

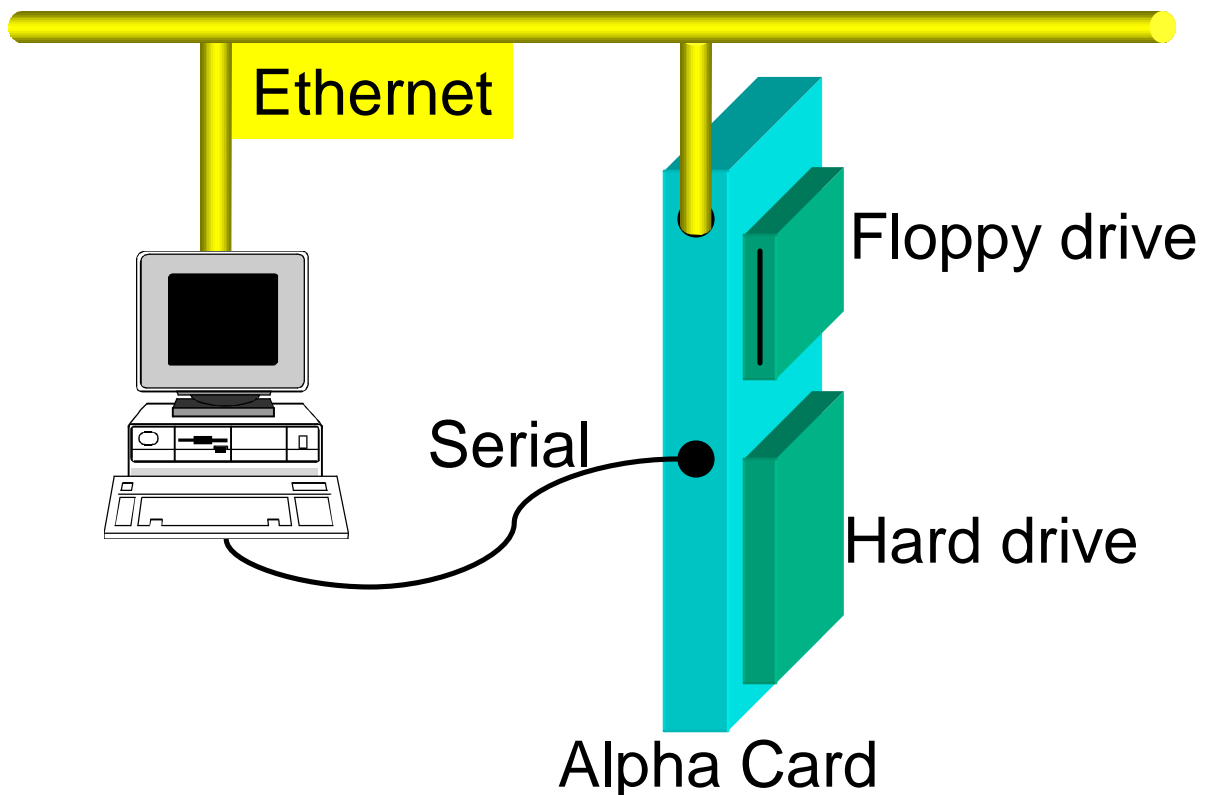
DØ Level 2 Trigger Software for Run II



**R. Moore,
Michigan State University**

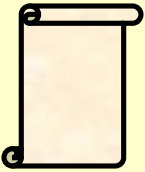
Software Environment

- **Main environment is DEC's debug monitor**
 - **compile using DEC C++**
 - **executable download via ethernet**
 - **serial console for interactive applications (commissioning etc.)**

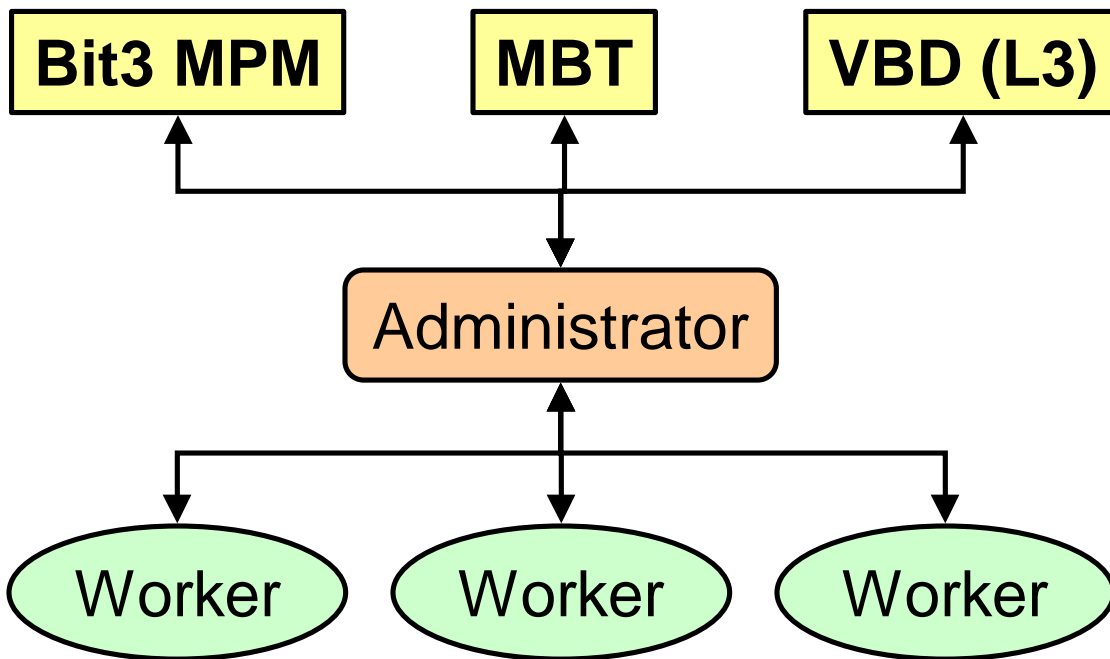


Software Environment II

- **Why choose this?**
 - No OS to interrupt code
 - SDK from DEC allows normal UNIX code to compile
 - allows remote debugging over ethernet
- **Limitations**
 - no dynamic memory management
 - lacks nice OS features (protected memory, telnet access etc.)
- **But most of these features add overhead so we should avoid them**



Basic Design



- **Two types of alpha**
 - administrator manages the workers
 - workers process data
- **Two worker networks**
 - each event to each worker
 - each event to one worker



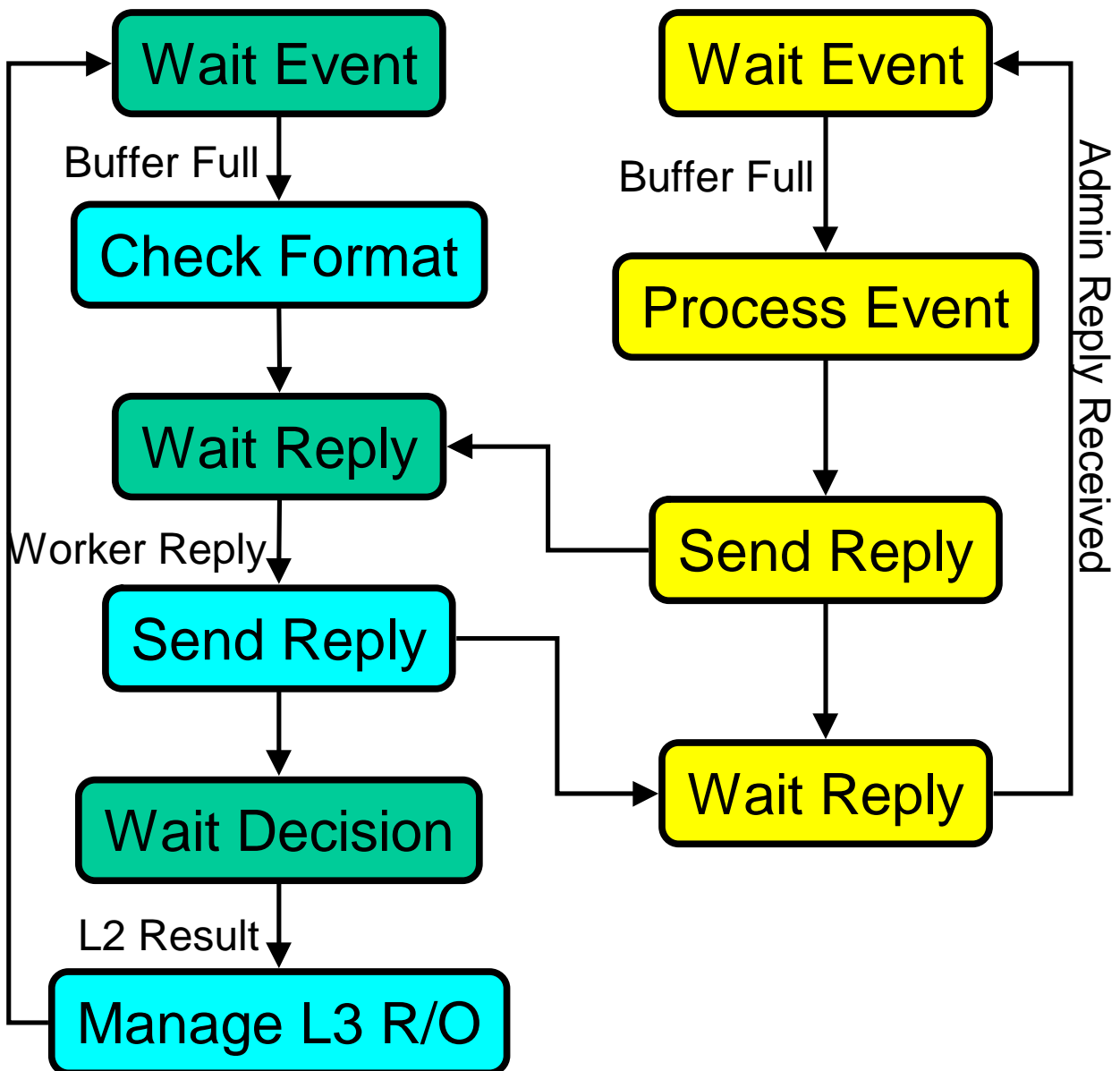
The Administrator

- **Initialization**
 - Looks for configuration data in **Bit3**
 - notifies workers where to find their setups and awaits confirmation of success
 - writes ‘setup complete’ flag in **Bit3**
 - verification of configuration data (before attempting to parse) done solely by executable version number (see L2 Global talk)

Event Loop

Administrator States

Worker States



Worker « Admin

- Admin ® Worker
 - tells worker which buffer to process next
 - gives one (or more) buffers to allocate
- Worker ® Admin
 - tells admin which buffer has been processed
 - Global workers tell admin the L2 trigger bits for the event

Multi-Worker Loops

- **Two possibilities to deal with multiple workers:**
 - **Lockstep**
 - **all workers complete processing before any process next event**
 - **event loop identical to single worker, wait for all replies**
 - **Non-Lockstep**
 - **each worker told to process next event when its ready**
 - **event loop has generic wait state**
- **Choice of lockstep/non-lockstep made in admin:**
workers identical for both

Error Handling

- **Three responses to errors detected in workers or admin**
- **Non-fatal**
 - pass event to L3 and flag error
 - e.g. checksum fails
- **Serious**
 - raise L1 busy and request SCL initialize (reset buffers+cards)
 - e.g. synchronization error
- **Fatal**
 - SYSRESET (reboot alphas, download executables etc)
 - e.g. administrator hangs



The Worker

- **Only has three states**
 - **processing data**
 - **waiting for an event**
 - **waiting for an admin reply**
- **Simple calling interface for user code**
 - **initialize**
 - **process buffer**
 - **collect status**
 - **reset status**
 - **reset**
- **Automatically generated IO code for object read/write**



Monitoring

- **Collect status events**
 - after processing event each worker writes status data to internal buffer
 - admin collects buffers (with its own) and writes them to Bit3, then signals TCC
- **Unbiased Sample Events**
 - each worker adds all its input data to the end of its output buffer
 - whole buffer written to L3
 - allows verification of decisions



Debugging



- **Emulation**
 - software test of data transport code
- **Simulation**
 - tests worker code only
 - large number of simultaneous users
 - little/no hardware expertise required
 - low numbers of events
- **Test stand**
 - close to online environment
 - limited availability
 - low numbers of events



Debugging II



- **Shadow Nodes ('spare' alphas in online system)**
 - **high event rates**
 - **synchronous**
 - processes every event
 - could act as hot spare if main worker fails
 - difficult to resynchronize in case of failure
 - **asynchronous**
 - processes most recent event when ready
 - more time to analyse failures
 - more decoupled from system



Debugging III



- **Core dump files (used by UNIX debuggers)**
 - possible to write code to create a fake core dump from debug monitor (but difficult...)
- **Use of Linux/Alpha**
 - gives full OS environment
 - useful for hardware diagnostic programs (remote access, UNIX environment...)
 - possible to use as online environment
 - provides core dumps for free!
 - performance issues may be a problem...

Software Status

- **All software designed to level of class diagrams**
- **Some basic utilities written and tested e.g. FIFO, interrupt handler, RTC interface...**
- **Administrator prototype tested with worker stub-code in simple emulator environment**
- **Currently implementing**
 - **I/O routines**
 - **hookup to simulation**
- **Starting soon (in parallel)**
 - **hardware device code**