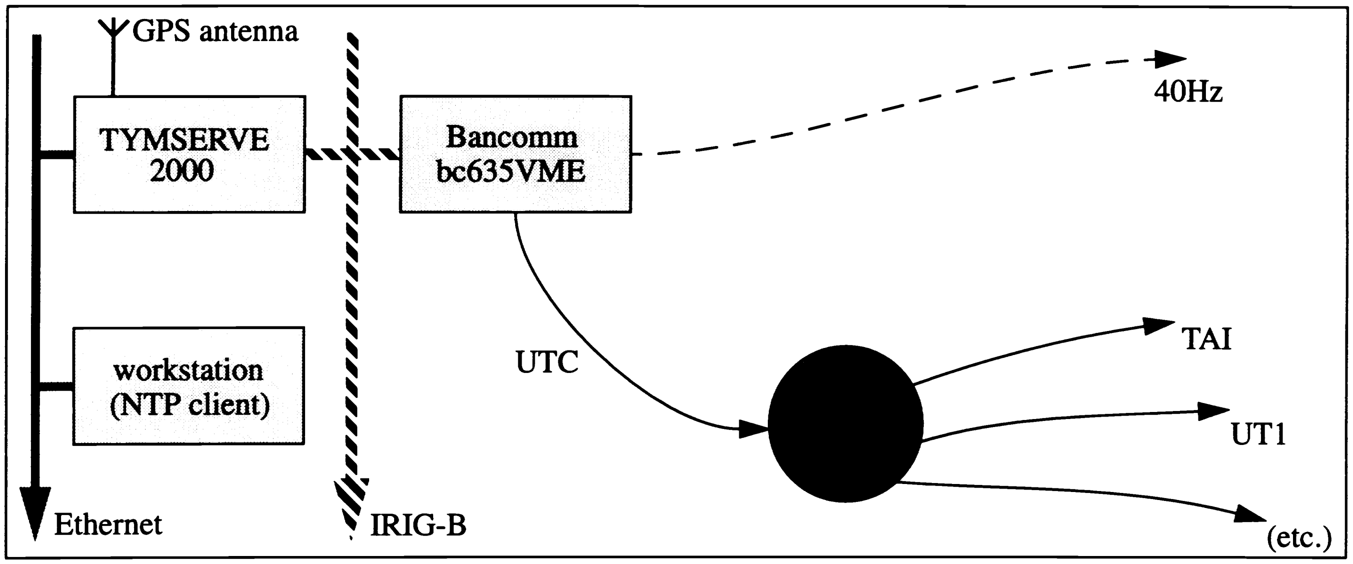


Figure 4. Time subsystem

4. Implementation

This section explains how EPICS tools have been used to implement the control system.

4.1. Hardware environment

The control system runs on two VME systems (three once the chopper software has been completed). All use Force CPU/40 (25MHz Motorola M68040) boards.

1. TDC (telescope drives controller) runs pointing and axes control.
2. ASC (auxiliary systems controller) runs dome, rotators, secondary and data acquisition.
3. CIE (chopper interface electronics) will run the chopping secondary.

User interfaces all run under Unix (SunOS 4.1.x; soon to be Solaris). Refer to Figure 5 for an illustration of these computers and their responsibilities.

4.2. EPICS database

The system overview of Section 3 was deliberately biased towards illustrating data flow. This is because a significant part of the system has been implemented directly from data flow diagrams.

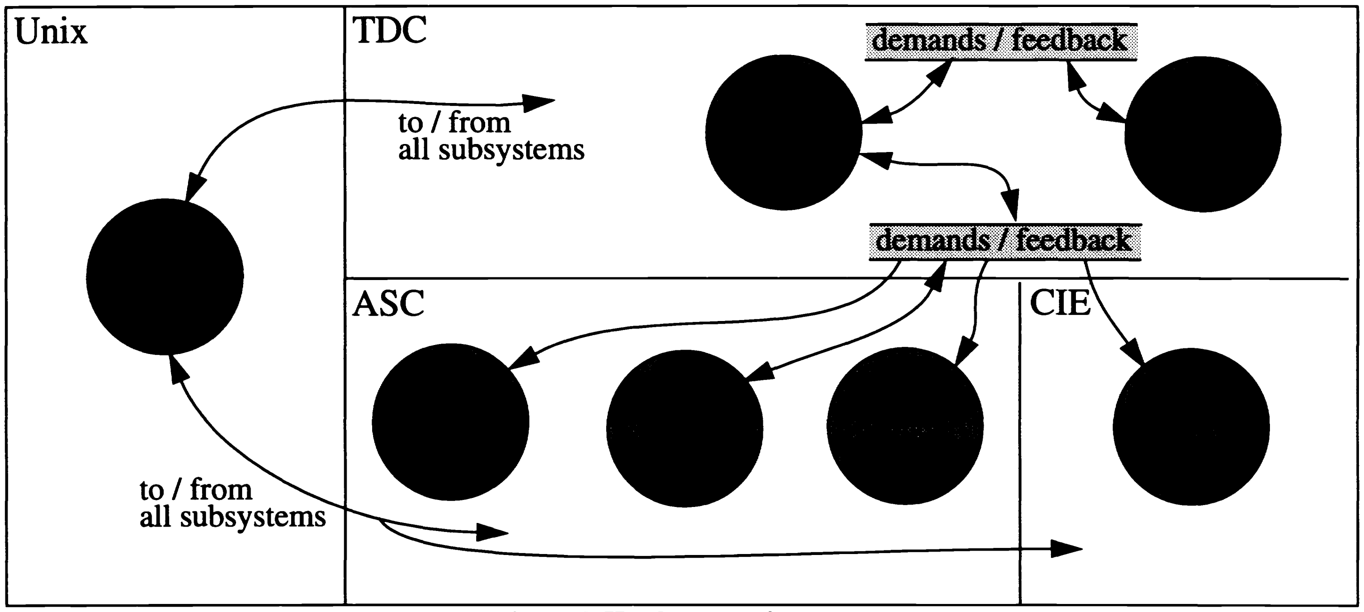


Figure 5. Hardware environment

A commercial electronics circuit design package (Capfast) is used to "draw" these diagrams, using "wires" to connect objects. EPICS refers to these objects as "records". Each record has an associated type, e.g. "ai" (analog input), "bo" (binary output), "calc" (calculation) or "sub" (subroutine).

Most records, including all those mentioned in the previous paragraph, correspond to code supplied as part of the EPICS system. Where special processing is required, custom records can be written. The control system uses eleven custom records and it should perhaps come as no surprise to discover that they correspond exactly to the processing "bubbles" in Figures 2, 3 and 4.

Subsystem	**Custom records**
pointing	slow, medium, fast, offset, guider
axes	interpolation[a], trajectory, shaper, servo, encoder
time	convert

Table 1. Custom records

a. A general-purpose interpolation record has also been written (it's already used for rotators) and will soon replace the axes-specific interpolation record.

The Capfast drawings, which are processed automatically to produce a downloadable EPICS database, clearly mirror the structure already presented. Figure 6 shows the top-level pointing subsystem Capfast drawing.

4.3. Coordination

EPICS database processing handles steady-state operation but cannot handle the coordination necessary for initialization, initiating moves, handling faults and so on. Such coordination is performed by EPICS sequences. Source code written in terms of states, events and transitions is pre-processed into C (and then compiled).

The standard state model illustrated in Figure 7 has been applied to the axes, dome, rotator and secondary subsystems.

4.4. Device access

All telescope hardware is accessed via VME interface boards. Most of these boards are the same as the ones in use on Keck I.

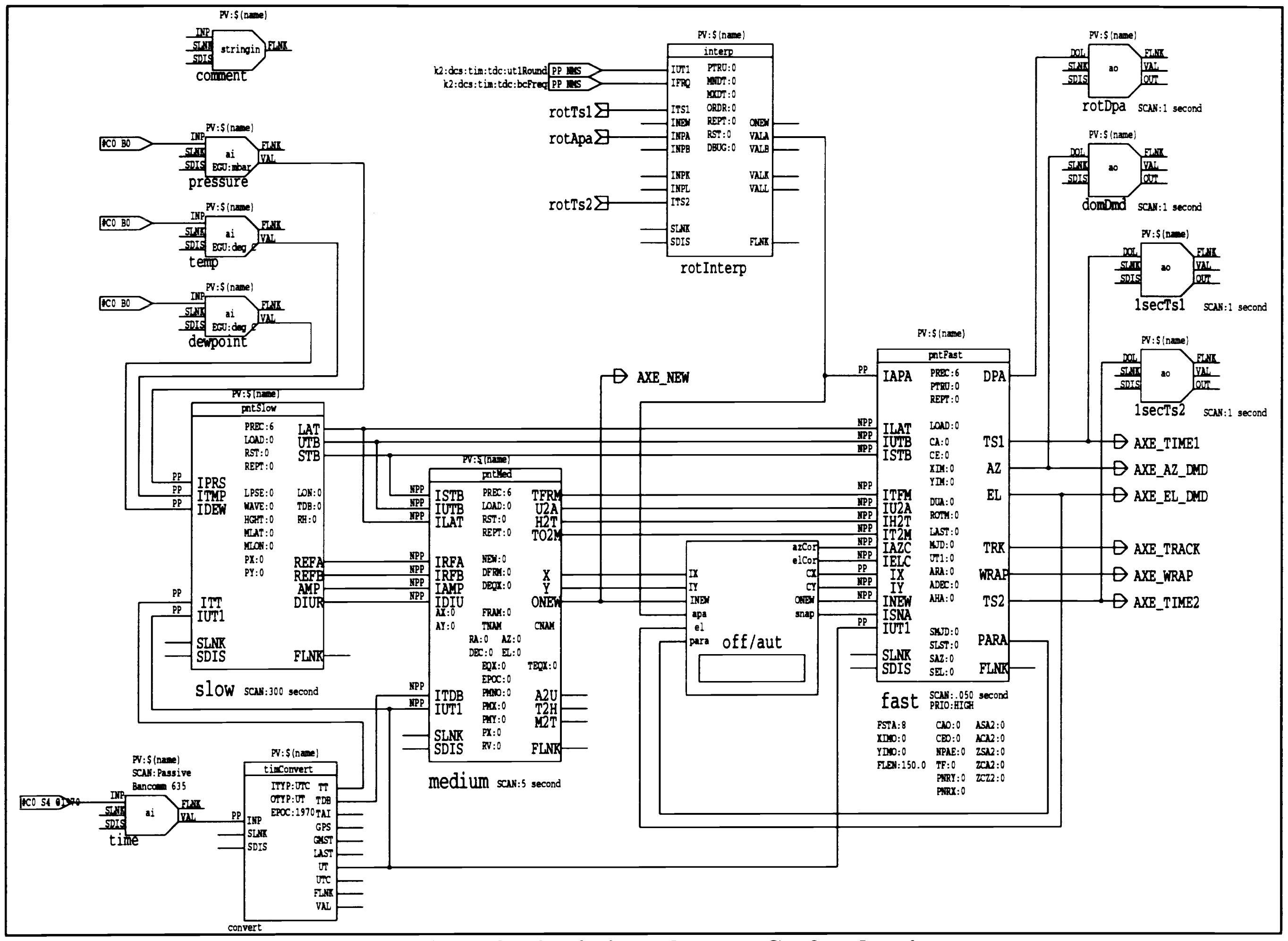
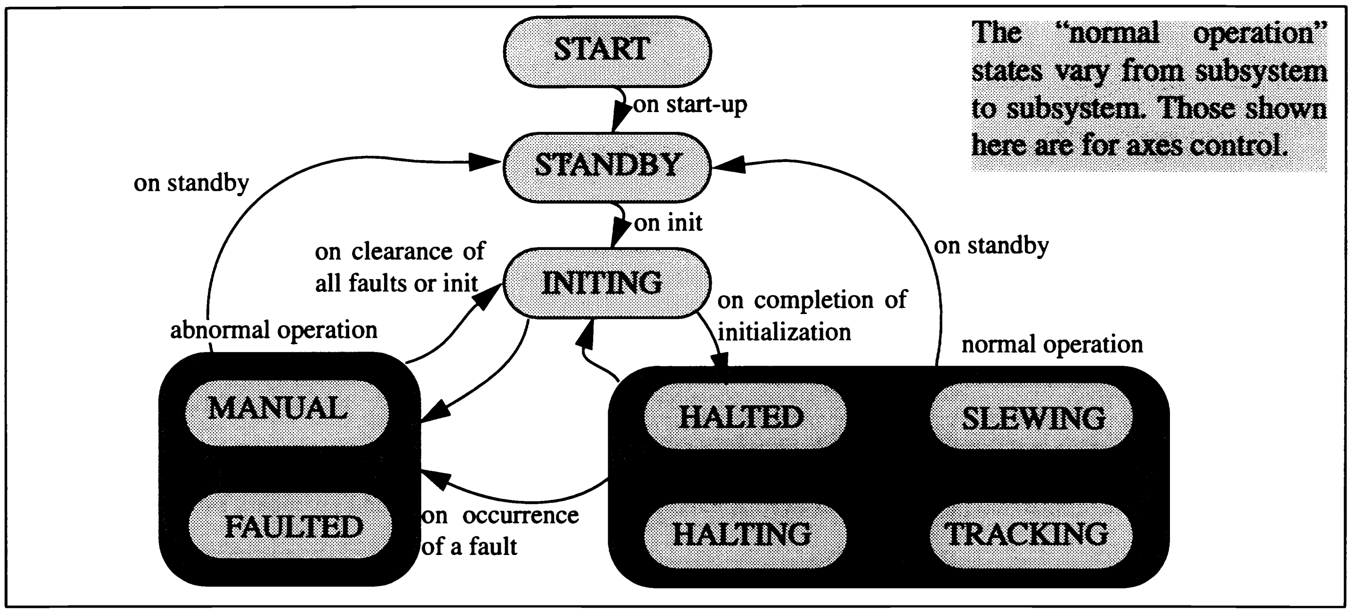


Figure 6. Top-level pointing subsystem Capfast drawing



START
on start-up
STANDBY
on standby
on init
on clearance of all faults or init
INITING
on completion of initialization
on standby
abnormal operation
normal operation
MANUAL
FAULTED
on occurrence of a fault
HALTED
SLEWING
HALTING
TRACKING
The "normal operation" states vary from subsystem to subsystem. Those shown here are for axes control.

Figure 7. Standard state model

EPICS requires a device driver to convert between EPICS device-independent concepts such as "analog input" (e.g. a channel on an ADC board) or "binary output" (e.g. a bit on a digital output board), and the details of the hardware.

For some boards, EPICS drivers were already available. Most new drivers were written by contract programmers who already had extensive EPICS experience.

Interface	Description	Driver
Bancomm bc635	time and frequency processor	UKIRT
Data Translation dt140x	digital to analog board	contract
XYCOM xy212	digital input board	existing
XYCOM xy220	digital output board	existing
XYCOM xy540	analog to digital board	contract
Keck counter card	custom encoder interface	contract
Force SIO-2	serial board, interfaces to dome (Allan-Bradley PLC), secondary (Compumotor) and barcodes	Keck serial driver; contract generic serial support; contract device-specific support

Table 2. VME interface boards and EPICS drivers

5. User interfaces and tools

5.1. EPICS monitors

A very powerful attribute of EPICS is that any record field can be accessed directly by name from a remote machine (e.g. a Unix workstation). In addition, an application can request that it be notified whenever a record field changes value or severity (e.g. an axis goes into a limit). Such a notification is referred to as a "monitor".

Monitors are used extensively in user interfaces, the alarm handler, and for archiving data. Refer to Figure 8 for an illustration of their use.

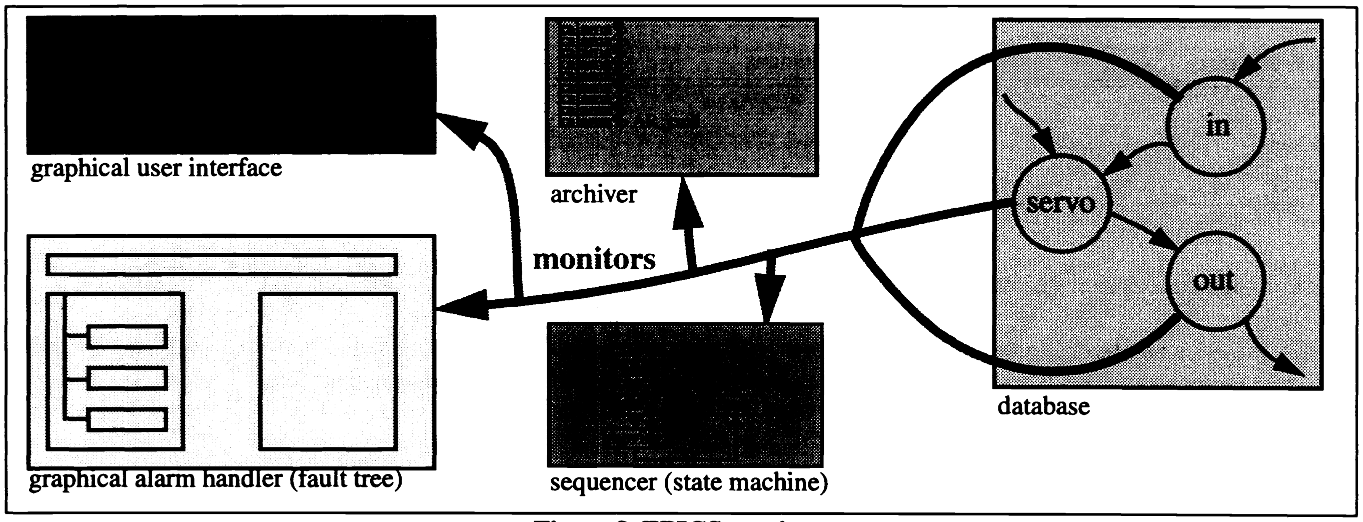

graphical user interface
archiver
monitors
in
servo
out
database
graphical alarm handler (fault tree)
sequencer (state machine)

Figure 8. EPICS monitors

5.2. Graphical interfaces

EPICS provides a graphical editor / display manager (edd/dm), which allows fast interactive creation of engineering displays. This has been used extensively; Figure 9 shows an example screen. The Motif editor / display manager (medm) has not yet been used at Keck.

Figure 9. Example edd/dm screen

A generic layer between the existing KTL (Keck Task Library) keywords and the corresponding EPICS record fields has been created. This allows existing Keck I keyword-based user interfaces to be run on Keck II and has eased the transition from Keck I to Keck II for the observing assistants. The main such user interfaces are

User Interface	Environment	Description
sky	Motif + dataViews	catalog search and slew to target
facsum	dataViews	high-level telescope status display
xguide	Motif	guider and offsetting control

Table 3. Graphical user interfaces

The commercial dataViews package is now used for new user-level (non engineering) GUIs.

EPICS provides an alarm handler, which displays a fault tree, alerts the operator to the location of any problem, and offers guidance on recovery. This is a very valuable and powerful tool; Figure 10 shows an example screen.

5.3. Data archiving

The EPICS archiver tool is used for saving engineering and calibration data. It works by monitoring specified EPICS record fields. Each monitor message includes a time-stamp, the field name, its alarm state and its value. For example, it can be given a list of the field names which correspond to the inputs and outputs of the azimuth interpolation, trajectory, shaper, servo and encoder records. It will then write these values (most of which change at 40Hz) to a disk file for off-line analysis. This can be done on the running system without affecting its operation.

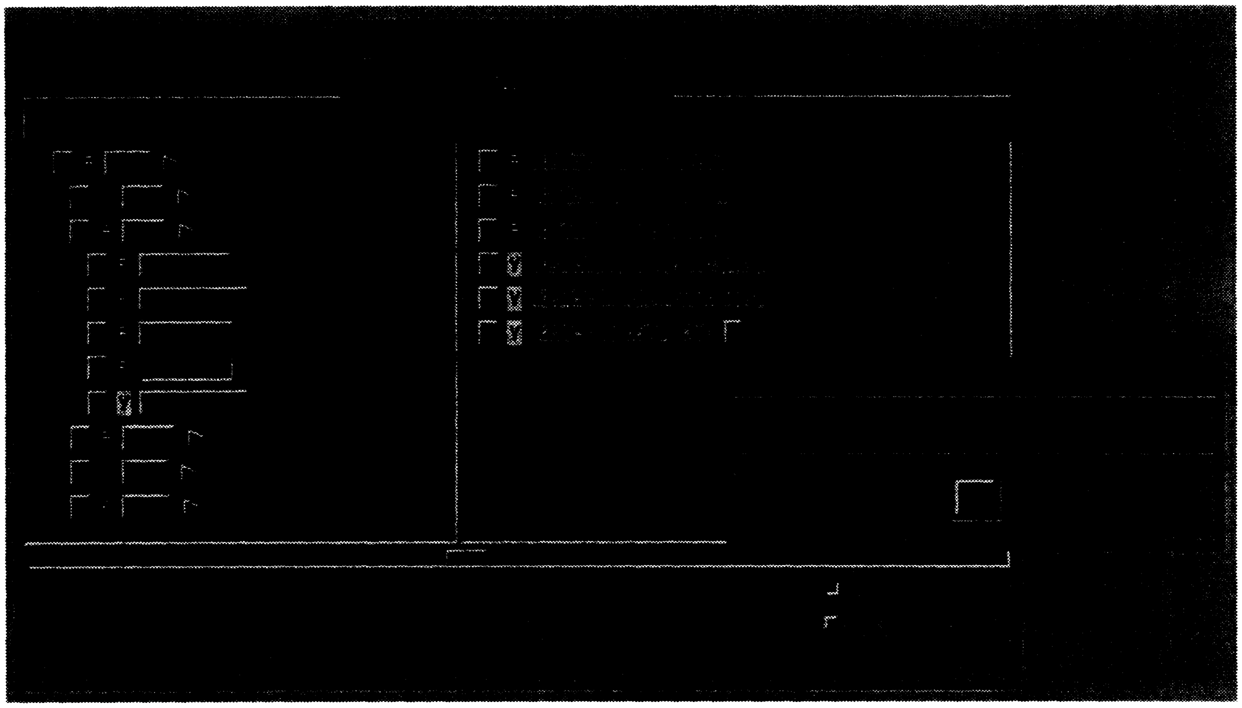


Figure 10. Example alarm handler screen

A set of observatory-wide quantities is being defined, which will be sampled throughout the night (every night) every five seconds or so. This will give a complete time-stamped record of system behavior at this time resolution. Adding or deleting a quantity to be logged merely involves editing a configuration file.

Archiver data can be analyzed using a standard EPICS tool, but IDL is instead being used because of its superior data analysis facilities. It is also intended that archiver data be ingested into a Sybase database but plans for this are at an early stage.

5.4. Support tools

The control system has been developed using the UAE development environment. This provides mechanisms for organizing directories, building executable files, and managing releases. It is based on initial work at Keck and is now a collaboration between Keck and UKIRT. It is used by several groups in the EPICS community and has been recommended for use by all Gemini software developers.

Various other tools and utilities have been developed for doing things like managing summit software versions, loading initial values for record fields, and analyzing calibration data.

6. Statistics

6.1. Database statistics

The TDC database currently has 219 records, of which 88 are attached to hardware.

Subsystem	Number of records
AXE	138
PNT	40

Table 4. TDC EPICS database

Subsystem	Number of records
MON[a]	32
TIM	9

Table 4. TDC EPICS database

a. These are hardware records which are monitored for faults, but are not needed for system operation.

The ASC database currently has 844 records, of which 289 are attached to hardware.

Subsystem	Number of records
ROT	485
SEC	283
DOM	67
TIM	9

Table 5. ASC EPICS database

Numbers of records will increase as new facilities are added and more hardware is monitored for faults.

6.2. Timing statistics

Execution times for the pointing records have been measured.

Record	Execution frequency	Execution time (25MHz M68040)
Fast	20Hz	1775μs
Medium	Every 5 seconds	375μs
Slow	Every 5 minutes	144ms

Table 6. Pointing timing statistics

So also have execution times for the axes control records.

Record	Execution frequency	Execution time (25MHz M68040)
Encoder	40Hz (azimuth and elevation)	2100μs (azimuth); 1550μs (elevation)[a]
Servo	40Hz (azimuth and elevation)	550μs (each)
Shaper	40Hz (azimuth and elevation)	150μs (each)
Trajectory control	40Hz	200μs
Interpolation	40Hz	800μs

Table 7. Axes control timing statistics

a. Different because azimuth has two encoders

The critical region between the 40Hz interrupt and the final velocity demand being written to the hardware is about 1.2ms, rather over the specification of 1ms. Analysis shows that application code does not start executing until 350μs after the interrupt and that the first (azimuth) velocity demand is written 800μs after the interrupt. The 350μs delay is caused by the generic routines which wake up the application code; it can probably be reduced substantially by use of more specific wake-up code (which seems justified in this special case).

A total of 61 records are processed during the 40Hz axes processing. 12 of these are partially or completely processed during the critical region.

7. Current developments

High-level coordination of subsystems is being implemented, the aim being to present the user with a unified telescope rather than a set of independent subsystems. This will include new GUIs.

Autoguiding is currently using Keck I guider cameras and software.

1. The interface between the guider and the telescope has being changed to use the new EPICS system.
2. A general-purpose guider/telescope interface has been defined, which will handle guiding corrections from guider cameras, instrument detectors or the adaptive optics system.

Pointing code was originally a direct port of the Keck I algorithms. These are being enhanced to implement a reverse pointing transformation and a virtual guide telescope.

Keck I chopper code is being ported to the Keck II EPICS environment. The new system will control both the Keck I chopper and the Keck II Infrared Fast Steering Mechanism (IFSM).

Dome shutter control will initially be open loop. Some hardware work is required before closed loop control can be implemented.

8. Future developments

User interfaces will become simpler and more intuitive, will usually be implemented in dataViews, and will make maximal use of graphical techniques for representing information.

The Keck II control system is to be retro-fitted on Keck I. Keck I hardware is similar but not identical to Keck II. Hardware differences must be addressed, and naming conventions must be applied rigorously so that the two systems can run independently or (in the future) together.

Interfaces between adaptive optics and telescope subsystems are currently being designed. The telescope, when asked to offset, will "ask permission" of the AO system before proceeding. This will allow AO loops to be opened before the telescope offset begins, and then closed again on completion of the offset.

Procedures such as pointing tests and rotator center calibration will be automated.

9. Acknowledgments

The following members of the Keck software group have been involved in this effort.

Person	Areas of responsibility
Hilton Lewis	porting of axes and pointing algorithmic code, pointing integration, overall pointing and axes control expert
William Lupton	development environment, build procedures, support tools, overall design, time support, calibration data analysis, EPICS support
Kevin Tsubota	axes control and initial pointing integration, dome control, rotator control, Capfast support
Allan Honey	device and driver support, secondary control

Table 8. Software group members

Person	Areas of responsibility
Sarah Quady	board-level monitoring implementation, displays, secondary control
Al Conrad	KTL user interfaces, KTL / EPICS interface
Myrna Tsubota	telescope simulator, meteorological data server, guider interface
Liz Chock	meteorological data server, environmental data server

Table 8. Software group members

Nick Rees wrote the UKIRT Bancomm driver (with contributions from William Lupton). Rees and Lupton have collaborated on the UAE development environment.

The W. M. Keck Observatories are operated as a scientific partnership between the California Institute of Technology and the University of California. The observatories are made possible by the generous financial support of the W. M. Keck Foundation.

10. References

1. H. Lewis, W. Lupton, K. Tsubota, A. Honey, S. Quady, "Software Engineering for the Keck II Telescope Control System," *in these proceedings*.
2. R. Dalesio, M. Kraimer, A. Kozubal, "EPICS architecture," in *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems*, KEK, Tsukuba, Japan, pp. 278-281 (or `http://www.aps.anl.gov/asd/controls/epics/EpicsDocumentation/WWWPages/EpicsDocs/Epics-General/EPICS_Architecture.ps`) (1991).
3. W. Lupton, "Software Infrastructure for the Keck II Telescope," in *Telescope Control Systems*, Patrick T. Wallace, Editor, Proc. SPIE 2479, pp. 140-151 (1995).
4. P. Wallace, "Proposals for Keck telescope pointing algorithms", *Appendix A to Keck Drive and Control System Software Design Requirements Document CARA 106* (1987).