Introduction to Reconfigurable Computing

Pao-Ann Hsiung
Embedded Systems Laboratory
National Chung Cheng University
Chiayi, Taiwan, ROC.
http://www.cs.ccu.edu.tw/~pahsiung/courses/rc/
pahsiung@cs.ccu.edu.tw

Outline

- Why Reconfigurable Computing?
- What is Reconfigurable Computing?
- From Codesign to Reconfiguration
- Reconfiguration Tools and Platforms
- Design and Verification Methodologies
- Application Examples

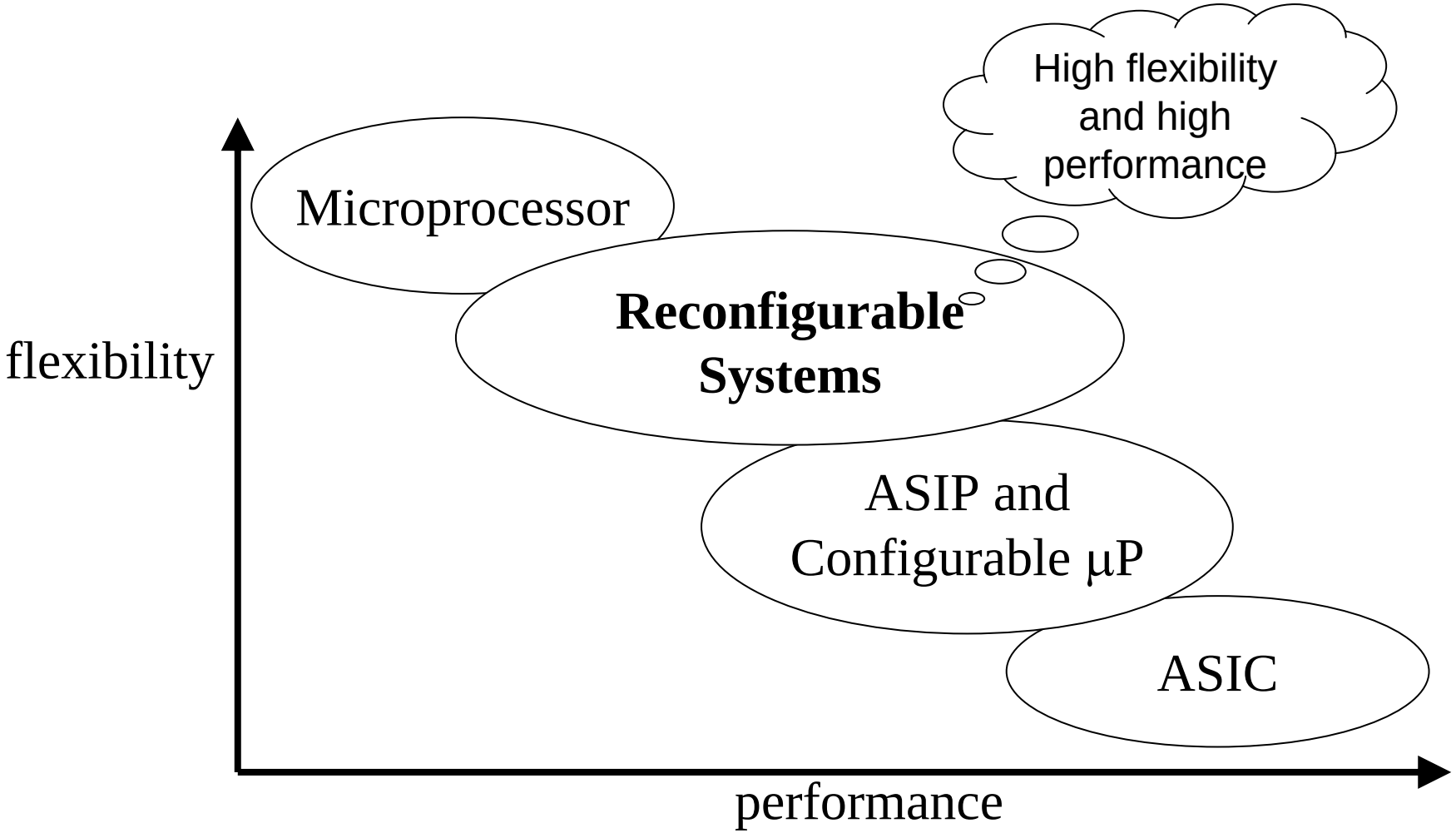
Outline

- Why Reconfigurable Computing?
- What is Reconfigurable Computing?
- From Codesign to Reconfiguration
- Reconfiguration Tools and Platforms
- Design and Verification Methodologies
- Application Examples

Why Reconfigurable Computing?

- Memory access and I/O bottleneck in microprocessor architecture performance
- Increasingly higher NRE (non-recurring engineering) costs and longer design time for nanometer ASIC development
- Applications are getting complex and dynamic, ASICs, ASIPs, and configurable processors are inflexible in coping with dynamic changes
- Difficult hardware-software codesign due to cumbersome prototyping

Filling in the gap!



Comparing MP, ASIP/CP, ASIC, RC

Architecture	Resource	Algorithm	Performance	Cost	Power	Flexibility	Design Effort
MP	Fixed	Programmable	Low	**Low**	Med	**High**	Low
ASIP/CP	Static Recnf	Programmable	Med	**Med**	Low ~ Med	Low	High
ASIC	Fixed	Fixed	**High**	High	Low	Low	High
RC	Dynamic Recnf	Reconfigurable	**Med ~ High**	**Low ~ Med**	High	**High**	Med

MP: Microprocessor, CP: Configurable Processor,
RC: Reconfigurable Computing, Recnf: Reconfiguration, Med: Medium

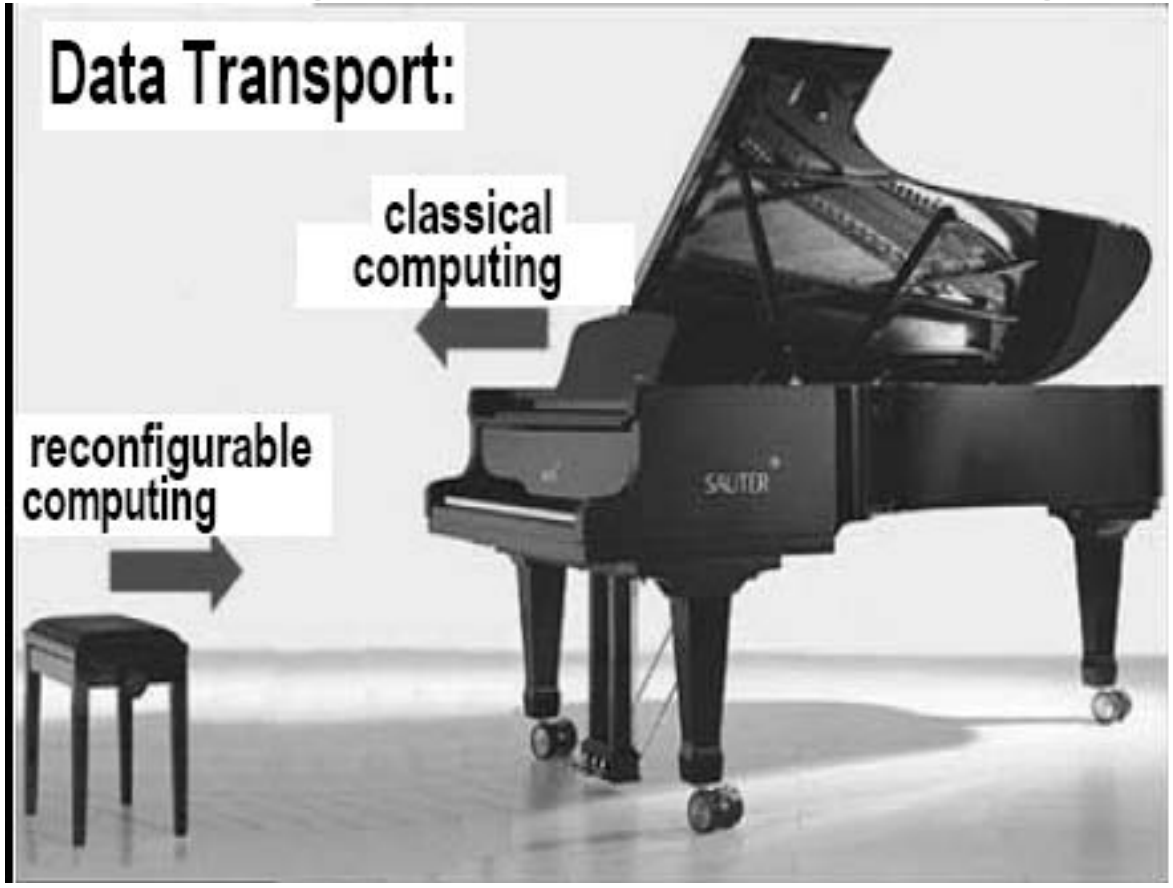
Benefits of Reconfigurable Computing

- Flexible enough to cope with dynamic changes
  - Full Static / Partial Dynamic reconfiguration
- Post-fabrication reconfiguration
  - Reduces development time
    - No backend design flow
  - Reduces NRE costs
    - No fabrication required

Benefits of Reconfigurable Computing

- Application specific design increases resource utilization and performance
  - Precise data size configuration
  - Loop pipelining
- Spatial parallelization is superior to temporal parallelization (μp) in accelerating application performance
  - 10X ~ 100X speedup of data-intensive applications
    - Multi-port distributed memory blocks allow highly parallel memory accesses

Classical Computing vs. Reconfigurable Computing



* R. Hartenstein, “Why we need reconfigurable computing education,” *1st International Workshop on Reconfigurable Computing Education*, March 2006.

Current Disadvantage of Reconfigurable Computing*

- High power consumption
- Immature and non-standard design flows, tools, and methodologies increase design time to as much as 10 times a non-reconfigurable system
- Poor support for partial dynamic reconfiguration

*As of February 2007

Nevertheless, still widely pervasive

- Embedded Systems and SoCs
- Digital Signal Processing
- Image Processing
- Network Security
- Bioinformatics
- Supercomputing
- Boolean SATisfiability (SAT)
- Spacecrafts
- Military applications

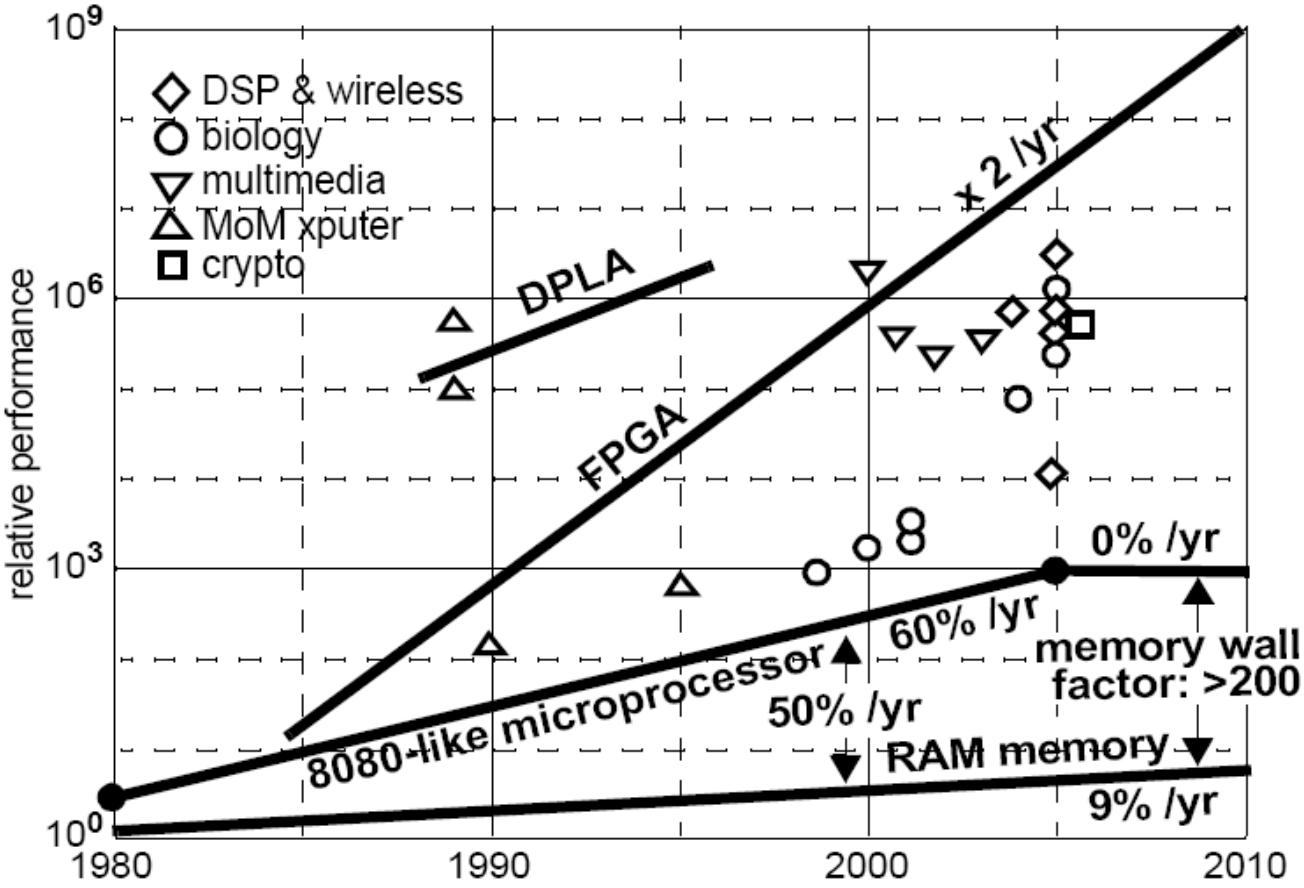
M. Gokhale, P. Graham, *Reconfigurable Computing*, Springer, 2005 (Chapters 5~9)

Pervasiveness of Reconfigurable Computing

- Google hits
  - "**`reconfigurable computing`**"
    - 296,000 in Jan 2006*
    - 1,040,000 in Feb 2007 (more than 3X in one year!)
  - "**`FPGA`**"
    - 9,190,000 in Feb 2007 (approx. 10 million)
- Pervasiveness of RC
  - http://fpl.org/pervasiveness.html

* R. Hartenstein, "Why we need reconfigurable computing education," *1st International Workshop on Reconfigurable Computing Education*, March 2006.

Why so pervasive? Speedup!!!



* R. Hartenstein, “Why we need reconfigurable computing education,” *1st International Workshop on Reconfigurable Computing Education*, March 2006.

FFT: FPGA is faster than ASIC! Why?

Implementation Technology	CMOS Tech.	FFT Size	Input Data Format	Clock Rate	Time per FFT	Power
Dillon Eng. (Virtex-II Pro)	130 nm	4096	2x18-bit, fixed pt.	200 MHz	3.84 μs [125]	< 4 W [125]
Dillon Eng.	150 nm	4096	2x18-bit	160 MHz	4.8 μs [125]	< 4 W [125]
(Altivec, VSIPL)						
Dillon Eng. (Virtex-II Pro)	130 nm	4096	2x32-bit, floating pt.	200 MHz	30.7 μs [125]	< 4 W [125]
Dillon Eng. (Virtex-II)	150 nm	4096	2x32-bit, floating pt.	160 MHz	38.4 μs [125]	< 4 W [125]
Eonic PowerFFT (ASIC)	180 nm	4096	2x32-bit, floating pt.	128 MHz	48 μs [138]	< 2 W [138]
Radix RaCE FFT (ASIC, 2 Chips)	N/A	**2048**	2x18-bit, bl. floating pt.	84 MHz	24.4 μs [336]	< 3.5 W [336]
CRI Pathfinder 2 (ASIC)	250 nm	**2048**	2x32-bit floating pt.	120 MHz	31 μs [124]	< 2.5 W (estimated)

FPGA uses the latest and fastest CMOS fabrication technology, while the technology adoption by ASICs is delayed due to NRE costs and design time.

2048 points ∴needs 2X time

Education of RC

- Education on RC
  - 1st Workshop on Reconfigurable Computing Education, Germany, March 2006
  - 2nd Workshop on Reconfigurable Computing Education, Brasil, May 2007
  - http://helios.informatik.uni-kl.de/RCeducation/

Outline

- Why Reconfigurable Computing?
- What is Reconfigurable Computing?
- From Codesign to Reconfiguration
- Reconfiguration Tools and Platforms
- Design and Verification Methodologies
- Application Examples

What is Reconfigurable Computing?

- Reconfigurable Computing
  - A discipline in which system or application functions can be changed by configuring a fixed set of logic resources through memory settings (our definition)
    - Functions: transforms, filters, codec, protocol, …
    - Fixed set of logic resources: logic block, I/O block, routing block, memory block, application-specific block (e.g. MAC, DSP, MP, …)
    - Memory settings: configuration bits

Other Definitions of Reconfigurable Computing

- “A reconfigurable computer is a device which computes by using post-fabrication spatial connections to compute elements.”

  - Andre Dehon

- “General purpose custom hardware”

  - Seth Copen Goldstein

- “On the fly ASIC”

  - Fadi Kurdahi

- “Logic-on-demand”

  - Virtual Computer Corporation (VCC)

Programmable vs. Reconfigurable?

- Programmable
  - Computation is performed by fetching, decoding, and executing a few of a pre-defined set of instructions
- Reconfigurable
  - Computation is performed by configuring logic resources into application-specific functions

Programmable vs. Reconfigurable?

Char.	Programmable	Reconfigurable
Cfg Level	Instructions	Logical bits
Datapath	Spatially fixed	Spatially configurable
Inst Depth	Megabytes (in cache)	A few configurations
Memory hierarchy	Pre-defined, cache, limited ports	User-defined, no cache, multiple ports
Acceleration	Pre-defined, hidden	User-defined, explicit
Orientation	Control-oriented (fetch, decode, execute)	Data-oriented (data-path pipelining)
Resources	Suboptimal use	Near optimal use
Parallelism	Temporal	Spatial

Reconfigurable System Design Choices

- Application-specific datapath width
  - For example, 5 bits, 17 bits
- Memory hierarchy design
  - Lookup Table (LUT)-based, flip-flop, on-chip SRAM, on-chip processor cache, on-board memory, …
- Execution control
- Function unit allocation
  - Control Logic Block (CLB), MAC, multipliers, DSP, …
- Spatial parallelism, pipelining

Characteristics of Reconfigurable Computing Resources

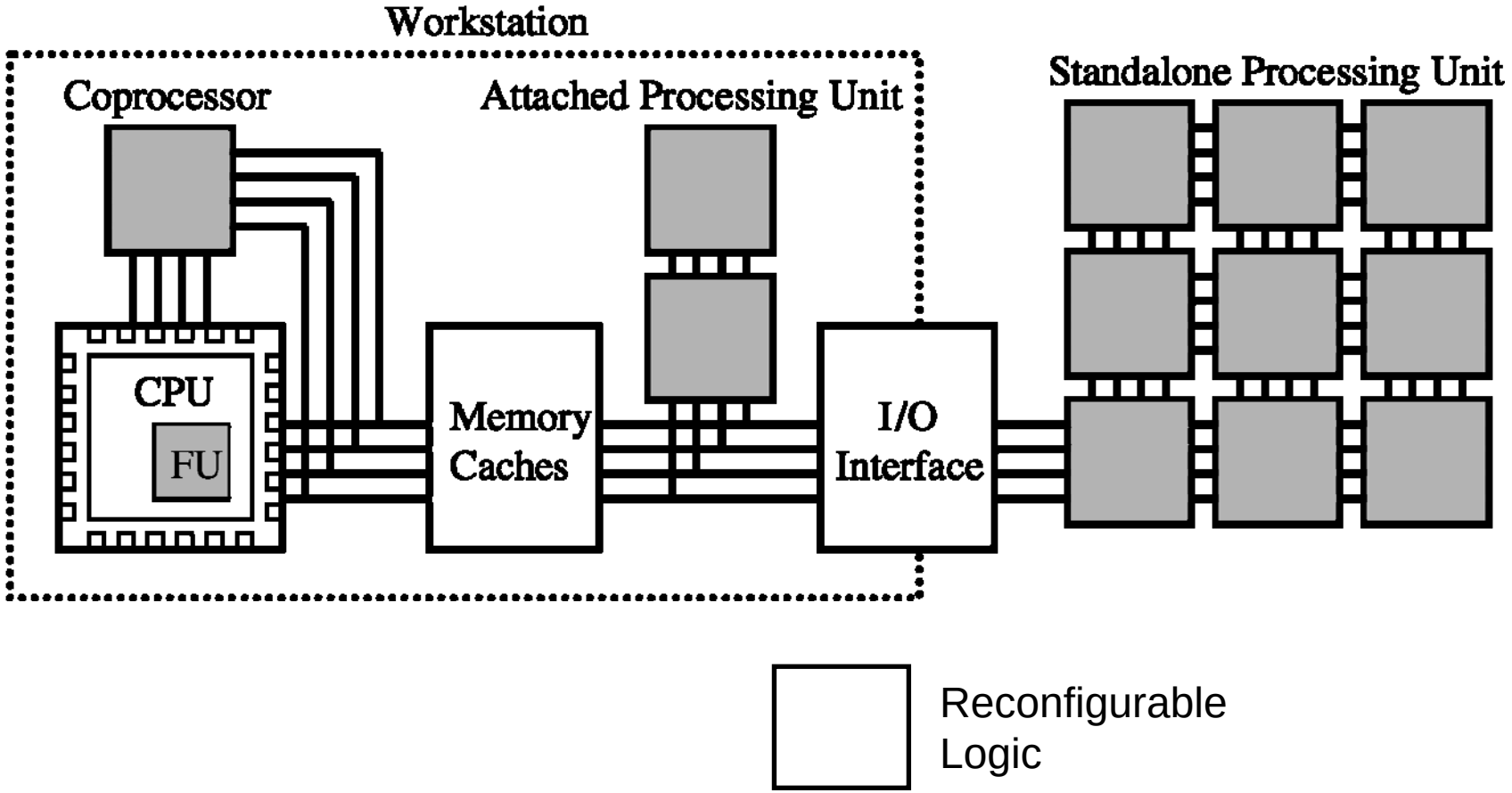
- Granularity
  - Fine-grained, Coarse-grained
  - Instruction, Function, Process
- Coupling
  - System, IP, Coprocessor, Micro-architecture
- Routing Dimensionality
  - Mesh, island, row, systolic, …

(Details in Chapter 2)

Granularity

Attribute	Fine-grained	Coarse-grained
Datapath width	1 bit	32 bits, …
Reconfiguration	Gate level	RT level
Reconfigurable unit	LUT, CLB, LAB, …	ALU, FP, …
Config time	milliseconds	microseconds
Clock cycle	~ 0.5 GHz	~ 1 – 3 GHz
Example	FPGA	rDPA

Coupling


Workstation
Coprocessor
Attached Processing Unit
Standalone Processing Unit
CPU
FU
Memory
Caches
I/O
Interface
Reconfigurable
Logic

Routing Dimensionality (1/3)

- Mesh based (cellular) architectures
  - 2D array, nearest neighbor connections, e.g. XC6200, CLi/Atmel 6000 FPGA, Plessey/Pilkington ERA
- Island style architectures
  - Local and global connections, e.g. Xilinx FPGA

Routing Dimensionality (2/3)

- Long line routing architectures
  - Long horizontal or vertical lines, e.g. Altera FPGA, Actel ProASIC FPGA
- Row based architectures
  - 1D communication, e.g. Garp, Chimaera, Actel Act-1 FPGA

Routing Dimensionality (3/3)

- Systolic architectures
  - Data-streams driven by data counters (no PC)
  - Proposed by Kung and Leiserson [1978]
  - e.g. KressArray by Rainer Kress

Outline

- Why Reconfigurable Computing?
- What is Reconfigurable Computing?
- From Codesign to Reconfiguration
- Reconfiguration Tools and Platforms
- Design and Verification Methodologies
- Application Examples

Reconfigurable System = RC + MP

- Most reconfigurable systems or platforms either have a hard microprocessor core embedded or soft microprocessor cores can be configured
  - Altera: ARM 922T (hard), Nios II (soft)
  - Xilinx: PowerPC 405 (hard), Microblaze (soft)
- Hard or soft bus-based communication architectures
  - Two hard AHBs in Altera Excalibur FPGA
  - Soft AMBA in Xilinx Virtex-II Pro and Virtex-4 FPGA

In a reconfigurable system …

- Hardware tasks run on the logic resources
- Software processes run on the microprocessor cores
- HW and SW communicate through buses and shared memories
- OS schedules all computations and communications

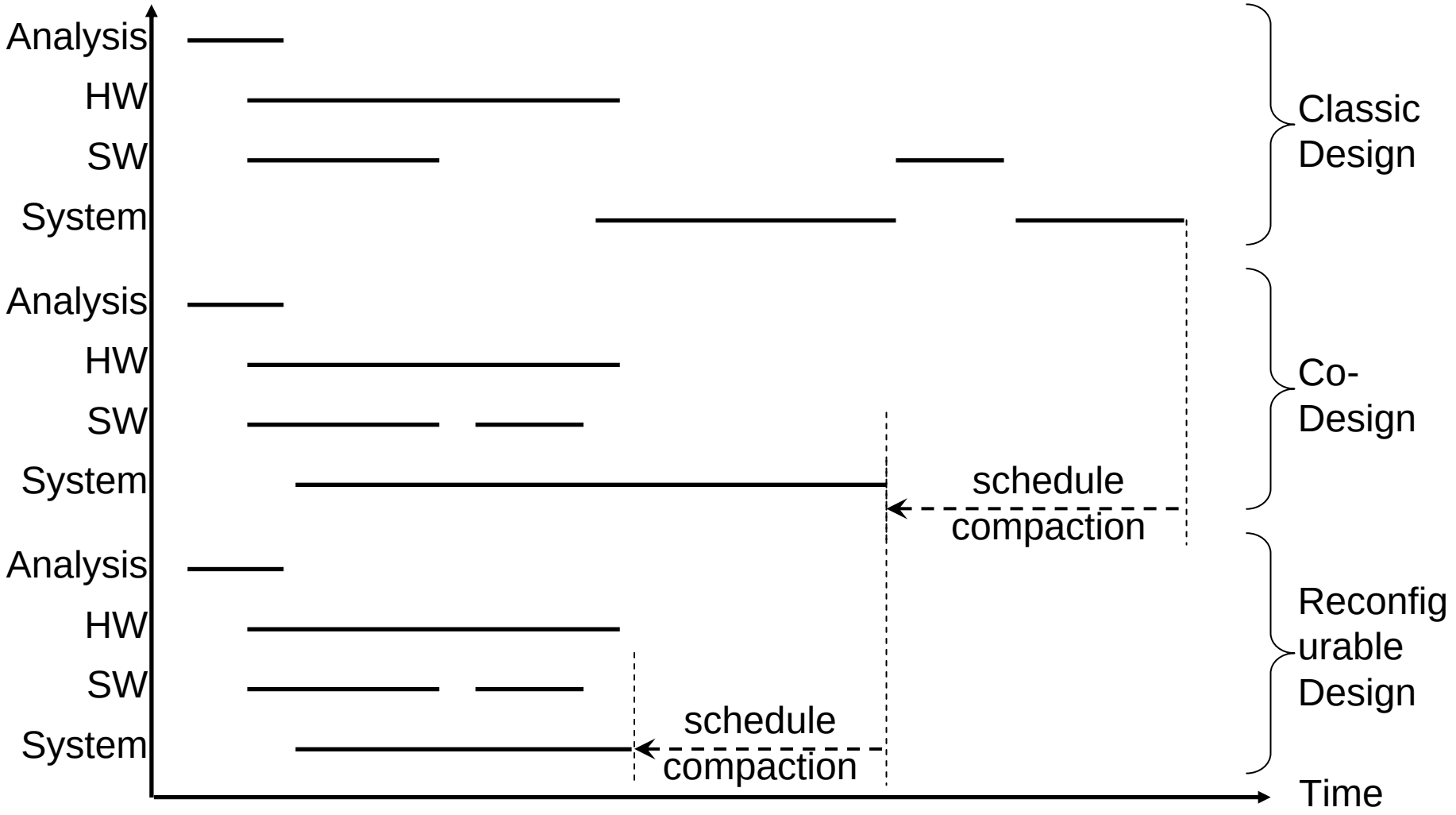
Main Problems of HW-SW Codesign

- HW-SW cosimulation is time-consuming
  - Can evaluate only a few design alternatives
  - Need a lot of error-prone optimizations
- Estimations of hardware and software design performance are inaccurate
  - Suboptimal system design results
- Backend design impossible
  - Poor estimations of physical characteristics such energy consumption, real-time attributes, …

Reconfigurable Computing to the Rescue!!!

- Reconfigurable logics, though slower than ASICs, are much faster than software cosimulation tools
  - Can evaluate more design alternatives
  - No need of error-prone cosimulations
- “HW in hardware” and “SW in software” allow accurate estimations of hardware and software performance
  - More optimal design results
- Post-fabrication configuration allows physical characteristics to be estimated with very high accuracy
  - Accurate estimations of HW area, timing, power, ...

Schedule Compactions



Impact on CS/EE Education

- Current status quo
  - SW: programming language, data-structure, algorithm, OS
  - HW: digital systems, computer organization, VLSI design, CAD, …
- Future trend (with the aid of RC!)
  - Micro: Hardware-software codesign, system design and analysis (OS/CA), …
  - Macro: Application design (game), …

Outline

- Why Reconfigurable Computing?
- What is Reconfigurable Computing?
- From Codesign to Reconfiguration
- Reconfiguration Tools and Platforms
- Design and Verification Methodologies
- Application Examples

Reconfiguration Tools

- Hardware Design
  - FPGA synthesizer, placer, router, simulator, debugger, analyzer
- Software Design
  - Compiler, simulator, debugger, …
- System Design
  - Codesign tools, algorithm synthesizer, system synthesizer, evaluation frameworks

Reconfigurable HW Design Tools

- FPGA Suite Tools
  - Xilinx ISE Foundation
  - Altera Quartus
  - Mentor Graphics FPGA Advantage
- FPGA Synthesizer
  - Synplicity Synplify Pro
  - Synopsys FPGA Compiler
  - Mentor Graphics Leonardo Spectrum
  - Mentor Graphics Precision Synthesis

Reconfigurable HW Verification Tools

- Simulator
  - Mentor Graphics Modelsim
  - Cadence NC Sim, Scirocco Simulator, Verilog-XL
  - Verisity Spexsim
  - Synopsys VCS
- Analysis Tools
  - Xilinx Chip Scope Pro

Reconfigurable SW Design Tools

- Compiler
  - gcc, lcc, …(vendor proprietary tools)
- Assembler
  - Specific to microprocessor
- Locator/Linker
  - Specific to microprocessor and compiler
- Instruction Set Simulator (ISS)
  - Specific to microprocessor

Reconfigurable SW Verification Tools

- System-Level Simulators
  - SystemC-based environments such as Coware's Platform Architect
- Debuggers
  - Front end debugger (gdb, Turbo debugger)
  - Remote debugger
  - Logic Analyzer

Reconfigurable System Design Tools

- Hardware-Software Codesign
  - Xilinx Platform Studio
- Algorithm Synthesizer
  - Xilinx System Generator
  - Altera DSP Builder (Quartus II ⇔ Matlab)
- System Synthesizer
  - Altera SOPC Builder (IP blocks from Celoxica)

Reconfigurable System Verification Tools

- Evaluation Frameworks
  - SystemC-based framework, e.g. Perfecto [13], SyCERS [12], Coware's Platform Architect
  - C++-based framework, e.g. OCAPI-XL [2]

Reconfiguration Platforms

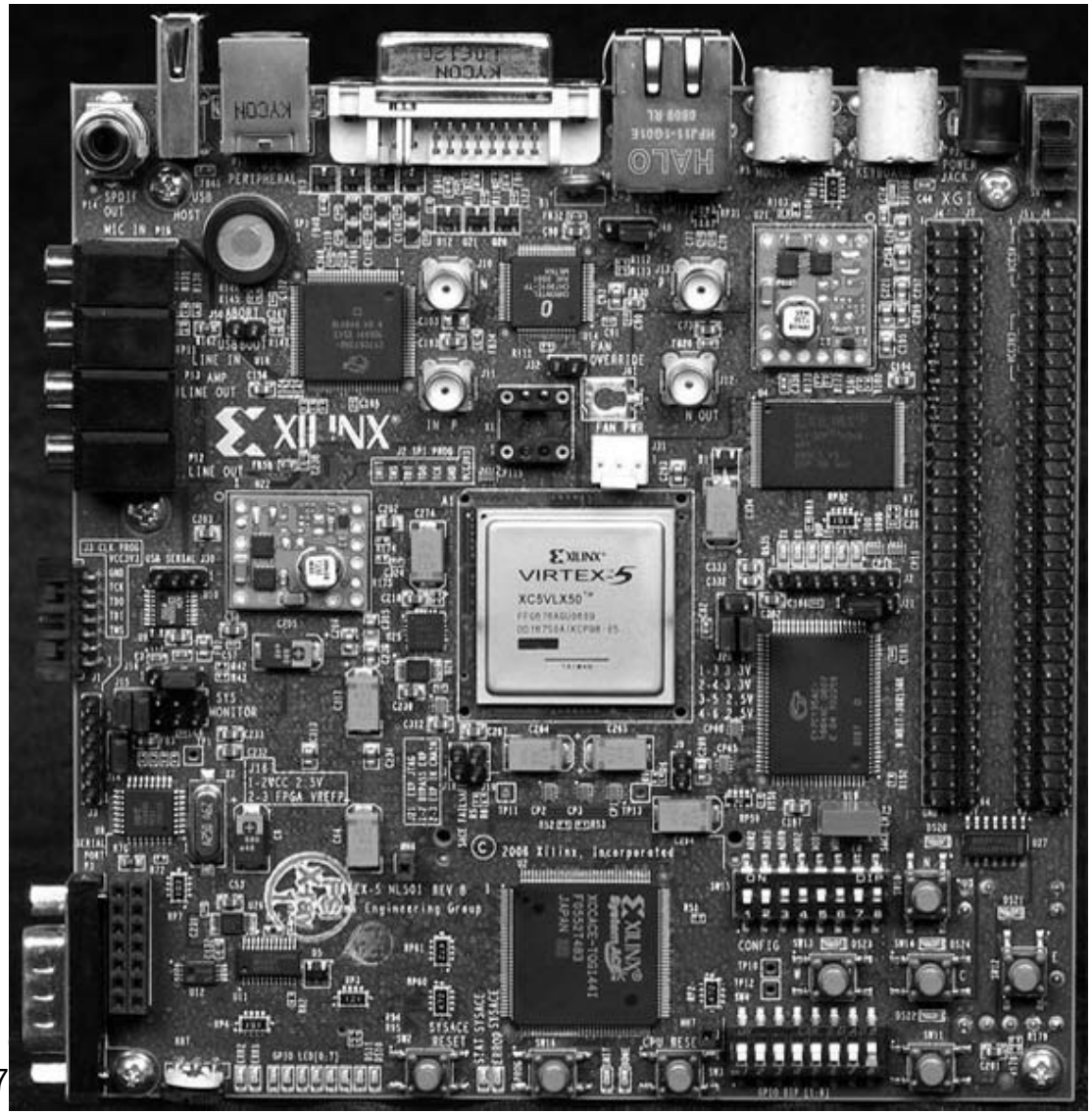
- Reconfigurable Logic + Microprocessor
  - Xilinx
    - PowerPC, MicroBlaze, PicoBlaze
    - Virtex-II Pro, Virtex-4, Virtex-5
  - Altera
    - Nios II
    - Cyclone, Cyclone II, Stratix, Stratix II, Stratix III
    - First integration
      - Excalibur: APEX 20KE + ARM 922T

Reconfiguration Platforms

- Application-oriented FPGAs
  - Xilinx Virtex-5 (65nm)
    - LX: Logic applications
    - LXT: Logic applications with advanced serial connectivity
    - SXT: Signal processing applications
    - FXT: Embedded systems
  - Altera Stratix III (65nm)
    - L: Logic applications
    - E: DSP and memory-rich applications
    - GX: multi-gigabit transceivers

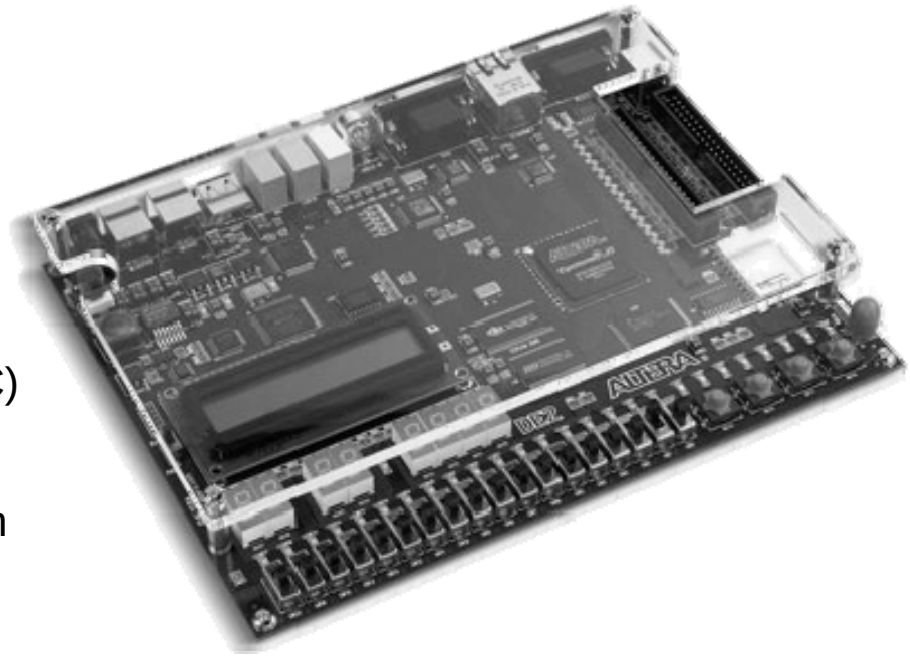
Reconfiguration Platforms (Xilinx ML501)

- XC5VLX50FFG676
- DDR2 SODIMM (256 MB)
- ZBT SRAM (1 MB)
- Linear Flash (32 MB)
- System ACE™ CF (Compact Flash)
- Platform Flash
- SPI Flash
- JTAG Programming Interface
- External Clocking (2 Differential Pairs)
- USB (2) - Host and Peripheral
- PS/2 (2) - Keyboard, Mouse
- RJ-45 - 10/100 Networking
- RS-232 (Male) - Serial Port
- Audio In (2) - Line, Microphone
- Audio Out (2) - Line, Amp, SPDIF, Piezo Speaker
- Video (DVI/VGA) Output
- Single-Ended and Differential I/O Expansion
- GPIO DIP Switch (8)
- GPIO LEDs (8)
- GPIO Pushbuttons (5)



Reconfiguration Platforms (Altera DE2)

- Cyclone II EP2C35F672C6
- Built-in USB-BlasterTM cable
- 10/100 Ethernet
- RS232
- Video out (VGA 10-bit DAC)
- Video in (NTSC/PAL/multi-format)
- USB 2.0 (type A and type B)
- PS/2 mouse or keyboard port
- Line in/out, microphone in (24-bit Audio CODEC)
- Expansion headers (76 signal pins)
- Infrared port
- 8-MBytes SDRAM, 512K SRAM, 4-MBytes flash
- SD memory card slot
- 16 x 2 LCD display
- Eight 7-segment displays
- 18 toggle switches
- 18 red LEDs, 9 green LEDs
- Four debounced push-button switches
- 50-MHz crystal for FPGA clock input
- 27-MHz crystal for video applications
- External SMA clock input



Outline

- Why Reconfigurable Computing?
- What is Reconfigurable Computing?
- From Codesign to Reconfiguration
- Reconfiguration Tools and Platforms
- Design and Verification Methodologies
- Application Examples

Design and Verification Methodologies

- Platform
  - Boards
    - Xilinx ML320, ML410, ML501
    - Altera DE2, DE1, UP3-12, UP3-06, UP2
    - Numerous other vendors …

Design and Verification Methodologies

- Architectures
  - Component-based:
    - Caronte Architecture (black box) [11],
    - Adriatic Project (Dynamically Reconfigurable Fabric, DRCF) [2]
  - Threaded model:
    - Adriatic Project (OCAPI/XL in SystemC) [2]
  - Slot-based: PaDReH [1]
  - 1D-based: CCU Architecture
  - 2D or tile-based:
  - Virtual Logic Architecture: CCU2 Architecture

Design and Verification Methodologies

- Simulation Framework
  - SystemC based
    - Perfecto [13]
    - SyCERS [12]
    - Adriatic project [2]
  - C++ based
    - OCAPI-XL [2]

Design and Verification Methodologies

- Techniques
  - Hardware-Software Partitioning
  - Hardware IP Placement
  - Hardware Task Scheduling
  - Operating System for Reconfigurable Systems (OS4RS)
    - Hardware-Software Communication
    - Task Relocation (HW to SW and vice versa)

Design and Verification Flows (1)

- System-oriented
  - SystemC based
    - PaDReH [1], Adriatic Project [2]
  - UML-SystemC based
    - Tseng & Hsiung [3]
- Software-oriented
  - Threaded model
    - OCAPI/XL (C++) with SystemC [2]

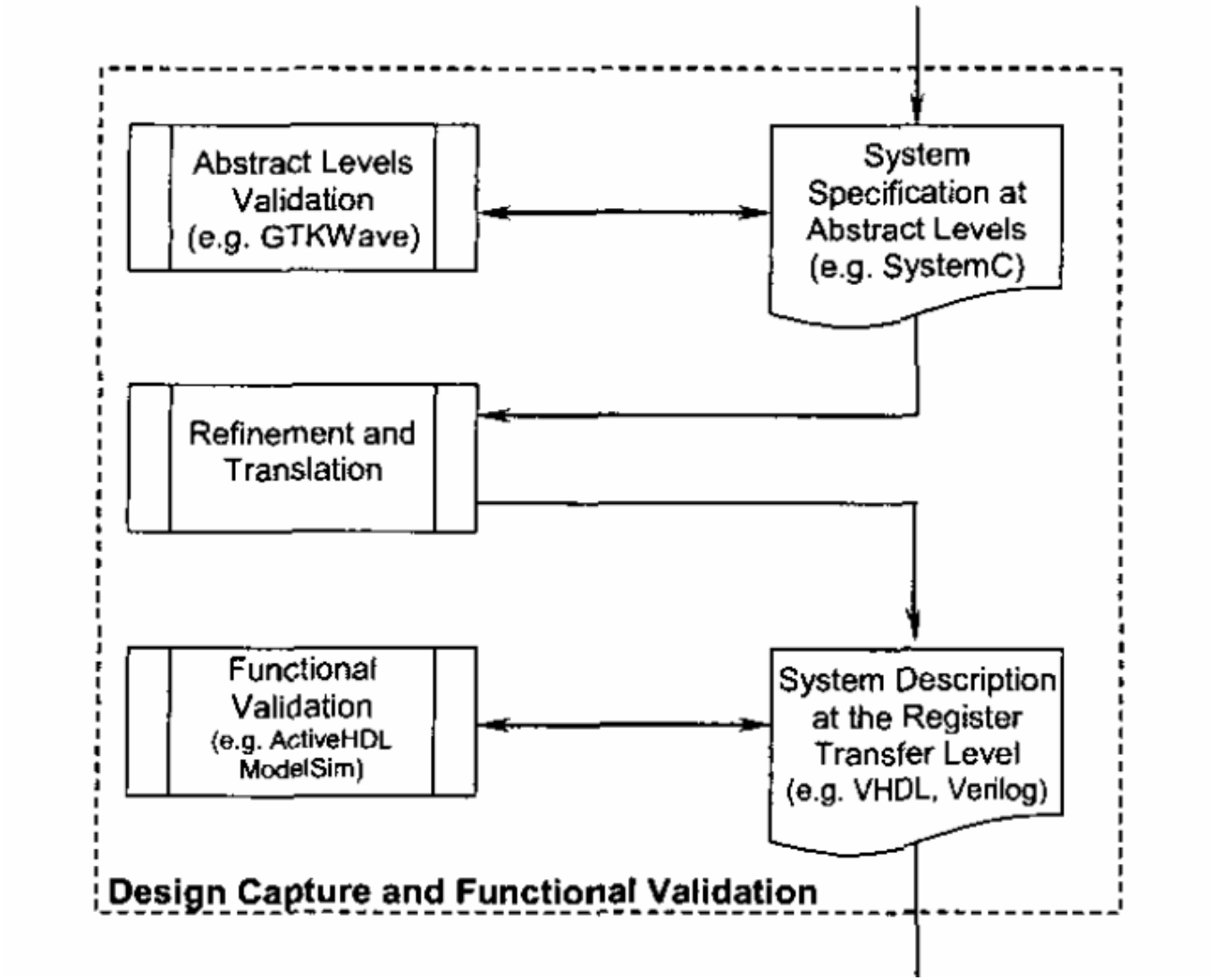
Design and Verification Flows (2)

- Hardware-oriented
  - Paged hardware (temporal partitioning)
    - SCORE (Stream Computation Organized for Reconfigurable Execution) [4]
  - Stages of computation
    - Janus (Just Another HDL (JHDL) based) [5]
  - Glyph based
    - CHAMPION (Khoros Cantata visual programming software, automatic partitioning for multi-FPGA) [8]

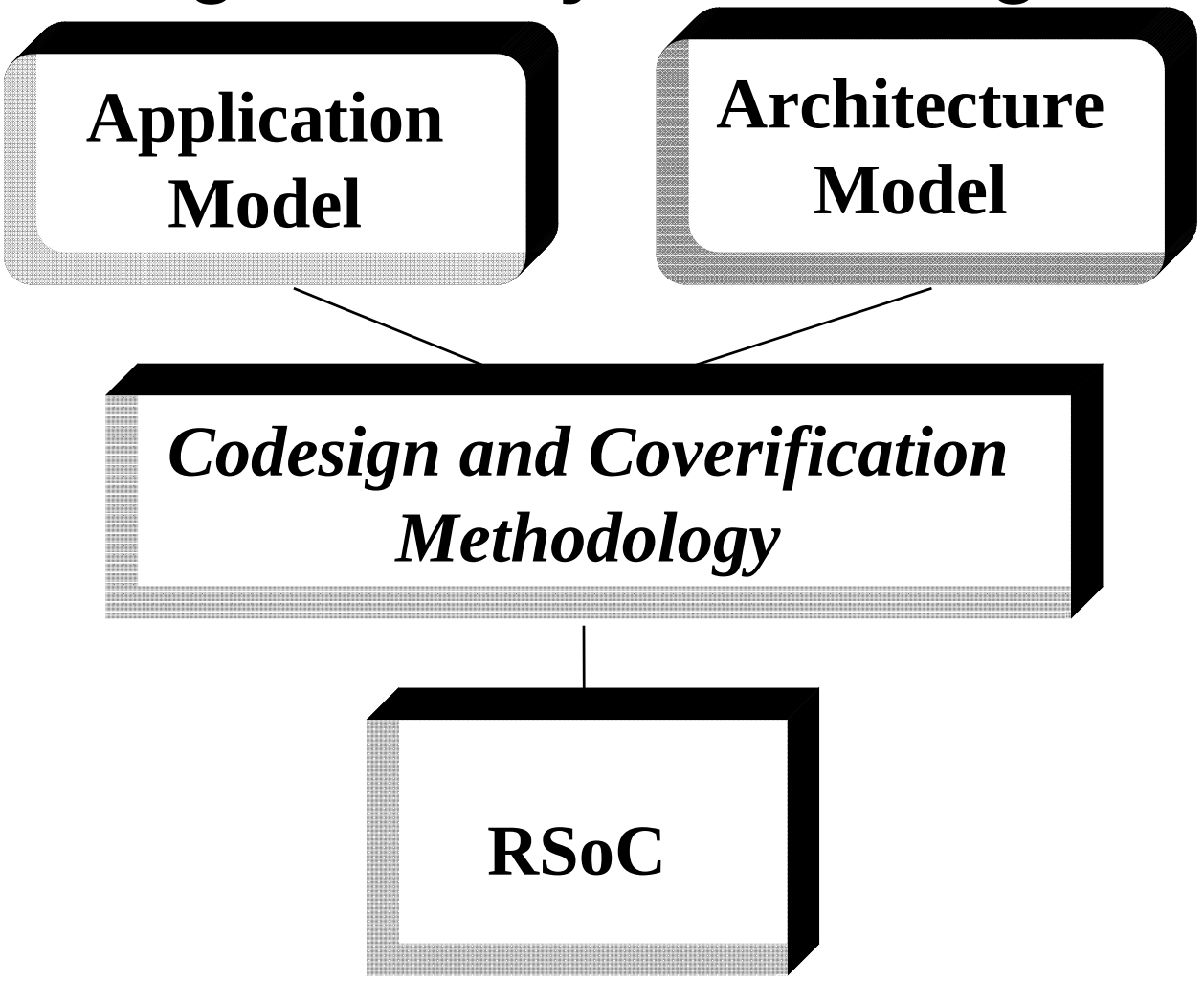
Design and Verification Flows (3)

- Task-oriented
  - Task graph based
    - SPARCS (Synthesis and Partitioning for Adaptive Reconfigurable Computing Systems) [10]
  - Task based
    - FSS (FPGA Support System) [6]
  - Problem graph and architecture graph
    - Eisenring and Platzner [9]
  - Graphical models
    - Model-Integrated Development Environment for Adaptive Computing (MIDE) [7]

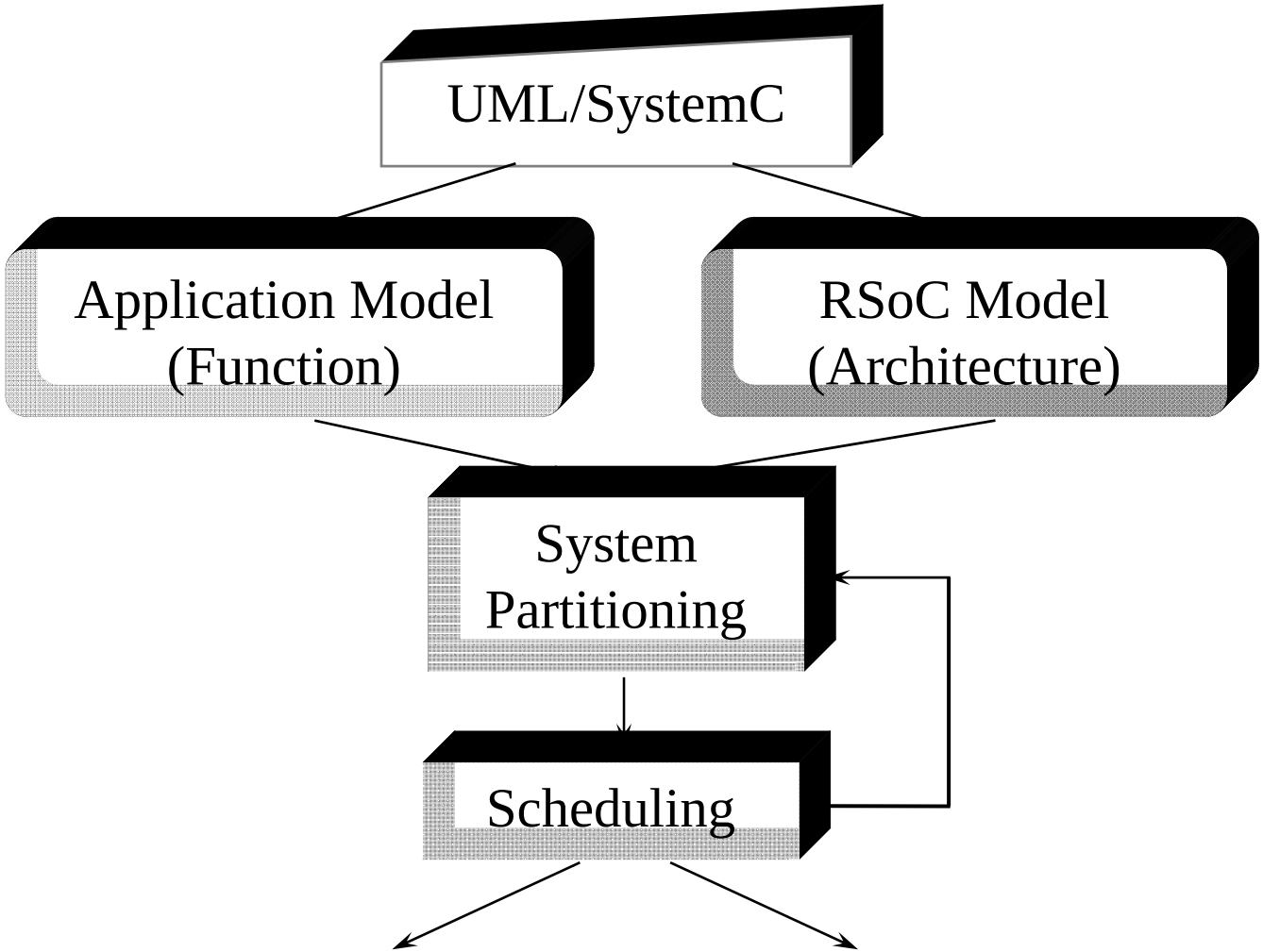
PaDReH: Design Flow for Dynamically and Partially Reconfigurable Systems



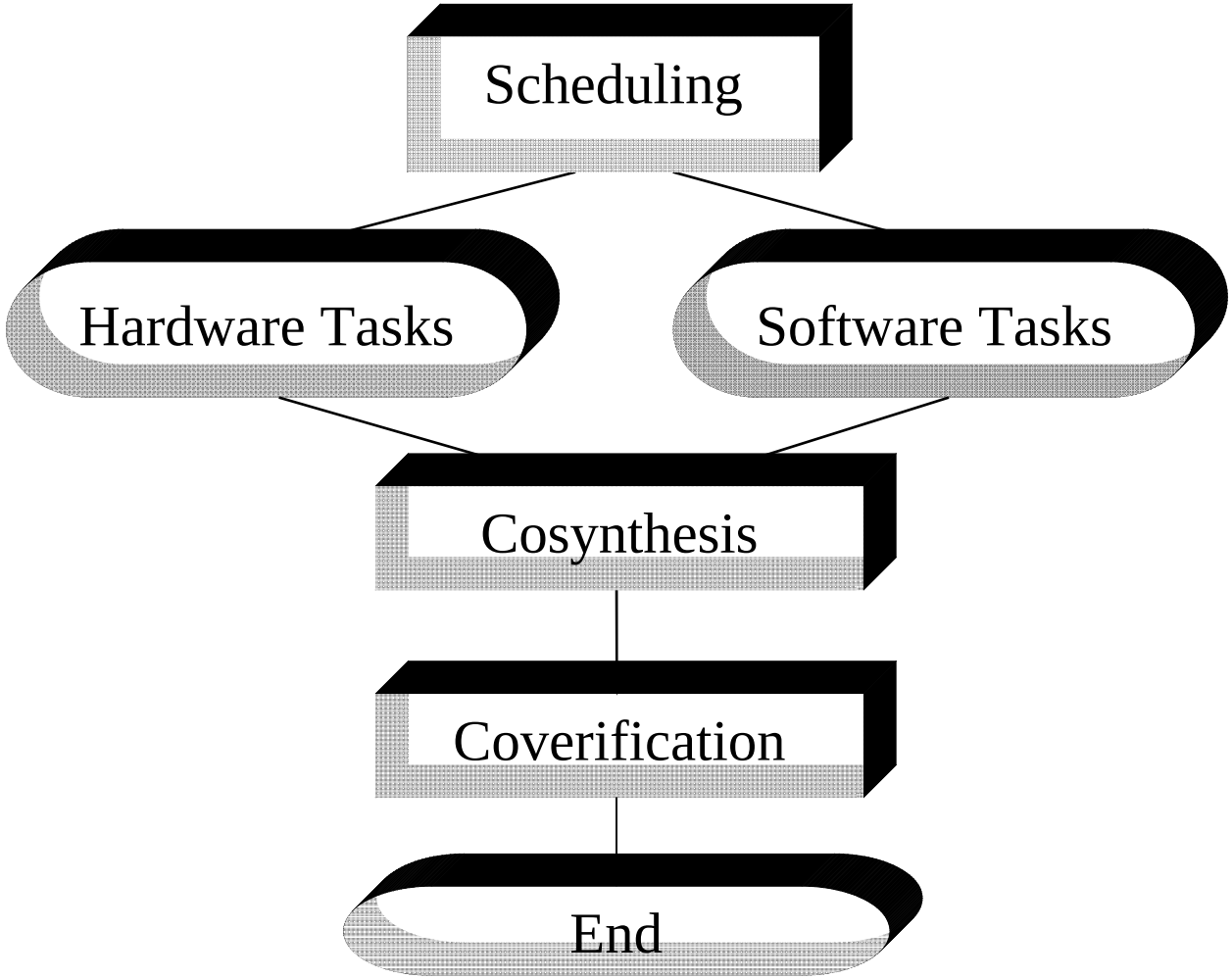
Tseng & Hsiung's UML-SystemC based Reconfigurable System Design Flow



Codesign in UML-SystemC Design Flow (1/2)



Codesign in UML-SystemC Design Flow (2/2)



Outline

- Why Reconfigurable Computing?
- What is Reconfigurable Computing?
- From Codesign to Reconfiguration
- Reconfiguration Tools and Platforms
- Design and Verification Methodologies
- Application Examples

Application Examples

- Dot Product

```
int acc;
int coeff[n], data[n];
for(i=0;i<n;i++)
    acc += coeff[i]*data[i];
```

C code

- Kernel operation in
  - matrix multiply
  - image processing convolution

Conventional vs. Reconfigurable

Dot Product	**Conventional**	**Reconfigurable**
Data type (**`acc`**)	int	8~32 bits (MAC, registers, memory)
Data array (**`coeff[n], data[n]`**)	Loading into cache is transparent to compiler	Explicitly mapped to on-chip and off-chip RAMs
Loop (**`for {…}`**)	Instruction set architecture	Data path and control
Performance	Hidden in MP	Spatial pipeline
Design Effort	Easy 1x	Hard 10x

Application Speedup due to RC

Algorithm	Reconfigurable System	Comparison CPU	Speedup
DNA Matching [31, 32]	SPLASH-2	SPARC 10, Cray2, 16K CM2	4300, 300, 200
RSA Crypto [5]	PAM	Alpha 150 MHz	17.8
SAT [33]	IKOS VirtualLogic SLI Emulator (16 Xilinx XC4013E)	Sun 5	100
Serpent Block Cipher [34]	Xilinx Virtex XCV1000	200 MHz Pentium Pro	18
Sieving for factoring large numbers [35]	Mojave	200 MHz UltaSparc	28
String Pattern Matching [36]	Xilinx XC6216	300 MHz Pentium II	29.9
LZ data compression [37]	Wildforce (4 Xilinx 4036XLA)	450 MHz Pentium II Xeon	30
Traveling Salesman [38]	SPLASH-2	125 MHz PA-RISC	4
Spec92 [26]	MIPS+RC	MIPS	1.12
Shape Adaptive Template Matching [39]	Virtex 1000E	1.4 GHz Pentium 4	7000
RC4 Key Search Engine [40]	Xilinx Virtex XCV1000-E	1.5 GHz Pentium 4	58

Source: [15]

References

1. Carvalho, E., Calazans, N., Briao, E., Moraes, F. PaDReH – A framework for the design and implementation of dynamically and partially reconfigurable systems, *SBCCI*, ACM Press, September 2004.
2. Voros, N. S. and Masselos, K. *System Level Design of Reconfigurable Systems-on-Chip*, Springer, 2005.
3. Tseng, C.-H. and Hsiung, P.-A. UML-Based Design Flow and Partitioning Methodology for Dynamically Reconfigurable Computing Systems, In: *Proceedings of the 2005 IFIP International Conference on Embedded and Ubiquitous Computing* (EUC'2005, Nagasaki, Japan), LNCS, Vol. 3824, pp. 479-488, Springer Verlag, December 2005.
4. Caspi, E., DeHon, A., and Wawrzynek, J. A streaming multithreaded model, In: *Third Workshop on Media and Stream Processors*, 2001.
5. Lehn, D., Hudson, R., and Athanas, P. Framework for architecture-independent run-time reconfigurable applications, *In: SPIE Proceedings,* November, 2000.

References

6. Edwards, M., and Green, P. Run-time support for dynamically reconfigurable computing systems. *Journal of Systems Architecture,* v. 49, pp. 267-281. 2003
7. Bapty, T., Neema, S., Scott, J., Sztipanovits, J., and Asaad, *S.* Model-integrated tools for the design of dynamically reconfigurable systems, Technical Report #ISIS-99-01, ISIS, Vanderbilt University, 2000.
8. Natarajan, S., Levine, B., Tan, C., Newport, D., and Bouldin, D. Automatic Mapping of Khoros-based Applications to Adaptive Computing Systems. *In: MAPLD.99,* pp. 101-107, 1999.
9. Eisenring, M., and Platzner, M. A framework for run-time reconfigurable systems, *Journal of Supercomputing* v. 21, pp. 145-159, 2002.
10. Ouaiss, I., Govindarajan, S., Srinivasan, V., Kaul, M., and Vemuri, R. *An* Integrated Partitioning and Synthesis System for Dynamically Reconfigurable Multi-FPGA Architectures. *In: RAW98,* 1998.

References

11. Donato, A., Ferrandi, F., Redaelli, M., Santambrogio, M.D., and Sciuto, D. Caronte: A complete methodology for the implementation of partially dynamically self-reconfiguring systems on FPGA platforms. pp. 321-322, FCCM 2005.
12. Amicucci, C., Ferrandi, F., Santambrogio, M.D., Sciuto, D. SyCERS: a SystemC design exploration framework for SoC reconfigurable architecture. pp. 63-69, ERSA 2006.
13. Hsiung, P.-A., Huang, C.-H., and Liao, C.-F. Perfecto: A SystemC-based performance evaluation framework for dynamically partially reconfigurable systems, In *Proceedings of the 16th IEEE International Conference on Field Programmable Logic and Applications*, (FPL'2006, Madrid, Spain), pp. 190-198, IEEE CS Press, August 2006.
14. Gokhale, M. and Graham, P. S. *Reconfigurable Computing: Accelerating Computation with Field-Programmable Gate Arrays*, Springer, 2005.
15. Kastner, R., Kaplan, A., and Sarrafzadeh, M. *Synthesis Techniques and Optimizations for Reconfigurable Systems*, Kluwer Academic Publishers, 2004.