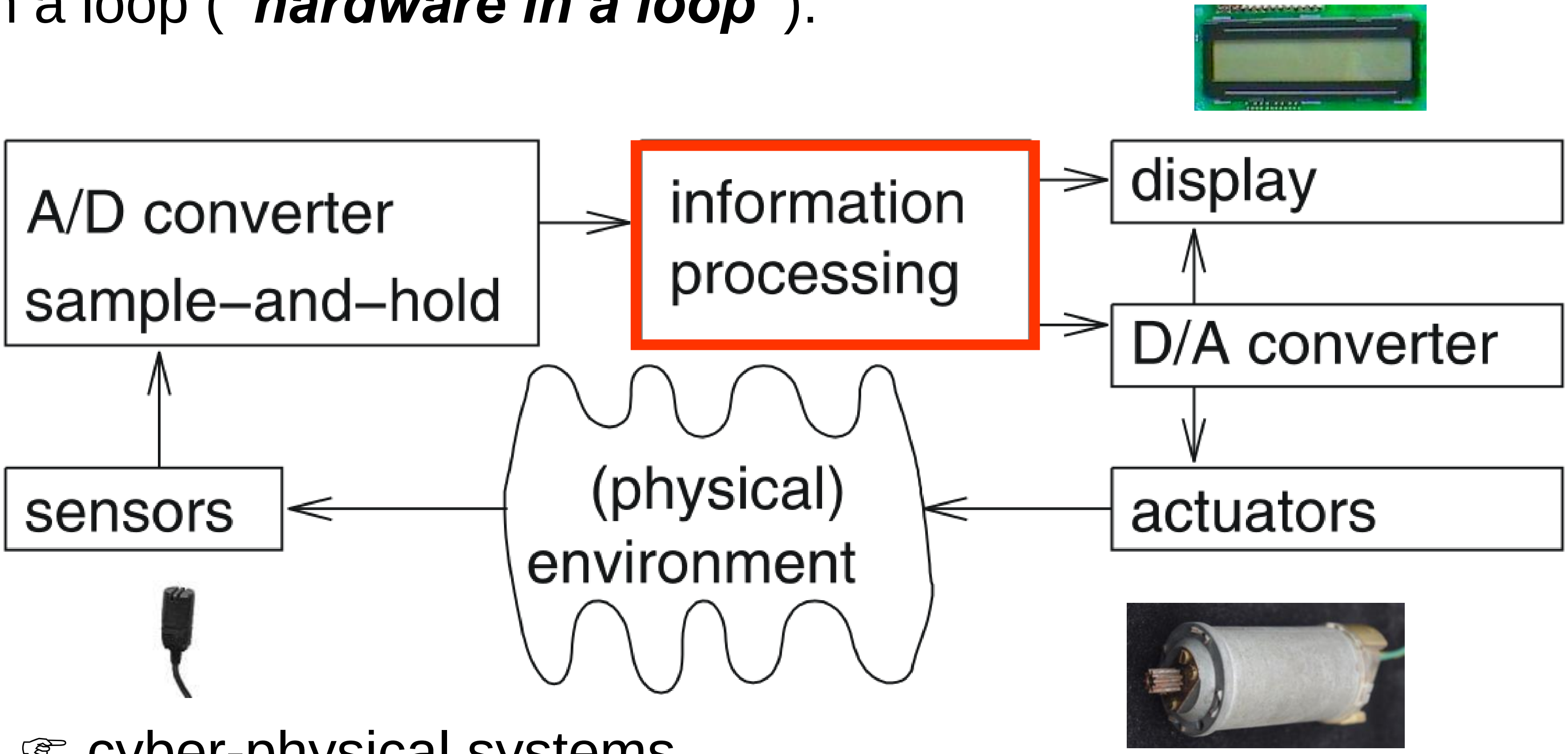
Embedded Systems 14

REVIEW: Embedded System Hardware

Embedded system hardware is frequently used in a loop (***“hardware in a loop“***):

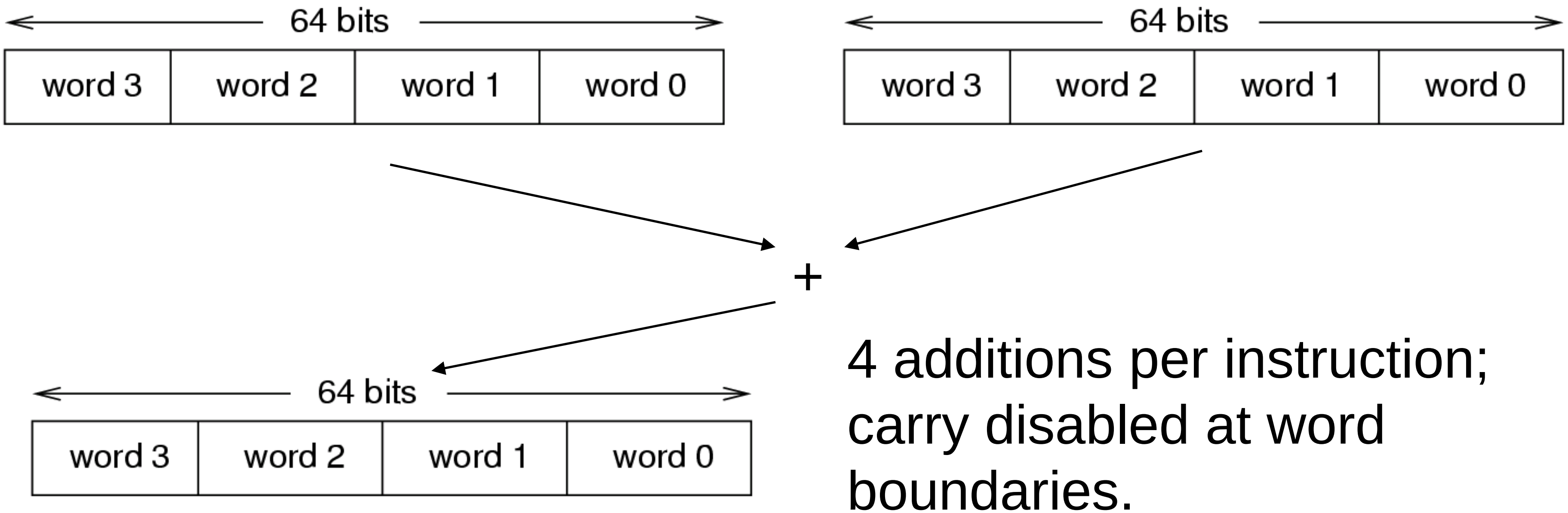


☞ cyber-physical systems

REVIEW: Single-instruction, multiple-data (SIMD)

- Multimedia instructions exploit that many registers, adders etc are quite wide (32/64 bit),
- whereas most multimedia data types are narrow (e.g. 8 bit per color, 16 bit per audio sample per channel)

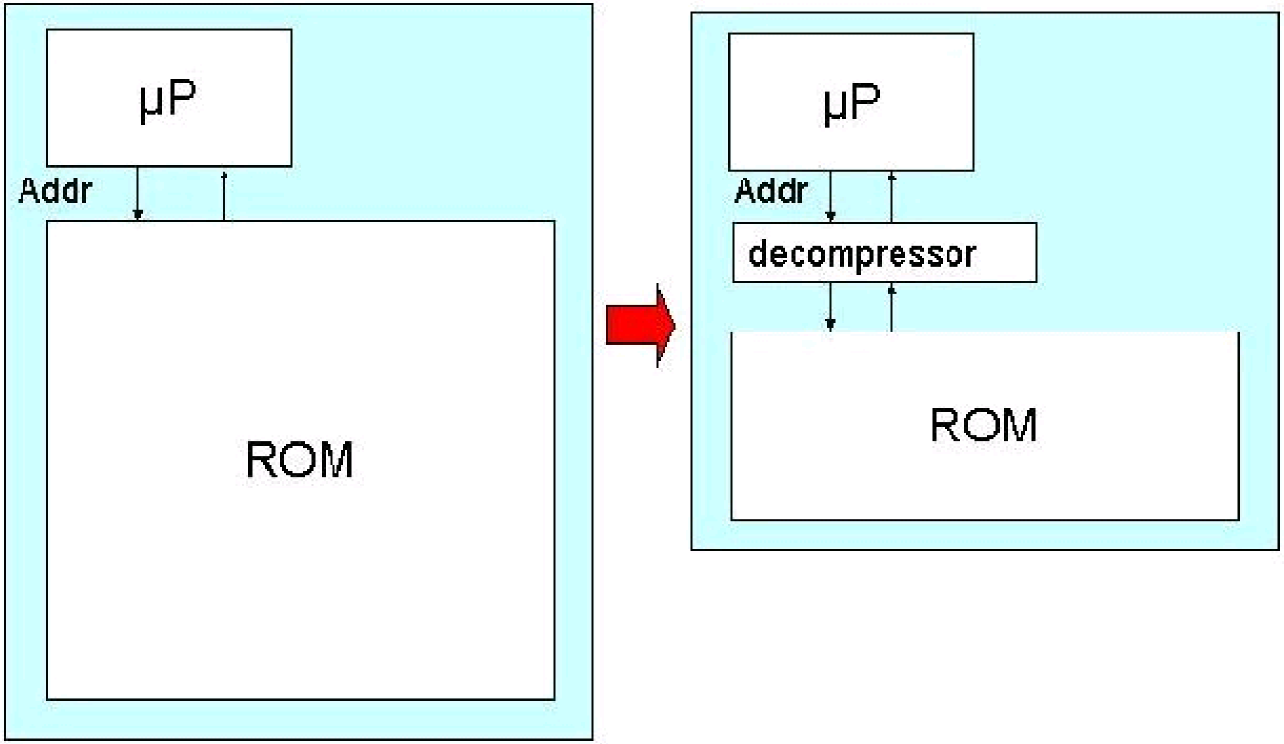
☞ 2-8 values can be stored per register and added. E.g.:



4 additions per instruction; carry disabled at word boundaries.

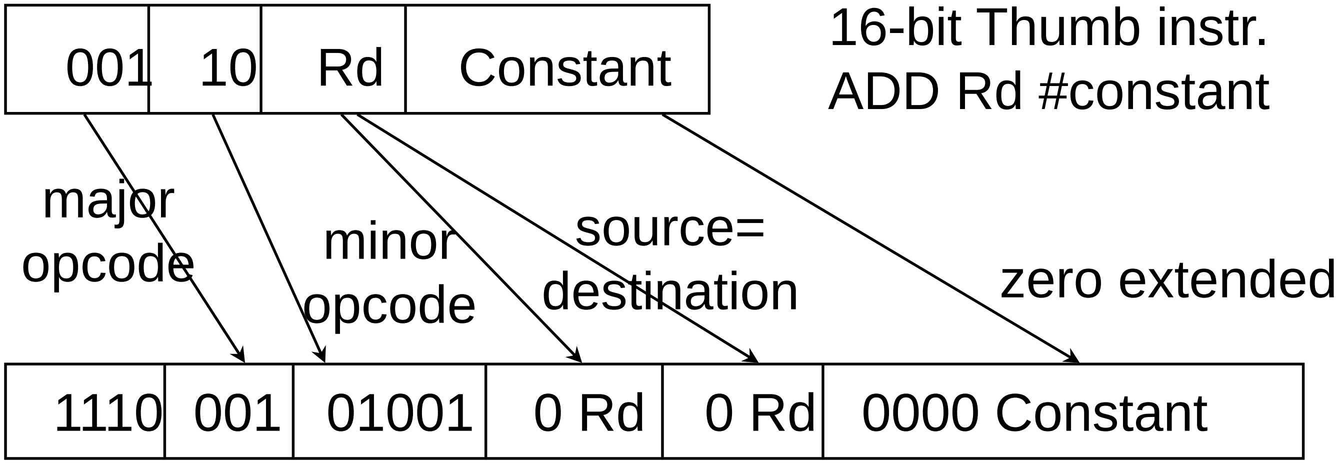
Code-size efficiency

- **CISC machines**: RISC machines designed for run-time-, not for code-size-efficiency
- **Compression techniques:** key idea



Code-size efficiency

- **Compression techniques (continued):**
  - 2nd instruction set, e.g. ARM Thumb instruction set:



16-bit Thumb instr.
ADD Rd #constant

Dynamically decoded at run-time

- Reduction to 65-70 % of original code size
- 130% of ARM performance with 8/16 bit memory
- 85% of ARM performance with 32-bit memory

[ARM, R. Gupta]

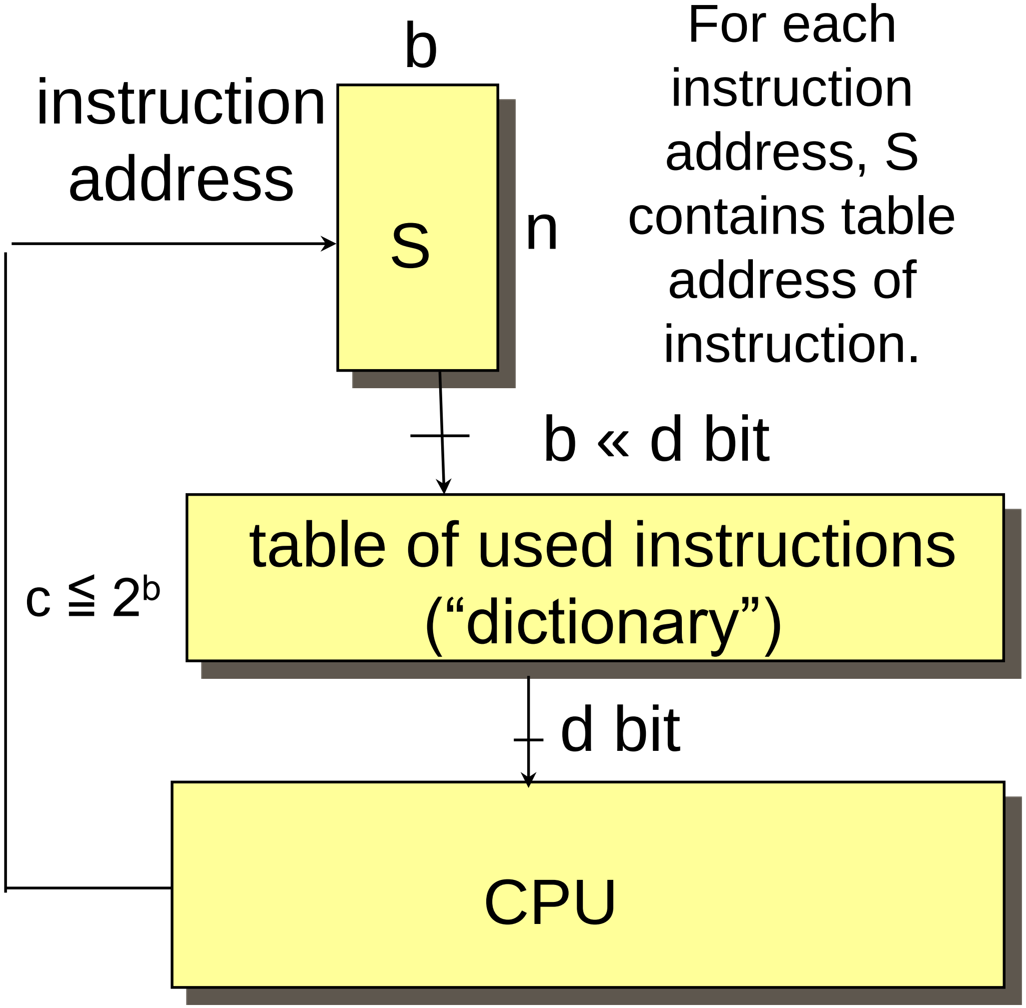
Same approach for LSI TinyRisc, …
Requires support by compiler, assembler etc.

Dictionary approach, two level control store (indirect addressing of instructions)

“*Dictionary-based coding schemes cover a wide range of various coders and compressors.*
Their common feature is that the methods use some kind of a dictionary that contains parts of the input sequence which frequently appear.
The encoded sequence in turn contains references to the dictionary elements rather than containing these over and over.”

[Á. Beszédes et al.: Survey of Code size Reduction Methods, Survey of Code-Size Reduction Methods, *ACM Computing Surveys*, Vol. 35, Sept. 2003, pp 223-267]

Key idea (for *d* bit instructions)



For each instruction address, S contains table address of instruction.

Uncompressed storage of *n d*-bit-wide instructions requires *nxd* bits.

In compressed code, each instruction pattern is stored only once.

small

Hopefully, *nxbcxd < nxd.*

Called nanoprogramming in the Motorola 68000.

Cache-based decompression

- Main idea: decompression whenever cache-lines are fetched from memory.
- Cache lines ↔ variable-sized blocks in memory
  ☞ line address tables (LATs) for translation of instruction addresses into memory addresses.
- Tables may become large and have to be bypassed by a line address translation buffer.

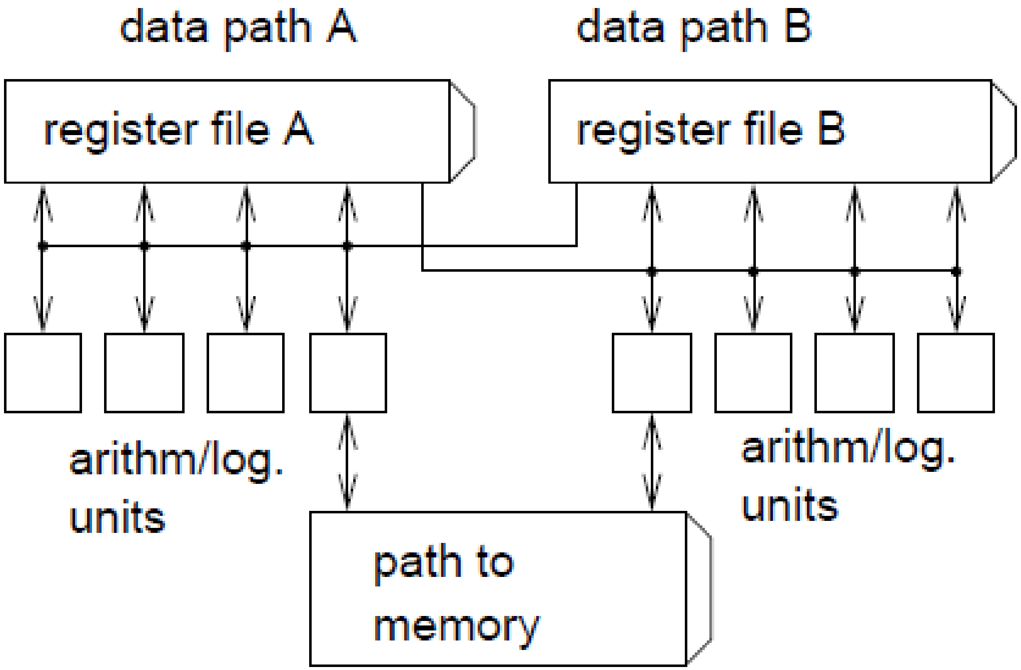
[A. Wolfe, A. Chanin, MICRO-92]

Partitioned register files

- Many memory ports are required to supply enough operands per cycle.
- Memories with many ports are expensive.

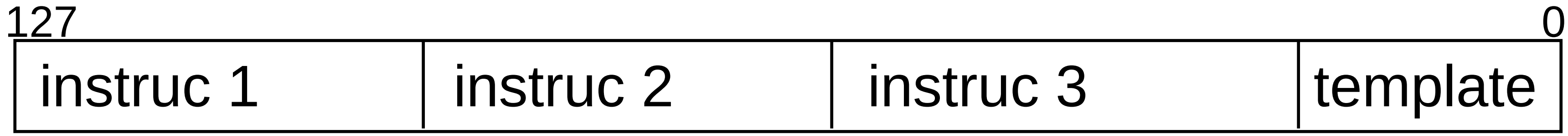
☞ Registers are partitioned into (typically 2) sets

e.g. for TM 320C6x



More encoding flexibility with IA-64 Itanium

3 instructions per **bundle:**



127

0

instruc 1

instruc 2

instruc 3

template

Instruction grouping information

There are 5 instruction types:

- A: common ALU instructions
- I: more special integer instructions (e.g. shifts)
- M: Memory instructions
- F: floating point instructions
- B: branches

The following combinations can be encoded in templates:

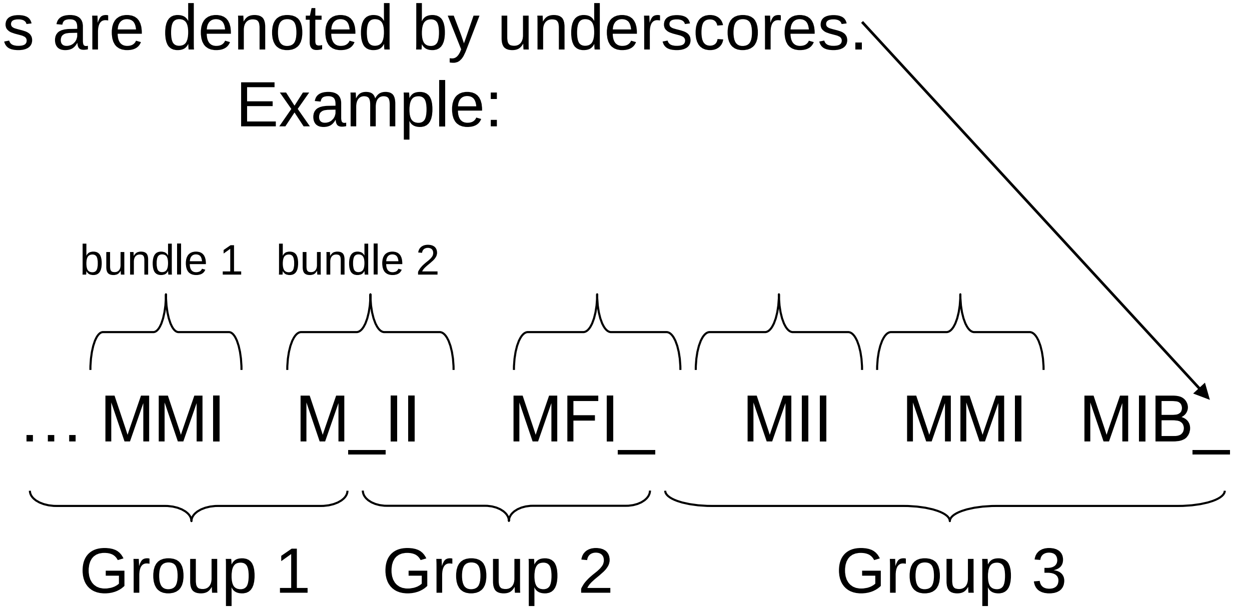
- MII, MMI, MFI, MIB, MMB, MFB, MMF, MBB, BBB, MLX with LX = *move 64-bit immediate* encoded in 2 slots

Templates and instruction types

End of parallel execution called **stops.**

Stops are denoted by underscores.

Example:



Very restricted placement of stops within bundle.

Parallel execution within groups possible.

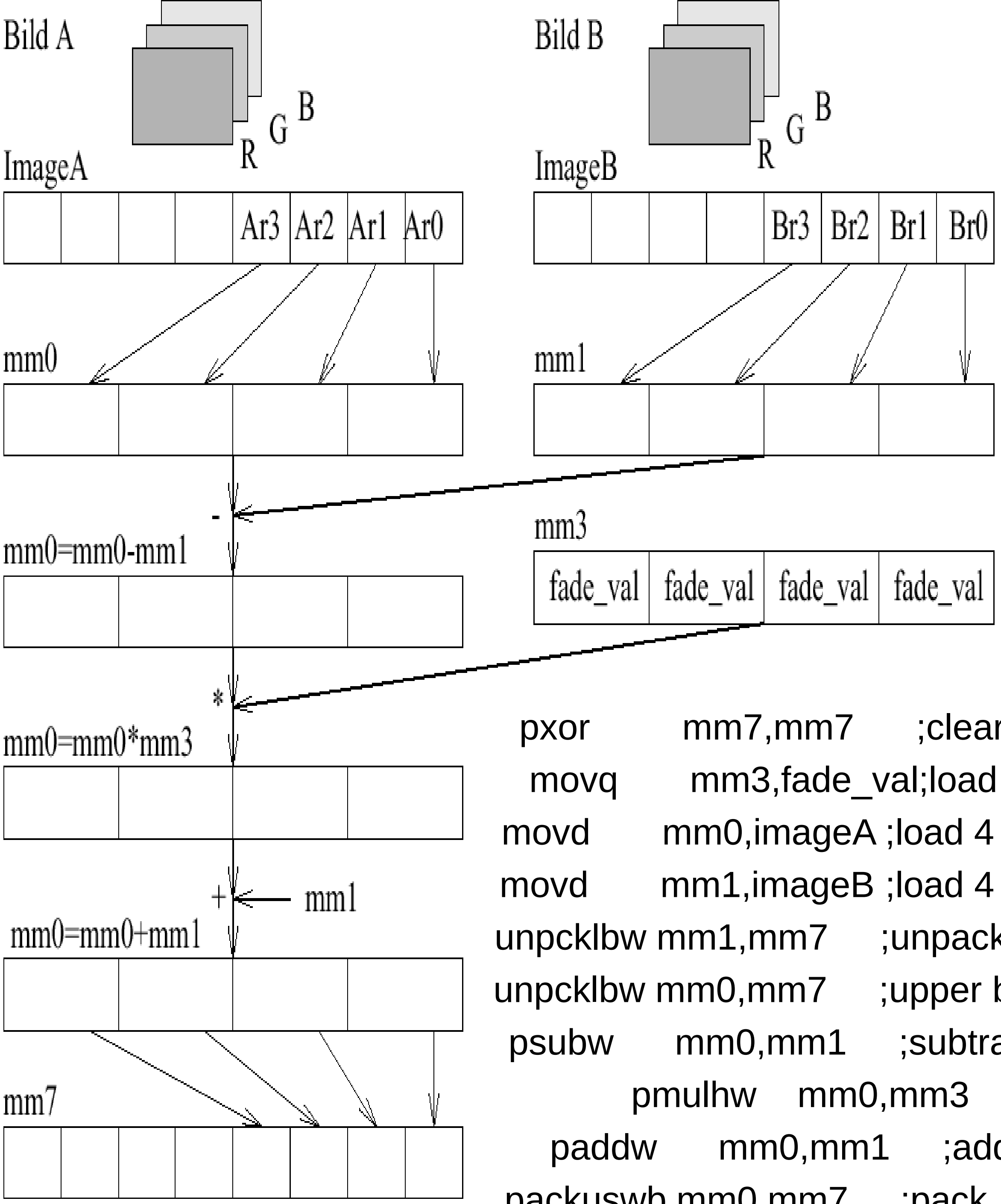
Parallel execution can span several bundles

Pentium MMX

Example: Scaled interpolation between two images

Next word = next pixel, same color.

4 pixels processed at a time.

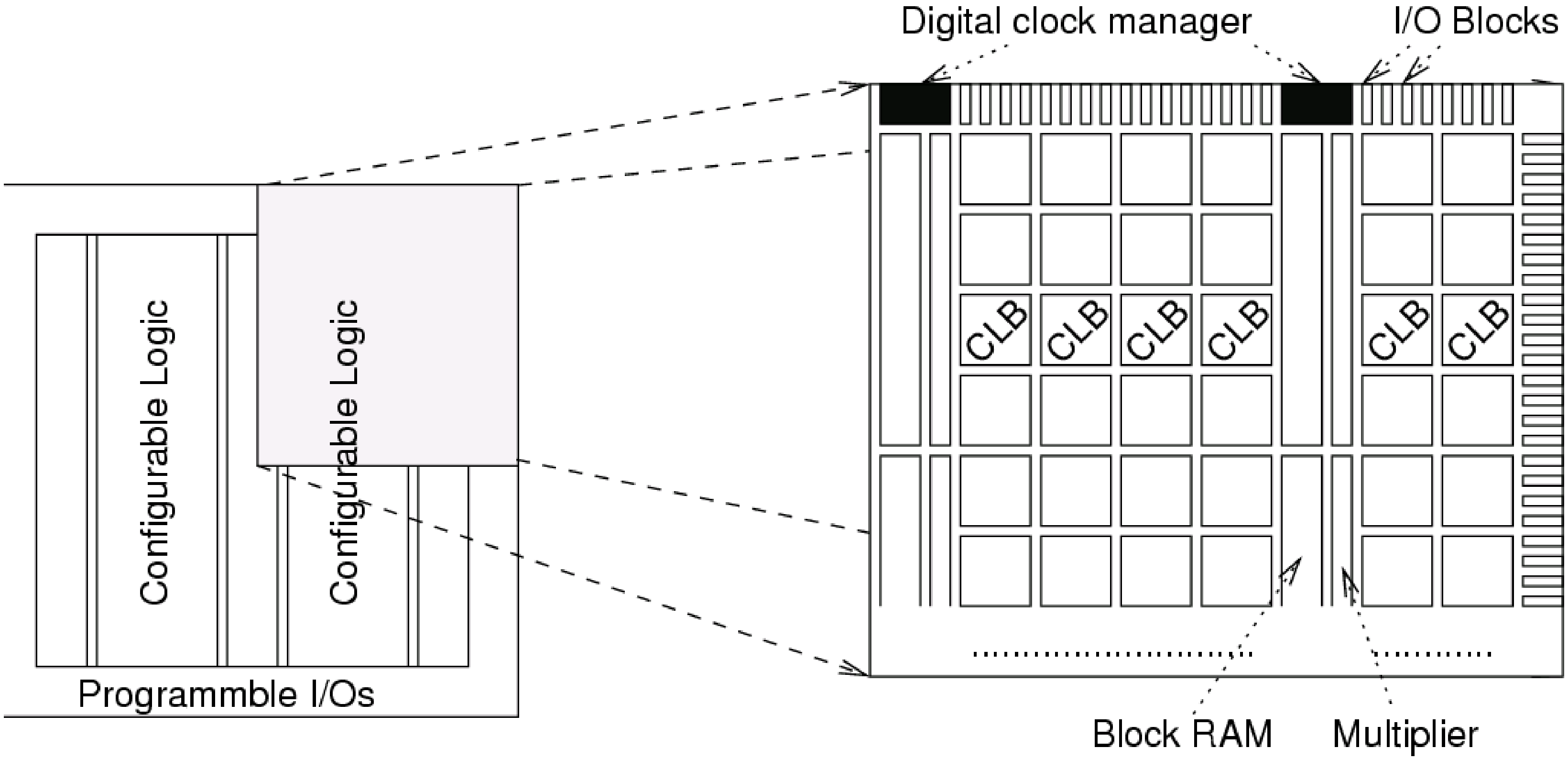


```
pxor        mm7,mm7      ;clear register mm7
movq        mm3,fade_val;load scaling value
movd        mm0,imageA ;load 4 red pixels for A
movd        mm1,imageB ;load 4 red pixels for B
unpcklbw mm1,mm7      ;unpack,bytes to words
unpcklbw mm0,mm7      ;upper bytes from mm7
psubw       mm0,mm1      ;subtract pixel values
pmulhw    mm0,mm3      ;scale
paddw       mm0,mm1      ;add to image B
packuswb mm0,mm7      ;pack, words to bytes
```

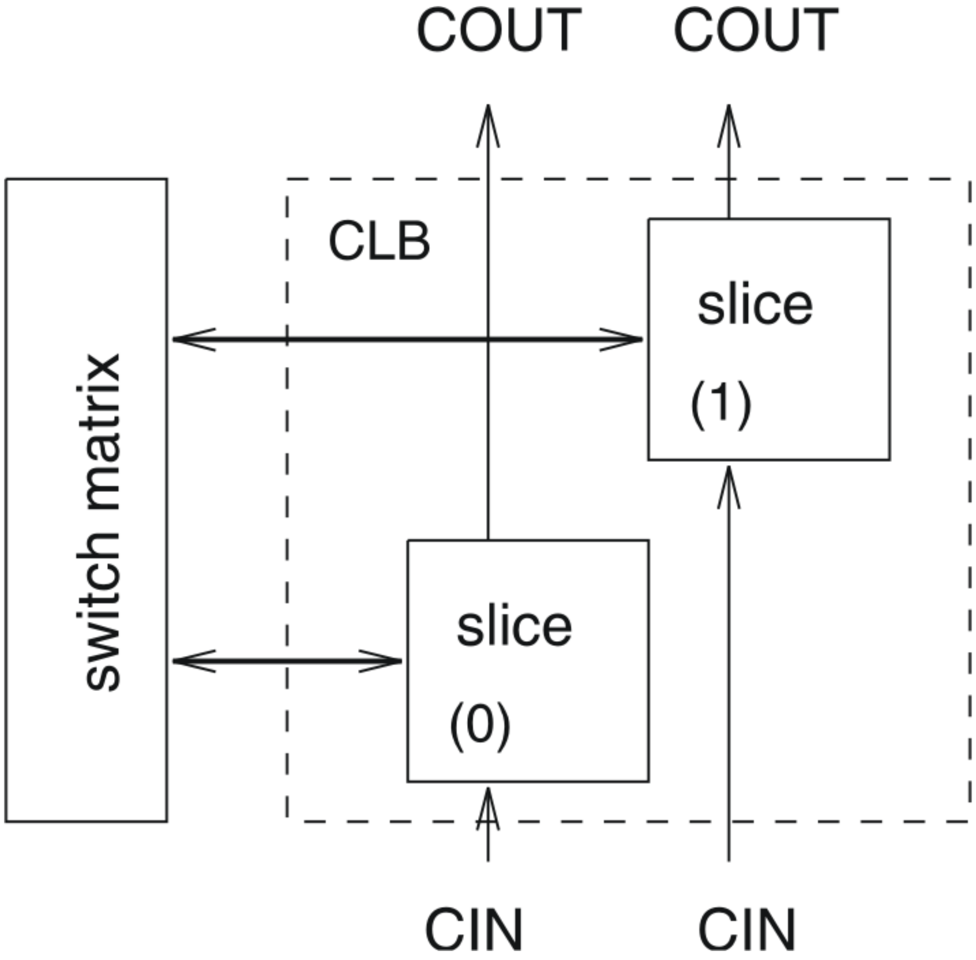
Reconfigurable Logic

- Full custom chips may be too expensive, software too slow.
- Combine the speed of HW with the flexibility of SW
  - ☞ HW with programmable functions and interconnect.
  - ☞ Use of configurable hardware;
    common form: field programmable gate arrays (FPGAs)
- ☞ Applications: bit-oriented algorithms like
  - encryption,
  - fast “object recognition“ (medical and military)
  - Adapting mobile phones to different standards.
- Very popular devices from
  - XILINX (XILINX Virtex 6 are recent devices)
  - Actel, Altera and others

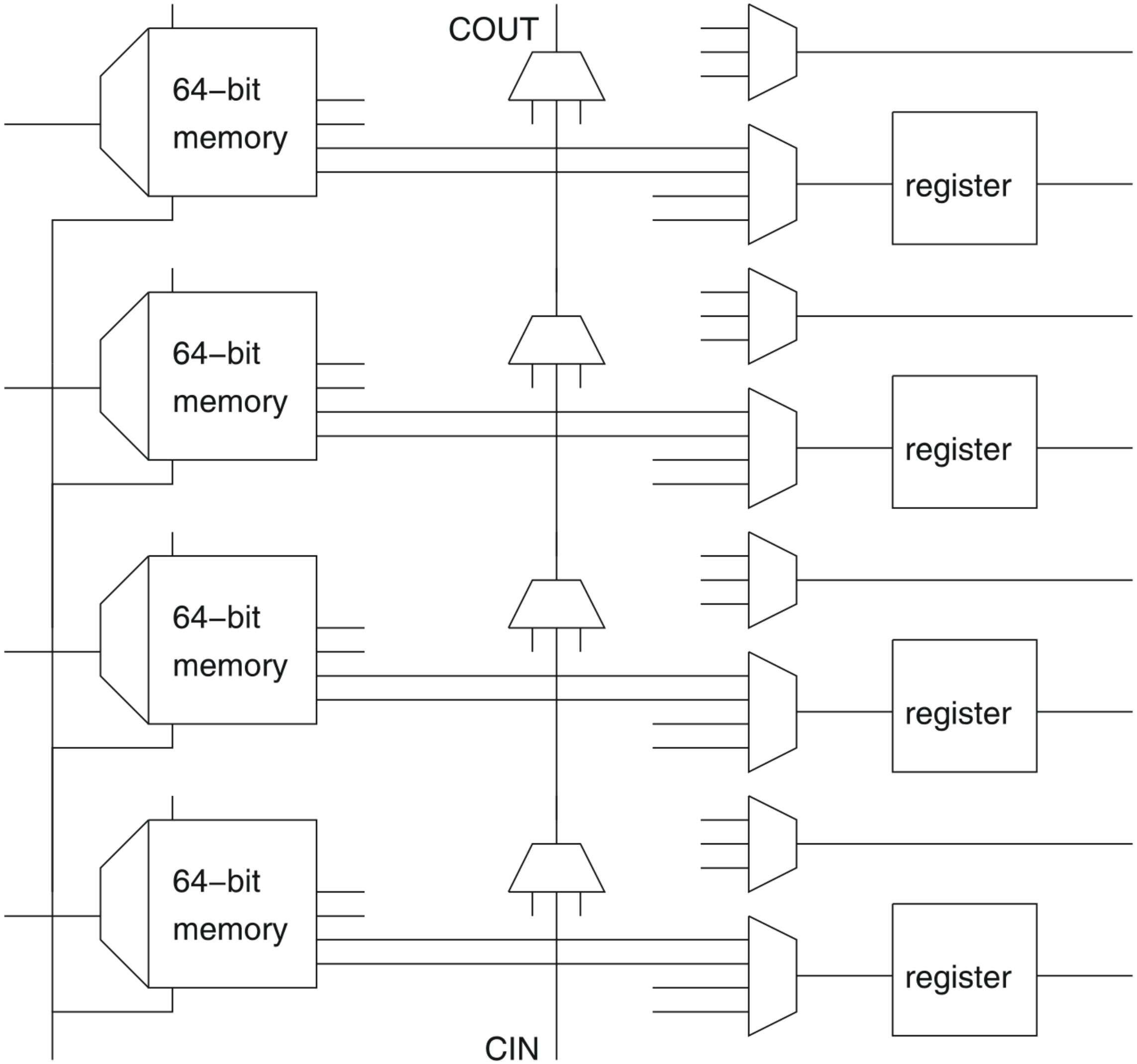
Floor-plan of VIRTEX II FPGAs



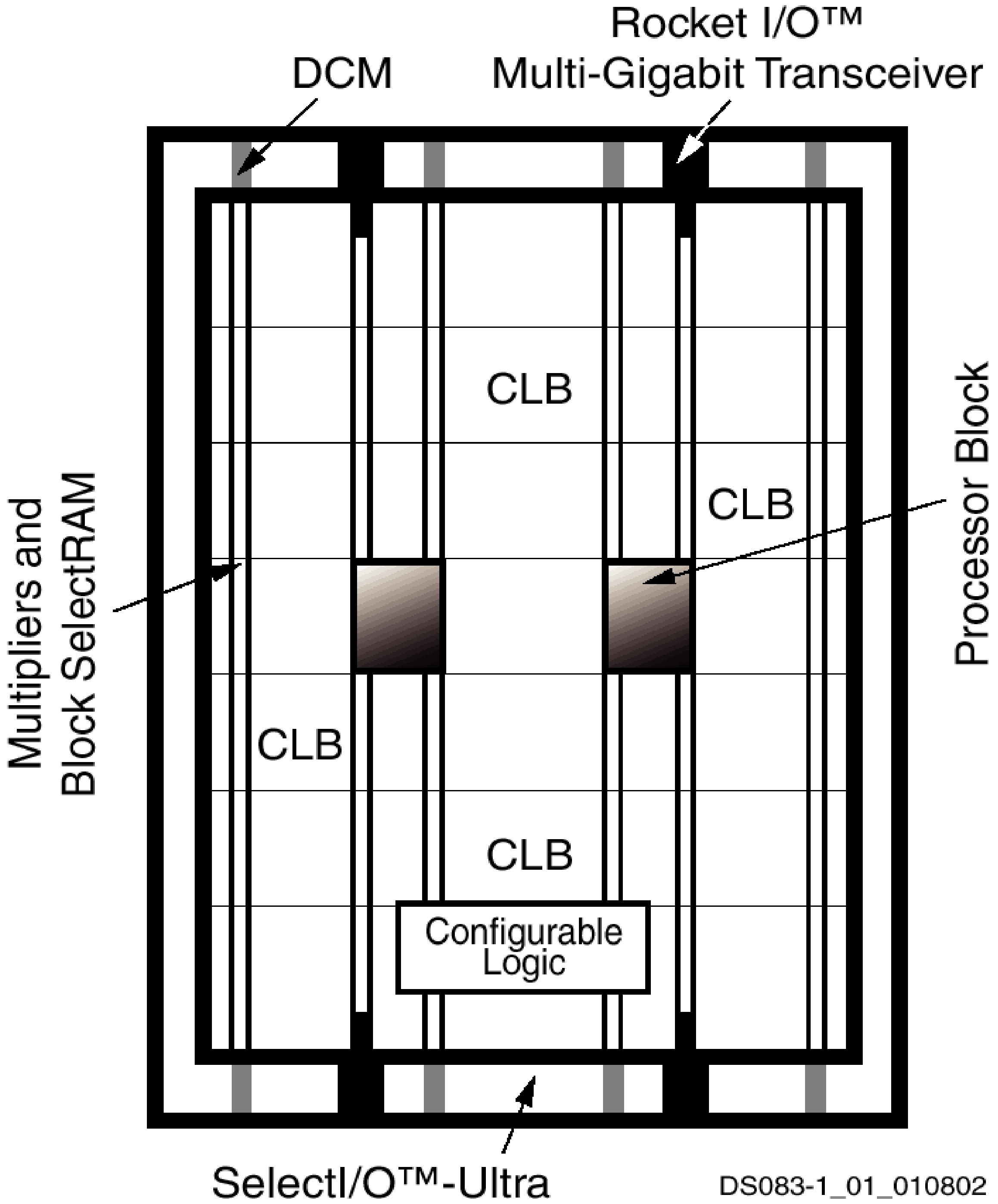
Virtex 5 Configurable Logic Block (CLB)



Virtex 5 Slice (simplified)



Memories typically used as look-up tables to implement any Boolean function of ≤ 6 variables.



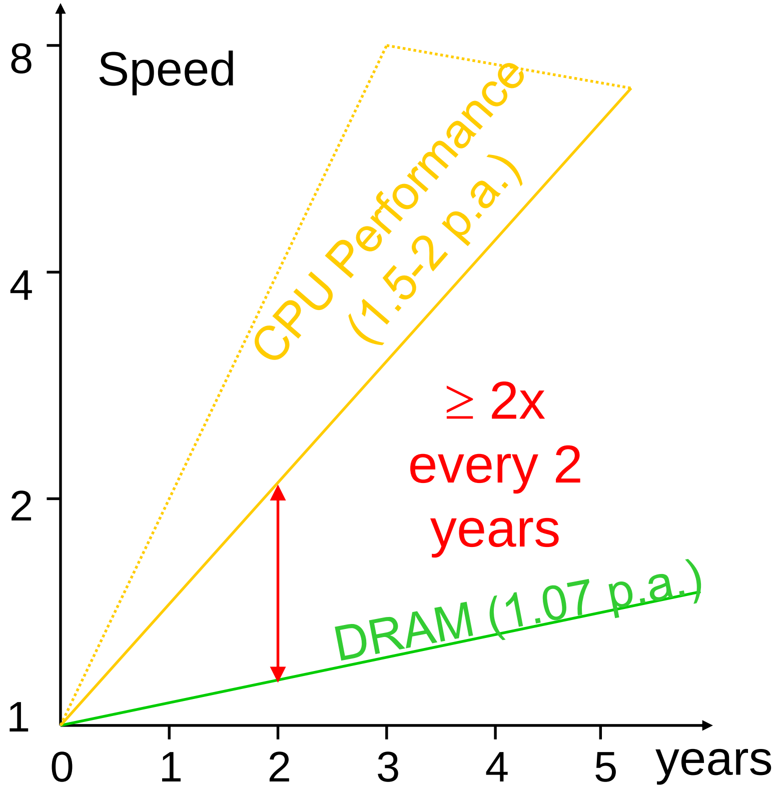
Virtex II Pro Devices include up to 4 PowerPC processor cores

Virtex 5 Devices include
up to 2 PowerPC processor cores

[© and source: Xilinx Inc.: Virtex-II Pro™ Platform FPGAs: Functional Description, Sept. 2002, //www.xilinx.com]

Memory

- Speed gap between processor and main DRAM increases



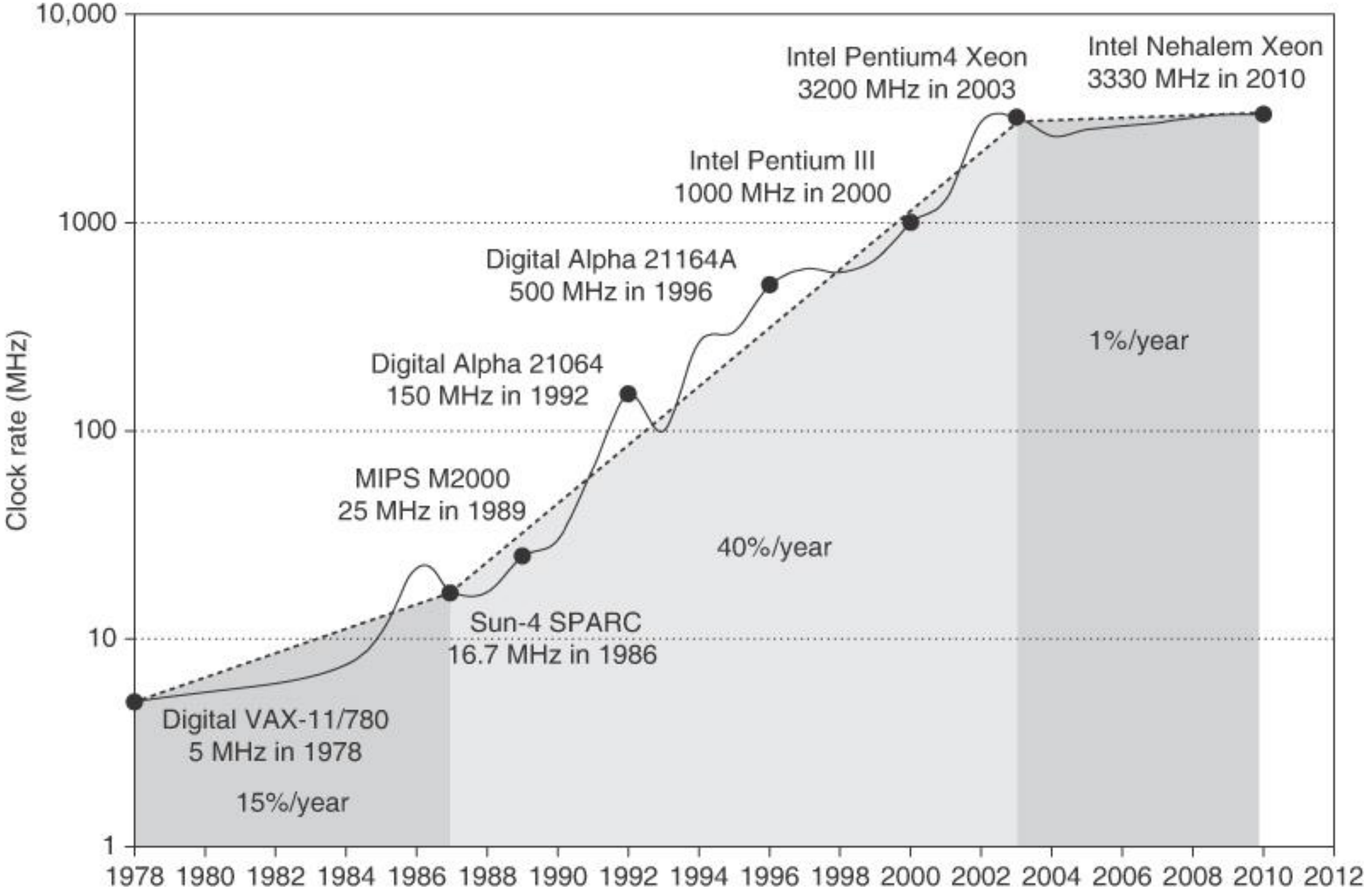
Similar problems also for embedded systems

☞ Memory access times >> processor cycle times

☞ “Memory wall” problem

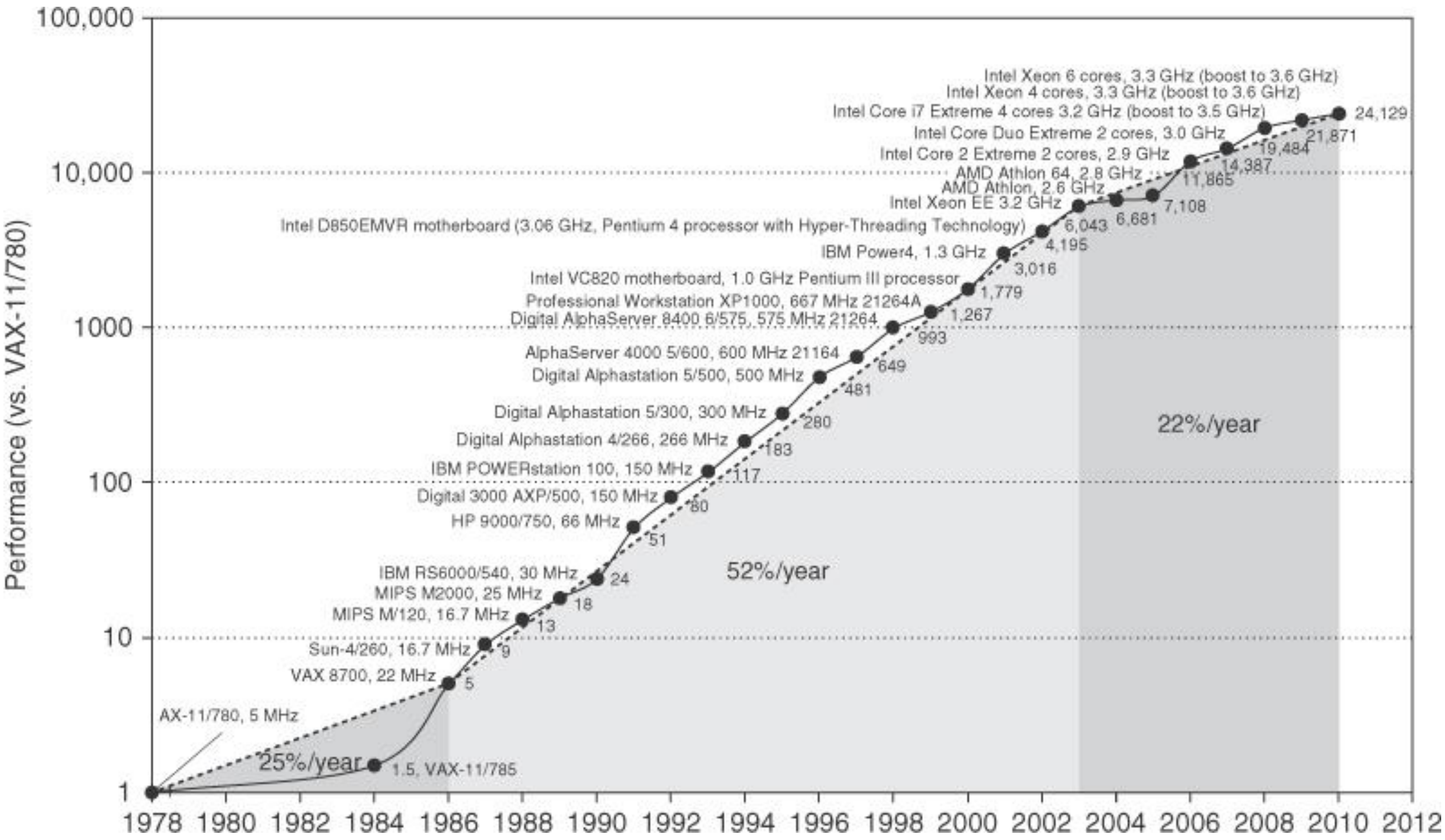
[P. Machanik: Approaches to Addressing the Memory Wall, TR Nov. 2002, U. Brisbane]

Clock speed



[Hennessy/Patterson: Computer Architecture, 5th ed., 2011]

Parallel performance

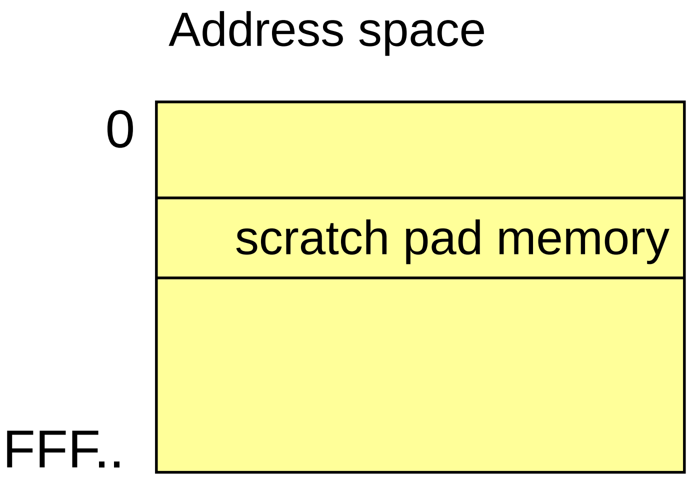


Copyright © 2011, Elsevier Inc. All rights Reserved.

[Hennessy/Patterson: Computer Architecture, 5th ed., 2011]

Hierarchical memories using scratch pad memories (SPM)

SPM is a small, physically separate memory mapped into the address space



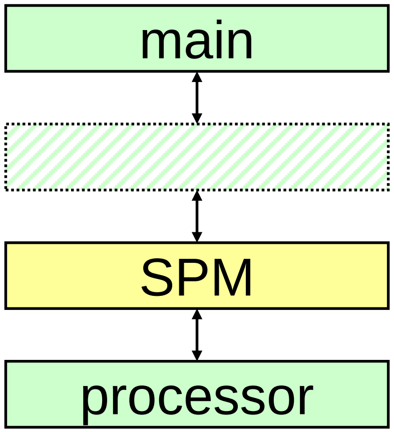
Address space

0

scratch pad memory

FFF..

Hierarchy

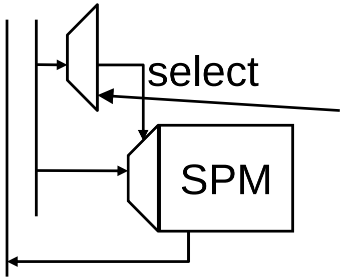


main

SPM

processor

no tag memory

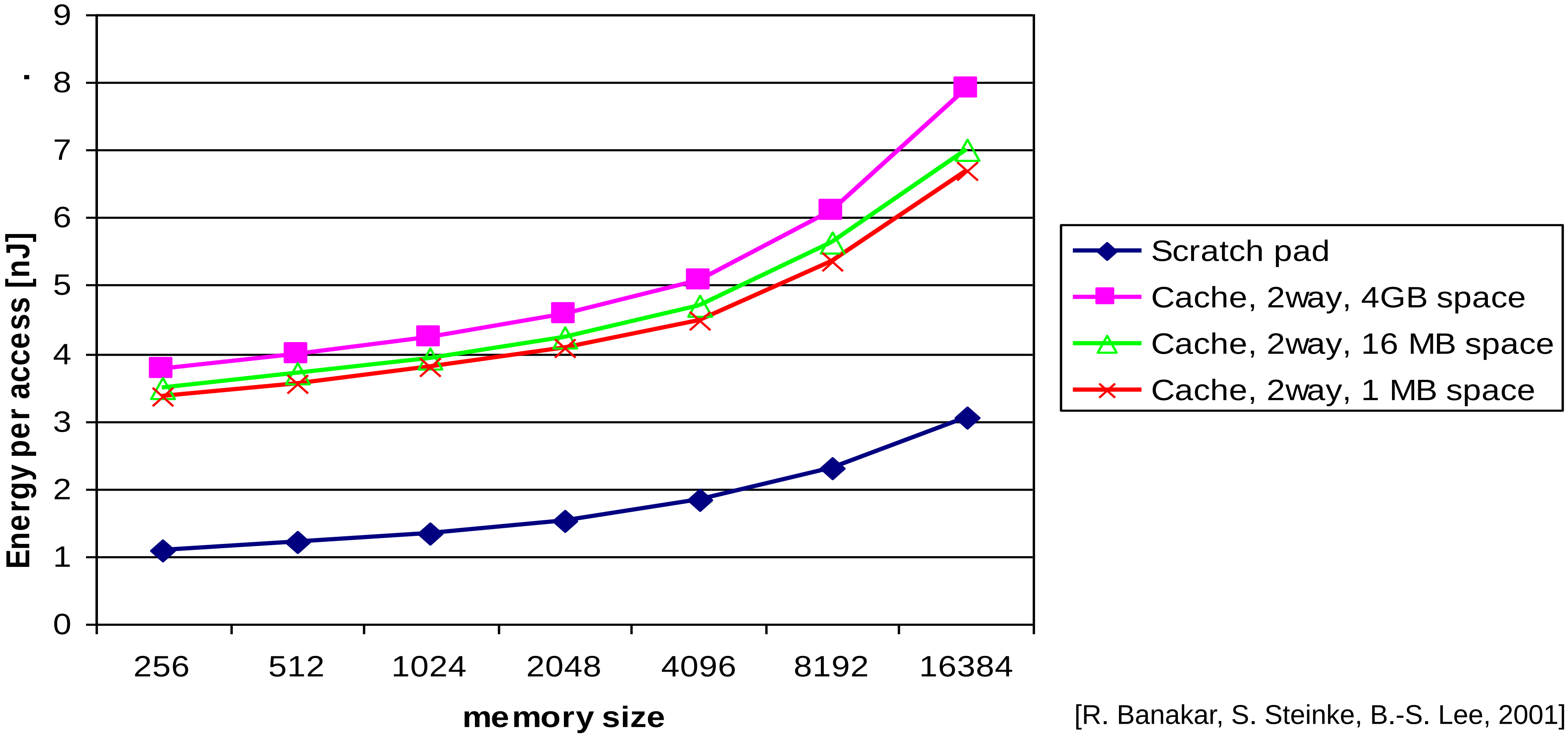


select

SPM

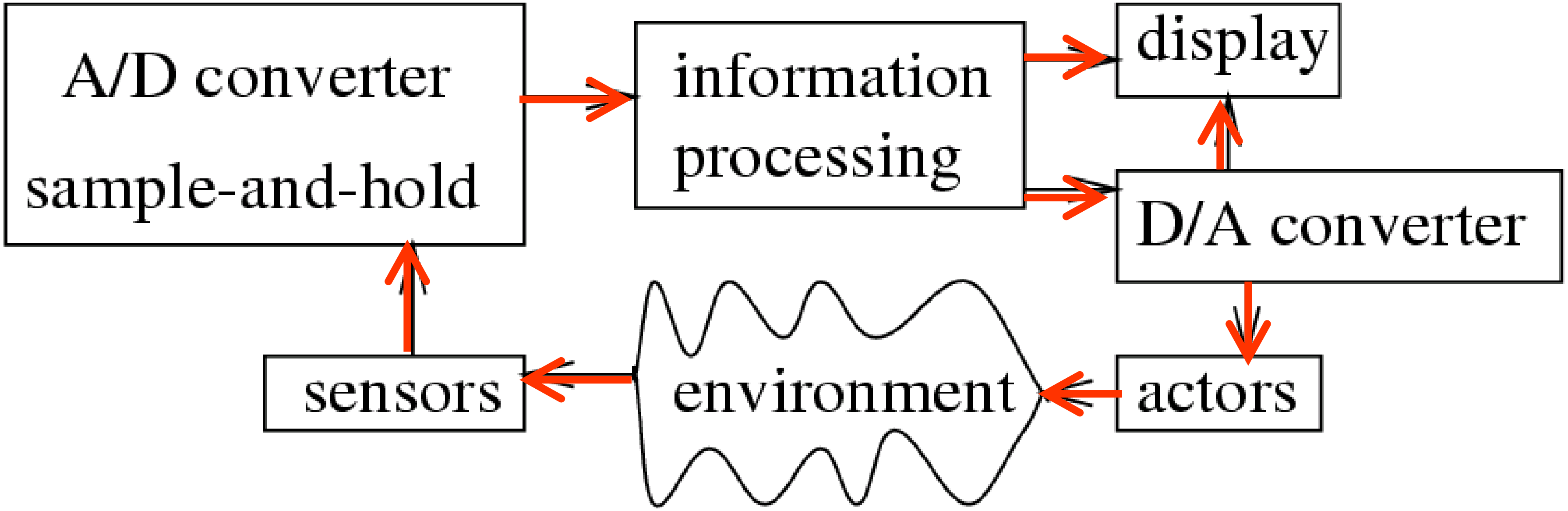
Selection is by an appropriate address decoder (simple!)

Energy consumption per memory access



[R. Banakar, S. Steinke, B.-S. Lee, 2001]

Communication

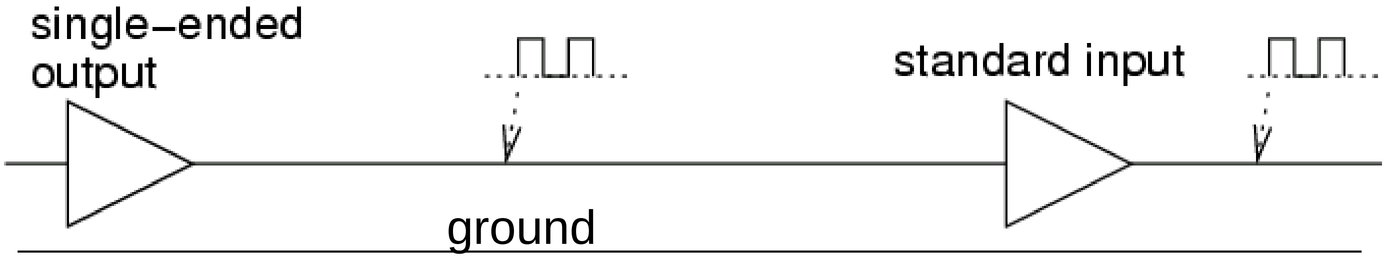


Communication requirements

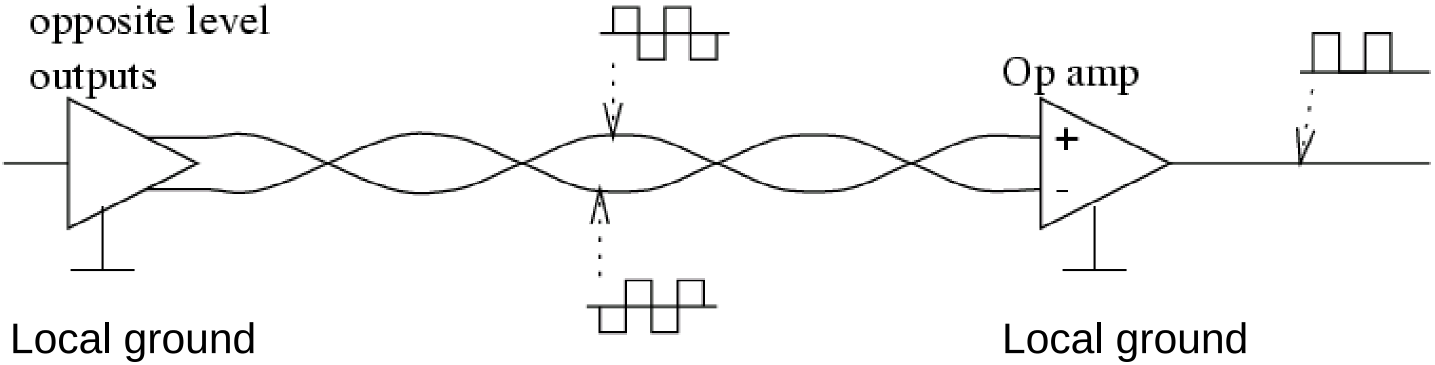
- Real-time behavior
- Efficient, economical
  (e.g. centralized power supply)
- Appropriate bandwidth and communication delay
- Robustness
- Fault tolerance
- Diagnosability
- Maintainability
- Security
- Safety

Basic techniques: Electrical robustness

- Single-ended vs. differential signals



Voltage at input of Op-Amp positive → '1'; otherwise → '0'



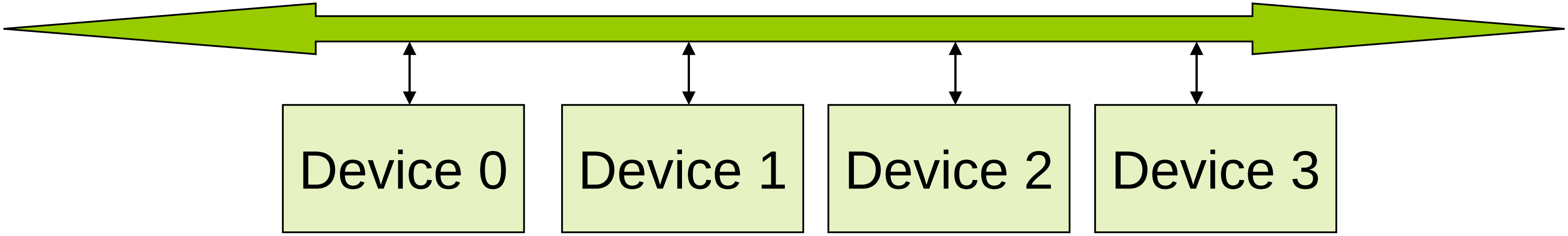
Combined with twisted pairs; Most noise added to both wires.

Evaluation

- **Advantages:**
  - Subtraction removes most of the noise
  - Changes of voltage levels have no effect
  - Reduced importance of ground wiring
  - Higher speed
- **Disadvantages:**
  - Requires negative voltages
  - Increased number of wires and connectors
- **Applications:**
  - USB, FireWire, ISDN
  - Ethernet (STP/UTP CAT 5/6 cables)
  - differential SCSI
  - High-quality analog audio signals (XLR)

Priority-based arbitration of communication media

For example, consider a bus



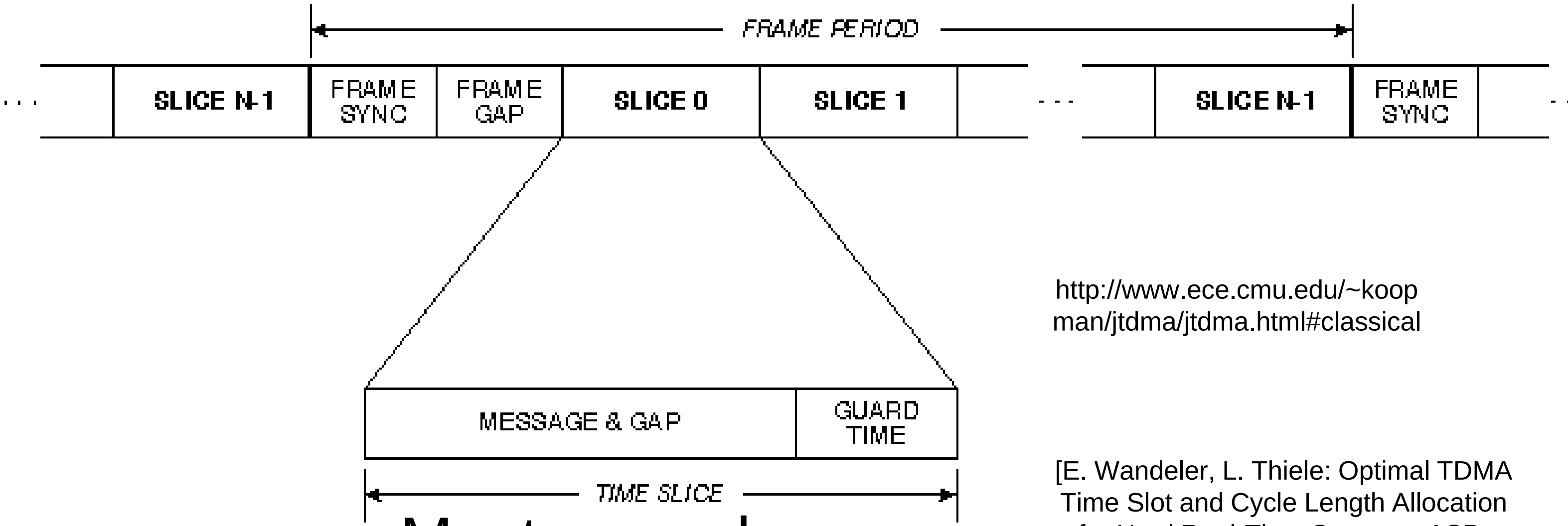
- ▪ Bus arbitration (allocation) is frequently priority-based
- ☞ Communication delay depends on communication traffic of other partners
- ☞ No tight real-time guarantees, except for highest priority partner

Ethernet

- Carrier-sense multiple-access/collision-**detection** (CSMA/C**D**, Standard Ethernet): no guaranteed response time.
- Alternatives:
  - token rings, token busses
  - Carrier-sense multiple-access/collision-**avoidance** (CSMA/C**A**)
    - WLAN techniques with request preceding transmission
    - Each partner gets an ID (priority).
      After bus transfer: partners try setting their ID on the bus;
      Partners detecting higher ID disconnect themselves. Highest priority partner gets guaranteed response time; others only if they are given a chance.

Time division multiple access (TDMA) busses

- Each communication partner is assigned a fixed time slot. Example:

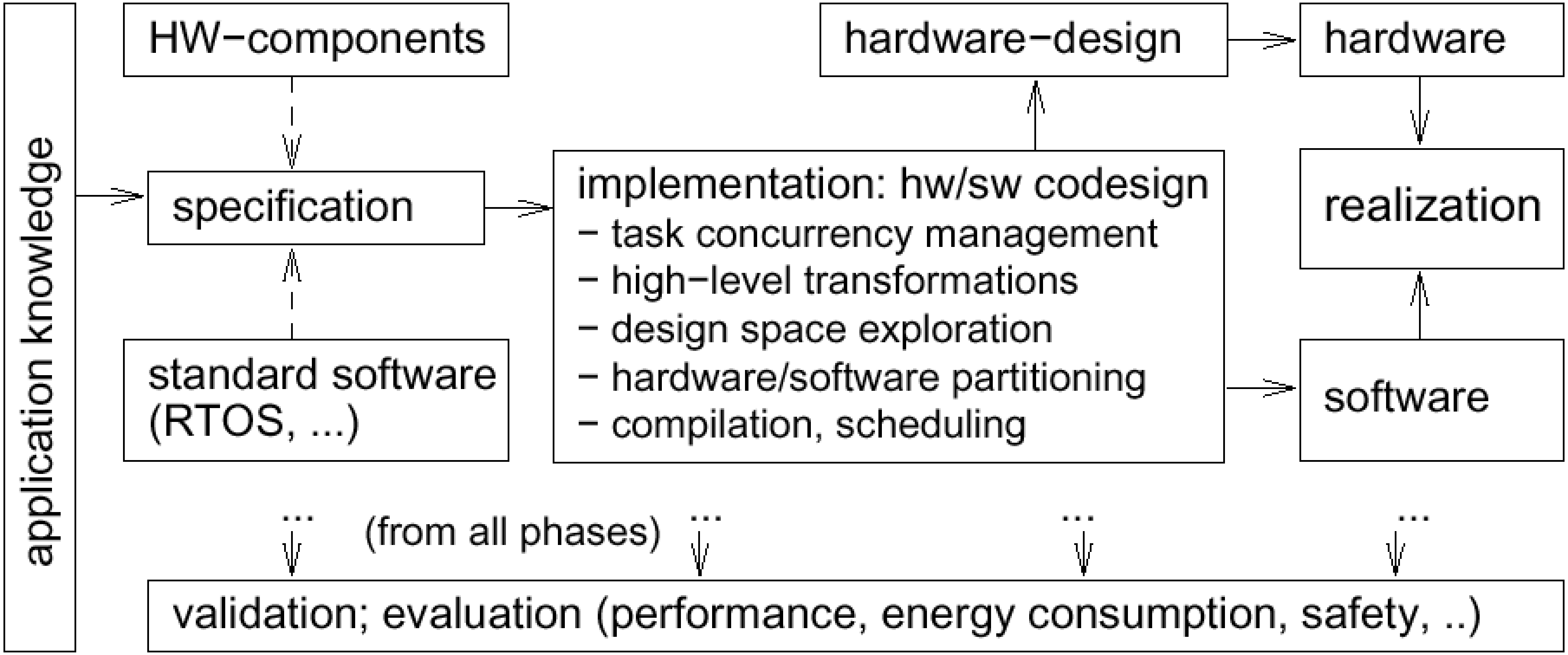


http://www.ece.cmu.edu/~koopman/jtdma/jtdma.html#classical

[E. Wandeler, L. Thiele: Optimal TDMA Time Slot and Cycle Length Allocation for Hard Real-Time Systems, ASP-DAC, 2006]

  - Master sends sync
  - Some waiting time
  - Each slave transmits in its time slot
- **TDMA resources have a deterministic timing behavior**
- **TDMA provides QoS guarantees in networks on chips**

Overview of embedded systems design



application knowledge

HW−components

specification

standard software (RTOS, ...)

implementation: hw/sw codesign
- task concurrency management
- high−level transformations
- design space exploration
- hardware/software partitioning
- compilation, scheduling

hardware−design

hardware

realization

software

... (from all phases) ... ...

validation; evaluation (performance, energy consumption, safety, ..)