Goals for this Panel

- System-level design: are we solving the right problems with the right solutions at the right time?
  - Certain technical solutions were too early or too late for the problem they were trying to solve
- Make a self-critical evaluation of what was done right or wrong in co-design / system-level research since its beginnings
- What are the main system-level design problems today and in the future?
- What can and should be done to address them?
- Continue doing it, or should we go back to the drawing board and think of brand new research approaches?
A quick trip down memory lane...

- **Circa 1993**
  - Behavioral synthesis on steroids (lots of papers, press coverage, tools)

- **Circa 1996**
  - Behavioral synthesis not really materializing in the market place
  - Funds (for industry and research grants) drying out
  - Academics shift focus to HW/SW Co-Design

- **Circa 1997**
  - Co-Design becomes very popular
  - Realization that Co-Design is a much bigger problem (e.g., software, architecture, power, performance, etc.)
A quick trip down memory lane...

- **Circa 1998**
  - Some approaches really start to pay off (e.g., co-simulation techniques, models of computation)

- **Circa 2002**
  - Apply “old” solutions to bigger co-design problems (in the hope they will work this time...)
  - Several companies at DAC this year showing products with behavioral synthesis techniques under the covers
Some Lessons Learned

- **HW/SW co-design starting from single specification**
  - Interesting from an academic point of view, but...
  - IT NEVER HAPPENS!

- **Systems don’t come in one big executable spec**

- **Links to implementation are essential**
  - HW/SW co-design, then throw it over the wall to designers, does not work!
Some Lessons Learned

- Algorithmic exploration can give you a much bigger bang for the effort, ... and make it architecture-aware
  - “What?! You’re willing to give me TWO integer multipliers?? If I’d known that, I would have inverted those two loops and rewritten the second computation. By the way, if I do that, 16-bit multipliers will do it just as fast, I won’t need 32-bit units.”
  - This is hard enough to do manually when the architecture is fixed. Automatic joint algorithmic and architecture exploration is beyond us.

(courtesy of Josh Fisher)
Key Challenges for Systems Design

Relative Effort by Designer Role

- Software effort overtakes hardware effort at 130 nm
- Architecture effort overtakes physical design at 90 nm

Source: IBS, November 2002, Coware Presentation
The Future

- We cannot afford to repeat the mistakes of the behavioral synthesis era

- Lack of cooperation and infrastructure

- Behavioral synthesis:
  - No common parser, no common datastructures, no common benchmarks, no common metrics for comparing algorithms
  - No overwhelming success
The Future

- **Co-Design is a much, much bigger problem**
- **Need to establish a robust Co-Design INFRASTRUCTURE for accelerating research**
  - Common language(s) and data models
  - Common platforms
  - Common evaluation metrics
  - Not flashy research, but essential!

- **Need to think beyond languages and levels of abstraction**
  - Higher-levels of abstraction are not enough to compensate for the much faster growing system complexity
  - Need new model of abstraction