Hardware/Software Codesign of Embedded Systems

DESIGN METHODOLOGIES

Voicu Groza
SITE Hall, Room 5017
562 5800 ext. 2159
Groza@SITE.uOttawa.ca
DESIGN METHODOLOGIES

• Embedded system design process
• Design Methodology
• Introduction to Hardware/Software CoDesign
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
  – flexibility requirements
  – 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 (IP market)

• numerous constraints and design objectives; examples: cost, power consumption, timing constraints, dependability

• reduced and overlapping design cycles
Embedded System Requirements

• Reactive systems.
  - The system never stops.
  - The system responds to signals produced by the environment.

• Real-time systems.
  - Timing constraints on task execution.
  - Hard and soft constraints.
Example: Engine Control Unit (ECU)

Task: control the torque produced by the engine by the timing fuel injection and spark.

- Major constraints:
  - Low fuel consumption
  - Low exhaust emission
ECU
Control injection time (3 sub-tasks)

compute air flow

Throttle position
Engine speed
Air temperature
Air pressure
Engine temperature

Compute injection time

air flow

Look-up table

injection time

drive actuators

signals
ECU option 1

CPU has to:
- Process input data
- Compute outputs
- Control actuators

May not meet timing requirements
ECU option 2

- CPU has to
  - Process input data
  - Compute outputs
- FPGA Control actuators

Analog inputs $\rightarrow$ A/D $\rightarrow$ 16 bit CPU $\rightarrow$ FPGA $\rightarrow$ Actuations

Digital inputs $\rightarrow$
ECU option 3

- DSP Process input data
- CPU Computes outputs
- FPGA Control actuators
Embedded system design process

Requirements definition

Specification

System architecture development

SW development
- application SW
- compilers etc.
- operating syst.

Interface design
- SW driver dev.
- HW interface synthesis

HW design
- HW architecture design
- HW synthesis
- Physical design

Integration and test

support
(CAD, test, ...)

customer/ marketing
system architect

SW developer

HW developer

Reused components

SW

HW

Voicu Groza, 2008
Observations in the design process

- Increasingly concurrent design of hardware and software with partially incomplete or variable specification
  => tight and permanent cooperation of hardware and software designers, system architects and customer/marketing required

- Narrow time-to-market windows require a safe "first-time-right" design process
  – early detection of systematic design flaws is crucial
  – reliable design times and precisely predictable product data are more important than design time minimization
  *prerequisite: reliable estimations => designer experience*

- Increased productivity through reuse of components and functions
  – function and component libraries required
  – problem: function migration between different technologies, between hardware and software
Definition of Design Methodology

* A design methodology defines the models, methods, and tools and their relations employed in the design process
* Design models are used to represent design entities
* Design languages are used to express design models
* Modeling frames define modeling concepts and classes of modeling languages
* Methods define the design transformations
* Tools support the designer in modeling, analyzing, and transforming the design entities
Abstraction of Design Transformations

Methodology

Method A → Method B → Method D → Method F

Method C → Method E

Tool 1 → Tool 3

Tool 2
Modeling Frames in a Methodology

**Specification Frame**
- FSM
- OO model

**Design Frames**
- Algorithms
- Interconnection model

**Architecture Frames**
- Architecture model
- HWSW interconnection model
Items a Design Methodology Usually Specifies

* Syntax and semantics of inputs and outputs of design steps
* Methods for transforming input to output
* Kind of components to be used in the design implementation
* Definition and ranges of constraints
* Mechanism for selection of architectural styles
* Control strategies (scenarios or scripts)
* A methodology must make a conscious tradeoff between an authoritarian and a liberal style
Design methodology - Components

- Input description style
- Synthesis tools (and algorithms) at each abstraction level
- Design libraries and components to be used in design implementation
- Specification and propagation of constraints
- User Interaction mechanisms (knobs and gadgets)
- Conceptualization environment
- Verification methodology at each abstraction level
- Reiteration hints if constraints are not satisfied
History

Languages and tools
- Structured programming
- Compilers
- Pascal
- Smalltalk
- Ada
- Harel Statechart
- Specification frameworks
- Methodology frameworks

Methodologies
- Structured Design
- Structured Analysis
  (Jackson diagram)
- Structured design with Ada
- Statemate
- OOD, OOA
- OMT
- UML by OMG
- Concurrent Engineering
- Total Quality Management
Design Methodology – Past
Capture-and-Simulate

System specification (natural language)

Informal exploration

Processor
Informal spec.

SW design

Processor
C code

ASIC 1
Informal spec.

HW design

ASIC 1
RTL struct

Implement, Integrate & Test

ASIC 2
RTL struct

ASIC 2
Informal spec.

Very costly iterations
Problems with Past Design Method

- Lack of unified hardware-software representation
- Partitions are defined a priori
  - Can't verify the entire system
  - Hard to find incompatibilities across HW-SW boundary
- Lack of well-defined design flow
  - Time-to-market problems
  - Specification revision becomes difficult

=> Need Hardware-Software Co-Design
Hardware/Software CoDesign

- **Hardware/software CoDesign:**
  Combined design of HW & SW

- **Why Codesign?**
  - Design process optimization
    ⇒ Increased design productivity
  - Design optimization
    ⇒ More design alternatives can be explored
    ⇒ A good solution can be found by balancing HW/SW
    ⇒ Improved product quality
  - To meet strict design constraints
    ⇒ power dissipation.
    ⇒ physical constraints, such as size, weight,
    ⇒ safety and reliability constraints.
  - To design systems on a chip.
Tasks & Requirements

• **Tasks** (… automatic!?)
  • Co-specification & Co-modeling
  • Co-design process integration & optimization
  • Co-verification
  • Design optimization & co-synthesis

• **Requirements:**
  • Implementation Independent
    ⇒ Stress system design issues
    ⇒ Allow different hardware and software styles
  • Unified design approach
    ⇒ Facilitates system specification
    ⇒ Easy HW-SW trade-off evaluation
    ⇒ Flexible HW-SW partitioning

• **All co-design problems require computer support**
  ⇒ Computer aided hardware/software co-design is the new challenge
Design Methodology - Present

Insufficient exploration
- non-optimized design
- spec. errors

System level description

HW & SW design

HW & SW implementation

System specification (natural language)

Informal exploration

Processor functional spec.

ASIC 1 functional spec.

ASIC 2 functional spec.

SW Synthesis

HW Synthesis

Processor C code

ASIC 1 RTL

ASIC 2 RTL

Compilation, Logic synthesis, Physical design

COSIMULATION
Another View of CoDesign:

Design of HW & SW using

- the same methods/tools
- the same people
Design Methodology – Future Describe-and-Synthesis

System functional specification

Exploration
Specification refinement

System level description

HW & SW design

HW & SW implementation

Behavior 1  Behavior 2  Behavior 3

Transformation Allocation Partitioning Estimation
Transformation Allocation Partitioning Estimation

Memory  Interfacing  Arbitration  Generation
Memory  Interfacing  Arbitration  Generation

Processor functional spec.
Processor functional spec.
Processor functional spec.

ASIC 1 functional spec.
ASIC 1 functional spec.
ASIC 2 functional spec.

ASIC 1 RTL
ASIC 2 RTL.

SW Synthesis
HW Synthesis

Compilation, Logic synthesis, Physical design
Compilation, Logic synthesis, Physical design

C code
C code

COSIMULATION
COSIMULATION
Codesign Terminology

**Specification**
- Describing the functionality of the design

**Partitioning**
- Assigning sub-behaviors to the system component
- Sequence of steps from specification to implementation

**Design Methodology**

**Synthesis**
- Converting the functionality to an architecture

**CoSimulation**
- Simultaneously simulating custom software and hardware parts of the design
Computer-Aided Codesign

• Modeling, validation and verification
  - Co-specification.
  - Co-simulation.
  - Co-verification.

• Architecture selection:
  - Allocation of system components (processing elements, storing elements, and communication elements)
  - Hardware/software partitioning.

• Cosynthesis:
  - Hardware synthesis.
  - Software compilation and code generation.
  - Interface synthesis.
  - Modification of hardware/software boundary.
Modeling of Embedded Systems

• Heterogeneous nature of embedded systems.
• No global system models, except for restricted domains.
  – Finite-state machines
  – Petri nets.
  – Flow graphs.
• Functional simulation models (software programs) vs. specification models (e.g., SDL, VHDL, Statecharts).
HW/SW Partitioning

- Speed-up software execution.
  - By migrating software functions to dedicated hardware.

- Reduce cost of hardware implementation.
  - By migrating hardware functions to software.

- Guarantee real-time constraints.
  - By migrating the timing-critical portion to ASIC.
Code Synthesis after Partitioning

- Software functions identified by program threads.
- A single processor requires thread serialization or interleave.
- Scheduling of threads and instructions.  
  - To satisfy performance or timing constraints.
- System-level run-time scheduler to synchronize software and hardware functions.
Codesign of General-Purpose Processors

• Architectural support for operating systems.
• Software design for multiprocessors.
• Cache design and tuning:
  - Cache size selection.
  - Control schemes.
• Pipeline control design:
  – Hardware control mechanisms.
  – Compiler design.
NEXT....

State-oriented models
• Finite-state machines ++
Specify - Explore - Refine

• Three step design methodology growing
  – Specify desired functionality
  – Explore alternative design options
  – Refine specification for next level of design

• Applicable at system, behavioral and RT levels
  – Hierarchical modeling methodology

• One level’s implementation is the next level’s specification