Introduction to Tutorials

For the students of the "Digital Systems Design" courses, a series of 6 tutorials have been developed that want to guide, step by step, the students through the design techniques of a dedicated logic circuit on a programmable logic device (FPGA).

The tutorials are of increasing difficulty and, with the exception of the tutorial on the Modelsim system, they are designed to be carried out in sequence, in fact in each tutorial the concepts and principles developed in the previous steps are considered already acquired. On the contrary, the tutorial on Modelsim is essentially focused on the use of this tool useful for the simulation at various levels of abstraction of a logic circuit, and therefore it can be carried out at any time and requires a minimum of skills on the analysis of sequential logic circuits. All tutorials are widely accompanied by a vast bibliography, useful for any in-depth study of the topics covered, and also in this case the bibliography of each tutorial is not to be considered complete and limited to the tutorial itself, but it must be integrated with that of the previous tutorials and constitutes an integral part of the bibliography for the tutorials to follow.

All tutorials have been developed to be used directly on the "Terasic DE1-SoC" development board that mounts an Altera Cyclone V FPGA device as well as numerous other peripherals. Their use on different development boards such as the DE0-Nano requires appropriate variations in specifications and constraints.

The first three tutorials are oriented to the creation of dedicated logic circuits

In the **first** tutorial you will learn the rudiments of description, simulation at various levels of abstraction and the synthesis of a very simple logic circuit, up to verifying its operation on the development board. The circuit proposed in this case uses two buttons through which to act separately to turn on and / or off a LED placed on the development board. In this tutorial we will also lay the first foundations for the use of the Verilog HDL language for the description and synthesis of logic circuits.

In the **second** tutorial a more complete project will be developed centered on the creation of some adders, comparing their characteristics. This tutor will deepen the use of the "System Builder" as an interface for the realization of complex circuits, will analyze the realization of a system at a hierarchical level and will deepen the aspects related to the simulation of the same. We will also consider the critical issues related to the propagation times of the signals and how to analyze and manage them through the creation of specific circuits and with the imposition of appropriate time constraints.

In the **third** tutorial we will create a simple "frequency meter" or a system capable of estimating the period of a periodic digital signal and evaluating its "duty cycle", displaying the values in real time on the seven-segment display of the board. In this tutorial we will deepen the issues related to the creation of meters, access to peripherals (such as the GPIO and the display with 7-segment LEDs). We will see how to create a hierarchical system using different input tools, as well as an in-depth study of the Verilog code. In addition, simple devices external to the board (NE555-based oscillators) will be used to generate signals to be used as inputs using the appropriate interfaces. In addition, the "In System Source & Probe" tool will be used to control and observe the signals inside the FPGA.

In the **fourth** tutorial we will configure the board to be able to drive an external LED matrix display according to the SPI protocol. The purpose of this tutorial is to familiarize ourselves with external interfaces and with the generation of signals that underlie particular specifications through the definition of a system suitable for the purpose. The use of some tools, developed within Quartus, will also be deepened: the first called "Signal TAP Logic Analizer" useful for monitoring the temporal evolution of signals inside and outside the FPGA, the second, called "In-System Memory Editor", useful for interfacing with any memories to write or read their content.

The following tutorials are instead more oriented to the creation of "embedded" systems and computer architecture: The FPGA mounted on the DE1-SoC board has in fact sufficient hardware resources to be able to create, exploiting its internal logic blocks, both a processor (soft processor) configurable *ad hoc* by the user (NIOS II) both in specifications and in the structure of the peripherals, and a dual core ARM hardware processor (hard processor) with several peripherals already made on silicon.

In the **fifth**  tutorial you will create simple systems based on the Nios-II processor capable of communicating with some peripherals. In particular, we will see how to program (in C) and how to debug such a system. You will learn how to design Hardware blocks designed ad hoc to be interfaced with the NIOS processor in order to create a series of blocks dedicated to various functions that can be easily reused within specific projects.

In the **sixth** tutorial we will extend what we have learned in the use of the ARM processor and its perifiers and we will learn how to create a complete system that makes use of this processor and opt for customized peripherals.

It is suggested to carry out the tutorials independently and completely. The texts and additional software material available on the course page are fully sufficient for the purpose. The VerilogHDL and C codes are in fact reported in extended form in order to allow a simple operation of (Cut & Paste) while maintaining all the operations to be followed step-by-step, which allow the student to appropriate the specific procedure to achieve the result.

There is also available (in compressed form) the complete project developed in the various tutorials in question, made through the "Quartus II v22.1 Web edition" system, but it should be emphasized that what is available is a "frozen" version of the project in its final form and therefore lacks all the steps and variants that could have been the subject of in-depth study during the tutorial itself.