Towards Teaching Embedded Parallel Computing: An Analytical Approach

Zain Ul-Abdin and Bertil Svenson
High-Performance Embedded Computing

• Advanced embedded signal processing systems typically require:
  – High-performance
  – Low power
  – Low cost
  – High engineering efficiency

• There is a need for highly parallel computing technology

• However, the increasing programming complexity will affect the software and product development process

There is a need for advanced embedded systems designers to have in-depth and up-to-date knowledge of parallel computing
Teaching the Embedded Aspects

• Typical characteristics:
  – Real-time guarantees
  – Tough reliability and security requirements in safety critical applications
  – Tight integration of software and hardware development
  – Energy efficiency and battery life-time
  – Specific hardware augmentation with heterogeneous cores

• Emergence of manycore technology

• The aim of the Embedded Parallel Computing (EPC) course is to teach the students the methods to deal with the constraints imposed by embedded systems
Intended Learning Outcomes

• Describe the most important parallel architecture models and parallel programming models

• Practically demonstrate understanding by programming parallel computer systems intended for embedded applications

• Judge, evaluate and discuss how the choice of programming model and method influences, e.g., execution time and required resources,

• Relate the state of the art in the area

• Understand scientific articles in the area
Embedded Parallel Computing Course

• Offered in the Master’s program “Embedded and Intelligent Systems
• 7.5 ECTS credits i.e., 1/8 of an academic year
• Assumed Knowledge:
  – Computer organization, digital-logic design, and calculus
  – Common programming experience
• The course has been given for 10 years
  – Earlier with the name of Parallel Computer Architecture
  – The change in the course orientation towards manycore processors is reflected in the change of name
Traditional Approaches

• Stovepipe model
  – Architecture and Software as distinct courses usually in the junior/senior years

• Patt and Patel
  – Logic design, assembly language, plus C
  – Ideal as a pre-req for our course

• Problems with the traditional approaches
  – Changing CS scene
  – Needs rethinking at “core”
  – Infuse research interest
An Analytical Approach for the EPC course

• Discovery as opposed to instruction
• Learn the fundamental components before applying it to solve real-world problems
• Involves active learning by relying on:
  – Discipline-specific action
  – Problem-Solving skills
• ‘Learning by Doing’ metaphor
  – Emphasizes on the creative elements of the learners
• Based on constructivist learning theory
  – Engages the students in analyzing and building knowledge
• Story telling approach
  – Keeps the student interest alive
Teaching Format for the EPC Course

- Lectures
  - Imparts declarative knowledge
  - Used reflective technique for content development

- Laboratory and Project
  - Deals with delivering functioning knowledge by undertaking practical tasks

- Seminars
  - Develops knowledge-building discourse
Lectures Part

• Course literature
  – “Computer Architecture: A Quantitative Approach” by Hennessy and Patterson
  – Other complementary texts:
    • Books by Kornaros and Wolf
    • Research Articles exemplifying architectures and models of computations

• Topics taught:
  – Motivation for parallelism and energy-efficiency
  – Forms of parallelism: SIMD, MIMD, dataflow, …
  – Parallel programming models and MoC: MPI, CSP, KPN, …

• Incorporates interactive tasks
Laboratory and Project Part

• Implement parallel processing tasks on manycore HW platforms
  – Ambric: 45 brics (360 processors) @333 MHz
    • Programming languages
      • aJava
      • aStruct
    • Proprietary IDE
  – Epiphany: 16 cores @600MHz
    • Programming language
      – ANSI C
    • Eclipse based IDE
Laboratory and Project Part

• Open Virtual Platform (OVP) infrastructure for early design stage simulation

• Main features:
  – Easy to create:
    • Virtual platforms of many peripherals and many processors
    • Own processors and peripherals models
  – Fast instruction accurate simulations
  – Use for application, OS, embedded software development
Seminar Part

• Course participants make detailed studies on selected topics and conduct seminars
• Seminars are moderated by the teacher
• Seminar topic areas:
  – Processor arrays
  – Graphic processors
  – Reconfigurable architectures
  – Network-on-chip
  – Energy-Efficiency
  – Parallel programming models
Experiences

• Educate each other in an organized way!
• Inspires the students to contribute to research
  – MS theses
  – Internships
  – PhD studies
• Research orientation has changed gradually – to accompany needs in industry (e.g., manycore technology, models of computation, programming languages, SW development tools)
• Produces MS students with the right competence
  – Our former students have joined prominent research groups and industry all over the world
Future Developments

• Laboratory in Embedded Environment
  – Introduce high-level programming languages and tools for the manycore platforms in place of proprietary tools
  – Incorporate tools to support distance-learning students

• Adopt blended course model
  – Lectures and seminars delivered through distance learning platforms
Concluding Remarks

• Presented an analytical approach towards imparting declarative and functioning knowledge
  – Combining computer architecture and embedded software
  – Take the journey together exploring hardware and the software abstractions
  – Emphasize connectedness

• An integrated course
  – Interactive lectures
  – Lab sessions with self-directed project
  – Seminars

• Develops reasoning process among students and allows them to apply the gained knowledge practically
Thank you for your attention!

Email: Zain-ul-Abdin@hh.se