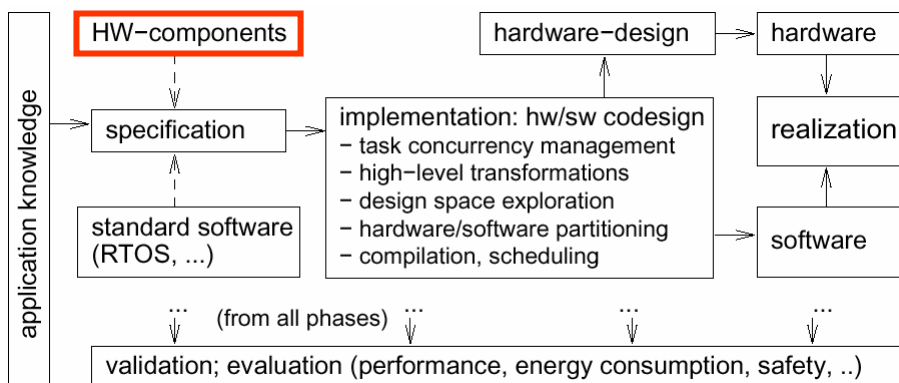


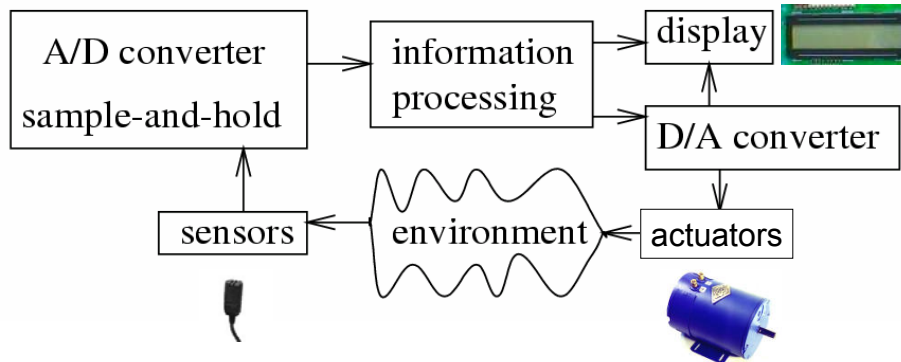


Overview of embedded systems design



Embedded System Hardware

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



BF - ES

- 3 -

Many examples of such loops

- Heating
- Lights
- Engine control
- Power supply
- ...
- Robots



BF - ES

Heating: www.masonsplumbing.co.uk/images/heating.jpg
Robot: Courtesy and ©: H.Ulbrich, F. Pfeiffer, TU München

- 4 -

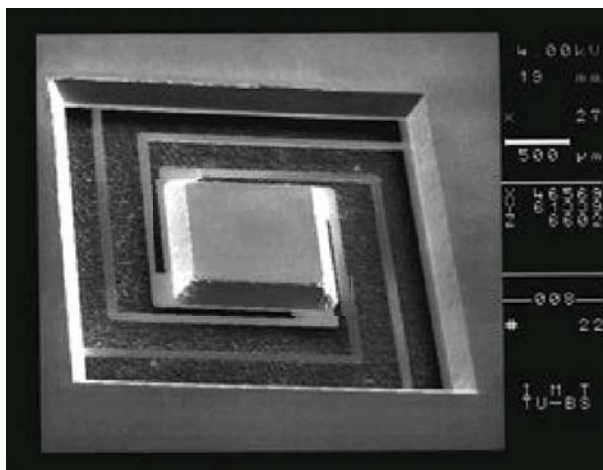
Sensors

- Processing of physical data starts with capturing this data.
- Sensors can be designed for virtually every physical and
- chemical quantity
 - including weight, velocity, acceleration, electrical current, voltage, temperatures etc.
 - chemical compounds.
- Many physical effects used for constructing sensors.
- Examples:
 - law of induction (generation of voltages in an electric field),
 - light-electric effects.
- Huge amount of sensors designed in recent years.

BF - ES

- 5 -

Example: Acceleration Sensor



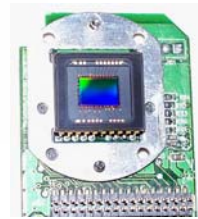
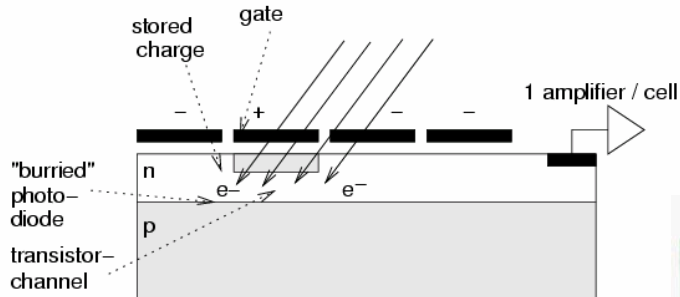
Courtesy & ©: S. Bütgenbach, TU Braunschweig

BF - ES

- 6 -

Charge-coupled devices (CCD) image sensors

Based on charge transfer to next pixel cell

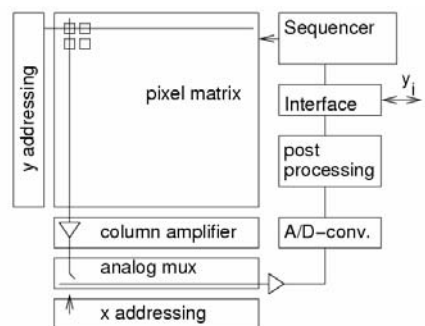
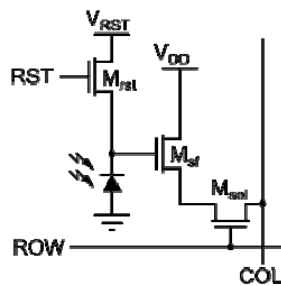


- Mature technology
- Medium to high-end compact digital cameras

BF - ES

- 7 -

CMOS image sensors

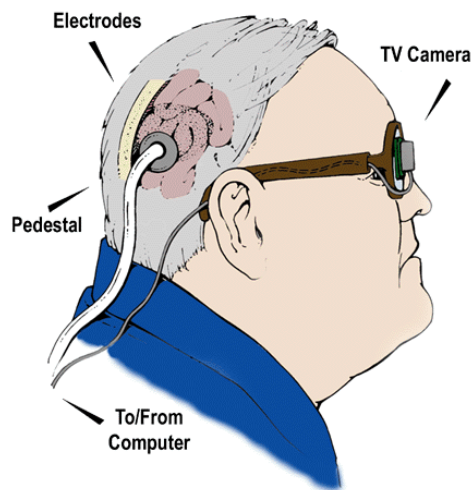


- Lower power consumption
- Lower cost
- Based on standard production process for CMOS chips, allows integration with other components.
- low cost devices
- Automotive
- medical

BF - ES

- 8 -

Artificial eyes

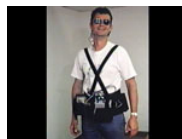
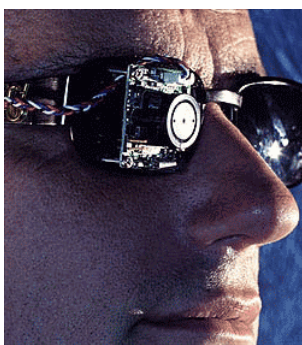


© Dobelle Institute

BF - ES

- 9 -

Artificial eyes (2)



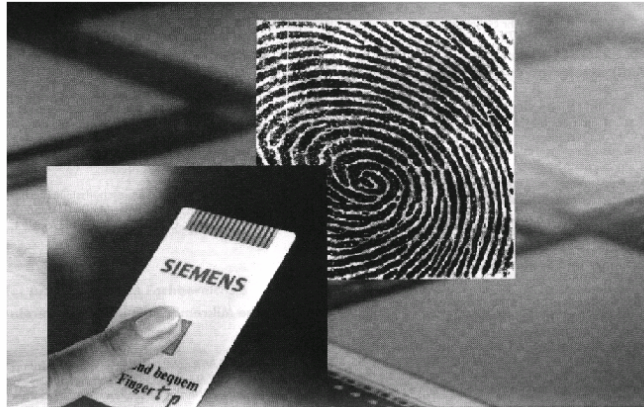
© Dobelle Institute

BF - ES

- 10 -

Example: Biometrical Sensors

Example: Fingerprint sensor (© Siemens, VDE):



Matrix of 256 x 256 elem.
Voltage ~ distance.
Resistance also computed. No fooling by photos and wax copies.
Carbon dust?

Integrated into ID mouse.

BF - ES

- 11 -

Other sensors

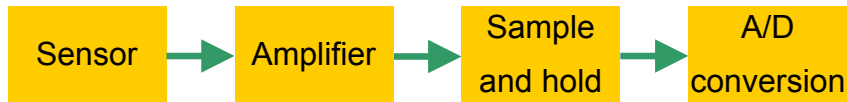
- Rain sensors for wiper control („Sensors multiply like rabbits“ [ITT automotive])
- Pressure sensors
- Proximity sensors
- Engine control sensors
- Hall effect sensors



BF - ES

- 12 -

Standard layout of sensor systems for contin. entities



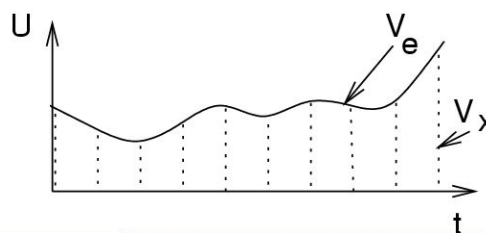
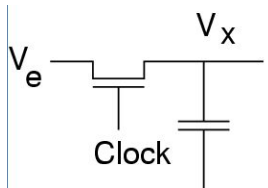
- Sensor: detects/measures entity and converts it to electrical domain
 - May entail ES-controllable actuation: e.g. charge transfer in CCD
- Amplifier: adjusts signal to the dynamic range of the A/D conversion
 - Often dynamically adjustable gain: e.g. ISO settings at digital cameras, input gain for microphones (sound or ultrasound), extremely wide dynamic ranges in seismic data logging
- Sample + hold: samples signal at discrete time instants
- A/D conversion: converts samples to digital domain

BF - ES

- 13 -

Discretization of time

V_e is a mapping $\mathbb{R} \rightarrow \mathbb{R}$



V_x is a **sequence** of values or a mapping $\mathbb{Z} \rightarrow \mathbb{R}$

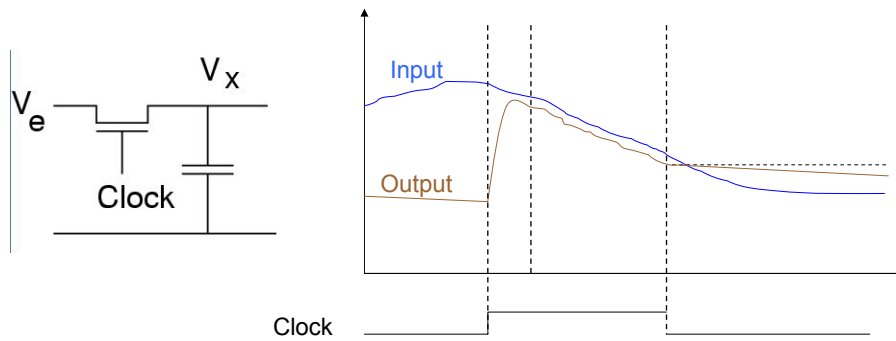
Discrete time: sample and hold-devices.

Ideally: width of clock pulse $\rightarrow 0$

BF - ES

- 14 -

Sample and Hold



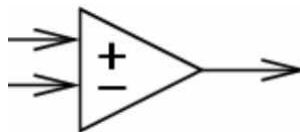
BF - ES

- 15 -

Discretization of values: A/D-converters

1. Flash A/D converter (1)

- Basic element: analog comparator



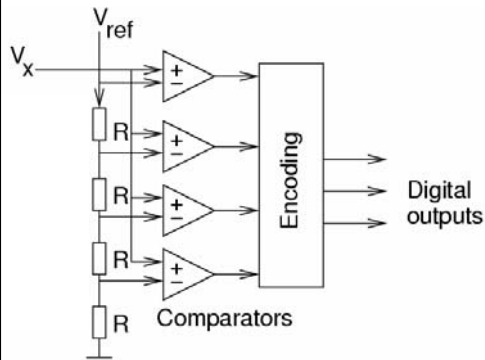
- Output = '1' if voltage at input + exceeds that at input -.
- Output = '0' if voltage at input - exceeds that at input +.
- Idea:
 - Generate n different voltages by voltage divider (resistors), e.g. V_{ref} , $\frac{3}{4} V_{ref}$, $\frac{1}{2} V_{ref}$, $\frac{1}{4} V_{ref}$.
 - Use n comparators for parallel comparison of input voltage V_x to these voltages.
 - Encoder to compute digital output.

BF - ES

- 16 -

Discretization of values: A/D-converters

1. Flash A/D converter (2)



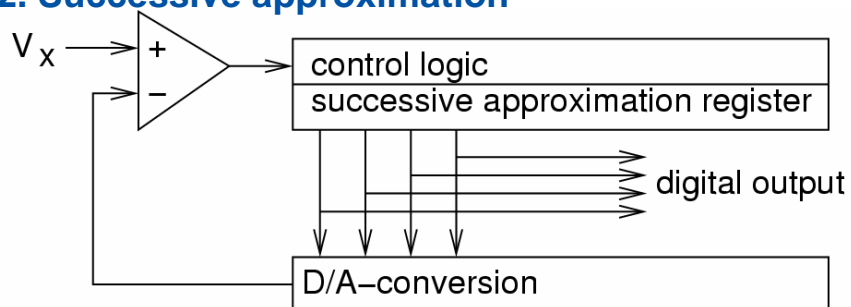
- Parallel comparison with reference voltage
- **Applications:** e.g. in video processing

BF - ES

- 17 -

Discretization of values

2. Successive approximation



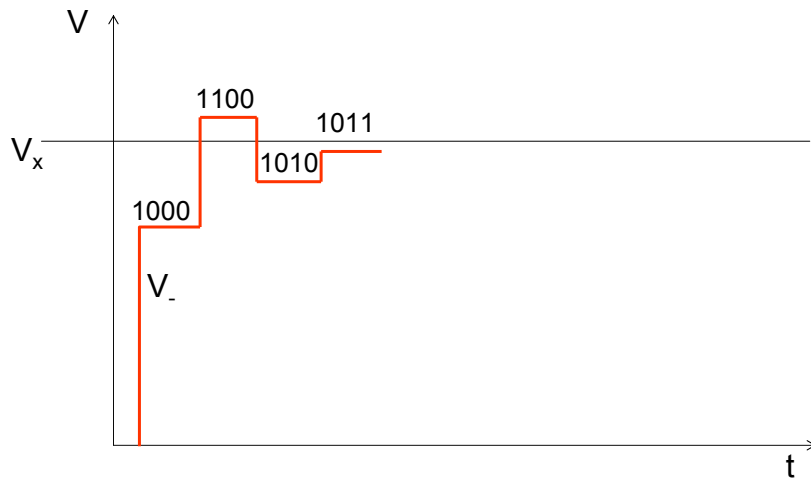
Key idea: binary search:

- Set MSB='1'
- if too large: reset MSB
- Set MSB-1='1'
- if too large: reset MSB-1

BF - ES

- 18 -

Successive approximation (2)

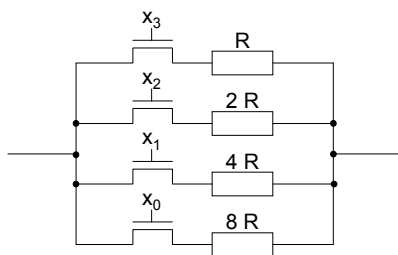


BF - ES

- 19 -

Digital-to-Analog (D/A) Converters

- Convert digital value to conductivity proportional to the digital value

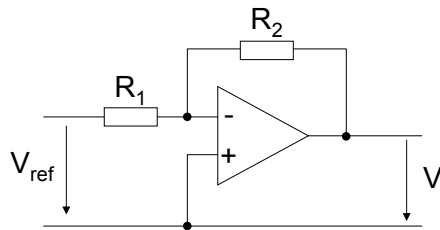


BF - ES

- 20 -

Operational amplifier

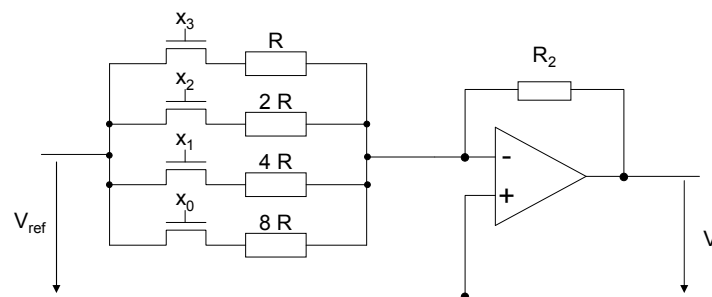
- Use operational amplifier to convert conductivity to voltage: $V = -V_{\text{ref}} R_2 / R_1$



BF - ES

- 21 -

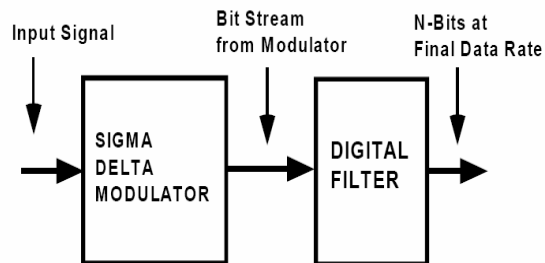
Digital-to-Analog (D/A) Converters (3)



BF - ES

- 22 -

sigma-delta A/D converter

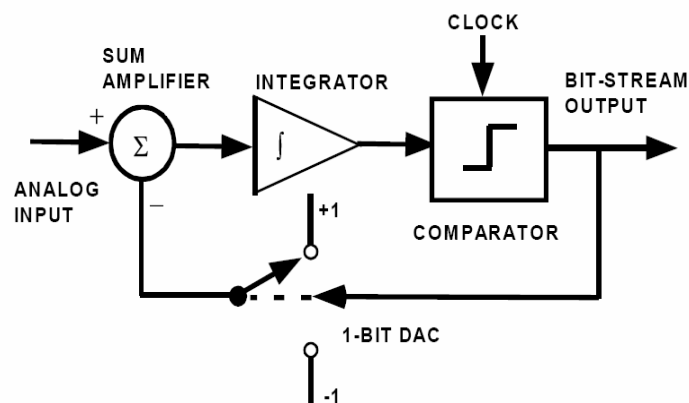


- Modulator generates bit stream whose density of ones (= sliding average value) matches the analog input
- Bit rate many times higher than the final data rate
- Digital filter essentially pursues averaging over sufficiently wide window

BF - ES

- 23 -

1st order sigma-delta modulator



- Generates bit stream whose density of ones (= sliding average) matches the analog input

BF - ES

- 24 -

Actuators and output

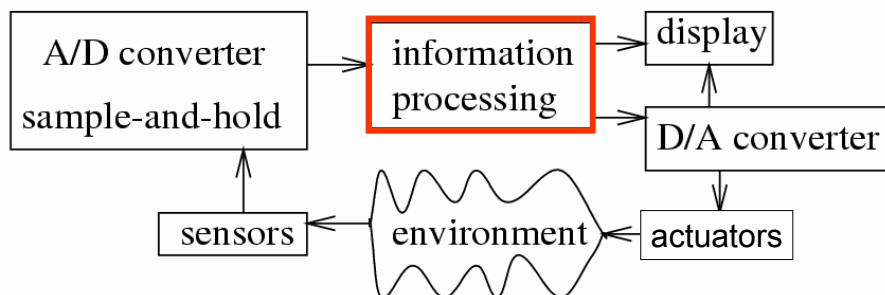
- Huge variety of actuators and outputs, impossible to represent
- Two base types:
 - analogue drive (requires D/A conversion, unless on/off sufficient)
 - CRTs, speakers, electrical motors with collector
 - electromagnetic (e.g., coils) or electrostatic drives
 - piezo drives
 - digital drive (requires amplification only)
 - LEDs
 - stepper motors
 - relais, electromagnetic valve (if actuation slope irrelevant)

BF - ES

- 25 -

Embedded System Hardware

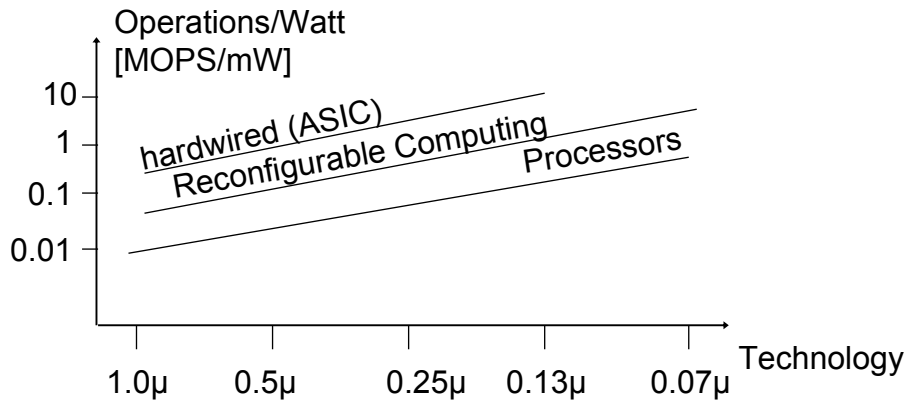
- Embedded system hardware is frequently used in a loop („*hardware in a loop*“):



BF - ES

- 26 -

Hardware Efficiency

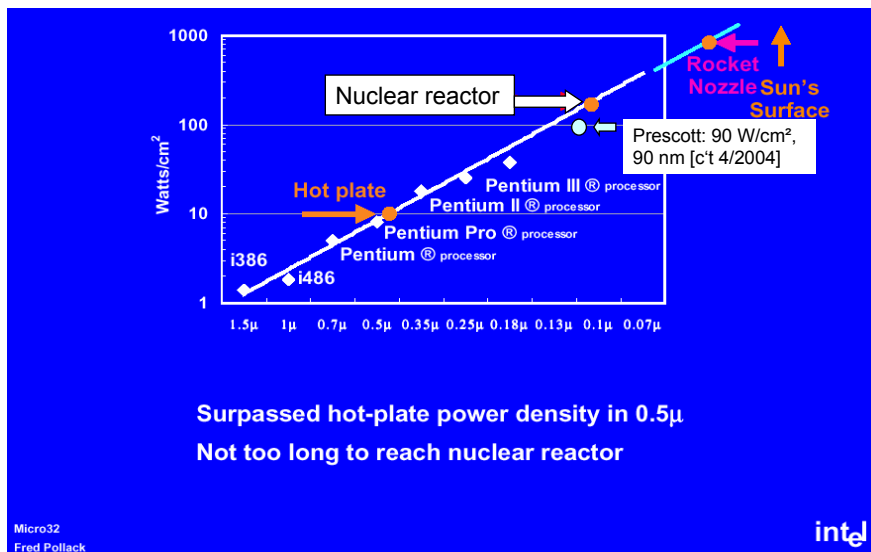


[H. de Man, Keynote, DATE'02;
T. Claasen, ISSCC99]

BF - ES

- 27 -

Power density continues to get worse



- 28 -

Surpassed hot (kitchen) plate ...? Why not use it?

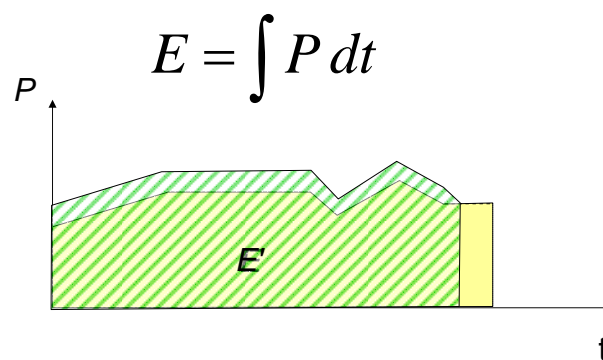


http://www.phys.ncku.edu.tw/~htsu/humor/fry_egg.html

BF - ES

- 29 -

Power and energy are related to each other



In many cases, faster execution also means less energy, but the opposite may be true if power has to be increased to allow faster execution.

BF - ES

- 30 -

Low Power vs. Low Energy Consumption

- Minimizing the **power consumption** is important for
 - the design of the power supply
 - the design of voltage regulators
 - the dimensioning of interconnect
 - short term cooling
- Minimizing the **energy consumption** is important due to
 - restricted availability of energy (mobile systems)
 - limited battery capacities (only slowly improving)
 - very high costs of energy (solar panels, in space)
 - cooling
 - high costs
 - limited space
 - reliability
 - long lifetimes, low temperatures

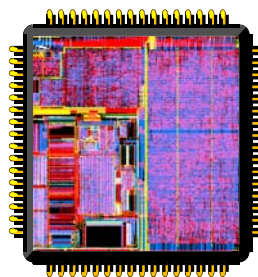


BF - ES

- 31 -

Application Specific Circuits (ASICs) or Full Custom Circuits

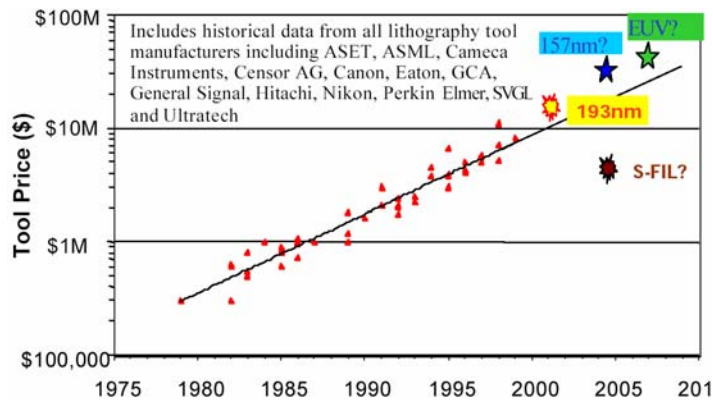
- Custom-designed circuits necessary
 - if ultimate speed or
 - energy efficiency is the goal and
 - large numbers can be sold.
- Approach suffers from
 - long design times,
 - lack of flexibility (changing standards) and
 - high costs (e.g. Mill. \$ mask costs).



BF - ES

- 32 -

Mask cost for specialized HW becomes very expensive



Trend towards implementation in Software

[http://www.molecularimprints.com/Technology/tech_articles/MIL_COO_NIST_2001.PDF]

BF - ES

- 33 -

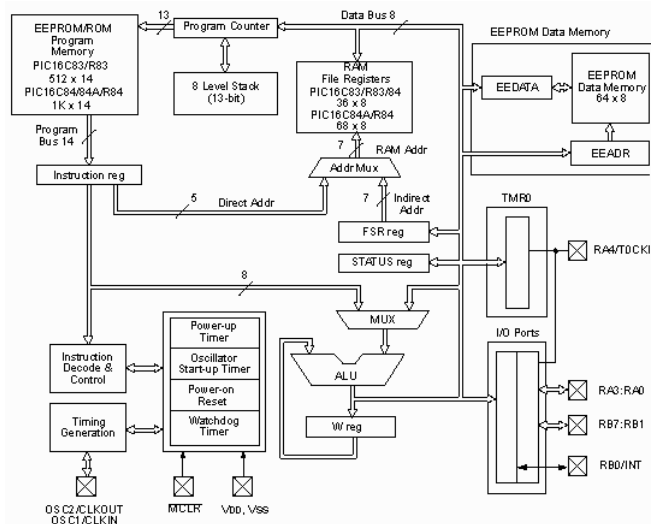
Micro-controllers

- Integrate several components of a microprocessor system onto one chip
 - CPU, Memory, Timer, IO
- Low cost, small packaging
- Easy integration with circuits
- Single-Purpose

BF - ES

- 34 -

Example: PIC16C8X



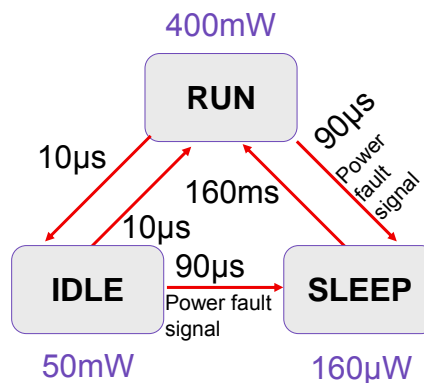
BF - ES

- 35 -

Dynamic power management (DPM)

Example: STRONGARM SA1100

- **RUN**: operational
- **IDLE**: a sw routine may stop the CPU when not in use, while monitoring interrupts
- **SLEEP**: Shutdown of on-chip activity



BF - ES

- 36 -

Fundamentals of dynamic voltage scaling (DVS)

Power consumption of CMOS circuits (ignoring leakage):

$$P = \alpha C_L V_{dd}^2 f \text{ with}$$

α : switching activity

C_L : load capacitance

V_{dd} : supply voltage

f : clock frequency

BF - ES

- 37 -

Fundamentals of dynamic voltage scaling (DVS)

Power consumption of CMOS circuits (ignoring leakage):

$$P = \alpha C_L V_{dd}^2 f \text{ with}$$

α : switching activity

C_L : load capacitance

V_{dd} : supply voltage

f : clock frequency

Delay for CMOS circuits:

$$\tau = k C_L \frac{V_{dd}}{(V_{dd} - V_t)^2} \text{ with}$$

V_t : threshold voltage

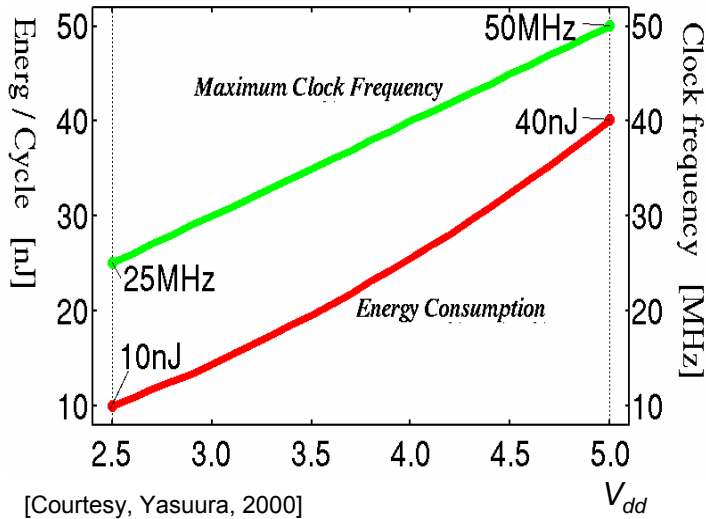
($V_t < \text{than } V_{dd}$)

☞ Decreasing V_{dd} reduces P quadratically, while the run-time of algorithms is only linearly increased
 $E = P \times t$ decreases linearly (ignoring the effects of the memory system and V_t)

BF - ES

- 38 -

Voltage scaling: Example

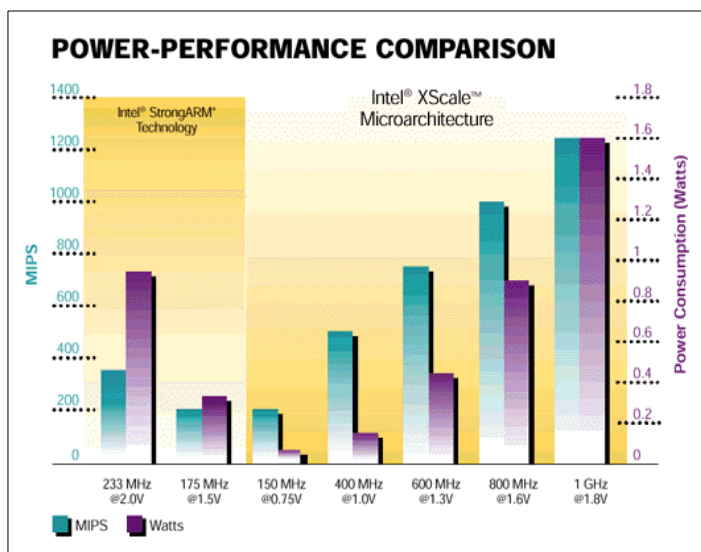


[Courtesy, Yasuura, 2000]

BF - ES

- 39 -

Variable-voltage/frequency example: INTEL Xscale



BF - ES

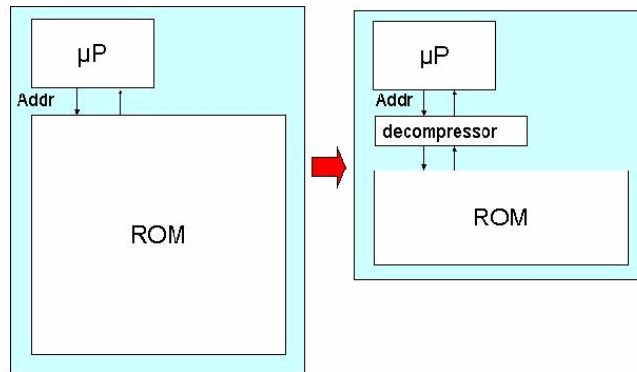
OS should schedule distribution of the energy budget.

From Intel's Web Site

- 40 -

Key requirement #2: Code-size efficiency

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

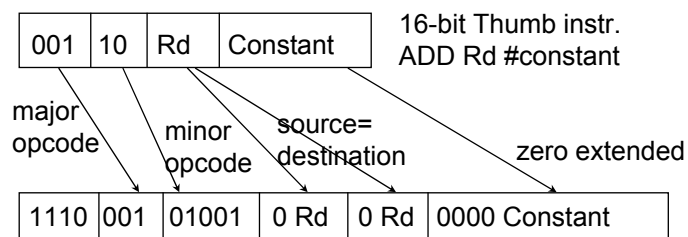


BF - ES

- 41 -

Code-size efficiency

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



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.

BF - ES

- 42 -

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

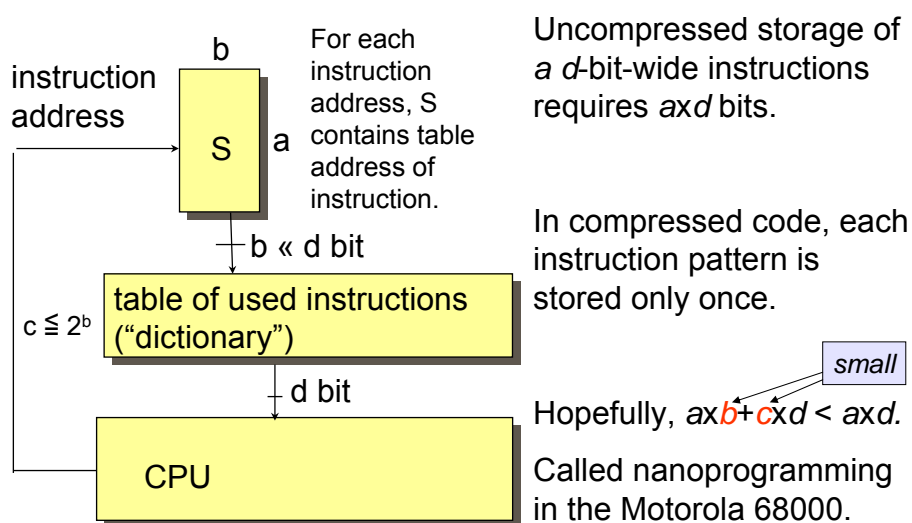
“Dictionary-based coding schemes cover a wide range of various coders and compressors. Their common feature is that the methods use some kind of a dictionary that contains parts of the input sequence which frequently appear. The encoded sequence in turn contains references to the dictionary elements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

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

BF - ES

- 43 -

Key idea (for d bit instructions)



BF - ES

- 44 -

Cache-based decompression

- Main idea: decompression whenever cache-lines are fetched from memory.
- Cache lines \leftrightarrow 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]

BF - ES

- 45 -

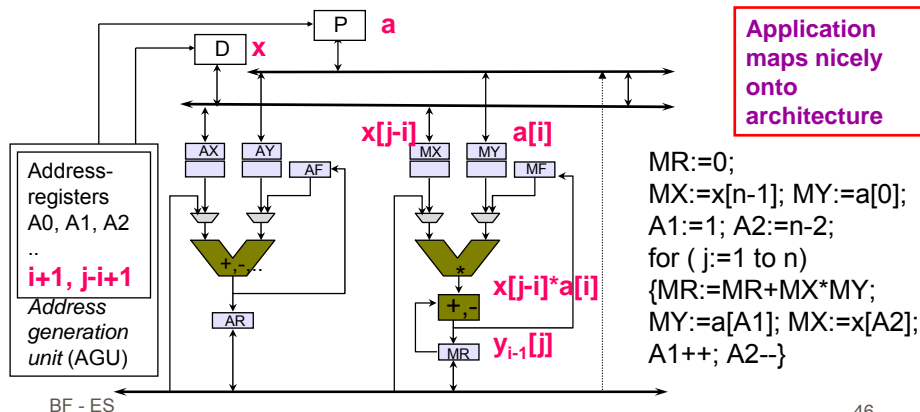
Key requirement #3: Run-time efficiency

- Domain-oriented architectures -

Application: $y[j] = \sum_{i=0}^{n-1} x[j-i] * a[i]$

$\forall i: 0 \leq i \leq n-1: y_i[j] = y_{i-1}[j] + x[j-i] * a[i]$

Architecture: Example: Data path ADSP210x



BF - ES

- 46 -

DSP-Processors: multiply/accumulate (MAC) and zero-overhead loop (ZOL) instructions

MR:=0; A1:=1; A2:=n-2; MX:=x[n-1]; MY:=a[0];

for (j:=1 to n)

{MR:=MR+MX*MY; MY:=a[A1]; MX:=x[A2]; A1++; A2--}

Multiply/accumulate (MAC) instruction

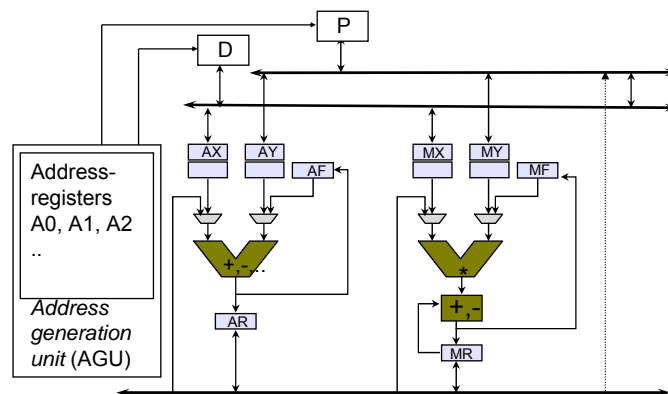
Zero-overhead loop (ZOL) instruction preceding MAC instruction.
Loop testing done in parallel to MAC operations.

BF - ES

- 47 -

Heterogeneous registers

Example (ADSP 210x):



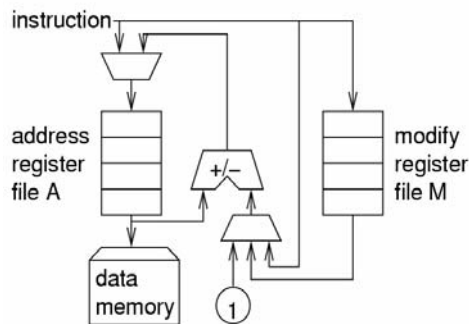
Different functionality of registers An, AX, AY, AF, MX, MY, MF, MR

BF - ES

- 48 -

Separate address generation units (AGUs)

Example (ADSP 210x):



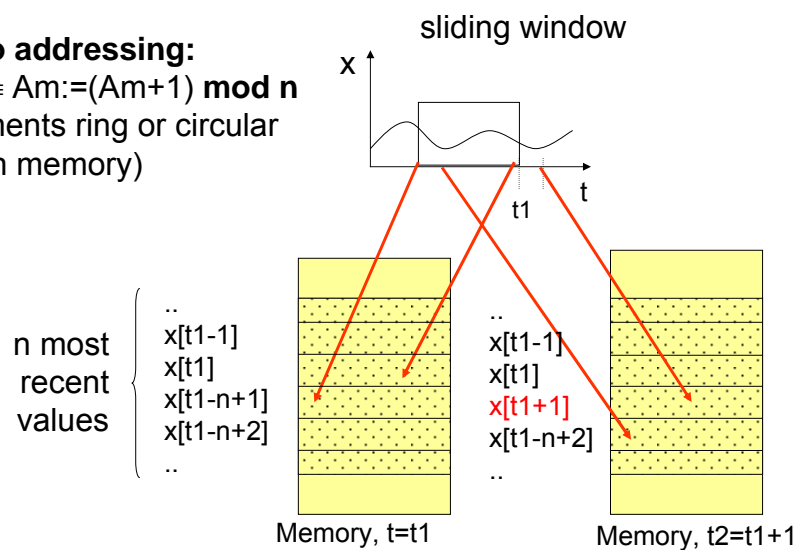
- Data memory can only be fetched with address contained in A,
 - but this can be done in parallel with operation in main data path (takes effectively 0 time).
 - $A := A \pm 1$ also takes 0 time,
 - same for $A := A \pm M$;
 - $A := \langle \text{immediate in instruction} \rangle$ requires extra instruction
- ☞ Minimize load immediates
- ☞ Optimization in optimization chapter

BF - ES

- 49 -

Modulo addressing

Modulo addressing:
 $A_{m++} \equiv A_m := (A_m + 1) \bmod n$
 (implements ring or circular buffer in memory)



BF - ES

- 50 -

Saturating arithmetic

- Returns largest/smallest number in case of over/underflows

- Example:

a		0111
b	+	1001
<hr/>		
standard wrap around arithmetic		(1)0000
saturating arithmetic		1111
<hr/>		
(a+b)/2:	correct	1000
	wrap around arithmetic	0000
	saturating arithmetic + shifted	0111 „almost correct“

- Appropriate for DSP/multimedia applications:
 - No timeliness of results if interrupts are generated for overflows
 - Precise values less important
 - Wrap around arithmetic would be worse.