Special Topics in Advanced Digital System Design

by

Dr. Lesley Shannon

Email: lshannon@ensc.sfu.ca

Course Website: http://www.ensc.sfu.ca/~lshannon/

Simon Fraser University

Slide Set: 4

Date: November 1, 2011

Slide Set Overview

- Embedded Systems
  - Definition
  - Design Methodology
- Specifying the Application's Platform
  - SoC? Custom/ASIC/FPGA?
  - Processor Choices
  - Other hardware and software choices

Slide Set Overview

- An implementation study example
  - To demonstrate why we have to make choices in our implementation platform to meet application specification requirements
  - Example Application:
    - A digital camera

Slide Set Overview

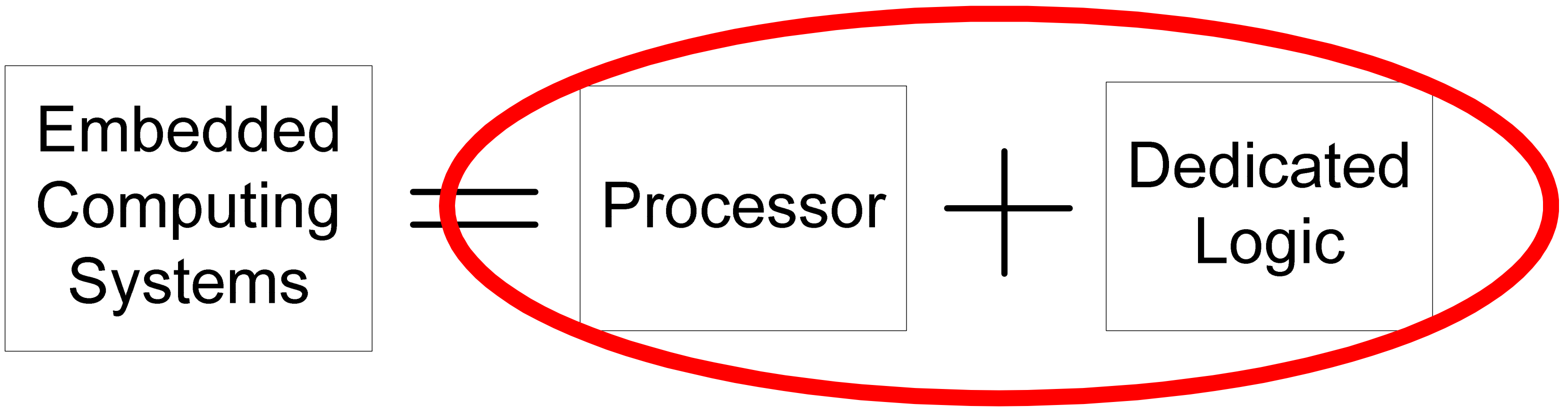
- We've been talking about the EDK tools, FPGAs, and FPGA CAD Flow and Compilers
- Now we're going back to how you make design implementation decisions throughout the design process
  - Once you have a specification, you have to choose a platform that will allow you to meet this specification.

Embedded Computing Systems

Embedded Computing Systems = Processor + Dedicated Logic

- Design Constraints:
  - Area
  - Power
  - Performance
  - Cost

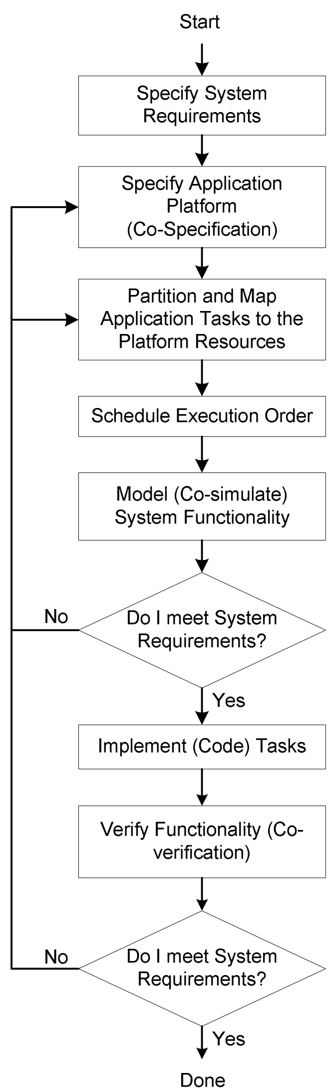
Embedded Computing Systems



- Design Constraints:
  - Area
  - Power
  - Performance
  - Cost

Embedded System Design Methodology: see Handout

Embedded System Design Methodology



System-on-Chip

- What are the other options?
  - Multiple dedicated integrated circuits on a PCB
  - A personal computing station
    - completely software implementation
  - What are other options?
- Why implement a System-on-Chip?
  - Performance constraints
  - Area constraints
  - Why else?

System-on-Chip

- If you choose to implement an SoC, it could be a:
  - Custom Designed ASIC
  - Standard Cell ASIC
  - Structured ASIC
  - FPGA

Custom Designed ASIC

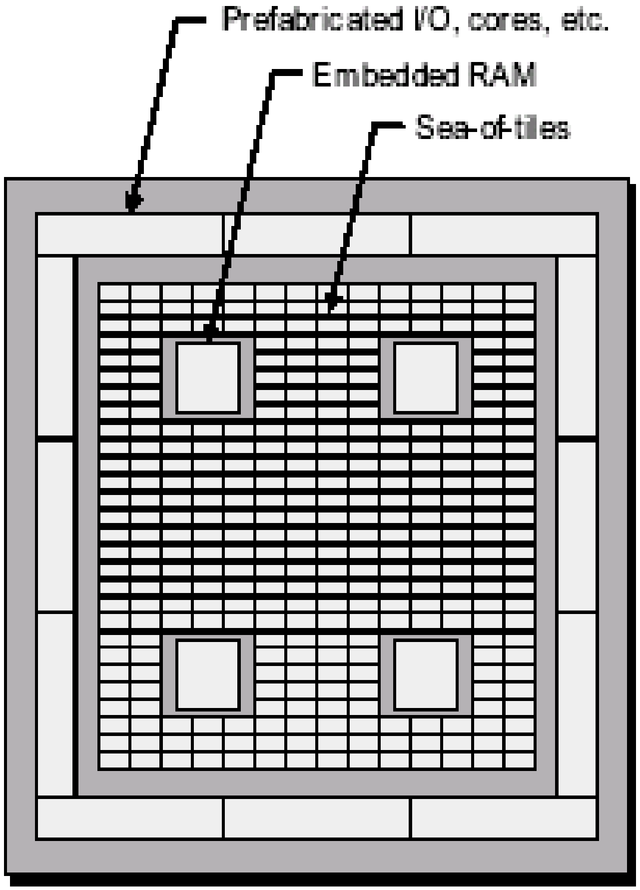
- Hand Tailored at the Transistor Level to perform Application Functionality
  - Could be digital or analog circuitry or mixed signal
  - Can you think of some examples that might use this technology?
    - Hint: think of standard cell libraries

Standard Cell ASIC

- Foundries (e.g. TSMC) have libraries of standard cells
  - Can be very low-level (inverter)
  - Can represent basic modules (e.g. full adder)
  - Can you think of some examples that might use this?

Structured ASIC Architecture

- Two Main Levels
  - Structured Elements
    - Combinational and sequential function blocks
    - Can be a logical or storage element
  - Array of Structured Elements
    - Uniform or non-uniform array styles
    - A fixed arrangement of structured elements

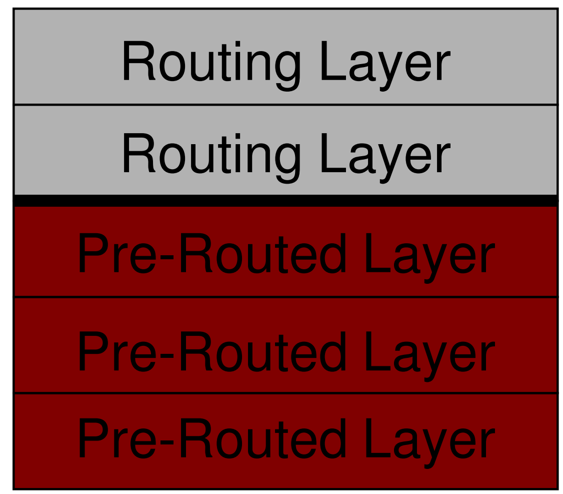


Structured ASIC Concept

- A Structured ASIC falls between an FPGA and a Standard Cell-based ASIC
- Structured ASIC's are used mainly for mid-volume level designs
- The design task for structured ASIC's is to map the circuit onto a fixed arrangement of known cells

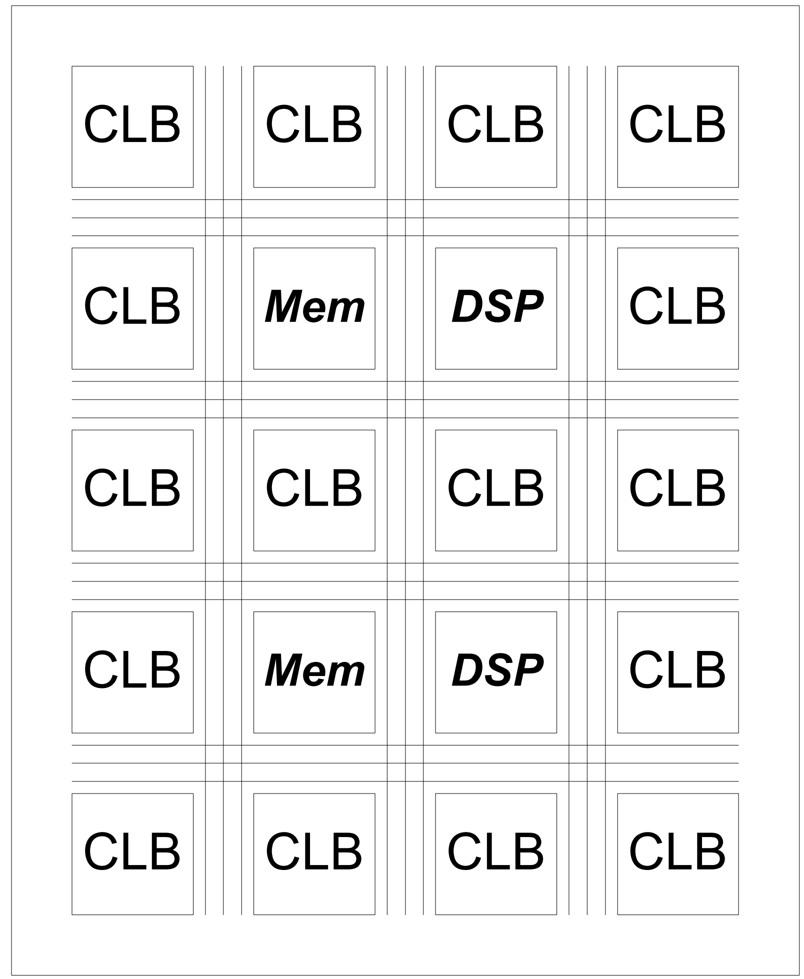
Structured ASIC Concept

- Largely Prefabricated
  - Components are “almost” connected in a variety of predefined configurations
  - Only a few metal layers are needed for fabrication
  - Drastically reduces turnaround time
- Commercially available:
  - Altera’s HardCopy



Field Programmable Gate Arrays

- Prefabricated programmable circuitry
- Configurable Logic Blocks (CLBs)
- Heterogeneous Blocks (Memory & DSP)
- Configurable Routing
- Can implement >30 uprocs!



Pros and Cons of Custom ICs/Standard Cell ASICs/Structured ASICs/FPGAs

FPGA vs Standard Cell ASIC

- Easy to Design
- Short Development Time
- Low NRE Costs
- Design Size Limited
- Design Complexity Limited
- Performance Limited
- High Power Consumption
- High Per-Unit Cost

- Difficult to Design
- Long Development Time
- High NRE Costs
- Support Large Designs
- Support Complex Designs
- High Performance
- Low Power Consumption
- Low Per-Unit Cost (at high volume)

Structured ASIC's Combine the Best of Both Worlds (the HardCopy sales pitch) 

Structured ASIC Advantages

- Low NRE cost
  - Implementation engineering effort
  - Mask tooling charges
- High performance
- Low power consumption
- Less Complex
  - Fewer layers to fabricate
- Small marketing time
  - Pre-made cell blocks available for placing

Custom ASICs

- Homework: From the above, figure out the pros and cons of custom ASICs.

Processor Choices

Processor Choices

- You could pick:
  - An off the shelf processor, or
  - “Roll out your own” processor
- You could also create a more complex platform by combining a processor with hardware units

Processor Choices

- If you just use a single processor
  - Then write software to run on that processor
- But what type of processor do you use?
  - Pentium and friends are good for desktop use
    - Often not a good choice for embedded systems
      - Too complex (means it is more expensive)
      - Not necessarily suitable for real-time
      - High-end processors might use a lot of power
- Other Processors are available…

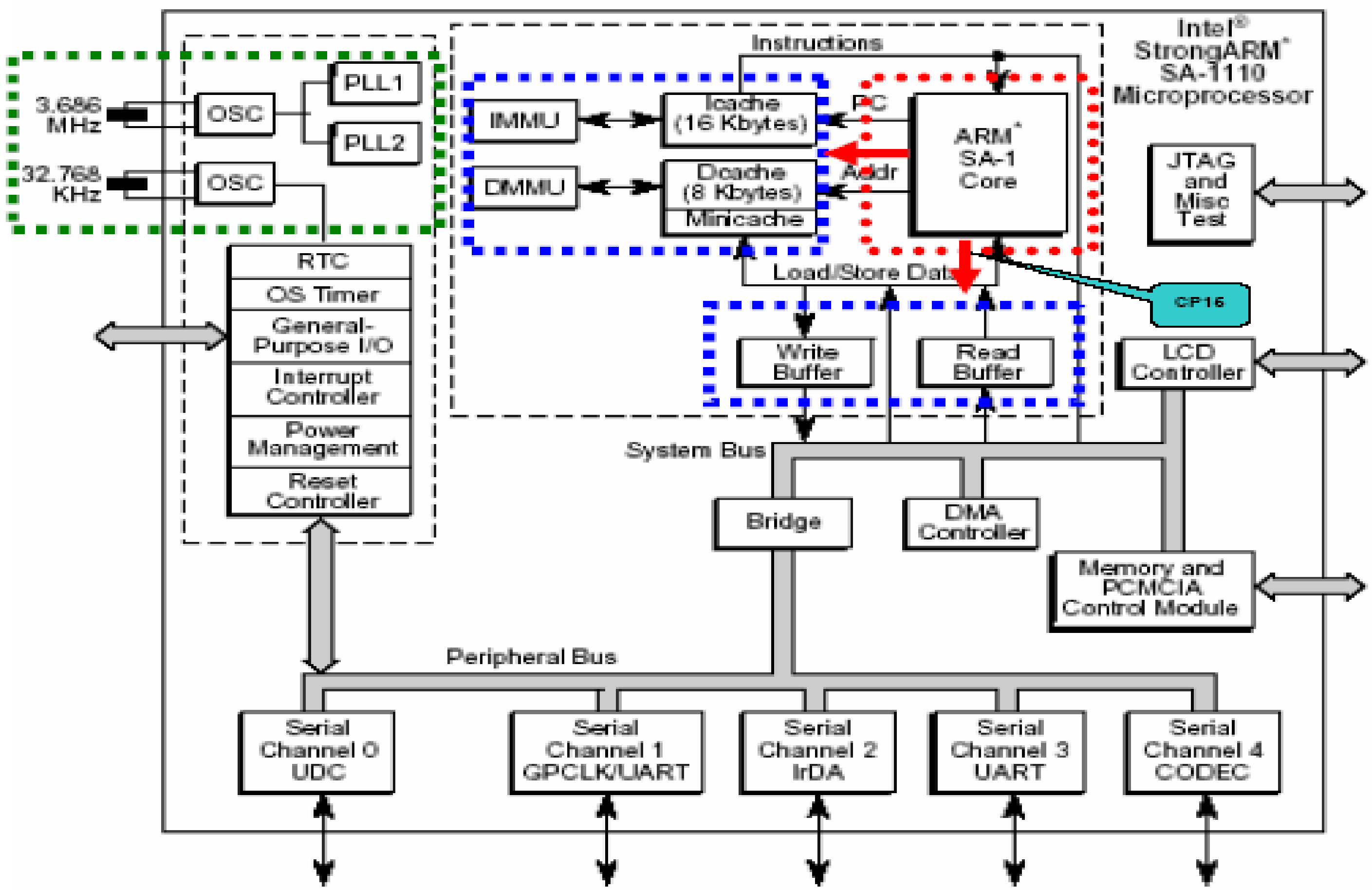
Things to think about

- Overall, things you should think about when choosing a processor:
  - Cost
  - 8-bit, 16-bit, 32-bit?
  - Do you need floating point, or is fixed point arithmetic OK?
  - Power dissipation?
  - Availability?
  - Programming model?
  - Interfaces to the real world? Timers?

- In general, a ***soft*** real-time system may be implementable on a normal workstation, but an embedded system rarely would be.

Simpler Processors

- MIPS, ARM, etc:



Recall our ipod

- The Apple iPod is powered by two ARM7TDMI microprocessors.

The Samsung Miniket Camcorder

- 5 ounces (with battery)
- capture D1-resolution video (720 x 480, the same resolution as DVDs) at 30 frames/sec, with audio
- 10-power optical zoom, along with 100-power digital zooming
- Contains a Samsung SoC containing an ARM processor

Microcontrollers

- Such as the Motorola 68HC11
  - Combine a simple processor with interface blocks
  - Usually one or more timers (important for real-time!)

Digital Signal Processors

- Processors optimized specifically for DSP applications
  - Lots of parallel multiply-accumulates
  - Several different programming models
    - (C, Matlab, etc)
  - Could be either fixed point of floating point
    - Motorola 5600 = Fixed Point
    - Motorola 9600 = Floating Point
- Good if you want to process signals in real time, but not really suitable for control-type applications

“Rolling out your own” processor

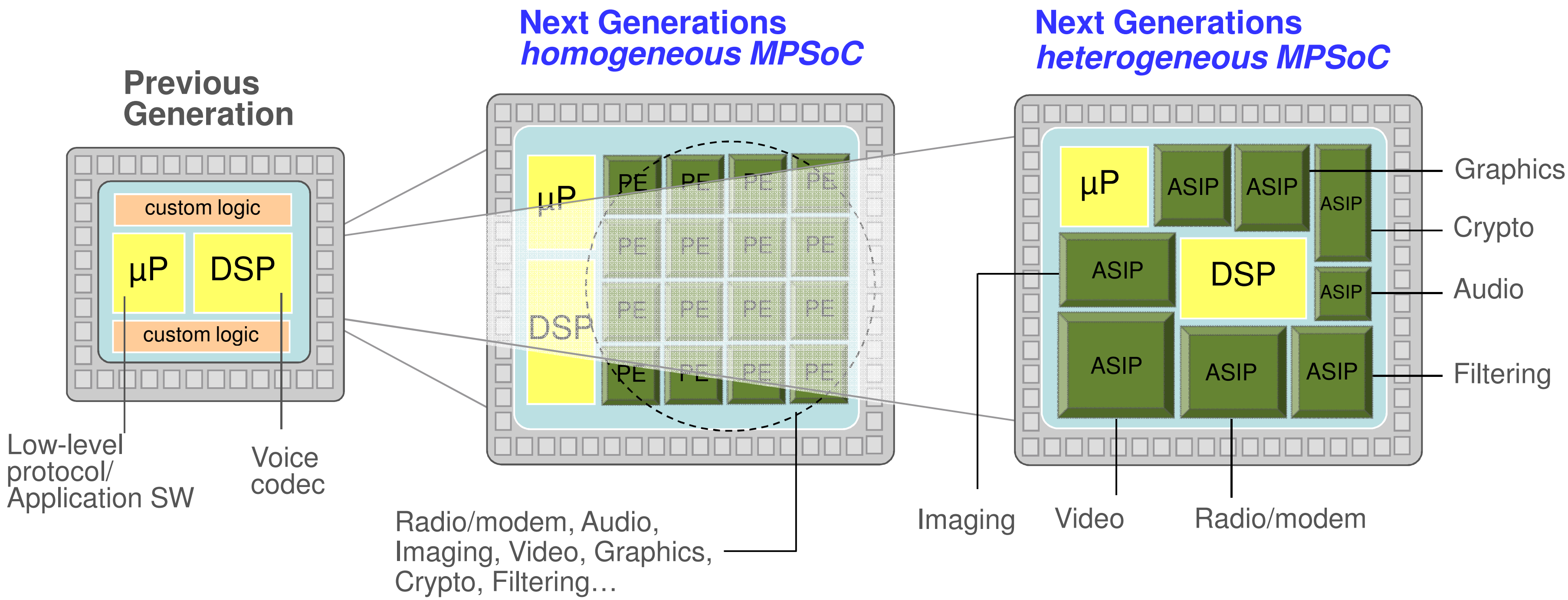
- You could create your own processor just for your application as an Application-Specific Instruction-set Processor (ASIP)
  - Could get exactly the cost/power/performance point you want
  - But, designing a processor takes time
    - Think of the man years put into designing the next generation pentium chips (100s of man-years)

“Rolling out your own” processor

- You could create your own processor just for your application as an Application-Specific Instruction-set Processor (ASIP)
  - Another option:
    - Some companies provide tools to allow you to customize a base processor for a given application
    - No need to design a processor from scratch
    - But sometimes the degree of customization possible is limited
  - The following slides are taken from a keynote address at ASAP and highlight things you need to consider when designing an ASIP

ASIPs in Multi-Core SoC

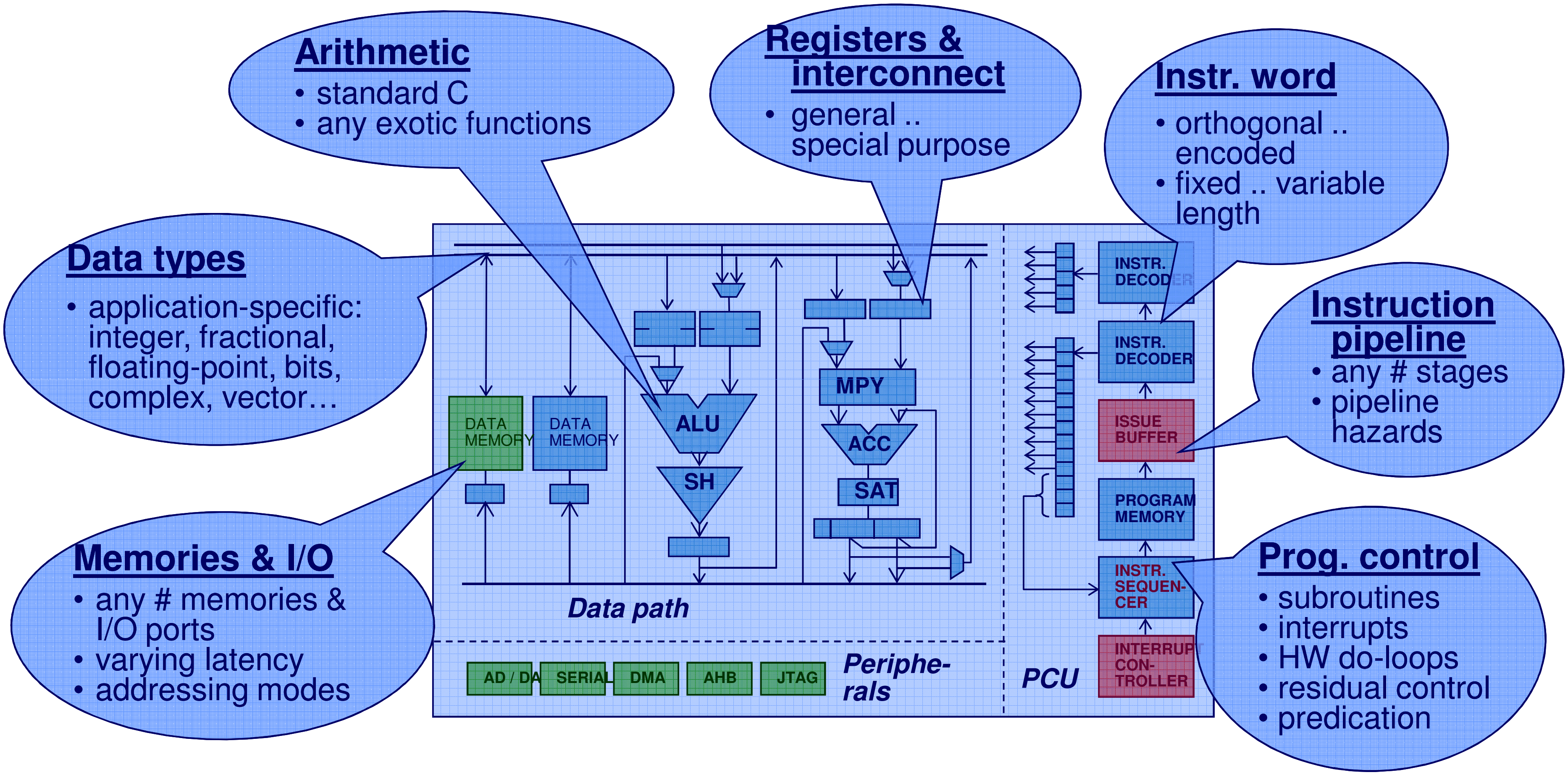
- System-on-Chip becomes Sea-of-Cores



- Heterogeneous best for deep submicron power challenge
  – Each ASIP optimised for its function: minimal logic, balanced parallelism
  – Local communication
  – Power gating based on system requirements

From Keynote at ASAP 2008 by Gert Goossens from Target the ASIP Company

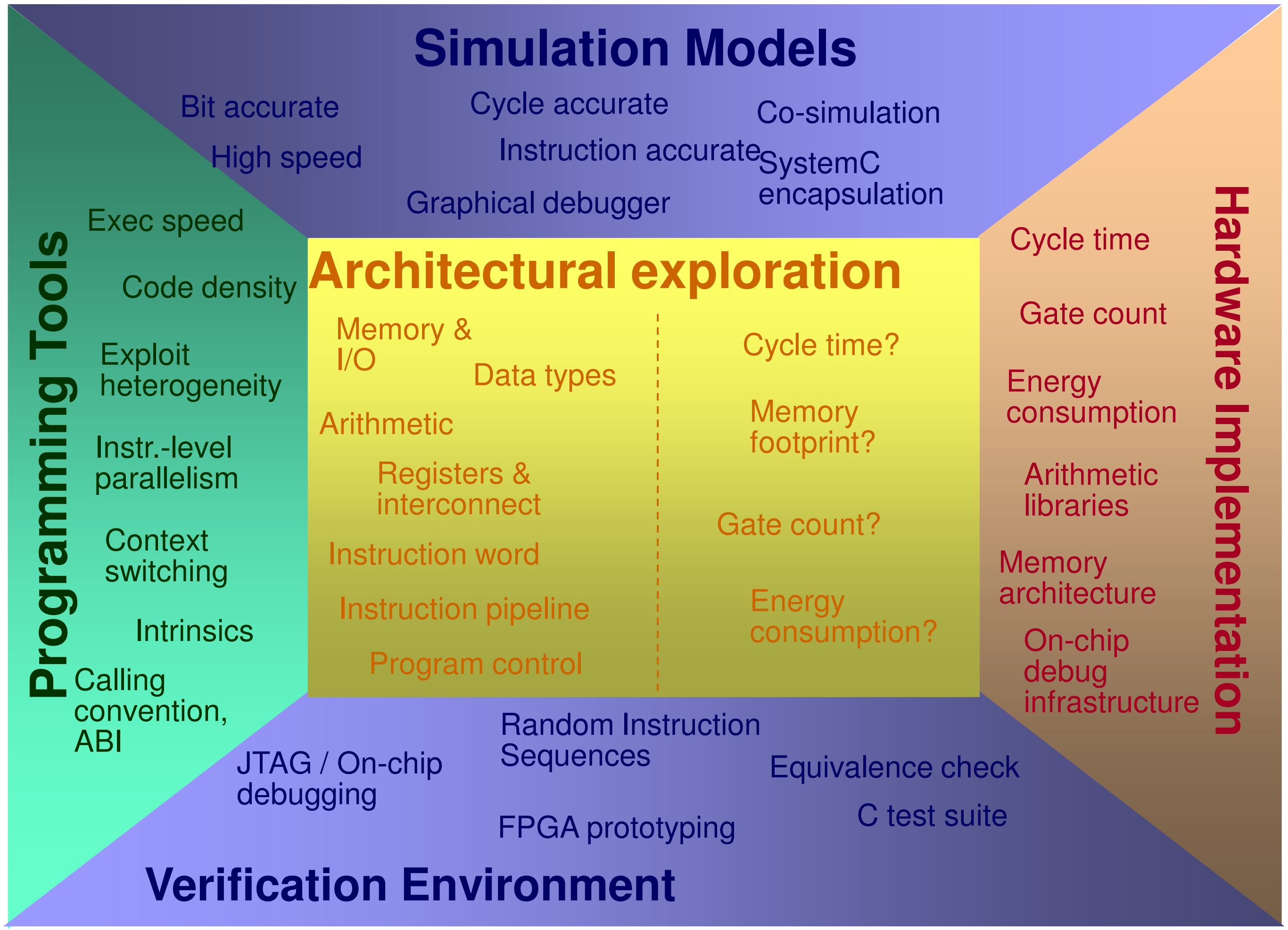
Architectural variation of ASIPs



• True architectural exploration: opportunity & challenge…

From Keynote at ASAP 2008 by Gert Goossens from Target the ASIP Company

ASIP Designer's Challenges



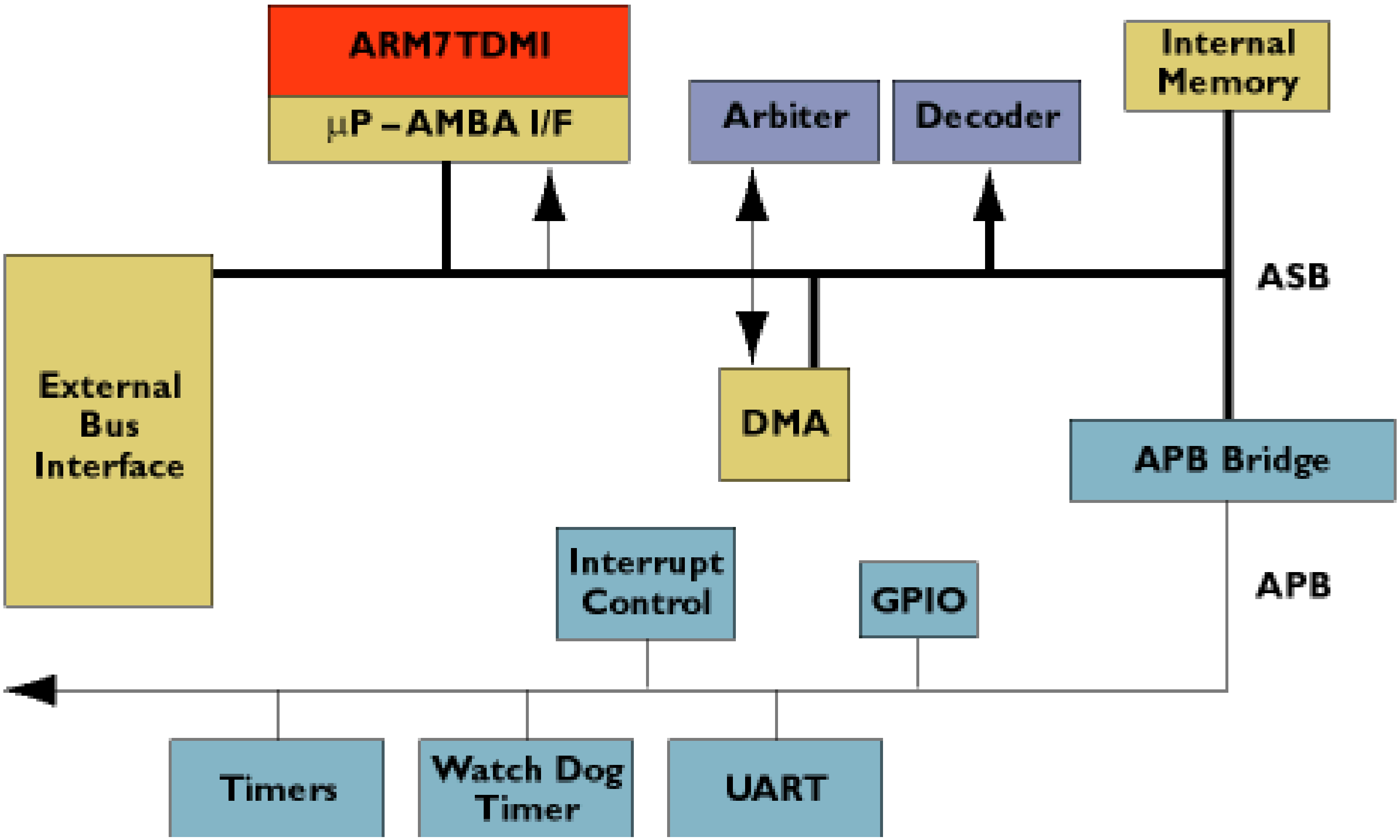
From Keynote at ASAP 2008 by Gert Goossens from Target the ASIP Company

Flexible Processor Examples

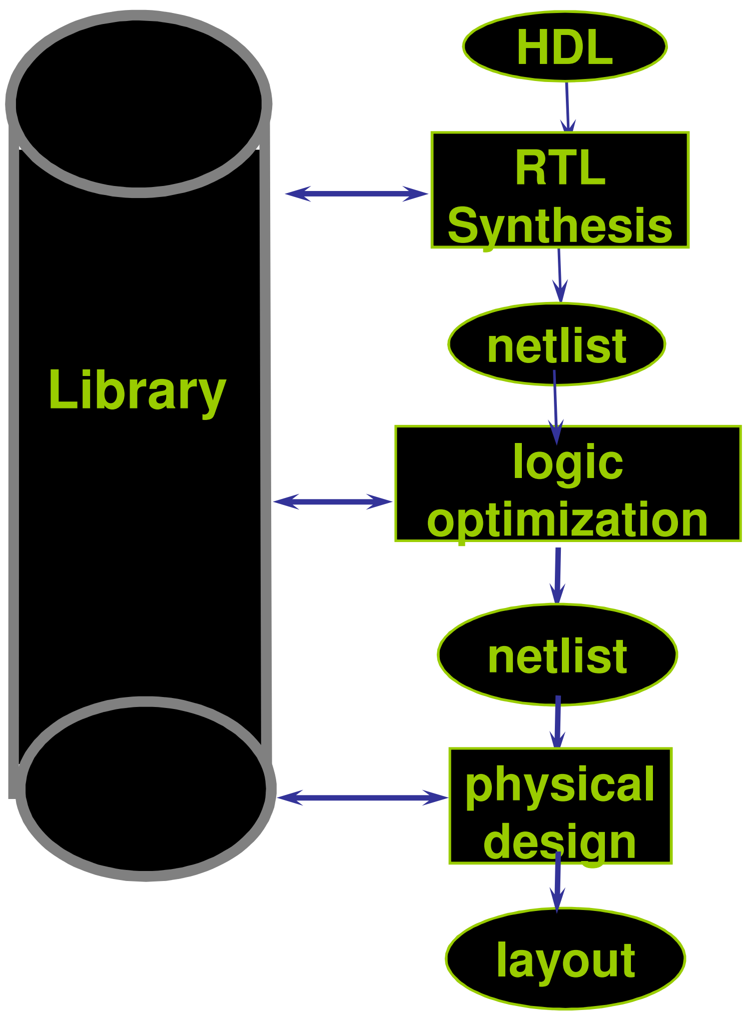
- ARM allows for configurability via AMBA bus
- Offers ``prime cell'' peripherals which hook into AMBA Peripheral Bus (APB)
  - UART
  - Real Time Clock
  - Audio Codec Interface
  - Keyboard and mouse interface
  - General purpose I/O
  - Smart card interface
  - Generic IR interface

- http://www.arm.com/Pro+Peripherals/PrimeCell/index.html

An ARM plus your choice of peripherals



Can be even more flexible…

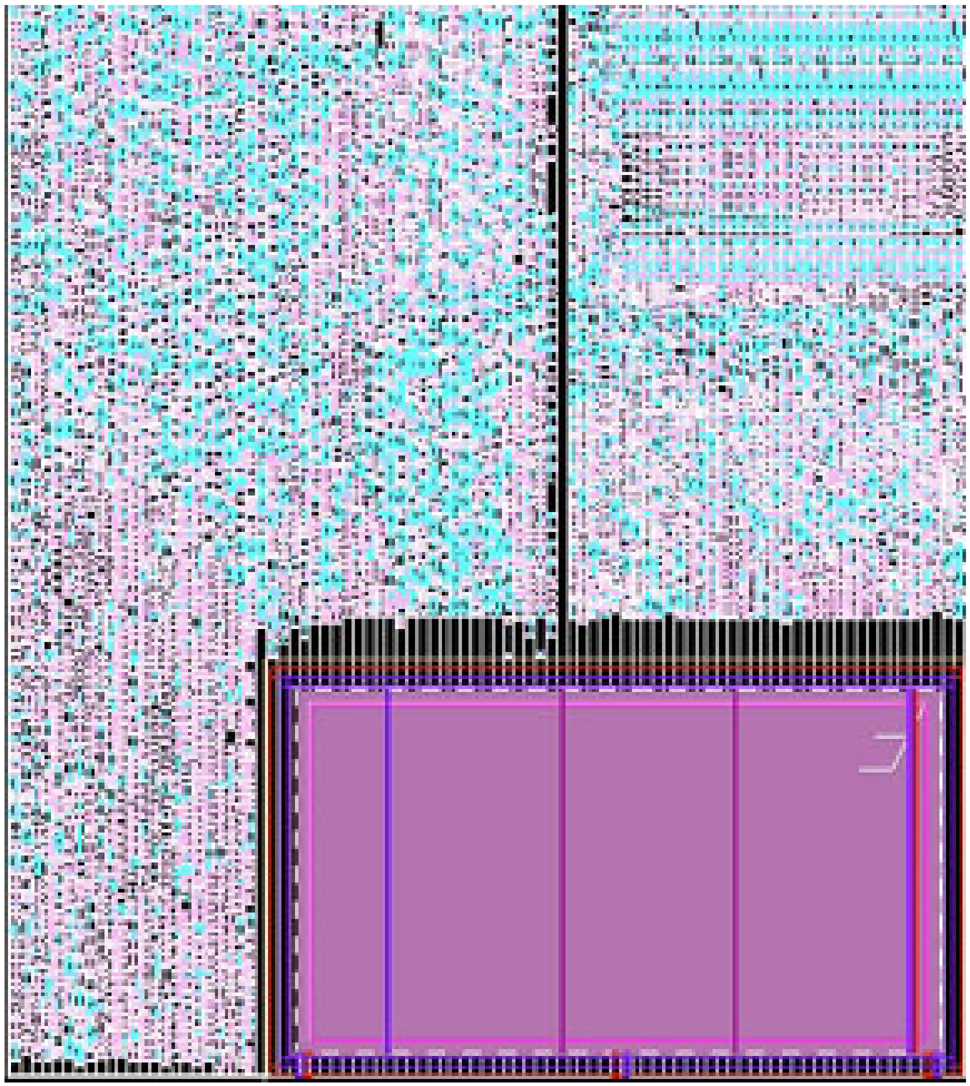


Synthesis of a processor core from an RTL description allows for:

- Lots more configurability

Examples:
- ARM7
- Motorola Coldfire
- Tensilica Xtensa

ARM7



ERROR: stackunderflow
OFFENDING COMMAND: ~

STACK: