Development of tools supporting FPGA reconfigurable hardware

MEANDER
Design Framework
Presentation Outline

- Current state of academic design tools
- Proposed design flow
- Proposed graphical user interface
- Comparison results
- Conclusions
Motivation

- Lack of complete academic design flow
  - (RTL- VHDL → Bit stream)
- Lack of “open source code” synthesizer
- Lack of “open source code” tool for bit-stream generation
- Existing “incomplete” design flows are in text mode (no GUI)
- Lack of manuals for the most of the existing tools
- Most of the tools are designed for different operating systems (SUN OS, Linux, BSD, …)
Design Flow Development

- Design flow based on open-source code CAD tools
  - Linux Operating System
  - C/C++ language
  - Input format: RT VHDL, Structural VHDL, EDIF, BLIF
  - Output: Configuration Stream
  - Technology Independent
  - Portability (e.g. x486, SPARC)
  - Run on a local machine or through the Internet/Intranet
  - Modularity: each tool can run as a standalone tool
  - Graphical User Interface (GUI)
  - Performance, Area and Power Consumption
  - Minimum requirements: x486, 64 MB RAM, 30 MB HD
Design Flow

Circuit

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Generation of BLEs/Cluster (T-VPack)

Architecture (DUTYS)

Placement/Routing (EX-VPR)

FPGA Configuration (DAGGER)

Legend

- Existing tool
- Modified & Extended
- New tool
Design Flow

Circuit

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Architecture (DUTYS)

Placement/Routing (EX-VPR)

Generation of BLEs/Cluster (T-VPack)

FPGA Configuration (DAGGER)

Legend
- Existing tool
- Modified & Extended
- New tool

Circuit description in VHDL
Design Flow

Circuit

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Legend

- Green: Existing tool
- Orange: Modified & Extended
- Maroon: New tool

Architecture (DUTYS)

Generation of BLEs/Cluster (T-VPack)

Placement/Routing (EX-VPR)

FPGA Configuration (DAGGER)

Performs syntax checking and simulation
Design Flow

Circuit

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

PowerModel

Generation of BLEs/Cluster (T-VPack)

Placement/Routing (EX-VPR)

Architecture (DUTYS)

FPGA Configuration (DAGGER)

Synthesizes the VHDL circuit to EDIF format

Legend
- Existing tool
- Modified & Extended
- New tool
Design Flow

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Generation of BLEs/Cluster (T-VPack)

Placement/Routing (EX-VPR)

FPGA Configuration (DAGGER)

Legend:
- **Existing tool**
- **Modified & Extended**
- **New tool**

Modifies the EDIF file in order to be compatible to our design flow.
Design Flow

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Translation of EDIF to BLIF format

Architecture (DUTYS)

Generation of BLEs/Cluster (T-VPack)

Placement/Routing (EX-VPR)

FPGA Configuration (DAGGER)

Legend
- Existing tool
- Modified & Extended
- New tool
Design Flow

Syntax check (VHDL Parser) & Simulation (FreeVHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Generation of BLEs/Cluster (T-VPack)

Architecture (DUTYS)

Placement/Routing (EX-VPR)

FPGA Configuration (DAGGER)

Legend:
- Green: Existing tool
- Yellow: Modified & Extended
- Pink: New tool

Performs logic optimization and technology mapping to LUTs & F/Fs
Design Flow

Circuit

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Generation of BLEs/Cluster (T-VPack)

Architecture (DUTYS)

Placement/Routing (EX-VPR)

FPGA Configuration (DAGGER)

Legend
- Existing tool
- Modified & Extended
- New tool

Packs LUTs/FFs into BLEs and Clusters
Design Flow

Circuit

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Architecture (DUTYS)

Generation of BLEs/Cluster (T-VPack)

Placement/Routing (EX-VPR)

FPGA Configuration (DAGGER)

Describes the FPGA architecture

Legend
- Existing tool
- Modified & Extended
- New tool
Design Flow

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Generation of BLEs/Cluster (T-VPack)

Placement/Routing (EX-VPR)

FPGA Configuration (DAGGER)

Legend:
- Existing tool
- Modified & Extended
- New tool

Performs placement and routing. Also extracts power estimation results.
Design Flow

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Generation of BLEs/Cluster (T-VPack)

Architecture (DUTYS)

Placement/Routing (EX-VPR)

FPGA Configuration (DAGGER)

Legend
- Existing tool
- Modified & Extended
- New tool

Performs circuit power estimation
Design Flow

Circuit

Syntax check (VHDL Parser) & Simulation (FreeHDL)

Synthesis (DIVINER/LEONARDO)

Modification of .edif files (DRUID)

Translation to .blif (E2FMT)

Logic Optimization & Technology Mapping (SIS)

Architecture (DUTYS)

Generation of BLEs/Cluster (T-VPack)

Placement/Routing (EX-VPR)

FPGA Configuration (DAGGER)

Configures the FPGA

Legend
- Existing tool
- Modified & Extended
- New tool
VHDL Parser

- JAVA based tool
- Checks VHDL file for syntax errors
- In case of error, a message locating it is displayed
- In addition there are some suggestions to correct it
- A manual for the tool has been written
FreeHDL

- Is a VHDL simulation tool running on Linux
- It is part from the FreeHDL Project
- Is capable to show in graphic mode the signal transactions
- Supports the VHDL-93 standard
- Is similar to many commercial tools, like V-System
- A manual for the tool has been written
Goal: Synthesizes circuit descriptions from VHDL to EDIF

There is no other available academic synthesizer

Current Status: Supports a subset of VHDL-described components

⇒ The circuits should consist of logic and F/Fs

A manual for the tool is prepared
Goal: Modification of EDIF file to another EDIF file compatible with the remaining tools of the design flow

Automatically generates basic modules not supported by E2FMT libraries (arithmetic units, MUXs, decoders, comparators) in gate level.

Renames component instances, signal and port names and libraries accordingly.

No other existing academic tool makes those modifications.

Tool manual prepared.
(instance modgen_add_0 (viewRef INTERFACE (cellRef add_zu_zu_zu_0 (libraryRef OPERATORS ))))

(instance modgen_add_0 (viewRef rtl (cellRef adder_gen_z (libraryRef USER_LIB ))))

(port (array (rename output3 "output3(2:0)") 3 ) (direction OUTPUT))

(port output3_0 (direction OUTPUT))
(port output3_1 (direction OUTPUT))
(port output3_2 (direction OUTPUT))

(portRef (member output3 x) (instanceRef reg2 ))
(portRef output3_x (instanceRef reg2 ))
DRUID OPERATION (2/2)

(cell add_zu_zu_zu_0 (cellType GENERIC)
    (property (rename a0 "\$GENERIC") (string "add"))
    (property (rename a1 "$size") (string "z"))
    (property (rename a2 "$signed") (string "false"))
    (view INTERFACE (viewType NETLIST)
        (interface
            (port cin (direction INPUT))
            (port (array (rename a "a(z-1:0)") z )(direction INPUT))
            (port (array (rename b "b(z-1:0)") z )(direction INPUT))
            (port (array (rename d "d(z-1:0)") z )(direction OUTPUT))
            (port cout (direction OUTPUT)))

(cell adder_gen_z (cellType GENERIC)
    (view rtl_adder (viewType NETLIST)
        (interface
            for(i=0; i=i+1; i=z-1)
            { (port a_i (direction INPUT)) }
            ...
            (contents
                (instance inv_i0_addz (viewRef INTERFACE (cellRef INV (libraryRef PRIMITIVES ))))
            ...
            for(i=0; i=i+1; i="z-1")
            { (net b_i_adder_gen_z (joined
                (portRef b_i "z-i-1")
                (portRef in1 (instanceRef and2_i_adderz_1 )) )
            ...

MEANDER Design Framework – VLSI Design and Testing Center – Democritus University of Thrace

Goal: Translation of netlist from EDIF to BLIF format

Features:
- BLIF format is required from the existing academic tools
- Better than existing EDIF2BLIF tool, since it can handle more complex EDIF files
- The translation is based on the IWLS’93 library
- The technology library has been extended in order to support more complex components
- The source code was modified to embody PowerModel tool within the design flow
- Tool manual prepared
T-VPack

- Groups the lookup-table and the F/F to BLE or to Cluster

![Diagram](image)

- Advantages:
  - The size of the LUT can be specified
  - It is easy to modify the source code, in order to handle more complex structures

- Disadvantage:
  - Any modification to BLE types must be compatible with the placement and routing tool

- We determined the way that this tool could handle user-specified structures of BLEs

- Tool manual prepared
DUTYS

- **Goal:** Generates the architecture file of the target fine-grain reconfigurable hardware to be used by VPR tool

- **Features:**
  - Description of I/O pads
  - Description of Relative Channel Widths
  - Logic Block Description
  - Detailed Routing Architecture Description: switch box, connection box, segment
  - Description of timing analysis parameters

- Tool manual prepared
Goal: Places and routes the circuit

Combined with PowerModel, it extracts information about the power consumption of the circuit

The source code has been modified in order to support extra features

A model that estimates device area in $mm^2$ was added for 0.18μm STM

Tool manual prepared
Power Model

- It estimates the dynamic, the short-circuit, and the leakage power consumption of an island-style FPGA
- It is integrated to the proposed design flow, after porting from Solaris-OS to Linux
- Tool manual prepared
DAGGER

- It is the first academic “open source” bit-stream generation tool
  - Input from: DUTYS, T-VPack, VPR, and PowerModel
  - Novel configuration algorithm was developed

- The DAGGER tool can support partial reconfiguration (if hardware supports it)
- It can be used with the same way for programming BLEs or clusters
- It can be used for a variety of FPGA architectures
- Supports FPGA arrays with large size
- The bit-stream is in encrypted binary format
- Tool manual prepared
Configuration Bitstream Generator: Structure of DAGGER Files

Array A

Admissible connection point to routing channel
Admissible connection point to switch box
Prohibited connection point

Intermediate BLE
Example of DAGGER output

LUT PROGRAMMING

| Input pad: p_1gat_0 | pos_x = 0 | pos_y = 5 |
| Input pad: p_6gat_3 | pos_x = 6 | pos_y = 4 |
| Input pad: p_7gat_4 | pos_x = 6 | pos_y = 5 |
| Input pad: p_2gat_1 | pos_x = 0 | pos_y = 3 |
| Input pad: p_3gat_2 | pos_x = 4 | pos_y = 6 |
| Output pad: out:p_2 | pos_x = 4 | pos_y = 0 |
| Output pad: out:p_2 | pos_x = 4 | pos_y = 6 |

CLB: n_n14 | pos_x = 4 | pos_y = 5 | LOGIC = 0000000011111111 | use f/f = 0
CLB: n_n5 | pos_x = 4 | pos_y = 4 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n13 | pos_x = 5 | pos_y = 4 | LOGIC = 0000000011111111 | use f/f = 0
CLB: n_n9 | pos_x = 4 | pos_y = 3 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n2 | pos_x = 2 | pos_y = 3 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n6 | pos_x = 2 | pos_y = 2 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n1 | pos_x = 3 | pos_y = 2 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n3 | pos_x = 5 | pos_y = 3 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n10 | pos_x = 3 | pos_y = 1 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n7 | pos_x = 4 | pos_y = 2 | LOGIC = 0000000000001111 | use f/f = 0
CLB: p_23gat_9 | pos_x = 4 | pos_y = 1 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n0 | pos_x = 3 | pos_y = 4 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n11 | pos_x = 3 | pos_y = 4 | LOGIC = 0000000000001111 | use f/f = 0
CLB: p_22gat_10 | pos_x = 3 | pos_y = 5 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n15 | pos_x = 1 | pos_y = 3 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n12 | pos_x = 5 | pos_y = 5 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n8 | pos_x = 2 | pos_y = 4 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n4 | pos_x = 2 | pos_y = 4 | LOGIC = 0000000000001111 | use f/f = 0
CLB: n_n16 | pos_x = 1 | pos_y = 5 | LOGIC = 0000000000001111 | use f/f = 0

Intermediate BLE
Example of placement and routing (EX-VPR)

ISCAS Benchmark: C17

- Initial placement of logic blocks
- Final placement of logic blocks
- Circuit after the finish of routing stage
- Zoom-in to the architecture of an FPGA
- Circuit fully routed
Graphical User Interface
Graphical User Interface

- It integrates most of the tools that are described previously
- There is no other academic implementation of such a complete graphical design chain
- Advantages of GUI:
  - The GUI is friendly for the non-experienced designer
  - The end-user does not need to know Linux
  - It is possible to run it from local PC or through Internet / Intranet
  - The source code can be modified (extended) in order to add more tools
- This interface can run from the local PC as well as though the Internet. In both situations it runs on the web-browser
- Can program the FPGA that is attached to the user’s PC
- It is not needed to use all the tools of the design chain
- Manual prepared
# Qualitative comparisons

<table>
<thead>
<tr>
<th>FEATURE</th>
<th>AMDREEL</th>
<th>XILINX</th>
<th>Univ. TORONTO</th>
<th>ALLIANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Input Format</td>
<td>VHDL/VERILOG</td>
<td>VHDL/VERILOG</td>
<td>BLIF</td>
<td>VHDL</td>
</tr>
<tr>
<td>Synthesizer</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Format Translation</td>
<td>✓</td>
<td>-</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Architecture Description</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Place &amp; Route</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Bitstream Generation</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Back annotation</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Power Estimation</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Area Estimation</td>
<td>✓</td>
<td>-</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>GUI</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Remote Access to GUI</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>User Manual</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>O/S</td>
<td>LINUX</td>
<td>SOLARIS/ WINDOWS/ LINUX</td>
<td>SOLARIS</td>
<td>LINUX</td>
</tr>
</tbody>
</table>

- ✓: supported
- ✗: not supported
- -: not required

MEANDER Design Framework – VLSI Design and Testing Center – Democritus University of Thrace
Presentation Outline

- Current state of academic design tools
- Proposed design flow
- Proposed graphical user interface
- Comparison results
- Conclusions
Quantitative Results: Frequency

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>AMDREL</th>
<th>XILINX</th>
</tr>
</thead>
<tbody>
<tr>
<td>b01</td>
<td>200</td>
<td>0</td>
</tr>
<tr>
<td>b02</td>
<td>250</td>
<td>200</td>
</tr>
<tr>
<td>b03</td>
<td>100</td>
<td>50</td>
</tr>
<tr>
<td>b04</td>
<td>150</td>
<td>100</td>
</tr>
<tr>
<td>b06</td>
<td>200</td>
<td>150</td>
</tr>
<tr>
<td>b07</td>
<td>100</td>
<td>75</td>
</tr>
<tr>
<td>b09</td>
<td>250</td>
<td>200</td>
</tr>
<tr>
<td>b10</td>
<td>100</td>
<td>50</td>
</tr>
<tr>
<td>b11</td>
<td>200</td>
<td>150</td>
</tr>
<tr>
<td>b12</td>
<td>100</td>
<td>75</td>
</tr>
<tr>
<td>b13</td>
<td>250</td>
<td>200</td>
</tr>
<tr>
<td>b14</td>
<td>150</td>
<td>100</td>
</tr>
<tr>
<td>b15</td>
<td>200</td>
<td>150</td>
</tr>
<tr>
<td>b15_1</td>
<td>100</td>
<td>50</td>
</tr>
<tr>
<td>b17</td>
<td>150</td>
<td>100</td>
</tr>
<tr>
<td>b17_1</td>
<td>50</td>
<td>25</td>
</tr>
<tr>
<td>b20</td>
<td>200</td>
<td>150</td>
</tr>
<tr>
<td>b20_1</td>
<td>100</td>
<td>50</td>
</tr>
<tr>
<td>b21</td>
<td>150</td>
<td>100</td>
</tr>
<tr>
<td>b21_1</td>
<td>50</td>
<td>25</td>
</tr>
<tr>
<td>b22</td>
<td>200</td>
<td>150</td>
</tr>
<tr>
<td>b22_1</td>
<td>100</td>
<td>50</td>
</tr>
</tbody>
</table>

MHz

Benchmark

Maximum Frequency

Graph showing the maximum frequency for AMDREL and XILINX benchmarks.
Quantitative Results: # LUTs

![Number of LUTs](chart.png)

Benchmark

0 2000 4000 6000 8000 10000 12000
LUTs

b01 b02 b03 b04 b06 b07 b09 b10 b11 b12 b13 b14 b15 b15_1 b17 b17_1 b20 b20_1 b21 b21_1 b22 b22_1

Number of LUTs

- AMDREL
- XILINX
## Quantitative Results: Combinational circuit

<table>
<thead>
<tr>
<th></th>
<th>AMRDEL Embedded FPGA</th>
<th></th>
<th>XILINX SPARTAN2 xc2s15</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Combinational: C1908</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Synthesis</strong></td>
<td><strong>Used/Avail.</strong></td>
<td><strong>Usage(%)</strong></td>
<td><strong>XST(synthesis)</strong></td>
<td><strong>Used/Avail.</strong></td>
</tr>
<tr>
<td>#primary IOs</td>
<td>58/96</td>
<td>60</td>
<td>#primary IOs</td>
<td>58/90</td>
</tr>
<tr>
<td># LUTs</td>
<td>148/320</td>
<td>46</td>
<td>Num. of LUTs</td>
<td>106/384</td>
</tr>
<tr>
<td>#Latches</td>
<td>0/320</td>
<td>0</td>
<td>#Latches</td>
<td>0/384</td>
</tr>
<tr>
<td><strong>VPR</strong></td>
<td></td>
<td></td>
<td><strong>P&amp;R</strong></td>
<td></td>
</tr>
<tr>
<td><strong>CLBs</strong></td>
<td>30/64</td>
<td>47</td>
<td>CLBs</td>
<td>27/96</td>
</tr>
<tr>
<td>IOs</td>
<td>58/96</td>
<td>60</td>
<td>bonded IOBs</td>
<td>58/90</td>
</tr>
<tr>
<td>delay (ns)</td>
<td>19,6</td>
<td></td>
<td>delay (ns)</td>
<td>30</td>
</tr>
<tr>
<td>power (mW)</td>
<td>7,4</td>
<td></td>
<td>power (mW)</td>
<td>42,4</td>
</tr>
<tr>
<td>signal power (%total)</td>
<td>44%</td>
<td></td>
<td>signal power (% of total)</td>
<td>29%</td>
</tr>
<tr>
<td>logic power (% of total)</td>
<td>56%</td>
<td></td>
<td>logic power (% of total)</td>
<td>71%</td>
</tr>
</tbody>
</table>
## Quantitative Results: Sequential circuit

<table>
<thead>
<tr>
<th></th>
<th><strong>AMDREL Embedded FPGA</strong></th>
<th></th>
<th><strong>XILINX SPARTAN2 xc2s15</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sequential: S1423</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Synthesis</strong></td>
<td>Used/Avail.</td>
<td>Usage(%)</td>
<td>XST(synthesis)</td>
<td>Used/Avail.</td>
</tr>
<tr>
<td>#primary IOs</td>
<td>23/96</td>
<td>24</td>
<td>#primary IOs</td>
<td>23/90</td>
</tr>
<tr>
<td># LUTs</td>
<td>313/320</td>
<td>98</td>
<td>Num. of LUTs</td>
<td>264/384</td>
</tr>
<tr>
<td>#Latches</td>
<td>74/320</td>
<td>23</td>
<td>#Latches</td>
<td>75/384</td>
</tr>
<tr>
<td><strong>VPR</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>P&amp;R</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLBs</td>
<td>64/64</td>
<td>100</td>
<td>CLBs</td>
<td>73/96</td>
</tr>
<tr>
<td>IOs</td>
<td>23/96</td>
<td>24</td>
<td>bonded IOBs</td>
<td>23/90</td>
</tr>
<tr>
<td>delay (ns)</td>
<td>25,6</td>
<td></td>
<td>delay (ns)</td>
<td>30</td>
</tr>
<tr>
<td>power (mW)</td>
<td>5,5</td>
<td></td>
<td>power (mW)</td>
<td>32,1</td>
</tr>
<tr>
<td>signal power (%total)</td>
<td>32%</td>
<td></td>
<td>signal power (% of total)</td>
<td>51%</td>
</tr>
<tr>
<td>logic power (% of total)</td>
<td>68%</td>
<td></td>
<td>logic power (% of total)</td>
<td>49%</td>
</tr>
</tbody>
</table>
AMDREL Power Analysis

- 61% Routing Power
- 35% CLB Power
- 4% Clock Power
- 0% Leakage Power
AMDREL Power Analysis

Total Power

Total Power with Low-swing

Benchmarks

uWatt

b01  b02  b03  b04  b06  b07  b09  b10  b11  b13  b14  b20  b20_1  b21  b21_1  mac32
AMDREL Power Analysis

Power Consumption

Xilinx
AMDREL

Benchmark

b08  b10  b11  b14  b20  b20_1  b21  b21_1
## Comparison of Configuration Bitstream sizes

<table>
<thead>
<tr>
<th>Device</th>
<th>Config. bits</th>
<th>LUTS/FFs</th>
<th>Device</th>
<th>Config. bits</th>
<th>LUTS/FFs</th>
</tr>
</thead>
<tbody>
<tr>
<td>8×8</td>
<td>33,600</td>
<td>320/320</td>
<td>XCS05</td>
<td>53,984</td>
<td>200/360</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>XCS10</td>
<td>95,008</td>
<td>362/616</td>
</tr>
<tr>
<td>18×18</td>
<td>294,880</td>
<td>1620/1620</td>
<td>XC2S50</td>
<td>559,200</td>
<td>1536/1536</td>
</tr>
<tr>
<td>27×27</td>
<td>1,649,120</td>
<td>3645/3645</td>
<td>XC2S150</td>
<td>1,040,096</td>
<td>3456/3456</td>
</tr>
<tr>
<td>34×34</td>
<td>2,331,240</td>
<td>5780/5780</td>
<td>XCV300</td>
<td>1,751,808</td>
<td>6144/6144</td>
</tr>
<tr>
<td>39×39</td>
<td>2,748,034</td>
<td>7605/7605</td>
<td>XCV400</td>
<td>2,546,048</td>
<td>9600/9600</td>
</tr>
<tr>
<td>47×47</td>
<td>3,984,034</td>
<td>11045/1045</td>
<td>XCV600</td>
<td>3,607,968</td>
<td>13824/13824</td>
</tr>
<tr>
<td>48×48</td>
<td>4,024,454</td>
<td>11520/11520</td>
<td>XCV800</td>
<td>4,715,616</td>
<td>18816/18816</td>
</tr>
<tr>
<td>59×59</td>
<td>16,435,763</td>
<td>17405/17405</td>
<td>XCV1000</td>
<td>6,127,744</td>
<td>24576/24576</td>
</tr>
</tbody>
</table>
Conclusions

• Complete Design Flow: Input VHDL ➔ Output Bitstream

• The single only complete design flow in academia based on open-source tools and running on Linux

• Promising comparisons results with commercial design flow
More info:
AMDREL website: http://vlsi.ee.duth.gr/amdrel
Email: dsoudris@ee.duth.gr – ksiop@ee.duth.gr