



BAE SYSTEMS

Model Based Design of Heterogeneous Processing Architectures

Ian Alston & Bob Madahar

BAE SYSTEMS

Advanced Technology Centre, Great Baddow

Content

- **Introduction**
- **System-Level Design Languages**
 - Definition
 - Example Language - Handel C
- **ESPADON Design Flow Principle**
- **Extended Design Flow**
 - Requirements Analysis
 - Definition
 - Integration with GEDAE
- **Summary**

Introduction

- **Current DSP approach**

- COTS multi-processor boards
 - parallel architectures, modular & highly scalable
- Application Specific Integrated Circuits (ASICs)
 - time critical aspects, difficult to design & implement
 - Not cost effective for low volumes?

- **How to deal with increased complexity of DSP Systems?**

- Higher Performance, low latency, 'hard' real-time Systems
- Dynamic Multi-mode, multi-function Systems

- **Leverage rapid advances in microelectronics** (~1BT/device by 05!)

- Reconfigurable/reprogrammable FPGAs, SoCs, NoCs,...
- Increasing Performance/Cost ratios - *replacement for ASICs*

- **System Design lagging HW advances**

- **NEED** high level languages & tools for rapid development

System-Level Design Languages

• Definition

- Definition of “system” is itself vague
 - One definition of System-level Design is:
 - “the design of a product at the conceptual level, including hardware/software co-design, design partitioning and specification writing” (Gartner Dataquest)
- Features of an SLD language:
 - design decisions at high level of abstraction
 - integrated approach
 - executable specification
 - *a functional model*
 - » *allows the functional behaviour of the system to be investigated*
 - links to implementation

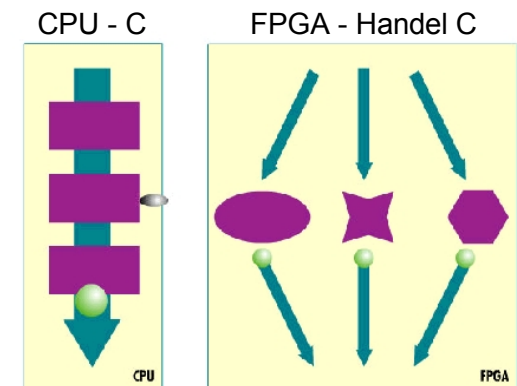
System-Level Design Languages (2)

Two basic approaches to providing an SLD language:

- **Extending the syntax of existing language (e.g. Handel C)**
 - Requires development of separate compilers
- **Providing “class” libraries for an extensible language (e.g. SystemC)**
 - additional libraries provided to model hardware aspects
 - standard compilers & development tools can be used
 - simulation of executable specification only
- **Both require conversion of executable specification into:**
 - executable software for chosen target processor
 - hardware implementation possibly via an HDL
 - interface synthesis - a means to link the S/W & H/W

Example Language - Handel C

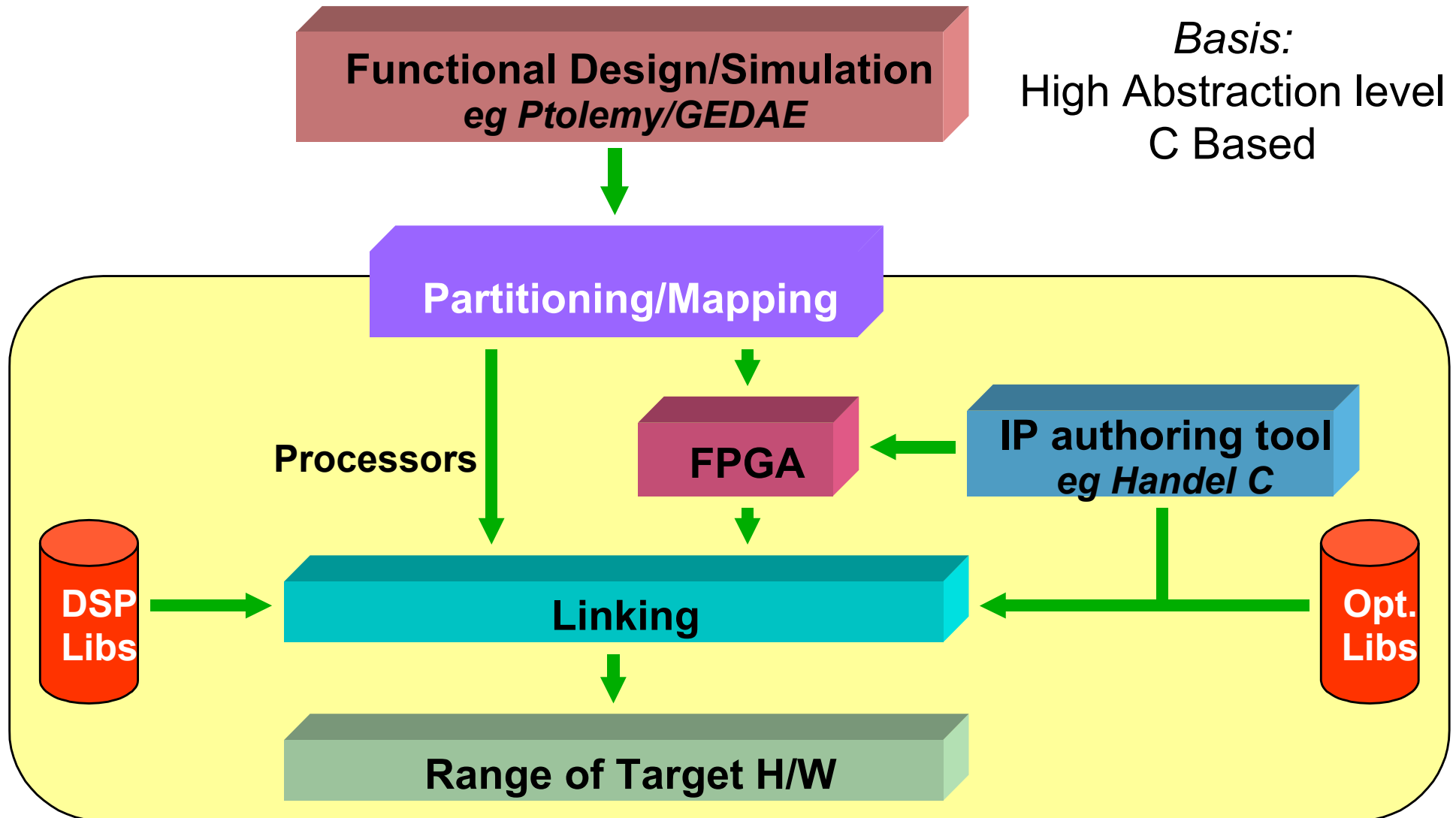
- **Based on:**
 - communicating sequential processes (Hoare) & OCCAM language
- **Aim:**
 - to compile high level algorithms directly into gate level hardware
 - using EDIF - Electronic data-interchange format
- **Uses syntax of conventional C plus:**
 - parallel constructs - *par statement*
 - I/O constructs - *channel & interface statements*
 - integer types - *bit width specifiers*
- **Supporting development environment (DK1)**
 - complexity & performance estimates
 - supports major FPGA vendors (Xilinx & Altera)



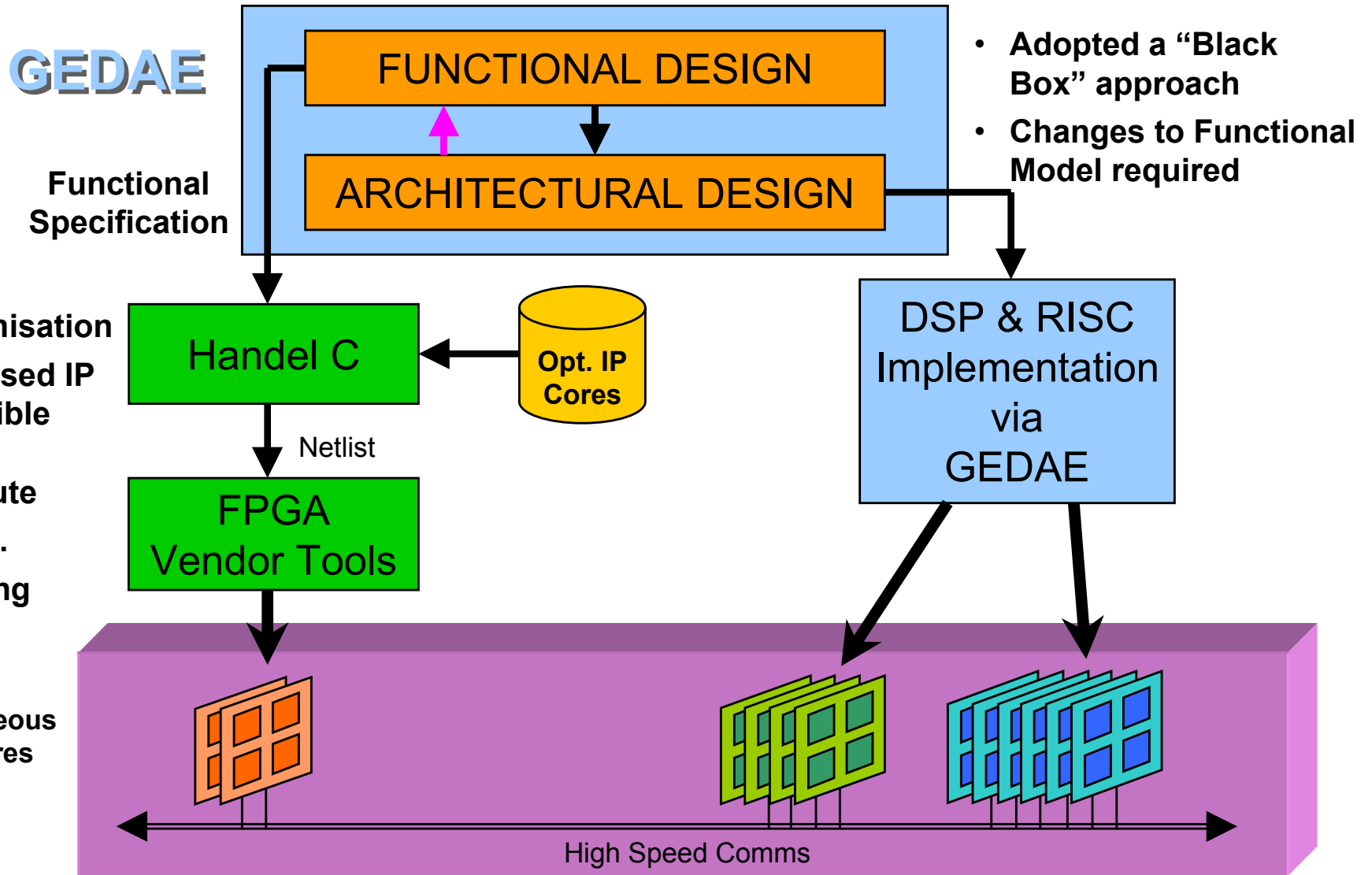
ESPADON Design Flow Principle

“Equivalence between Processors & FPGAs design flows”

Basis:
High Abstraction level
C Based



Handel C & Gedae Integration



- Re-coding
- In-line optimisation
- Uses optimised IP where possible
- Place & Route
- Bit File Gen.
- Programming

Heterogeneous Architectures

High Speed Comms

Heterogeneous Support in Gedae

- **Requirements:**

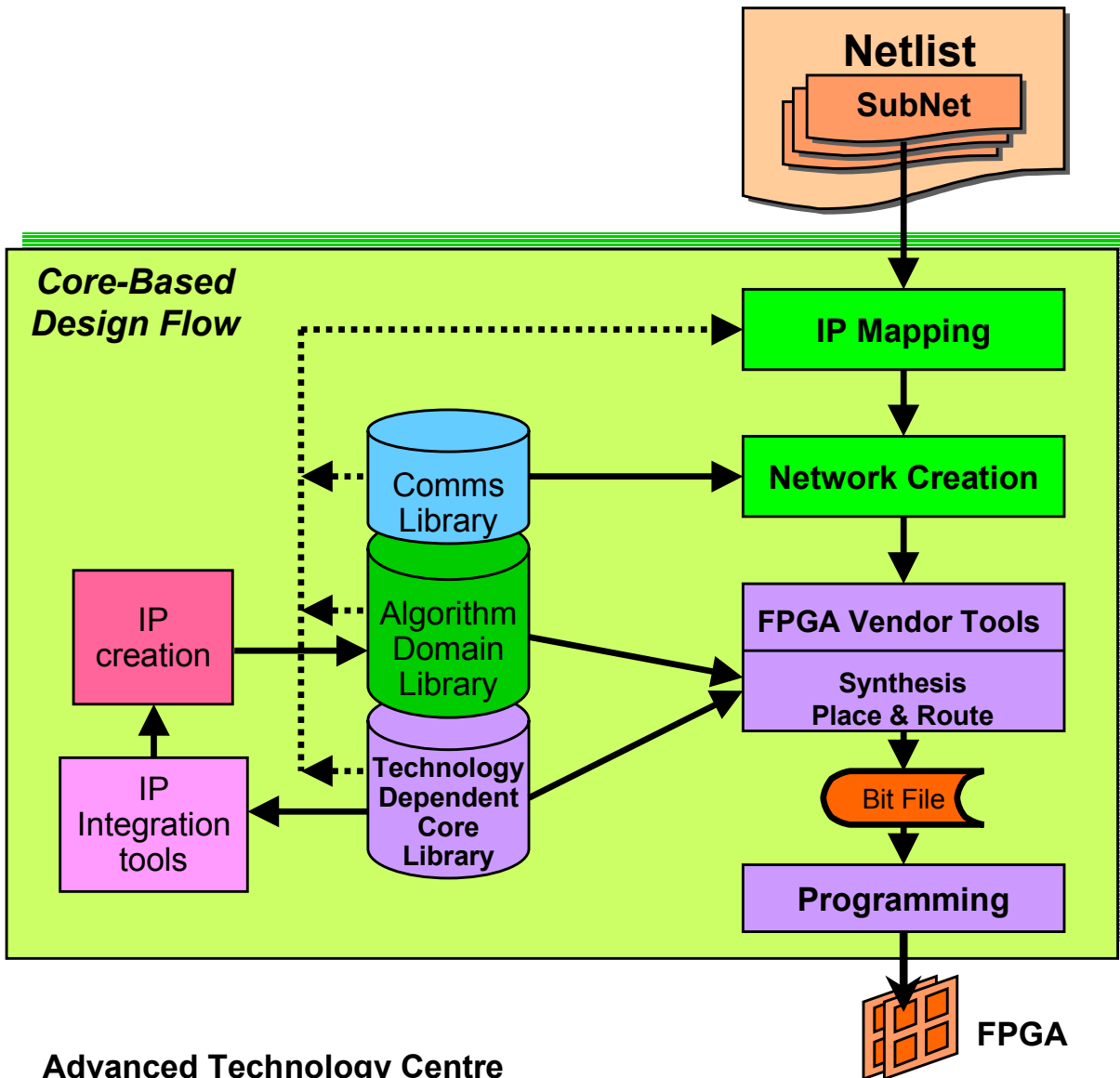
- Graphical development & seamless route to implementation
- Support hybrid targets (+ trace) - DSPs, RISCs, FPGAs
 - SoCs and NoCs in the future!
- Facilitate performance trade-off studies during mapping
 - minimise the rework during mapping changes
- FPGA implementation:
 - FPGA firmware programming capability for **system/software** engineers
 - *“hardware engineering for software engineers”*
 - Use optimised IP cores - at all levels of functional hierarchy
 - Take advantage of the inherent parallelism in FPGAs
 - Facilitate generation of new IP blocks/cores

Heterogeneous Support in Gedae (2)

- **Possible Routes:**
 - Low-level implementation (RTL)
 - HDLs or schematic capture - *architectural exploration difficult*
 - System-level design languages
 - refinement through various levels of abstraction - *register transfer level spec.*
 - maturity of techniques - *many lack efficient route to implementation*
 - Embedded processor cores
 - processor based design flows - *compiler efficiency & clock speed issues*
 - quick route for implementation - *especially of unsupported functions*
 - Core-based flow
 - High level of abstraction - *inherent parallelism in algorithms exploited*
 - Optimised IP Cores - *efficient implementation route*

Proposed Core Based FPGA Design Flow

“future extension to SoCs?”



Input Netlist:

- connectivity netlist - not executable
- subnets for each “processor”
- heirarchical

IP Mapping:

- converts functional reference to IP
- all levels of hierarchy
- switchable - on/off

Network Creation:

- provides comms between IP cores
- uses HDL or System Level Design Language

IP Core Libraries:

- comms, algorithmic, vendor
- multiple implementations

IP Creation:

- population of IP libraries
- multiple languages / tools

Requirements for Integration with Gedae

- **Netlist Export**

- Should be simple to provide - non-destructive to current flow
- Post partitioning & scheduling
- Different Levels: *Primitive, Partition, Model*

- **Building and Executing the Application**

- FPGA build process - very long compared to processor based
- Background process - enables other work to proceed

- **Data Communications**

- Run-time parameters - no embedded scheduler/kernel
 - parameter control program (Handel-C or micro-processor core)
- Data transfer to/from Host - unbalanced processing and asynchronous behaviour
 - buffering & blocking transfers
- Inter-processor communications
 - dedicated point-to-point, virtual channel and shared memory

Requirements for Integration with GEDAE (2)

- **Library Support**

- range of possible/supported data-types greater than processors
- need N-bit signed and unsigned integers and fixed point representations
- basic library of processing primitives - arithmetic & signal processing functions
- "multi-representation" library
- performance estimates for GSIM

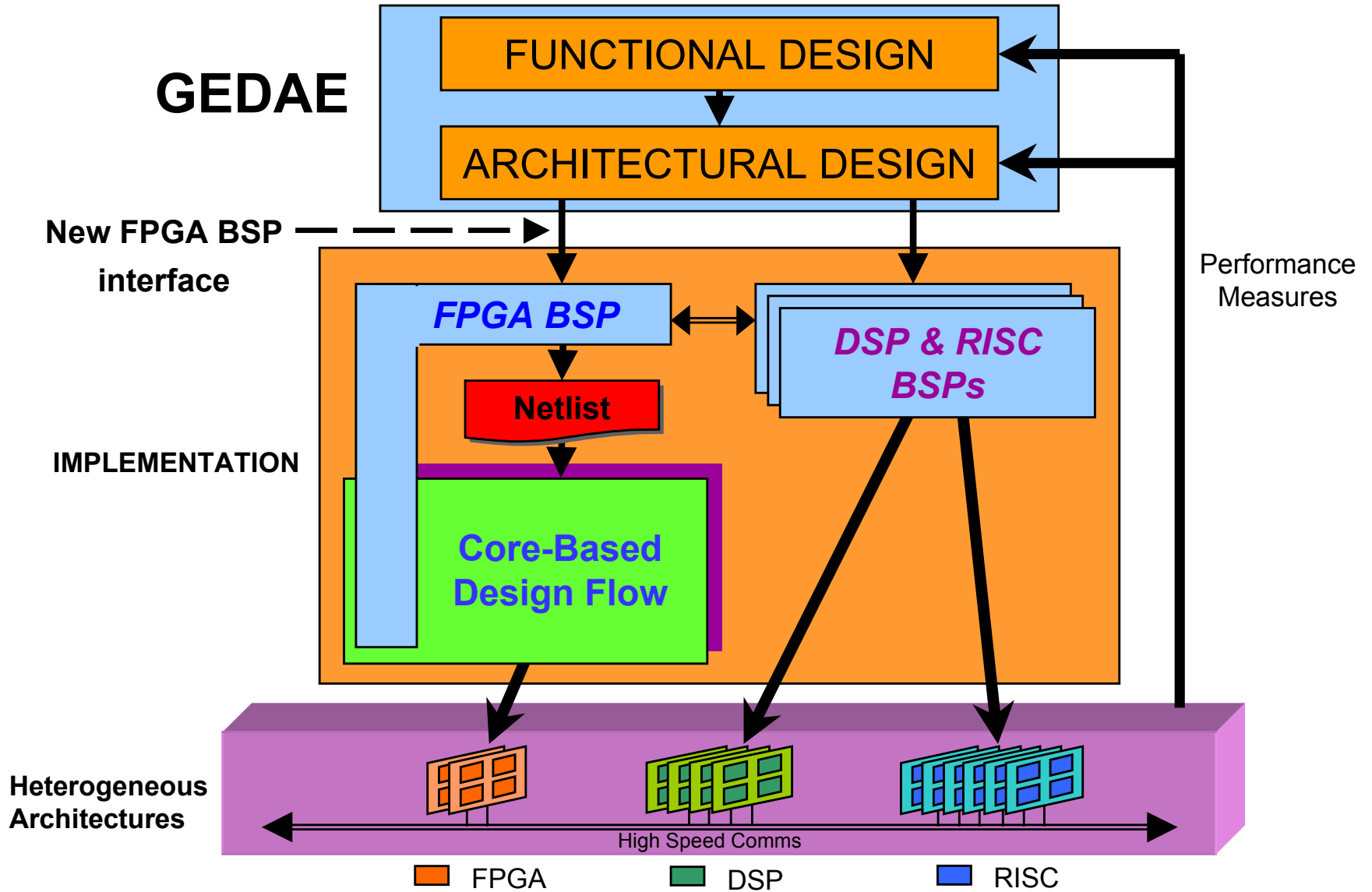
- **Heterogeneous Support**

- multiple BSPs in a single Gedae host - **potential problems at present**
- ability to add inter-platform IPC to an existing BSP

- **Monitoring & Debug**

- Trace Data Collection - **everything concurrent in FPGA!**
 - suitable data should be defined, collected and displayed
- Interface to Signal Monitoring - **rebuild times when adding scopes, echos, ...**
 - nodes in FPGA should be monitorable without rebuild

Integration with GEDAE



Summary

- **System-level design languages:**
 - enable hardware and software co-design
 - enable hardware to be synthesised directly at higher levels of abstraction
 - maturity and efficiency still an issue
- **ESPADON design flow:**
 - used Handel-C to achieve x7 improvement in lifecycle cost
 - inefficient implementation compared to HDLs
- **Core-based design flow:**
 - mirrors the requirements of the conventional rapid prototyping flow
 - efficient implementation route
 - enables efficient architectural trade-off studies
 - provides flexible design flow
 - *complementary to Gedae RTL - integration of RTL into flow simple*

Contact Information

Ian Alston & Bob Madahar

BAE SYSTEMS

**BAE SYSTEMS Advanced Technology Centre
West Hanningfield Rd, Gt. Baddow,
Chelmsford, CM2 8HN, UK**

ian.alston, bob.madahar@baesystems.com

Tel: +44 1245 242195, ...2262

Fax: +44 1245 242124

"© BAE SYSTEMS 2003. All rights reserved."

"Unless BAE SYSTEMS (Operations) limited has accepted a contractual obligation in respect of the permitted use of the information and data contained herein such information and data is provided without responsibility. BAE SYSTEMS (Operations) Limited disclaims all liability arising from its use."