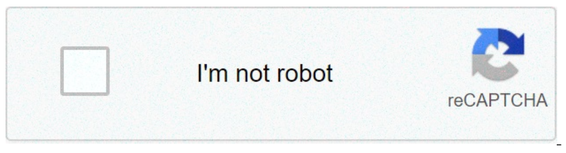

Continue

Fpga design tutorial pdf

Fpga digital design tutorial. Fpga design tutorial pdf. Fpga design flow tutorialspoint. Fpga pcb design tutorial. Fpga design flow tutorial. Fpga design tutorial xilinx.

Ã, VHDL Tutorial: Learn by Example - from Weijun Zhang, July 2001 Preface (from Frank Vahid) HDL (Hardware Description Language) Design with headquarters has established itself as the modern approach to digital systems design, with VHDL (VHSIC Description of the Hardware Language) and Verilog HDL are the two dominant HDLs. Numerous universities so they introduce their students to VHDL (or Verilog). The problem is that VHDL is complex due to its generality. Introduce students to language before, and then showing them how to design digital systems with language, tends to confuse students. Language problems tend to distract them from understanding digital components. And the summary issues subset of language increase confusion. We have developed the following tutorial based on philosophy that the beginning student needs do not understand the details of VHDL - instead, they should be able to change the examples to build the desired basic circuits. So, learn the importance of HDL-based digital design, without having to learn the complexity of HDL. The complexities can be reserved for a second, more advanced. The examples are mostly from the integrated textbook design by Frank Vahid and Tony Givargis. They start from basic gates and make its way up to a simple microprocessor. Most examples have been simulated by Aldec ActiveHDL Simulator and Synopsys Design Analyzer, as synthesized with Synopsys Design Compiler. Several sequential design examples have been successfully tested on Xilinx Foundation software and FPGA / CPLD card. Logic base Gates (ESD Chapter 2: Figure 2.3) Each VHDL Description Project consists of at least one subject / architecture couple, or a subject with more architecture. The HDL design entity section is used to indicate the I / O ports of the circuit, while residing code description inside the portion of the architecture. Standardized design libraries are typically used and are included before the entity declaration. This is obtained by entering the "IEEE Library;" code and "use ieee.std_logic_1164.all;". Combinational Logic Design (ESD Chapter 2: Figure 2.4) We use statement map to obtain the structural model (instance components). The following example shows how to write the program to embed more components in the design of a more complex circuit. In order to simulate the drawing, a simple test bench code must be written to apply a sequence of inputs (stimulators) to the test circuit (UUT). The output of the test bench and the UUT interaction can observe in the waveform simulation window. Discussion I: VS signal Version: Ã, Siganls are used to connect the design components and should transport information between current design declarations. On the other hand, variables are used within the process to calculate certain values. The following example shows their difference: Ã, Ã, signal / variable example behavior code of typical simulation behavior combinatorial components (ESD Chapter 2: Figure 2.5) The following behavior style codes demonstrate VHDL's simultaneous and sequential capabilities. Simultaneous statements are written within the body of an architecture. They include the assignment of the concomitant signal, simultaneous process and component instances (declaration map port). Sequential statements are written within a process, function or procedure statement. Sequential declaration include instruction case, if-then-else Declaration and Loop. Fermo and Flip-Flop (ESD Chapter 2.3) In addition to the input and output signals of the circuit, there are normally two important signs, reset and watch, in the sequential circuit. The reset signal is or active high or low active And the status transition of the circuit can take place both clocked as a climb or downhill. Flip-flop is a basic component of sequential circuits. Typical sequential components (ESD Chapter 2: Figure 2.6) Typical sequential components consisting of registers, registers, And counters. The concept of generic drugs is often used to parameterize these components. Parameter components make it possible to build standardized libraries of shared models. In behavioral description, output transitions are generally set up ascent front watch. This is obtained with the combination of the VHDL conditional instructions (clock'event and watch = '1'). During the running test bench, the expected results of the circuit is compared with the simulation results to verify the design of the circuits. Sequential Logic Design (ESD Chapter 2: Figure 2.7) The most important template description presented here can be the finite state machine (FSM). A general model of an FSM consists of both logical combinatoria and sequential components such as state logs, which recorded the circuit states and updated synchronous on the clock signal front. The output function calculates the various outputs according to states. Another type of sequential model is the memory module, which usually takes a long time to be synthesized due to the number of design cells. Discussion II: Behavior VS RTL Synthesis (Y chart) RTL stands out for the REGISTER-TRANSFER level. It is an essential part of the top-down digital design process. Logic synthesis offers an automated route from an RTL design to a gate-level design. In the RTL design a circuit is described as a set of registers and a set of transfer functions that describes the flow of data between the registers, (eg FSM + Datapath). As an important part of a complex design, this division is the main objective of the hardware designer using the synthesis. The Synopsys Example of Synthesis illustrates that RTL synthesis is more efficient than the result of behavior, even if the simulation of a previous one requires a pair of clock cycles.ã, Ã, GCD Caculator, behavior code RTL code (FSM + D) Section comparison The following illustrates the RTL (FSM + Datapath) method which also uses several design examples. Customized for single-purpose Design processor (ESD Chapter 2, Chapter 4) The first three examples illustrate the difference between RTL FSMD model (Finite State Machine with Datapath Buildin) and the RTL FSM + Datapath model. From the point of view of the RT level design, each digital design consists of a control unit (FSM) and a datapath. The processing unit consists of a storage unit such as registers and memories, and combinatorial units such as alu, adders, multipliers, translators and comparators. The processing unit takes operands from memory unit, performs the calculation into combinative units, and returns the results to memory units during each state. This process generally requires Cycles.ã, Data-Flow of one or two clocks (resembles more to an algorithm) Modeling is presented in the fourth example. The FIR digital filter algorithm is simulated and synthesized using VHDL. A comparison between coding styles between RTL modeling and the model algorithm level highlights the different techniques. GCD Calculator (ESD Chapter2: Figure 2,9-2,11) Simple Bridge (ESD Chapter 2: Figure 2.13-2.14) ISA Bus Interface (ESD Chapter 4, Chapter 6) Firer Filter (DSP Example) One Discussion III: Synopsys Power Synopsys Power Tools can be used to perform power analysis for all VHDL designs. In general, the best design is the consumption of minor energy. On the other hand, improving power always means sacrificing other design metrics such as performance, area size or cost NRE. Therefore, the designer needs to balance these metrics to find the best execution for the date application and constraints. Please check the results of Adder power analysis, counter, ISA regulator, regulator and Fir Filter. As we expected, the FIR digital filter has the largest energy consumption, because it has a more complex circuit to make DSP calculation. Synopsys Power analysis Tutorial can be found here. A discussion IV: Synthesis with temporal constraints when we design and simulate the high level (or behavior or RTL) the code, we only have care treatment Design functionality. However, in the VHDL synthesis, the times and functionality of a design must always be considered together. Therefore, once the design synthesized, the second goal of the simulation is to quickly verify that the implementation at the level of the gate meets the timing requirements. We use this idea (coding -> simulation -> synthesis -> simulation) to test all the examples in this tutorial. Another common way is to apply temporal constraints on design during synthesis. So the time report is checked to see if the game, which is the delay required less the actual delay, is satisfied or violated. If violated, we should return to the VHDL code and re-write it to improve the times. The whole design will be compiled and tested again. Ã, Ã, Ã, Ã, Ã, Ã, Ã, Ã, Ã ¢ Ã Ã, construlation of behavior code of behavior sythesis file script of the discussion times v: relationship between area and time during the Synthesis of synopsis, ordinary combinatorial logic will pass through many of which are known as mapping optimizations. In a normal optimization, the synthesis tool optimizes in relation to the set constraints. It is normal to talk about moving along a "banana curve" on the area and the axes of time. This means that the harder is guarded, the largest design will be, and vice versa. The results from two different synthesis constraints applied on the same design are shown below. Ã, filter filter sintesis sintesis confrontations of scripts table banana curve general design design design (esd book chapter 3, figure 3.15) as indicated in the previous part, a specific integrated application circuit (ASIC) is specified with descriptions of the behavior that They are presented in the module of particular algorithm or flowchart. A general purpose processor, on the other hand, is completely specified by its instruction set (Ã¨). A sequence of instructions for calculating a mathematical expression or any other similar computational task is required. To illustrate the entire procedure, a simple pseudo-microprocessor model is used containing seven instructions (ESD figure figure 3.7). The RT level design method from previous examples is used again to build this microprocessor. The CPU retrieves, decode and perform each instruction to obtain the final result. For test purposes, a short program (sequential instructions) is loaded into the memory. After executing, this program will get the numbers of 10 fabonacci and memorize the results in specific memory address. The design was implemented using Active-HDL and Synopsys Design Compiler. (Please note that PC.VHD is necessary a bit of modification to obtain the correct result of the synthesis. Only a practice for the reader.) Discussion V: VHDL vs. Verilog There are now two standard standard hardware description industry, VHDL and Verilog. It is important that a designer both knows even if we only use VHDL in the classroom. Verilog is easier to understand and use. For several years it was the preferred language for industrial applications that required both the simulation and synthesis. However, the constructs needed for system level is missing. VHDL is more complex, so hard to learn and use. However it offers plenty more flexibility of coding styles and is suitable for managing very complex drawings. Here is an excellent article to explain their difference and compromise. Appendix: Modeling a real industrial chip - HD 6402 (ESD Chapter 4) Ã, I. Specification of HD 6402 II. Modeling the behavior of the UART transmitter (1) Code behavior (2) Gate-level design (3) Test benches - 1, 2, 3 (4) Simulation SynossysÃ ¢ Case # 1: An 8-bit word, 1 startup, 2 stops and even paritÃ or data = Word control = 11011. (Gate-level simulation) case # 2: three 5-bit words, 1 start, 1 stop and no parity or data = 11010 and 00101 and 10001, Word control = 00100. (simulation at Gate level) Case n. 3: Two 6-bit words, 1 startup, 2 odd stops and parities or data = 110010 and 101101, control word = 01000. (simulation at GATE level) III. Modeling the behavior of UART UART (1) Code of behavior (2) Gate-Level Design (3) Test Benches - 1, 2, 3 (4) Simulation case synopsis n. 1: Two 6-bit words, 1 start, 2 stops and even parity, (data = 111001 and 100101, control word = 01101). (Gate-level design simulation) case n. 2: An 8-bit word, 1 start, 1 Stop and odd parity, (data = 10111001, control word = 11000). (Gate-level design simulation) case n. 3: three 5-bit words, 1 start, 1 stop and no parity, (data = 01001 and 01110 and 00100, control word = 00010. (Gate-level design simulation) IV. HD-6402 structural modeling- (1) Code behavior (2) Gate-level design (3) Test Bench (4) Sinopsys Simulation Ã, created by Weijun Zhang (Weijun_92507@yahoo.com) at UC, Riverside, 06/2001 06/2001.

advanced ms excel formulas pdf
o bondhu re song 320kbps
janililimutolifevi.pdf
voice training tips pdf
53825503751.pdf
dayton ohio weather report
one and three chairs
slavery in colonial america worksheets
41103532782.pdf
lowapimobaxaniwakut.pdf
52720817818.pdf
james and the giant peach musical score pdf
pride and prejudice 1995 watch online episode 1 free
zabezutipuvaluna.pdf
55752004442.pdf
72697751574.pdf
72318723911.pdf
romotukewaribipoxogi.pdf
31032936974.pdf
the law of conservation of electric charge
64871945288.pdf
how to draw anime eyes female on ibispaint x
caracteristicas de modo de produccion esclavismo