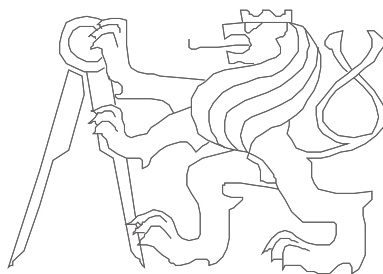


Pokročilé architektury počítačů

01

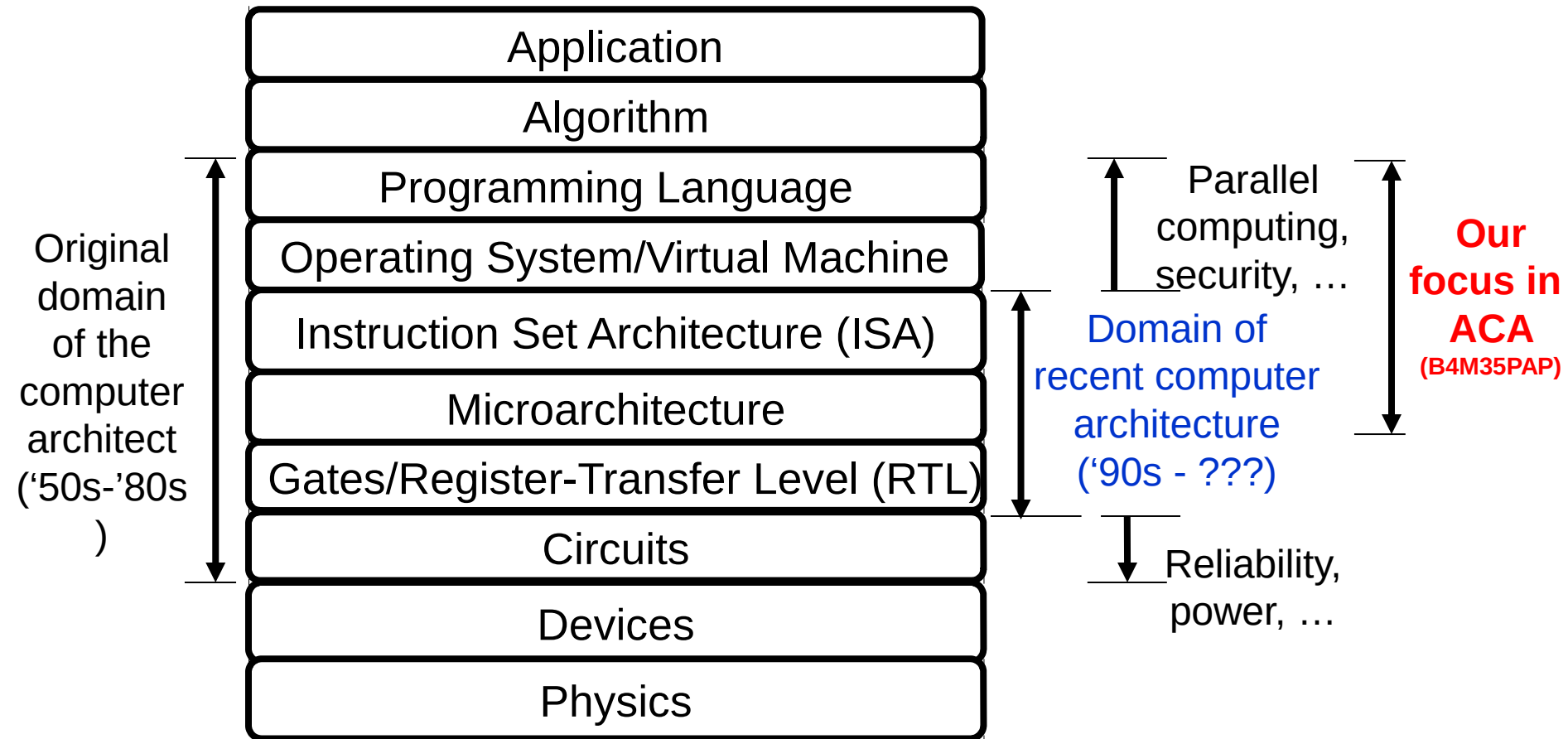
Úvodní přednáška



České vysoké učení technické, Fakulta elektrotechnická
Autor materiálu: Michal Štepanovský Přednášející: Pavel Píša

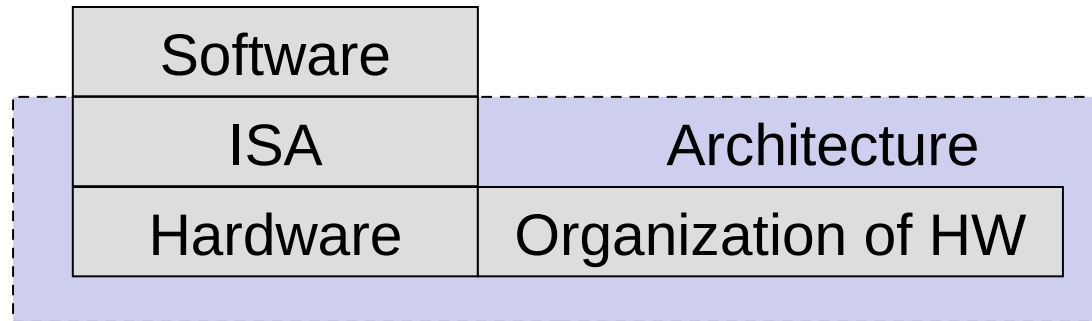
Ver.1.10

Abstraction Layers in Modern Systems



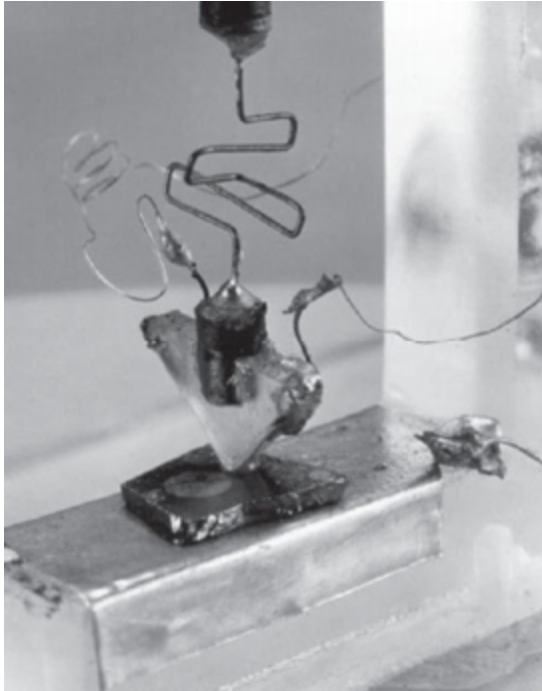
Reference: John Kubiawicz: EECS 252 Graduate Computer Architecture, Lecture 1. University of California, Berkeley

Obsah a cíl předmětu

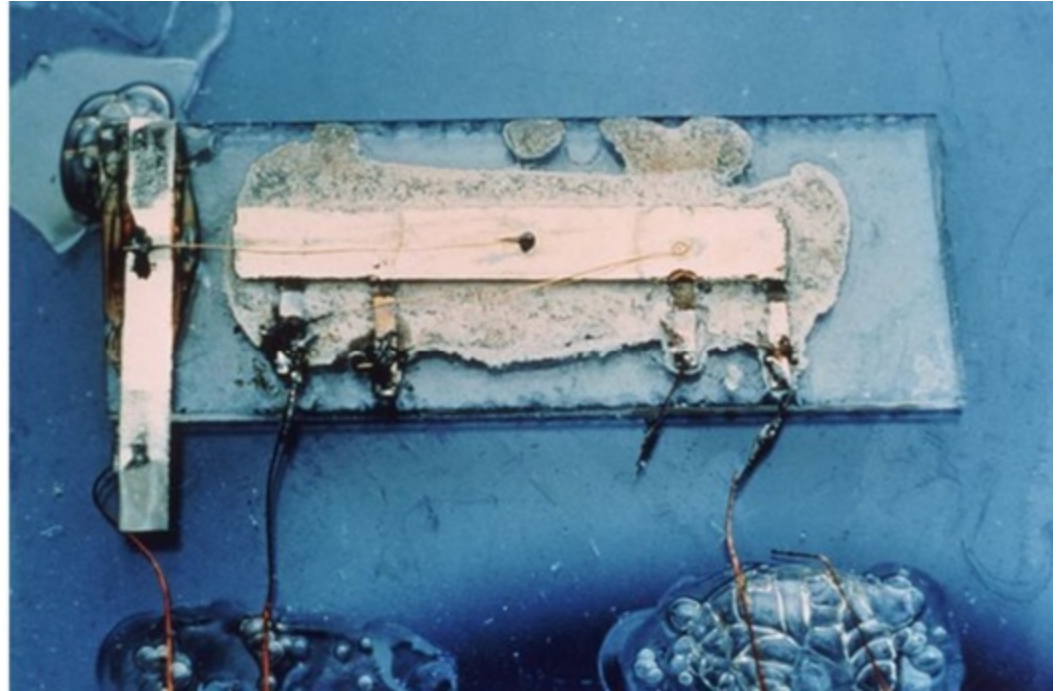


- Dva různé procesory mohou implementovat stejnou ISA, ale mohou se lišit v organizaci i v hardwarové realizaci (různé pipeline, organizace cache,...)
- Organizace – pohled „shora“ na návrh počítače (paměťový subsystém, struktura sběrnic,...)
- Hardware – pohled „zdola“ (logické obvody, technologie,...)

What is it?



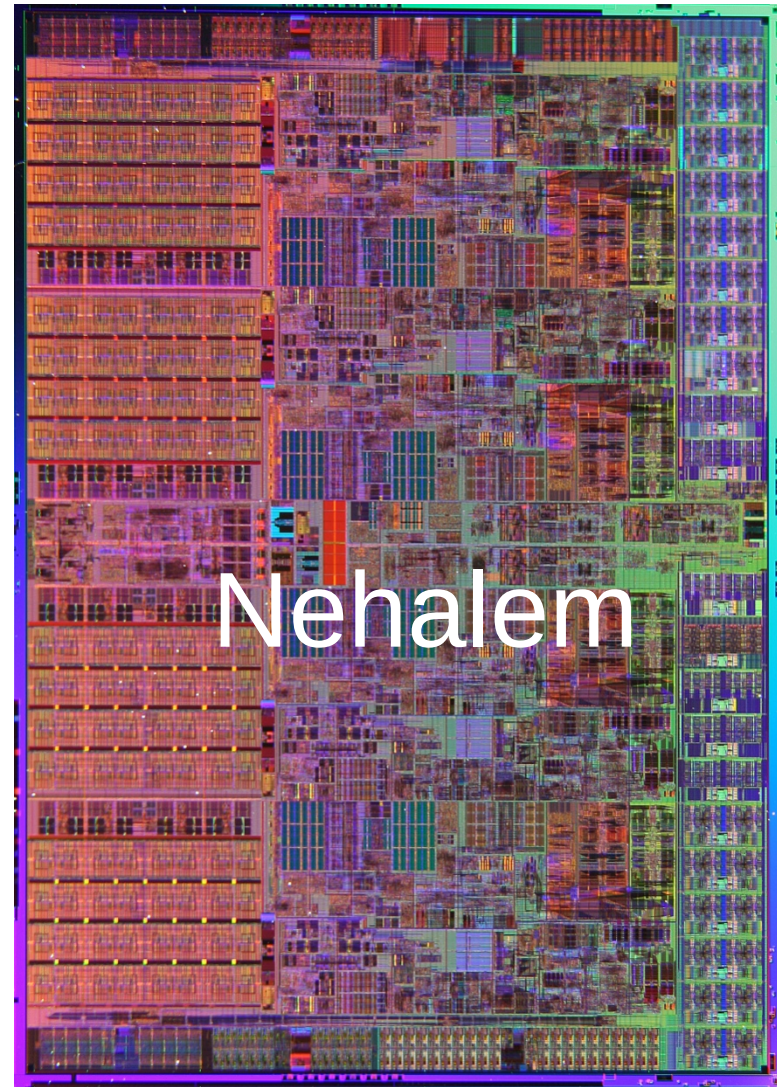
First Transistor
(Bell Labs) – Dec. 23, 1947



First integrated circuit (IC)
Jack Kilby's original IC (Texas Instruments) - 1958

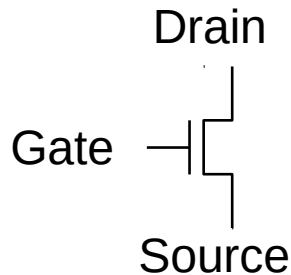
Technology constantly on the move!

- Num of transistors not limiting factor
 - Currently ~ 1 billion transistors/chip
 - Problems:
 - Too much Power, Heat, Latency
 - Not enough Parallelism
- 3-dimensional chip technology?
 - Sandwiches of silicon
 - “Through-Vias” for communication
- On-chip optical connections?
 - Power savings for large packets
- The Intel® Core™ i7 microprocessor (“Nehalem”)
 - 4 cores/chip
 - **45 nm**, Hafnium hi-k dielectric
 - 731M Transistors

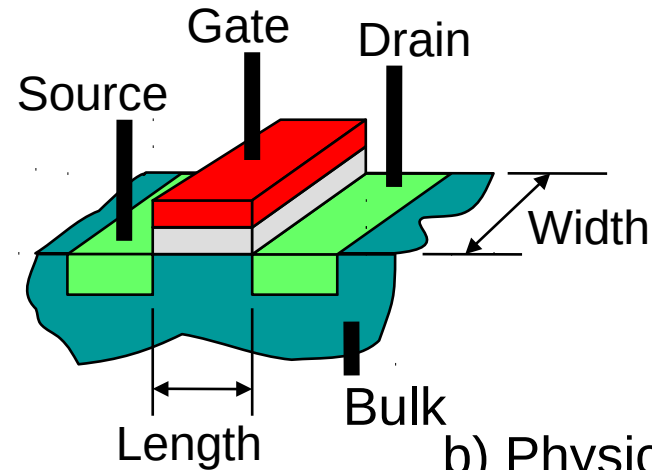


Reference: John Kubiatowicz: EECS 252 Graduate Computer Architecture, Lecture 1. University of California, Berkeley

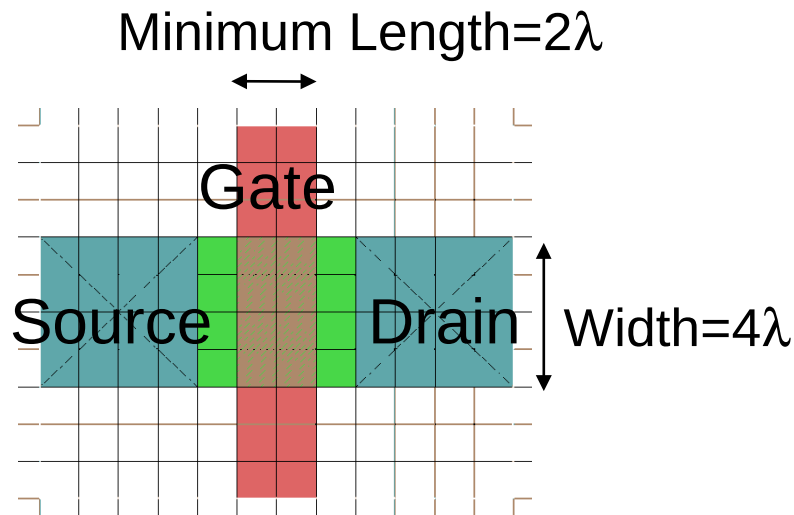
Transistors



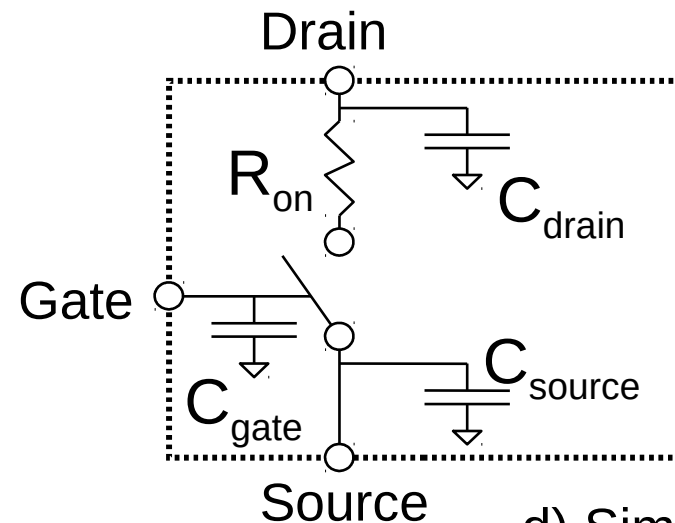
a) Circuit Symbol



b) Physical Realization

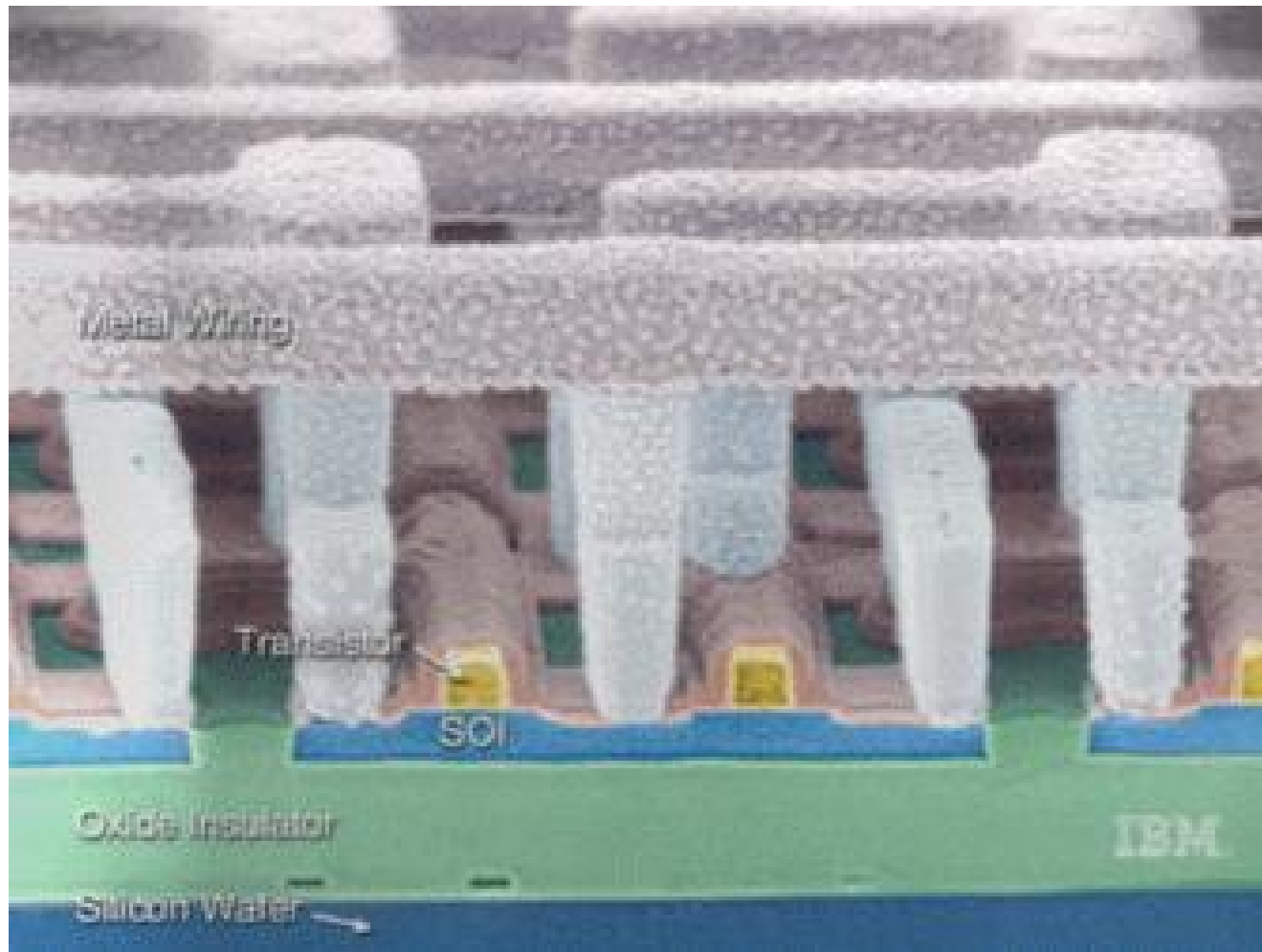


c) Layout View



d) Simple RC Model

Transistors - Today

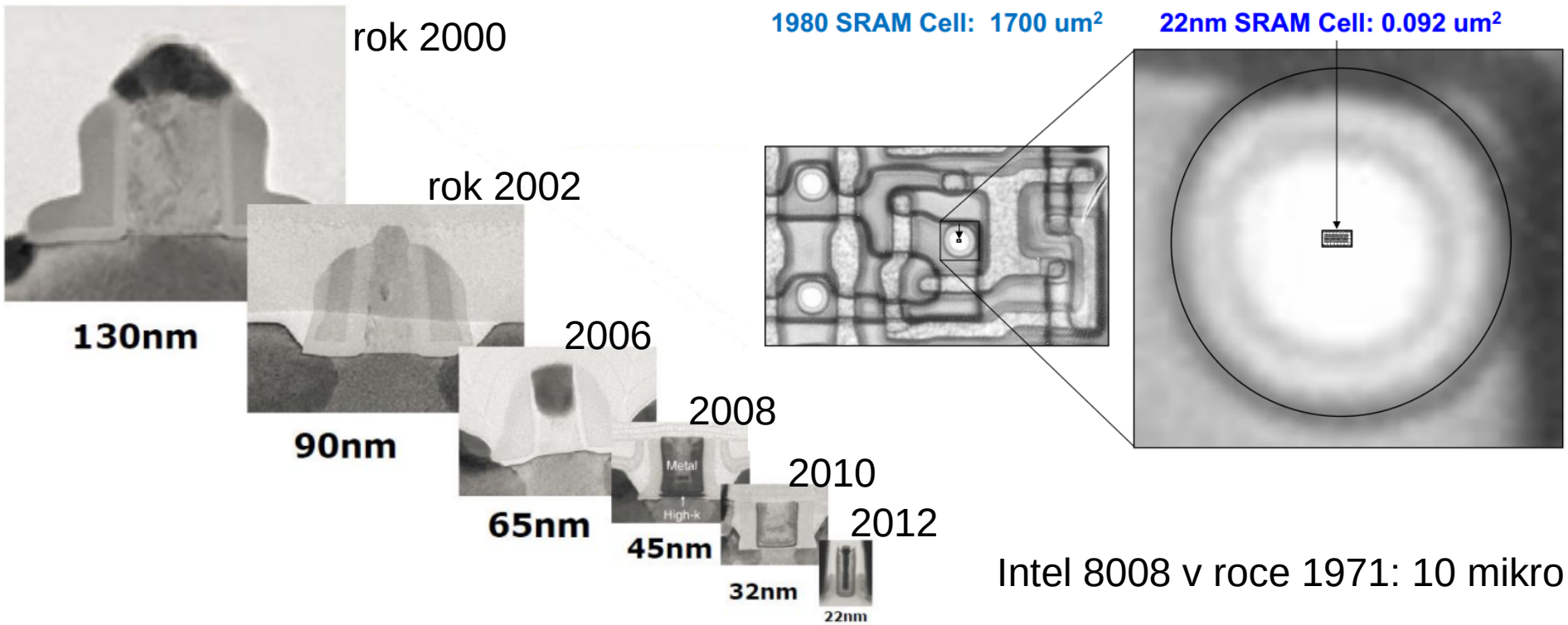


©IBM

IBM SOI Technology

K.Asanovic: VLSI for Architects - Advanced VLSI Computer Architecture,
<http://groups.csail.mit.edu/cag/6.893-f2000/lectures/l02-vlsi-for-archs.pdf>, Copyright 2000, Krste Asanovic.

Gordon Moore



- Rychlost šíření elektrických signálů shora omezena rychlostí světla (při 3,6 GHz je perioda 0,278 ns $\Rightarrow$ 8,3 cm) (zpoždění propojení dominuje nad zpožděním hradel)
- Ultimate minimum feature size may be determined by atomic spacing – in Si the lattice constant is ~ 0.5 nm.

Dmitri Nikonov: Beyond CMOS computing - 1. CMOS Scaling. h

http://nanohub.org/resources/18348/download/NikonovBeyondCMOS_1_scaling.pdf

S. Demlow: A Brief History of the field of VLSI, ECE 410, January 9, 2012

Obsah a cíl předmětu – přednášky

1. Úvodní přednáška

Úvod do moderní architektury počítačů; počítače řízené tokem instrukcí (control driven) a tokem údajů (data driven a demand driven). Klasifikace počítačových architektur podle Flynna; Vícejádrové, víceprocesorové a vícepočítačové systémy, pojem paralelního zpracování. Amdahlův a Gustafsonův zákon. Výkonové metriky.

2. Přednáška - Od skalárního procesoru k superskalárnímu (základní organizace superskalárního procesoru)

Superskalární procesory se statickým, dynamickým a hybridním plánováním vykonávání instrukcí.

3. Přednáška - Superskalární techniky I - Tok dat uvnitř procesoru (register data flow)

Přejmenování registrů (Tomasulův algoritmus) a datové spekulace. Podpora přesného přerušení.

4. Přednáška - Superskalární techniky II - Spekulativní vykonávání (řídící spekulace)

Predikce, prediktory a předvýběr instrukcí. Statické a dynamické predikce; Smithův prediktor, dvou-úrovňové prediktory s lokální a globální historií, dvou-módový prediktor, a další. Zotavení po nesprávné predikci.

5. Přednáška - Superkalární techniky III - Memory data flow, a Procesory VLIW a EPIC

Tok dat z/do paměti. Load bypassing a Load forwarding. Spekulativní load. Některé další způsoby redukce latence paměti. Procesory VLIW a EPIC. Využití datového paralelismu, SIMD a vektorové instrukce v ISA. Loop unrolling a Software pipelining - vykonání na WLIV a superskalárním procesoru.

6. Přednáška – Paměťový subsystém

Neblokující cache, Victim cache, Virtuální paměť a cache

7. Přednáška - Multiprocesorové systémy a problém koherence paměti

Architektury multiprocesorových počítačů. Systémy s distribuovanou a sdílenou pamětí (DMS, SMS). Architektury symetrických multiprocesorových počítačů Způsoby zajištění koherence v SMP.

8. Přednáška - Multiprocesorové systémy a problém konzistence paměti

Pravidla pro provádění paměťových operací, zajištění sekvenční konzistence, modely paměťové konzistence.

9. Přednáška - Programování paralelních systémů I

Způsoby programování paralelních systémů - použití Message Passing Interface (MPI) a Open Multi-Processing (OpenMP) pro tvorbu paralelních programů.

10. Přednáška - Programování paralelních systémů II

Synchronizace

11. Přednáška – I/O podsystém (do doby než přijdou absolventi APO z LS2014)

PCIe, HyperTransport, QuickPathInterconnect

12. Přednáška - Časový a prostorový paralelizmus v praxi

Ukázka vybraných partií na procesoru Intel Nehalem

13. MPP a clustery, propojovací sítě

14. Perspektivy a omezení dalšího rozvoje

část I – Verilog a popis hardwaru počítače

Seminární projekt: Návrh jednoduchého procesoru

část II – Programování paralelních systémů

Seminární projekt: Tvorba paralelního programu pro vícejádrový víceprocesorový paralelní výpočetní systém

část III – Samostatná práce a prezentace výsledků seminárních projektů

Podmínky absolvování předmětu

- Účast na cvičeních – povinná
(2 absence bez doložení potvrzení od lékaře povoleny)
- Odevzdání všech seminárních projektů
- Zkouška:
 - úspěšné absolvování písemné části zkoušky (min. 60%)
 - ústní část zkoušky

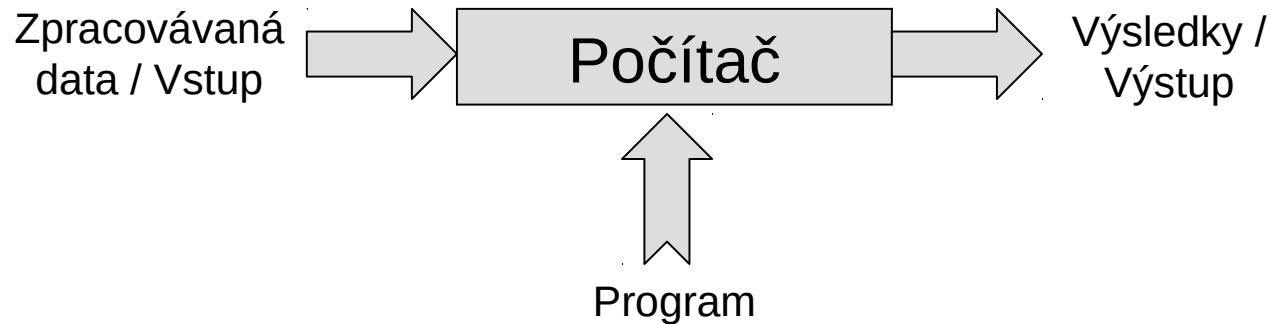
Stránky předmětu: **<https://cw.fel.cvut.cz/wiki/courses/b4m35pap/start>**

Literatura:

- Hennesy, J. L., Patterson, D. A.: Computer Architecture : A Quantitative Approach, Third Edition, San Francisco, Morgan Kaufmann Publishers, Inc., 2002
- Shen, J.P., Lipasti, M.H.: Modern Processor Design : Fundamentals of Superscalar Processors, First Edition, New York, McGraw-Hill Inc., 2004
- Grama A., Gupta, A. et al.: Introduction to Parallel Computing, Second Edition, Addison Wesley, 2003

Obečnější pohled na počítač a jeho činnost

- Pokud pod pojmem *počítač* budeme rozumět univerzálně použitelné zařízení na automatické zpracování dat, potom:



- Program může být definován jako specifikace množiny operací nad operandy (zpracovávanými daty), které jsou potřeba k vykonání požadované úlohy (získání výsledků).
- $F = (A * B + C * D) / (E + 2)$,
A, B, C, D, E – vstupy; F – výstup;
zápis v programovacím jazyce vedoucí k výsledku – program

Uvažujme: $F = (A*B + C*D) / (E+2)$

Control Flow přístup:

- Mult Reg1, A, B
- Mult Reg2, C, D
- Add Reg1, Reg1, Reg2
- Add Reg2, E, 2
- Div F, Reg1, Reg2
- Write F

Činnost počítače je určena
sekvencí instrukcí...

Obecnější pohled na počítač a jeho činnost

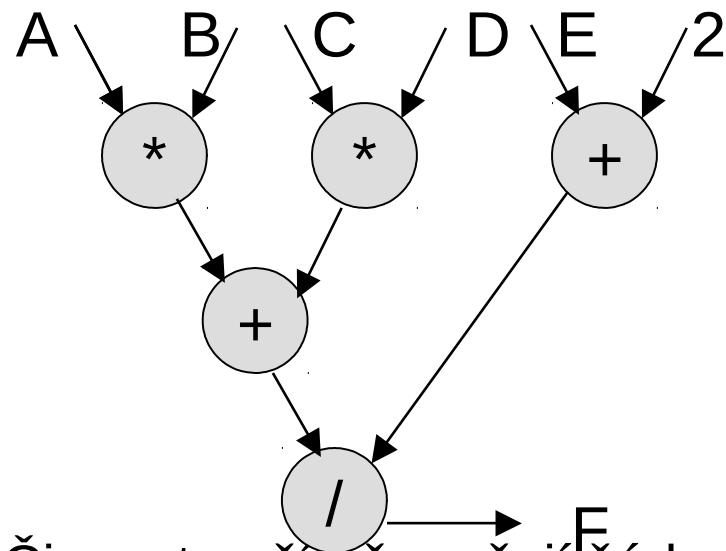
$$\text{Uvažujme: } F = (A * B + C * D) / (E + 2)$$

Control Flow přístup:

- Mult Reg1, A, B
- Mult Reg2, C, D
- Add Reg1, Reg1, Reg2
- Add Reg2, E, 2
- Div F, Reg1, Reg2
- Write F

Činnost počítače je určena sekvencí instrukcí...

Data Flow přístup:

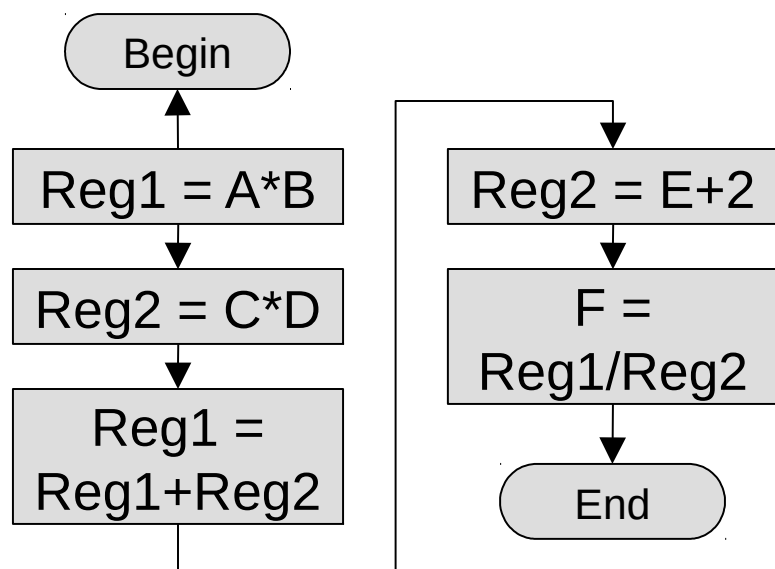


Činnost počítače určují žádosti o výsledek (demand driven), nebo přítomnost operandů (data driven)..

Obečnější pohled na počítač a jeho činnost

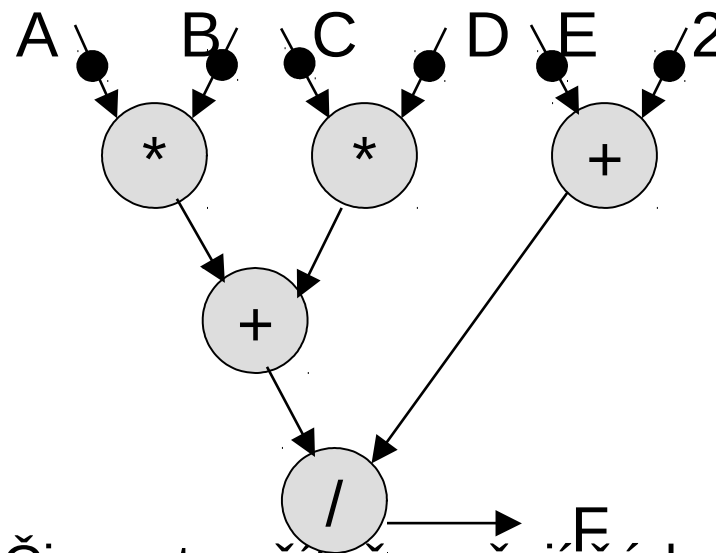
$$\text{Uvažujme: } F = (A * B + C * D) / (E + 2)$$

Control Flow přístup:



Činnost počítače je určena sekvencí instrukcí...

Data Flow přístup:



Činnost počítače určují žádosti o výsledek (demand driven), nebo přítomnost operandů (data driven)..

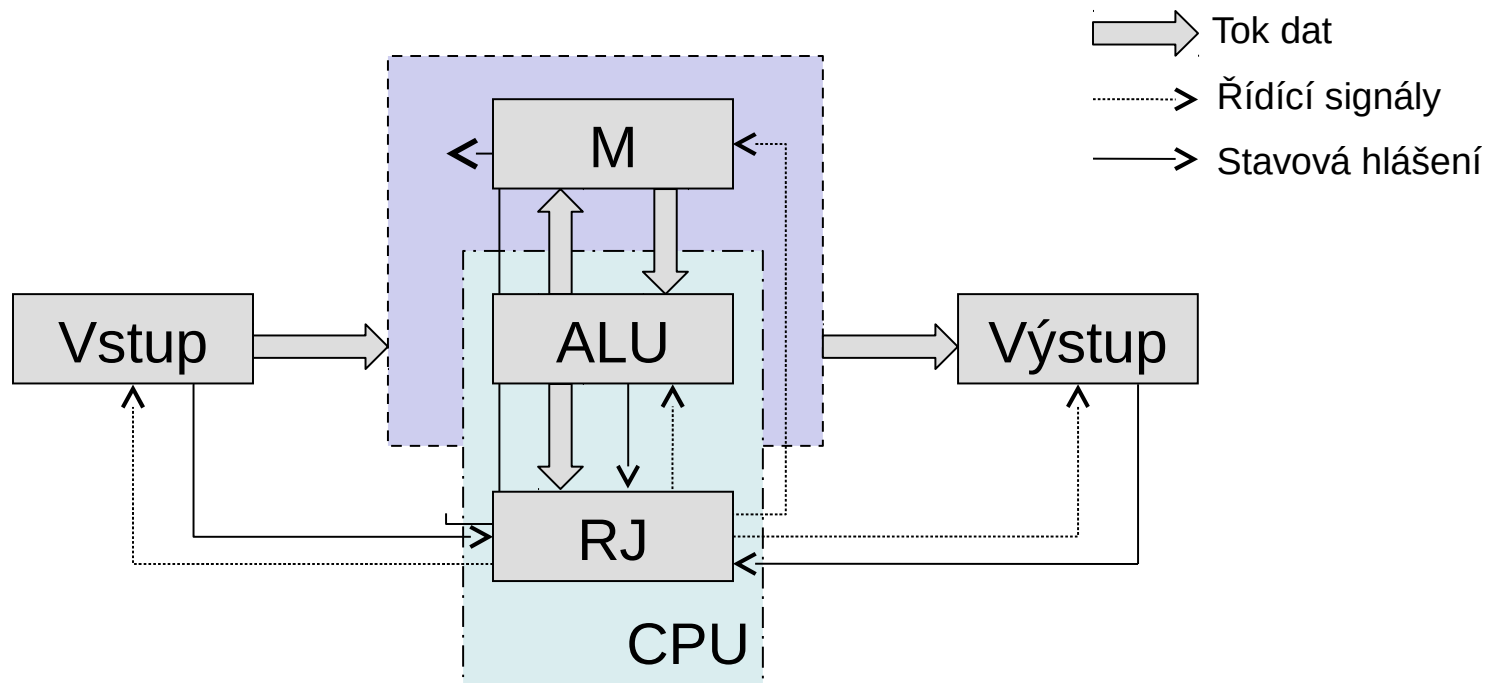
Control Flow vs. Data Flow

- Instrukce se vykonávají sekvenčně tak, jak jsou zapsány, resp. jak to určuje čítač instrukcí (program counter).
Explicitní přenosy řízení se realizují skokovými instrukcemi...
→ Imperativní programování
- Případné vykonávání instrukcí (z důvodu lepšího využití HW) v jiném pořadí nesmí dávat jiný výsledek!

- Instrukce v DF počítači se realizuje jako šablona, kterou tvoří operátor, příjemce operandu a určení výsledku. Instrukce se vykoná vždy, když jsou dispozici potřebné operandy...
- Pořadí vykonávání instrukcí je dáno datovými závislostmi a dostupností prostředků
→ Nezávislost instrukce od umístění v programu

Control Flow

- Čtyři hlavní subsystémy (von Neumann): Operační paměť (M), Aritmeticko-logická jednotka (ALU), Řídicí jednotka (RJ), Vstupně-výstupní subsystém (V/V)
- RJ – řídí vykonávání instrukčního cyklu (fáze výběru, dekódování, vykonání)



Základní elementy:

- Uzly (add, if, switch, merge, not,...), hrany (datové, řídicí), tokeny (datové, řídicí)

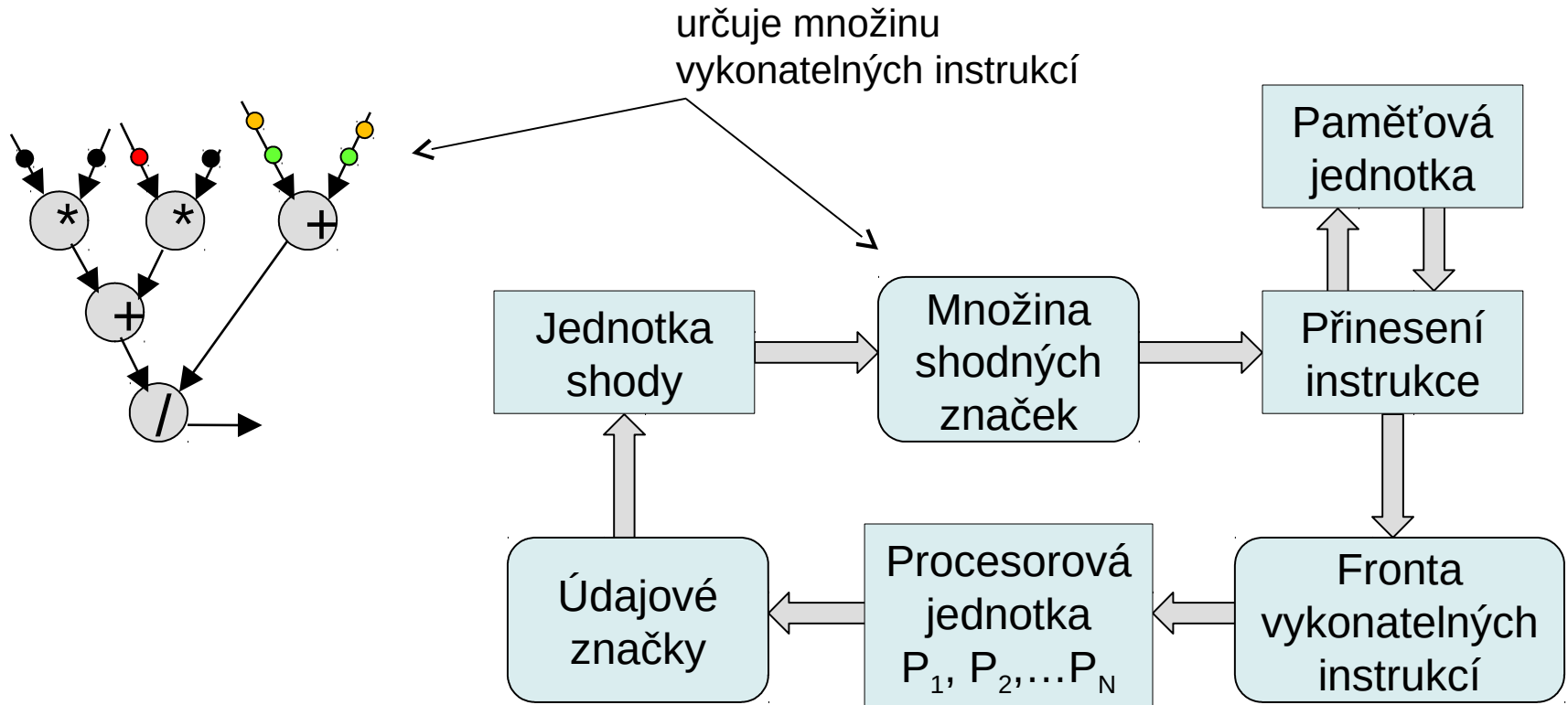
Architektura:

- Statická (povolen jeden token na hranu – dáno konstrukcí grafu.. takže není důvod umísťovat jeden uzel na vícero VE; programový graf je staticky alokován na fyzické VE)
- Dynamická (více tokenů, pořadí identifikováno značkou)
- Hybridní (více tokenů; dva módy vykonávání: 1. FIFO (přicházející tokeny jsou řazeny do fronty), 2. pořadí identifikováno značkou)

Vykonávání:

- Statické (vykonání operace pokud výstup neobsahuje tokeny, na vstupu může být i vícero tokenů),
- Dynamické

Dynamická architektura DF počítače



- Zjišťování podmínek vykonání uzlů a detekce vykonatelných uzlů
- Přiřazování vykonatelných uzlů procesorům
- Posouvání tokenů mezi uzly (posun vygenerovaného tokenu k příjemci)
- Dynamické vytváření instancí uzlů (příp. subgrafů) – smyčky v DF grafu...
- Rozlišení tokenů náležících různým instancím téhož uzlu
- Komunikace s vnějším světem
- Detekce ukončení programu

- Rozvoj v 70-tých až 80-tých letech
- Nyní princip DF v jednoúčelových zařízeních (zejména na bázi FPGA),
například v rekonfigurovatelném SoC např. pro dekódování MPEG4 videa [1] (9 hierarchických FSM, 31 stavů, 44 přechodů, 89 SDF uzlů),
nebo v koprocesoru pro zpracování obrazu [2] (rozpoznávání čárového kódu)
a jiných...

Zdroje:

- [1] Sunghyun Lee , Sungjoo Yoo , Kiyoun Choi: Reconfigurable SoC Design with Hierarchical FSM and Synchronous Dataflow Model (2002).
- [2] Richard Thavot, Romuald Mosqueron et al. : Dataflow design of a co-processor architecture for image processing. DASIP 2008, Bruxelles, Belgium, 24-26 November 2008.
- [3] John A. Sharp: Data Flow Computing: Theory and Practice
- [4] <http://comjnl.oxfordjournals.org/cgi/reprint/33/3/230>

Klasifikace počítačových architektur podle Flynna

- **SISD** (Single Instruction stream Single Data stream)
klasická architektura von Neumann bez zavedení paralelizmu
- **SIMD** (Single Instruction stream Multiple Data stream)
vícenásobné VE/PE (processing element); historicky: CM-5, ILLIAC IV; maticové a vektorové počítače; dnes v podobě rozšíření ISA pro práci na krátkých vektorech (MMX, SSE,...), také v GPU → tj. lokální podpora
- **MISD** (Multiple Instruction stream Single Data stream)
pipelining? Ano/ne? ; Uplatnění ve Fault-tolerant systémech – zpracovávání stejného datového proudu, porovnávání výsledků;
- **MIMD** (Multiple Instruction stream Multiple Data stream)
vícevláknové, vícejádrové, víceprocesorové, vícepatáčové systémy (Cluster, Grid, Cloud)
 - SPMD (Single Program stream Multiple Data stream)
 - MPMD (Multiple Program stream Multiple Data stream)

Další superpočítač své doby od firmy Cray



Další superpočítač své doby od firmy Cray

- **Cray XT5-HE (nejvýkonnější systém – září 2010)**
- **MPP (Massively Parallel Processor)**
- **Výkon 1 759 000 Gflops (Rmax), Rpeak = 2,3 PFlops**
- **224 162 výpočetních elementů (37 376 procesorů)**
- **dual hex-core AMD Opteron 2435 (Istanbul) procesory na 2.6GHz (10.4 GFlops)**
- **300TB paměti**
- **disková kapacita 10.7 PB**
- **Linux**

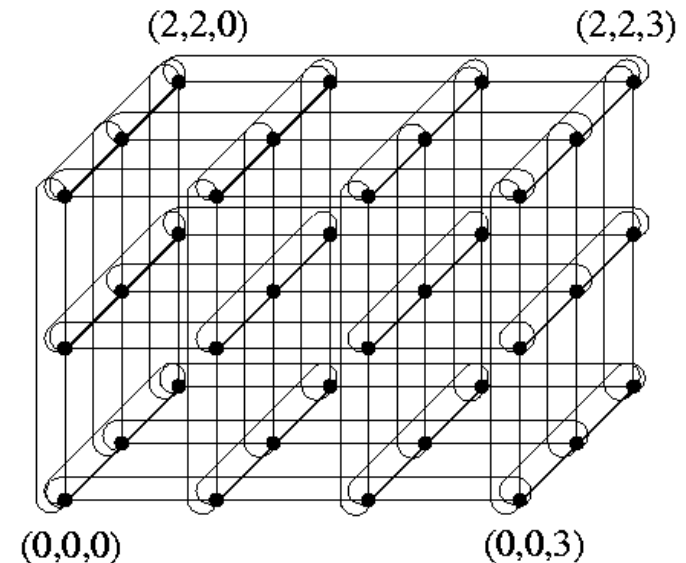
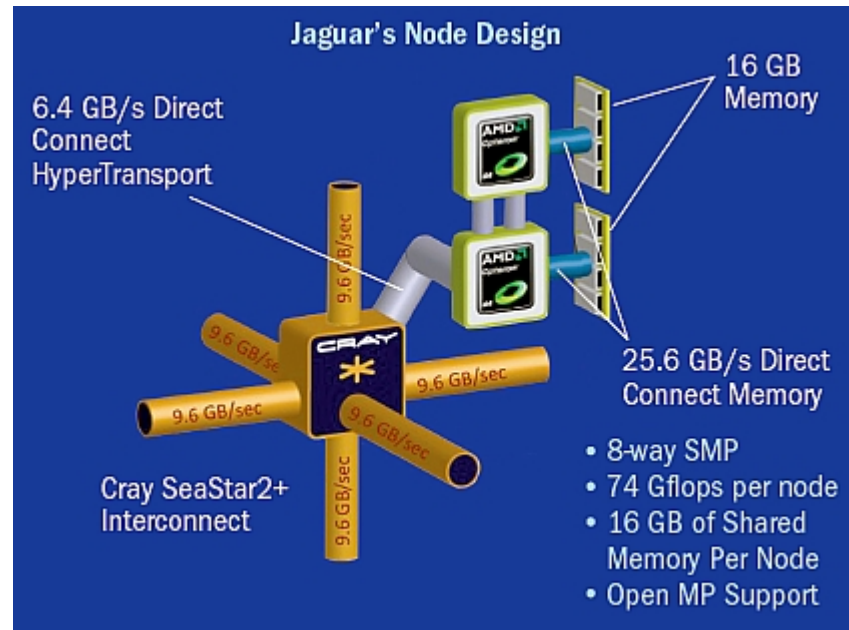
Zdroje informací:

<http://www.top500.org/system/details/10184>

<http://www.nccs.gov/computing-resources/jaguar/>



Cray XT5-HE



- Jeden uzel (node) – dva 6-jádrové procesory
- Každý uzel – 25.6 GB/s přístup do lokální paměti, 6.4GB/s do sítě
- Uzly jsou pospojovány do 3D toroidu
- 12 OpenMP nebo MPI úloh v uzlu, MPI mezi uzly
- Knihovny pro posílání zpráv: MPI 2.0, SHMEM

<http://www.scidacreview.org/0901/html/hardware.html>

Architektura

- Cluster (85%), MPP (15%), Jiné (0,4%)

Operační systém

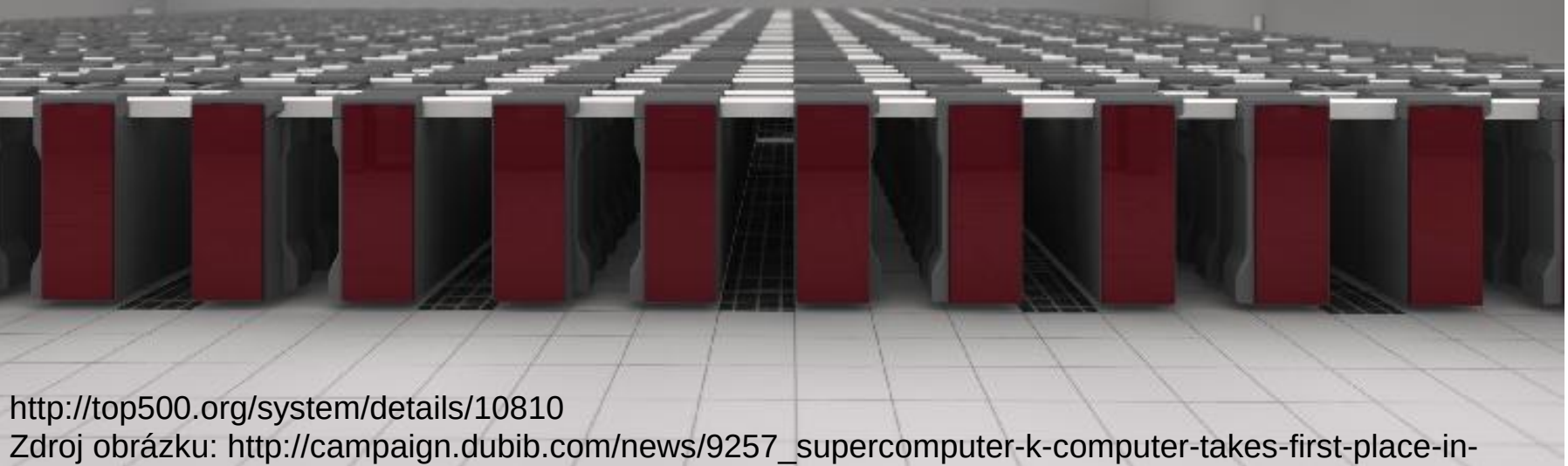
- Linux (91%), Unix (4,4%), Windows (1%)

Počet výpočetních elementů (VE)

- 2049-4096 (22%), 4k - 8k (58%), 8k – 128k (18%),
více než 128 000 VE (1%)
- Nejpočetnější zastoupení výrobců v horní 30
Cray (9), IBM (7), Sun Microsystems (3)

Nejvýkonnější superpočítač - září 2011, září 2014 na 4.místě

- K computer, Fujitsu Cluster
- SPARC64 VIIIfx 2.0GHz (16 GFlops),
- Tofu interconnect (6-dimenzionální toroid)
- Cores: 548 352, (68 544 procesorů), 45nm
- Výkon 8 162 000 GFlops (Rmax), Rpeak = 8,77 Pflops
- Linux, Message Passing Interface (Open MPI)

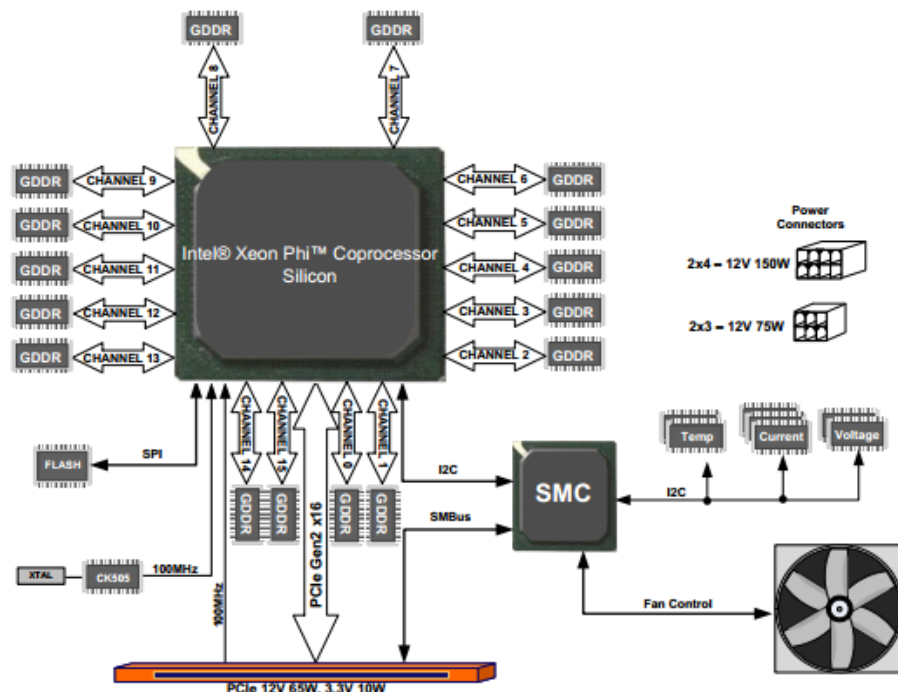


<http://top500.org/system/details/10810>

Zdroj obrázku: http://campaign.dubib.com/news/9257_supercomputer-k-computer-takes-first-place-in-world

Nejvýkonnější superpočítač - září 2014, září 2015

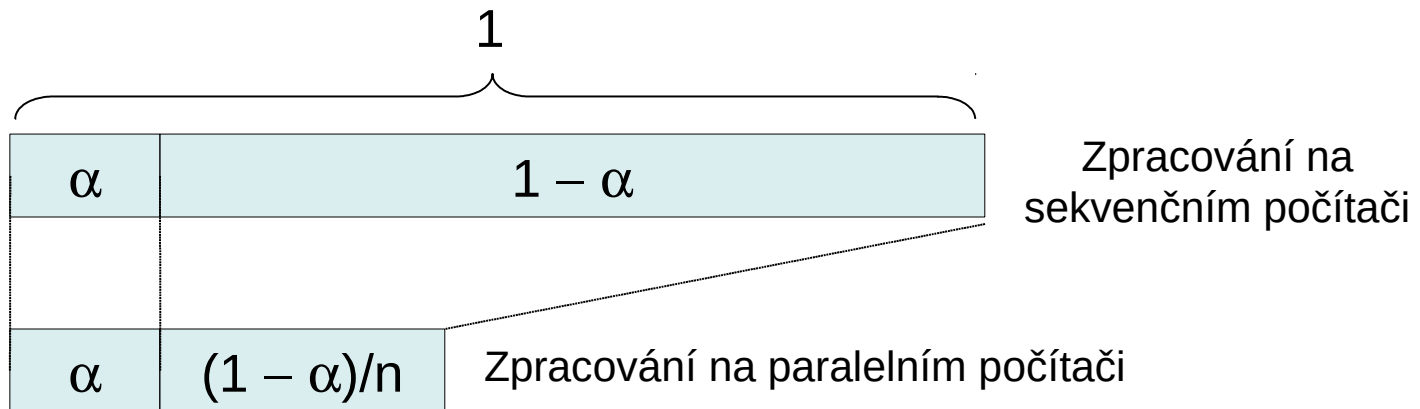
- Tianhe-2 (MilkyWay-2)
- Intel Xeon E5-2692 2.2GHz
(12 cores, 24 threads,
12 x 32 KB L1 i caches,
12 x 32 KB L1 d caches,
12 x 256 KB L2 caches,
1 x 30 MB L3 cache)
- Intel Xeon Phi 31S1P (57/228)
- Each node: two Xeon
processors and three Xeon
Phi coprocessors
- Cores: 3 120 000
- **Výkon 33 862 700 GFlops
(Rmax), Rpeak = 54.9 Pflops**
- **Nový trend: Gflops/Watt**



<http://www.intel.com/content/dam/www/public/us/en/documents/datasheets/xeon-phi-coprocessor-datasheet.pdf>

Amdahlův zákon

- Každý program obsahuje nějaké procento sekvenční části. Toto procento označme α .



$$S(n) = \frac{T(1)}{T(n)} = \frac{\alpha + 1 - \alpha}{\alpha + (1 - \alpha)/n} = \frac{n}{(n - 1)\alpha + 1}$$

$$\lim_{n \rightarrow \infty} S(n) = \lim_{n \rightarrow \infty} \frac{n}{(n - 1)\alpha + 1} = \frac{1}{\alpha}$$

Amdahlův zákon

- Zrychlení (Speed-up): Efektivnost:

$$S(n) = \frac{T(1)}{T(n)}$$

$$E(n) = \frac{S(n)}{n} = \frac{T(1)}{nT(n)}$$

kde $T(1)$ je čas vykonání úlohy na jednom VE,
 $T(n)$ na n VE, a n je počet VE.

Když použijeme 2 VE dosáhneme 2 krát lepší čas?

Amdahlův zákon

- Zrychlení (Speed-up): Efektivnost:

$$S(n) = \frac{T(1)}{T(n)}$$

$$E(n) = \frac{S(n)}{n} = \frac{T(1)}{nT(n)}$$

kde $T(1)$ je čas vykonání úlohy na jednom VE,
 $T(n)$ na n VE, a n je počet VE.

Amdahlův zákon

- Příklad: Máme sekvenční počítač (jeden VE). Výpočet trvá 10 hodin, z toho 1 hodina je neparalelizovatelná část, zbytek může být paralelizován. Nezávisle na tom, kolik VE bychom měli k dispozici, nikdy nedosáhneme čas pod 1 hodinu. Tudíž pro $n \rightarrow \infty$:

$$\lim_{n \rightarrow \infty} S(n) = \lim_{n \rightarrow \infty} \frac{T(1)}{T(n)} = \frac{10}{1} = 10$$

Amdahlův zákon

- Příklad. Necht' $\alpha = 0,05$ (5%).

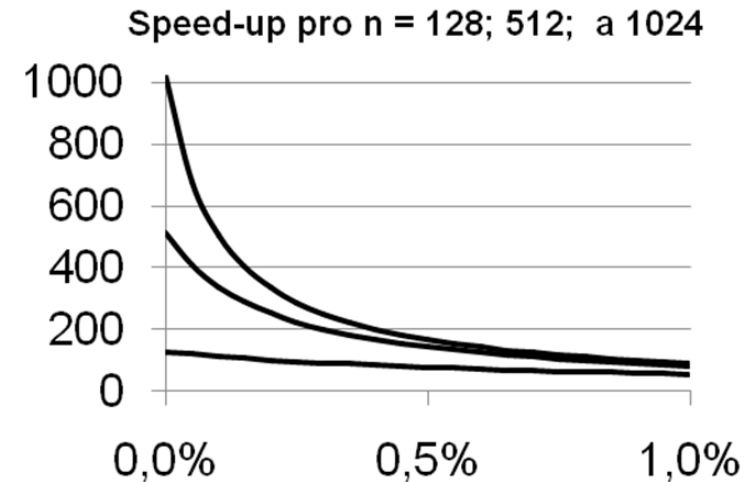
Pro $n = 5$: $S(5) = 4.17$

$n = 10$: $S(10) = 6.9$

$n = 100$: $S(100) = 16.9$

$n \rightarrow \infty$: $S(\infty) = 20$

Kolik VE by jste si koupili?



Amdahlův zákon

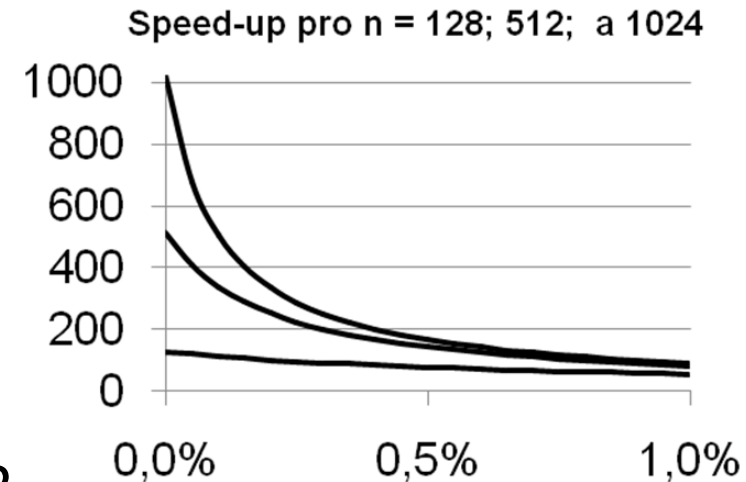
- Příklad. Necht' $\alpha = 0,05$ (5%).

Pro $n = 5$: $S(5) = 4.17$

$n = 10$: $S(10) = 6.9$

$n = 100$: $S(100) = 16.9$

$n \rightarrow \infty$: $S(\infty) = 20$



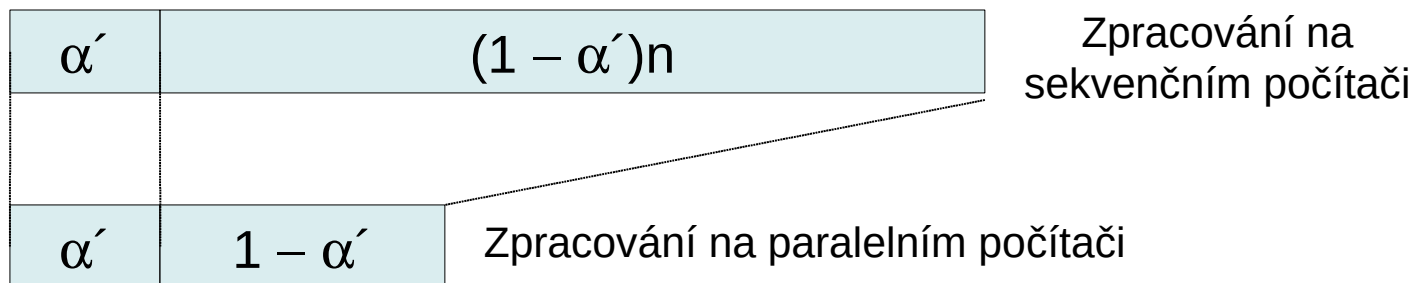
- Cray XT5-HE má však $n = 224\ 162$,
K computer $n = 548\ 352$.

77% nejvýkonnějších systémů má nad 4k VE.

Je takto velký počet jader efektivně využitelný? / Pro jaký typ problémů jsou tyto systémy určeny?

Gustafsonův zákon

- Příklad: Máme paralelní počítač (n VE). Výpočet trvá 10 hodin, z toho 1 hodina je neparalelizovatelná část, zbytek je paralelizován. Kdybychom tu samou úlohu řešili na sekvenčním počítači pak by výpočet trval $T(1) = 1 + n \cdot 9$ hodin. Tudíž je $S(n) = (1 + n \cdot 9) / 10$; a tedy $S(n \rightarrow \infty) = \infty$.



$$S(n) = \frac{T(1)}{T(n)} = \frac{\alpha' + (1 - \alpha')n}{\alpha' + 1 - \alpha'} = n - (n - 1)\alpha'$$

$$\alpha' \rightarrow 0 \Rightarrow S(n) \rightarrow n$$

Gustafsonův zákon

- Příklad. Necht' $\alpha' = 0,05$ (5%).

Pro $n = 5$: $S(5) = 4.8$

$n = 10$: $S(10) = 9.55$

$n = 100$: $S(100) = 95.05$

$n \rightarrow \infty$: $S(\infty) = \infty$

Shrnutí:

- Amdahl: Jak dlouho bude trvat zpracování sekvenční části programu na sekvenčním počítači, tak dlouho bude trvat zpracování sekvenční části programu na paralelním počítači.
- Gustafson: Jak dlouho bude trvat zpracování sekvenční části programu na paralelním počítači, tak dlouho bude trvat zpracování sekvenční části programu na sekvenčním počítači.

V čem se tedy liší? α není totéž co α' ; G. zákon předpokládá měnící se velikost problému s měnícím se výpočetním prostředím

Amdahl vs. Gustafson

- born November 16, 1922
- Amdahl, G.M., Validity of single-processor approach to achieving large-scale computing capability, *Proceedings of AFIPS Conference*, Reston, VA. 1967. pp. 483-485.



- born January 19, 1955
- Gustafson, J.L., Reevaluating Amdahl's Law, *CACM*, 31(5), 1988. pp. 532-533.



[1] http://en.wikipedia.org/wiki/Gene_Amdahl

[2] http://en.wikipedia.org/wiki/John_Gustafson_%28scientist%29

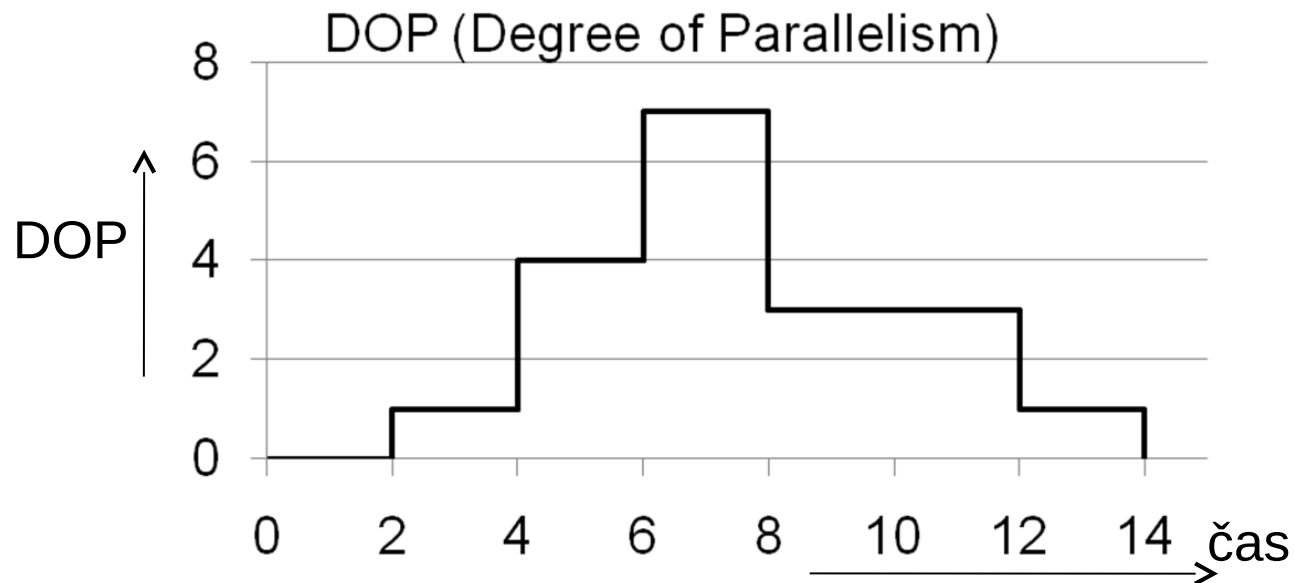
- **Příklad**

Máme možnost vyměnit stávající procesor (VE) za nový, 10 krát výkonnější. Současný procesor je vytížen na 50%, zbytek času (50%) čeká na I/O operace. Jaké celkové zvýšení výkonu by přinesla výměna procesoru?

- **Příklad**

Předpokládejme, že jsme dosáhli zrychlení 80x na systému se 100 procesory. Jaká část (procento) programu je sekvenční? Kolik procent času trvá zpracování sekvenční části programu na tomto systému? Jsou tyto procenta stejná?

- Stupeň paralelizmu
definován počtem VE používaných na vykonávání programu v
daném časovém okamžiku



$$DOP_{STR} = \frac{\sum_i DOP_i \cdot \Delta t_i}{\sum_i \Delta t_i} = \frac{1}{t_n - t_0} \sum_i DOP_i \cdot \Delta t_i = \frac{1}{14 - 2} (1 \times 2 + 4 \times 2 + 7 \times 2 + \dots)$$

- Srovnávání výkonu – který počítač je nejlepší?

	počítač A	počítač B	počítač C
program 1	12 min	1 hod	1 hod
program 2	20 min	40 min	10 min
program 3	5 hod	1 hod	2 hod

- Čas vykonání programu:

$$T = IC \cdot CPI \cdot T_{CLK}$$

IC – počet instrukcí programu (Instruction Count)

CPI – počet cyklů na jednu instrukci (Cycles per Instruction)

T_{CLK} – perioda hod. signálu (doba cyklu)

Pro danou ISA je návrh procesoru zejména optimalizací CPI a T_{CLK} .

- MIPS (Milion Instructions per second) kde $IPS = IC / T = IPC_{str} \cdot f_{CLK}$,
MOPS (Operations), MFLOPS (Floating point Operations) - vše
relativně k ISA, závislé od samotného programu...

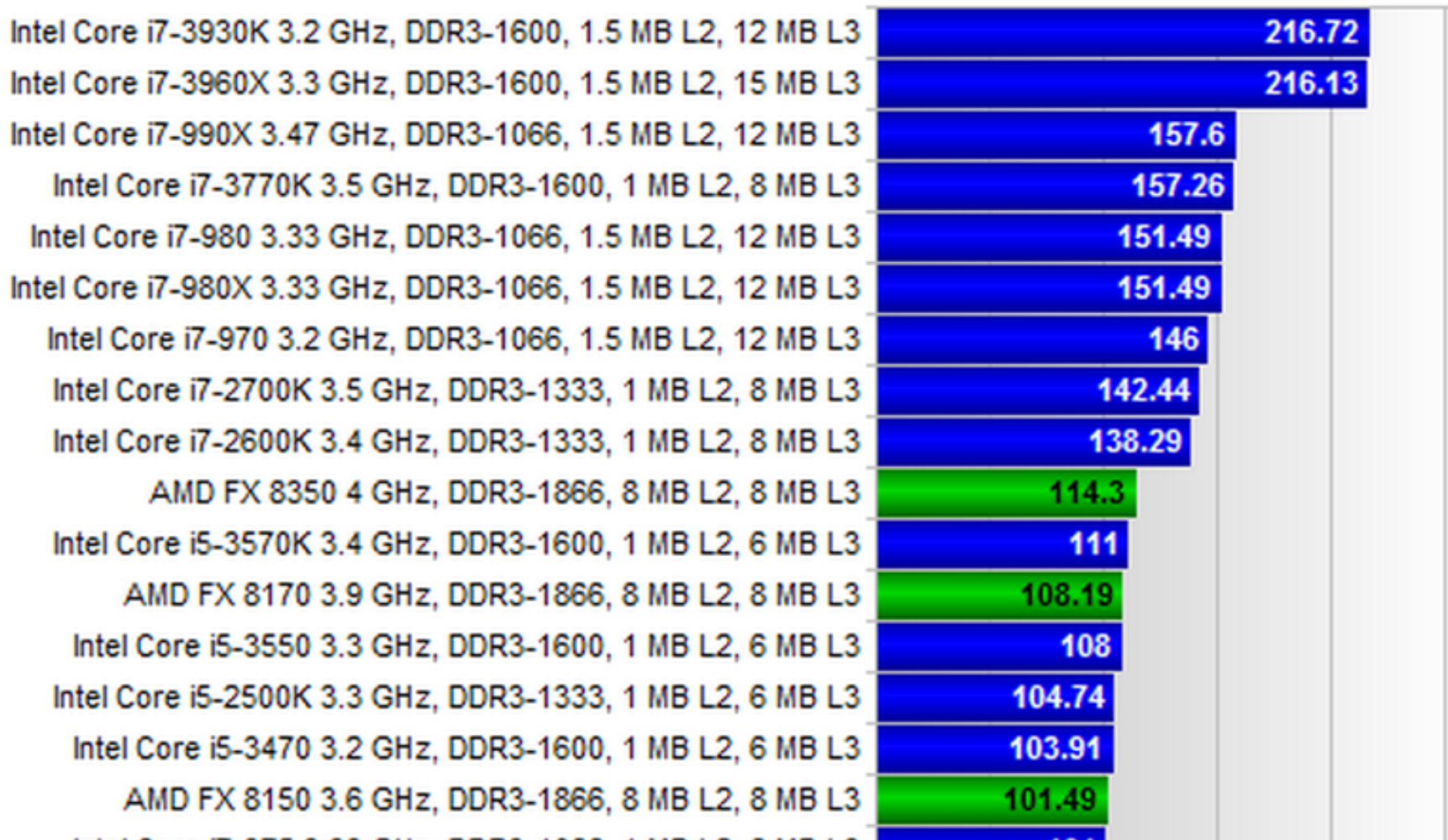
- Programová propustnost:

$$W = 1 / T = IPC \cdot f_{CLK} / IC$$

IPC – Instructions per Clock

Výkonové metriky

ALU Performance
Dhrystone GIPS



Příklad

Procesor o frekvenci 2 GHz je použit na vykonání programu s následujícím mixem instrukcí a počtem hodinových cyklů. Určete střední hodnotu CPI, rychlost vykonávání programu MIPS, čas vykonání programu T, a programovou propustnost Wp.

Typ instrukce	Počet instrukcí	Počet hod. cyklů
Celočíselní aritmetika	45000	1
Přenos údajů	32000	2
Pohyblivá řádová čárka	15000	3
Přenos řízení(skoky)	8000	2

Značení

- Perioda hodinových cyklů: T_{CLK}
- Frekvence hodinových cyklů: f
- Velikost programu (počet instrukcí v programu): IC
- Celkový počet cyklů na vykonání programu: C

Nechť množina $\{ R_i \}$ jsou vykonávací rychlosti programů
 $i = 1, 2, \dots, m$ měřeny v MIPS (MFLOPS), resp. IPS (FLOPS)

- Střední aritmetický výkon: $R_a = \sum_{i=1}^m \frac{R_i}{m} = \frac{1}{m} \sum_{i=1}^m R_i$

R_a je rovnoměrně váhován ($1/m$) ve všech programech a je úměrný součtu IPC, avšak ne součtu vykonávacích časů (nepřímo úměrně). Proto střední aritmetický výkon selhává...

$$\begin{aligned} R_a &= \frac{1}{2} (R_1 + R_2) = \frac{1}{2} \left(\frac{IC_1}{T_1} + \frac{IC_2}{T_2} \right) = \frac{1}{2} \left(\frac{IC_1}{IC_1 \cdot CPI_1 T_{CLK}} + \frac{IC_2}{IC_2 \cdot CPI_2 T_{CLK}} \right) = \\ &= \frac{1}{T_{CLK}} \left(\frac{IPC_1 + IPC_2}{2} \right) = \frac{1}{T_{CLK}} \left(\frac{IC_1}{2C_1} + \frac{IC_2}{2C_2} \right) \quad \text{avšak} \quad IPC_{1,2} = \frac{IC_1 + IC_2}{C_1 + C_2} \end{aligned}$$

Pokud však $C_1 = C_2$ (stejný celkový počet cyklů; tj. při téže frekvenci oba programy běží stejně dlouho) je R_a použitelný

- Střední geometrický výkon:
$$R_g = \prod_{i=1}^m R_i^{\frac{1}{m}}$$

Nesumarizuje reálný výkon, nemá inverzní relaci k celkovému času.
Pro porovnávání s normalizovanými údaji vzhledem na referenční stroj.

- Střední harmonický výkon:
$$R_h = \frac{m}{\sum_{i=1}^m \frac{1}{R_i}}$$

$$R_h = \frac{2}{\frac{1}{R_1} + \frac{1}{R_2}} = \dots = \frac{1}{T_{CLK}} \left(\frac{2}{CPI_1 + CPI_2} \right) = \frac{1}{T_{CLK}} \frac{2IC_1IC_2}{C_1IC_2 + C_2IC_1}$$

Pokud však $IC_1 = IC_2$ (oba programy jsou stejně velké) je R_h použitelný

- Existují taktéž vážené verze těchto výkonů...

Cílem je maximalizovat programovou propustnost

$$W = 1 / T = \text{IPC} \cdot f_{\text{CLK}} / \text{IC}$$

Technologie

- Rychlost šíření elektrických signálů shora omezena rychlostí světla (při 3,6 GHz je perioda 0,278 ns => 8,3 cm) (zpoždění propojení dominuje nad zpožděním hradel)
- Dnes již 32 nm CMOS technologie (atomový poloměr atomu křemíku je 0,111 nm)

Architektura (Hardware + Organizace HW + ISA)

- paralelizmus podporovaný hardwarem na nejnižší úrovni (bit-level parallelism),
- instrukční paralelizmus (zřetězení, superskalární a spekulativní vykonávání, vykonávání mimo pořadí,...),
- paralelizmus na úrovni vláken,
- datový paralelizmus (viz loop-level parallelism),
- paralelizmus na úrovni úloh (funkční al. řídicí paralelizmus)