EPJ Web of Conferences **245**, 01016 (2020) https://doi.org/10.1051/epjconf/202024501016
CHEP 2019

Integration of custom DAQ Electronics in a SCADA Framework

Luís Granado Cardoso[1,*], *Clara* Gaspar[1], *João* Viana Barbosa[1], *Federico* Alessio[1], *Beat* Jost[1], *Niko* Neufeld[1], *Markus* Frank[1], *Rainer* Schwemmer[1], and *Paolo* Durante[1]

[1]CERN, Geneva, Switzerland

Abstract. LHCb is one of the 4 experiments at the LHC accelerator at CERN. During the upgrade phase of the experiment, several new electronic boards and Front End chips that perform the data acquisition for the experiment will be added by the different sub-detectors.
These new devices will be controlled and monitored via a system composed of GigaBit Transceiver (GBT) chips that manage the bi-directional slow control traffic to the Slow Control Adapter(s) (SCA) chips. The SCA chips provide multiple field buses to interface the new electronics devices (I2C, GPIO, etc). These devices will need to be integrated in the Experiment Control System (ECS) that drives LHCb.
A set of tools was developed that provide an easy integration of the control and monitoring of the devices in the ECS. A server (GbtServ) provides the low level communication layer with the devices via the several user buses in the GBT-SCA chip and exposes an interface for control to the experiment SCADA (WinCC OA), the fwGbt component provides the interface between the SCADA and the GbtServ and the fwHw component, a tool that allows the abstraction of the devices models into the ECS. Using the graphical User Interfaces or XML files describing the structure and registers of the devices it creates the necessary model of the hardware as a data structure in the SCADA. It allows then the control and monitoring of the defined registers using their name, without the need to know the details of the hardware behind. The fwHw tool also provides the facility of defining and applying recipes - named sets of configurations - which can be used to easily configure the hardware according to specific needs.

1 Introduction

The LHCb experiment is one of the detectors collecting data at the LHC accelerator at CERN. It is specialized in b-physics and is composed of several sub-detectors and subsystems. All of these subsystems have specialized custom electronic boards and devices to acquire the data from the events produced by the collisions of the LHC beams in the detector. During the upgrade phase, many of these components will be replaced, to be capable of handling the increased luminosity delivered by the accelerator. The control and monitoring of the new subdetectors electronics will be achieved with the usage of a radiation hard chipset - the GigaBit Transceiver (GBT) and the Slow Control Adapter (SCA) chips [1] - which provide the bidirectional optical links and various user-configurable interfaces in order to meet the

*e-mail: luis.granado@cern.ch

requirements of the different front-end ASICs of the subdetectors. In order to easily and reliably control and configure these devices they need to be integrated in the SCADA System (WinCC OA) [2] so they can be driven by the Experiment Control System (ECS). Due to the custom nature of these devices, this requires the implementation of a framework that is capable of handling the communication with the front-end devices via all the available field-buses, providing a link to the ECS and abstracting the description of the hardware and model it into the data structures used by WinCC OA.

2 Architecture

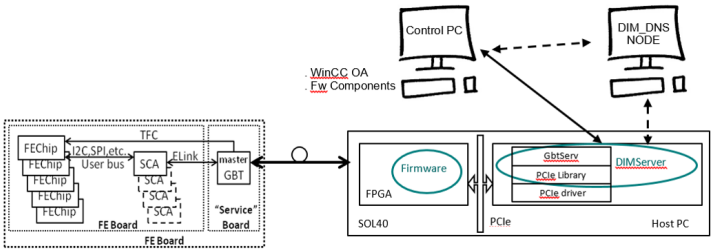

Figure 1. Architecture of the control system for the front-end electronics boards

Fig. 1 is a depiction of the typical architecture of the control system for the Front End electronics devices of the experiment. The Front End Boards (FEBs) include one or more SCA chips, which provide the communication interfaces to the devices of the board via the several available field-buses (I2C, SPI, etc.); the SCA chip(s) are connected via Elinks to a master GBT, which is connected through optical links to a board that contains an FPGA, connected to a remote PC. This PC is then connected to (or is itself) a control PC that is part of the distributed ECS. On the software side, the framework for the control system is composed of 3 main components:

- A GBT Server (GbtServ) – runs on the host PC that holds the FPGA board
- A GBT Client (fwGbt) – runs on the controls PC
- A hardware abstraction tool (fwHw) – installed on the controls PC

All the LHC experiments at CERN use as their SCADA system the software WinCC OA. In order to reduce duplication of development, the JCOP (Joint Controls Project) [3] was created in order to provide common controls solutions for the four experiments. JCOP provides a framework for the creation of JCOP components – WinCC OA packages containing all the required user panels, libraries, scripts and other software – that can be easily installed and distributed. The 2 latter components of the control system software for the FE electronic devices (fwGbt and fwHw) are developed as Framework components.

3 The GBT Server

The Gbt server (GbtServ) implements the lower level communication with the FPGA board devices and firmware, the master GBT, and the SCA chips. It also implements the connection

to the Control System, acting as the bridge between the Control System and the hardware devices and firmware.

For the local FPGA board, the GbtServ can control and monitor devices using the I2C and SPI protocols, and can communicate with the FPGA firmware using the local bus (typically PCIe).

For the remote Front End devices, which are connected via the GBT-SCA chipset, the GbtServ provides control and monitoring through the multiple field-buses supported by the GBT-SCA (I^2C, GPIO, SPI, ADC, DAC, JTAG).

This server is based on DIM (Distributed Information Management system) [4], which can easily be interfaced from WinCC OA.

DIM is a communication system for distributed / mixed environments and it provides a network transparent inter-process communication layer. It is based on the server/client paradigm and it uses the concept of publishing/subscribing services and commands.

The GbtServ provides the connection to the control system as it implements services and commands via DIM for all the available field-buses, both to the local FPGA board and remote GBT-SCA chips, which are then exposed to the GBT client (fwGbt) in the Control System.

The Control System abstraction tool (fwHw) can also send a description of the connected hardware and the GbtServ exposes DIM services and commands based on these descriptions.

4 The fwGbt component

The way the GBT interface is integrated into WinCC OA requires the installation of a specific framework component, a GBT client, which provides the basic communication between the SCADA and the GbtServ running on the interface PC.

The fwGbt is developed as a JCOP Component and connects to the GbtServ by subscribing to the DIM services and commands published by the server.

For each of the existing protocols it implements a minimum of 3 types of operation – read, write, write-read – and enables the communication with the electronics at a low level. To each type of protocol correspond 2 DIM services (one for readings status and other for writings status) and a DIM command (for operations) subscription from the GbtServ. The fwGbt also allows for an easy debugging of all the communications by providing an easy to use User Interface for tests (Fig. 2). This UI allows access to all the implemented protocols and their different settings from a central place. It can be used to verify what the status of the connection with the GbtServ is and how the hardware reacts and replies to different commands.

This component provides the lower-level functions to access the different types of registers, which are used by the abstraction tool (fwHw).

5 The fwHw component

The framework Hardware tool (fwHw) [5] is developed as a JCOP component, allowing for easy installation by the subdetectors. The tool provides a user interface, which allows for the easy modelling and configuration of the existing hardware devices and can also serve as a debugging tool. However, its purpose is mostly setting up the system and the different subdetectors or subsystems should then implement their hierarchical control Finite State Machine (FSM) trees and their own user interfaces more suitably geared to the operation of their electronics.

Figure 2. fwGbt GBT Client User Interface

5.1 Hardware abstraction

The fwHw tool models the hardware into WinCC OA datapoint structures. The WinCC OA data structure is a treelike structure, where a logical model is defined in Datapoint Types, which can be instantiated in datapoints that share this logical model. Similarly the fwHw tool produces the logical model of the electronic devices as Datapoint Types, but providing an interface more adapted to the modelling of electronic devices. In the fwHw tool, the data is hierarchical organized with the following types:

- Registers - The registers are the representation of a register on a real hardware device. A register can be of any size and implement any type of protocol. The hardware tool allows the configuration of each register according to the settings necessary to access the hardware via the different protocols, the size of the register and the type of readout it will require. The registers can also represent a local variable, i.e. a logical data item of a given area (e.g. a string which holds a filename to configure a given area). These are the leaves or the lower level nodes of a hardware device model tree.
- Areas/Sub-Areas - Areas are logical groups of registers and/or other areas, which represent a logical division of a part of the hardware device (e.g. a chip or a group of chips). Defined areas can be declared as sub-areas of other areas and can be used multiple times. For example, in Fig. 3 we can see that both areas 2 and 3 have the same type and thus have the same structure. The defined areas can also be used by different Device Types, so if some devices share a common structure, there is no need to duplicate similar structures.
- Device Types - The topmost node of the description of the hardware is the Device Type. The device type is the definition of a device (e.g. a board) with the areas and registers that compose it and the definition of the interface to the hardware they will have.

After the model of the hardware device is defined, we can create as many instances of that model as required and configure the particular settings for those devices. The created models hide the complexity of the communication with the hardware. Settings like the specific protocol, addresses, sub-addresses and all others necessary to access the hardware registers are

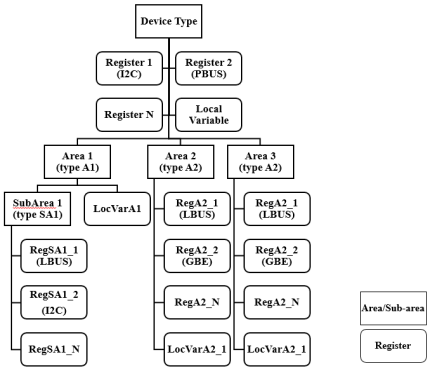

Figure 3. Example of a model of a device

included in the model and this allows the access of the hardware registers simply by using the name defined in the model.

5.2 Hardware model creation

There are three ways to create the model of an electronics device using the fwHw tool:

- via the User Interface Panels: The fwFw tool provides a user interface, which can be used to create the models of the electronics devices to be configured. It is intuitive and easy to use, however it can be cumbersome to create a device model with many registers/sub-areas, as the registers need to be created one by one and the sub-areas will also need to be included one by one into other areas. Another disadvantage is that it is not particularly easy to update a given model if the changes required are somewhat extensive.
- via scripts: The tool also provides the functionality of using a script to create the hardware models; this improves the creation of big hardware models as well as improves the update or modifications to those models. It can also be used to automate the creation of devices quite easily. It requires writing of scripts using the provided library functions, which requires a learning curve and the scripts may be quite long.
- via XML description files: There is also the possibility of creating the hardware models by describing them in XML files. These XML files will then be used to generate the scripts, which will create the hardware models in the fwHw tool. The usage of an XML hardware description file has the advantage of being easily readable, being validated against an XML schema file and being easily changed or updated. It is also foreseen in the future to have these XML files generated automatically from other descriptions (e.g. an FPGA firmware file).

  The implementation of the configuration via XML description files allows also for the easy export of the currently defined models of hardware in a project to be adapted by different sub-detectors or sub-systems.

5.3 Hardware interface

The hardware devices for the upgraded experiment will be controlled via the GBT-SCA chipset. The connection between the electronic devices and the Control System is achieved,

at a lower level by the GbtServ and the fwGbt tool. These provide the hardware interface to the fwHw tool.

After an electronic device is modelled in the fwHw tool, devices of this type can be instantiated. Specific settings need then to be set, in order to configure the GbtServer connection through which the device will be connected.

Once the devices are instantiated in the fwHw tool, it is needed to make available to the GbtServ the names of the declared registers of the model. The configuration is automatically passed to the GbtServ at startup, and the server is then aware of all the existing registers of the hardware as well as all the settings necessary to access them, their size and the desired readout mode.

The control PC is connected to the host PC where GbtServ is running via the network. The GbtServ is aware of the modelled devices, the necessary settings to access them, and the required readout modes (poll, update on change, etc).

5.4 Configuration DB and Recipes

One of the components provided by JCOP is the fwConfigurationDB [6]. This component provides a way to store and apply different sets of data to WinCC OA datapoints. It introduces the concept of recipes - named configurations for a given device or set of datapoints. These configurations can be both static configurations in order to setup equipment (e.g. configuring a spare device that is replacing one that failed) as well as dynamic configurations (e.g. configure the devices for different LHC modes of operation).

The fwHw Tool provides an interface for the configuration of recipes for the defined hardware models and this allows for the easy configuration of the devices according to specific needs. For instance it is possible to set specific values to the registers of the devices when the LHC provides physics conditions for data taking.

The created recipes can be both cached locally in the systems where they will be used and also stored on an Oracle database.

6 Integrated Solution

A final integrated solution will look like the schema in Fig. 4. Here we can see an example of a fully configured and integrated system, where all the components are running:

- GbtServ is providing the lower level communication layer with the hardware;
- fwGbt is configured and it's User Interface can be used as Test UI to debug the communication from the Control System to the hardware;
- The model of the hardware devices has been abstracted with the fwHw tool, and the existing devices have been instantiated;
- Custom User Interfaces for configuration and monitoring of the modeled devices have been easily developed and integrated in the global ECS Operation UI.

7 Conclusion

The developed framework provides a complete, easy and reliable solution to integrate custom electronics devices into the LHCb Experiment Control System. Even though the existing electronics devices in LHCb are of varied types, the developed tools provide a way to easily address all the devices and a way to easily abstract and create models of these devices, so they can be integrated and controlled from the global Experiment Control System. It is also

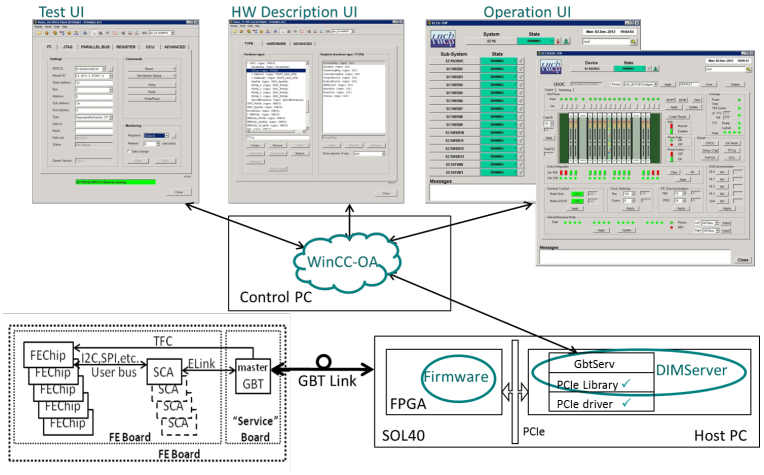

Figure 4. Schema of a complete integrated solution.

easy to modify or extend the already existing models without major impact to the existing system.

These tools are able to facilitate the integration of the subdetectors electronic devices as they provide a base for the subdetectors to integrate their devices into their ECS control trees and develop their specific user interfaces.

The usage of fwHw tool to abstract the hardware models also makes possible an easier control of the devices by providing a way to interface the registers of these devices using named registers. This hides the complexity of the communication from the user and enables the possibility of using the configuration DB and recipes to configure the electronics according to the specific needs of the experiment in different operation modes. It also provides a base for increasing the automation of the running of the experiment [7].

References

[1] F. Alessio and R. Jacobsson, JINST Topical workshop on electronics for particle physics **51**, (2012)

[2] SIMATIC WinCC Open Architecture made by ETM Professional Control GmbH, Eisenstadt, Austria: http://www.etm.at

[3] O. Holme et al., Europhysics Conf. Abstr. **29J**, WE2.1-6O (2005)

[4] C. Gaspar and M. Donszelmann, IEEE Real Time Conference, 156-158 (1993)

[5] fwHw Tool: http://lhcb-online.web.cern.ch/lhcbonline/ecs/FWHW/default.html

[6] L. Abadie et al., Europhysics Conf. Abstr. **29J**, MO4A.2 (2005)

[7] C. Gaspar and B.Franek, IEEE Transactions on Nuclear Science **53**, No.3, 974-979 (2006)