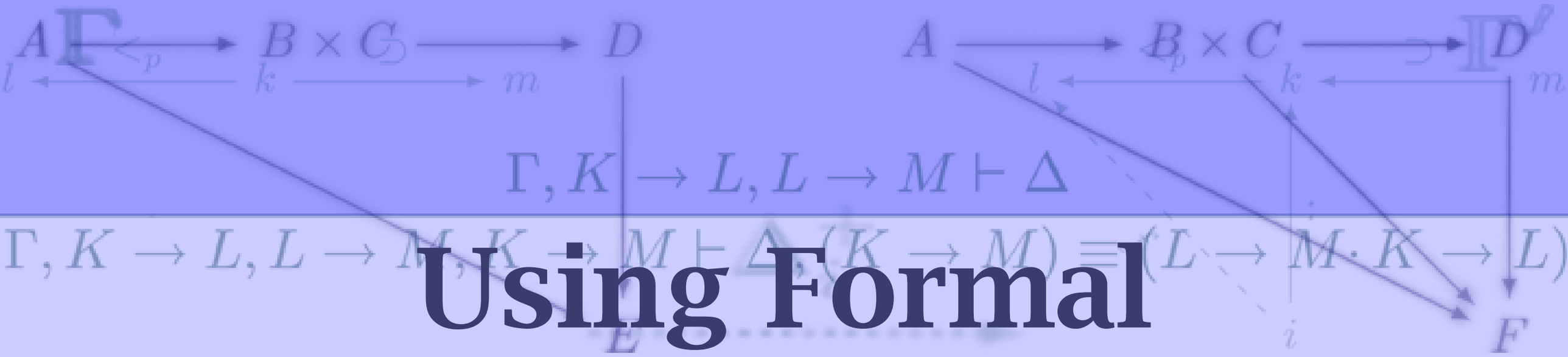


Using Formal Methods for VLSI Development

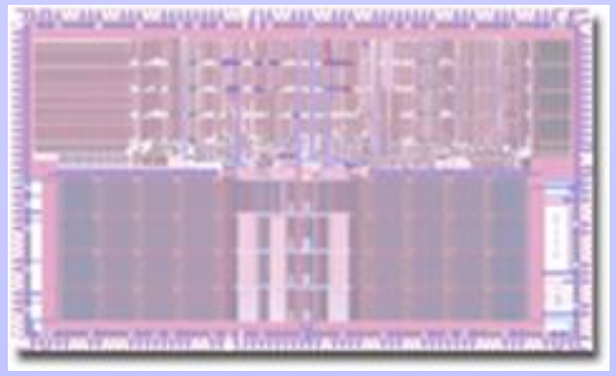
Using Java and JML in the “Real World”

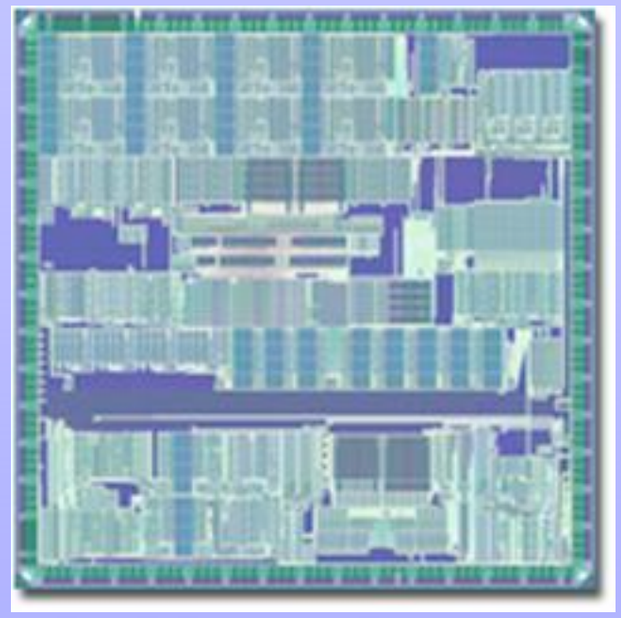
Joseph Kiniry

Department of
Computer Science
University College Dublin

Asynchronous VLSI

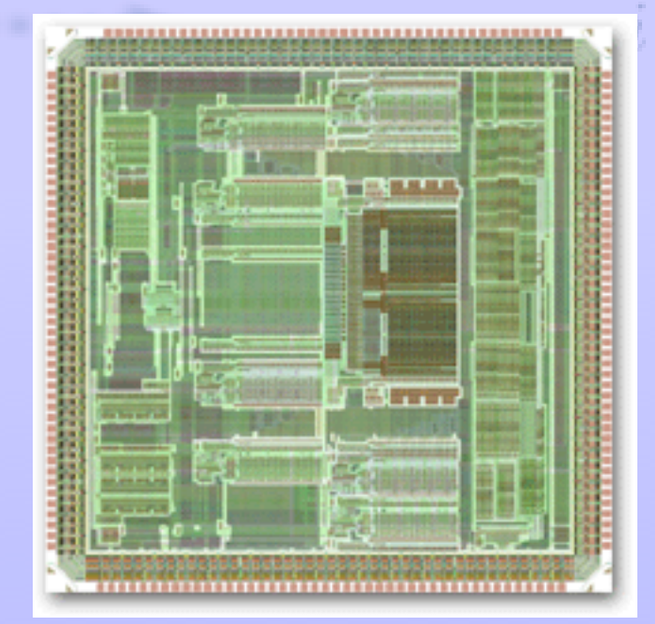
- What is AVLSI?
  - delay insensitive circuits
    - power invariant
    - design scalability
    - process invariant
- Manchester vs. Caltech

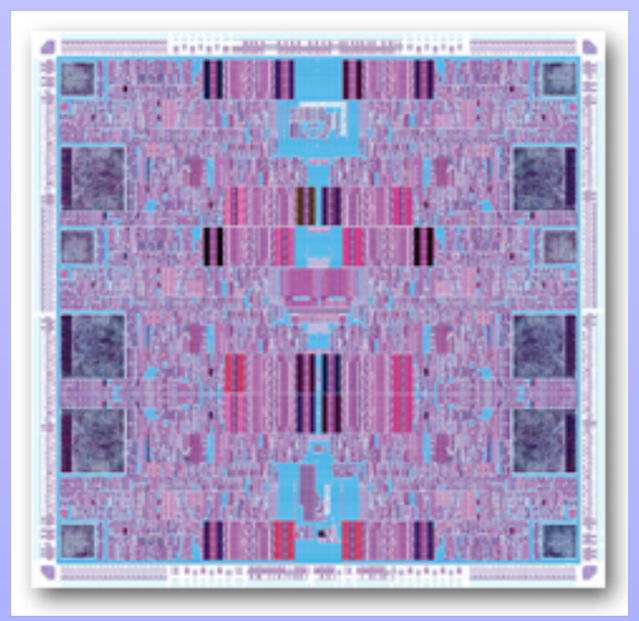




Typical VLSI Process

- high level specification (e.g., VHDL)
- low-level specification (e.g., Verilog)
- automated layout
- 99% commercial tools





An AVLSI Design Process

- multiple specification levels
  - multiple Java realizations
  - CSP (Concurrent Sequential Processes)
    - production rules
    - Verilog
    - automated and manual layout
- cosimulation for behavioral equivalence
  - testing for checking formal refinement

Unit Testing and Cosimulation

- must test at multiple granularities
  - cell, unit, CPU
- test with and without an operating system
  - minimal test OS and Linux
- test at all refinement levels
  - a test written in Java does not necessarily conform to any test written for CSP

Challenges

- performance
  - Your try simulating a processor in Java!
- scalability
  - massive memory and thread use
- robustness
  - if simulation takes five days and your simulator crashes after four…
- correctness!
  - you cannot patch a fabricated chip

Observations on Arrival

- major misuse of concurrency
- data structure abuse
- aimless optimization
- untracked requirements changes
- no documentation process

Recommendations and Response

- refine the software engineering process
  - particularly wrt docs and specs
- use commercial tools where appropriate
  - analysis with JProbe and jProfiler
  - revision control with Perforce
  - simulation with Cadence

Recommendations and Response (2)

- Open Source tools where appropriate
  - custom code coverage with Gretel
  - metrics with JavaNCSS and SlocCount
  - documentation with SGML and LaTeX
  - specification with JML
  - build system with Ant

Convincing the Boss and Coworkers

- lead by example
- gather hard data and present it well
- use the “soft sell”
  - suggest solutions and solve other people’s problems in intriguing ways
- convince key personnel
  - key developers, managers, executives, and board members

Problems and Resistance

- speed and memory issues *jmlc* and *jmlrac*
  - non-linear system compilation impact
- configurability of compilation and testing
  - unit of configuration is the class
  - would prefer Eiffel approach with configurability by assertion type
- lack of support from above
  - long-term prospects for use low

Positive Results

- performance
  - typical: 10 minute change for 10%
  - atypical: 1 man-month for 1000%
- memory use
  - garbage collection abuse
  - iterators, events, and string buffers
  - operating system VM abuse
  - overall memory size

Positive Results (2)

- configuration management
  - plain text configuration files
    - properties, bundles, and custom
- system monitoring
  - domain-specific run-time monitoring framework
- process changes
  - trading JML for English docs