

Efficient Modeling of Embedded Systems using Computer-Aided Recoding

Rainer Dömer
doemer@uci.edu

With contributions by Gunar Schirner, Pramod Chandraiah

Center for Embedded Computer Systems
University of California, Irvine




Outline


- Embedded Systems
- Design Challenges
- Computer-Aided Recoding
- Interactive Source Recoder
- Experiments and Results
- Conclusions

Embedded Systems


- System embedded into another system
 - Constraints from external input
 - Application specific
- Omnipresent in our environment
 - In many application domains
 - In 2005 [Source Netrino]
 - Only 2% of all processors in workstations
 - Remaining 8.8 billion in embedded systems
 - Pervasive




Source: Miele
EECS Colloquium, 04/20/2011



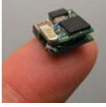
Source: Philips




Source: www.trouper.com



Source: www.medicacorp.com/
R. Doemer



Source: P. Chou, UCI



Source: Edumaticator

3

Embedded Systems

- Product challenges
 - Often mobile
 - Performance constrained
 - Tightly coupled software and hardware
 - Specialized
 - Complex
 - E.g. Car: MB E-class
 - » 55 ECUs
 - » 5 busses



Source: Motorola Inc



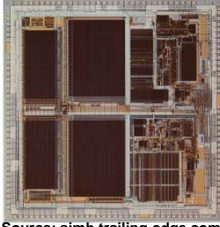
Source: Daimler

R. Doemer


4

Embedded System Design

- Design Advantages
 - Application known at design time
 - Environment known at design time
 - Allows for customized / optimized solution
 - Improved performance
 - More functionality
 - At lower power
- Customized HW/SW
 - Application Specific Integrated Circuit (ASIC)
 - Multi-Processor System-on-Chip (MPSoC),
 - Complete embedded system integrated on a chip
 - Containing multiple processors
 - Field Programmable Gate Array (FPGA)
 - Circuit board with off-the-shelf-components



Source: simh.trailing-edge.com



Source: Xilinx

EECS Colloquium, 04/20/2011
R. Doemer
5

Embedded System Design

- Productivity Gap
 - Hardware design gap
 - + Software design gap
 - = System design gap

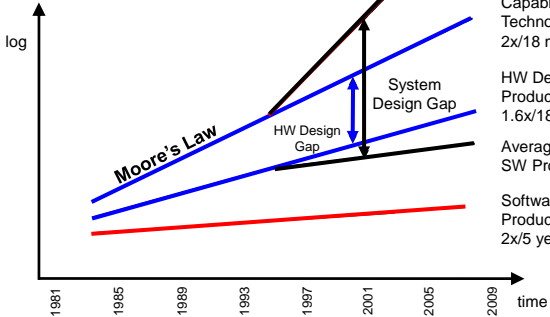
Additional SW required for HW
2x/10 months

Capability of Technology
2x/18 months

HW Design Productivity
1.6x/18 months

Average HW + SW Productivity

Software Productivity
2x/5 years



(source: "Hardware-dependent Software", Ecker et al., 2009)

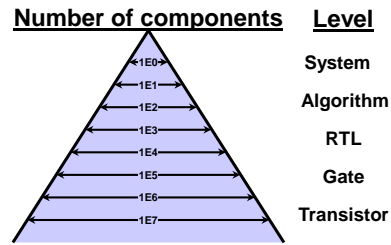
EECS Colloquium, 04/20/2011
R. Doemer
6

Embedded System Design

- How can we overcome the productivity gap?

International Technology Roadmap for Semiconductors (ITRS) 2004:
higher-level abstraction and specification is the first promising solution

- System Level Design
 - Unified HW and SW design
 - Higher level of abstraction
 - Fewer, more complex components
 - Maintain system overview
 - Without overwhelming details
 - Compose a system of algorithms
 - System Level Design Languages
 - SpecC [Gajski et. al, 2000]
 - SystemC [Groetker et. al, 2002]



Source: "System Design: A Practical Guide with SpecC", 2001

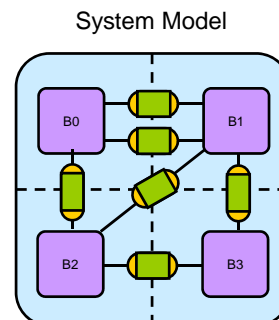
EECS Colloquium, 04/20/2011

R. Doemer

7

Embedded System Design

- System Level Modeling
 - Abstract description of a complete system
 - Hardware + Software
- Key Concepts in System Modeling
 - Explicit Structure
 - Block diagram structure
 - Connectivity through ports
 - Explicit Hierarchy
 - System composed of components
 - Explicit Concurrency
 - Potential for parallel execution
 - Potential for pipelined execution
 - Explicit Communication and Computation
 - Channels and Interfaces
 - Behaviors / Modules



EECS Colloquium, 04/20/2011

R. Doemer

8

Computer-Aided Recoding

- Embedded System Design Flow
 - Input: System model
 - Output: MPSoC platform
- Actual Starting Point
 - C reference code
 - Flat, unstructured, sequential
 - Insufficient for system exploration
- Need: System model
 - System-Level Description Language (SLDL)
 - Well-structured
 - Explicit computation, explicit communication
 - Potential parallelism explicitly exposed
 - Analyzable, synthesizable, verifiable
- Research: Automatic *Re-Coding*
 - How to get from flat and sequential C code to a flexible and parallel system model?

C Code

Re-Coding ↓

System Model

↓

MPSoC Platform

EECS Colloquium, 04/20/2011
R. Doemer
9

Motivation

- Extend of Automation
 - Refinement-based design flow
- Automatic
 - Specification model down to implementation
 - Example: SCE (mostly automatic)
 - MP3 decoder: less than 1 week
- Manual
 - Source code transformations
 - C reference code to SpecC specification model
 - MP3 decoder: 12-14 weeks!
- Automation Gap
 - 90% of overall design time is spent on re-coding!
- Proposal: Automatic Recoding

C Reference Code

Manual ↓ 12-14 weeks

Recoding (dashed box)

Specification Model

Architecture Exploration

Automatic ↓ Less than 1 week

Architecture Model

Comm. Exploration

Communication Model

Implementation

Source: System Design: A Practical Guide with SpecC
R. Doemer 10

EECS Colloquium, 04/20/2011
R. Doemer
10

Problem Definition

- How to get from flat, sequential C code to a flexible, parallel system model?
- Recoding
 - Create structural hierarchy
 - Expose concurrency (parallelize/pipeline)
 - Partition code and data
 - Expose communication
 - Eliminate pointers
 - Make the code compliant to the design tools, ...
- Our approach
 - Computer-Aided Recoding
 - Interactive source code transformations

C code

Recoding

System Model

EECS Colloquium, 04/20/2011 R. Doemer 11

Computer-Aided Recoding

- Complete Automation is Infeasible!
 - Today's parallelizing compilers are largely ineffective
 - Heterogeneous architectures
 - Complexity of embedded applications
 - Hard problems (eliminating pointers, exposing parallelism, etc.)
 - Modeling requires understanding of the application
 - Recoding is not a monolithic transformation
 - Multiple transformations in application-specific order
- Interactive Approach
 - “Designer-in-the-loop”
 - Designer can utilize application knowledge
- *Designer-controlled* Transformations
 - Designer makes decisions
 - Tool automatically transforms the source code

EECS Colloquium, 04/20/2011 R. Doemer 12

Overcoming the Specification Gap

- Recoding Transformations

Recoding

EECS Colloquium, 04/20/2011
R. Doemer
13

Overcoming the Specification Gap

- Recoding Transformations
 - Creating structural hierarchy [ASPDAC'08]
 - Code and data partitioning [DAC'07]
 - Creating explicit communication [ASPDAC'07]
 - Recode pointers [ISSS/CODES'07]

Create Hierarchy Partition Code and Data Expose Communication Recode Pointers ...

EECS Colloquium, 04/20/2011
R. Doemer
14

Interactive Source Recoder

- Implementation
 - Integrated Development Environment (IDE)
- *Cute* tool is a union of
 - Text editor
 - Abstract Syntax Tree (AST)
 - Parser
 - Transformations
 - Code generator

EECS Colloquium, 04/20/2011
R. Doemer
22

Interactive Source Recoder

- Interactive Environment
 - Scintilla + QT + AST + Transformations
- Basic editing
 - Syntax highlighting
 - Auto-completion
 - ...
- Recoding Transformations
 - Dependency analysis
 - Code and data splitting
 - Variable re-scoping
 - Port insertion
 - ...

EECS Colloquium, 04/20/2011
R. Doemer
28

Experiments and Results

- We have conducted various sets of experiments
- Goals
 - Responsiveness of the “compiler in the editor”
 - Estimated Productivity Gains
 - Extrapolation based on the number of lines of code changed
 - Measured Productivity Gains
 - Class of graduate students
- Design examples
 - GSM Vocoder (voice codec in mobile phones)
 - MP3 Decoder (audio decoder, e.g. iPod)
 - Fixed-point version
 - Floating-point version
 - JPEG Encoder (image encoder, e.g. digital camera)
 - ...

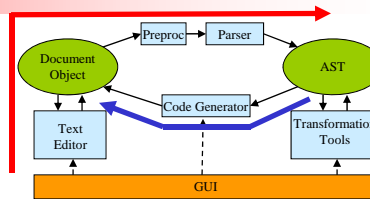
EECS Colloquium, 04/20/2011

R. Doemer

29

Experiments and Results: Responsiveness

- Why measure Responsiveness ?
 - To check feasibility
- Responsiveness
 - Response to designer actions
 - Time to synch AST
 - On editing
 - Time to synch Editor
 - On transformation
 - Depends on the size of the AST
- Design examples
 - JPEG, MP3, GSM
 - << 1 sec (on a 3 GHz Linux PC)
 - File I/O overhead (20%)



| Operation | Simple | JPEG | MP3 | GSM |
|---------------------|------------------|------------------|------------------|------------------|
| Lines of code | 174 | 1642 | 7086 | 7492 |
| Objects in AST | 1073 | 5338 | 31763 | 26009 |
| Synch AST | 0.15 secs | 0.19 secs | 0.68 secs | 0.55 secs |
| Synch Editor | 0.16 secs | 0.20 secs | 0.73 secs | 0.59 secs |

EECS Colloquium, 04/20/2011

R. Doemer

30

Experiments and Results

- **Productivity Gain**
 - Creating structural hierarchy
 - Manually
 - estimation
 - Automatically
 - measured
- **Results**
 - Manual time
 - weeks
 - Recoding time
 - minutes

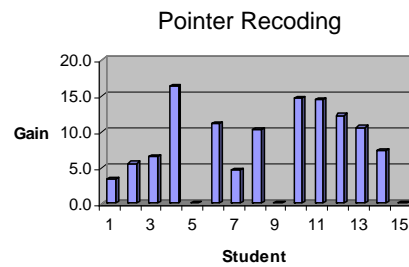
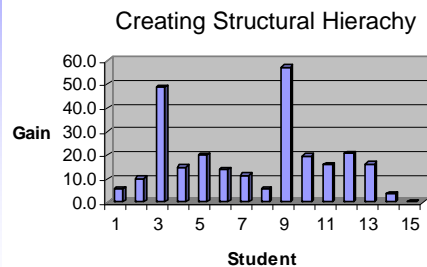
| Properties | JPEG | Float-MP3 | Fix-MP3 | GSM |
|---------------------|-----------|-----------|-----------|-----------|
| Lines of C code | 1K | 3K | 10K | 10K |
| C Functions | 32 | 30 | 67 | 163 |
| Lines of SpecC code | 1.6K | 7K | 13K | 7K |
| Behaviors created | 28 | 43 | 54 | 70 |
| Re-Coding time | ≈ 30 mins | ≈ 35 mins | ≈ 40 mins | ≈ 50 mins |
| Manual time | 1.5 weeks | 3 weeks | 2 weeks | 4 weeks |
| Productivity gain | 120 | 205 | 120 | 192 |

[ASPDAC'08]

➢ Significant productivity gains!

Experiments and Results: Productivity

- **Measured Productivity Gains**
 - Class of 15 graduate students
 - Recode an MP3 design example
 - Manually (given detailed instructions)
 - Automatically (using the Source Recoder)
- **Results**



– Productivity factors vary, but show significant gains!

Conclusions

- Embedded System Design
 - Start from higher level of abstraction
 - Need flexible system models in SLDL
- Motivation
 - Automation gap between C reference and SLDL system models
 - 90% of the overall design time spent on “coding” and “re-coding”
 - Need for design automation
- Problem
 - Complete automation is difficult
- Approach
 - *Computer-Aided Recoding* using Source Recoder
 - Designer-in-the-loop
- Results
 - Significant productivity gains
- Future work
 - Research and develop more transformations
 - Improve interactive graphical environment

References

- [ASPDAC'07] P. Chandraiah, J. Peng, R. Dömer, "*Creating Explicit Communication in SoC Models Using Interactive Re-Coding*", Proceedings of the Asia and South Pacific Design Automation Conference 2007, Yokohama, Japan, January 2007.
- [IESS'07] P. Chandraiah, R. Dömer, "*An Interactive Model Re-Coder for Efficient SoC Specification*", Proceedings of the International Embedded Systems Symposium, "Embedded System Design: Topics, Techniques and Trends" (ed. A. Rettberg, M. Zanella, R. Dömer, A. Gerstlauer, F. Rammig), Springer, Irvine, California, May 2007.
- [DAC'07] P. Chandraiah, R. Dömer, "*Designer-Controlled Generation of Parallel and Flexible Heterogeneous MPSoC Specification*", Proceedings of the Design Automation Conference 2007, San Diego, California, June 2007.
- [ISSS+CODES'07] P. Chandraiah, R. Dömer, "*Pointer Re-coding for Creating Definitive MPSoC Models*", Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis, Salzburg, Austria, September 2007.
- [ASPDAC'08] P. Chandraiah, R. Dömer, "*Automatic Re-coding of Reference Code into Structured and Analyzable SoC Models*", Proceedings of the Asia and South Pacific Design Automation Conference 2008, Seoul, Korea, January 2008.
- [TCAD'08] P. Chandraiah, R. Dömer, "*Code and Data Structure Partitioning for Parallel and Flexible MPSoC Specification Using Designer-Controlled Re-Coding*", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems vol. 27, no. 6, pp. 1078-1090, June 2008.
- [DATE'09] R. Leupers, A. Vajda, M. Bekooij, S. Ha, R. Dömer, A. Nohl, "*Programming MPSoC Platforms: Road Works Ahead!*", Proceedings of Design Automation and Test in Europe, Nice, France, April 2009.

Interested in learning more?

- EECS Graduate Courses
on Embedded System Design

EECS222A: *System-on-Chip Description and Modeling*

EECS222B: *System-on-Chip Design and Exploration*

EECS222C: *System-on-Chip Software Synthesis*

EECS222D: *System-on-Chip Hardware Synthesis*

- Course A is prerequisite for B, C, and D;
or: consent of instructor
- Offered regularly since Fall'07
- Instructors: Dömer (A, C), Gajski (B, D)