ATLAS

ATLAS MDT ROD Production Readiness Review: Design Overview

Document Version: 1.0
Document Date: October 26th, 2005
Document Status: 5th Release

1. Abstract

An overview of the design of the production prototype of the ATLAS MDT ROD is presented.

Keywords: Atlas, MDT, ROD

2. Institutes and Authors

NIKHEF Amsterdam: H. Boterenbrood; P. Jansweijer; G. Kieft; J. Vermeulen
IMAPP, Dept. of Experimental High Energy Physics, Radboud University Nijmegen: A. König; T. Wijnen

3. Revision Log

Table 1: Document Change Record

1. Document title: ATLAS MDT ROD Readiness Review Design Document			
2. Document Reference Number			
4. Issue	**5. Revision**	**6. Date**	**7. Reason for Change**
0	1	13/10/2005	Birth.
0	2	25/10/2005	Expanded.
0	3	25/10/2005	Further expanded.
0	9	26/10/2005	Almost final version.
1	0	26/10/2005	Final version for PRR

Contents

4. Introduction

4.1. Purpose of this document

The purpose of this document is to provide information on the design of the ATLAS MDT ROD [1] for the Production Readiness Review. The final design of the MDT ROD, the MROD-X is a further development of the MROD-1, which was subject of an Intermediate Design Review on 12 November 2003 [2].

4.2. Glossary, acronyms and abbreviations

CSM: Chamber Service Module [9], mounted on MDT
GOL: Gigabit Optical Link [10]
MDT: Monitored Drift Tube chamber [6]
MROD: Read Out Driver for MDT chambers of the ATLAS Muon Spectrometer
MROD-0: test implementation of the MROD using boards originally developed for ATLAS TDAQ studies
MROD-1: first prototype MROD, event fragment building in software
MROD-X: pre-production version of the MROD, can be operated in a MROD-1 compatible mode, but also in a new mode with event fragment building by an FPGA.
ROL: Read-Out Link [11]

4.3. References

[1] H. Boterenbrood et al., “The Read-Out Driver for the ATLAS MDT Muon Precision Chambers”, ATL-COM-MUON-2005-013, http://cdsweb.cern.ch/search.py?sysno=002528191CER
[2] IDR of the MDT ROD: http://agenda.cern.ch/fullAgenda.php?ida=a035913
[3] P. Jansweijer and T. Wijnen, "MROD-X Design, Changes with respect to the MROD-1 Design", 26 October 2005, http://www.nikhef.nl/ pub/experiments/atlas/daq/mrod/mrodprr/MROD_X_Design.pdf
[4] P. Jansweijer and T. Wijnen, "MROD-X-In Programmers Manual", 26 October 2005, http://www.nikhef.nl/ pub/experiments/atlas/daq/mrod/mrodprr/MROD_X_In.pdf
[5] P. Jansweijer and T. Wijnen, " MROD-X-Out Programmers Manual ", 26 October 2005, http://www.nikhef.nl/ pub/experiments/atlas/daq/mrod/mrodprr/MROD_X_Out.pdf
[6] The ATLAS Muon spectrometer design report. CERN-LHCC 97-022, The ATLAS Collaboration (1997), http://atlasinfo.cern.ch/Atlas/GROUPS/MUON/TDR/Web/TDR.html
[7] C. Posch et al., MDT-ASD, “CMOS front-end for ATLAS MDT”, ATL-MUON-2002-003, http://bmc.bu.edu/bmc/asd/asd_chip.html
[8] Y. Arai, “AMT-3 (ATLAS Muon TDC version 3) User's Manual”. Rev. 0.32. 18-05-2005, http://atlas.kek.jp/tdc/amt3/index.html
[9] J. Chapman et al., “CSM: On-chamber readout system for the ATLAS MDT Muon Spectrometer”, IEEE Trans. Nucl. Sci., vol. 51, 2004, pp. 2196-2200
[10] G. Cervelli et al., “GOL: A 0.13-mu-m CMOS Serializer for Data and Trigger Optical Links In Particle Physics Experiments.”, IEEE Trans. Nucl. Sci., vol. 51, 2004, pp. 836-841
[11] CERN S-LINK, http://hsi.web.cern.ch/HSI/s-link/
[12] ATLAS Readout Link recommendation, see http://atlas.web.cern.ch/Atlas/GROUPS/DAQTRIG/DIG/ROD
[13] HOLA S-Link documentation: http://hsi.web.cern.ch/HSI/s-link/devices/hola
[14] ATLAS High-Level Trigger, Data-Acquisition and Controls Technical Design Report, ATLAS TDR 016, CERN/LHCC/2003-022, June 2003, http://atlas-proj-hltdaqdcs-tdr.web.cern.ch/atlas-proj-hltdaqdcs-tdr/
[15] APEX 20K FPGA, http://www.altera.com/products/devices/apex/
[16] Analog Devices SHARC DSPs, http://www.analog.com/processors/processors/sharc/index.html
[17] http://atlas.web.cern.ch/Atlas/GROUPS/DAQTRIG/DataFlow/RODCrateDAQ/RCD.html
[18] Virtex-II Pro FPGA, http://www.xilinx.com/products/tables/fpga.htm
[19] T.A.M. Wijnen, “The MROD data format and the tower partitioning of the MDT chambers”. ATL-DAQ-2003-023, 2003, http://doc.cern.ch/ archive/electronic/cern/others/atlnot/Note/daq/daq-2003-023.pdf
[20] H. Boterenbrood and J. Vermeulen, "Results of MROD-X performance tests", 26 October 2005, http://www.nikhef.nl/pub/experiments/atlas/daq/mrod/mrodprr/MRODX-performance-tests.pdf

5. Overview of MDT ROD Design

5.1. Introduction

In the following sections first an overview of the precursor of the production version of the MROD-X design, the MROD-1, is presented, followed by a short overview of the MROD-X design. The functionality of the MROD-X is a superset of that of the MROD-1, while the software environment used is identical. This is possible due to the use of SHARC DSPs in both designs. A description of some technical details judged to be relevant and not available from other documents is presented in section 6. In a companion document [3] the extra functionality of the MROD-X compared to that of the MROD-1 is described in detail. Detailed information on registers and interrupts needed for software development for the MROD is available in two separate manuals [4] [5].

5.2. System Overview

The ATLAS muon subsystem consists of 1172 Monitored Drift Tube (MDT) chambers [6] with a total of about 300,000 individual drift tubes. The largest MDT chambers contain 432 drift tubes. The chambers form 192 $\Delta\phi\Delta\eta$ towers, 164 towers consist of five or six chambers, the other 28 of seven or eight chambers. The measurements with the MDT chambers require time recordings with an accuracy of 1 ns. A schematic overview of the MDT read-out chain is depicted in Fig. 1. On-chamber Time to Digital Converters (TDCs) generate time stamps from the wire signals. The data from the TDCs are sent via the Chamber Service Module (CSM) to the MROD (MDT Read-Out Driver). As there are 192 towers a total of 192 MRODs is needed.

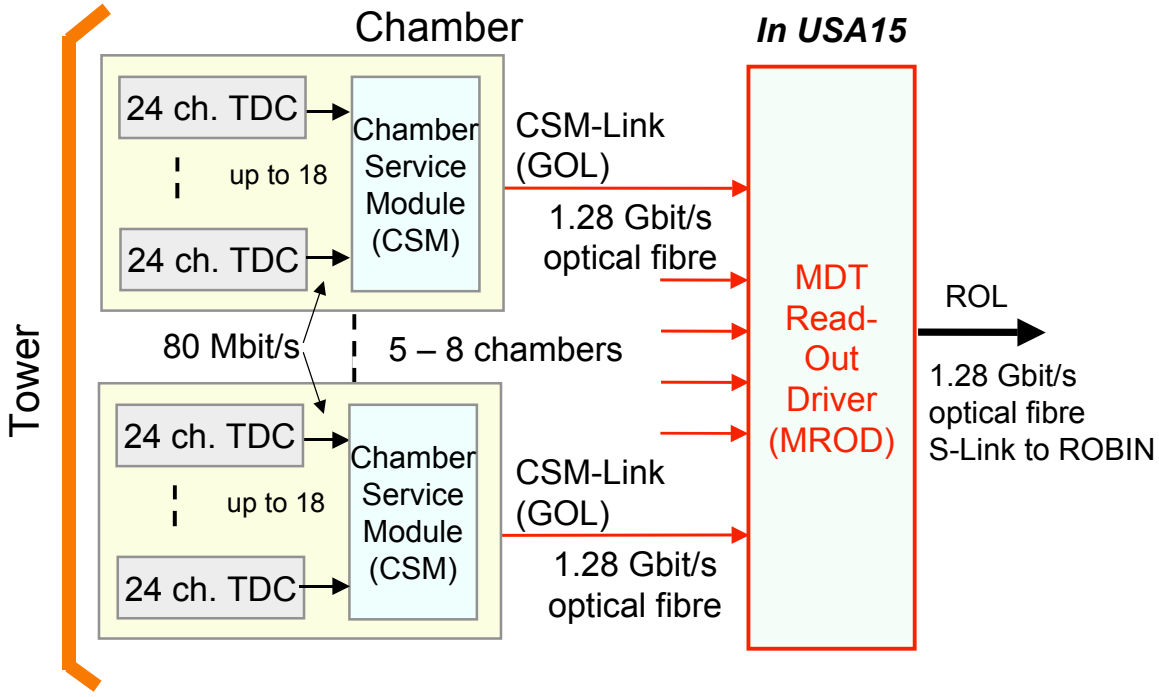

Figure 1. Overview of the MDT read-out chain.

Directly on the MDT chambers, front-end cards equipped with ASD (Amplifier-Shaper-Discriminator) chips [7] and 24-channel TDC-chips (ATLAS Muon TDC or AMT [8]) process the wire signals. Each front-end card thus serves a maximum of 24 drift tubes and the largest MDT chambers have 18 front-end cards with as many TDC-chips. The TDCs are entirely data driven and record time stamps for any wire signal above threshold, which are stored in the internal buffer memory. Upon receipt of a level-1 trigger accept, each TDC-chip selects from its internal buffer any time stamps pertaining to this particular trigger and sends them, enclosed within a header and a trailer word, over a serial 80 Mbit/s point-to-point connection to the Chamber Service Module (CSM) [9]. The header and trailer words are also output when no time stamps are found. The CSM deserializes

the 32-bit TDC data words, multiplexes them, and outputs the resulting data stream after serialization via a 0.8 or 1.28 Gbit/s (net throughput) optical link (GOL) [10] to the MDT Read Out Driver (MROD)[1]. The CSM therefore acts as a time division multiplexer. Each TDC has its fixed time slot in which a single 32-bit word of it is stored. If a TDC does not output data during its time slot, a word containing zeros is inserted by the CSM in the data stream as a placeholder. Note that the optical link provides a strictly one-way connection from the CSM to the MROD. The MRODs are physically located in a different underground area (USA15) than the detector itself, shielded from the radiation in the detector cavern. The length of the optical fibers connecting CSMs and MRODs is of the order of 100 m.

The main task of the MROD is to receive the data streams from five to eight MDT chambers, which together form a "tower". The MROD builds event fragments from the incoming data and sends these via an S-Link [11][12][13] connection, the Read-Out Link (ROL) to a Read-Out Buffer (ROB, located on a ROBIN card), from where the data can be retrieved by the second-level trigger and by the Event Builder [14]. In addition, the MROD detects and reports errors and inconsistencies in the incoming data streams (and where possible initiates corrective action). Ideally it also collects statistics and it allows to "spy" on the data. Moreover, zero suppression and data reduction schemes are implemented in the MROD.

5.3. MROD-1 Design

The MROD-1 has a modular design with separate input and output sections called MRODin and MRODout respectively. Physically the MROD-1 prototype is built as a two-slot wide 9U VME64x module with three dual-input daughter boards (the MRODins) mounted on a motherboard, the MRODout. Each MRODin serves two input channels. Per input channel the MRODin has one Altera APEX 20K200 Field Programmable Gate Array (FPGA) [15] and an associated dual-ported 1 MByte Zero Bus Turnaround (ZBT) buffer memory. In addition each MRODin has one SHARC DSP [16]. The MRODout contains two SHARC DSPs, and one APEX 20K100 FPGA that connects to the common external bus of the two DSPs, to the VME64x bus and to an S-Link interface. All DSPs in the MROD-1 prototype are strongly interconnected by means of their SHARC-links. A schematic overview of the MROD-1 is given in Fig. 2.

The design philosophy behind the MRODin is that, in the absence of any errors, the FPGA will process the incoming data without any intervention of the MRODin DSP. The FPGA demultiplexes the CSM data and removes words only containing zeros, and thus reconstructs the data streams of the individual TDCs. These streams are stored in the memory associated with the FPGA, each in its own partition. On the fly the FPGA recognizes the TDC trailer words and reads the event number. The trailer word flags that the TDC has completed the transmission of data for that particular event. The individual TDCs produce their data strictly time ordered event per event. However the data streams of the different TDCs of one chamber are not necessarily in phase, while the event fragments to be transferred to the DSP need to contain all data associated with the same event number from all TDCs. In order to detect that all these data have arrived and are stored in the memory, in the FPGA a bit in a two dimensional bit array is set upon arrival of a TDC trailer word. The TDC number determines the column of the bit. The 4 least significant bits of the event number, extracted from the TDC trailer word, define its row. Once all or a programmable subset of the bits in the row that is associated with the "expected event number" are set, the event data of one entire chamber are complete. The (expected) event number and the bits in the row corresponding to it are sent to the output controller which collects the data from the different memory partitions, each containing data of one of the TDCs. Note that for this mechanism to work properly, it is crucial that each TDC sends at least the trailer word, even in cases when it has no data. Error conditions, which may result from corrupted trailer words or erratically behaving TDCs, are detected on the basis of the bit states.

[1] With 80 Mbit/s transfers between the TDCs and CSM and the original rate of 0.8 Gbit/s for the GOL link, buffer overflows in the CSM are not excluded. A higher GOL data rate would alleviate the problem, an increase of the data rate on the GOL link to 1.28 Gbit/s has therefore been agreed, but at this rate buffer overflows in the CSM still may occur. The MRODin FPGAs can handle the higher rate, as they are clocked at 40 MHz (this has been checked with a loopback connection on one of the inputs of the MROD-X). In the MROD-X clocking the FPGAs at 50 MHz may be possible, so that a GOL link data rate of 1.6 Gbit/s (max. GOL rate > 3 Gbit/s [10]) could be handled. With this rate the absence of buffer overflows in the CSMs under any condition can be guaranteed.

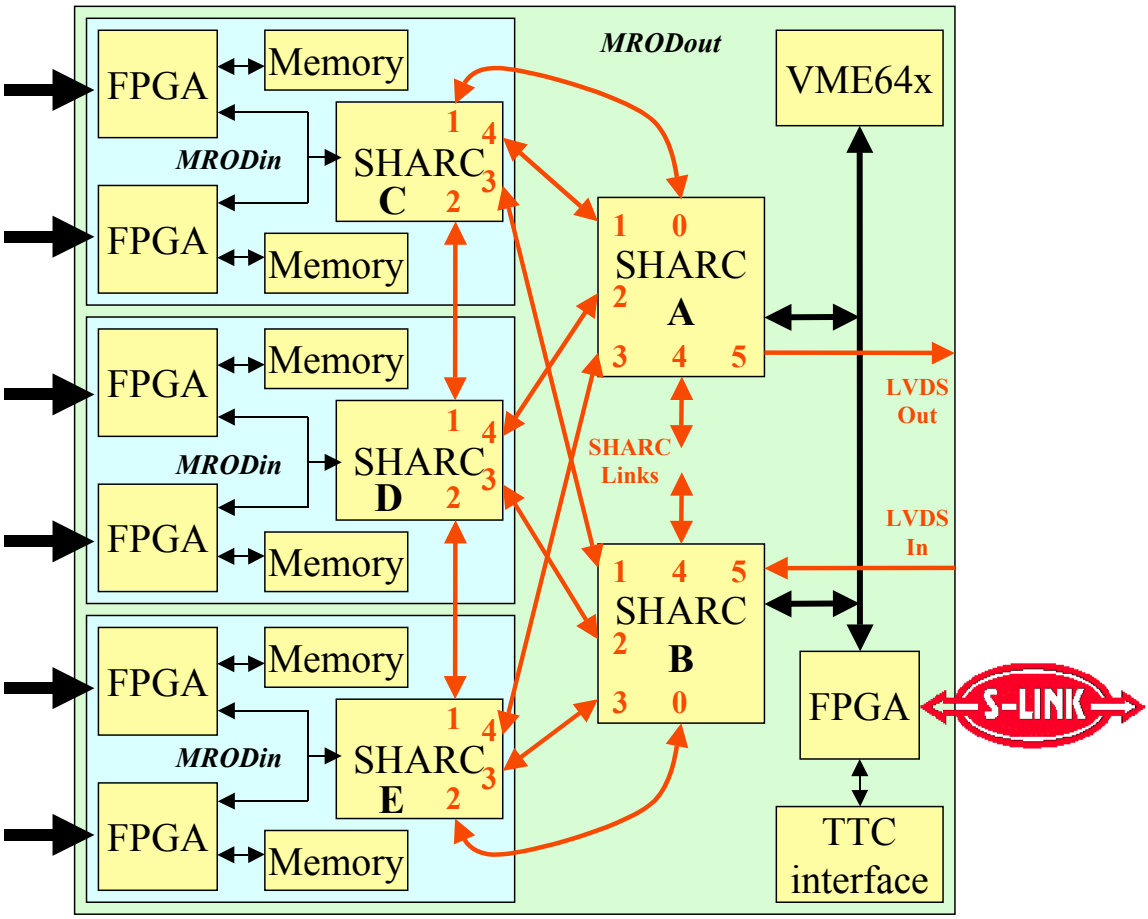

Figure 2. Schematic overview of the design of the MROD-1.

Once an event is complete, its data are transferred to the internal memory of the MRODin DSP by way of a DMA transfer, with handshaking between the FPGA and DSP. In this transfer, empty TDC envelopes may optionally be skipped, i.e. zero suppression may be applied. At this point also the chamber (or CSM) level envelope is generated and the TDC data are enclosed to form the event fragment. Together with the event and envelope data the FPGA passes via a separate FIFO one word containing the length of the fragment and the 12-bit event number to the DSP. The availability of this "length word" indicates that all event data has been read from the ZBT memory. Due to pipelined data handling these data may not all have arrived yet in the DSP memory when the information in the FIFO becomes available.

The FPGA checks for a number of error and exception conditions:

- parity errors on the TDC to CSM link (this parity is checked and errors are encoded in the data by the CSM),
- link and/or parity errors on the CSM to MROD link,
- memory partition overflow,
- incorrect or too long event fragments from a TDC,
- absence of expected trailer words or corruption of trailer words, as mentioned above.

In case an error is detected, the FPGA may interrupt the DSP, which is then assumed to intervene appropriately. For very serious conditions (too large event fragment, memory partition overflow, absence of data) the FPGA will independently decide to ignore (shut off) an individual TDC channel until the error condition is removed and integrity of the data is assured. This is flagged in one of the envelope words. Also in this case the DSP will be notified by means of an interrupt.

The MRODin DSP serves two chambers. Once the event fragments from both chambers are stored in the internal memory of the DSP, the two data blocks are, one after the other and always in the same order, passed on to one of the MRODout DSPs via one of the SHARC-links.

The task of the MRODout is to build the event fragments from the data received from the three MRODins and to output the fragments via the Read-Out Link (S-Link). At relatively low trigger rates, the MRODout may also perform monitoring tasks, collect statistics, copy events to the VME interface for spying purposes etc.

The MROD interfaces to the central Timing, Trigger and Control (TTC) system of ATLAS and to the Busy logic. The MROD will assert the Busy signal if internal congestion is detected, caused e.g. by assertion of the XOFF signal of the ROL by the ROB downstream of the MROD.

The DSPs are booted via the VME bus, the MRODin DSPs via their SHARC-links (in Figure 2 the links connecting A to link 4 of C, to link 4 of D and to link 4 of E), the MRODout DSPs via their internal memories. This is possible as both internal memory and SHARC-link interfaces of the MRODout DSPs can be accessed directly from the VME bus. Per DSP a combined loader and server program can be run on the crate processor, which runs Linux. A run-time library linked to the DSP program facilitates Unix-like handling of command line arguments, and terminal and file I/O by the DSP program. These facilities have proven to be essential for checking and debugging the software. For data taking the MRODs in a crate (12 in the final system) will be under control of and communicating with the ROD Crate DAQ (RCD) software [17]. In the 2004 test beam the MRODs were booted with the help of scripts started by the RCD. Since then better integration with the RCD has been achieved. The DSPs can now be booted directly by an MROD specific plug-in of the RCD and initialization parameters can be communicated. Also fetching of event data from the MROD via the VME bus is possible. This is important for commissioning of the MRODs in the absence of operational ROBINs to connect to.

Communication between the RCD and MROD DSPs is necessary during running for control and acquiring error and monitoring information. For this purpose a communication framework has been developed and implemented which makes it possible to send messages to individually addressed DSPs, to broadcast messages to the DSPs as well as to send messages from any DSP to the RCD. The framework also supports a special type of messages for terminal output. These are sent with the help of a method of which the parameters are identical to a subset of those possible for the “printf” function of the C standard I/O library. Text strings, numbers and formatting information are passed as part of the message, so that locally no processor time needs to be spent on parsing and formatting. A small buffer is associated with this method. Messages are deleted when the buffer overruns, but a count of lost messages is stored in the first message that again is accepted by the buffer. The messages are transferred between the DSPs via SHARC-links not used for event data transfers and not used for booting. Using these links a ring of interconnected DSPs is formed, via which data are always transmitted in one direction. One of the MRODout DSPs communicates via its internal memory with the crate processor. All messages between the DSPs again are transferred under DMA control. The framework is operated on the basis of polling and consumes a minor fraction of the DSP processor time, as it needs to be invoked only relatively infrequently with respect to the polling loop associated with event data processing. The framework has been tested with the help of the boot and server program mentioned earlier, it has been verified that the impact on the maximum event rate that can be handled is negligible. Integration with the RCD is to be accomplished soon.

To ease code development and debugging an emulation of the complete MROD has been implemented as a C++ program, which can be compiled for any environment with a suitable compiler and a command line interface. In the program each DSP is represented by an object, from which the code for the real MROD is invoked. Only a very small amount of code needs to be changed for the emulation, this is achieved with the help of conditional compilation. All DMA transfers are emulated and are started in the same way as on the real hardware. The exact timing of the various transfers is not emulated, but it is possible to control the relative order of the transfers. This has helped to reproduce errors and to find and remove the bugs responsible for these errors. Furthermore it is possible to use a modern Integrated Development Environment with support for interactive symbolic debugging and code browsing facilities, resulting in a considerable speed-up of the development and debugging process.

With the approach outlined, the speed and throughput requirements of the input stage of the MROD are guaranteed by the MRODin FPGAs. At the same time the DSPs, which can be programmed in the high level languages C or C++, allow for adaptability and flexibility in dealing with errors and exception conditions. However, taking care of the data flow at event rates up to 100 kHz is a demanding task for the DSPs. With 6 input links active and 3 MRODins sending their data to one of the MRODout DSPs the processing in the latter

DSP limits the maximum rate for small events[2] to 80 kHz. The part of the software needed to avoid overflow of the FIFO in the output S-link interface has been found to reduce the maximum achievable rate considerably, a maximum rate of 110 kHz has been measured without it. Applying both DSPs of the MRODout for fragment building can be expected to result in an increase of the maximum event rate that can be handled. As two SHARC-links per MRODin would be used for event data transport the bandwidth between MRODins and MRODout DSPs would also increase.

The MROD-1 prototype has been used on a routine basis in the 2003 and 2004 runs of the H8 test beam at CERN, Geneva. In total three MROD-1 modules were used in H8 to read out up to 15 MDT chambers. In addition to the H8 deployment, MROD-1 modules are used at NIKHEF, Amsterdam, in the quality assessment procedure, using cosmic rays, for the MDT chambers constructed at NIKHEF.

5.4. MROD-X design

The MROD-1 prototype has been evaluated on the basis of extensive general experience and in particular of the results of the throughput measurements. The performance tests showed that the SHARC-links between MRODin and MRODout form a bottleneck and moreover that most of or all the processing power of the DSPs may be needed for controlling data transfers and adding of headers and trailers to the data. The fast serial FPGA interconnection technology coming to the market offers possibilities not available at the time of the design of the MROD-1. It has been decided to change the design of the next and final MROD and make use of the Xilinx RocketIO technology, which allows inter-FPGA transfers of up to 3 Gbit/s. The new design utilizes Xilinx Virtex-II Pro FPGAs [18]. In this design (called MROD-X) the modular MRODin / MRODout structure is retained, see Figure 3 and Figure 4. The RocketIO connections are superimposed on the MROD-1 design, yielding direct connections between the MRODin and MRODout FPGAs. The RocketIO links run at 1.28 Gbit/s (net throughput). The DSPs and the SHARC-links are left in place and are now relieved from the burden of taking care of the MROD-internal event data transfers. They may be fully exploited to check and monitor the data stream and to collect statistics. They also provide the possibility to use the MROD-X in an MROD-1 compatible mode. This mode is of interest for deployment for commissioning and initial running, as existing software and firmware (ported to the new FPGAs used) can be used. The entire MROD-X has been implemented on a single board as a single-slot 9U VME64x module. Four 8-channel and two 6-channel MROD-X boards have been produced. In Figure 5 the assembled PCB of an 8-channel module is shown together with an indication of the data paths. On the six-channel boards the optical interfaces labeled with In4a and In4b, the FPGAs labeled with 4a and 4b and the SHARC DSP labeled with F are absent. On one of those boards also SHARC DSP B is absent. It is anticipated that the MROD-X can be operated without this SHARC, foreseen is a production of 176 6-channel boards and of 44 8-channel boards, all without SHARC DSP B.

[2] For larger events the bandwidth of the SHARC-link between the MRODin and MRODout DSPs of 40 MByte/s determines the maximum rate, see [1].

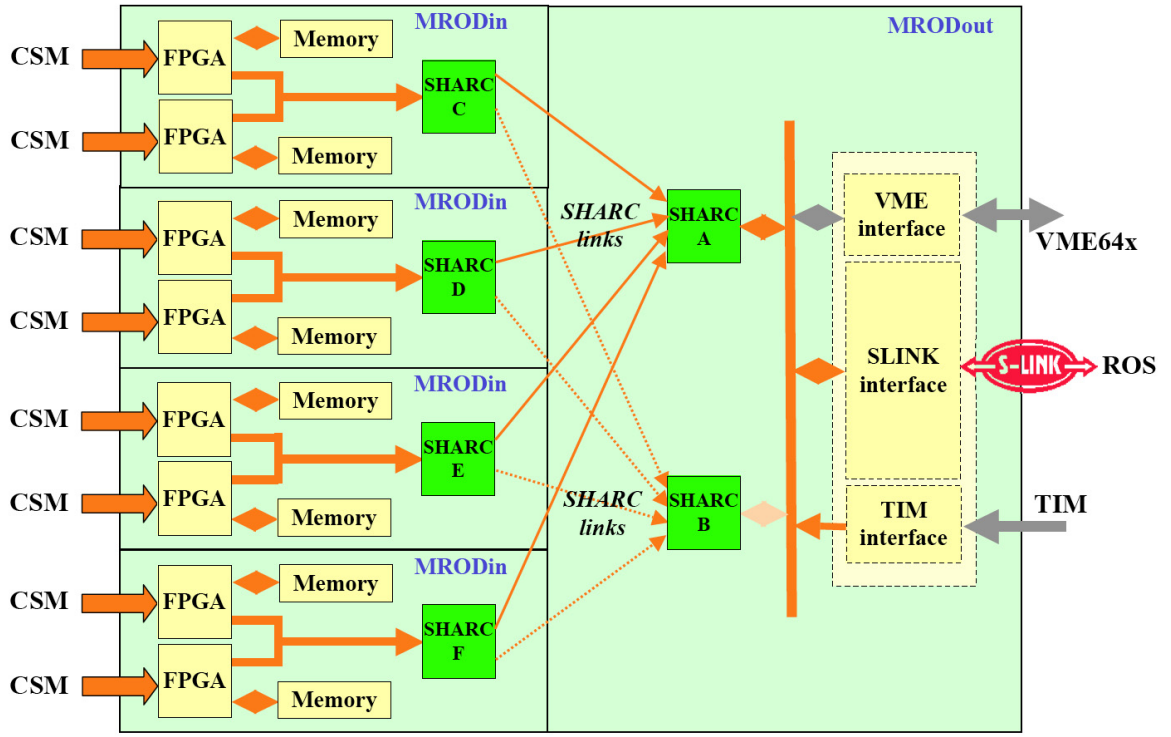

Figure 3. Schematic overview of the design of the MROD-X, with data paths used for operation in MROD-1 compatible mode indicated.

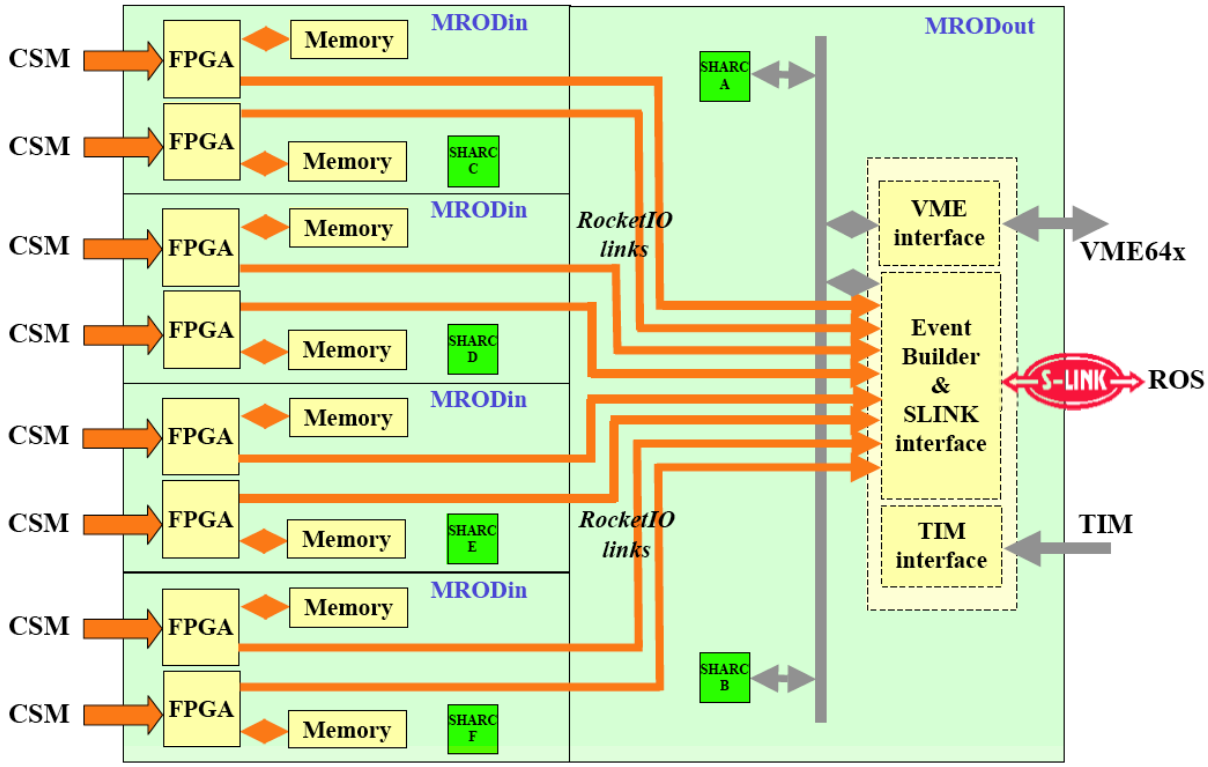

Figure 4. Schematic overview of the design of the MROD-X, with data paths used for operation in the new MROD-X mode indicated.

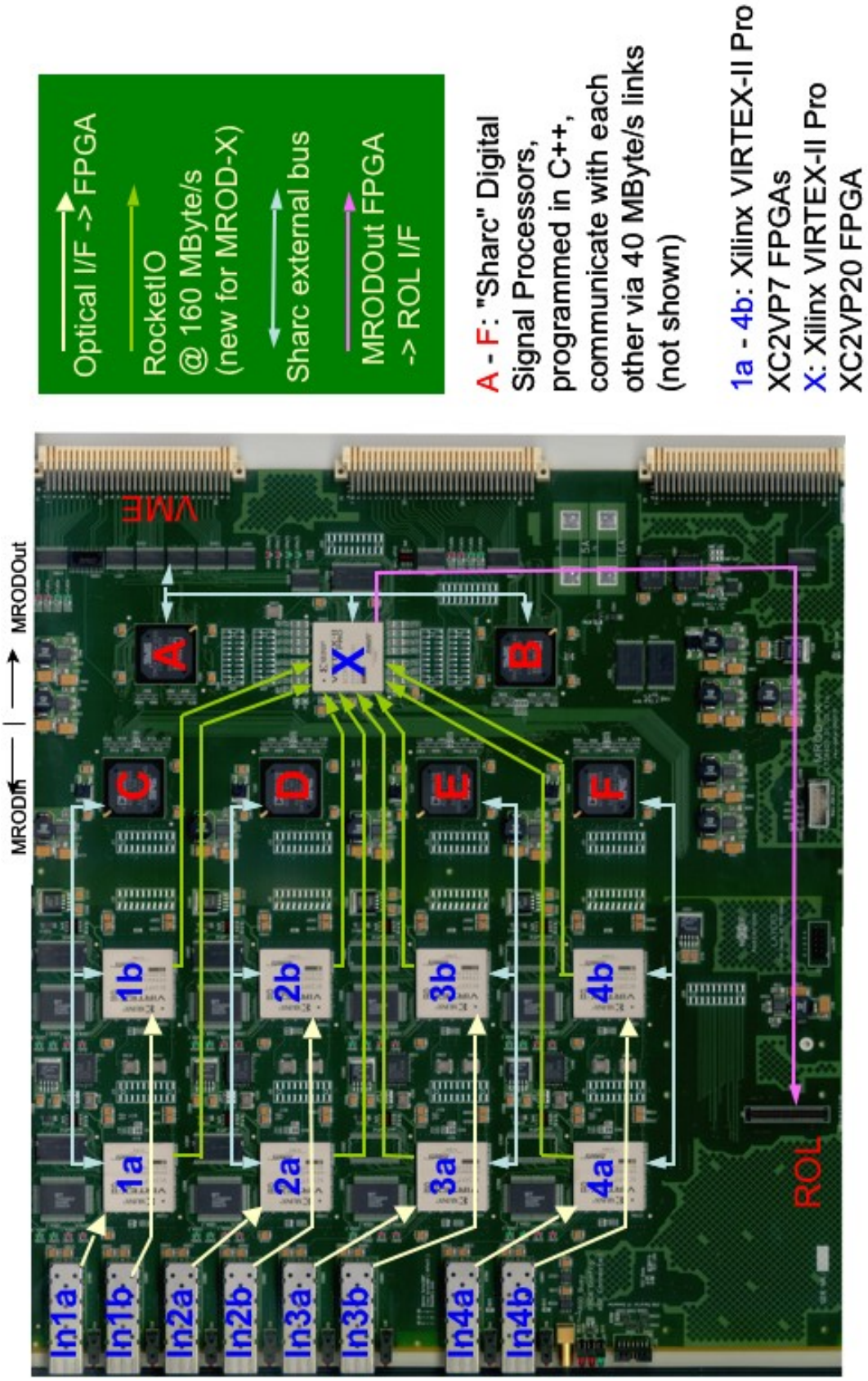

Figure 5. The assembled PCB of an 8-channel MROD-X. Data paths, except Sharc links, are indicated. The stiffening bar and the front panel are not yet mounted.

6. Technical details of the MROD-X design

6.1. Board Power

The MDT ROD is powered from the 3.3V and 5 V available on the VME P1 and P2 backplanes. Additionally 3.3 V is available via the P3 backplane (The "TIM" backplane). On-board switching regulators are used to provide additional supply voltages, generated from the 3.3 V, see figure 6. The currents specified are worst case values.

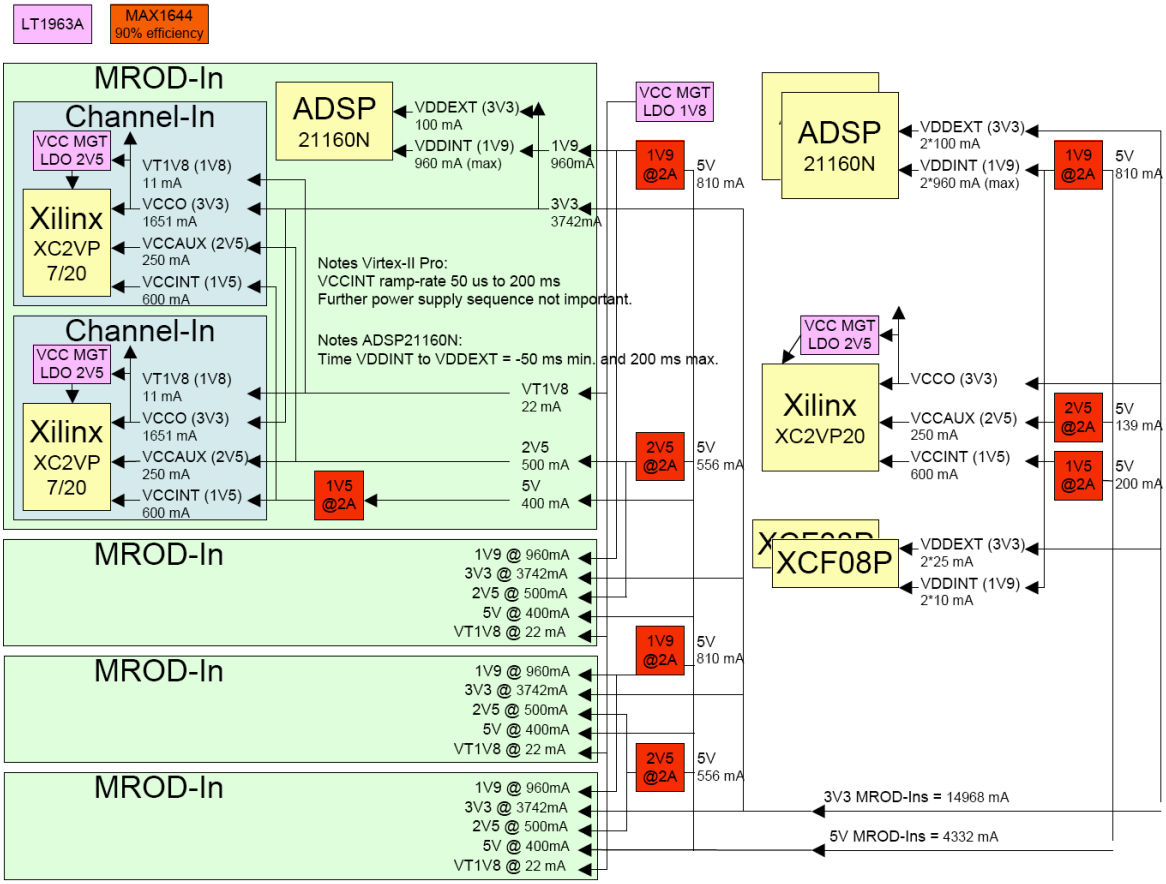

Figure 6. Supply voltages in MROD-X.

We have measured the power consumption of a fully equipped MROD-X by measuring the currents drawn. For results see Table 2.

MROD-X, SHARCs running "REGTEST" and "BLINK"			For 12 MROD-X modules per Crate
U [V]	I [A]	P [W]	I [A]
5	**2**	**10**	**24**
12	0	0	0
48	0	0	0
3.3	**11**	**36.3**	**132**
-12	0	0	0
		46.3	

Table 2 Currents drawn by and power consumption of a fully equipped MROD-X.

For comparison Table 3 contains an overview of the currents that can be delivered by the power supplies in the 9U VME crates. It can be concluded that currents drawn by and power dissipation of the MROD-X are within the allowable limit.

UEP 6021-LHC9u Power Supply		
U [V]	I [A]	P [W]
5	100	500
12	10	120
48	12	576
3.3	300	990
-12	10	120
		2306

Table 3. Maximum currents and power supplied by the power supplies in a 9U VME crate.

6.2. MROD-X clocks

Figure 7 provides an overview of the clocks in the MROD-X design. there are separate clocks for each pair of GOL links (50 or 80 MHz), for the FPGAs and Sharc DSP of each MRODin (40 MHz), for the RocketIO links (80 MHz) and for the FPGA and SHARC DSPs of the MRODout (40 MHz). The LHC clock is used for receiving TTC data (by the MRODins as well as the MRODOut). In the current design the MRODout FPGA and DSPs also run from the LHC clock if present, otherwise the 40 MHz clock is used. In the production version the MRODout FPGA and DSPs are foreseen to run at 50 MHz and the RocketIO links at 1.6 Gbit/s (net throughput), to allow maximum utilization of the available ROL bandwidth of 200 MByte/s. This requires a slight modification in the circuitry and in the lay-out of the board, as well as changing clock frequencies from 40 to 50 and from 80 to 100 MHz (for the RocketIO links). The MRODout of one of the prototypes has been modified to make running at 50 MHz possible, no problems have been found so far but more testing is required. Running the MRODin FPGAs and DSPs at 50 MHz and the GOL links at 1.6 Gbit/s (viability still to be demonstrated) is an option to be decided on for the production version.

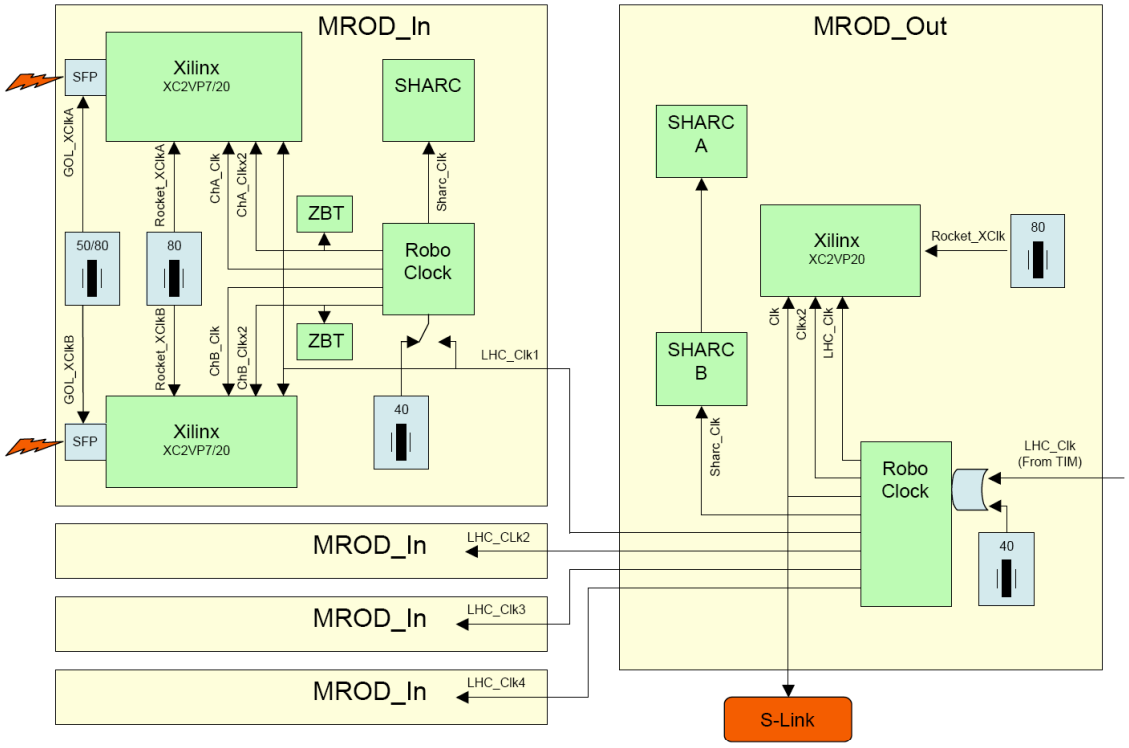

Figure 7. Clocks in the MROD-X.

6.3. *JTAG chain for the SHARC DSPs*

The SHARC DSPs can be booted via JTAG. This is not used for normal operation, but can be useful for debugging or for testing outside a crate (in particular for temperature tests in a climate chamber). The JTAG chain configuration depends on the presence of F and B, as shown in Figure 8.

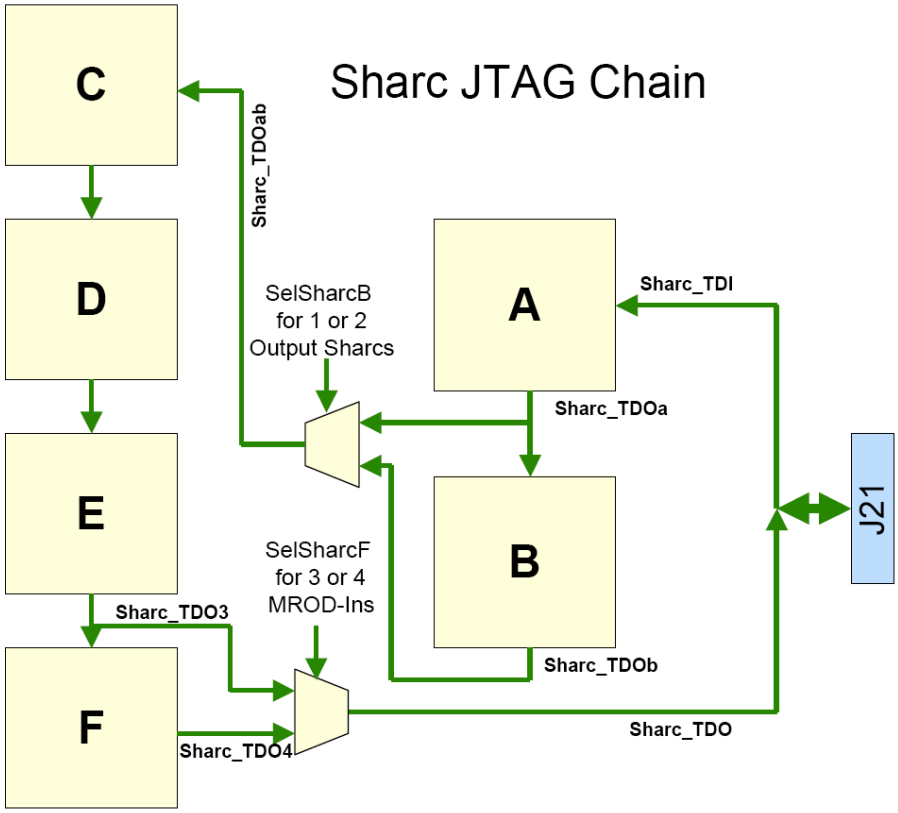

Figure 8. JTAG chain for the SHARC DSPs.

6.4. *Reset procedure*

The procedure for resetting the whole board or individual parts of it is dictated by the reset topology shown in Figure 9. After a VME SYSRESET or after a VME cycle with address modifier 10 addressing the module (see also [4]), the FPGAs are all reconfigured, the SHARC DSPs are all reset and the MRODin FPGA pipelines and associated buffers are put into "freeze" mode. Resetting of the MRODin SHARC DSPs individually needs to be done by software running on one of the MRODout SHARC DSPs (or by software running on the crate controller using one of the DMA controllers of one of the MRODout SHARC DSPs) by writing in a register a "0" to the bit corresponding to the DSP to be reset. Resetting of both MRODout SHARC DSPs only, can be done by writing to a CSR register from the VME bus.

Individual MRODin FPGAs can be reset with the help of flag signals of the SHARC DSPs, for resetting the MRODOut FPGA a VME SYSRESET or a VME cycle with address modifier 10 addressing the module is necessary. Invidual RocketIO links, the GOL links and the S-link can be reset by writing to registers in the FPGAs. Booting all DSPs requires first resetting SHARC DSP A and B (MRODout), then booting a program on SHARC DSP A (MRODout), which has to reset SHARC DSPs C, D, E and F (MRODin SHARCs) (resetting C to F can also be done by a separate program booted before booting the program to be run on A), next booting these SHARC DSPs and SHARC DSP B. Use of the communication framework in its present form requires that SHARC DSP C is booted last. This sends a special "startup" message around the ring network formed with the help of SHARC links (see next section) signaling that all DSPs have been booted. After arrival of the message in C normal operation can start.

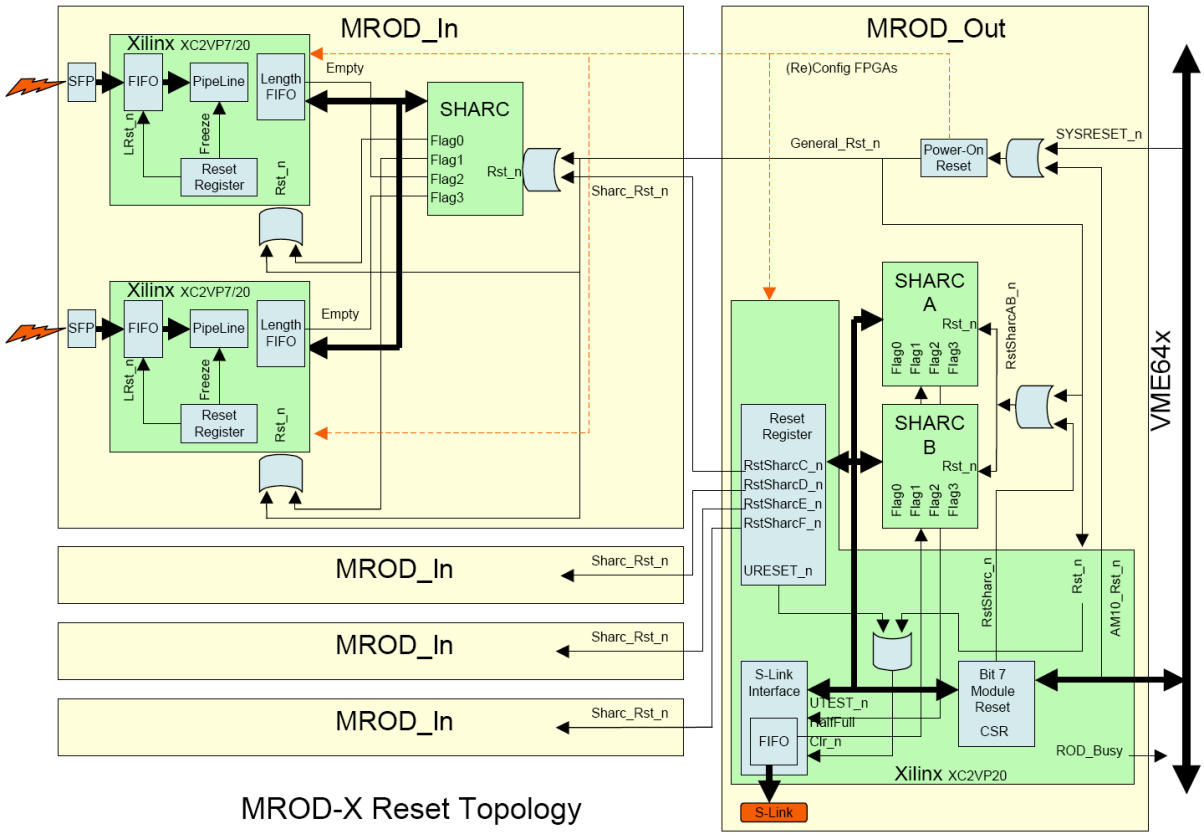

Figure 9. Reset topology of the MROD-X.

6.5. SHARC link architecture

In Figure 10 an overview of the SHARC link connections between the SHARC DSPs is shown.

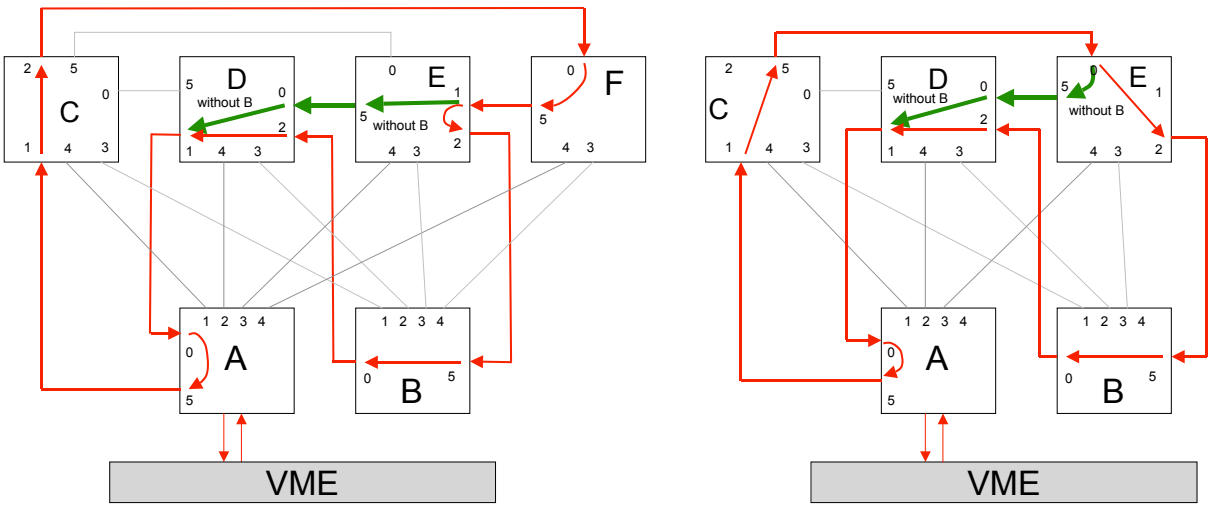

Figure 10. SHARC link architecture for MROD-X. The ring network as used by the communication framework is indicated in red for configurations with (left) and without (right) SHARC DSP F. The thick green arrows indicate how a ring network can be formed without B. The numbers refer to the numbers of the SHARC links.

7. The MROD System

The full MROD system will occupy 16 9U VME64x crates in 4 racks in USA-15. As indicated in Figure 11, each crate contains 12 MROD modules, giving a total of 192 modules for the full system. Four groups of 4 crates correspond to the 4 DAQ partitions (forward and barrel A side, barrel and forward C side). Each crate contains a TTC Interface Module (TIM) and a dedicated backplane for fanning out TTC data to the MROD modules and connecting the Busy outputs of the MROD modules to the TIM. Finally there is one VP315 crate controller per crate, running the standard ROD Crate DAQ software. The USB port of the VP315 is used to connect to the JTAG connections of the MROD modules, as described in [3] and allowing for remote updating of the FPGA configuration data stored in FLASH memory in the MROD modules. For a detailed description of the tower partitioning and the allocation of crates in USA15 see [19].

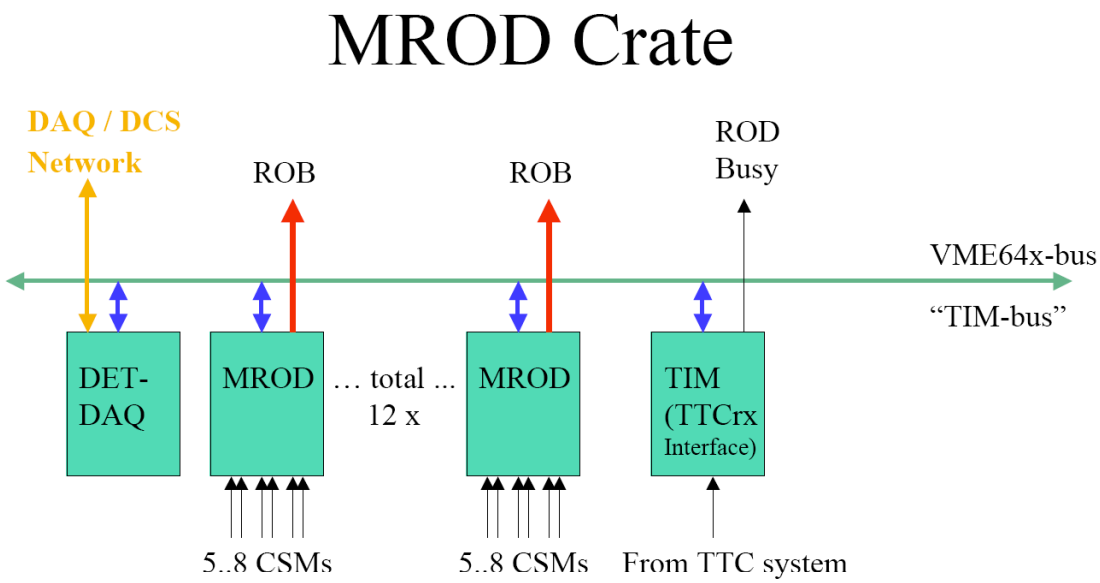

Figure 11. Schematic drawing of MROD crate.

8. MROD-X Performance

Detailed measurement results are available in the form of a set of slides [20]. Here a summary is presented from the results. The MROD-X has been used in the mode in which event fragments are passed from the MRODin FPGAs to the MRODout FPGA via the RocketIO Links and with event fragment building by the MRODout FPGA.

The maximum effective bandwidth available for transfer of TDC data on a GOL link is 89.2 MByte/s for a 25 MHz clock and 142.7 MByte/s for a 40 MHz clock and not 100, respectively 160 MByte/s due to the overhead introduced by separator words and idle words. The scheme for idle words used by the internal data generators of the MROD, i.e. 2 idle words after every 32 data words, has been assumed.

For the inverse of the maximum rate that can be sustained by the MRODin FPGA can be written:

Inverse rate = (25 + 5 * nTDCs + nWords) * time per cycle

nTDCs is the number of active TDCs, nWords is the total number of TDC words read by the FPGA from the buffer memory, irrespective of whether these words are output or not (words read may not be output if zero-suppression is switched on). For a small number of words per TDC and if only a single GOL link is used the throughput is therefore limited by the MRODin FPGA, but only for event rates higher than 100 kHz, the maximum LVL1 accept rate. For a larger number of words per TDC (> 8 words per TDC for 18 TDCs active for

a 25 MHz clock GOL link) and when only a single GOL link is used, the bandwidth of the GOL link limits the throughput, but again the event rate is above 100 kHz for up to 12 words per TDC, see Figure 12. The throughput for a small number of words per TDC in combination with the use of only two GOL links again is limited by the MRODin FPGAs. However, for all other cases the throughput is limited by the Read-Out link and by the overhead associated with event building in the MRODout FPGA.

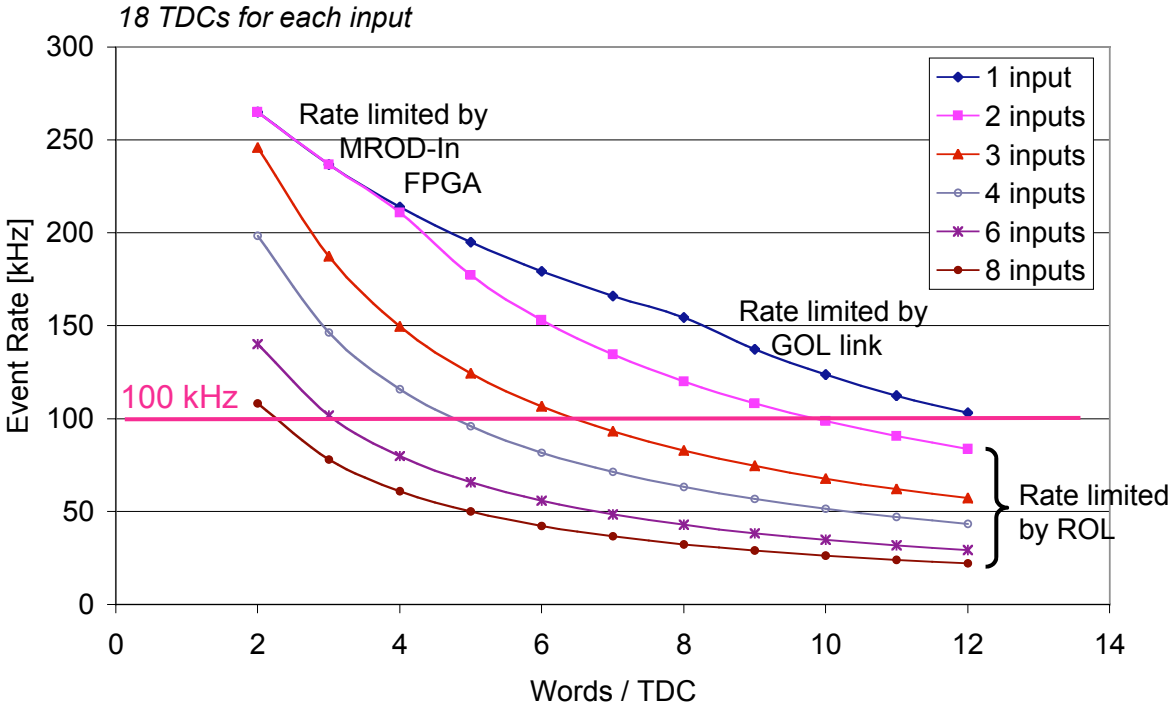

Figure 12. Maximum event rate as function of the the number of words per TDC, all TDCs output the same number of words.

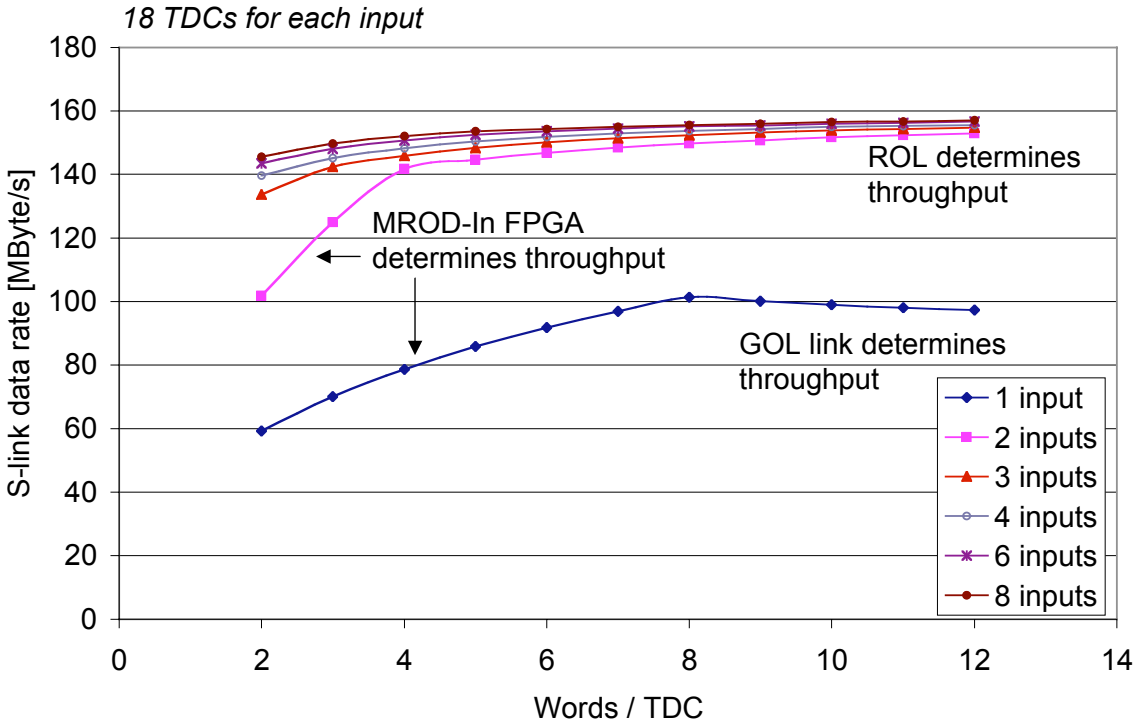

Figure 13. Maximum data rate as function of the the number of words per TDC, all TDCs output the same number of words.

For the inverse of the maximum rate that can be sustained by event building can be written:

Inverse rate = (35 + 6 * nGOLs + nWords) * time per cycle .

nGOLs is the number of GOL links used, nWords is the number of TDC words output (i.e. without words added by the MROD). The overhead of 35 + 6 * nGOLs cycles causes the maximum output bandwidth to be smaller than the nominal Read-Out Link bandwidth, see Figure 13.

9. Appendix A

9.1. List of Tables

9.2. List of Figures