INTRODUCTION TO FPGA DESIGN IN QUARTUS (REMOTE EDITION)

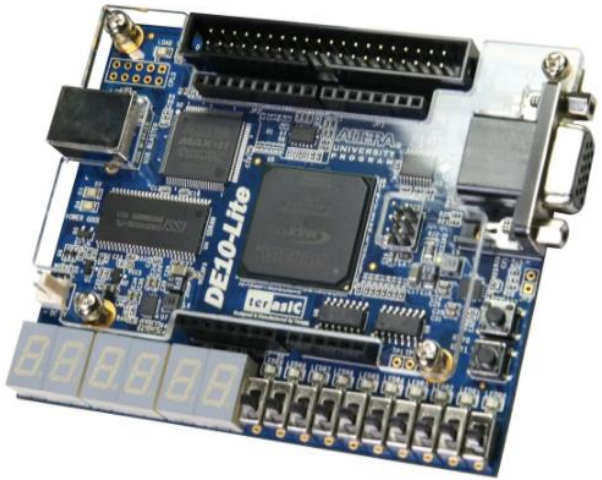
November 2020

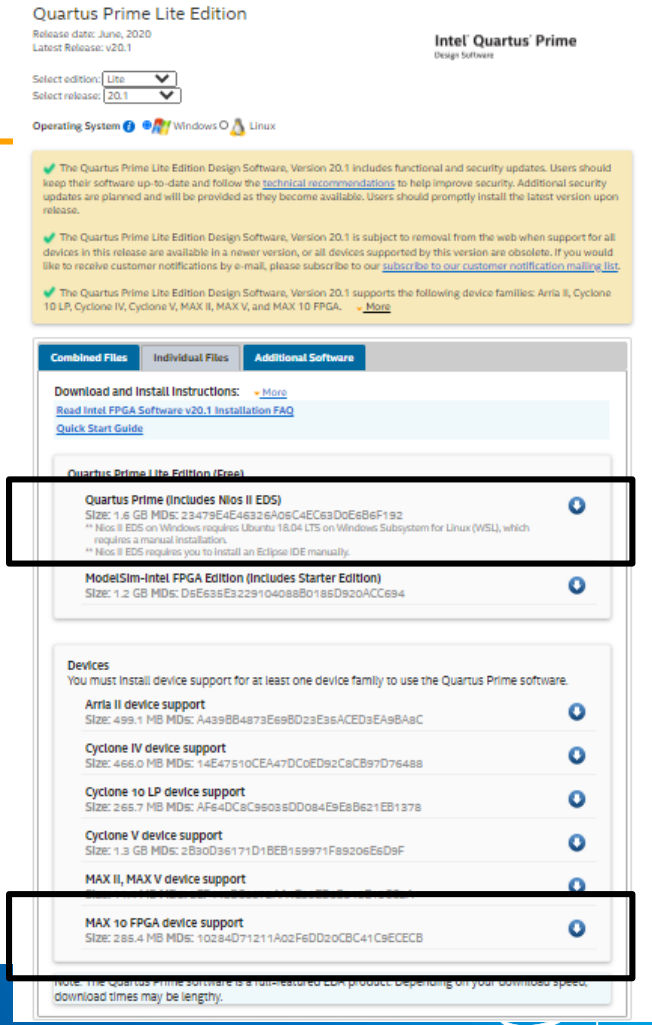
TOPICS

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

COMPANION LAB

1. Download the Quartus Prime Lite Software:
2. We will run with a remote DE10-Lite development board





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

THE RISE OF NEW MARKETS

THE BIRTH OF FPGAs

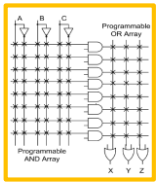
1970

First PROM; programmable, but permanent

1971

First EPROM; erased using UV light

1975



First PLA; limited to sum of products

1983

Altera founded

EEPROM

1983

1984

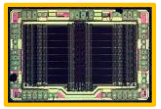
First EPLD; re-programmable CMOS

1985

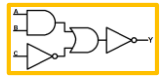
First FPGA; 64 logic blocks w/ two three-input LUTs

1988

High-density CPLD; full subsystems



1989



Altera launches first Integrated graphic design environment

SOURCE: https://www.altera.com/about/company/history.html

INTEL'S DEEP HISTORY WITH FPGAs

1968 — Intel is founded by Robert Noyce and Gordon Moore

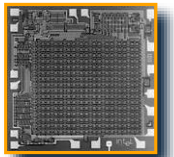
1971 — World's first microprocessor

1983 — Founded

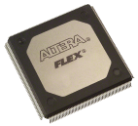
1984 — First PLD

1984 — Intel and Altera establish a joint marketing agreement

1994 — Altera purchases Intel's PLD business



1995 — First embedded RAM



2000 — First soft micro-processor

2002 — First embedded DSP

2012 — SoC FPGA

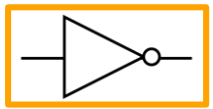
2013 — Intel and Altera start foundry relationship for 14nm FPGAs and SoCs

2015 — Intel acquires Altera

2016 — System-in-Package (SIP)

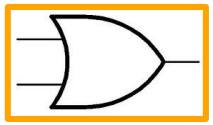
BACK TO THE BASICS

A	Z
0	1
1	0



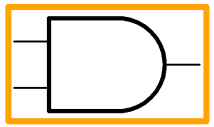
Inverter
Z = ~A

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1



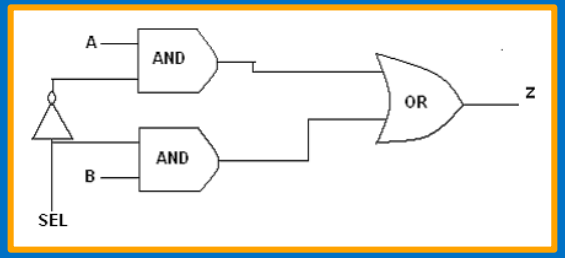
OR
Z = A | B

A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1



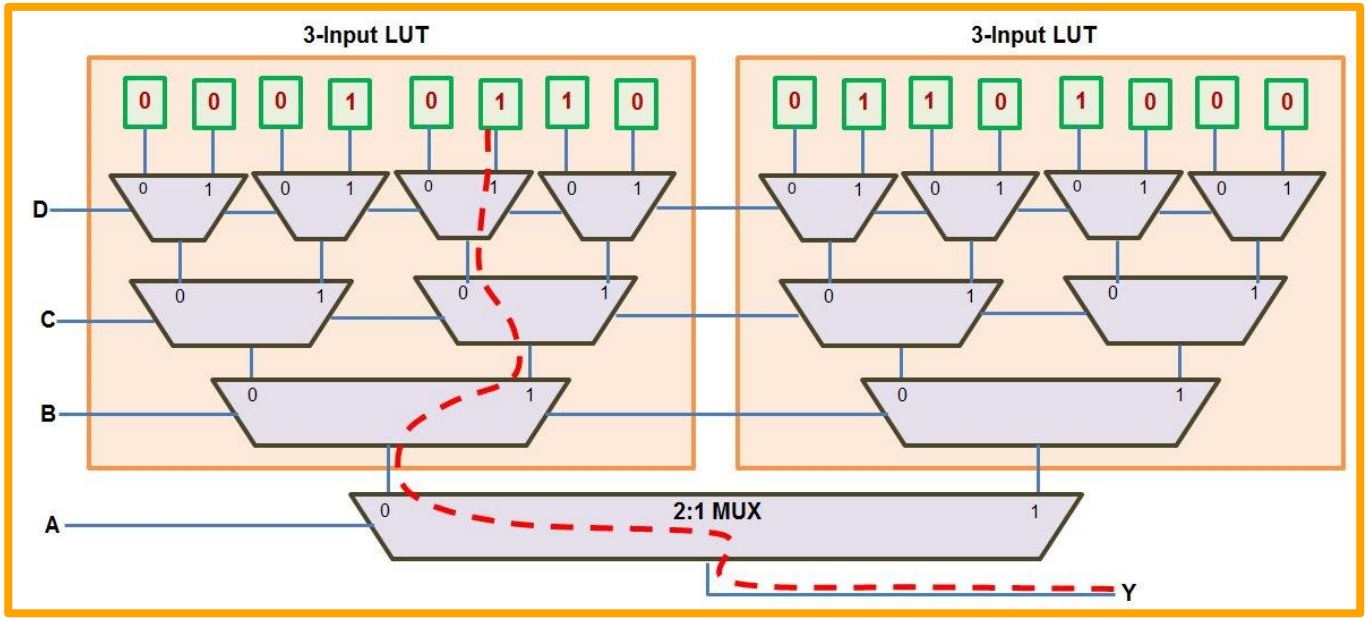
AND
Z = A & B

S	A	B	Z
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

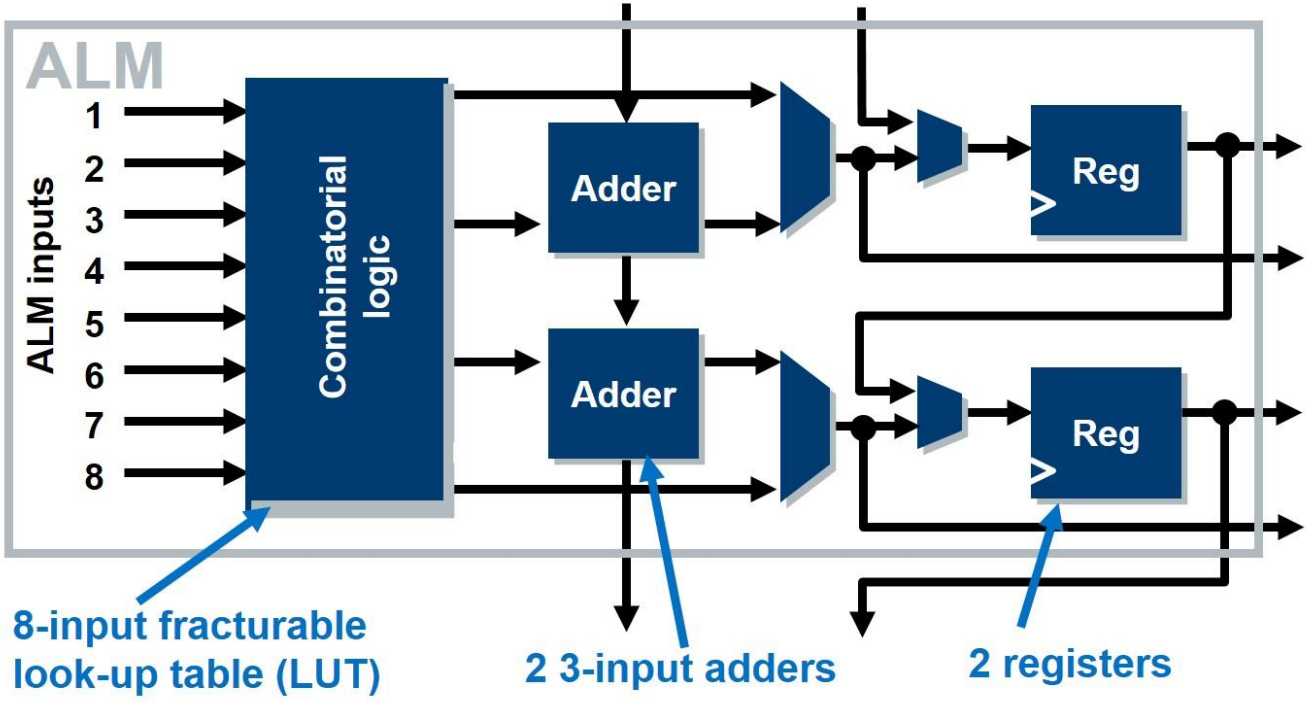


2:1 MUX
Z = (~S & A) | (S & B)

LOOK-UP TABLE (LUT): THE FOUNDATION

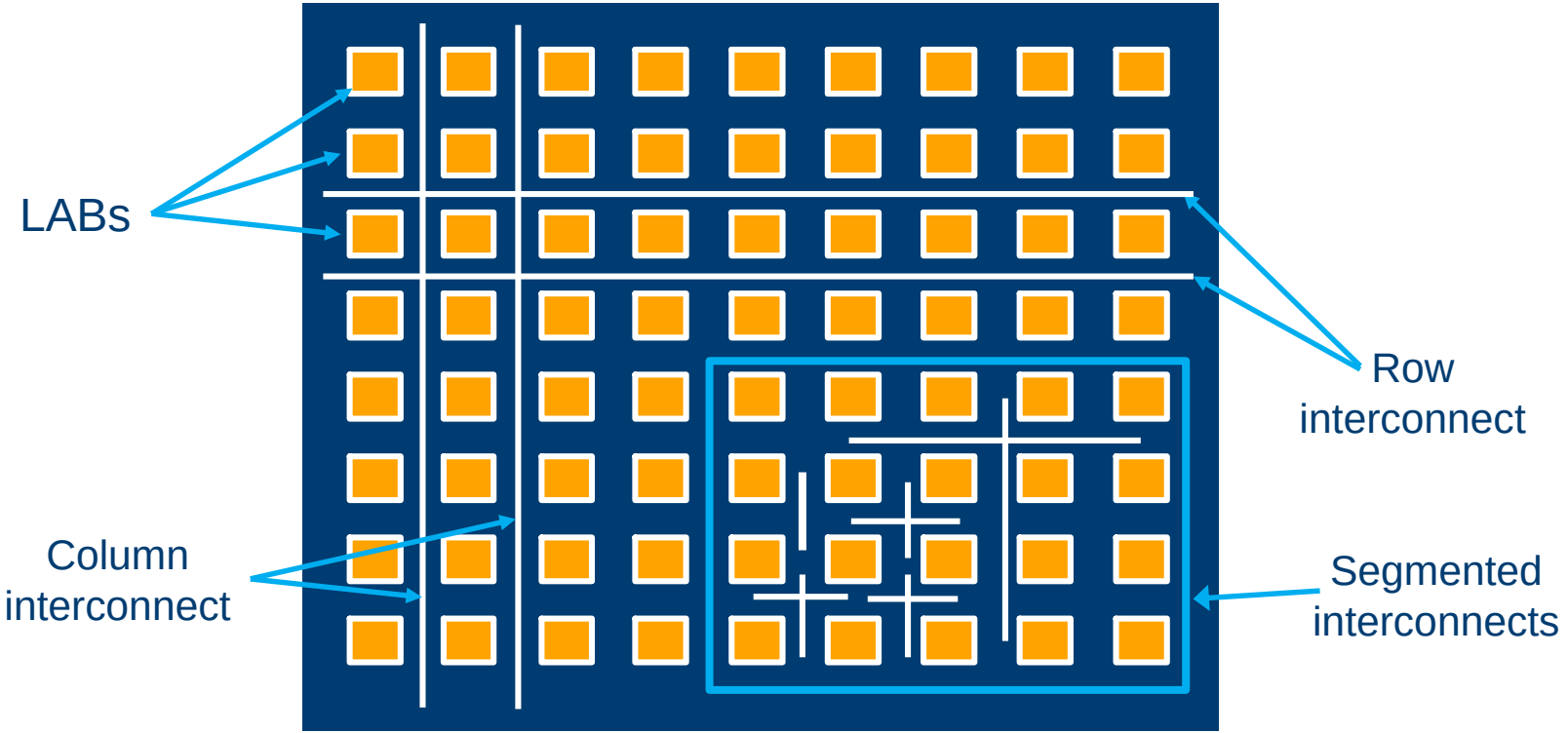


LOGIC ARRAY (BUILDING) BLOCKS



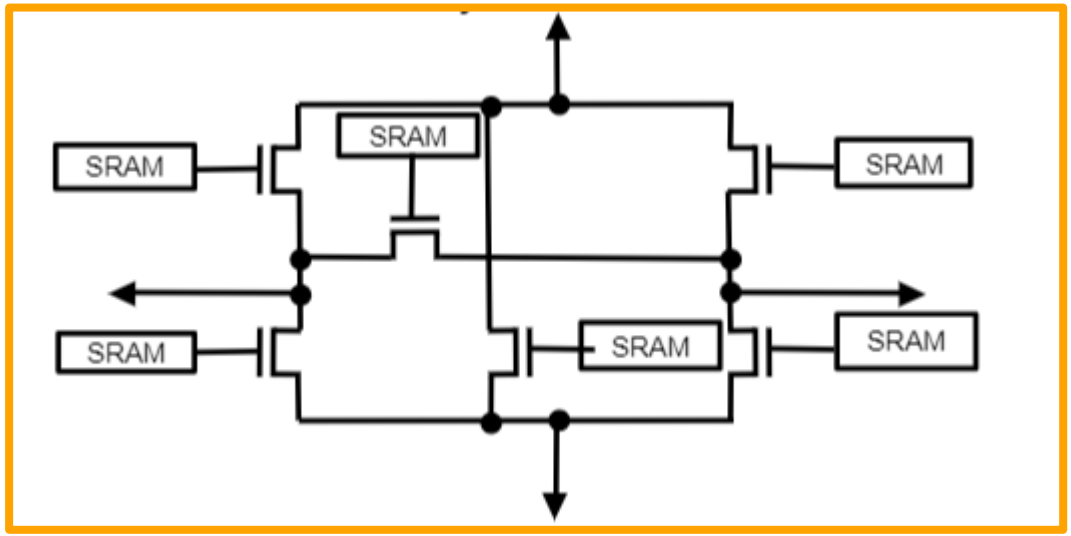
x 10

BUILDING THE ARRAY



HOW ITS PROGRAMMED

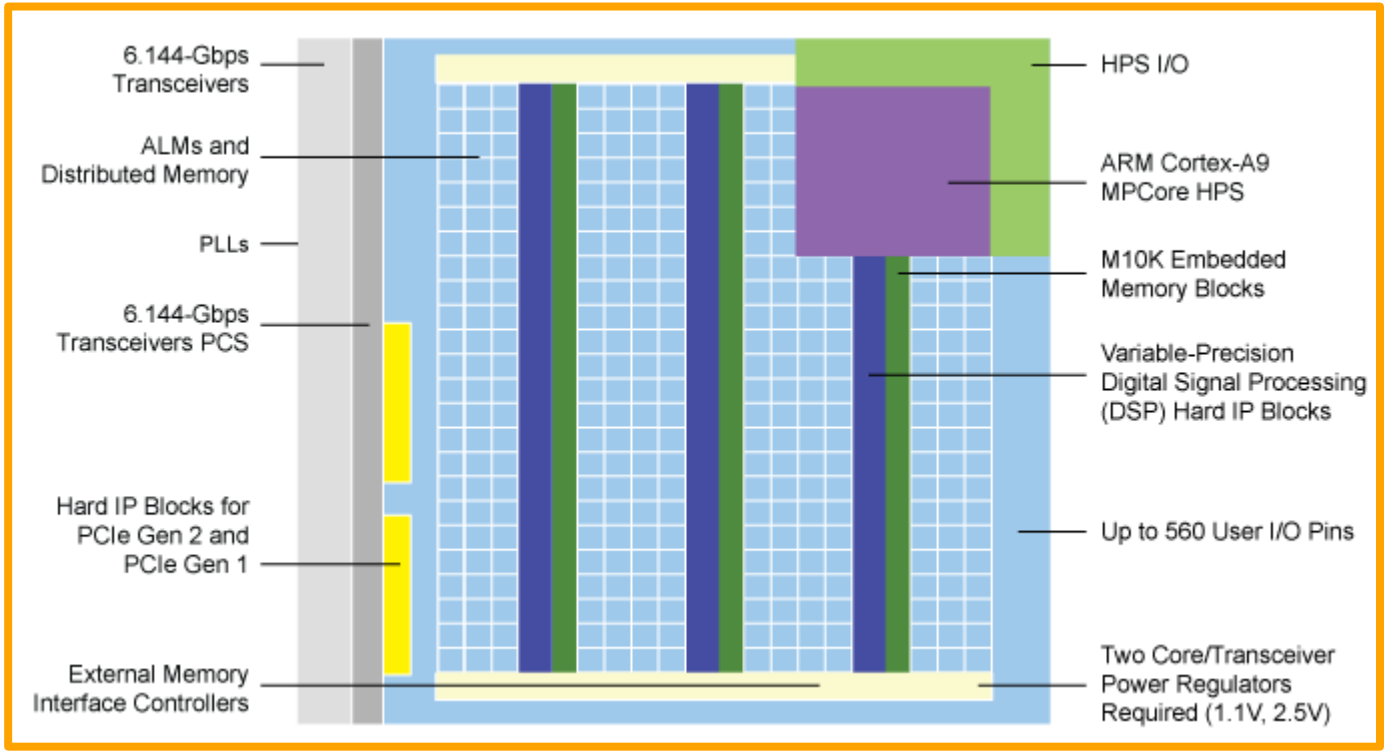
Row/Column Interconnect Junction



Programming info stored in a external non-volatile device

- Active: programmed automatically at power-on
- Passive: Intelligent host (CPU) controls programming

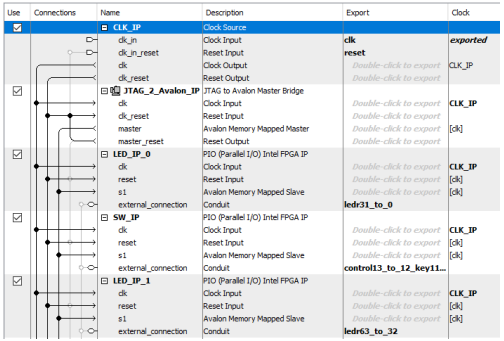
MODERN FPGAS



Source: https://www.altera.com/products/fpga/cyclone-series/cyclone-v/features.html

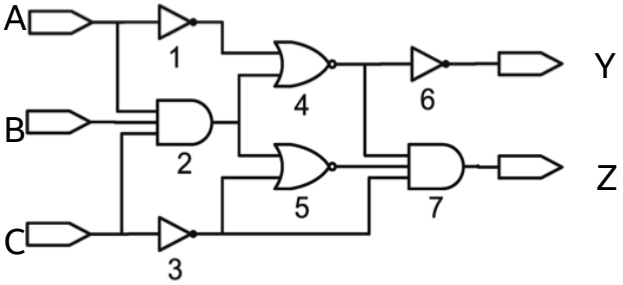
DESCRIBING FPGAS

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




```
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();
}
```

Y = ~(~(~A | (A&B&C)));
Z = (~(~A | (A&B&C))) & ((A&B&C) | ~C) & ~C);

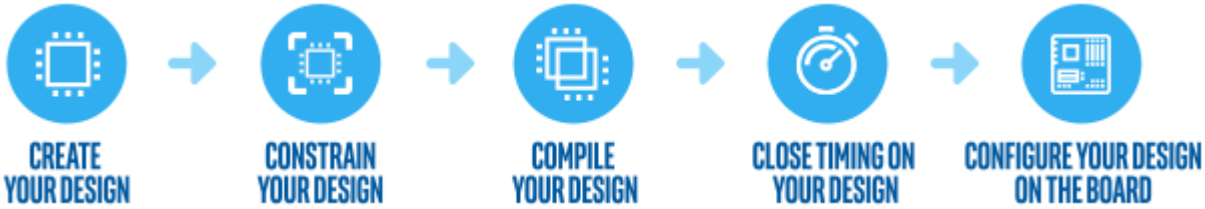


WHAT IS IP (INTELLECTUAL PROPERTY)?

- 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.

INTRO TO 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

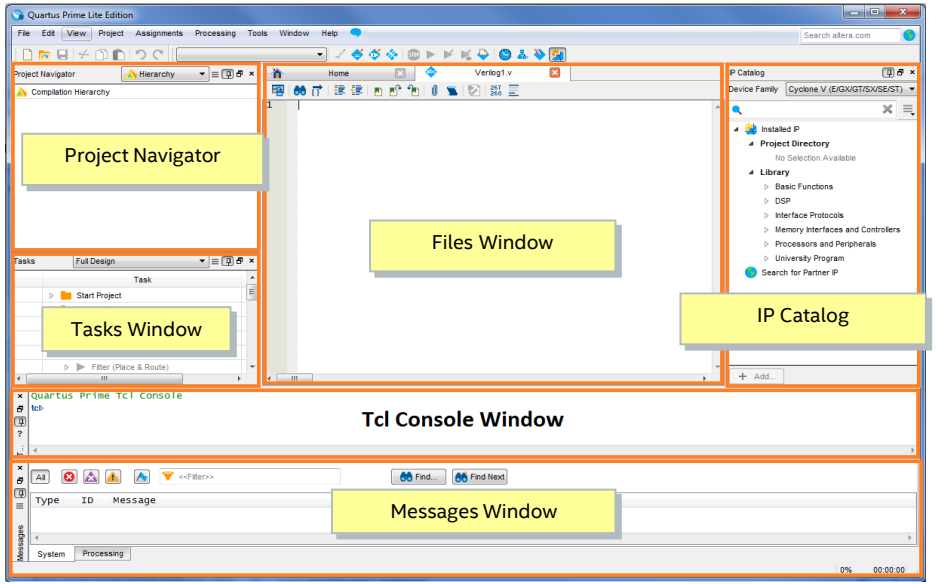


- Many formats to program an FPGA
  - In this class we will use a ".sof" file (SRAM object file)
  - The .sof file is "volatile" and needs to be reprogrammed every time the board is restarted

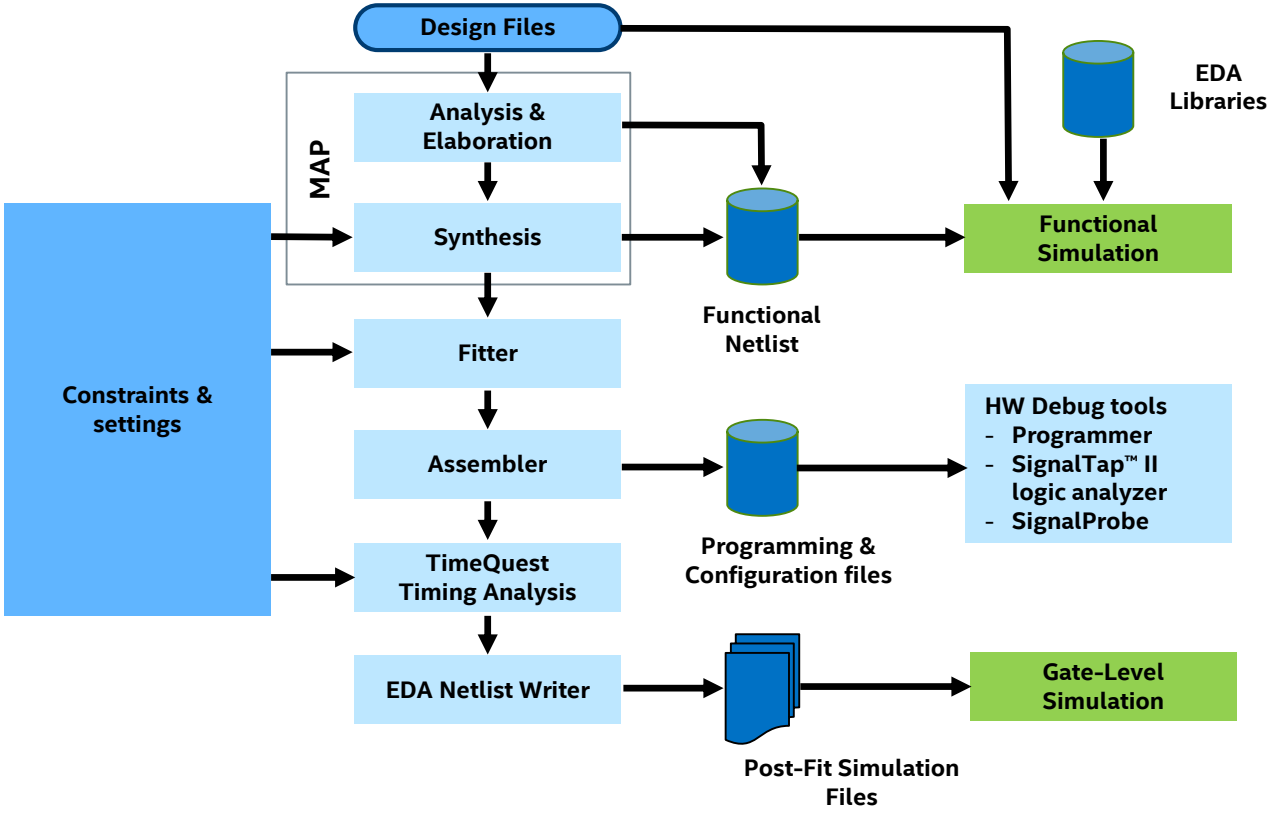
QUARTUS USER INTERFACE

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.

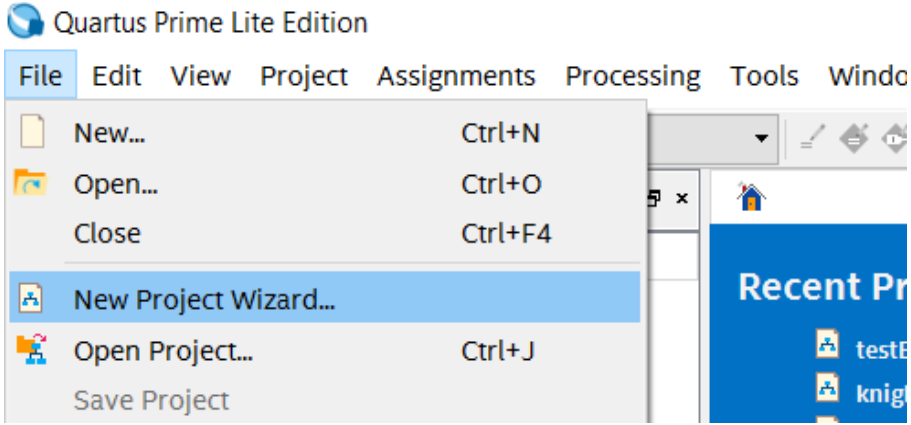


TOOLS OVERVIEW



NEW PROJECT WIZARD

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

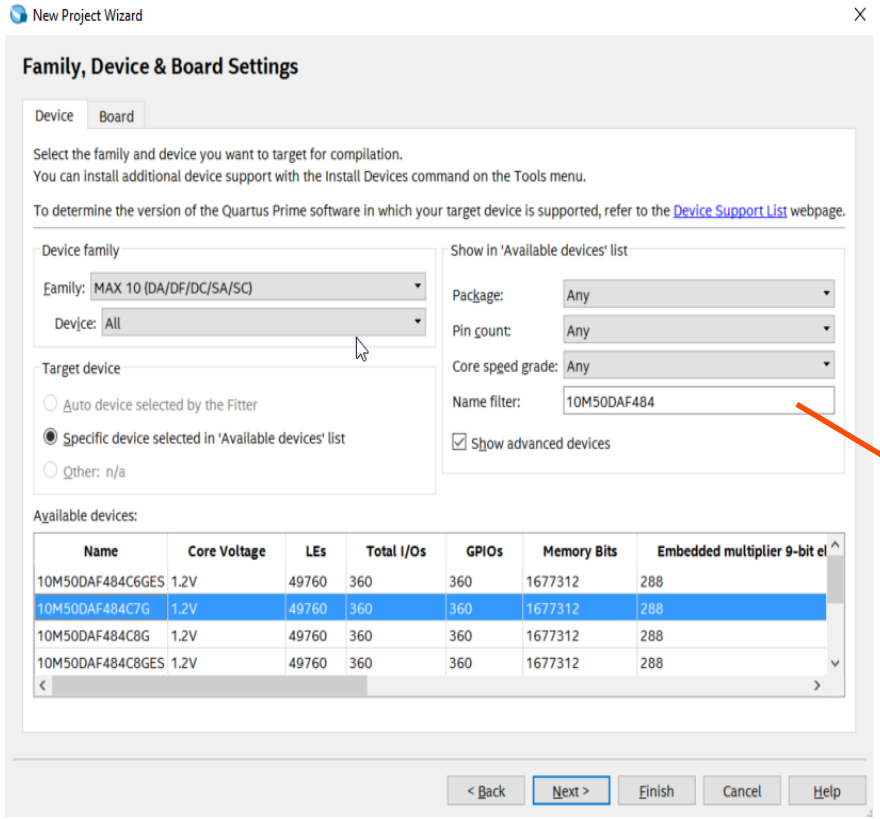


Helpful Tips

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.

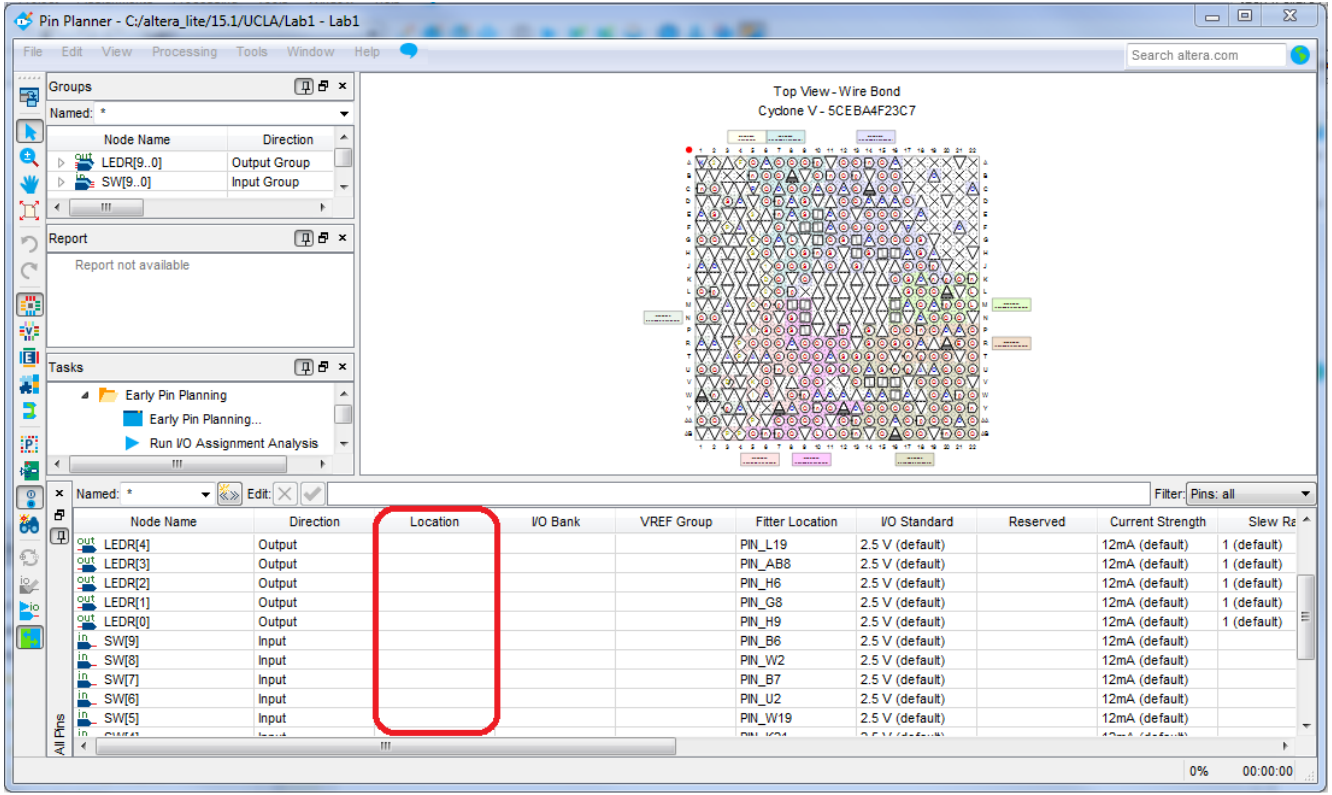
HINT: FAMILY & DEVICE SETTINGS



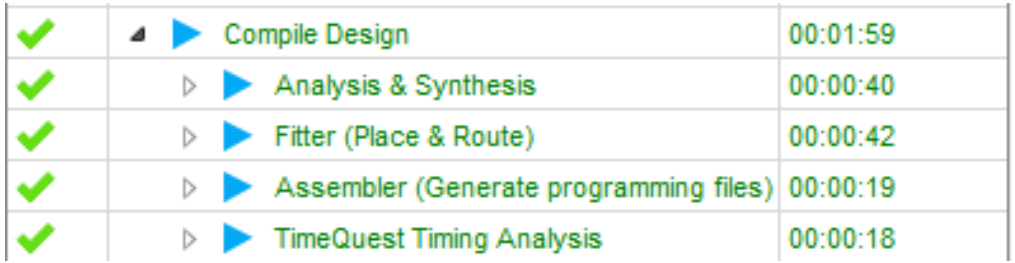
Expand the window so you can see all the fields

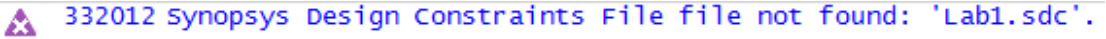
Get the part number for your specific device by looking on the chip on your board or the side of the box.

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
LEDR[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 - 5CEBA4F23C7
Named: * Edit: Filter: Pins: all
Node Name Direction Location I/O Bank VREF Group Fitter Location I/O Standard Reserved Current Strength Slew Ra
LEDR[4] Output PIN_L19 2.5 V (default) 12mA (default) 1 (default)
LEDR[3] Output PIN_AB8 2.5 V (default) 12mA (default) 1 (default)
LEDR[2] Output PIN_H6 2.5 V (default) 12mA (default) 1 (default)
LEDR[1] Output PIN_G8 2.5 V (default) 12mA (default) 1 (default)
LEDR[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)
All Pins
0% 00:00:00

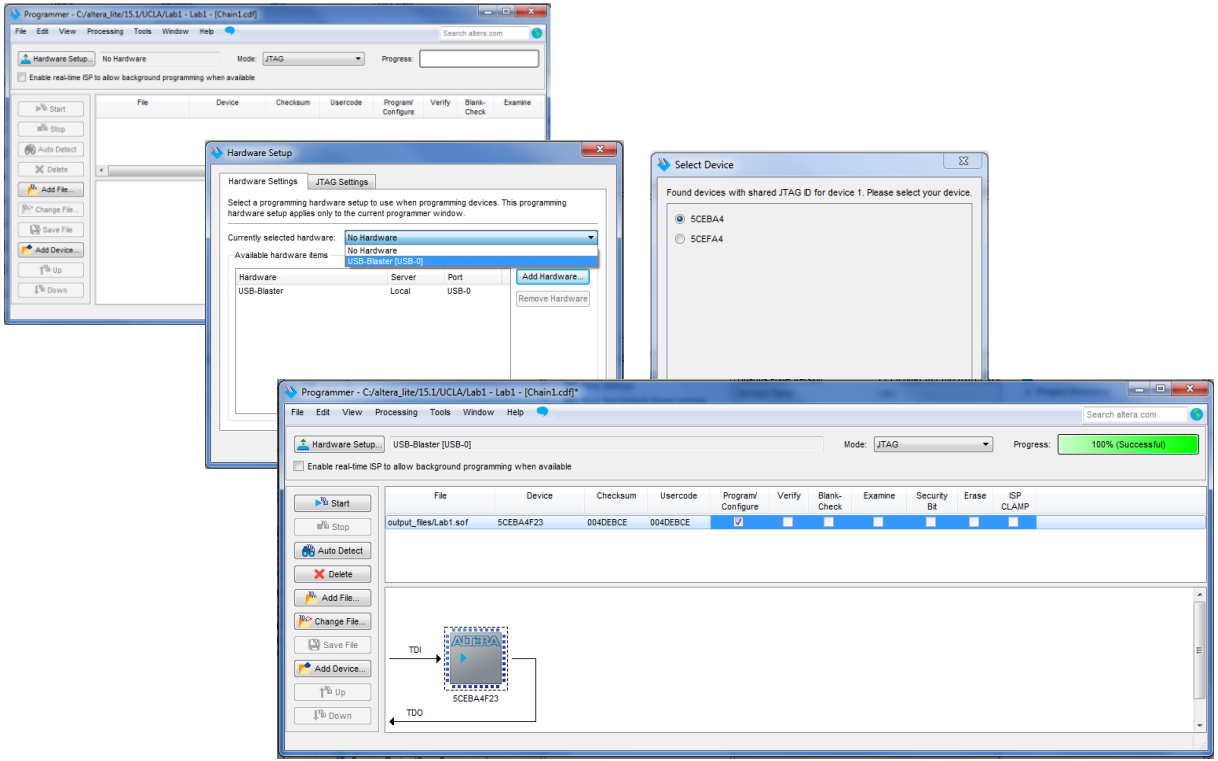
COMPILE YOUR DESIGN



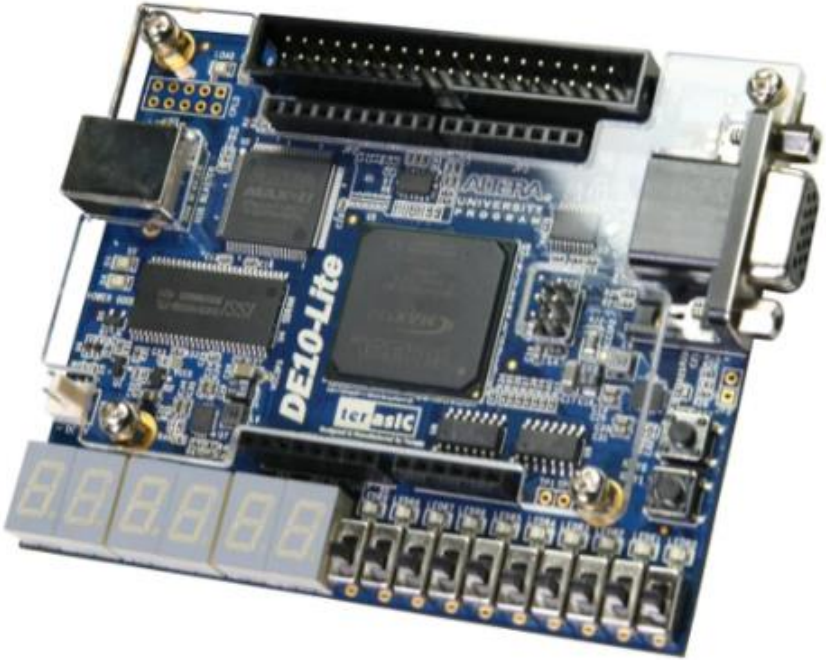


Warnings shown in blue won't prevent your design from compiling or being programmed, but they could indicate possible bugs. This lab does not have any design constraints, so the .sdc file is not needed. You will learn how to create one in the timing analysis workshop.

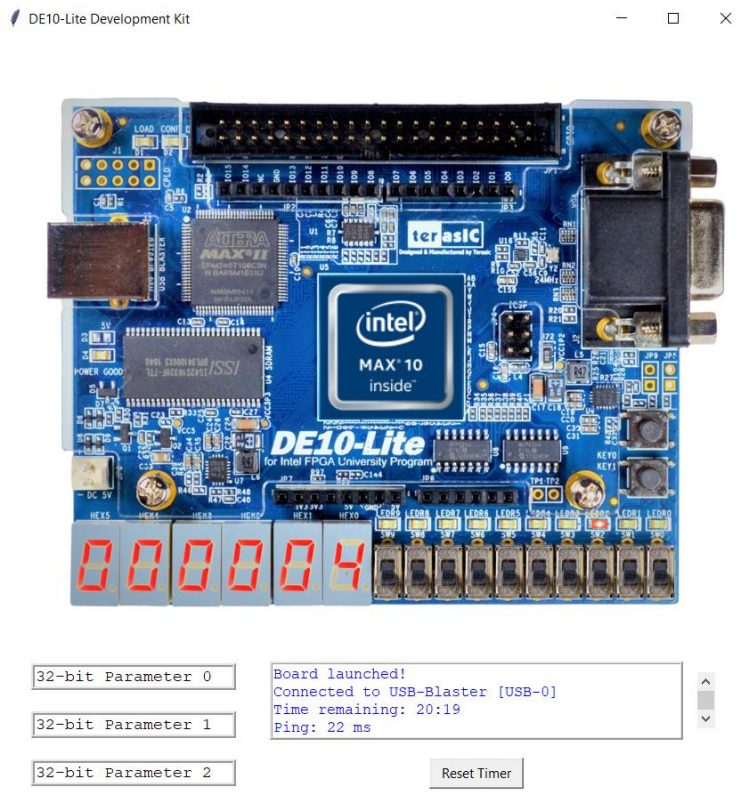
PROGRAM YOUR FPGA



TEST YOUR DESIGN



Live Hardware



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

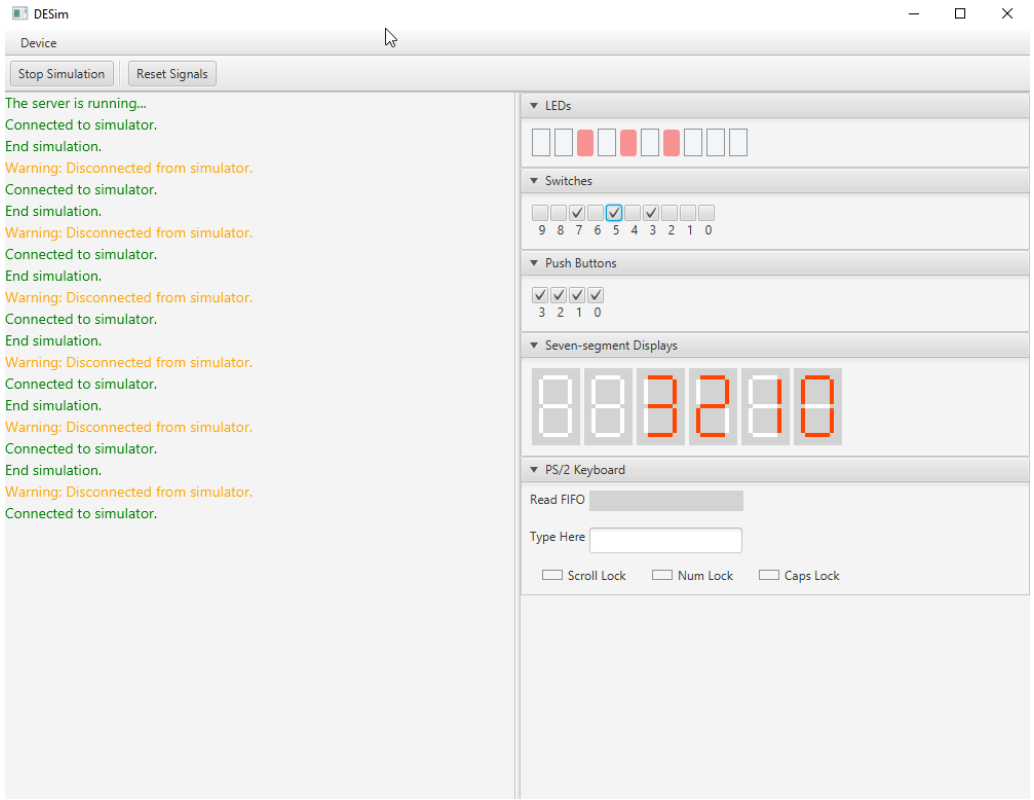
SIMULATION CONSOLE FOR FPGAS

Supports: Switches, Push Buttons, LED, 7-Segments, PS2 character i/f and VGA Display

Modelsim mimics behavior of board

Good for initial bring-up and debug

- No need for viewing waveforms



OVERVIEW

Support for remote GPIO-based labs.

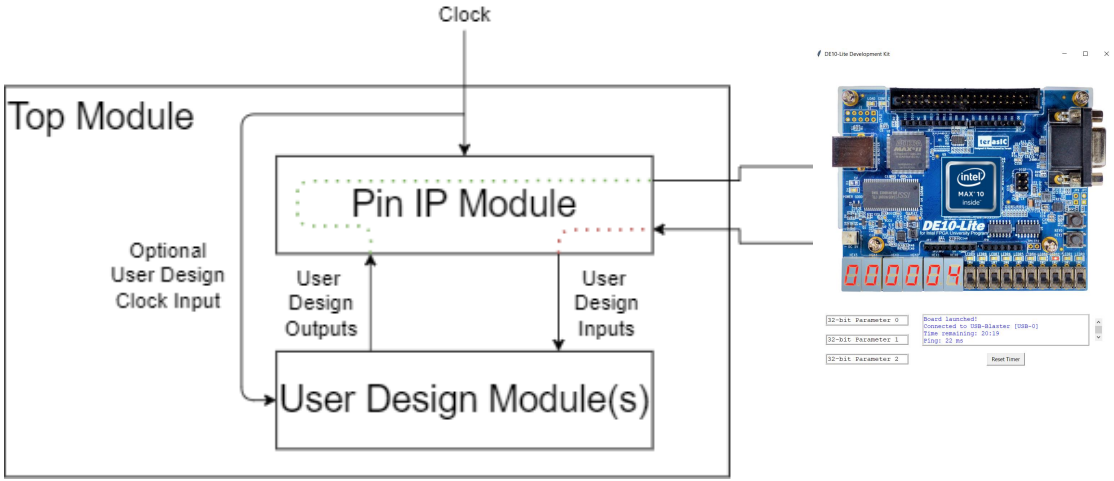
- Good for undergraduate curriculum.

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



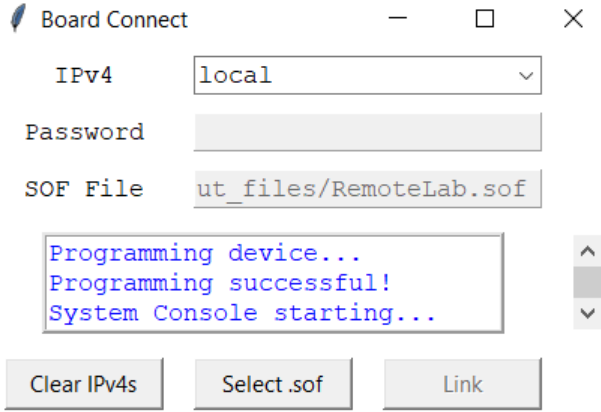
CONNECTING TO DEVELOPMENT KIT

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.

HOSTING DEVELOPMENT KITS

Requirements:

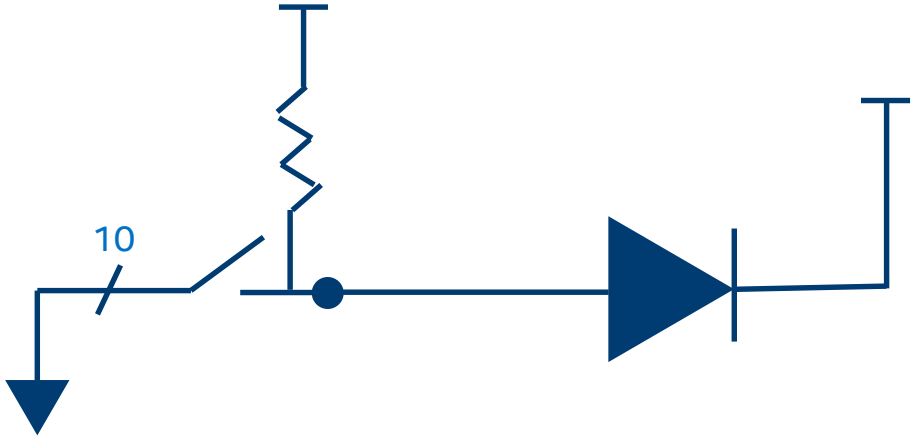
- Linux Ubuntu 18.04
- Static IP assignment
- Enough USB connections (supports USB hubs)
- Quartus Prime Lite Programmer and Tools package
- Clients with VPN privileges

Run setup bash script as root

Distribute IPv4/passwords to clients

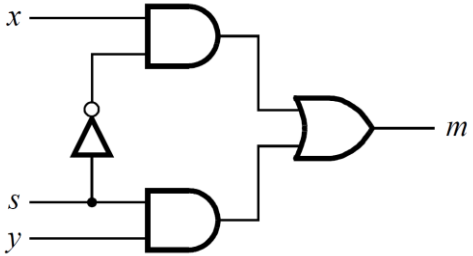
- Saved in jtag_server_passwords.txt

FIRST LAB: SWITCH TO LED



assign LED = SW;

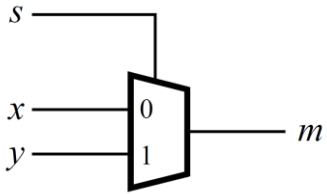
NEXT LAB: MULTIPLEXER



Circuit

s	m
0	x
1	y

Truth table

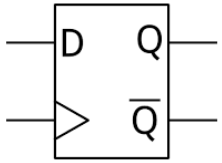


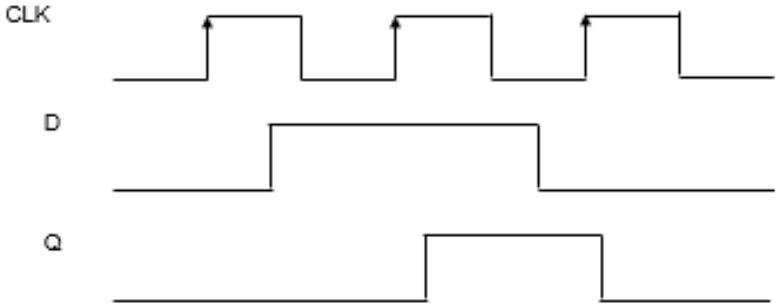
Symbol

The multiplexer can be described by the following Verilog statement:

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

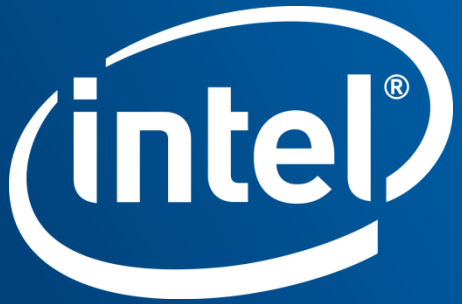
KNIGHTRIDER





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.


intel®