Experiment 8: Microcontroller on an FPGA

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
142	5 April 2015		M

Contents

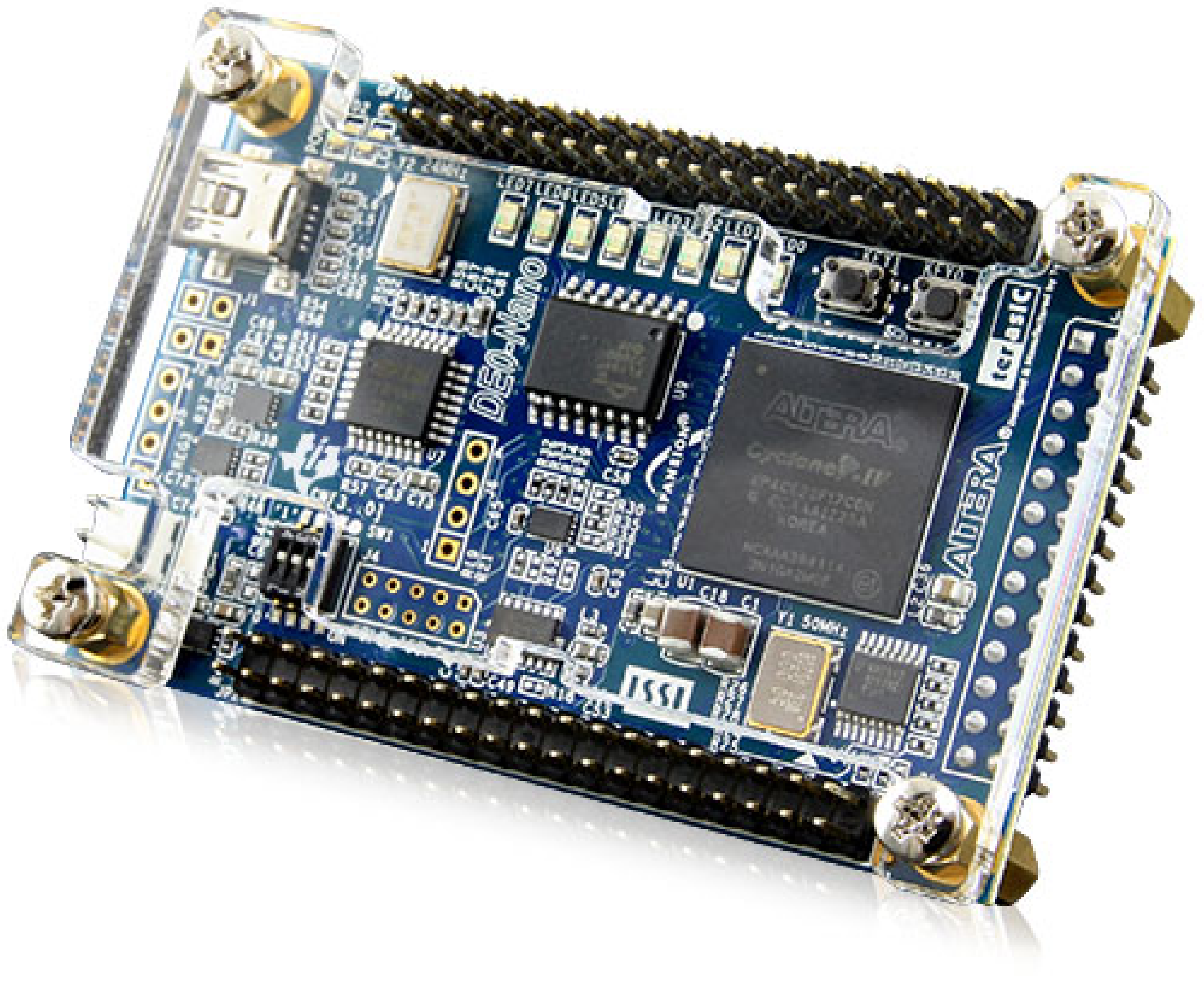
1 Objectives

In this experiment, you will learn about the hardware of a microcontroller by:

- Building a microcontroller system on an FPGA by integrating Altera's Nios II soft processor and a few peripherals
- Writing software for the FPGA-based microcontroller system

2 Parts List

- Altera DE0-Nano FPGA board



- USB A-Type to Mini-B cable

3 Background

This experiment involves two main tasks:

1. Building the hardware for a microcontroller system using an FPGA device.

   We will use Altera's *DE0-Nano* FPGA development board, which incorporates a small FPGA device and a number of peripherals. To create the microcontroller system, we will configure the FPGA device to implement a soft processor and a number of required components for the processor to function properly.

   To configure the FPGA, we will use Altera's *Quartus II* design software. To build the soft processor system, we will use Altera's *Qsys* system integration software to implement a system around Altera's Nios II soft processor core.

2. Developing software to exercise your microcontroller system.

   We will use Altera's Eclipse-based *Nios II Embedded Design Suite (EDS)* software development environment to build software for the Nios II-based hardware system.

Altera Software
You can obtain and install all the required software for this experiment freely from Altera's website. You will want to download and install *Quartus II Web Edition software*, which includes Qsys, the Nios II EDS, and Altera IP Library.
For more details on available Altera software, software licensing, download options, and hardware and software requirements, consult the Altera Software Installation and Licensing document.

3.1 The DE0-Nano FPGA Board

The DE0-Nano is a low-cost, low-power, portable, compact board (49 mm x 75 mm) aimed at developing embedded soft processor systems using the Nios II processor.

BOARD FEATURES

- Three-axis accelerometer with 13-bit resolution
- Eight-channel, 12-bit resolution analog-to-digital (A/D) converter
- Expansion headers: two 40-pin headers and one 26-pin header
- Two-pin external power header
- 32-MB SDRAM
- 2-Kb EEPROM
- Eight green LEDs
- Four dual in-line package (DIP) switches
- Two push-button switches

3.2 The Nios II Processor

Nios II is a RISC soft processor architecture. A soft processor is a processor that can be implemented on reconfigurable logic, e.g. an FPGA.

NIOS II PROCESSOR FEATURES

- Full 32-bit instruction set, data path, and address space
- 32 general-purpose registers
- 32 interrupt sources
- External interrupt controller interface for more interrupt sources
- Optional floating-point instructions for single-precision floating-point operations

- Access to a variety of on-chip peripherals, and interfaces to off-chip memories and peripherals
- Hardware-assisted debug module enabling processor start, stop, step, and trace under control of the Nios II software development tools
- Optional memory management unit (MMU) to support operating systems that require MMUs
- Software development environment based on the GNU C/C++ tool chain and the Nios II Software Build Tools (SBT) for Eclipse

The Nios II Processor Reference Handbook states that:

> A Nios II processor system is equivalent to a microcontroller or "computer on a chip" that includes a processor and a combination of peripherals and memory on a single chip. A Nios II processor system consists of a Nios II processor core, a set of on-chip peripherals, on-chip memory, and interfaces to off-chip memory, all implemented on a single Altera device. Like a microcontroller family, all Nios II processor systems use a consistent instruction set and programming model.
>
> — Nios II Processor Reference Handbook

Tip
For more information on the Nios II processor, consult its extensive documentation.

3.3 Design Flow

Unlike previous experiments, we need to create the hardware of the microcontroller system before we can program it.

In order to create a Nios II soft processor system on the Altera DE0-Nano FPGA board, and write software for it, you are going to use the following software tools:

Qsys
used to specify the Nios II processor core(s), memory, and other components your system requires. Qsys automatically generates the interconnect logic to integrate the components in the hardware system.

Quartus II
used to compile all design files, including those generated by Qsys, into an FPGA configuration file, known as an *SRAM Object File* (`.sof`), which can be downloaded into the FPGA device to implement the designed system.

Nios II EDS
the Nios II Embedded Design Suite includes Nios II Software Build Tools (SBT) for Eclipse, which is an eclipse installation preconfigured to use a set of plugins to support developing software for the Nios II processor. To create a new Nios II C/C++ application project, the Nios II SBT for Eclipse uses information from the files generated by Qsys to learn about the target hardware.

Hence, the general flow involves the following steps:

1. Use Qsys to generate the hardware description of your processor system. In addition to the HDL files, Qsys generates an `.sopcinfo` file that describes the system.
2. Use Quartus II to compile the hardware description into an FPGA configuration file (`.sof`), and to download the configuration file into the FPGA to implement the system's hardware.
3. Use Nios II SBT for Eclipse to develop the software for the configured hardware. Nios II SBT for Eclipse learns about the hardware from the Qsys-generated `.sopcinfo` file, in order to compile the software for the generated hardware.

3.4 Creating a Quartus II Project

We can start by creating a Quartus II project first, or by creating a Qsys project first. What matters is that the Qsys project should eventually be referenced in the Quartus II project. In our case, we will start by creating a Quartus II project.

The DE0-Nano kit ships with a convenient software utility called *System Builder*, which creates preconfigured Quartus II projects for the DE0-Nano board. For example, it automatically configures the project for the FPGA device in the DE0-Nano, and configures the pin locations for the selected peripherals.

Run the DE0-Nano's System Builder utility, and choose the following configuration options:

- CLOCK
- LED x 8
- EEPROM, 2KB
- SDRAM, 32MB

Then, press *Generate* to create a Quartus II project.

Important
Avoid using directories with spaces in their names for your Quartus II or Nios II EDS projects.

3.5 Building the Processor System Using Qsys

We would like to build a Nios II system that includes the following hardware components:

- Nios II/s core with 2 KB of instruction cache
- 20 KB of on-chip memory
- Timer
- JTAG UART
- Eight output-only parallel I/O (PIO) pins
- System ID component

Tip
For more information about these and other components, refer to the Embedded Peripherals IP User Guide.

To build this system, run Qsys from the *Tools* menu in Quartus II, and follow the instructions in the Nios II Hardware Development Tutorial, page 1-11 (*Define the System in Qsys* section).

3.6 Integrate the Qsys System into the Quartus II Project

To integrate the Qsys system with the Quartus II project, we need to:

1. Add the Qsys system to the Quartus II project
2. Instantiate the Qsys system

3. Connect the ports of the Qsys sytem to signals in the Quartus II project.

For Quartus II to be able to recognize the Qsys system, the Qsys system, represented by its Quartus II IP file (`.qip`), must be added to the Quartus II project as follows:

1. From the Quartus II menu, select *Project* > *Add/remove Files in project*
2. Click the browse button (`...`) next to the *File name* field
3. Select the file `<qsys_project_directory>/synthesis/<qsys_project_name>.qip`
4. Click *Add* to include `.qip` file in the project, then click *OK* to close the dialog box

To instantiate the Qsys-generated Nios II system, and to connect each port of the Qsys system instance to the appropriate signal in the top-level module of the Quartus II project, use the following Verilog instantiation code in the top-level module of your Quartus II project, which is typically named `<quartus_project>.v`, where `<quartus_project>` is the name of your Quartus II project.

Verilog Code to Instantiate the Qsys-Generated System

```
<qsys_project> u0(
    .clk_clk (CLOCK_50),
    .reset_reset_n (1'b1),
    .led_pio_external_export (LED)
);
```

About the Qsys-system-instantiation Verilog Code
In the Verilog code above, replace <qsys_project> with the name of your Qsys project.
The code creates an instance, named `u0`, of the Qsys system, and maps, i.e. connects, the ports of the Qsys system (the names following the periods) to signals declared in the module in which this code resides (the names between parentheses).
`1'b1` is a single-bit constant value of `1`.
The exported port names of the Qsys system are derived from the Qsys system definition.

3.7 Compile and Download the Hardware Design

The Quartus II hardware compiler consists of a set of modules that perform different compilation steps. The modules are *Analysis & Synthesis*, the *Fitter*, the *Assembler*, and the *TimeQuest Timing Analyzer*. To obtain the downloadable `.sof` FPGA configuration file, we need to run the assembler. Running the assembler will trigger all other required modules.

After compiling the Quartus II Project, connect the DE0-Nano board to your PC in order to download the hardware design.

Note
To download the FPGA configuration data file (`.sof`) to a the FPGA device, you use Altera's USB-Blaster download cable, which interfaces a USB port on your host computer to the Altera FPGA.
The USB-Blaster cable requires a driver for the host computer to recognize it. For details on using the USB-Blaster and installing its driver, refer to the USB-Blaster Download Cable User Guide.
The driver has already been installed on the lab PCs.

To download your hardware design to the FPGA:

1. Run the programmer from the *Tools* menu in Quartus II
2. Click the *Hardware Setup* button and choose *USB-Blaster* if it is not selected
3. Click the *Start* button to start downloading the `.sof` file to the FPGA chip on your board.

Important
Don't close the *OpenCore Plus Status* dialog when it appears.

Tip
For more details on downloading your design to the FPGA, refer to the *Download the Hardware Design to the Target FPGA* section of the Nios II Hardware Development Tutorial (page 1-31).

3.8 Software Development Using Nios II SBT

We will first start with a simple program to explore the software development process. We will use the *Hello World Small* template program by following the instructions on the Nios II Hardware Development Tutorial, page 1-32 (*Develop Software Using the Nios II SBT for Eclipse* section), only use the *Hello World Small* template instead of the *Count Binary* template.

Tip
The difference between the *Hello World Small* template and the *Hello World* template is that the former is configured to generate an optimized-for-space program that would fit in the small on-chip memory that was created in Qsys.
You can use the *Hello World* template instead of the *Hello World Small* template, but you would then need to adjust the properties of the BSP project in order to minimize the memory footprint of the software, as described on page 1-34 of the Nios II Hardware Development Tutorial.

To make the program slightly more interesting, replace your code with the one on page 1-9 of the My First Nios II Software Tutorial.

Important
In the function call `IOWR_ALTERA_AVALON_PIO_DATA`, replace the first argument with your system's base address of the PIO peripheral. Look the address up in your `system.h` file.

To understand how this program works, read the *Why The LED Blinks* section on page 1-10.

4 Tasks

4.1 Build and Download the Hardware Design

1. Using the *System Builder* program, create a Quartus II project for the DE-Nano board. Configure your project to use the board's CLOCK, LEDs, EEPROM, and SDRAM.
2. Build a Nios II system using Qsys.
3. Instantiate your Nios II system in the Quartus II project.
4. Compile and download the hardware design to the DE0-Nano board.

4.2 Build and Download the Software

1. Create a software project for your Nios II system using Nios II SBT for Eclipse. Use the Hello World template.
2. Run Hello World application on your Nios II system on the DE0-Nano board.
3. Create and run another application that blinks an LED on the DE0-Nano.
4. Create a third program that blinks all eight LEDs on the DE0-Nano sequentially.

4.3 Discussion

- What peripherals are readily available for inclusion in this microcontroller system? (list three)
- What peripherals would you add to your microcontroller systems?
- What is the address of your PIO peripheral, which is driving the LEDs?
- How can you change it?

5 Resources

[altera-install]
Altera Corporation. *Altera Software Installation and Licensing*. MNL-1065. 2014.12.15.
https://www.altera.com/en_US/pdfs/literature/manual/quartus_install.pdf

[nios-ii]
Altera Corporation. *Documentation: Nios II Processor*.
https://www.altera.com/products/processors/support.html

[nios-ii-ref]
Altera Corporation. *Nios II Processor Reference Handbook*. NII5V1-13.1. February 2014.
https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/nios2/n2cpu_nii5v1.pdf

[periph-ip-ug]
Altera Corporation. *Embedded Peripherals IP User Guide*. UG-01085. 2014.24.07.
https://www.altera.com/en_US/pdfs/literature/ug/ug_embedded_ip.pdf

[nios-ii-hw-tut]
Altera Corporation. *Nios II Hardware Development Tutorial*. TU-N2HWDV-4.0. May 2011.
https://www.altera.com/en_US/pdfs/literature/tt/tt_nios2_hardware_tutorial.pdf
Newer revision (Quartus II 14.0+): *Nios II Gen2 Hardware Development Tutorial*. AN-717. 2014.09.22
https://www.altera.com/en_US/pdfs/literature/an/an717.pdf

[usb-blaster]
Altera Corporation. *USB-Blaster Download Cable User Guide*. UG-USB81204-2.5. April 2009.
https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/ug/ug_usb_blstr.pdf

[nios-ii-sw-tut]
Altera Corporation. *My First Nios II Software Tutorial*. TU-01003-2.1. December 2012.
https://www.altera.com/en_US/pdfs/literature/tt/tt_my_first_nios_sw.pdf

6 Grading Sheet

Task	Points
Build the processor system using Qsys	2
Instantiate the processor system in a Quartus project	2
Run the *Hello World* program	2
Run an LED-blinking program	2
Discussion	2