Reconfigurable technology &tools: the FASTER approach &future opportunities

Dionisios Pnevmatikatos

pnevmati@ics.forth.gr

Reconfigurable Technology

Technology for practical adaptable hardware systems

- Can add/remove components at run-time/product lifetime
- Flexibility at hardware speed (not quite ASIC)
- Parallelism at hardware level (depending on application)
- Ideally: alter function & interconnection of blocks

Implementation in:

- FPGAs: fine grain, complex gate plus memory and DSP blocks
- Coarse Grain (custom) chips: multiple ALUs, multiple (simple) programmable processing blocks, etc.

FASTER Motivation

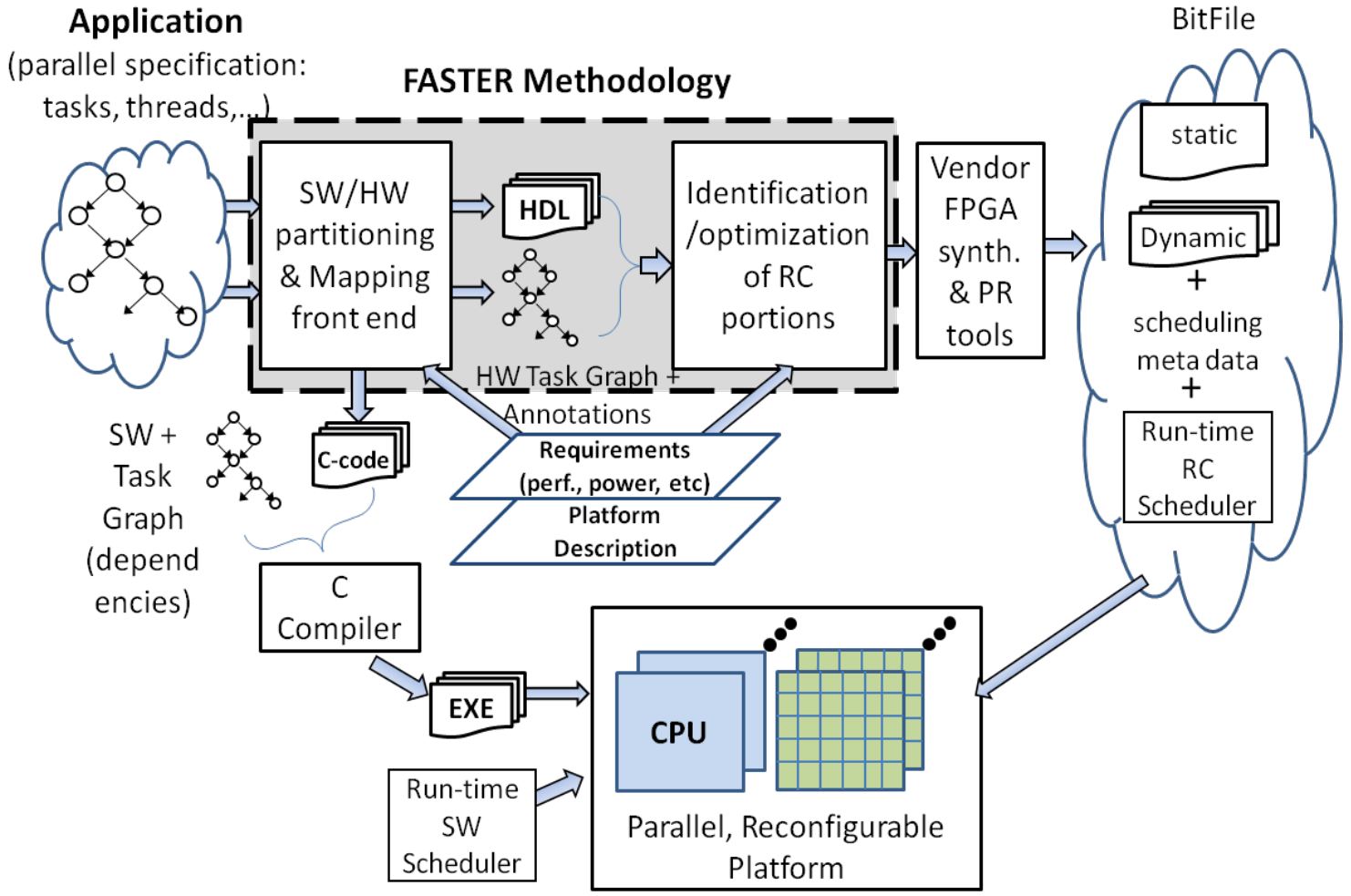
Creating reconfigurable systems is not straightforward!

- The designer has to:
  - Identify portions to be reconfigured
  - Establish a schedule that (a) respects dependencies while at the same time (b) achieves performance and other constraints
  - Manage the system resources (reconfiguration area mainly)
  - Reconfiguration cost is substantial (use wisely)
  - Verify a *changing* system!
- Tool support for these tasks is esoteric to say the least
- Resource management is up to the user
- Verification: *any* support today?

FASTER Goals & Innovation

- Include **reconfigurability** as **an explicit design concept** in computing systems design, along with **methods and tools** that support run-time reconfiguration in the entire design methodology
- Provide a framework for analysis, synthesis and verification of a reconfigurable system
- Provide efficient and transparent runtime support for partial and dynamic reconfiguration, including micro-reconfiguration
- Demonstrate usability & performance on commercial applications (Maxeler, ST Microelectronics, Synelixis)

FASTER Overall Methodology



High-level Analysis & Reconfigurable System Definition (led by PDM)

Analyse each application to:

- Define the application components
  - Static part, reconfigurable modules, software part
- Provide analytical model of a reconfigurable design
  - Relate application attributes with implementation parameters
  - Estimate metrics (speed, area, power)
- Identify and optimize performance and constraints on the target reconfigurable system
  - Execution time
  - Floorplanning/Placement
  - Reconfiguration time

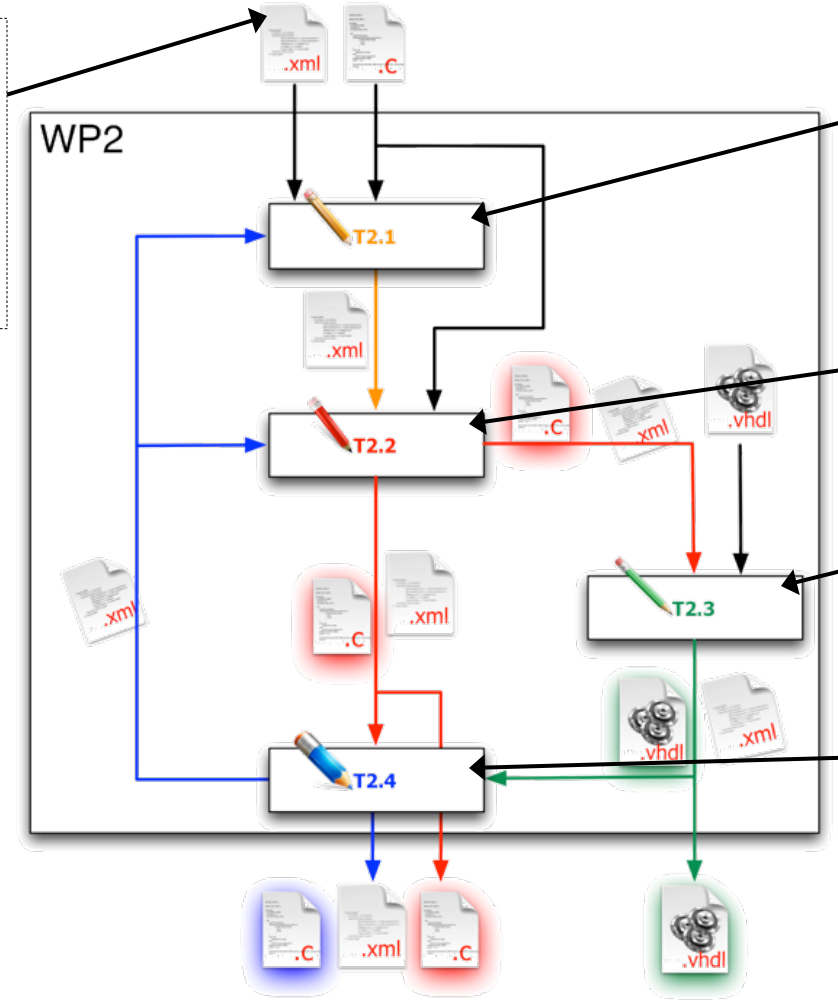
High-level Analysis & Reconfigurable System Definition – cont'd

Achieve these by identifying:

- partitioning of the input specification in HW/SW components
- implementation(s) of the modules to be realized as HW accelerators
- the most appropriate level of reconfigurability for HW components: none, micro, region based
- foorplanning constraints (size and shape)
- placement requirements
- power constraints
- a baseline schedule for application's execution

High-level Analysis & Reconfigurable System Definition – Proposed Flow

- Platform Architecture
- App Task Graph
- Performance Characteristics



High-level analysis
Estimation of metrics (power, speed, area)

App task profiling + Identification of reconfigurable cores

Optimization of app for micro-reconfiguration

Off-line scheduling and mapping into reconfigurable regions

Micro-reconfiguration (led by Gent)

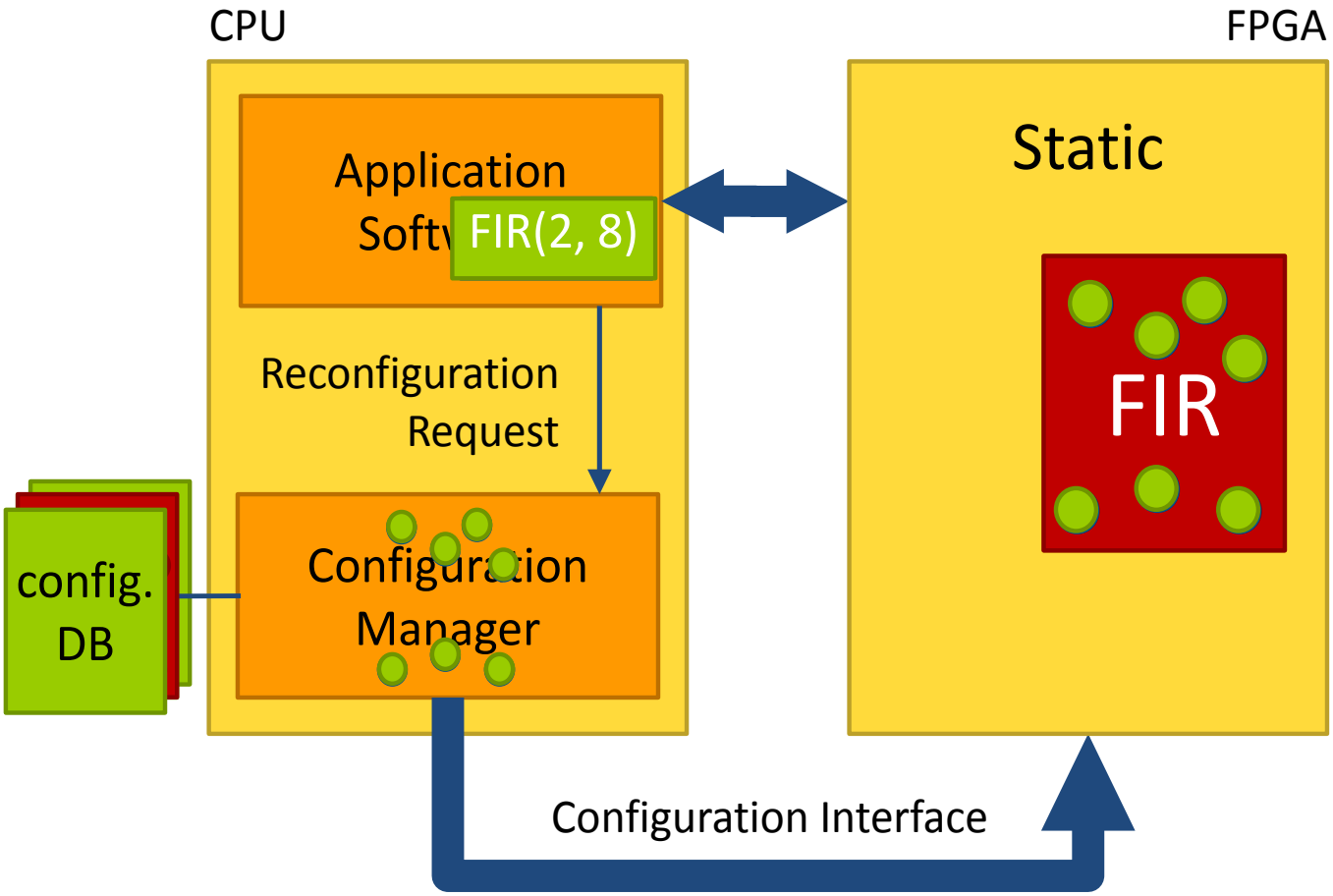
In some applications we can identify fast changing inputs vs. slow-changing "parameters"

- Parameters trigger a small-scale reconfiguration

We want to:

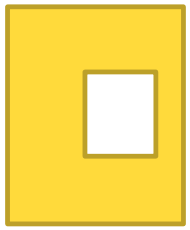
- Identify parameters
- Create bitfile with "holes"
- Parameter values => reconfiguration bits for missing "holes"
- Fine grain, faster reconfiguration time!
- Extend the idea from logic (TLUT) to wires (TCON)

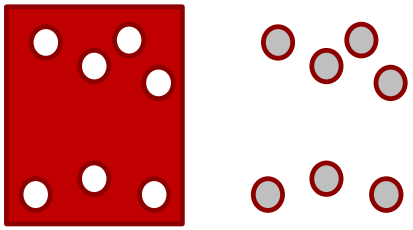
Micro-reconfiguration (led by Gent)



Micro-reconfiguration (led by Gent)

Static HDL	Param. HDL
Synthesis	Synthesis*
Tech. Mapping	Tech. Mapping*
Place & Route	Place* & Route*
Static Config.	Param. Config.





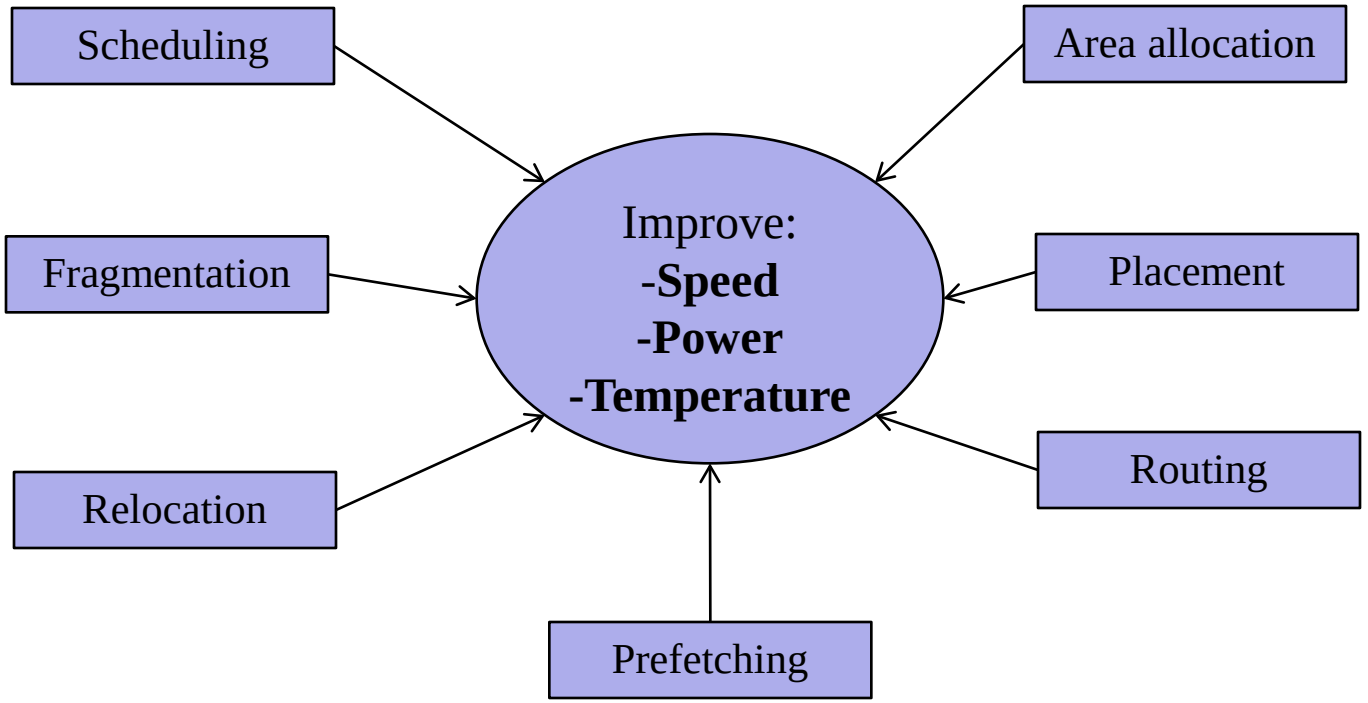
Verifying Reconfigurable Systems (led by Imperial)

- Study design validation approaches: simulation, emulation and formal verification
- Extend symbolic simulation to dynamic aspects of reconfigurable design
- In some cases static approaches may not be able to verify the entire RC system => will use run-time verification. Address and minimize impact on:
  - Speed, area and power
  - Light-weight architectural support

Run-time System (Chalmers/FORTH)

- Provide support for partial & dynamic reconfiguration
  - Extend the OS capabilities
  - Seamless, transparent easily integrated into the existing system
  - Handle efficient on-line scheduling and placement of task modules
- Evaluate reconfiguration overhead
- Propose advanced mechanisms to support
  - Scheduling
  - Relocation
  - Fragmentation = f(relocation, scheduling)
  - Area allocation
- Bottom-line: Extent the flexibility of run-time support

FASTER Run-time System



Demonstration and Use

- Demonstrate the effectiveness of the FASTER tool-chain with three complex applications from different application domains and on commercial platforms:
  - (a) Reverse Time Migration (RTM), a computational seismography algorithm (**Maxeler**)
  - (b) Global Illumination and Image Analysis (**ST**), and
  - (c) a Network Intrusion Detection System (**Synelixis**)
- Evaluate the FASTER tool flow on designer productivity in the design and verification process.
- Metrics: application speed, cost, and power consumption.

Expected Results & Conclusions

FASTER is a focused project that builds on combined partner expertise as well as on past research work & projects

In the context of this project we hope to demonstrate:

- 20% productivity improvement in implementation and verification of dynamically changing systems
- 50% total ownership cost reduction for NIDS and RTM systems
- 2x performance improvement under power constraints for Global Illumination and Image Analysis application

Challenges & Opportunities

Tool support for analysis & system definition

Specification of changing system(s)

Reconfigurable granularity: influenced by (influences???) tools and applications

Architectural support for reconfiguration (vendor?)

Metrics: include design effort/time, total ownership cost