

CS6813: Research Seminar on Reconfigurable Hardware

Lockwood : Spring 2003

Review of: Infrastructure for Design and Management of Relocatable Tasks in a Heterogeneous Reconfigurable System-on-Chip

•Paper by:

- J-Y. Mignolet, V. Nollet, P. Coene, D. Verkest, S. Vernalde, R. Lauwereins (IMEC – Belgium)
- Design, Automation, and Test in Europe (DATE), 2003

•On-line as:

–http://www.imec.be/design/pdf/reconfig/date_03_infrastructure.pdf

•Reviewed by: Chris Neely

Paper Objectives

- Describes their architecture for using FPGAs as multimedia accelerators for embedded systems.
- Presents a performance evaluation of their HW/SW M-JPEG decoder.
- Attempts to synergize OS research for reconfigurable computing.
- IMEC
 - **IMEC** (Interuniversity **M**icro**E**lectronics **C**enter) is an independent research center based in Europe. Research is conducted in microelectronics, nanotechnology, enabling design methods and technologies for ICT systems.
- References:
 - (2) FCCM '97 and '00
 - (2) Int'l. Conf. Engineering Reconfig. Sys. and Arch. Jan. '02 & Jun. '02
 - (1) FPL'00
 - (1) Tech report
 - (1) CODES'01
 - (1) IEEE Transactions on VLSI
 - (1) MAPLD -- JBits paper (Guccione)

Networked Portable Multimedia Device

- **Watching a movie**
 - While sitting on the train, Mr. Smith feels like watching a movie. He turns on his portable device and picks a movie from the network, or a previously downloaded one. He connects his set of earphones, leans back and enjoys the show.
- **Playing a game**
 - Some time later, one of the scenes bores him. Mr. Smith decides to play a 3D game. However, he doesn't want to miss out on any of the action. Ready to switch right back to the movie, he places it in a corner of the screen when he starts playing.



<http://www.imec.be/design/reconfig/scenario.shtml>

New Class of Multimedia Accelerators (p.1)

- **Motivation**
 - Multimedia applications are usually computationally intensive and have much parallelism.
 - Networked portable multimedia devices are usually computationally limited. FPGAs can be used to increase computation ability with low power consumption.
 - 3-D applications and video decoders have different architectural requirements.
 - So, make flexible multimedia accelerators using FPGAs and dynamically reprogram the architecture.
 - Multimedia applications are constantly upgraded.
 - The HW architecture is made reprogrammable.

HW/SW Multitasking Platform (p. 2):

- “The ability to reschedule a task either in hardware or software will be an important asset in a reconfigurable SoC”
 - Software-like tasks can be spawned, deleted and pre-empted in HW (as is done in software)
 - Downloaded the architecture that best fits the running applications
 - *“... first to our knowledge that HW/SW multitasking is addressed in such a way that the OS can relocate tasks in either HW or SW”*
 - The location or relocated tasks shouldn't affect how other tasks are communicating with it.

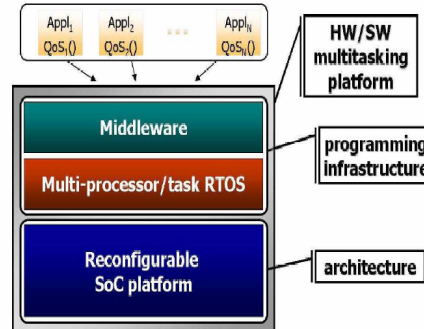


Figure 1. Our research activity

Communication Scheme (p.3)

- Reconfigurable Fabric
 - FPGA Chip is partitioned into tiles, which will fit one OS task per tile
 - Trying to determine a useful level of granularity
 - Fine-Grain multitasking works well for complex SoCs [Marescaux et al, FPL 2002]
 - Course-Grain provides an easier level for SW/HW abstraction
 - “Soft” interconnection networks (implemented using the reconfigurable fabric) for passing messages between tasks
 - Meant to be fixed interfaces between tiles; to be replaced using ASIC technology.
 - Messages are same format for HW and SW
 - Research goal to create efficient interconnection network

Heterogeneous Arrays

- OS4RS = OS for reconfigurable systems
 - Primary function is to manage/schedule tasks.
 - OS4RS controls the way messages are routed in the interconnection network via hardware task routing tables. Secondary function is to update the routing tables.
 - Implements a hardware abstraction layer (HAL)
 - developer will not need to know what specific resources their applications will use
 - Task rescheduling is complex problem– like how do you determine state equivalence between a software and hardware task?
 - Currently runs on a port of Linux for the iPAQ. “ported the real-time services to it”

Routing Resources (p.3)

- Middleware layer – platform abstraction and QoS aware rescheduling of tasks ; partitions applications into tasks using QoS considerations
 - They want middleware to function like a virtual machine and parse unified code that can be either software or hardware
 - Small footprint is desired
 - Current vision is to provide uniform behavior for hardware and software

Communication Architecture

- HW/SW Multiproc. Arch.
 - Run-time reconfigurable hardware blocks
 - Coupled with instruction set processor (ISP)
- Messages passed between P3 and PC pass the HAL
- Between PA and PB messages do not pass the HAL, they use packet-switched interconnection

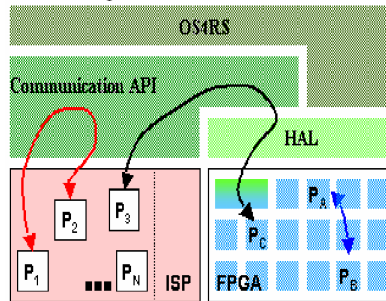


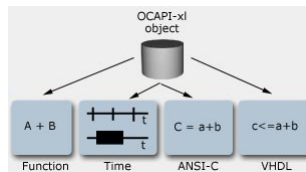
Figure 2: Message passing between tasks.

More Communication Details (p.3)

- Packet-switched interconnection network solves same OS issues listed by [Diessel and Wigley]
 - task placement
 - location independence
 - routing
 - inter-task communication
- Constraining their designs to tile regions to simplify the management software to do partial reconfiguration (tradeoff between area and run-time overhead)
 - At design time, task placement is determined using place and route tools → irregular task footprint
 - However, they want the management software to arrange tasks in the reconfigurable fabric at run-time
 - If the task shape is irregular, then the management software needs to run a fitting algorithm
- Currently they have a partial bitstream for every tile. Better alternative would be to use JBits for a single bitstream with run-time reconfiguration.

Object-oriented design environment (p.3)

- System Level modeling with a high level of abstraction. (C/C++ Style code)
- More information about OCAPI at this site:
 - http://www.imec.be/design/design/ocapi_publications.shtml
- Represents application as communicating threads and semaphores. Automatic code generation is done for both hardware and software.
- Gives uniform behavior for both SW and HW implementations



Heterogeneous Context Switching Issues

- Spatial context switches vs. time based context switches
- How do you guarantee run-time availability of hardware tiles?
Improving context switch latency and reconfiguration time.
- SW state equivalence to HW task state?
- What is the right level of abstraction for a task state?
- Targeting a single code stream that mixes configuration bitstream w/ SW instructions
- Context switch points should be low overhead

Task Switching

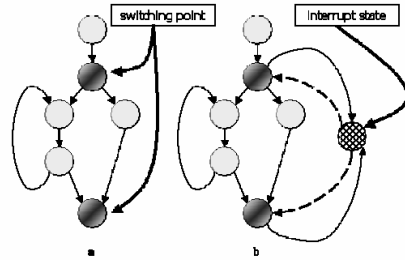


Figure 3: Relocatable task

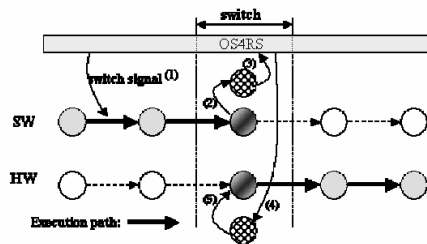


Figure 4: Task switching: from software to hardware.

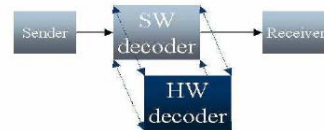


Figure 6: Relocatable decoder

Gecko Demonstrator (p.5)

- **Hardware**
 - Xilinx Virtex 2 XC2V6000 (6M gate FPGA)
 - iPAQ (Intel StrongARM 206MHz) connected via expansion bus
- **Application**
 - Motion JPEG frame decoder application – send thread passes coded frames one by one to the decoder thread
 - Decode thread sends the blocks to a receive thread, which displays the images



Figure 5. The T-ReCS Gecko demonstrator

<http://www.imec.be/design/reconfig/gecko.shtml>

Relocatable M-JPEG Decoder

- Software Implementation
 - 6 fps (95% CPU load)
- Hardware Implementations
 - Fixed Standard Huffman Tables
 - 9570 LUTS (13%)
 - 40 MHz
 - 23 fps
 - Huffman Table Optimization Support
 - 15,901 LUTS (21%)
 - 40 MHz
 - 23 fps
 - DualPort RAM Interface to iPAQ limited performance
 - DPRAM accessed at 20 MHz, FPGA-CPU memory bottleneck
 - iPAQ memory access rate is 103 MHz
 - Added wait states

Conclusions (p.6)

- Presented a method for handling context switches between software and reprogrammable hardware
- Reconfigurable systems can achieve greater performance than software.
- Run-Time Reconfiguration:
 - Allows Virtual hardware
 - Expands logic size though device reuse

Comments

- Paper appears to build upon prior work in OS research for reconfigurable systems, as well as making their own contributions.
- Idea is similar to reconfigurable coprocessor for accelerating kernel loops. They are trying to exploit task/thread level parallelism.
- More info at:
 - <http://www.imec.be/design/reconfig/publications.shtml>

OCAPI-xl Reference

- G. Vanmeerbeeck P. Schaumont S. Vernalde M. Engels I. Bolsensvery
- http://www.imec.be/design/pdf/Ocapi/codes_01_hardware.pdf

OCAPI-xl is a C++ class library for true unified Hardware / Software modeling that allows taking partitioning decisions anywhere in the design flow was presented. This is achieved by describing the system model in a refinable, a target implementation independent way. Communication is expressed using a software-like approach, using messages, semaphores and shared variables. To keep processes target independent, it uses a lazy definition approach, or generates use-based interfaces, depending on the chosen implementation target. To model external system components, and to get system ports to these components, a powerful FLI method was introduced. It also provides code generation possibilities to get synthesizable code out of your system model both for hardware and software.

As a proof of concept for the OCAPI-xl methodology, a standalone NetCam was designed implementing an interface to a digital CMOS image sensor, a GIF engine, a network layer and an interface to an off-the-shelf ethernet controller. This was achieved in only 14 man-months, thanks to our methodology.

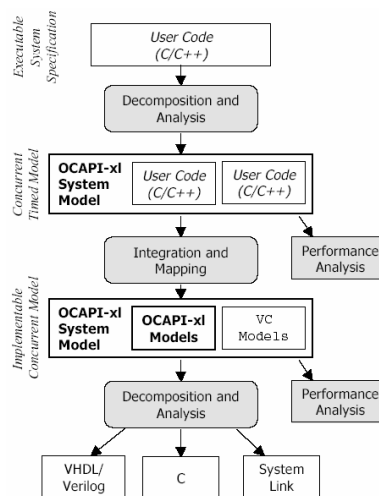


Figure 1: the OCAPI-xl design flow