7. Multiprocessors

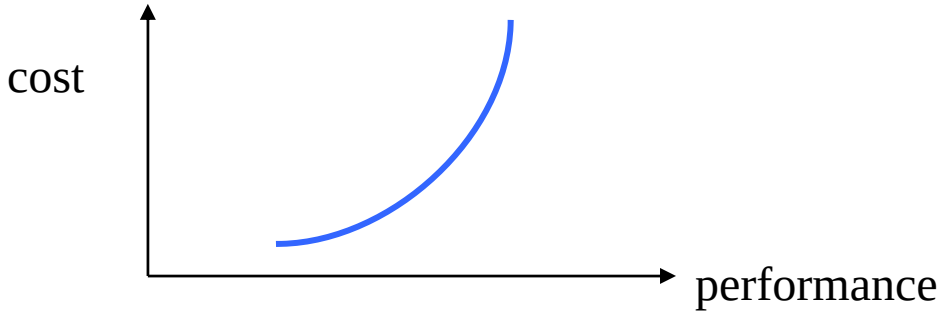
- Why multiprocessors?
- CPUs and accelerators.
- Multiprocessor performance analysis.

Why multiprocessors?

- Programming a single CPU is hard enough.
- Why make life more difficult by adding more processors?
- PE: processing element for computation
  - Whether is is programmable or not.
- Multiprocessors tend to have regular architectures
  - Several identical processors that can access a uniform memory space

Why multiprocessors?

- There are a variety of different multiprocessor architectures
- Better cost/performance.
  - Match each CPU to its tasks or use custom logic (smaller, cheaper).
  - CPU cost is a non-linear function of performance.

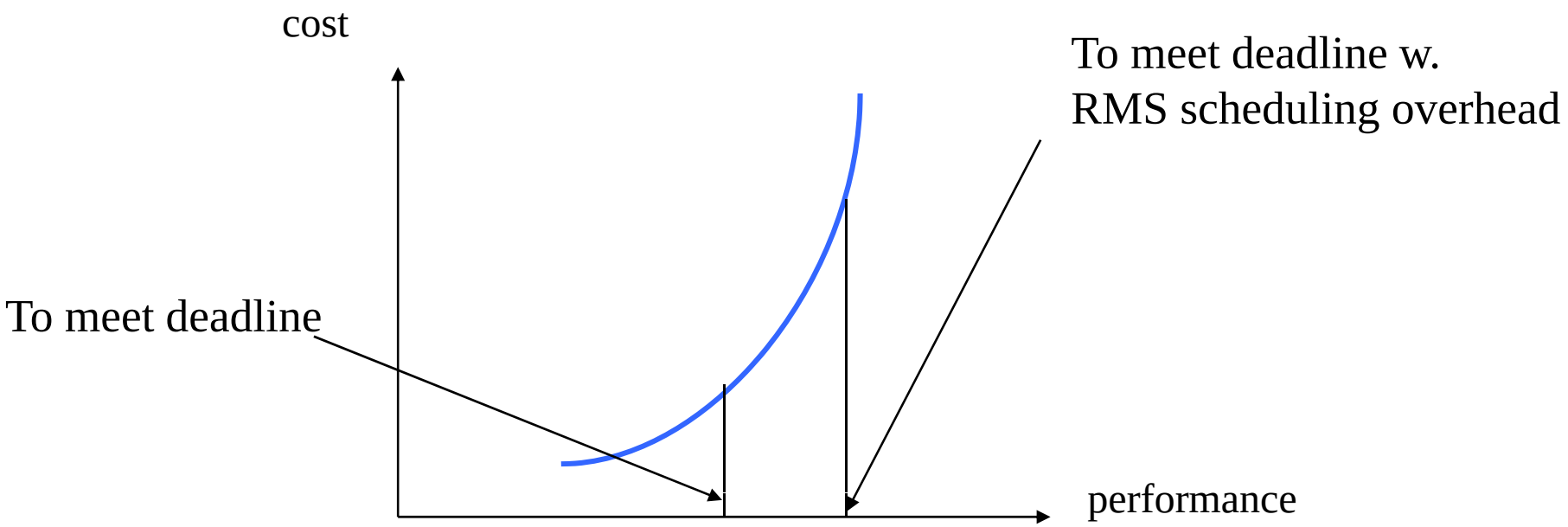


Power
Performance
Cost (Area)

Why multiprocessors?

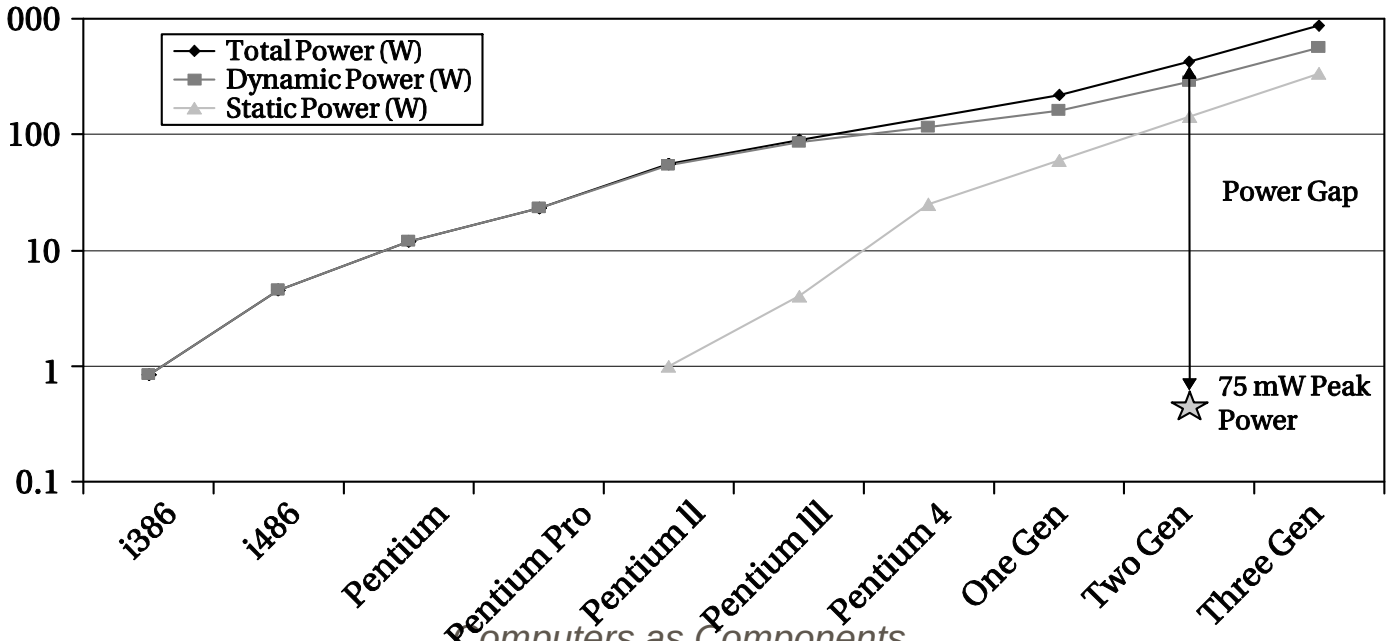
- Splitting the application across multiple processors entails higher engineering cost and lead times.
- Better real-time performance.
  - Put time-critical functions on less-loaded processing elements.
  - Remember RMS utilization---extra CPU cycles must be reserved to meet deadlines.

Why multiprocessors?



Why multiprocessors?

- Using specialized processors or custom logic saves power.
- Desktop processors are not power-efficient enough for battery-powered applications.



(battery)

Why multiprocessors?

- May consume less energy.
- May be better at streaming data.
- May not be able to do all the work on even the largest single CPU.

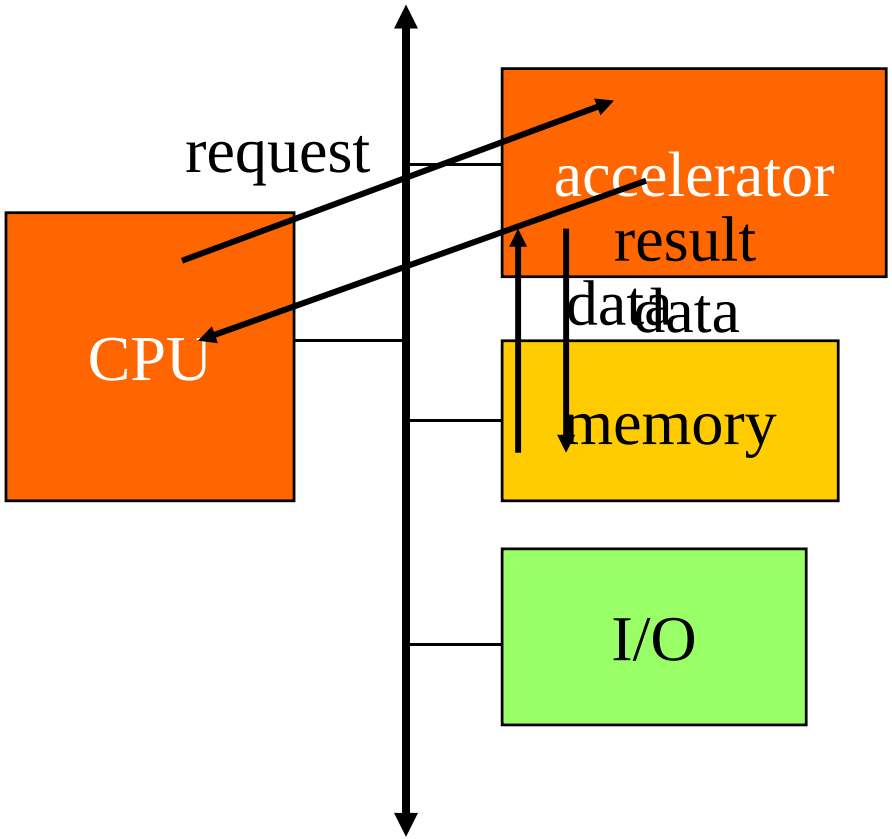
Why multiprocessors?

- Good for processing I/O in real-time.
- May consume less energy.
- May be better at streaming data.
- May not be able to do all the work on even the largest single CPU.
- A thread per processor
  - no context switching

Accelerated systems

- Use additional computational unit dedicated to some functions?
  - Hardwired logic.
  - Extra CPU.
- Hardware/software co-design: joint design of hardware and software architectures.

Accelerated system architecture



Accelerator vs. co-processor

- A co-processor executes instructions.
  - Instructions are dispatched by the CPU.
- An accelerator appears as a device on the bus.
  - Its interface is functionally equivalent to an I/O device
  - is controlled by it registers.

Accelerator implementations

- Application-specific integrated circuit.
- Field-programmable gate array (FPGA).
- Standard component.
  - Example: graphics processor.

System design tasks

- Design a heterogeneous multiprocessor architecture.
  - Processing element (PE): CPU, accelerator, etc.
- Program the system.

Accelerated system design

- First, determine that the system really needs to be accelerated.
  - How much faster is the accelerator on the core function?
  - How much data transfer overhead?
- Design the accelerator itself.
- Design CPU interface to accelerator.

Accelerated system platforms

- Several off-the-shelf boards are available for acceleration in PCs:
  - FPGA-based core;
  - PC bus interface.

Accelerator/CPU interface

- Accelerator registers provide control registers for CPU.
- Data registers (buffers) can be used for small data objects.
- Accelerator may include special-purpose read/write logic.
  - Especially valuable for large data transfers.
  - DMA to transfer a large volume of data without intervention of CPU

System integration/debugging

- Try to debug the CPU/accelerator interface separately from the accelerator core.
- Build scaffolding to test the accelerator.
- Hardware/software co-simulation can be useful.

Caching problems

- Main memory provides the primary data transfer mechanism to the accelerator.
- Programs must ensure that caching does not invalidate main memory data.
  - CPU reads location S.
  - Accelerator writes location S.
  - CPU writes location S.

Synchronization

- As with cache, main memory writes to shared memory may cause invalidation:
  - CPU reads S.
  - Accelerator writes S.
  - CPU reads S.

Mobile Phone Trends

TABLE I
MOBILE PHONE TRENDS IN 5-YEAR INTERVALS.

year	1995	2000	2005	2010	2015
cellular generation	2G	2.5-3G	3.5G	pre-4G	4G
cellular standards	GSM	GPRS UMTS	HSPA	HSPA LTE	LTE LTE-A
downlink bitrate [Mb/s]	0.01	0.1	1	10	100
display pixels [×1000]	4	16	64	256	1024
battery energy [Wh]	1	2	3	4	5
CMOS [ITRS, nm]	350	180	90	50	25
PC CPU clock[MHz]	100	1000	3000	6000	8500
PC CPU power [W]	5	20	100	200	200
PC CPU MHz/W	20	50	30	30	42
phone CPU clock[MHz]	20	100	200	500	1000
phone CPU power [W]	0.05	0.05	0.1	0.2	0.3
phone CPU MHz/W	400	2000	2000	2500	3000
workload [GOPS]	0.1	1	10	100	1000
software [MB]	0.1	1	10	100	1000
#programmable cores	1	2	4	8	16

Power and Battery Capacity

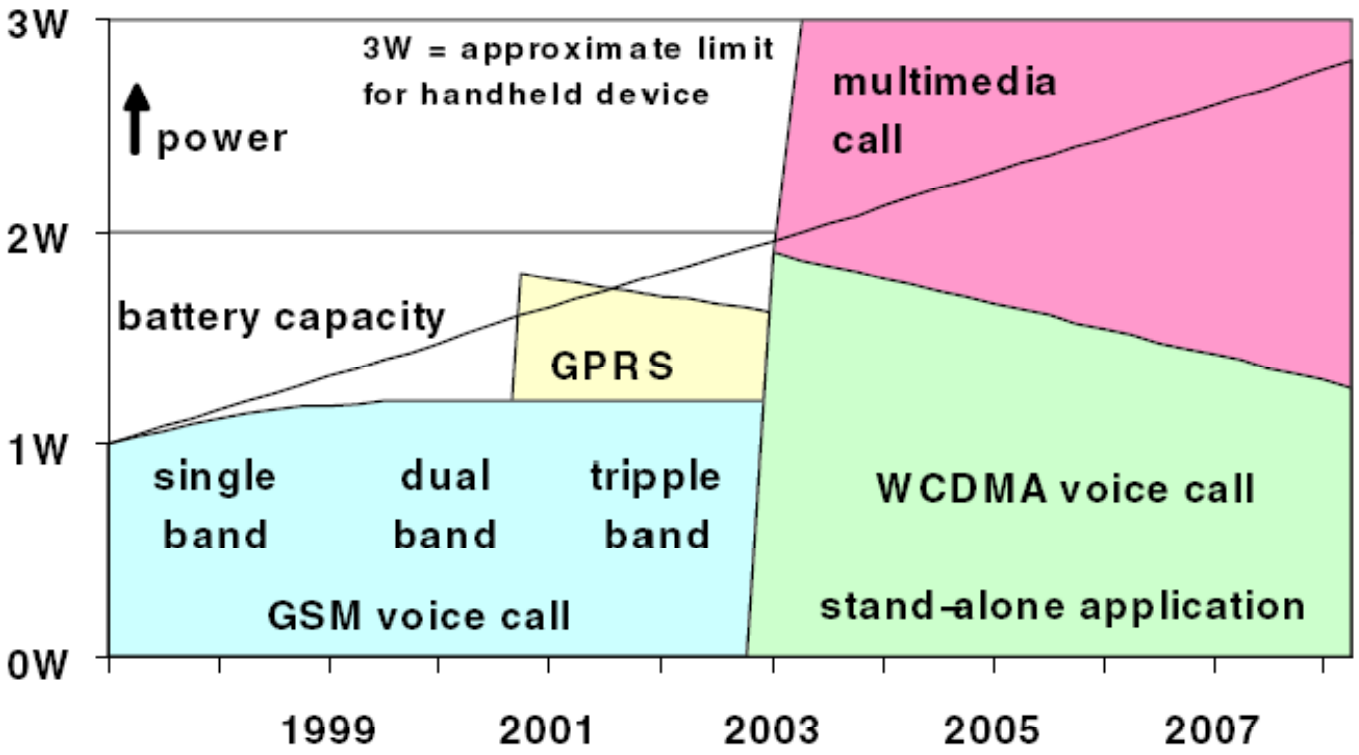
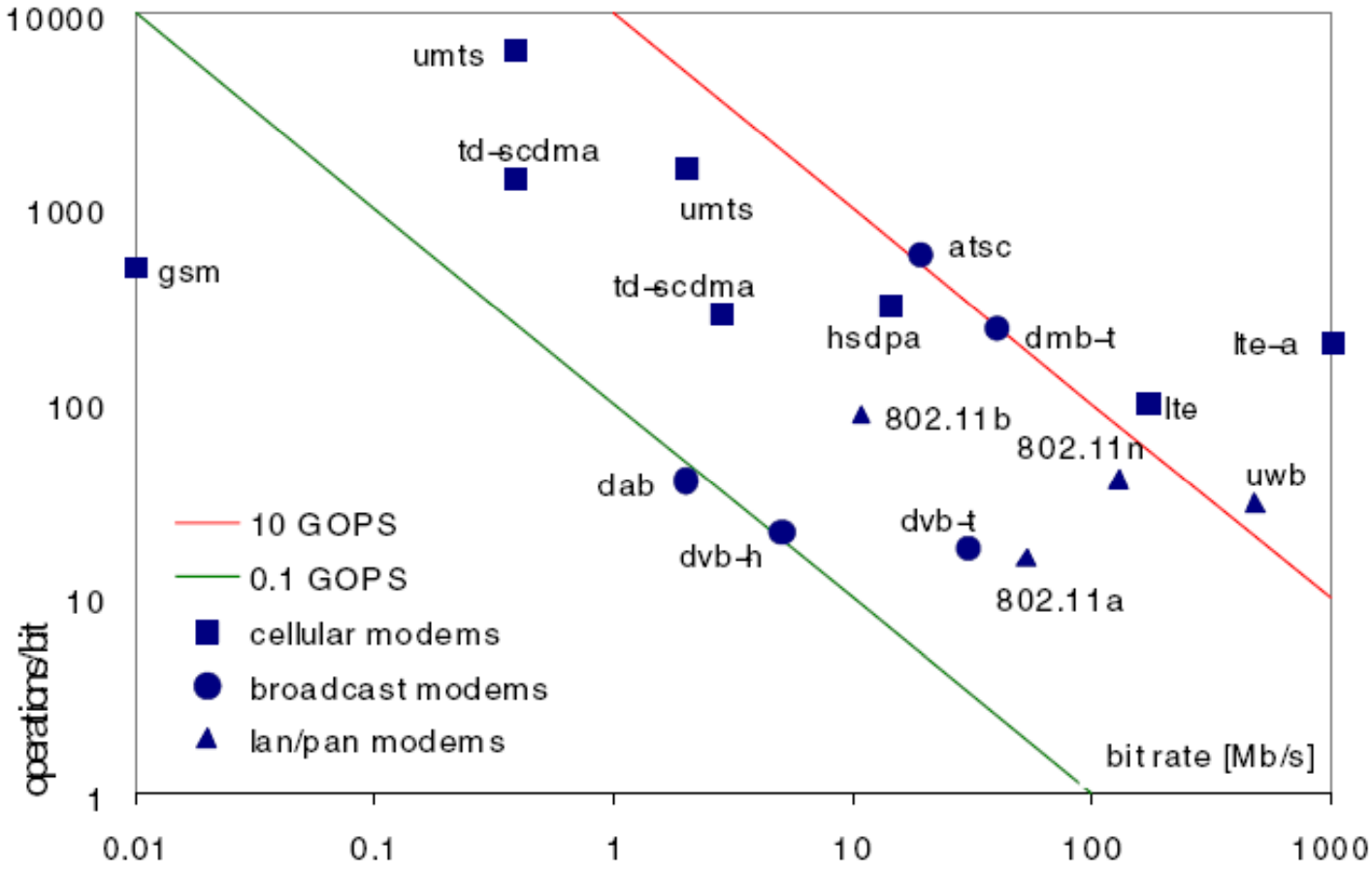
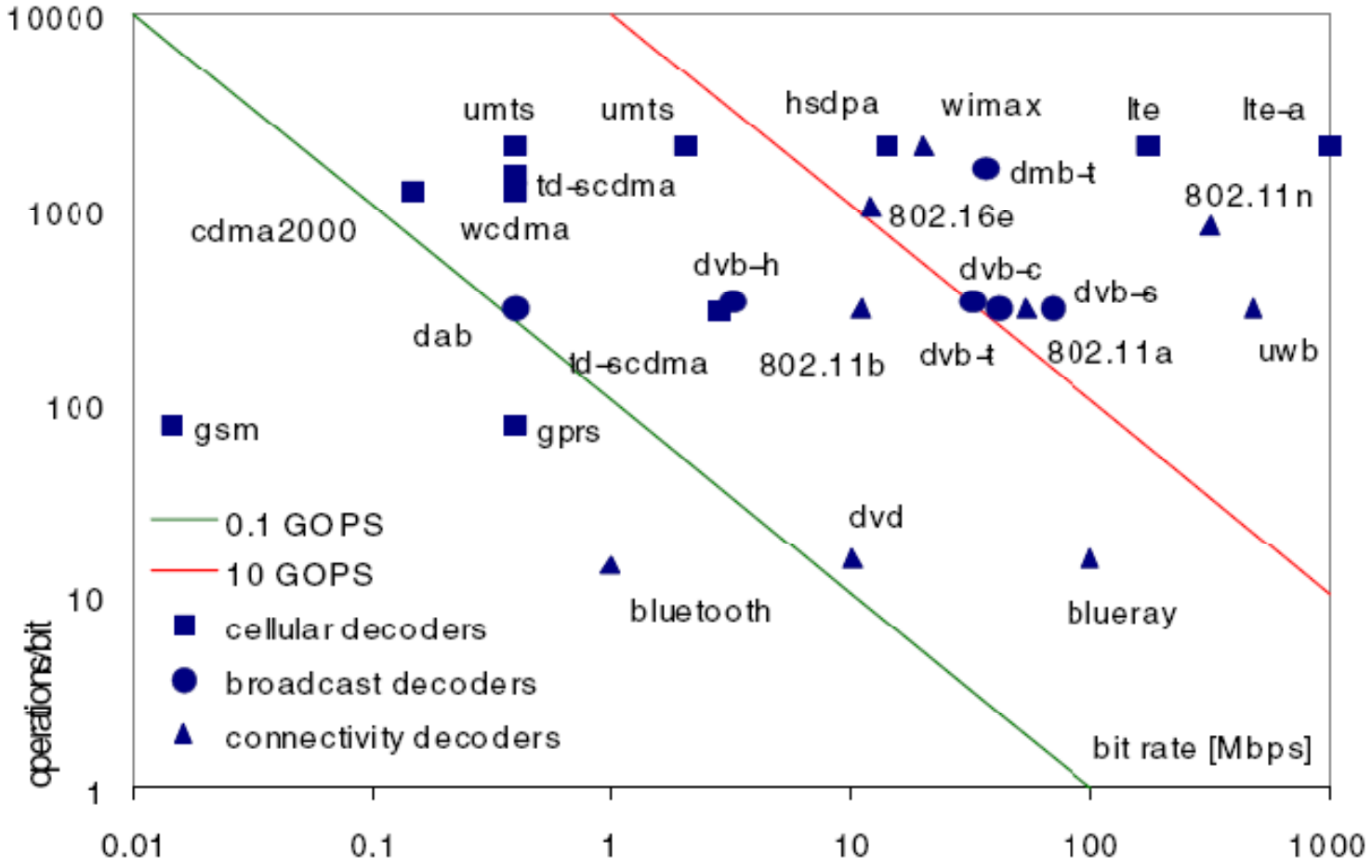


Fig. 1. Battery capacity and power consumption at maximum output power level in cellular transmitters, adapted from [3].

Radio Demodulation Workload



Radio Decoding Workload



3.5G Workload

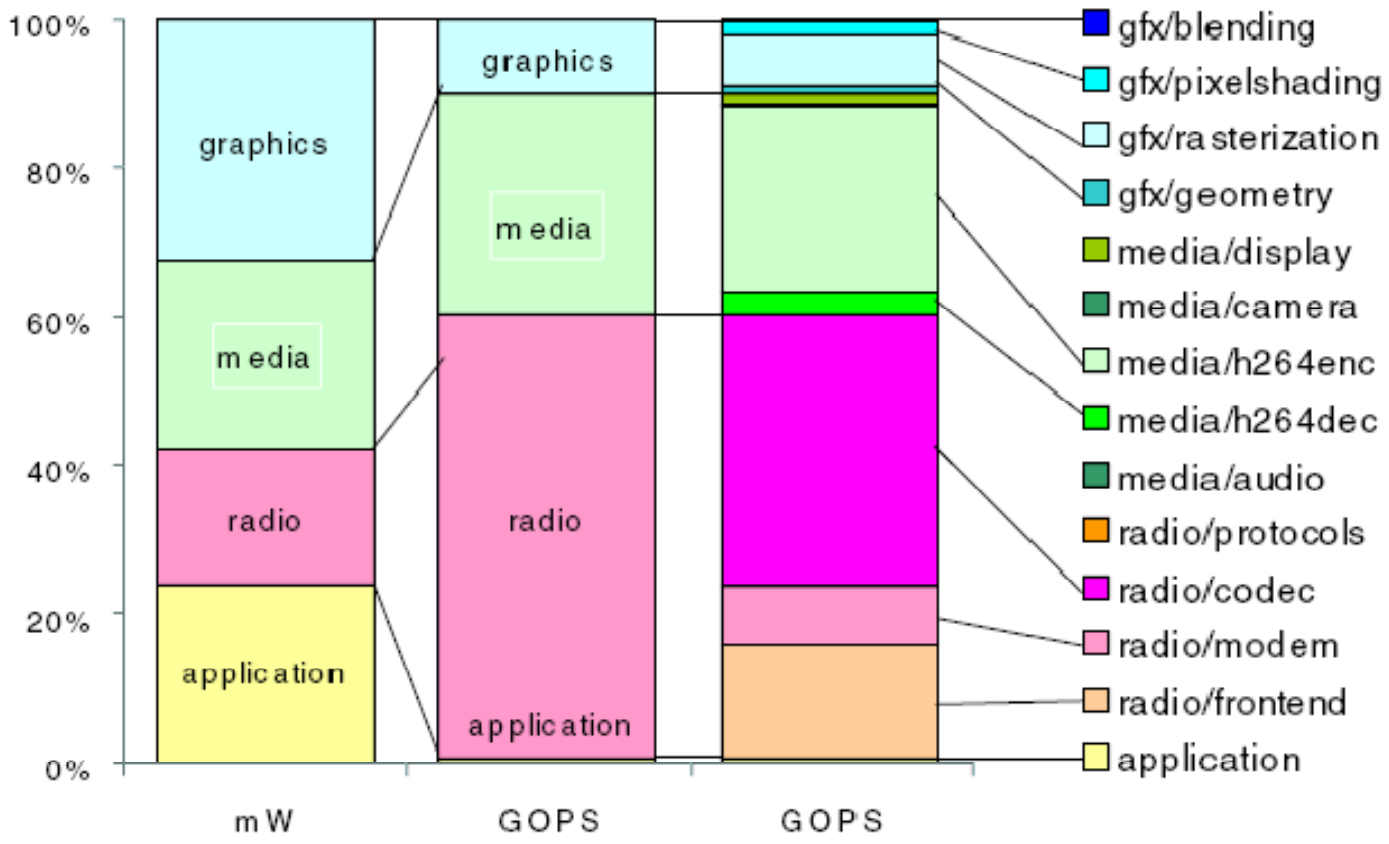


Fig. 3. 3.5G workload (power consumption) as fractions of 100GOPS (1W).

Workload vs Energy/operation

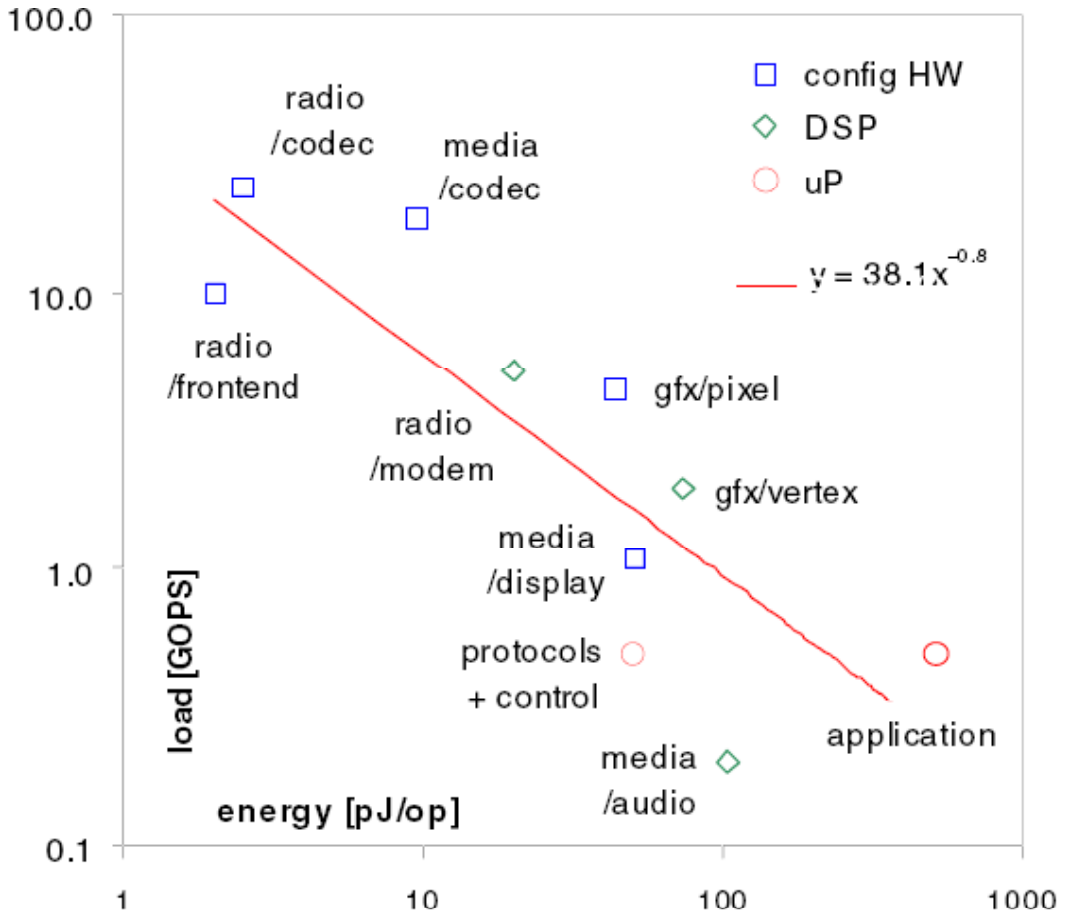


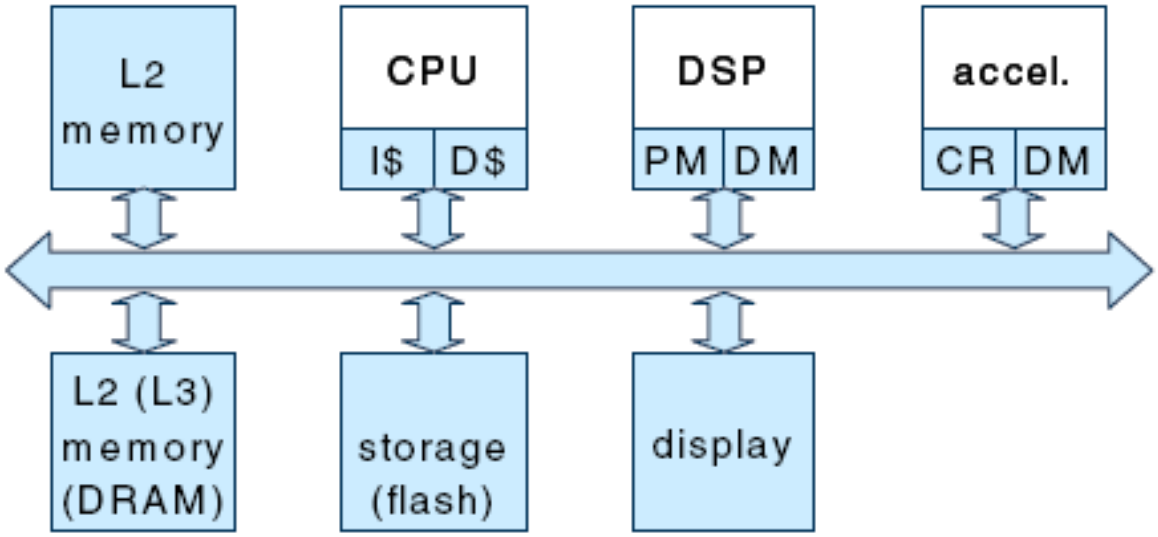
Fig. 4. Workloads [GOPS] versus energy/operation [pJ].

Value of Programming

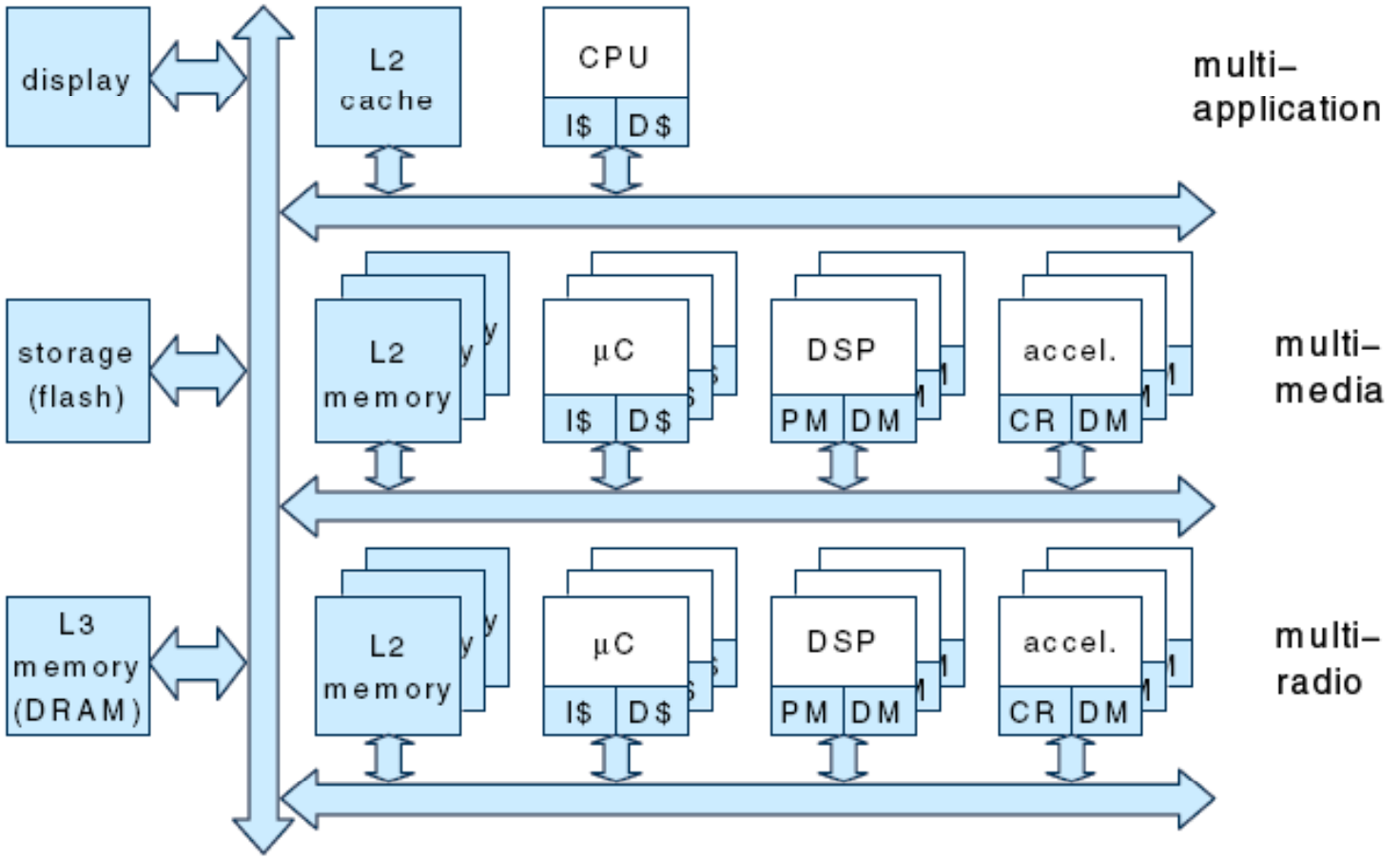
TABLE II
VALUE-AFFORDABILITY OF PROGRAMMABLE SOLUTIONS (EXAMPLES).

	radio	video	3D graphics
very high	protocol stacks		geometry proces.
high	channel estimation		pixel shading
medium	demodulation	motion estimation	
low	turbo decoder (i)fft	entropy (de)coding deblocking	
very low	filters	filters scaling	rasterization pixel blending

2/2.5G Dual-core Architecture



3/3.5G Multi-core Architecture



Cell-phone Chips

TABLE III
PUBLISHED INDUSTRIAL CELL-PHONE CHIPS ("..." DENOTES A SHARED RADIO-MEDIA CORE).

year	ref	source	cmos	total	application		radio		media	
			nm	# cores	core(s)	MHz	core(s)	MHz	core(s)	MHz
1992	[1]	Philips	1000	1	n.a.		KISS-16-V2	...	...	20
2000	[2]	Infineon	250	2	CPU	78	Oak	...	...	78
2001	[9]	Samsung	180	2	ARM9		Teaklite		n.a.	
2003	[6]	Toshiba	130	3	n.a.		n.a.		3x RISC	125
2004	[3]	Nokia	130	2	CPU	50	DSP	...	...	160
2004	[10]	Qualcomm	130	3	ARM9	180	DSP	95	DSP	95
2004	[11]	Renesas	130	2	CPU	216	n.a.		DSP	216
2005	[12]	NEC	130	4	ARM9	200	n.a.		2xARM9 + DSP	200
2006	[13]	ST	130	2	ARM8	156	ST122 DSP	...	...	156
2007	[14]	Infineon	90	2	ARM9	380	TEAKlite	...	...	104
2008	[15]	Renesas et al	65	4	ARM11	500	ARM9	166	ARM11 + SHX2	500
2008	[16]	TI	45	5	ARM11	840	?		C55 + ?	480
2008	[17]	NEC	65	3	ARM11	500	ARM11	250	DSP	500
2009	[18]	Panasonic	45	4	ARM11	486	ARM11	245	2xDSP	216
2009	[19]	Renesas	65	2	CPU	500	n.a.		SHX2	500

Power Management Knobs

TABLE IV

POWER MANAGEMENT KNOBS: f_C DENOTES CLOCK FREQUENCY, V_{DD} AND V_t DENOTE SUPPLY AND THRESHOLD VOLTAGE, AND P_D AND P_S DENOTE DYNAMIC AND STATIC POWER CONSUMPTION.

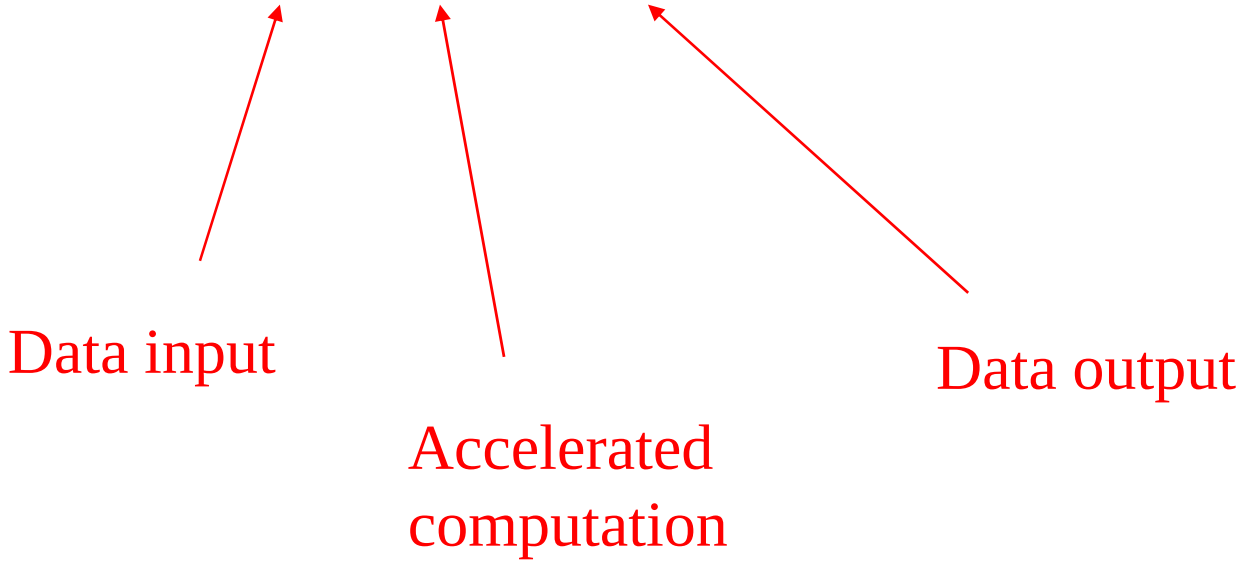
	knob		throughput	power
stop the clock:	$f_C \downarrow 0$	$\Rightarrow$	$\downarrow 0$	$P_D \downarrow 0$
frequency scaling (FS)	$f_C \downarrow$	$\Rightarrow$	$\downarrow$	$P_D \downarrow$
voltage scaling (VS)	$V_{DD} \downarrow$	$\Rightarrow$	$\downarrow$	$P_D \downarrow$
power down	$V_{DD} \downarrow 0$	$\Rightarrow$	$\downarrow 0$	$P_S \downarrow 0$
forward body bias (FBB)	$V_t \downarrow$	$\Rightarrow$	$\uparrow$	$P_S \uparrow$
reverse body bias (RBB)	$V_t \uparrow$	$\Rightarrow$	$\downarrow$	$P_S \downarrow$

Accelerator speedup

- Critical parameter is speedup: how much faster is the system with the accelerator?
- Must take into account:
  - Accelerator execution time.
  - Data transfer time.
  - Synchronization with the master CPU.

Accelerator execution time

- Total accelerator execution time:
  - $t_{accel} = t_{in} + t_x + t_{out}$



  - $t_{accel} = \max \{t_{in}, t_x, t_{out}\}$ if pipelined

Accelerator speedup

- Assume loop is executed n times.
- If the software loop is replaced with the accelerator, compare accelerated system to non-accelerated system:
  - $S = n(t_{CPU} - t_{accel})$
    $= n[t_{CPU} - (t_{in} + t_x + t_{out})]$

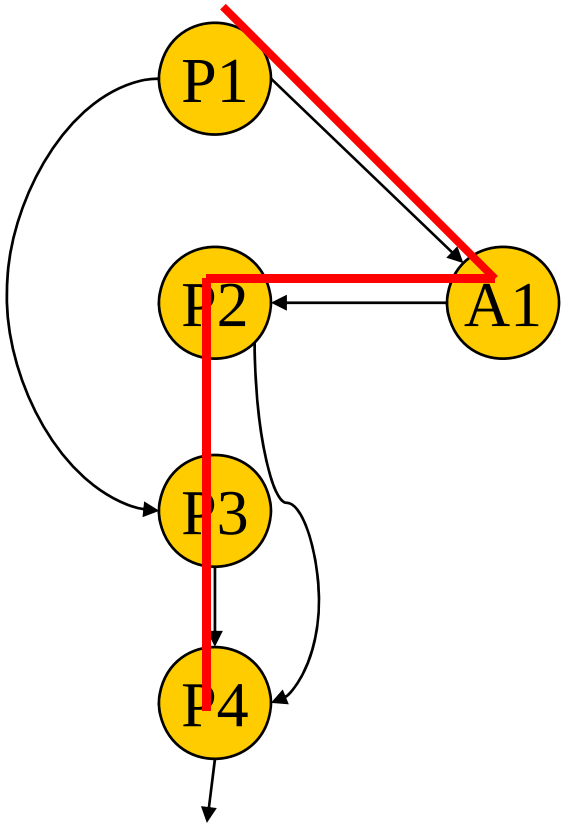
Execution time on CPU

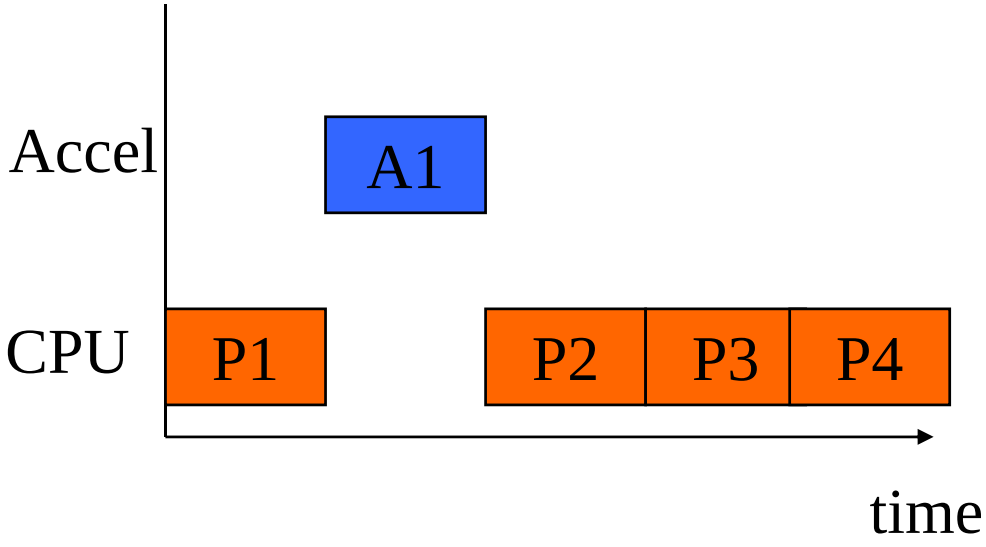
Single- vs. multi-threaded

- One critical factor is available parallelism:
  - single-threaded/blocking: CPU waits for accelerator;
  - multithreaded/non-blocking: CPU continues to execute along with accelerator.
- To multithread, CPU must have useful work to do.
  - Synchronization: software must also support multithreading.

Total execution time

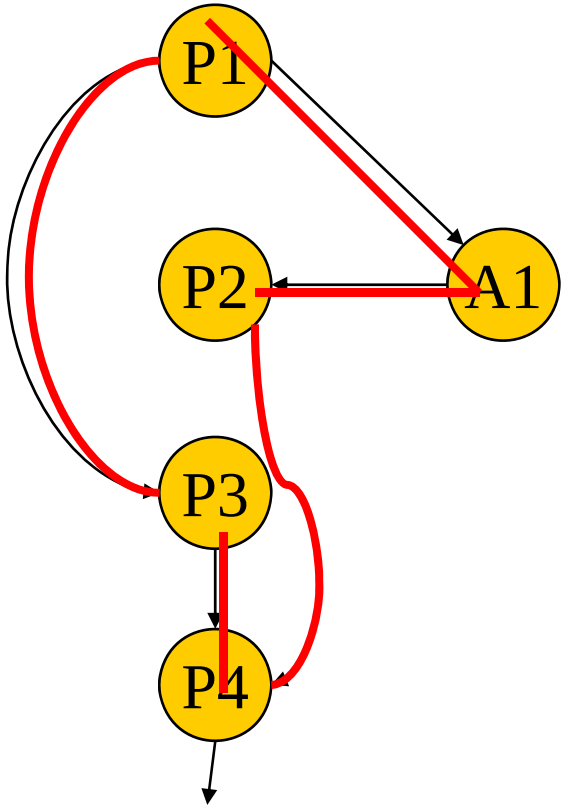
- Single-threaded:

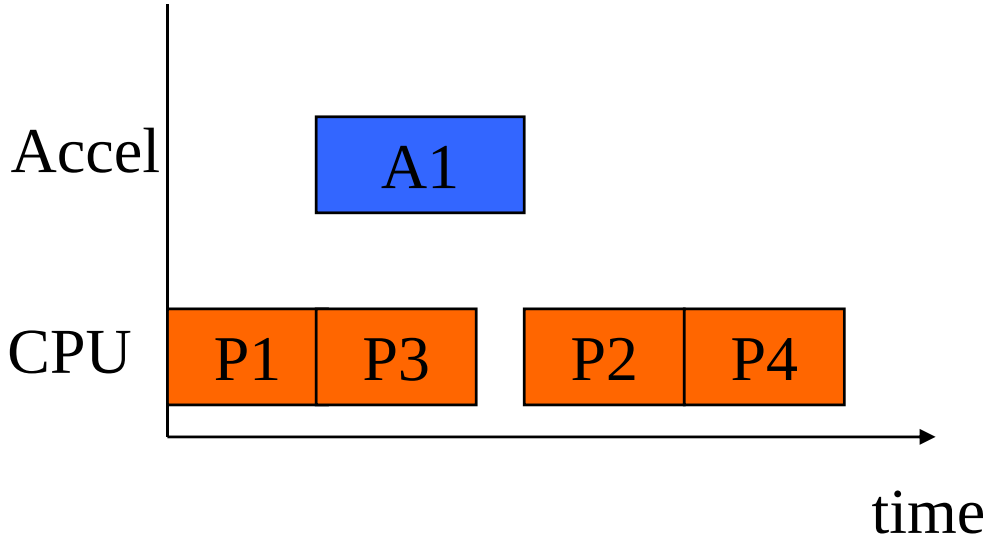




Total execution time

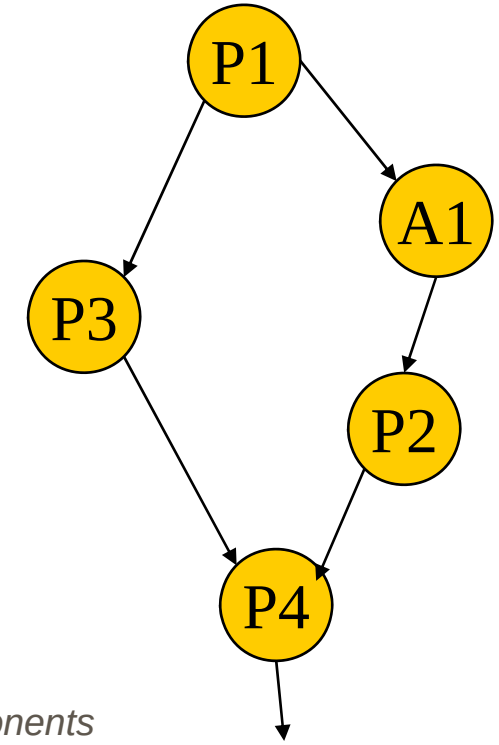
- Multi-threaded:





Execution time analysis

- Single-threaded:
  - Count execution time of all component processes.
- Multi-threaded:
  - Find longest path through execution.



Sources of parallelism

- Overlap I/O and accelerator computation.
  - Perform operations in batches, read in second batch of data while computing on first batch.
- Find other work to do on the CPU.
  - May reschedule operations to move work after accelerator initiation.

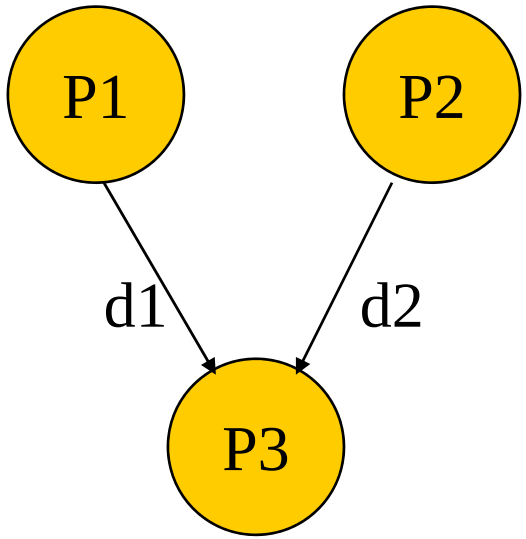
Data input/output times

- Bus transactions include:
  - flushing register/cache values to main memory if necessary;
  - time required for CPU to set up transaction;
  - overhead of data transfers by bus packets, handshaking, etc.

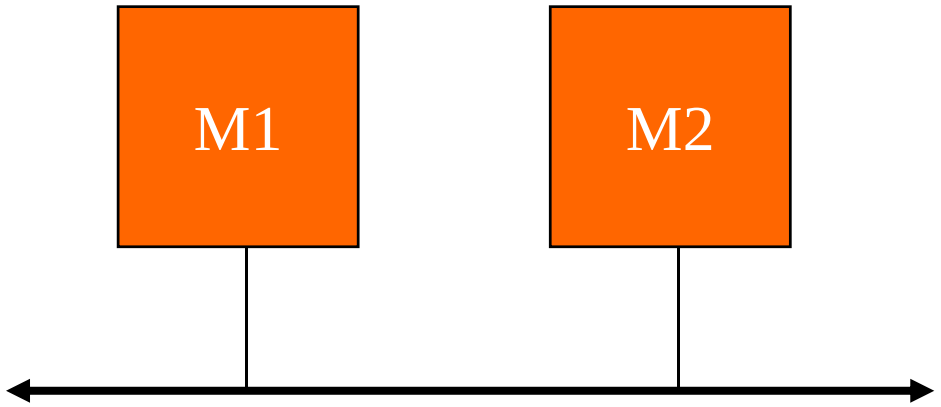
Scheduling and allocation

- Must:
  - schedule operations in time;
  - allocate computations to processing elements.
- Scheduling and allocation interact, but separating them helps.
  - (Alternatively) allocate, then schedule.

Example: scheduling and allocation



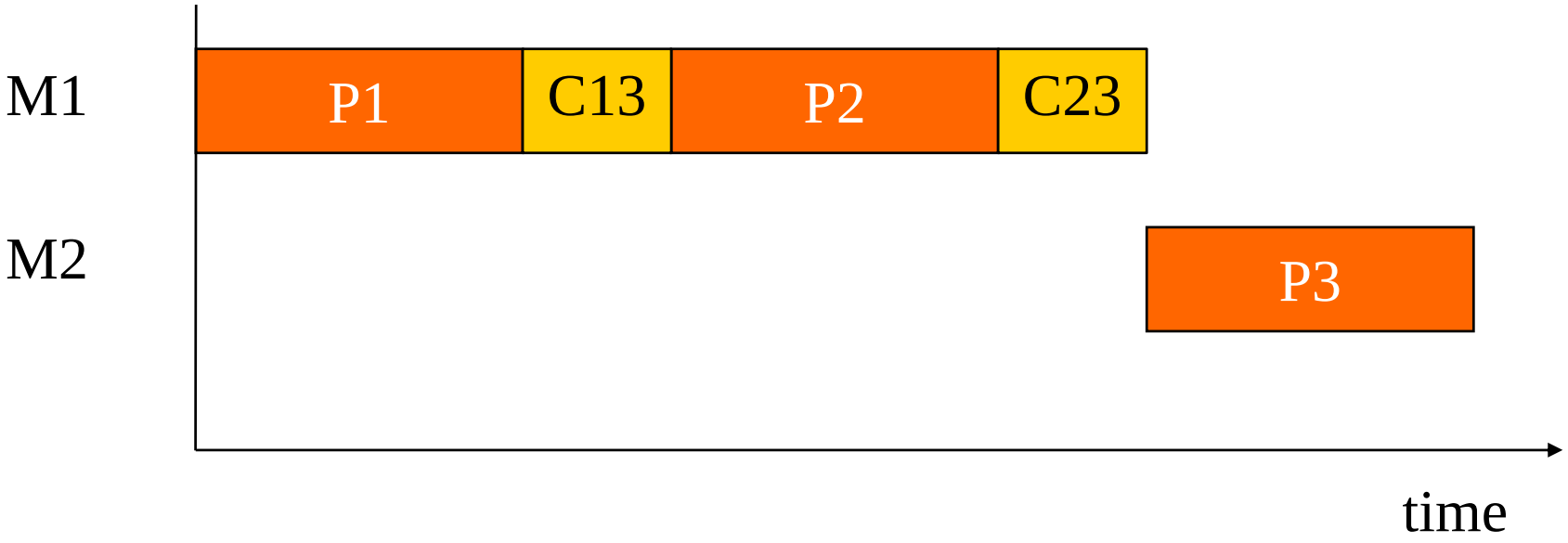
Task graph



Hardware platform

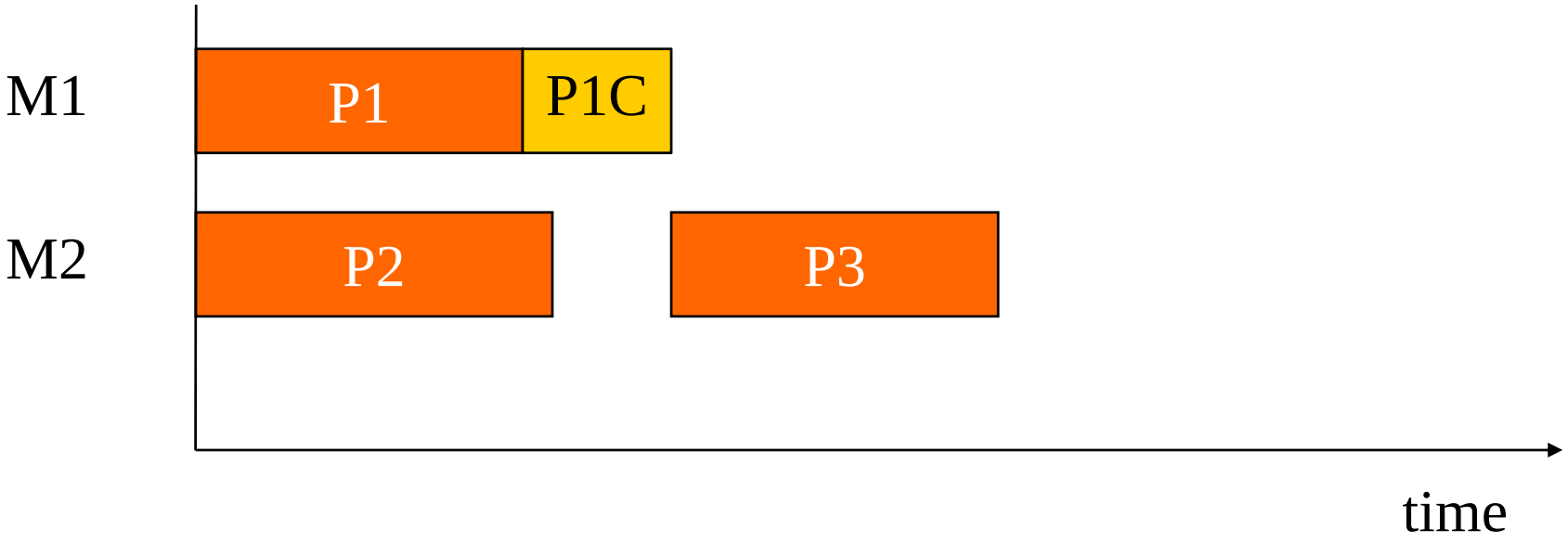
First design

- Allocate P1, P2 -> M1; P3 -> M2.



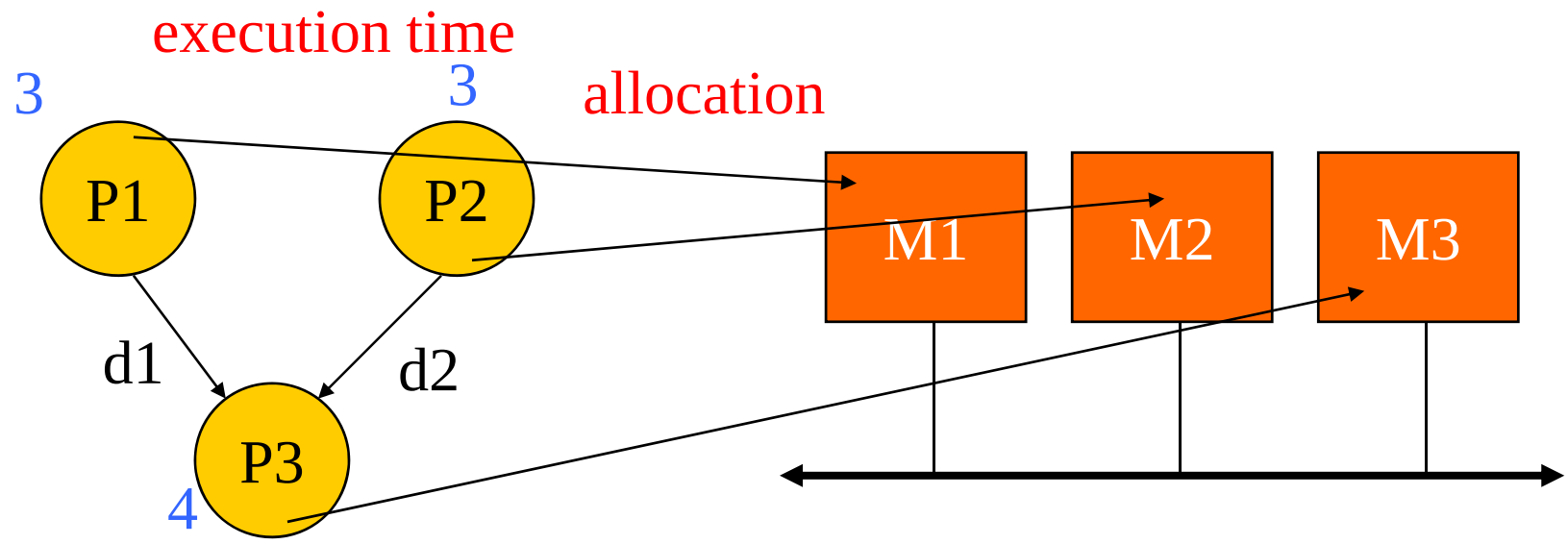
Second design

- Allocate P1 -> M1; P2, P3 -> M2:



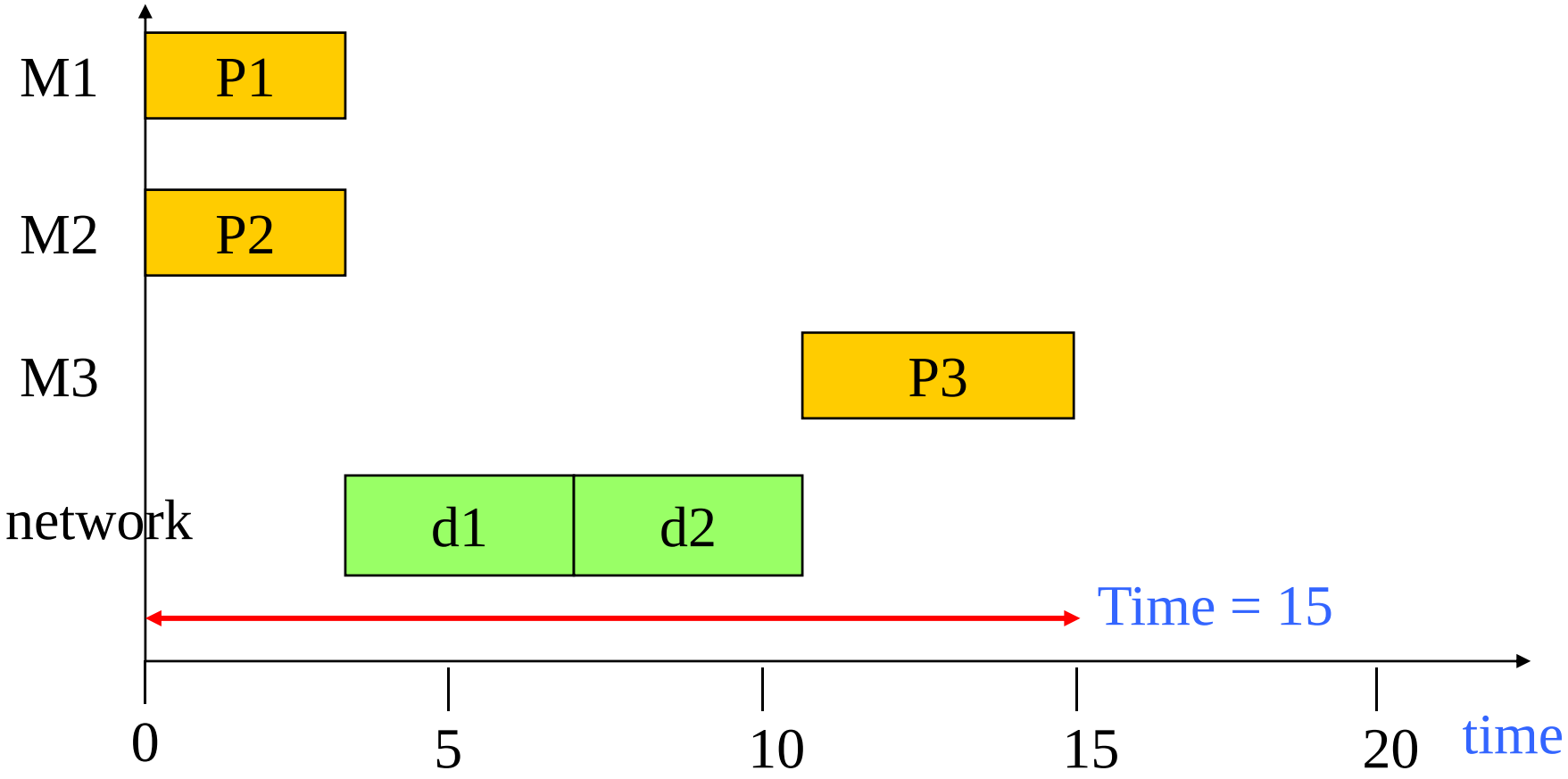
Example: adjusting messages to reduce delay

- Task graph:
- Network:



Transmission time = Td1=Td2=4

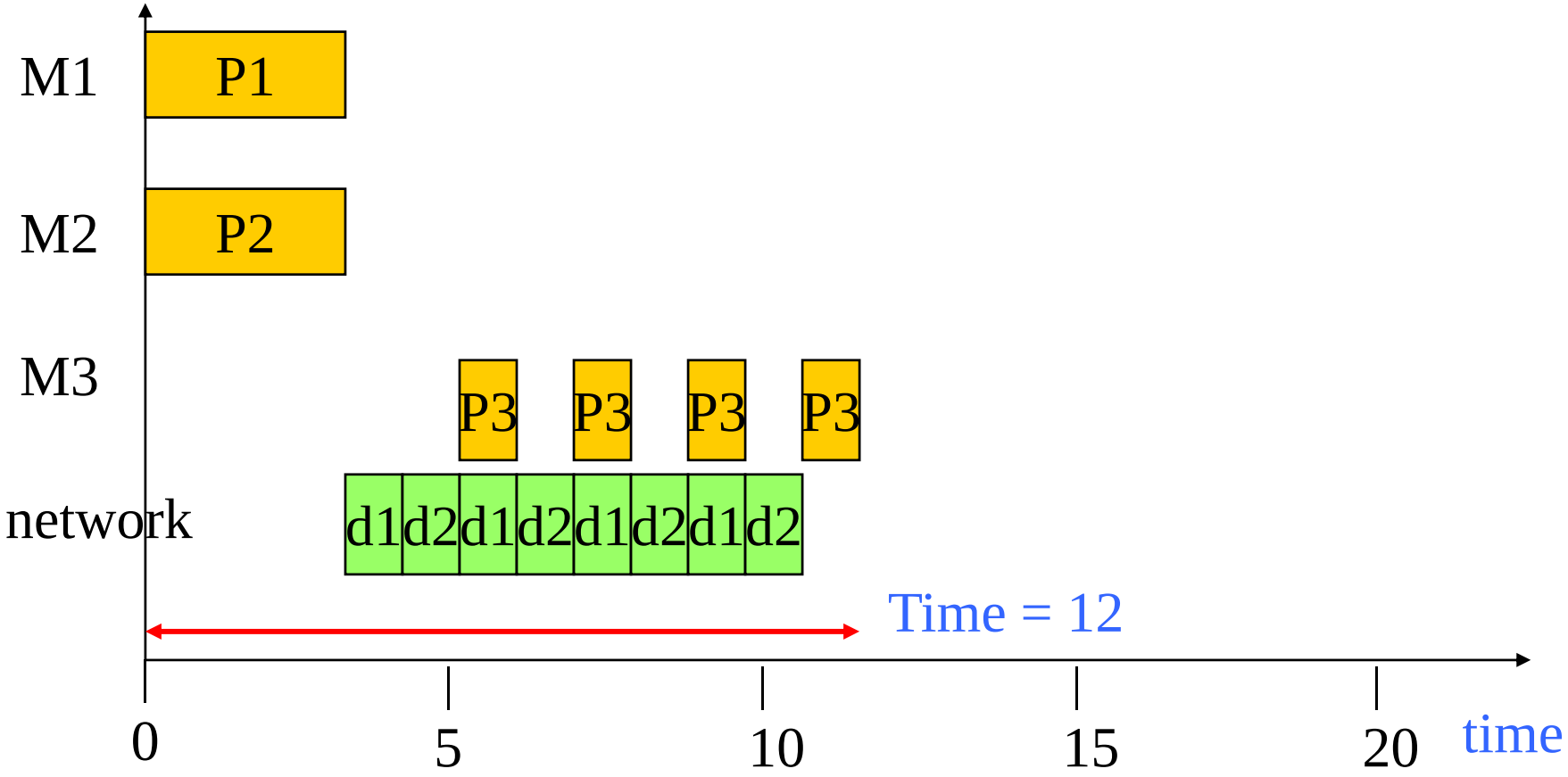
Initial schedule



New design

- Modify P3:
  - reads one packet of d1, one packet of d2
  - computes partial result
  - continues to next packet

New schedule



Buffering and performance

- Moving data in microprocessor incurs significant and sometimes unpredictable costs
- Buffering may sequentialize operations.
  - Next process must wait for data to enter buffer before it can continue.
- Buffer policy (queue, RAM) affects available parallelism.

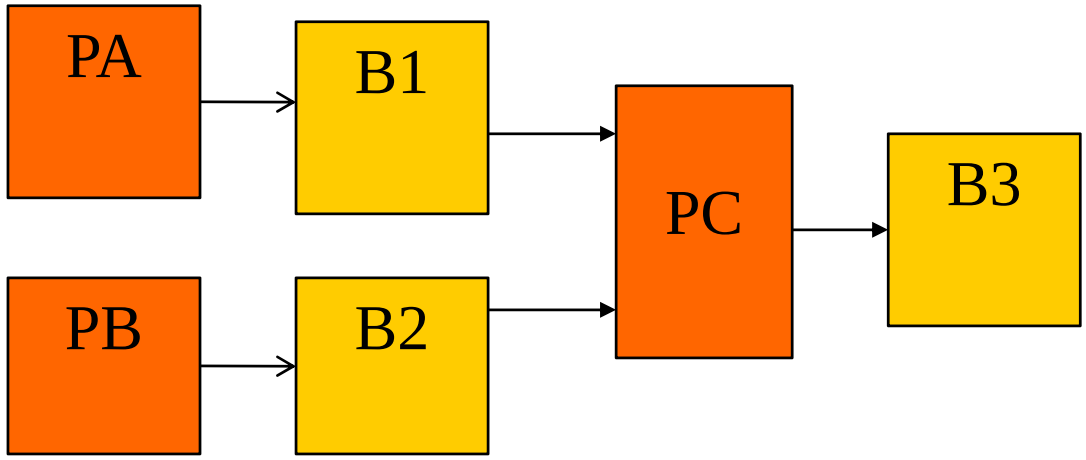
Copying an array

- Read a data from the source memory
- Transfer the data through the bus
- Write the data into the destination memory
- The transfer rate is limited by the slowest element

Buffers and latency

- Three processes separated by buffers:

PA: copying array A
PB: copying array B
PC: computing C[i]= f(A[i],B[i])



Buffers and latency schedules

(2n+1) cycle latency

A[0]
A[1]
...
B[0]
B[1]
...
C[0]
C[1]
...

Must wait for all of A before getting any B

3-cycle latency

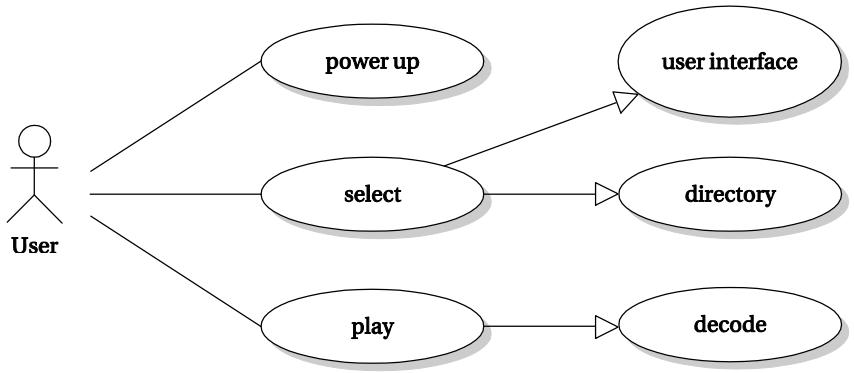
A[0]
B[0]
C[0]
A[1]
B[1]
C[1]
...

Multiprocessors

- Consumer electronics systems.
- Cell phones.
- CDs and DVDs.
- Audio players.
- Digital still cameras.

Consumer electronics use cases

- Multimedia: stored in compressed form, uncompressed on viewing.
- Data storage and management: keep track of your multimedia, etc.
- Communication: download, upload, chat.

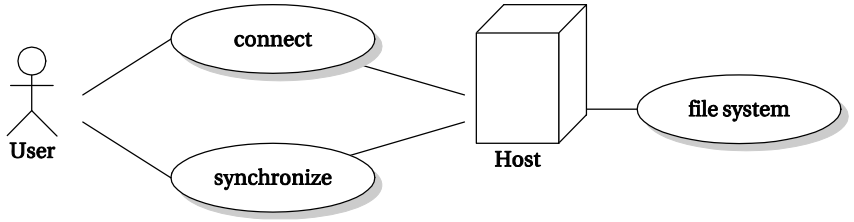


Non-functional requirements for CE

- Often battery-operated, strict power budget.,
- Very inexpensive.
- User interface must be capable but inexpensive.

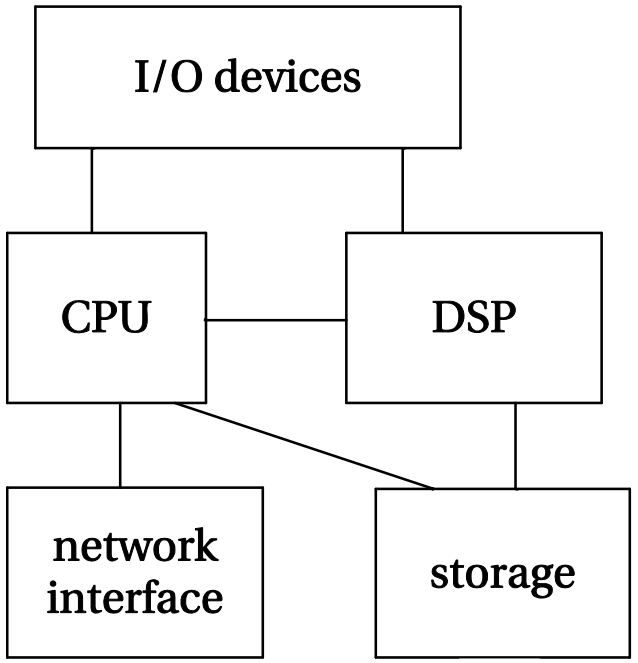
CE devices and hosts

- Many devices talk to host system.
  - PC host does things that are hard to do on the device.
- Increasingly, CE devices communicate directly over the internet, avoiding the host for access.



Platforms and OS

- Many CE devices use a DSP for signal processing and a RISC CPU for other tasks.
- OS runs on the CPU
  - maintain processes for concurrency and file systems
- I/O devices include buttons, screen, USB.

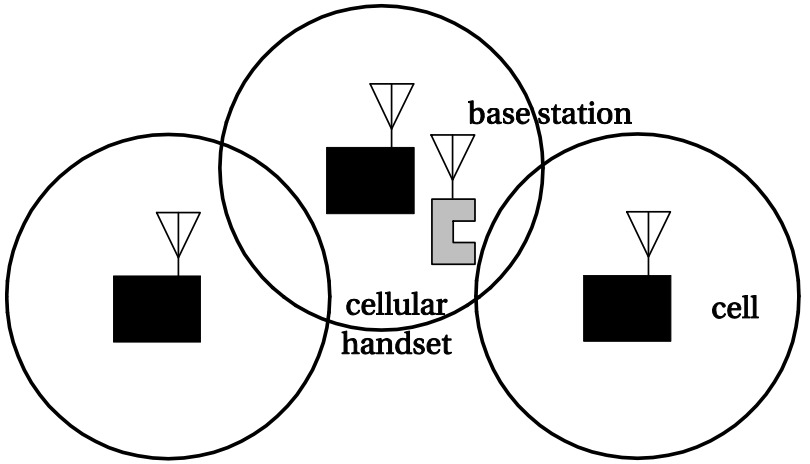


Flash memory

- Flash is widely used for mass storage.
- Flash wears out on writing (up to 1 million cycles).
  - Directory is most often written, wears out first.
- Flash file system has layer that moves contents to levelize wear.
  - Hides wear leveling from API.

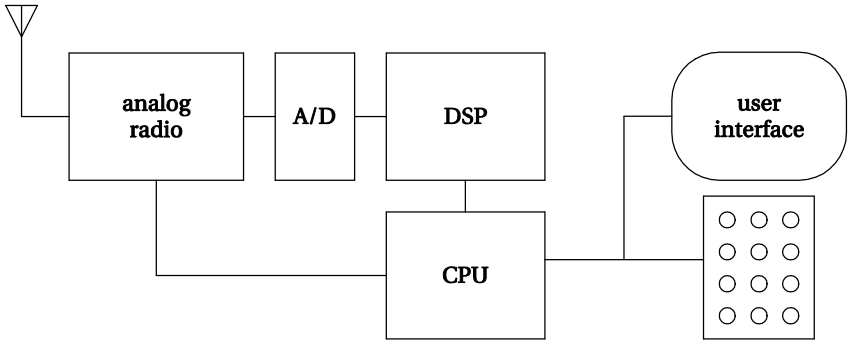
Cell phones

- Most popular CE device in history; most widely used computing device.
  - 1 billion sold per year.
- Handset talks to cell.
- Cells hand off handset as it moves.



Cell phone platforms

- Today's cell phones use analog front end, digital baseband processing.
  - Future cell phones will perform IF processing with DSP.
- Baseband processing in DSP:
  - Voice compression.
  - Network protocol.
- Other processing:
  - Multimedia functions.
  - User interface.
  - File system.
  - Applications (contacts, etc.)



Compact disc players

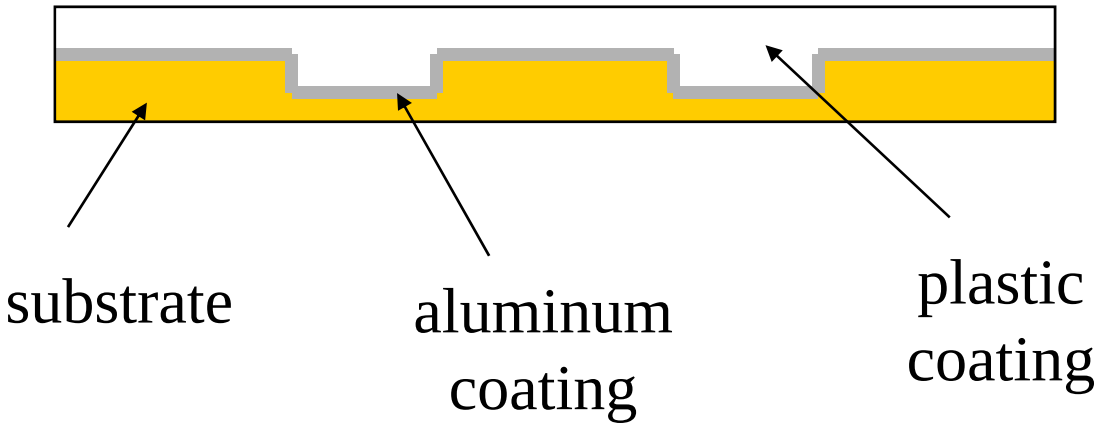
- Device characteristics.
- Hardware architectures.
- Software.

CD digital audio

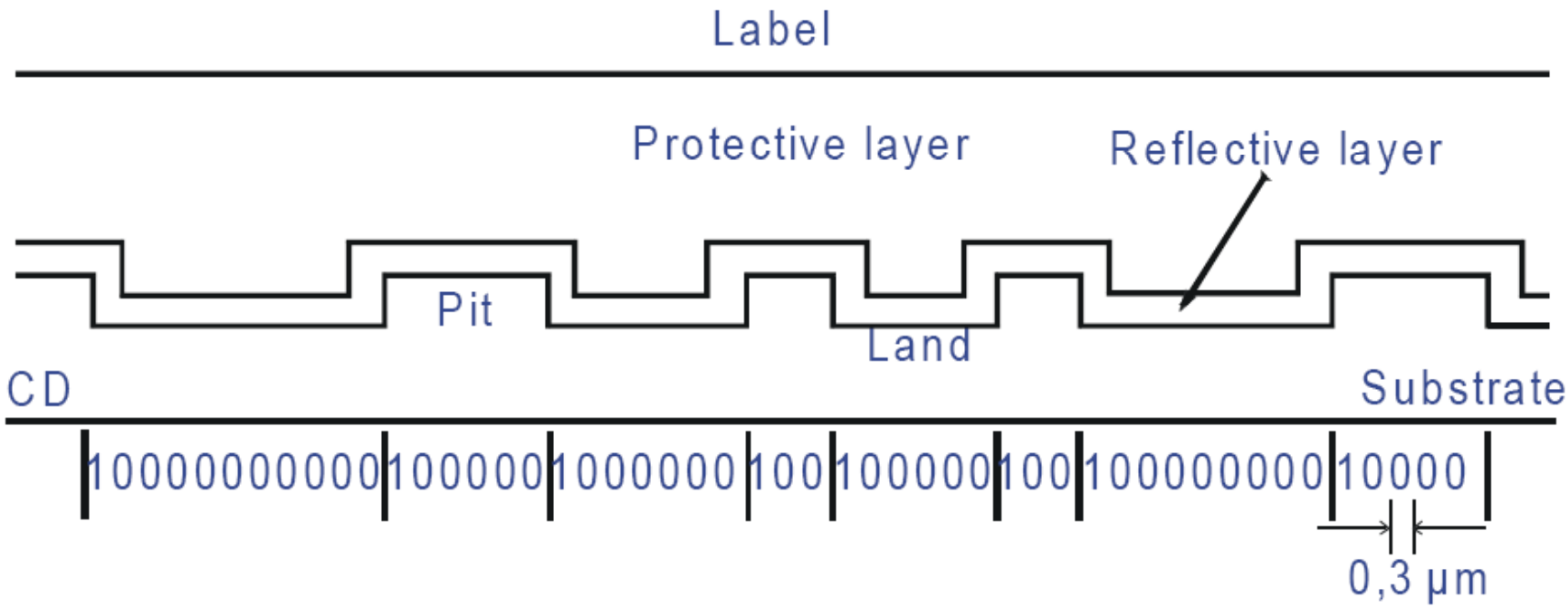
- 44.1 kHz sample rate.
- Quantization: 16 bit samples.
- Pulse coded modulation (PCM)
- Stereo (2 channels)
  - ~1.4 Mbit/s.
- Additional data tracks.
- S/N: ~ 6 dB/bit, 16 bit , 98dB

Compact disc

- Playback time : max 74 min
  - 747 Mbyte
- Data stored on bottom of disc:



CD-DA: pits and Lands

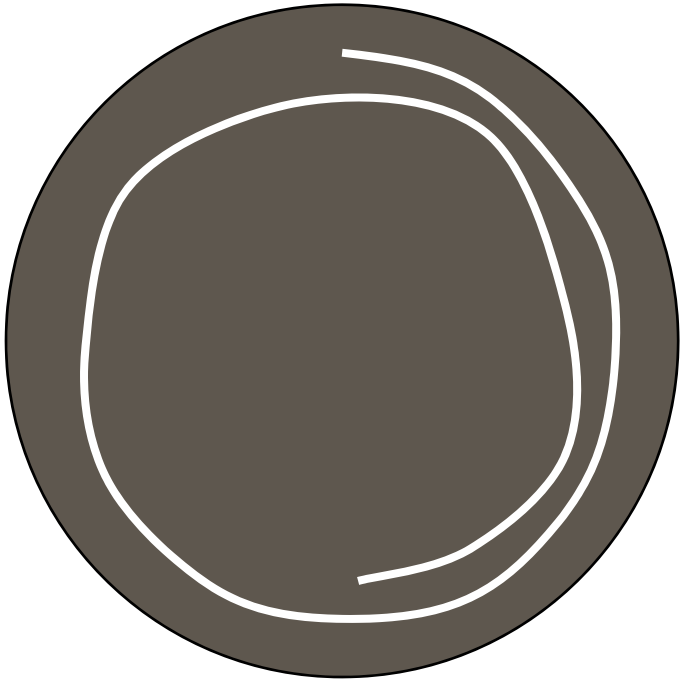


CD medium

- Rotational speed: 1.2-1.4 m/s
- Constant linear velocity (CLV).
- Track pitch: 1.6 microns.
- Diameter: 120 mm.
- Pit length: 0.8 -3 microns.
- Pit depth: .11 microns.
- Pit width: 0.5 microns.
- Laser wavelength: 780 nm.

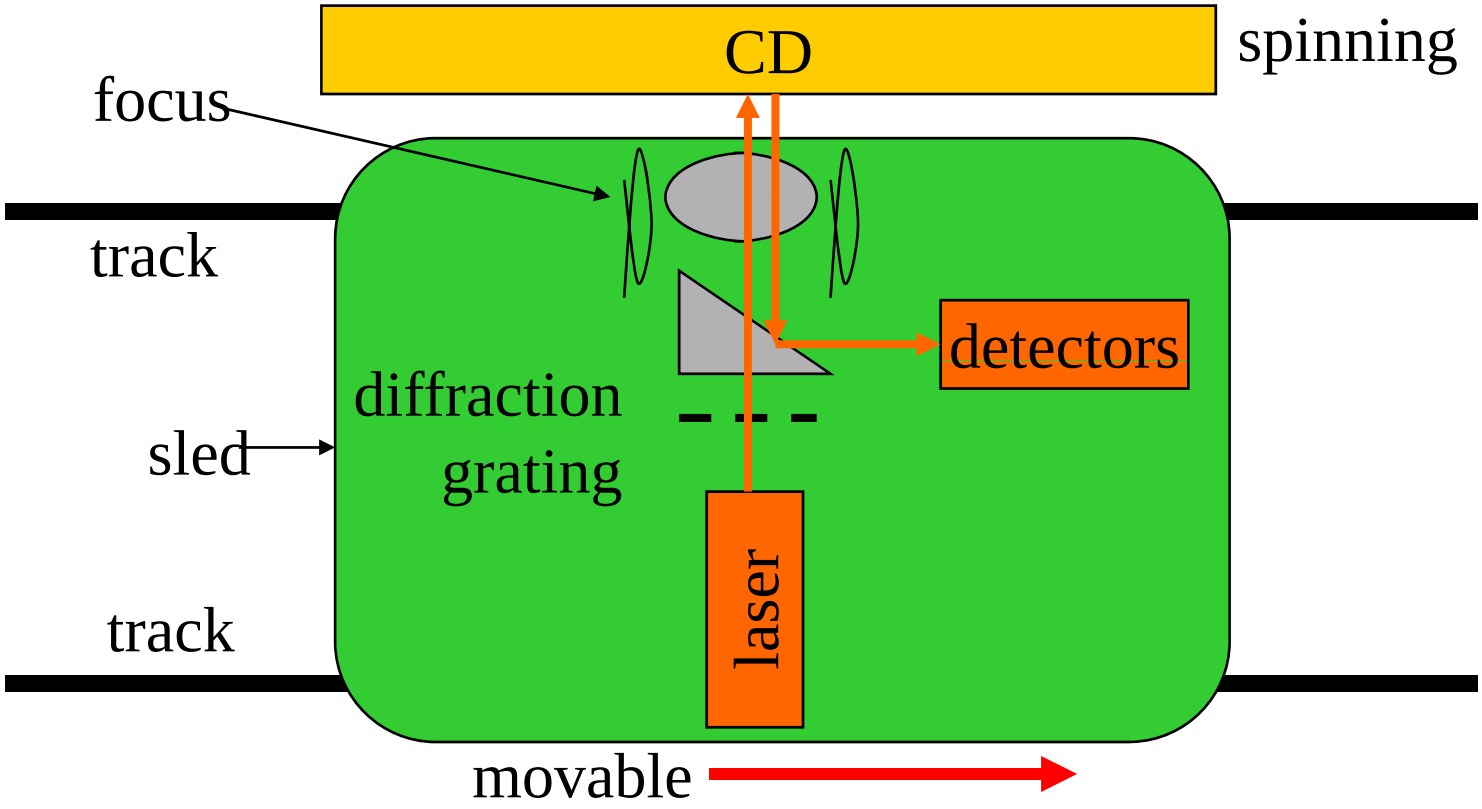
CD layout

- Data stored in spiral, not concentric circle:
- One spiral with approx. 20k turns



CD mechanism

- Laser, lens, sled:

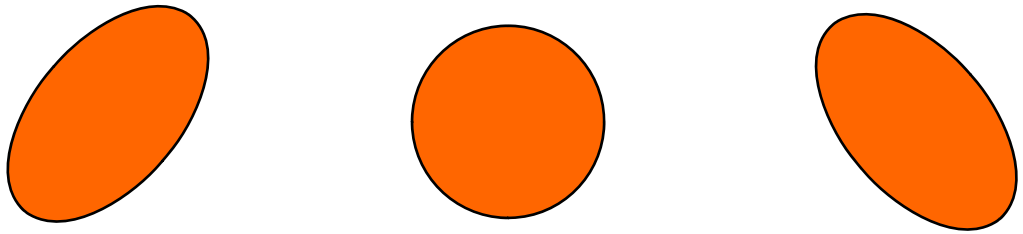


Laser pickup sled

- It is movable
- It is comprised of
  - the laser,
  - a system of lenses,
  - a photodetector, and
  - a motor which moves the sled.

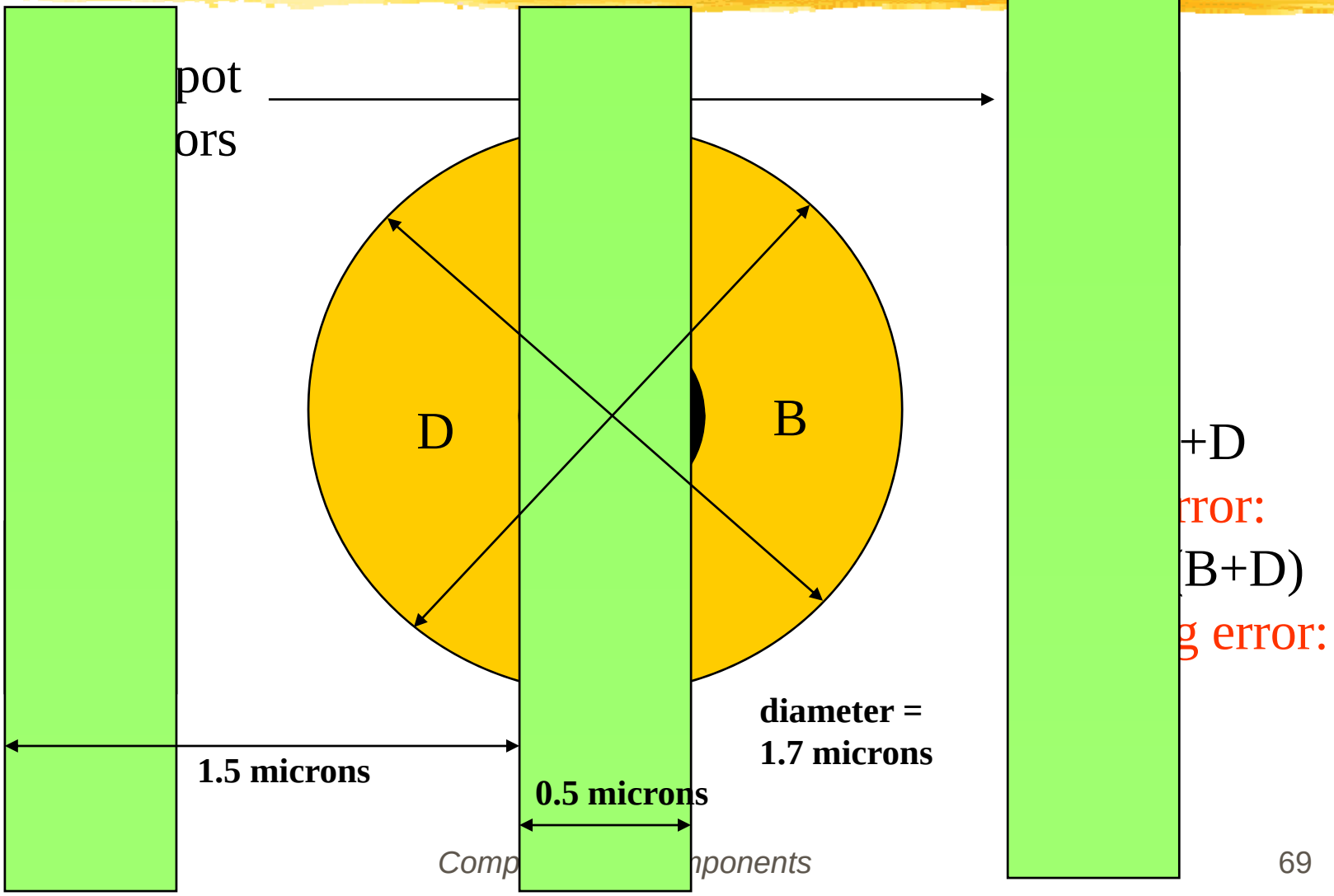
Laser focus

- Focus controlled by vertical position of lens.
- Unfocused beam causes irregular spot:



Out of focus In focus Out of focus

Laser pickup



Servo control

- Four main signals:
  - focus (laser) @ 245 kHz;
  - tracking (laser) @ 245 kHz;
  - sled (motor): @ 800 Hz;
  - Disc motor.

} Optical pickup

EFM

- Eight-to-fourteen modulation:
  - Fourteen-bit code guarantees a maximum distance between transitions to minimize the transition rate.
  - To guarantee pits of specific lengths, the CD standard requires that there are at least 2 and at most 10 zeroes between every 1.
  - The shortest possible pit (or land) thus represents 3 EFM bits (100), and the longest 11 EFM bits (10000000000)
  - 256 are chosen from 267 combinations that satisfies the constraints

EFM

Example from the code conversion table

data bits	channel bits
00000000	01001000100000
00000001	10000100000000
…	…

EFM

Concatenation of independent 14 bit values could lead to a violation of:

- minimum distance of 2 bits between Ones
- maximum distance of 10 bits between Ones
- => three additional ***merging (filling) bits***

CD-DA: Eight-to-Fourteen Modulation Example

Audio Bits	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 1
Modulation Bits	0 1 0 0 1 0 0 0 1 0 0 0 0 0	1 0 0 0 0 1 0 0 0 0 0 0 0 0
Filling Bits	0 1 0	1 0 0
Channel Bits	0 1 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0
On the CD-DA	l p p p l l l p p p p l l l l l l p p p l l l l l p p p p p p p p p

Error correction

- CD capacity: 6.99 GB raw, 700 MB formatted.
- CD interleaves Reed-Solomon blocks to reduce effects of large data gaps (scratches and other bursty errors).
- The time required to complete Reed-Solomon coding depends greatly on the number of erasure bits

Error correction

Two-level Reed-Solomon code with frame interleaving ("Cross Interleaved Reed Solomon Code"):

- **First level:** byte level, EDC and ECC. Two groups, each with four correction bytes for 24 data bytes:
  - 1st group: correction of single byte errors
  - 2nd group: correction of double byte errors, detection of further errors
- **Second level: frame interleaving**
  - frame: 588 channel bits = 24 audio data bytes
  - distribution of consecutive data bytes and corresponding ECC bytes over adjacent frames

Error correction

Error rate: 10^{-8} (~ 1 bit in 100 million bits (!)

- Exact correction of 4000 data bits possible
- 4000 data bits * 0.3 µm/channel bit
- hence: burst errors within 2.5 mm can be corrected

With interpolation: Up to 12,300 data bits (~ 7 mm)

CIRC encoding

- Cross-interleaved Reed-Solomon coding.
  - Interleaves to reduce burst errors.
- Each 16-bit sample split into two 8-bit symbols.
  - Pulse coded modulation (PCM)
- Sample split into two symbols.
- Six samples from each channel (=24 symbols=192 bits) are chosen to make a frame, which is encoded as a 224 bits.
- It will be eventually encoded as 588 bits

CIRC encoding

Each frame consists of

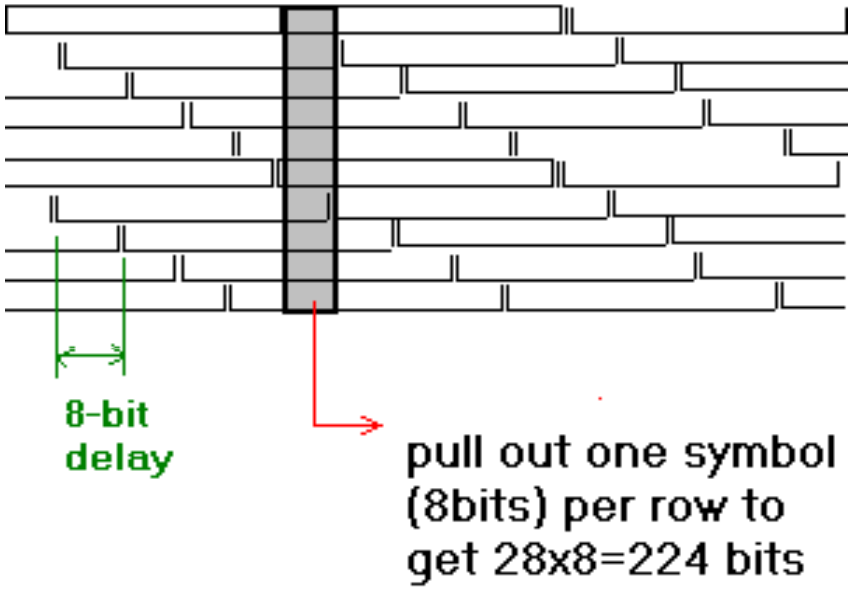
- **Data**
  - two groups of 12 audio data bytes each (actual data)
- **Error detection and correction code**
  - two groups of four parity bytes
  - Computed according to the Reed-Solomon code
- **Control&display byte**
  - Together with control&display bytes of other frames it forms the subchannel stream.
  - Example: subchannel byte for track start identification
- **Synchronization pattern**
  - Start of a frame
  - 12 x “1” + 12 x “0” + 3 merging bits = 27 bits

Encoding – the numbers

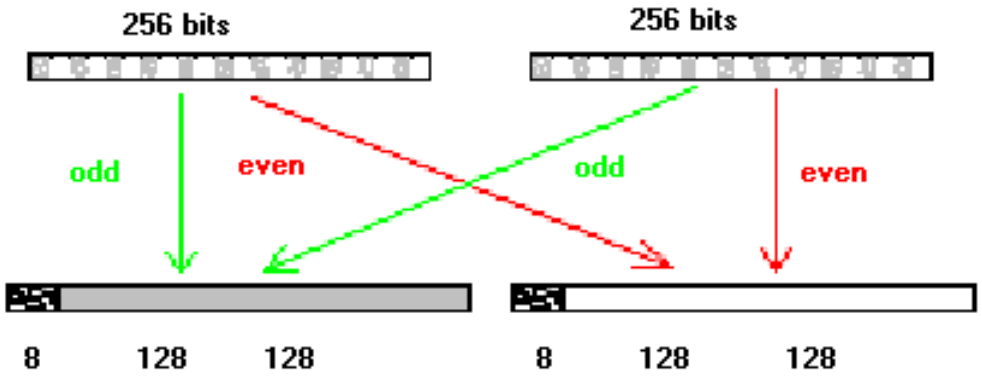
- The codewords from the first code are interleaved into a virtually infinite array of 28 rows of symbols over GF(256).
- We pull out 8 binary columns (one symbol) to obtain a 28x8=224-bit frame which is then encoded using another Reed-Solomon code to obtain a codeword of length 256 bits.

Interleaving to disperse errors

- Codewords of first code are stacked like bricks
- 28 rows of vectors over GF(256)
- Extract columns and re-encode using second Reed-Solomon code



Splitting Odd and Even Bits



Further Processing

- So 256+8=264=33x8 bits, carrying six samples, or 192 information bits, gets encoded as 588 channel bits on the disk
- This represents 0.000136 seconds of music

What actually goes on the disc?

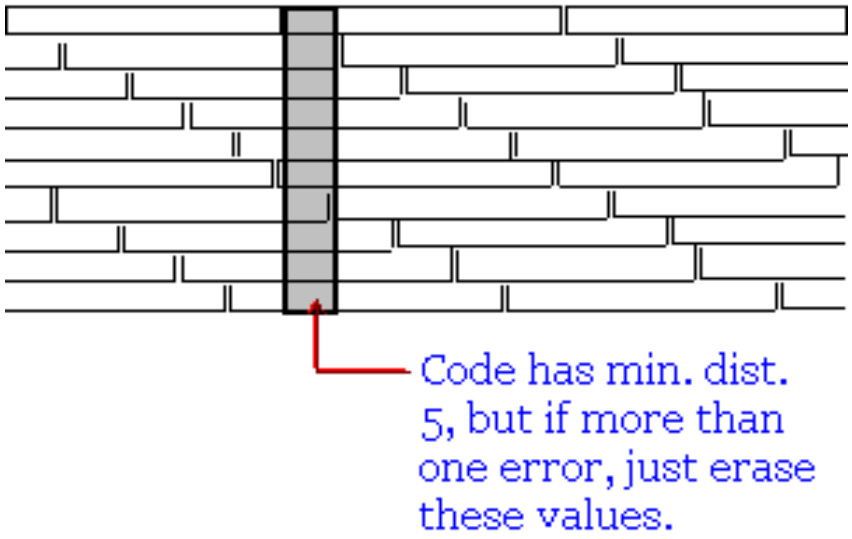
- We must do this 7,350 times per second
- So CD player reads 4,321,800 bits per second of music produced
- To get 74 minutes of music, we must store

74x60x4321800 = 19,188,792,000

bits of data on the compact disc!

When in doubt, erase

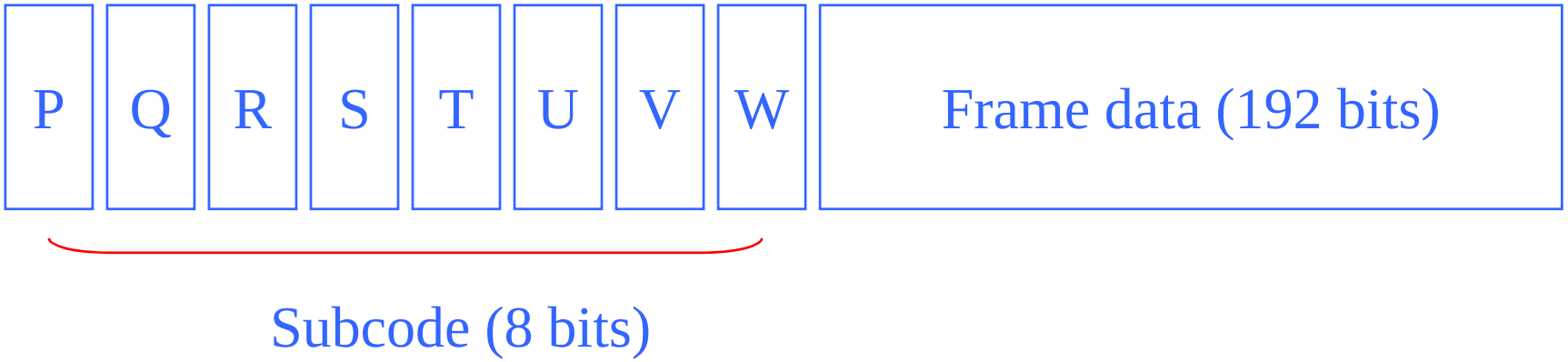
- Inner code has minimum distance 5 (over GF(256))
- Rather than correct two-symbol errors, the CD just erases the entire received vector.



So...how good is it?

- The two Reed-Solomon codes team up to **correct** 'burst' errors of up to 4000 consecutive data bits (2.5 mm scratch on disc)
- If signal at time t cannot be recovered, **interpolate**
- With smart data distribution, this allows for recovery from burst errors of up to 12,000 data bits (7.5 mm track length on disc)
- If all else fails, **mute**, giving 0.00028 sec of silence.

CD frame



Control word

- 8-bit control word for every 32-symbol block:
  - P: 1 during music/lead-in, 0 at start of selection.
  - Q: track number, time, etc (spread over 98 frames).
  - R, S, T, U, V, W: reserved.

Control and error correction

- Skips caused by physical disturbance.
  - Wait for disturbance to subside.
  - Retry.
- Read errors caused by disc/servo problems.
  - Detect error.
  - Choose location for retry.
  - Retry.
  - Fail and interpolate.

Retry problems

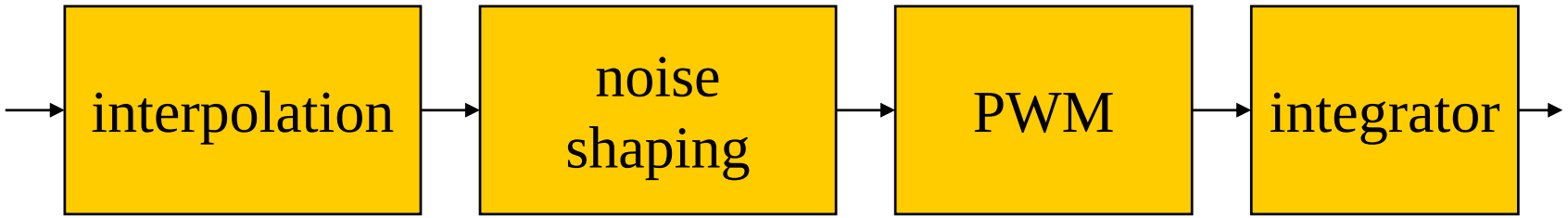
- Data is stored in a spiral.
  - Can’t seek track as on magnetic disc.
  - Sled servo is very coarse.
- Data is only weakly addressed.
  - Must read data to know where to go.

Audio playback

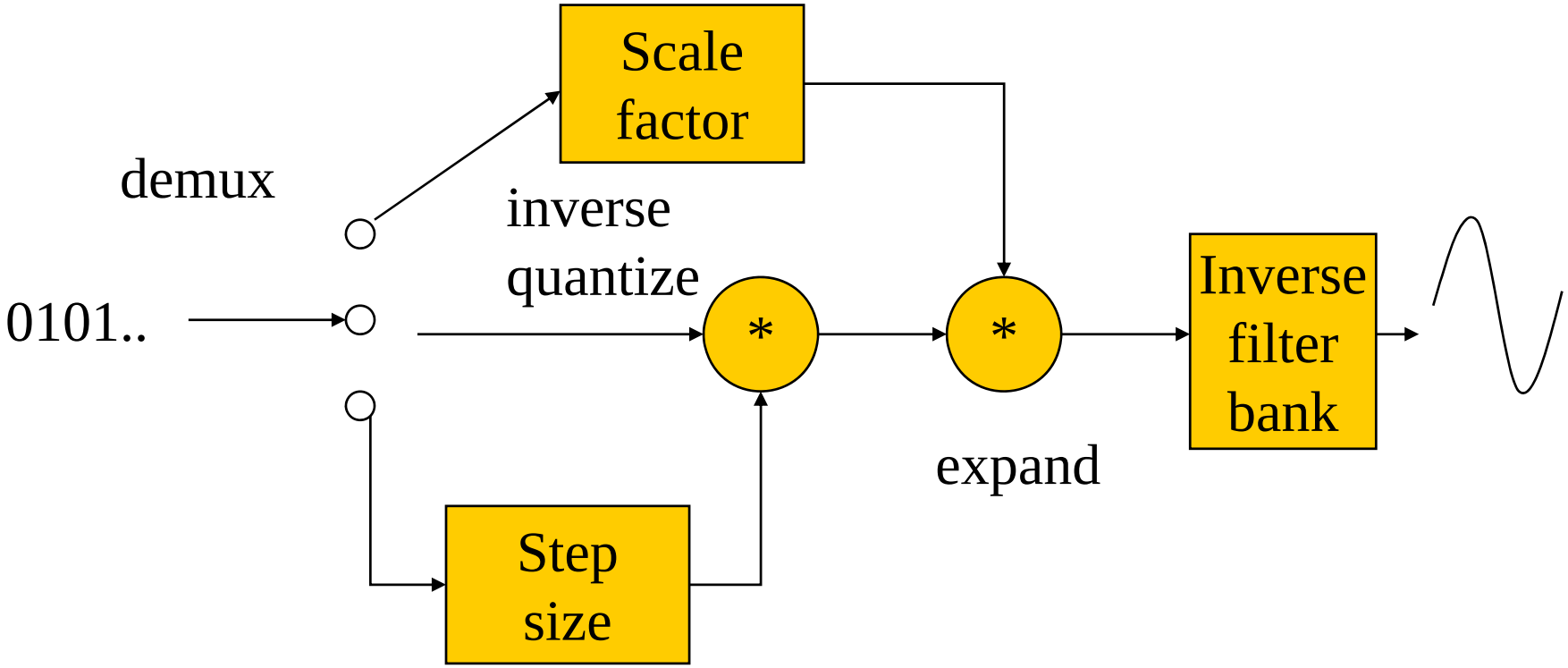
- Audio CD needs no audio processing.
- Tasks:
  - convert to analog;
  - amplify.

Digital/analog conversion

- 1-bit MASH conversion:



MPEG Layer 1 decoder



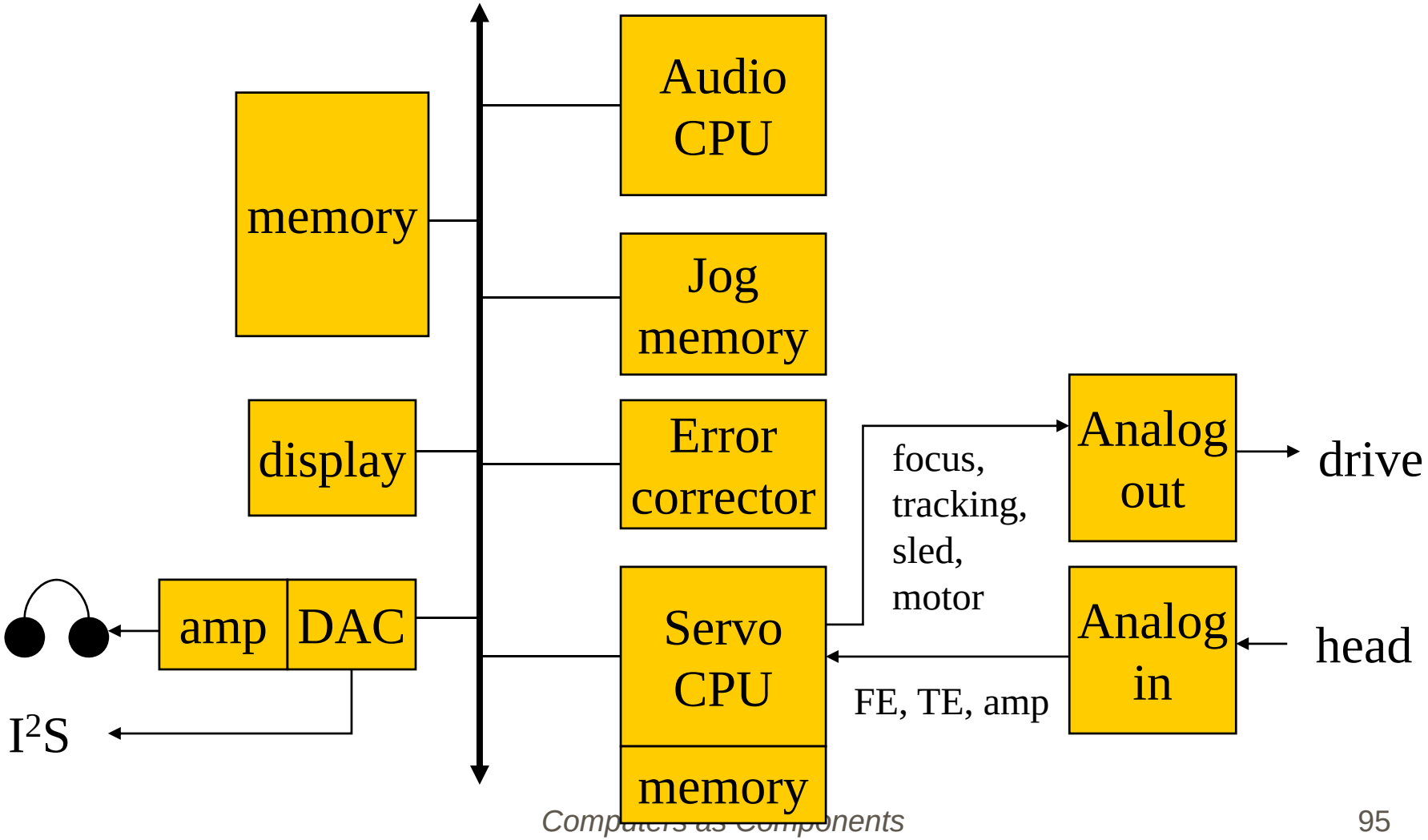
MP3

- Decoding is easier than encoding, but requires:
  - decompression;
  - filtering.
- Basic CD standard for data discs.
- No standards for MP3 disc file structure: player must understand Windows, Mac, Unix discs.

Jog/skip memory

- Read samples into RAM, play back from RAM.
- Modern RAMs are larger than needed for reasonable jog/skip.
- Jog memory saves some power.

CD/MP3 player



DVD format

- Similar to CD, but:
  - shorter wavelength laser;
  - tighter pits;
  - two layers of data.

Audio on DVD

- Alternatives:
  - MP3 on data DVD (stereo).
  - Audio track of video DVD (5.1).
  - DVD audio (5.1).
  - SACD (5.1).

MPEG audio standards

- Layer 1:
  - Lossless compression of subbands + optional simple masking model
- Layer 2:
  - More advanced masking model.
- Layer 3:
  - Additional processing for lower bit rates.

MPEG audio rates

- Input sampling rates:
  - 32, 44.1, 48 kHz.
- Output bit rates:
  - 23, 48, 64, 96, 112, 128, 192, 256, 384 kbits/sec.
- Output can be mono, dual-channel (bilingual, etc.), stereo.

Other standards

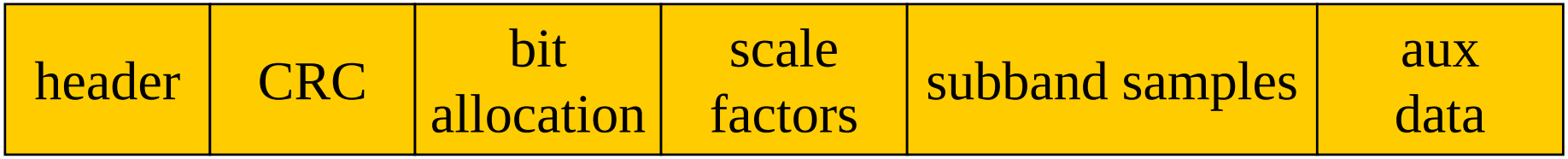
- Dolby Digital (AC-3):
  - Uses modified discrete cosine transform.
- ATRAC (MiniDisc):
  - Uses subband + modified DCT.
- MPEG-2 AAC.

MPEG Layer 1

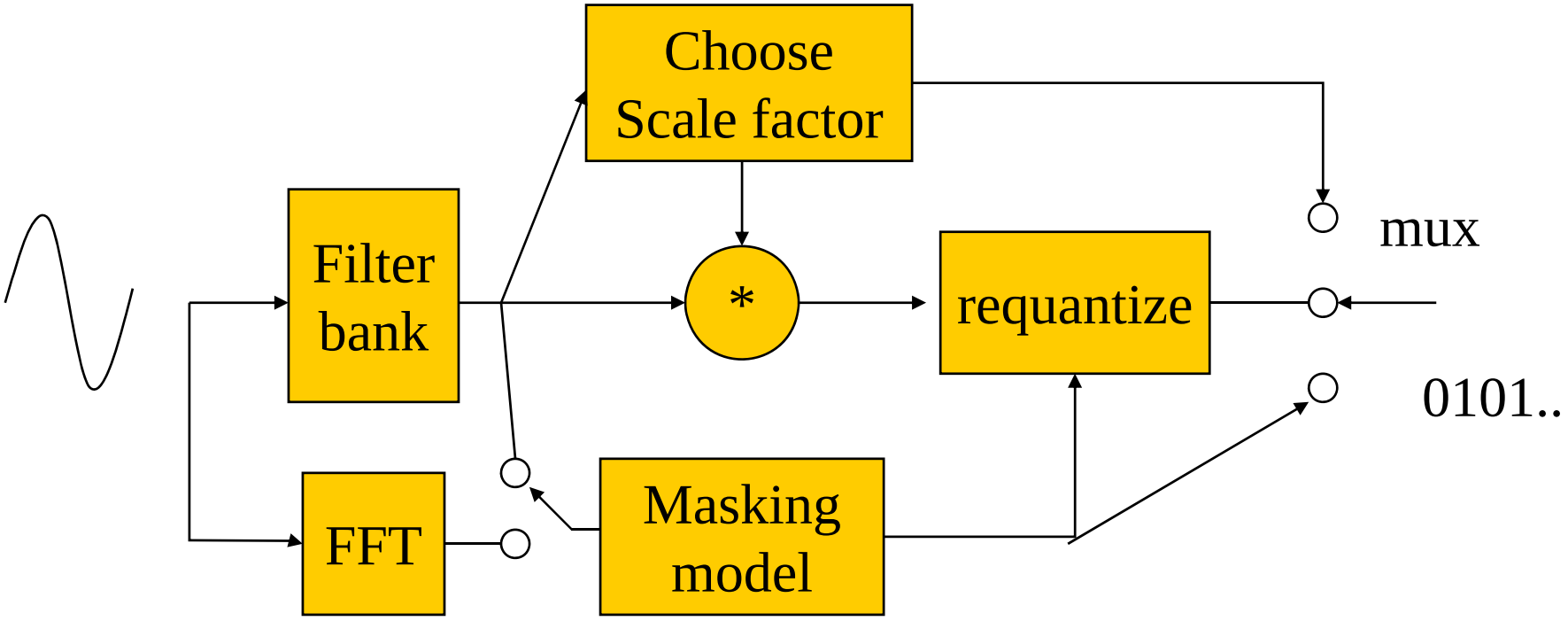
- 384 samples/block at all frequencies.
  - Equals 8 ms at 48 kHz.
- Optional masking model.
  - Driven by separate FFT for better accuracy.

MPEG Layer 1 data frame

- Bit allocation codes specify word length in each subband.
- Scale factors give gain for each band.

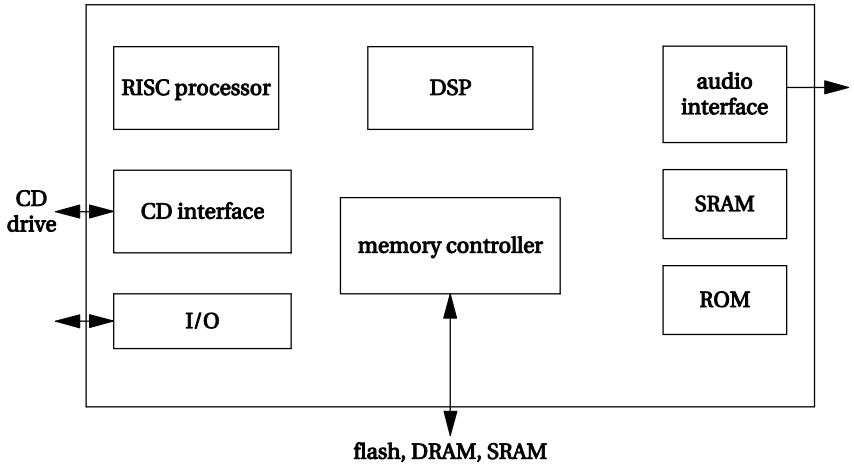


MPEG Layer 1 encoder



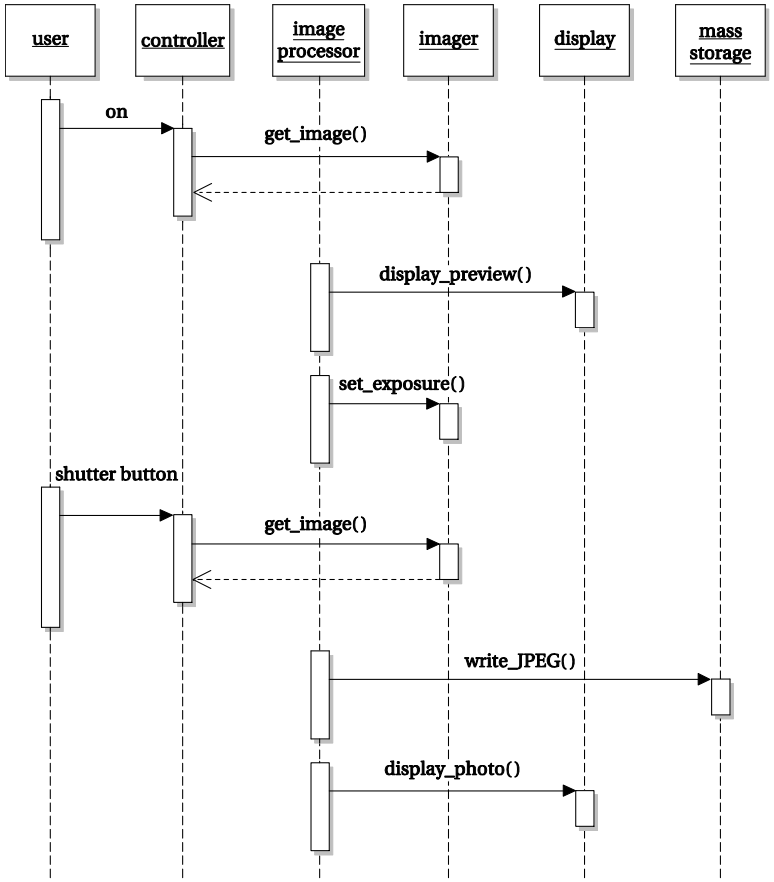
Audio players

- Audio players may use flash, hard disk, or CD for mass storage.
- Decompression requires small amount of CPU:
  - 10% of ARM7.
- File system must be compatible (FAT).



Digital still cameras

- DSC must determine exposure before taking picture.
- After taking picture:
  - Improve image quality.
  - Compress.
  - Save as file.



Digital still camera architecture

- DSC uses CPU for general-purpose processing, DSP for image processing.
- Internal memory buffers the passes on the image.
- Display is lower resolution than image sensor.
  - Image must be downsampled.

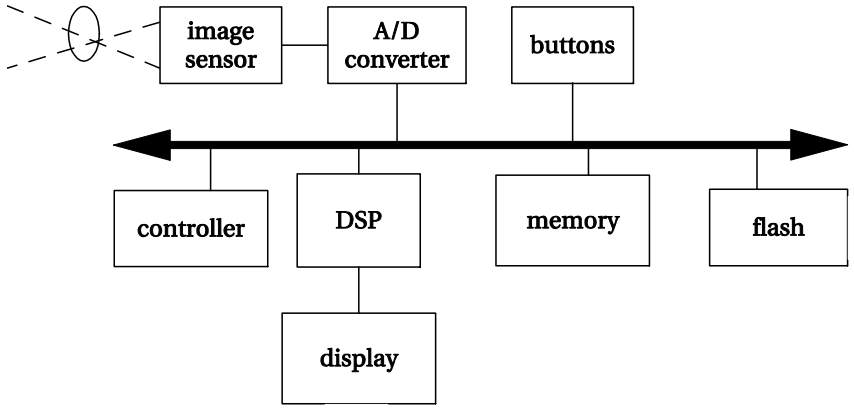
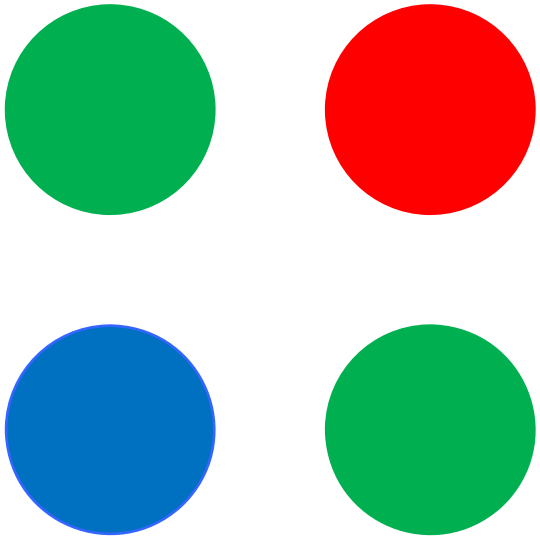


Image capture

- Before taking picture:
  - Determine exposure.
  - Determine focus.
  - Optimize white balance.



Bayer pattern

Image processing

- Must perform basic processing to get usable picture:
  - Bayer->RGB interpolation.
- DSCs perform many functions formerly performed by photoprocessors for film:
  - Image sharpening.
  - Color balance.

File management

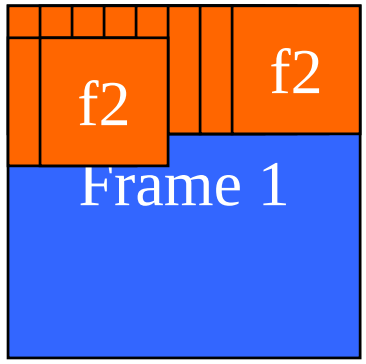
- EXIF standard gives format for digital pictures:
  - Format of data in a file.
  - Directory structure.
- EXIF file includes:
  - Image (JPEG, etc.)
  - Thumbnail.
  - Metadata (camera type, date/time, etc.)

Accelerators

- Example: video accelerator

Concept

- Build accelerator for block motion estimation, one step in video compression.
- Perform two-dimensional correlation:

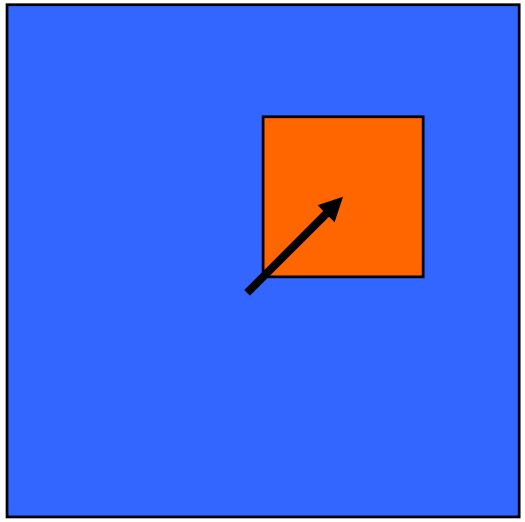


Block motion estimation

- MPEG divides frame into 16 x 16 macroblocks for motion estimation.
- Search for best match within a search range.
- Measure similarity with sum-of-absolute-differences (SAD):
  - $\Sigma \mid M(i,j) - S(i-o_x, j-o_y) \mid$

Best match

- Best match produces motion vector for motion block:



Full search algorithm

```
bestx = 0; besty = 0;
bestsad = MAXSAD;
for (ox = - SEARCHSIZE; ox < SEARCHSIZE; ox++) {
   for (oy = -SEARCHSIZE; oy < SEARCHSIZE; oy++) {
        int result = 0;
        for (i=0; i<MBSIZE; i++) {
                for (j=0; j<MBSIZE; j++) {
                        result += iabs(mb[i][j] - search[i-
ox+XCENTER][j-oy-YCENTER]);
```

Full search algorithm, cont'd.

```
                }
            }
            if (result <= bestsad) { bestsad = result; bestx =
    ox; besty = oy; }
        }
}
```

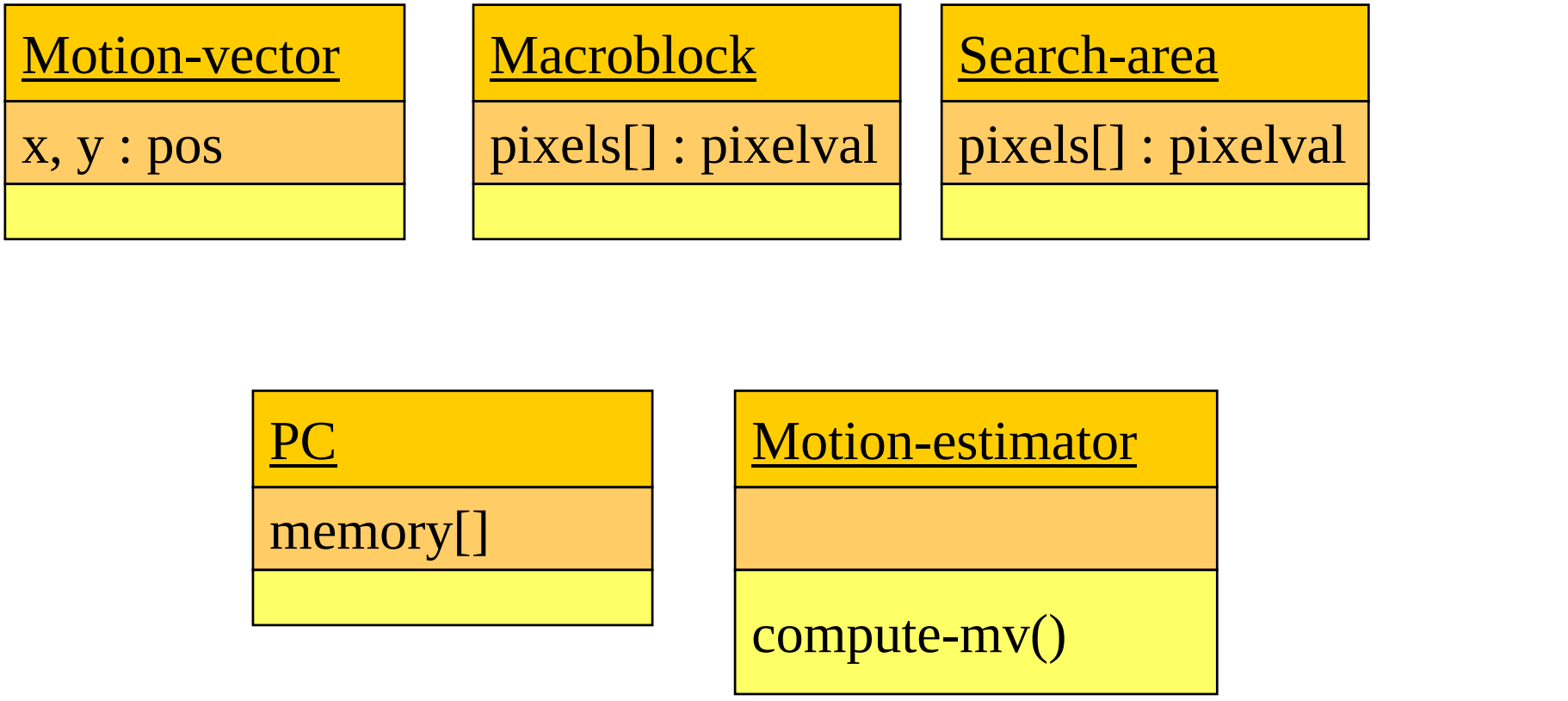
Computational requirements

- Let MBSIZE = 16, SEARCHSIZE = 8.
- Search area is 8 + 8 + 1 in each dimension.
- Must perform:
  - n_{ops} = (16 x 16) x (17 x 17) = 73984 ops
- CIF format has 352 x 288 pixels -> 22 x 18 macroblocks.

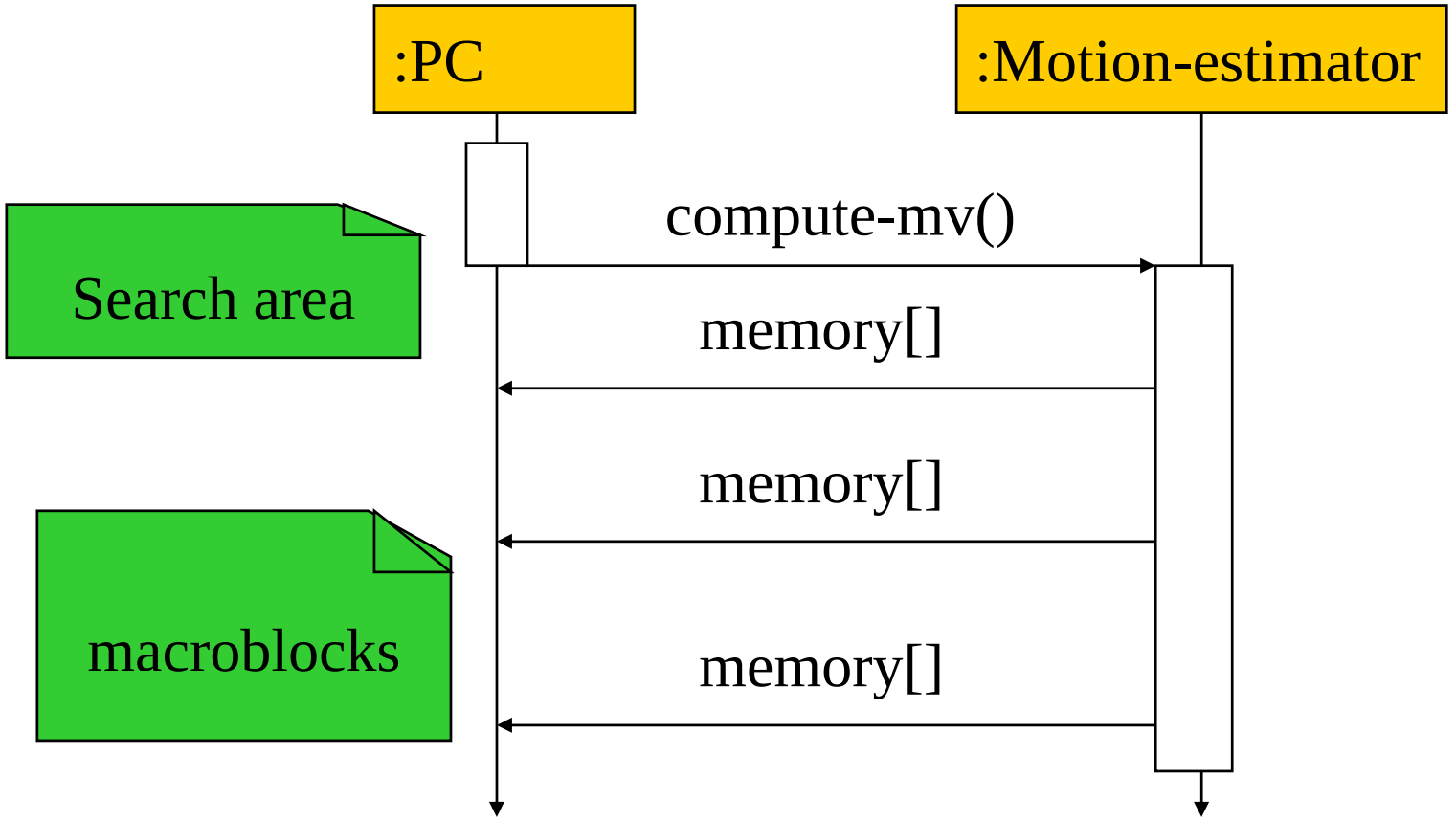
Accelerator requirements

name	block motion estimator
purpose	block motion est. in PC
inputs	macroblocks, search areas
outputs	motion vectors
functions	compute motion vectors with full search
performance	as fast as possible
manufacturing cost	hundreds of dollars
power	from PC power supply
physical size/weight	PCI card

Accelerator data types, basic classes



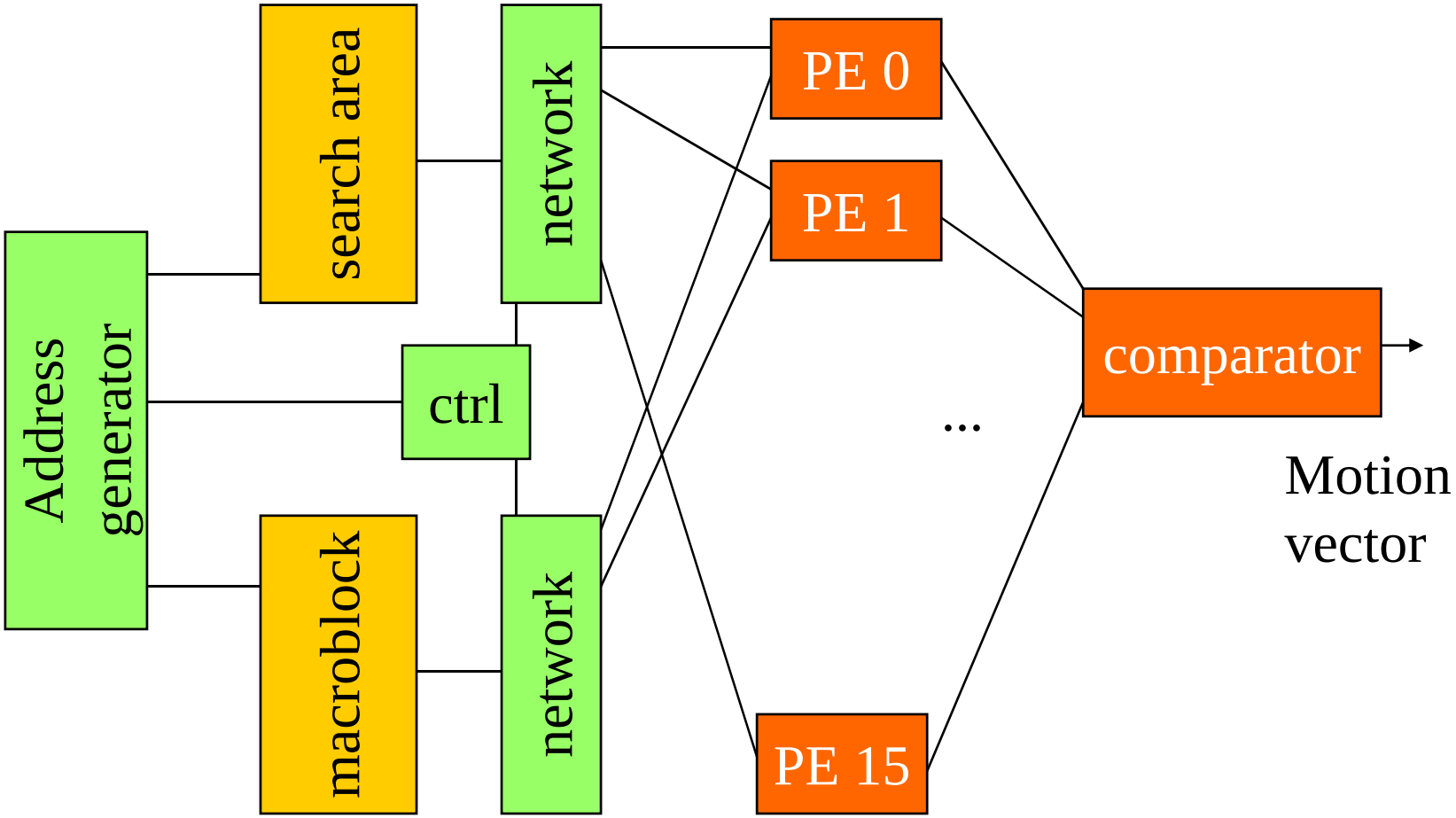
Sequence diagram



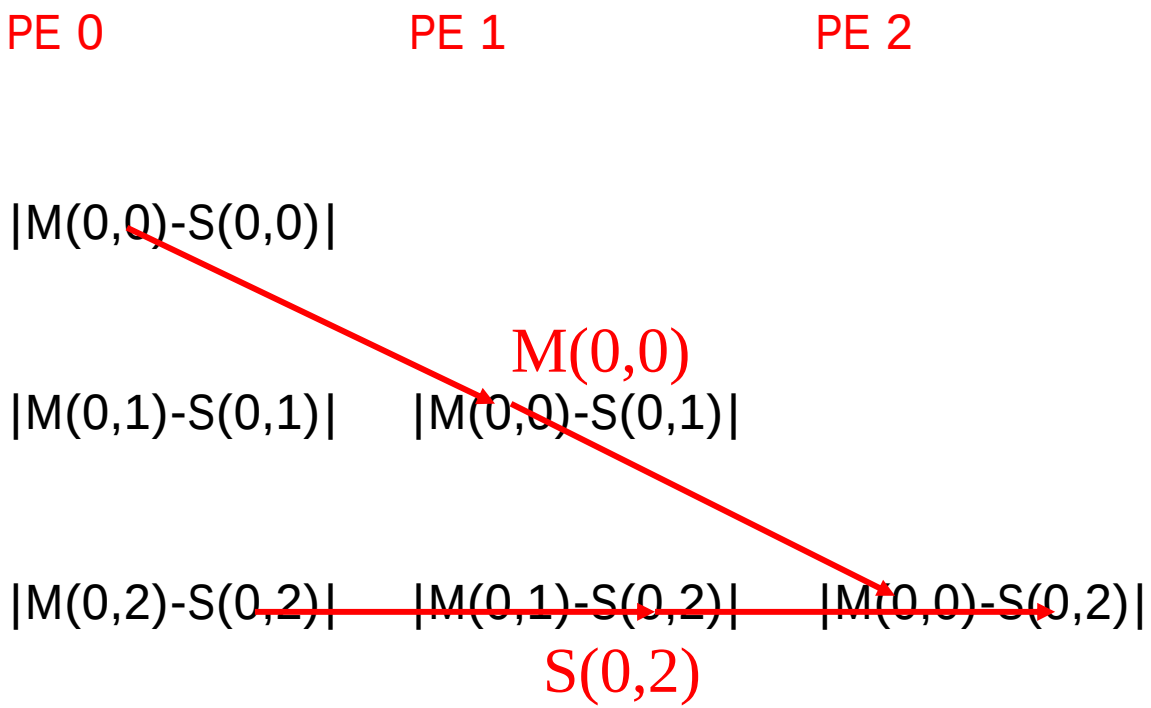
Architectural considerations

- Requires large amount of memory:
  - macroblock has 256 pixels;
  - search area has 1,089 pixels.
- May need external memory (especially if buffering multiple macroblocks/search areas).

Motion estimator organization



Pixel schedules



System testing

- Testing requires a large amount of data.
- Use simple patterns with obvious answers for initial tests.
- Extract sample data from JPEG pictures for more realistic tests.