A Unified Interconnection Network with Precise Time Synchronization for the CBM DAQ-System

Frank Lemke*1, David Slogsnat*2, Niels Burkhardt*3, Ulrich Bruening*4

*Computer Architecture Group, University of Heidelberg
68131 Mannheim, Germany

1frank.lemke@ziti.uni-heidelberg.de
2david.slogsnat@ziti.uni-heidelberg.de
3niels.burkhardt@ziti.uni-heidelberg.de
4ulrich.bruening@ziti.uni-heidelberg.de

Abstract—This paper focuses on the interconnection network used as a part of the Data Acquisition System of the Compressed Baryonic Matter experiment at the Facility for Antiproton and Ion Research in Darmstadt. This experiment will have special demands on the Data Acquisition System like limited space for hardware, radiation tolerance, flexible enough for all required operation modes of the detectors and support for synchronization mechanisms. The specials of the CBM networks are using only a single bidirectional fiber link for all network abilities and providing a deterministic latency message for precise time synchronisation. This led to the development of a new network and protocol.

Keywords—Data acquisition, Synchronization, Synchronous optical network, Transport protocols, Networks

I. INTRODUCTION

This paper describes a part of the Data Acquisition (DAQ) system of the Compressed Baryonic Matter (CBM) experiment at the Facility for Antiproton and Ion Research (FAIR) at GSI Darmstadt [1]. The FAIR [2] organization works together with the GSI [3] for constructing and running the planned FAIR facility. This facility will extend the existing GSI Linear Accelerator and the GSI Synchrotron. Main goal of the project is to provide highest beam intensities, high quality beams, high beam energy, high beam power, and efficient parallel usage. The planned extension consists of two high-energy superconducting synchrotrons built on top of each other in a subterranean tunnel, also of five collector, cooler, and storage rings and numerous new detectors serving five fields of physics [4]. The accelerator complex will deliver intense (5·1011) pulsed uranium beams at 2.7 GeV/u for U36+, intense (4·1013) pulsed proton beams at 29 GeV, high-energy ion beams of maximum energies around 45 GeV/u for Ne40 and close to 35 GeV/u for fully stripped U92+. The storage Rings, equipped with beam cooling facilities, internal targets, and inring experiments, will provide beams at energies 0.74 GeV/u for U36+ and for antiproton beams at energies of 3 GeV up to a maximum energy of 14 GeV. The FAIR extension is shown in Fig. 1.

The CBM experiment will investigate heavy-ion collisions in fixed target geometry, key observables are detection of open charm and light and heavy vector mesons in the di-electron and di-muon channel [1]. The main detectors are Silicon Tracking System (STS), Ring Imaging Cherenkov detector (RICH), Transition Radiation Detectors (TRD), Muon Chamber/absorber system (MUCH), Resistive Plate Chambers (RPC), Electromagnetic Calorimeter (ECAL), and the Projectile Spectator Detector (PSD) [5]. These CBM detectors have special demands on the DAQ System. Self-triggered front-end electronic (FEE) modules are used to collect data from the detectors. The complete event selection will be done after event building in a processor farm. The data flow of the detector will be up to 1TB/s. Special synchronization mechanisms are needed for synchronization of the detectors. Radiation tolerance must be provided by the frontend part of the network. The communication topology must be flexible enough for all operation modes. Space and budget are limited. This led to the usage of only a single bidirectional fiber link for clock distribution, time synchronization, control messages, and data streams. This solution seems to be efficient to provide all required features. The fiber solution also overcomes the length restriction and the problem of different voltage level at the detectors. The development under consideration of all this facts led to the solution for the unified interconnection network with precise time synchronization used in the CBM DAQ system network presented in this document.
II. CBM NETWORK AND DAQ STRUCTURE

The CBM network is a hierarchical structured network which connects three different kinds of boards, the Readout Controller Board (ROC) located close to the FEE, the Data Combiner Board (DCB) in the inner network part and the Active Buffer Board (ABB) at the backend of the network [6]. The main experiment data flow is unidirectional from the detector frontend electronics to the cluster farm at the backend. Control transactions are mostly initiated from the backend and data is exchanged in both directions. Administration packets are transmitted only on link level and are not routed through the network. A structural overview with indicated message streams is shown in Fig. 2.

![CBM Network Structure with Message Streams](image)

The ROC, also known as SysCore Board [7], is responsible for interfacing to the network links on one side and connecting to various FEEs of different kinds of detectors. This is not done directly, but by connecting to front-end boards (FEB), which carry the FEE. These FEBs can be of different types. In the current version they are assembled with a varying amount of n-XYTER ASIC [8] chips. The n-XYTER chips are self triggered and data driven. The ROC completely initializes and controls all its connected FEBs. The measured data from the n-XYTERs is always collected by a ROC and subsequently send through the links of the network. The network consists of an arbitrary number of hierarchical levels of DCBs also known as HTX-Board [9]. These boards are used for bundling small data packets to larger ones and combining the data. The current version enables the connection of up to 5 ROCs. For later development steps a first filtering of data could be implemented in the DCBs. The DCB further transmits the data to an ABB. This ABB [10] represents the interface to the commodity DAQ cluster farm. During processing the data, an ABB sorts it considering the temporal and special information. Immediately afterwards it is written into the specified local buffers. Multiple of these ROC-DCB-ABB chains are used in parallel for one detector system. The Data Acquisition Backbone Core (DABC) [11] is used as a general purpose software framework for the implementation of the DAQ.

III. STATE OF THE ART

Networks used for time synchronization must support the one-to-all communication in order to distribute the central clock event to all endpoints. All these event messages must arrive at the endpoints after a fixed travelling time. A simple solution to this requirement is to use a dedicated network just for time synchronization and make all paths to the endpoints of equal length.

Trying to synchronize endpoints in a packet oriented general purpose network like a LAN is difficult due to the variable latencies for message transports. Switching of network paths is the cause for the variable latencies. Special protocols have been developed to synchronize endpoints as close as possible, e.g. the network time protocol (NTP) [12] which allows timesyncs in the range of some microseconds. The precise time protocol (PTP) [13] allows a more precise synchronization. The CBM network requires time synchronization with a resolution of <40ps and this can only be achieved by special hardware support.

In a fused network, various traffic classes can interfere with each other and the goal to achieve a deterministic latency for the time synchronization messages is difficult. The solution is to provide a dedicated hardware path through all switching nodes with fixed latency. The arbitrated output path of the switch must have a mechanism for inserting the clock event without variable timing. This can be achieved with the Priority Insertion Method where the clock event message is inserted in the outgoing stream immediately when it arrives at the link port [14].

Using the fixed latency path to the endpoint and return the event in the same way to the clock source allows to measure the distance to the endpoint and thus allows having variable but fixed distances from clock source to each endpoint. This method is similar to the Cristian algorithm [15].

IV. CBM NETWORK AND PROTOCOL

The CBM network protocol takes into account that three different functions are integrated into the network. The detector data transport, the slow control and the timing synchronization use the same optical link and lead to three specific traffic classes. The link layer protocol has been specially designed to support this feature. Furthermore, each traffic class can be handled with different fault tolerance levels. An important aspect of the proposed network protocol is the SECDEC ability over 8b/10b-based serial links. Single bit errors that occur on the link may cause an error multiplication, generating up to 5 bit errors in the same code word. Plain forward-error correction in the 8bit domain thus is overly expensive. Previously it has been proposed to insert ECC data between 10b characters. Thus is most efficient for larger data blocks. But even for 96bit ECC-controlled blocks, the bandwidth overhead that is introduced by this method is 17% [20].

The entire set of link-layer control phits, which is for example used to frame packets, is fault tolerant with SECDED capabilities. Optimized packet structures achieve a high link utilization of about 90%. The payload of Detector Transport...
Messages is protected by a CRC for error detection. An automatic hardware retransmission mechanism guarantees the reliability of Detector Control Messages.

Time synchronization and various special event signalling are provided by DLMs. The deterministic latency for these messages is achieved by priority request insertion.

A. Traffic Classes

Three different traffic classes are provided in order to fit the requirements of the specific message types in the network. Each traffic class is represented by a virtual channel to make these classes independent of each other. This enables quality of service features. In particular, the classes have different priorities to access the physical link. The following traffic classes are supported:

- Deterministic Latency Messages (DLM)
- Data Transport Messages (DTM)
- Detector Control Messages (DCM)

The link layer has a speciality build in for the time synchronization of large networks, the Deterministic Latency Messages (DLM). This type of message has a fixed length with a packet size of only \((16+2)\) bits. They are more a special control character than a message because they carry no real payload. Instead they carry information that is directly encoded in the control characters. DLMs, as their name indicates, must always have a deterministic latency in the entire network. Therefore one of the most important features is priority request insertion, this guarantees an insertion of DLMs at any given time into the link even during data or control packets are send. Also, latency within the FPGA, and in particular within the link ports, must be constant and known beforehand. Therefore, DLMs take a dedicated data path within the FPGA that provides these features. Also the network physical layer has to provide a deterministic latency. This must be ensured at the initial restart of the system, but as well at any time later if components should be reinitialized or reset. The accumulated round trip time for DLMs must always have the same total number of clock cycles in the network. Therefore the used FPGAs have to provide a serializer/deserializer (SerDes) module with the possibility to achieve a deterministic latency. The used Xilinx FPGAs do support this only with a special configuration and an additional algorithm to guarantee the right latency setup. With such an implementation, DLMs are an ideal vehicle for synchronization or time critical service purposes message signalling. For the traffic class of Data Transport Messages (DTM) the most important ability is high bandwidth for streaming the detector data. This requirement is reached by the usage of an optimized packet structure. This leads to link utilization for data of \(64/70 = 91.428\%\) \((64/87.5 = 73.142\%\) with 8b/10b coding). Data Transport Messages can have any fault tolerance level, but currently data correction is not implemented, only detecting errors and marking erroneous packets is required. This is realized with a CRC and a marking mechanism using special characters. In case error correction should be needed in the future, a link-level retransmission as it exists for DCMs may be added.

In contrast to DTM, Detector Control Messages (DCM) as the third message class needs a high level of fault tolerance. Therefore they are protected with CRCs and automatically retransmitted in case of an error.

Service packets like acknowledgements or idles are handled by the link port and are not visible to the core module. They are also coded as special characters to provide fault tolerance. The credit based flow control is handled internally in the link port modules.

B. Packet Format

Physical transfer digits (phit) of 16 data bits plus 2 control bits are the smallest data unit in the network. For the serial transmission on the fiber, each phit is translated into two 10b code words. A phit may either be data payload of a packet, or a control character.

Packet structures are shown in Fig. 3. DTM and DCM packets may have a payload of 8 to 64 bytes. Every flit is framed with start and end phits and a 16bit CRC. Although theoretically, no framing of a flit would be required, the start of a flit and the end-of-flit character provides additional security at little cost and also avoids carrying a length field in the header which must be interpreted to know the length.

The first phit of a packet is a start-of-packet (SOP) control character for DTM and start-of-special-control (SOSC) for DCM. End-of-packet (EOP) control characters mark the end. The start control characters also include information about the virtual channel in which a packet flows.

DLMs are somehow different from this concept, as they do not contain payload information. Instead, they consist of two control characters.

C. Control Characters

Besides start and end characters, a link protocol must support DLMs, credits and acknowledgements. Additionally, idle, retransmission and management characters exist. As the number of 8b/10b K characters is by far not sufficient to encode these control characters, a combination of K and D characters is used [19]. To avoid confusions between control characters and 8b/10b characters, control characters are referred to as control phits.

The first character of a control phit is one of K.28.7, K.27.7, K.28.3, which have a hamming distance \(HD>=3\). The second character is one of a set of D characters with \(HD>=4\). There is no closed formula to generate such a set of D.
characters. Instead, the set has been found by a brute-force search through the set of D characters. Decoding with error correction and detection are thus performed by a lookup-table based approach.

K and D characters are combined in such a way that all single bit errors in a phit can be corrected and all double bit errors can be detected, thus providing SECDEC capabilities [16].

D. Error Correction & Detection

Single bit errors in control phits can be corrected using the SECDEC mechanisms described above. Assuming that there are no hardware defects, double bit errors within the same control phit are extremely rare. Loss of data in such a case is acceptable. Nevertheless, it is important to detect such an error in order to prevent inconsistent system states. In the case of detected double or multi bit errors in control characters the link will thus be reinitialized.

Besides this protection of the protocol’s control information, another important topic is how payload information is protected. For both the payloads of DTMs and DCMs, the receiver can detect errors by checking the packets CRC. It is acceptable to have data loss for DTMs, these will simple be discarded if an error has been detected.

For DCMs, such a data loss is not acceptable. DCMs are protected by a retransmission-based protocol. All packets that are transmitted over the link are also copied to a retransmission buffer in the sending link port. If the receiving part of the link positively acknowledges the reception of a packet, the buffer space can be freed. If the acknowledgement is a negative acknowledgement, the sending part will instead initiate a retransmission of all flits in the retransmission buffer. This mechanism can also cope with lost or corrupted acknowledgements using a sequence identifier within the acknowledgements.

As the retransmission occurs on link-level and the latency of link transmission and processing is very low, a hardware-implemented retransmission FIFO has a low FPGA resources usage.

E. Link Initialization

The goal of the link initialisation is to ensure a proper link start-up after system power up, reset, or cable hot plug. The link initialization checks the availability of the link cable and tries to establish a bidirectional link channel.

During link initialization, each link port starts sending a predefined pattern. This allows the Xilinx multi-gigabit transceivers (MGT) on both sides of the link to synchronize on the stream, to recover the clock and to properly de-serialize the incoming data. If the MGTs signal, that they can recover a stable clock, this is the first indication that there is actually an active receiver on the link. The link port will then continue to send the pattern, and will also check the incoming data for correctness. After a defined period of receiving correct initialization data, a link port will try to do a handshake with the remote link port. If the other side did not detect an error either, the handshake succeeds and the link switches to operation mode. Otherwise, initialization is restarted again.

The complete CBM network modules are coded in Verilog HDL. Their control interface is optimized for adaption in all parts of the network and delivers easy to use valid-stop synchronization for all three traffic classes. All low level link information, which is not needed by the upper layers, is intentionally non transparent through the interface.

V. Synchronization Mechanism

The most important ability to enable synchronization over fiber links is to guarantee a deterministic latency over the links. Therefore the FPGA’s serial transceivers have to be configured in a special way for always starting with the same basic latency after link initialization. The control logic implementation must also allow a deterministic insertion of synchronization signals or characters into the link.

For Xilinx FPGAs, we use the Multi-Gigabit Transceivers (MGTs), as well as the GT transceivers. Different control logic parts of the MGTs and GTPs have to be configured to guarantee a reproducible deterministic behaviour. The basic configuration is derived from the MGT user guide [17] and is adapted with the following steps. The wrapper generated by the RocketIO Wizard had to be modified to have access to the barrel shifter position signal RXLOSSOFSYNC of the alignment block to read out the deserialized data. The fabric interface had to be chosen with the width of 32 bit to guarantee deterministic behaviour at the interface. This is required because the internal MGT width is 32 bit and multiplexing must be avoided. Also the derived parallel clock generated by the clock data recovery (CDR) can lead to different barrel shifter positions. To obtain deterministic latency the CDR can be reset for altering the barrel shifter position of alignment until the predefined value is reached. This is achieved by restarting the low level initialization of the GT11_INIT_RX module. Because of the difference between the local clock oscillators the resulting barrel shifter position varies and the right alignment will always be achieved after a number of resets. Through this configuration method a reproducible and deterministic latency is achieved after link initialization when the same bit file is used for all FPGAs.

Another very important fact which has to be considered is the usage of an identical clock in all parts of the network. Therefore the clock must be derived from a single oscillator and the system must provide a SerDes bit precise deterministic latency. This is needed to achieve the same deterministic latency after restarting the network or parts of it. Therefore at least the send clocks of the transmitters in the network must be identically and the deterministic logic parts need to be source synchronous. A system clock to ensure this could be provided to all parts by a separate clock distribution net or through a clock recovery for reusing the input clock in the transmit path. We decided against the overhead of a separate network, but for a clock recovery mechanism.

The clock recovery mechanism possible within the Xilinx FPGAs of the CBM network delivers a peak-to-peak jitter between 80ps and 100ps. This jitter is not precise enough to
be used as SerDes clock, because it does not fulfill the MGT specification [18] maximum value of 40ps. The solution is to use an additional jitter cleaner device to clean the received clock for CDR and feed it back into the FPGA as reference clock for the transmit path. Fig. 4 shows the jitter cleaner extension board developed for proving the feasibility of this implementation method.

When designing this device the idea was to use only standard COTS parts. It should not only be usable for jitter cleaning, but also as clock source. As connector to plug it onto a system board a standard mezzanine connector is provided. This connector delivers the default recovered clock from the FPGA and all configuration signals for the IC at the extension board. The optional clock source is a Silicon Labs SI570, a via I2C configurable oscillator. Due to our bad experiences regarding jitter quality of a clock switching within the FPGA, we use a separate clock switching IC. A Texas Instruments CDCLVD110 is used to select the FPGA clock between the onboard clock and the recovered clock. The chosen clock is then cleaned or configured within the National Semiconductor LMK03000 chip. The clock jitter of the cleaned clock for peak-to-peak is below 40ps, which is enough for usage as reference clock for the MGTs and GTPs. The magnitude of the measured RMS jitter is below 10ps. Fig. 5 shows a jitter measurement with a cleaned clock of the jitter cleaner board.

The jitter cleaned clock has been used as transmit clock within a test setup at our lab and passed all requirement of the MGTs.

As every other control phit, DLMs are coded with a 1 bit forward error correction. After initialization of the epoch counter in the ROCs, arriving DLMs are only used to check that the detector is still synchronous. After the loss of several DLMs it is assumed, that an error occurred and a resynchronization starts. Thus even a complete DLM could get lost in the network without influencing the functionality. Additionally there are DLMs available for user defined special event signalling with deterministic latency. The character coding of all currently available DLMs is depicted in Table 1.

<table>
<thead>
<tr>
<th>Name</th>
<th>Upper Byte</th>
<th>Lower Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLM0</td>
<td>K 27.7</td>
<td>D 10.3</td>
</tr>
<tr>
<td>DLM1</td>
<td>K 27.7</td>
<td>D 14.1</td>
</tr>
<tr>
<td>DLM2</td>
<td>K 27.7</td>
<td>D 20.1</td>
</tr>
<tr>
<td>DLM3</td>
<td>K 27.7</td>
<td>D 20.6</td>
</tr>
<tr>
<td>DLM4</td>
<td>K 27.7</td>
<td>D 22.3</td>
</tr>
<tr>
<td>DLM5</td>
<td>K 27.7</td>
<td>D 28.2</td>
</tr>
<tr>
<td>DLM6</td>
<td>K 27.7</td>
<td>D 28.5</td>
</tr>
<tr>
<td>DLM7</td>
<td>K 27.7</td>
<td>D 6.2</td>
</tr>
<tr>
<td>DLM8</td>
<td>K 27.7</td>
<td>D 14.6</td>
</tr>
<tr>
<td>DLM9</td>
<td>K 27.7</td>
<td>D 3.1</td>
</tr>
<tr>
<td>DLM10</td>
<td>K 27.7</td>
<td>D 11.2</td>
</tr>
<tr>
<td>DLM11</td>
<td>K 27.7</td>
<td>D 17.2</td>
</tr>
<tr>
<td>DLM12</td>
<td>K 27.7</td>
<td>D 25.3</td>
</tr>
<tr>
<td>DLM13</td>
<td>K 27.7</td>
<td>D 17.5</td>
</tr>
<tr>
<td>DLM14</td>
<td>K 27.7</td>
<td>D 3.6</td>
</tr>
<tr>
<td>DLM15</td>
<td>K 27.7</td>
<td>D 5.3</td>
</tr>
</tbody>
</table>

This could be a trigger mechanism for certain detector readout features or a special control implementation. The usage of this message based synchronization requires beside the described specific FPGA configuration and clock recovery abilities, also a deterministic and a structural separated hardware implementation. In addition the insertion into the link message stream with fixed latency is necessary. This could happen within a deterministic time multiplexing scheme or with a guaranteed latency for an insertion at any time. The later solution is more flexible and was chosen for the implementation. We call this feature priority request insertion. This enables the possible insertions shown in Fig 6, between two packets, within message parts or even right into the data payload of a packet. Therefore it has to be ensured that there is never more than one DLMs inserted into a packet. This leads to a minimum interval size of 70 bytes / 2 bytes per cycle = 35 cycles.

The cleaned recovered clock together with the user-defined MGT/GPT configurations for achieving a fixed initialization state guarantees the deterministic behaviour of the link. This enables the special feature of the CBM network, the Deterministic Latency Message. DLMs are packets with the size of a single 16 bit phit within the network. This enables a phit-based synchronization of the network. Therefore the latency for each link is measured during link initialization and DLMs are periodically send to synchronize an epoch marker.

![Fig. 4 Jitter Cleaner Extension Board](image)

![Fig. 5 Jitter Cleaned Clock](image)

![Fig. 6 DLM Insertion Examples](image)
VI. Conclusions

Our experiences with networks in the area of cluster computing led us to the development of the protocol and the FPGA- and PCB implementations.

The network combines three different functions onto the same network fabric, while past solutions often used different networks for these features. Our unified solution is expected to save costs and manpower in particular during the build-up phase. Deterministic Latency Messaging together with low-jitter and synchronous clock recovery allow for precise time synchronization in this self-triggered detector system.

The network protocol is scalable, as an example additional virtual channels may be added in the future, or link speeds may be increased. Also, error correction for data packets may be added. This allows for reuse in future detector systems with minimal changes. At the same time, an effective link bandwidth of up to 73% is reached with 8/10b coding. Further improvements can be achieved by a more promising method the use of scrambling. Self-synchronous scramblers do not multiplicate errors. They also could overcome the 20% bandwidth overhead that 8b/10b coding introduces. However, scrambling is worse when it comes to DC-balance and maximum running lengths. Without full control and knowledge over the FPGA serializers, such an approach is currently a game of luck.

The new concept of the network has been implemented, verified by simulation and tested in system prototypes. The functionality and usability of all developed concepts is proven in hardware during numerous lab tests. Also, other groups that participate in this project successfully tested our design. The next step in development will be the construction of a larger demonstrator. Demonstrator I is currently under construction and will be tested during the test beam time in August or September 2009.

Acknowledgments

We gratefully acknowledge the following person for his outstanding support: Walter F.J. Mueller, GSI Helmholtzzentrum fuer Schwerionenforschung GmbH.

We also want to thank Sven Schenk from the Computer Architecture Group for bringing his experiences with Xilinx and Altera serializers into this project.

References

[17] “Virtex-4 RocketIO Multi-Gigabit Transceiver”, UG076 (v4.1), Xilinx Corporation, San Jose, USA.
[18] “Virtex-4 FPGA Data Sheet: DC and Switching Characteristics.”, DS302 (v3-4), Xilinx Corporation, San Jose, USA.