Beta should meet or exceed specs (critical specs italicized), and provide this functionality
In cases where it can’t, need to explain how that’s dealt with

Occupies 2 slots; power: +12 2A; +5 20A; +3.3 10A; -5.2 1.25A; -12 1A; total, 176W

Processing power 500 MHz 21164
  2 integer, 2 floating pipes; 64 bits wide (used in bit searches and PIO)
  96kB on-chip L2 cache; 3-way associative
    enough cache for data for event
    + enough instructions for 24 μsec of non-looping execution
  (L3 cache (4MB) does not work, doesn’t SEEM to be needed
  because of characteristics of L2 cache above)
SI95 15  SF95  21

Executes L2 algorithms < 50μsec/event with new data each event

Linux/RT Unix OS (include licenses in costs)
  • Association with physical memory (for buffers and device drivers)
  • Possibility of direct memory mapping (possibly running in MMU-less mode) (?)
  • Low overhead operating system (zero/low CPU use during running)
    o E.g. preemptive multitasking, with trigger code at (nearly) highest or priority
  • Interrupt handler for DMA and SCL_INIT (source code?)
  • Core dump files – need to load and examine core files at a later date
  • Offline environment:
    o need code development workstation
    o must be able to run offline simulator on same platform for trigger verification
  • Efficient modern C++ compiler:
    o Support for ANSI C++; especially namespaces and templates
    o Support for KAI compiler would be a big bonus!
  • Data formats specified as little endian; 1 and 2-byte instructions
  • (JTAG interface into CPU would be nice)

PCI devices: VME, TSI, DMA, PIO

VME via Universe II 32 bit wide used
  16MB/s for L3 readout
  Geographic addressing for initialization
  Programmable interrupts
  16, 24, and 32 bit addressing modes for handling VME devices in crate

TSI:
  J2 interface: (TTL levels)
  DØ minimum: 4 inputs, 6 outputs
  IN: 2 inputs generate PCI interrupts
    L2_Answer_Ready; VBD_DONE
  OUT: VBD_START; 5 “worker interrupt out”
CDF uses more; see web sites

Front panel interface:

DØ minimum: 32 bit ECL Output register (used for monitoring)
CDF: 32 bit LVDS; bi-directional (split unknown)
**Mbus DMA**  
80-100MB/s (at about 40% PCI occupancy)
Address translation for 10 bits of destination base address
16 events x 64 sources
reserved Mbus addresses 00yy where yy is lower 10 bits; 00 is upper 22 bits = 0
64 KB input FIFO
DMA destination set up by CPU
Time to re-set destinations after an event ~ .2 µsec/source
Time for PCI startup of DMA: 5-8 X 30ns PCI cycles after PCI mastership
Mbus mastered by MBT
Broadcast across processors
Support for MBT control protocol
PCI Interrupt at end of full-event DMA (all sources)
for all boards in broadcast
AP_FIFO Empty logic
set DONE_OUT
read MOD_DONE register (20 bits or so)
set(Administrator) or read START_LOAD Toggle Administrator/Worker
DMA may be turned off for debugging
Any other Mbus signals used:

**Mbus PIO**  
16MB/s 1µsec latency even during DMA
latency driven by CIA buffers filling; pauses every .3-.6 µsec in DMA
Note: 64 bit PCI means 16B Mbus PIO in 2 PCI cycles, not 4
2 modes: 1 MBus cycle ever 128 PIO bits (2 PIO cycles), one every PIO cycle (64 bit)
current firmware uses only 32-bit PCI cycles; to be upgraded

**Mbus Arbitration** in PECL, estimated < 15ns

Master/Write to Mbus devices (32 bit address)
Worker Output to Global; Global Answer to L2HWFW via MBT
Worker Answer; Administrator Reply (sender end)
Master/Read from Mbus devices (32 bit address)
Admin get L2 Answer (from MBT)
Slave devices: processor-programmable 1MB window: Mbus address space to memory
Slave/Write by other Mbus device (e.g. another alpha writes to my mailbox)
Worker Answer; Administrator Reply (mailbox end)
Slave/Read by other Mbus device (e.g. it reads back of my mailbox for checking)