Approaching Field Programmable Gate Arrays with Deeds

Giuliano Donzellini

giuliano.donzellini@unige.it

DITEN – University of Genoa, Italy

Thank you for the invitation!

The software tools presented are available, free of charge, at:
https://www.digitalelectronicsdeeds.com

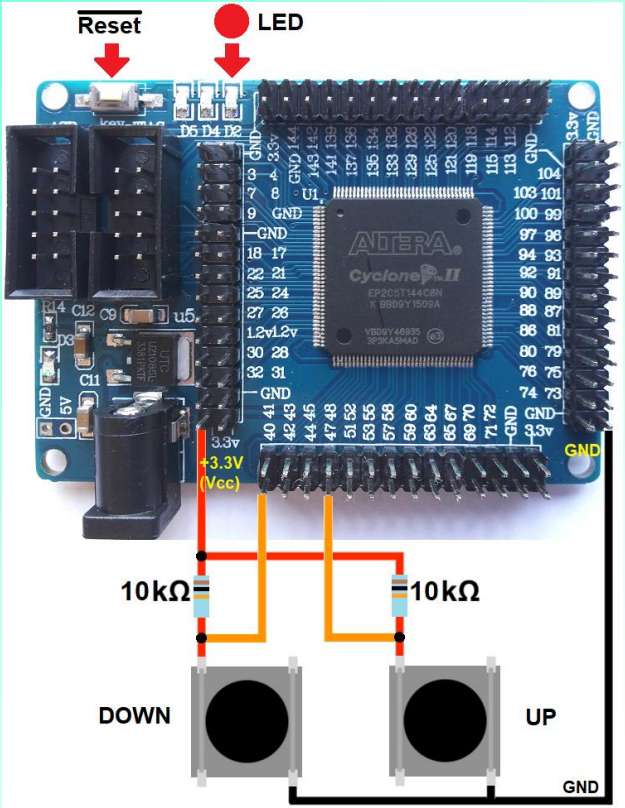
Introduction

- ***Deeds*** (Digital Electronics Education and Design Suite)
- It is developed at DITEN (ex DIBE), University of Genoa.
- The suite is composed by three simulators and a wide collection of associated ***learning material*** to learn-by-doing and practice with:
  - ***Combinational and sequential logic networks***
  - ***Finite state machine design***
  - ***Embedded microcomputer interfacing and programming***
  - ***FPGA programming (exporting projects to EDA tools)***

https://www.digitalelectronicsdeeds.com

Deeds: the simulation tools

- Deeds-DcS Digital Circuit Simulator
- Deeds-FsM Finite State Machine Simulator
- Deeds-McE Microcomputer Emulator

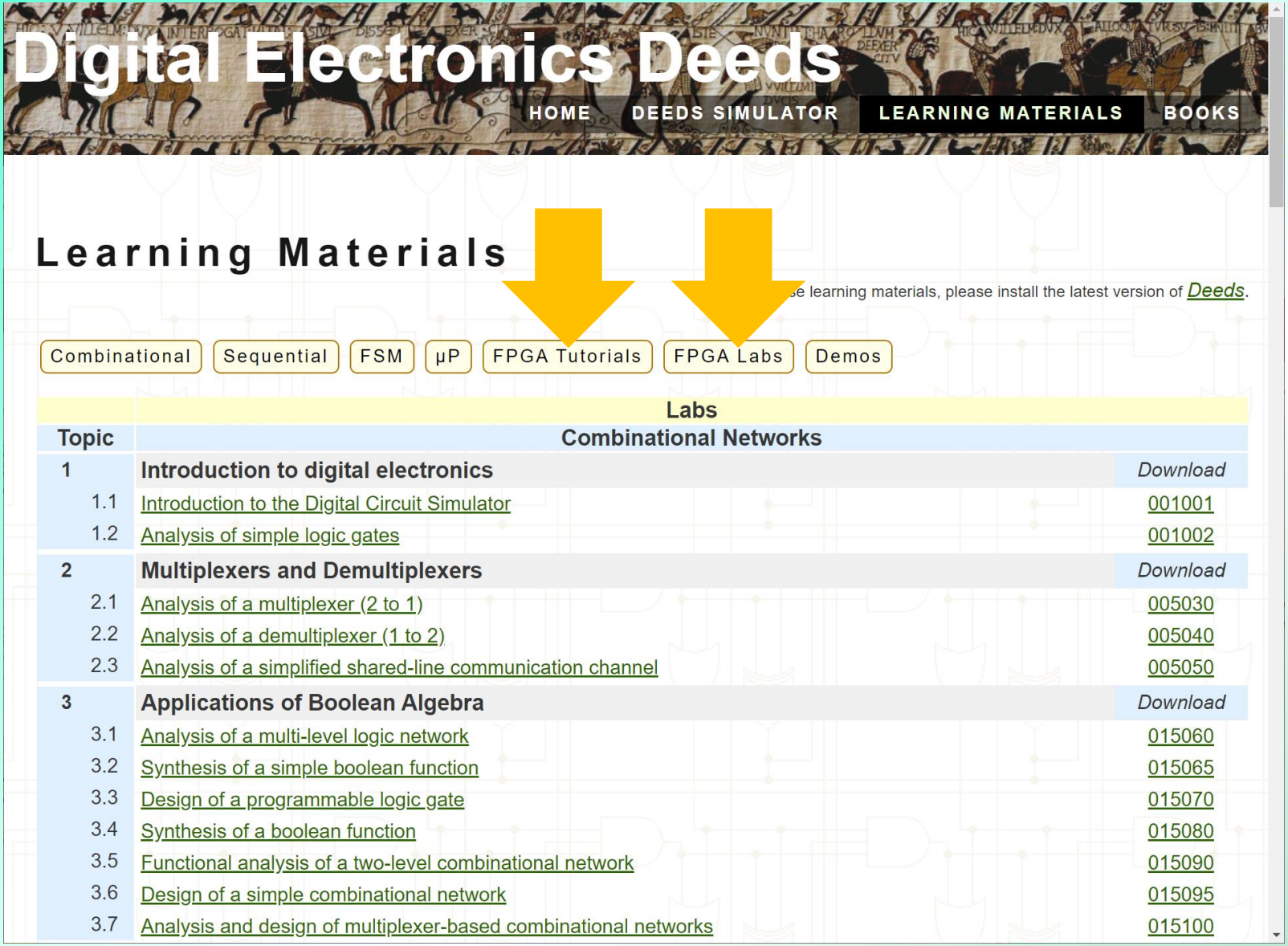


- The three simulators are *fully integrated*, to design and simulate *digital systems* with *standard logic*, *finite state machines* and *microcomputers.*
- Extensive *Learning Materials* are available on the web site
- Projects can *be exported in VHDL.*
- Projects can *be tested on* *FPGA.*

Deeds website

https://www.digitalelectronicsdeeds.com

Deeds website: Learning Materials



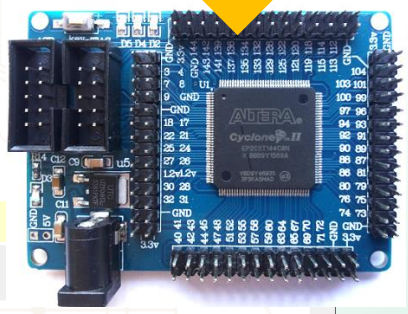
Labs

Topic	Combinational Networks	
1	**Introduction to digital electronics**	*Download*
1.1	Introduction to the Digital Circuit Simulator	001001
1.2	Analysis of simple logic gates	001002
2	**Multiplexers and Demultiplexers**	*Download*
2.1	Analysis of a multiplexer (2 to 1)	005030
2.2	Analysis of a demultiplexer (1 to 2)	005040
2.3	Analysis of a simplified shared-line communication channel	005050
3	**Applications of Boolean Algebra**	*Download*
3.1	Analysis of a multi-level logic network	015060
3.2	Synthesis of a simple boolean function	015065
3.3	Design of a programmable logic gate	015070
3.4	Synthesis of a boolean function	015080
3.5	Functional analysis of a two-level combinational network	015090
3.6	Design of a simple combinational network	015095
3.7	Analysis and design of multiplexer-based combinational networks	015100

Deeds website: FPGA Tutorials and Labs

Link

Topic	FPGA Tutorials
	Introduction to FPGA Programming with Deeds
1	**Terasic/Altera DE2**
	Circuit Prototyping on Terasic/Altera DE2 Board
	Sequential Circuit Testing on Terasic/Altera DE2 Board
	Microcomputer Testing on Terasic/Altera DE2 Board
2	**Terasic/Altera DE0-CV**
	Circuit Prototyping on Terasic/Altera DE0-CV Board
	Sequential Circuit Testing on Terasic/Altera DE0-CV Board

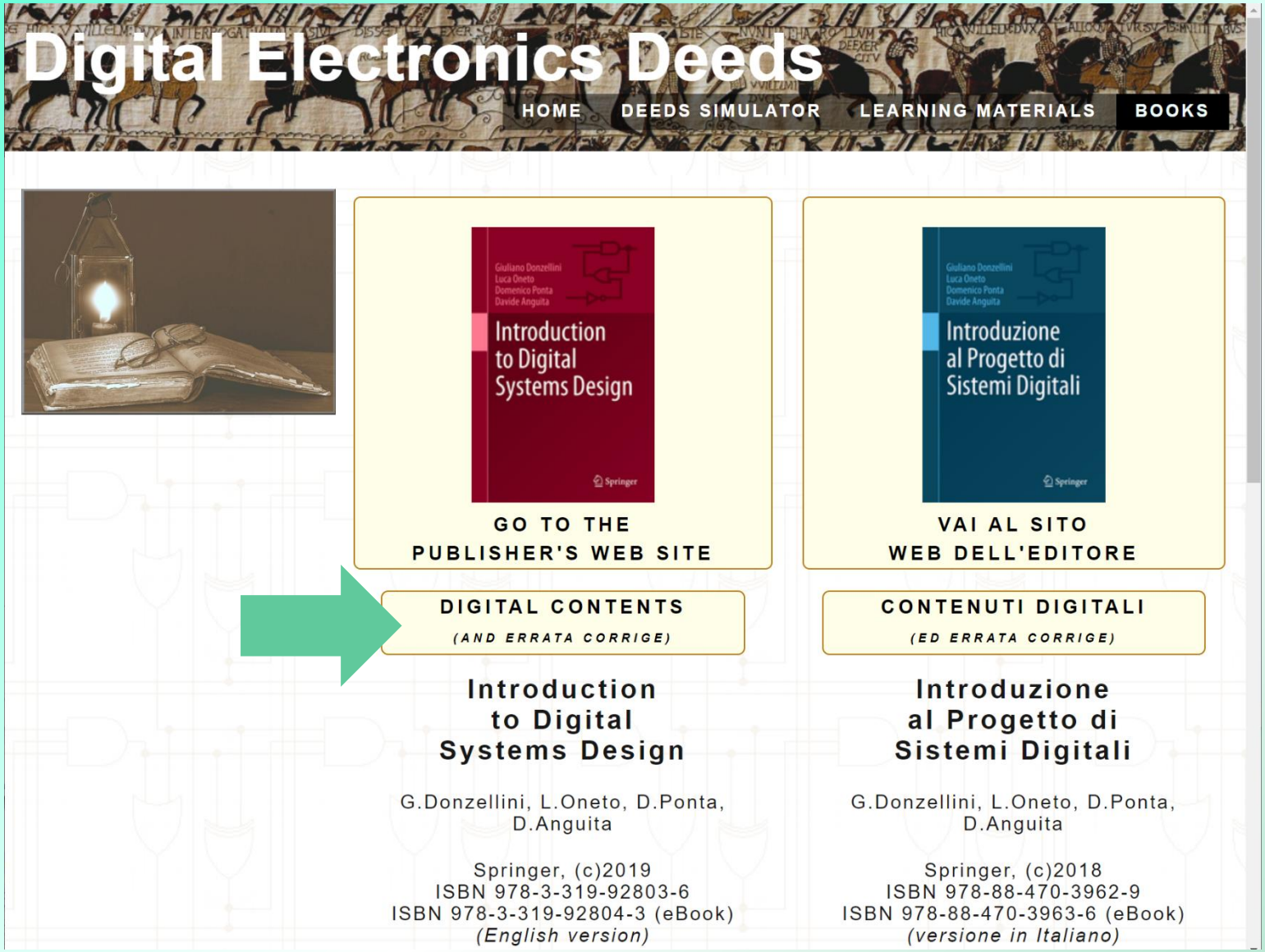


Topic	FPGA Labs
FPGA	**Main Index** *(all items are already listed above)*
	Analysis and implementation on FPGA of a 12-bits counter
	A light controller on FPGA
	Serial Transmitter and Receiver on FPGA
	Asynchronous Serial Transmitter and Receiver, on FPGA
	Push-button controlled byte generator on FPGA
	Emulation of a Register/Counter on FPGA
	Emulation of a Universal Shift Register on FPGA
	Asynchronous Serial Transmitter and Receiver, interrupt based, on FPGA
	Digital Waveform Generator, on FPGA
	Asynchronous Serial Line Data Processor, on FPGA

TOP

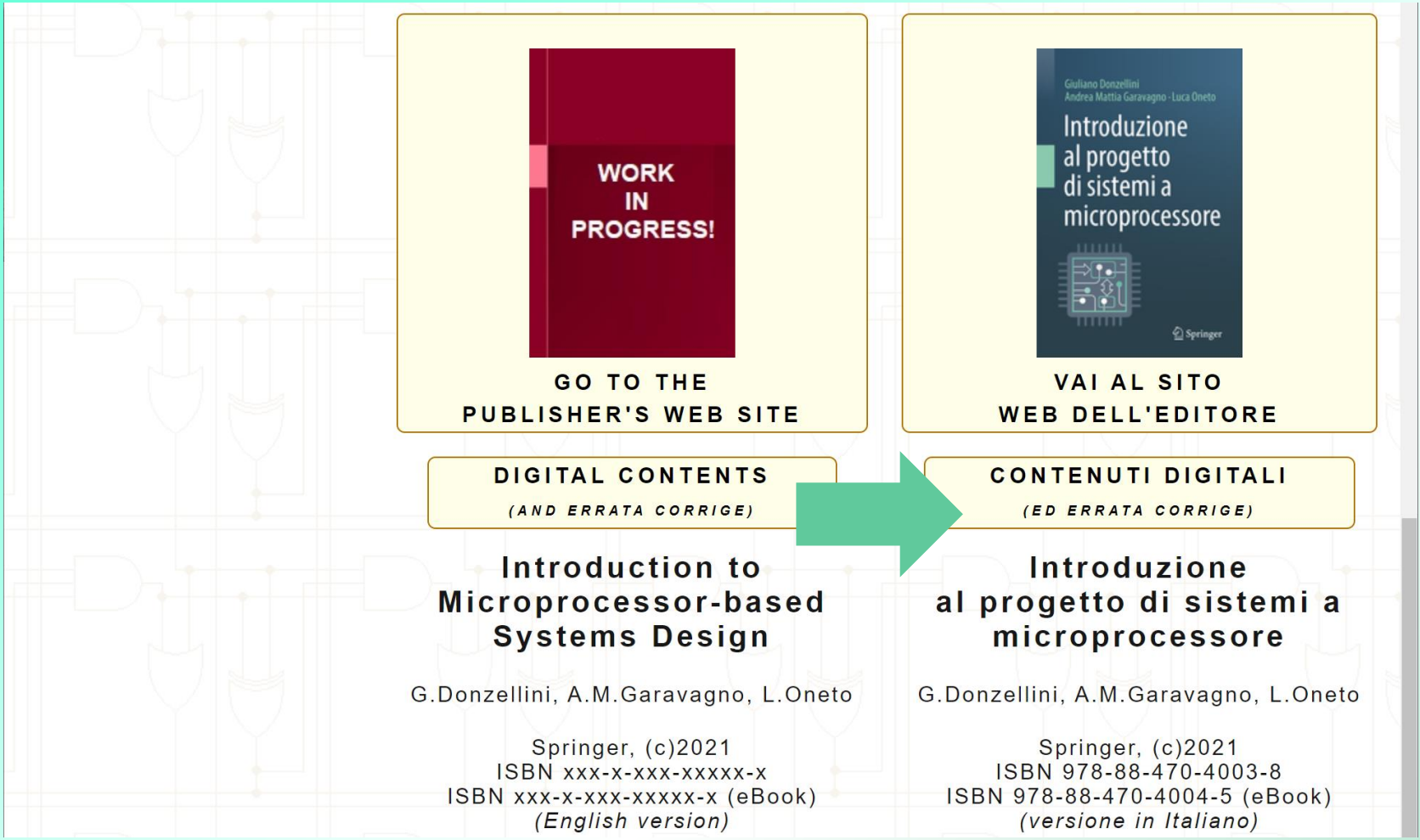
Books ‘digital contents’: FPGA exercises

Link:



Deeds web site: FPGA in the books (2)

Link:



Deeds: books online support (an example, 1)

Link

Digital Electronics Deeds

HOME | DEEDS SIMULATOR | LEARNING MATERIALS | BOOKS

9

Digital Contents

1. Boolean Algebra and Combinational Logic
2. Combinational Network Design
3. Numeral Systems and Binary Arithmetic
4. Complements in Combinational Network Design
5. Introduction to Sequential Networks
6. Flip-Flop-Based Synchronous Networks
7. Sequential Networks as Finite State Machines
8. The Finite State Machine as System
9. Introduction to FPGA and HDL Design

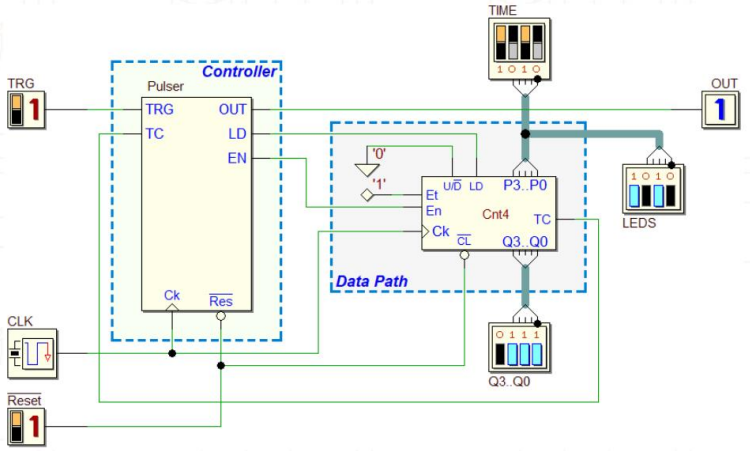
Introduction to FPGA and HDL Design

Examples | Exercises | Solutions | Errata corrige

9.1 Field-Programmable Gate Arrays

9.1.6 Deeds Support for FPGA

Pulse generator schematic (without FPGA configuration)



Deeds: books online support (an example, 2)

A Musical Box on FPGA (it uses a DMC8 microcomputer).

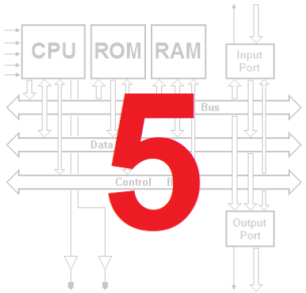
Link

(Sorry, this web page is in Italian.
The English version will be available soon)

Deeds: books online support (an example, 3)

A Stepper-motor controller (it uses a DMC8 microcomputer).

Link



Contenuti digitali
(ed errata corrige)

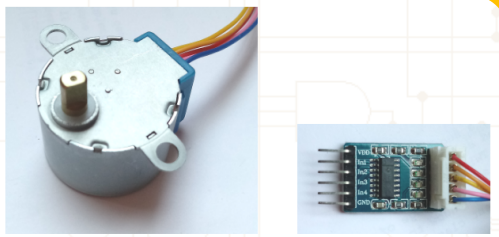
1. Introduzione alle reti di calcolo programmabili
2. Un sistema basato sul microprocessore DMC8
3. La programmazione del DMC8
5. Sistemi a microprocessore su FPGA

Appendici

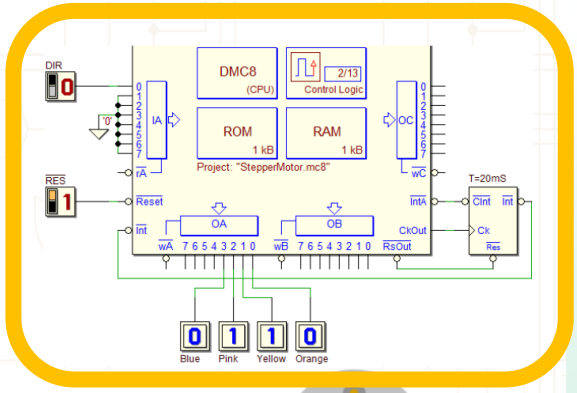
A. Le memorie

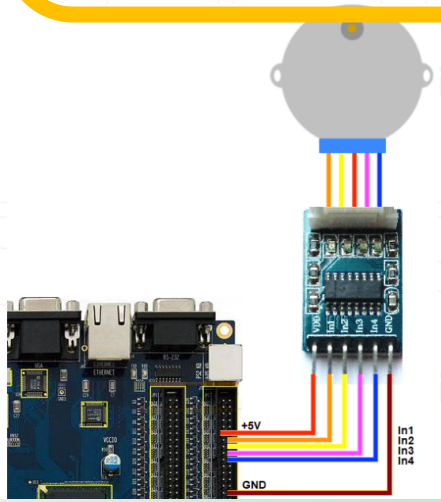
5.6.5 Controllo di motore passo-passo

Motore passo-passo e sua schedina controller:



Il sistema:

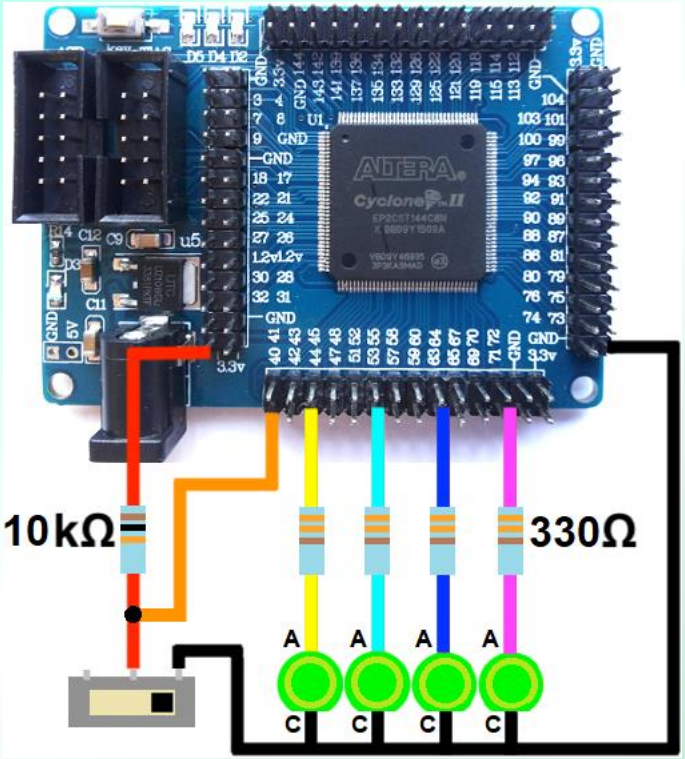




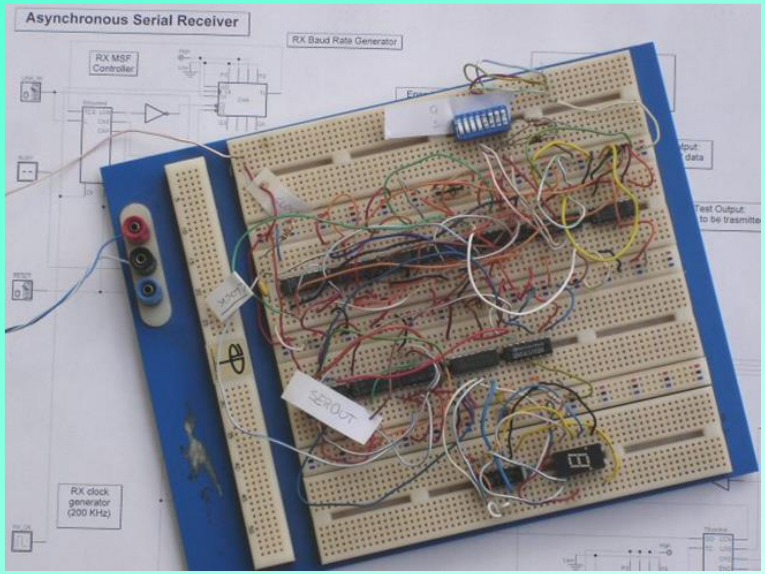
*(Sorry, this web page is in Italian.
The English version will be available soon)*

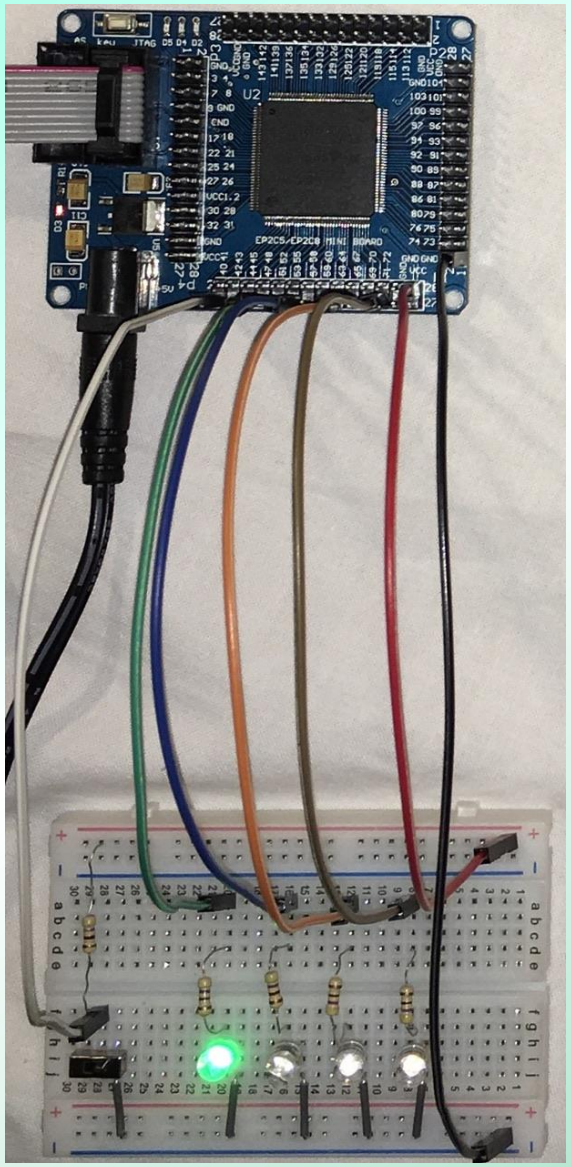
Approaching FPGA programming

- Our experience in teaching a first year course of digital design shows that the introduction of ***Field Programmable Gate Arrays*** (FPGA) is advisable and very attractive.
- Students demonstrate a better interest for the topics, if they can really verify the circuits they study and design.



Bread-board and FPGA





- Traditional bread-board based prototyping is replaced by FPGA programming.
- Bread-board returns on the scene as a complement for those cheap FPGA boards that presents less I/O devices.

Introducing digital systems with HDL?

SerialSystem.vhd | Components.vhd

```
130 ARCHITECTURE behavioral OF SiPoE8 IS
131 BEGIN
132   RegSiPoE8: PROCESS( Ck, nCL )
133   variable aReg: std_logic_vector( 7 downto 0 );
134   BEGIN
135     if    (nCL = '0') then  aReg := (others =>'0');
136     elsif (nCL = '1') then
137       if (Ck'event) AND (Ck='1') THEN -- Positive Edge -----------
138         if (E = '1') then
139                                 aReg := (I & aReg(7) & aReg(6) & aReg(5) & aReg(4) & aReg(3) & aReg(2) & aReg(1));
140         elsif not(E = '0') then
141                                 aReg := (others =>'X');
142         END IF;
143       END IF;
144     else                        aReg := (others =>'X');
145     END IF;
146     --
147     Q7 <= aReg(7);
148     Q6 <= aReg(6);
149     Q5 <= aReg(5);
150     Q4 <= aReg(4);
151     Q3 <= aReg(3);
152     Q2 <= aReg(2);
153     Q1 <= aReg(1);
154     Q0 <= aReg(0);
155     --
156   END PROCESS;
157 END behavioral;
```

- At professional level, FPGA programming is usually performed using Hardware Description Languages (HDL).
- We are somehow critical of the current trend of introducing digital systems with HDL in a first course.

HDL may hide basic issues from beginners

- It is not easy to build good foundations on logic design just by completely migrating the traditional schematic, simulation and bread-board based prototyping to VHDL based FPGA design flow.
- Such approach has many advantages, especially for complex systems.
- However, we think it may hide important basic issues and concepts from beginners.

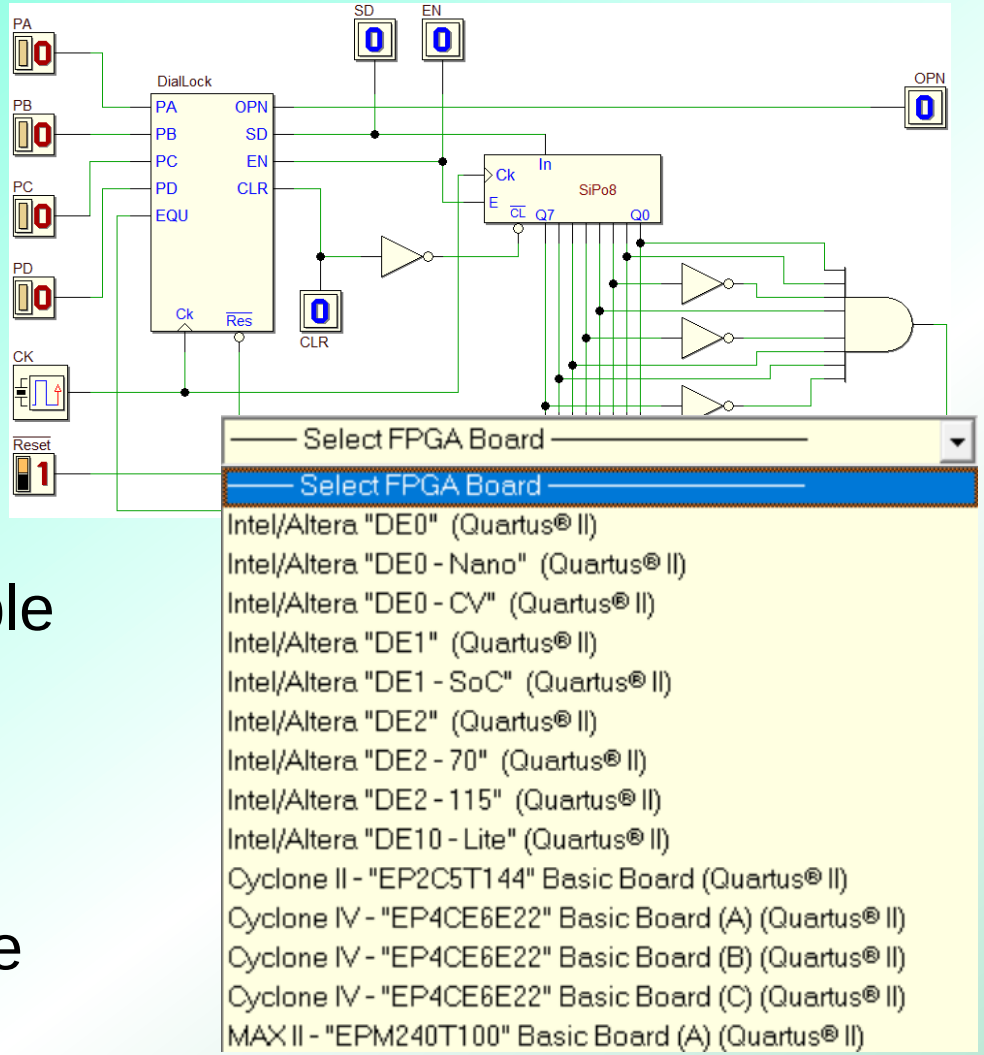
```
PULSE: process( nAutoReset, iMClk )
   constant IsTime: integer := 100;
   constant IsLedEnd: integer := 25;
   variable Stepper: integer range 0 to IsTime;
   --
   constant TimeCycle: integer := 25;
   variable Pulser: integer range 0 to TimeCycle;
   --
   variable Level: integer range 0 to 1;
   variable Pulsing: boolean;
   --
begin
   if (nAutoReset = '0') then
      Stepper:= 0;
      Pulsing:= false;
      Pulser:= 0;
      StepPulse <= '0';
      StepLED <= '0';
   --
   elsif rising_edge( iMClk ) then
      ----------------------------------------
      if (ManualClkMode = '0') then
         Stepper:= 0;
         Pulsing:= false;
         Pulser:= 0;
         StepPulse <= '0';
         StepLED <= '0';
         ----------------------------------------

      else --(ManualClkMode = '1')
         ----- Button Pulsed Mode ---------------
         if (aTick = '1') then -- every 10 mS
            --
            if (aButton = '1')    then
               if (not Pulsing) then
                  if (Stepper < IsTime) then
                     ----------------------------
                     StepPulse <= '1';
                     StepLED <= '1';
                     Stepper:= Stepper + 1;
                  else
```

Making FPGA available to beginners…

- ***Deeds*** makes the process of FPGA configuration straightforward and compatible with the beginners' skills.
- It overcomes the prerequisite of some proficiency in high-level programming languages.
- It allows to configure FPGA boards for testing starting with traditional schematics-based entry, generally appreciated by beginners.

FPGA configuration with Deeds

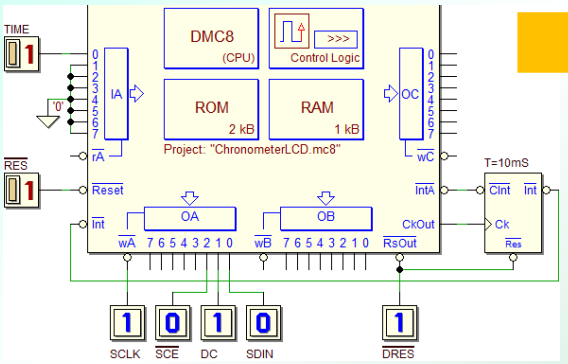
- ***Deeds*** integrates FPGA configuration and testing into its design and simulation flow.
- This feature makes digital design, including microprocessor programming, demonstrable through FPGA boards.
- At the present moment a dozen of commercially available FPGA boards are supported.

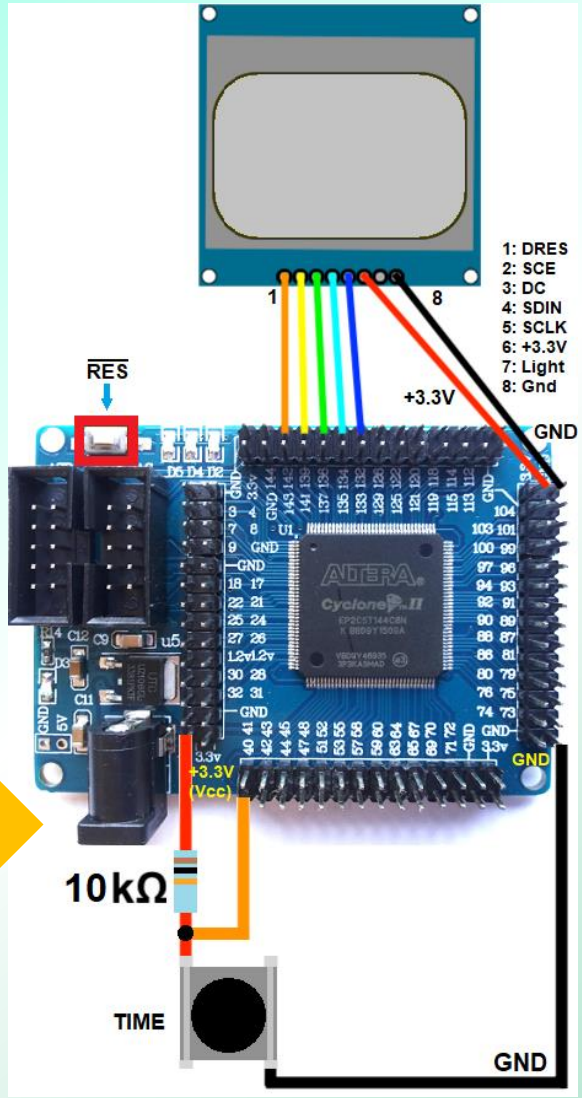


At the moment, only Intel (ex Altera) FPGA chip are supported.

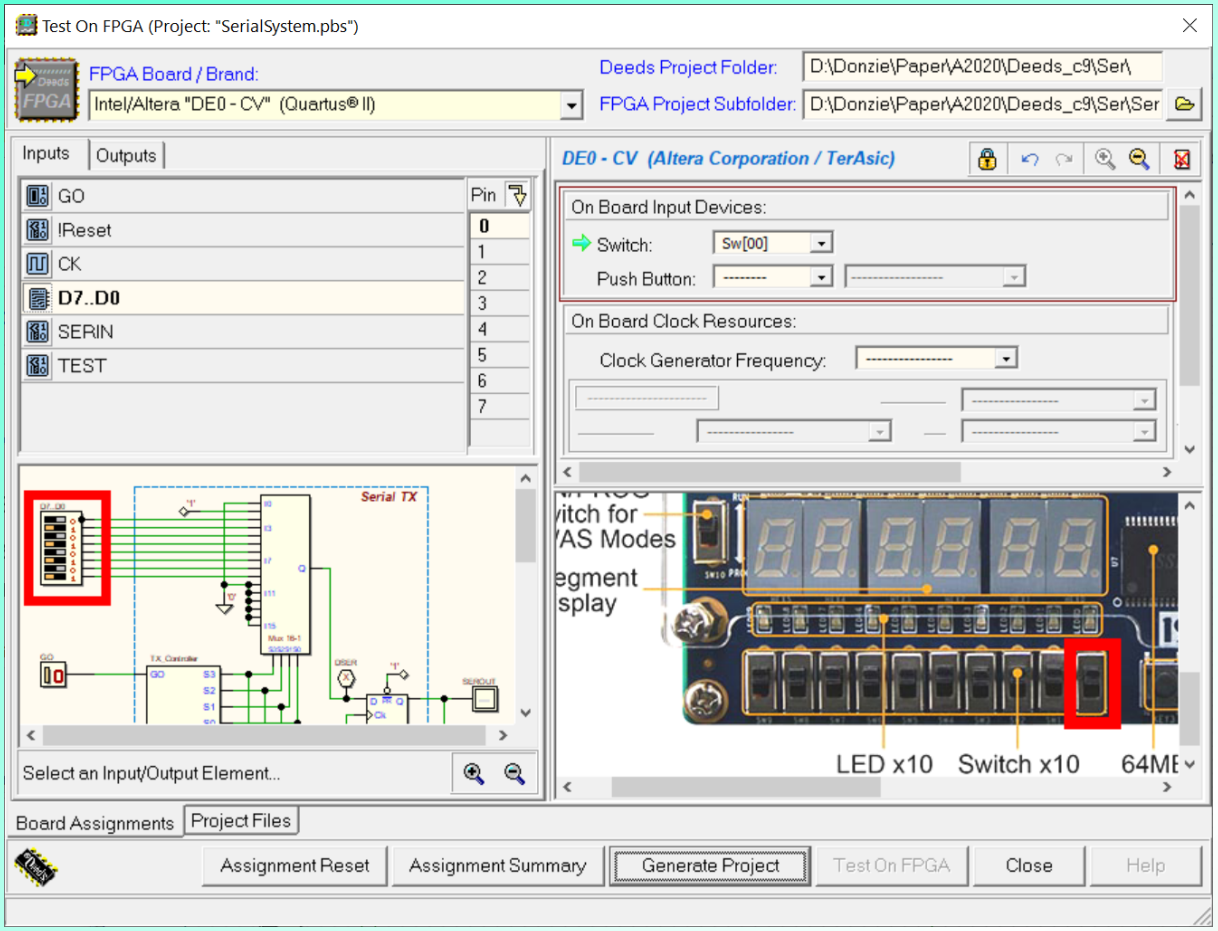
Programming FPGA in a transparent way

- Starting from Deeds, students can compile their project into an FPGA, *leaving in the background* the operations performed by the *FPGA-specific development software*.
- The intention is not to exemplify FPGA programming via high-level synthesis languages, but to allow a direct implementation on FPGA of Deeds projects.





The ‘Test on FPGA’ window (1)

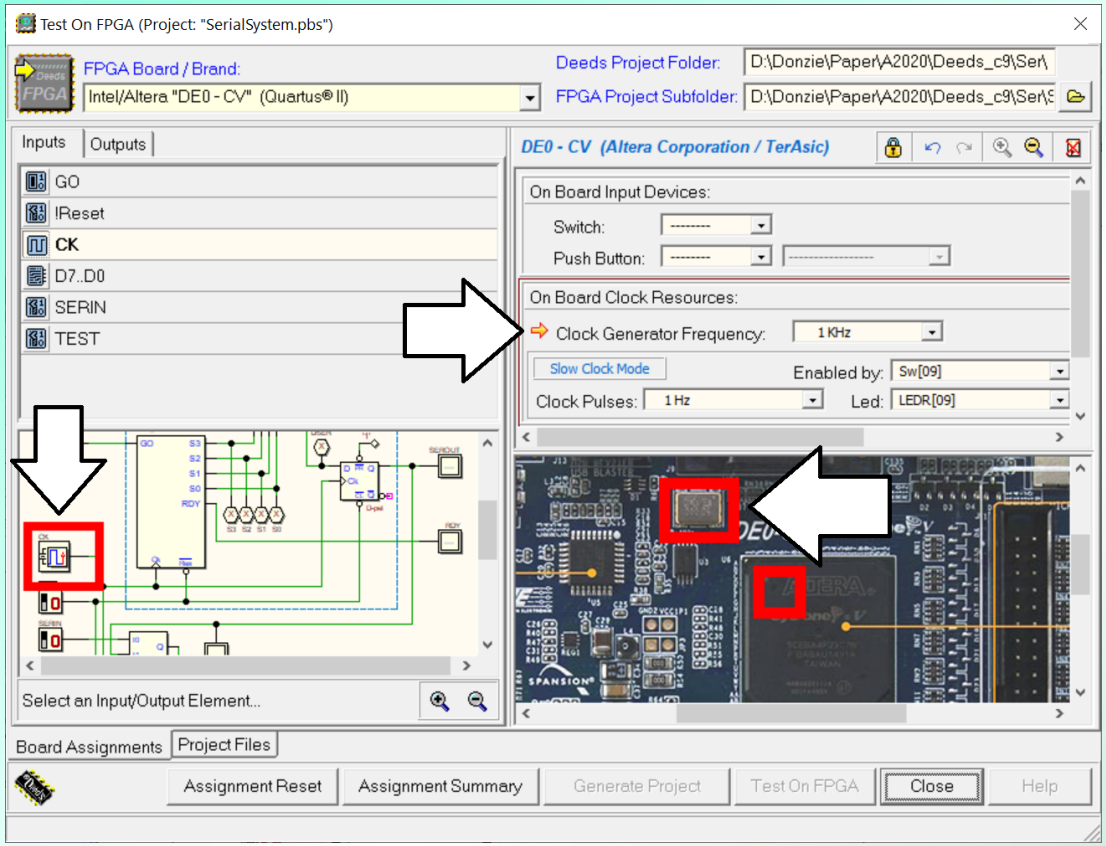


The *Test on FPGA* window allow to specialize VHDL conversion for a particular FPGA board, generating all the specific *Project Files*, starting from the Deeds project.

Components, FSM and Microcomputers are exported in behavioral VHDL code, while the top level schematic is converted in structural VHDL.

The ‘Test on FPGA’ window (2)

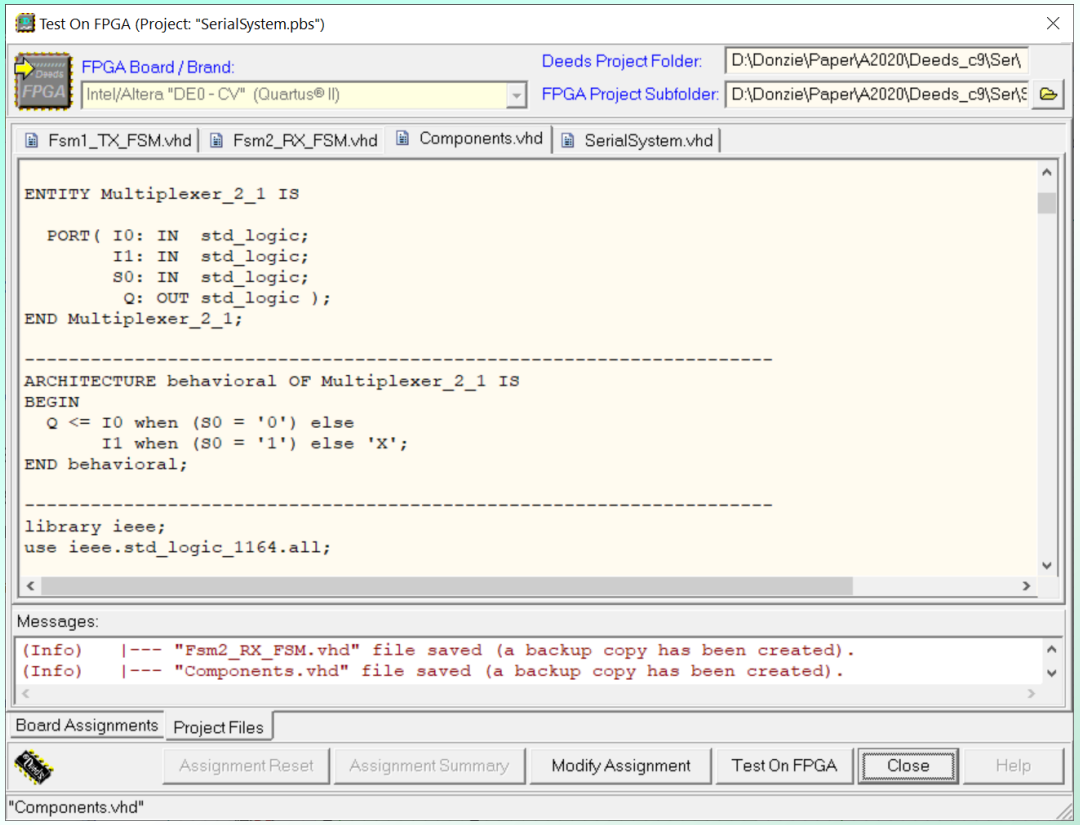
- The user selects each I/O termination in the Deeds schematic (see on the left), with the purpose of associating it to a device or resource available on board (on the right).
- Associations can be done *in advance by teachers*.



- In the present example, a clock resource on the FPGA board has been associated to the clock component in the Deeds schematic (assigning a clock frequency of 1KHz).

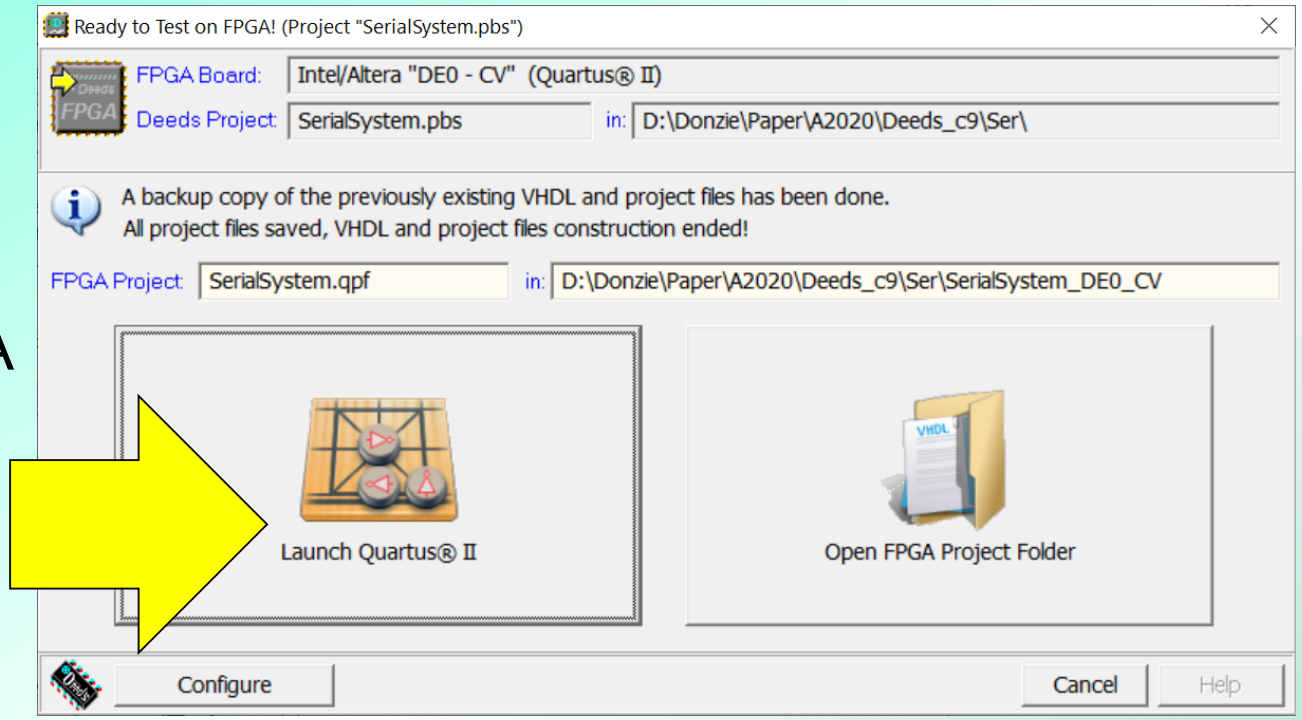
The generated VHDL code

- When the conversion ends, the VHDL files are presented to the user through a dedicated window.
- Users with a good VHDL familiarity can export, edit and re-use the generated code in a professional design tool of their choice.
- The VHDL generator builds also the project files required by the FPGA specific tool.



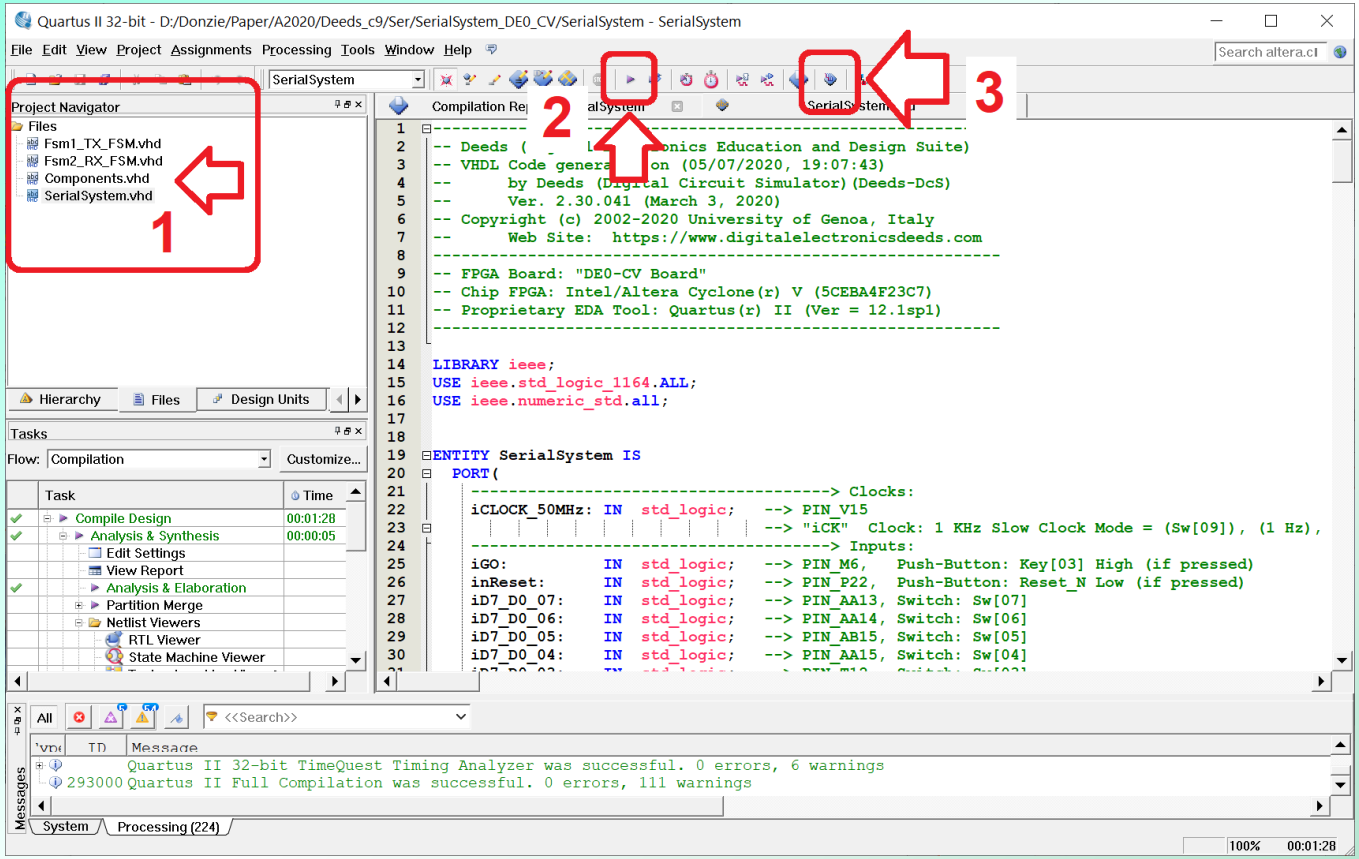
Now we can launch the FPGA tool

- According to the associations between the schematic and the board devices, Deeds defines all FPGA pins and board connections (in the generated project files).
- Once terminated the project generation, the user is invited to launch the FPGA software specific for the chip (Quartus® II).



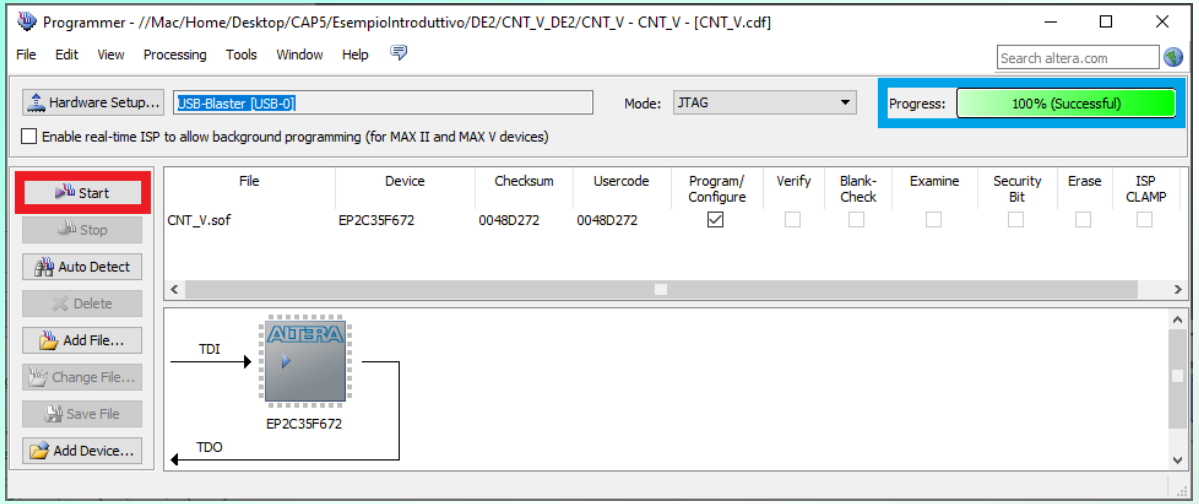
The role of the specific FPGA tool

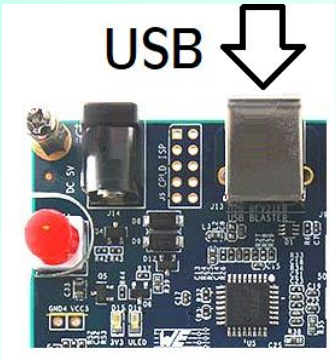
- Now we have the project files opened in Quartus® II, ready to be processed *(1)*.
- The student needs only to compile them *(2)*; *no operation on the VHDL code* is needed by the beginner.
- Finally, the student will load the project binaries into the FPGA board, using the programmer command *(3)*.



FPGA programmer module

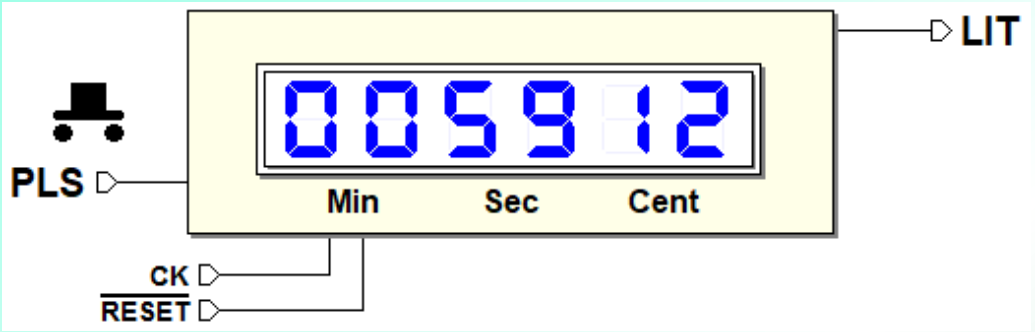
- The *Programmer* software module allow to load the binaries into the FPGA chip, through its JTAG interface.
- Normally, a USB serial port is used to program the FPGA boards.
- The programming hardware can be already on board, or external.





A laboratory session: an example

- We propose to the students to design and test on FPGA a *digital chronometer (*)*.
- Requested resolution is one hundredth of a second, maximum time measurable about one hour.

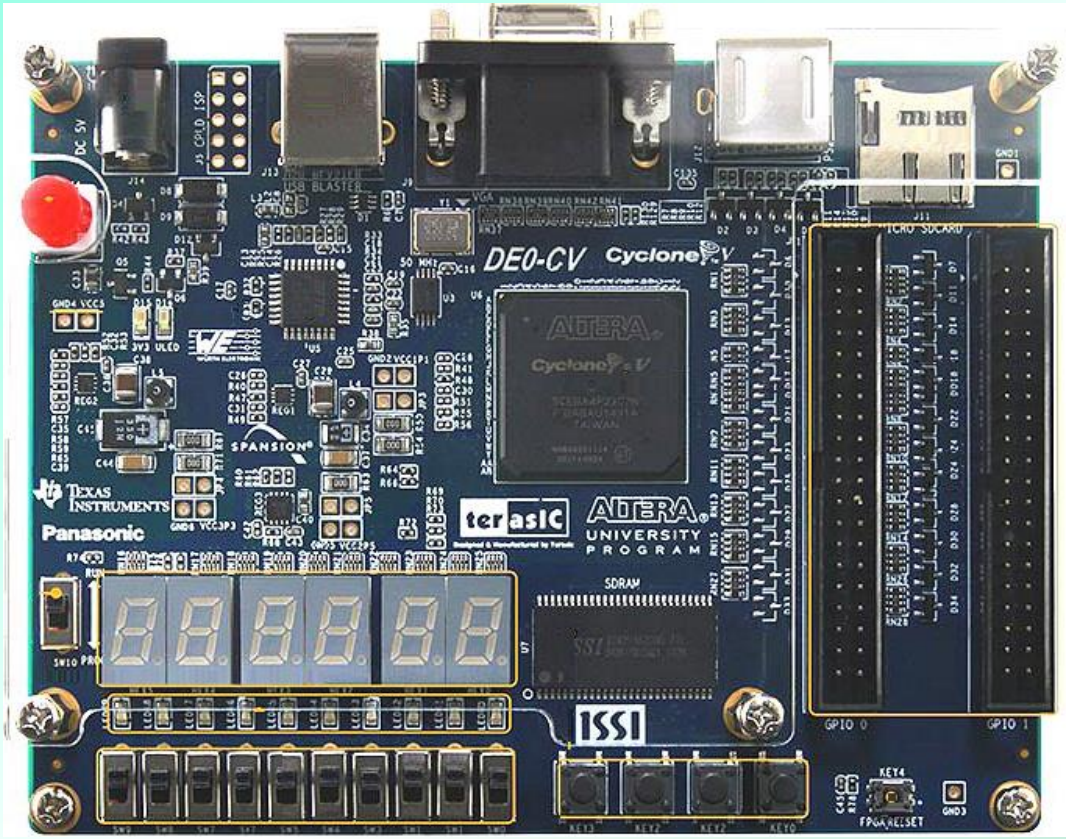


- Display will show six decimal digits, two for the minutes, two for the seconds and two for the hundredths of second.
- When the pushbutton PLS is pushed and then released, it start counting.
- On the second pressure it stops and the time will be readable on the displays; the third pressure resets all.
- The lamp LIT is activated for all the time PLS is pushed.
- The frequency of the clock is 100 Hz.
- We implement it on the DE0-CV FPGA board (Terasic/Altera).

(*) This exercise is proposed in the book "Introduction to Digital System Design": click here to download the files from the online support

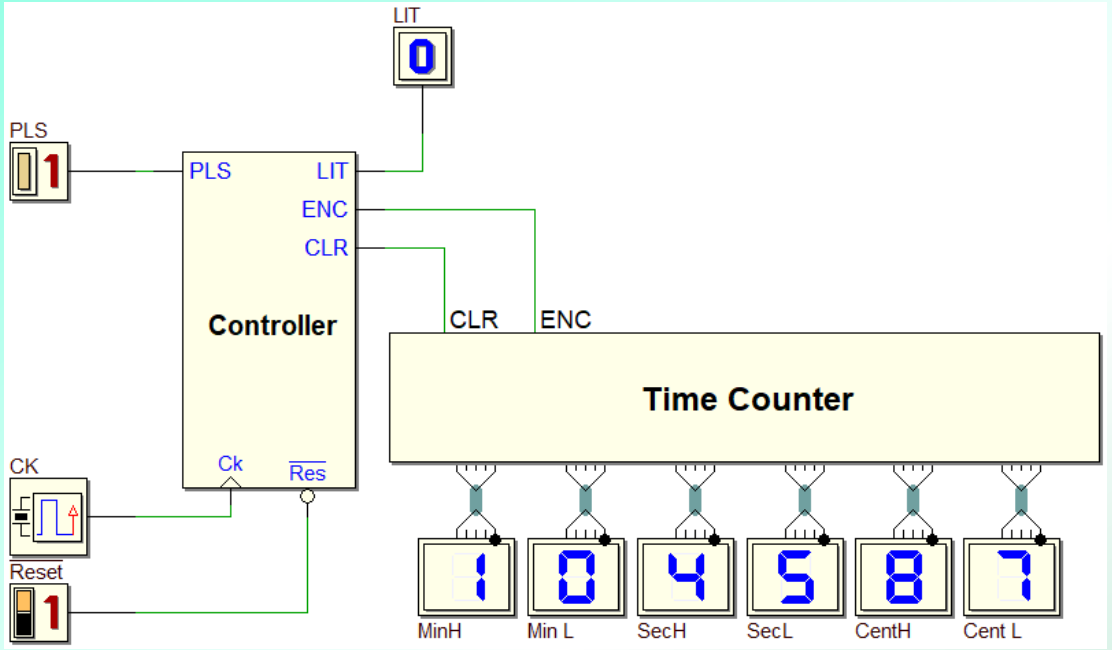
The FPGA Board

- The DE0-CV board makes available the required six seven-segments displays, push-buttons, switches and LEDs.
- The native 50 MHz clock generator will be scaled down automatically by Deeds, according to the project setting.



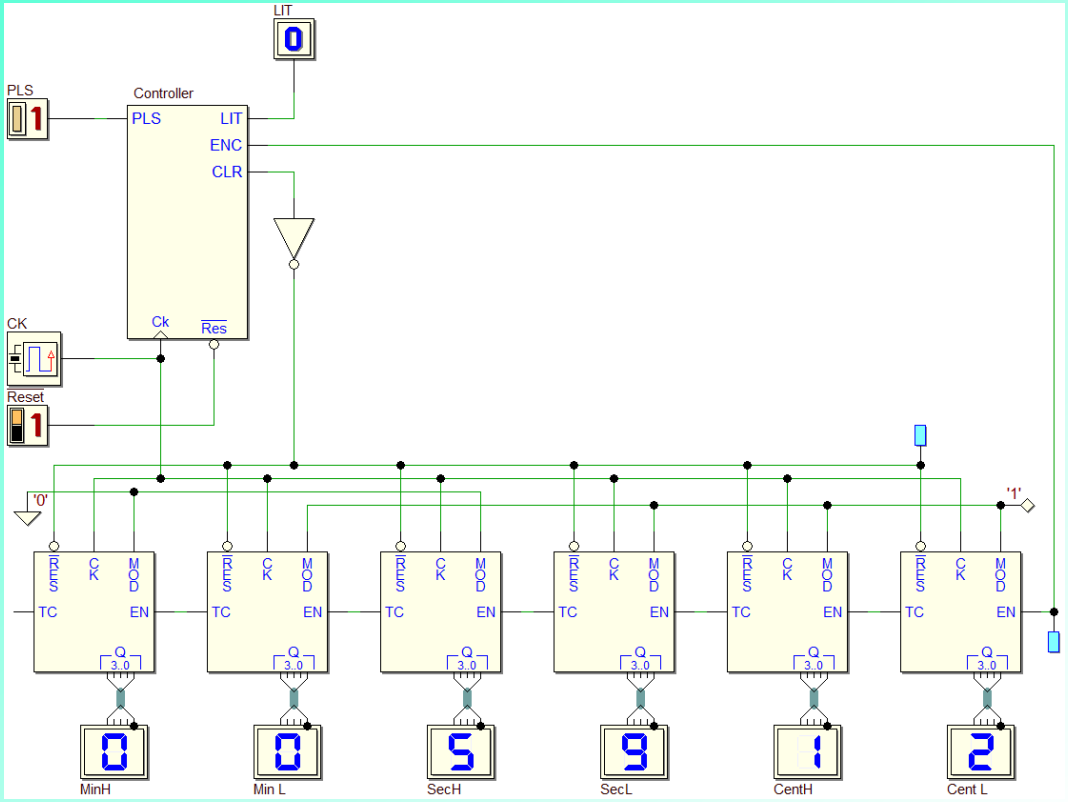
Design guidelines

- A controller-datapath structure is suggested to the student.
- The datapath includes the time counter and the associated displays.
- The controller reads the button PLS and handles the lamp LIT and the time counter controls.



- The time counter is cleared by CLR and will count when ENC='1'.
- We suggest to implement the controller as a Finite State Machine (FSM) that will track the PLS input.

Time counter architecture



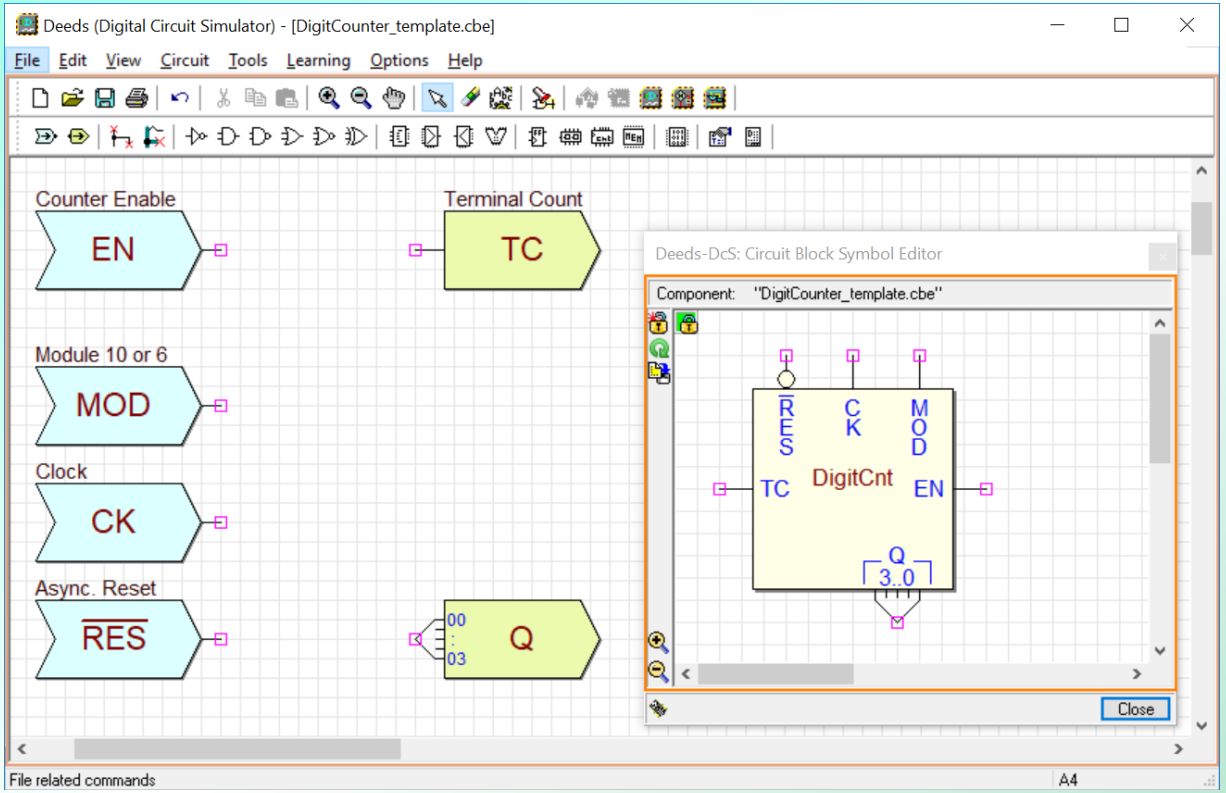
- The suggestion given to the students is to divide the counter in six BCD (Binary Coded Decimal) sub-modules.
- The interior of each module is left to the creativity of each one.
- In Deeds, each module elements can be defined as a CBE (Circuit Block Element).

- Each module receives a count enable EN from the one on the right side, and generates a TC (Terminal Count) to enable the one placed on its left side.

The 'templates' to speed up the design (1)

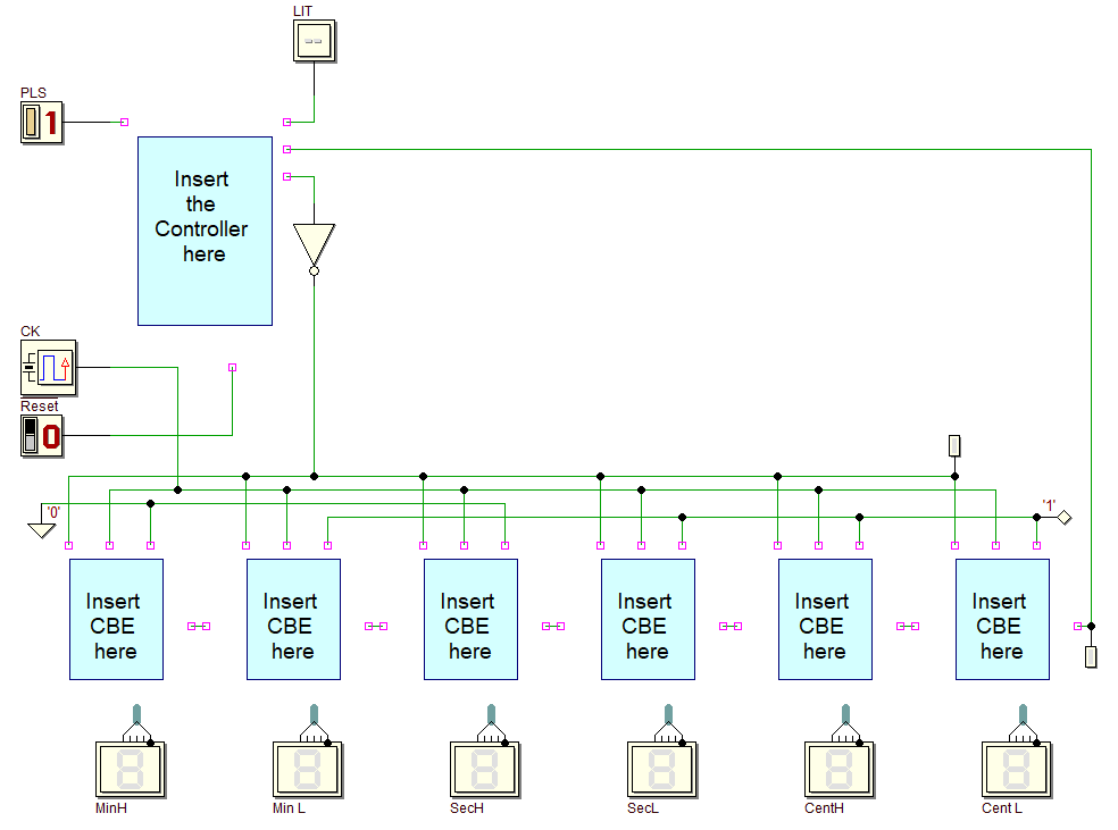
- A circuit 'template' can speed up the student work.
- The general uniformity given by templates is useful for the teacher too, to simplify checking of the quality of designs.

- CBE blocks are perfect to define circuit templates.
- In the figure, the CBE component template.
- I/O pins, and the component symbol are already defined.



The 'templates' to speed up the design (2)

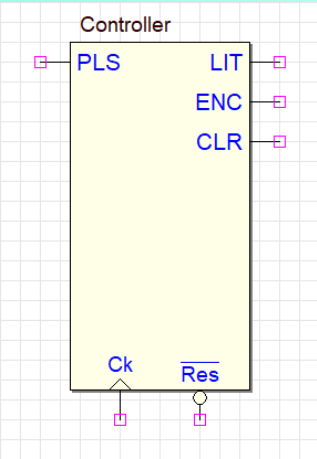
- A template of the top level circuit is also very useful. *Why?*
- In Deeds, *Input / Output* components have a special role.

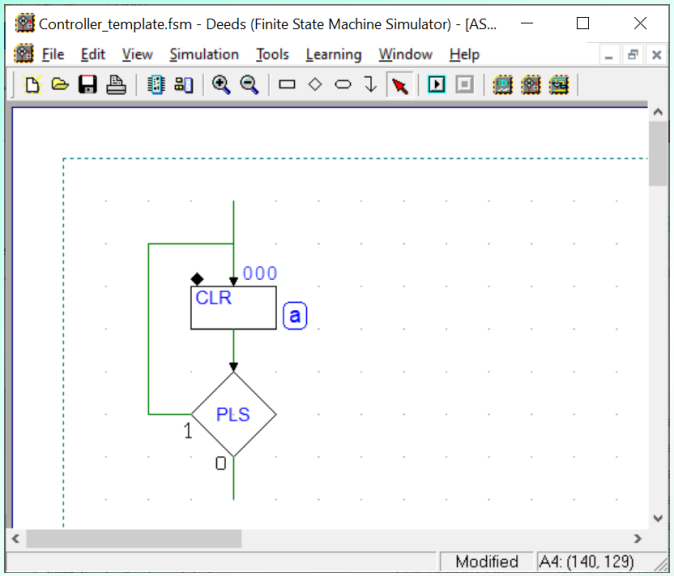


- *Timing Diagram:* Input components memorize the timing sequences used for the simulation.
- *Test on FPGA* window: Input and Output components store the association between the schematic and the physical devices on the FPGA board.

The ‘templates’ to speed up the design (3)

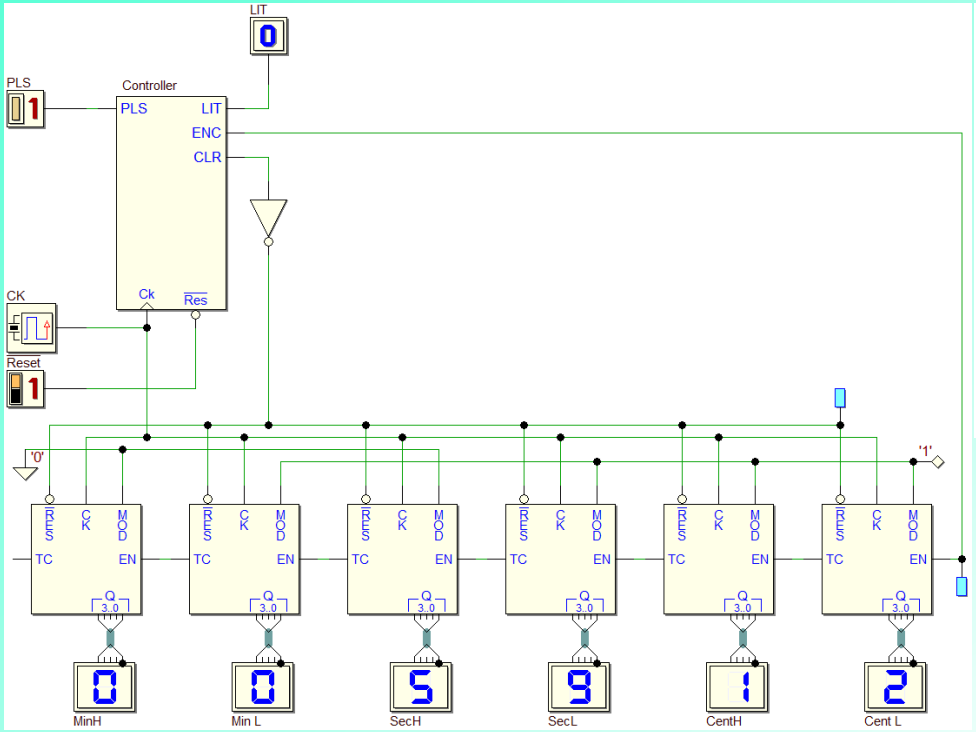
- You can give to the student a *top project file empty*, but already prepared for the FPGA prototyping.
  - You can test the solution in advance by simulating and prototyping it.
  - When all is working, it is only necessary delete from the circuit all, or part of, the circuit elements.
  - Leave in place the Input and output component, just to grant timing test sequences and the FPGA board configuration, to grant clear and comparable solutions from all the students.

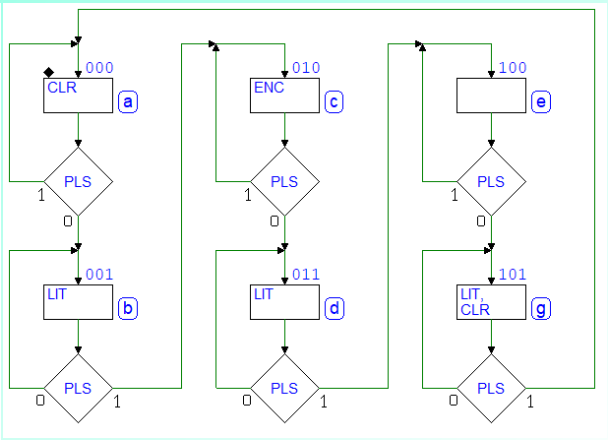


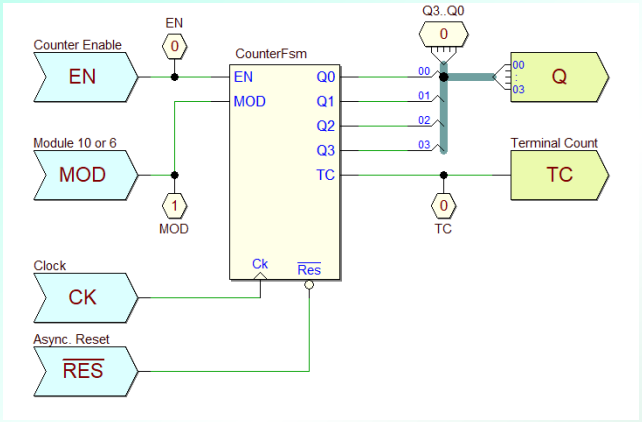


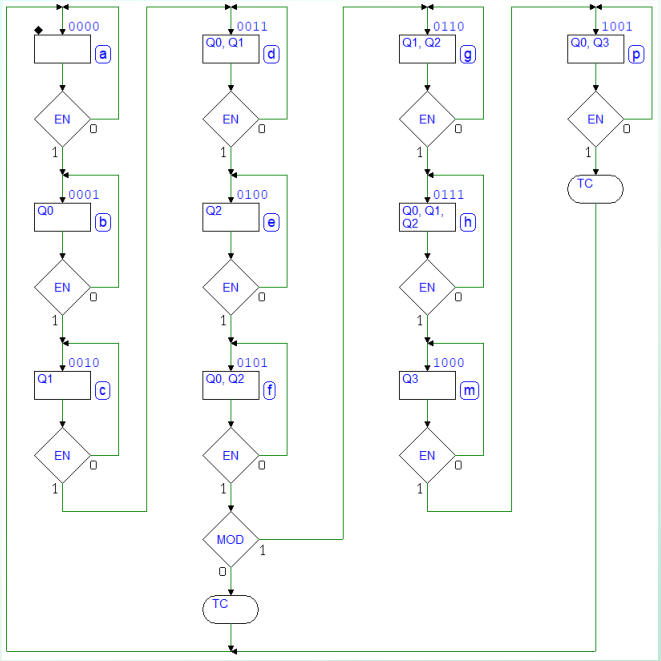
- Templates can be defined also for the FSM components (see the figures aside).
- The component symbol has been defined, but the ASM-chart presents only a partial suggestion.

The final circuit of the chronometer



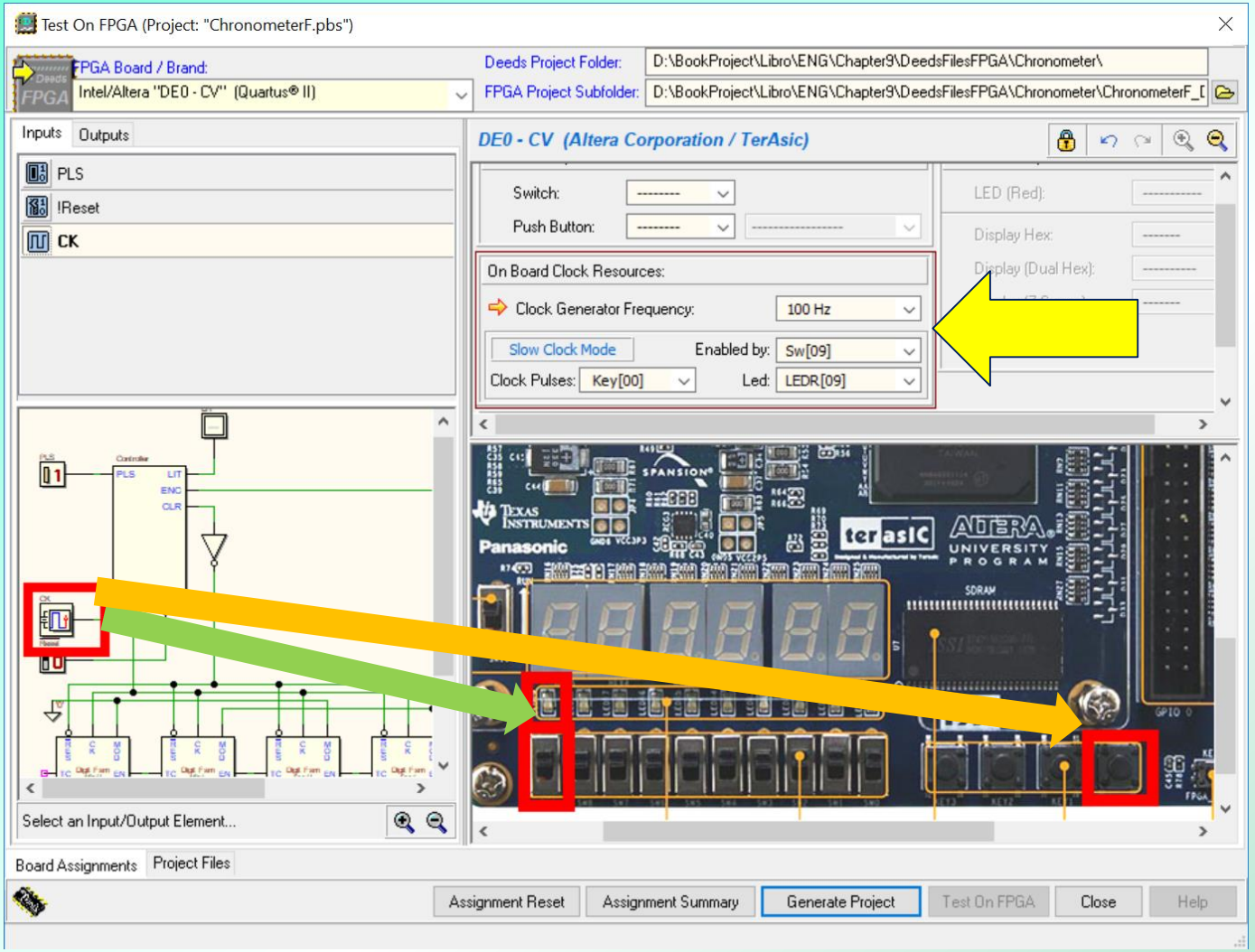






Specialized inputs: the programmable clock

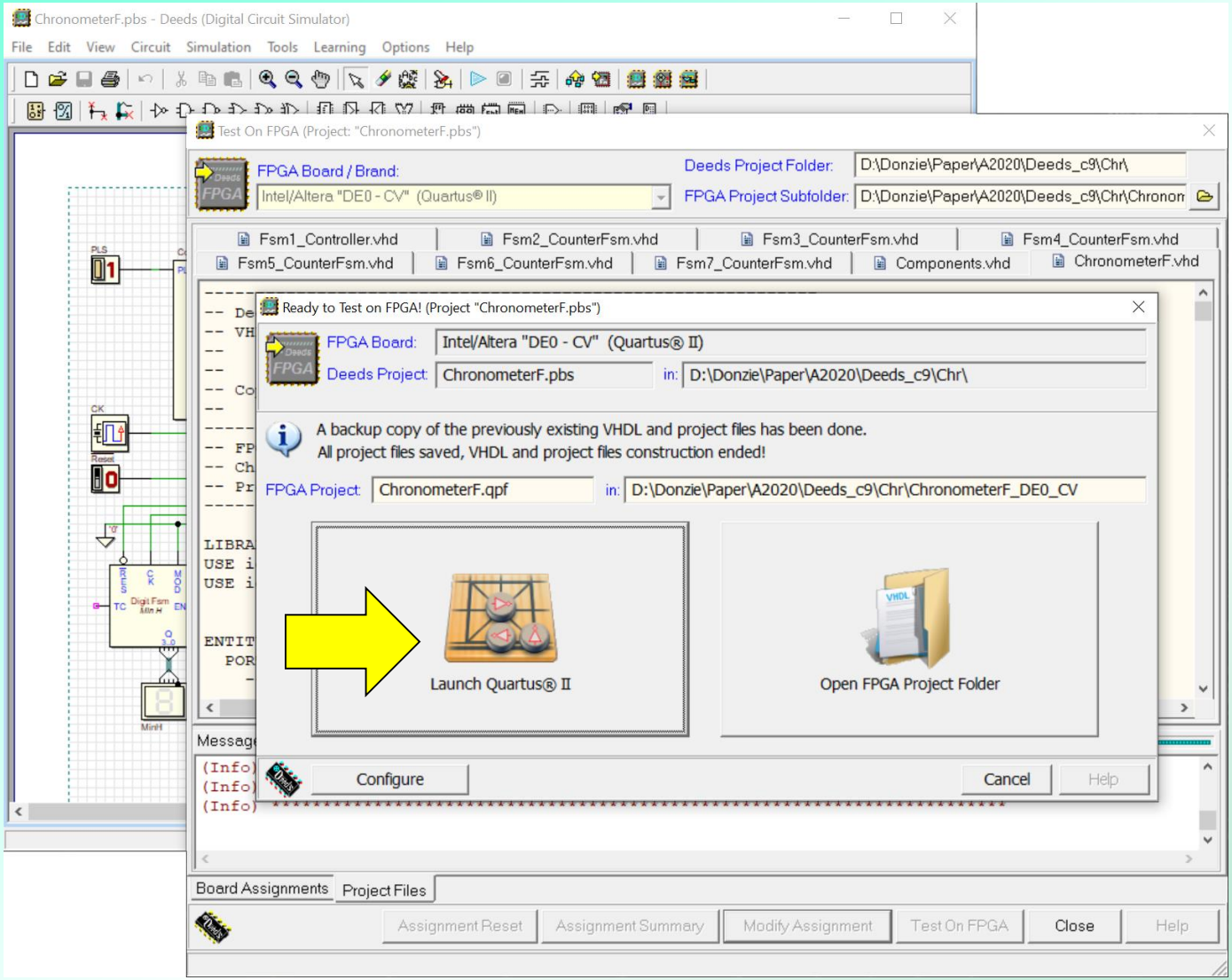
- *'Clock'*
- *'Slow Clock'*
- Fck = 100*Hz*
- *Sw[09]*: enables the slow mode.
- *Key[00]*: to pulse the clock in slow mode.
- *LEDR[09]*: to shows clock pulses.



(The associations have been already all defined in the template)

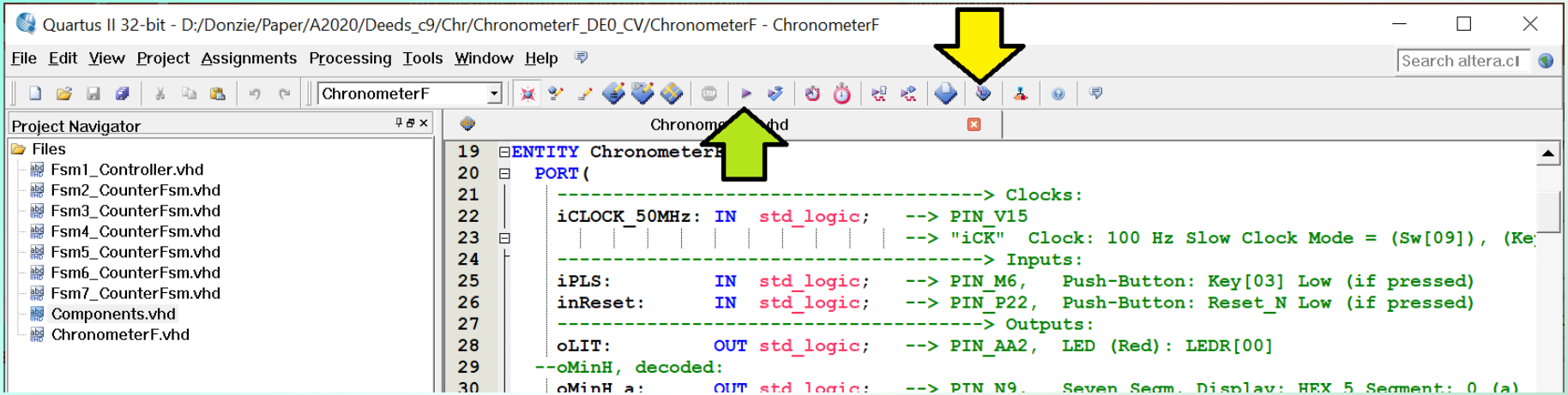
Exporting the VHDL project

- Now students are ready to launch the specific FPGA tool to compile the FPGA project files (Quartus® II).

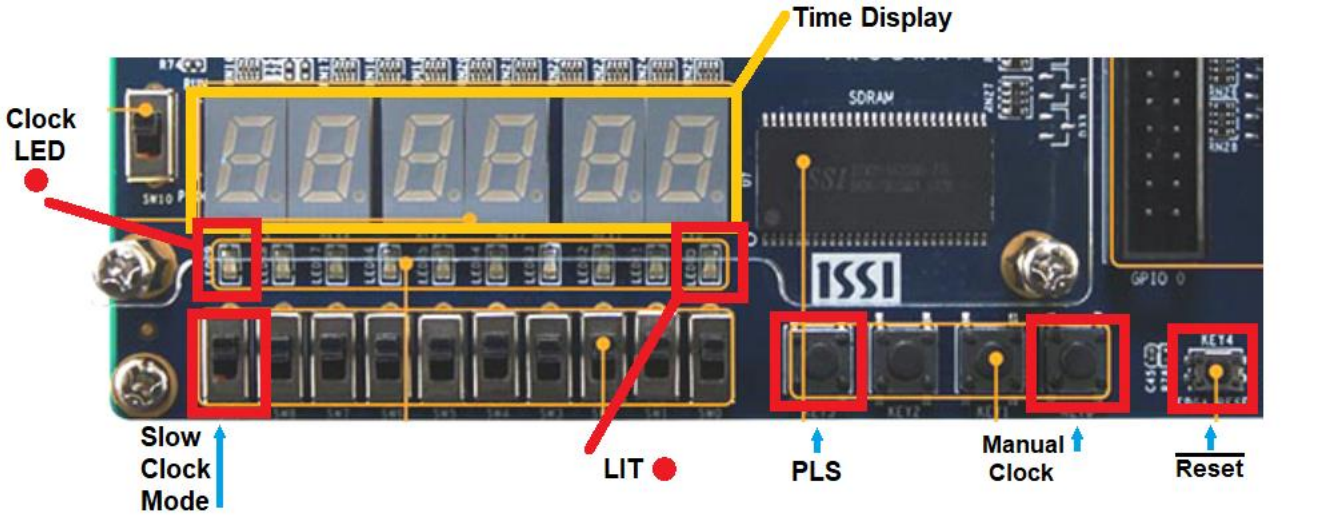


Compiling the VHDL project

- The project in Quartus® II is ready to be compiled (green arrow).
- The top VHDL file contains the structural description of the digital network schematic.
- The other VHDL files collect the behavioral descriptions of all the components used by the network, included user-defined FSM, micro-computers, ROM contents…
- After the compilation, we are able to transfer the circuit to the FPGA using the programmer module and the USB connection (yellow arrow).

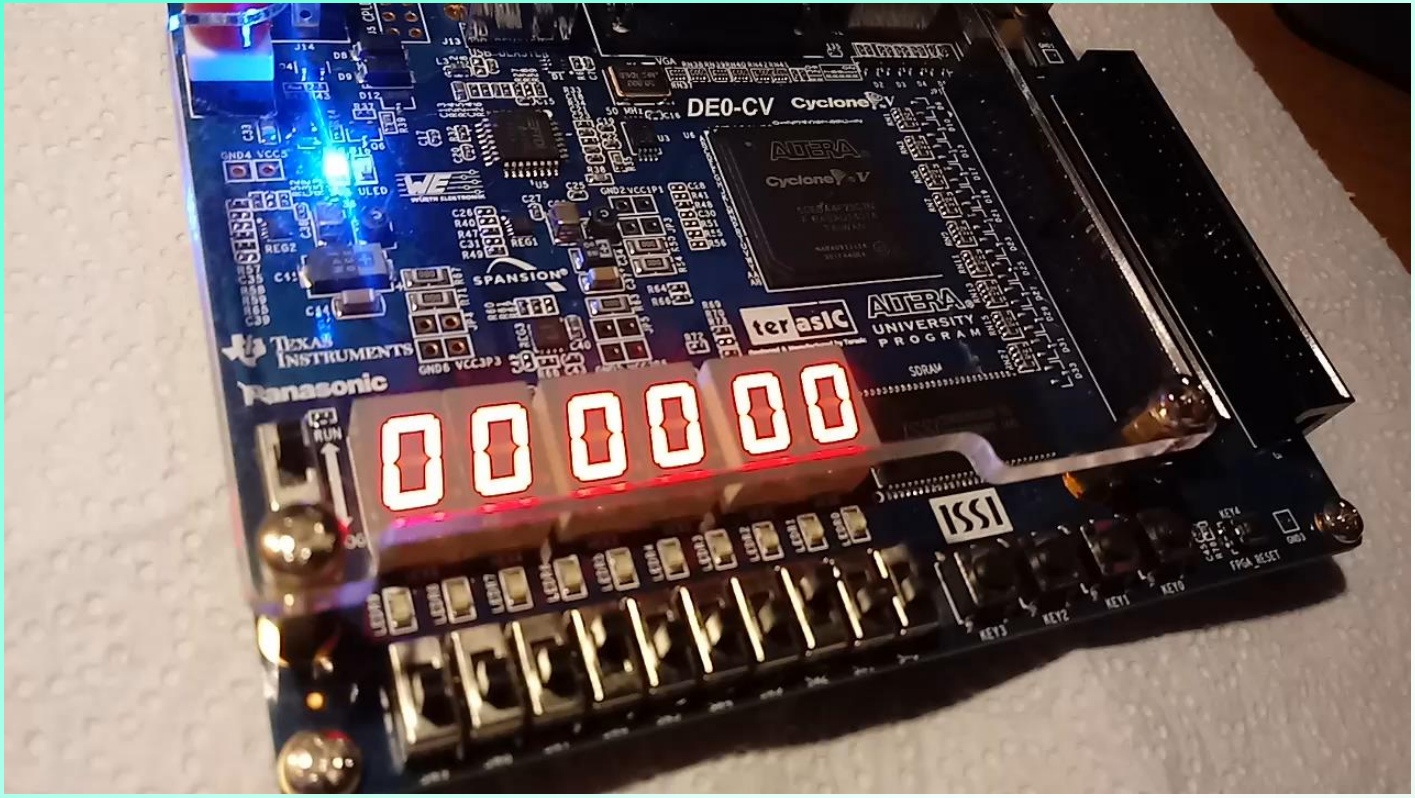


Testing the ‘physical’ chronometer



- Laboratory assignments always include a board synoptic view of the I/O of the circuit under test.
- Switches, push-buttons and LEDs of the FPGA board are put in evidence to facilitate the testing of the circuit.

The chronometer at work



Thank you for the interest and participation!

https://www.digitalelectronicsdeeds.com