Atlas SCT and Pixel ROD FPGA Based Implementation

Atlas - LBNL/Wisconsin Group

K. Dao, D. Fasching, O. Hayes, R. Jared

K. Marks, M. Nagel

March 20, 1999

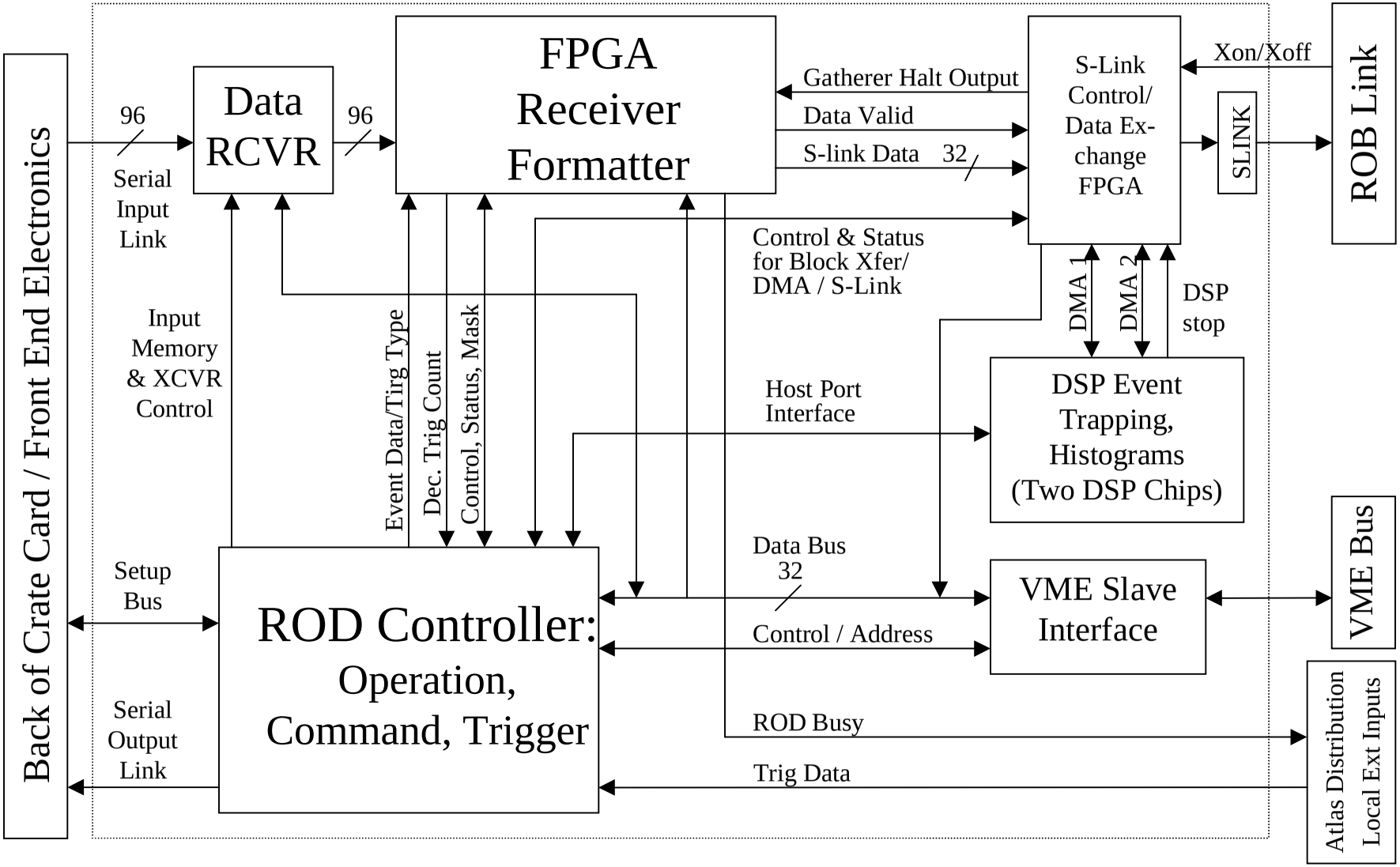
ROD Overview Notes

- 3.3V FPGA Decoder/Gatherer with DSPs for back-end processing and operation control.
  - Different firmware for SCT or Pixel RODs (same hardware).
  - Creates a “hybrid” ROD that exploits the advantages of FPGAs/DSPs.
  - In the main data path, FPGAs provide performance and predictability.
  - For Calibrations and Histograms, DSPs provide high level processing and access to a large memory.
- 9U VME card, custom backplane.
- Back-of-Crate Card for I/O.
- SCT: 96 input links
- Pixel: 36 links w/ TOT or 48 w/o TOT. B-layer TBD
- Deterministic, 40 Mwords/sec output performance.
- Supports 100 kHz trigger rate without an upgrade.

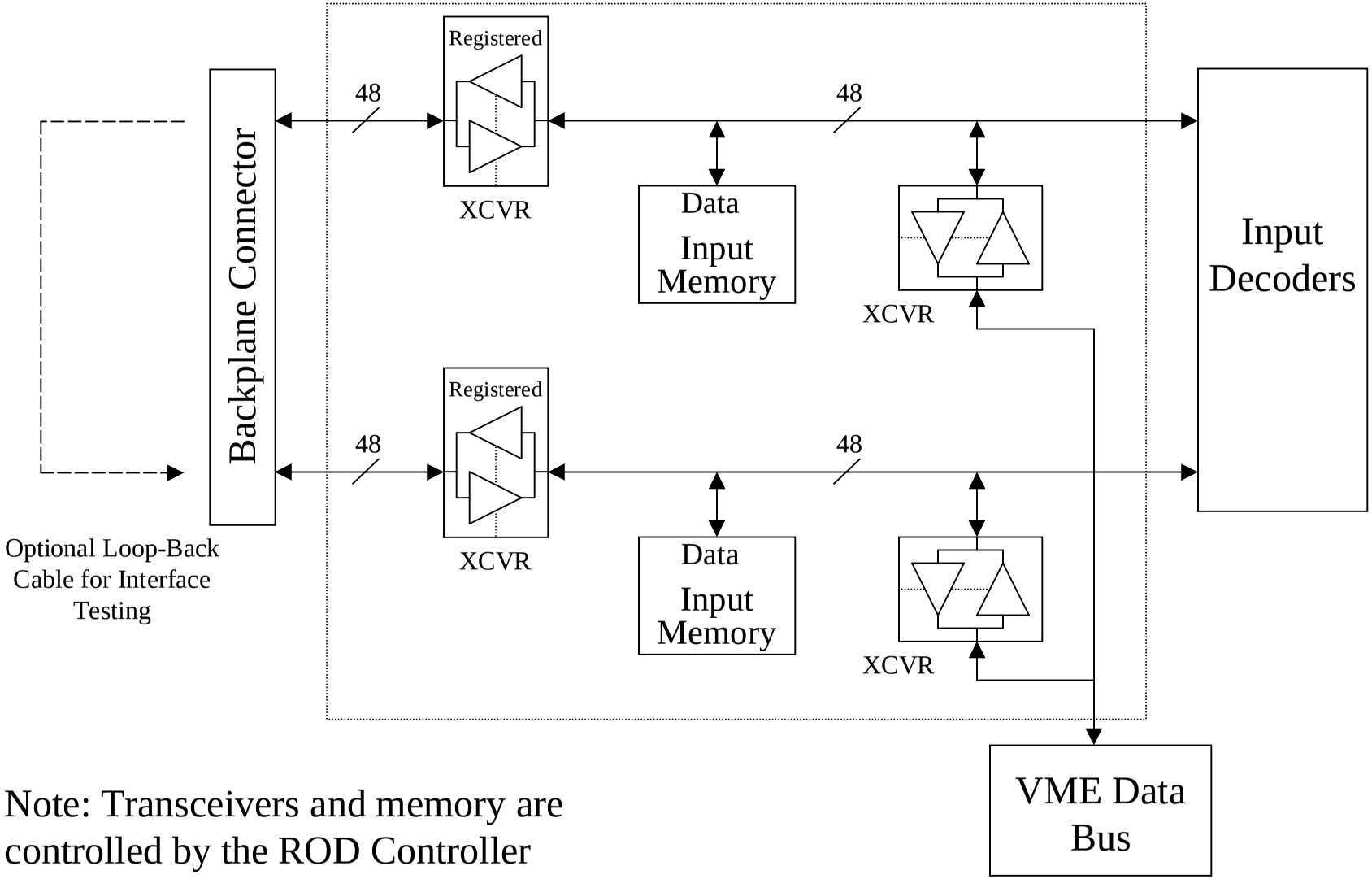
ROD Overview Notes 2

- Register based interface with DSP list processor.
- Fiber Optic S-Link to Readout Buffers.
- 12 FPGAs, 1 Million+ gates of programmable logic.
- 3 DSPs, 4800 MIPS of DSP processing power.
- Cypress Semiconductor VME Slave Interface chipset.
  - Block Transfers - 80 Mbytes/second.
  - Simple interface and supports standard VME transactions w/ interrupts.
- Clock and Trigger.
  - Clock and trigger signals are available on the backplane from the Trigger Interface Module in the crate.
  - LEMOs can provide clock and trigger inputs for standalone testing.
    - Spare LEMOs can provide additional FPGA configurable I/O signals.

ROD Implementation: FPGA Front End / DSP Back End



Data Receiver Implementation

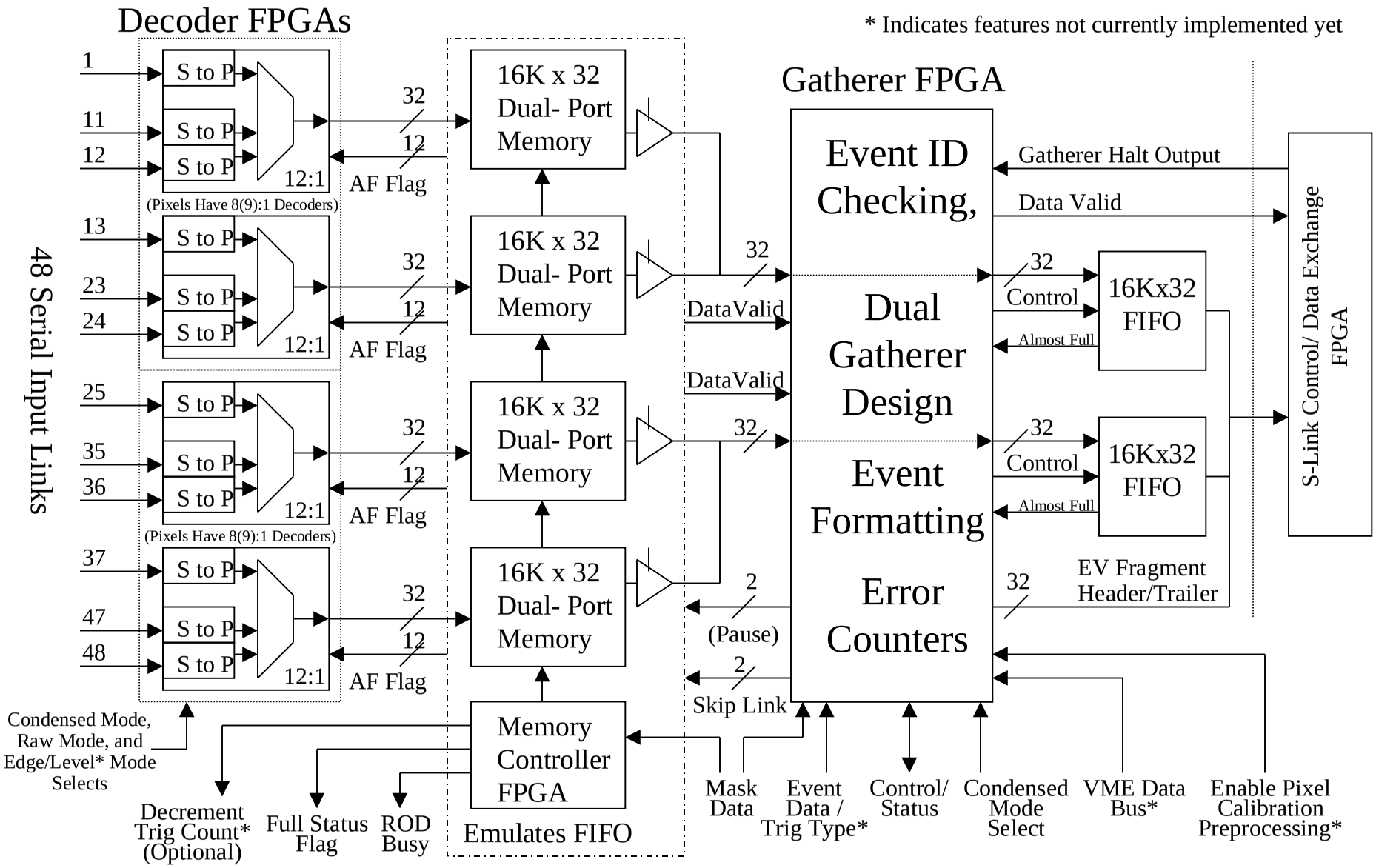


Note: Transceivers and memory are controlled by the ROD Controller

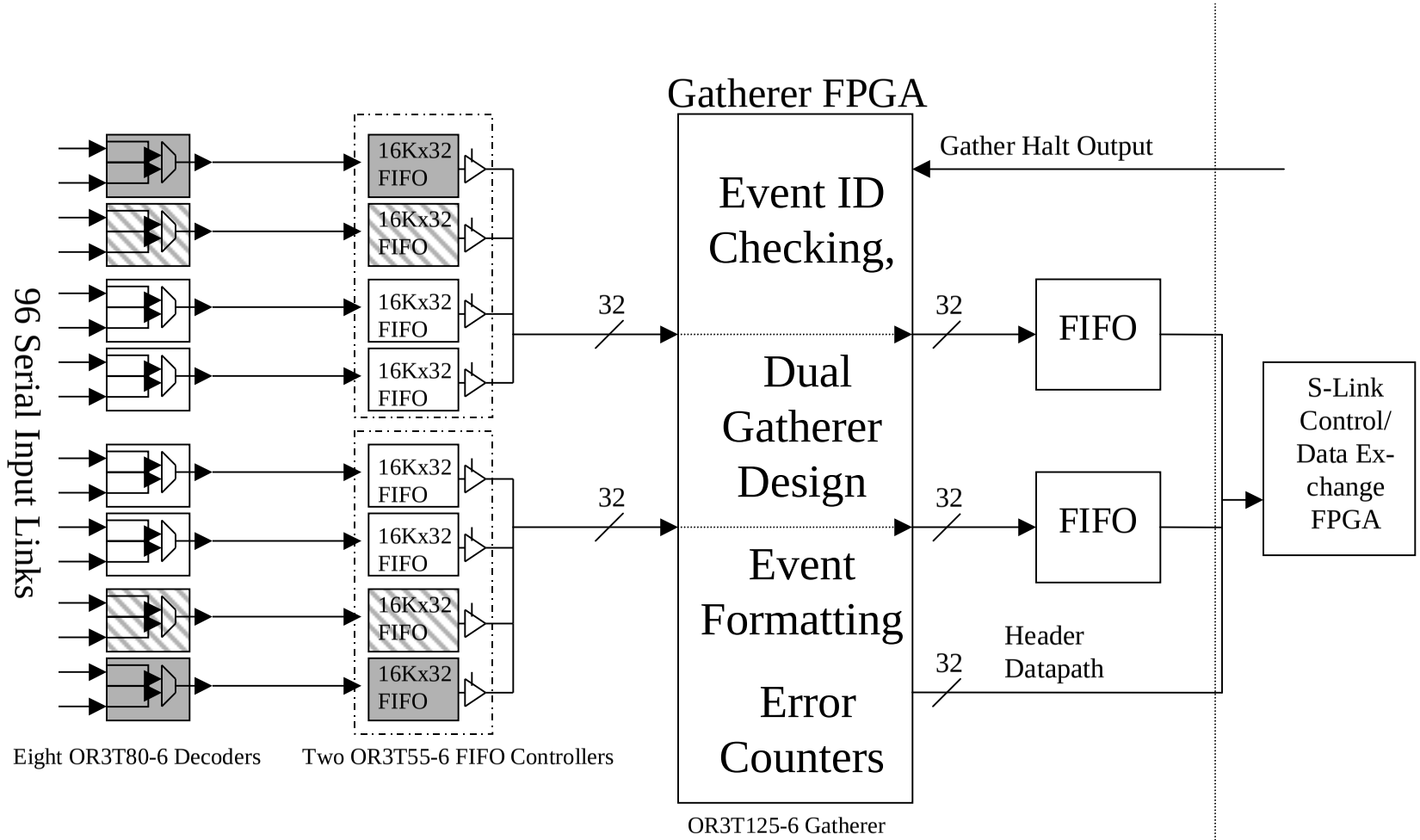
Data Receiver Notes

- Registers data from back-of-crate card to input decoders to resynchronize signals.
- Input Memories.
  - Play or record data to perform diagnostics and tests.
  - Start of record operation can be triggered by the ROD Controller.
  - Two groups of 64K x 48 bits, single-ported synchronous memories.
- Backplane interface connector can be tested with a loopback cable.
  - One memory plays while the other records.

Receiver Formatter - 48 SCT / 32 Pixel Links



Receiver Formatter - 96 SCT / 48(36) Pixel Links



The completely shaded components need not be loaded in the Pixel ROD. If there is Pixel TOT info (i.e. 36 link ROD) then the hash shaded components can also be removed. SCT Decoder FPGA has 12 links. Pixel Decoder FPGA has 8 Links w/o TOT and 9 links w/TOT.

Receiver Formatter - Decoder

- Decoder FPGAs have separate versions of logic for Strip and Pixel Modules.
  - SCT: 12 Links per FPGA.
  - Pixel: 8 links per FPGA without TOT or 9 links with TOT.
  - Pixel B Layer may require another FPGA version supporting fewer links.
- Serial-to-Parallel conversion with error checking.
- Data from 12(8/9) Links written sequentially to dual-port memory on a round robin basis.
- Decoder can keep up with any input data rate.
- Only headers / trailers written when FIFO is almost full.
- Individual links can be put in raw data mode or condensed hit data mode (SCT).

Receiver Formatter - FIFO Controller

- FIFOs are implemented as FPGA w/ Dual-Port Mem.
- Sorts data by sequentially transferring one event of data from each link to the Gatherer.
- When FIFO Dual Port Memory is almost full:
  - Force the Decoder into header/trailer only mode on the links that are almost full to reduce bandwidth.
  - If Memory continues to fill up, the *ROD Busy* signal is issued to stop triggers.
  - If the Memory still goes full, then an error flag is issued to interrupt the ROD controller - fatal error condition.
    - This should not be possible unless ROD Busy is masked off or the front end electronics have malfunctioned.
- Support flow control - Gatherer can assert pause signal.

Receiver Formatter - FIFO Controller 2

- If data from a link times out, the link can be skipped.
- Individual links can be masked off.
- Optional features ???
  - Mask off Individual links that are malfunctioning on the fly.
  - Counts triggers that have returned on all links so ROD can protect the front ends from too many triggers.
    - This may be too conservative and can result in triggers being suppressed unnecessarily because of the latency from the front end through the decoders.
  - Load-able constants for the almost full and rod_busy thresholds (they are now constants in the FPGA firmware).

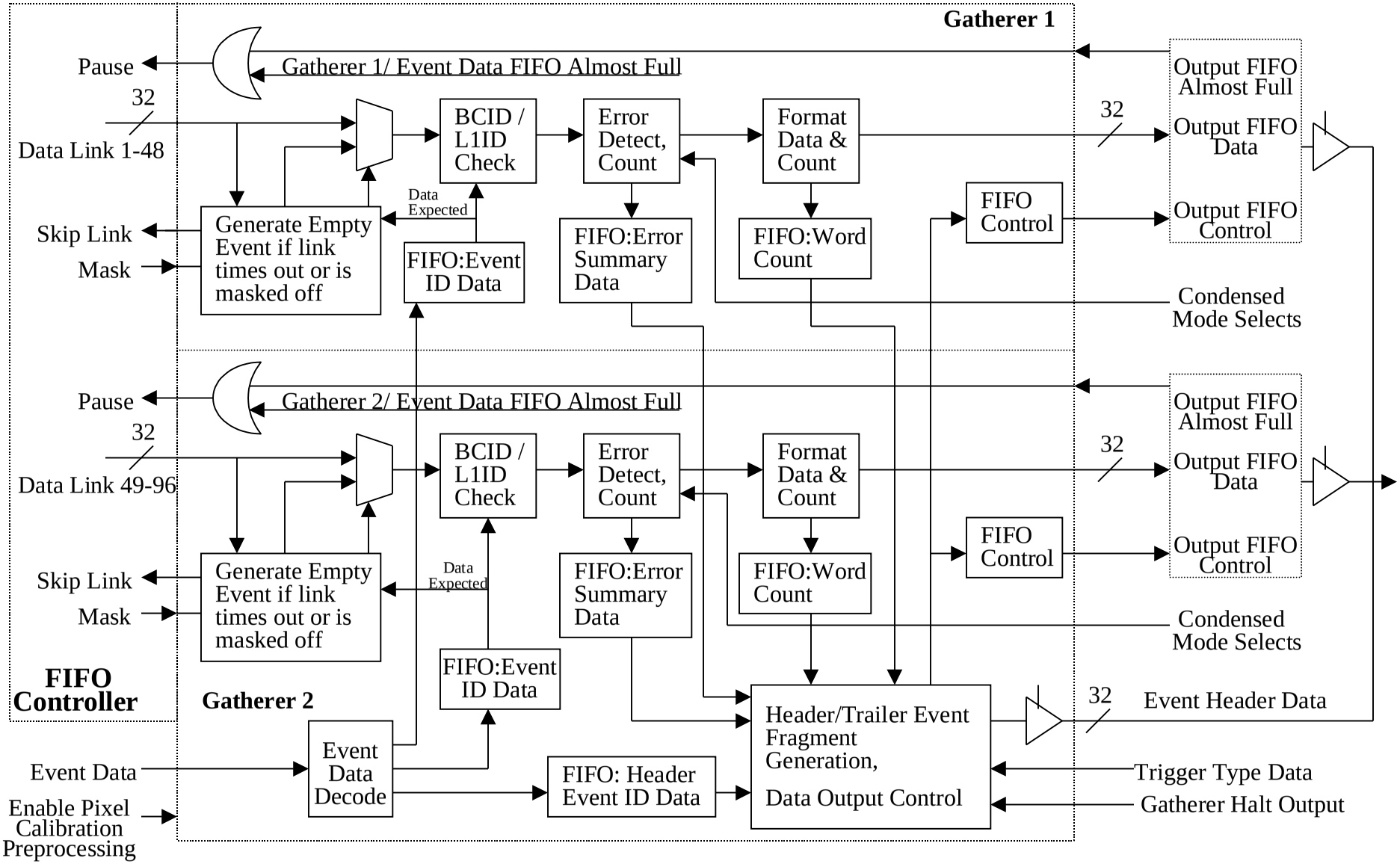
Receiver Formatter - Gatherer

- Performs event BC/L1 ID checking, error counting, and formats output data into ROB format.
- If “Gatherer Halt Output” is issued, data can still flow through the Gatherer and fill the output memory.
  - This will cause the gatherer to apply backpressure to pause the FIFO Controller and causes the Dual Port Memories to fill up.
- Gatherer can timeout waiting for data from a link.
  - Flags error and tells FIFO Controller to skip to the next link.
- Errors are maskable on a link-by-link basis.
- One global error counter that can interrupt the ROD Controller after a programmable number of errors.
  - Separate, maskable 8 bit error counters per link to be added.

Receiver Formatter - Gatherer 2

- Works on two parallel streams of data from half of the input links to mask the FIFO Controller's link-to-link latency.
  - The output FIFO memory is thus needed to derandomize data between gatherer data streams - helps achieve 40 MWords/sec bandwidth.
- Pixel Gatherer can perform pre-processing of data to reduce DSP processing time during calibrations.
- Optional Features
  - When the FIFO Controller goes full, generate empty event fragments with an error flag set indicating there was a FIFO overflow.
  - Allow the output memories to be loaded from VME so that data patterns can generated to test the DSPs or the Data Exchange FPGA.
  - Word counts can go into event fragment header.
  - In case of consecutive BC ID errors on one link (i.e. a missing or an extra event), a correction algorithm can be implemented.

Dual Gatherer FPGA Design



Gatherer - Functional Blocks

- Empty Event Generation:
  - If a link is masked off, a header/trailer word with no hits and no errors is generated as a place holder.
  - If a link times out, an empty event is also generated but with a flag bit set to identify the timeout error.
- BC ID / L1 ID Check:
  - Compares expected BC ID / L1 ID with the decoded data and flags any errors.
  - Other data just passes through.
  - Separate Event FIFOs provide L1/BCID data so each gatherer data path can work independently on different events.
  - Overwrites header data to add link numbers and flags bits.

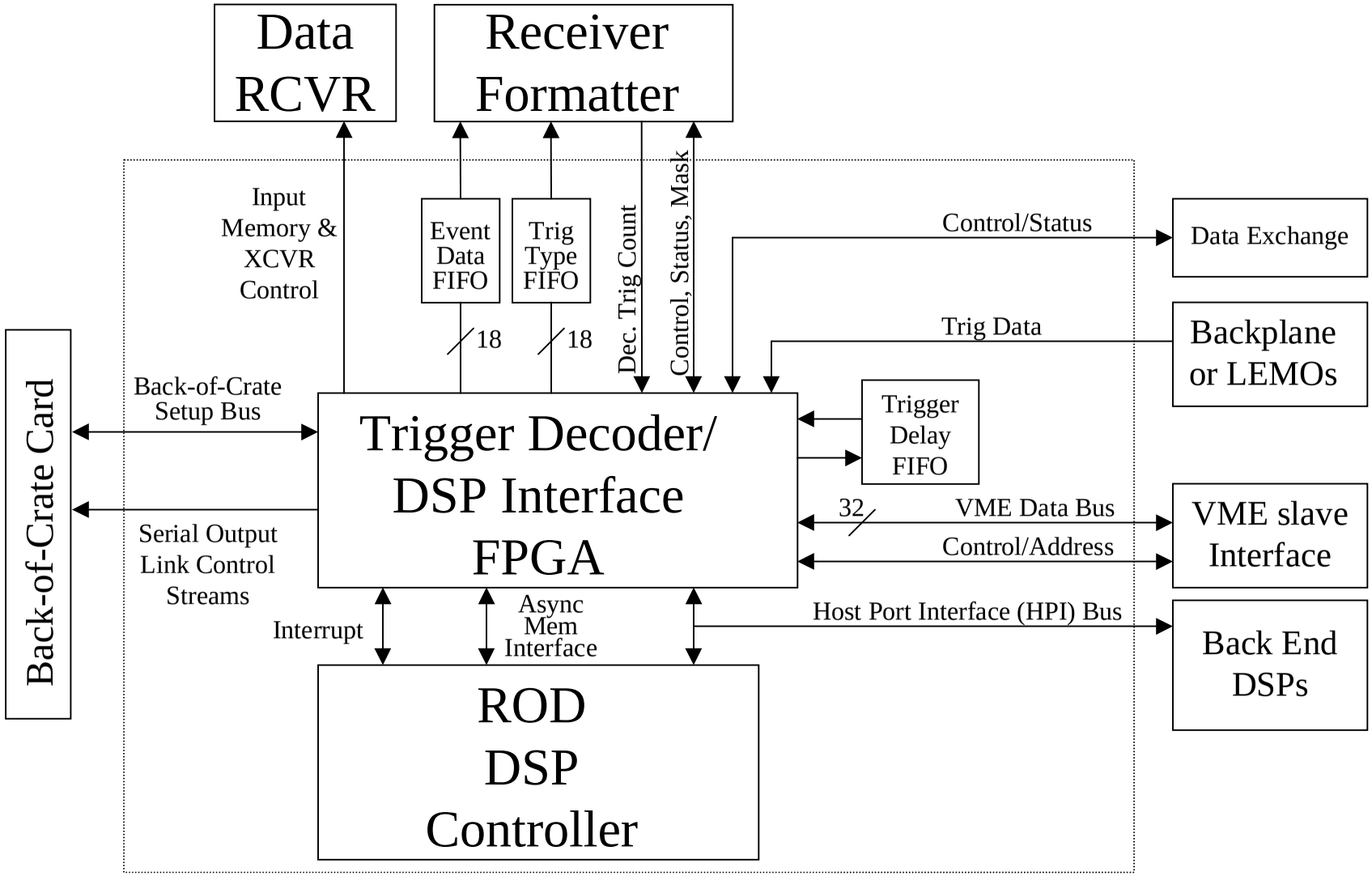
Gatherer - Functional Blocks 2

- Error Detect:
  - Looks at data bit fields to identify errors.
  - Generates the error summary block for the event.
  - One maskable error counter per link.
  - One global error counter w/ programmable interrupt threshold.
  - Error counters can be read by ROD Controller.
- Format Data and Count:
  - Removes front end trailer words unless the trailer error flag or the almost full flag was set.
  - Removes data from links with no hit data and no errors.
  - After unnecessary halfwords are removed, it re-packs all the data into 32 bit words so there are no gaps in the data.
  - Counts how many words were sent to the output FIFOs over an event and writes the count into a FIFO.

Gatherer - Functional Blocks 3

- Header/Trailer Block Generation, Data Output Control:
  - After the data counts are available from both Gatherers, an event is delivered to the S-LINK.
    - An Event Header block is first transmitted to the S-Link.
    - Using the data counts, data is then burst from the output FIFOs.
    - Finally, the summary and event fragment trailers are transmitted.
    - Will not output data until the trigger type data has arrived.
- Pause:
  - If The DSP asserts Stop or the S-Link asserts Xoff, data will not be transmitted, but the Gatherer can still process data and write to the output memories.
  - If any of the internal Gatherer FPGA FIFOs are almost full or the output FIFOs are almost full, the FIFO controller will be told to pause.
  - Output FIFOs are big enough to hold largest SCT/Pixel Event.

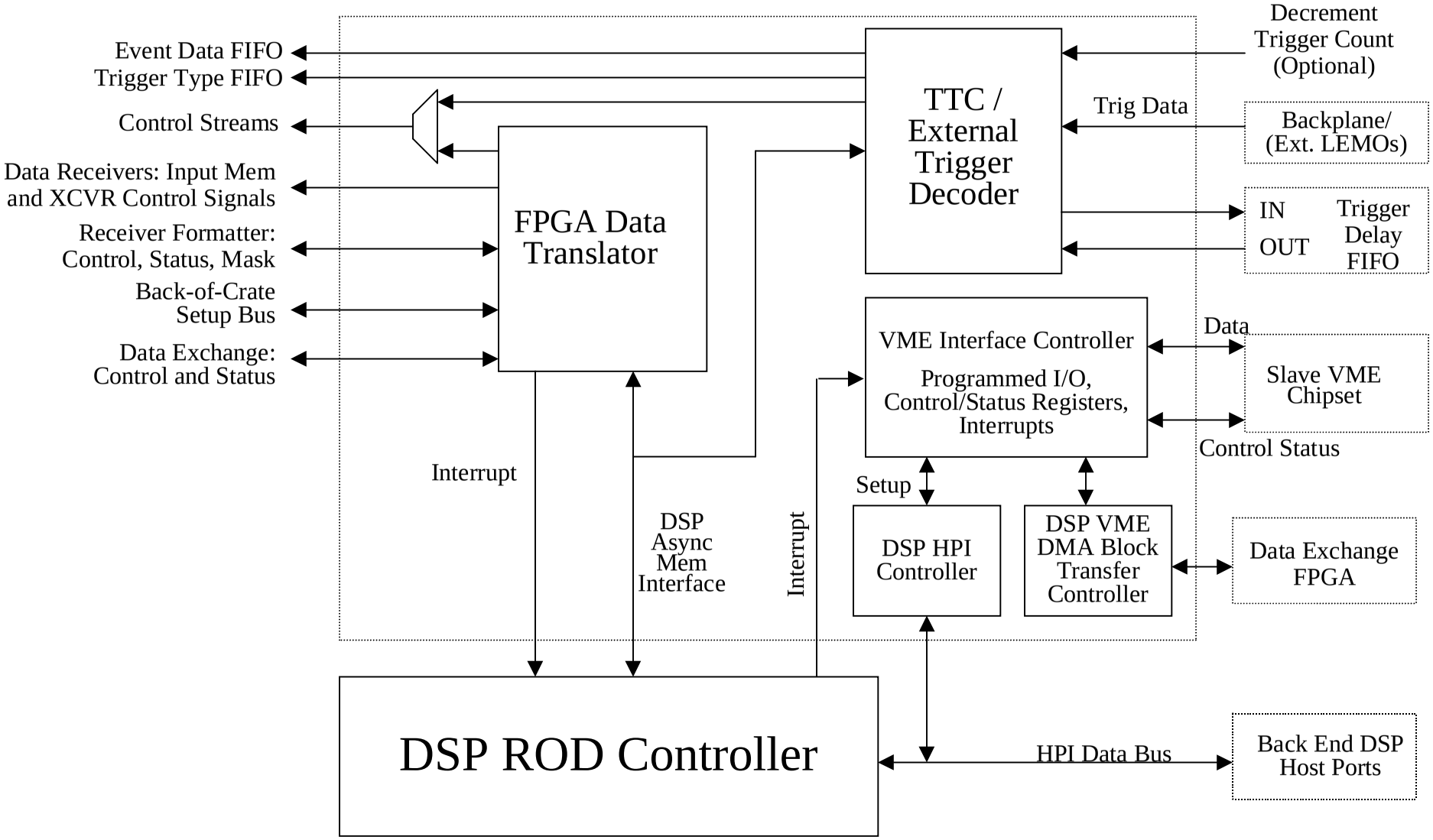
DSP ROD Controller Design



ROD Controller Notes

- Controls the operation of the ROD and coordinates the flow of data on the board.
  - i.e. read/write to memory, issue resets, set mask, check status.
- Works with the VME interface chipset to implement the memory-mapped I/O interface and generate VME interrupts.
- Processes commands from the TTC module or the external LEMOs to generate triggers and calibration signals.
  - Delay FIFO generates arbitrary trigger, strobe and accept patterns.
- Can execute a long command sequence without tying up the ROD Crate Controller.
- Event FIFO stores a list of expected trigger IDs that the Gatherer can check against.
  - Second FIFO queues the late arriving trigger type data.
- 4 control streams for all SCT links; will consider 48 (36) for Pixels.
- Keeps count of the number of pending triggers to prevent the front end buffers from overflowing (optional feature).

Trigger Decoder / DSP Interface FPGA



ROD Controller with DSP

- Support FPGA is required for a fixed latency trigger.
- This FPGA must also translate DSP commands into the necessary control/status signals for non-DSP chips.
  - The DSP asynchronous memory interface would be used to communicate with the support FPGA.
  - This FPGA would not be very complex.
- DSP allows high level control operations to be performed.
  - Performance is less critical in the control path.

ROD Controller with DSP 2

- The DSP offers flexibility since DAQ/Calibration requirements are not well defined.
  - Software can function at a higher level to relieve the processing load on the crate host and reduce VME bus traffic.
  - Can aide in post-processing data at beam/system tests (although the back-end DSPs can already do this).
- Option to have FPGA controller on the final ROD remains open.
  - Experience gained with prototype ROD may point this way.
  - Clearer DAQ and detector requirements may also point this way.

Back End DSPs

- Performs histogramming, spying, and event trapping.
  - Spying supports 1KHz event readout at the crate level.
  - Calibrations: raw data, histograms, summaries to reduce VME traffic and load on crate processor, or comparison to reference histograms.
  - 128Mbit SDRAM memory per DSP.
  - The DSPs are removed from the datapath allowing them to work independently without affecting the ROD data taking.
    - Clean separation from DSPs allow FPGAs to be fully simulated.
  - Can trap specific events, i.e. embedded calibrations.
    - Uses Trigger Type Info.
    - However, only one of every few events can be trapped.

Back End DSPs 2

- DSP can perform additional post processing of data at system tests and test beams.
- Data Exchange FPGA routes data to S-Link / DSPs.
  - Use internal FPGA dual port memory to asynchronously connect to DSPs.
  - During runtime monitoring, data is continuously DMA'd into the internal SRAM of each DSP at 20 Mwords/sec.
    - DSP captures the data it needs, throws away the rest.
    - Can tell gatherer to stop if it needs to temporarily stop data from flowing.
- DSP can play test pattern or calibration data to S-Link.
- During pixel calibrations, the gatherer will preprocess data to reduce the workload of the DSP.
  - SCT has far fewer elements so FPGA preprocessing may not be needed to reduce calibration time.
  - 2 DSPs with FPGA decoding give good calibration performance.

Design Status

- Features listed in the slides for the SCT Decoder, FIFO Controller, and Gatherer have been implemented unless otherwise noted.
- SCT Receiver/Formatter FPGAs simulated together and tested against a C behavioral model.
  - SCT Decoder written and tested with 1000 events.
  - SCT FIFO Controller written and tested with 1000 events.
  - SCT Gatherer is written and tested with 300 Events (~ 5 msec).
- Pixel Decoder version also available.
  - No configuration mode logic, not yet tested thoroughly.
- Interface FPGA in ROD Controller partially written.
  - can load configuration data into FPGAs using simple interface.

Design Status 2

- DSP code for hybrid has been discussed for feasibility but not written.
- Behavioral model simulates FIFO and output bandwidth behavior with extremely good accuracy.
  - Behavioral model thus provides a very good platform for studying the performance of the ROD.
  - High confidence that the ROD will have a negligible data loss rate.
- Receiver Formatter FPGAs all route successfully at 40 MHz.
  - High confidence that the ROD will work and that the FPGA based ROD implementation is feasible and can be designed in less time.
  - Decoder: 49% Logic, 86% PFU utilization, 63 of 172 I/O pins.
  - FIFO Controller: 37% Logic, 71% PFU utilization, 187 of 289 I/O pins.
  - Gatherer: 33% Logic, 64% PFU utilization, 261 of 343 I/O pins.
    - FPGA routing takes about 3 hours once the right cost table is known.

Flexibility

- FPGAs and DSPs are both programmable and therefore flexible devices.
- VHDL is a portable language that is independent of development tools and a particular manufacturer's FPGA.
- A processor allows for flexibility out of the real-time data path, i. e. the flexibility of a processor in a real-time environment is difficult to predict.

Concluding Notes

- This “hybrid” design exploits the strengths of FPGA and DSP technology.
  - The principle ROD requirement is negligible data loss; this must be known with certainty. FPGAs offer:
    - Performance and Determinism.
    - Simulation, Verification, Debugging, CAD Tool Package. Leads to faster development with fewer revisions.
    - State machines are less prone to hang.

    ➡ Therefore, FPGAs are preferred in the data path.
  - Silicon calibration and monitoring requirements are also demanding. DSPs offer:
    - Ability to process calibration data.
    - Large memory for fast calibrations and monitoring.
    - Ability to coordinate high level calibration runs.

    ➡ Therefore, DSPs are helpful with back-end and control functions.

Concluding Notes 2

- Hardware architecture is very open and straightforward.
- The mix of DSP and FPGA technology, and their clean separation in the architecture, also maps well to the manpower and expertise of the responsible institutions.
- A lot of progress has been made to show that this proposed ROD solution is feasible given the cost, schedule and labor constraints.