Overview

- Why VLSI?
- Moore's Law.
- The VLSI design process.

Why VLSI?

- Integration improves the design:
  - lower parasitics = higher speed;
  - lower power;
  - physically smaller.
- Integration reduces manufacturing cost: (almost) no manual assembly.

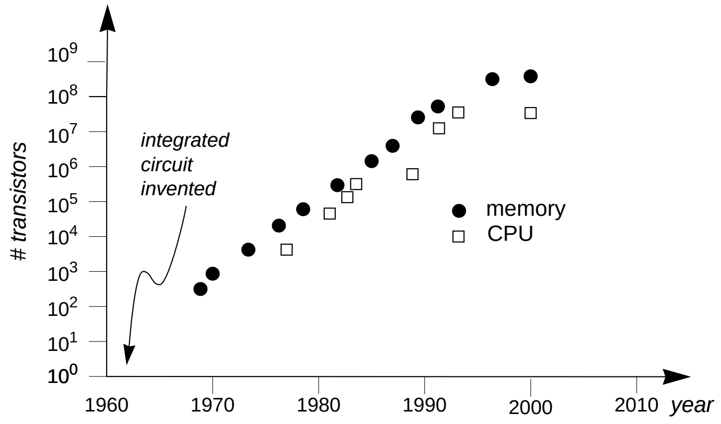
VLSI and you

- Microprocessors:
  - personal computers;
  - microcontrollers.
- DRAM/SRAM.
- Special-purpose processors.
- *System-on-a-chip*: all the above and analog circuitry on one die.

Moore's Law

- Gordon Moore: co-founder of Intel.
- Predicted that number of transistors per chip would grow exponentially (double every 18 months).

Moore’s Law plot



The VLSI design process

- May be part of larger product design.
- Major steps:
  - specification;
  - architecture;
  - logic design;
  - circuit design;
  - layout.

The steps

- Specification: function, cost, etc.
- Architecture: large blocks.
- Logic: gates + registers.
- Circuits: transistor sizes for speed, power.
- Layout: determines parasitics.
- Test vectors for testing.
- (Application software).

Challenges in VLSI design

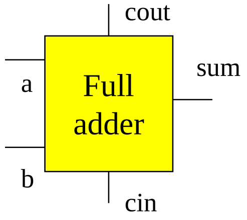
- Multiple levels of abstraction: transistors to CPUs.
- Multiple and conflicting constraints: low cost and high performance are often at odds.
- Short design time: Late products are often irrelevant.

Dealing with complexity

- Divide-and-conquer: limit the number of components you deal with at any one time.
- Group several components into larger components:
  - transistors form gates;
  - gates form functional units;
  - functional units form processing elements;
  - etc.

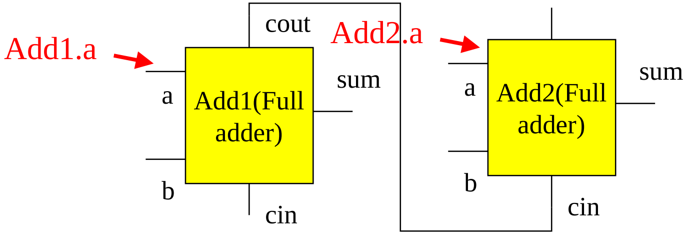
Hierarchical absraction

- Interior view of a component:
  - components and wires that make it up.
- Exterior view of a component = type:
  - body;
  - pins.



Instantiating component types

- Each instance has its own name:
  - add1 (type full adder)
  - add2 (type full adder).
- Each instance is a separate copy of the type:



Signal abstraction

- Analog and time-continuous currents and voltages become binary and time-discrete;
- *Bits* become *words*;
- Groups of words (e.g. address & data) become a *transaction* (e.g. read action on a bus).

Timing abstraction

- At *circuit* level: clock is just an ordinary signal;
- At *register-transfer* level (RTL) : a clock only connects to flipflops;
- At *algorithmic* level: no explicit clock, number of clock cycles not yet known.

Top-down vs. bottom-up design

- *Top-down* design adds functional detail.
  - Create lower levels of abstraction from upper levels.
- *Bottom-up* design creates abstractions from low-level behavior; it *reuses* subblocks with proven characteristics.
- Good design needs both top-down and bottom-up efforts.

Design validation

- Must check at every step that errors haven’t been introduced–the longer an error remains, the more expensive it becomes to remove it.
- Forward checking: compare results of less- and more-abstract stages.
- Back annotation: copy performance numbers to earlier stages.

Manufacturing test

- Not the same as design validation: just because the design is right doesn’t mean that every chip coming off the line will be right.
- Must quickly check whether manufacturing defects destroy function of chip.
- Must also speed-grade.

The cost of fabrication

- Current cost: about \$2-3 billion.
- Typical fab line occupies about 1 city block, employs a few hundred people.
- Most profitable period is first 18 months-2 years.

Cost factors in ICs

- Recurrent costs:
  - silicon area (about 0.07 \$/mm^2)
  - packaging (0.1 upto several dollars)
  - testing (about 0.05 \$/sec)
- Typical IC: 10 to 100 mm^2, 1 to 10 seconds of test time.
- Non-recurrent costs:
  - design time (about 100 \$/hour)
  - mask sets (in the order of 100 k\$ per set)
- It often happens that silicon is not “first time right” → new mask sets per redesign.