AN INNOVATIVE IMPLEMENTATION TECHNIQUE OF A REAL-TIME SOFT-CORE PROCESSOR

Reza Raeisi, California State University, Fresno; Sudhanshu Singh, California State University, Fresno

Abstract

The objective of this project was to experience and develop rapid prototypes of a System-on-chip (SoC) by using a soft-core processor implementation on a Field Programmable Gate Array (FPGA). The research project described here was a partnership program between Altera Corporation and CSU, Fresno, to enhance the quality of both undergraduate and graduate education in the Electrical and Computer Engineering department. As engineering education is changing in response to the major technological changes in Electronics Design Automation (EDA) tools, the Altera Corporation donated the required EDA tools to build an industry-verified Digital Design Environment Laboratory.

Presented here is the use of these tools for a reconfigurable hardware-software co-design implementation of embedded systems on a FPGA using the µClinux Real Time Operating System (RTOS). This software-hardware co-design technique is useful for the design of both soft-core and hard-core processors and is ideal for teaching an embedded-system design course. This technique allows students to customize the exact set of several Central Processing Unit (CPU) peripherals and the interfaces needed for engineering design applications.

Introduction

One of the purposes of this research partnership project was to enhance and transcend graduate research in the field of soft-core processor embedded-system design. Also, to apply an industry-verified electronic design automation tool set for learning soft-core embedded systems in undergraduate education. Any device that includes a programmable computer, but itself is not a general-purpose computer, is termed as an embedded system. An embedded system is a special-purpose computer, which is designed to perform certain dedicated functions. This can be from portable devices such as cell phones and MP3 players, to a large stationary installation like bank teller machines or systems controlling power plants.

In general, embedded systems are not recognizable as regular computers, but instead are recognized as specific computers that usually do not interface with the real world through familiar personal-computer interface devices such as a mouse, keyboard or graphic-user interfaces. Instead, they interface with the real world through unusual interfaces such as sensors and other communication links. Soft-core embedded systems are preferred for teaching over hardware-embedded and general computers because of their design flexibility and ease of development. Students have the option of customizing their own embedded systems with the required peripheral subsystems.

Most time-constrained resource allocations and task scheduling across a spectrum of subsystems, such as sensor and actuator processing, communications, CPU, memory and other peripheral devices, are today required to use RTOS in order to meet the system response. FPGA-based RTOS embedded systems using soft-core processors are increasingly used in a variety of applications such as aircraft autopilots, avionics and navigation systems, anti-lock braking systems, and traction control systems. They have been adapted more commercially nowadays and are gaining popularity in educational institutions for teaching embedded-system courses [1]. Hence, it is essential to apply this new pedagogy for teaching embedded systems. In this study, the authors integrated a small, fast and efficient real-time operating system (OS), µClinux, with a soft-core processor and implemented them on FPGA platforms. The ultimate goal was to disseminate the use of soft-core processor experience for graduate research, classroom/laboratory teaching and learning in undergraduate education. The plan was to enhance the quality and complexity of the laboratory experiments, senior design and masters projects.

The focus of this study was the dissemination of the practical details of implementing and integrating a RTOS with a soft-core processor for interfacing with real-time applications through different I/O subsystems. Aside from this experience, another outcome of the project was the development of a digital-design environmental laboratory taking place to provide students a new perspective on digital design using better tools and equipment, than are currently being used in industry. The initial lesson learned from this experience led to the development of a new embedded-system design course "ECE 178". This is a senior-level course taken by students in the seventh semester of their program, before their senior-design project. This course has already gone through its approval stages at the University and was published in the 2009-2010 University general catalogs. It is scheduled to be taught during the spring of 2010, when an assessment of the course will be made.

Processor Overview

Embedded systems are implemented using the following classification of processors:

1. Hard-core processors
2. Soft-core processors

The hard-core processor is embedded in the form of silicon inside an FPGA, where a soft-core processor will use the programmable logical structure of an FPGA to implement the processor. They both have the advantage of using the FPGA logical elements for configuring and implementing peripherals interface to both soft-core and hard-core processors. Both design approaches are very common among manufacturers, but using the soft-core-processor approach is more flexible in academic environments. The soft-core-design approach allows students to specify the processor organization, functionality, and different peripheral connectivity. Therefore, using a FPGA soft-core processor for student design projects is more practical and can save both time and money as they can use and customize a variety of peripherals at their disposal. However, the hard-core processor tends to be faster because of faster clock rates, consumes less power, but they are not reconfigurable and have large development costs. Above all, a soft-core processor targeting FPGA is flexible because its parameters can be changed at any time by reprogramming the device [2]. There are many different kinds of soft-core processors available on the market including:

1. ARM
2. Microblaze
3. Nios II

A soft-core processor version of ARM has been implemented in an FPGA called ARM Cortext-M1 by Dominic Pajak [2]. The ARM Cortext-M1 processor is a streamlined three-stage 32-bit Reduced Instruction Set Computer (RISC) processor that includes: configurable instruction and data memories, optional OS support and system timer, 1 to 32 interrupts, fast or small multiplier and removable debug features.

The MicroBlaze soft-core processor is included as part of Xilinx Embedded Development Kit (EDK). The EDK comes with a standard set of peripherals including timers, UARTs, interrupt controllers, and external flash and memory controllers. There are many OSs to support the Xilinx MicroBlaze soft-core processor including µClinux.

The Nios II embedded processors were introduced in 2001 into the electronics industry by Altera as the viable commercial processor specially created for embedded-system designs in FPGAs [3]. Since then, it has been used widely in the industry and academia. It is a 32-bit soft-core processor, which is defined in a hardware descriptive language (HDL). It can be implemented in Altera's FPGA devices (DE-2) by using the Quartus II CAD system. The soft-core nature of the Nios II processor allows students to specify and generate a custom Nios II core, tailored for specific project requirements. Nios II is comparable to Xilinx MicroBlaze with RISC-type architecture. The Nios II platform was chosen because Altera Corporation agreed to be a partner in this study and provide the tools needed for the development of the embedded-systems project using µClinux RTOS. Some features of the Nios II include: access to up to 2GB of external address space, optional tightly-coupled memory for instructions and data, pipeline architecture, dynamic branch prediction, up to 256 custom instructions, and JTAG debug-module capability.

Nios II processors also allow for:

1. Customization of the CPUs, peripherals and the interfaces.
2. Increased performance by implementing real-time embedded-system applications.
3. Lower laboratory costs by not spending additional money on a hardware microcontroller board.

The Nios II processor can be used with a variety of other components to form a complete system. Altera's DE-2 development and educational board contains several components that can be integrated into a Nios II system. An example of such a system is shown in Figure1. Its arithmetic and logic operations are performed on operands in the general-purpose registers. The data are moved between the memory and these registers by means of Load and Store instructions. The word length of the Nios II processor is 32 bits. All registers are 32 bits long. The Nios II architecture uses separate instruction and data buses, which is often referred to as the Harvard architecture [4].

Soft-core Processor Real-Time Operating System (RTOS) Implementation

Real-time and embedded systems operate in constrained environments in which computer memory and processing power are limited. They often need to provide their services within strict time deadlines to their users and to the surrounding world. It is these memory, speed and timing

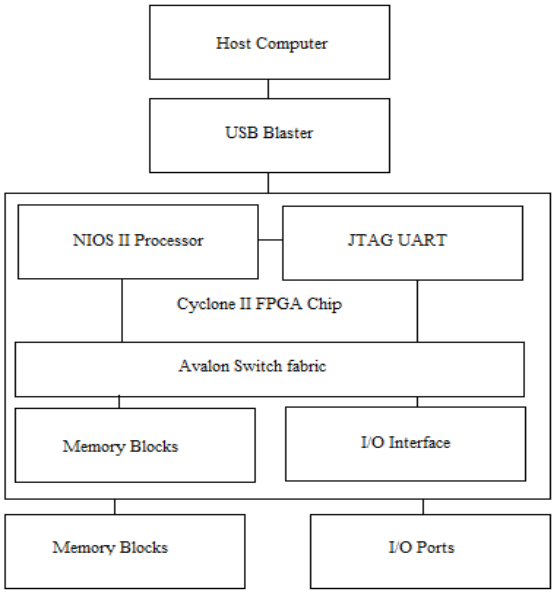

Figure 1. Nios II system implemented on the DE-2 board

constraints that dictate the use of real-time operating systems in embedded software. RTOS kernels hide from application software in the low-level details of system hardware, while at the same time providing several categories of services to the application software. These include: task management with priority-based preemptive scheduling, reliable inter-task communication and synchronization, non-fragmenting dynamic memory allocation, and basic timer services [5]. The µClinux kernel supports multiple soft-core CPU platforms including Altera's Nios II architecture. The main advantage of this operating system is that it is an open-source project and it is smaller than the regular Linux kernels. Most features of Linux kernels are available, like process control, file system, networking, and device drivers [6].

Altera's DE-2 board block diagram is shown in Figure 2. The DE-2 board and Quartus II software are used to implement the Nios II soft-core processor that supports a flexible memory option and I/O device combination. Experiments implemented using the NIOS II soft-core processor are real-time tasks using interrupt programming, and connections with I/O devices such as audio, video, USB, network and memory expansion. Quartus II is used to configure the available memory on the DE-2 board for the execution of small application programs. However, for larger application programs, whose size is beyond the capacity of the available on-chip memory, a Bootloader is placed into the on-board memory. The Bootloader will start running on boot-up and can receive a larger program binary file—over a serial interface—from external memories such as, SRAM, SDRAM, and Flash, for execution on the soft-core processor.

The Nios II Integrated Development Environment (IDE) is the standalone program that helps us to accomplish our task of implementing µClinux over the FPGA device. Nios II IDE 9.0 is the latest version of the software and can be downloaded from the Altera website. The distribution for µClinux can be obtained from http://Nioswiki.jot.com/WikiHome/. The Nios II community develops and releases the latest kernels according to the Altera software release. Because of the licensing issue, the authors have built the uClinux kernel in a Linux environment and then transferred the kernel image into the Windows to complete the project. Running the µClinux on a DE-2 board requires two steps. First, the FPGA must be conFigured to implement the Nios II processor system [7], and second the µClinux kernel image must be downloaded into SDRAM on the DE-2 board. Both configuration steps can be accomplished via the Nios II 9.0 command shell. Before starting the configuration of the DE-2 board, the power cable should be connected, the DE-2 board should be turned ON, and the USB cable connected between the PC and the USB blaster port on the DE-2 board.

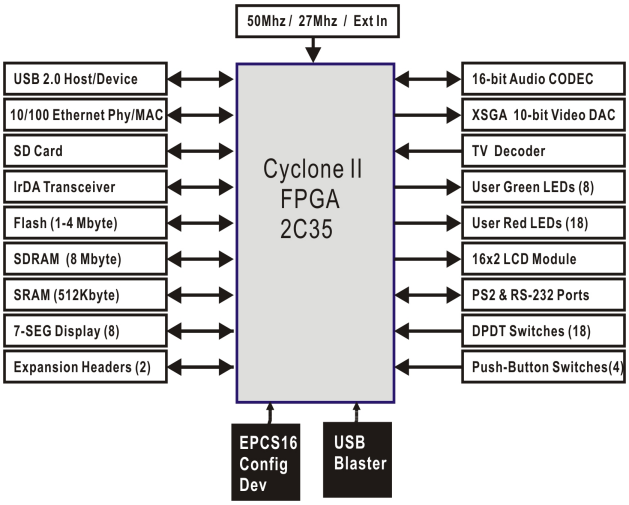

Figure 2. Altera DE-2 board block diagram

An existing Nios II project from the demonstrations directory of the enclosed DE-2 CD-ROM was used. The authors chose the DE2_NIOS_HOST_MOUSE_VGA project and used the *wget* command in the Linux terminal to download the µClinux distribution. A basic µClinux kernel image was built using the *make menuconfig* command. The completed image was located at Nios-linux/uClinux-dist/image/zimage. Zimage is a compressed form of a kernel image. The Linux kernel takes care of expanding the image at bootup. On Linux systems, vmLinux is a statically-linked executable file that contains the Linux kernel in one of the executable file formats supported by Linux, including ELF, COFF and a.out. To configure the FPGA and download the zImage to the processor, the following command steps were written into the Nios II 9.0 command shell [7].

Step 1. ConFigure the FPGA:
Nios2-conFigure-sof DE2_NIOS_HOST_MOUSE_VGA.sof
Step 2. Download and run the kernel image:
Nios2-download
-g zImage_DE2_NIOS_HOST_MOUSE_VGA_v1.6

After the kernel image was downloaded onto the DE-2 board, µClinux in *Nios2-terminal* became active and was ready, as shown in Figure 3.

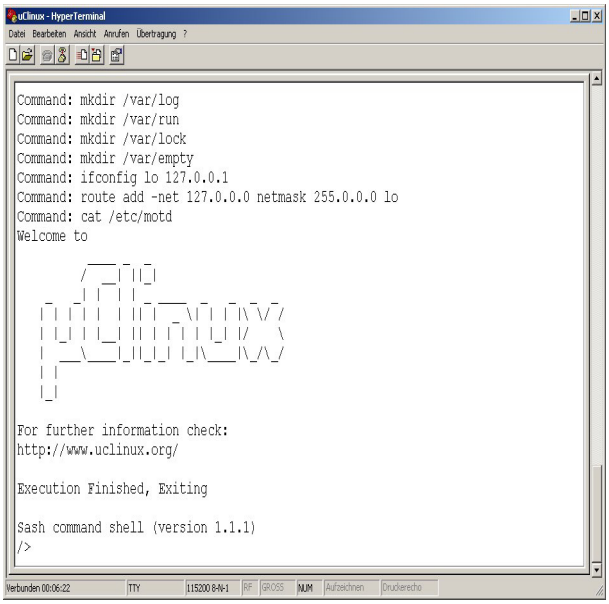

Figure 3. uClinux Implementation on Nios II

Application of the Soft-Core Processor

Once the µClinux is configured, the Nios II system was ready for use. The next step was to customize the kernel and add a user application. Next, after logging into the Linux platform, the *make menuconfig* operation was performed. One of the application experiments was to complete the Ethernet interfacing and connecting the board to the outside world. This was done by invoking FTP and Telnet. Then, the Ethernet network support was activated during the *make menuconfig* command.

The Ethernet connection was tested using the *ifconfig* command. The *ifconfig* command allows the operating system to set up network interfaces and the user to view information about the conFigured network interfaces. A valid IP address is displayed after the label *inet addr* as shown in Figure 4, which shows that the DE-2 board was successfully communicating on the network.

```
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

/> dhcpcd&
[22]
/>
/> ifconfig
eth0      Link encap:Ethernet  HWaddr 00:07:ED:00:00:00
          inet addr:128.59.151.179  Bcast:128.59.151.255  Mask:255.255.248.0
          UP BROADCAST NOTRAILERS RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3374 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:350749 (342.5 KiB)  TX bytes:2832 (2.7 KiB)
          Interrupt:6 Base address:0x10f8

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

/>
```

Figure 4. ifconfig result

Furthermore, the Nios II processor developed was used to implement a variety of experiments. Examples are writing a device driver for the LCD controller interface and I/O and interrupt programming. Figure 5 shows the experience with the pulse-width modulation (PWD) of a specific duty-cycle to manipulate analog circuitry from a digital domain application.

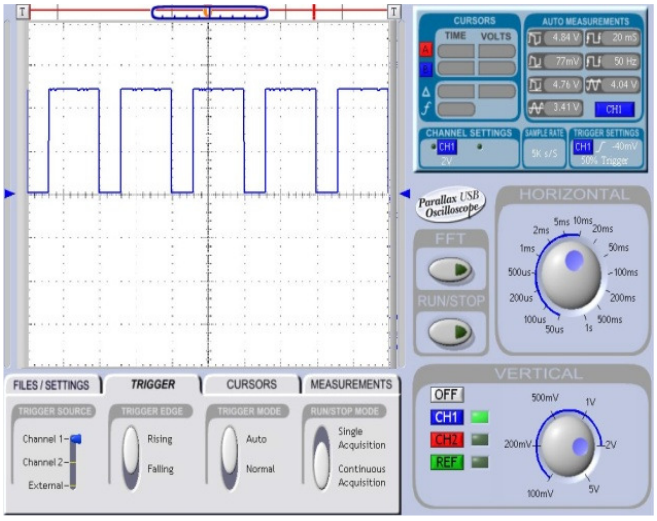

Figure 5. Pulse width modulation using Nios II

Conclusion

As a result of this study, a new course, ECE 178 embedded systems, was developed. This course demonstrates how soft-core-processor embedded systems can be implemented on a real-time operating system on an FPGA. During this project, various applications in the Linux kernels were examined. The Ethernet-connection application and some other I/O and interrupt programming was implemented. Also, the Altera grant impacted three master-student projects, in-progress at the time of publication. Now, the platform to transcend graduate research and teaching this new approach for learning embedded-system concepts for the undergraduate students in the ECE 178 embedded-systems course is set. Overall, FPGA soft-core processors for learning and teaching will enable us to define a variety of laboratory experiments with different complexity in each design. Also, it provides students with a better learning experience by having their design components at-hand and be more economical by reusing the components for different design projects.

References

[1] Tyson S. Hall and James O. Hamblen, "Using an FPGA Processor Core and Embedded Linux for Senior Design Projects", Proceedings of the IEEE International Conference on Microelectronics Systems Education, Issue. 3-4 pp. 33 – 34, June 2007.

[2] "Embedded Design with FPGAs and ARM Cortex-M1"-Dominic Pajak, ARM, Jean Labrosse, Micrium and Mike Thompson, Actel from the April-2008 embedded systems conference.

[3] Altera Homepage, www.altera.com

[4] Introduction to the Altera SOPC Builder- Quartus II Development Software Literature.

[5] Zongqing Lu, Xiong Zhang, Chuiliang Sun, "An Embedded System with uClinux based on FPGA", IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, pp. 691-694, 2008.

[6] Philipp Lutz , "Device drivers and Test application for a SOPC solution with Nios II soft-core processor and μClinux" Masters Thesis, University of Applied Sciences, Augsberg, 2008.

[7] J.O Hamblen, T.S Hall and M.D. Furman, "Rapid prototyping of digital systems", Chapter 18 Springer Press, 2007.

Biographies

REZA RAEISI is an Associate Professor in the Electrical and Computer Engineering Department at California State University, Fresno. He is also the Graduate Program Coordinator for the ECE department. His research interests include integrated circuits, embedded systems, and VLSI-CAD technology. He serves as Pacific Southwest regional director of American Society of Engineering Education. He is an entrepreneur with over 20 years of domestic and international experience and professional skills in both industry and academia. Dr. Raeisi may be reached at rraeisi@csufresno.edu

SUDHANSHU SINGH is a graduate student in Electrical Engineering in the College of Engineering, at the California State University, Fresno. He received his bachelor's degree in Electrical Engineering from Gujarat University. His interests involve VLSI design, which includes the physical design and timing analysis, embedded systems. Mr Singh may be reached at singh0601@csufresno.edu