L12: Reconfigurable Logic Architectures

Acknowledgements:


- Frank Honore

- Lecture Notes prepared by Professor Anantha Chandrakasan
History of Computational Fabrics

- **Discrete devices**: relays, transistors (1940s-50s)
- **Discrete logic gates** (1950s-60s)
- **Integrated circuits** (1960s-70s)
  - e.g. TTL packages: Data Book for 100’s of different parts
- **Gate Arrays (IBM 1970s)**
  - Transistors are pre-placed on the chip & Place and Route software puts the chip together automatically – only program the interconnect (mask programming)
- **Software Based Schemes (1970’s- present)**
  - Run instructions on a general purpose core
- **Programmable Logic (1980’s to present)**
  - A chip that be reprogrammed after it has been fabricated
  - Examples: PALs, EPROM, EEPROM, PLDs, FPGAs
  - Excellent support for mapping from Verilog
- **ASIC Design (1980’s to present)**
  - Turn Verilog directly into layout using a library of standard cells
  - Effective for high-volume and efficient use of silicon area
Reconfigurable Logic

- Logic blocks
  - To implement combinational and sequential logic

- Interconnect
  - Wires to connect inputs and outputs to logic blocks

- I/O blocks
  - Special logic blocks at periphery of device for external connections

Key questions:
- How to make logic blocks programmable? (after chip has been fabbed!)
- What should the logic granularity be?
- How to make the wires programmable? (after chip has been fabbed!)
- Specialized wiring structures for local vs. long distance routes?
- How many wires per logic block?
Programmable Array Logic (PAL)

- Based on the fact that any combinational logic can be realized as a sum-of-products.
- PALs feature an array of AND-OR gates with programmable interconnect.

![Diagram of PAL structure](image)
Each input pin (and its complement) sent to the AND array
OR gates for each output can take 8-16 product terms, depending on output pin
“Macrocell” block provides additional output flexibility...
From Cypress

- Combinational/active low
- Combinational/active high

From Lattice Semiconductor

- Outputs may be registered or combinational, positive or inverted

<table>
<thead>
<tr>
<th>$S_1$</th>
<th>$S_0$</th>
<th>Output Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Registered/active low</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Registered/active high</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Combinational/active low</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Combinational/active high</td>
</tr>
</tbody>
</table>

0 = Programmed EE bit
1 = Erased (charged) EE bit
RAM Based Field Programmable Logic - Xilinx

Programmable Interconnect

Configurable Logic Blocks (CLBs)

I/O Blocks (IOBs)
The Xilinx 4000 CLB

Simplified Block Diagram of XC4000 Series CLB (RAM and Carry Logic functions not shown)
Two 4-input Functions, Registered Output and a Two Input Function

Simplified Block Diagram of XC4000 Series CLB (RAM and Carry Logic functions not shown)
5-input Function, Combinational Output

Simplified Block Diagram of XC4000 Series CLB (RAM and Carry Logic functions not shown)
N-LUT direct implementation of a truth table: any function of n-inputs.

N-LUT requires $2^N$ storage elements (latches)

N-inputs select one latch location (like a memory)

Why Latches and Not Registers?

Latches set by configuration bitstream

4LUT example
Configuring the CLB as a RAM

Memory is built using Latches not FFs

Read is same a LUT Function!

16x2
Xilinx 4000 Interconnect

Single- and Double-Length Lines, with Programmable Switch Matrices (PSMs)
Wires are not ideal!
Xilinx 4000 Flexible IOB

Adjust Transition Time

Outputs through FF or bypassed

Adjust the Sampling Edge
Add Bells & Whistles

Gigabit Serial

Hard Processor

Multiplier

I/O

Programmable Termination

Clock Mgmt

BRAM

The Virtex II CLB (Half Slice Shown)
Adder Implementation

LUT: $A \oplus B$

$Y = A \oplus B \oplus Cin$

Dedicated carry logic

1 half-Slice = 1-bit adder
1 CLB = 4 Slices = 2, 4-bit adders

64-bit Adder: 16 CLBs


A[3:0] → CLB0 → Y[3:0]

CLBs must be in same column

(First Carry Chain)

(Second Carry Chain)
Virtex II Features

Double Data Rate registers

Digital Clock Manager

Embedded Multiplier

Block SelectRAM
The Latest Generation: Virtex-II Pro

- High-speed I/O
- Embedded PowerPC
- Embedded memories
- Hardwired multipliers
- FPGA Fabric
- Courtesy Xilinx
Altera’s New Stratix Architecture

Up to 11,310 LE’s, 10Mbits RAM
10 LE’s per LAB

Embedded DSP feature: 9x9, 18x18, 36x36 with 52-bit accumulator
Technology Mapping: Schematic/HDL to Physical Logic units

Compile functions into basic LUT-based groups (function of target architecture)

always @(posedge Clock or negedge Reset)
begin
    if (! Reset)
        q <= 0;
    else
        q <= (a & b & c) | (b & d);
end
Design Flow – Placement & Route

- **Placement** – assign logic location on a particular device

- **Routing** – iterative process to connect CLB inputs/outputs and IOBs. Optimizes critical path delay – can take hours or days for large, dense designs

Challenge! Cannot use full chip for reasonable speeds (wires are not ideal).

Typically no more than 50% utilization.
module adder64 (a, b, sum);
  input [63:0] a, b;
  output [63:0] sum;
  assign sum = a + b;
endmodule

64-bit Adder Example
How are FPGAs Used?

- **Prototyping**
  - Ensemble of gate arrays used to emulate a circuit to be manufactured
  - Get more/better/faster debugging done than with simulation

- **Reconfigurable hardware**
  - One hardware block used to implement more than one function

- **Special-purpose computation engines**
  - Hardware dedicated to solving one problem (or class of problems)
  - Accelerators attached to general-purpose computers (e.g., in a cell phone!)

**Logic Emulation**

FPGA-based Emulator
(courtesy of IKOS)
FPGA provide a flexible platform for implementing digital computing

A rich set of macros and I/Os supported (multipliers, block RAMS, ROMS, high-speed I/O)

A wide range of applications from prototyping (to validate a design before ASIC mapping) to high-performance spatial computing

Interconnects are a major bottleneck (physical design and locality are important considerations)

“College students will study concurrent programming instead of “C” as their first computing experience.”

-- David B. Parlour, ISSCC 2004 Tutorial