Hardware-Software Codesign

1. Introduction

Lothar Thiele
Contents

- What is an Embedded System?
- Levels of Abstraction in Electronic System Design
- Typical Design Flow of Hardware-Software Systems
Embedded Systems

Embedded systems (ES) = information processing systems embedded into a larger product

Examples:

Main reason for buying is **not** information processing
Embedded Systems

- external process
- human interface
- embedded system
- sensors, actuators
Parallel and Distributed Target Platforms
Example: Intel

48 cores vs. 4 cores
More Examples

Intel Xeon Phi
(5 Billion transistors, 22nm technology, 350mm² area)

Oracle Sparc T5
Embedded Multicore Example

Recent development:

- Specialize multicore processors towards real-time processing and low power consumption
- Target domains:
Multiprocessor systems-on-a-chip (MPSoCs)

SH-MobileG1: Chip Overview

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Die size</td>
<td>11.15mm x 11.15mm</td>
</tr>
<tr>
<td>Process</td>
<td>90nm LP</td>
</tr>
<tr>
<td></td>
<td>8M(7Cu+1Al)</td>
</tr>
<tr>
<td></td>
<td>CMOS dual-Vth</td>
</tr>
<tr>
<td>Supply voltage</td>
<td>1.2V(internal), 1.8/2.5/3.3V(I/O)</td>
</tr>
<tr>
<td># of TRs, gate, memory</td>
<td>181M TRs, 13.5M Gate, 20.2 Mbit mem</td>
</tr>
</tbody>
</table>

©2007. Renesas Technology Corp., All rights reserved.

MPSoC '07

Everywhere you imagine.
RENEASAS

Multiprocessor systems-on-a-chip (MPSoCs)
Multiprocessor systems-on-a-chip (MPSoCs)

Samsung Galaxy S6
- Exynos 7420 System on a Chip (SoC)
- 8 ARM Cortex processing cores
  (4 x A57, 4 x A53)
- 30 nanometer: transistor gate width
Multiprocessor systems-on-a-chip (MPSoCs)

Samsung Galaxy S6
- Exynos 7420 System on a Chip (SoC)
- 8 ARM Cortex processing cores (4 x A57, 4 x A53)
- 30 nanometer: transistor gate
Zero Power Systems and Sensors
Zero Power Systems and Sensors
Comparison

- **Embedded Systems**
  - Few applications that are known at design-time.
  - Not programmable by end user.
  - Fixed run-time requirements (additional computing power not useful).
  - Criteria:
    - cost
    - power consumption
    - predictability
    - meeting time bounds
    - …

- **General Purpose Computing**
  - Broad class of applications.
  - Programmable by end user.
  - Faster is better.
  - Criteria:
    - cost
    - average speed
Design Challenges

- **Challenges in the design of embedded systems**
  - increasing *application complexity* even in standard and large volume products
    - large systems with legacy functions
    - mixture of event driven and data flow tasks (see next chapter)
    - examples: multimedia, automotive, mobile communication
  - increasing *target system complexity*
    - mixture of different technologies, processor types, and design styles
    - large systems-on-a-chip combining components from different sources, distributed system implementations
  - numerous *constraints and design objectives*
    - examples: cost, power consumption, timing constraints, temperature
Implementation Alternatives

- General-purpose processors
- Application-specific instruction set processors (ASIPs)
  - Microcontroller
  - DSPs (digital signal processors)
- Programmable hardware
  - FPGA (field-programmable gate arrays)
- Application-specific integrated circuits (ASICs)
Contents

- What is an Embedded System?
- Levels of Abstraction in Electronic System Design
- Typical Design Flow of Hardware-Software Systems
Abstraction, Models and Synthesis

**Model**
- Formal description of selected properties of a system or subsystem
- A model consists of data and associated methods

**Classification of models**
- Degree of abstraction, granularity
  - hardware: system, architecture, logic, transistor,
  - software: module, block/class, function/method, ...
- View
  - behavior, structural, physical

**Synthesis**
- Linking adjacent levels of abstraction (refinement)
- Stepwise adding of structural information
Levels of Abstractions

- **System**
  - **Function**
  - **Object Code**
  - **Process/Module**
  - **Architecture**

- **Levels of Abstractions**
  - **SW**
    - Function
    - Object Code
  - **HW**
    - Architecture
    - RTL

- **Structure**
  - Gate-level models
  - Switch-level models
  - Circuit-level models
  - Device-level models
  - Layout models

- **Behavior**
Contents

► What is an Embedded System?

► Levels of Abstraction in Electronic System Design

► Typical Design Flow of Hardware-Software Systems
System Design

Specification → System Synthesis

System Synthesis:
- Specification
- Estimation

SW-Compilation
- Instruction Set
- HW-Synthesis

Intellectual Prop. Code
- Instruction Set

Machine Code
- HW-Synthesis

Net lists
- Intellectual Prop. Block
Fixed Processor Architecture

- Specification
- System Synthesis
- SW-Compilation
- Instruction Set
- HW-Synthesis
- Estimation
- Intellectual Prop. Code
- Machine Code
- Net lists
- Intellectual Prop. Block
Application Specific HW Block

- Specification
  - System Synthesis
    - SW-Compilation
      - Intellectual Prop. Code
        - Machine Code
    - Instruction Set
      - Instruction Set
    - HW-Synthesis
      - Intellectual Prop. Block
        - Net lists

Estimation
System-Level Design

System-level design is a complex synthesis tasks
- software synthesis and code generation
- hardware synthesis
- interface and communication synthesis
- hardware/software partitioning and component selection
- hardware/software scheduling

Major Components:
- application specification
- design space exploration and system optimization
- estimation
The Mapping Problem
HW/SW Mapping and Scheduling

- **Hardware/software mapping**
  - Partitioning of system function to programmable components (software), hard-wired or parameterized components (hardware) or application specific instruction set processors.

- **Similarity** to scheduling and load distribution problem in real-time operating systems
  - Time constraints, context switch and context switch overhead, process synchronization and communication

- **Differences** to real-time operating systems
  - Larger design space with very different solutions
  - High optimization requirements (motivation for hardware design)
  - Underlying hardware is not fixed
**HW/SW Mapping and Scheduling**

- Similarity to allocation (or load distribution) problem in high-level synthesis (or real-time operating systems)

![Diagram showing HW/SW mapping and scheduling](image)
The principle of synthesis based on abstraction only makes sense if there are powerful estimation methods available:

- Estimate properties of the next layer(s) of abstraction.
- Design decisions are based on these estimated properties: If the estimation is not correct (or not accurate enough), the design will be sub-optimal or even not working correctly.