Computer Architecture

R. Poss
Computer Systems Architecture group (UvA)
e-mail: r.c.poss@uva.nl

Components and vocabulary

Von Neumann vs Harvard

- Data memory vs code memory: where are the instructions coming from?
- **Harvard architecture**: separate memory for code and data
- **Von Neumann architecture**: same memory for both
- Most architectures nowadays use a hybrid system:
  - Separate Instruction and Data cache (I-Cache / D-Cache)
  - Data in I-Cache not automatically updated when writing to D-cache, illustion of partially separate memories

Buffer vs memory

- **Buffer**: one or more memory cells arranged in a FIFO
  - Producer-consumer interface between two or more hardware components
- **Memory**: one or more memory cells arranged in an array
  - With address selection circuit to determine which cell is read from/written to in one access operation

Register vs main memory

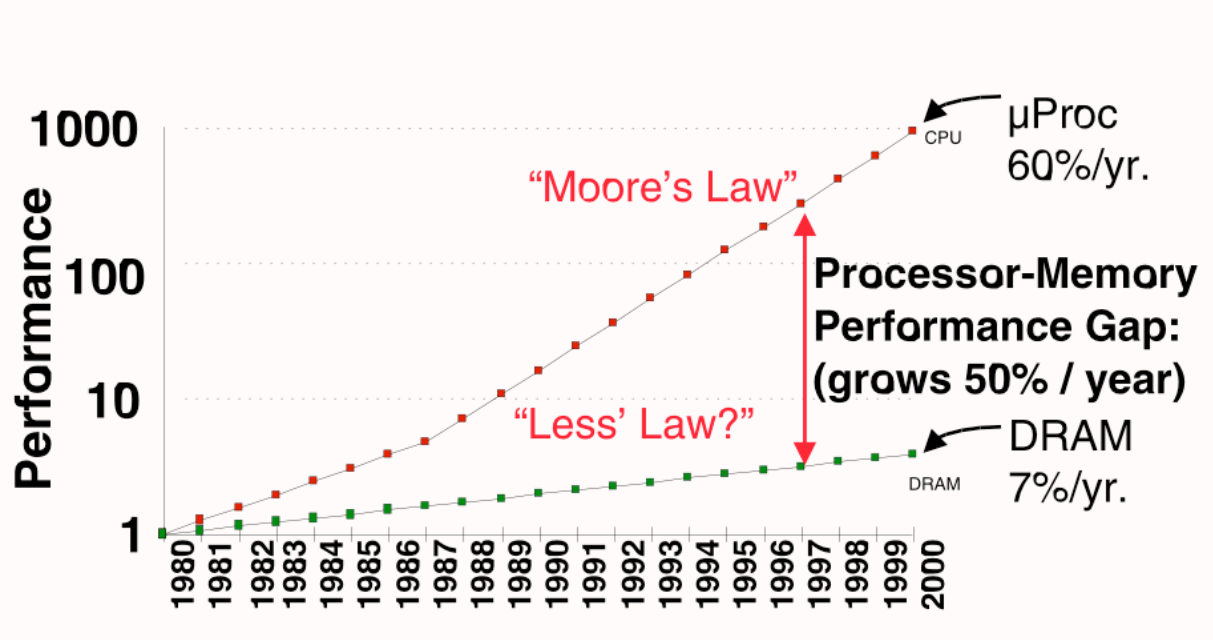
- **Register file**:
  - on-chip, close to the processor; SRAM: faster, larger/bit, small capacity
  - indexed using fixed offsets in instruction codes
- **Main memory** or scratchpads in MPSoCs
  - off-chip or farther from the processor; DRAM: slower, smaller/bit, large capacity
  - indexed using variable offsets in registers or other memory cells
- Both are forms of "memory" from the **architect**'s perspective
  the word "memory" designates any component with an address / value interface.
  - NB: from the software/compiler/OS perspective "memory" is not registers

Complexity vs Critical path

- **Complexity**:
  number of operations/circuits needed to produce an output from an input
- **Critical path**: longest chain of operations to the output from input
- **Complexity and critical path are different**
  - think parallelism again
- But: critical path establishes a lower bound on (circuit) complexity
  - In component design, criticial path sets lower bound on cycle time

Memory technology

Memory wall



- "Memory wall" = divergence between CPU and RAM speed
- We can increase bandwidth by introducing concurrency in memory access (e.g. through pipelining accesses)
  - but this requires regular access patterns
  - random accesses to main memory can cause severe performance degradation

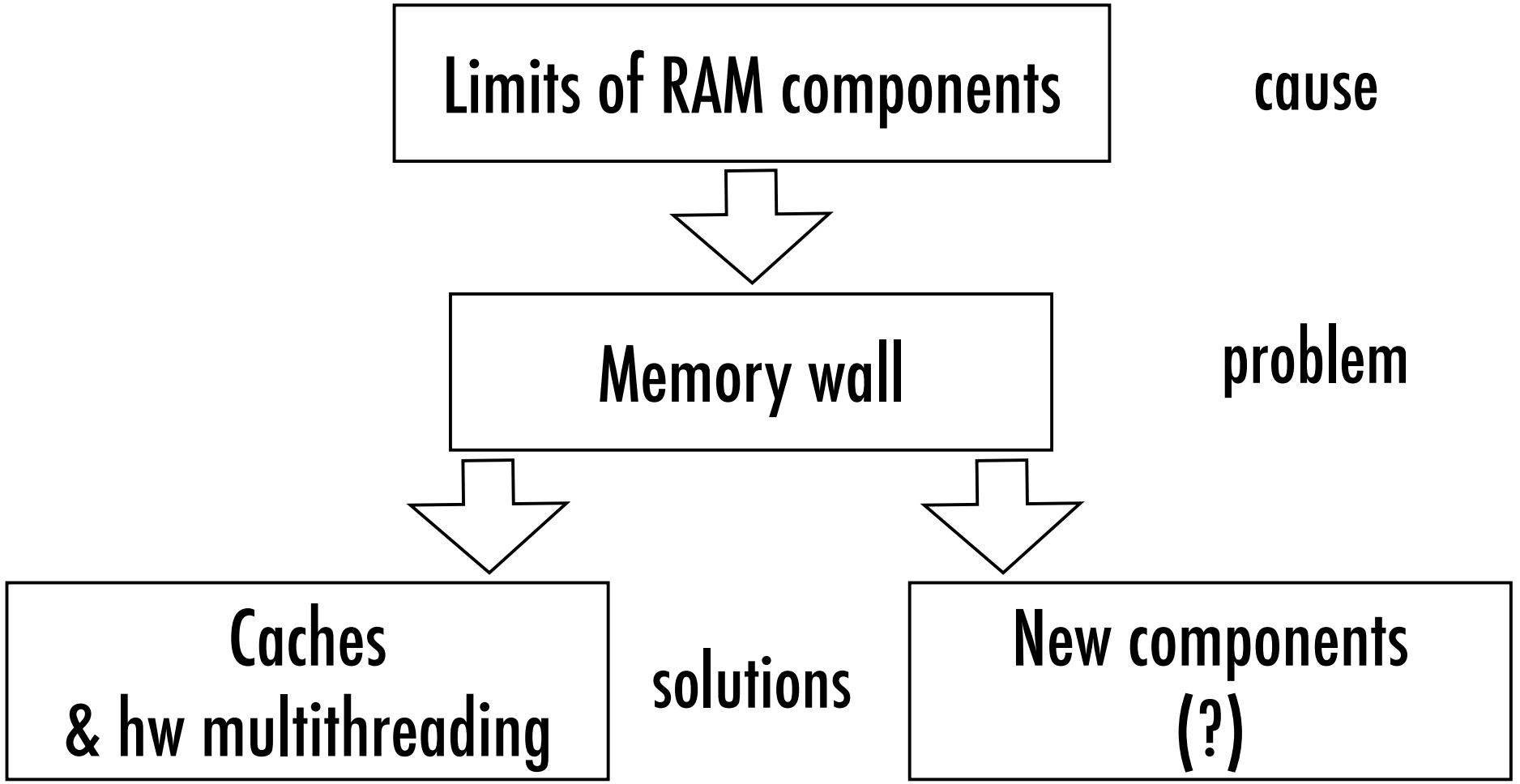
Memory hierarchy - issues

- Conflicting requirements in a memory system: we want both **large** and *fast*
- Electronic systems have higher latencies as they increase in size
  - **Speed of light is approximately 1ns for 30cms**
    - N.b. 1 ns is 3 clock cycles in a state of the art processor (3GHz)
  - **Pins, wires, connectors etc. all add resistance and capacitance which delay signals significantly**
- A memory hierarchy *attempts* to make a large slow memory appear fast by buffering data in smaller + faster memories close to the processor

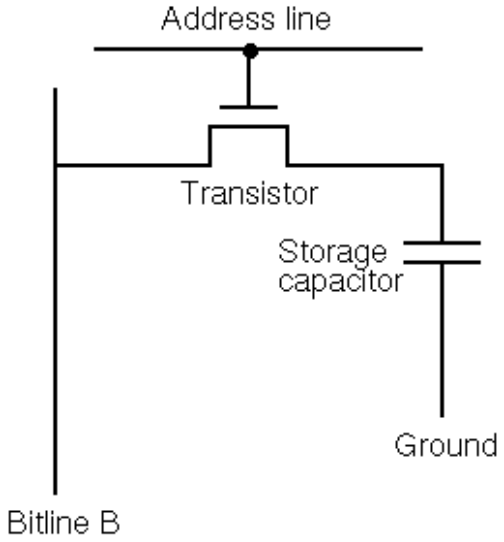
Memory hierarchy - issues

- Can make memories faster but this requires more power
- Need to drive long wires on chip across memory array – to do this fast needs more power
- Memory performance is a *compromise* between power and performance
- (As is processor performance today)

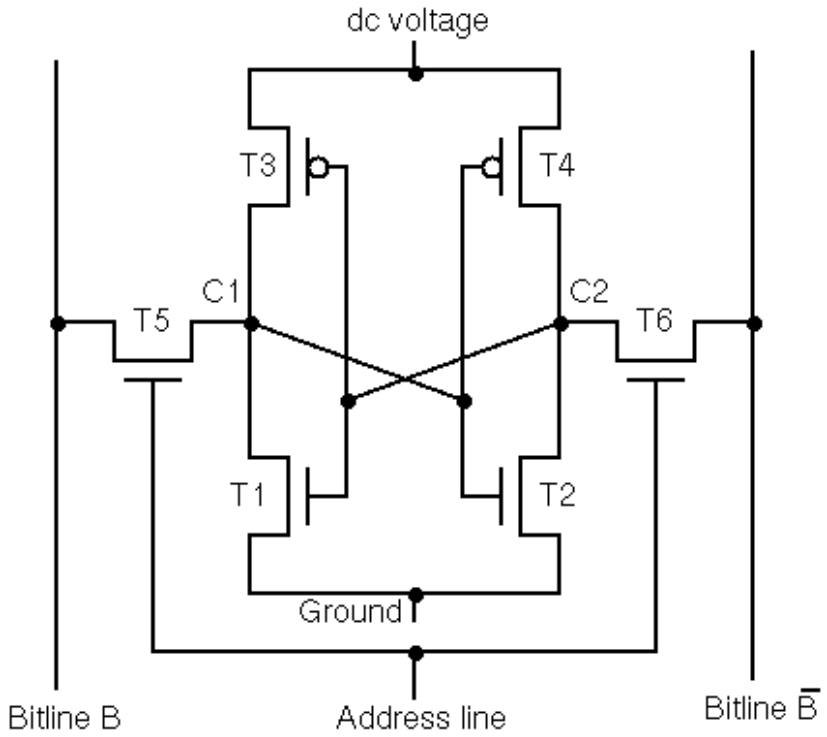
Summary


Limits of RAM components
cause
Memory wall
problem
Caches
& hw multithreading
solutions
New components
(?)

RAM cell designs



DRAM cell

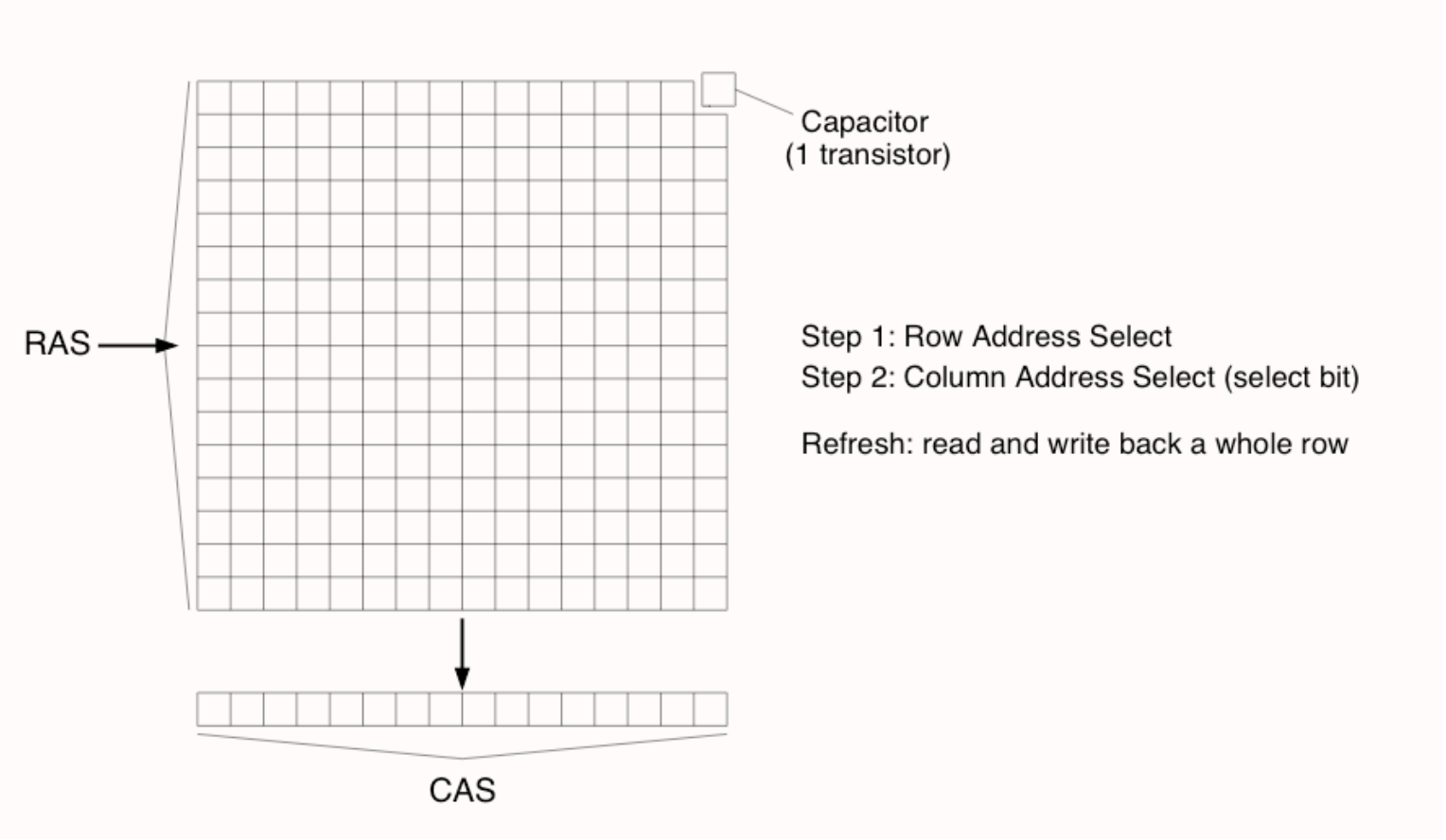


SRAM cell

Components - DRAM

- DRAM - is a very **dense** form of RAM - it is volatile
  - **it uses a capacitor to store data and one transistor to access it - access is destructive & data must be re-writtem**
  - **charge also leaks from the capacitor which stores the data so data must be refreshed periodically ("dynamic" RAM)**
- Typical DRAM chip characteristics
  - **256-1024 Mbit and 2-800MHz cycle**
- Uses two cycle Row/Column addressing (RAS/CAS)
  - **two-stage access and the requirement to rewrite data contribute to slow cycle time**
  - **but good design can help for regular accesses**

DRAM - RAS/CAS addressing



Components - SRAM

- SRAM - is less dense but faster than DRAM
  - **uses a four transistors to store data and a two further transistors to access it - access is non-destructive**
  - **data is stable while the RAM has power connected ("static" RAM)**
- Typical SRAM chip characteristics
  - **~64Mbit and ~1GMz cycle**
    - although SRAM cycle times are similar to DRAM, SRAM is true random access memory
      DRAM can only read consecutive bits at the cycle rate, therefore DRAM has a much larger latency time
  - SRAM is also used for memory on the processor chip
    - **registers** and **cache** both use SRAM technology
- **the smaller the memory the "faster" it operates: lower latency**

Bandwidth vs. Latency

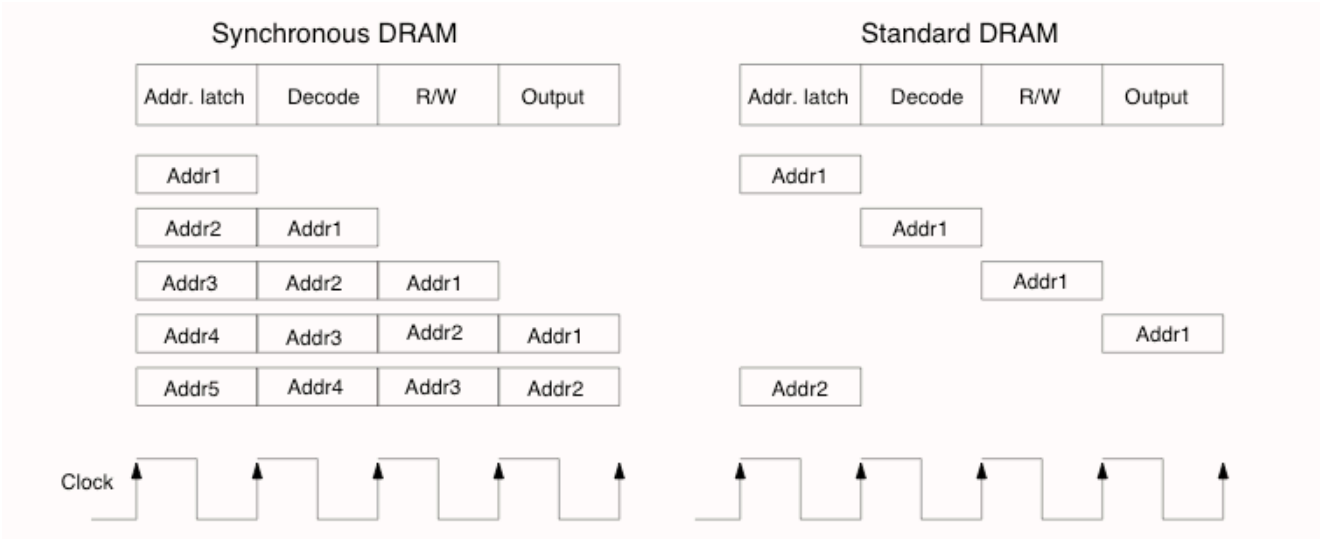
- **Memory latency** is the delay required to obtain a specific item of data (measured in seconds)
  - **This is larger in DRAM than in SRAM**
    - SRAM can access any bit each cycle
    - DRAM is restricted to bits in the same row, CAS cycles
- **Memory Bandwidth** is the rate at which data can be accessed (e.g. bits per second)
  - **Bandwidth unit is normally 1/cycle time (1 access at a time)**
  - **This rate can be improved by concurrent access**

Improving DRAM bandwidth

- Using locality to get maximum bandwidth
  - **One RAS multiple CAS e.g.**
    - Fast page mode DRAM
    - E(xtended) D(ata) O(utput) RAMs
  - **Burst-mode DRAMs**
    - Burst EDO RAM
    - S(ynchronous)DRAM
- By improving the interface
  - **DDR SDRAM and RAMBUS**

Example - Synchronous DRAM

- SDRAM uses a form of pipelining
  - **DDR SDRAM** - double data rate uses transfers on both rising and falling clock edges



BUT...
this does
not decrease
the latency!

Summary

- Hope for new dense + fast technology, maybe **memristors**?
- Most common solution: **cache** data
  - *Requires locality of access, or memory reuse*
- Design processors that **tolerate high-latency memory accesses** – "don't wait do something else"
  - *Trend: hardware multithreading in current and future chips*