Project information

Project full title	European network for developing new horizons for RIs
Project acronym	EURIZON
Grant agreement no.	871072
Instrument	Research and Innovation Action (RIA)
Duration	01/02/2020 – 31/01/2024
Website	https://www.eurizon-project.eu/

Deliverable information

Deliverable no.	2.4
Deliverable title	Functional tests of the developed readout solution
Deliverable responsible	WUT
Related Work-Package/Task	WP2, task 2
Type (e.g. Report; other)	Report
Author(s)	Wojciech M. Zabołotny and Marek Gumiński, for the WP2.2 participants
Author(s) affiliation	Warsaw University of Technology (WUT)
Dissemination level	Public
Document Version	1.0
Date	23.01.2024
Download page	https://www.eurizon-project.eu/results/deliverables

Document information

Version no.	Date	Author(s)	Comment
1.0	24.01.2024	Wojciech M. Zabołotny, Marek Gumiński (WUT)	

Table of Contents

Spis treści

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

Introduction

The initial aim of that task in the initial CREMLINplus project was the development of the readout chain for the BM@N experiment at JINR, utilizing the synergy with preparing the readout chain for the CBM experiment at GSI.
However, after the Russian aggression against Ukraine, the Project was converted into the EURIZON project and the goals of the task WP2.2 have been reoriented accordingly.

Instead of developing the readout chain for a single experiment, a more versatile, modular solution suitable for different applications was needed. The main aim was to enable work on mCBM and CBM DAQ in remote sites that are not allowed to use the GBTX-based readout boards or the CRI1 boards. Another purpose was to provide a limited readout solution for quality-assurance sites to control the tested front-end electronics and receive the data.
An additional expected result was providing a base solution for developing of readout chains for other experiments.

The general view of the developed readout chain is shown in Figure 1.

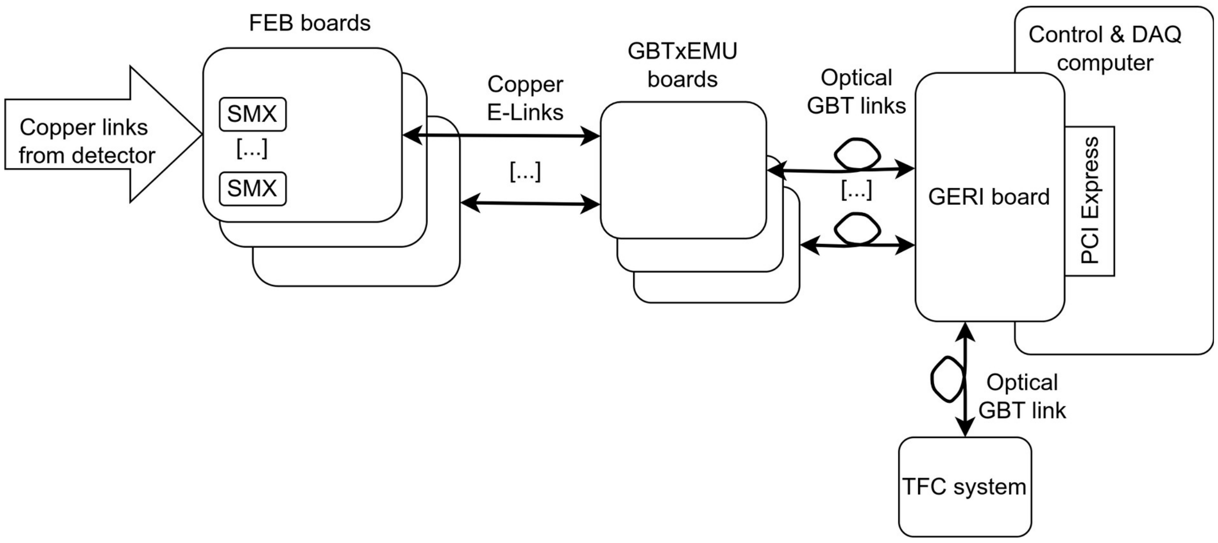


Figure 1. General view of the readout chain based on GERI and GBTxEMU boards. The FEB boards contain the SMX front end ASICs. The GBTxEMU is a replacement for the GBTx-based CROB boards used in the standard CBM readout chain. The GERI board is a replacement for the CRI board used in the standard CBM readout chain.

Background

Certain goals were achieved in the first phase of the project when it was realized as CREMLINplus. The basic readout chain based on the GBTxEMU board controlled by the old DPB board [1] was created and tested. The tests have proven its ability to acquire the data in the triggerless mode correctly.
The concept of the GERI board based on a commercial TEC0330 board from Trenz has been created, and necessary modifications (including replacement of the clock generator and adding routing of the clock signal) have been defined.

Development of IP cores for the standard CBM readout chain and the one created in the Project was started. Those included the new SMX controller (HCTSP bridge) and a PCIe-Wishbone bridge.
Most of the results achieved in that phase have been published in [P1] and [P2].

Achievements in the second phase of the Project

During the second phase, significant work was oriented toward developing the final GERI firmware. The dedicated DMA engine for high-performance data acquisition was created. It was designed with testability in mind. Therefore, the hardware architecture, the associated Linux driver, and the sample control and data acquisition application was tested in the modified QEMU emulator (described in [P3]). The block diagram of the emulation environment is shown in Figure 2. A hardware implementation using the new High-Level Synthesis (HLS) approach was created and tested based on the results achieved.

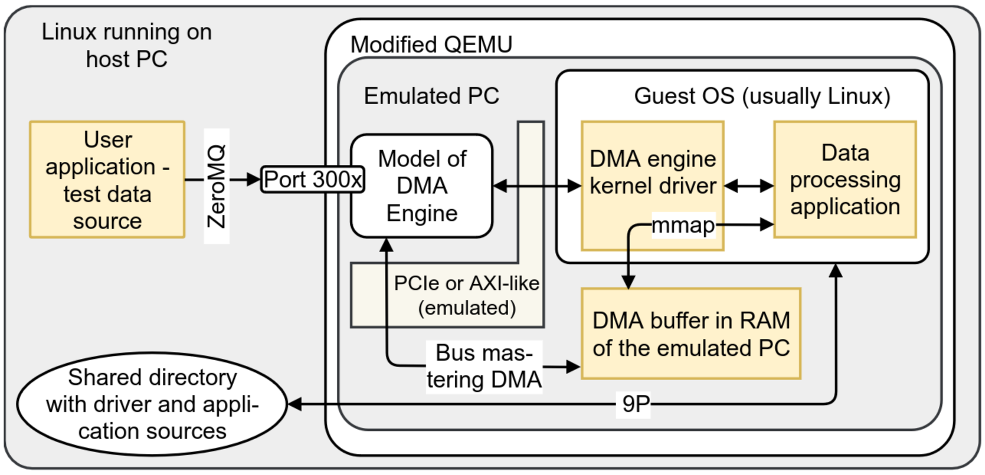

Linux running on host PC
User application - test data source
ZeroMQ
Port 300x
Modified QEMU
Emulated PC
Model of DMA Engine
Guest OS (usually Linux)
DMA engine kernel driver
Data processing application
mmap
PCIe or AXI-like (emulated)
DMA buffer in RAM of the emulated PC
Bus mastering DMA
Shared directory with driver and application sources
9P

Figure 2. The block diagram of the QEMU-based system used for the verification of the DMA engine architecture and for the associated software development. Image reproduced from [P3].

The DMA engine enables the acquisition of data split into packets. Different packets may be processed in parallel with multiple CPU cores. Additionally, it contains a PCIe-Wishbone bridge. The DMA engine is described in publication [P4], and its block diagram is shown in Figure 3.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

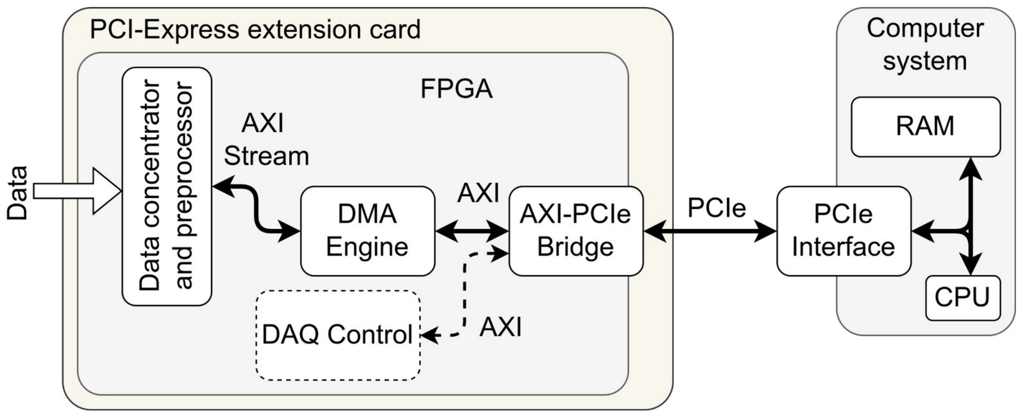


Figure 3. The block diagram of the DMA engine implemented with HLS technology. Diagram reproduced from [P4].

Another important solution for GERI firmware developed and tested in the framework of EURIZON project was the data concentrator. To solve the problems faced during the earlier work on data preprocessing and concentration in the CBM readout chain [2,3], the new concentration scheme with the interconnection network was proposed. The first solution [P5] was based on the Beneš network, simplified using the brute-force approach analysis. Such a concentrator was sufficient for collecting data from 8 inputs but required relatively big memory storing possible network configurations. Further analysis of the concentrating network resulted in the development of a scalable (tested for 16 inputs in hardware and for 32 inputs in simulation) and significantly simpler concentrator described in [P6] (the block diagram of the concentrator is shown in Figure 4). That concentrator has been successfully integrated into the GERI firmware.

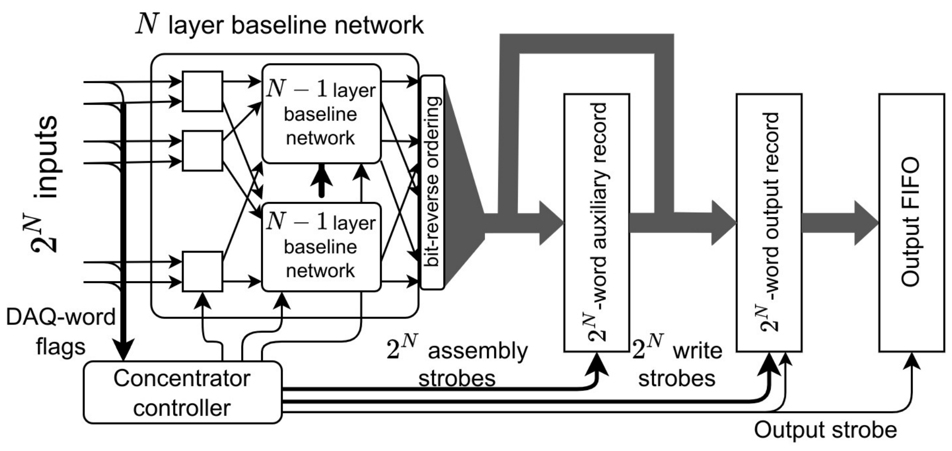


Figure 4. Block diagram of the scalable data concentrator. Diagram reproduced from [P6].

The GBTxEMU firmware was also subjected to further improvements. To better match the needs of testing sites for the CBM experiment at FAIR, the firmware has been modified to enable a 160 MHz E-Link clock in FEE communication, with runtime switchable of E-Link clock frequency. Additionally, the code was refactored for more flexible E-Link pins allocation and E-Link signals routing.
The GERI firmware has also been modified to support 80 MHz and 160 MHz E-Link clock frequency. However, the frequency selection must be done before the firmware compilation.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

To properly reconstruct tracks and build the events, the system clock and time counters in the GERI board and connected GBTxEMU boards must be properly synchronized with the reference clock and time value delivered to the GERI board. In the CBM experiment, the dedicated TFC (Time and Fast Control) system will be responsible for providing those signals. For the readout chain developed in the Project a model of that system has been implemented as a modification of the DPB firmware [1] running on an AFCK board. To test the proper synchronization, the TFC model transmits the reference clock and time counter via the GBT link and generates an LVDS time signal corresponding to the lowest bit of the time counter. The signal is inverted when the selected bit of the time counter changes its value – resulting in the injection of time markers every 2^N clock period.

The synchronization system shown in Figure 5 has already been implemented in GBTxEMU. A similar synchronization system shown in Figure 6 has been added in GERI.

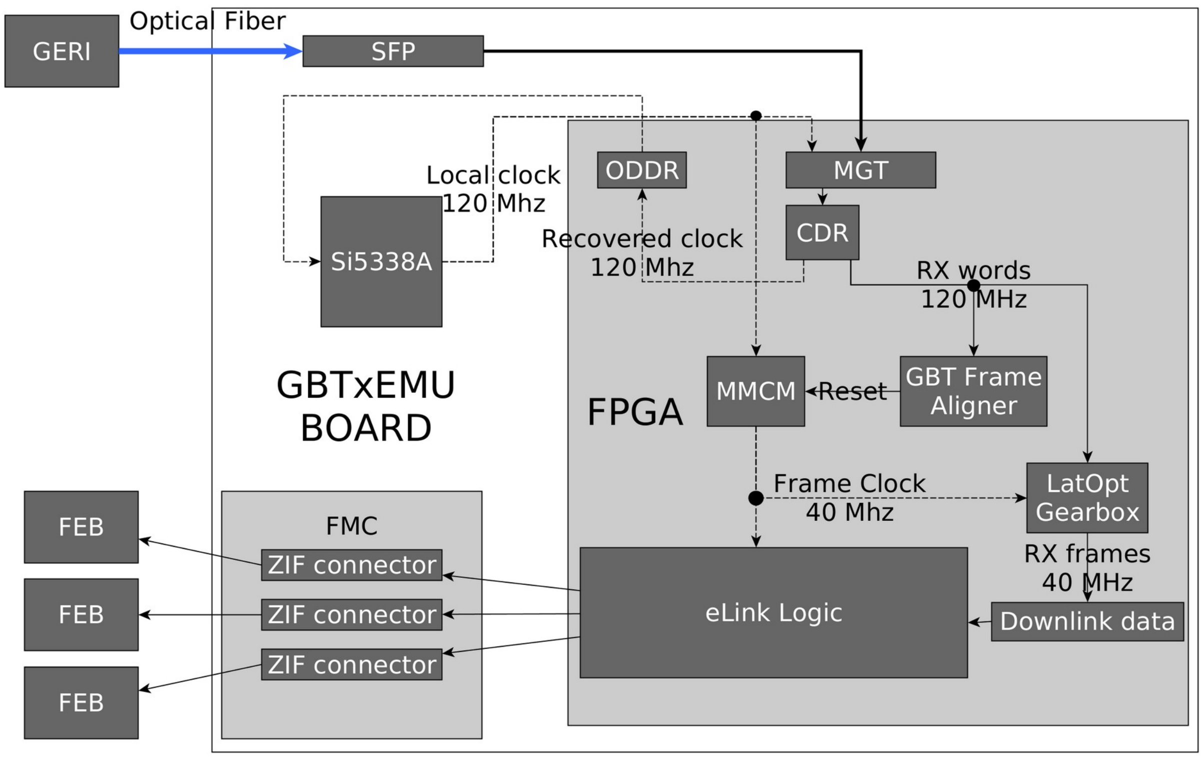


Figure 5. Clock recovery, jitter cleaner and synchronization system in GBTxEMU board.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

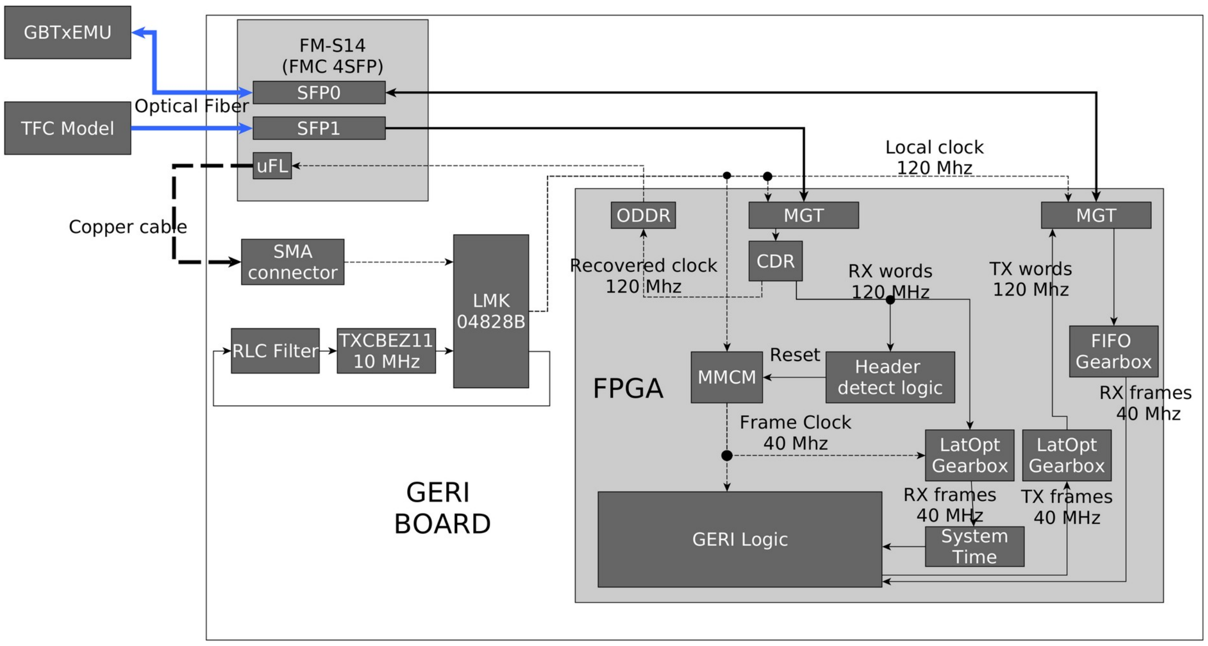


Figure 6. Clock recovery, jitter cleaner, and synchronization system in the GERI board.

Additionally, especially for the synchronization tests, the GERI firmware was modified to transmit the time counter value in unused bits of the downlink frame. The GBTxEMU has been extended with a receiver that samples the LVDS timing signal at a frequency of 960 MHz, detects the time marker and records its position with ca. 1 ns resolution related to the active slope of the recovered reference clock. That position, together with the time counter value are written to the Wishbone-accessible FIFO. The whole system for testing the synchronization is shown in Figure 7.

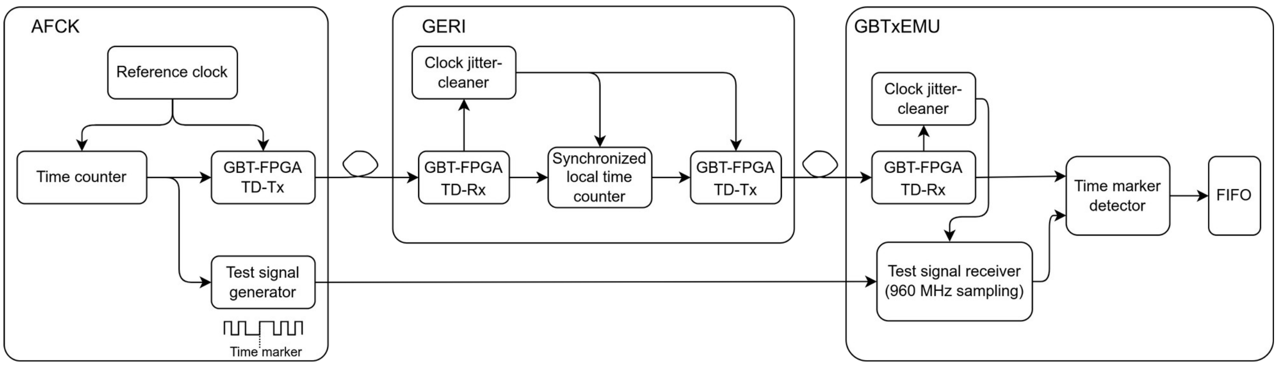


Figure 7. The whole system for testing the synchronization of the readout chain.

The software supporting the readout chain consists of the Linux driver for the GERI board and the data acquisition application written in C, and of the control scripts written in Python.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

Functional tests performed

The following tests have been performed to verify the essential functionalities of the prepared readout solution. The test have been ordered so that the particular test relies on the correct result of the previous ones.

Test of communication between the Linux driver and GERI board

The FPGA firmware was uploaded to the GERI board located in a PC computer, and the computer was restarted. Then the presence of the device was tested with the `lspci` utility:

```
# lspci -v -d 32ab:8018
24:00.0 Memory controller: Device 32ab:8018
     Subsystem: Device 32ab:0007
     Flags: fast devsel, IRQ 79
     Memory at a0000000 (64-bit, non-prefetchable) [size=256M]
     Memory at bfce0000 (64-bit, non-prefetchable) [size=128K]
     Memory at b0000000 (64-bit, non-prefetchable) [size=128M]
     Capabilities: [80] Power Management version 3
     Capabilities: [c0] Express Endpoint, MSI 00
     Capabilities: [100] Advanced Error Reporting
     Capabilities: [300] Secondary PCI Express
     Kernel modules: wzdaq_drv
```

The board was properly recognized as a PCI Express device supported by the wzdaq_drv driver. After loading the driver, the memory areas were properly mapped, the interrupt line was connected, and the device was recognized as a bus-mastering DMA device. Additionally, the special files for communication with the board were created:

```
# insmod wzdaq_drv.ko
# dmesg
[…]
[549397.972252]  mmio_start=bfce0000, mmio_end=bfcfffff,
mmio_len=20000, irq=79
[549397.972276]  agwb_start=b0000000, agwb_end=b7ffffff,
agwb_len=8000000
# lspci -v -d 32ab:8018
24:00.0 Memory controller: Device 32ab:8018
     Subsystem: Device 32ab:0007
     Flags: bus master, fast devsel, latency 0, IRQ 79
     Memory at a0000000 (64-bit, non-prefetchable) [size=256M]
     Memory at bfce0000 (64-bit, non-prefetchable) [size=128K]
     Memory at b0000000 (64-bit, non-prefetchable) [size=128M]
     Capabilities: [80] Power Management version 3
     Capabilities: [c0] Express Endpoint, MSI 00
     Capabilities: [100] Advanced Error Reporting
     Capabilities: [300] Secondary PCI Express
     Kernel driver in use: wzab_daq1
     Kernel modules: wzdaq_drv
# ls -l /dev/my_*
```

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

```
# ls -l /dev/my_*
crw------- 1 root root 238, 0 Jan 22 19:56 /dev/my_ctrl0
crw------- 1 root root 239, 0 Jan 22 19:56 /dev/my_daq0
```

The test results confirmed correct operation of the GERI board with the driver

Test of communication between the controlling software and GERI board

After loading the driver, the Python script for controlling the GERI board was loaded. The access to its internal registers was verified.

```
# python3
>>> import versatile_dma_iface
>>> geri = versatile_dma_iface.versatile_dma_iface("/dev/my_ctrl0")
>>> import agwb
>>> regs = agwb.top(geri,0x0)
>>> print(hex(regs.ID.read()))
0x1ed91fca
>>> print(hex(regs.VER.read()))
0xadd15038
>>>
```

The test has proven the correct operation of the access to the internal Wishbone bus of the GERI board from the Python-based control software.

Test of communication between the controlling software and GBTxEMU boards

After loading the driver, the Python script for configuring the GERI board and establishing the communication with GBTxEMU was loaded. The access to GBTxEMU internal registers was verified.

```
# python3 -i gbt_access.py 2
Waiting for tfc rx lock
RB_CLKin0_LOS        : 0
RB_CLKin0_SEL        : 1
RB_CLKin1_LOS        : 0
RB_CLKin1_SEL        : 0
RB_CLKin2_SEL        : 0
RB_DAC_VALUE         : 513
RB_HOLDOVER          : 0
RB_PLL1_LD           : 0
RB_PLL1_LD_LOST      : 1
RB_PLL2_LD           : 1
RB_PLL2_LD_LOST      : 0
Locked!
Waiting for frame clock alignment with GBT frame
Waiting for MMCM lock
MMCM locked
GBT RX Ready
```

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

```
GBT frame clock not locked to GBT frame!
Resetting MMCM
Waiting for MMCM lock
MMCM locked
GBT RX Ready
GBT frame clock not locked to GBT frame!
Resetting MMCM
Waiting for MMCM lock
MMCM locked
GBT RX Ready
GBT frame clock not locked to GBT frame!
Resetting MMCM
Waiting for MMCM lock
MMCM locked
GBT RX Ready
GBT frame clock not locked to GBT frame!
Resetting MMCM
Waiting for MMCM lock
MMCM locked
GBT RX Ready
GBT frame clock locked to GBT frame!
{'link_ready': 1, 'rx_ready': 1, 'rx_header_locked': 1, 'tx_ready':
1, 'tx_phaligned': 1, 'tx_phaligned_val': 1}
Succeeded in : 1
Read GBTxEMU firmware ID: 0x1ed91fca and VER: 0x52c6231d
>>>
```

The test results agree with expectations. After a few attempts, the appropriate alignment of the GBT transmitter frame clock is achieved. Then, the proper operation of the GBT uplink incoming from the GBTxEMU is detected. Afterward, access to the ID and VER registers in the GBTxEMU is correctly performed.

Test of communication between the controlling software and FEE ASICs in FEB boards

After loading the driver, the Python script for configuring the GERI board and establishing the communication with the FEE ASICs (SMX) was loaded.

The script has correctly recognized the configuration of the connected FEB board:

```
# python3 -i gbt_access_part.py 1
{'link_ready': 1, 'rx_ready': 1, 'rx_header_locked': 1, 'tx_ready': 1,
'tx_phaligned': 1, 'tx_phaligned_val': 1}
Succeeded in : 1
Clock:2
Read GBTxEMU firmware ID: 0x1ed91fca and VER: 0x52c6231d>>> print(se)
[
Setup Element:
  Group: 1
  Downlink: 0
  Uplinks: [0, 1, 2, 3, 4, 5, 6, 7]
  ASICs Map:
    ASIC address 0x0: (ASIC uplink, uplink): (0, 4)
    ASIC address 0x1: (ASIC uplink, uplink): (0, 0)
    ASIC address 0x2: (ASIC uplink, uplink): (0, 5)
    ASIC address 0x3: (ASIC uplink, uplink): (0, 1)
    ASIC address 0x4: (ASIC uplink, uplink): (0, 6)
    ASIC address 0x5: (ASIC uplink, uplink): (0, 2)
    ASIC address 0x6: (ASIC uplink, uplink): (0, 7)
    ASIC address 0x7: (ASIC uplink, uplink): (0, 3)
  Clock Phase Characteristic:
    Optimal Phase: 24
    Window Length: 70
    Eye Windows:
      Uplink  0:
____________________________________________________________XXXXXXX_____________
      Uplink  1:
____________________________________________________________XXXXXXX_____________
      Uplink  2:
______________________________________________________________XXXXX_____________
      Uplink  3:
____________________________________________________________XXXXXXX_____________
      Uplink  4:
______________________________________________________________XXXXXXX___________
      Uplink  5:
______________________________________________________________XXXXXXXX__________
      Uplink  6:
______________________________________________________________XXXXXX____________
      Uplink  7:
____________________________________________________________XXXXXXX_____________
  Data phase characteristics:
    Uplink 0:
      Optimal Phase: 13
      Window Length: 60
      Eye Window: ____________________________________________XXXX________________
    Uplink 1:
      Optimal Phase: 11
      Window Length: 57
      Eye Window: ________________________________________X___XXX_________________
    Uplink 2:
      Optimal Phase: 23
      Window Length: 45
      Eye Window: X_____________________________________________X_XXXX____________
    Uplink 3:
      Optimal Phase: 12
      Window Length: 58
      Eye Window: __________________________________________X_XXXX________________
    Uplink 4:
      Optimal Phase: 29
      Window Length: 46
      Eye Window: __XXXXX______________________________________________XXXXX______
    Uplink 5:
```

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

```
      Optimal Phase: 29
      Window Length: 46
      Eye Window: __XXXXX______________________________________________XXXX_______
    Uplink 6:
      Optimal Phase: 27
      Window Length: 48
      Eye Window: XXXX________________________________________________XXXX________
    Uplink 7:
      Optimal Phase: 24
      Window Length: 47
      Eye Window: X_______________________________________________XXXX____________
]
>>>
```

Access to the internal registers of the SMX chips was verified:
Reading the SMX addresses:

```
>>> sxs[1].read(192,22)
1
>>> sxs[2].read(192,22)
2
```

Read/write access:

```
>>> sxs[2].read(192,14)
0
>>> sxs[2].write(192,14,1234)
>>> sxs[2].read(192,14)
1234
>>>
```

The test has shown correct communication between the controlling software and the FEE ASIC (SMX) chips in the FEB board.

Test of transmission of data from FEE ASICs to the PC hosting the GERI board

The following preparatory steps were performed before the test:

- Loading the GERI driver.
- Running the script for initialization of huge-pages-based data buffer
- Starting the data acquisition application

```
# ./wzdaq_app_geri 0
I'm trying to open our device!
I'm trying to set number of HP!
I'm trying to set DMA buffer!
I'm trying to start engine!
I'm trying to switch on IRQ!
```

Then the Python script configuring the SMX chips to send the test data was started:

```
# python3 -i gbt_access_dma.py 1
```

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

```
{'link_ready': 1, 'rx_ready': 1, 'rx_header_locked': 1, 'tx_ready':
1, 'tx_phaligned': 1, 'tx_phaligned_val': 1}
Succeeded in : 2
Read GBTxEMU firmware ID: 0x1ed91fca and VER: 0x52c6231d
```

The data-receiving application started to receive the packets with acquired data:

```
[…]
Evt:52, first:387982, last:39929f, w2=0, w3=0
00000034579acce7,0000001b72831125,0000000000000000,0000000000000000,
247ffb36247ffaea,24fefbe524fcf3c9,24fefbe524fcf3c9,247ffd76247ffd44,
[...]
24c51457247ff958,247ffbc6247ffae8,247ffb38240007c0,247ffd4a247ffc8a,
00000034ed9acce7,0000001b72c01a25,0000000000000000,0000000000000000,
Evt:53, first:39929f, last:3aa94c, w2=0, w3=0
00000035579acce7,0000001b72c01a25,0000000000000000,0000000000000000,
247ffeee247ffdd8,22201994240004e8,247fff30247ffea6,247ff85024c8208f,
[...]
247fff4822e0820d,247ff8da24cc30c4,24da69a9247ff91e,24da69a924cd34d2,
00000035ed9acce7,0000001b72fd2325,0000000000000000,0000000000000000,
Evt:54, first:3aa94c, last:3bc0b2, w2=0, w3=0
00000036579acce7,0000001b72fd2325,0000000000000000,0000000000000000,
247ffab024cd34d2,247ffcd2247ffb30,247ffd7a247ffc52,247ffef424000420,
[...]
247ffa44247ff9d6,22eaaaa4247ffa3c,22ebaeb2247ffb5a,24d6596724ecb2c3,
00000036ed9acce7,0000001b733a2c25,0000000000000000,0000000000000000,
Evt:55, first:3bc0b2, last:3cd881, w2=0, w3=0
00000037579acce7,0000001b733a2c25,0000000000000000,0000000000000000,
24d6596724ecb2c3,247ffd98247ffdce,247fffaa247ffe50,247ff8fe247fff84,
[...]
24dc71ce24f8e382,247ff9e8247ff8fa,222019e424dd75d8,247ffb48247ffa38,
00000037ed9acce7,0000001b73773525,0000000000000000,0000000000000000,
[…]
```

The reception of data could be controlled using the appropriate GERI register

```
>>> g.regs.datapath.triv_proc.ctrl.run.writef(0) # Stop reception
>>> g.regs.datapath.triv_proc.ctrl.run.writef(1) # Start reception
```

The received data were stored in the file and verified for correctness.
The test has proven the correct transmission of data.

Test of synchronization of the readout chain

A test of synchronization was performed in the configuration shown in Figure 7. The TFC model running in the AFCK board was working continuously. The GBTxEMU and GERI were initialized, and afterwards, the synchronization was checked in two ways. The first test provided very fast

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

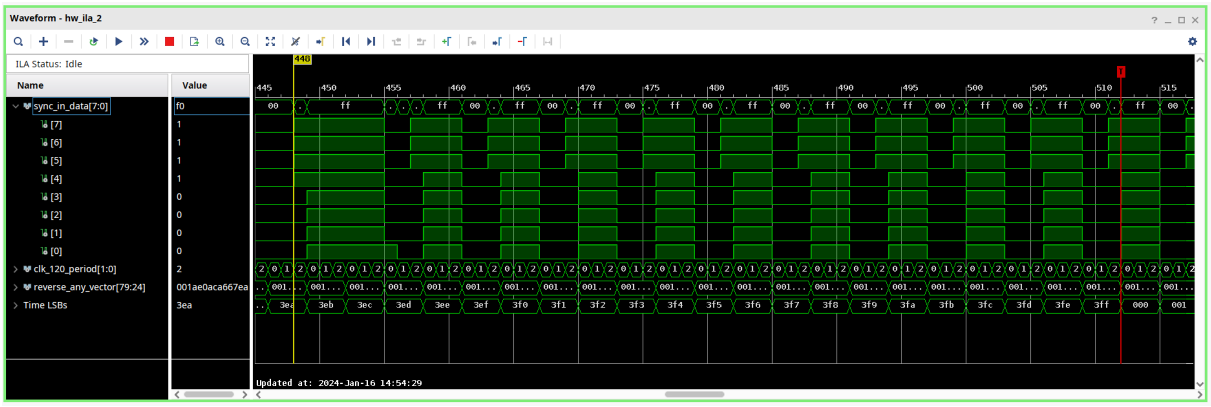
verification. The delay between the center of the received time marker and the moment when the selected bit in the received time counter value has changed was checked using the ILA analyzer. The example recorded waveforms are shown in Figure 8. The fluctuation of delay by ca. 1 ns is visible.

The measurement was repeated after reinitialization of GBTxEMU and/or GERI. The results have shown that the delay is stable with accuracy of ca. 1 ns between reinitializations. However, the recompilation of the firmware may change the delay.

For verification of the long-term stability of the delay, the FIFO containing information about the time position of the central slope of the time marker was read periodically. Those tests have shown that the delay remains stable in the range of ±1 ns.

The tests have proven correct synchronization and coherent delay after reinitialization of GBTxEMU. However, small delay changes have been observed if the GBTxEMU was reinitialized shortly after reinitialization of the GERI board. That problem may be worked around, but it is worth further investigation and elimination.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.



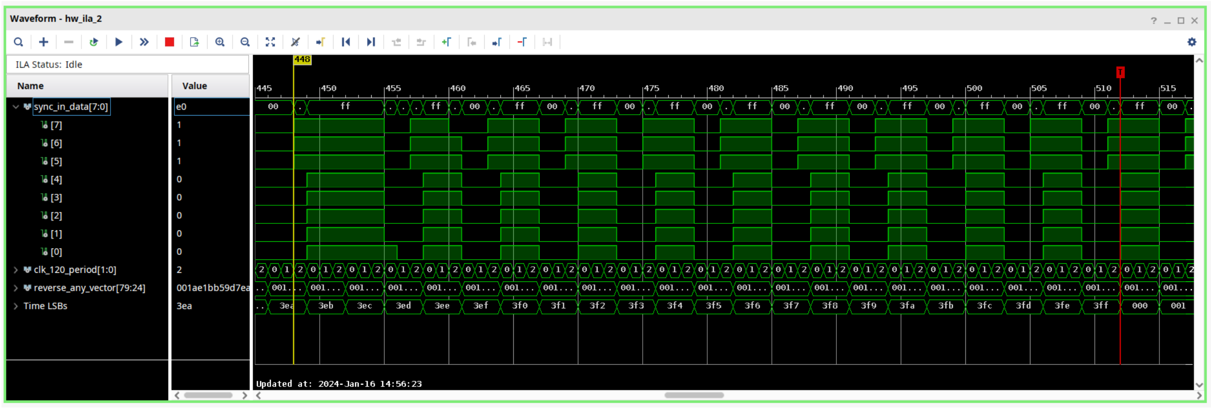


Figure 8. Two recordings from ILA logic analyser showing the delay between the received time marker and the associated change of the received time counter value. The change of the time counter is delayed, because it is transmitted by two GBT links in series. The sync_in_data signal is the output of the deserializer capturing the data at 960 MHz.

Conclusions

The functional tests have shown that the developed readout solution may be a basis for preparing various readout configurations. It is possible to use the full chain consisting of GERI and GBTxEMU boards, or one of its components (either GERI or GBTxEMU) with the standard mCBM or CBM readout chain. That configuration may be especially useful for testing purposes.
The components of the developed DAQ chain are used at GSI for STS assembly testing (for Full module performance tests, Burn-in tests, and Ladder assembly tests) and at GSI and KIT for STS module assembly tests (see Figure 9).
The developed chain is also planned to be used in external experiments, including the J-PARC E16 Experiment and the STRASSE Experiment t RIBF in RIKEN Nishina Center.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.

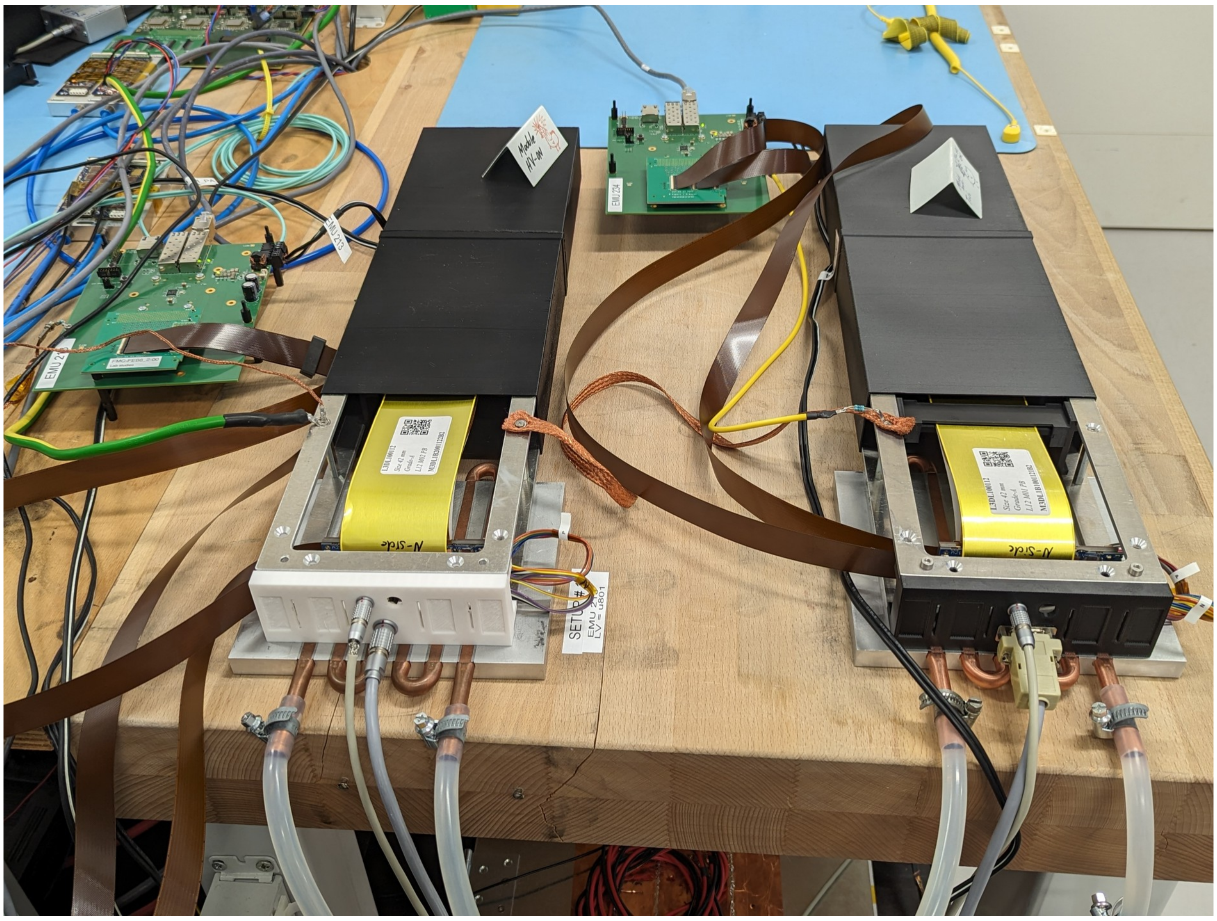


Figure 9. The GBTxEMU boards used for testing the STS detector modules in GSI.

Publications being result of the Project

[P1] D. Dementev et al., “Fast Data-Driven Readout System for the Wide Aperture Silicon Tracking System of the BM@N Experiment,” Physics of Particles and Nuclei 52, no. 4 (July 1, 2021): 830–834, DOI: 10.1134/S1063779621040213.

[P2] W.M. Zabołotny et al., “GBTX Emulator for Development and Special Versions of GBT-Based Readout Chains,” Journal of Instrumentation 16, no. 12 (December 1, 2021): C12022, DOI: 10.1088/1748-0221/16/12/C12022 (open access preprint at https://arxiv.org/abs/2109.11591).

[P3] W. M. Zabołotny, “QEMU-Based Hardware/Software Co-Development for DAQ Systems,” Journal of Instrumentation 17, no. 04 (April 4, 2022): C04004, DOI: 10.1088/1748-0221/17/04/C04004 (open access preprint at http://arxiv.org/abs/2109.14735).

[P4] Zabołotny, Wojciech Marek. 2023. “Versatile DMA Engine for High-Energy Physics Data Acquisition Implemented with High-Level Synthesis,” Electronics 12, no. 4: 883 (February 9, 2023). DOI: 10.3390/electronics12040883 .

[P5] M. Gumiński et al., “Beneš Network Based Efficient Data Concentrator for Triggerless Data Acquisition Systems,” Electronics 12, no. 6 (March 17, 2023): 1437.
DOI: 10.3390/electronics12061437
[P6] W.M. Zabołotny, “Scalable Data Concentrator with Baseline Interconnection Network for Triggerless Data Acquisition Systems”. Electronics 13, no. 1. (December 23, 2023): 81.
DOI: 10.3390/electronics13010081

References

[1] W.M. Zabołotny and G. Kasprowicz "Data processing boards design for CBM experiment", Proc. SPIE 9290, Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2014, 929023 (25 November 2014); DOI: 10.1117/12.2073377

[2] M. Gumiński et al. “High-Speed Concentration of Sorted Data Streams for HEP Experiments.” Acta Physica Polonica B Proceedings Supplement 11, no. 4 (2018): 689.
DOI: 10.5506/APhysPolBSupp.11.689.

[3] M. Gumiński et al. “Sorting of STS-XYTER2 Data for Microslice Building for CBM Experiment.” In Proceedings of Topical Workshop on Electronics for Particle Physics — PoS(TWEPP2018), 143. Antwerp, Belgium: Sissa Medialab, 2019. DOI: 10.22323/1.343.0143.

[4] W.M. Zabołotny et al. "Control and Diagnostics System Generator for Complex FPGA-Based Measurement Systems" Sensors 21, no. 21 (November 6, 2021): 7378.
DOI: https://doi.org/10.3390/s21217378

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871072.