Software support for X2000 Hardware

Blair Lewis / Len Day

Blair.F.Lewis@jpl.nasa.gov /

Len.Day@jpl.nasa.gov

June 6, 2001

Agenda

- Charter
- Software overview
- Abstract Device Model
- Device drivers
- 1394 Hardware Abstraction Layer (HAL)
- Test tools
- CPU Performance measurement
- Development status

Charter

- Among other things the Europa Orbiter Operating System and Avionics Support team is chartered to:
  - Provide flight device drivers for all X2000 hardware
  - Use these drivers to support the X2000 hardware integration and test
  - Provide test tool support for testing / exercising the X2000 hardware

Software Overview

- To achieve these ends the following have been developed:
  - Abstract Device Model (ADM)
    - An object oriented framework containing the common code for implementing drivers such as I/O queuing, I/O completion notification, event reporting
  - Concrete instantiations of the ADM for each of the X2000 devices
  - A complimentary implementation called "Command Lists" which enables repetitive execution and allows commands for various devices to be executed with specified temporal relationships
  - A test tool called LLE (Low Level Exerciser) useful for both software and hardware test

Software overview cont

- Portability
  - Primary target is VxWorks, nevertheless we are maintaining portability
  - Minor portions of the code such as hooking interrupts are necessarily OS-dependent
  - All other OS interaction is done using a wrapper called ACE (Adaptive Computing Environment) from the University of Washington
    - ACE is ported to many environments
  - We currently do much of our software testing under Solaris using simple software simulators instead of hardware
  - Full bit-level simulator hooks in place, simulators to be developed in the future.
  - All drivers in C++

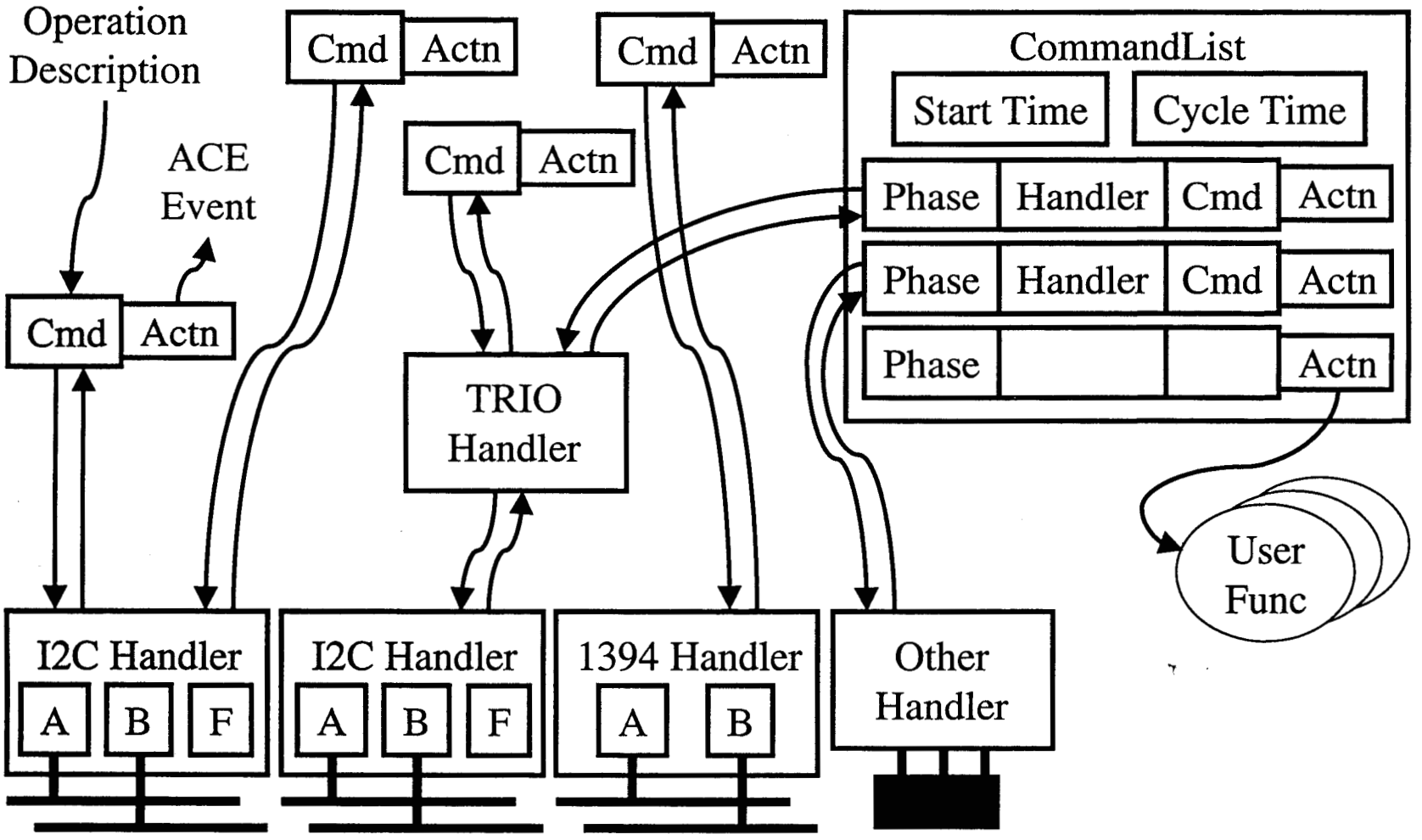
Abstract Device Model

- General goals:
  - Provide a uniform functional interface across a set of device drivers
  - Provide general-purpose methods to handle sets of related I/O operations
  - Provide a flexible framework for device driver implementation

Abstract Device Model

- Use objects to represent actions
  - Command objects represent I/O operations.
  - Action objects represent how a user is to be notified of an asynchronous operation's completion.
- Use a Command/Handler model
  - Users (clients) send Commands with attached Action objects to a Handler that executes the command and uses the Action objects to perform the notification of completion.
  - Actions can set semaphores, do callbacks, set bits, virtually anything

Abstract device model



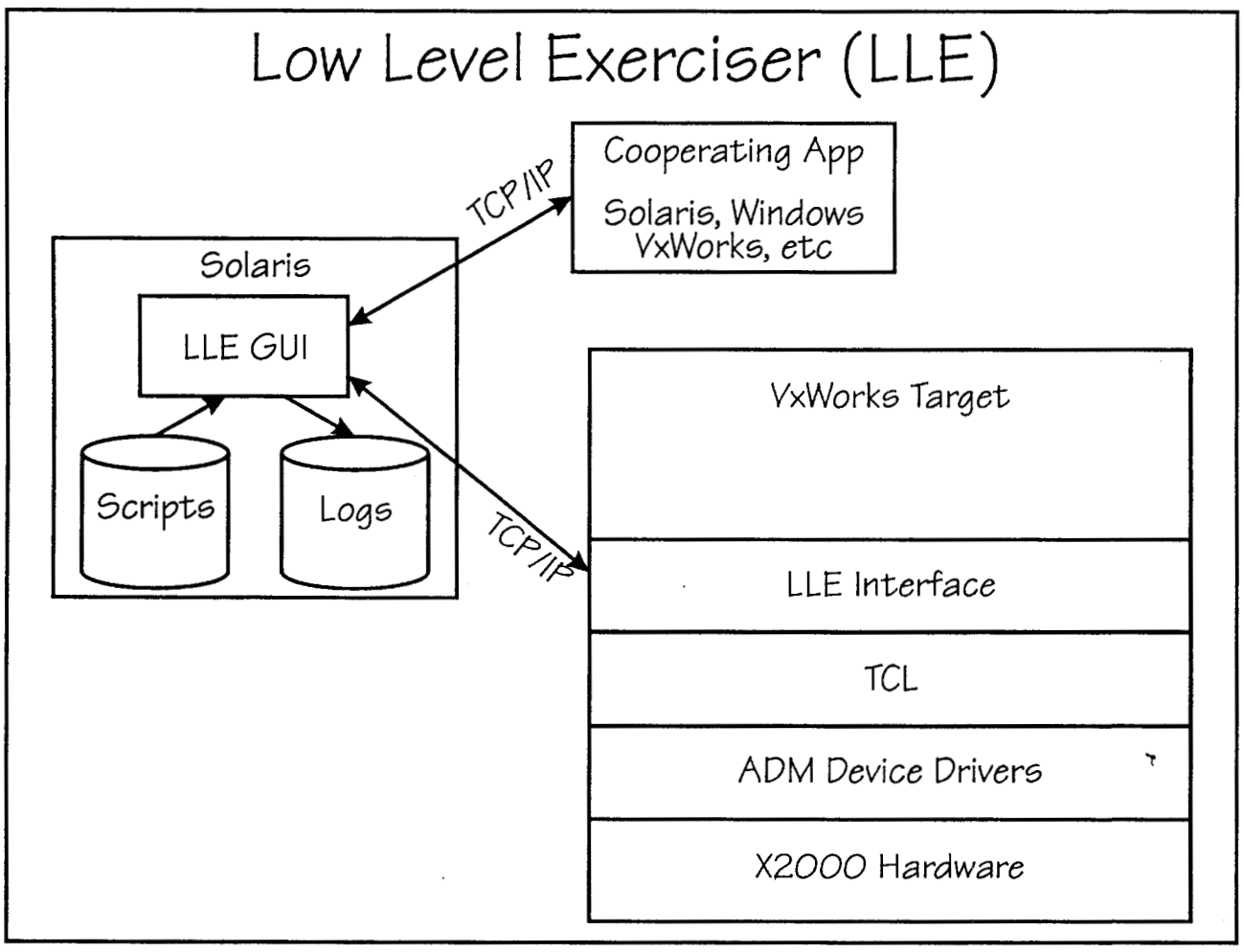
Device drivers

- DIO
  - 4 separate drivers, 1394, I2C, UART and "Custom logic" (timers, discretes)
- SIA
  - Incorporates instrument and telecom interfaces
- NVM
  - There is a filesystem layer planned in addition to the driver
- TRIO
- PSS

1394 Hardware Abstraction Layer (HAL)

- Provides access to all 1394 hardware features
- Implemented mostly in C
  - C++ parts can be replaced by C code
- Less abstract than the ADM interface
  - ADM interface is implemented on top of the HAL
  - ADM Interface a higher-level view of the isochronous and asynchronous operations and contains address space management.

Test tools


Low Level Exerciser (LLE)
Cooperating App
Solaris, Windows
VxWorks, etc
TCP/IP
Solaris
LLE GUI
Scripts
Logs
TCP/IP
VxWorks Target
LLE Interface
TCL
ADM Device Drivers
X2000 Hardware

Test Tools

- LLE GUI implemented in TCL/TK
  - Provides command input, script input, timestamped logging to file and the screen
  - Provides interfaces to two external applications for transfer of commands and output
  - Can interleave commands / output from both
  - Not actually dependent on ADM or other programs, could be used with any application that can send and receive via a socket

Test tools

- On the target we run a TCL interperter (V8.0)
- ADM drivers register TCL commands to exercise I/O operations, build and maintain command lists, verify data and I/O status, etc.
- TCL commands / scripts can be loaded including complex procedure definitions
- Scripts have assess to the full TCL language as well as the additional commands created by the drivers
- Not specifically dependent on the LLE GUI, can be operated by any TCP/IP-aware application (sometimes we use telnet for software test).

Test Tools

- Support by the LLE GUI for an additional application
- Allows another application to participate in the test
- In our application for the PSS the application is a logic analyzer interface program
- Can be any application which can take input and display output via a socket
- This implementation is currently in progress

CPU Performance measurement

- We have developed a VxWorks-specific package to monitor performance
- Uses the PPC 750 performance monitoring registers
- Collects statistics on a per-task basis
- Reports CPU time, # of instructions, L1 cache rate, memory use (malloc / free)
- Currently being enhanced to report 1394, I2C and SIA bus traffic
- Summarizes output
- Reports to screen or file
- Intended for eventual flight use

Development status

- All device drivers are nearing completion
  - All of them run on the X2000 hardware, being debugged in parallel with the hardware debug
- LLE is operational and being used extensively for hardware and software test
  - Currently adding additional interface

Software support for X2000 Hardware

Blair Lewis / Len Day

Blair.F.Lewis@jpl.nasa.gov /
Len.Day@jpl.nasa.gov

June 6, 2001

Agenda

- Charter
- Software overview
- Abstract Device Model
- Device drivers
- 1394 Hardware Abstraction Layer (HAL)
- Test tools
- CPU Performance measurement
- Development status

Charter

- Among other things the Europa Orbiter Operating System and Avionics Support team is chartered to:
  - Provide flight device drivers for all X2000 hardware
  - Use these drivers to support the X2000 hardware integration and test
  - Provide test tool support for testing / exercising the X2000 hardware

Software Overview

- To achieve these ends the following have been developed:
  - Abstract Device Model (ADM)
    - An object oriented framework containing the common code for implementing drivers such as I/O queuing, I/O completion notification, event reporting
  - Concrete instantiations of the ADM for each of the X2000 devices
  - A complimentary implementation called "Command Lists" which enables repetitive execution and allows commands for various devices to be executed with specified temporal relationships
  - A test tool called LLE (Low Level Exerciser) useful for both software and hardware test

Software overview cont

- Portability
  - Primary target is VxWorks, nevertheless we are maintaining portability
  - Minor portions of the code such as hooking interrupts are necessarily OS-dependent
  - All other OS interaction is done using a wrapper called ACE (Adaptive Computing Environment) from the University of Washington
    - ACE is ported to many environments
  - We currently do much of our software testing under Solaris using simple software simulators instead of hardware
  - Full bit-level simulator hooks in place, simulators to be developed in the future.
  - All drivers in C++

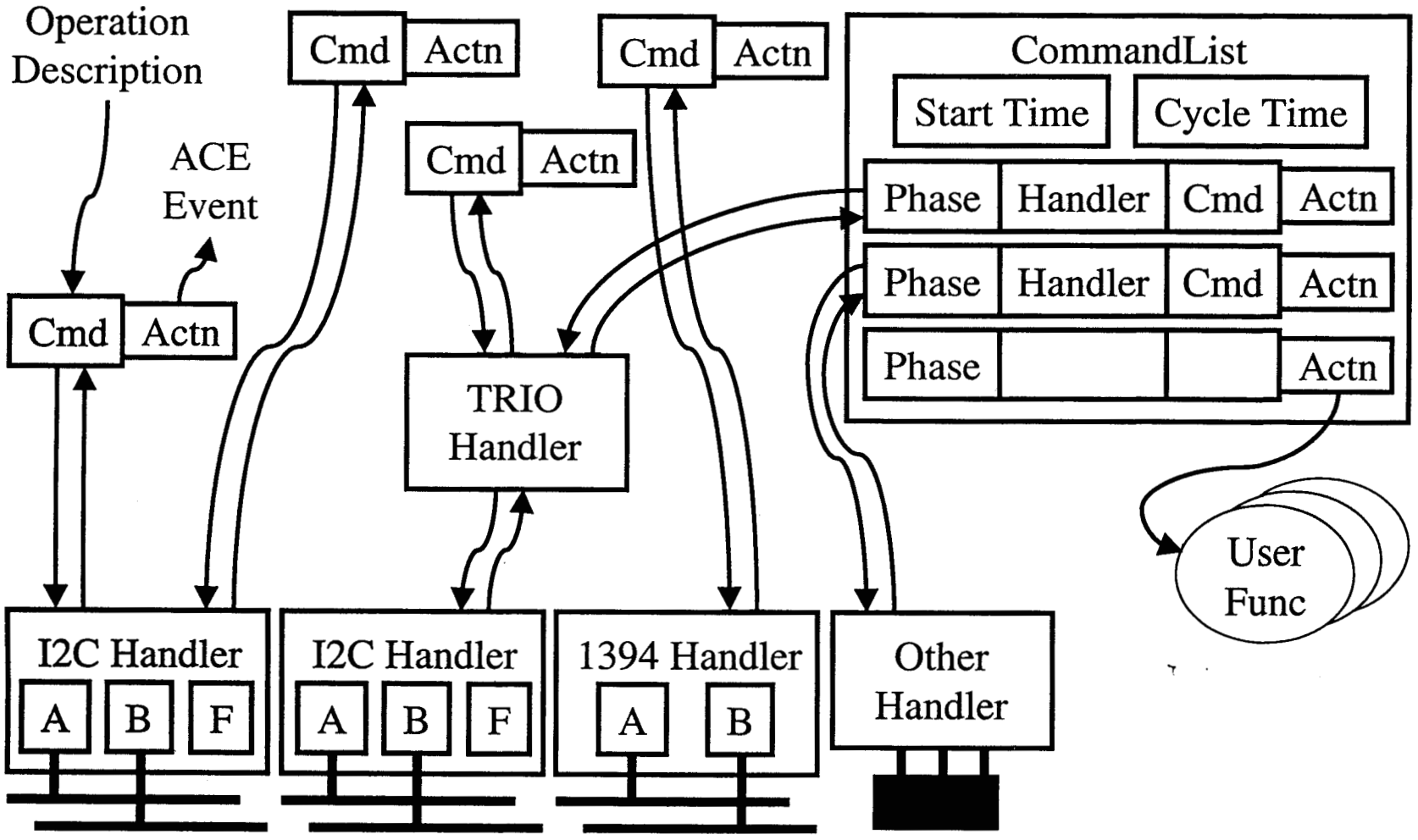
Abstract Device Model

- General goals:
  - Provide a uniform functional interface across a set of device drivers
  - Provide general-purpose methods to handle sets of related I/O operations
  - Provide a flexible framework for device driver implementation

Abstract Device Model

- Use objects to represent actions
  - Command objects represent I/O operations.
  - Action objects represent how a user is to be notified of an asynchronous operation's completion.
- Use a Command/Handler model
  - Users (clients) send Commands with attached Action objects to a Handler that executes the command and uses the Action objects to perform the notification of completion.
  - Actions can set semaphores, do callbacks, set bits, virtually anything

Abstract device model



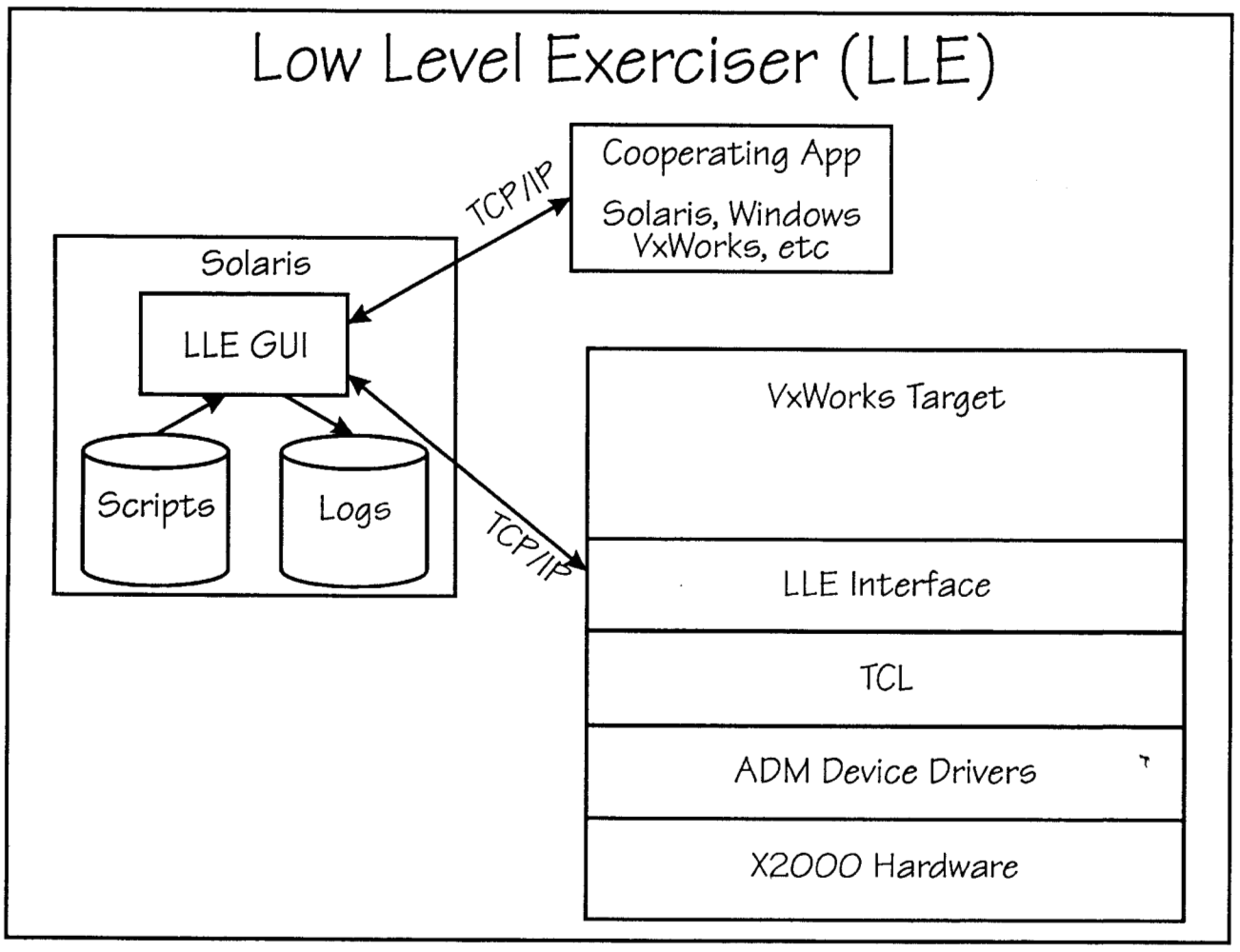
Device drivers

- DIO
  - 4 separate drivers, 1394, I2C, UART and "Custom logic" (timers, discretes)
- SIA
  - Incorporates instrument and telecom interfaces
- NVM
  - There is a filesystem layer planned in addition to the driver
- TRIO
- PSS

1394 Hardware Abstraction Layer (HAL)

- Provides access to all 1394 hardware features
- Implemented mostly in C
  - C++ parts can be replaced by C code
- Less abstract than the ADM interface
  - ADM interface is implemented on top of the HAL
  - ADM Interface a higher-level view of the isochronous and asynchronous operations and contains address space management.

Test tools


Low Level Exerciser (LLE)
Cooperating App
Solaris, Windows
VxWorks, etc
TCP/IP
Solaris
LLE GUI
Scripts
Logs
TCP/IP
VxWorks Target
LLE Interface
TCL
ADM Device Drivers
X2000 Hardware

Test Tools

- LLE GUI implemented in TCL/TK
  - Provides command input, script input, timestamped logging to file and the screen
  - Provides interfaces to two external applications for transfer of commands and output
  - Can interleave commands / output from both
  - Not actually dependent on ADM or other programs, could be used with any application that can send and receive via a socket

Test tools

- On the target we run a TCL interperter (V8.0)
- ADM drivers register TCL commands to exercise I/O operations, build and maintain command lists, verify data and I/O status, etc.
- TCL commands / scripts can be loaded including complex procedure definitions
- Scripts have assess to the full TCL language as well as the additional commands created by the drivers
- Not specifically dependent on the LLE GUI, can be operated by any TCP/IP-aware application (sometimes we use telnet for software test).

Test Tools

- Support by the LLE GUI for an additional application
- Allows another application to participate in the test
- In our application for the PSS the application is a logic analyzer interface program
- Can be any application which can take input and display output via a socket
- This implementation is currently in progress

CPU Performance measurement

- We have developed a VxWorks-specific package to monitor performance
- Uses the PPC 750 performance monitoring registers
- Collects statistics on a per-task basis
- Reports CPU time, # of instructions, L1 cache rate, memory use (malloc / free)
- Currently being enhanced to report 1394, I2C and SIA bus traffic
- Summarizes output
- Reports to screen or file
- Intended for eventual flight use

Development status

- All device drivers are nearing completion
  - All of them run on the X2000 hardware, being debugged in parallel with the hardware debug
- LLE is operational and being used extensively for hardware and software test
  - Currently adding additional interface