DEVELOPMENT OF MONITORING TOOLS FOR FELIX PROJECT

Anastazija Zivkovic[1]

Supervisors: Felix Zahn, Ph.D. & Carlo Alberto Gottardo, Ph.D.

ATLAS Trigger and Data Acquisition (TDAQ) System

CERN Summer Student Programme 2021

This document consists of a report about the task I have been working on during the CERN Summer Student Programme. The work described here is a part of the new FELIX project that will be implemented in the ATLAS Data Acquisition System during Phase I upgrade. LHC is currently under shutdown and that is why this upgrade is necessary for future data acquisition because it is able to provide a higher bandwidth for the readout system and to update the hardware. Specifically, my work consisted of the development of monitoring tools for FELIX. Access to the felix-monitoring tools is very straightforward and the content is shown using charts so that the whole process of analysis of the data could be simplified. The assignment is accomplished by receiving the data directly from FIFO and publishing it on an E-link. I used C++, HTML, JavaScript, AJAX, Highcharts and felix-star software commands to access FELIX data and to visualize it on a GUI running on a webserver. This data includes the fid, data rate, block rate, chunk rate, buffer events, polls and interrupts.

1. Introduction

CERN (for Conseil Européen pour la Recherche Nucléaire) was established in 1952 as a European Council for Nuclear Research and it is located near Geneva on Swiss/French border. Its main area of research is particle physics because nowadays the understanding of matter goes deeper than the nucleus. CERN adopts very large number of scientists that come from different countries to unify their knowledge and skills and contribute to different experiments. Currently there is 23 CERN member states, 3 Associate Member States in the pre-stage to Membership and 7 Associate Member States [1]. The most of the experiments are using LHC (The Large Hadron Collider) which is the world's largest and most powerful particle accelerator/collider. It has been in use since 2008 and since then it has been continuously upgraded. The LHC contains a large 27-kilometre long ring inside of which particles are accelerating at 13 TeV center-of-mass energy [2] using superconducting magnets. Locations where collisions of the accelerated particles happen are at the points of four particle detectors: ATLAS, CMS, ALICE and LHCb. These are portrayed in Figure 1.1

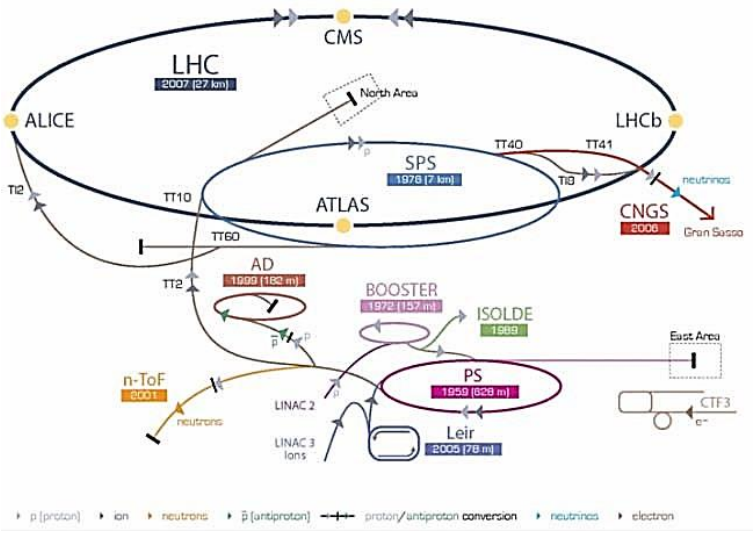

Figure 1.1. The four LHC experiments ATLAS, CMS, ALICE and LHCb are the points where beam particles collide at high energies. Image source: "Facts and figure about the LHC", URL: https://home.cern/resources/faqs/facts-and-figures-about-lhc.

2. Purpose of FELIX

ATLAS experiment is one of the four main experiments in CERN where particle collisions happen, but not all

[1] Student at the Faculty of Electrical Engineering, Computer Science and Information Technology Osijek (J. J. Strossmayer University of Osijek, Croatia)

collisions (also called events) are worth analyzing. That is the reason why a trigger and data acquisition system (TDAQ) is an important part of ATLAS experiment. It is responsible for selecting in real time only those events that could be interested and are possibly important for scientists and for further investigations. After the current LHC shutdown, collision energy and luminosity will increase in the current LHC Phase-I upgrade and ATLAS will need to provide the same physics performance once when LHC starts running again [2]. That is why currently the ATLAS experiment needs to be upgraded and a part of this upgrade is the FELIX (*Front End LInk eXchange*) system.

FELIX is a new central data distribution system that forwards data in both directions between the ATLAS detectors and data filters and processors [3]. Currently each detector has its own point-to-point link and FELIX is a substitute to it that will make maintenance and monitoring easier. It assures a central data distribution layer in the DAQ chain [3] and it interfaces custom radiation tolerant optical links from front-end electronics (named E-links) using PCIe Gen3 cards and switched Ethernet network. In this way, FELIX is implemented as a router that enables software running on commercial servers [4] and routes between custom serial links from front end ASICs and FPGAs to data collection and processing components via a commodity switched network [5]. This process is displayed in Figure 2.1.

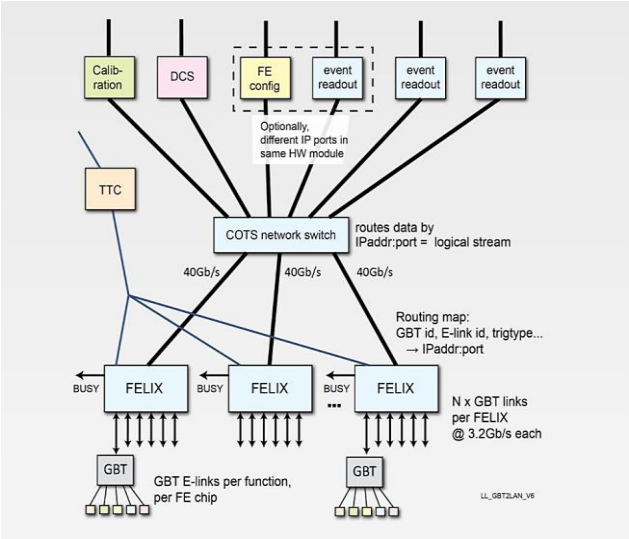

Figure 2.1. FELIX system acting as a router. Image source: "The ATLAS FELIX Project - Introduction to FELIX", URL: https://atlas-project-felix.web.cern.ch/atlas-project-felix/.

3. *felixcore* and *felix-star* software

The main part of FELIX software used to be the FELIX core application (called *felixcore*) which is a multi-threaded application that controls the FPGA cards via PCIe registers. It reconstructs the chunks of data from buffers and forwards it to the location which is allocated to receive the data from this application. NetIO, which is a network protocol-agnostic layer, is used when forwarding the data through the network [5]. Moreover, it is important to mention that *felixcore* can forward the data to the FPGAs as well.

FELIX core software is working in a way that the user interacts with it using network endpoints in order to manage receiving or sending the data from or to E-links. In this way, software systems connect with FELIX using this application. Among these software systems is a monitoring system which is responsible for monitoring of operational data and *felixcore* supports this process via a web front-end and publishing E-link information using the FELIX bus system [6]. The *felix-cor*e software is illustrated in Figure 3.1.

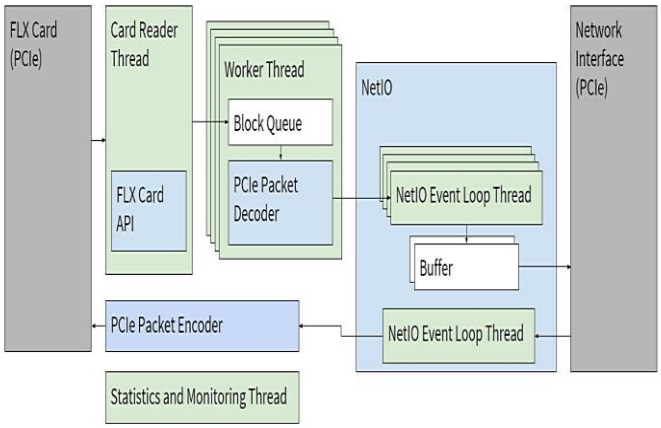

Figure 3.1. The architecture of the *felix-core* software. Image source: FELIX – User Manual.

However, it is important to mention that *felixcore* is an old application which had been used for the first FELIX tests and it is not in use anymore. An application currently used as the central process of a FELIX system is called *felix-star*, shown in Figure 3.2., a single-threaded, eventloop-based software, which enables RMDA-based asynchronous communication. Also, it is enhanced with new data types, a new felix-bus system, etc. The actual communication processes are implemented in the NetIO-

next library. For now, *felix-star* can output monitoring information formatted in JSON in local UNIX FIFOs and publish the same information over e-links [6].

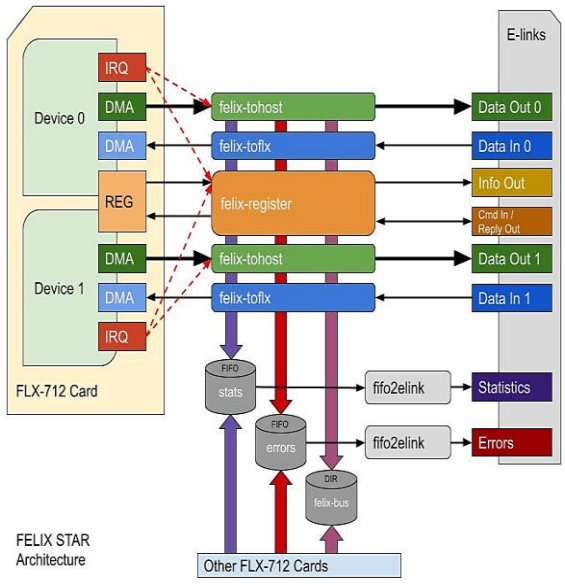

Figure 3.2. The architecture of the *felix-star* software. Image source: FELIX – User Manual.

4. Development of *felix-monitoring* tools

My work on CERN Summer Student Programme consisted of developing *the felix-monitoring* tools which are responsible for visualizing the statistics data that is published on an E-link by *felix-star*. The first step in the development of this tool was to use *felix-file2host* tool that enables reading data from a file and emulating it to the *felix-tohost* software. That was a necessary step because the whole process of developing was held on an lxplus virtual machine without a *felix card*. The data given by starting this command is in JSON format and it has been published in FIFO files that were previously created for the statistics and error data. Because it is in JSON format, it was necessary to make a function that will convert it from this format into regular data types.

The second step in development was use of *felix-fifo2elink* command. General options of this command are portrayed in Table 1. It is able to read the statistics and error FIFO files from the previous step and publish the information as an E-link. This command can be used following these guidelines:

#source felix-fifo2elink [options] <fifo> <hostname_or_ip> <port> <elink> (<did> <cid>)...

Table 1: General options

Option	Description
-fifo	FIFO (JSON) to read from.
-hostname_or_ip	Hostname or IP to publish from.
-port	Port number to use.
-elink	Elink to use for FID.
-did	DetectorID to use for FID.
-cid	ConnectorID to use for FID.

A user can access this application by accessing the *felix-star* software and creating two FIFO files for statistics and errors in LINUX terminals. After this, the second step is running the *felix-file2host* and *felix-fifo2elink* commands in separate terminals. The last step is running *felix-monitoring* tools. By default the web front-end is available on port 8080. To access the web front-end there should be used a web browser which points to *http://hostname:8080*. The whole process of the project's development is visualized in Figure 4.1.

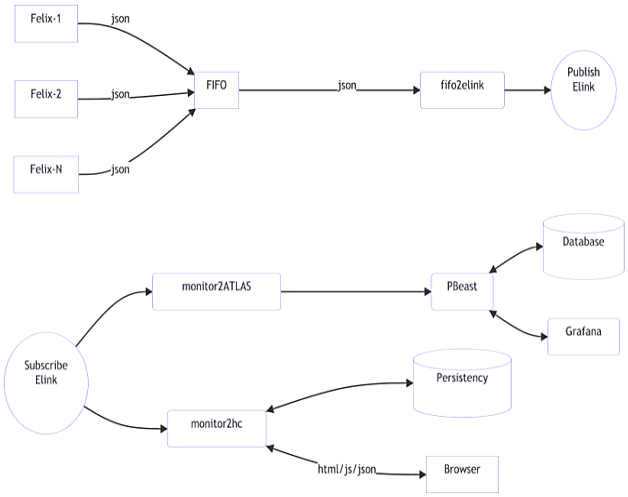

Figure 4.1. The development process of multiple monitoring tools. The felix-monitoring-tool chain is represented in the top part of the diagram and the bottom part from "Subscribe Elink" to "Browser".

The project consists of multiple project files that are indicated in Figure 4.2. In the *main* function, in the file *main_fmclient*, two FIFO files for statistics and errors are made and then the *start_web_server* function is called, which is responsible for starting own web server from *web_server.cpp* file. JSON file produced in FIFO when calling the *read_fifo* function is sent to the *read_data* function that converts the input to the *string* values. This function is defined in *functions.cpp* file and her output are fields of strings. Each field is dedicated to one of the parameters that are shown on charts and in the table in *felix-monitoring* web app. Later these fields are input for other functions called in *web_server.cpp* file, where the output of these functions are JSON fields. In *index.html* is defined the layout of webpage and it is connected with *test.js* file, which is using AJAX technology to get the JSON content coming from *web_server.cpp* file. This project's program allows using of numerous processes due to usage of threads.

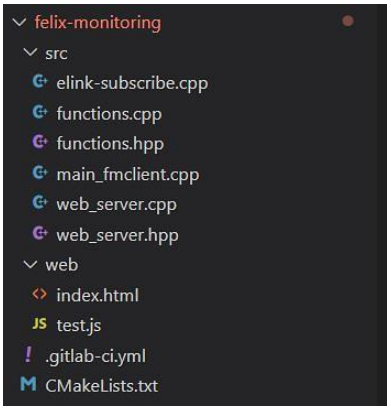

Figure 4.2. The *felix-monitoring* project files.

In the development process, C++, HTML, JavaScript, JSON, AJAX, Bootstrap, jQuery and Highcharts were used to set up my own web server, monitoring and giving status information via a web front-end. Web server is handling all HTTP requests which are constantly refreshing every 5 seconds if a button "Start Time Interval" is clicked so that a user can access to the current data on charts and in the table. The charts have parameter values on their y-axis and real-time values on their x-axis. When clicked on the charts, app displays the parameter value from the current point and local date and time. The values that are used for making charts are fid, data rate, block rate, chunk rate, buffer events, polls and interrupts. The fid value is presented in a table due to its almost constant permanence. The final result is shown in Figure 4.3.

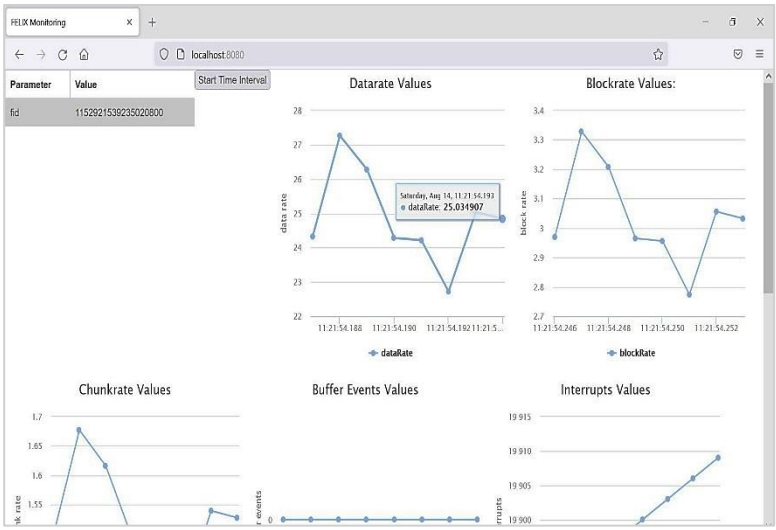

Figure 4.3. The *felix-monitoring* web app.

5. Conclusion

Until now, FELIX core *felixcore* application was used when accessing the data output parameters such as free memory, E-links from host, queue size, block_rate, chunk_rate etc. Testing it has shown that this application uses the data and old data types and then calculates its statistics. It is complicated process and sometimes not necessary.

Therefore, *felix-monitoring* has been proposed as a new solution. It is a lightweight, simplified application within *felix-star* software and provides a friendly, homogeneous interface that shows the data which is coming directly from FIFO. It means that this application does not have a need for calculating the data and its statistics, it uses properly data types and is an improved solution for the Phase I LHC upgrade. With this interface, users of *felix-monitoring* can easy access all the data they need for their own projects and in that way increase their effectiveness.

Acknowledgments

I want to thank CERN for giving me this great opportunity to be a part of this year's Summer Student Programme and to have an insight in all of the CERN resources. It included working with people from all over the world on an interesting and challenging project, attending the lectures and workshops on the most incredible science breakthroughs and visiting fascinating experiments. I am especially thankful to supervisors Felix Zahn, Ph.D. and Carlo Alberto Gottardo, Ph.D., for guiding me through every step of the project.

References

[1] "CERN – Our Member States", URL: https://home.cern/about/who-we-are/our-governance/member-states, (6 Aug 2021)

[2] M. Trovato, on behalf of the ATLAS TDAQ Collaboration: "FELIX: The New Readout System for the ATLAS Detector," 2019 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2019

[3] J. Schumacher, C. Plessl, W. Wandelli.: "Interfacing Detectors and Collecting Data for Large-Scale Experiments in High Energy Physics Using COTS Technology", presented 09 Jun 2017, 2017, URL: https://cds.cern.ch/record/2268506, (23 Jul 2021)

[4] "The ATLAS FELIX Project - Introduction to FELIX", URL: https://atlas-project-felix.web.cern.ch/atlas-project-felix/, (24 Jul 2021)

[5] N. Ilic, J. Vermeulen, S. Kolos: "FELIX: the new detector interface for the ATLAS experiment", 23rd International Conference on Computing in High Energy and Nuclear Physics (CHEP 2018), 2019, URL: https://www.epj-conferences.org/articles/epjconf/abs/2019/19/epjconf_chep2018_01023/epjconf_chep2018_01023.html, (24 Jul 2021)

[6] FELIX – User Manual