SYSTEMS ON CHIP (SOC) FOR EMBEDDED APPLICATIONS

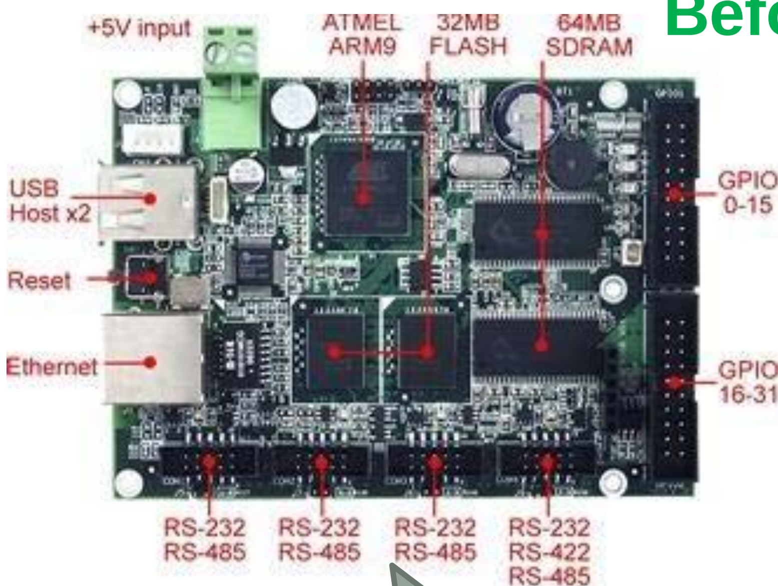
Embedded System

- “System”
  - Set of components needed to perform a function
  - Hardware + software + ….
- “Embedded”
  - Main function not computing
  - Usually not autonomous
  - Usually a computer inside a system
  - Application specific
  - Subject to constraints

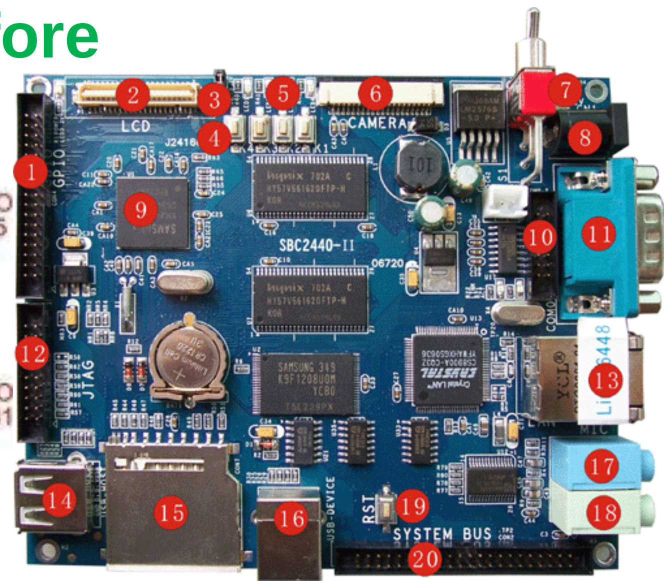
System on chip

- Definition
  - (nearly) complete embedded system on a single chip
- Usually includes
  - Programmable processor(s)
  - Memory
  - Accelerating function units
  - Input/output interfaces
  - Software
  - Re-usable intellectual property blocks (HW + SW)

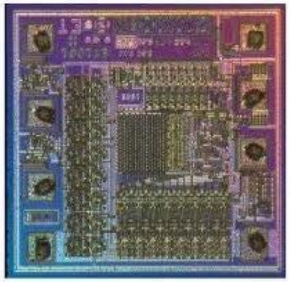
SoC Design Goal



Before

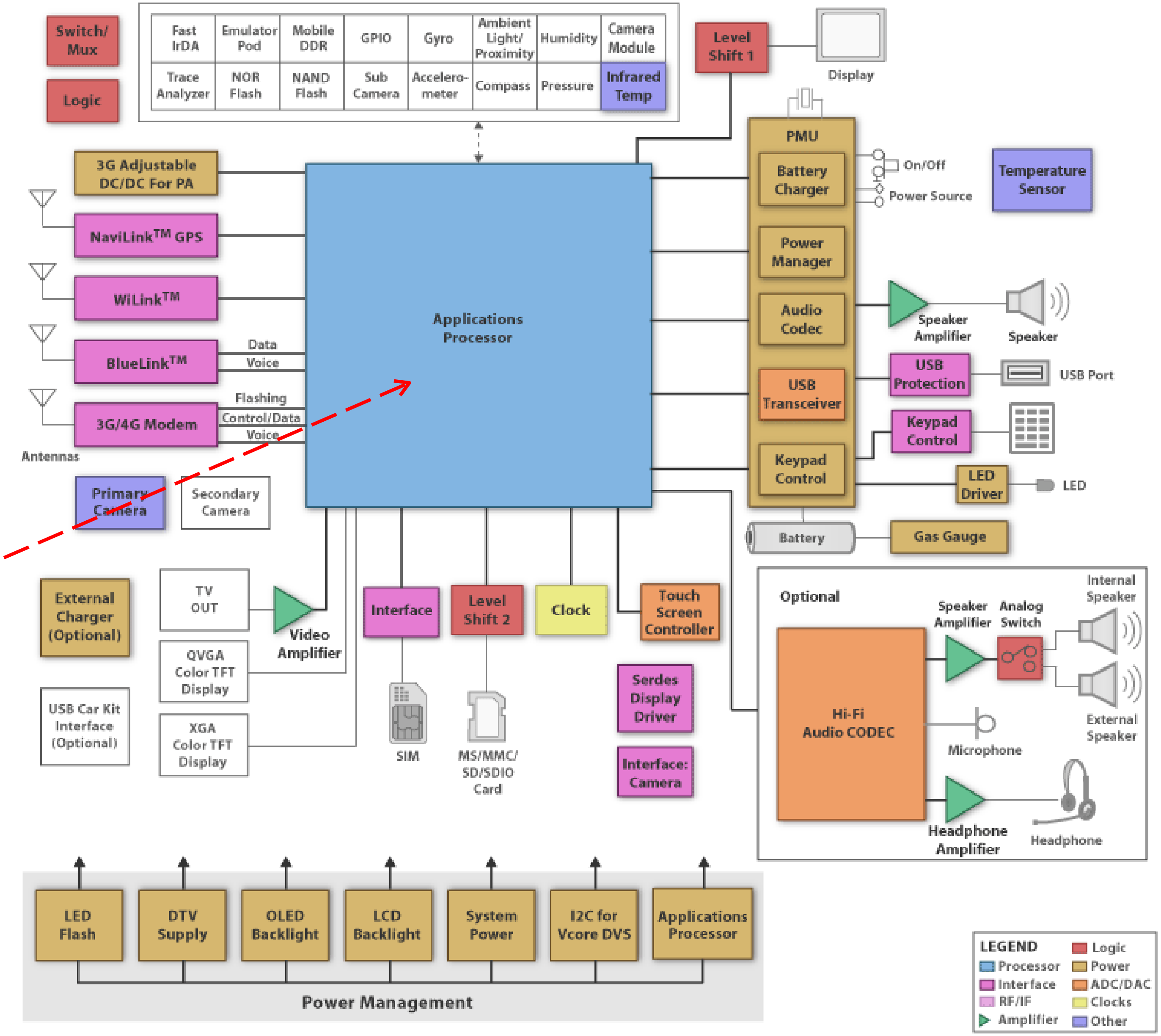


After



T.I . smartphone reference design

Main SoC

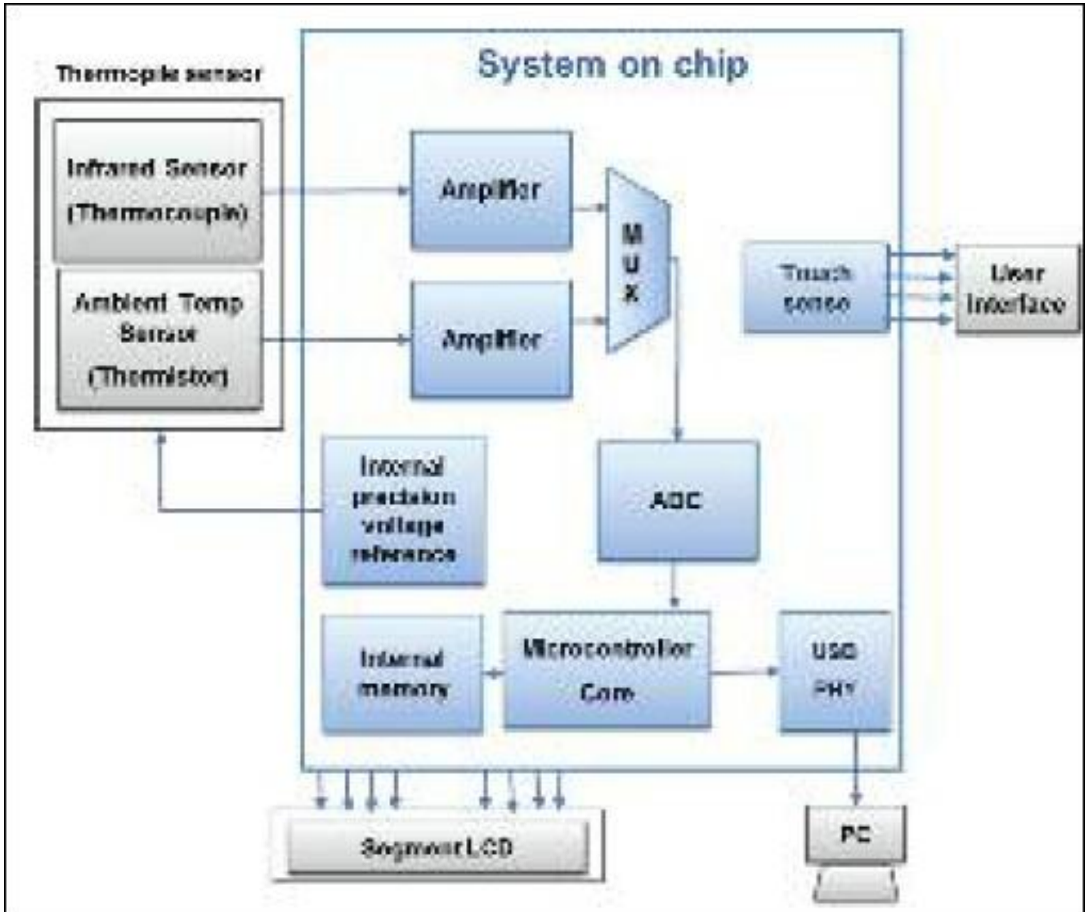


Apple "A5" SoC

- Used in *iPad 2* and *iPhone 4S*
- Manufactured by Samsung
  - *45nm, 12.1 x 10.1 mm*
- *Elements (unofficial):*
  - *ARM Corex-A9 MPCore CPU - 1GHz*
    - *NEON SIMD accelerator*
  - *Dual core PowerVR SGX543MP2 GPU*
  - *Image signal processor (ISP)*
  - *Audience "EarSmart" unit for noise canceling*
  - *512 MB DDR2 RAM @ 533MHz*

iPhone 4 circuit board (with A4 SoC)

Example: blood pressure monitor SoC



SoC Challenges

- SoC Designs
  - More complex, more functions, higher gate counts
  - Faster, cheaper, smaller
  - More reliable
- How to handle complexity?
  - System design at multiple abstraction levels
  - Integration of heterogeneous technologies & tools
  - Signal integrity & timing
  - Power management
  - SoC test methodology

SoC design with re-usable IP modules

- IP = intellectual property
  - HW or SW block
  - Designed for reuse
  - Need for standards (VSIA)
- Platform-based SoC design
  - Organized method
  - Reduce cost and risk
  - Heavy re-use of HW and SW IP
- Steps in re-use
  - Block -> IP -> integration architecture

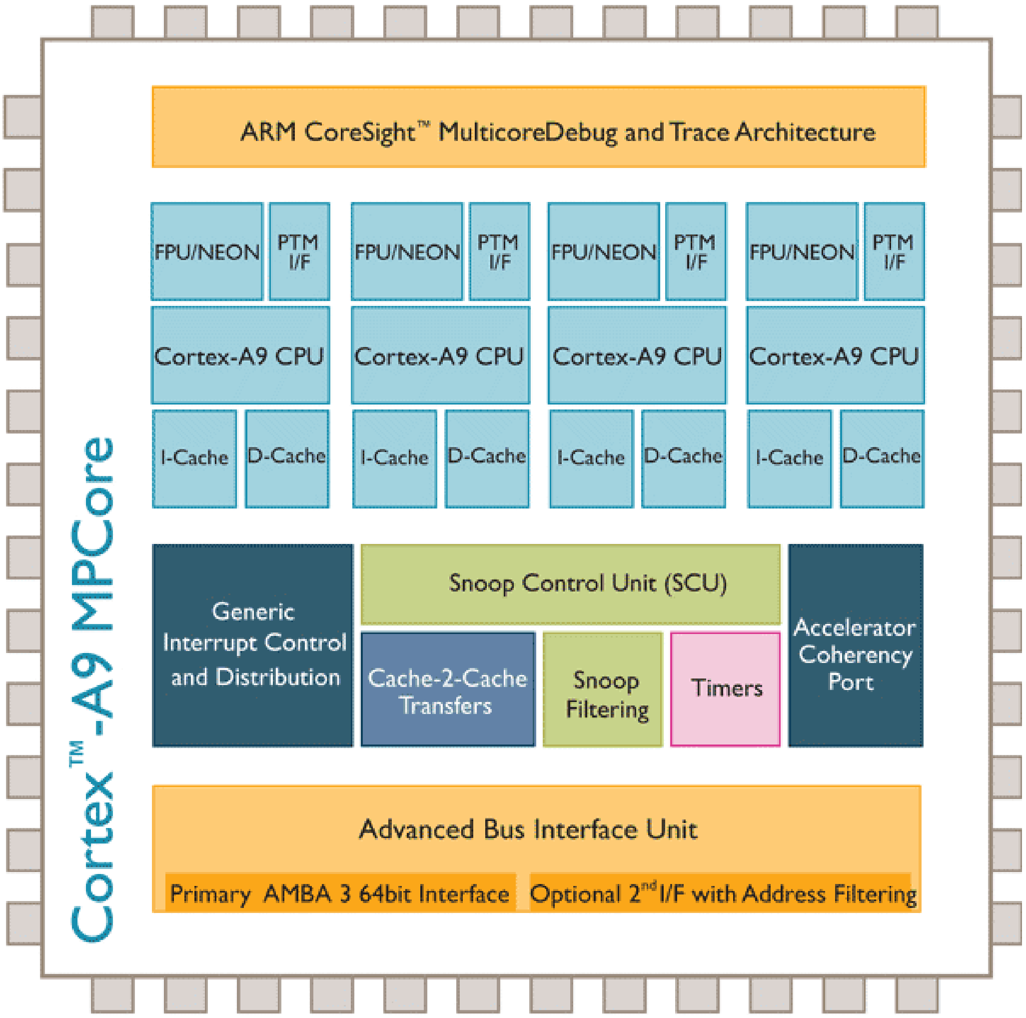
ARM IP (ARM makes no hardware)

- Processors
  - Cortex A, Cortex R, Cortex M, ARM11, ARM9, ARM7, SecurCore
- Multimedia IP - graphics, video, audio
  - Mali-T604 GPU graphics processor
  - Mali-VE6 video engine
- System IP
  - CoreLink – interconnect & memory controllers
    - Supports Cortex and Mali processors
    - AMBA – Advanced Memory Bus Architecture
  - CoreSight – debug and trace IP (build into SoC)
- ARM “Artisan” Physical IP
  - Logic IP, Standard Cells, Memory Compilers, Interface IP
  - *Technology-specific*

ARM SoC-based products

Joe Bungo: CPU Design Concept to SoC

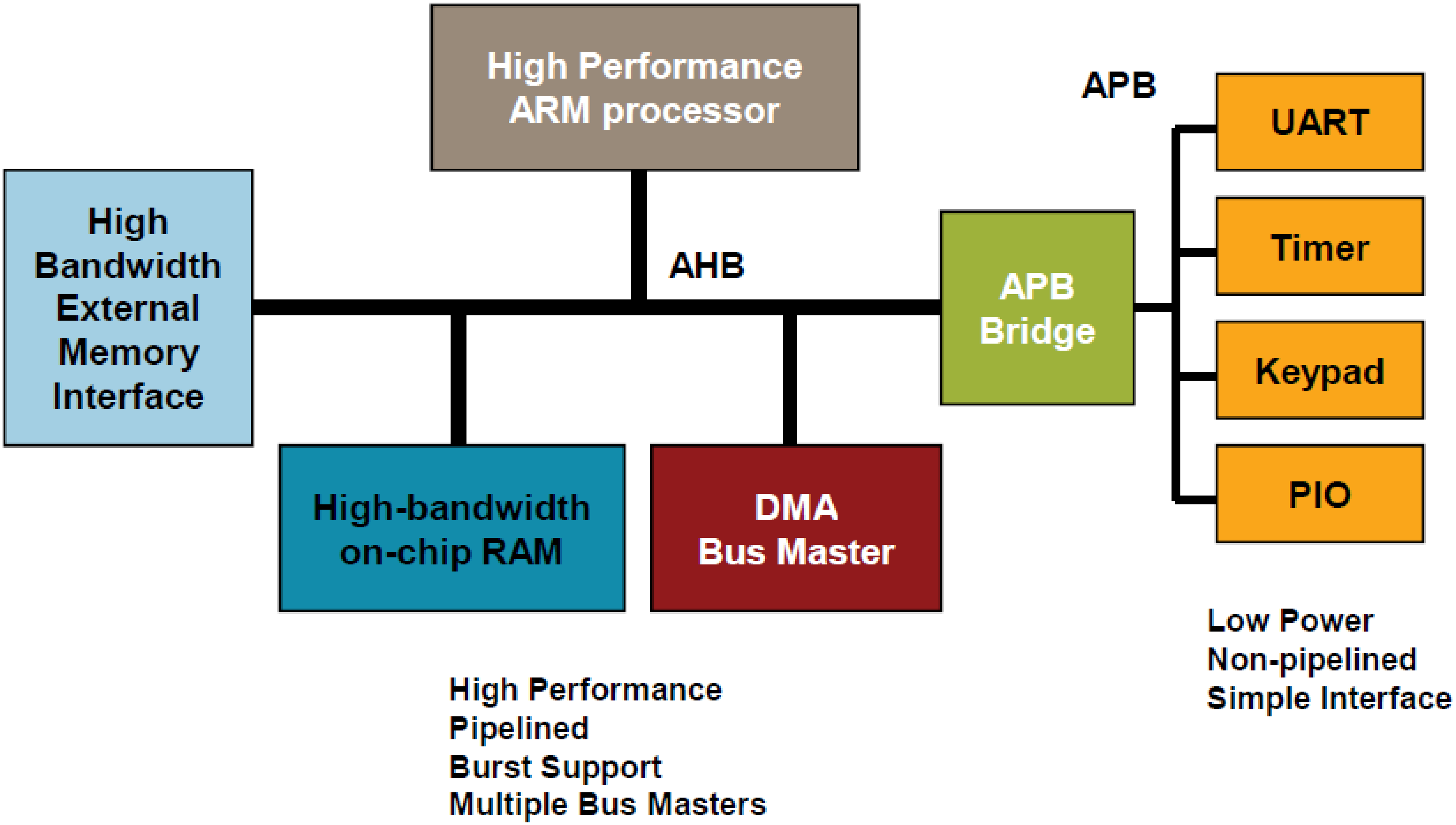
ARM Cortex-A9 MPCore



ARM Advanced Microcontroller Bus Architecture (AMBA)

- On-chip interconnect specification for SoC
- Promotes re-use by defining a common backbone for SoC modules using standard bus architectures
  - AHB – Advanced High-performance Bus (system backbone)
    - High-performance, high clock freq. modules
    - Processors to on-chip memory, off-chip memory interfaces
  - APB – Advanced Peripheral Bus
    - Low-power peripherals
    - Reduced interface complexity
  - ASB – Advanced System Bus
    - High performance alternate to AHB
  - AXI – Advanced eXtensible Interface
  - ACE – AXI Coherency Extension
  - ATB – Advanced Trace Bus

Example AMBA System



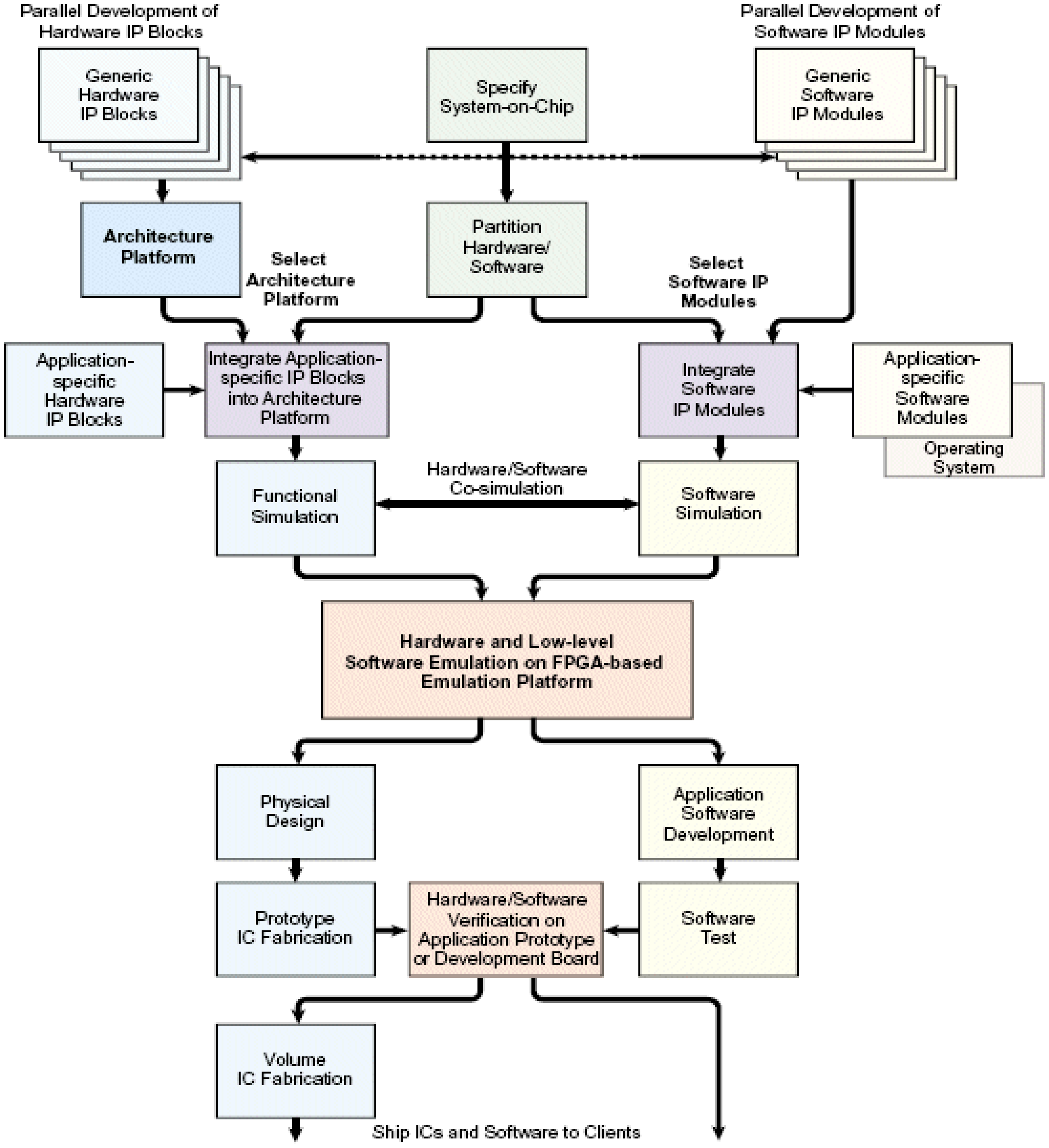
Joe Bungo: CPU Design Concept to SoC

SoC Design Process

- Customer requirements
- System specification
- Architecture design
  - Hardware vs. software
- Component design
- Integration
- Verification
- Manufacture
- Test

Model, simulate,
and evaluate
at each stage

SoC Design Flow



High-Level Performance Modeling

- **Identify Workloads**
  - Based on target market
  - Standard benchmarks (spec, EEMBC)
  - O/S based “real” benchmarks – browser, real apps
- **Performance Models**
  - C-based, highly configurable
  - Internally developed (no EDA vendor)
  - Fast Instruction-Set Based Model
    - No timing information, but very fast
    - Used for statistics collection and coarse algorithm development (i.e. branch prediction scheme, load/store address patterns)
  - Abstracted Pipeline
    - Reasonably accurate, longer development time
    - More specific to microarchitecture

Higher levels of abstraction for SoC

- ESL (Electronic system level)
  - RTL (register transfer level) to TLM (transaction-level modeling)
  - VHDL to SystemC to UML
- HW/SW co-design
  - Simulation models/emulators of hardware to develop software while hardware is being developed
  - Need new tools
  - Consider the whole system
  - Large optimization potential
  - Combination of formal, semi-formal and non formal techniques

Unit RTL

- Synthesizable HDL models
- Split work into units based on functionality
  - Verilog language of choice
  - Write low-level constructs only (assign, case)
  - Why?
    - Portability; we target multiple partners and have to target
    - ‘lowest-common denominator’ design tools
  - Know your RTL! Easier to count gates “on-the-fly”

- Orderly bring-up; integration as soon as possible

ARM Design Simulation Models (DSM)

- HDL behavioral models of ARM cores
  - for functional and "in some cases, timing" simulation
  - derived from ARM core RTL code
  - full device functionality
  - register visibility
  - configure cache and memory sizes
  - compatible with VHDL/Verilog simulators (ex. ModelSim)
- Back-annotation capable, timing accurate
  - accept timing through SDF files
  - min/typ/max pin-to-pin delays
  - setup/hold/pulse checks
- Also called "Design Sign-Off Models"
  - generated from technology-specific netlist of a core

SoC integration

- Once core is built, integrated with other cores into chip
- Many millions of gates; can we abstract this out?
- System Design
  - SystemC model – transaction level, no timing
  - Can chain processor/peripheral models together to test OS
- Cycle-level system simulation
  - compiled model
  - no internal visibility
  - faster runtimes
  - smaller, simulator won’t run out of memory!

ARM Development Tools

- Software development
  - ARM Development Studio 5 (DS-5)
    - For ASICs and ASSPs
    - Compilers, debugger, system performance analyzer, real-time system simulator
  - Keil Microcontroller Development Kit (MDK)
    - For embedded microcontrollers
    - Cortex M, Cortex-R4, ARM7, ARM 9 devices
    - Compilers, debugger, simulators
- Models
  - ARM Fast Models – virtual platforms for software development before silicon

Conclusions

- SoC design requires different design approach than traditional ASICs
  - More modeling & simulation at higher abstraction levels
- Heavy use of IP, re-usable modules, platform-based design
  - SoC design team must work with IP vendor and foundry
- Use platform design & standard interfaces between IP
- Hardware/software co-design
- Many design challenges