Monitor Interfacing through Soft Core Processor

Gendlal M. Vaidya
PG Student
Department of Computer Science Engineering,
G.H. Raisoni College of Engineering, Nagpur
India

D. V. Padole, Ph.D
Professor
Department of Electronics Engineering,
G. H. Raisoni College of Engineering, Nagpur
India

ABSTRACT
This paper presents co designing of software and hardware which is used for the soft core processor based applications. The ALTERA's SOPC (system on programmable chip) development tool is used to implement the system, which contains an NIOS II processor. To interface the monitor, we need the ALTERA's DE1 board on which the RS232 serial port is available. It also provides Field Programmable Gate Array (FPGA) which is loaded first with the Nios II soft core processor and then Real Time Operating System (RTOS) installed on this FPGA based architecture contained with the soft core processor.

Keywords
SOPC Builder, FPGA, RTOS, Nios II soft core processor

1. INTRODUCTION
In the past few years, embedded systems have been changing our lives and are used in a diverse range of products, such as mobile phones, home appliances, medical equipments and so on. System-on-a-programmable-chip (SOPC) is currently the primary interest in the embedded systems and digital design. It enables to employ a large field programmable gate array(FPGA) that contains both memory and logic elements along with an intellectual property(IP) processor core to implement a computer and custom hardware for system-on-a-chip(SOC) applications. During the past seven years, several commercial reduced instruction set computer (RISC) processor cores have been introduced [1]. As the growing popularity of SOPC designs and the increasing importance of programmable logic, the educational system must cater to the needs of both Computer Science and Electrical Engineering students at the undergraduate level.

Embedded Systems Architecture design is the task of selecting and programming a suitable configuration of components for a given system application. Programmable chip companies, with the help of Moore's law, are providing us with amazing selection of components to do this. Traditionally, the creation of embedded system architectures used to be relatively straightforward: use a microcontroller for flexibility and add hardware peripherals for specialized functions. Now designers can add multiple component types (e.g. FPGAs, DSPs, and Application-Specific Instruction-set Processors) to find the optimum over multiple design objectives, including system flexibility, power consumption, design cost, and design time. A central idea in hardware/software co design is to merge two design processes: hardware design uses spatial decomposition and is well suited for performance, while software design uses temporal decomposition and is well suited for flexibility. A successful combination of hardware and software enables designers to obtain solutions that are the right combination of flexibility and performance. The hardware/software co-design as a simplified version of the more complex trade-off that needs to be made during embedded systems architecture design, namely the partitioning between platform architecture and platform function. For this reason, we think that hardware/software co-design is the proper starting point for education in this area.

Among programmable components, FPGA platforms have been very successful in providing a target that equally suits software design and hardware design. Several courses have explored this in the context of co design. We also note that there is a complementary view to embedded systems design which starts from a software-centric system view (rather than a hardware-centric system view). In this case, the problem being addressed is how to teach architecture-specific software. The Embedded Software consortium in Taiwan, for example, has defined a software curriculum because of the high add-on value that software can bring to hardware design .Vanderbilt University has defined and embedded-software and systems concentration in their engineering curriculum to address the specific needs of embedded software that interacts with electrical, mechanical and other hybrid systems.

2. LITERATURE SURVEY
2.1 Altera's DE-1 Board:
The Nios II processor core is a soft-core central processing unit that could program onto an Altera field programmable gate array (FPGA). If different Quartus II and NIOS II EDS version used, there will have some small difference during the operation. Other kinds of development board based on Altera FPGA chip also supported.

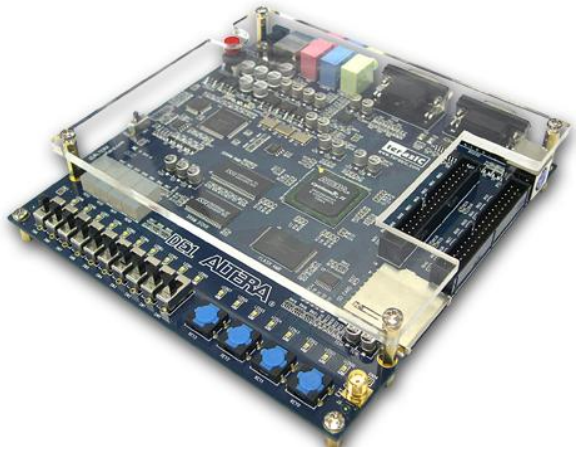

Figure 1: DE1 Board

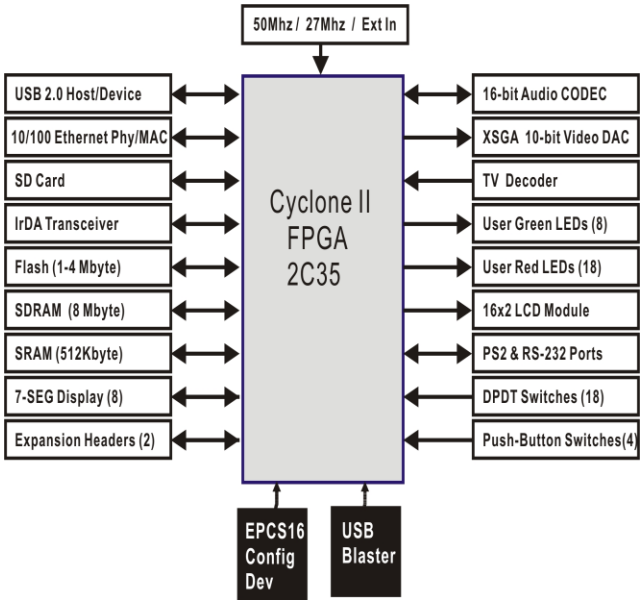

Figure 2: DE1 Block Diagram

The NIOS II standard hardware system provides the following necessary components:

- Nios II processor core, that's where the software will be executed
- On-chip memory i.e RAM to store and run the software
- JTAG link for communication between the host machine and target device
- hardware (typically using a USB-Blaster cable)
- LED peripheral I/O (PIO), be used as indicators

2.2 Field Programmable Gate Array (FPGA's)

A field-programmable gate array (FPGA) is an integrated circuit designed to be configured by a customer or a designer. The FPGA configuration is usually specified using a hardware description language (HDL), similar to that used for an application-specific integrated circuit. FPGAs can be used to implement any logical function that an ASIC could perform. The capability to update the functionality after delivery, partial reconfiguration of a portion of the design and the low non-recurring engineering costs relative to an ASIC design, offer advantages for many applications.

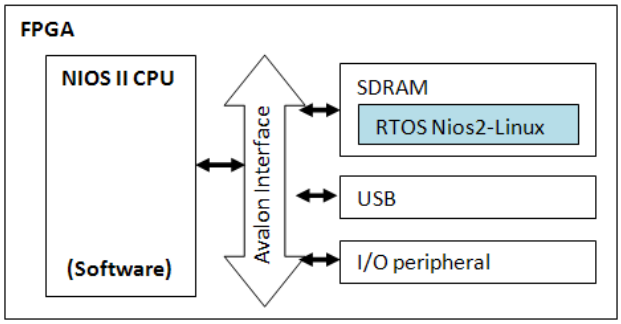

Figure 3: Embedded System Architecture

FPGAs contain programmable logic components called "logic blocks", and a ladder of reconfigurable interconnects that allow the blocks to be "wired together"—somewhat like many (changeable) logic gates that can be inter-wired in (many) different configurations. Logic blocks can be configured to execute complex combinational functions, or merely simple logic gates like AND and XOR. In most FPGAs, the logic blocks also consist of memory elements, which can be simple flip-flops or more complete blocks of memory.

3. SOPC TECHNOLOGY OVERVIEW

3.1 SOPC and Processor Cores

Hard-processor core is an embedded-processor core which is designed in dedicated silicon and implemented in an FPGA chip. It provides performance, power consumption and cost tradeoffs between a traditional application-specific integrated circuit (ASIC) and an FPGA. Several FPGA manufacturers provide a number of different processor types. For example, Altera offers an ARM processor core embedded in its APEX 20KE family FPGAs. Xilinx's Virtex-II Pro family FPGAs generally contain up to four PowerPC processor cores on chip. Cypress Semiconductor also offers a variation of the SOPC system. Cypress's Programmable-System-on-a-Chip (PSOC) is based on an M8C processor core with configurable logic arrays designed to implement the peripheral interfaces, which include counters, timers, A to D converters, D to A converters, universal asynchronous receivers–transmitters (UARTs) and so on[2], [3]. Soft cores, such as Xilinx's MicroBlaze and Altera's Nios-II processors, use configurable programmable logic gates in FPGA to implement the processor. Soft core processors can be very flexible and feature-rich which allow designers to determine arithmetic logic unit (ALU), data bus, width peripherals and memory-address space parameters on their own. However, such flexibility comes at a cost of performance and power consumption. For instance, soft cores operate at slower system clock and consume more power than an equivalent hard core processor. Implementing a soft processor core in FPGA, which mainly uses logic elements, only costs a few dollars on the basis of current prices of FPGAs. The remainder resources of FPGA can be used to build application-specific module, such as, user's control logic, digital filters and so forth. Traditional SOC devices such as ASICs and custom very-

large-scale-integration integrated circuits (VLSI ICs) offer better performance, but they also have large development costs and longer design cycles [4]. FPGA-based SOPC approach is easier, faster, smaller, and economical is a good choice.

3.2 SOPC Development Platform

The development platform consists of a Altera's FPGA device Cyclone EP2C35,Flash, off chip SRAM, SDRAM, LCD, SD interface, CF interface, IDE interface, LED, microphone, ADC, DAC, RS232, RS485 and so on. The development kit includes a complete set of tools for SOPC design including Quartus II v12.1. Some third-party vendors also provide software for the development of systems based on the Nios-II processor. And there are many embedded operating system support for the Nios-II processor such as eCos, embOS, Erika Enterprise, Euros RTOS, MicroC/OS-II, ThreadX, Linux, uClinux and so on [7].

3.3 SOPC Design Tools

The configurable parameters include cache, memory, data path width, address space, UARTs, general-purpose I/O, Ethernet controllers, SDRAM controllers and so on. Once the processor parameters are determined in the GUI interface, the processor core is produced in the form of a netlist file.This file and additional user logic module can be included as block box into top HDL design. Then, the full hardware design containing processor core and any additional user logic is compiled (synthesis, place and route, etc.),and the FPGA can be programmed with the resulting SOF or POF file using the related tool. So far, the hardware design is completed.
The next step is to write and simultaneously design the software that can be executed on the soft-processor core. Library files and associated C header files that are customized for the specific processor core are generated along with the netlist files at the same time in the hardware design stage mentioned above. The designer can compile code for an operating system ported to the processor core. The operating systems including MicroC/OS-II, ThreadX, Linux, uClinux are available from third-party vendors.
After a program binary file has been generated, it must be loaded into the processor's program and/or data memories to run. This can be done in several ways depending on chip architecture on the printed circuit board (PCB) and the memory configuration of the processor. If the application program is small and can stored into on-chip memory on the FPGA, then the code can be initialized in the on-chip memory. However, on-chip memory is very limited, and this method is not usually the choice. The application code can also be stored on an external EEPROM, Flash, and other form of nonvolatile memory. And the code can be executed directly from the external memory.

4. HARDWARE SOFTWARE CODESIGN

When the hardware as well as Software are used in parallel manner while designing any architecture, then the system is referred as a co designed system. This concept is also known as partitioning, in which hardware as well as software parts of the system are separated in parallel manner.

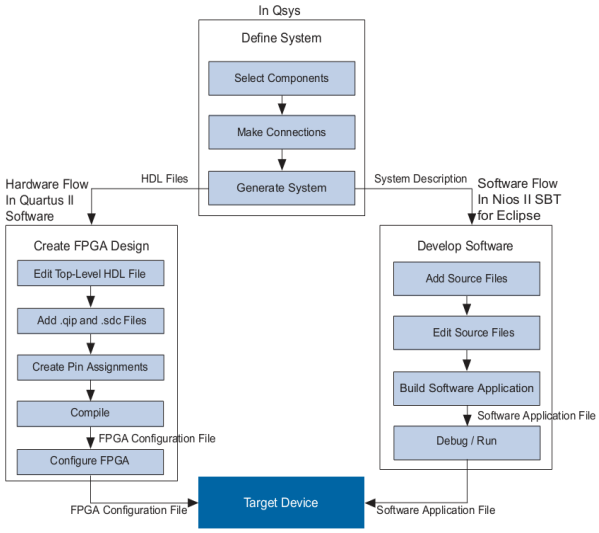

Figure 4: Hardware Software Co design

The modes of this project are:

(a) The soft core processors will be dumped on the Field Programmable Gate Array (FPGA's) which is generally available on Altera's DE-1 board.
(b) Any Real Time Operating System (RTOS) will be installed on that FPGA platform.
(c) Finally the application which can be easily implemented on this RTOS will be executed.

5. CONCLUSION AND FUTURE WORK

This paper gives the preliminary exploration of the SOPC design. FPGA and general-purpose SOPC kit saves both time and money by minimizing the necessity for supplementary hardware that is often needed to construct prototypes. To implement SOPC systems, first design both the hardware and software subsystems required to make many different decisions in partitioning systems and tradeoffs between hardware and software. The widespread of embedded control systems, microprocessor embedded processing systems, and multi-core processing systems will emphasize the importance of SOPC system education.

This work is depends upon the real time operating systems which will be installed on the FPGA platform where the sufficient memory for this RT OS is required. As every system requires its own processor, therefore the Nios II soft core processor will be programmed onto that platform. Once this setup gets ready, and then any applications like LED blinking, interfacing with monitors, etc and many more things can be done which are easily supported by that RTOS.

6. REFERENCES

[1] C. Snyder. (2000) "FPGA Processor Cores Get Serious," Cahners Microprocessor Report. http://www.MPRonline.com

[2] Julio Perez Acle, Matteo Sonza Reorda, Massimo Violante "Implementing a safe embedded computing system in SRAM-based FPGAs using IP cores: A case

study based on the Altera NIOS-II soft processor" in the proceeding of IEEE 2011

[3] P.C.Eng, M.Khalil-Hani " FPGA-Based Embedded Hand Vein Biometric Authentication System" in the Proc of TENCON 2009, IEEE 2009

[4] Cai Ken, Liang Xiaoying, Liu Chuanju "SOPC based flexible architecture for JPEG enconder" in the Proceedings of 2009 4th International Conference on Computer Science & Education, 2009 IEEE

P. Yiannacouras, J. Rose, and J. G. Steffan, "The micro architecture of FPGA based soft processors, in Proc. Int. Conf. CASES, pp 202–212 2005

[5] P.Metzgen,"Optimizing a high performance32-bit processor for programmable logic," in International Symposium on System-on-Chip2004. IEEE Computer Society, 2004,p.13

[6] Nios II, Altera. [Online]. Available: http://www.altera.com/products/ip/processors/nios2

[7] Pico Blaze 8-bit Embedded Microcontroller User Guide for Spartan- 3,Virtex-II and Virtex-II Pro FPGAs, Xilinx, Inc., November 2005

[8] Kentaro Sano "SW and HW Co-design of Connect6 Accelerator with Scalable Streaming Cores" IEEE 2011.

[9] MicroBlaze Processor Reference Guide Embedded Development Kit EDK8.2i, Xilinx, Inc, June2011.

[10] R. Lepetenok, AVR Core, opencores.org, http://www.opencores.com/projects/avr_core/,March 2007

[11] LEON2 Processor User's Manual, Gaisler Research, July 2005.

[12] NIOS-II Processor Reference Manual, Altera Corporation, San Jose, CA USA, 2009