THE P0 FEEDBACK CONTROL SYSTEM BLURS THE LINE BETWEEN IOC AND FPGA *

N. DiMonte#
Advanced Photon Source, ANL, Argonne, IL 60439, U.S.A.

Abstract

The P0 Feedback system is a new design at the Advanced Photon Source (APS) primarily intended to stabilize a single bunch in order to operate at a higher accumulated charge. The algorithm for this project required a high-speed DSP solution for a single channel that would make adjustments on a turn-by-turn basis. A field programmable gate array (FPGA) solution was selected that not only met the requirements of the project but far exceeded them. By using a single FPGA, we were able to adjust up to 324 bunches on two separate channels with a total computational time of ~6×10^9 multiply-accumulate operations per second. The IOC is a Coldfire CPU tightly coupled to the FPGA, providing dedicated control and monitoring of the system through EPICS [1] process variables. One of the benefits of this configuration is having a four-channel scope in the FPGA that can be monitored on a continuous basis.

INTRODUCTION

The Advanced Photon Source (APS) experiences beam instabilities in both the transverse and longitudinal planes. The P0 Feedback system will be used to stabilize a selected bunch. This is accomplished by using an Altera Stratix II FPGA coupled with a Coldfire CPU running EPICS with RTEMS. This paper will discuss how the CPU and the FPGA are tightly coupled to control 648 32-tap finite-impulse filters.

FPGA COMPONENT

In selecting an FPGA for this project, consideration for the number of DSP functions that a FPGA contained and the rate that these DSP function could operate were important factors. Another factor was the development time cycle, which was set for six months. To avoid developing a complex circuit board with an FPGA containing over 1000 connections and taking into consideration the high-speed analog circuit, an Altera DSP Development Kit [2] was selected. This kit contains a high-speed Stratix II EP2S60 chip on a prototyping platform that allows development of high-performance DSP designs. Other key features of this board include two high-speed 12-bit analog-to-digital (A/D) converters, two 14-bit digital-to-analog (D/A) converters, and additional connectors to interface to other analog circuit boards. Provisions also exist to connect external I/O to the development board. In this project, an external daughterboard containing two 14-bit A/D converters was adapted to the analog connectors. One of the I/O connectors was populated with an APS event timing receiver while the other I/O connectors were used to interface a Coldfire CPU module.

The primary function of the FPGA is to process the beam position data though a programmable 32-tap FIR (finite infinite response) filter. Each tap in this filter uses two 9-bit DSP blocks to create an 18-bit-wide filter, using a total of 64 DSP blocks per channel. This project requires 288 DSP blocks less than half of the available DSP blocks provided by the EP3S60. Both channels share a common coefficient register, which can be modified while data is being processed. Before the data is processed by the FIR filter, the sampled data is preprocessed through a programmable high-pass filter. The output of the FIR filter is then passed through a programmable delay before being output to a D/A converter (see Fig. 1).

The analog circuit is clocked at 88 MHz, which is a quarter of the storage ring operating frequency at APS. Originally this design was to focus on only the P0 (first of 1296) bunch, but since this FPGA has sufficient internal RAM, the design was restructured to include every fourth bunch, for a total of 324 bunches per turn. Each bunch stores its own 32 18-bit values for the FIR filter. By using this method, all 324 bunches are processed in parallel within 3.68 μs, the time it takes one revolution of beam in the storage ring. Once the first bunch (P0) trigger is detected, both channels operate simultaneously, looking like 648 individual FIR filters. The computation speed for all 648 filters is approximately 6×10^9 multiply-accumulate operation per second.

IOC COMPONENT

The FPGA has the ability to load its own CPU core called the "Nios II" [3] processor. However, an EPICS board support package (BSP) would have to be developed and maintained. Maintenance on this BSP can be time consuming with the constant upgrades in the FPGA development tools coupled with the various chip families. With the rapid growth in FPGA technologies, it was uncertain that a reliable internal BSP could be maintained; another option had to be used. At the APS, a uCdimm Coldfire 5282 microcontroller module (uC5282) [4], a complete system about the size of a laptop computer memory module, has been integrated into the P0 Feedback system. The Coldfire module has many additional features which make this a very attractive CPU. The basic core of this module is based on a 32-bit RISC architecture. The module includes a 10/100 BaseT Ethernet controller, a multiply and accumulate (MAC) for

*Work supported by U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences, under Contract No. DE-AC02-06CH11357.
#npd@aps.anl.gov

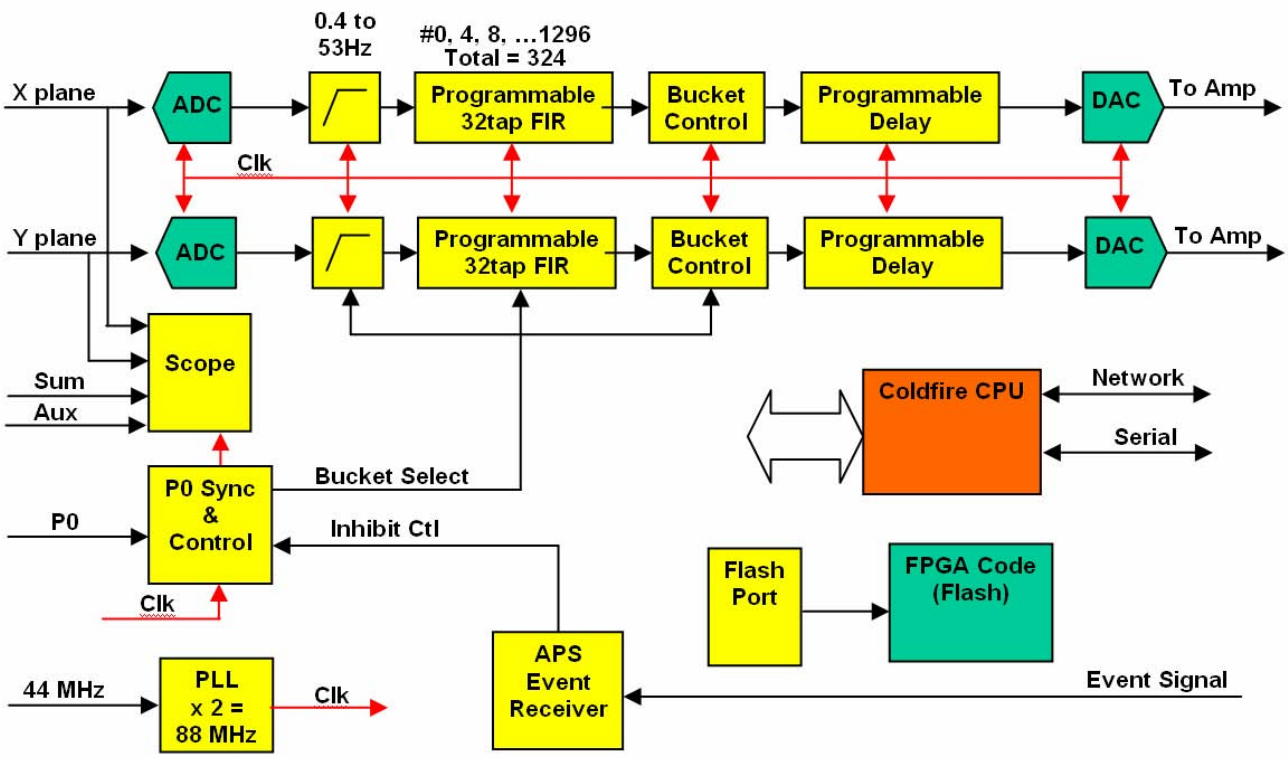

Figure 1: Block diagram of the P0 Feedback system.

DSP, two RS232 ports, one QSPI (Queued Serial Peripheral Interface), hardware timers with pulse-width modulation, a CAN bus controller, and eight 10-bit A/D channels. This CPU had already been successfully installed in other systems at the APS, giving us a high degree of confidence on the reliability of its use.

The CPU is directly connected to the FPGA and uses the Nios II bus architecture called the "Avalon Bus" [5] to access the various functions of the P0 Feedback system. Altera provides a tool to configure the Avalon Bus called SOPC [6], which stands for System-On-a-Programmable-Chip. To interface to this Avalon Bus, a master Avalon Bus component was developed at the APS called the ColdFire Bridge. This component translates the bus signals from the ColdFire CPU to the Avalon Bus specification and also resolves the difference in the two clock domains.

When connected to the Coldfire Bridge, this Avalon Bus looks similar to a VMEbus. A good example of this is with the APS Event Receiver. The classic APS Event Receiver is in a VMEbus form factor with all of its hardware on both the front panel and through the P2 connector. But, the core logic in this module was transplanted into an Avalon Bus slave component. Only the required inputs and outputs are utilized in this system though the provided I/O connectors. The interface logic in the Avalon component is much simpler than its VME counterpart. Since most of the address decoding is handled on the Avalon Bus, the old VME interface was removed and replaced with simple select logic. All of the main functionality was maintained, including the register addressing, register layouts, and the ability to generate interrupts. In fact, the old VxWorks device support was changed slightly only to support RTEMS [7] system calls. The Avalon component of the APS Event Receiver is configured with the same address as the VME counterpart. This is done using the same configuration statement in the EPICS startup command file that specifies the address and vector number.

The development of the other slave components, illustrated in Fig. 2, use the same simple Avalon Bus interface. Once a component is created, it is added in the SOPC builder, the address and IRQ are set, and then it is compiled. SOPC will generate a system.h file that can be linked to the ASYN [8] device support for each of these components, making the EPICS build even simpler. Even the "scope" component required little time to develop in EPICS. This was facilitated by the use of the Generic Transient Recorder (GTR) [9] for the oscilloscope functions in the FPGA. This device support provides all the necessary controls functions to operate a transient recorder or oscilloscope. Several examples of VME-based transient recorders using GTR device support are provided in the package. This facilitated the GTR support for this FPGA scope, which required very little effort to adapt from these examples. In this FPGA application, data from all four A/Ds are connected to the scope function with each channel using 4 k of memory. Upon receiving an interrupt indicating memory is at least half full, data is read from the scope. The GTR support displays the recorded data according to the user setting in the GTR control panel. Data from the scope is used to monitor the

effects on this system, which aid in adjusting the P0 Feedback parameters.

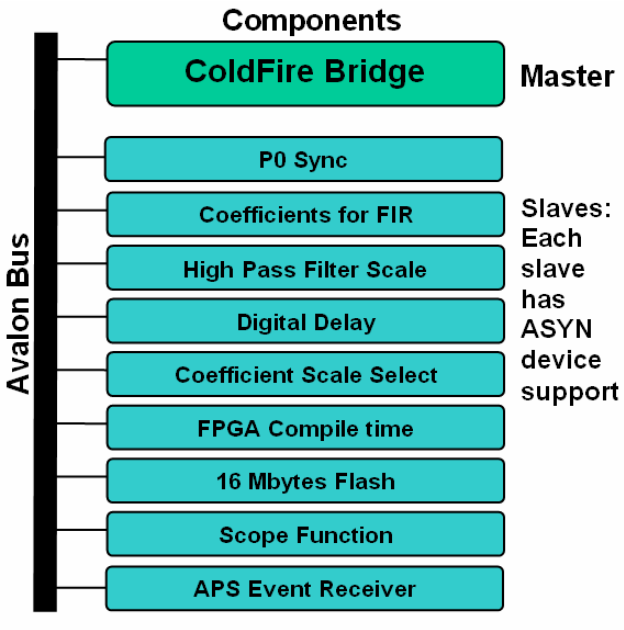

Figure 2: Avalon Bus components for P0 Feedback system.

CONCLUSION

Combining a ColdFire CPU with an Altera FPGA blurs the line when it comes to what is considered a traditional communication bus, like VME. Using the Avalon Bus in this configuration made it easy to transplant an existing function into this structure. Using EPICS R3.14, GTR, and ASYN device support further reduced the development time needed for this project due to simpler device support requirements. In all, the development time was reduced from six months down to two.

ACKNOWLEDGMENTS

The author would like to acknowledge Eric Norum for the many helpful discussions, his work in creating the ColdFire Bridge, and his support in both RTEMS and ASYN. All of these contributed greatly to the success of this project.

REFERENCES

[1] EPICS, http://www.aps.anl.gov/epics
[2] Altera DSP Development Kit with Stratix II, http://altera.com/products/devkits/altera/kit-dsp-2S60.html.
[3] Altera Nios II Integrated Development Environment, http://altera.com/support/ip/processors/nios2/ide/ips-nios2-ide.html.
[4] ColdFire, http://arcturusnetworks.com/coldfire5282.html.
[5] AvalonBus, http://altera.com/products/software/products/sopc/ avalon/nio-avalon_bus.htm
[6] SOPC, http://altera.com/products/software/products/sopc/sop-index.html
[7] RTEMS, http://www.aps.anl.gov/epics/base/RTEMS.php
[8] ASYN, http://www.aps.anl.gov/epics/modules/soft/asyn/
[9] GTR, http://www.aps.anl.gov/epics/modules/analog/gtr/index.html