FPGA and Xilinx ISE
FPGA Basics

- What is FPGA
  - Field Programmable Gate Array
  - An FPGA is a regular structure of logic cells (modules) and interconnect, which is under the designer’s complete control.
  - An FPGA is really some programmable logic with a whole bunch of programmable wires.

- How to program
  - Volatile
    - SRAM-Based, reprogrammable
  - Non volatile
    - Anti-fuse, one time programmable
Inside FPGA

- All Xilinx FPGAs contain some basic resources
  - Slices (grouped into Configurable Logic Blocks (CLBs))
    - Contain combinatorial logic and register resources
  - IOBs
    - Interface between the FPGA and the outside world
  - Programmable interconnect
  - Other resources
    - Memory
    - Multipliers
    - Processors
    - Clock management
Slices and CLBs

- Each Virtex™-II CLB contains four slices
  - Local routing provides feedback between slices in the same CLB, and it provides routing to neighboring CLBs
  - A switch matrix provides access to general routing resources
Simplified Slice Structure

- Each slice has
  - Two 4-input look-up tables (LUTs)
    - Any 4-input logic functions
  - Four outputs
    - Two registered outputs, two non-registered outputs
  - Carry logic
    - Fast arithmetic logic
  - Other controls
    - e.g. set/reset
Virtex-II Pro Features

- Up to 24 RocketIO™ Multi-Gigabit Transceiver (MGT) blocks
  - Serializer and deserializer (SERDES)
  - Fibre Channel, Gigabit Ethernet, XAUI, Infiniband compliant transceivers, and others
  - 8-, 16-, and 32-bit selectable FPGA interface
  - 8B/10B encoder and decoder

- PowerPC™ RISC processor blocks
  - Thirty-two 32-bit General Purpose Registers (GPRs)
  - Low power consumption: 0.9mW/MHz
  - IBM CoreConnect bus architecture support
# Virtex-II-Pro Datasheet

<table>
<thead>
<tr>
<th>Device</th>
<th>RocketIO Transceiver Blocks</th>
<th>PowerPC Processor Blocks</th>
<th>Logic Cells</th>
<th>CLB (1 = 4 slices = max 128 bits)</th>
<th>18 X 18 Bit Multiplier Blocks</th>
<th>Block SelectRAM+</th>
<th>Maximum User I/O Pads</th>
</tr>
</thead>
<tbody>
<tr>
<td>XC2VP2</td>
<td>4</td>
<td>0</td>
<td>3,168</td>
<td>1,408</td>
<td>12</td>
<td>12</td>
<td>216</td>
</tr>
<tr>
<td>XC2VP4</td>
<td>4</td>
<td>1</td>
<td>6,768</td>
<td>3,008</td>
<td>94</td>
<td>28</td>
<td>504</td>
</tr>
<tr>
<td>XC2VP7</td>
<td>8</td>
<td>1</td>
<td>11,088</td>
<td>4,928</td>
<td>154</td>
<td>44</td>
<td>792</td>
</tr>
<tr>
<td>XC2VP20</td>
<td>8</td>
<td>2</td>
<td>20,880</td>
<td>9,280</td>
<td>290</td>
<td>88</td>
<td>1,584</td>
</tr>
<tr>
<td>XC2VPX20</td>
<td>8(4)</td>
<td>1</td>
<td>22,032</td>
<td>9,792</td>
<td>306</td>
<td>88</td>
<td>1,584</td>
</tr>
<tr>
<td>XC2VP30</td>
<td>8</td>
<td>2</td>
<td>30,816</td>
<td>13,696</td>
<td>428</td>
<td>136</td>
<td>2,448</td>
</tr>
<tr>
<td>XC2VP40</td>
<td>0(3), 8, or 12</td>
<td>2</td>
<td>43,632</td>
<td>19,392</td>
<td>606</td>
<td>192</td>
<td>3,456</td>
</tr>
<tr>
<td>XC2VP50</td>
<td>0(3) or 16</td>
<td>2</td>
<td>53,136</td>
<td>23,616</td>
<td>738</td>
<td>232</td>
<td>4,176</td>
</tr>
<tr>
<td>XC2VP70</td>
<td>16 or 20</td>
<td>2</td>
<td>74,448</td>
<td>33,088</td>
<td>1,034</td>
<td>328</td>
<td>5,904</td>
</tr>
<tr>
<td>XC2VPX70</td>
<td>20(4)</td>
<td>2</td>
<td>74,448</td>
<td>33,088</td>
<td>1,034</td>
<td>308</td>
<td>5,544</td>
</tr>
<tr>
<td>XC2VP100</td>
<td>0(3) or 20</td>
<td>2</td>
<td>99,216</td>
<td>44,096</td>
<td>1,378</td>
<td>444</td>
<td>7,992</td>
</tr>
</tbody>
</table>

**Notes:**
1. -7 speed grade devices are not available in Industrial grade.
2. Logic Cell \( \approx (1) \) 4-input LUT + (1)FF + Carry Logic
3. These devices can be ordered in a configuration without RocketIO transceivers. See Table 3 for package configurations.
4. Virtex-II Pro X devices equipped with RocketIO X transceiver cores.
FPGA Design Flow (Xilinx ISE)

Plan & Budget

Implement
- Translate
- Map
- Place & Route

Create Code/Schematic

HDL RTL Simulation

Synthesize to create netlist

Functional Simulation

Timing Simulation

Create BIT File

Attain Timing Closure
Design Entry

- Plan and budget
- Two design-entry methods: HDL or schematic
- Whichever method you use, you will need a tool to generate a netlist for implementation
  - Netlist: A text file that describes the actual circuit to be implemented at very low (gate) level
- Simulate the design to ensure that it works as expected!
Xilinx Implementation

- Once you generate a netlist, you can implement the design.
- There are several outputs of implementation:
  - Reports
  - Timing simulation netlists
  - Floorplan files
  - FPGA Editor files
  - and more!
What is Implementation?

- Implementation includes many phases
  - **Translate**: Merge multiple design files into a single netlist
  - **Map**: Group logical symbols from the netlist (gates) into physical components (slices and IOBs)
  - **Place & Route**: Place components onto the chip, connect the components, and extract timing data into reports
- Each phase generates files that allow you to use other Xilinx tools
  - Floorplanner, FPGA Editor, XPower
Timing Closure

1. Use Proper Coding Techniques
2. Drive Your Synthesis Tool
3. Specify Pin Constraints
4. Apply Global Xilinx Constraints
5. Reasonable Performance Objectives
6. Increase Place & Route Effort Level
7. Apply Multi-Cycle and False Path Constraints
8. Review Critical Paths in the Code
9. Apply Critical Path Constraints in Synthesis
10. Implement with MAP-Timing Options or Run Multi-Pass Place & Route
11. Floorplan

- Implement
  - Done
  - Yes
  - No
  - All Multi-Cycle and False Paths not identified
  - Floorplan hindered timing
- Start Over/Rewrite Code
Download

- Once a design is implemented, you must create a file that the FPGA can understand
  - This file is called a bitstream: a BIT file (.bit extension)
- The BIT file can be downloaded directly into the FPGA, or the BIT file can be converted into a PROM file, which stores the programming information
JTAG and Boundary Scan Technology

- In the 1980s, the Joint Test Action Group (JTAG) developed a specification for boundary-scan testing that was standardized in 1990 as the IEEE Std 1149.1, and later revised in 1993 (titled 1149.1a).

- Boundary-scan architecture
  - Each boundary-scan cell including a multiplexer and latches is assigned to each pin on the device
  - Boundary-scan cells can capture data from pin or core logic signals, or force data onto pins.
    - The captured data is serially shifted out and externally compared to the expected results
    - Forced data is serially shifted into the boundary-scan cells
  - Boundary-scan cells form a serial data path called the scan path or scan chain.
Boundary Scan