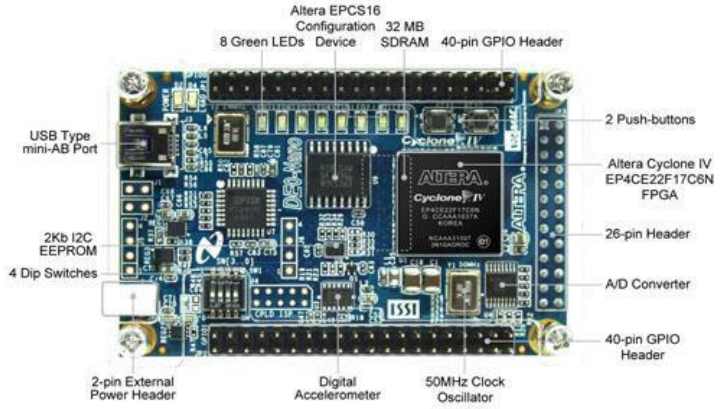
ELEC 2531

Lab 3

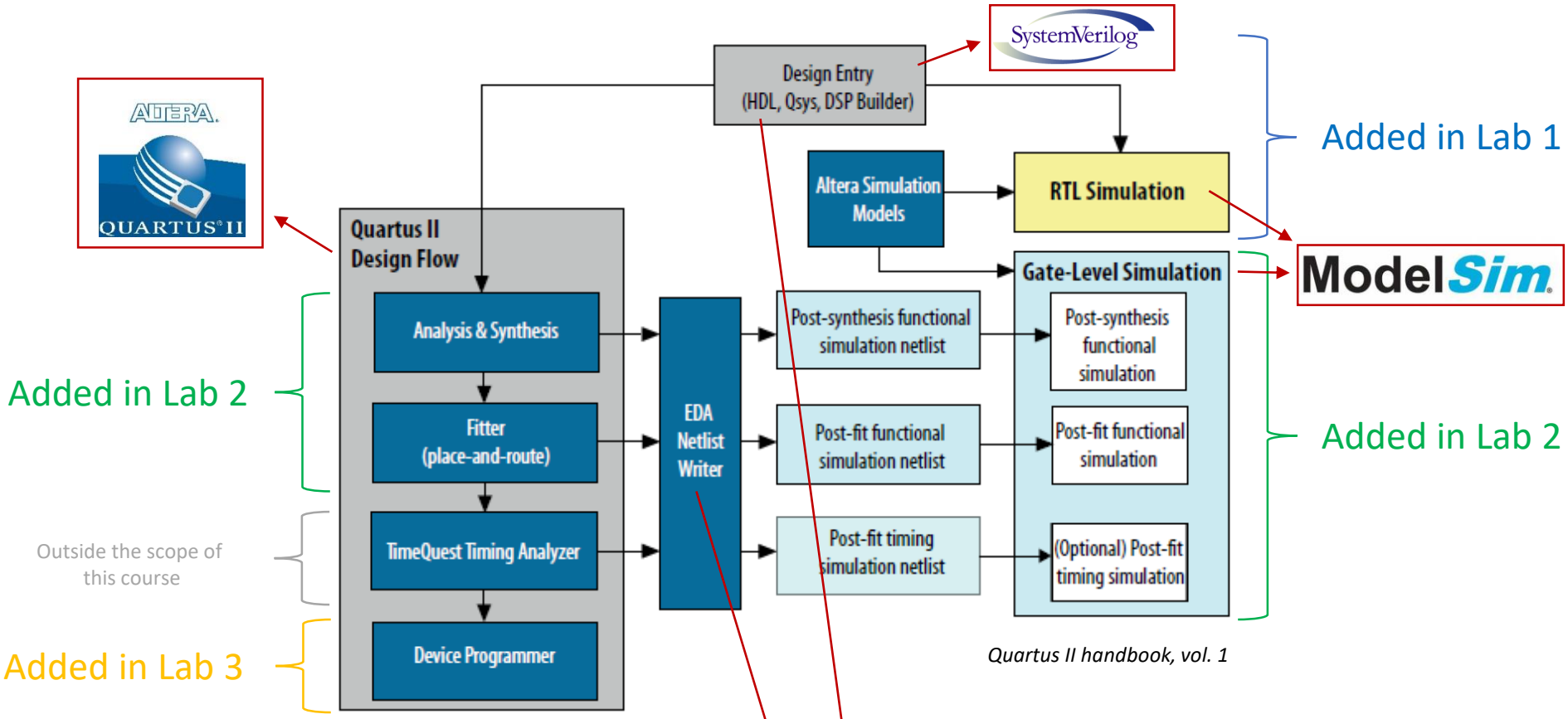
Introduction to **DE0-Nano board** & **Cyclone IV FPGA**:

- Generation of a SystemVerilog design for the Cyclone IV FPGA using Altera's **SystemBuilder**
- Synthesis, Place & Route using Quartus, RTL and Gate Level Simulation using ModelSim
- **Programming the FPGA** using Quartus
- In-circuit **SignalTap II** logic analyzer



Electronic Design Automation (EDA) Flow for FPGA

- We use a programming language - (System)Verilog - to **describe** **(HDL)** a circuit (our design entry) that we can simulate at different levels (ModelSim) and use to program a physical device (our FPGA, using Quartus).



- A circuit description (HDL) file is called a **netlist**. The netlist your write in (System)Verilog is processed by Quartus into lower abstraction level netlists (`Synthesis + Place&Route`) and then finally into a configuration file (*`Assembler`*) intended to program the FPGA (*`Device Programmer`*).

A quick reminder on the Raspberry-Pi

- Part of your personal tasks for this course consist of
  - Setting up your Raspberry-Pi (WiFi connection, SSH remote access, SFTP file transfer)
  - Getting familiar with the Linux environment (master the basics of the command line)
  - Integrate the basics of the Python programming language

-> 100% of you should these three points by the end of Week 7 (hence for the 4th lab)

Raspberry PI

- Get started
  - Plug in your Raspberry PI
  - Configure your Raspberry PI to local settings (language, key
  - Configure the I2C and SMBus (maybe already be done from
  - Connect your Raspberry PI to Ethernet and your local WIFI
  - Connect your Raspberry PI to UCL WIFI
    - We recommend to use UCLouvain-prive !
    - It is possible to access from eduroam with SSH the PI c
  - Install FileZilla on your PC (to transfer files between the PI a
  - Download the Python3 script "MyIPaddress.py" and copy th
  - Configure crontab to execute the script on reboot (select nar
    - `@reboot python3 /home/pi/MyPI-Nano/MyIPaddress`
  - Access to your PI with SSH

- Raspberry PI web site
  - Help
  - Get started
  - Documentation
  - Remote access

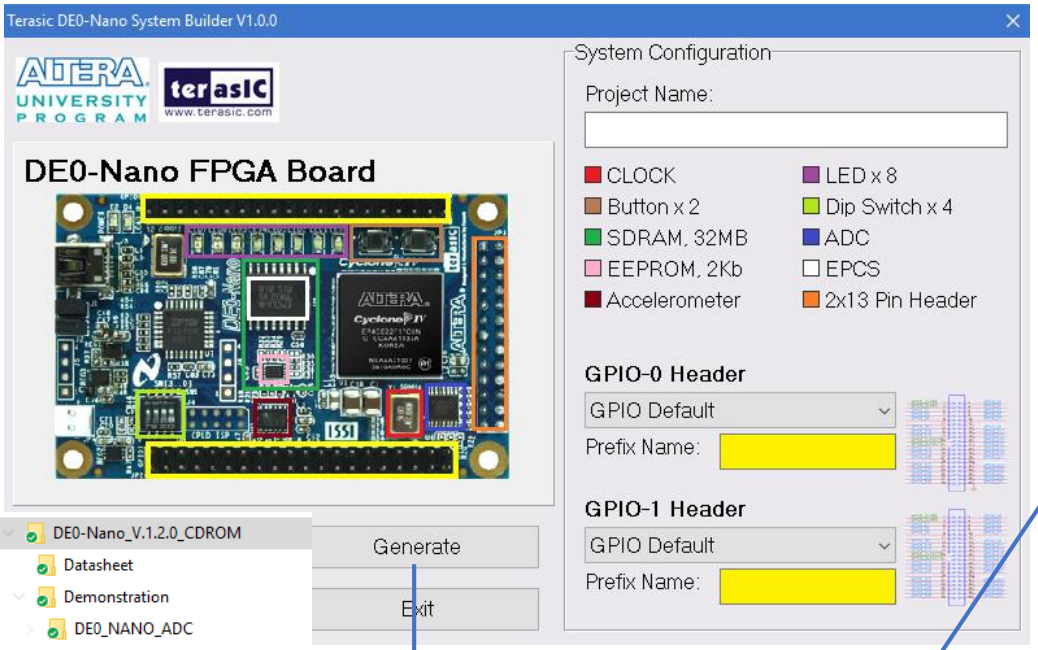
In constrast to HDL design for FPGA, embedded systems and especially the Raspberry-Pi benefit from large online resources and communities. It is hence important to use that as much as possible

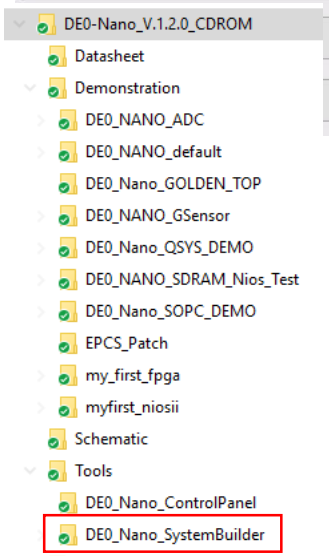
- Resources from the reference book
  - Chapter 9 : I/O Systems (9.1 to 9.3.4)
  - Getting Acquainted with Linux and the Raspberry PI

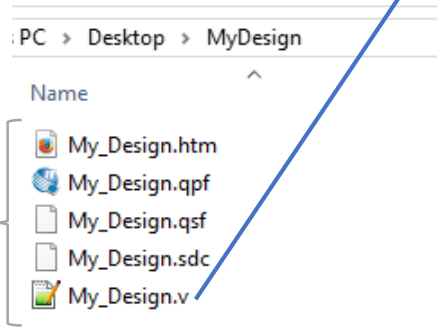
Primary to consult: concise while complete resource to get familiar with SSH connection, Linux environment, DDD and SFTP transfer

Quick Restructuring: the DE0-Nano board

- The cyclone IV FPGA, yes, but many other stuff; let's review them:





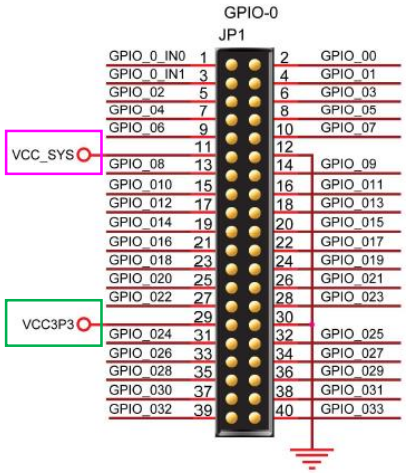



```
//=============================================================
//  This code is generated by Terasic System Builder
//=============================================================

module My_Design(

      ///////////// CLOCK /////////
      input CLOCK_50,

      ///////////// LED /////////
      input [7:0] LED,

      ///////////// KEY /////////
      input [1:0] KEY,

      ///////////// SW /////////
      input [3:0] SW,

      ///////////// SDRAM /////////
      input [12:0] DRAM_ADDR,
      // (...)
      input        DRAM_WE_N,

      ///////////// EPCS /////////
      output EPCS_ASDO,
      input EPCS_DATA0,
      output EPCS_DCLK,
      output EPCS_NCSO,

      ///////////// Accelerometer and EEPROM /////////
      output inputG_SENSOR_CS_N,
      input G_SENSOR_INT,
      output I2C_SCLK,
      inout I2C_SDAT,

      ///////////// ADC /////////
      output ADC_CS_N,
      output ADC_SADDR,
      output ADC_SCLK,
      input ADC_SDAT,

      ///////////// 2x13 GPIO Header /////////
      inout [12:0] GPIO_2,
      input [2:0]  GPIO_2_IN,

      ///////////// GPIO_0, GPIO_0 connect to GPIO Default /////////
      inout [33:0] GPIO_0,
      input [1:0]  GPIO_0_IN,

      ///////////// GPIO_1, GPIO_1 connect to GPIO Default /////////
      inout [33:0] GPIO_1,
      input [1:0]  GPIO_1_IN
);
```

Quick Restructuring: the DE0-Nano board

- Digital clock:
  - `CLOCK_50`: on-board 50MHz clock oscillator (chip in red box, on the previous slide)
  - Other clocks may be derived from `CLOCK_50` using PLLs (4 available in the FPGA), see Lab-3B

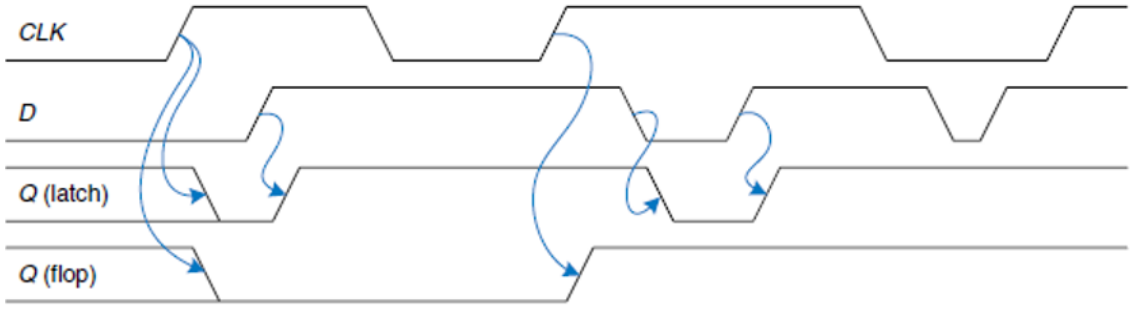
- Supply voltages:
  - Main source = mini-USB (5V)
  - The Cyclone IV FPGA is fed with 1.2V, 2.5V and 3.3V DC supply voltages, produced by dedicated chips on the board (linear regulators)
  - 5.0V and 3.3V pins are available on the GPIO-0 and GPIO-1 headers



- Memories:
  - Is used to store designs on the board (see INGI2315, ELEC2103):
    - 8MB Flash (Spansion EPCS64, non-volatile)
  - Can be addressed explicitly by the user in designs (see INGI2315, ELEC2103):
    - 32MB SDRAM (volatile)
    - 2kB EEPROM (non-volatile)
  - Are used by the Quartus compiler to build the required memory blocks of your design:
    - 1kB M9K blocks (volatile, x 66 => max. 600kbits)
    - Logic elements: contain dedicated logic registers (max. 22kbits)

Quick Restructuring: Timing & Delays

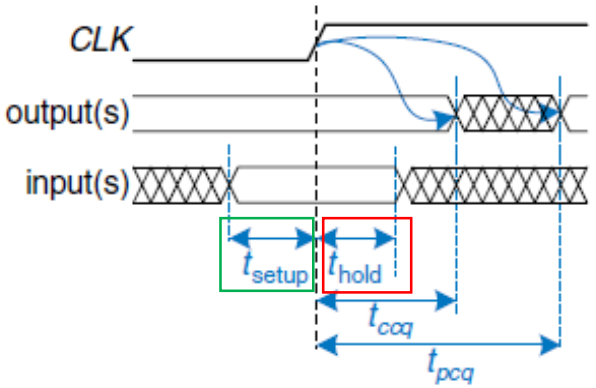
- Time is an important factor in digital circuits:
  - Every wire, gate or component introduce delays (e.g. propagation)
  - Sequential devices (flip-flops, latches, …) need a digital clock to work …

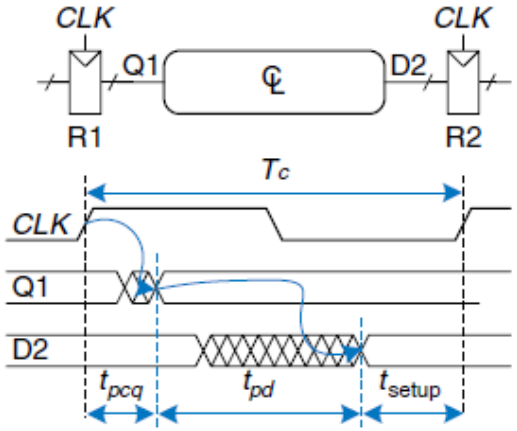


SystemVerilog: always_latch

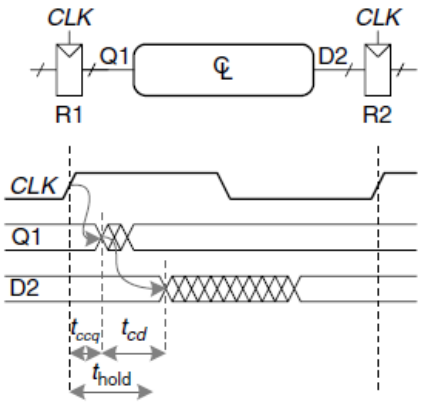
SystemVerilog: always_ff

… and therefore introduce timing considerations in the design





$$T_c \geq t_{pcq} + t_{pd} + t_{\text{setup}}$$



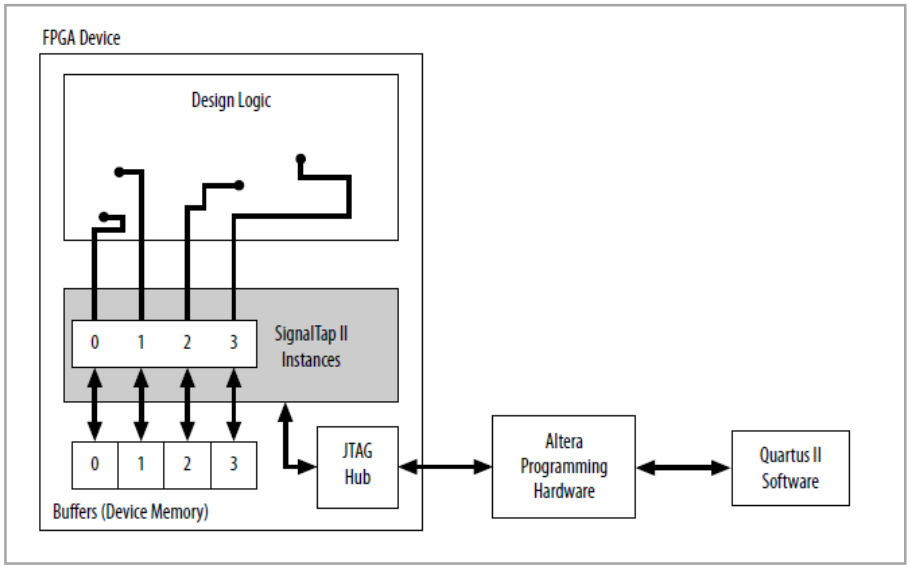
$$t_{ccq} + t_{cd} \geq t_{\text{hold}}$$

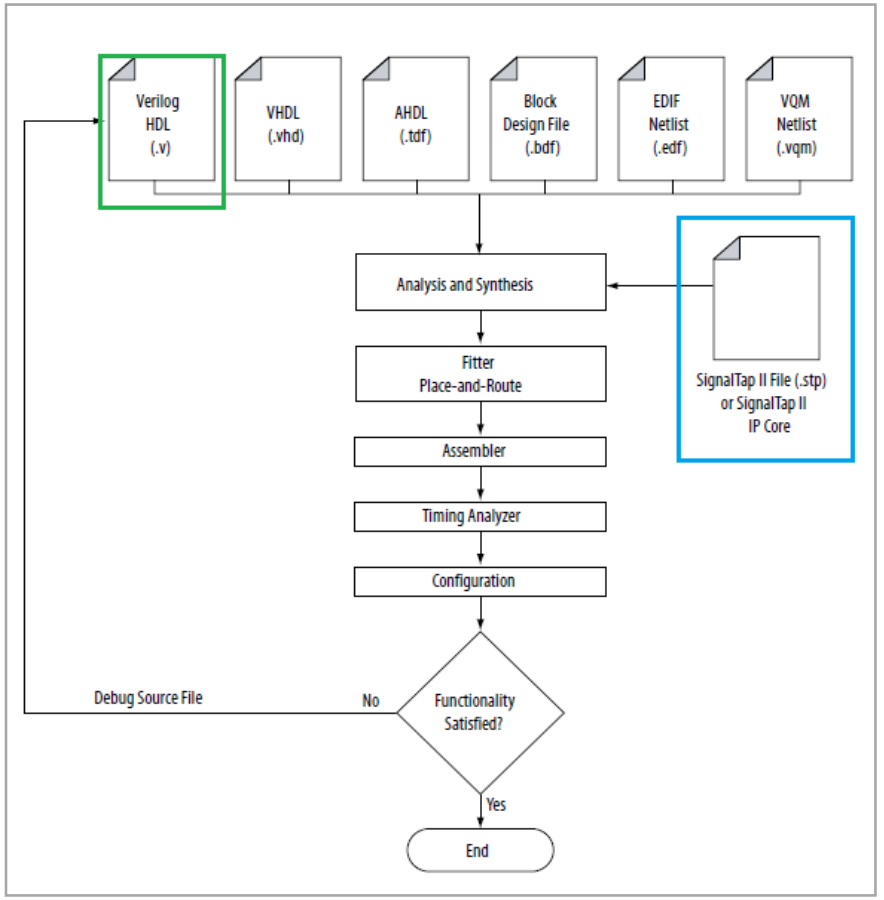
Harris & Harris, chap. 3.5

Quick Restructuring: Signal Tap II

Quartus Handbook, vol. III, chap. 13

- In-circuit logic analyzer: often neglected by students (tired of new tools ?) but **(1)** it is an incredibly useful tool (ELEC2103 !) and **(2)** we will ask you to deliver Signal Tap waves at evaluations 2 and/or 3 (!PIPE!) -> you should take as seriously as the rest





Before SignalTap

Flow Summary	
Flow Status	Successful - Sun Oct 09 13:32:11 2016
Quartus II 64-Bit Version	15.0.0 Build 145 04/22/2015 SJ Web Edition
Revision Name	DE0_NANO
Top-level Entity Name	DE0_NANO
Family	Cyclone IV E
Device	EP4CE22F17C6
Timing Models	Final
Total logic elements	40 / 22,320 (< 1 %)
Total combinational functions	39 / 22,320 (< 1 %)
Dedicated logic registers	28 / 22,320 (< 1 %)
Total registers	28
Total pins	11 / 154 (7 %)
Total virtual pins	0
Total memory bits	0 / 608,256 (0 %)
Embedded Multiplier 9-bit elements	0 / 132 (0 %)
Total PLLs	0 / 4 (0 %)

After SignalTap

Flow Summary	
Flow Status	Successful - Sun Oct 09 13:47:37 2016
Quartus II 64-Bit Version	15.0.0 Build 145 04/22/2015 SJ Web Edition
Revision Name	DE0_NANO
Top-level Entity Name	DE0_NANO
Family	Cyclone IV E
Device	EP4CE22F17C6
Timing Models	Final
Total logic elements	1,223 / 22,320 (5 %)
Total combinational functions	690 / 22,320 (3 %)
Dedicated logic registers	1,028 / 22,320 (5 %)
Total registers	1028
Total pins	11 / 154 (7 %)
Total virtual pins	0
Total memory bits	352,256 / 608,256 (58 %)
Embedded Multiplier 9-bit elements	0 / 132 (0 %)
Total PLLs	0 / 4 (0 %)