Distributed and Parallel Real-time Control System Equipped FPGA-Zynq and EPICS Middleware

Sangil Lee, Changwook Son, and Hyojae Jang

Abstract—**Zynq series of Xilinx FPGA chips are divided into Processing System (PS) and Programmable Logic (PL), as a kind of SoC (System on Chip). PS with the dual-core ARM CortexA9 processor is performing the high-level control logic at run-time on linux operating system. PL with the low-level Field Programmable Gate Array (FPGA) built on high-performance, low-power, and high-k metal gate process technology is connecting with a lot of I/O peripherals for real-time control system. EPICS (Experimental Physics and Industrial Control System) is a set of open-source-based software tools which supports for the Ethernet-based middleware layer. In order to configure the environment of the distributed control system, EPICS middleware is equipped on the linux operating system of the Zynq PS. In addition, a lot of digital logic gates of the Zynq PL of FPGA-Zynq evaluation board (ZedBoard) are connected with I/O pins of the daughter board via FPGA Mezzanine Connector (FMC) of ZedBoard. An interface between the Zynq PS and PL is interconnected with AMBA4 AXI. For the organic connection both the PS and PL, it also used the linux device driver for AXI interface. This paper describes the content and configuration of the distributed and parallel real-time control system applying FPGA-Zynq and EPICS middleware.**

I. Introduction

RAON is a new heavy ion accelerator under construction in South Korea, which is to produce a variety of stable ion and rare isotope beams to support various researches for the basic science and applied research applications[1]. RAON, which is a colossal machine, is composed of many control devices, experimental equipments, and additional utilities. A big scientific experiment is required by the unity of the heterogenous distributed control system according to its characteristic and the fast response for a particular device. Furthermore, these requirements for the distributed and parallel environments must be satisfied at the same time. The distributed processing is to handle dividing one job into several processes and the parallel processing is to handle the multiple jobs at the same time. To satisfy the general requirements under the distributed environment, it needs to the middle-ware layer to support for the distributed control system. For the purpose of implementing it, the control system of RAON has decided to apply the EPICS (Experimental Physics and Industrial Control System) software framework. "EPICS is a set of open source software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as a particle accelerators, telescopes and other large scientific experiments"[2]. Additionally FPGA chips are used in general for monitoring or control system of the fast response time. There are also many control systems that adopted FPGA among the RAON equipments, such as timing system, LLRF control system, power supply cotrol system and so on. A lot of devices or systems have implemented the control system configuring CPUs with OS and FPGA chips using VME or cPCI interface. The low-power clustering application for parallel computing is also one of the fields utilizing the GPU or FPGA. Likewise, the configuration of the specific hardwares, such as GPU processing elements or FPGA gate logics, it can be saved the prorated cost of the CPU for control and analysis algorithms. Zynq series of Xilinx FPGA chips are called the programmable SoC. Within the one chip it consists of the arm-based CPU and the logic gates of FPGA. PS with the dual-core ARM CortexA9 processor is performing the high-level control logic at run-time on linux operating system. PL with the low-level FPGA is connecting with a lot of I/O peripherals for real-time control system. Utilizing zynq SoC, it could configure an environment of the distributed and parallel processing at a time. In addition, the control system may provide an abstract layer integrated with EPICS middleware using Open Computing Lanaguage (OpenCL)[3] for the heterogenous parallel processing hardwares in future plan.

F. A. SI Lee is with the Institute for Basic Science, 70 Yuseong-daero, CO 34047 South Korea (telephone: +82-42-878-8854, e-mail: silee7103@ibs.re.kr.

S. B. CW Son is with the Institute for Basic Science, 70 Yuseong-daero, CO 34047 South Korea (telephone: +82-42-878-8831, e-mail: scwook@ibs.re.kr.

T. C. HY Jang is is with the Institute for Basic Science, 70 Yuseong-daero, CO 34047 South Korea (telephone: +82-42-878-8788, e-mail: lkcom@ibs.re.kr.

II. Distributed Processing Environment

Configuration of Fig.1 shows the overall of the control system for the distributed and parallel processing. The PS of zynq is equipped with the EPICS software framework on the cross-compiled linux OS of the ARM processor. The parameter values for the motor control is set through the EPICS interface on PS. The data communication between PS and PL uses the AXI interface of the ARM Advanced Microcontroller Bus Architecture (AMBA). Software module for AXI interface between both should be also developed in the linux device driver. The types of motor which can be supported by a low voltage drive board of the Analog Device[4] are brushless DC, PMSM, brushed DC or stepper motor.

A. Linux on Zynq PS

To operate linux OS on the ARM processor of the zynq PS there consists of:

- ARM Cross Compile Tool Chain (arm-linux-gnueabihf)

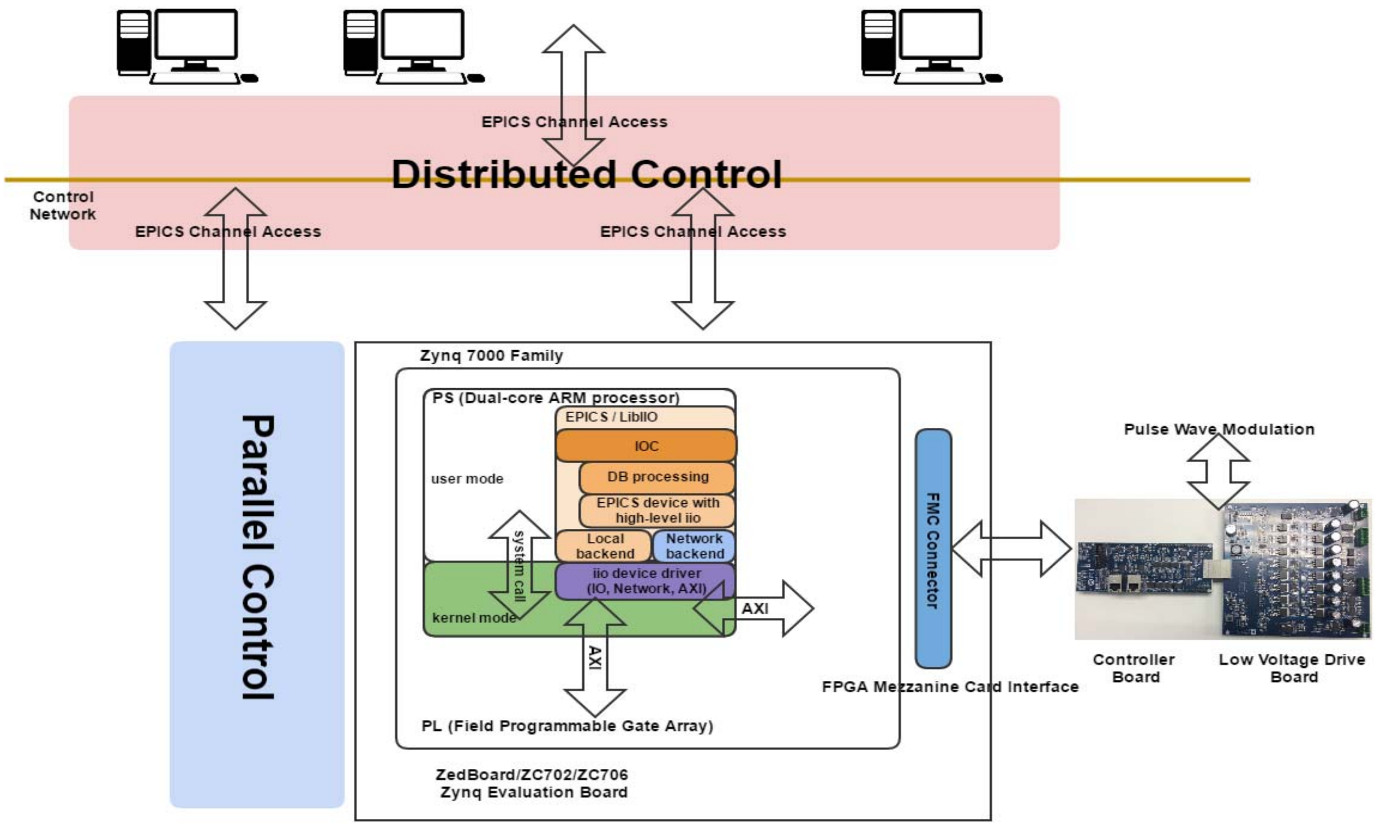

Fig. 1. Distributed and Parallel Control System Configuration

- Linux Kernel Source (Linaro)
- Bootloader (BOOT.BIN)
- Board Support Package (Linux Device Tree)
- Root File System (Busybox)

”Linaro is a engineering organization that works on free and open-source software such as the Linux kernel, the GNU Compiler Collection (GCC), power management, graphics and multimedia interfaces for the ARM family of instruction sets and implementations thereof as well as for the heterogeneous system architecture”[5]. The linaro kernel source is the ubuntu-based kernel source including a lot of device driver such as zynq device tree. The bootloader of PS uses U-Boot[6] supporting for the zynq device of Xilinx. To boot the zynq device, it should be made up the boot.bin file created by Vivado tool. The boot.bin file consists of fsbl.elf (First Stage BootLoader), u-boot.elf (Second BootLoader), uImage (kernel zImage for uboot), zynq.bif (Boot Image Format file) and user.bit (bitstream file of FPGA). The linux kernel is using the root file system generated by the Busybox[7] tool.

B. Device Driver on Linux

Linux device driver for zynq PS uses Industrial IO (IIO) module of the Analog Devices. Libiio[8] is a library that has been developed by Analog Devices for easy interface with IIO devices. Localbackend module of Fig. 1 goes into kernel mode from user mode through the system call and accesses the iio device driver via ”struct file_operations”. Basically the iio device driver is a character device type.

C. EPICS Middleware Framework

EPICS IOC is being implemented using the libiio library that abstracted the low-level details of the hardware, as shown in Fig. 1. The device support routine of EPICS is implemented using the libiio library to read or write the parameter values for the motor control. The motor control parameters become the Process Variables (PVs) of EPICS and are carried out the EPICS database processing according to EPICS scan rate. Also, it is going to develop additional EPICS waveform PV to monitor the Pulse Width Modulation (PWM) signal of the low voltage drive board.

III. Parallel Processing Environment

A. Zynq Evaluation Board

The ZC706 evaluation board for All Programmable SoC (AP SoC) provides a hardware environment for developing and evaluating designs targeting the Zynq-7000 AP SoC. The ZC706 board has the SubMiniature version A (SMA) port which is the programmable user clock for the Transistor-Transistor Logic (TTL) out signal of the timing board. The reason which selected the board is to have the user clock port to receive input from the external trigger.

B. Controller for Motor Drive

As shown Fig. 1, controller board communicates with ZC706 via FPGA Mezzanine Card (FMC) connector. FMC connector of the controller board is connected to XADC interface, digital I/O and sensors, two gigabit ethernet modules, power line and so on. XADC and digital I/O signals among those connections are delivered to the low voltage drive board via the drive board connector. The low voltage drive board received those signals generates PWM signal through the Metal-Oxide-Semiconductor Field Effect Transister (MOSFET) gate driver and drives the stepper motor. The low voltage drive board is operated in 12∼24V DC power from the external power source.

C. FPGA Interface

FPGA code reuses the Hardware Description Lanaguage (HDL) code which is provided by the Analog Devices. FPGA HDL code is develeped by VerilogHDL using Vivado of Xilinx. When the motor controller was purchased from Analog Devices, it operated on ZedBoard. It should be changed to operate the ZC706 board as well as ZedBoard. It is going to transplant the code of ZedBoard into ZC706 board.

TABLE I shows the software modules that used to incorporate a distributed process control system and a parallel processing control system.

TABLE I
SOFTWARE MODULES

Software	Contents	Module
EPICS	Base R3.14.15.2	PS of Zynq
Libiio	Industrial I/O library supplied Analog Devices	
Kernel	Linaro kernel source including iio device driver	
Bootloader	U-Boot for zynq	
IOC	In-house using Libiio	
FPGA	Analog Devices and In-house code	PL of Zynq
Vivado	Vivado14.4 including SDK with node lock license	Tool
Busybox	for rootfs system (free)	
Toolchain	for ARM cross-compile (GNU)	

IV. SOFTWARE DEFINED SYSTEM ON CHIP PLATFORMS

The SDSoC (Software Defined System On Chip)[9] environment is an Eclipse-based Integrated Development Environment (IDE) for implementing embedded systems using the zynq programmable SoC platform.The SDSoC environment includes support for the ZC702, ZC706, MicroZed, ZedBoard and Zybo development boards featuring the Zynq-7000 AP SoC.

An SDSoC platform can define as follows:

- Hardware and Software Architecture
- Application Context including Linux OS
- Bootloader
- Device Drivers for I/O or AXI Interface
- Root File System
- External Memory Interface
- Custom Input/Output
- User-defined Libraries

A series of the works that were previously described in chapter [2,3] can be easily configured using the SDSoC platform as shown in Fig.2. Fig.2 shows the configuration of the software

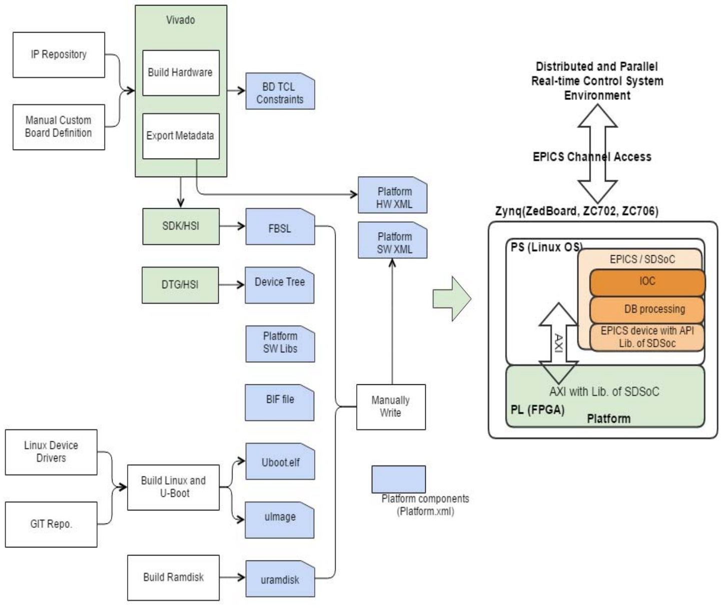

Fig. 2. Control System Configuration using SDSoC

platform to perform the application context using SDSoC on zynq PS, such as a linux kernel image, boot loader, root file system, device tree, and the hardware platform to generate the FPGA logic through vivado on zynq PL. After this configuration, EPICS middleware that cross-compiled on the linux OS is interconnected with the C libraries through the SDSoC. Therefore it does not need to develope the AXI interface linux device driver.

V. FUTURE WORK

In order to connect the distributed processing system and parallel processing system more effectively, it should provide a unified interface layer for the specific various parallel processing hardwares. For the purpose of developing the unified parallel processing interface layer it can be used Open Computer Lanaguage (OpenCL).

A. OpenCL for Parallel Processing

OpenCL[10] is a framework or programming model for writing programs that execute across heterogeneous platforms consisting of CPUs, GPUs, DSPs, FPGAs and other processors or hardware accelerators and was developed by the Khronos group. Hardware manufacturer must provide application program interface libraries to satisfy the specification according to OpenCL version. Fig.3 shows the configuration of the unified interface layer for the parallel processing of GPU (NVIDIA or AMD) and FPGA (Xilinx or Altera) using OpenCL. The data interface between the parallel processing

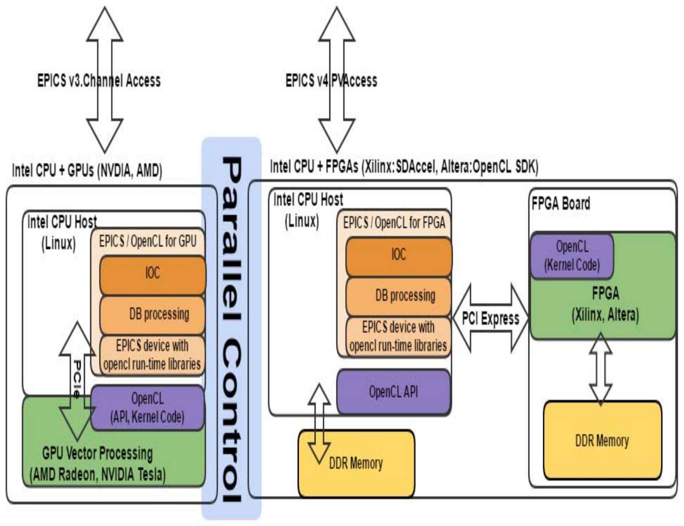

Fig. 3. OpenCL-based Parallel Control System Configuration

hardware and the host make up with PCI Express (PCIe) bus. This configuration for the FPGA hardware is required the DDR memory separately. The GPU already uses the built-in DDR memory. Eventually the kernel code of OpenCL is performed as parallel processing code through OpenCL interface API. The integration of the OpenCL code and the EPICS IOC code means the integration of the distributed system and parallel system.

B. EPICS Version 4

EPICS v3 is not enough to hold the contents of the much more complex scientfic data structure like a image data. Reflecting this requirement, EPICS v4[11] provides efficient storage, access, and communication, of memory resident structured data. The EPICS v4 Normative Types[12] are a collection of structured data types that can be used by the application level of EPICS V4 network endpoints, to interoperably exchange scientific data. Adopting EPICS v4 as the distributed system it will be interfaced large amounts of data and complex data to be processed in the parallel processing of a variety of experimental devices.

VI. Conclusion

Big scientific experiment facilities such as accelerators, nuclear fusion, and telescope are required by the unity of the heterogenous distributed control system and the fast response according to its characteristic. These requirements for the distributed and parallel environments must be satisfied at the same time. If the distributed control system using EPICS makes a connection with the high speed parallel processing of FPGA, it is possible to improve the performance and efficiency of the control system. Zynq SoC can be considered as an ideal device to satisfy with the distributed and high-speed parallel control system at the same time. More and more, the control environment of devices used in the big science experiment of the future will be used the distributed processing environment combined with the parallel processing environment. Therefore, it expect to be increased the requirements for the unified software abstraction layer to operate the heterogenous parallel proccessing hardwares."FPGA-based heterogeneous system (CPU + FPGA) using the OpenCL standard has a significant time-to-market advantage compared to traditional FPGA development using lower level hardware description languages (HDLs) such as Verilog or VHDL"[13]. As a result, utilizing the OpenCL standard on the heterogeneous parallel processing hardwares may offer significantly higher performance and the unified abastract layer at much lower power.

Acknowledgment

This work is supported by the Rare Isotope Science Project funded by Ministry of Science, ICT and Future Planning **(MSIP)** and National Research Foundation **(NRF)** of Korea (Project No. 2011-0032011).

References

[1] Y. K. Kwon, *et. al*,"Status of Rare Isotope Science Project in Korea", Few-Body Syst 54, 961-966, (2013).
[2] EPICS website, http://www.aps.anl.gov/epics/
[3] OpenCL, https://www.khronos.org/opencl/
[4] Analog Devices website, http://www.analog.com
[5] Linaro Document website, https://en.wikipedia.org/wiki/Linaro
[6] U-Boot Document website, http://www.denx.de/wiki/U-Boot
[7] Busybox Document website, http://www.busybox.net/
[8] Industrial I/O Document website, https://wiki.analog.com/resources/tools-software/linux-software/libiio
[9] SDSoC Document, http://www.xilinx.com/support/documentation/sw_manuals/xilinx2016_1/ug1146-sdsoc-platforms-and-libraries.pdf
[10] OpenCL Definition, https://en.wikipedia.org/wiki/OpenCL
[11] EPICS Version 4, http://epics-pvdata.sourceforge.net/
[12] EPICS Version 4 Normative Types, http://epics-pvdata.sourceforge.net/docbuild/normativeTypesCPP/tip/documentation/ntCPP.html
[13] OpenCL on Altera, https://www.altera.com/en_US/pdfs/literature/wp/wp-01173-opencl.pdf