CONCEPT AND ARCHITECTURE OF THE RHIC LLRF UPGRADE PLATFORM*

K. S. Smith[#], T. Hayes, F. Severino, BNL, Upton, NY 11973, U.S.A.

Abstract

The goal of the RHIC LLRF upgrade has been the development of a stand alone, generic, high performance, modular LLRF control platform, which can be configured to replace existing systems and serve as a common platform for all new RF systems. The platform is also designed to integrate seamlessly into a distributed network based controls infrastructure, be easy to deploy, and to be useful in a variety of digital signal processing and data acquisition roles. Reuse of hardware, software and firmware has been emphasized to minimize development effort and maximize commonality of system components. System interconnection, synchronization and scaling are facilitated by a deterministic, high speed serial timing and data link, while standard intra and inter chassis communications utilize high speed, non-deterministic protocol based serial links. System hardware configuration is modular and flexible, based on a combination of a main carrier board which can host up to six custom or commercial daughter modules as required to implement desired functionality. This paper will provide an overview of the platform concept, architecture, features and benefits.

INTRODUCTION AND MOTIVATION

Embedded processors and Field Programmable Gate Arrays (FPGAs) have become ubiquitous features of modern digital control and signal processing systems. Development of the RHIC LLRF Upgrade Platform has focused on leveraging these technologies with the aim of creating a modular platform architecture which is suitable for deployment as an upgrade to the existing RF systems at the Collider-Accelerator Department (C-AD), as well as for new systems under development.

An overarching motivation for the upgrade has always been the need to accommodate very limited personnel resources. When upgrade work began in 2005, the C-AD LLRF group consisted of three engineers, and that has since expanded to six. On and off through the period there has also been one technician dedicated to LLRF. The group is responsible for operation, maintenance and development of LLRF systems across the C-AD complex (RHIC, AGS, Booster, and EBIS) and for development of LLRF systems for all new projects. With the resources available, application specific solutions for each system were simply not practical.

It was recognized that platform development would present a significant array of engineering challenges. Consideration was given to Commercial Off The Shelf (COTS) solutions. Based on prior experiences employing COTS hardware at both RHIC and SNS Ring LLRF systems, it was felt that the limitations and compromises inherent to such solutions outweighed the benefits nominally associated with an off the shelf solution.

SYSTEM CONCEPT

With the decision taken to develop a custom solution in-house, the overall goals for the system were defined. The desire to develop a system which was generic and modular led to the fundamental concept that underpins the entire system – that it should comprise a platform. In general terms, a platform can be defined as a base of technologies upon which other technologies or systems are built. In the context of our objective, the term platform refers to a combination of hardware, software and firmware which can be combined and configured to create digital control and signal processing systems suitable to a variety of applications.

SYSTEM ARCHITECTURE

The overall system architecture derives from the approach taken to implementing the general platform concept. Different parts of the architecture were developed relatively independently from one another, addressing specific system goals. The final architecture was incrementally developed from these components, and in fact remains under development. As will be evident, the interdependence between hardware and software is a pervasive attribute of this architecture, a characteristic shared by most modern digital systems. The remainder of this paper will attempt to provide a top down description of the platform architecture, starting with the physical hardware, working down through the embedded system at its core, the development environment and finally discussing system integration, synchronization and scaling.

Hardware Platform

The hardware platform [1] provides an example of the modularity and design reuse concepts employed throughout the architecture. Physical platform hardware comprises carrier and daughter modules, typically residing in a 3U 19” rack-mount chassis. The platform is designed to be stand-alone, not requiring any separate crates or supporting backplane technologies. Although a chassis can operate as a functional system with nothing more than a line cord attached, external connections for signal IO, Ethernet, custom high speed serial links, reference clocks, etc. are naturally required to build more complex systems integrating multiple platform chassis.

*Work supported by Brookhaven Science Associates, LLC under Contract No. DE-AC02-98CH10886 with the U.S. Department of Energy

[#]ksmith@bnl.gov

Daughter modules provide the application specific functionality for a particular system implementation. Two modules have been designed so far: A 4-channel RF ADC [2] and a 4-channel RF DAC [3]. Daughter modules share a common FPGA based back end hardware and software architecture. This supports customizable software and firmware defined functions for the module, interfacing to the carrier and connection to standard board peripherals (DDR2 SO-DIMMs and FLASH memory, etc.). This commonality also helps ensure that developing new modules requires redesign of only the front end, with reuse of the back end resulting in reduced design times and increased likelihood of first turn success.

The carrier forms the base of the hardware and software platforms. The software platform will be discussed in the following section. Up to six daughter modules can be installed in any combination on a carrier. The carrier shares all of the back end peripherals of the daughters, and adds all the infrastructure required to support internal platform functionality and external interfacing - high speed serial interfacing to and between daughter modules, Ethernet and other standard control system interfaces, high speed serial fiber interfaces for chassis to chassis communication, low noise reference clock distribution, signal filtering and buffering, system health monitoring, power distribution, cooling, etc.

Together, the carrier and daughter modularity provides for a very flexible physical configuration of the platform which can be readily adapted to a variety of applications and requirements. This configuration or definition of the hardware functionality is provided by the software platform.

Software Platform

The software platform [4] is based upon the embedded processors available in the FPGAs on both the carrier and the daughter modules. At a minimum, all FPGAs contain a single embedded PPC440 processor, but also allow instantiation of soft processor core implementations for added flexibility and functionality.

Carrier FECs

On the carrier, the role of the PPC440 processor is to support interfacing to the RHIC control system as a Front End Computer (FEC) [5]. The control system is based on a series of FECs connected to a dedicated controls network. FECs provide a standardized interface between the controls network clients (console applications, data loggers, displays, etc.) and the accelerator system hardware (RF, Instrumentation, Power Supplies, etc.). They also provide for network caching of hardware configuration parameters, allowing for automated restoration of operation following reboots, resets or power cycles. The FEC implementation on the carrier is completely transparent to the control system, using a nearly identical VxWorks kernel implementation. LLRF platform FECs are indistinguishable from any other standard FEC in the control system, allowing absolutely seamless integration of the LLRF platform into the existing controls infrastructure.

Because the FEC is integrated in the carrier FPGA, standard Control System data and timing links receivers are also integrated into the platform. In traditional RHIC VME based systems, the RHIC Event Link [6] and Real-Time Data Link (RTDL) [7] require VME decoder modules to connect them with both the FEC and the other VME hardware. Custom IP cores have been developed to integrate these functions, allowing both carrier and daughter modules to directly decode RHIC timing and data links with no need for additional external hardware.

FEC access to the carrier FPGA configuration FLASH enables remote reconfiguration the carrier via Ethernet. At the console level, this is accomplished by selecting a configuration file on the network file server, and pushing a download button to proceed. This is an extremely important capability when systems are deployed in the field. The same capability is extended to the daughters modules.

Carrier – Daughter Communication

Communication between the carrier FEC and daughter modules is accomplished using point to point 2.5 Gbps serial links based on the Xilinx Aurora protocol [8]. The PPC440 FEC executes a library of driver functions which provide unified memory mapped access to daughter module resources via these links. Stand-alone code in an FPGA soft processor core (Xilinx μBlaze [8]) manages the transaction in the daughter module. Various types of access are supported, and new ones are readily developed as extensions to the driver library when needed. An example follows:.

Daughter FPGA IP (e.g. a digital RF synthesizer implementation) employs a standardized bus interface to the μBlaze processor core. In typical implementations, this interface exposes configuration registers, data buffers, etc. to the carrier FEC via the unified memory map. Adding new instances of existing IP is accomplished by simply connecting via the standard bus interface and updating a hardware definition file used by the FEC. If completely new IP is added, the above process is unchanged, but any requisite driver module must of course be added to the FEC environment.

Platform Development

With the basic architecture of the hardware and software platforms established, a description of platform development is in order. To create a functioning system, one needs to define the FPGA internal system architecture in terms of software and firmware modules. This is accomplished using a vendor provided software tools, in this case, the Xilinx Platform Studio development environment [8].

Software and firmware development for the LLRF platform is generally compartmentalized by board. Unique projects are created for each board (Carrier, DAC Daughter Module, ADC Daughter Module and so on). Each project in turn is roughly divided into two major pieces which are tightly integrated but developed separately: 1) Standard embedded system components, and 2) Custom IP core components.

Embedded System Development

The embedded system provides all of the infrastructure mentioned in the Software Platform section. After the hardware architecture for a specific board has been defined, numerous standard as well as custom peripherals can be added as modules to the overall embedded design. Standard peripherals include DDR2 memory interfaces, Ethernet interfaces, etc., and are combined using standard peripheral interconnects. All of this occurs in a GUI environment through drag and drop type operations. Peripheral configuration settings (e.g. a base address) are adjusted as needed.

Custom IP Core Development

Custom core components are developed when no standard component is provided within the development environment to address a system specific requirement. Some are developed to support the embedded system functionality. An example would be the core used for the integrated Event Link and RTDL receiver functionality. Other custom cores are developed to implement daughter specific functionality, such as the DAC DDS core [3] and the ADC Bunch to Bucket Phase detector core [2]. Once developed however, these cores can be instantiated in the architecture as often as needed. So for example, the four channel DAC DDS module uses four identical instances of the common DAC DDS core.

When system design has been completed, the vendor implementation tools generate a bit file. This file contains the binary program which resides in the configuration FLASH memory for the target board, and is used to configure the internal programmable resources of the FPGA upon power up.

LLRF Platform Integration

The previous sections describe the LLRF platform architecture and development process required to use the platform. For small or simple systems, a single platform may well suffice. More complicated applications however require a system which can be scaled in size, tightly integrated and often synchronized.

To ensure synchronous operation of all interconnected systems, a 100 MHz ultra low phase noise master reference clock is distributed to all chassis. All required system clocks are then derived from this master clock.

The Update Link [9] is the means by which we implement our concept of system integration. This link provides deterministic timing and low latency data communication between all sub-systems. It utilizes gigabit serial transceivers in the FPGAs, using fiber optic connections between multiple chassis and LVDS connections within a chassis.

Data on the link consists of encoded packets representing both events (triggers) and actual data. Transmissions on the link are bracketed by a periodic "Update Event" code broadcast which is followed by a 48-bit time-stamp. The update period was chosen to be 10 μs, or 1000 cycles of the 100 MHz master reference clock.

All sub-system carrier and daughter FPGAs employ custom IP cores to utilize this link. The receiver core in particular required very careful design to ensure repeatable and deterministic latency. This is the most critical component for assuring repeatable and deterministic synchronization of sub-systems, and the design effort has not been without its struggles [9].

The utility of this link in achieving synchronization, integration and scaling of systems cannot be overstated. In the case of the RHIC LLRF system, reset and alignment of synthesizers distributed across multiple chassis becomes trivial. Intricate manipulations of cavity phases and frequencies are easily managed using synchronous capture and broadcast of this data to and from any part of the system. System interconnections can be easily changed and restored simply by altering ID parameters for the encoded Update Link data and events.

CONCLUSION

The RHIC LLRF Upgrade Platform has been developed with the goal of providing a flexible, modular and scalable architecture which will support our current applications and satisfy new ones for the foreseeable future. The platform has been recently commissioned at both RHIC [10] and the RHIC EBIS injector [11]. To date the platform has demonstrated its versatility and utility, meeting the design goals as originally defined.

REFERENCES

[1] T. Hayes, et al., "A Hardware Overview of the RHIC LLRF Platform", these proceedings.

[2] K. Smith, et al., "A Bunch to Bucket Phase Detector for the RHIC LLRF Upgrade Platform", these proceedings.

[3] T. Hayes, et al., "A High Performance DAC/DDS Daughter Module for the RHIC LLRF Platform", these proceedings.

[4] F. Severino, et al., "Embedded System Architecture and Capabilities of the RHIC LLRF Platform", these proceedings.

[5] J. Morris, et al., "Status Report for the RHIC Control System", Proc. ICALEPCS 2001.

[6] B. R. Oerter, "Accelerator Timing at the Relativistic Heavy Ion Collider", Proc. ICALEPCS 1999.

[7] H. Hartmann, "The RHIC Real Time Data Link", Proc. IEEE PAC 1997, Vancouver.

[8] http://www.xilinx.com Accessed: 03-22-2011.

[9] T. Hayes, et al., "A Deterministic, Gigabit Serial Timing, Synchronization and Data Link for the RHIC LLRF", these proceedings.

[10] K. S. Smith, et al., "Commissioning Results from the Recently Upgraded RHIC LLRF", these proceedings.

[11] S. Yuan, et al., "Commissioning and Performance of the BNL EBIS LLRF System", these proceedings.