Introduction

This paper addresses aspects of implementing a control system for electronics boards in order to perform remote FPGA programming, register control, and monitoring, as well as interfacing it to an expert system. Based on the ideas presented a run control has been built for the Timing and Fast Control (TFC) system of the LHCb experiment at CERN.

Given that a set of electronics boards are controlled from a common type of control interface which has access to set of board control buses, the aim here is to:

- Define a generic data structure which reflects the control configuration of a board and on which the remote control system can perform control actions, either directed from a user interface or from an expert system, and verify the integrity of the register control.
- Provide a simple and economical remote access mechanism to any board resource type independent of the bus type.
- Provide a simple and economical mechanism which allows monitoring counter and status information in the electronics boards by data register subscription.

Device Description

Each electronics board type is described by a PVSS data point type with an overall structure common to all board types. Each individual board is stored as an instantiation of the corresponding data point type. The data point reflects the entire state of all the controllable resources on the board.

- BoardIdentifier: Board identifier of the particular board.
- FPGAcode: Structure with the file pointers to the FPGA programming files.
- State: Structure used to interface the control system to a supervisory expert system for automated control.
- Register Settings and Register Readings: To verify register write actions automatically in PVSS, the requested and the read values are stored.
- Parameter Settings and Parameter Readings: Since there is not always a one-to-one relation between physical registers and the functional parameters, the data point also stores the values of the parameters. The physical parameters are divided into logical groups according to the board function to which they belong. These are the values acted upon and displayed in the user interfaces.
- Actions: Set of common structures which are associated with the DIM commands and services.

Currently PVSS script functions perform the configuration actions and other global actions, and make the automatic translation between physical registers and functional parameters and vice-versa.

Examples of Graphics User Interfaces

Application on the LHCb Timing Fast Control System

A complete local run control system with user interfaces and low level control of the electronics boards has been implemented for the LHCb Timing and Fast Control system. The TFC system consists of some 75 boards of five different types. The most complex board has over 250 functional parameters in 150 physical registers. The control system has been in use during the entire construction of the LHCb detector and is now in use for the testing and commissioning of the detector Front-End electronics.

Control System Architecture

The remote control system is based on the industrial distributed SCADA system PVSS II from B&M. The electronics boards carry a commercial Credit Card PC (CCPC) with Ethernet from Digital Logic, AG, Switzerland.

Using the Distributed Information Management (DIM) package, a generic DIM server has been implemented on the CCPC which performs the low-level control of all resources on each electronics board. The DIM server is linked with functions to operate data transactions via a parallel 32-bit Local Bus, I2C, JTAG and an 8-bit General Purpose I/O interface. The DIM server publishes on the control network a list of commands and services through which all the board resources are accessed from PVSS:

FPGA programming:
- DownloadFPGA(struct{data, id}) - Command to program FPGAs. The data consists of the identifier of the FPGA and the programming data in the form of STAPL byte-code.
- FPGAloadStatus(struct{status}) - Service to inform the control system about the status of the programming.

Register control:
- ReadWriteRegister(struct{struct{method, address, data, mask, write}}) - Command to read or write registers from PVSS. It has a dynamic length meaning that any number and types of data registers may be accessed in a single action. The parameter method indicates the access method for each register which in the current implementation is Local Bus (0), I2C (1) or GPIO (2). The address is the full physical address, the mask allows modifying only specific bit fields of a wider register, and the parameter write indicates if the action is a write (1) or only a read action (0).

Data subscription:
- SubscribeRegister(struct{struct{address, interval}}) - Command to request the DIM server to update counter or status registers with a specified interval. Data consists of a list of register addresses and the update interval.
- UpdateSubscribeRegister(struct{struct{address, data}}) - Service to update an array of counter or status registers in PVSS which are more subscribed to. The DIM server will update this service according to the specified interval.

Parameter - Register Translation

Currently the register-parameter translation and the global control actions are hard coded in PVSS scripts. Although it doesn’t have a prohibitive effect on the current implementation, the performance of the PVSS script is inherently rather low. Secondly, the fact that the translation between registers and parameters is hard coded implies that there is a lack of generality.

An improvement is underway in which the data point contains the mapping between physical registers and functional parameters in a structure of descriptors. For each parameter the descriptor contains the name of the parameter, the address of the register in which it is located, and the position and size of the bit field. This makes it possible for a generic function to read the descriptors at start up and set up the translation runtime. In order to improve on the performance the aim is to implement this as a PVSS API manager.