FPGA: What? Why?

Politecnico di Milano
NECST Meeting Room
9 November, 2016

Antonio R. Miele
Marco D. Santambrogio
Politecnico di Milano
Reconfiguration

The process of physically altering the location or functionality of network or system elements. Automatic configuration describes the way sophisticated networks can readjust themselves in the event of a link or device failing, enabling the network to continue operation.

Gerald Estrin, 1960
Outline

• A bird’s eye view on the Reconfigurable Computing
• Xilinx FPGA
• The roadmap
Reconfigurable computing is defined as the study of computation using reconfigurable devices.

Christophe Bobda, 2007
Reconfigurable Hardware

“Reconfigurable computing is intended to fill the gap between hardware and software, achieving potentially much higher performance than software, while maintaining a higher level of flexibility than hardware”

Evolution of implementation technologies

• Logic gates (1950s-60s)
• Regular structures for two-level logic (1960s-70s)
  – muxes and decoders, PLAs
• Programmable sum-of-products arrays (1970s-80s)
  – PLDs, complex PLDs
• Programmable gate arrays (1980s-90s)
  – densities high enough to permit entirely new class of application, e.g., prototyping, emulation, acceleration
Gate Array Technology (IBM - 1970s)

- Simple logic gates
  - combine transistors to implement combinational and sequential logic
- Interconnect
  - wires to connect inputs and outputs to logic blocks
- I/O blocks
  - special blocks at periphery for external connections
- Add wires to make connections
  - done when chip is fabbed
    - “mask-programmable”
  - construct any circuit
Field-Programmable Gate Arrays

- 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?
- how to connect the wires?
- after the chip has been fabbed
The Academic Efforts

- The Reconfigurable Architecture Workstation (RAW) - MIT
- The Matrix Architecture - MIT
- The Reconfigurable Multimedia Array Coprocessor (REMARC) - Stanford
- MorphoSys - University of California, Irvine
- Chimaera – Northwestern
- PipeRench - CMU
- RaPiD - University of Washington
- Garp – UC Berkeley
- Bee2- UC Berkeley
5 W

- **who** controls the reconfiguration
- **where** the reconfiguration controller is located
- **when** the configurations are generated
- **which** is the granularity of the reconfiguration
- **what** dimension the reconfiguration operates
Reconfigurable Architectures Characterization

• SoC (System on Chip)
  – Embedded Vs External
  – Complete Vs Partial
  – Dynamic VS Static

• SoMC (System on Multiple-Chip)
  – Embedded Vs External
  – Complete Vs Partial
  – Dynamic VS Static
What’s next

• A bird’s eye view on the Reconfigurable Computing

• Xilinx FPGA
  – Technology
  – CLB, Slice, LUT
  – Frame
  – Configuration bitstream

• The roadmap
Commercial FPGA Companies

% of Total Market

Lattice

FPGA

Actel

$2.9B

$0.6B

Xilinx

Lattice

Altera

PLD

Xilinx

Lattice official website
FPGA market share @ 2013

Preferred FPGA Vendor's Dev's

- Xilinx: 66%
- N/A: 17%
- Altera: 16%
- Other: 1%

Source: FPMM Surveys (N=4702)
Xilinx Programmable Gate Arrays

- CLB - Configurable Logic Block
- Built-in fast carry logic
- Can be used as memory
- Three types of routing
  - direct
  - general-purpose
  - long lines of various lengths
- RAM-programmable
  - can be reconfigured
Configurable Logic Blocks

• CLBs made of Slices
  – sVirtex-E 2-slice
  – VIIP 4-slice

• Slices made of LookUp Tables (LUTs)

• LookUp Tables
  – 4-input, 1 output functions
  – Newest FPGA 2 6-input 2 output
Simplified CLB Structure
Lookup Tables: LUTs

- LUT contains Memory Cells to implement small logic functions
- Each cell holds ‘0’ or ‘1’.
- Programmed with outputs of Truth Table
- Inputs select content of one of the cells as output

3 Inputs LUT -> 8 Memory Cells

3 – 6 Inputs

Multiplexer MUX

Static Random Access Memory SRAM cells
Example: 4-input AND gate

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
The Virtex CLB

2-Slice Virtex-E CLB
Details of One Virtex Slice

Virtex-E Slice
Implements any Two 4-input Functions

Virtex-E Slice
CLB

Switch Box

4-Slice VIIP CLB

SLICE

TBUF

SLICE_X66Y74
Interconnection Network

- Configurable Logic Blocks
- Interconnection Network
- I/O Signals (Pins)
Example

- Determine the configuration bits for the following circuit implementation in a 2x2 FPGA, with I/O constraints as shown in the following figure. Assume 2-input LUTs in each CLB.
CLBs configuration
Placement: Select CLBs
Routing: Select path

Input1

CLB0

SB0

CLB1

SB1

SB2

SB3

Input2

CLB2

SB4

CLB3

Input3

Output

SB1

Configuration bits

0 0 0 0

SB4

Configuration bits

0 1 0 0

0 0 0 0
Configuration Bitstream

- The configuration bitstream must include ALL CLBs and SBs, even unused ones
  - CLB0: 00011
  - CLB1: ?????
  - CLB2: 01100
  - CLB3: XXXXX
  - SB0: 000000
  - SB1: 000010
  - SB2: 000000
  - SB3: 000000
  - SB4: 000001
The configuration bitstream

- Occupation must be determined only on the basis of
  - Number of configuration words
  - Initial Frame Address Register (FAR) value
Frame and Configuration Memory

- **Virtex-II Pro**
  - Configuration memory is arranged in vertical frames that are one bit wide and stretch from the top edge of the device to the bottom.
  - Frames are the smallest addressable segments of the VIIP configuration memory space.
    - All operations must act on whole configuration frames.

- **Virtex-4**
  - Configuration memory is arranged in frames that are tiled about the device.
  - Frames are the smallest addressable segments of the V4 configuration memory space.
    - All operations must therefore act upon whole configuration frames.
Xilinx Virtex-4: frame organization
Logistic

• Friday 20 Nov 2015
  – @9.00 am in Seminar Room
    A. Alario with Santa
  – @11.00am in Seminar Room
    A. Alario with A. Miele
Some Definitions

- **Object Code**: the executable active physical (either HW or SW) implementation of a given functionality

- **Core**: a specific representation of a functionality. It is possible, for example, to have a core described in VHDL, in C or in an intermediate representation (e.g. a DFG)

- **IP-Core**: a core described using a HD Language combined with its communication infrastructure (i.e. the bus interface)

- **Reconfigurable Functional Unit**: an IP-Core that can be plugged and/or unplugged at runtime in an already working architecture

- **Reconfigurable Region**: a portion of the device area used to implement a reconfigurable core
Reconfigurable Region Definition

The flows require constraints to be satisfied when defining RRs in the UCF (User Constraints File) file

 AREA_GROUP "RR1" RANGE = SLICE_X28Y64:SLICE_X41Y127;
 AREA_GROUP "RR1" RANGE = RAMB16_X2Y9:RAMB16_X2Y15;
Area Constraints

Xilinx VIIP

Xilinx S3

Xilinx V4
Xilinx FPGA and Configuration Memory

- Bitstream files
- ICAP
- SelectMAP
- Reconfiguration Controllers
FPGA EDA Tools

- Must provide a design environment based on digital design concepts and components (gates, flip-flops, MUXs, etc.)
- Must hide the complexities of placement, routing and bitstream generation from the user. Manual placement, routing and bitstream generation is infeasible for practical FPGA array sizes and circuit complexities.
Computer-aided Design

• Can't design FPGAs by hand
  – way too much logic to manage, hard to make changes
• Hardware description languages
  – specify functionality of logic at a high level
• Validation - high-level simulation to catch specification errors
  – verify pin-outs and connections to other system components
  – low-level to verify mapping and check performance
• Logic synthesis
  – process of compiling HDL program into logic gates and flip-flops
• Technology mapping
  – map the logic onto elements available in the implementation technology (LUTs for Xilinx FPGAs)
CAD Tool Path (cont’d)

• Placement and routing
  – assign logic blocks to functions
  – make wiring connections

• Timing analysis - verify paths
  – determine delays as routed
  – look at critical paths and ways to improve

• Partitioning and constraining
  – if design does not fit or is unroutable as placed split into multiple chips
  – if design it too slow prioritize critical paths, fix placement of cells, etc.
  – few tools to help with these tasks exist today

• Generate programming files - bits to be loaded into chip for configuration
What’s next

• A bird’s eye view on the Reconfigurable Computing
• Xilinx FPGA

• The roadmap
  – The 90% – 10% Rule
  – Programmable System on a Chip
  – Multi-FPGA
  – Complex heterogeneous adaptive systems
The 90% – 10% Rule

• 90% of the execution is spent in **10% of the code**
  – Inner loops in algorithms
  – Computational intense code

• 10% of the execution is spent in **90% of the code**
  – Exceptions
  – User interaction

• The **10% computational intense code** has to be executed as hardware on **reconfigurable devices**

• The **90% exception code** is run as executable files on **processors**
Programmable System on a Chip

• No longer just a bunch of reconfigurable elements
• DSPs, GPP, reconfigurable elements, etc. etc...
Heterogeneous Multi-FPGA system
Intel Stellarton

- Heterogeneous Multicore
  - An Intel Atom E6XX processor
    - # Cores: 1
    - # Threads: 2
    - L2 Cache: 512 KB
  - An Altera Field Programmable Gate Array
Complex Heterogeneous Systems

• Due to the complexity in the demand, the system has to be heterogeneous and able to autonomously adapt and evolve
  – FPGAs
  – DSPs
  – GPP (Multi-cores)

• Adaptive systems learn how they can be used to address a particular problem
  – Respond to user goals
  – Build self-performance models
  – Identify what they needs to learn
  – Adapt to changing goals, resources, models, operating conditions
  – Gracefully adapt to failures
  – Optimize their own behavior
Heterogeneous Complex Systems

• Ryft ONE
  – Big Data infrastructure due to an FPGA-accelerated architecture

• IBM Power8
  – Introducing the Coherent Accelerator Processor Interface (CAPI) port that is layered on top of PCI Express 3.0

• Microsoft Catapult
  – Stratix V (Arria 10 FPGA)

• OpenPower Foundation
Questions...

FPGA: What? Why?

Politecnico di Milano
NECST Meeting Room
3 November, 2016

Antonio R. Miele
Marco D. Santambrogio
Politecnico di Milano