FPGA

CPU Architecture

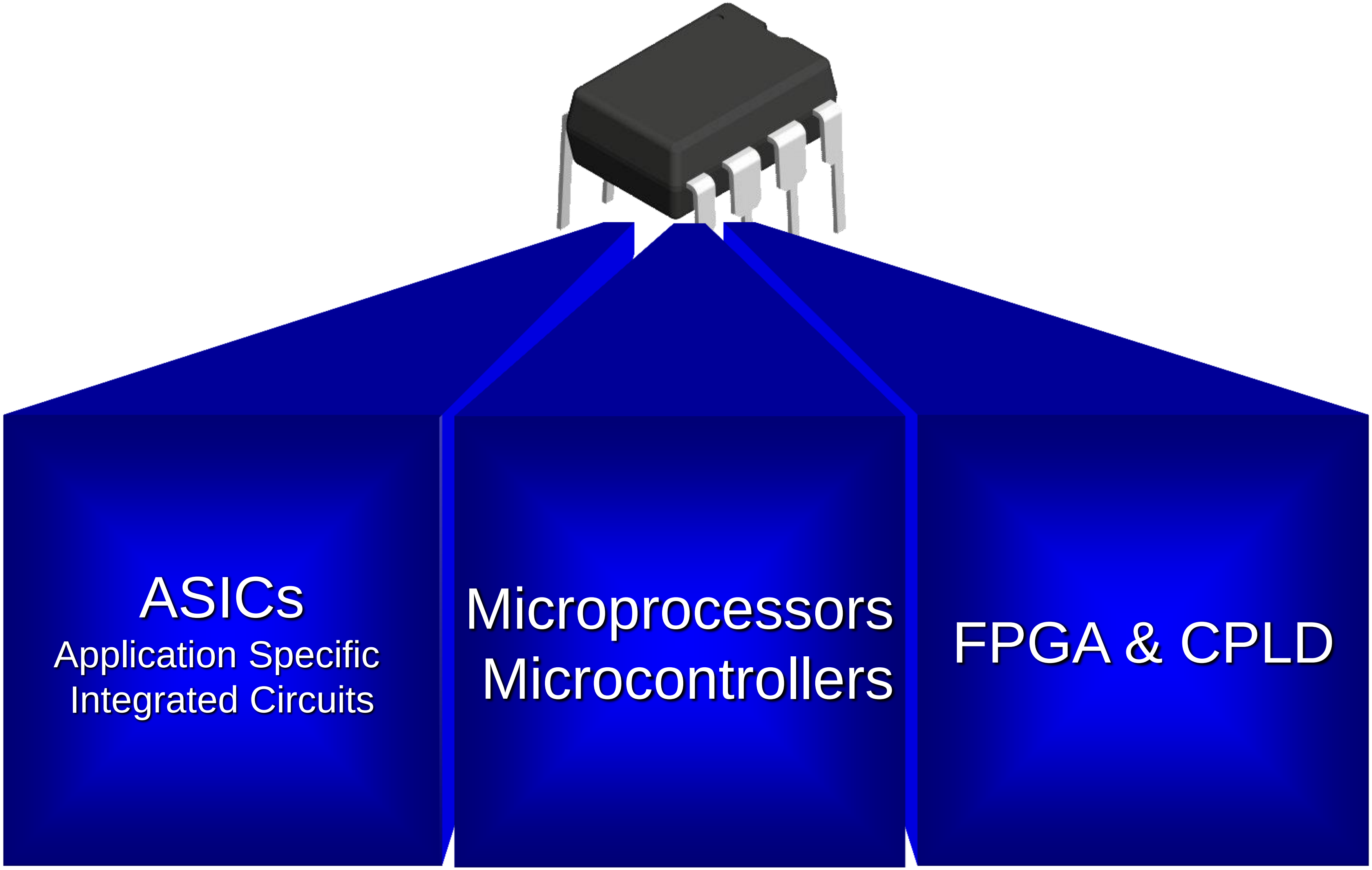
Agenda

- What is a programmable logic
- What is an FPGA
- Altera Cyclone II 20 FPGA
- Design Flow
  - Coding
  - Compiling
  - Pin Assignment
  - Configuring the board
  - Debugging (Signal TAP)
  - PLLs and Templates

Agenda

What is a programmable logic

Digital Semiconductor Chips



ASICs

Application Specific
Integrated Circuits

Microprocessors
Microcontrollers

FPGA & CPLD

Programmable logic

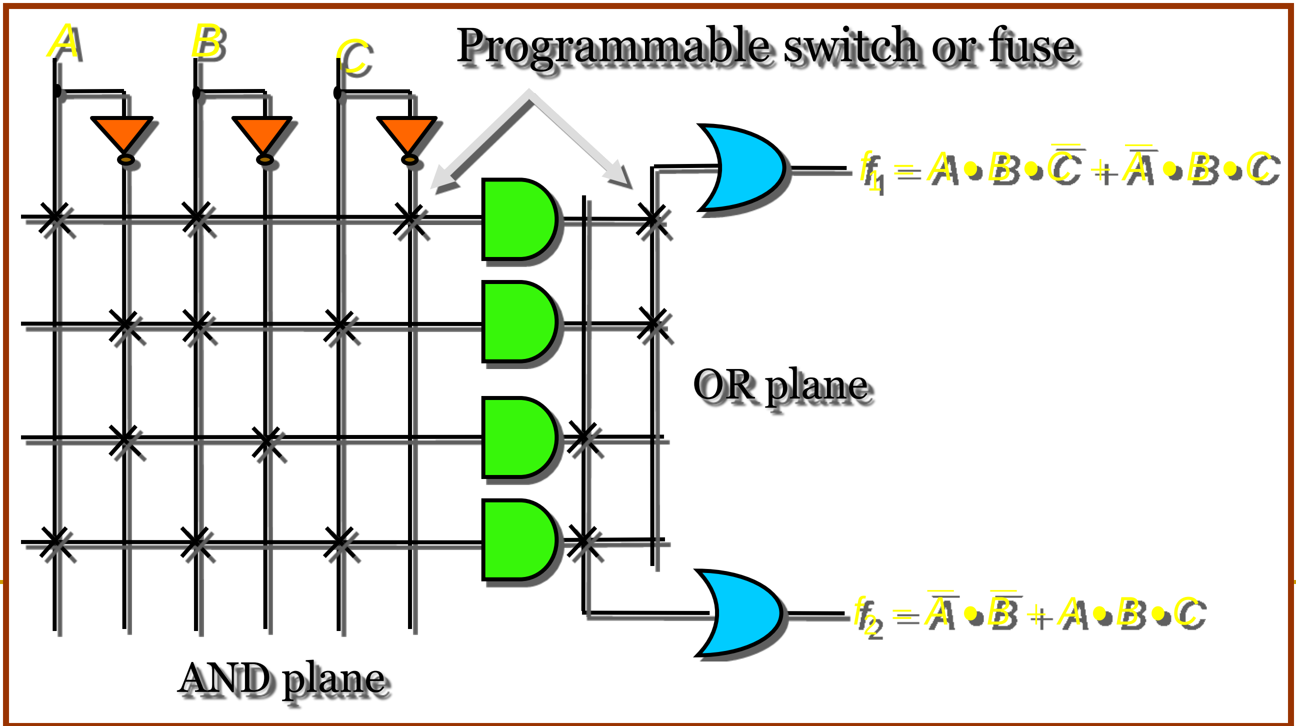
- An integrated circuit that can be programmed/reprogrammed with a digital logic of a curtain level.
- Started at late 70s and constantly growing
- Now available of 100th K Flip-Flops in a single chip.

Advantages

- Short Development time
- Reconfigurable
- Saves board space
- Flexible to changes
- No need for ASIC expensive design and production
- Fast time to market
- Bugs can be fixed easily

How it Began (70's): PLA

- Programmable Logic Array
- First programmable device
- 2-level and-or structure
- One time programmable

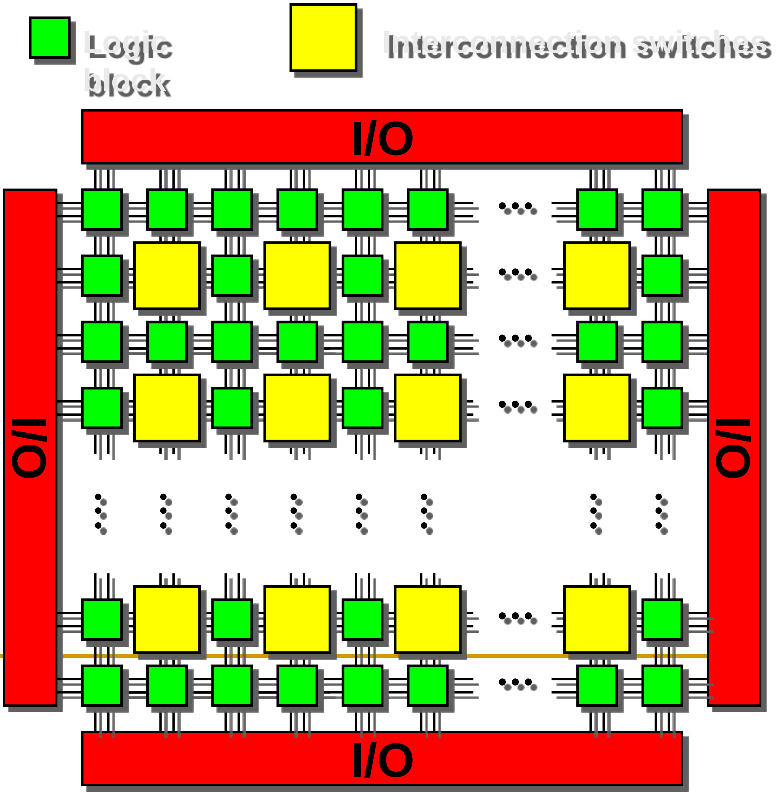

A
B
C
Programmable switch or fuse
$f_1 = A \bullet B \bullet \overline{C} + \overline{A} \bullet B \bullet C$
OR plane
$f_2 = \overline{A} \bullet \overline{B} + A \bullet B \bullet C$
AND plane

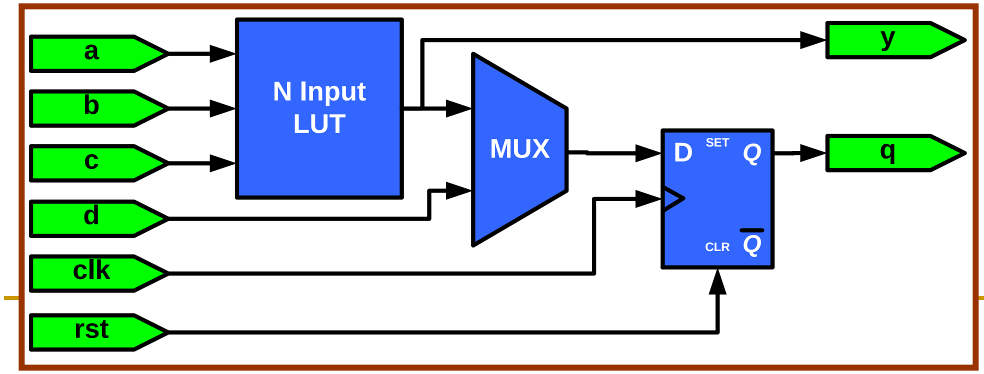
Agenda

What is an FPGA

FPGA - Field Programmable Gate Array

- Programmable logic blocks (Logic Element “LE”)
  Implement combinatorial and sequential logic. Based on LUT and DFF.
- Programmable I/O blocks
  Configurable I/Os for external connections supports various voltages and tri-states.
- Programmable interconnect
  Wires to connect inputs , outputs and logic blocks.
  - clocks
  - short distance local connections
  - long distance connections across chip

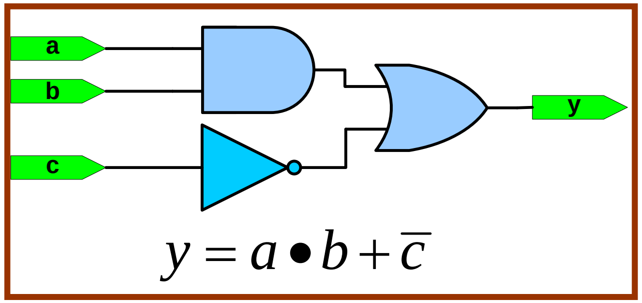




Configuring LUT

- LUT is a RAM with data width of 1bit.
- The contents are programmed at power up

Required Function

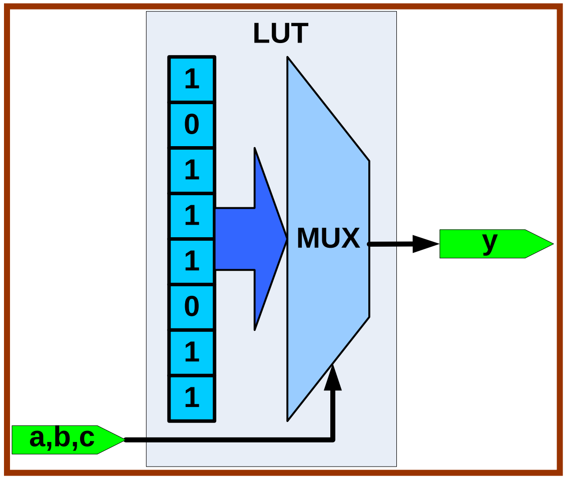


$y = a \bullet b + \overline{c}$

Truth Table

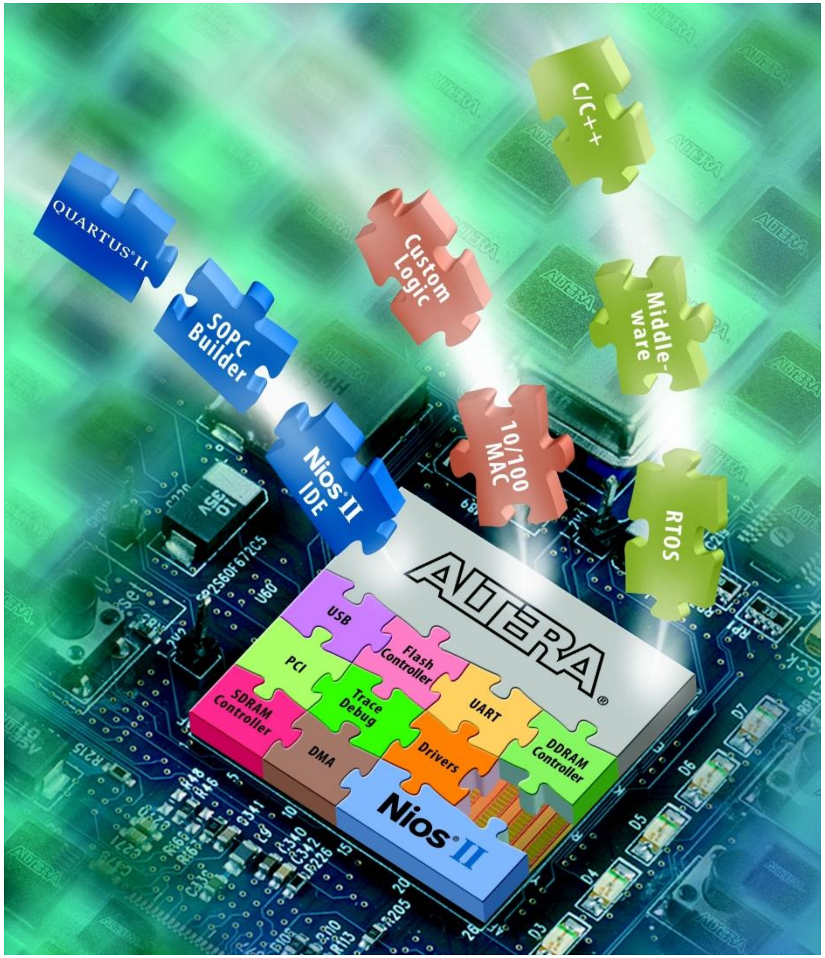
A	b	c	y
0	0	0	**1**
0	0	1	**0**
0	1	0	**1**
0	1	1	**1**
1	0	0	**1**
1	0	1	**0**
1	1	0	**1**
1	1	1	**1**

Programmed LUT

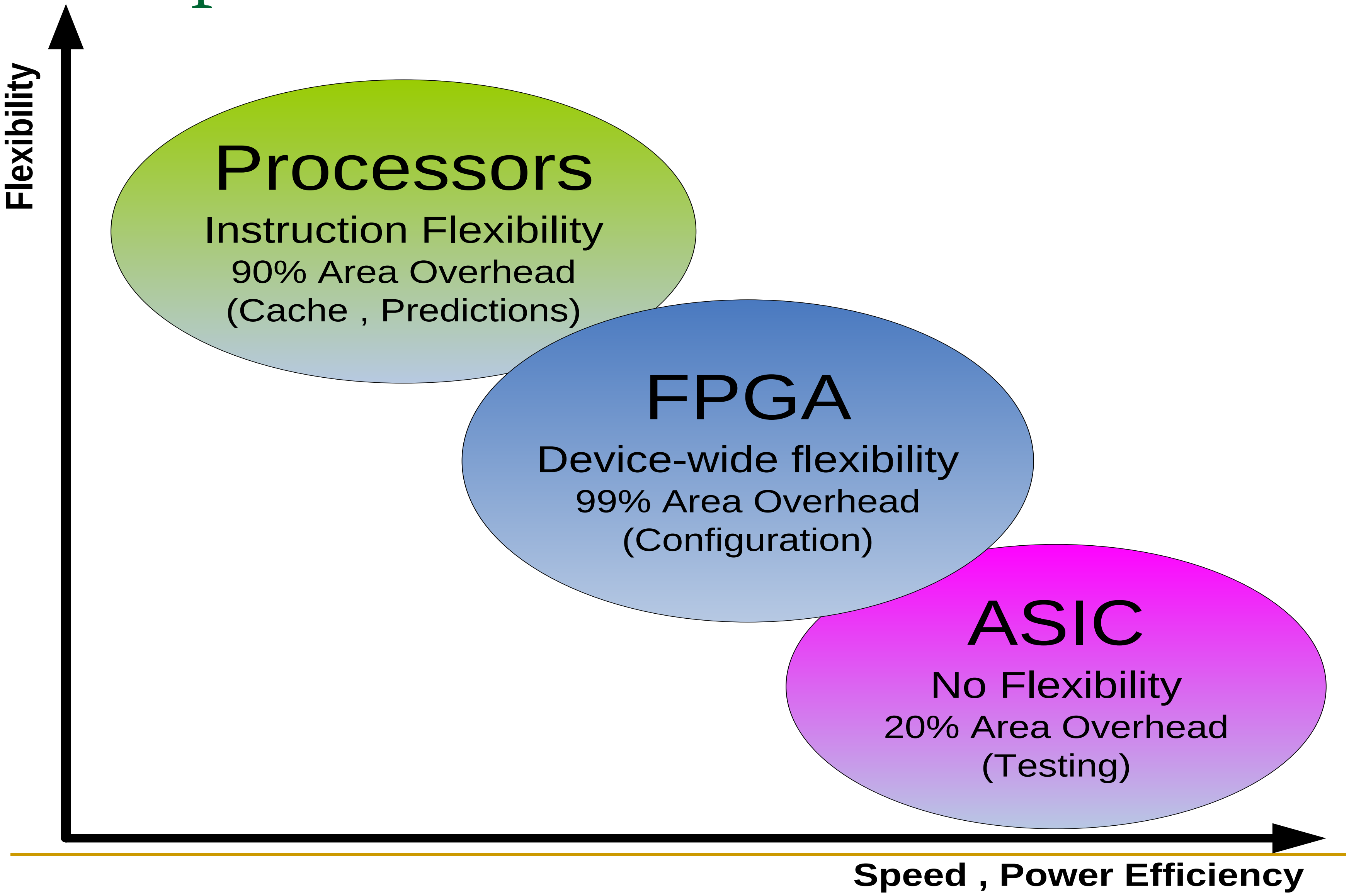


Special FPGA functions

- Internal SRAM
- Embedded Multipliers and DSP blocks
- Embedded logic analyzer
- PLLs
- Embedded CPUs
- High speed I/O
- DDR/DDRII/DDRIII SDRAM interfaces



Comparison



Flexibility
Processors
Instruction Flexibility
90% Area Overhead
(Cache , Predictions)
FPGA
Device-wide flexibility
99% Area Overhead
(Configuration)
ASIC
No Flexibility
20% Area Overhead
(Testing)
Speed , Power Efficiency

Usages

- Small quantity products
- Reconfigurable systems
- Upgradeable systems
- ASIC prototyping and emulation
- Education

Manufacturers

- Xilinx
- Altera
- Lattice
- Actel

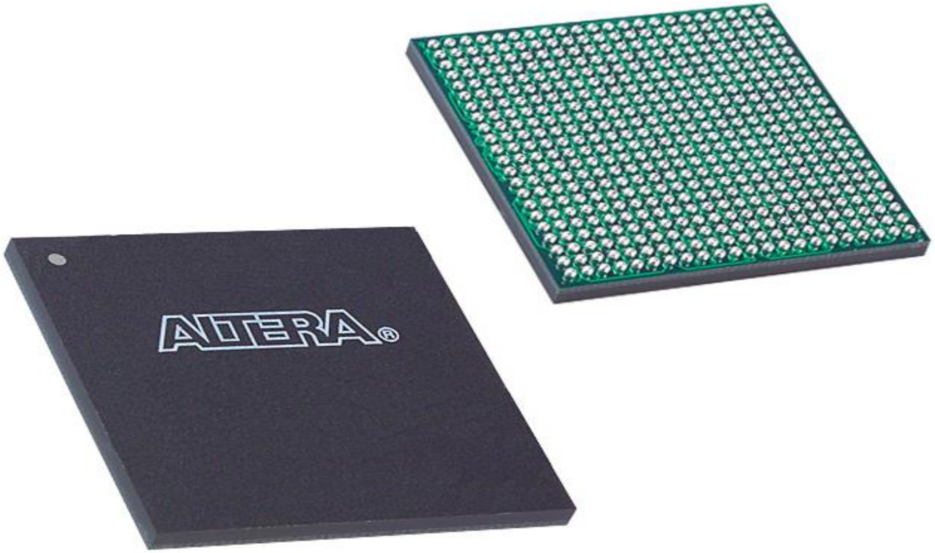
We will work with Altera FPGAs

Agenda

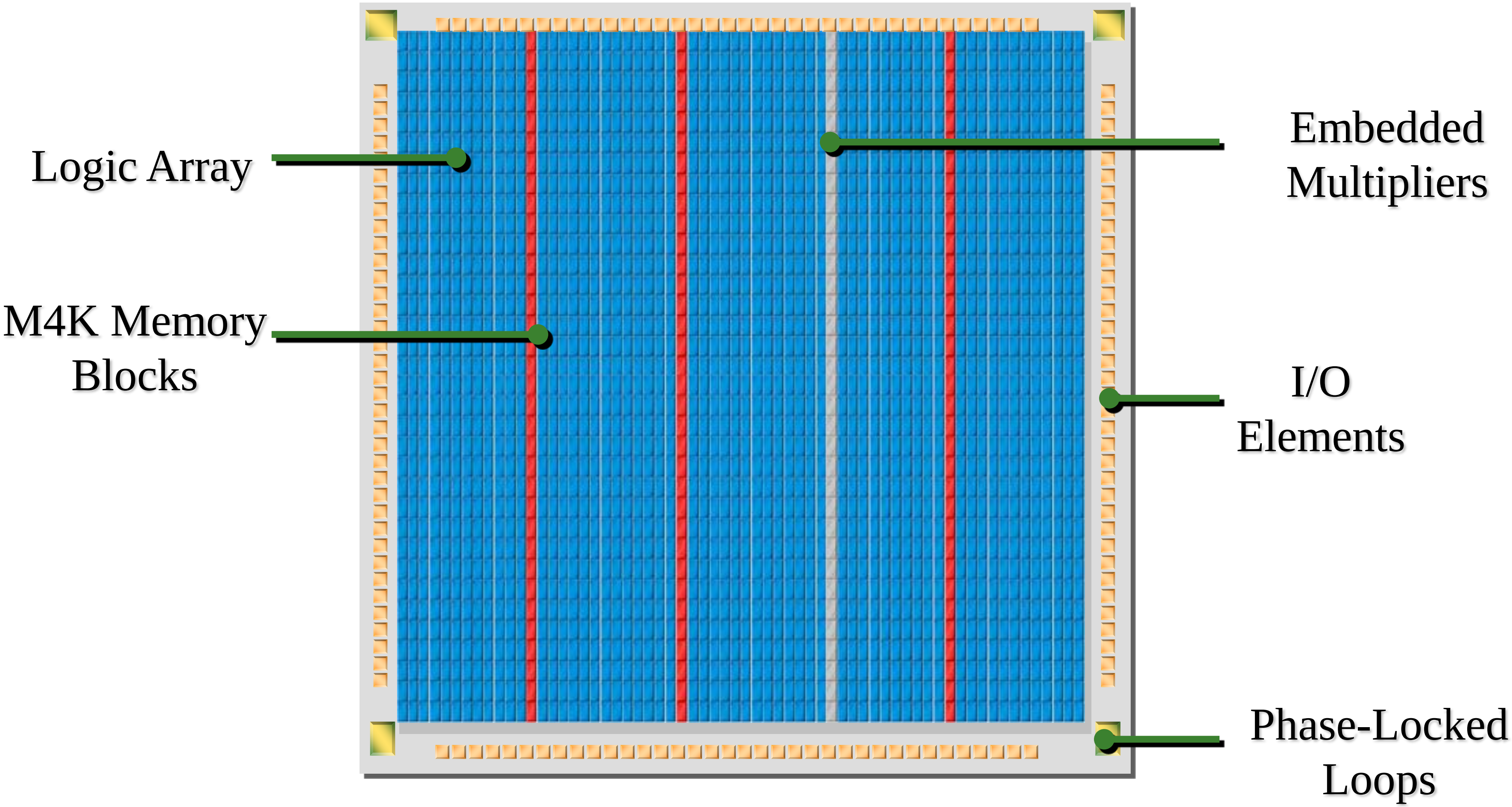
Altera Cyclone II 20 FPGA

Cyclone II - 20

- 18,752 LEs
- 52 M4K RAM blocks
- 240K total RAM bits
- 52 9x9 multipliers
- 4 PLLs
- 16 Clock networks
- 315 user I/O pins
- SRAM Based volatile configuration

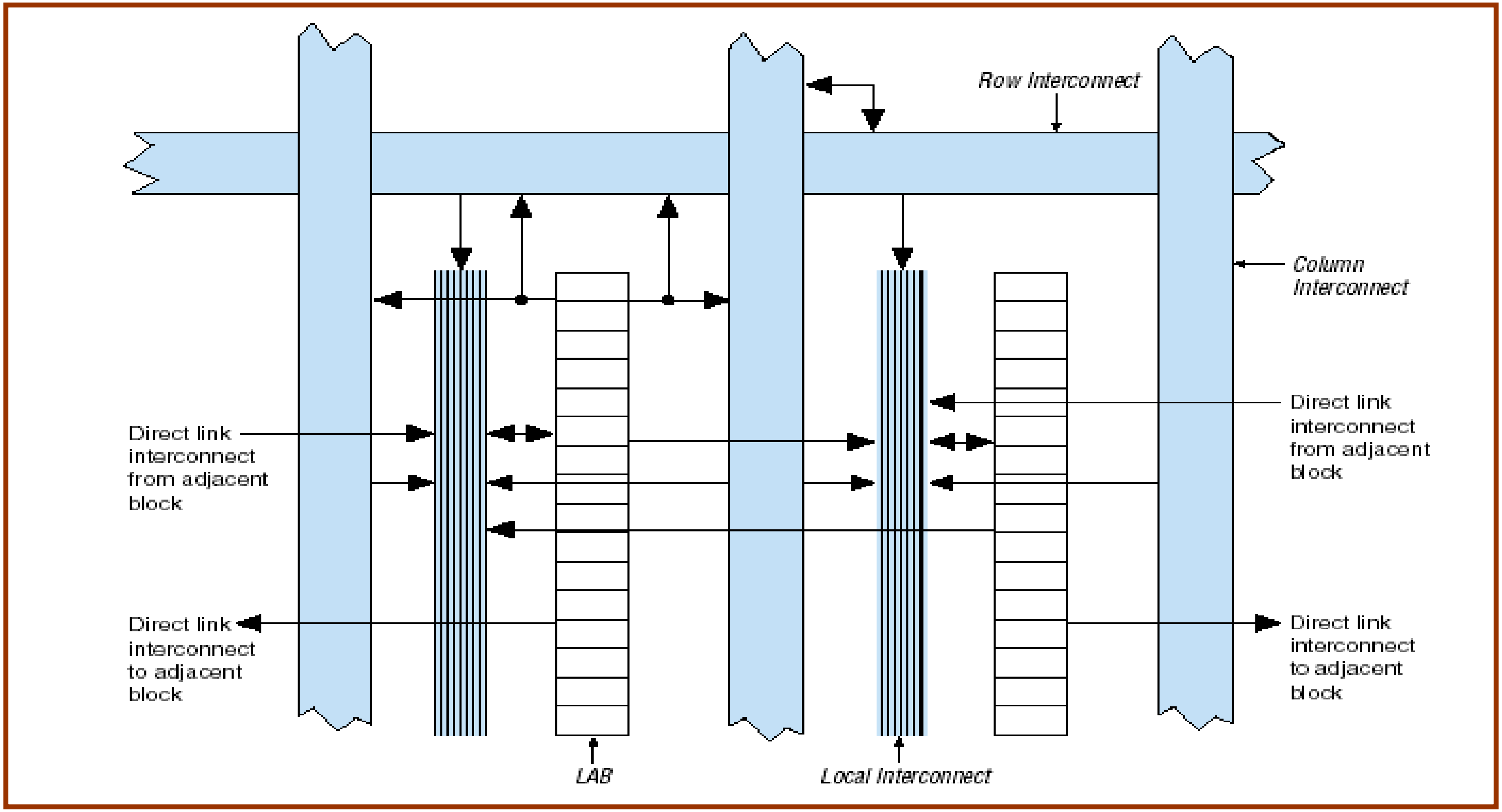


Cyclone II Internals



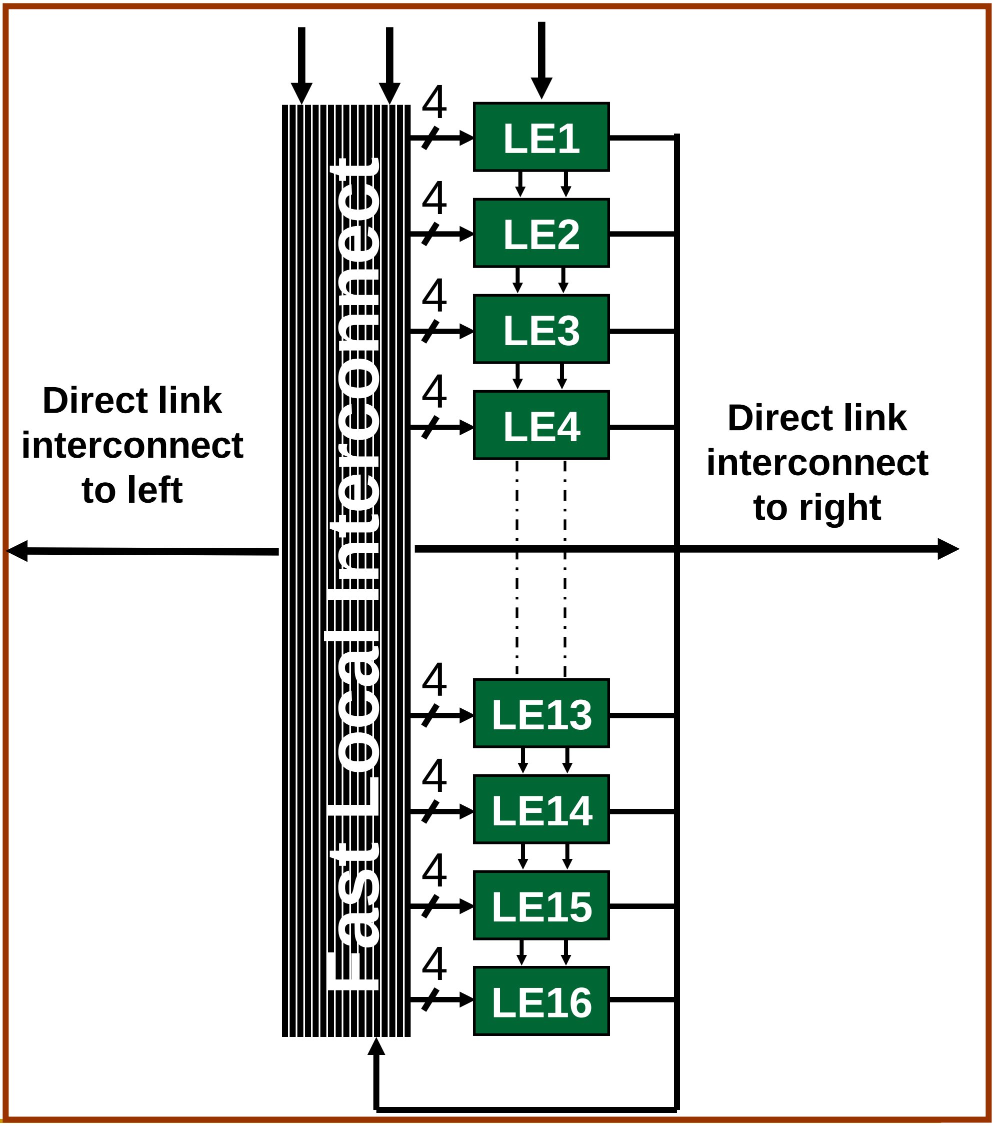
Cyclone II Logic Array

- Build of LABs (logic array blocks) and reconfigurable interconnect

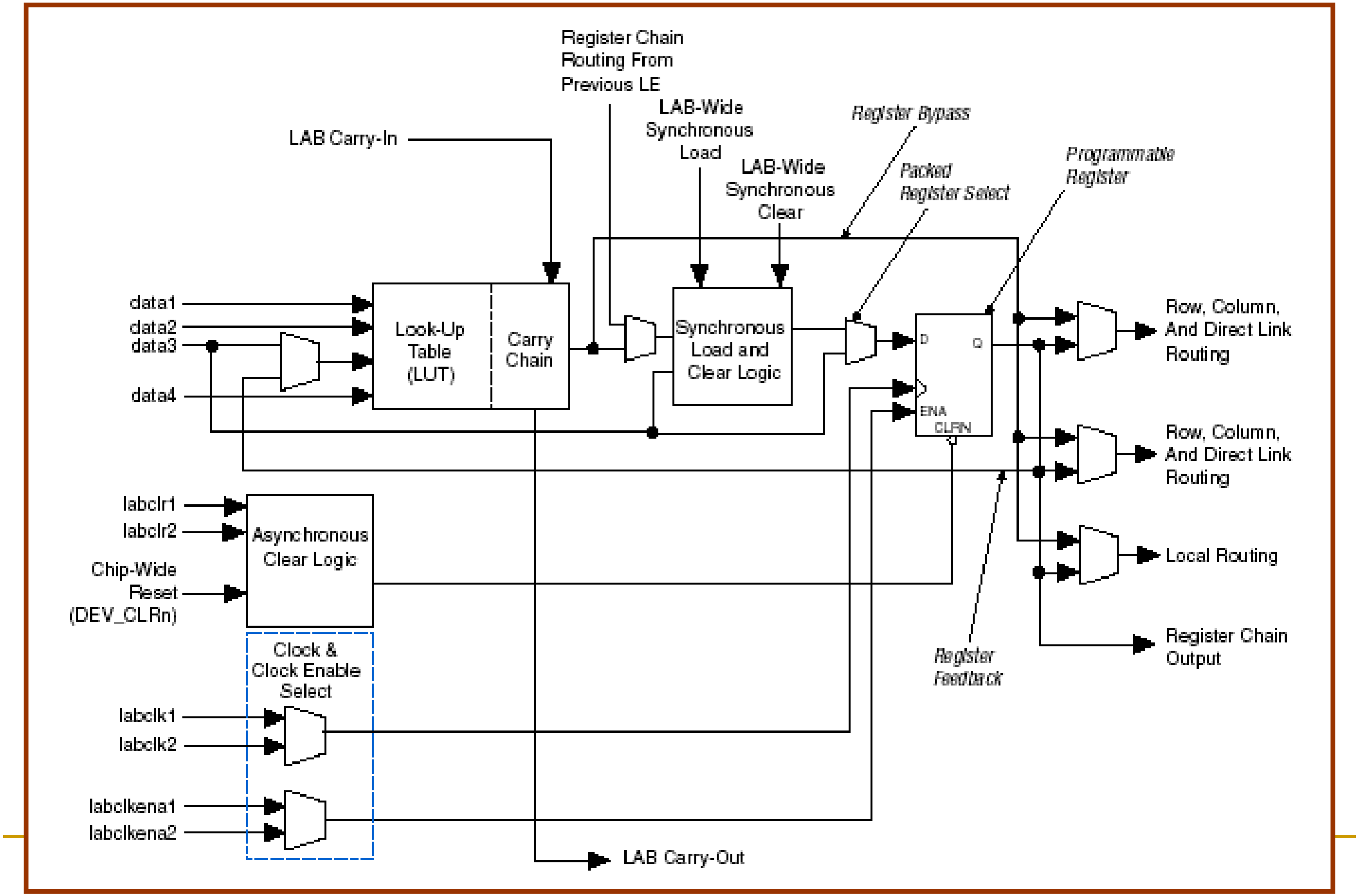

Row Interconnect
Column Interconnect
Direct link interconnect from adjacent block
Direct link interconnect from adjacent block
Direct link interconnect to adjacent block
Direct link interconnect to adjacent block
LAB
Local Interconnect

Cyclone II Logic Array Block (LAB)

- 16 LEs
- Local Interconnect
- LE carry chains
- Register chains
- LAB Control Signals
  - 2 CLK
  - 2 CLK ENA
  - 2 ACLR
  - 1 SCLR
  - 1 SLOAD

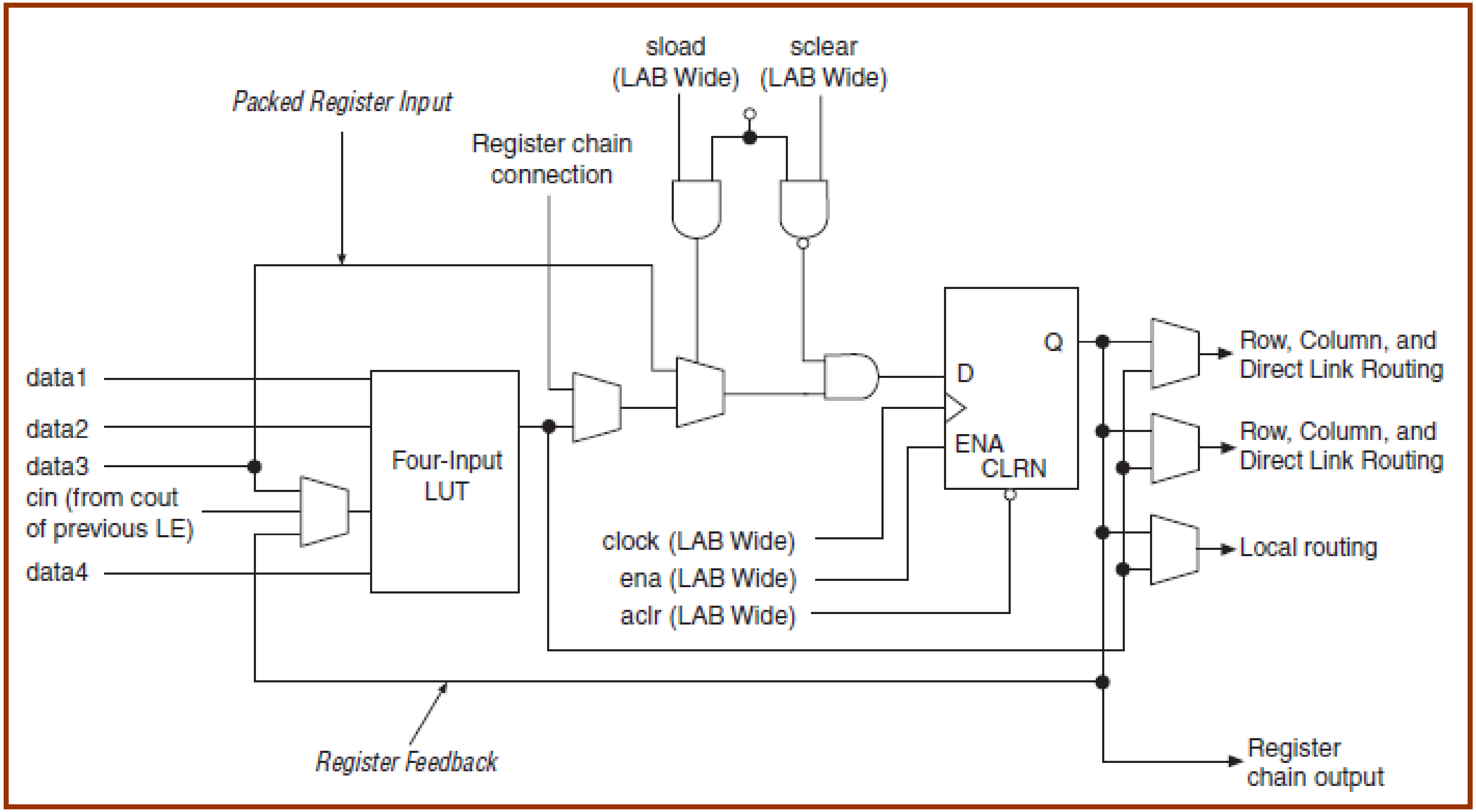

Fast Local Interconnect
4
LE1
4
LE2
4
LE3
4
LE4
4
LE13
4
LE14
4
LE15
4
LE16
Direct link interconnect to left
Direct link interconnect to right

Cyclone II Logic Element (LE)



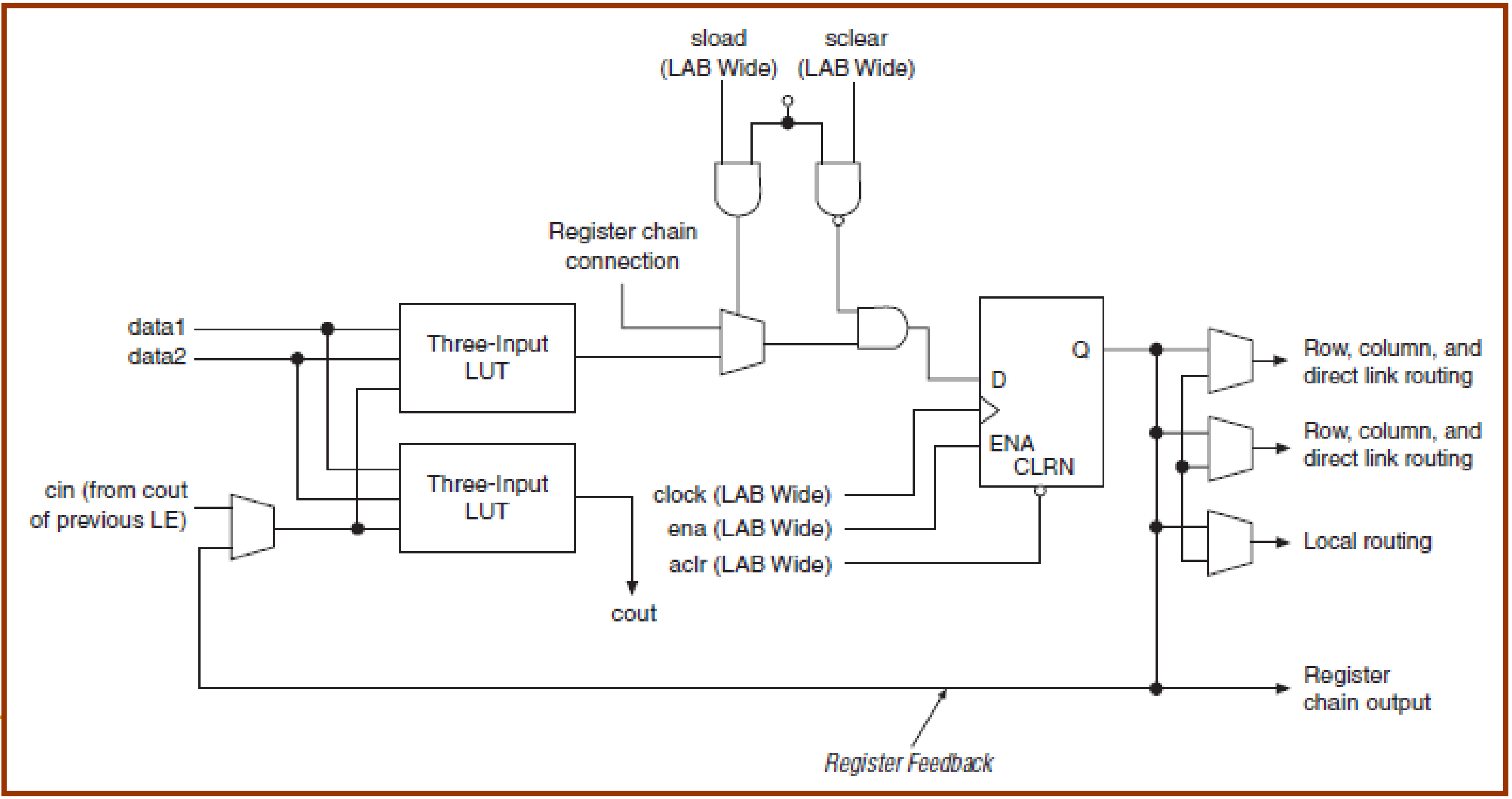
LE in Normal Mode

- Suitable for general logic applications and combinational functions.



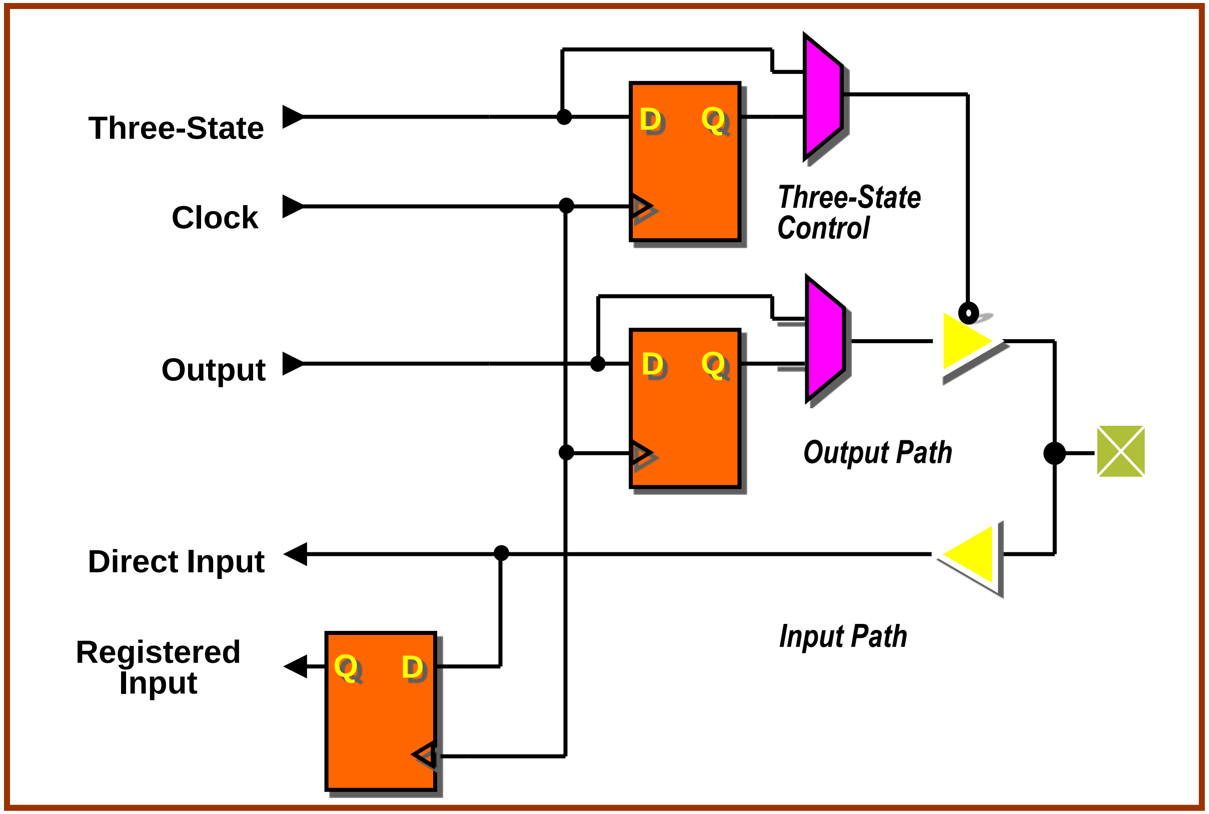
LE in Arithmetic Mode

- Ideal for implementing adders, counters, accumulators, and comparators.



Cyclone II I/O Features

- In/Out/Tri-state
- Fast Registers option

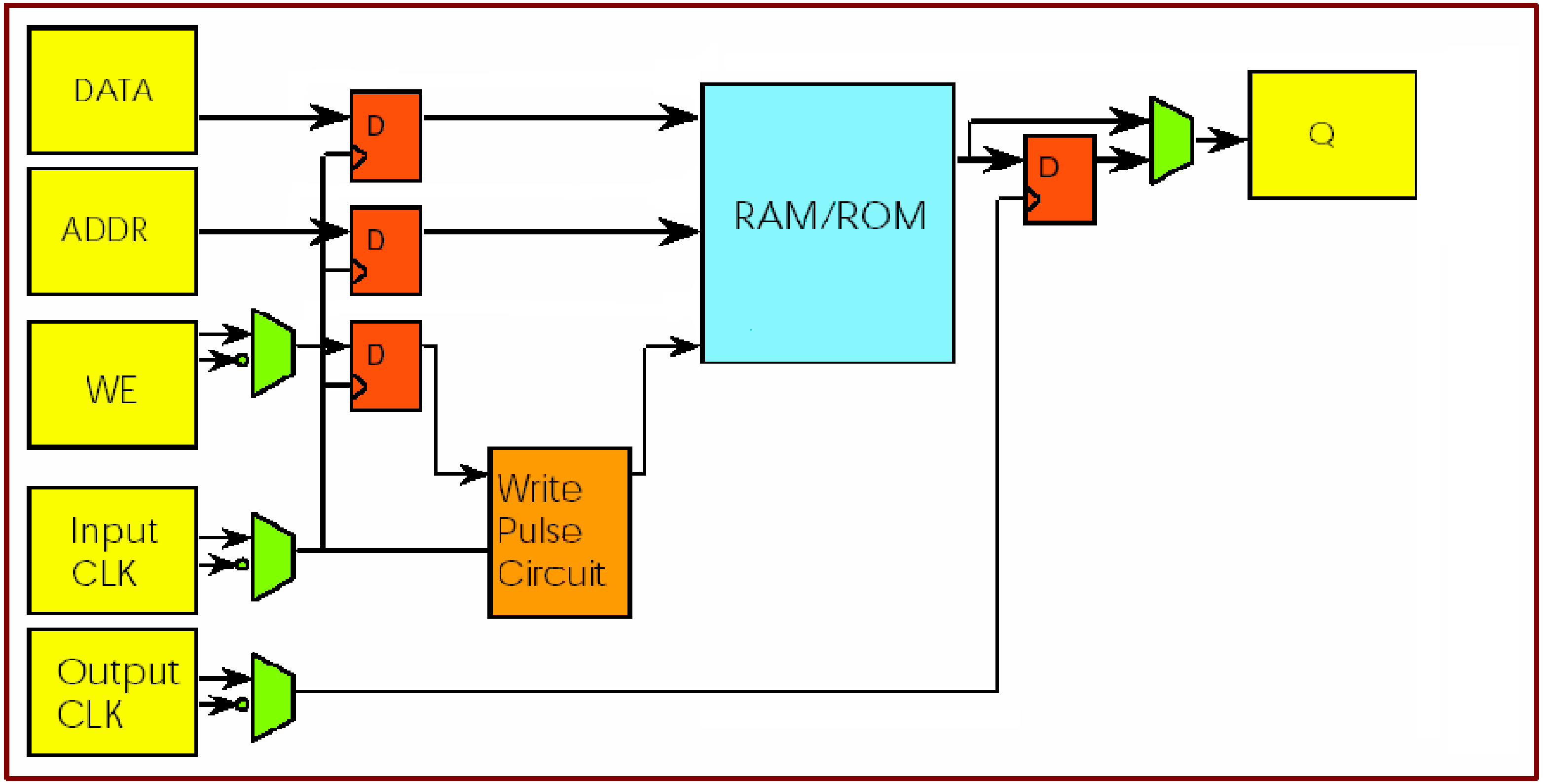


M4K Memory Blocks

- True Dual port RAM/ROM with dual clock
- Variable data width
  - 4K×1, 2K×2, 1K×4, 512×8, 512×9, 256×16, 256×18
  - 128×32, 128×36 (not available in true dual-port mode)
- Input data and address are registered
  - 1 Clock Write latency
- Output data can be registered
  - **Read latency of 1 or 2 clocks**
- Byte Enable

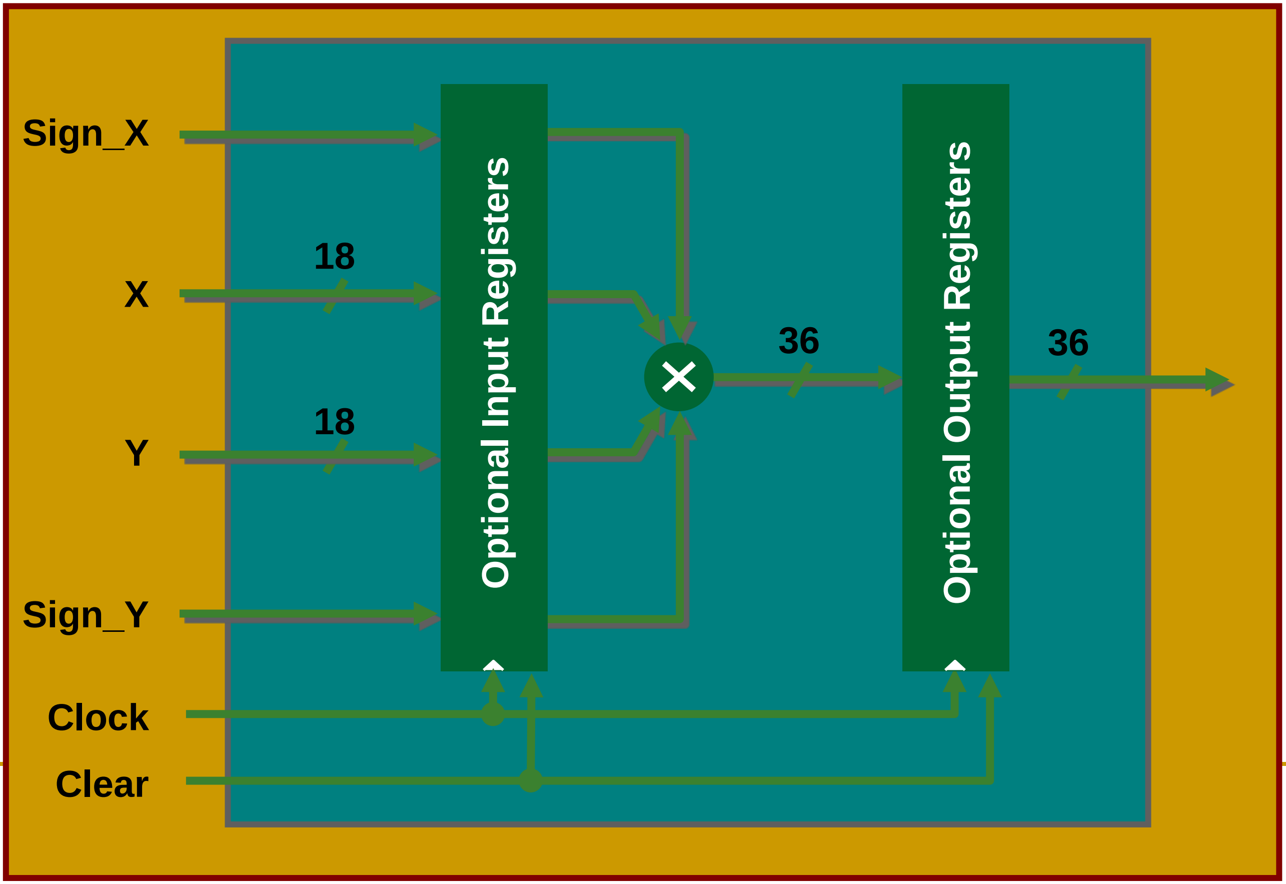
Cyclone II Memory Structure

- Read latency of 1-2 clocks
- Write latency of 1 clocks



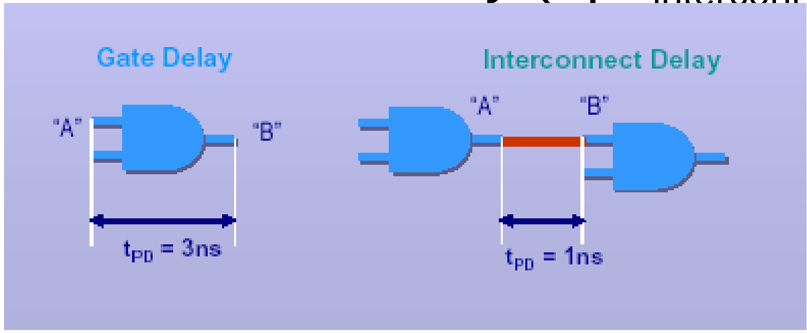
Cyclone II Multipliers

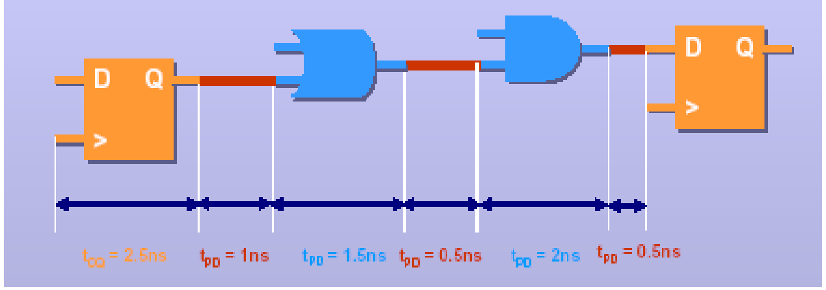
- 18x18 or 2 9x9 modes
- Up to 250MHz Performance



Delays and maximal frequency

- Gate delay (Tpd_{logic}) – Delay of logic element
- DFF delay (Tco)
- Setup time (Tsu) Very small.
- Interconnect delay ($Tpd_{interconnect}$)

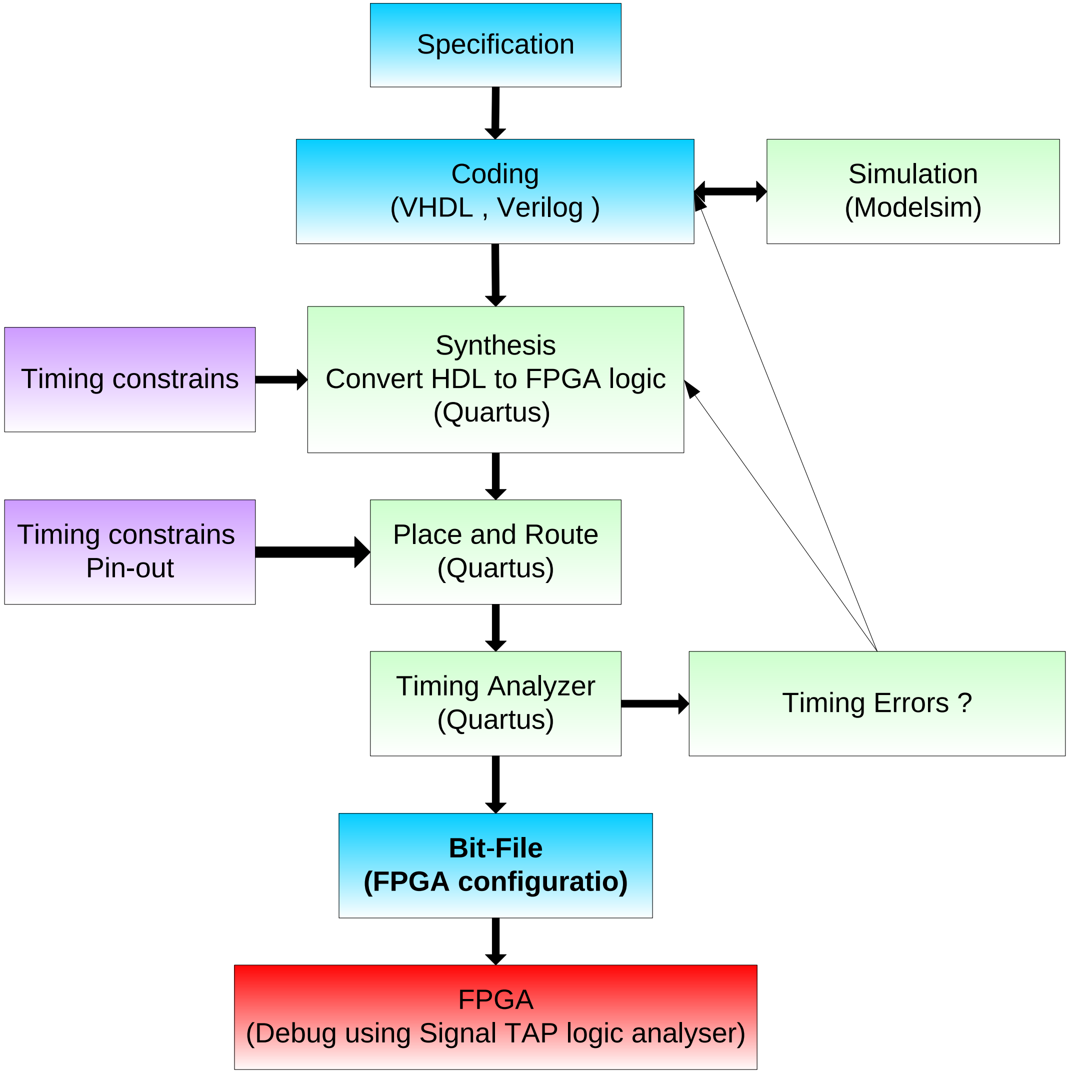




$$1/F_{max} = Tco + Tpd_{logic} + Tpd_{\ interconnect}$$

Maximum Frequency is the fastest speed a circuit containing flip-flops can operate.

Design flow


Specification
Coding
(VHDL , Verilog)
Simulation
(Modelsim)
Timing constrains
Synthesis
Convert HDL to FPGA logic
(Quartus)
Timing constrains
Pin-out
Place and Route
(Quartus)
Timing Analyzer
(Quartus)
Timing Errors ?
Bit-File
(FPGA configuratio)
FPGA
(Debug using Signal TAP logic analyser)

Design Rules

	ASIC	**FPGA**
Adder	CLA	Ripple Carry
Latch	Not Recommended	Not Recommended
Gated clock	Commonly used	Unacceptable
Tri-State	Commonly used	Only at I/O
Async RAM	Commonly used	Only Small

Break

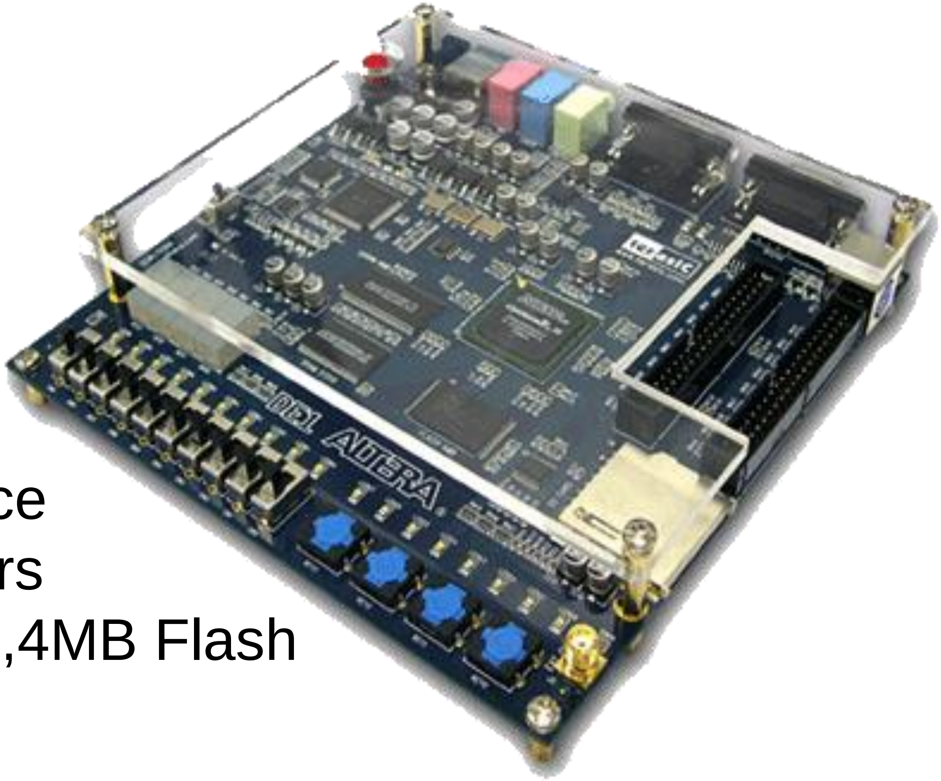
Agenda

Design Flow

- Coding and Compiling

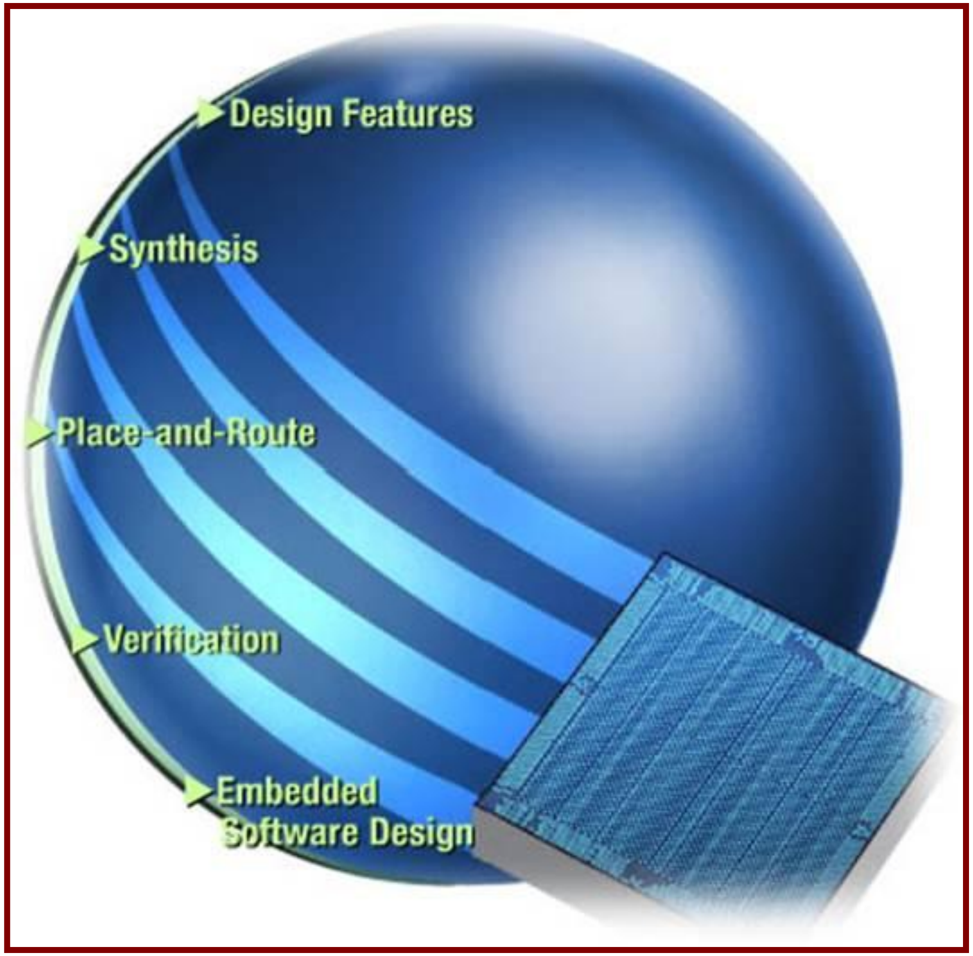
Altera DE1 FPGA Board

- Cyclone II EP2C20F484C6 FPGA
- 50MHz,27MHz and 24MHz oscillators for clock sources
- 4 pushbuttons
- 10 toggle switches
- 10 red and 8 Green LEDs
- 24-bit audio CODEC
- VGA DAC
- RS-232 interface
- SD Card socket
- PS/2 mouse/keyboard Interface
- Two 40-pin Expansion Headers
- 512KB SRAM , 8MB SDRAM ,4MB Flash



Quartus II Version 10.1

- Synthesis tool
- Place and Route
- Simulator
- Debugger
- Programmer
- And much more

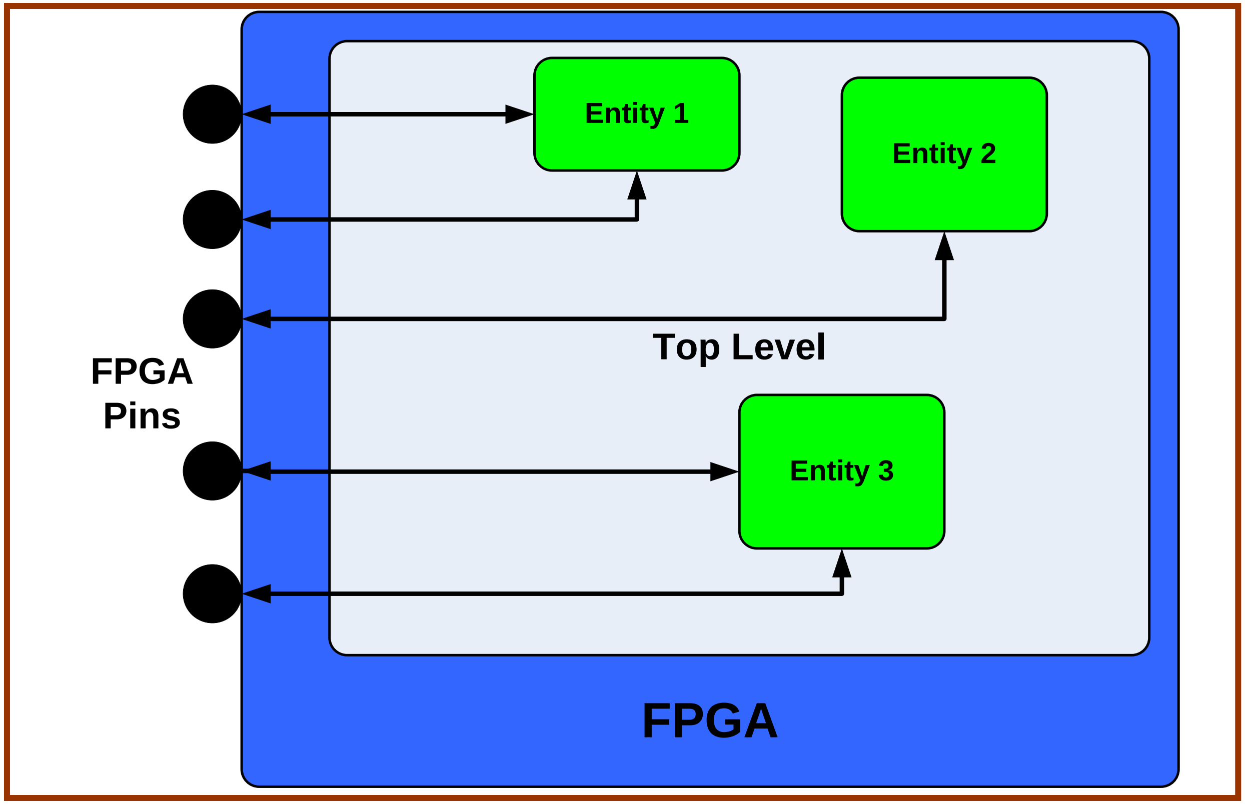


Project Files description

- .qpf Project file
- .qsf Settings file (timing , constrains , pin)
- .vhd Design file , must be at least a top level design file , its ports are directly connected to physical pins
- .stp Signal Tap file
- .vwf Simulation Waveform file
- .sof FPGA programming file

What is a Top Level

- Serves as a top level entity
- Connects to FPGA physical pins



Example application

- 32bit behavioral counter with enable
- 8 MSB connected to green LEDs
- Enable connected to switch
- Clock to 50MHz onboard oscillator

Example Code

```
library ieee;
use ieee.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity counter is port (
        clk,enable : in std_logic;
        q          : out std_logic_vector (7 downto 0));
end entity;

architecture rtl of counter is
    signal q_int : std_logic_vector (31 downto 0);
begin
    process (clk)
    begin
        if (rising_edge(clk)) then
            if enable = '1' then
                q_int <= q_int + 1;
            end if;
        end if;
    end process;
    q <= q_int(31 downto 24); -- Output only 8MSB
end rtl;
```

Exercise 1 – Coding and compiling

- Duration 5 minutes
- Purpose –
  - create a project,
  - add a VHDL file and compile
  - view the RTL drawing
  - Add system constraints
  - Perform full synthesis

Exercise 1 – Starting New Project

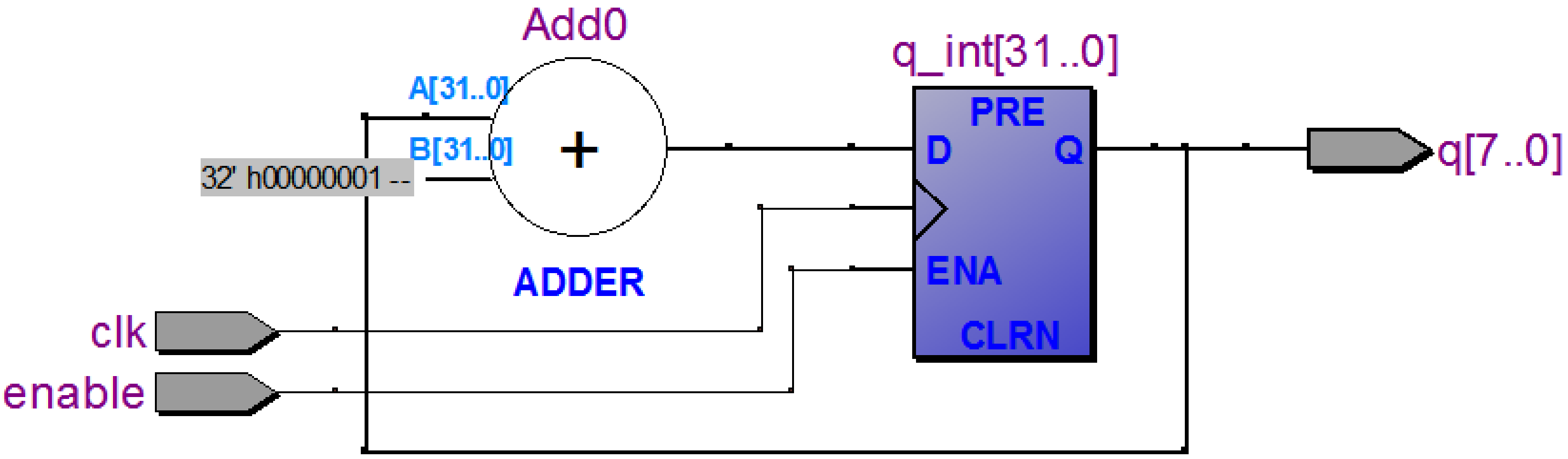
- Open Quartus II
- Start Wizard File->New Project Wizard and Click Next ,
- Specify Name of Project inside directory \lab2\ex1\ and click Next
- Add existing file “ex1.vhd” and click Next
- Specify FPGA
  - Cyclone II , EP2C20F484C6
- Click Next , Next and Finish

Exercise 1 – Compilation

- We can also create new files : File->New
- Set compilation order Assignments ->Settings->Files
- Change Top level to the name of the entity in ex1.vhd Assignments->Settings->General ->Top-level entity
- Analyze the project : Push Button
- View resource utilization at "Compilation Report"
- How many elements? What is the area?

Exercise 1 – Viewing Synthesis results

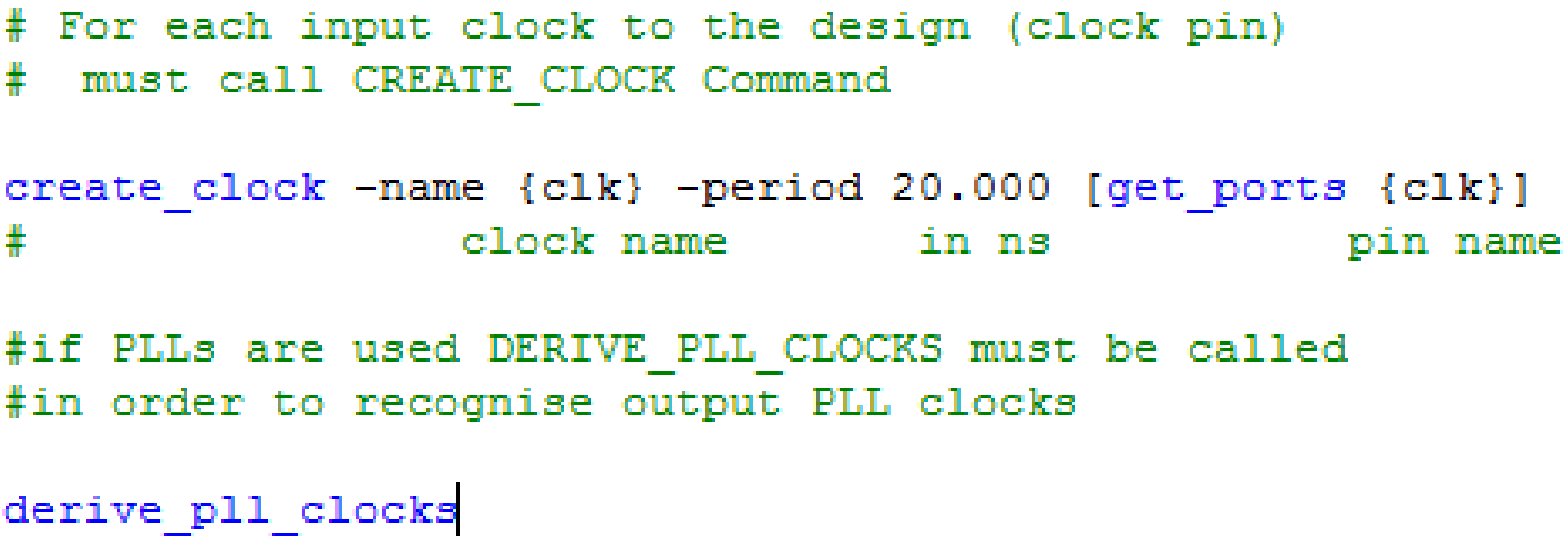
- View the RTL (Register Transfer Level)
  - Tools -> Netlist Viewers -> RTL viewer



- Technology Synthesis
  - Tools -> Netlist Viewers -> Technology map viewer

Exercise 1 – Setting system constrains

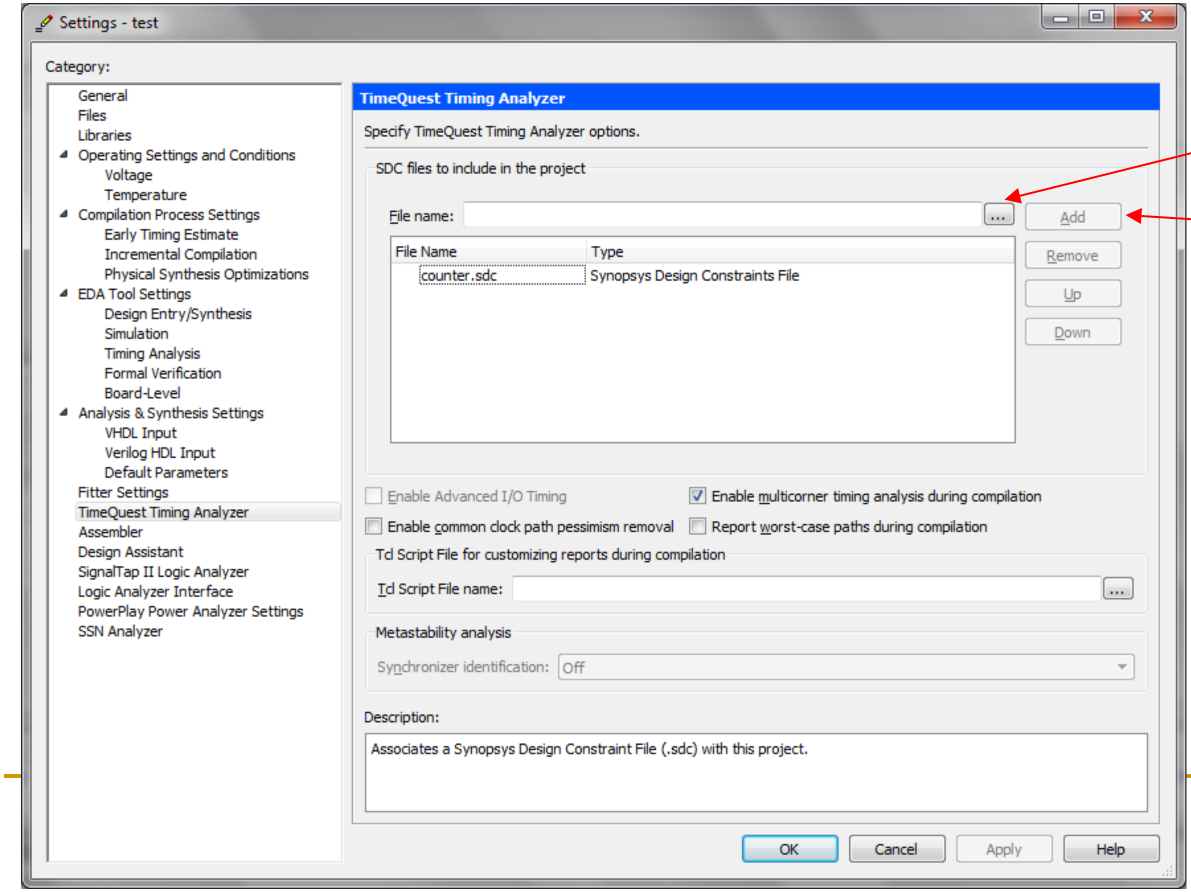
- Create new SDC file : File -> New ->Other Files -> Synopsys Design Constrains
- Type clocks constrains and Save , Example :



```
# For each input clock to the design (clock pin)
#  must call CREATE_CLOCK Command

create_clock -name {clk} -period 20.000 [get_ports {clk}]
#                clock name       in ns            pin name

#if PLLs are used DERIVE_PLL_CLOCKS must be called
#in order to recognise output PLL clocks

derive_pll_clocks
```

Exercise 1 – Setting system constrains

- Go to : Assignments->Settings->TimeQuest Timing Analyzer
- Add the SDC file



Settings - test

Category:

TimeQuest Timing Analyzer

Specify TimeQuest Timing Analyzer options.

SDC files to include in the project

File name: [ ] ...

File Name	Type
counter.sdc	Synopsys Design Constraints File

□ Enable Advanced I/O Timing
☑ Enable multicorner timing analysis during compilation
□ Enable common clock path pessimism removal
□ Report worst-case paths during compilation

Tcl Script File for customizing reports during compilation

Tcl Script File name: [ ] ...

Metastability analysis

Synchronizer identification: Off

Description:

Associates a Synopsys Design Constraint File (.sdc) with this project.

Click Browse , select file , then click Add

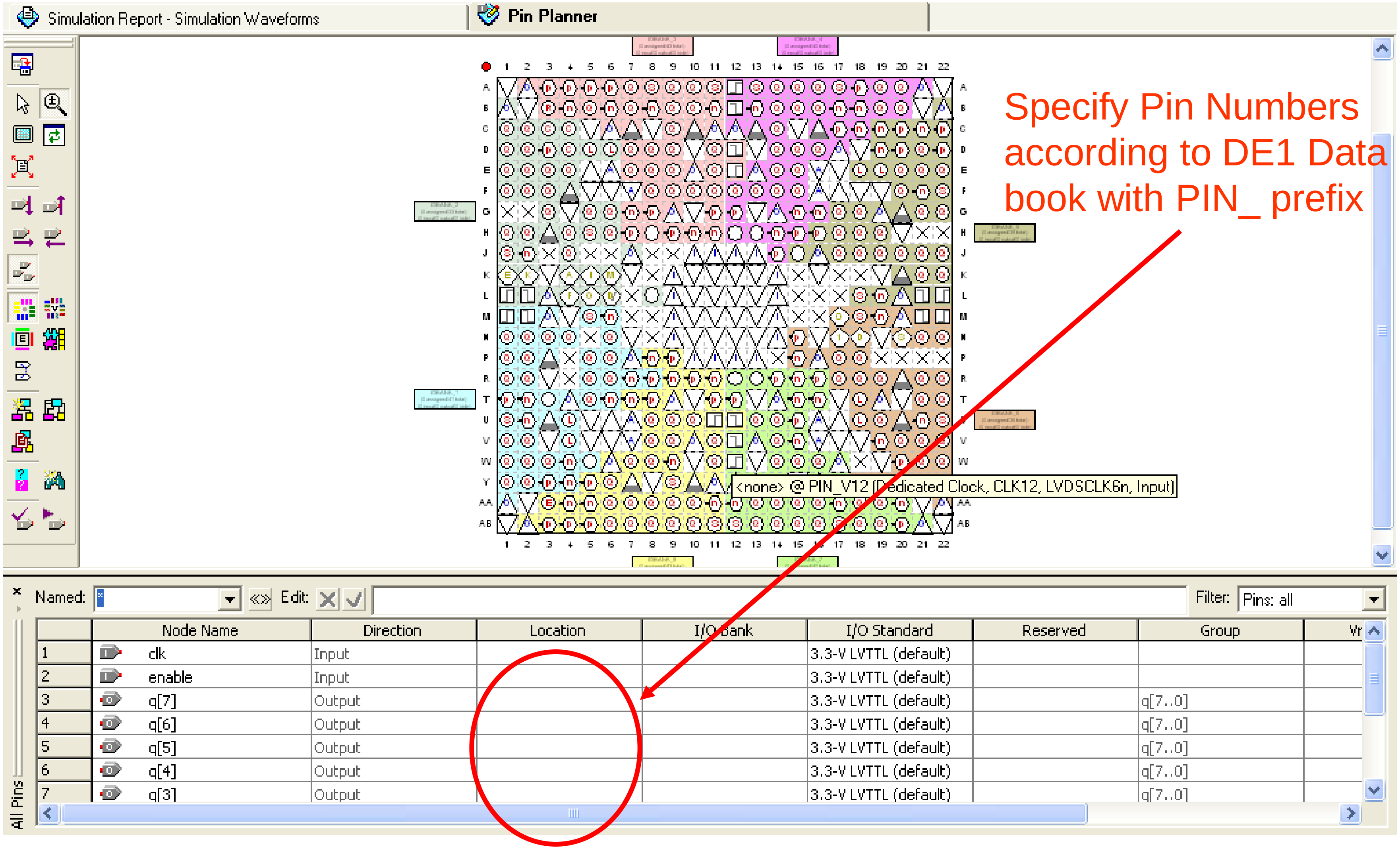
Agenda

Design Flow

- Pin Assignment

Pin Assignments

- Open Pin Planner (Assignments -> Pin Planner)



<none> @ PIN_V12 (Dedicated Clock, CLK12, LVDSCLK6n, Input)

Specify Pin Numbers according to DE1 Data book with PIN_ prefix

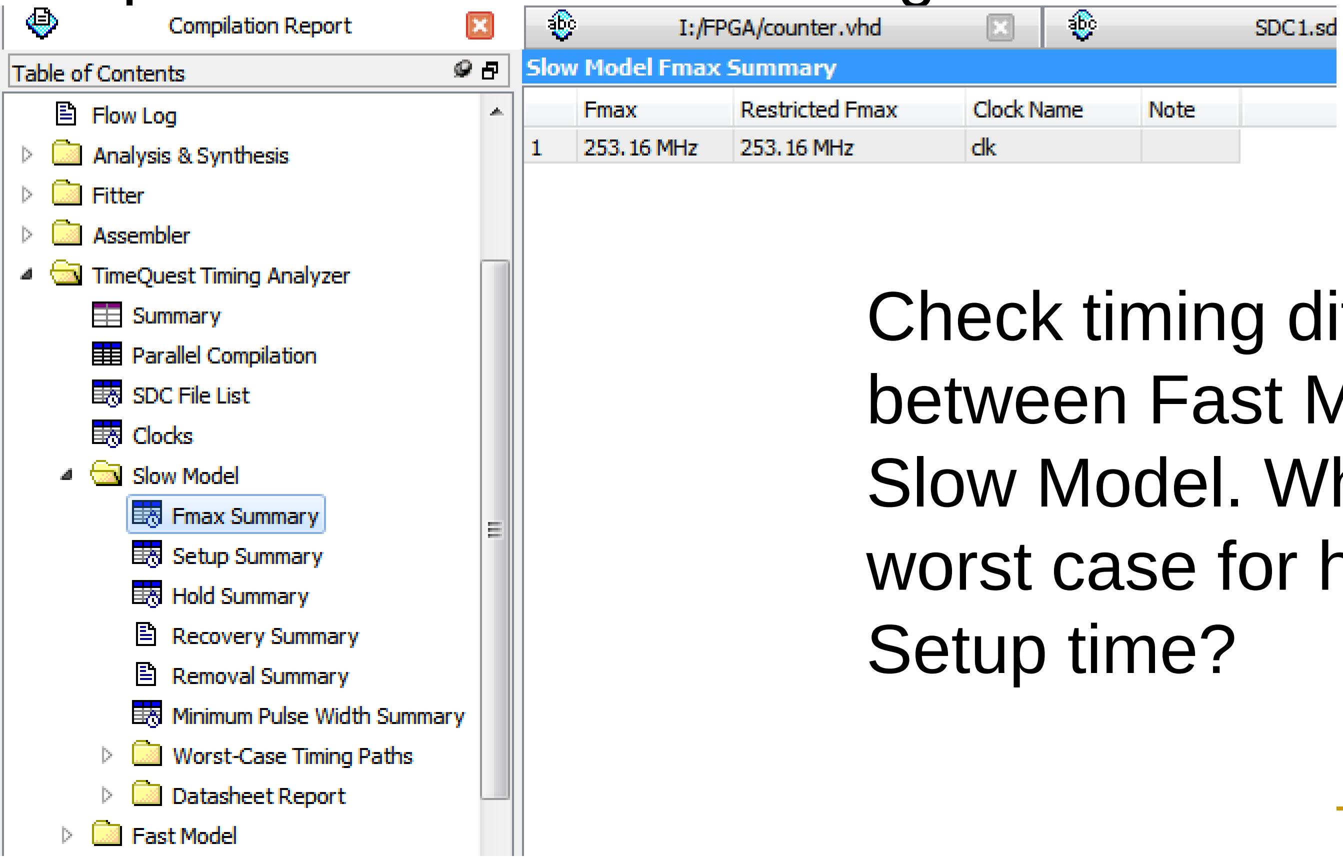
	Node Name	Direction	Location	I/O Bank	I/O Standard	Reserved	Group	Vr
1	clk	Input			3.3-V LVTTL (default)			
2	enable	Input			3.3-V LVTTL (default)			
3	q[7]	Output			3.3-V LVTTL (default)		q[7..0]	
4	q[6]	Output			3.3-V LVTTL (default)		q[7..0]	
5	q[5]	Output			3.3-V LVTTL (default)		q[7..0]	
6	q[4]	Output			3.3-V LVTTL (default)		q[7..0]	
7	q[3]	Output			3.3-V LVTTL (default)		q[7..0]	

Pin Assignments

- clk to Clock_50
- enable to Sw[0]
- q[0]..q[7] to LED Green[0]..[7]

Full compilation

- Press ▶ Button
- After compilation open timing analyzer in compilation report and see that all timings are OK.



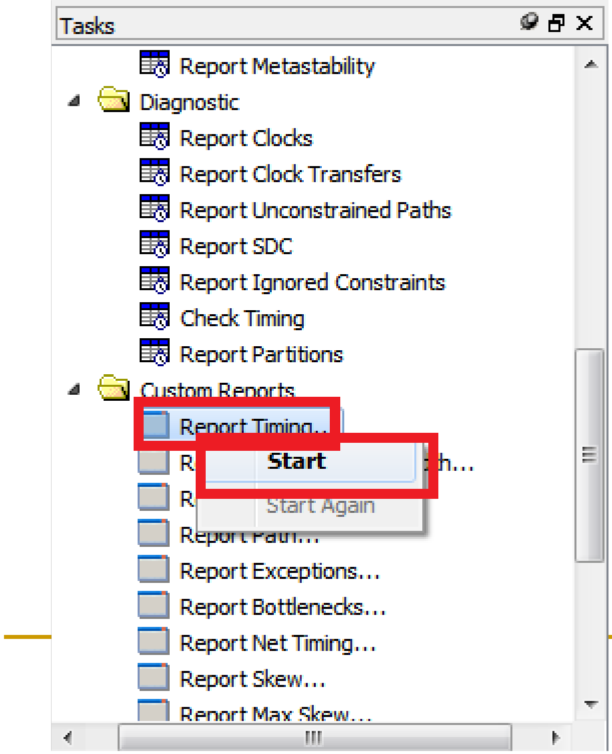
Slow Model Fmax Summary

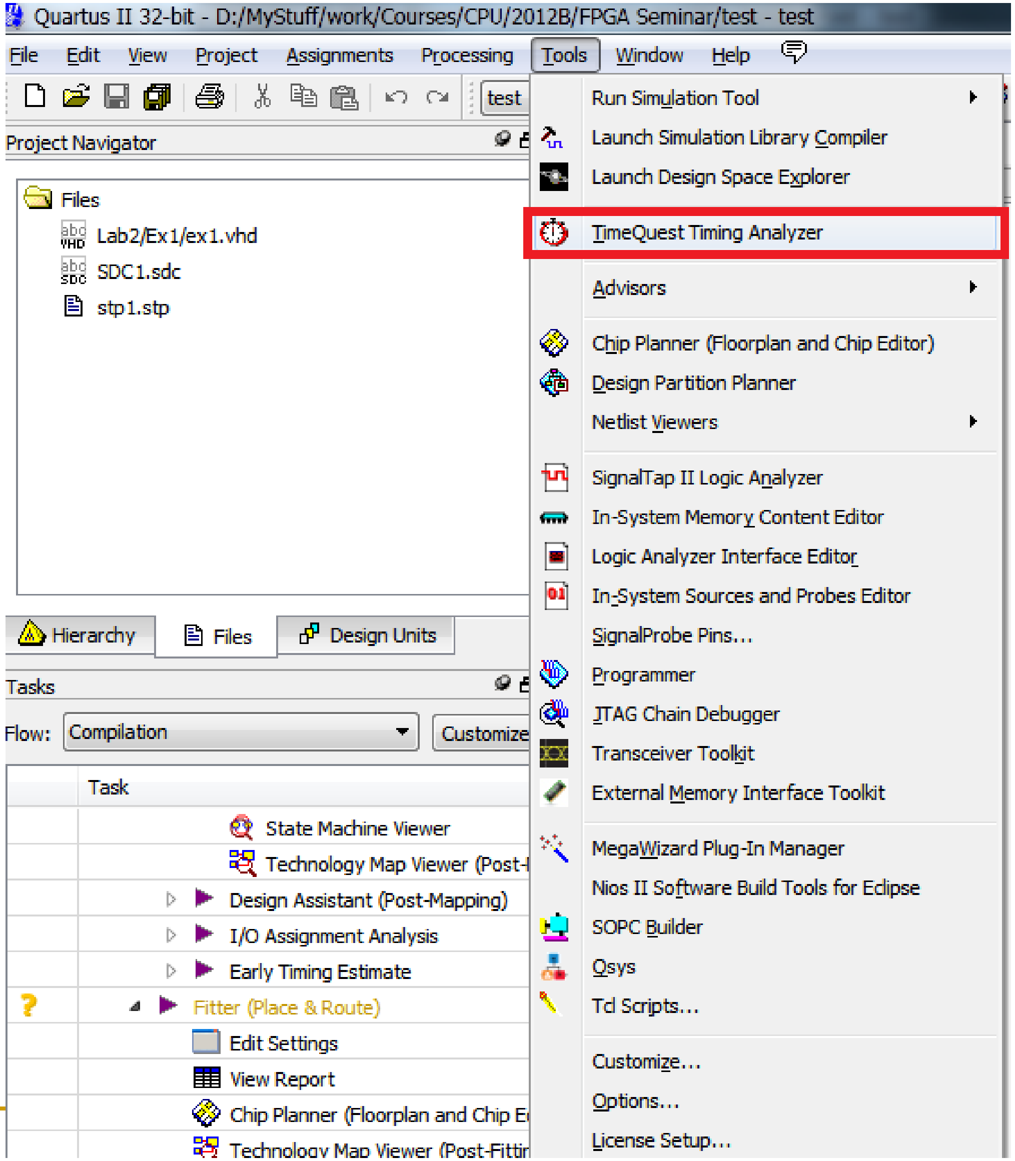
	Fmax	Restricted Fmax	Clock Name	Note
1	253.16 MHz	253.16 MHz	clk	

Check timing difference between Fast Model and Slow Model. Which is the worst case for hold time? Setup time?

Finding Critical Path location

- Start TimeQuest
- Right click Report Timing→Start

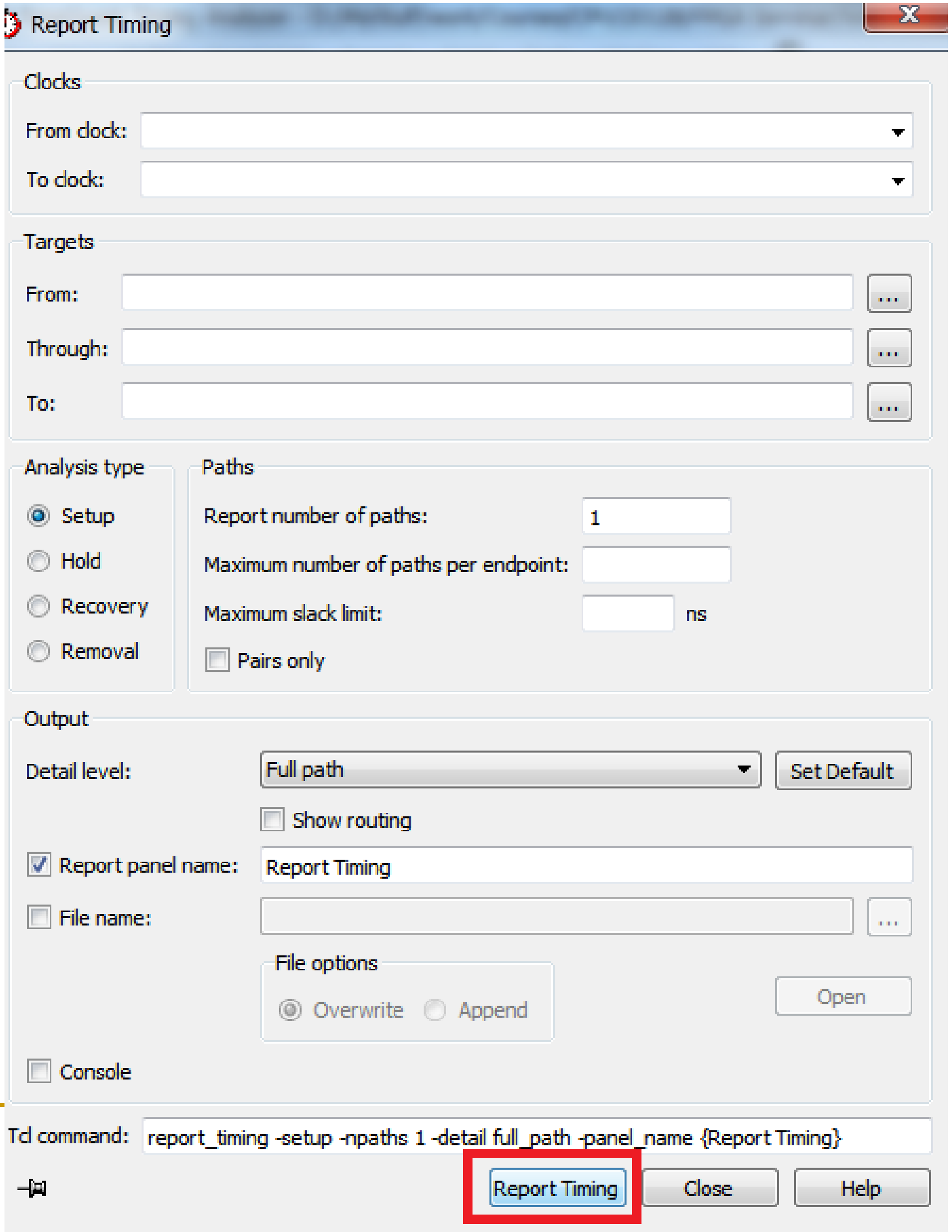




Quartus II 32-bit - D:/MyStuff/work/Courses/CPU/2012B/FPGA Seminar/test - test
File
Edit
View
Project
Assignments
Processing
Tools
Window
Help
test
Project Navigator
Files
Lab2/Ex1/ex1.vhd
SDC1.sdc
stp1.stp
Hierarchy
Files
Design Units
Tasks
Flow:
Compilation
Customize
Task
State Machine Viewer
Technology Map Viewer (Post-
Design Assistant (Post-Mapping)
I/O Assignment Analysis
Early Timing Estimate
Fitter (Place & Route)
Edit Settings
View Report
Chip Planner (Floorplan and Chip E
Technology Map Viewer (Post-Fittin
Run Simulation Tool
Launch Simulation Library Compiler
Launch Design Space Explorer
TimeQuest Timing Analyzer
Advisors
Chip Planner (Floorplan and Chip Editor)
Design Partition Planner
Netlist Viewers
SignalTap II Logic Analyzer
In-System Memory Content Editor
Logic Analyzer Interface Editor
In-System Sources and Probes Editor
SignalProbe Pins...
Programmer
JTAG Chain Debugger
Transceiver Toolkit
External Memory Interface Toolkit
MegaWizard Plug-In Manager
Nios II Software Build Tools for Eclipse
SOPC Builder
Qsys
Tcl Scripts...
Customize...
Options...
License Setup...

Finding Critical Path location

- Choose "Report Timing"



Report Timing

Clocks

From clock:

To clock:

Targets

From:

Through:

To:

Analysis type

● Setup

○ Hold

○ Recovery

○ Removal

Paths

Report number of paths: 1

Maximum number of paths per endpoint:

Maximum slack limit: ns

□ Pairs only

Output

Detail level: Full path

□ Show routing

☑ Report panel name: Report Timing

□ File name:

File options

● Overwrite ○ Append

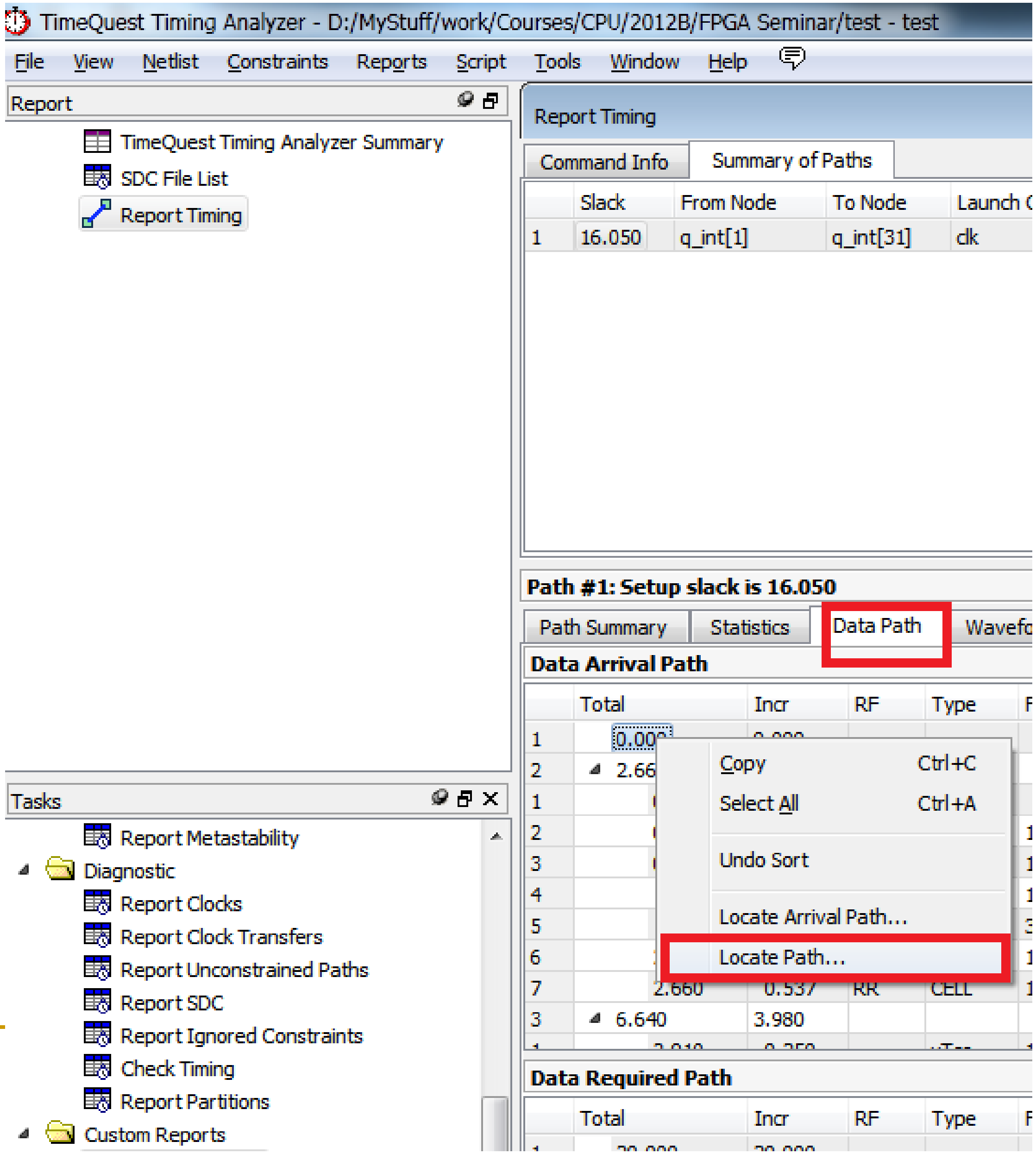
□ Console

Tcl command:

```
report_timing -setup -npaths 1 -detail full_path -panel_name {Report Timing}
```

Finding Critical Path location

- Choose
- “Data Path”
- “Locate Path”

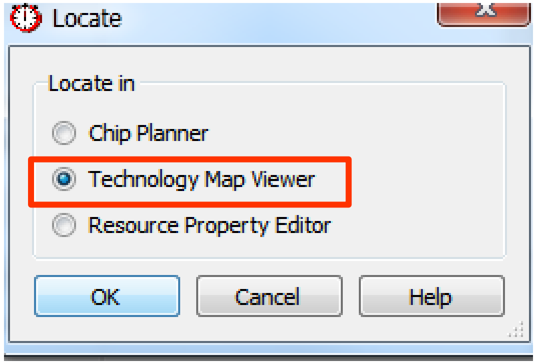


	Slack	From Node	To Node	Launch C
1	16.050	q_int[1]	q_int[31]	clk

Path #1: Setup slack is 16.050

Finding Critical Path location

- Technology Map Viewer



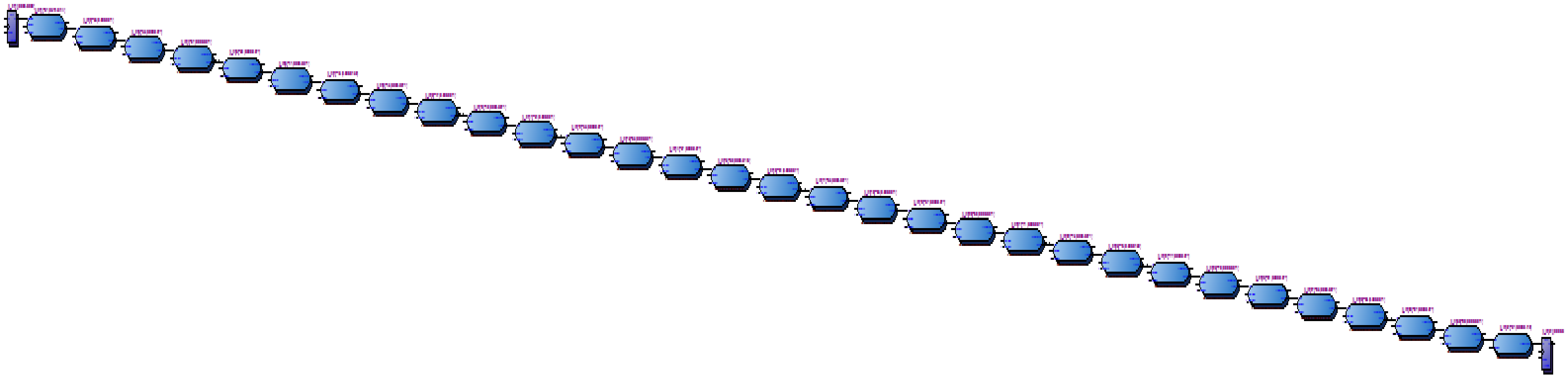
Locate

Locate in

○ Chip Planner

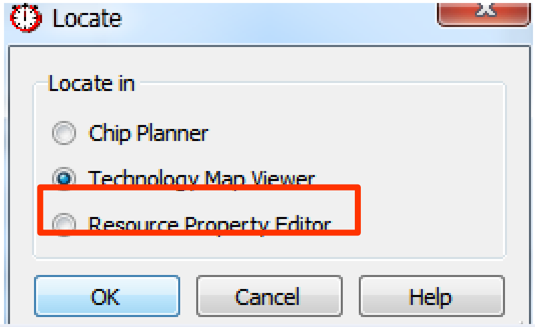
● Technology Map Viewer

○ Resource Property Editor



Finding Critical Path location

- Resource Property Editor



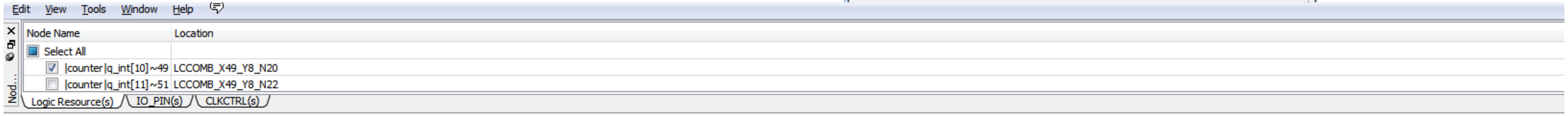
Locate

Locate in

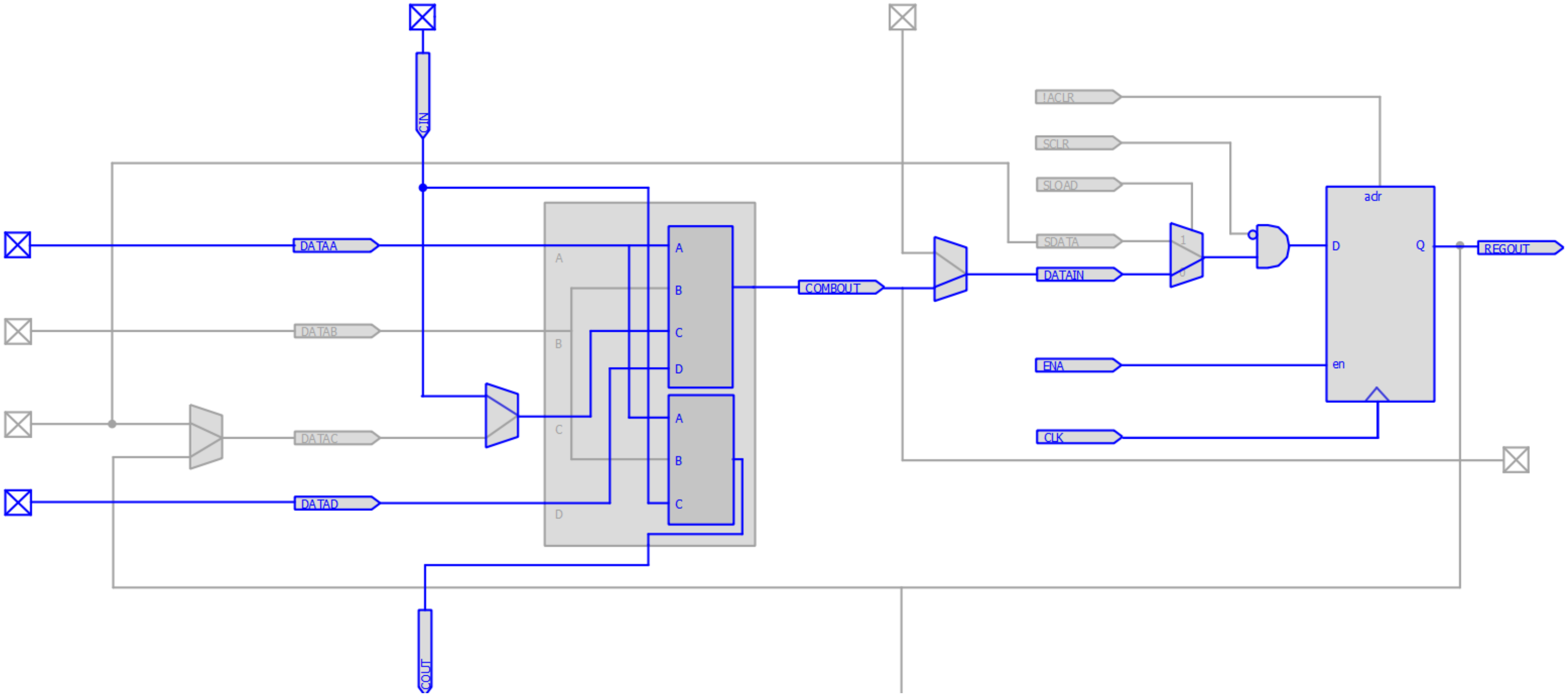
○ Chip Planner

● Technology Map Viewer

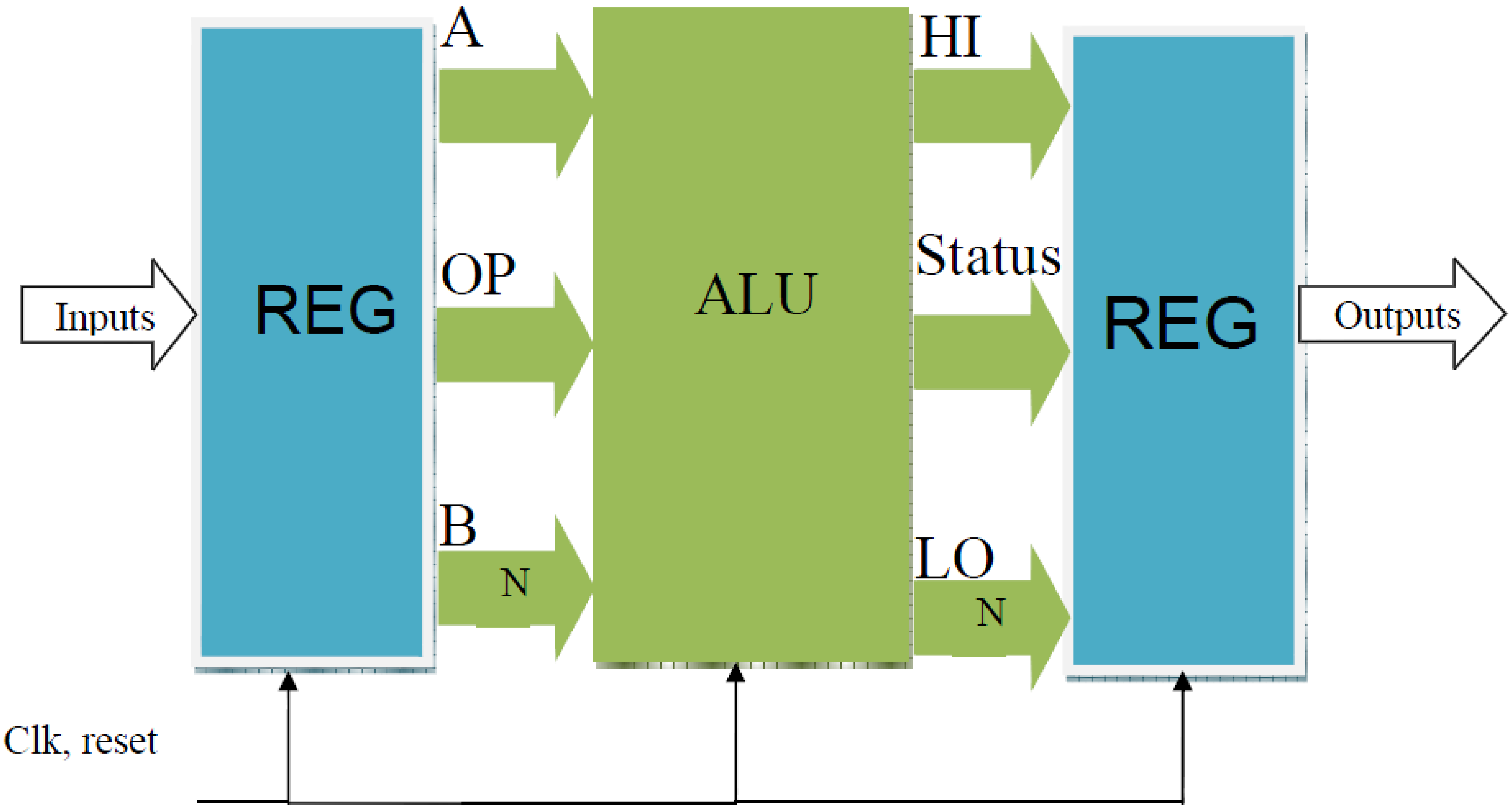
○ Resource Property Editor



Node Name	Location
□ Select All	
☑ |counter|q_int[10]~49	LCCOMB_X49_Y8_N20
□ |counter|q_int[11]~51	LCCOMB_X49_Y8_N22



Critical path - test



break

Agenda

Design Flow

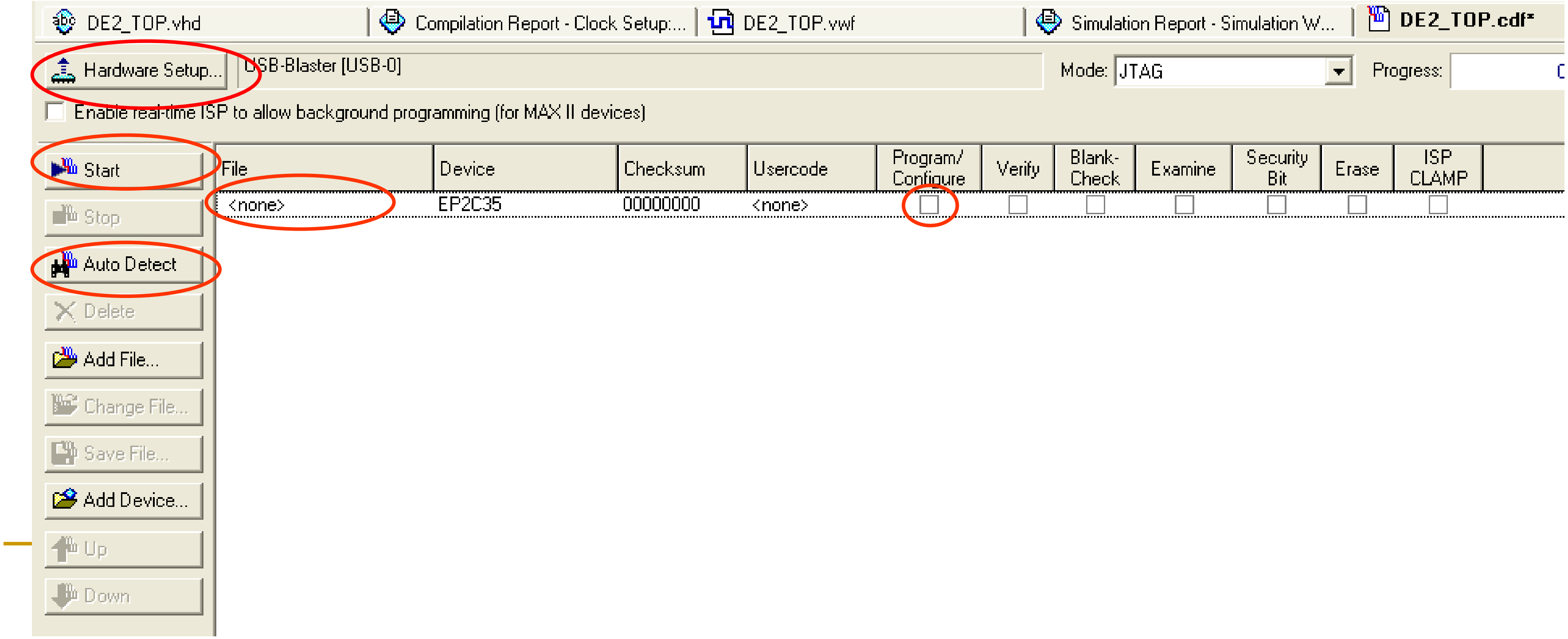
- Configuring the board

Programming the FPGA

- Power supply is Not Required
- Connect DE1 board to PC using USB cable
- Power on the board using RED button
- Push the programmer button in Quartus

Programming the FPGA - cont

- Push Hardware Setup and Select USB-Blaster
- Push the Auto Detect button
- Then double click the <none> and select sof File
- Check the "Program configure" box
- Push Start button



USB-Blaster [USB-0]

Mode: JTAG

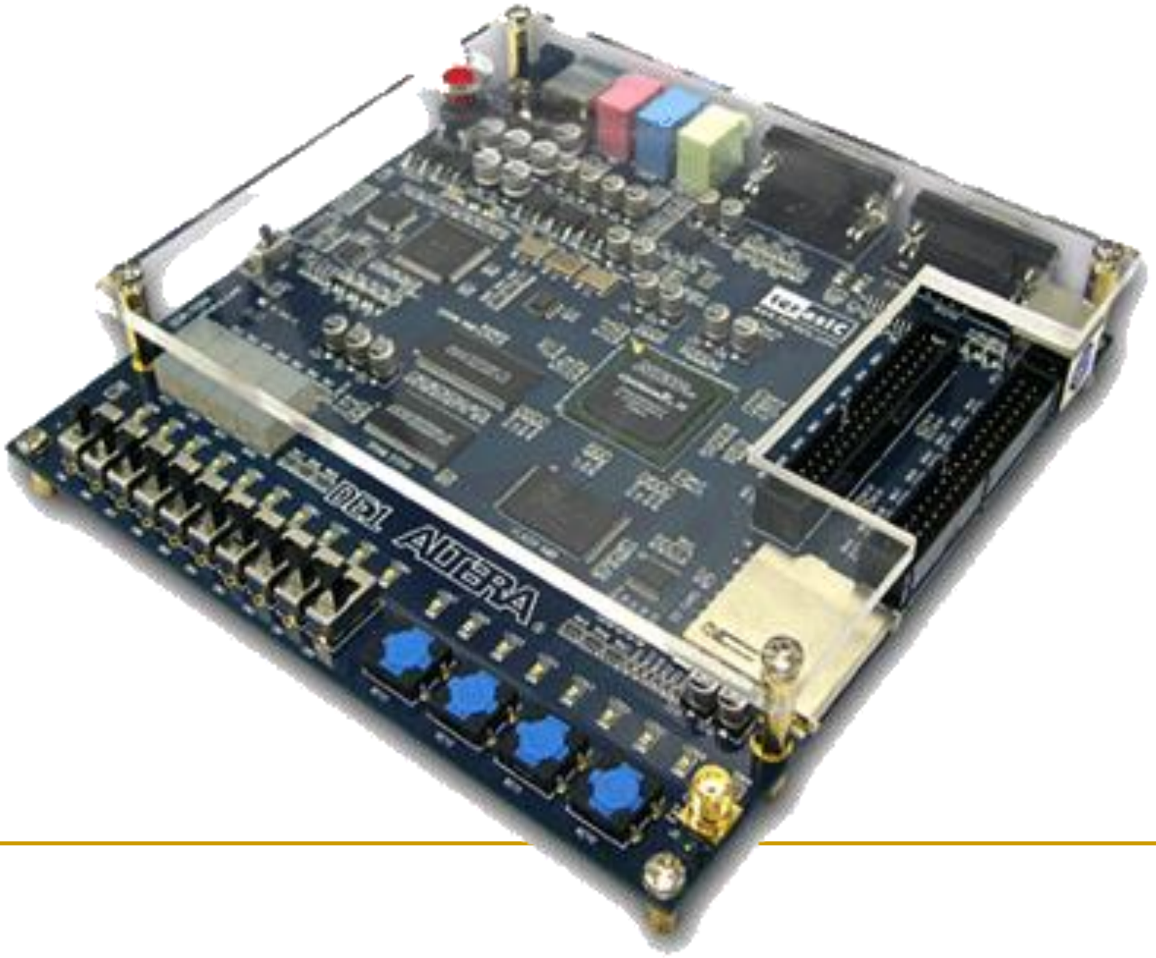
Progress:

□ Enable real-time ISP to allow background programming (for MAX II devices)

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase	ISP CLAMP
<none>	EP2C35	00000000	<none>	□	□	□	□	□	□	□

Turn on the counting machine

- Watch the green LEDs as they represent bits in the counter.
- Turn the counting machine on and off by pressing SW0.



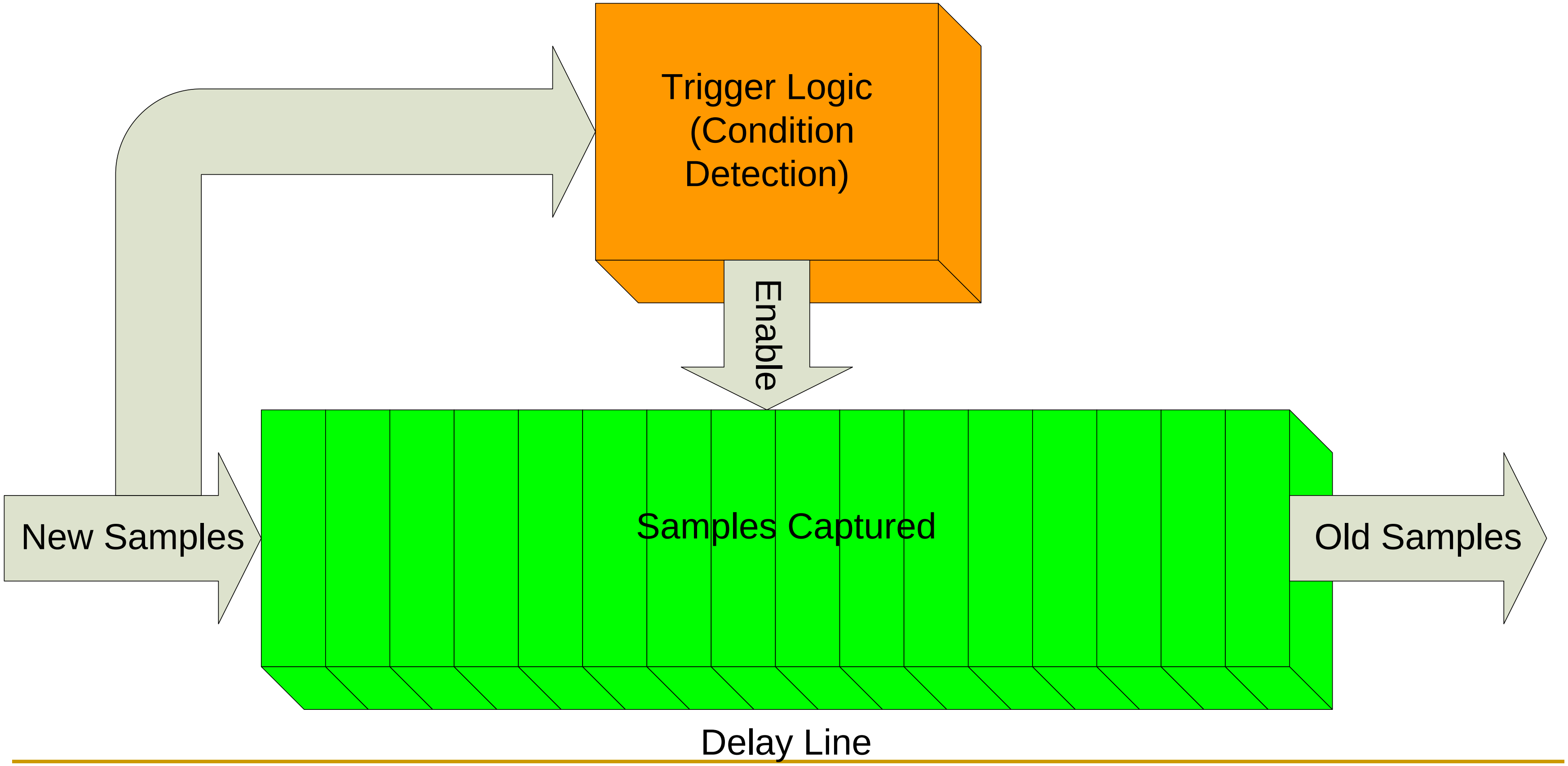
Agenda

Design Flow

- Debugging (Signal TAP)

Logic Analyzer

- Captures a time window of digital data
- Uses trigger to signal capture finish



Logic Analyzer Parts

- Sample Depth
  - Total number of samples in each capture (Size of the buffer)
- Sample Clock
  - Clock that samples the data into the buffer
- Trigger
  - A logic condition that stops the capture
- Trigger Options
  - Pre Trigger
  - Center Trigger
  - Post Trigger

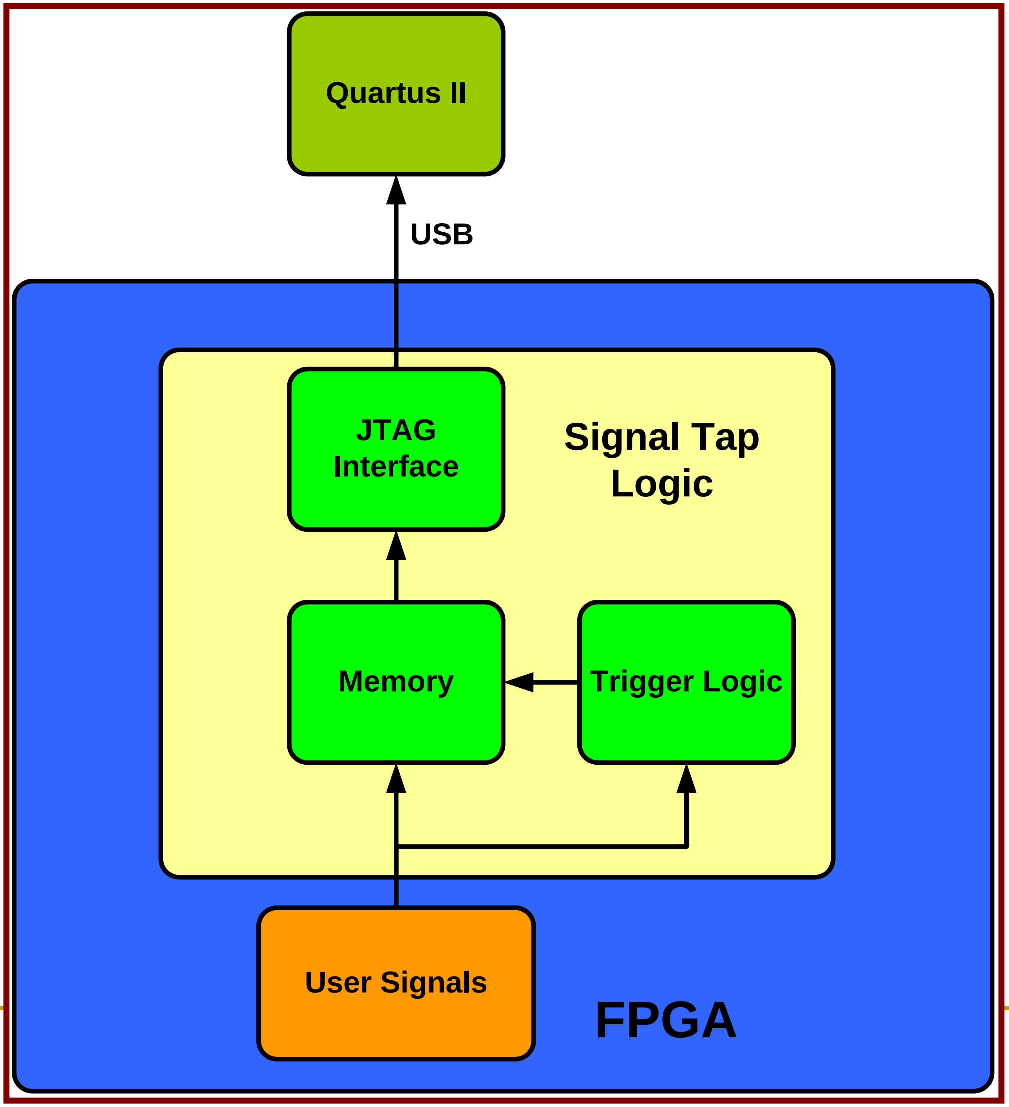
Signal TAP Pros & Cons

- Captures real time state of FPGA internal signals and pins (up to 200MHz)
- Connects to Quartus II through JTAG
- Do not require huge and expensive equipment
- Uses internal FPGA resources
  - Memory Blocks
  - Logic Elements
- Each time the captured signals list change the design must be recompiled

Signal TAP Features

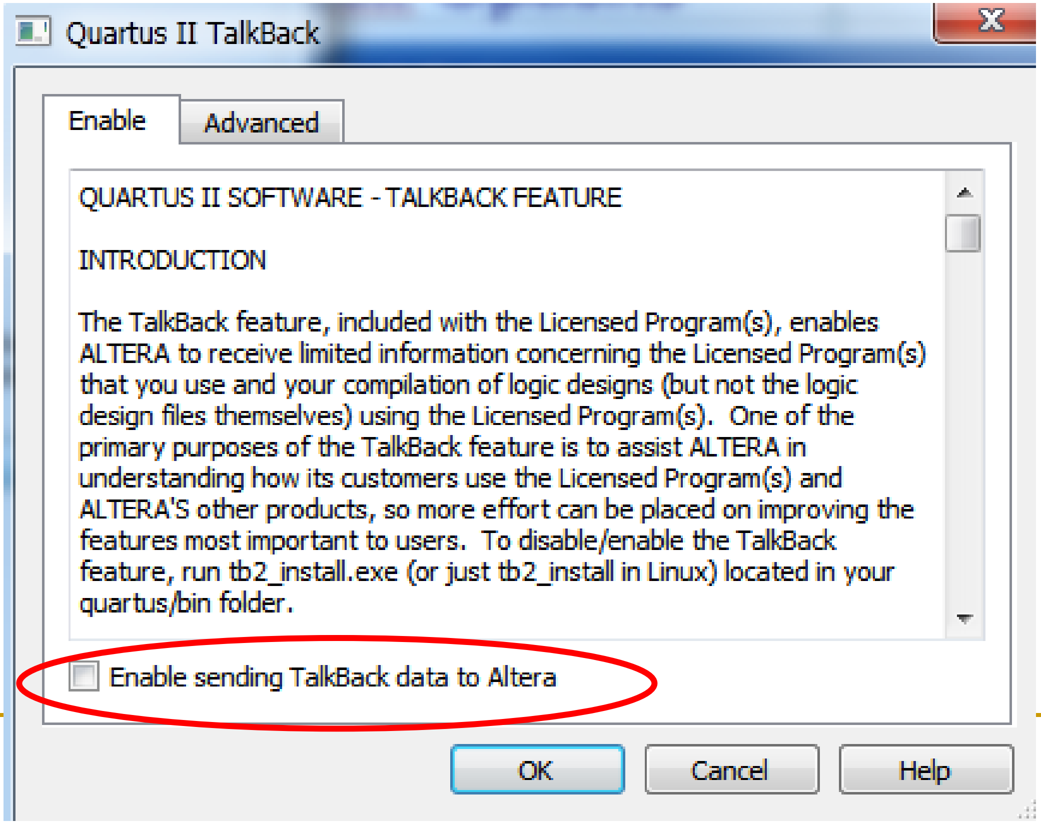
- Up to 1024 Data Channels
- Multiple Analyzers in One Device
  - Supports Analysis of Multiple Clock Domains
  - Each Analyzer Can Run Simultaneously
- Multiple Analyzers in One Device
- Up to 10 Trigger Levels Per Channel
- Up to 128K Samples Per Channel

Signal TAP - How it Works



Activating STP in Web Edition (no license)

- Go to : Tools→ Options -> Internet Connectivity -> TalkBack Options



QUARTUS II SOFTWARE - TALKBACK FEATURE

INTRODUCTION

The TalkBack feature, included with the Licensed Program(s), enables ALTERA to receive limited information concerning the Licensed Program(s) that you use and your compilation of logic designs (but not the logic design files themselves) using the Licensed Program(s). One of the primary purposes of the TalkBack feature is to assist ALTERA in understanding how its customers use the Licensed Program(s) and ALTERA'S other products, so more effort can be placed on improving the features most important to users. To disable/enable the TalkBack feature, run tb2_install.exe (or just tb2_install in Linux) located in your quartus/bin folder.

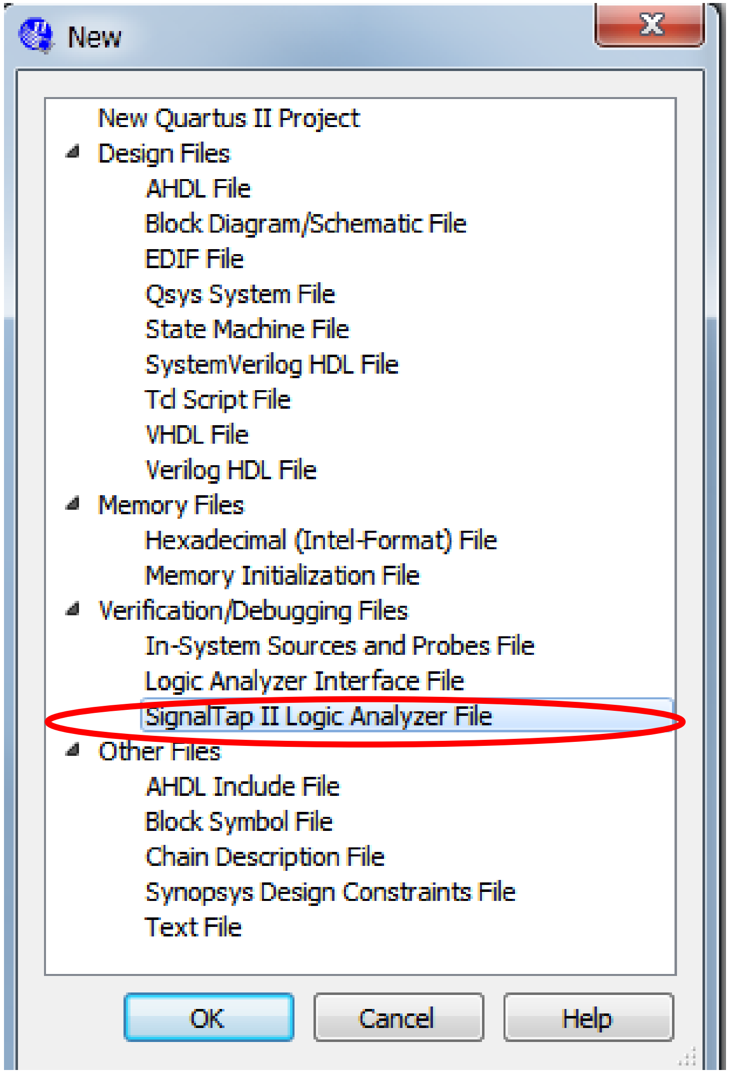
☐ Enable sending TalkBack data to Altera

Signal TAP Usage

- Create .STP File
  - Assign Sample Clock
  - Specify Sample Depth
  - Assign Signals to STP File
  - Specify Triggering
  - Setup JTAG
- Save .STP File & Compile with Design
- Program Device
- Acquire Data

Create new STP File

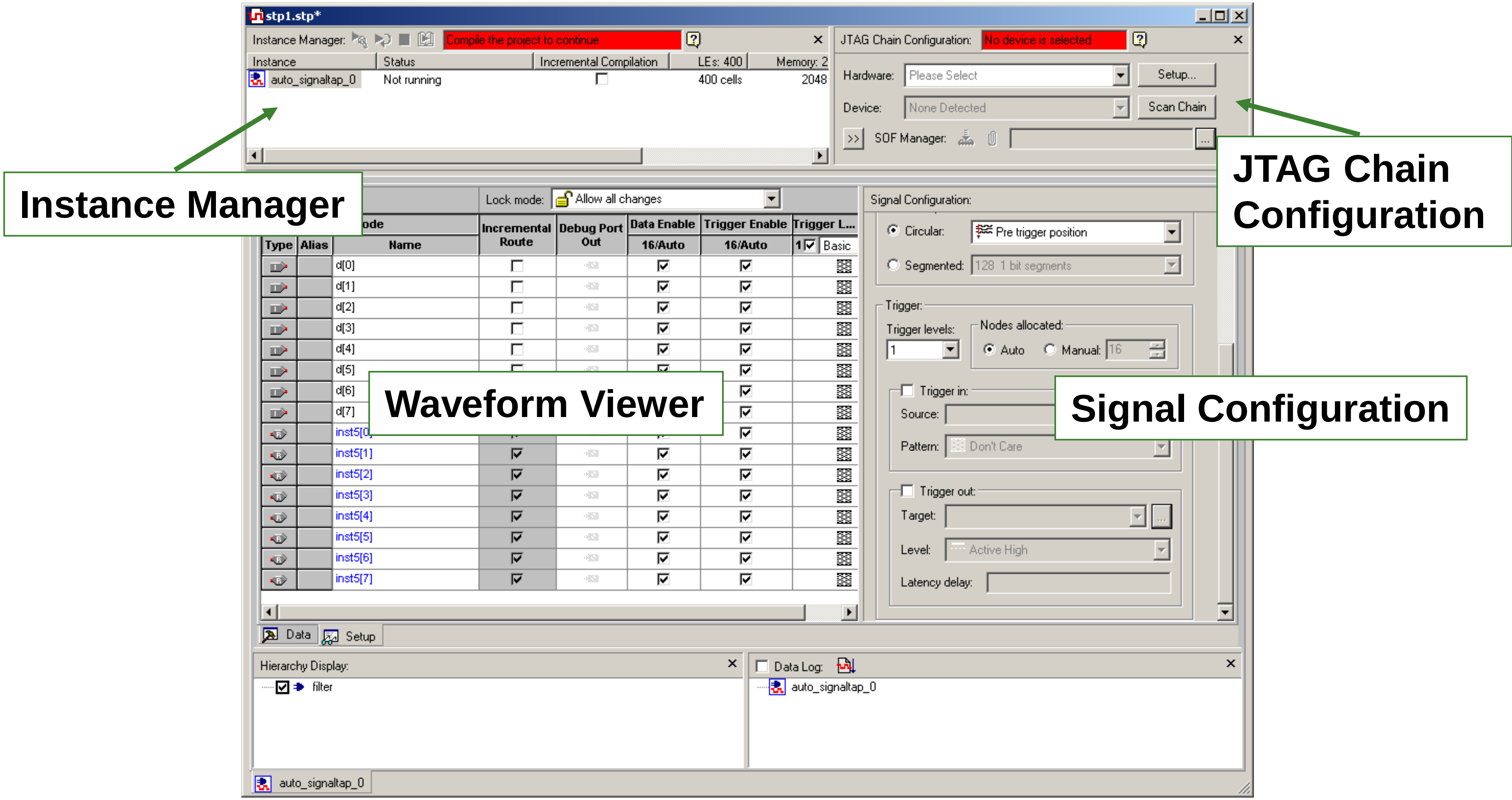
- File → New → Verification/Debugging Files → SignalTap II Logic Analyzer File



New

New Quartus II Project
- Design Files
  - AHDL File
  - Block Diagram/Schematic File
  - EDIF File
  - Qsys System File
  - State Machine File
  - SystemVerilog HDL File
  - Tcl Script File
  - VHDL File
  - Verilog HDL File
- Memory Files
  - Hexadecimal (Intel-Format) File
  - Memory Initialization File
- Verification/Debugging Files
  - In-System Sources and Probes File
  - Logic Analyzer Interface File
  - SignalTap II Logic Analyzer File
- Other Files
  - AHDL Include File
  - Block Symbol File
  - Chain Description File
  - Synopsys Design Constraints File
  - Text File

STP Components



Instance Manager

JTAG Chain Configuration

Waveform Viewer

Signal Configuration

stp1.stp*

Instance Manager: Compile the project to continue

Instance	Status	Incremental Compilation	LEs: 400	Memory: 2
auto_signaltap_0	Not running	□	400 cells	2048

JTAG Chain Configuration: No device is selected

Hardware: Please Select

Device: None Detected

SOF Manager:

Lock mode: Allow all changes

Type	Alias	Name	Incremental Route	Debug Port Out	Data Enable 16/Auto	Trigger Enable 16/Auto	Trigger L... 1 ☑ Basic
		d[0]	□		☑	☑	▨
		d[1]	□		☑	☑	▨
		d[2]	□		☑	☑	▨
		d[3]	□		☑	☑	▨
		d[4]	□		☑	☑	▨
		d[5]	□		☑	☑	▨
		d[6]				☑	▨
		d[7]				☑	▨
		inst5[0]				☑	▨
		inst5[1]	☑		☑	☑	▨
		inst5[2]	☑		☑	☑	▨
		inst5[3]	☑		☑	☑	▨
		inst5[4]	☑		☑	☑	▨
		inst5[5]	☑		☑	☑	▨
		inst5[6]	☑		☑	☑	▨
		inst5[7]	☑		☑	☑	▨

Signal Configuration:

● Circular: Pre trigger position

○ Segmented: 128 1 bit segments

Trigger:

Trigger levels: 1

Nodes allocated: ● Auto ○ Manual: 16

□ Trigger in:

Source:

Pattern: Don't Care

□ Trigger out:

Target:

Level: Active High

Latency delay:

Data Setup

Hierarchy Display:

☑ filter

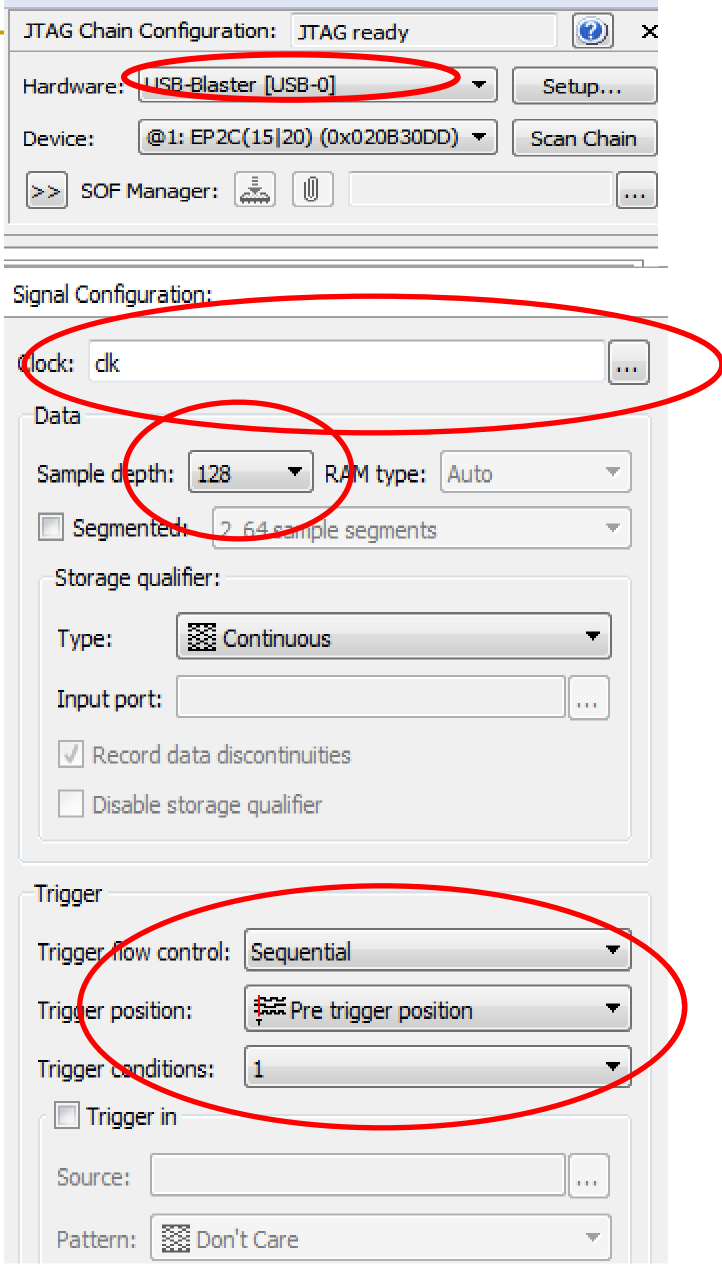
□ Data Log:

auto_signaltap_0

auto_signaltap_0

STP Setup

- Select USB-Blaster
  - In JTAG Configuration
- Set Sample Clock
  - Use Global Clock
  - Every Sample taken at Clock Rising Edge
  - Cannot Be Monitored as Data
- Specify Sample Depth
- Set Trigger mode
  - Sequential
  - Specify Trigger Position
    - Pre , Center , Post
  - Select Number of trigger conditions



JTAG Chain Configuration: JTAG ready

Hardware: USB-Blaster [USB-0]

Device: @1: EP2C(15|20) (0x020B30DD)

SOF Manager:

Signal Configuration:

Clock: clk

Data

Sample depth: 128

RAM type: Auto

□ Segmented: 2 64 sample segments

Storage qualifier:

Type: Continuous

Input port:

☑ Record data discontinuities

□ Disable storage qualifier

Trigger

Trigger flow control: Sequential

Trigger position: Pre trigger position

Trigger conditions: 1

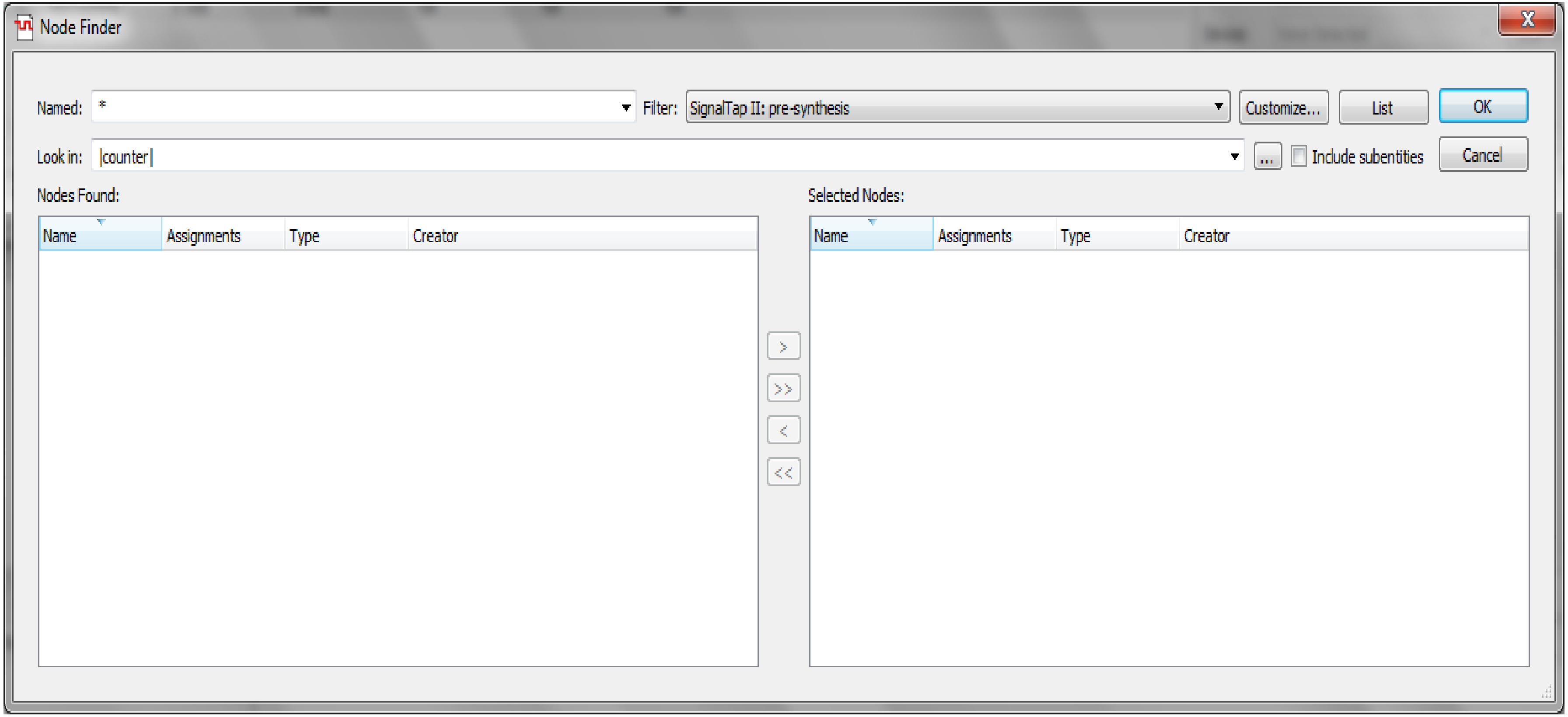
□ Trigger in

Source:

Pattern: Don't Care

Adding Signals

- To open Node Finder
- In signals setup right click and select add nodes or double click.
- Important : Select SignalTap II: Pre synthesis in Filters



Named: *

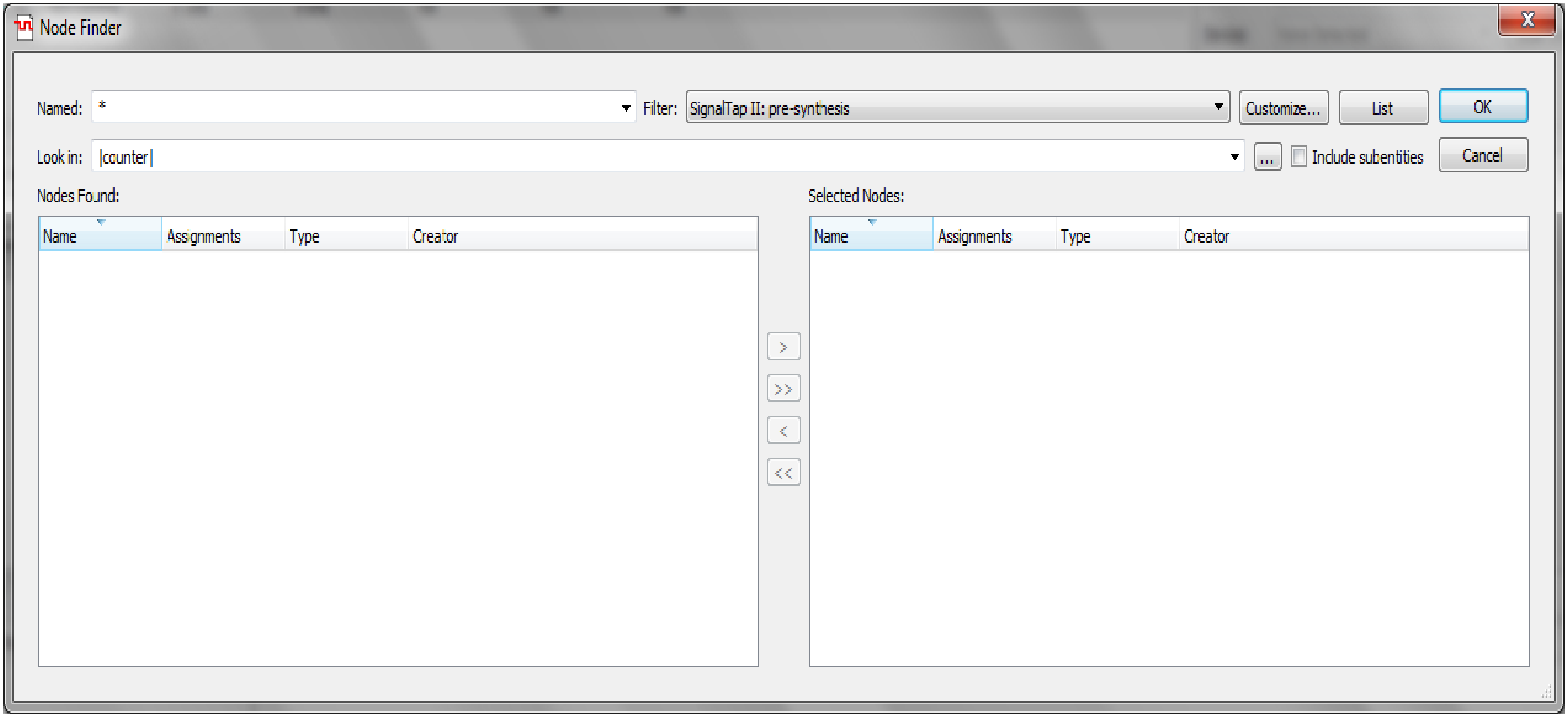
Filter: SignalTap II: pre-synthesis

Look in: |counter|

□ Include subentities

Adding Signals

- Specify entity for search “Look in”
- Click List , double click signals in left window to add
- Click OK



Node Finder

Named: *

Filter: SignalTap II: pre-synthesis

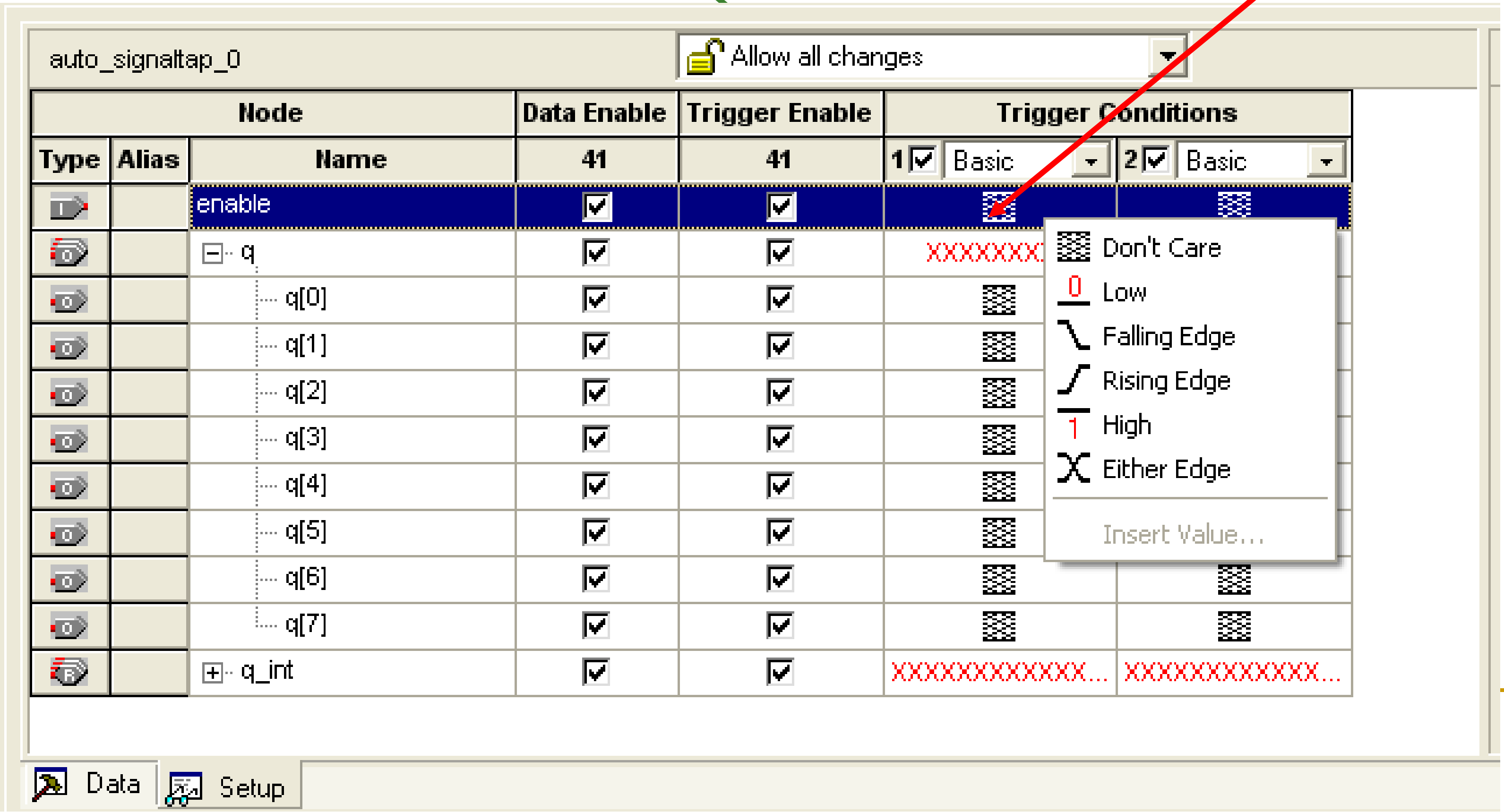
Look in: |counter|

☐ Include subentities

Setting Triggers

- All signals must satisfy trigger condition to cause data capture

Right-Click to Set Value



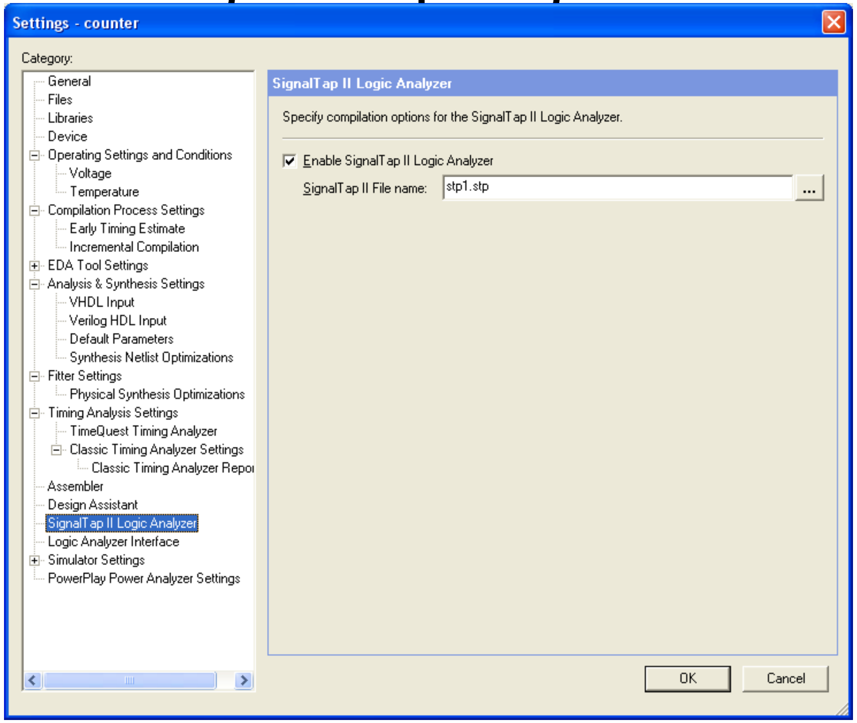
auto_signaltap_0

Allow all changes

Node			Data Enable	Trigger Enable	Trigger Conditions	
Type	Alias	Name	41	41	1 ☑ Basic	2 ☑ Basic
		enable	☑	☑	▨	▨
		⊟ q	☑	☑	XXXXXXXX	
		q[0]	☑	☑	▨	
		q[1]	☑	☑	▨	
		q[2]	☑	☑	▨	
		q[3]	☑	☑	▨	
		q[4]	☑	☑	▨	
		q[5]	☑	☑	▨	
		q[6]	☑	☑	▨	▨
		q[7]	☑	☑	▨	▨
		⊞ q_int	☑	☑	XXXXXXXXXXXX...	XXXXXXXXXXXX...

STP Compilation

- Save The STP file
- Open Assignments -> Settings -> SignalTap logic analyzer
  - Specify the STP File to Compile with Project


Settings - counter
Category:
General
Files
Libraries
Device
Operating Settings and Conditions
Voltage
Temperature
Compilation Process Settings
Early Timing Estimate
Incremental Compilation
EDA Tool Settings
Analysis & Synthesis Settings
VHDL Input
Verilog HDL Input
Default Parameters
Synthesis Netlist Optimizations
Fitter Settings
Physical Synthesis Optimizations
Timing Analysis Settings
TimeQuest Timing Analyzer
Classic Timing Analyzer Settings
Classic Timing Analyzer Repor
Assembler
Design Assistant
SignalTap II Logic Analyzer
Logic Analyzer Interface
Simulator Settings
PowerPlay Power Analyzer Settings
OK
Cancel

SignalTap II Logic Analyzer

Specify compilation options for the SignalTap II Logic Analyzer.

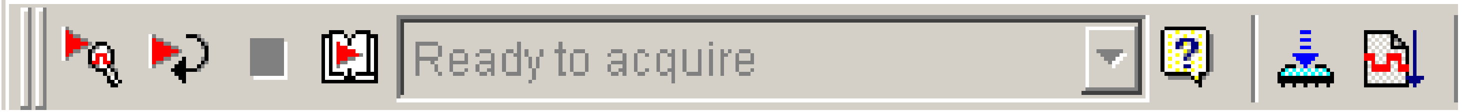
☑ Enable SignalTap II Logic Analyzer

SignalTap II File name: stp1.stp

- Run Full Project Compilation and reprogram FPGA

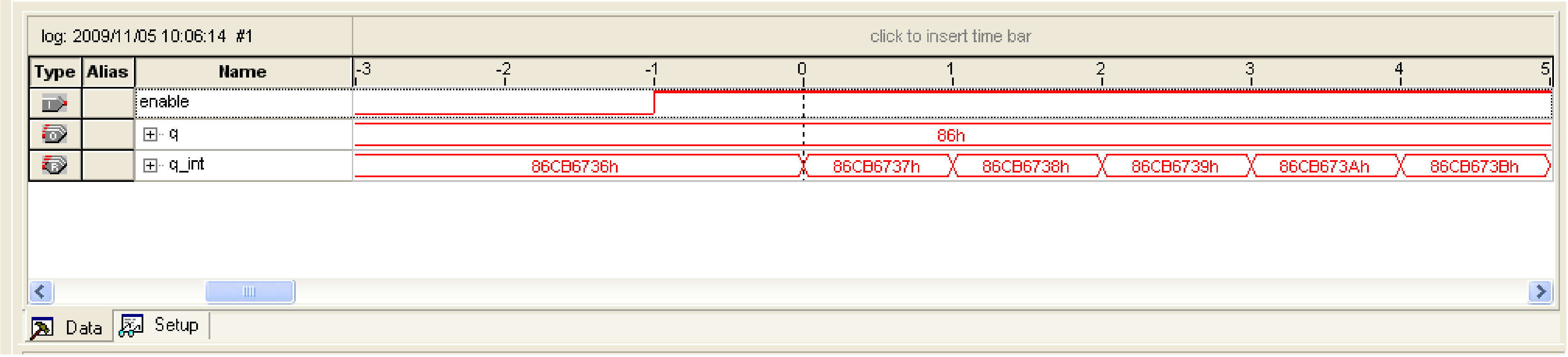
Acquiring Data

- Signal Tap II Toolbar & STP File Controls



Ready to acquire

  - Run
  - Autorun
  - Stop



log: 2009/11/05 10:06:14 #1

click to insert time bar

Type	Alias	Name	-3	-2	-1	0	1	2	3	4	5
		enable									
		⊞ q	86h								
		⊞ q_int	86CB6736h			86CB6737h	86CB6738h	86CB6739h	86CB673Ah	86CB673Bh	

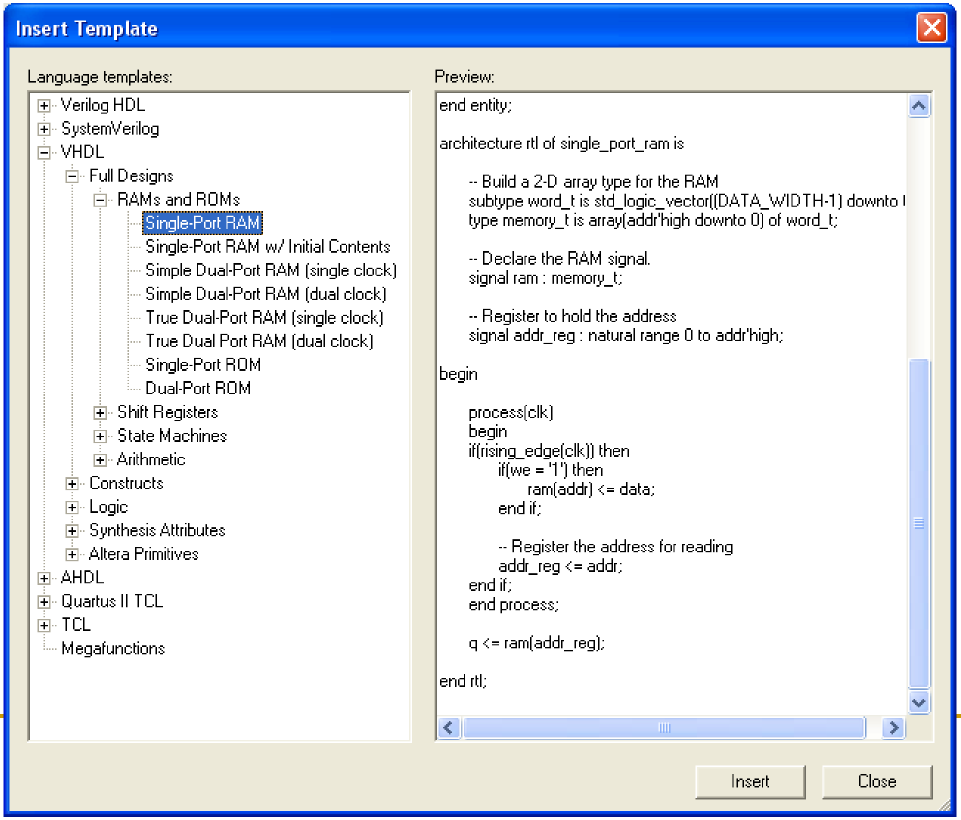
Agenda

Design Flow

- PLLs and Templates

Hardware Templates

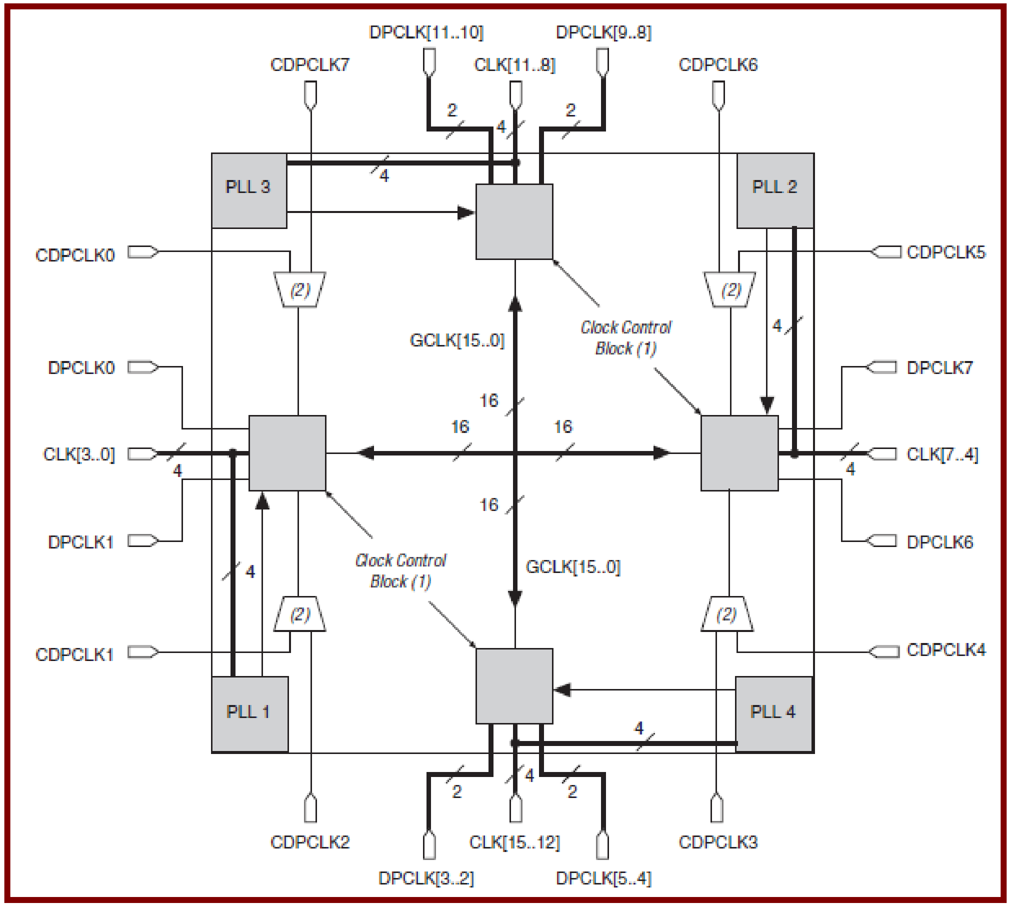
- While in VHDL file push button on the left
- In a template window select the needed logic template



```
end entity;

architecture rtl of single_port_ram is

	-- Build a 2-D array type for the RAM
	subtype word_t is std_logic_vector((DATA_WIDTH-1) downto 
	type memory_t is array(addr'high downto 0) of word_t;

	-- Declare the RAM signal.
	signal ram : memory_t;

	-- Register to hold the address
	signal addr_reg : natural range 0 to addr'high;

begin

	process(clk)
	begin
	if(rising_edge(clk)) then
		if(we = '1') then
			ram(addr) <= data;
		end if;

		-- Register the address for reading
		addr_reg <= addr;
	end if;
	end process;

	q <= ram(addr_reg);

end rtl;
```

Cyclone II Clocking

- 16 Global Clocks
- 4 PLLs
- External Oscillator Must be used



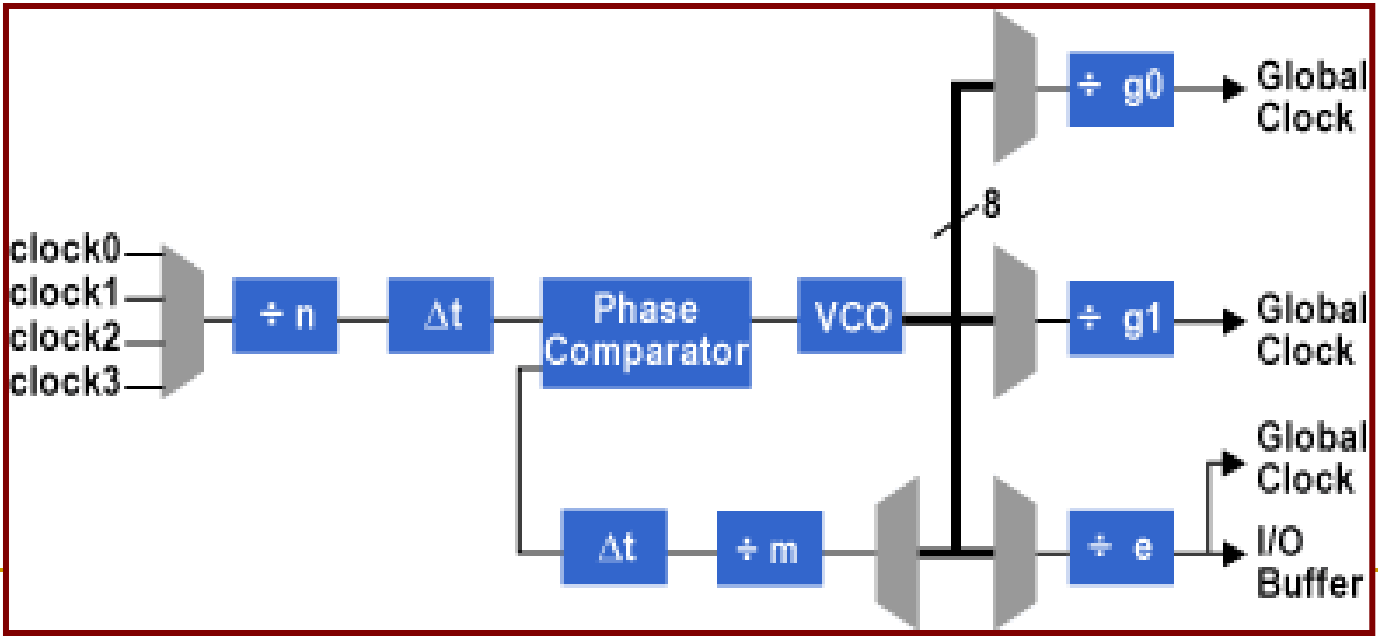
Cyclone II PLL

- Enables division and multiplication of clocks
- 3 Outputs with different frequencies
- Input clock can be only from clock pin
- Different Phase shifts between outputs can be achieved

Cyclone II PLL Structure

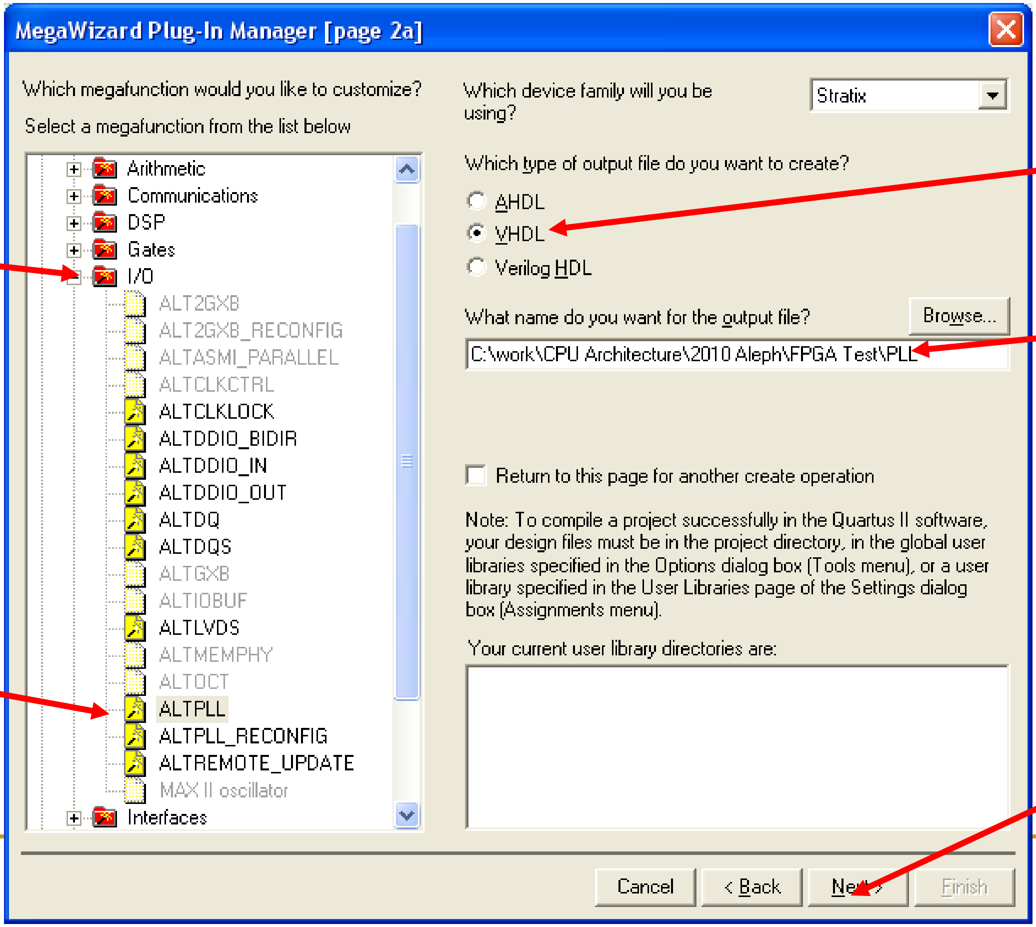
- Output frequencies are related each to other

$$\boxed{Fout(i) = Fin\frac{m}{n \cdot g(i)}}$$



PLL

- Open MegaWisard Tools->MegaWisard Plug in manager and click Next



MegaWizard Plug-In Manager [page 2a]

Which megafunction would you like to customize?

Select a megafunction from the list below

- Arithmetic
- Communications
- DSP
- Gates
- I/O
  - ALT2GXB
  - ALT2GXB_RECONFIG
  - ALTASMI_PARALLEL
  - ALTCLKCTRL
  - ALTCLKLOCK
  - ALTDDIO_BIDIR
  - ALTDDIO_IN
  - ALTDDIO_OUT
  - ALTDQ
  - ALTDQS
  - ALTGXB
  - ALTIOBUF
  - ALTLVDS
  - ALTMEMPHY
  - ALTOCT
  - ALTPLL
  - ALTPLL_RECONFIG
  - ALTREMOTE_UPDATE
  - MAX II oscillator
- Interfaces

Which device family will you be using? Stratix

Which type of output file do you want to create?

○ AHDL

● VHDL

○ Verilog HDL

What name do you want for the output file? C:\work\CPU Architecture\2010 Aleph\FPGA Test\PLL

□ Return to this page for another create operation

Note: To compile a project successfully in the Quartus II software, your design files must be in the project directory, in the global user libraries specified in the Options dialog box (Tools menu), or a user library specified in the User Libraries page of the Settings dialog box (Assignments menu).

Your current user library directories are:

Open I/O

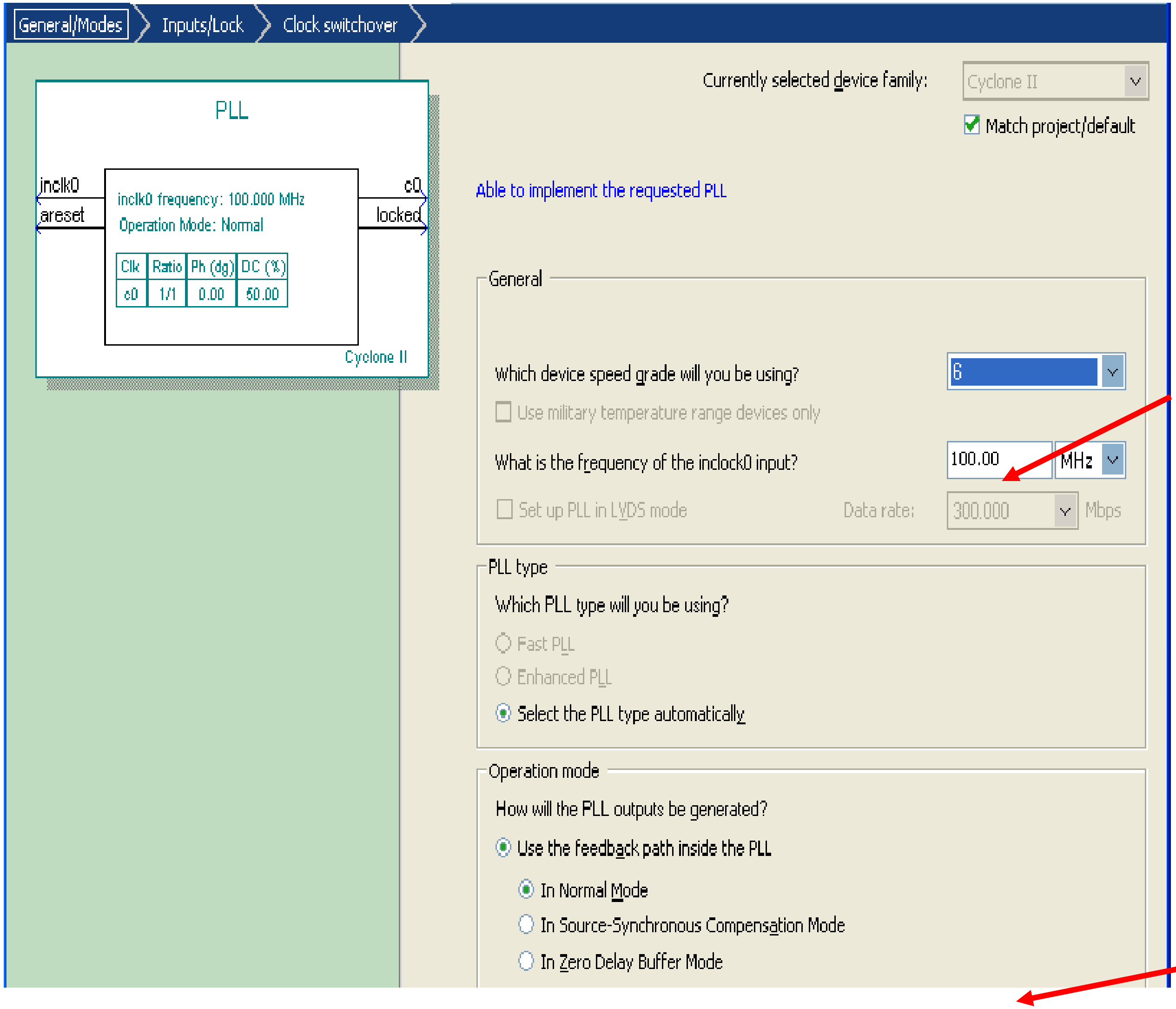
Select ALTPLL

Select VHDL

Set Name

Click Next

PLL Cont



inclk0 frequency: 100.000 MHz

Operation Mode: Normal

Clk	Ratio	Ph (dg)	DC (%)
c0	1/1	0.00	50.00

Currently selected device family: Cyclone II

☑ Match project/default

Able to implement the requested PLL

General

Which device speed grade will you be using? 6

□ Use military temperature range devices only

What is the frequency of the inclock0 input? 100.00 MHz

□ Set up PLL in LVDS mode Data rate: 300.000 Mbps

PLL type

Which PLL type will you be using?

○ Fast PLL

○ Enhanced PLL

● Select the PLL type automatically

Operation mode

How will the PLL outputs be generated?

● Use the feedback path inside the PLL

● In Normal Mode

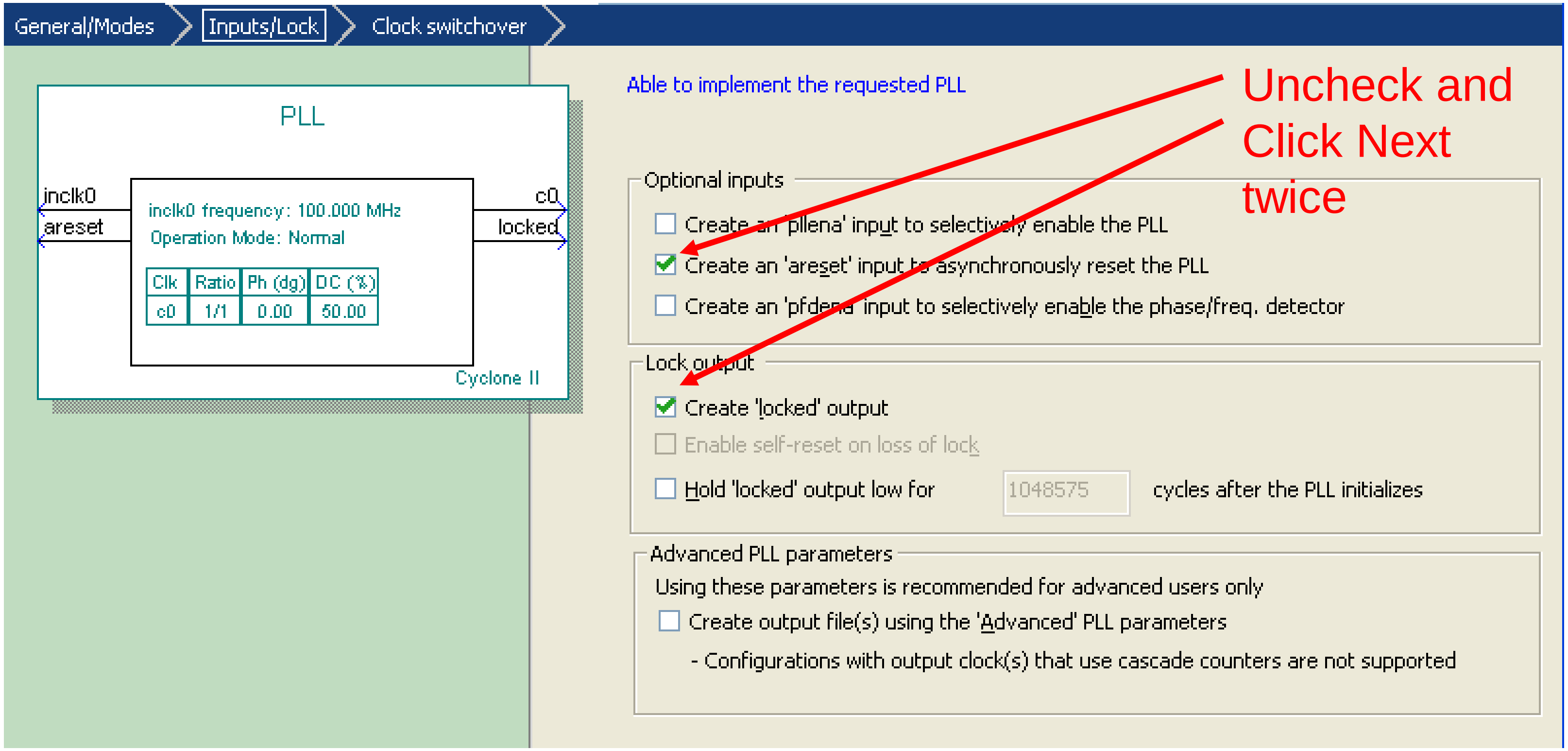
○ In Source-Synchronous Compensation Mode

○ In Zero Delay Buffer Mode

1. Set Input Clock

2. Click Next

PLL Cont.



inclk0 frequency: 100.000 MHz
Operation Mode: Normal

Clk	Ratio	Ph (dg)	DC (%)
c0	1/1	0.00	50.00

Able to implement the requested PLL

Optional inputs

☐ Create an 'pllena' input to selectively enable the PLL

☑ Create an 'areset' input to asynchronously reset the PLL

☐ Create an 'pfdena' input to selectively enable the phase/freq. detector

Lock output

☑ Create 'locked' output

☐ Enable self-reset on loss of lock

☐ Hold 'locked' output low for 1048575 cycles after the PLL initializes

Advanced PLL parameters

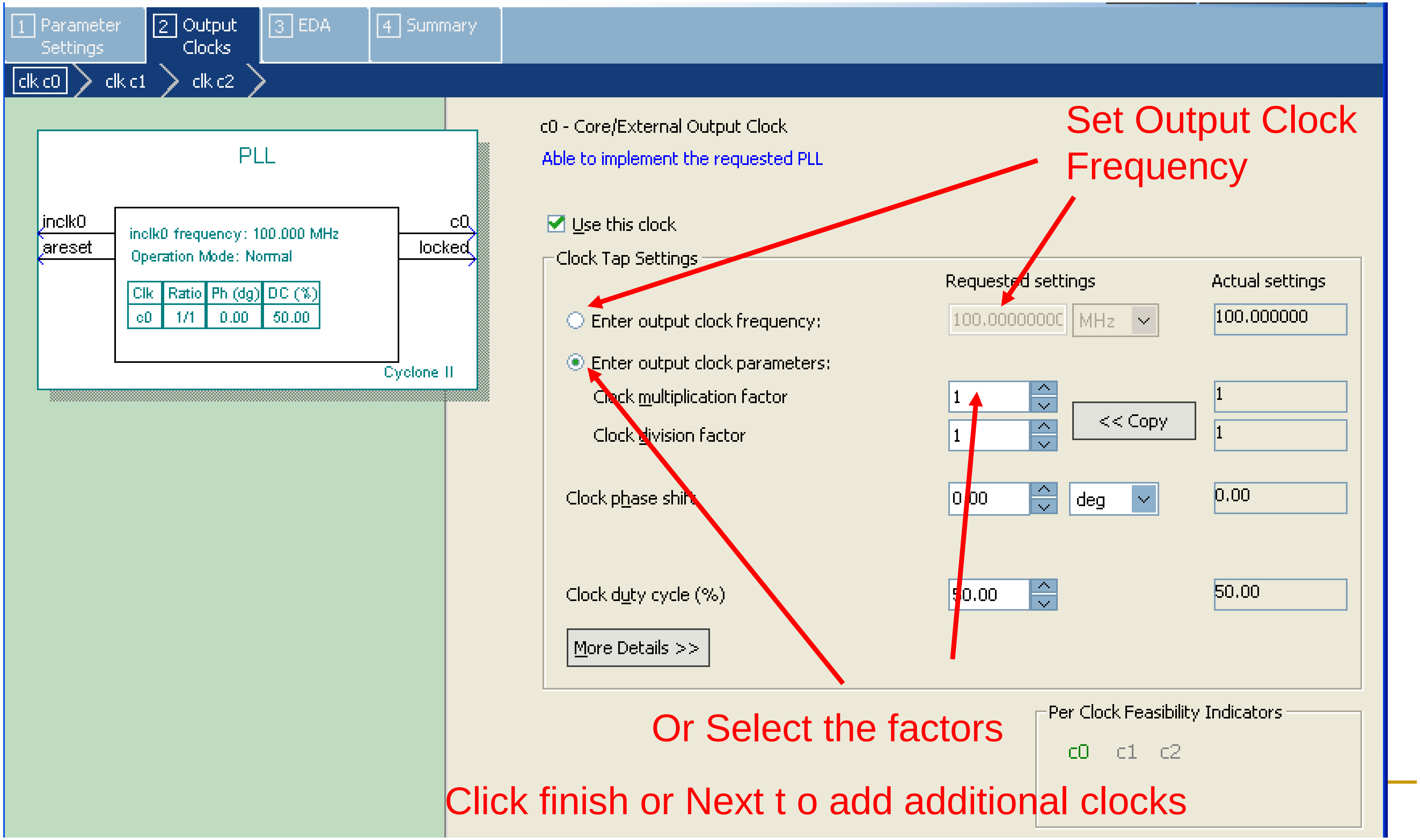
Using these parameters is recommended for advanced users only

☐ Create output file(s) using the 'Advanced' PLL parameters

- Configurations with output clock(s) that use cascade counters are not supported

Uncheck and Click Next twice

PLL Cont



inclk0 frequency: 100.000 MHz

Operation Mode: Normal

Clk	Ratio	Ph (dg)	DC (%)
c0	1/1	0.00	50.00

c0 - Core/External Output Clock

Able to implement the requested PLL

☑ Use this clock

Clock Tap Settings

	Requested settings		Actual settings
○ Enter output clock frequency:	100.0000000C	MHz	100.000000
● Enter output clock parameters:			
Clock multiplication factor	1		1
Clock division factor	1		1
Clock phase shift	0.00	deg	0.00
Clock duty cycle (%)	50.00		50.00

Set Output Clock Frequency

Or Select the factors

Click finish or Next t o add additional clocks

References

- Quartus user manual
- DE1 board data sheet
- Altera web site.
- Cyclone II Data sheet

Any questions?