



# System-on-Chip and Network-on-Chip Design



Grant Martin, Tensilica Inc.

1<sup>st</sup> Semester in Master's  
Seong Min Jo



- Introduction
- System-on-a-Chip
- System-on-a-Programmable Chip
- IP Cores
- Virtual Components
- Platforms and Programmable Platforms
- Integration Platforms and SoC Design
- Overview of the SoC Design Process
- System-Level Design
- Interconnection and Communication Architectures for SoC
- Computation and Memory Architecture for SoC
- IP Integration Quality and Certification Methods and Standards
- Summary



- SoC (System-on-Chip)
  - + Major revolution in IC design
  - + The integration of all or most of electronic components
  
- Today's chipset would become tomorrow's chip
  - + The move to SoC in the mid-1990s (0.35 ~ 0.25  $\mu\text{m}$ )
  - + More and more components are integrated
    - Digital baseband function
    - Analog components
    - Passive components
    - Sensor, actuator, biological, nanotechnology
  
- What is most important about a SoC
  - + Design a SoC as an integrated system



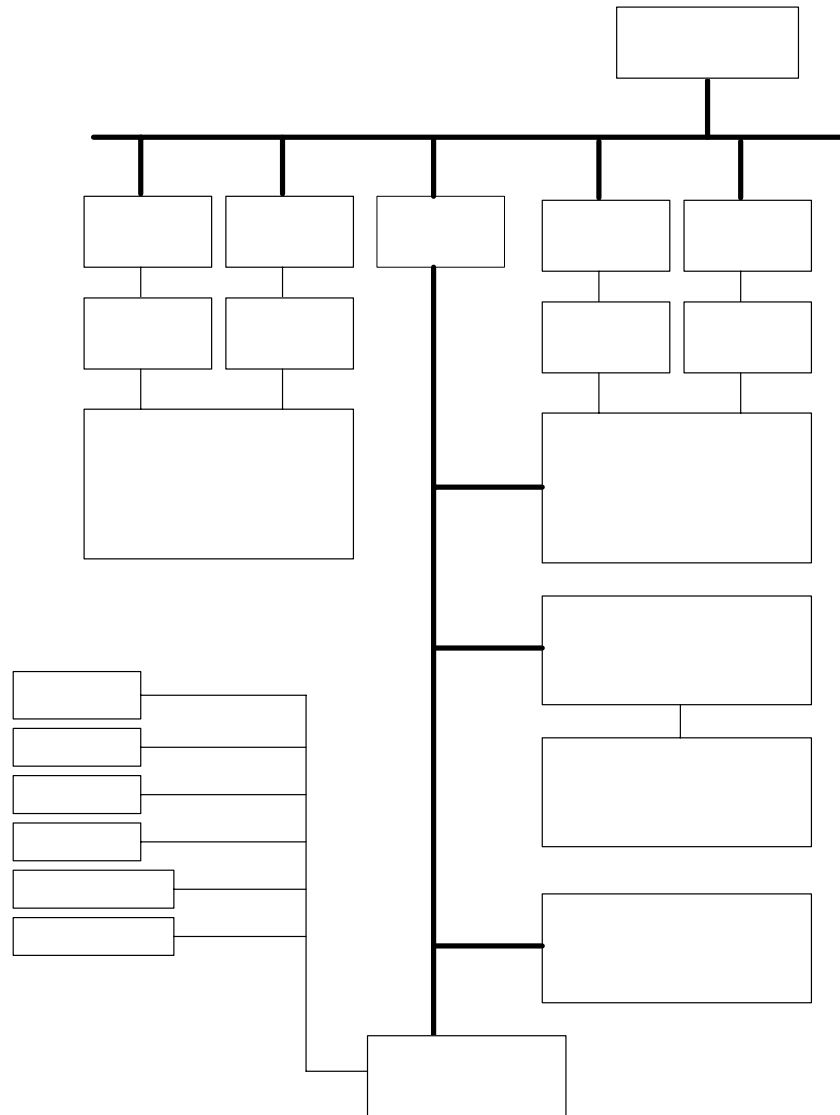
## ■ SoC

- + Complex integrated circuit or integrated chipset, which combines the major functional elements or subsystems of a complete end product into a single entity
- + RISC processor, DSP
- + On-chip communication structures
  - Processor bus, peripheral bus
- + On-/off-chip memory
- + HW-based accelerating units
- + Analog components
- + Sensor and actuators
- + Optical communications interfaces

## ■ SoC Design

- + Encompass HW and SW components
- + RTOS (Real-Time OS), device driver, middleware, application
- + HW-SW trade-off, partitioning decision, SW architecture

# A Typical SoC Device for Consumer Applications





## ■ FPGA (Field-Programmable Gate Arrays)

- + Reconfigurable logic part
- + Highly flexible design
  - Ex) Some algorithms can be partitioned into SW part (RISC processor), HW part (Reconfigurable logic)
- + Avoid expensive NRE (NonRecurring Engineering)

## ■ MPGA (Metal-Programmable Gate Array)

- + Intermediate form of SoC between ASIC and FPGA
- + Slower design creation and higher NRE than FPGA
- + Better cost, performance and energy consumption than FPGA



## ■ IP reuse

- + Impossible that the design of SoC started from scratch
- + Design productivity gap
  - Difference between the rate of increase of complexity and the rate of increase in designer productivity
  - IP reuse can reduce the design productivity gap.
  - Reduce the design risk and time
- + Designer creates SoC products that span multiple design disciplines and domains

## ■ Problems

- + Standards for IP creation, evaluation, exchange, and integration
  - VSIA (Virtual Socket Interface Alliance)
    - Hard and soft IP
    - Lack of formal and acknowledged adoption of it IP standards
    - Many companies have reuse programs internally.
- + Business issue
  - IP evaluation, purchase, delivery, use
  - VCX (Virtual Component Exchange)
  - Ad hoc supplier



- Virtual Socket or Virtual Component
  - + Design interface which an IP core must satisfy, and design models integration information
  - + Conform in the development and verification processes to well-established design processes and quality standards
  - + Conform to one or other well-accepted standards for IP reuse
  - + Fabricated at least once, and characterized post-fabrication to ensure that they have validated claims
  - + Reused at least once by an external design team, and usage reports and feedback should be available
  - + Rated for quality using an industry standard quality metric such as OpenMORE or the VSI Quality standard
- Focused on defining the standards and processes to turn the ad hoc reuse into a well-understood and reliable process for acquiring and reusing virtual components





## ■ Platform-based design

- + A planned design methodology which reduces the time and effort required, and risk involved, in designing and verifying a complex SoC
- + More integrated approach
- + Assembles groups of components into reusable platform architecture
- + Advantages
  - Increase in design productivity
  - Reduction in risk
  - Ability to utilize pre-integrated virtual components from other design domains more easily
  - Ability to reuse SoC architectures created by experts
- + Industrial platforms
  - Full application platforms
    - Philips Nexperia, TI OMAP – vehicle for specific product domains
  - Processor-centric platforms
    - ARM PrimeXsys
  - Reconfigurable or highly programmable platforms
    - Xilinx Platform FPGA, Altera's SOPC



## ■ SoC design process

### + Design the basic platform

- Enough implementation must be done to allow the platform and its constituent libraries to be fully characterized and modeled for reuse.
- Appropriate configuration programs or scripts to allow automatic creation of a configured platform during derivative design

### + Create and qualify the derivative design based the basic platform

## ■ Meta-platform

### + Platform FPGAs, SOPC

### + Obtain a basic set of IP cores such as IP-embedded processors, on-chip buses, special IP blocks, and a variety of other pre-qualified IP blocks from companies such as Xilinx and Altera.

### + Customize their own application platform by adding application domain-specific IP libraries

### + Derivative design teams configures it and selects the IP blocks.





## ■ SoC requirements analysis

- + Defining and specifying a complex SoC
  - Marketing definition of the end product
  - Resulting characteristics (Functional, non-functional)

## ■ SoC architecture

- + Defining the basic structure of the desired SoC
- + Decide the communication architecture
- + Choose processor

## ■ System-level design

- + HW-SW partitioning
- + Construction of system-level models
  - Ad hoc C/C++ based models -> SystemC model
- + Analysis of functioning and non-functional attributes through simulation and other analytical tools
- + Identify the required IP blocks



- IP acquisition
  - + Ensure completeness and correctness of IP prior to use
    - Resolve any problems with quality early with suppliers
  
- Build a transaction-level golden testbench
  - + Form the basis for a more elaborated design model using transaction-level abstractions
  - + Verify the micro-architecture of the design and detailed design models for HW IP at the HDL level
  
- Define the SoC SW architecture
  - + OS, device driver, middleware
  
- Configure and floorplan SoC micro-architecture
  - + Deal with a more physical and detailed logical basis
    - All the IP blocks are properly and fully configured
    - The basis micro-architectures have been fully defined and configured
    - HW implementation can proceed



- DFT architecture and implementation
  - + Implement the test architecture
  - + Standard test interface (such as JTAG ports)
  
- AMS (Analog/Mixed-Signal) HW implementation
  - + AMS blocks use them to interface to the external world
  - + Considerable works in defining how AMS IP blocks should be created to allow them to be more easily integrated into mainly digital SoCs
  
- HW IP assembly and integration
  - + Assembling design blocks agreed on architecture for communications, bussing, clocking, power, etc.
  
- SW assembly and implementation
  - + Assembling the SW IP and validating it to conformance to interfaces and expected operational quality
  - + Verify as much of the SW in its normal system operating context as possible



## ■ HW and HW-SW verification

- + Largest consumers of design time and effort
- + A large number of different verification tools and techniques
  - SW-base simulation, HW emulator, HW accelerator, FPGA, bonded-core-based rapid prototyping
- + Formal technique
  - Equivalence checking, model/property checking
- + Mixed approaches to HW-SW verification
  - ISS (Instruction Set Simulators) + HW emulator
- + Assertion-based verification
  - Certain properties must be satisfied
  - Certain error condition never occur
  - Semiformal verification
    - Formal property checking + simulation-based assertion checking
- + How do we know that verification is done?

## ■ Final SoC HW assembly and verification

- + Final place and route of the chip, any hand-modifications required, final physical verification
  - Design rule checking

## ■ Fabrication, testing, packaging, and lab verification

- + Verification of system SW running in context of the HW design



- A key aim of IP reuse and of platform-based design
  - + Make the “back end design implementation processes” easier
  - + Shift the major design phase for SoC up in time and in abstraction level to the system level
- The fundamental nature of IP-based design of SoC has a stronger influence on the system level.
- Major tasks
  - + DSE (Design Space Exploration)
    - Deciding on and validating the basic system architecture and choice of IP blocks
  - + SoC platform customization for a particular derivative
  - + Partitioning
    - HW-SW
    - SW-SW
- Function-architecture codesign
  - + The functional intent of the system
  - + The architectural structure of the system
  - + Build explicit mappings between the functional view of the system and the architectural view





- Current SoC architecture deal in fairly traditional hierarchies of standard on-chip buses
  - + ARM's AMBA, IBM's CoreConnect
- NoC (Network-on-Chip) communication architectures
  - + Based packet switched network
  - + Research topics



## ■ Computation

### + Primary processor used in SoC

- Embedded RISCs
  - ARM processors, PowerPC, MIPS
- Configurable processors
  - Tensilica, ARC

### + Embedded DSP

- TI, Motorola, ParthusCeva

### + In future SoCs, they may incorporate networks of heterogeneous configurable processors to offer large amounts of computational parallelism

## ■ Memory

### + Level 1 cache memory is widely used.

### + Consider the structure, size, and configuration of the memory hierarchy



## ■ Issues of the design reuse aspects of SoC

### + The issue of IP quality

- VSIA, OpenMORE have only a limited extent.
- Need the formal certification process

### + The issue of reusable IP creation

- IP is not created for reuse
- Reusable IP block has 50 to 200% more efforts needed than IP used only once.
- Planned and systematic IP reuse and investment give a high chance of achieving significant productivity



- Reduce productivity gap
  - + High level of design reuse
  - + Reuse the existence of the third party and internal IP groups

# THANK YOU