Some goals 10 years ago

- unified&systematic HW/SW design starting from single specification
  - trend in SW engineering and high-level synthesis at that time

- design assumption: software on “hard“ processor(s) with “soft“ HW components
  - influenced by strong shake-out in processor architectures
  - few embedded architectures used as “work horses” (8051, 68000, ...)
  - ASIPs only in large volume products with extremely narrow cost margins
  - promize of fast HW component generation by a highly automated design process (high-level synthesis, automated layout)

⇒ focus on EDAs
What we did right

• systematic problem formalization to apply EDA algorithms
• systematic identification and research into EDA bottlenecks (memory, communication, interfacing)
• architecture template based approach to limit design space
• tried to include software engineering and RTOS
• support of design iteration, refinement, and constraint management
  – no waterfall design processes
• aimed at coherent view of the design process
What we learned

• underestimated EDA problems
  – HW design automation did hardly move beyond RT level
    • HW was „harder“ then we thought (arch. variety, wiring cost, analog, RF, ...)
    • high-level synthesis had limited impact
  ⇒ HW reuse became the dominant productivity driver

• reuse leads to different design and EDA problems
  ⇒ more focus on design integration rather than component design

• HW reuse taken to system level: platform based design
  ⇒ HW flexibility and APIs become major issue

• lack of good business models
  ⇒ few point tools accepted
So, academia can work on the right things now?

- lack of infrastructure
  - reuse requires access to libraries
    libraries still very limited or not accessible and very limited – any work in design space exploration is still a problem
  - lack of benchmarks
    • little work beyond TGFF
    • industry often does not have benchmarks itself due to lack of systematic system level design style
- lack of focus in commercial system level EDA
  - companies buy each other, rewrite tools, come up with new languages, MoC
  - more worry about business models and than about technical progress
  - still major library problem
What should academia work on?

- HW/SW platform API and middleware become key co-design research fields
  - HW dependent SW used to increase portability and platform reuse
  - resource management, power management, and communication primitives for complex MpSoC structures
  - not appropriately covered by classical RTOS

- higher level of automation required for reconfigurable and evolving (organic) computing
  - design tools become part of the target system
  - partly autonomous design decisions

- how could we work in these areas if research infrastructure was so difficult up to now?
Get involved in ES design

• use application projects as research drivers
  – many open problems in system performance optimization, resource sharing, communication, ..., to which we can contribute
  – develop understanding for upcoming problems and get feedback
  – find champions in industry and academia