

Introduction to FPGA Design using Quartus (Remote Console)

Intel Programmable Solutions Group

Larry Landis

Senior Manager University Outreach



- Larry Landis – University Outreach Senior Manager, Intel Programmable Solutions Group
- 32 years in semiconductor industry – design, sales, marketing, training (Altera 2004)
- Adjunct Lecturer for Digital Electronics Course – Santa Clara University



BSEE



MSEE



Adjunct Lecturer



Topics

- FPGAs at Intel
- Fundamentals of Digital Electronics
- FPGA Architecture
- Intel® Quartus® Prime Design Software
- FPGA Design Flow

Intel® FPGA Academic Ecosystem



- Train the next generation of FPGA Designers
- Increase Intel® FPGA presence in academia



- Academic access to the latest generation of Intel FPGAs



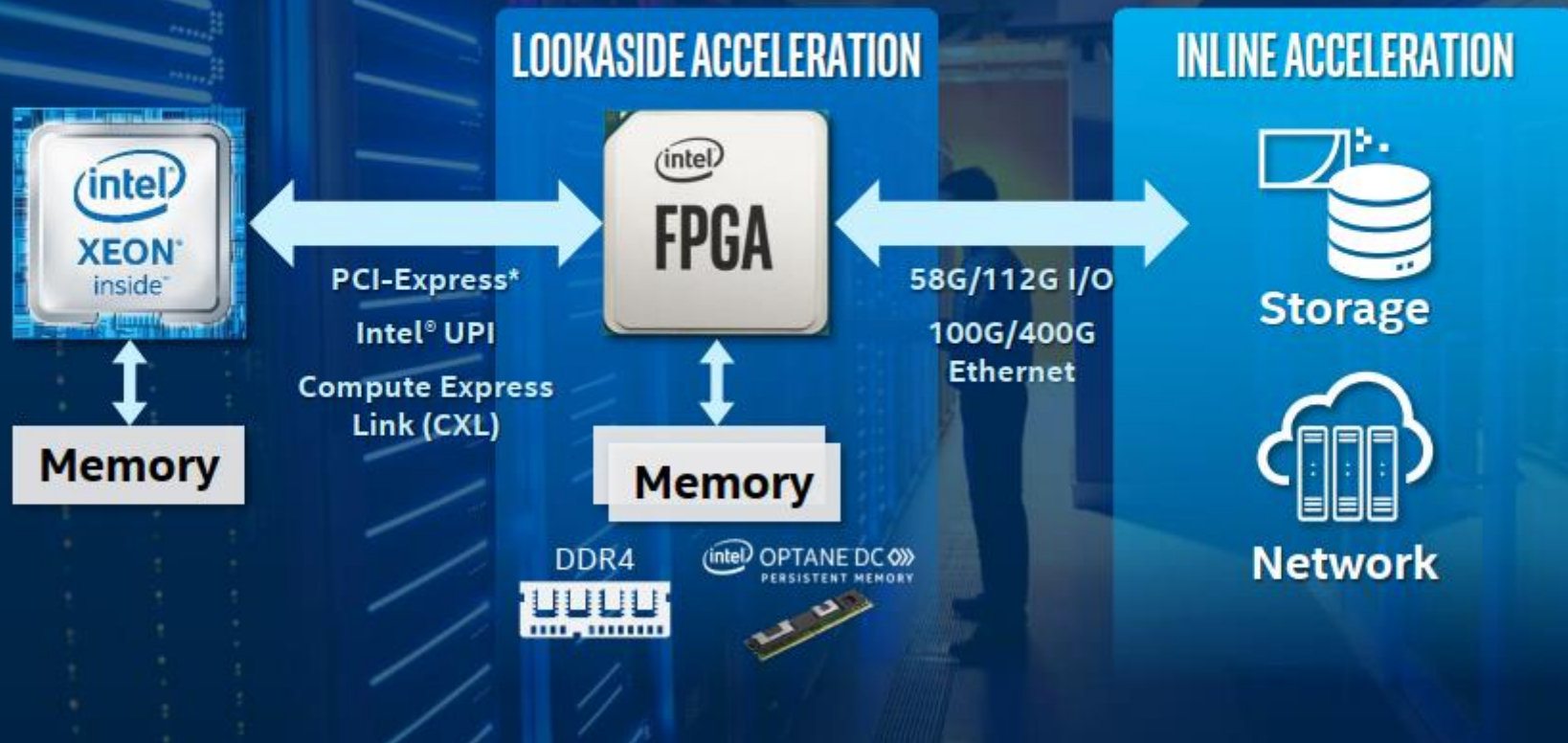
- Nurture the talent pipeline for Intel and our customers
- Engage research on Intel FPGAs

INTEL® FPGA EE COURSEWORK OFFERINGS

- Undergraduate
 - Digital Logic
 - Digital Systems
 - Computer Organization
 - Embedded Systems



INTEL® FPGAS ACCELERATING THE CLOUD & ENTERPRISE



Website Access for the Intel® FPGA Devcloud

<https://software.intel.com/en-us/devcloud/FPGA>



The screenshot shows the Intel FPGA Devcloud website. At the top, there is a blue header with a hamburger menu icon and the text "Development Tools" on the left, and the Intel logo on the right. Below the header is a large blue banner with the title "FPGA Design Development and Workloads for Hardware Acceleration". Under the title, there is a paragraph: "Develop programmable solutions and validate your workloads on leading FPGA hardware with tools optimized for Intel® technology. Use this cloud solution in the classroom to support acceleration engineering curriculum." Below this paragraph is a white button with the text "Sign Up →". Below the banner is a white section with the text "Overview | Intel® DevCloud for oneAPI | Edge" centered. Below this is the heading "What You Can Do" centered. At the bottom, there are three icons with corresponding text: 1. An icon of a circuit board with a gear and a hand, with the text "Develop custom hardware IP and FPGA designs". 2. An icon of a document with a gear, with the text "Teach advanced FPGA-based university courses". 3. An icon of a bar chart with a checkmark, with the text "Benchmark hardware acceleration performance".

Development Tools

intel

FPGA Design Development and Workloads for Hardware Acceleration

Develop programmable solutions and validate your workloads on leading FPGA hardware with tools optimized for Intel® technology. Use this cloud solution in the classroom to support acceleration engineering curriculum.

Sign Up →

Overview | Intel® DevCloud for oneAPI | Edge

What You Can Do

-  Develop custom hardware IP and FPGA designs
-  Teach advanced FPGA-based university courses
-  Benchmark hardware acceleration performance

Field Programmable Gate Array (FPGA)



- Flexible, multi-functional reprogrammable silicon
- Custom hardware functionality
- Bare-metal speed and reliability
- Truly parallel in nature

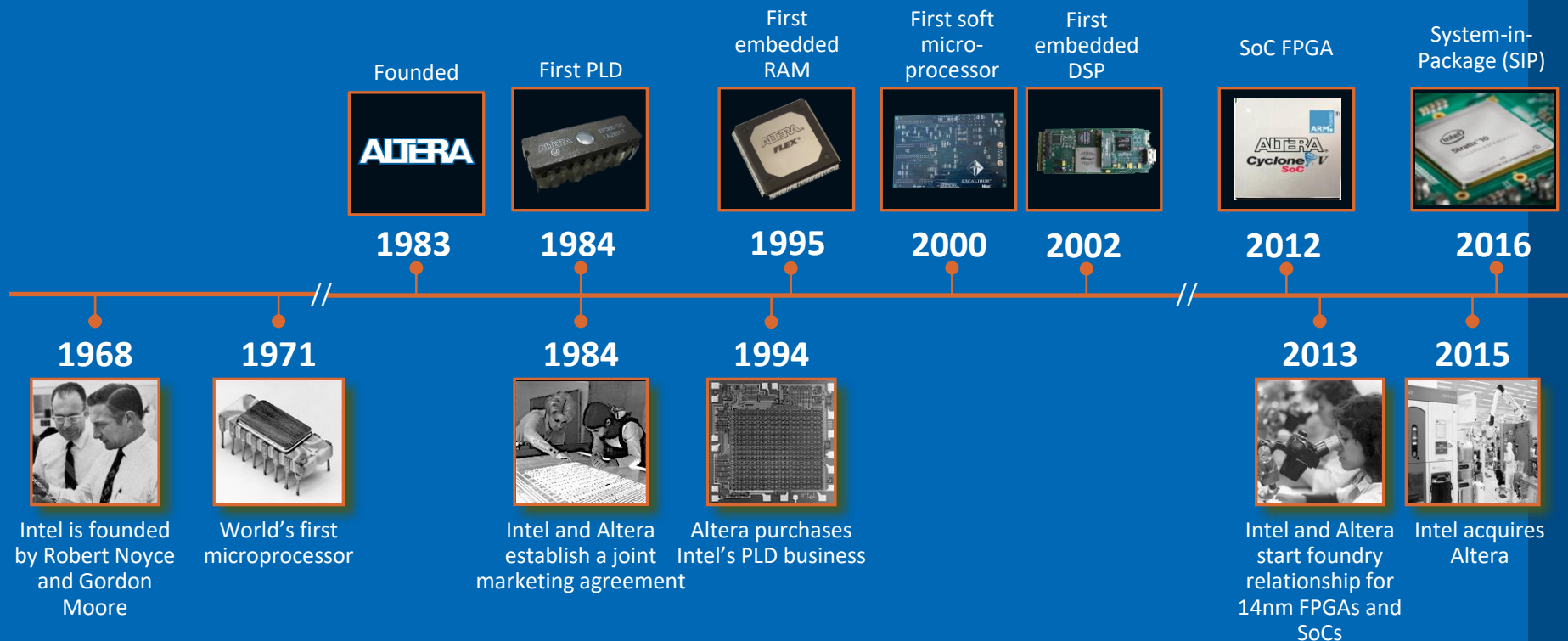
Benefits of FPGA Technology

- Flexibility
- Time to market
- Performance
- Reliability
- Long-Term Maintenance – reprogram if features change or bugs found
- Many different applications – 5G, Data Center, Industrial, DSP
- Excellent prototyping vehicle

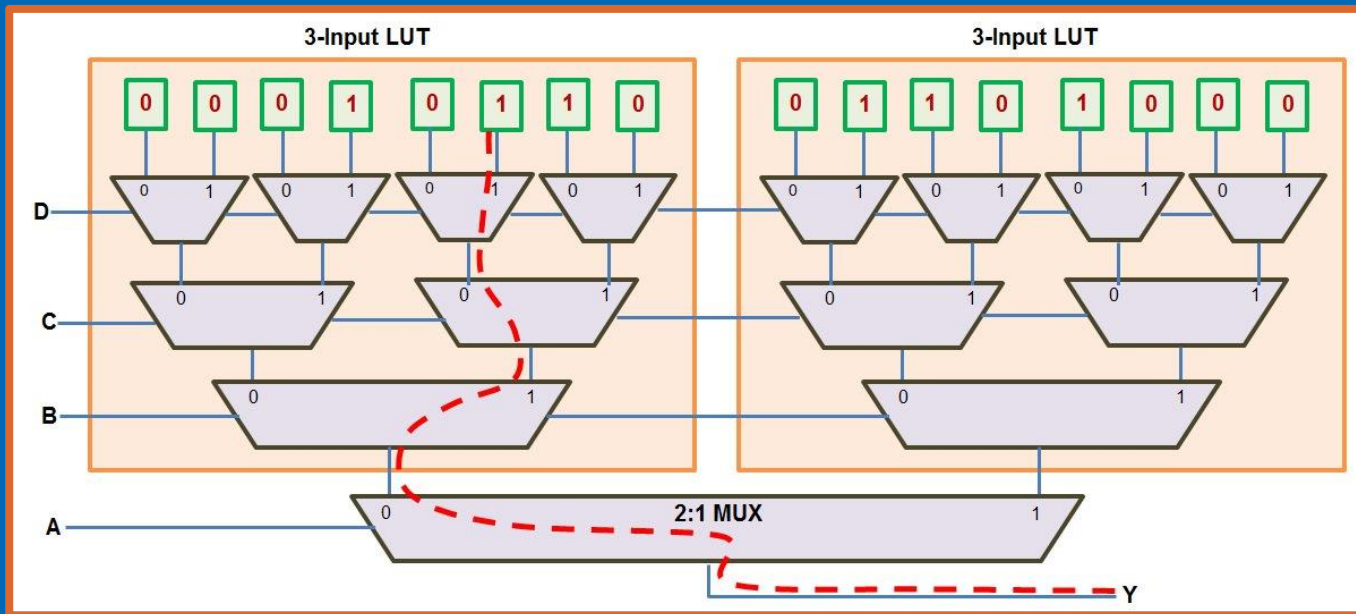
Rise of new markets



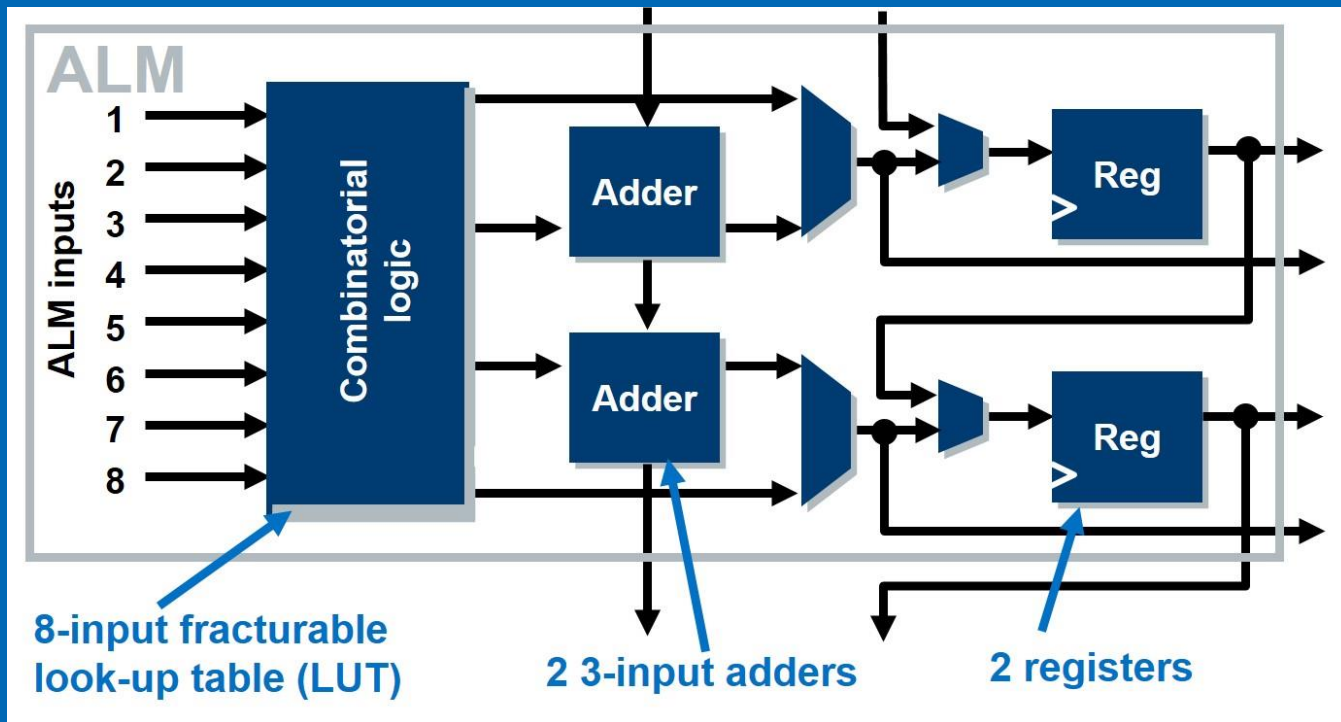
Intel's history with FPGAs



Look-Up Table (LUT): the foundation



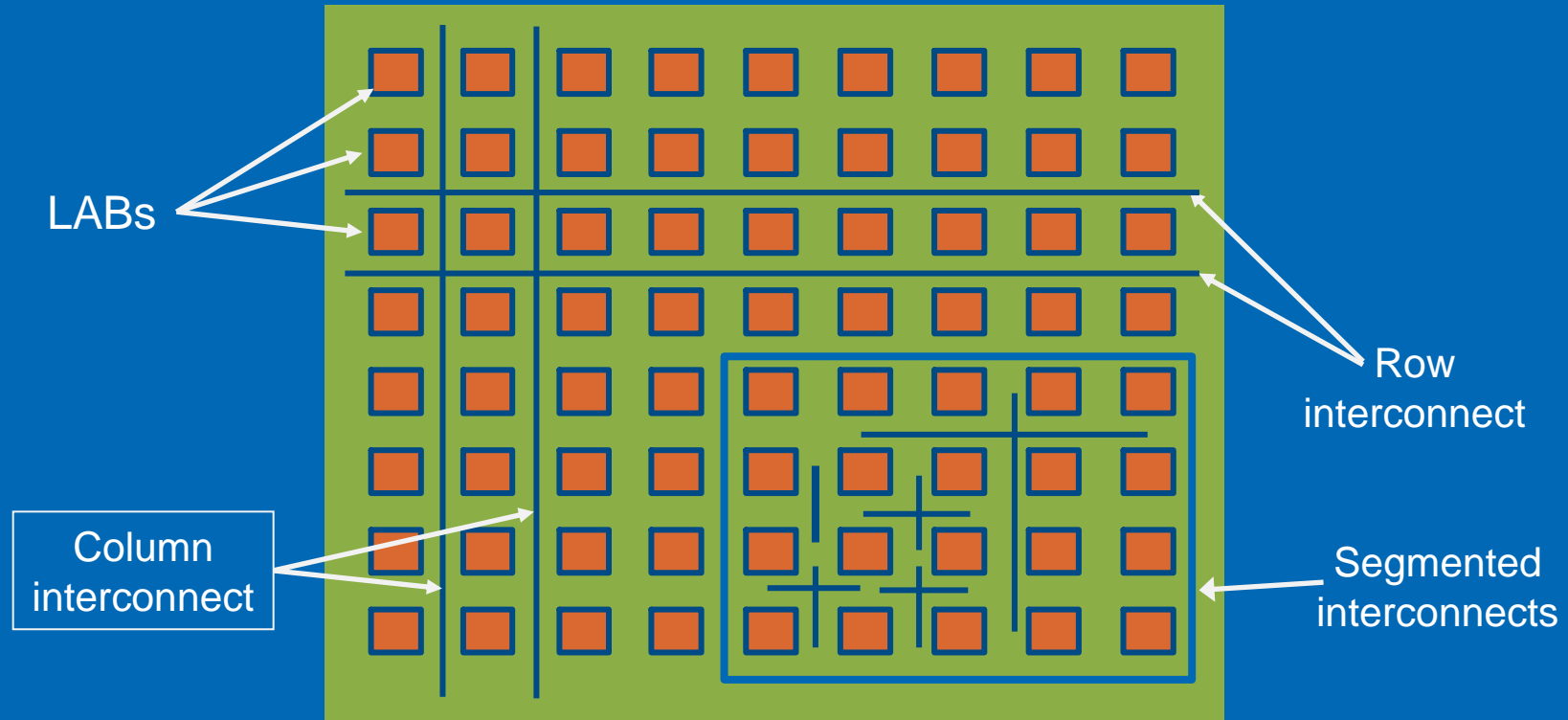
Logic Array Blocks



x 10

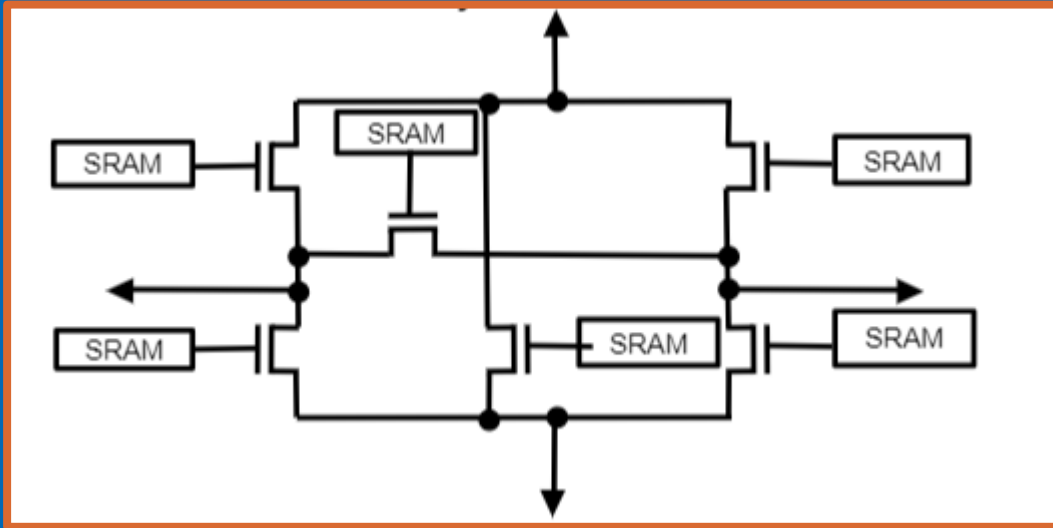
*Number of inputs and ALMs per LAB vary by product family

Building the Array



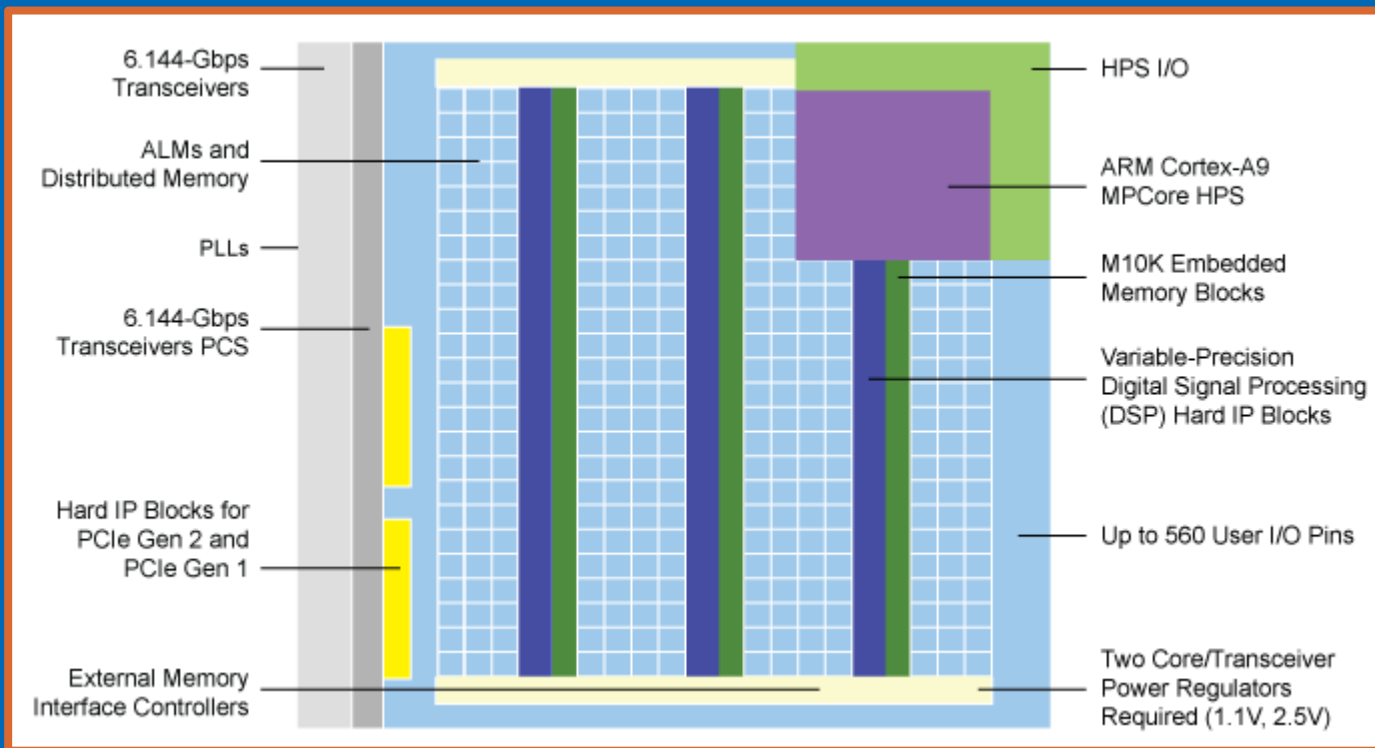
How switching fabric is programmed

Row/Column Interconnect Junction



Programming info stored in a
external non-volatile device
Active: programmed
automatically at power-on
Passive: Intelligent host
(CPU) controls
programming

FPGAs “Hardened” features

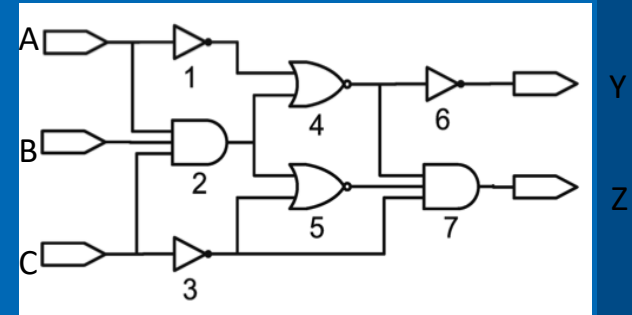


Describing FPGAs

- Schematics
- System Integration Tools – predefined blocks
- Hardware Description Languages (HDLs)
 - Verilog, VHDL are most popular
- High level languages
 - “HLS”
 - OpenCL
 - Data Parallel C++

Use	Connections	Name	Description	Export	Clock
<input checked="" type="checkbox"/>		CLK_IP	CLK Input	clk	exported
		clk_in	Clock Input		
		clk_in_reset	Reset Input	reset	
		clk	Clock Output	Double-click to export	CLK_IP
		clk_reset	Reset Output	Double-click to export	
<input checked="" type="checkbox"/>		JTAG_2_Avalon_IP	JTAG to Avalon Master Bridge		
		clk	Clock Input	Double-click to export	CLK_IP
		clk_reset	Reset Input	Double-click to export	
		master	Avalon Memory Mapped Master	Double-click to export	[clk]
		master_reset	Reset Output	Double-click to export	
<input checked="" type="checkbox"/>		LED_IP_0	PIC (Parallel I/O) Intel FPGA IP		
		clk	Clock Input	Double-click to export	CLK_IP
		reset	Reset Input	Double-click to export	[clk]
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]
		external_connection	Conduit	ledr31_to_0	
<input checked="" type="checkbox"/>		SW_IP	PIC (Parallel I/O) Intel FPGA IP		
		clk	Clock Input	Double-click to export	CLK_IP
		reset	Reset Input	Double-click to export	[clk]
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]
		external_connection	Conduit	control13_to_12_key11...	
<input checked="" type="checkbox"/>		LED_IP_1	PIC (Parallel I/O) Intel FPGA IP		
		clk	Clock Input	Double-click to export	CLK_IP
		reset	Reset Input	Double-click to export	[clk]
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]
		external_connection	Conduit	ledr63_to_32	

```
try {  
  
    sycl::queue q(sycl::default_selector{});  
  
    const float A(aval);  
  
    sycl::buffer<float,1> d_X { h_X.data(), sycl::range<1>(h_X.size()) };  
    sycl::buffer<float,1> d_Y { h_Y.data(), sycl::range<1>(h_Y.size()) };  
    sycl::buffer<float,1> d_Z { h_Z.data(), sycl::range<1>(h_Z.size()) };  
  
    q.submit([&](sycl::handler& h) {  
  
        auto X = d_X.template get_access<sycl::access::mode::read>(h);  
        auto Y = d_Y.template get_access<sycl::access::mode::read>(h);  
        auto Z = d_Z.template get_access<sycl::access::mode::read_write>(h);  
  
        h.parallel_for<class nstream>( sycl::range<1>(length), [=] (sycl::id<1> it) {  
            const int i = it[0];  
            Z[i] += A * X[i] + Y[i];  
        });  
    });  
    q.wait();  
}
```

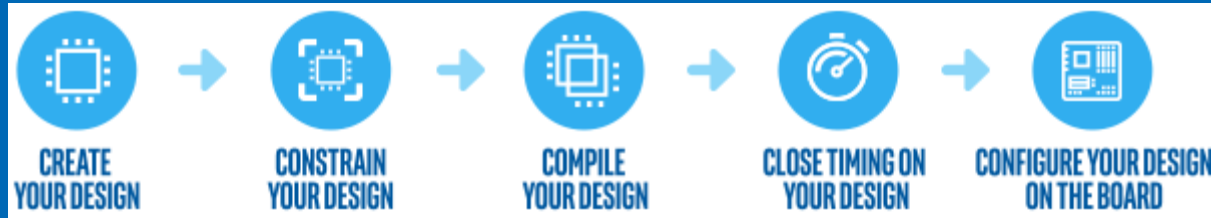


What is IP (Intellectual Property)

- Complex functions that Intel designs for our customers so they don't have to design it themselves
 - Sometimes IP is free
 - The more complex stuff costs since its expensive to develop and make sure it works
- Examples: Ethernet Controller, PCIe Controller, soft processor, multiplier functions, etc.

Basics of Quartus

- Intel® Quartus® Prime Design Software is a tool for FPGA, SOC and CPLD design
- Includes synthesis, debug, optimization, verification and simulation
- Takes a description of an FPGA (schematic or HDL) and determines how the lookup tables are programmed

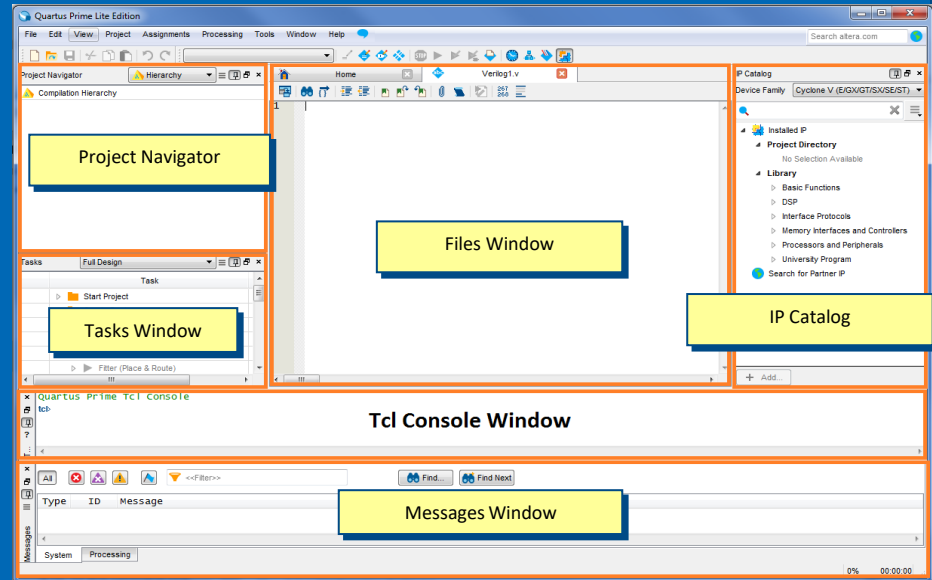


- Many formats to program an FPGA
 - In this class we will use a “.sof” file (SRAM object file)
 - The .sof file is “volatile” and needs to be reprogrammed every time the board is restarted

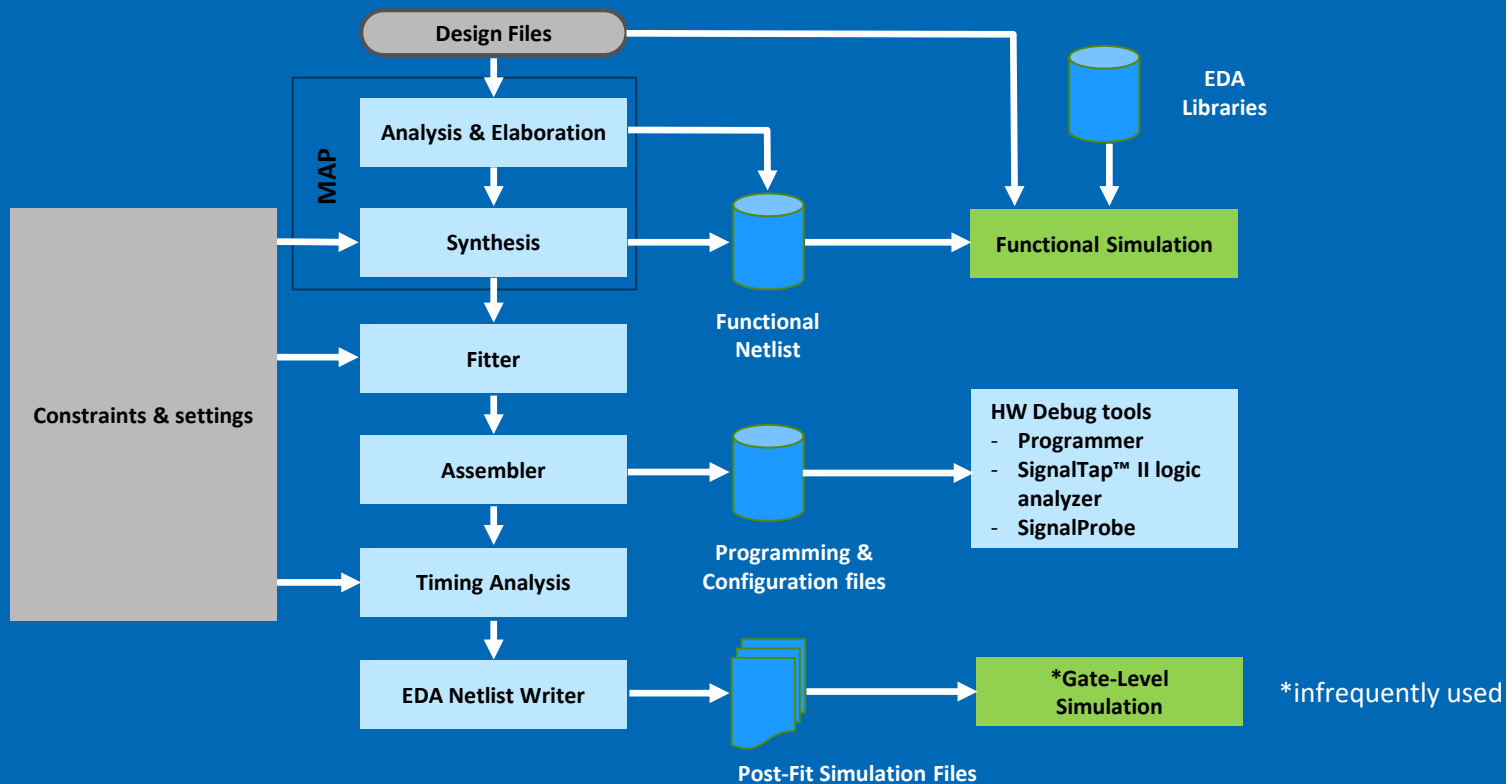
Quartus User Interface

■ Quartus Prime Software Main Window

- **Project Navigator** shows your project hierarchy, source files, design units, IP and design revisions in your project.
- **Tasks** window shows the status of the design and can be used to run or re-run parts of the design flow
- **Messages** window outputs messages from each process of the run.
- **Files** window has tabs for the report browser, open design files and any other file opened by the user.
- **IP Catalog** window is open by default and is used to generate IP functions that are to be used in your design.

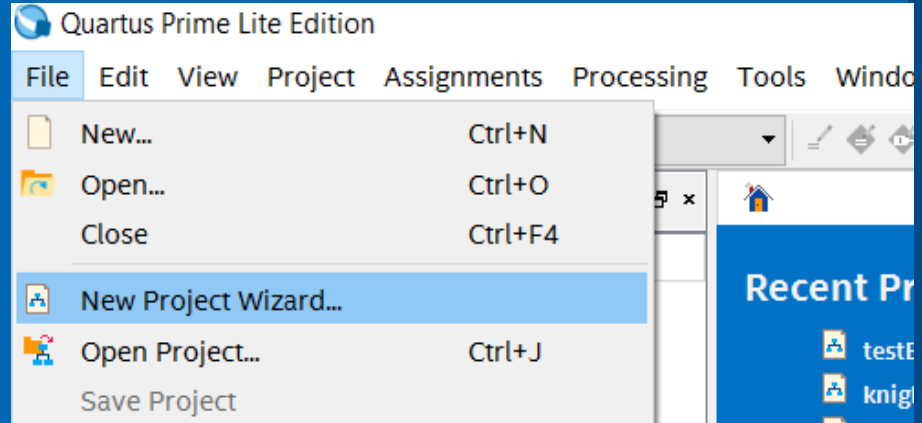


Tools Flow



New Project Wizard

1. Name project
2. Set Working Directory & Top-Level Entity
3. Add source files
4. Select Device
5. EDA tool settings



All settings can be modified later. Some steps can be skipped.

The top level entity must match the top level module in your design exactly (case sensitive) in order to avoid a compile error.



Helpful
Tips

Family & Device Settings

New Project Wizard

Family, Device & Board Settings

Device Board

Select the family and device you want to target for compilation.
You can install additional device support with the Install Devices command on the Tools menu.

To determine the version of the Quartus Prime software in which your target device is supported, refer to the [Device Support List](#) webpage.

Device family

Family: MAX 10 (DA/DF/DC/SA/SC)

Device: All

Target device

☐ Auto device selected by the Fitter

☒ Specific device selected in 'Available devices' list

☐ Other: n/a

Show in 'Available devices' list

Package: Any

Pin count: Any

Core speed grade: Any

Name filter: 10M50DAF484

☒ Show advanced devices

Available devices:

Name	Core Voltage	LEs	Total I/Os	GPIOs	Memory Bits	Embedded multiplier 9-bit cl
10M50DAF484C6GES	1.2V	49760	360	360	1677312	288
10M50DAF484C7G	1.2V	49760	360	360	1677312	288
10M50DAF484C8G	1.2V	49760	360	360	1677312	288
10M50DAF484C8GES	1.2V	49760	360	360	1677312	288

< Back Next > Finish Cancel Help

Expand the window so you can see all the fields

Get the part number for your specific device by looking on the chip on your board or the side of the box.

Pin Planner

Pin Planner - C:/altera_lite/15.1/UCLA/Lab1 - Lab1

File Edit View Processing Tools Window Help

Search altera.com

Groups

Named: *

Node Name	Direction
LED[9..0]	Output Group
SW[9..0]	Input Group

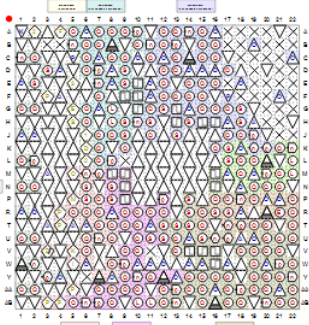
Report

Report not available

Tasks

- Early Pin Planning
 - Early Pin Planning...
 - Run I/O Assignment Analysis

Top View - Wire Bond
Cyclone V - 5CEB44F23C7



Named: * Edit: X Filter: Pins: all

Node Name	Direction	Location	I/O Bank	VREF Group	Filter Location	I/O Standard	Reserved	Current Strength	Slew Rate
LED[4]	Output				PIN_L19	2.5 V (default)		12mA (default)	1 (default)
LED[3]	Output				PIN_AB8	2.5 V (default)		12mA (default)	1 (default)
LED[2]	Output				PIN_H6	2.5 V (default)		12mA (default)	1 (default)
LED[1]	Output				PIN_G8	2.5 V (default)		12mA (default)	1 (default)
LED[0]	Output				PIN_H9	2.5 V (default)		12mA (default)	1 (default)
SW[9]	Input				PIN_B6	2.5 V (default)		12mA (default)	
SW[8]	Input				PIN_W2	2.5 V (default)		12mA (default)	
SW[7]	Input				PIN_B7	2.5 V (default)		12mA (default)	
SW[6]	Input				PIN_U2	2.5 V (default)		12mA (default)	
SW[5]	Input				PIN_W19	2.5 V (default)		12mA (default)	

0% 00:00:00

Compile your design

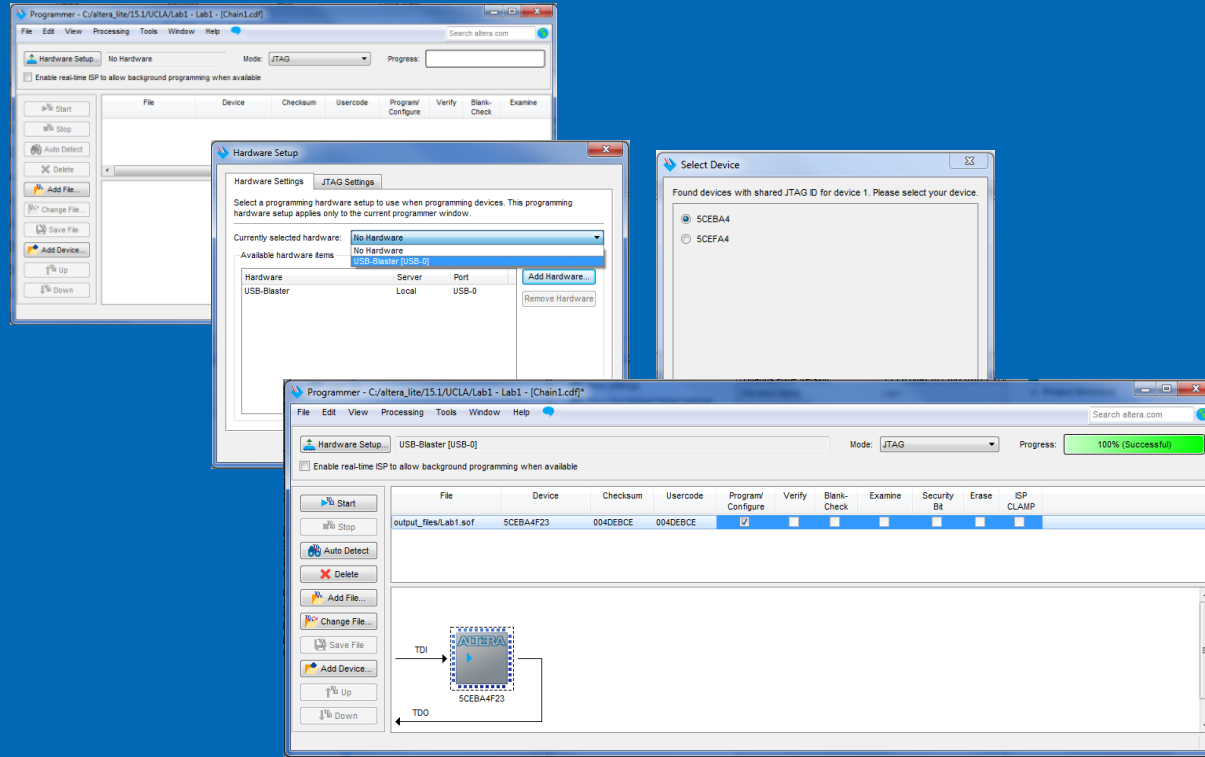
✓	▶ ▶ Compile Design	00:01:59
✓	▶ ▶ Analysis & Synthesis	00:00:40
✓	▶ ▶ Fitter (Place & Route)	00:00:42
✓	▶ ▶ Assembler (Generate programming files)	00:00:19
✓	▶ ▶ TimeQuest Timing Analysis	00:00:18

⚠ 332012 Synopsys Design Constraints File file not found: 'Lab1.sdc'.

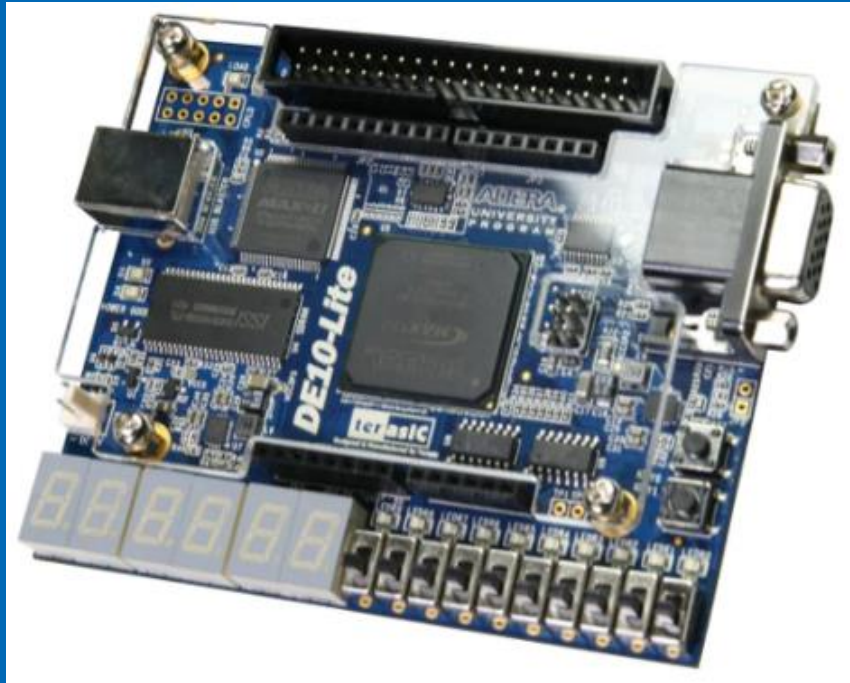
Helpful
Tips

Warnings shown in blue won't prevent your design from compiling or being programmed, but they could indicate possible bugs. This lab does not have any design constraints, so the .sdc file is not needed. You will learn how to create one in the timing analysis workshop.

Program your FPGA



Test your design



Live Hardware

DE10-Lite Development Kit

The screenshot shows the DE10-Lite Development Kit software interface. At the top, there's a title bar with the text 'DE10-Lite Development Kit'. Below the title bar is a large image of the development kit board. The board features an 'intel MAX 10 inside' logo, a 'terasic' logo, and 'DE10-Lite for Intel FPGA University Program' text. The board is populated with various components, including a USB-B connector, a USB-A connector, a DVI-D connector, and a series of BNC connectors. Below the board image, there are three input fields for '32-bit Parameter 0', '32-bit Parameter 1', and '32-bit Parameter 2'. To the right of these fields is a text box displaying the following information: 'Board launched!', 'Connected to USB-Blaster [USB-0]', 'Time remaining: 20:19', and 'Ping: 22 ms'. Below the text box is a 'Reset Timer' button.

32-bit Parameter 0

32-bit Parameter 1

32-bit Parameter 2

Board launched!
Connected to USB-Blaster [USB-0]
Time remaining: 20:19
Ping: 22 ms

Reset Timer

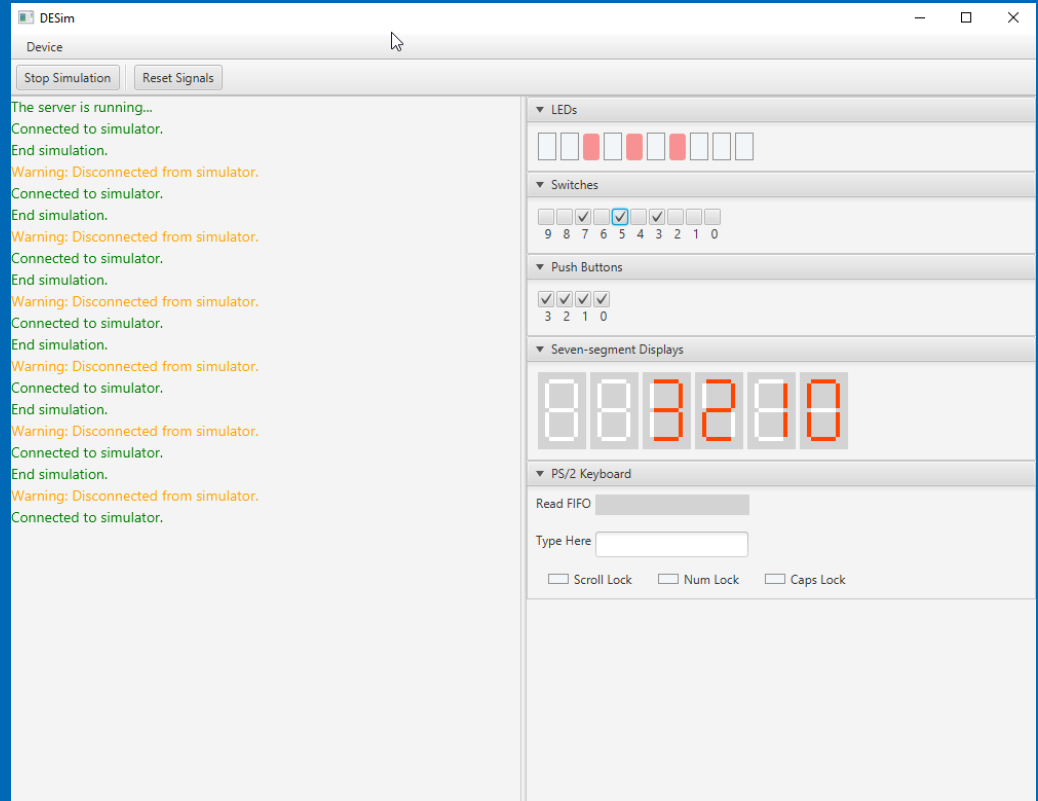
Remote Console

Covid-19: New challenges for remote learners

- In June 2020, we launched initiatives to help remote learners
- Addresses situation where student does not have a physical board in their possession
- Simulation based only requires Modelsim
 - Quick compiles
 - More setup with IP
 - Slower clock speed
- Hardware based requires Intel Quartus Prime-Lite
 - Longer compiles, but hardware accurate
 - More complex networking, adds a level of hierarchy in the design

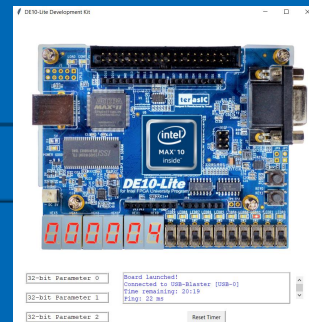
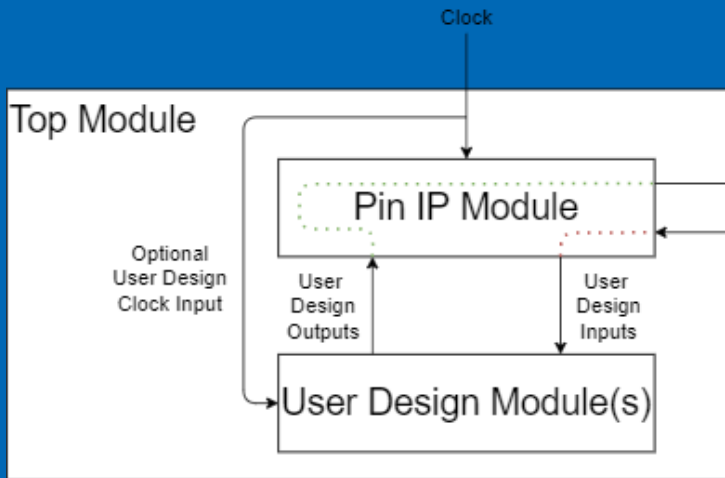
Simulation Console for FPGAs

- Supports: Switches, Push Buttons, LED, 7-Segments, PS2 character i/f and VGA Display
- Modelsim mimics behavior of board
 - No need for viewing waveforms



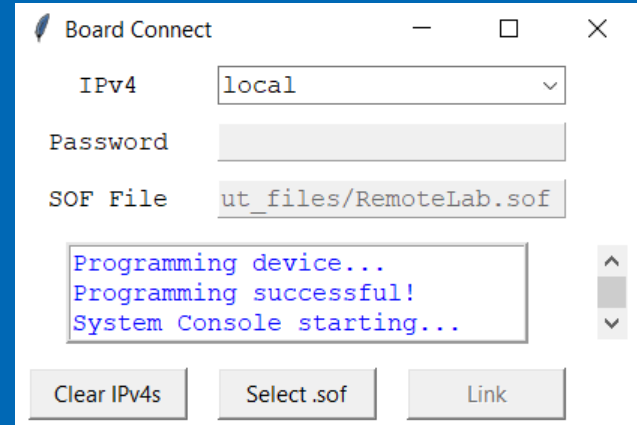
Remote console used in today's lab

- Support for remote GPIO-based labs.
 - Good for undergraduate curriculum.
- Leverages existing Quartus Prime Lite installation.
 - No additional software needed.
 - Up and running in minutes.
- Requirements:
 - Windows machine
 - Quartus Prime Lite
 - Access to local or remote development kit



Connecting to Development Kit

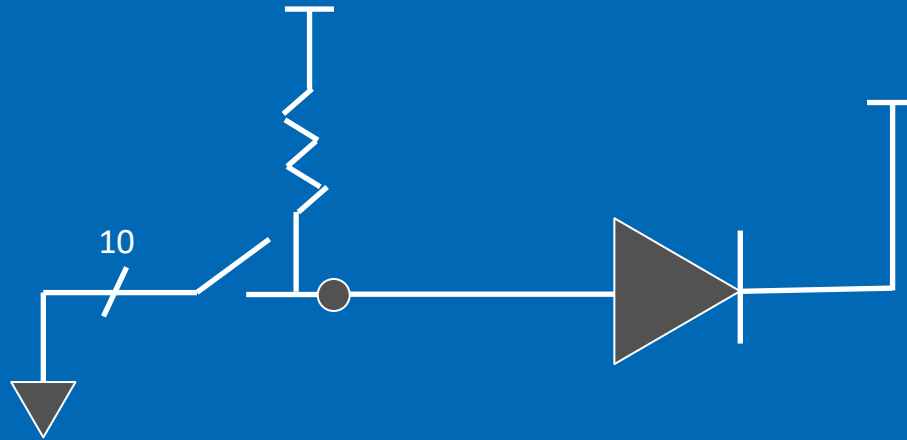
- Right-click launch task
- Select "Start"
- If development kit is locally connected keep IPv4 as "local"
 - Otherwise a host's IPv4 and password are used.
- Select project .sof.
- Link



New Remote Console Projects

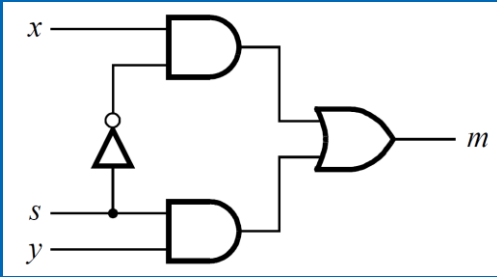
- Download the blankProject.qar example project for your dev kit.
- Unarchive the project into Quartus.
- Add source files and connect new design to top-level wrapper.
- Debug and compile.
- Launch Remote Console and validate functionality in hardware.

First Lab: Switch to LED



assign LED = SW;

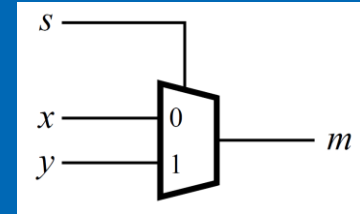
Next lab: Multiplexer



Circuit

s	m
0	x
1	y

Truth table

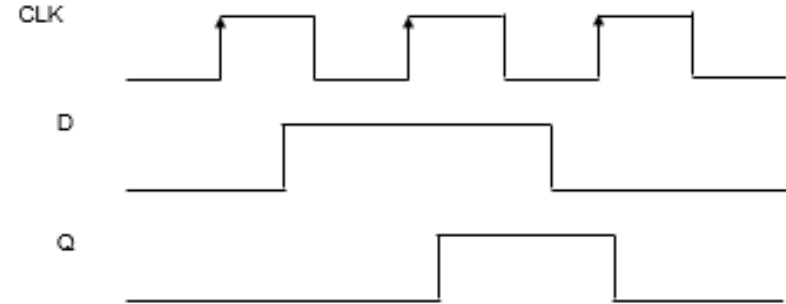
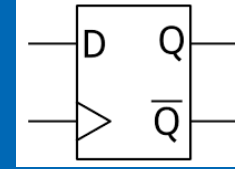


Symbol

The multiplexer can be described by the following Verilog statement:

```
assign m = (~s & x) | (s & y);
```

Knight Rider



Final Tips

- When Quartus Prime Lite first starts for the very first time it might ask you about purchasing a license, select Run Quartus, all licenses are free for this lab.
- If things fail to compile, check your top Level Entity Setting → Setting → Top Level Entity and make sure that the module <design> matches your top level entity, including case.
- Check the LEDR[0] and LEDR[9] pins carefully in the Knight Rider lab and see if they sequence properly. If not, study the code carefully!
- Sometimes copy and paste from files into Quartus has carriage return formatting errors. If you see run on lines with no carriage return, you need to copy things over line by line, or add the appropriate file to your project. It's better to click on the links and download the files.

Learning more

- DE10-Lite development kit - \$55 from Terasic with student discount
- <http://fpgauniversity.intel.com> – lots of helpful labs and tutorials
- Customer training site (just google Intel FPGA training)
 - In catalog, search on university
- Practice makes perfect!
- Thanks for attending!