National ICT Australia
Reconfigurable Computing Workshop

22 March, 2005

Meeting Notes

Table of Contents

National ICT Australia Ltd

Terry Percival
Director,Sydney Research Laboratory
Kensington

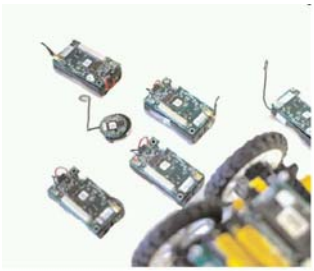
Australian Government
Department of Communications, Information Technology and the Arts
Australian Research Council

NICTA Members: ANU, UNSW, Department of State and Regional Development, BusinessACT Government

NICTA Partners: Victoria, The University of Sydney, Queensland Government, Griffith University, The University of Queensland

Mission

- *To be an enduring world class national research institute in Information and Communications Technology that generates national wealth.*



NICTA Members

- **NICTA Members**

Australian Government Department of Communications, Information Technology and the Arts Australian Research Council; BusinessACT Government; Department of State and Regional Development; UNSW

- **NICTA Partners**

QUT; Queensland Government; The University of Sydney; Griffith University; The University of Queensland; Victoria The Place To Be

Operating Pillars

Creating Australia's ICT Centre of Excellence

- Established on:
  - **Research** – Built on exceptional research talent
  - **Education** – Built on enhancing ICT education
  - **Commercialisation** – Built on consideration of use
  - **Collaboration** – Built on exceptional partnerships

Research

NICTA's research will produce significant social, environmental & economic benefits for Australia

- Central drivers of NICTA's research:
  - **Trusted Wireless Networks**
  - **From Data to Knowledge**
- Work is in progress to develop a portfolio of large scale projects under the Priority Challenges. The broad areas are:
  - ICT for Water conservation
  - Traffic Management
  - E-government
- NICTA is conducting research within 41 projects, five projects will deliver final results during 2005

Education

A nation's future is built on each generation's ability to improve social and economical conditions

- NICTA's PhD programs are building on traditional education programs
- Value-added degree programs within universities that we partner with
- Technical Broadening through extensive coursework

Our Progress

- Currently almost 100 students
- Industry experience is part of education NICTA-Telstra internship program for 21 students

Commercialisation

An intellectual property portfolio that is not used has no economic benefit

- Aim: to generate national wealth through the commercialisation of intellectual property through:
  - Flexibility in approach
  - Licensing of research and technology
  - Creation of spin-offs and joint ventures

Our Progress

- The first five provisional patents have been lodged
- Entrepreneur-in-Residence program underway

Collaboration

Some of the best ideas are born through the meeting of imaginations

- NICTA fosters research & commercialisation through an open & accessible culture that welcomes collaboration with business & technology organisations.

- NICTA collaborates with:
  - Small to Medium enterprises
  - Multinational ICT companies
  - Users of ICT
  - Researchers (national and international)

Our Progress

- IBM: Open Source
- Microsoft: Improved Web Services

The imagination driving Australia's ICT future

Using Off-The-Shelf Reconfigurable Hardware

ERTOS Program
Frank Engel
frank.engel@nicta.com.au

Australian Government
Department of Communications, Information Technology and the Arts
Australian Research Council

NICTA Members: ANU, UNSW, Department of State and Regional Development, BusinessACT
NICTA Partners: Victoria, The University of Sydney, Queensland Government, Griffith University, The University of Queensland

ERTOS Program

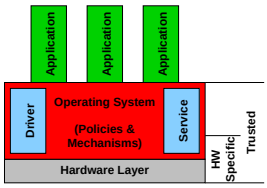
Operating System Support for Embedded Systems:

- Address typical constrains (power, size, price)
- Operating system based on micro-kernel (L4/Iguana)
- Reliability, trustworthiness
  - Modular component structure
  - Customizable (application, processor)
  - Real-time support
  - Third party code (driver/service, application)

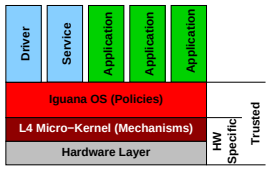
➜ *Embedded Software Framework*

ERTOS Program

Embedded Software Framework:



Monolithic Operating System Approach



ERTOS Micro-Kernel Approach

ERTOS Program
Reconfigurable Hardware

Application Driven Projects on Reconfigurable Hardware:

- Gain experience in reconfigurable SoC design
- Integrate reconfigurable HW into our embedded SW framework
- Provide OS support

Frank Engel
Background

Background:

- PhD at Dresden University of Technology, Dresden/Germany
- Vodafone Chair Mobile Communications Systems
- "Analyses and Concepts for Architectures of Application-Specific I/O-Processors"
- Focus:
  - Digital signal processing
  - Embedded processor design
  - HW implementation aspects

Frank Engel
Reconfigurable Hardware

My Research Focus:

- Operating system (OS) support for embedded processors
- Use of reconfigurable HW (FPGAs) as embedded systems
- Evaluation of commercial applications

➜ *FPGAs – peripheral device or HW function?*
➜ *Impact of reconfigurability on embedded software framework*

Projects

Current Projects:

- **Embedded Next Generation GNSS Platform**
  - FPGA-based GPS receiver platform
  - System-on-Programmable-Chip (SoPC) example
- **Algorithm and Architectural Investigation into a Real-Time Demonstrator of a New Receiver Algorithm**
  - L4/Iguana operating system on Xilinx FPGA / SoPC platform
  - Hard real-time application example

Project A

Embedded Next Generation GNSS Platform

System-on-Programmable-Chip (SoPC) Application

Project A SoPC Application

Motivation:

- GPS: Global Positioning System
- Growing interest in navigation applications
- New systems available soon
- Australia covered by at least four systems
- Local industry
- Good SoC example

Partner:

- Satellite Navigation and Positioning Group (SNAP) at UNSW

Project A SoPC Application

Objectives:

- **Research**
  - Joint initiative (ERTOS/SNAP) into FPGA based GNSS receivers
  - GPS enhancements and new signals (e.g. Galileo)
  - Case study into SoPC implementation process
- **Commercialisation**
  - IP module available to local industry and/or FPGA manufacturers
  - HW/SW framework for GPS application development
  - Design service (e.g. modifying/extending SW and signal processing HW)

Project A SoPC Application

Approach:

- **RF processing**
  - Use off-the-shelf ASIC
- **Signal processing (Altera FPGA logic)**
  - Port of commercial SW stack to soft-core processor
  - Adapt peripherals (HW & SW) to receiver architecture
- **Control processing (Altera NIOSII soft-core cpu)**
  - Design standard GPS signal processing module
  - Keep it generic for further extension

Project B

Algorithm and Architectural Investigation into a Real-Time Demonstrator of a New Receiver Algorithm

OS Support for Xilinx FPGA / SoPC Platform

PROJECT B
OS SUPPORT FOR XILINX FPGAS

Motivation:

- Mobile phone network
- Investigation into improved receiver principles
- Increased network capacity
- FPGAs often used in base stations
- Embedded operating system required

Cooperation:

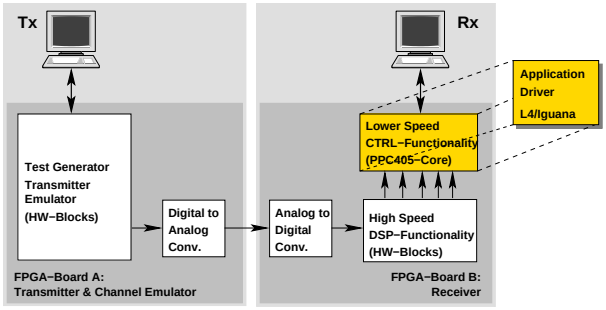
- NICTA's Wireless Signal Processing Program (WSP)

PROJECT B
OS SUPPORT FOR XILINX FPGAS

Objectives:

- **Research**
  - Port L4/Iguana OS to embedded PowerPC 405 architecture
  - Real-time issues arising from demo application
  - Support FPGA based SoC platform
- **Commercialisation**
  - Implement applications needed to run demonstrator
  - Integrate L4/Iguana OS into Xilinx FPGA tools
  - Get (local) network providers interested

PROJECT B
OS SUPPORT FOR XILINX FPGAS



Thanks

Reconfigurable Computing Projects II

Oliver Diessel

Overview

1. What is Reconfigurable Computing?
   - Static versus dynamic view
   - Examples of dynamically reconfigurable systems
2. Design flows for Reconfigurable Computing
3. Research projects
   I. Managing dynamically reconfigurable systems
   II. Modelling dynamically reconfigurable systems

1. What is Reconfigurable Computing?

- Use of reconfigurable devices to achieve a benefit over processor-based computing and/or custom devices
  - Currently involves FPGAs implementing algorithms as circuits
  - Look for enhanced performance, reduced power, reduced part count, greater reliability, greater flexibility
  - Small, but expanding niche; conditions most favourable in applications/markets with one or more of following characteristics:
    - Prototyping
    - Integration
    - Rapid development in protocols, standards, algorithms, architectures
    - Small to medium volume

Static versus Dynamic Reconfiguration

- Products are almost always statically configured
  - Underutilizes device capabilities
- How much do you want to integrate?
  - Can your system be partitioned into mutually exclusive time-multiplexed functions?
    - Do you need to provide additional hardware for these?
    - Can your desired functionality be provided as a single configuration?
- How rapidly do you want your system to respond to changes in its
  - Environment
  - Requirements

  i.e. how flexible, adaptive, or robust does your application need to be?
    - Is everything fixed at design time?

Examples

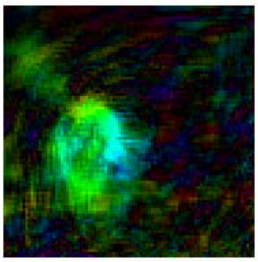
- Time-multiplexed application
  - Real-time Optical Flow
- Adaptive system
  - System responding to change in requirements/environment

Real-time optical flow computation

- Implement real-time optical flow algorithms using an FPGA
- Why?
  - Prototype hardware-based techniques
  - Faster processing = faster movement

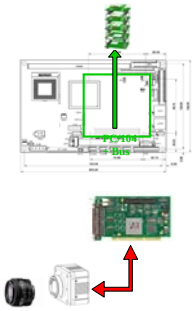
Optical Flow

- Determines velocity of pixels from frame to frame
  ⇒ Closer objects have higher relative velocity



System architecture

- Camera
  - 567x378 @ 27.4 fps
- Framegrabber
- Motherboard
  - P4-M 2.6GHz
  - 1024MB DDR 266
- BenNUEY board
- VirtexII XC2V6000

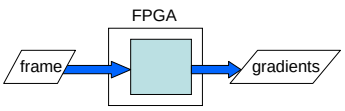


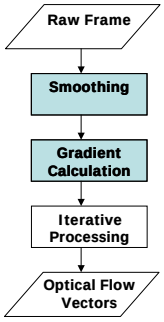
Overview of algorithm & mapping

Core iterative operation:

$$u_i^{k+1} = \frac{\sum_{j \in N(i)} u_j^k - \frac{1}{\alpha}(I_{x,y,i} \cdot v_i^k + I_{x,t,i})}{|N(i)| + \frac{1}{\alpha} I_{y,y,i}}$$

$$v_i^{k+1} = \ldots$$



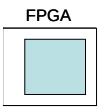


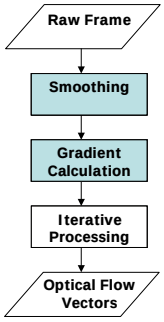
Overview of algorithm & mapping

Core iterative operation:

$$u_i^{k+1} = \frac{\sum_{j \in N(i)} u_j^k - \frac{1}{\alpha}(I_{x,y,i} \cdot v_i^k + I_{x,t,i})}{|N(i)| + \frac{1}{\alpha} I_{y,y,i}}$$

$$v_i^{k+1} = \ldots$$





Example: Adaptive System

- Change in requirements:
  - Optical flow → Optical flow + template matching
- Change in environment
  - Outdoor navigation → navigate indoors
- Fault tolerance
  - Adapt control equations
  - Share additional load

Example: Adaptive System

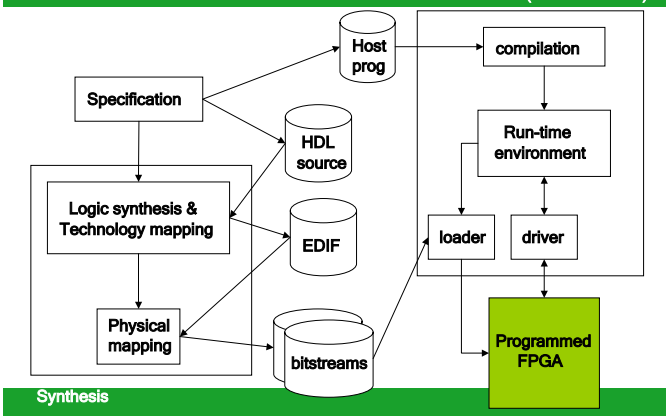
- Change in requirements:
  - Optical flow → Optical flow + template matching
- Change in environment
  - Outdoor navigation → navigate indoors
- Fault tolerance
  - Adapt control equations
  - Share additional load

How does one design such systems?

2. Design flows for Reconfigurable Computing

- Our contention is that design of statically configured systems is difficult
  - Not well supported
- Design of dynamically reconfigurable systems is harder
  - Almost no support

FPGA design flow (loosely-coupled system)



Desirable flows for Reconfigurable Computing

- Static:
  - Support high-level and component modelling using multiple modalities
  - Guide partitioning through understanding of tradeoffs
    - Hardware & software components, interfaces, memory, buses, power, cost
  - Efficient mappings
  - Support co-simulation and co-verification of integrated subsystems
  - Rapid prototyping
- Dynamic:
  - As above, PLUS
    - Model dynamism
    - Multiple partitions
      - Active set is event dependent
      - Optimize over all partitions
    - System management
      - Dynamic system

3. Research Projects

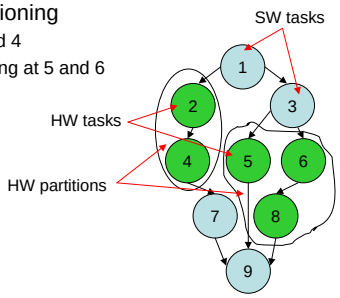
Research Project I

Managing Dynamic FPGA Task Sets

Oliver Diessel, Shannon Koh, Usama Malik, UNSW
Gordon Brebner, Xilinx Research Labs

Model definition: Tasks

- Task graph partitioning
  - Pipeline for 2 and 4
  - Parallel processing at 5 and 6



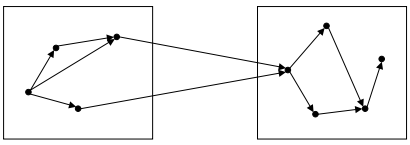
Partitioning

P1: Deciding which part of an application to implement in hardware

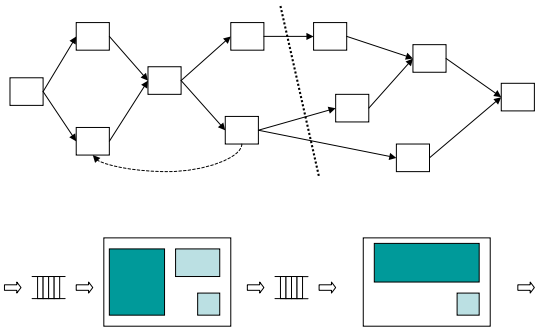
P2: Deciding how to fit a task graph to the available hardware
- Distinguish between spatial and temporal partitioning

Spatial partitioning

- Use min-cut/max-flow algorithm to minimize inter-chip or inter-partition communications
- Use feedback from placement and routing (allocation) algorithms



Temporal partitioning



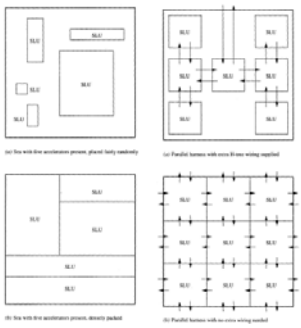
Implementation model: Swappable Logic Unit

2 models:

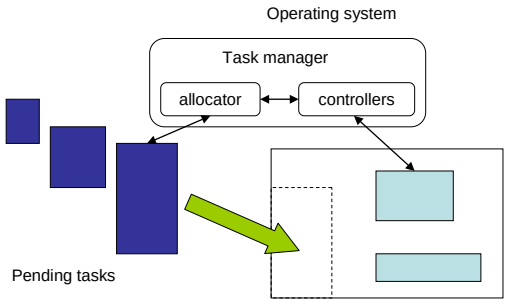
1. Sea of accelerators:
   - Logic flexibility
   - Performance less compromised
   - Potential for high utilisation
   - Problems with fragmentation
   - Problems routing
2. Parallel wiring harness
   - Ease of placement
   - Known delays
   - Lower performance
   - Reduced utilisation

[Brebner, 1996]

Sea of accelerators | Parallel wiring harness



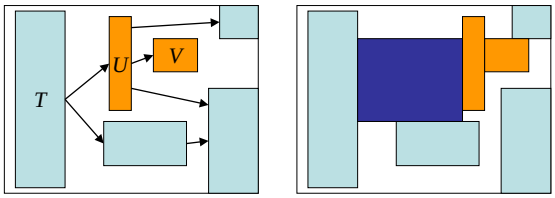
Managing variable partition sizes by partial rearrangement



Space-shared FPGA

Ordered compaction

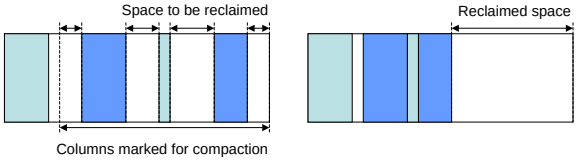
- "Slide" tasks along rows of FPGA cells to free space for incoming task



[Diessel,1997]

Logic-based compaction

- Ordered compaction frees required space by squeezing a subset of the tasks together
- Requires following enhancements:
  - marking method: shorten pattern as space found
  - compaction method: reloading usually proposed

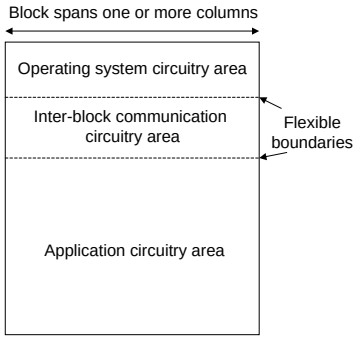


Before compaction **After compaction to left**

[Brebner & Diessel, 2001]

The 1D model

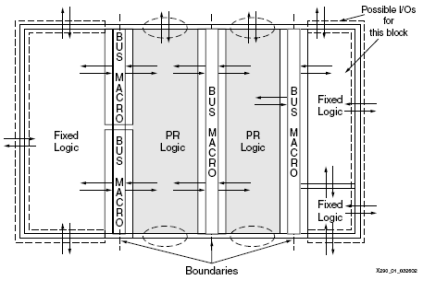
- Module occupies full array height & variable width
- Module's logic resource is composed of three parts:
  - area for system functions
  - communications area
  - application circuitry
- Need to keep overheads small



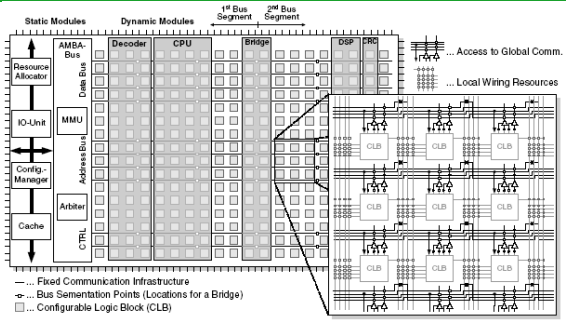
Xilinx Task-Based Reconfiguration

Advantages:
- Commercially available model
- Realisable

Disavantages:
- No dynamic sizing and placement
- Size and location in multiples of 4
- Bus macros must be used
- No parallel communication on same row



1 Dimensional Task Model



[Kalte *et al.*, 2004]

Research Project II

Towards High-Level Specification, Synthesis and Virtualization of Programmable Logic Designs

Oliver Diessel, Usama Malik, Keith So, UNSW
Jérémie Detrey, ENS-Lyon
George Milne, UWA

Big challenges facing RC

1. How best to exploit reconfigurable resources
   - still largely a "black art"
2. How to express algorithms as suitable digital systems
   - doing so linguistically
3. How to map these specifications to available hardware resources
   - having this step automated

Our goal

- To simplify the specification of reconfigurable systems
- To automate the generation of dynamically reconfigurable systems

Our methodology

- Model dynamic reconfiguration at the hardware level, i.e. capture capabilities of the hardware
- Develop compilation techniques that target these capabilities
- Develop syntactic structures that can be embedded into appropriate languages

Modelling Reconfiguration using a *Process Algebra*

Advantages:

- Natural (simple, yet powerful) expression of parallelism & synchronisation
- Verifiability

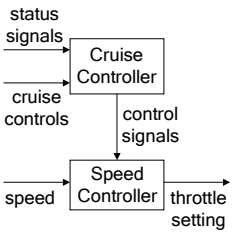
Disadvantages:

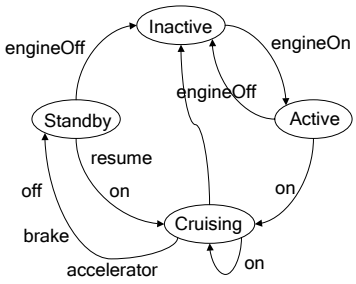
- Not well known by the FPGA community
- Existing PAs need to be enhanced

Progress to date

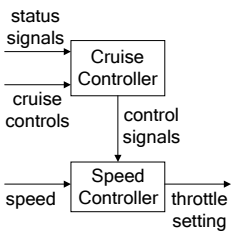
- Process descriptions mapped to hardware structures via syntax-directed translation
  - Process behaviours implemented as FSMs in compact logic blocks
  - Hierarchical design achieved through event abstraction and local process synchronisation
- Interpret specifications at run time, and dynamically reconfigure process logic to cope with limited chip area

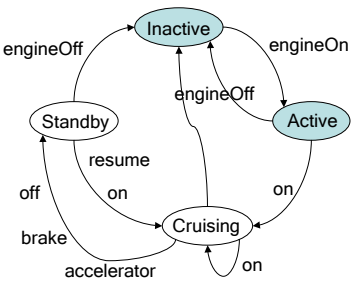
Example: Car Cruise Control



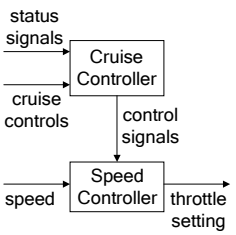


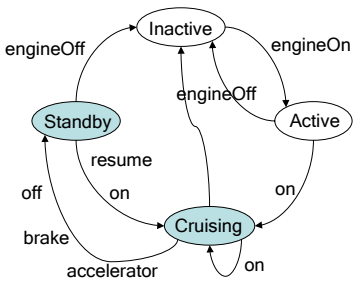
Example: Initial configuration



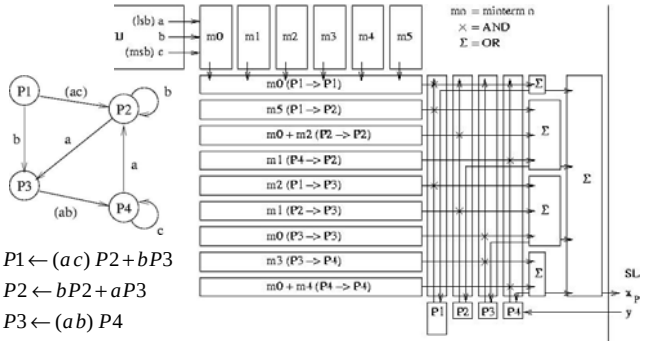


Example: Final configuration





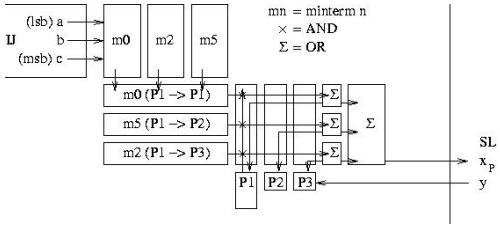
Process circuit implementation



$$P1 \leftarrow (a\,c)\,P2 + bP3$$
$$P2 \leftarrow bP2 + aP3$$
$$P3 \leftarrow (a\,b)\,P4$$
$$P4 \leftarrow cP4 + aP2$$

Modelling hardware virtualisation

- Suppose the array area for process P can only accommodate the behaviour for state P1
- To determine which transition occurred, boundary state registers for P2 and P3 are needed as well



Modelling dynamic reconfiguration

- Model 2 facets of dynamic systems
  1. Behavioural change
     - Change in function as mediated by change in logic
  2. Structural change
     - Change in composition as mediated by change in interconnection
- In a process algebra
  - Behavioural change equates to process evolution – transition from one state to another
  - Structural change equates to dynamic composition – composition guarded by some event

[Milne, 1999]

Applications

- Implementing time-varying control strategies
  - Mode switching
- Adjusting to available resources
  - Multi-tasking
  - Graceful degradation
- Coping with dynamic updates
  - User customizes system by selecting web-accessible modules

2 Reconfigurable Computing Research Directions

2.1 Dynamically Reconfigurable Systems Design

We're primarily interested in the design of dynamic reconfigurable computing systems

- View static RC as a special case

Research issues of interest to us:

1) Expressing & modelling dynamism
2) Effective spatial & temporal partitioning
3) Optimisation at various levels of design abstraction
4) ERTOS framework
    a) Support for embedded processors
    b) Support for dynamic tasks

2.2 Static Reconfigurable Systems Design

Statically configured systems are a special case of dynamically reconfigured systems

- Need to understand static cases to develop techniques for solving dynamic problems

Seek collaborative partners to explore aspects of better reconfigurable systems design

- Might start with static research & design projects

3 Discussion

3.1 How do you use FPGAs?

3.1.1 Robert Lang — Agere Systems Australia

- Application is chip development for mobile handsets.
- Use FPGAs for emulation.
- Don't use reconfigurable hardware in handsets, use ASICs.
  - Cost is extremely important.
- Desire to get ASICs design right first time; so emulation very important. Use emulation system designed by University of Newcastle, NSW.
- Didn't care about the performance during emulation. For example, sometimes 100 times slower in emulation, but can run emulation over a long period of time, e.g. overnight.
- Current system works, but we never have enough capacity.

3.1.2 Tony Proudfoot — G2 Microsystems

- Final application is 802.11b wireless communications.
- Use FPGAs for emulation.
- Can't fit whole digital design into one FPGA.
- More spatial partitioning.

3.1.3 Robert Dowle — SERCEL Australia

- Use in communications in geophysical equipment used for oil exploration.
- Marine acoustic, seismic data. Hostile environment. Long battery life. Power consumption is important!
- Reconfigure annually, i.e. firmware upgrades.
- On sea-bed to last two months. Surveillance device. Logging.
  - Devices rest on ocean floor at a depth up to 3 kilometres.
- Has a hard disk. DSP chip runs hard drive.
- Want less devices and less power. Processor architecture will always be too power hungry.
  - Current system draws 1.5 W. Want power down to 0.25 W.
  - Get 2 weeks operation or 1.5 months with large battery pack.
- Not a high-speed application.
- Applications today are not space and power critical.
- New area of autonomous systems, concerns become like mobile phones.
- Don't want to use ASICs because will only make 100s not 1000s.

3.1.4 Robert Vesetas — Thales Underwater Systems

- Application area is defence.
  - FPGA for telemetry and I/O pieces.
  - 3D ultrasonic imaging. 30 FPGAs. System requires Tera-operations/second.

- No real need for dynamic reconfiguration.
  - Often throw more hardware resources at a problem, than reconfigure.
- Low volume production; number of systems is in the 10s.
- Beam forming, DSP.
- Runs for hours. Reconfigurable.
- Maintaining skill sets is hard. More C programmers around.
- Often develop algorithms in MATLAB.
- Will keep watching brief on the reconfigurable computing technology.
- Push clock rates.

3.1.5 David Bettison — BAE Systems

- Have many static FPGA projects.
- Reconfigure rarely; for firmware upgrades.
- All use the same tool, from Mentor graphics.
- Current synthesis tools work in 1 hour, used to take overnight.
- Don't need the biggest and fastest designs.
- Custom designed board.
- Have in-house experts. Systems engineers. Architecture designs, software, mechanical do box design.
- Systems guys use MATLAB.
- Often use Mentor to simulate hardware.
- Defence clients. Product development is long. Products may take 10 years.
- Are happy with production processes.
- Quite low production. FPGAs are the choice.
- Often have limited space — can't throw in extra cards. But would use bigger FPGA.
- Future: Electronic warfare payloads in UAV.
  - May reconfigure every 10 minutes.
  - Requirements are: low weight and not too much power consumption.

3.1.6 Laurence Lau — ACMC@UQ

- Was in University of Queensland high performance group.
- Setting up Intellectual Property advisory in Hong Kong.
- Found client, with open source, reconfigurable camera.
  - Core problem was *data fusion*.
  - Laurence Lau could see commercial applications. But guy was one man band.
- Main interest is in *standardised tools*.

3.1.7 Steven Duvall — Intel Australia

- Now large number of computational problems.
- Internally to Intel: FPGAs are good for chip development.

3.2 Static FPGA design issues?

3.2.1 Robert Dowle — SERCEL Australia

- Static design is not a problem.

3.3 Discussion regarding access/availability of good FPGA designers

3.3.1 Robert Dowle — SERCEL Australia

- Robert used to work for Thales — getting good FGPA designers was hard.
- Consultants:
  - Consultants don't want to transfer skills.
  - Quality assurance problems when using consultants. Gurus differ when they come to analyse a problem.
- Had experience of inexperienced FPGA designers. One poor design expected to use 95% utilisation, 105% of clock.
- Would like NICTA people embedded in industry.
- Companies wary about putting staff into academia. Culture gulf. They lose focus and their 'industry edge'.

3.3.2 Robert Lang — Agere Systems Australia

- Need one good manager of FPGA team — one good in-house expert. Can't just use consultants.

3.3.3 Robert Vesetas — Thales Underwater Systems

- Started with zero people.
- Employed two or three FPGA experts. They are all gone now.

3.3.4 David Bettison — BAE Systems

- Often FPGA experts have nowhere to go within organisation once job is done.
- BAE is a project-based company.

3.4 Is it possible to get tools to enable a software engineer to do FPGA design?

3.4.1 Robert Vesetas — Thales Underwater Systems

- Can use a library approach.

3.5 Are you using the processor cores inside the FPGAs (hard/soft core)?

3.5.1 Robert Vesetas — Thales Underwater Systems

- Using Power PC (hard core) in Xilinx VirtexPro chips, but no soft core.

3.5.2 Robert Dowle — SERCEL Australia

- Conflict of Power PC embedded in FPGA and RC architecture?
- Bit slice, transputers?
- Oliver Diessel: huge granularity mismatch.

3.6 What is the future need in reconfigurable computing?

3.6.1 Steven Duvall — Intel Australia

- Convergence of fabrics.
  - Multi-core architecture.
  - Different fabrics, i.e. bit level, word level.
- Main Challenge: Want to make the programmer's development process look more like software than hardware.
- Three companies he knows of access an FPGA via an API.

3.7 If you were boss what would you have NICTA do?

3.7.1 David Bettison — BAE Systems

- BAE Want high-level tools: where do you partition. How to do it?

3.7.2 Laurence Lau — ACMC@UQ

- Want FPGA solutions accessible as APIs. Data fusion needs enough meta data around. Want new APIs flexible.

3.7.3 Robert Vesetas — Thales Underwater Systems

- Our guys use Intel library. Can do same in FPGA?
- Want FPGAs API library. Program in terms of primitives get away from RTL design.
- Acceleration ability to get development turn around up.
- Keep data in FPGA (avoid need to save states into external buffer when reconfiguring).

3.7.4 Robert Dowle — SERCEL Australia

- Dynamic side. Interesting, curious.
  - Sample application: swap networks between 3G and 2G.

- High-level language side. More interested in this.
  - Whole group doing simulation modelling. Runs 4-5 weeks on MATLAB to create dB curves.

3.7.5 Tony Proudfoot — G2 Microsystems

- Create MATLAB to Verilog converter.
- Wants dynamic MMX instructions.

3.8 What does industry want from NICTA?

3.8.1 Tony Proudfoot — G2 Microsystems

- Wants application examples of RC.
- Can it be used to achieve goal in signal processing.

4 Observations

4.1 General Observations

1) Attendees did not have an immediate application for reconfigurable computing — but there was considerable interest.
2) Attendees want to know how to use FPGAs as dynamically reconfigurable devices. They are interested to know; what techniques to use, what infrastructure to avail themselves of, and what performance to expect.
3) Some potential future applications of reconfigurable computing are envisaged.
4) Current users fall into two main categories:
   a) Designers using FPGA to integrate functionality.
   b) Testers using FPGAs to accelerate circuit verification.
5) There appeared to be agreement that design of static systems using FPGAs is well understood, even if it is difficult finding and keeping appropriately skilled designers.
6) Most users do not have size/space constraints on their use of FPGAs. Therefore, when more power is required, a more powerful FPGA can be substituted or extra circuitry added.
7) Most users do not have price constraints on their use of FPGAs.
8) There is little use being made of the 'system on a chip' capabilities of FPGAs with embedded processor cores.

4.2 Where to from here?

1) Pursue research directions identified.
2) Follow up with potential collaborations to develop FPGA-based (dynamically) reconfigurable systems.
3) Present (annual?) technical workshops including detailed design and analysis of real applications.

4.3 NICTA Research

NICTA's current research represents a 'bottom up' approach to solving the reconfigurable computing challenge. That is, current efforts aim to solve *generic* reconfigurable computing problems at the infrastructure level as identified in Section 2.1, point 4). This approach is in harmony with the orientation of NICTA's larger ERTOS programme. However, while valuable in itself, it should be questioned whether this research is likely to meet the needs of users, and whether it will have the impact expected of our organisation.

An alternative approach is to seek to research issues in a 'top down' manner. This approach attempts to tackle the problems listed in Section 2.1 in numerical order to provide as a goal a design flow that targets dynamic architectures. Such an approach, driven by user requirements, is more likely to solve real problems and make a substantial impact if successful. However, it is also more risky. It might require a realignment of NICTA's reconfigurable computing research efforts in the form of a cross-program project involving active contribution from the fields of computer design, software engineering, algorithms, compilers, formal methods, and operating systems. Are there grounds to consider moving to this approach? Further interaction with companies may provide the impetus and the necessary opportunities for collaboration to reconsider our current strategy.

Appendix

A Glossary

API	Application Program Interface
ASIC	Application-Specific Integrated Circuit
ERTOS	Embedded, Real-Time and Operating Systems (NICTA research programme.)
FPGA	Field Programmable Gate Array
MMX	Multi-media extension
RC	Reconfigurable Computing
RTL	Register Transfer Level (VHDL)
UAV	Unmanned Aerial Vehicle
Verilog	A hardware description language similar to VHDL
VHDL	VHSIC Hardware Description Language
VHSIC	Very High-Speed Integrated Circuits

B Attendee and Interested Parties List

Attendee	**Company**
David Bettison	BAE SYSTEMS
Robert Dowle	Sercel
Steven Duvall	Intel Australia
Robert Lang	Agere Systems
Lawrence Lau	ACMC@UQ
David Levy	University of Sydney
Tony Proudfoot	G2 Microsystems
Robert Vesetas	Thales Underwater Systems Pty Ltd
Athanassios Boulis	NICTA
Oliver Diessel	NICTA
Frank Engel	NICTA
Terry Percival	NICTA
Neil Temperley	NICTA
Couldn't attend:	
Dean Jackson	Agilent Technologies
Mark Rice	DSpace
Matt Simmons	Tenix
Kandeepan Sithamparanathan	NICTA

Would like to be kept informed:	
Neil Bergmann	University of Queensland
Chris Bishop	Intellidesign
Gary Francis	Cray Australia
John Kent	