A FRAMEWORK FOR HARDWARE INTEGRATION IN THE LHCB EXPERIMENT CONTROL SYSTEM

L. Granado Cardoso, C. Gaspar, R.Schwemmer, J. Barbosa, F. Alessio, CERN, Geneva, Switzerland
P-Y. Duval, CCPM, Marseille, France

Abstract

LHCb is one of the 4 experiments at the LHC accelerator at CERN. During the upgrade phase of the experiment (planned for 2018), 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 devices will need to be integrated in the Experiment Control System (ECS) that drives LHCb. Typically, they are controlled via a server running on a PC which allows the communication between the hardware and the experiment's SCADA (WinCC OA). A set of tools was developed that provide an easy integration of the control and monitoring of the devices in the ECS. The fwHw is a tool that allows the abstraction of the device's models into the ECS. Using 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.

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. In order to easily and reliably control and configure these devices they need to be integrated in the SCADA System (WinCC OA) [1] so they can be driven by the Experiment Control System (ECS). Due to the custom nature of these devices, their description varies greatly between subsystems, so, to effectively integrate these devices in WinCC OA there was the need to create a tool that could abstract the description of the hardware, model it into the data structures used by WinCC OA and provide a layer to interface with this hardware.

THE FWHW TOOL

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) [2] 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 framework Hardware tool (fwHw) [3] is developed as a JCOP component, allowing for easy installation by the sub-detectors. 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 sub-detectors 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.

Hardware Abstraction

The fwHw tool models the hardware into WinCC OA datapoint structures. The WinCC OA data structure is a tree-like 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 Figure 1 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

ISBN 978-3-95450-148-9

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.

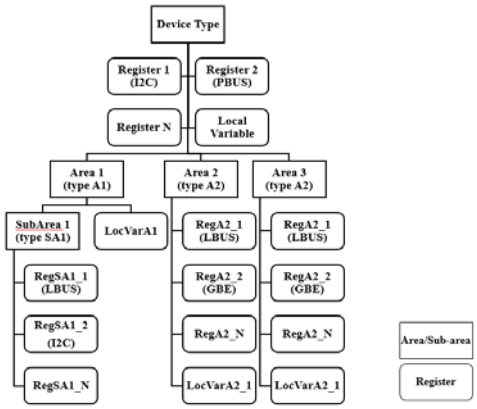

Figure 1: Example of a model of a device.

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

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.

Hardware Interface

The electronics devices in LHCb are controlled using specific interfaces. Currently two interfaces are used according to the location of the devices to be controlled:

- SPECS [4] – Serial Protocol for the Experiment Control System – used in radiation areas;
- CCPC [5] – Credit Card PC – a small embedded PC running Linux – used in non-radiation area;

A third interface will be used in the future:

- GBT [6] – Giga-Bit Transceiver – a radiation hard chipset – used in the future upgrade of LHCb

The type of registers available depends on the type of interface. A summary of the available register types is shown in table 1.

Table 1: Types of Registers per Interface

CCPC	SPECS	GBT
I^2C	I^2C	I^2C
JTAG	JTAG	JTAG
LBUS	PBUS	Memory Bus
GPIO	REG	PIA
GBE	DCU	SPI
-	-	ADC
-	-	DAC

These interfaces provide the connection between the ECS and the hardware devices. They run a server that communicates with the fwHw tool; this server is based on DIM (Distributed Information Management system) [7], 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 way these interfaces are integrated into WinCC OA require also the installation of specific components for each interface, which provide the communication between the fwHw tool and the DIM server running on the interface PC. These components provide the lower-level functions to access the different types of registers, which are used by the fwHw tool. They are appropriately named

ISBN 978-3-95450-148-9

fwSpecs, fwCcpc and fwGBT for the SPECS, CCPC and GBT interfaces respectively.

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 interface (or more exactly the DIM server) with which the device will connect.

Once the devices are instantiated in the fwHw tool it is needed to make available to the DIM server the names of the declared registers of the model. The configuration is automatically passed to the DIM server 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.

In Figure 2 we can see the fwHw tool user interface, showing declared hardware types on the left and the registers of that hardware type on the right.

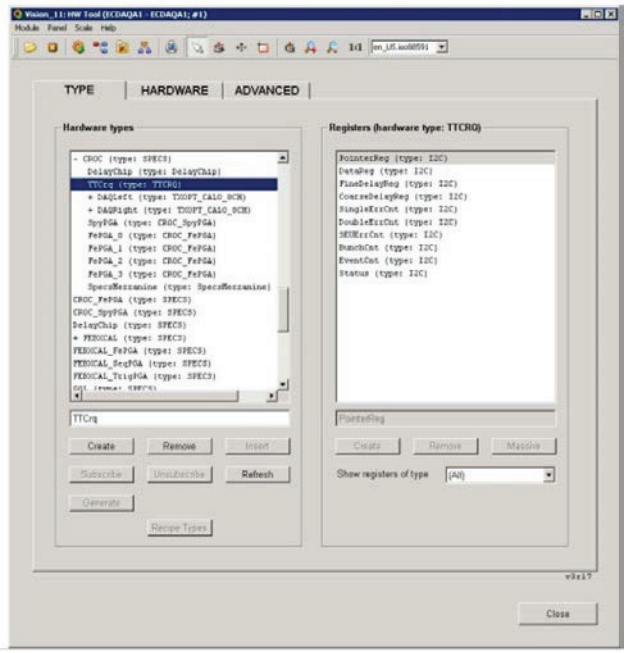

Figure 2: fwHw User Interface showing an existing model and the named registers for one device of this type.

As mentioned earlier, in the upgrade of LHCb, the new interface GBT will be used to interface the hardware both in radiation and non-radiation areas (it will link to both back-end and front-end devices). The GBT is a common CERN Serializer-Deserializer radiation-hard chipset for data, fast and slow control distribution to and from the electronics for the upgraded LHC experiments. This new interface requires the development of the new component fwGBT, which contains the lower level communication interface and the GBT DIM server.

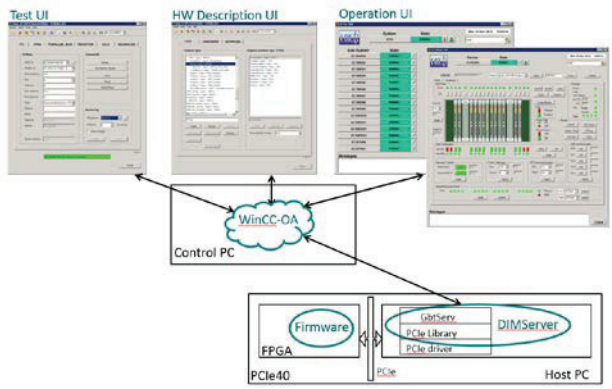

Figure 3: Architecture of the framework solution for the GBT Interface.

In Figure 3, we can see the schema of the architecture of the whole solution for the GBT Interface (the SPECS and CCPC interfaces share the same architecture), with a control PC running WinCC OA, which has the framework components installed (fwHw and fwGbt) and where the hardware is modelled and available for the FSM Control Tree and User Interfaces. The control PC is connected to the host PC where the DIM Server is running via the network. The DIM Server is aware of the modelled devices, the necessary settings to access them, and the required readout modes (poll, update on change, etc). It integrates the lower level libraries to communicate with the devices via the several available protocols.

Configuration DB and Recipes

One of the components provided by JCOP is the fwConfigurationDB [8]. 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.

CONCLUSION

The fwHw tool provides an 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, this tool provides a way to easily abstract and create models of these devices and control them from the global

ISBN 978-3-95450-148-9

Experiment Control System. It is also easy to modify the already existing models without major impact to the existing system.

The tool is able to facilitate the integration of the sub-detectors electronic devices as it provides a base for the sub-detectors to integrate their devices into their ECS control trees and develop their specific user interfaces (Figure 4), communicating with the hardware using the functions provided by the fwHw tool libraries.

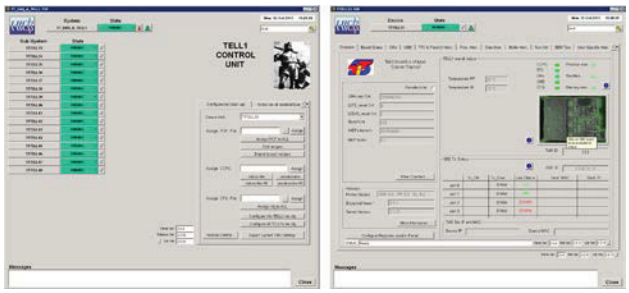

Figure 4: Control Tree and device panel for one of the LHCb sub-detectors.

This tool also makes possible to easily control 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 [9].

In the future upgrade of the LHCb, new interfaces will appear to replace or work along the existing ones. Due to the generic nature of the fwHw tool, these new interfaces can be easily integrated.

REFERENCES

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

[2] O. Holme et al., "The JCOP Framework", ICALEPCS 2005, Geneva, Switzerland.

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

[4] SPECS website: https://lhcb.lal.in2p3.fr/Specs/

[5] CCPC website: https://lhcb-online.web.cern.ch/lhcb-online/ecs/ccpc/default.htm

[6] F. Alessio and R. Jacobsson, "A new readout control system for the LHCb upgrade at CERN", JInst Topical workshop on electronics for particle phusics, 2012, Oxford, United Kingdom.

[7] C. Gaspar, "DIM - A Distributed Information Management System for the Delphi experiment at CERN", IEEE Real Time Conference, 1993, Vancouver, Canada.

[8] JCOP Framework Configuration DB website: https://wikis.web.cern.ch/wikis/display/EN/JCOP+Framework+Configuration+Database

[9] C. Gaspar and B. Franek, "Tools for the automation of large distributed control systems", IEEE Transactions on Nuclear Science., Vol. 53, NO. 3, June 2006.

ISBN 978-3-95450-148-9