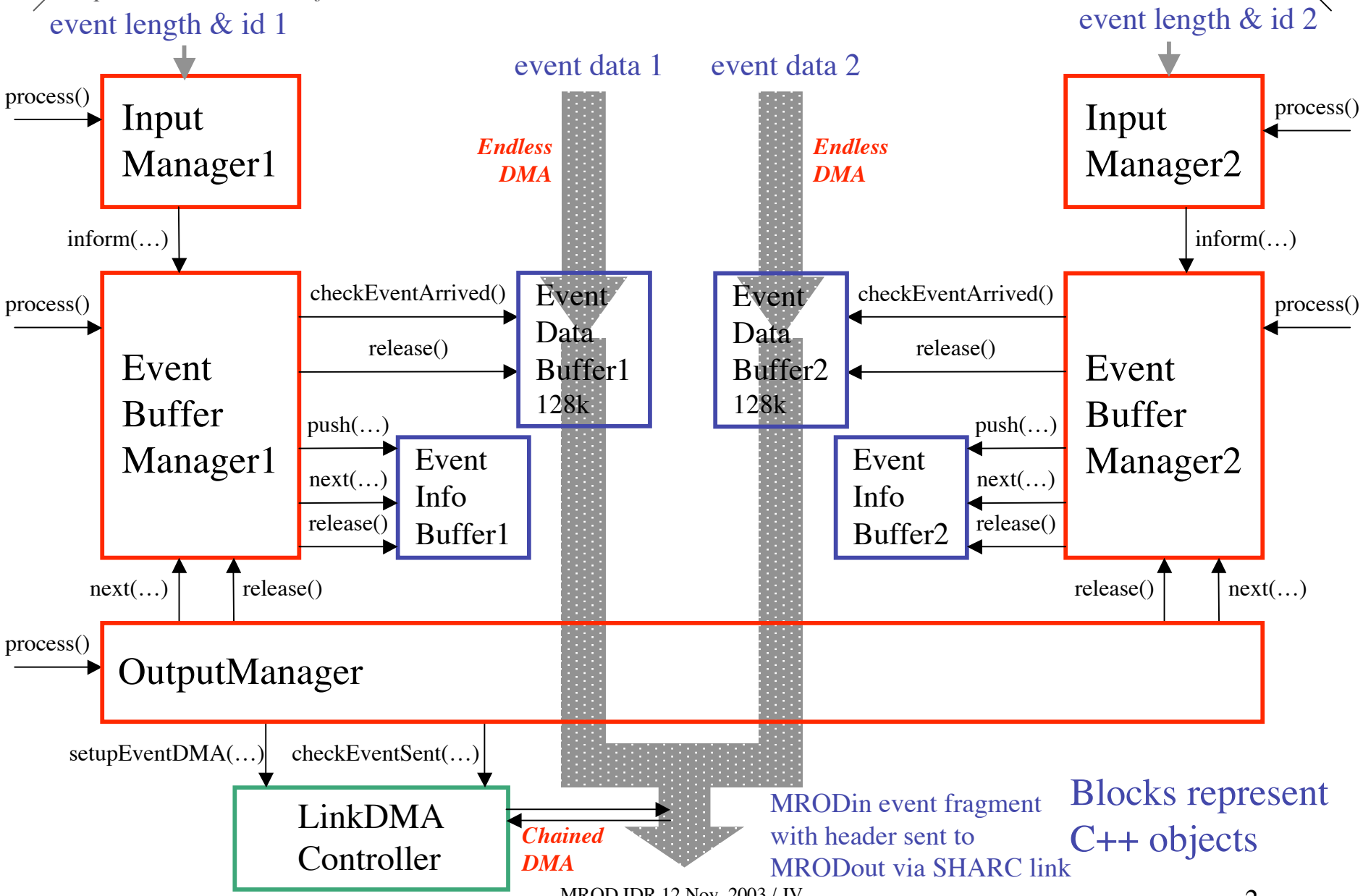
1. *MROD-1 software*

2. *Testing*

3. *TigerSHARC in stead of the 21160?*

MRODin software structure (data flow)

Separate data streams from 2 FPGAs via SHARC external bus



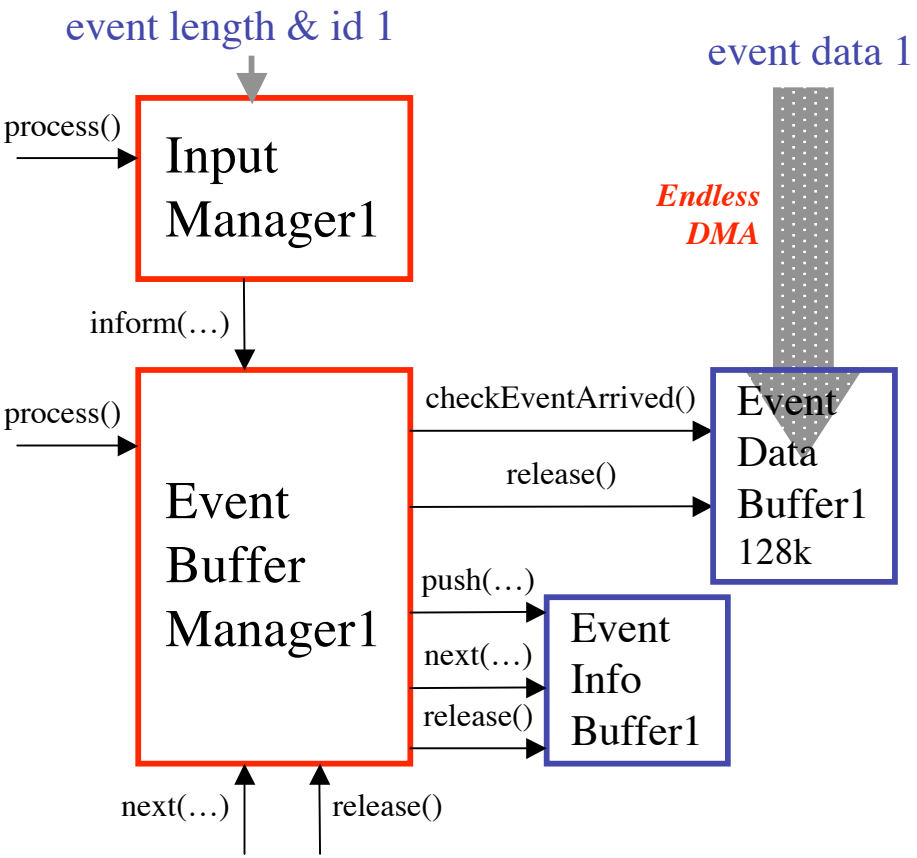
During normal running and for two active inputs this loop is executed:

```
while (running)
{
        theInputManager1.process();
        theEventBuffermanager1.process();
        theInputManager2.process();
        theEventBuffermanager2.process();
        theOutputManager.process();
        theControlManager.process();
}
```

Interrupts signal fault conditions only and do not occur during regular operation.

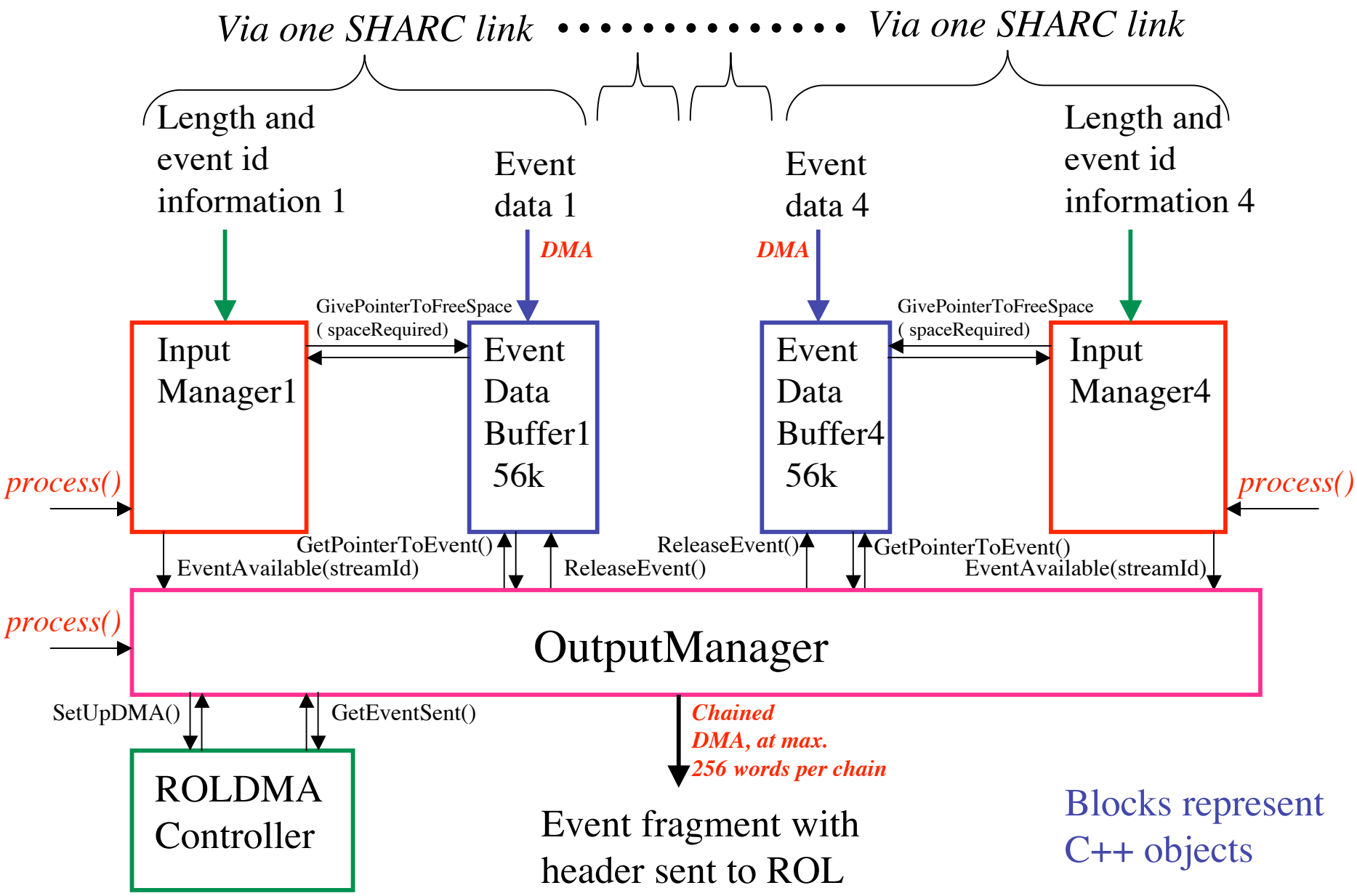
The Control Manager takes care of communication with a controller (via the MRODout) and of controlling state changes

NB: "process" methods are "inline" methods



Event data may be overwritten if the buffer is not emptied fast enough. This can happen in case of an XOFF on the ROL or for many large events arriving directly after each other. When the available buffer size in the EventDataBuffer drops below a certain limit an error flag is raised and events need to be skipped (the latter is not yet implemented). Halting the transfer of event data from the FPGA to the SHARC appears at first sight attractive, as there may be more buffer space available in the ZBT RAM. However, recovering from an overflow in the ZBT RAM is not possible by skipping events in a controlled way, as is possible for preventing overflow in the buffer in the SHARC. (NB: in the MROD-1 it is not possible to halt the transfer of event data from the FPGA to the SHARC temporarily).

MRODOut software structure (data flow)

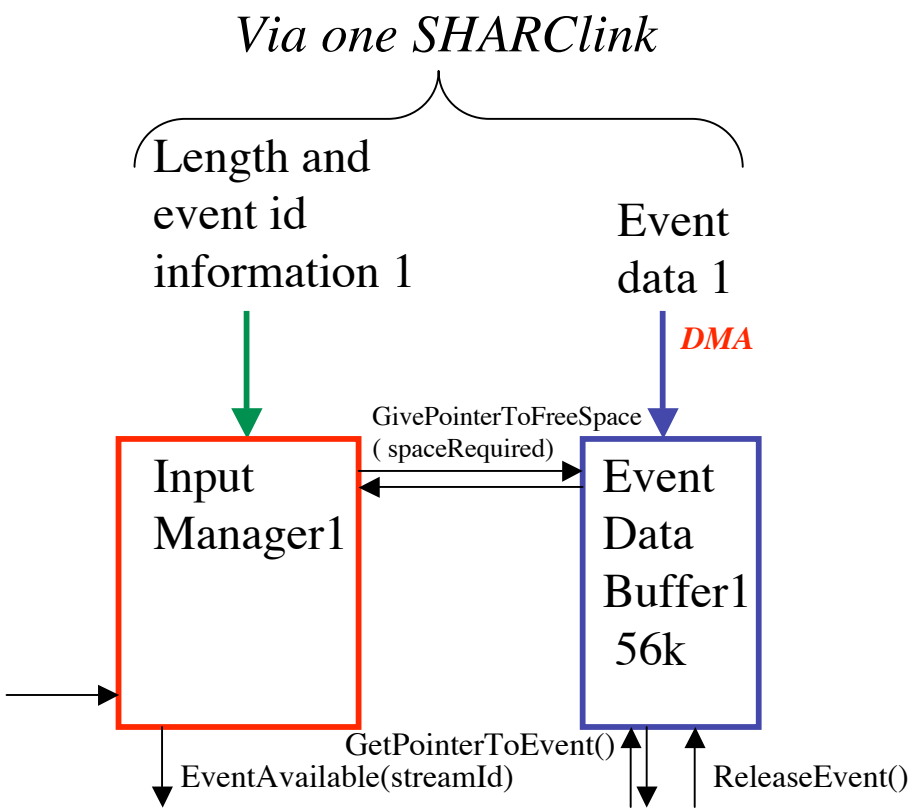


During normal running and for three active inputs a loop similar to the main loop of the MRODin is executed:

```
while (running)
{
        theInputManager1.process();
        theInputManager2.process();
        theInputManager3.process();
        theOutputManager.process();
        theControlManager.process();
}
```

Interrupts signal fault conditions only and do not occur during regular operation.

The Control Manager takes care of communication with the crate controller and of controlling state changes



A DMA for transfer of event data is only started when enough space in the buffer is present (14.4 kByte is needed for at max. 100 words per TDC and 18 TDCs per CSM). Transfers across a SHARC link are subject to handshaking -> event data cannot be overwritten.

Optimization is possible here with the help of fixed size DMA transfers or endless DMA and retrieval of the length and event id information from the buffer.

MRODout: at max. 256 words can be transferred as one block to the S-link, as the FIFO in the S-link interface has a size of 256 words. A write attempt to a full FIFO will result in a stall of the external bus until there is place again in the FIFO. There is a flag signalling that the FIFO is empty.
-> On the basis of this flag blocks of at max. 256 words can be safely transferred.

Under normal conditions the data leave the FIFO with the same speed (160 MByte/s) as they enter the FIFO. However, an XOFF on the ROL will cause the FIFO to fill (this caused problems last summer in the H8 testbeam, as a resulting bus hang prevents communication between the crate controller and the MRODout and may result in a VME-bus error).

Operational aspects

The SHARCs are booted via the VME-bus, the MRODout SHARCs directly via their internal memory, the MRODin SHARCs via their links. All can, after booting, communicate with a server program running on the crate controller, which provides support for terminal and file I/O. These facilities cannot be used for high rate data-taking, but are very useful for debugging and for error reporting.

Program development

The Visual DSP++ environment running under Windows is needed for cross-compiling and linking and producing loadable files. Program testing in an environment simulating SHARC and FPGA specific features has proven to be very useful for initial program development and debugging and for bug hunting. This is done under MACOS X or Windows with the Metrowerks Codewarrior IDE exploiting its debugging and code navigation features.

Status software

Working on transition to C++ on the basis of the software used in the H8 testbeam. This results in improved maintainability of the software, while optimization of the performance is no longer hampered by the lack of information hiding in the software written in C.

Performance

"H8 software", single channel input, output to S-link:
about 70 kHz max. event rate
C++ software, 6 channels, output to S-link, extrapolated: 72 kHz

One MRODin, one input channel, MRODout acting as data-sink, two words per TDC: 130 kHz

One MRODin, two input channels, MRODout acting as data-sink, empty events (only headers and trailers): 75 kHz (very recent result indicates 95 kHz to be possible)

One MRODin, one input channel, MRODout with S-link output, empty events (only headers and trailers): > 104 kHz (event rate limted by generator)

Two MRODins, one input channel, MRODout with S-link output, empty events (only headers and trailers): 87 kHz

Three MRODins, one input channel, MRODout with S-link output, empty events (only headers and trailers): 72 kHz *(1 output SHARC)*

MRODout program clearly can be optimized (e.g. chained DMA for output not yet implemented), further optimization of MRODin program also seems to be possible

2. Testing

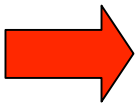
1. With programmable data generators in laboratory, HW/SW functionality, including error handling / speed
2. In simulated environment: for debugging software.
3. With cosmic-ray setup: "real-life" testing with full DAQ chain, setup to be used for MDT testing, planned start: 1 February 2004
4. H8, 2003 and in combined testbeam 2004

Several programmable data generators are available:

- *MRODout with appropriate software*, with fixed data pattern
  an event rate up to 130 kHz is possible with correct parity bits,
- *"CSMUX", FPGA based*, fragment size per TDC Poisson distributed,
  1 of each 256 fragments artificial large to simulate noise,
  very high frequency possible, driven by pulse generator,
  >> 100 kHz event rates possible. Three generators are
  available, more can be constructed. Errors can be introduced
  in the data (not in the present version),
- *SHaSlink (PCI card with prevous generation Sharc) based*,
  same software as for MRODout is used, but lower rates (~ 75 kHz).
  Rate can be increased if parity bits can be ignored. Five
  SHaSlink cards are available and can be synchronized to each other
  and to an MRODout functioning as data generator.

Additionally:

- the cosmic ray setup (5 chambers) can be used as high-rate data generator by lowering thresholds and triggering at random,
- for testing the MRODout the MRODins can be used in emulation mode.



Many possibilities for testing under a variety of conditions, which are being exploited.

NB: the data generated by the software generators tends to be very bursty, which can be helpful to find problems which may not show up when testing with identical time intervals between successive events.

Overview of results obtained so far with "CSMUX" for max. rate (in kHz) for Poisson distributed event size (average 4 words per TDC), truncated at the values indicated). For larger truncation values the bandwidth of 40 MByte/s of the link between MRODin and MRODout limits the event rate.

MRODs	CSMs	max TDC data words 0	1	2	3	7
1	1	> 103.5	99.0	93.6	84.7	66.0
	2	74.5				
2	1	87.5	83.8	83.4	75.3	59.5
	2					
3	1	72.5	72.5	72.2	68.1	52.9
	2					

3. TigerSHARCs in stead of the 21160?

The processing resources of the SHARC DSPs used may be sufficient for sustaining event rates up to 100 kHz, but not much headroom is left (if any) for other tasks.

Analog Devices has introduced the TigerSHARC as successor to the current SHARC series. A second generation TigerSHARC, the ADSP-TS20x series, is becoming available. These processors have clock frequencies of 500 / 600 MHz instead of the 80 MHz used for the SHARCs in the MROD-1. In addition a number of improvements have been made in the architecture and functionality of the device compared to the ADSP-21160 used in the MROD-1. Also the TigerSHARC has communication links (4 or 2) like the 21160, an MROD-2 design based on the TigerSharc could therefore be similar to the MROD-1 design.

	21160	TigerSHARC TS201, TS202, TS203
Internal memory	512 kByte	3, 1.5 or .5 MByte
Links	6 half duplex	4 or 2 full duplex
Max. link speed (1 direction)	100 MByte/s (practice: 40 MByte/s)	500 MByte/s
Link technology	4 bits parallel with handshaking, single-ended	4 LVDS pairs per direction, with hand-shaking and optional error detection (8-bit checksum per 16 Bytes)
Internal memory blocks	2, dual-ported SRAM, 64-bit wide (1 port connected via crossbar to 2 64-bit data busses, other port to peripheral bus)	6, DRAM, 128-bit wide (connected via crossbar to 4 128-bit data busses)
Link booting	only via link 4	via any link
DMA from/to link to/from external bus?	No	Yes

Possible caveat: internal pipelines in TigerSHARC longer (10 cycle instruction pipeline) than in 21160, could result in smaller gain in performance than suggested by difference in clock frequencies.

Timing for and behavior of TigerSHARC external bus differs from that of 21160 bus, links are different (signalling at 500 MHz is non-trivial). Getting all details right in an TigerSHARC based MROD-2 design may prove to be time-consuming.

Furthermore the experience with the 21160 has shown that first versions of processors and development software may be buggy and also that Analog Devices can be slow with delivery of relatively bug-free processors. The TS-20x series is just coming on the market ("sampling"), and NIKHEF is only a small customer for Analog Devices ....

We ordered a development kit with two TS-201s for evaluation of the possible gain in performance, delivery is expected in December.