ECE 545
Lecture 8
FPGA Devices & FPGA Design Flow

Required Reading
Xilinx, Inc.
Spartan-3 FPGA Family
Spartan-3 FPGA Family Data Sheet
Module 1:
• Introduction
• Features
• Architectural Overview
• Package Marking
Module 2:
• CLB Overview

Two competing implementation approaches
ASIC
Application Specific
Integrated Circuit
• designed all the way from behavioral description to physical layout
• designs must be sent for expensive and time consuming fabrication in semiconductor foundry

FPGA
Field Programmable Gate Array
• no physical layout design; design ends with a bitstream used to configure a device
• bought off the shelf and reconfigured by designers themselves

What is an FPGA?

Which Way to Go?
ASICS
High performance
Low power
Low cost in high volumes

FPGAs
Off-the-shelf
Low development cost
Short time to market
Reconfigurability
Other FPGA Advantages

- Manufacturing cycle for ASIC is very costly, lengthy and engages lots of manpower
- Mistakes not detected at design time have large impact on development time and cost
- FPGAs are perfect for rapid prototyping of digital circuits
- Easy upgrades like in case of software
- Unique applications
  - reconfigurable computing

Major FPGA Vendors

SRAM-based FPGAs
- Xilinx, Inc.
- Altera Corp.
  - Share about 85% of the market
- Atmel
- Lattice Semiconductor

Flash & antifuse FPGAs
- Microsemi SoC Products Group
  (formerly Actel Corp.)
- Quick Logic Corp.

Xilinx

- Primary products: FPGAs and the associated CAD software
- ISE Alliance and Foundation Series Design Software
- Programmable Logic Devices
- Main headquarters in San Jose, CA
- Fabless* Semiconductor and Software Company
  - UMC (Taiwan) (*Xilinx acquired an equity stake in UMC in 1996)
  - Seiko Epson (Japan)
  - TSMC (Taiwan)
  - Samsung (Korea)

Xilinx FPGA Families

- Old families
  - XC3000, XC4000, XC5200
  - Old 0.5µm, 0.35µm and 0.25µm technology. Not recommended for modern designs.
- High-performance families
  - Virtex (220 nm)
  - Virtex-E, Virtex-EM (180 nm)
  - Virtex-2 PRO (130 nm)
  - Virtex-4 (90 nm)
  - Virtex-5 (65 nm)
  - Virtex-6 (40 nm)
- Low Cost Family
  - Spartan-3A – derived from XC4000
  - Spartan-3A – derived from Virtex
  - Spartan-3E – derived from Virtex-E
  - Spartan-3A (90 nm)
  - Spartan-3E (90 nm) – logic optimized
  - Spartan-3A (90 nm) – I/O optimized
  - Spartan-3AN (90 nm) – non-volatile.
  - Spartan-3A DSP (90 nm) – DSP optimized
  - Spartan-6 (40 nm)

CLB Structure
General structure of an FPGA

Xilinx Spartan 3 CLB

CLB Slice = 2 Logic Cells

Xilinx Multipurpose LUT (MLUT)

Spartan 3 CLB Structure

CLB Slice Structure

- Each slice contains two sets of the following:
  - Four-input LUT
    - Any 4-input logic function (16x1 ROM)
    - or 16-bit x 1 sync RAM (SLICEM only)
    - or 16-bit shift register (SLICEM only)
  - Carry & Control
    - Fast arithmetic logic
    - Multiplier logic
    - Multiplexer logic
  - Storage element
    - Latch or flip-flop
    - Set and reset
    - True or inverted inputs
    - Sync. or async. control
LUT (Look-Up Table) Functionality

- Look-Up tables are primary elements for logic implementation
- Each LUT can implement any function of 4 inputs

5-Input Functions implemented using two LUTs

- One CLB Slice can implement any function of 5 inputs
- Logic function is partitioned between two LUTs
- F5 multiplexer selects LUT

MLUT as 16x1 RAM

Copyright © 2004 Mentor Graphics Corp. (www.mentor.com)
Distributed RAM

- CLB LUT configurable as Distributed RAM
- A single LUT equals 16x1 RAM
- Two LUTs implement Single and Dual-Port RAMs
- Cascade LUTs to increase RAM size
- Synchronous write
- Synchronous/Asynchronous read
- Accompanying flip-flops used for synchronous read

MLUT as 16-bit Shift Register (SRL16)

Shift Register

- Each LUT can be configured as shift register
- Serial in, serial out
- Dynamically addressable delay up to 16 cycles
- For programmable pipeline
- Cascade for greater cycle delays
- Use CLB flip-flops to add depth

Using Multipurpose Look-Up Tables in the Shift Register Mode (SRL16)

Inferred from behavioral description in VHDL for shift-registers with:
- one serial input, one serial output
- no reset, no set

Cascading LUT Shift Registers into Shift Registers Longer than 16 bits

- Register-rich FPGA
- Allows for addition of pipeline stages to increase throughput
- Data paths must be balanced to keep desired functionality
**Full-adder**

Alternative implementations

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>c_in</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>c_in</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>c_in</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>c_in</td>
<td>1</td>
</tr>
</tbody>
</table>

Implementation used to generate fast carry logic in Xilinx FPGAs

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>c_in</th>
<th>c_out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>c_in</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>c_in</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>c_in</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ x + y + c_{in} = (c_{out} \text{ s}_2) \]

\[ p = x \oplus y \]
\[ g = y \]
\[ s = p \oplus c_{in} = x \oplus y \oplus c_{in} \]

**Simplified View of Spartan-3 FPGA**

Carry and Arithmetic Logic in One Logic Cell

Hardwired (fast) logic
Simplified View of Carry Logic in One Spartan 3 Slice

Critical Path for an Adder Implemented Using Xilinx Spartan 3/Spartan 3E FPGAs

Number and Length of Carry Chains for Spartan 3 FPGAs

<table>
<thead>
<tr>
<th>Device</th>
<th>Number of Carry Chains</th>
<th>Bits per Column</th>
</tr>
</thead>
<tbody>
<tr>
<td>XC3S500</td>
<td>24</td>
<td>64</td>
</tr>
<tr>
<td>XC3S5000</td>
<td>40</td>
<td>96</td>
</tr>
<tr>
<td>XC3S5000</td>
<td>76</td>
<td>128</td>
</tr>
<tr>
<td>XC3S5000</td>
<td>80</td>
<td>192</td>
</tr>
<tr>
<td>XC3S5150</td>
<td>104</td>
<td>256</td>
</tr>
<tr>
<td>XC3S5200</td>
<td>128</td>
<td>512</td>
</tr>
<tr>
<td>XC3S5400</td>
<td>144</td>
<td>816</td>
</tr>
<tr>
<td>XC3S5700</td>
<td>160</td>
<td>416</td>
</tr>
</tbody>
</table>

Bottom Operand Input to Carry Out Delay $T_{OPCYF}$

0.9 ns for Spartan 3

Carry Propagation Delay $t_{BYP}$

0.2 ns for Spartan 3

$T_{ripple-add} = T_{FA}(x_{i-2}\rightarrow c_{i-1}) + (k-2)\cdot T_{FA}(c_{i-1}\rightarrow c_{i-0}) + T_{FA}(c_{i-0}\rightarrow c_{i})$
Carry Input to Top Sum Combinational Output Delay $T_{CINV}$

1.2 ns for Spartan 3

Critical Path Delays and Maximum Clock Frequencies (into account surrounding registers)

- 8 bits: 3.0 ns or 333 MHz
- 16 bits: 3.8 ns or 263 MHz
- 32 bits: 5.4 ns or 185 MHz
- 64 bits: 8.6 ns or 116 MHz

Fast Carry Logic

- Each CLB contains separate logic and routing for the fast generation of sum & carry signals
- Increases efficiency and performance of adders, subtractors, accumulators, comparators, and counters
- Carry logic is independent of normal logic and routing resources

Accessing Carry Logic

- All major synthesis tools can infer carry logic for arithmetic functions
  - Addition (SUM <= A + B)
  - Subtraction (DIFF <= A - B)
  - Comparators (if A < B then...)
  - Counters (count <= count + 1)

Logic Cell = ½ of a CLB Slice

<table>
<thead>
<tr>
<th>Part of Logic Cell</th>
<th>Mode of Operation</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-purpose Look-up Table (MLUT)</td>
<td>ROM (4-bit LUT)</td>
<td>A piece of combinational logic with 4 inputs and 1 output.</td>
</tr>
<tr>
<td></td>
<td>RAM</td>
<td>16 x 1 single-port RAM</td>
</tr>
<tr>
<td></td>
<td>SR (shift register)</td>
<td>Up to 16-bit shift-register with serial input, serial output, and no reset.</td>
</tr>
<tr>
<td>Carry &amp; Control</td>
<td>FF (Flip-flop)</td>
<td>2 stages of an adder.</td>
</tr>
<tr>
<td>Storage Element</td>
<td>Latch</td>
<td>2-bit latch, with optional set, reset, and enable.</td>
</tr>
</tbody>
</table>

CLB Slice = 2 Logic Cells

<table>
<thead>
<tr>
<th>Part of Logic Cell</th>
<th>Mode of Operation</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-purpose Look-up Table (MLUT)</td>
<td>ROM</td>
<td>Two pieces of combinational logic with 4 inputs and 1 output or 1 piece of combinational logic with 5 inputs and 1 output.</td>
</tr>
<tr>
<td></td>
<td>RAM</td>
<td>16 x 2 single-port RAM or 32 x 1 single-port RAM or 16 x 1 dual-port RAM.</td>
</tr>
<tr>
<td></td>
<td>SR (shift register)</td>
<td>Up to 32-bit shift-register with serial input, serial output, and no reset.</td>
</tr>
<tr>
<td>Carry &amp; Control (C&amp;C)</td>
<td>FF (Flip-flop)</td>
<td>2-bit register, with optional set, reset, and enable.</td>
</tr>
<tr>
<td>Storage Element (SE)</td>
<td>Latch</td>
<td>2-bit latch register, with optional set, reset, and enable.</td>
</tr>
</tbody>
</table>
Examples:

Determine the amount of Spartan 3 resources needed to implement a given circuit.
**Circuit 4: Top level**

- Input
- Previous_input
- Counter
- Output

**Other Components of Spartan 3 FPGAs**

**RAM Blocks and Multipliers in Xilinx FPGAs**

- RAM blocks
- Multipliers
- Logic blocks

**Dedicated Multiplier Block**

- A[17:0]
- RST
- CE
- RSTP

**Combinational and Registered Multiplier**

- A[17:0]
- B[17:0]
- C
- R

**Block RAM**

- Most efficient memory implementation
- Dedicated blocks of memory
- Ideal for most memory requirements
- 4 to 36 memory blocks in Spartan 3
  - 18 kbits = 18,432 bits per block (16 k without parity bits)
- Use multiple blocks for larger memories
- Builds both single and true dual-port RAMs
- Synchronous write and read (different from distributed RAM)
Block RAM can have various configurations (port aspect ratios)

![Block RAM Port Aspect Ratios](image_url)

<table>
<thead>
<tr>
<th>Block RAM Port Aspect Ratios</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>Single-Port Block RAM</td>
</tr>
<tr>
<td>WE</td>
</tr>
<tr>
<td>Dual-Port Block RAM</td>
</tr>
<tr>
<td>D[1:0]</td>
</tr>
<tr>
<td>Basic I/O Block Structure</td>
</tr>
<tr>
<td>Input/Output Blocks (IOBs)</td>
</tr>
</tbody>
</table>
IOB Functionality

- IOB provides interface between the package pins and CLBs
- Each IOB can work as uni- or bi-directional I/O
- Outputs can be forced into High Impedance
- Inputs and outputs can be registered
  - advised for high-performance I/O
- Inputs can be delayed
Design and implement a simple unit permitting to speed up encryption with RC5-similar cipher with fixed key set on 8031 microcontroller. Unlike in the experiment 5, this time your unit has to be able to perform an encryption algorithm by itself, executing 32 rounds...

Library
IEEE;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity RC5_core is
port (clock, reset, encr_decr: in std_logic;
      data_input: in std_logic_vector(31 downto 0);
      data_output: out std_logic_vector(31 downto 0);
      out_full: in std_logic;
      key_input: in std_logic_vector(31 downto 0);
      key_read: out std_logic);
end AES_core;

Specification / Pseudocode
VHDL description (Your Source Files)
Functional simulation
Post-synthesis simulation

Tools used in FPGA Design Flow
Functionally verified VHDL code
Xilinx XST
Synplify Premier
Xilinx IBE
Synthesis
Design
VHDL code
Netlist
Implementation
Bitstream

Synthesis Tools
Xilinx XST
Synplify Premier
... and others

Logic Synthesis
VHDL description
Circuit netlist
Circuit netlist (RTL view)

Mapping

RTL view in Synplify Premier

- General logic structures can be recognized in RTL view

Technology View in Synplify Pro

- Technology view is a mapped RTL view. It can be seen by pressing 'view' button or
- right-clicking on 'view' button

Crossprobing between RTL view and code

- Each port, net, or SM can be chosen by mouse click from the browser or directly from the RTL view.
- If you double-click on the element, its source code can be seen.

Viewing critical path

- Critical path can be viewed by pressing 'critical path' button.
- Delay values are written on each component of the path.
Implementation

- After synthesis the entire implementation process is performed by FPGA vendor tools.
Placing CLB SLICES

Routing Programmable Connections

Configuration

- Once a design is implemented, you must create a file that the FPGA can understand
- This file is called a bit stream: a BIT file (.bit extension)

- The BIT file can be downloaded directly to the FPGA, or can be converted into a PROM file which stores the programming information

Map report header

Release 8.1/Map 1.24
Xilinx Mapping Report File for Design 'Lab3Demo'

Design Information
Command Line : c:\Xilinx\bin\map.exe -p 3S1500FG320-4 -o map.ncd -pr b -i 4 -cm area -c 100 Lab3Demo.ncd Lab3Demo.pcf
Target Device : xc3s1500
Target Package : fg320
Target Speed : -4
Mapset Version : spartan3 -- $Revision: 1.34 $
Mapped Date : Tue Feb 13 17:04:54 2007
Map report

Design Summary
Number of errors: 0
Number of warnings: 0
Logic Utilization:
Number of Slice Flip Flops: 30 out of 26,624 (1%)
Number of 4 input LUTs: 38 out of 26,624 (1%)
Logic Distribution:
Number of occupied Slices: 33 out of 13,312 (1%)
Number of Slices containing only related logic: 33 out of 33 (100%)
Number of Slices containing unrelated logic: 0 out of 33 (0%)
See NOTES below for an explanation of the effects of unrelated logic
Total Number 4 input LUTs: 62 out of 26,624 (1%)
Number used as logic: 38
Number used as a route-thru: 24
Number of bonded IOBs: 10 out of 221 (4%)
IOB Flip Flops: 7
Number of GCLKs: 1 out of 8 (12%)

Related and Unrelated Logic
Related logic is defined as being logic that shares connectivity – e.g. two LUTs are “related” if they share common inputs.
When assembling slices, Map gives priority to combine logic that is related. Doing so results in the best timing performance.
Unrelated logic shares no connectivity. Map will only begin packing unrelated logic into a slice once 99% of the slices are occupied through related logic packing.
Note that once logic distribution reaches the 99% level through related logic packing, this does not mean the device is completely utilized. Unrelated logic packing will then begin, continuing until all usable LUTs and FFs are occupied.
Depending on your timing budget, increased levels of unrelated logic packing may adversely affect the overall timing performance of your design.

Place & route report

Asterisk (*) preceding a constraint indicates it was not met.
This may be due to a setup or hold violation.

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Requested</th>
<th>Actual</th>
<th>Logic Levels</th>
<th>Absolute Slack</th>
<th>Number of errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS_CLOCK = PERIOD TIMEGRP “CLOCK” 5 ns</td>
<td>5.000ns</td>
<td>4.137ns</td>
<td>2</td>
<td>0.863ns</td>
<td>0</td>
</tr>
<tr>
<td>TS_gen1Hz_Clock1Hz = PERIOD TIMEGRP “gen1Hz_Clock1Hz” 5 ns HIGH 50%</td>
<td>5.000ns</td>
<td>4.140ns</td>
<td>2</td>
<td>0.860ns</td>
<td>0</td>
</tr>
</tbody>
</table>

Timing summary:
Timing errors: 9  Score: 543
Constraints cover 574 paths, 0 nets, and 187 connections
Design statistics:
Minimum period: 5.140ns (Maximum frequency: 194.553MHz)

Post layout timing report

Clock to Setup on destination clock CLOCK

<table>
<thead>
<tr>
<th>Src:Rise</th>
<th>Src:Fall</th>
<th>Dest:Rise</th>
<th>Dest:Fall</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.140ns</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Xilinx FPGA Devices

Technology | Low-cost | High-performance
---|---------|-------------------
120/150 nm | Virtex 2, 2 Pro |
90 nm | Spartan 3 | Virtex 4 |
65 nm | Virtex 5 |
45 nm | Spartan 6 |
40 nm | Virtex 6 |

Altera FPGA Devices

<table>
<thead>
<tr>
<th>Technology</th>
<th>Low-cost</th>
<th>Mid-range</th>
<th>High-performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>130 nm</td>
<td>Cyclone</td>
<td>Stratix</td>
<td></td>
</tr>
<tr>
<td>90 nm</td>
<td>Cyclone II</td>
<td>Stratix II</td>
<td></td>
</tr>
<tr>
<td>65 nm</td>
<td>Cyclone III</td>
<td>Arria I</td>
<td>Stratix III</td>
</tr>
<tr>
<td>40 nm</td>
<td>Cyclone IV</td>
<td>Arria II</td>
<td>Stratix IV</td>
</tr>
</tbody>
</table>