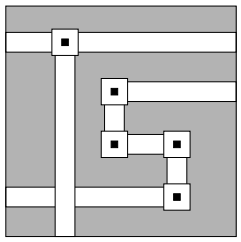


TRILOBYTE
SYSTEMS

Levels of Abstraction: The History of Custom MOS Design

Steve Golson
Trilobyte Systems

Phone: +1.978.369.9669
Email: sgolson@trilobyte.com
Web: http://www.trilobyte.com

1968 -- The beginning

- Hand-drawn schematics
- Standard cells -- new library for each product line
  - ~6 kinds of gates, ~4-5 speed ranges
- Gate sizing by hand using CT curves (load vs delay)
- Place & route by hand on mylar
- TTL breadboard for verification
- Transistor-level timing analysis using SPICE-like programs

Designer thinks about:
Gates, transistors, breadboard, and maybe library

Early 1970s -- some improvements

+ Module-level design (multiple designers working on one chip)
+ Software simulation (cycle-based)
+ On-chip bus design (trade wires for time)
+ Pitch-matched layout (datapath, PLA, ROM, RAM)
+ MOS-specific circuit techniques (dynamic logic)

Designer thinks about:
Gates, netlist simulation, floorplan, transistors

Early 1980s -- LSI Logic et al.

- Hand-drawn schematics
- Hand-typed netlist
+ Automatic place and route
- Netlist simulation
+ Gate-level timing analysis

Designer thinks about:
Gates, netlist simulation

Mid 1980s -- some improvements

+ Schematic capture
+ Automatic netlist generation
- Automatic place and route
- Netlist simulation
- Gate-level timing analysis

Designer thinks about:
Gates, netlist simulation

Early 1990s -- Verilog and synthesis

+ RTL Verilog
+ RTL simulation
+ Synthesized netlist
- Netlist simulation
- Automatic (?) place and route
- Gate-level timing analysis

Designer thinks about:
RTL code, RTL simulation, gates, netlist simulation

Late 1990s -- Behavioral synthesis

+ Behavioral Verilog, some generated by graphical HLDA tools
+ Behavioral Verilog and HLDA simulation
- RTL Verilog, some automatically generated by HLDA tool
- RTL simulation
- Synthesized netlist
- Netlist simulation
- Automatic (???) place and route
- Gate-level timing analysis
+ FPGA breadboard

Late 1990s -- Behavioral synthesis

Designer thinks about:
Behavioral code, RTL code,
gates, floorplan,
behavioral simulation, RTL simulation, netlist simulation, breadboard

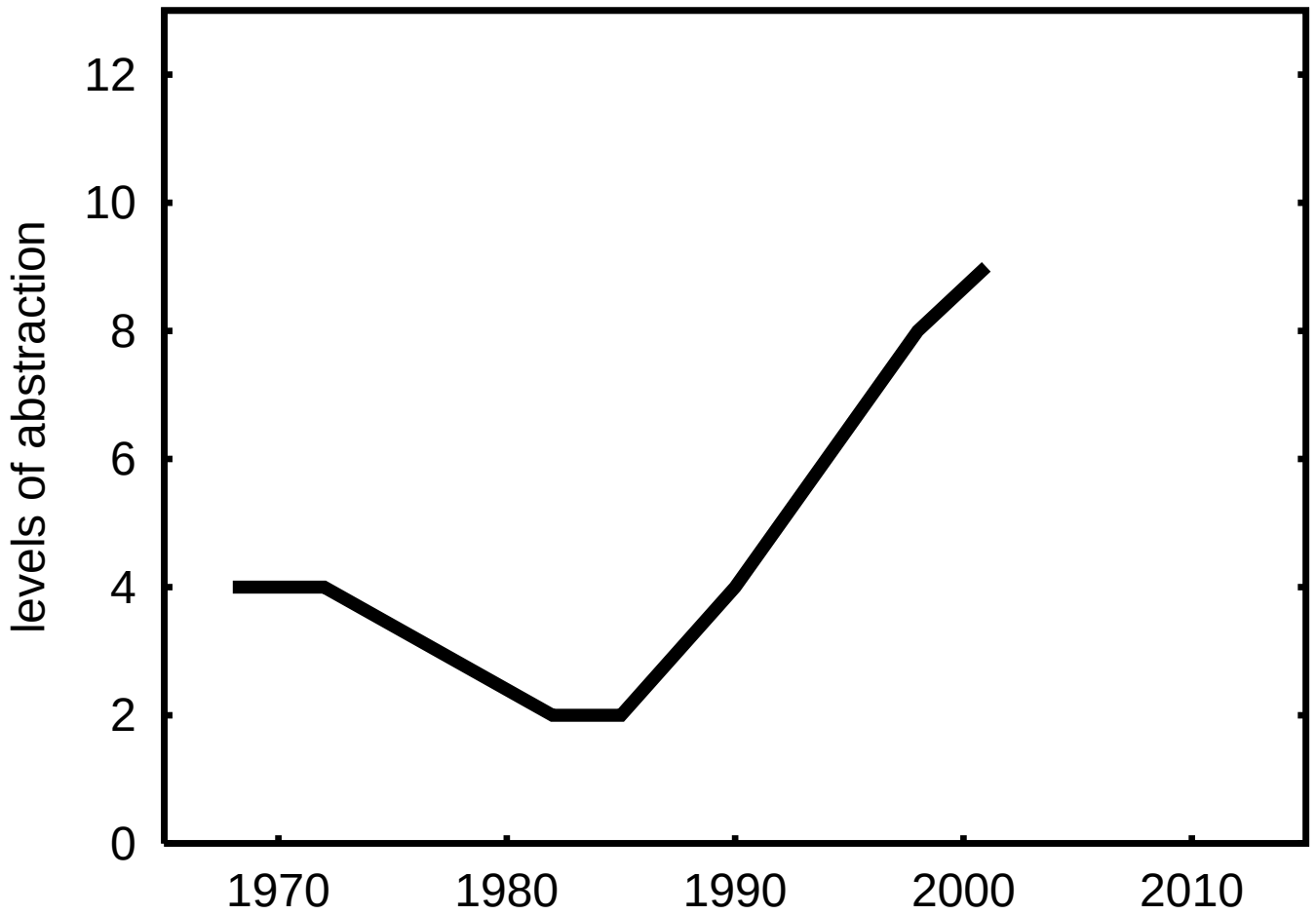
2001 and beyond the infinite…

+ System design language (C++, Superlog, SystemC, etc.)
+ System design language simulation
+ System design language synthesis
- RTL Verilog
- RTL simulation
- Floorplan
- RTL synthesized netlist & placement (physical synthesis)
- Netlist simulation
- Gate-level timing analysis

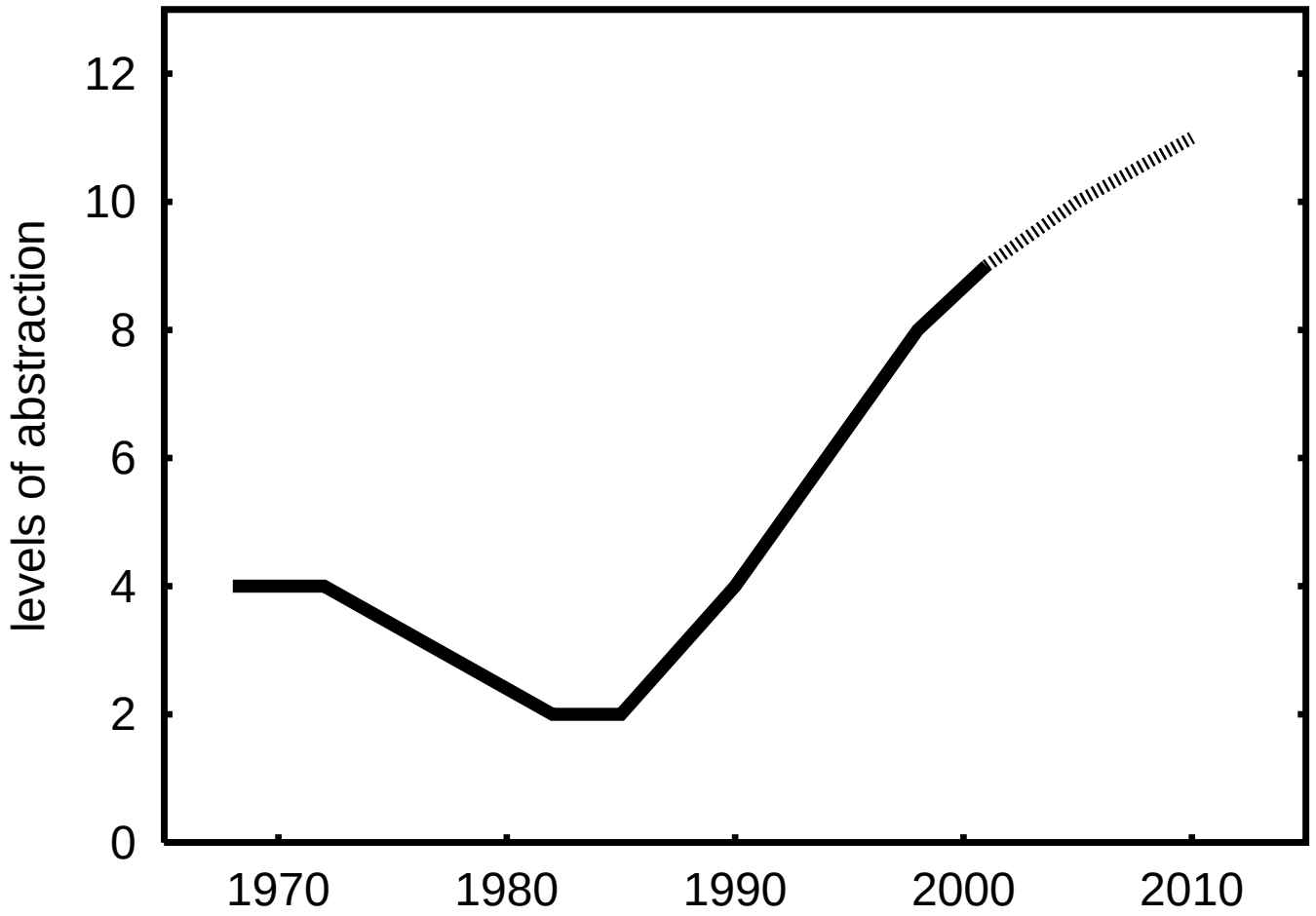
2001 and beyond the infinite…

Designer thinks about:
System code, RTL code,
gates, floorplan, placement,
system simulation, RTL simulation, netlist simulation, breadboard

Levels of abstraction today…



Levels of abstraction tomorrow??



What designers need…

What designers need

is to have abstraction levels *removed*,

not *added*

What designers need… is RTL signoff

Designer thinks about:
System code, RTL code,
System simulation, RTL simulation

Vendor thinks about:
everything else

Levels of abstraction: RTL signoff

What designers need…

- Physical synthesis
  - + Improve quality of synthesis
  - + Improve quality of place & route
  - + No more gate tweaks and layout mods
  - + No longer worry about gate-level timing
- Automatic verification of RTL vs netlist
  - + No longer worry about netlist simulation
- RTL analysis tools for timing, power, area
- Smarter IP lawyers

Things to consider about new EDA tools

- How do they fit into my existing design flow?
- Revolutionary or evolutionary?
- What problems do they solve?
- What new problems might they introduce?

Higher-level abstractions place you further away from an intuitive feel for hardware and timing: “How fast is it?”

Just because you can do something, doesn’t mean it is worth doing