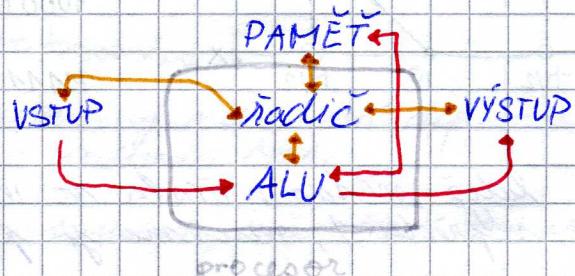


# Architektura počítačů

- CPU = Processor (Central Processing Unit)
  - přijímá od paměti instrukce a dat
  - decoding, executing programs instructions, performing calculations
- RAM (Random access memory) - paměť s libovolným přístupem
  - temporarily store dynamic data to enhance computer performance
  - dělí se na — DRAM (dynamická)  
SRAM (stacionární)
- BIOS (Basic Input / Output system)
  - interface between the operating system and hardware
- Complementary Metal Oxide Semiconductor RAM (CMOS RAM)
  - is kept alive by battery when PC is off  
→ store basics about PC configuration, date, time..
- GPU (Graphic Processing Unit)
  - graphical chips manage what's going on the display
- HDD (Hard disk drive)
  - dlouhodobě ukládání množství dat
- North bridge
  - systémový řadič, spolu se southbridge tvorí čipset
  - zajišťuje komunikaci mezi CPU, RAM, PC sběrnici a spojuje se southbridge
- South bridge
  - není přímo spojen s CPU (přes north bridge)
  - kontroluje I/O funkce
    - ↳ USB, audio, serial, BIOS...

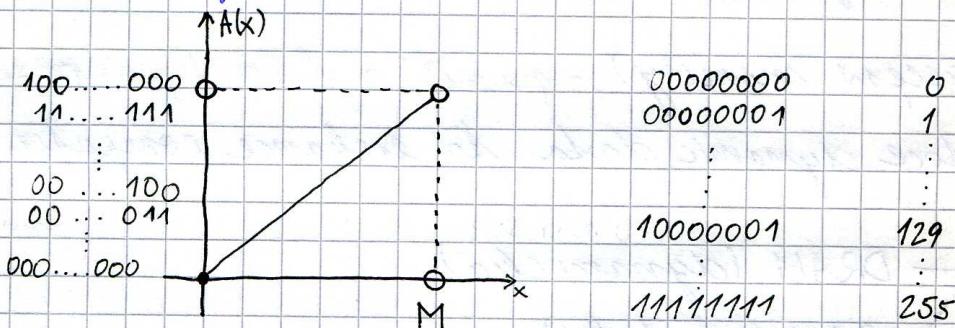


- MC (memory controller)

- spojuje Northbridge a RAM
- operace čtení a zápisu do paměti + refreshing

- Unsigned int

- např. když máme 8-bitové číslo, pak v rozsahu  $\langle 0, 255 \rangle$

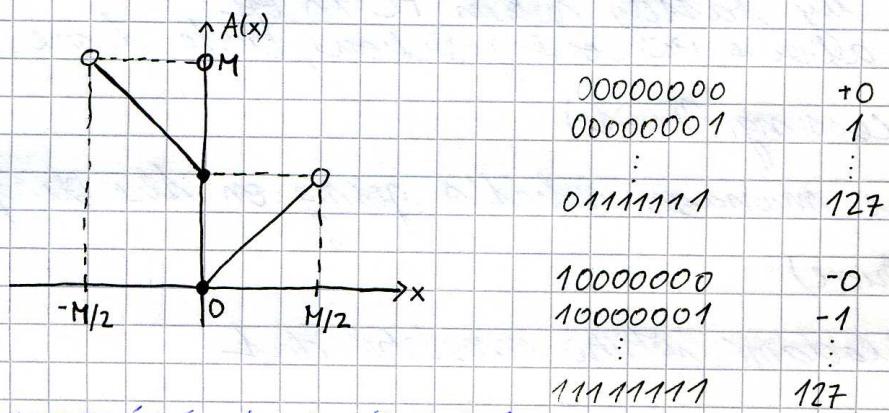


- Signed int

- první bit je znaménko ( $0 = „+“$ ,  $1 = „-“$ )

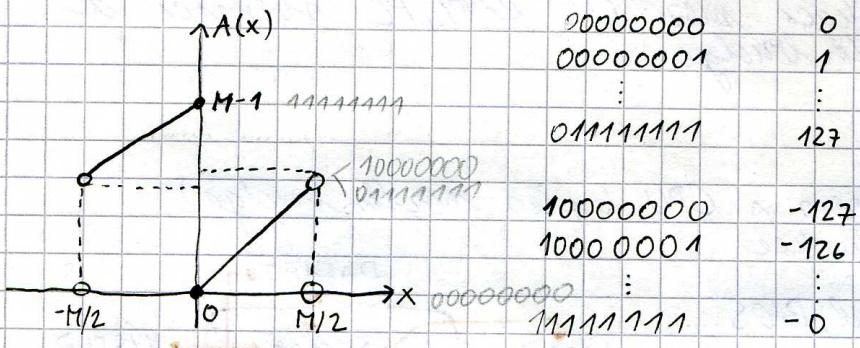
- když máme 8 bitů  $\langle -127, 127 \rangle$

$N$  bitů  $\langle -2^{N-1} + 1, 2^{N-1} - 1 \rangle$



- INVERZNÍ KÓD (jedničkový doplněk) - jen pro záporná

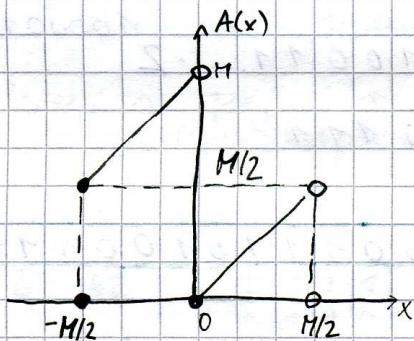
- $6 (0110) \rightarrow -6 (1001)$  ... bitová negace



pozn. když sečteme čísla v invertním kódu a dostaneme reprezentaci 1, musíme ji přidat na konec.

- u dvojkovku ne.

- DVOJKOVÝ DOPLNĚK - jen pro záporná
  - nejvhodnější varianta než inversní kód (úspora v HW)
  - $6(0110) \rightarrow -6(1010)$
  - přičtení 1 k nejménšemu bitu záporného čísla  
v invertivním kódu
  - postup
    - $6: 0110$
    - $\text{inv } 6: 1001 \ (-6)$
    - $\text{dd } 6: 1010 \ (-6)$
  - součet čísla s jeho dvojkovým doplňkem je 0



00000000	0
00000001	1
⋮	⋮
01111111	127
10000000	-128
10000001	-127
⋮	⋮
11111111	-1

↳ preplnení  
↳ rozsahu

- ADITIVNÍ KOD (s posunem tří nulou)
    - jiná možnost pro zobrazování čísel se známkem
    - k číslem přičítáme nejdešou známku nulou a začínáme

• Cislo typu **REAL** (mantissa x rámeček exponent)

$$+\infty \quad 0 \quad 1 \dots 1 \quad 0 \dots \dots \dots 0$$

$$-\infty \quad 1 \quad 1 \dots 1 \quad 0 \quad \dots \quad 0$$

+0.0 0 0...0 0...:...0

$$-0.0 \quad 1 \underbrace{0 \dots 0}_{8} \quad 0 \dots \dots \dots 0$$

→ znaménko      8-bit exponent      mantissa



pozn. Nan (Noz-a-number)

- polom je výsledek nejednoznačný nebo není definován pro daný vstup je uložena hodnota NaN

→ exponent nastavuje na same -1, mantissa nemá vliv

Pr.

Exp. bias 127

263.3

binary

1) 263 : 0001 0000 0111

$0,3 : 0100110011\dots$

$\frac{0,6}{0,1}$

1,2

0,4

:

desetinná čísla

0 8 mist

normalize

2)  $1,00000111 \times 0100110011\dots \cdot 2^8$

mantissa

exp → bin

3)  $1,000001110100110011\dots \cdot 2$

10000111\*

$$\cdot 127 + 8 = 135 \rightarrow 10000111$$

Rешение:

0 10000111 0000011101001100...

Pr

$$\begin{array}{r}
 0 10000110 \\
 + \underbrace{134}_{\text{exp } 2^7} \\
 \hline
 \end{array}
 \rightarrow \boxed{11010100} = 212$$

složka 1

Operace

- srovnání dvou čísel

- obrazy A a B se odečtou a zjistí se, zda je výsledek něž nebo roven 0

- sčítání

1) odečteme exponenty

2) mantisu čísla s menším exponentem posuneme doprava o počet bitů, který je roven rozdílu exponentů (o n bitů)

3) sčteme mantisy obou čísel

4) uříme počet nul před první 1

5) zmensíme prvního exponentu o počet malezujících nul

- Big x little Endian bite order

- the most significant bit is placed at the bit with the lowest address (BIG)

0x54237618 → 0x54 0x23 0x76 0x18

Př

$$7 = 0111$$

$$6 = 0110$$

$$-6 = 1010 \quad \text{...dvojicový doplněk}$$

$$\underline{7 + 6} = 13$$

$$\begin{array}{r} 0111 \\ 0110 \\ \hline 1101 = 13 \end{array}$$

$$\underline{7 - 6} = 1$$

$$\begin{array}{r} 0111 \\ 1010 \\ \hline 11001 = 1 \end{array}$$

přelečení

- přelečení

→ když ještě všechny původní používají rozsah

- např. při sčítání 2 kladujících nebo dvou záporných hodnot u dvojicového doplnku

přelečení se projeví r. tam, kdy součet 2 kladujících čísel postupně vezme odpovídající záporné hodnoty

$$7 \cdot 6 = 42$$

$$\begin{array}{r} 0111 \\ 0110 \\ \hline 0000 \\ 0111 \\ 0111 \\ \hline 0000 \\ \hline 0101010 = 42 \end{array}$$

Př

$$\cdot 228 = 11100100 = 1,1100100 \cdot 2^7$$

$$\text{exp} = 127 + 7 = 134 = 10000110$$

řešení: 0 10000110 1100100...00

$$\cdot 3,5 = 0011,1000... = 1,11000... \cdot 2^1$$

$$\text{exp} = 127 + 1 = 128 = 1000000000$$

řešení: 0 10000000 11000...00

pozn. denormalizované čísla

- velmi blízko 0 → není se skrylá 1

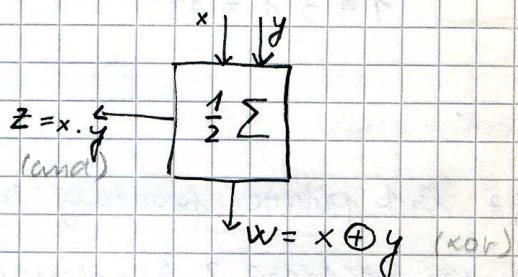
např. 0,011

# PŘÍZNAKOVÝ REGISTR

- příznaky Carry a Overflow
- informuje o správnosti výsledku

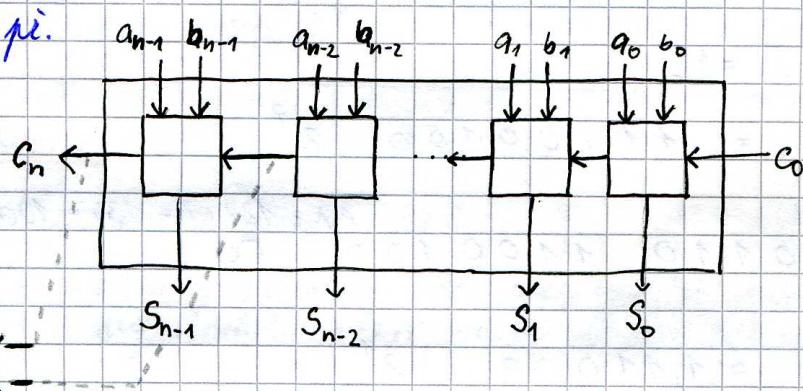
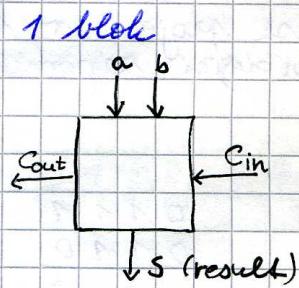
## 1) Polovinová sčítací

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



## 2) Celková sčítací

A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	01



• provice dílovou čísla velmi pomala

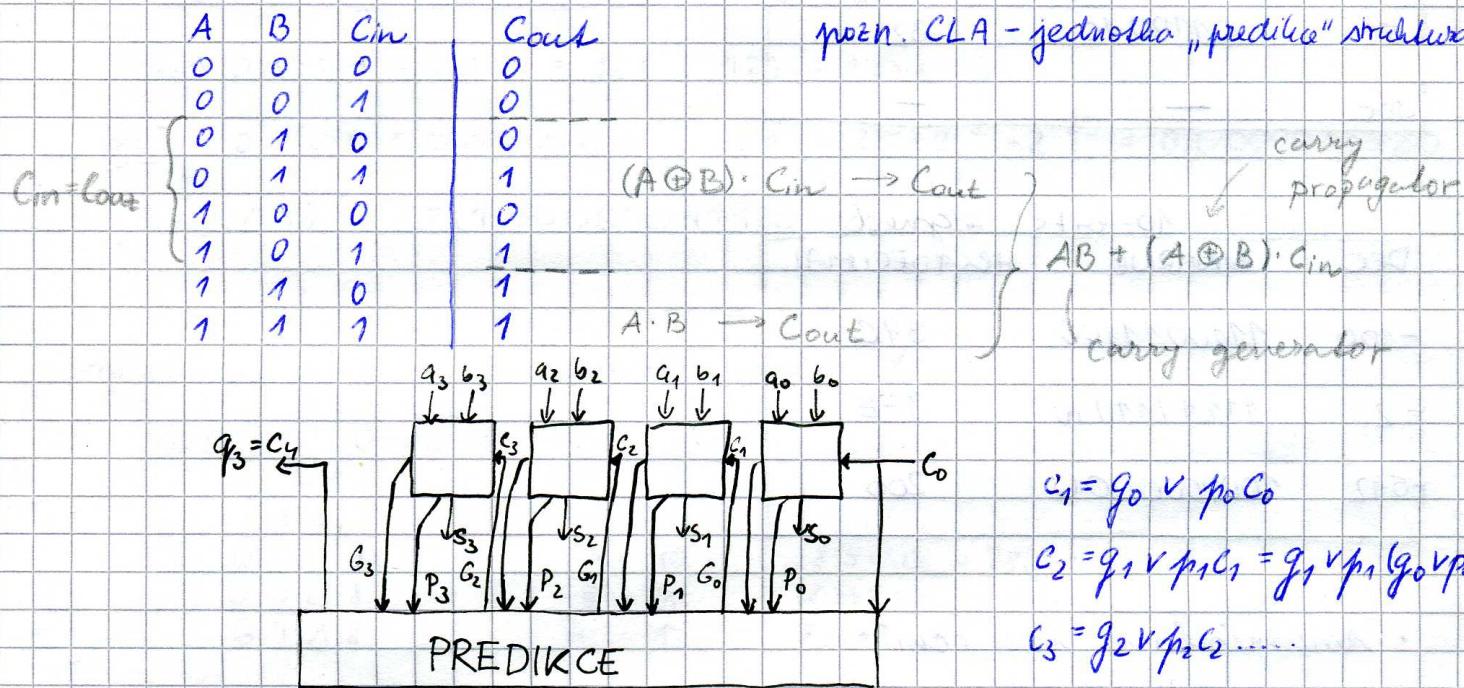
At. a pozdění jednotek hrádla

→ pro generování součtu dvou 2Nst

např. pro 64-dílovou sčítací doje 128 st

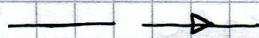
### 3) scitacia s „predikou“ prenosu (Carry Look - ahead)

- wychljuje výpočet prínosu sum, ře predpovídá hodnotu Carry
- pro výpočet Carry nejprve potřebujeme vypočítat Sum



#### znacení

WIRE / BUFFER



INV (NOT)



$X = 0$

$X = 1$

$F = 1$

$F = 0$

AND



platí oba

OR



platí alespoň jedno

XOR



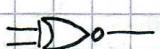
jsou rozdílná

NOR



polohu oba neplatí

EQU



jsou stejné

NAND



neplatí obě

# TEST

8-bits unsigned

DEC	BINÁRNĚ	HEXADECIMAL
100	0110 0100	64
150	1001 0110	96
50	0011 0010	32
300	—	—

10-bits signed

DEC	BINÁRNĚ	HEXADECIMAL
-100	1110011100	39C
-2	1111111110	3FE
-512	1000000000	200

- Anaměnlové rozšíření

- přidání bitu před číslo

- ↳ nemzaměnlová - přidávají se muly

- ↳ zaměnlová - nejvyšší bit čísla

- po každém sčítání a odčítání binárních čísel ALU nastaví 4 základní aritmetické příznaky (modem dobu, zda situace nastala)

- Carry - když je ztracený nejvyšší bit

- Overflow - přetížení pro signed

- Sign - kopie nejvyššího bitu čísla

- Zero - poslední následek obsahuje všechny bity 0

- hexadecimální interpretujeme jako unsigned

- logical left shift

- když přidáváme muly napravo takže je jeho posunutí 2

- Word - malivná délka daného počítací

- byte = Octet , 1/2 byte = 4 bits = nibble

12x 0

$$\cdot A = \underbrace{1000}_{1} \underbrace{1000}_{1} \underbrace{1000}_{1} \underbrace{1000}_{1} \underbrace{1000}_{1} \dots 000 \dots 00$$

$$\exp = 00010001 = 17 \quad 17 - 127 = -110$$

$$m = 000100010000 \dots 00 = 2^{-4} + 2^{-8} = 0,06640625$$

$$A = (1 + 0,06640625) \cdot 2^{-110} = (1,06640625) \cdot 2^{-110} = -8,2159 \cdot 10^{-34}$$

$$\cdot B = \underbrace{0011}_{1} \underbrace{1101}_{1} \underbrace{1010}_{1} \underbrace{0100}_{1} \underbrace{000 \dots 01}_{1}$$

$$\exp = 01111011 = 123 \quad 123 - 127 = -4$$

$$m = 0100100000 \dots 01 = 2^{-2} + 2^{-5} + 2^{-23} = 0,0000011920$$

$$B = (1 + m) \cdot 2^{-4} = (1,00000011920) \cdot 2^{-4}$$

### multiplication in hexadecimal base

$$0x0C35 \cdot 0x0008$$

$$\begin{array}{r}
 & \begin{smallmatrix} 1 & 2 \\ 0 & C & 3 & 5 \\ 0 & 0 & 0 & 8 \\ \hline 6 & 1 & A & 8 \end{smallmatrix} \\
 & \begin{array}{l} A = 10 \\ B = 11 \\ C = 12 \end{array} & \begin{array}{l} 8 \cdot 5 = 40_{10} = 28_{16} \\ (8 \cdot 3) \cdot 2 = 24_{10} = 1A_{16} \\ 8 \cdot C + 1 = 97_{10} = 61_{16} \end{array}
 \end{array}$$

### SEKVENČNÍ NAŠOBICKA

$$\begin{aligned}
 x &= 110 \dots \text{masobenec} \\
 y &= 101 \dots \text{masobitel}
 \end{aligned}$$

i	AC	MQ	A
0	000	101	110
1	110	101	-
2	011	010	-
3	001	001	-
4	111	101	-
5	011	110	-

první nastavení

záčátek cyklu,  $AC = AC + MB$

shift right

nic  $\rightarrow MQ = 0$

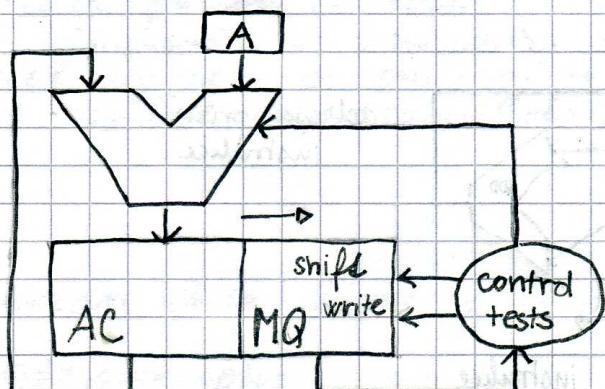
shift right

$AC = AC + MB$

if ( $MQ_0 \neq 1$ )

koniec cyklu

$MQ_0 \dots$  nejvýšší bit MQ



$$x \cdot y = 110 \cdot 101 = 11110$$

# HW DĚLÍČKA

	0 0 1 0 0 0
1 0 1	1 0 1 0 1 0
1 0	
1 0 1	
0 0 0 0	
0 0 0 0 1	
0 0 0 0 1 0	

101 → dělitel  
101010 → dělenec

$$42 / 5 = 8 \text{ zb. } 2$$

1 1 0	1 0 1 0 1 0	= 0 0 0 1 0 1
1 0	↓	
1 0 1	↓	
1 0 1 0	↓	
- 1 1 0	↓	
1 0 0 1		
- 1 1 0	↓	
0 1 1 0		

## POČÍTAČ DLE VON NEUMANNA

- Radíč - řídí činnost procesoru
- Alu
- Paměť
- vstup/výstup - I/O

dataová část  
- registry, další potřebné obvody  
jádro

PC (Program Counter)  
IR (Instruction Register)

## Základní cyklus počítací (instrukční cyklus)

1) počáteční nastavení (do Program Counteru se nastaví nějaká hodnota)

2) čtení instrukce

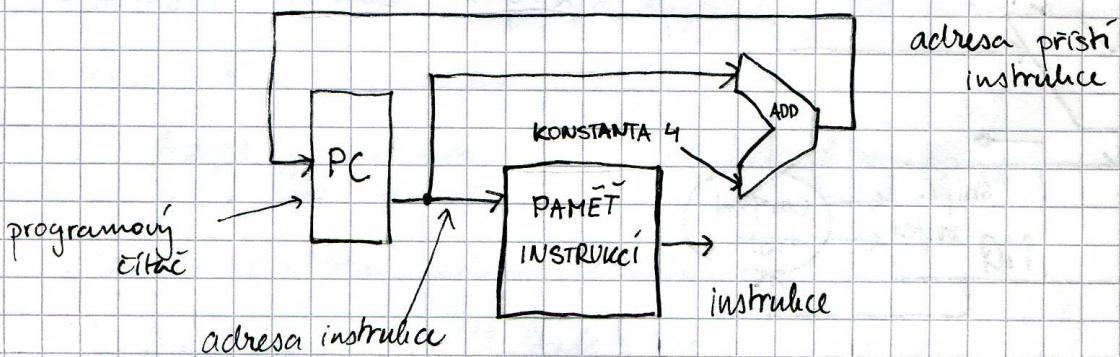
PC → adresa hl. paměti → čtení obsahu → Instruction Register

3) Dešifrování instrukce

4) provedení operace (včetně vyhodnocení efektivních adres, čtení operandů)

5) Možné přenosy (dolazy)

6) opakování bodu 2 (pokud něco doleží k přenosu)



## OTÁZKY

1) Za jakou dobu provede procesor s takto vysokou frekvencí  $16\text{ GHz} = 10^{-9}\text{ s}$ , 1000 instrukcí za předpokladu, že jedna instrukce provede na jeden dat?

$$t = \frac{1}{10^9} \cdot 1000 = 10^{-9} \cdot 1000 = 1\text{ ms} \Rightarrow 1000\text{ ms} = \underline{\underline{1\mu\text{s}}} \quad \left( \frac{1000}{10^9} \right)$$

2) Za jak dlouho provede tento procesor 1000 instrukcí za předpokladu že 20% instrukcí jsou přístupy do paměti. Jeden přístup do paměti trvá 70 ms.

$$800 \cdot 1\text{ ms} + 200 \cdot 70\text{ ms} = 0,8\mu\text{s} + 0,2 \cdot 70\mu\text{s} = 0,8\mu\text{s} + 14\mu\text{s} = \underline{\underline{14,8\mu\text{s}}}$$

3) Jak se změní doba potřebná ke vykonání instrukce, když nahradíme procesor 2GHz verzi, ale paměti zůstanou stejné?

$$t = \frac{1}{2} \cdot 10^{-9} \text{ s} = 0,5 \cdot 10^{-9} \text{ s} = 0,5\text{ ns} \quad (\text{na instrukci})$$

$$\rightarrow 800 \cdot 0,5\text{ ns} + 200 \cdot 70\text{ ns} = 0,4\mu\text{s} + 14\mu\text{s} = \underline{\underline{14,4\mu\text{s}}}$$

4) Jak se změní doba, když používáme původní 1GHz procesor, ale přidáme do systému cache? Tato cache má hit rade 80% a doba načtení požadovanu v cache je 5 ms.

$$\Delta t = 800 \cdot 1\text{ ms} + 160 \cdot 5\text{ ms} + 40 \cdot 5\text{ ns} + \cancel{40 \cdot 70\text{ ns}} = 0,8\mu\text{s} + 0,8\mu\text{s} + 3\mu\text{s} = \underline{\underline{4,6\mu\text{s}}}$$

## Register

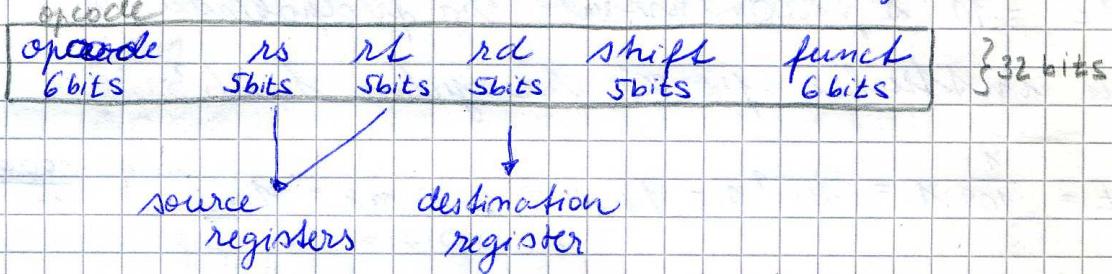
- malí a rychle uložití dat, které využívá procesor při své činnosti
- počet registrů je omezený → nepotřebná data & registrů jsou můžit kopirovány do operacní paměti

## Cache

- uložování data, čímž je následný přístup k nim vychýšší
- blízko procesorem

### 3 ZÁKLADNÍ TYPY INSTRUKcí

- R all the data values are located in registers



for example

- add, and, div, j (jump to address in register), xor, or, sll (set to 1 if less than), sllt (l. shift left), srl, sub...

- I instructions operate on an immediate register value

opcode	rs	rt	IMM-immediate	?	32 bits max
6 bits	5 bits	5 bits	16 bits		

for example

- addi, beq (branch if equal), bne (branch if not equal), lbu (load byte unsigned), lw (load word), ori, sb (store byte), sw (store word)

- J jumps

opcode	pseudo-address
6 bits	26 bits

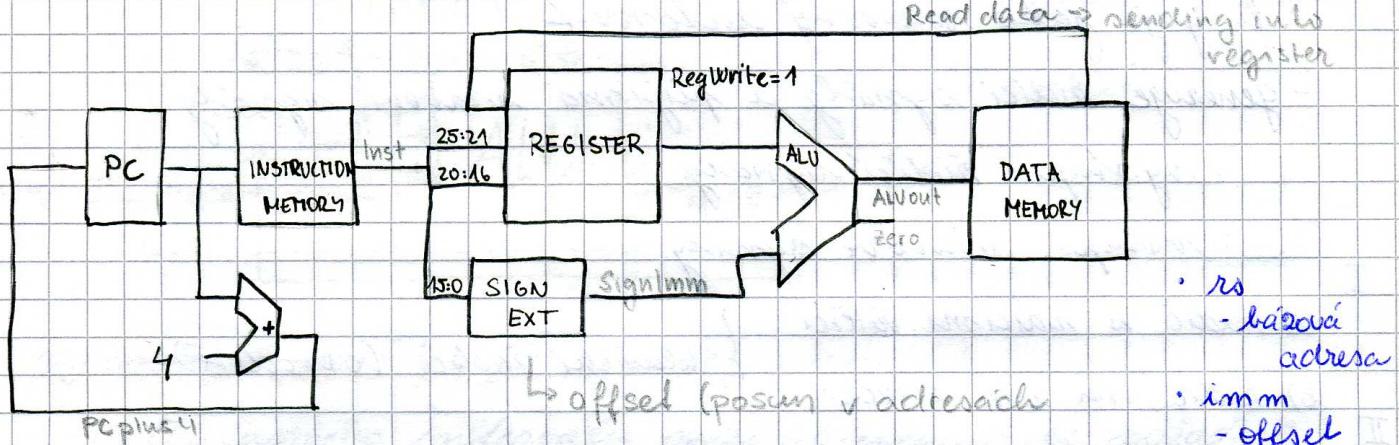
for example

- j (jump to address), jal (jump and link)

#### TABULKA

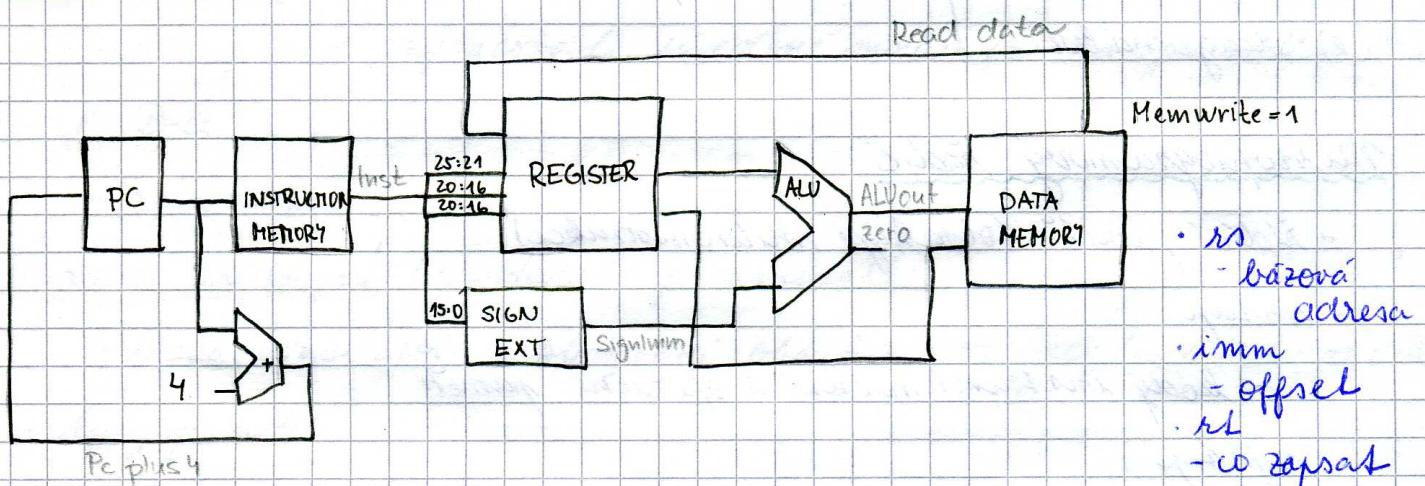
	ALUcontrol	RegWrite	MemWrite	ALUSrc	RegDst	MemtoReg
lw	+	1	0	1	0	1
sw	+	0	1	1	0	x
add	+	1	0	0	1	0
sub	-	1	0	0	1	0
and	&					
beq	-	0	0	0	x	x

LW stores data into registers



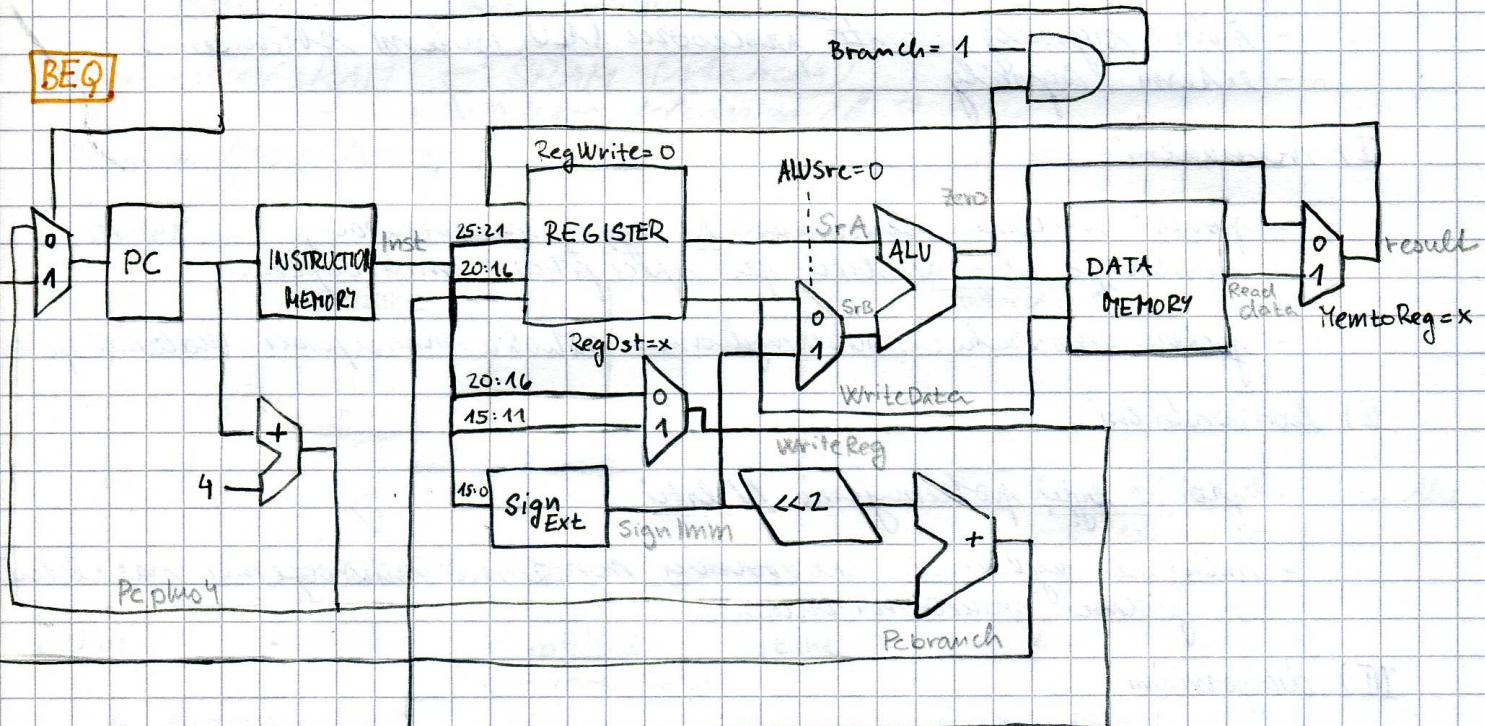
- rs
  - bázová adresa
- imm
  - offset
- rt
  - kde uložit

SW



- rs
  - bázová adresa
- imm
  - offset
- rt
  - co zapsat

BEQ



maximální frekvence procesoru

~ 980 000 instrukcí za sekundu ( $f_{CK} = 980 \text{ kHz}$ )

## ŘADÍC JEDNOCYKLOVÉHO PROCESORU (Control Unit)

- funguje jako stavový automat
- generuje řídící signály a přijima stavové signály
  - výstup: řídící signály
  - vstup: stavové signály

I.  
 a) řadíč s řídicimi režimy }  
 b) řadíč má liži čitací } klasické řadíče (obvodové)

II.  
 c) horizontální }  
 d) vertikální } řízené mikroprogramem (mikroprogramové)  
 e) diagonální }

### Mikroprogramový řadíč

- řídící paměť (obsahuje mikroinstrukce)
- vstup
  - bloky instrukcí načtené z operacní paměti
- výstup
  - řídící signály uvnitř procesoru (ALU, interní sběrnice...)
  - externí signály

#### I) vertikální

- jedna instrukce vykoná N typů mikrooperací, poslání je log.  $N$  bitů pro specifikaci mikrooperace
- jedna instrukce může vykonať mnoho mikrooperací (kódování poslání)

#### II) horizontální

- pro  $N$  typů potřebujeme  $N$  bitů
- můžeme vykonat libovolnou množinu mikrooperací paralelně v jedné mikroinstrukci

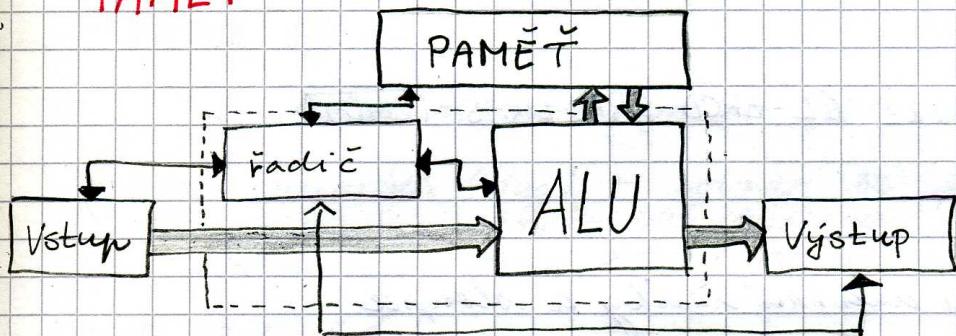
#### III) diagonální

- kompromis → některé bloky vertikálně a některé horizontálně

→ jedna se vlastní o počítací v počítaci

přev. pomalý; flexibilní; neefektivní pro výšená procesory

# PAMĚŤ



• 5 funkcícních jednotek

- vyhavarací doba paměti (kritický parametr)

- délka časového intervalu mezi objevením se požadavku a dobou, kdy jsou data k dispozici

- propustnost (vyhavarový parametr)

- schopnost spracovat uvedené množství na jednotku času

- latence

- spoždění

- doba přístupu (kastaralý parametr)

- vyhavarací doba + obnovení obsahu po destruktivním čtení

- typy paměti

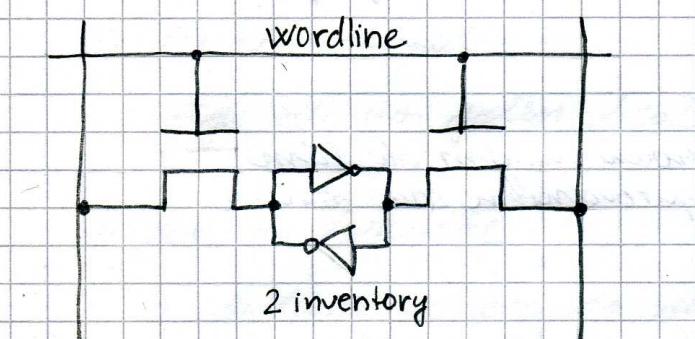
- RWM (RAM), ROM, FLASH

- provedení RAM

- SRAM (statická)
- DRAM (dynamická)

Random Access Memory

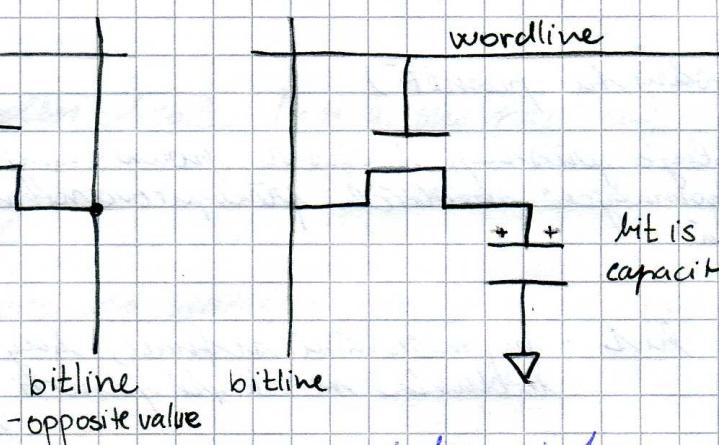
SRAM



6 transistors

- 2 transistors in cache inverter
- 4 transistors in the circuit

DRAM



1 transistors

slower, less space

- faster, more space

## parametry paměti

DDR<sub>x</sub>-yyyy / PC<sub>x</sub>-zzzz CL-ERCD-ERP-TRAS-(CMD)

CL... latency

ERCD... čas nutný mezi aktivací rácky a sloupcem

ERP... čas mezi uzavřením jednotky rácky a aktivací nového

TRAS... jak rácky musí cekat, než může být iniciován další přístup do paměti

CMD... čas mezi aktivací paměti a prvním možným příkazem

## paměťová hierarchie



- jedna až následující informace se může objevit na více místech hierarchické paměti
  - majdeme dle adresy
  - hledat začneme vzdáleji od procesoru (nejrychlejší vpravo)

## CACHE (skrytá paměť)

- rychluje přístup k často používaným datům  
ná „malých“ mědiček překoprovádění na mědičce rychlá

- cache hit - pojmenování situace, kdy požadovaná hodnota ve skryté paměti je

- cache miss - hodnota ve cache není

- cache line / cache block - soubor mimořádně jednotek mezi hierarchicky rozdílnými

- Block offset
  - to determine location in the blocks
- Block number (tag) - rovní se při cache hitu  
 ↳ jazyk blok doopravdy máme v cache
  - index odpovídající bloku v operační paměti (hodnota ukazatele)
- Validity bit
  - bit platnosti
  - indikuje, zda je obsah pole DATA níže platný
- Data
  - pole obsahující vlastní hodnoty na příslušných adresách
- Dirty bit
  - indikuje, že v cache již jiná hodnota než v hlavní paměti

### Typy cache (rychlý přehled)

$$HIT = (\text{tag} == \text{block number}) \& \text{validity}$$

#### 1) Fully associative

- obsahuje jen jeden set
  - stupň asociality je roven počtu bloků
  - adresa paměti se může mapovat kambolvit
- velmi druhá, používají se jené cache (prvmo mapování a s omezeným stupnem asociality)

#### 2) Direct-mapped

- je ude jen jeden blok (in a one particular line)

#### 3) Set-associative

- zadání adresy se rozdělí na dvě části

Memory :	tag	set	byte offset
Address			

- hit rate

- podíl počtu paměťových přístupů, které byly úspěšné při přístupu do téže části paměti hierarchie

- miss rate

- to samo, ale pro nesúspešný přístup

- miss penalty

- čas potřebný pro načtení bloku (čidají) z paměti méněj než jedna čtvrtina času

- average memory access time

- hit time + miss rate × miss penalty

pom. miss rate můžeme redukovat rozšířením velikosti bloku

↳ využití principu prostorové lokality

, což ale na druhou stranu znamená zvýšení počtu setí, což se projeví nárůstem miss rata

Pr.

8 bloků

adresa: 0xF0000014

Kam se umístit?

- průmo mapování
- s omezeným stupnem asociativity  $N=2$ :
- plné asociativní

Set = (adresa / 4) % (počet poliček)

Strategie uvolňování bloků tráče cache

- Náhodná (Random)

- vybere si libovolný blok

- LRU (Least recently used)

- musíme znát informace o posledním použití daného bloku

- LFU (Least Frequently used)

- ke každému bloku si pamatujeme informace o tom, jak často byl blok používán

- ARC (adaptive Replacement Cache)

- kombinace LRU a LFU

## Zápis dat procesorem do paměti

### monosidence dat

- samozřejmý požadavek na shodu obsahu stejných adres na různých médích

### write through

- současně se zápisem do cache se data kapiší do zápisové fronty a pak asynchronně do paměti

### write back

- data se do cache kapiší a poznamkou dirty

- ke shunckování kapišení dat do hlavní paměti dojde až v ohnisku případného nášení příslušného řádku cache, když krozi jí jízda abráska

+ další (write-combining, uncachable, write-protect)

## VIRTUÁLNÍ PAMĚŤ

- způsob správy operační paměti umožňující běžení mezi procesu souběžně paměťového prostoru, který je nezávislý jinak (nebo je doložice něčí) než fyzická paměť
- v současné běžných operačních systémech je virtuální paměť implementována pomocí virtuální paměti spolu se sloučením váním na disk, které rozšiřuje operační paměť o prostor na disku

VA - virtuální adresa → mapování → FA - fyzická adresa

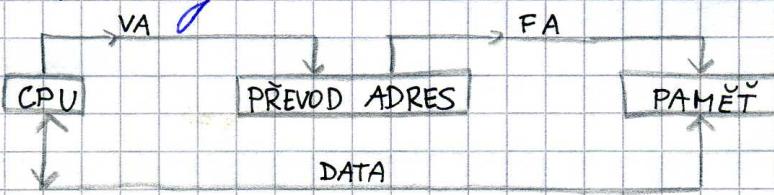
- jednotkovou mapováním jsou sloučeny typicky  $4Kb = 12$  bitů

↳ půjčování jednotlivým běžicím procesům

- každé virtuální stranice může odpovídat nejvýše jedné fyzické stranice, obecně lze neplatit

→ na jednu konkrétní fyzickou stranici může být namapováno více virtuálních stranek

⇒ můžeme sdílet paměť např. různými procesy nebo různými



## převod adres

- page table (tabulka stranek)
- mapovací funkce se nejčastěji implementuje
  - Look-up Table (vyhledávání tabulkou)
- O překlad virtuálních adres na fyzické se stará MMU (Memory Management Unit)
  - ↳ součástí CPU

## stránka

- stránka je typicky  $4\text{ kB} = 2^{12}$
- potud budeme mít adresu stránky, posloužíme si s ní sedy jinom 12 bitů na polohu (adresaci) v ní.  
→ rozloží 20 bitů (pro 32-bitovou adresu)
- Page Directory (Page Table) by měla obsahovat  $2^{20}$  položek  
→ nepraktické a nevhodné  
⇒ nás - využívají stránkovací

## SROVNÁNÍ VIRTUÁLNÍ PAMĚT × CACHE

VP	CACHE
stránka	blok / řádka
page fault	read / write miss
velikost stránek	velikost bloku
512 B - 8 kB	8 - 128 B
plen. asociativní	n-cestná, plen. asociativní
LRU (nejběž oběti)	LRU, ARC, CAC
write back	write back
pozn. oběť: uvolňování bloku	

# DISK

- nejdůležitější periferie, snadíme se myslit

RAID - Redundant array (independent disks)

- RAID 0

- zvyšuje výkon systému pětiček disků

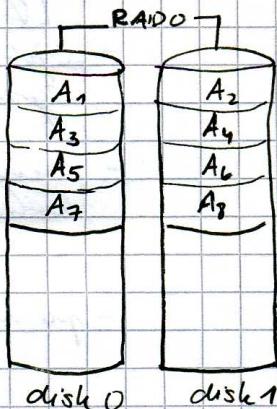
- ksv. striping

- RAID 1

- pro zvýšení spolehlivosti uložených dat

- ksv. mirroring

- nevyhluje, ale zvyšuje spolehlivost

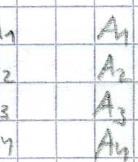


- RAID 5 ~ (RAID 6 → pomalejší zápis)

- uchovává paritní informace

- v degradovaném režimu se může jít o data uložená na vadném disku odvozovat z dat zdravých disků a parity

- rychluje čtení, zpomaliuje zápis



- RAID 10

- kombinace RAID 0 a RAID 1

→ vytváří se RAID 0 a ten se zde dělí na RAID 1

↳ výsledkem jsou dva RAID 1 obsahující identická čísla

- zvyšuje jak výkon, tak spolehlivost, avšak je nutno použít nejméně čtyři disky, nejlépe se stejnými parametry

# I/O PODSYSTÉMY

## • Interfejs (= rozhraní / propojení)

- společná komunikační část sdílena dvěma systémy, zařízeními nebo programy

- zahrnuje i protokoly k tomu určené a doplňkové řídící údaje uvedené k příslušnému propojení.

- přenos údajů
  - asynchronní
    - nezávislé z hlediska časování
  - synchronní
    - přenos konstantního obměnitou frekvence
    - isochronní
      - konstantní průměrná rychlosť

## • Standardní rozhraní v PC

• PCI sběrnice

• PCI Express point-to-point

• USB point-to-point (Universal Serial Bus)

• Fireware point-to-point

• Serial ATA point-to-point

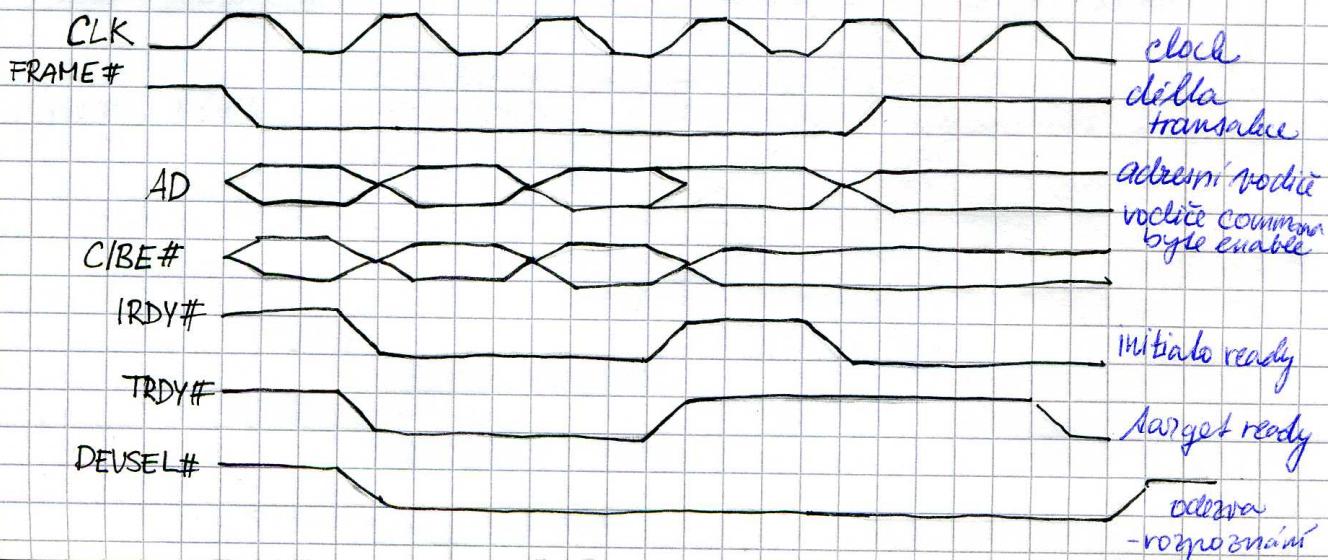
### PCI

- rozdíl sběrnicové transakce se rozděluje na 2 závěry:

• initiator (bus master)

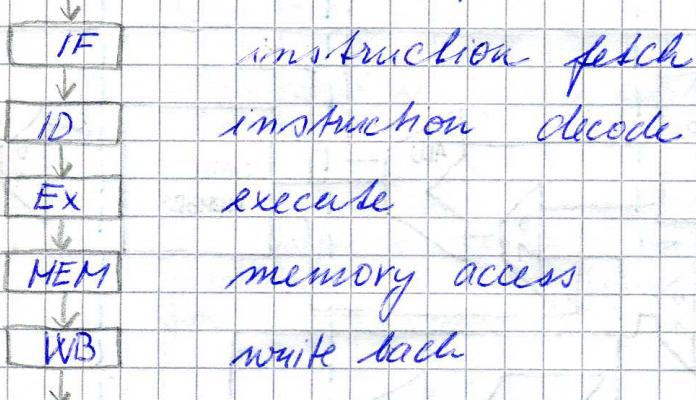
• target (slave) → může být více

- sběrnicový cyklus má obvykle adresovanou datovou fázi



# ZRĘCZENÉ VYKONÁVÁNÍ INSTRUKCIÍ

rozdělení do 5 stupňů



## • IF

- poslat PC do paměti a načíst aktuální instrukci

- aktualizace  $PC = PC + 4$

## • ID

- dekodování instrukce a načtení registru spec. v instrukci
- provedení testu rovnosti registru (kříž možnosti)
- pamětníkové rozšíření offsetu
- vypočet cílové adresy pro případ návratu

## • EX

- operace ALU

## • MEM

- v případě instrukce load/store → čtení/zápis do paměti

## • WB

- v případě instrukcí typu register-register nebo instrukce load - zápis myslidlem do RF

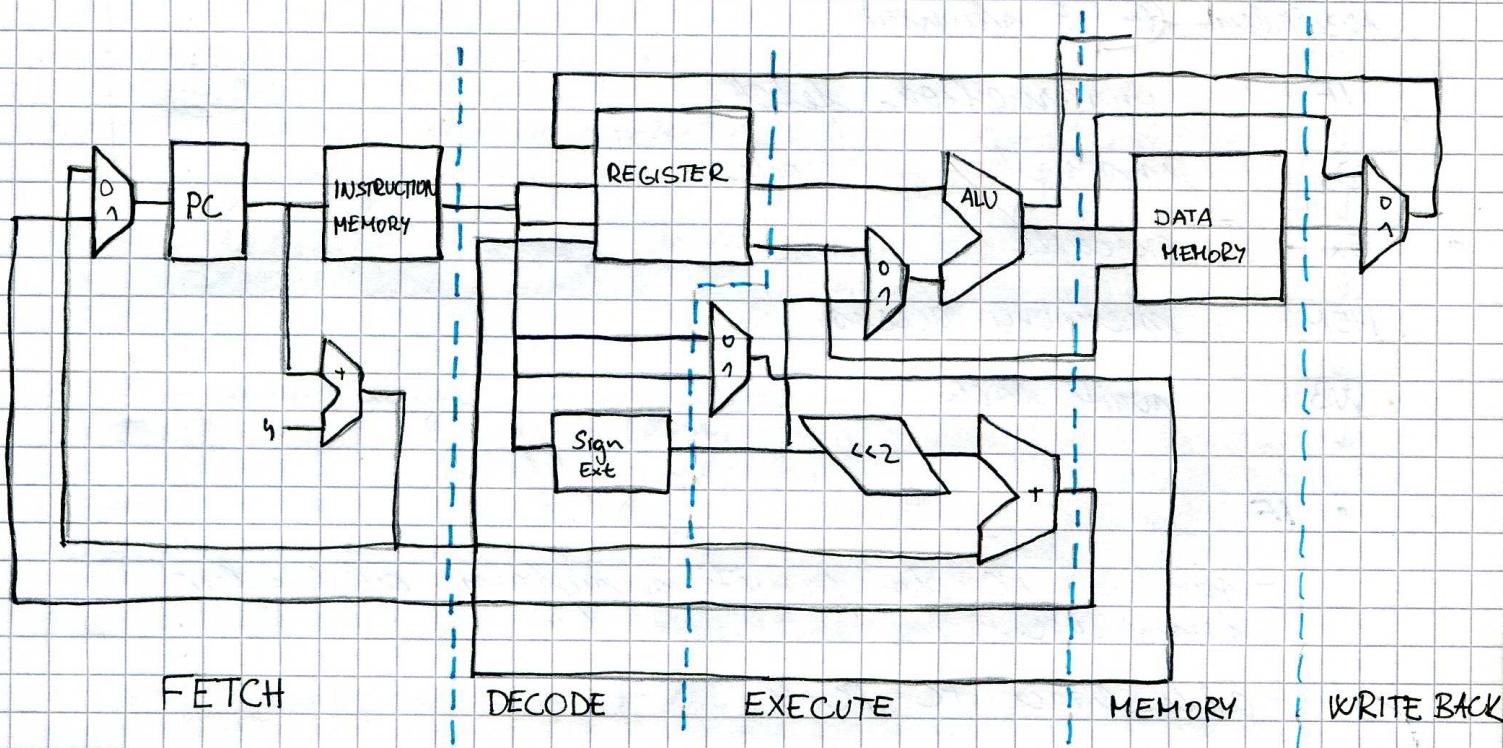
↳ myslidlo může přicházet z ALU nebo paměti  
pom. čas následní instrukcí k - stupňové pipeline

$$T_k = k \cdot \tau + (n-k) \tau$$

$$\tau = \max \{ T_i \}_{i=1}^k$$

$T_i$  propagation delay  
v jednom z  $k$  stupňů

## Zřežení myšlenkovací



## Hazardy

- způsobují pozastavení myšlenkovací (stall) nebo nvpřednutí pipeline

př.

1 2 3 4 5 6 7

Další hazard

IF ID EX MEM WB  
IF ID EX MEM WB

ADD R1, R2, R3  
SUB R4, R1, R3

- sub přičte neplatnou hodnotu R1

- řešení hazardů preposiláním (forwarding)

• pokud následná vznikla dřív než jí následující instrukce slunce ne potřebuje jej možné bude hazard řešit preposiláním (forwarding)

• například, když se použije zdrojové registry instrukce ve stupni EXE shodují s cílovým registrém ve stupni MEM nebo WB

→ čísla těchto registrů z těchto stupňů musejí být posílána do Hazard Unit

## - řešení hazardů pozastavením (stall)

- pokud následující instrukce počítače vyžaduje druh než slunečné vzniku je možné tento hazard řešit pozastavením
- pozastavení pipeline je prostředkem řešení hazardů
  - nevyužívá všechny propustnosti systému
- stupně pipeline předcházející stupni mohou hazard vzniknout jenom pozastaveny do doby, než jsou k dispozici následující pozadované následujícími instrukcemi, kdy jsou pak přesílány (forwarding)

## Využívání stupňů architektury

### - využívání

- cílem je rozdělit jednotlivé bloky do  $N$  stupňů tak, aby ve spojení se všech stupních bylo pokud možno stejné.

### záměr (pokud budeme ignorovat hazardy)

- procesor využívá různy typy instrukcí bezprostředně následující za hazardu sledovací instrukci (tj. mění PC v masém případě s opožděním jednotku cyklu) bez ohledu na platnost či neplatnost podnětných shod, pokud však tato sledovací instrukce není využívána v důsledku právě tétoho pravidla.

→ Shodovat instrukce shodí s opožděním jednotky cyklu (jedna instrukce)

### = branch-delay slot

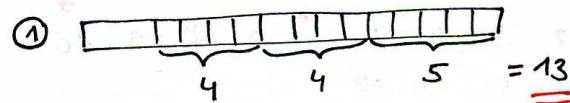
- uživatelem kompilátorem je lze rozepřít, že následující instrukce následující se v branch-delay slotu budou platné a naopak.

použ. nejsnadnější způsob je plnit delay slot přední instrukcí - nap. (nejméně efektivní)

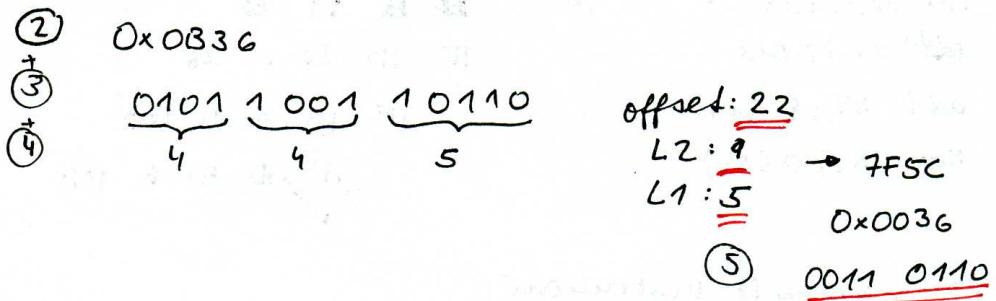
4.9

- ① • 16-bitů
- little endian
- 32B
- 2-úrovni
- 2B

⑥ → 3-úrovni



② 0x0B36



offset: 22

L2: 9 → 7F5C

L1: 5

0x0036

⑤ 0011 0110

⑦ rewrite / regodes

⑧ záleží na všech hodnotách

⑨ alibiuni spoje → A, B, C, F

⑩ 4-cestné → 16kB

Velikost bloku 32B

Velikost slova 32bitů 14B)

0x6D0

$$\cdot \text{setg} = \frac{\frac{16 \cdot 1024}{32}}{4} = \underline{\underline{128}}$$

⑪ Cache hity

$$\cdot \frac{512}{8} \cdot 7 + 1 = \underline{\underline{499}}$$

⑫ 0x6D0

$$\textcircled{12} \quad 1024 \Rightarrow 1024 - 499 = \underline{\underline{525}}$$

~~00110110 11010000  
10000000 00000000  
2<sup>3</sup>  
1885600~~

0000 0110 1101 0000  
↓ 3 2<sup>2</sup> (4B)  
54 (2<sup>3</sup>) = 8 slov

⑭ 16

⑮

⑯

(17) S forwardingen

	1	2	3	4	5	6	7	8	9	10
sub \$4, \$1, t1	IF	ID	EX	M	WB					
lw t2, 0x4(\$4)	IF	-	ED	EX	M	WB				
addi \$2, +2, 0x3			IF	ID	EX	M	WB			
add \$4, \$4, 0x4				IF	ID	EX	M	WB		
sw \$2, 0(\$4)					IF	ID	EX	M	WB	

(17) 06  
(18) 10

(19) • 8% miss rate instrukci

• 6% miss rate datové cache

• CPI (Clock Cycles per instruction) = 4 bez memory stalls

• penalizace 100 cyklů pro každý cache miss

• celkovou hodnotu CPI procesoru pro daný program, posunut 25 %

• využívající instrukci představují load a store instrukce.

⇒ celková hodnota CPI

$$(0,08 \cdot 1 \cdot 100) + (0,06 \cdot 0,25 \cdot 100) + 4 = \underline{\underline{13,5}}$$

29.6

- 2 GHz (procesor)
- 3 úrovně cache (16-cestně asociativní L3 - 8MB (instrukce i data))
  - ↳ latence 38 cyklů
- velikost bloku cache je 64B
- TLB plně asociativní o 64 pozicích
  - ↳ latence 3 cykly
  - vyhledávání probíhá paralelně a činnosti L1
- 64-bitové celočíselné registry
- 16GB paměti - frekvence 1200 MHz
  - ↳ latence je 44 cykly → 70 ns
- data jsou při čtení z danej úrovni přičtena po uplynutí latence bez ohledu na objem
- sekundární paměť kapacita  $\approx 500 \cdot 10^9 B$  (500GB)
- ideální přenosovou rychlosť pro čtení i zápis 540 MB/s  
a rychlosť náhodného čtení/zápisu 97 000 IOPS (Input/output per s)
- rozložit 8 kB stránky
- jedna pozice tabulky zabírá 8B
- virtuální adresa má síťku 53 bitů (počet bitů pro překlad)

① průměrný čas přístupu do paměti, pokud předpokládáme, že miss rate je následující:  $L1 d = 0,1 \quad L2 = 0,2 \quad L3 = 0,1$   
odpovidejte v mikrosekundách.

$$\text{3.19} \quad 2\text{GHz} = 2 \cdot 10^9 \text{Hz} \rightarrow 1 \text{cyklus} \frac{1}{2 \cdot 10^9} = 5 \cdot 10^{-10} \text{s}$$

$$L1: 4 \cdot 5 \cdot 10^{-10} \text{s} = 2 \cdot 10^{-9} \text{s}$$

$$L2: 0,1 \cdot 12 \cdot 5 \cdot 10^{-10} \text{s} = 6 \cdot 10^{-10} \text{s}$$

od L1

$$L3: 0,1 \cdot 0,2 \cdot 38 \cdot 5 \cdot 10^{-10} \text{s} = 3,8 \cdot 10^{-10} \text{s}$$

od L1 a L2

$$\text{paměť: } 0,1 \cdot 0,2 \cdot 0,1 \cdot [44(8,33 \cdot 10^{-10} + 70 \cdot 10^{-9} \text{s})] = 1200 \text{MHz} = 1,2 \text{GHz} = \frac{1}{1,2 \cdot 10^9} = 8,33 \cdot 10^{-10} \text{s}$$

$$\text{od } L1, L2, L3 \quad [44 \cdot 8,33 \cdot 10^{-10} + 70 \cdot 10^{-9} \text{s}] = 2,133 \times 10^{-10}$$

$$\text{součet} = 3,19 \cdot 10^{-9} \text{s} = \underline{\underline{3,19 \text{ ms}}}$$

② jedno-úrovnové stránkování, kolik položek bude mít stránkovací

③ tabulek

$$\rightarrow \text{offset} = 13 (\log_2 8192)$$

• 8 KB stránky = 8192 B

• jedna položka tabulky zabírá 8B

• virtuální adresa má 53 bitů

→ 1024 položek na 1 stránce  
 $= 2^{10}$

↳ potřebují 10 bitů  
na očíslování

~~53 10 53 2b~~

jedna položka 3bitů

opovídá: 53 bitů - 13 bitů offset = 40 bitů

↳ 2<sup>10</sup> bitů položek

④ page walk na příkladu (pri přehlade VA na FA)

- latence: 121 ns

•  $2^{10}$  }  $\left\{ \begin{array}{l} 2^{10} \\ 2^{10} \\ 2^{10} \\ 2^{10} \end{array} \right.$  jedna stránka 1 přístup do paměti  
4 stránky  $\Rightarrow 4 \cdot 121 = \underline{\underline{484 \text{ ms}}}$

⑤ jak dlouho bude trvat přehled VA pokud není potřeba mykat page walk

• latence mi nezajímá (mo page walk)

• TLB (dilá přehlada - memory cache) → 3 cykly, procesor:  $26 \text{ Hz} \rightarrow \frac{1}{2 \cdot 10^9} = 5 \cdot 10^{-10}$

$$\Rightarrow 3 \cdot 5 \cdot 10^{-10} = 1,5 \cdot 10^{-9} \text{ s} = \underline{\underline{1,5 \text{ ms}}}$$

⑥ kolik setů obsahuje L3 cache

• 16-cestné asociativní

• 8 MB = 8192 kB = 8388608 B

• velikost bloku cache 64 B

$$\left\{ \frac{8388608}{64} = 131072 \text{ bloků}$$

↳ 16-cestné : 16

16 bloků  
vedle sebe

⑦ 16 MB dokument (charong - 1 char = 1B) = 8192 setů

- odhadnout hit rate vyšší uvedené L1 cache

a hit rate vyšší uvedené TLB

• 4-cestné, 32 kB  $\Rightarrow 32768 \text{ B}$

• velikost bloku 64 B

$$\left\{ 512 \text{ bloků}$$

↳ 4-cestné : 4 = 128 setů

• 1 blok = 64 charů

↳ 1 miss + 63 hits

$$\left\{ \frac{63}{64} = \text{úspěšnost} = \underline{\underline{0,98}}$$

⑧ Zajíž je hledat rade pro nájde mnoho TLB.

⑨  $A = 0b10100101$   
 $B = 0b10001010$

$$\begin{array}{r} 00101111 \\ \hline \end{array}$$

⑩  $F = \bar{z} \cdot \bar{c} \cdot \bar{o}_V$

$$\begin{array}{r} 0011 \\ \hline \end{array}$$

⑪ PCIe již dnes využívá PCI sběrnice, se kterými není kompatibilní jak po hardwarové stránce tak z pohledu software  
- NE - je kompatibilní

⑫ V případě 64-bitové virtuální adresy je obvyklé používat několik bitů pro fyzickou adresu.

- ANO

⑬ V odprůdušené verzi jedno-fázového procesoru je ADDI instrukce nejdelsí instrukcí procesoru a od ní se odvíjí délka periody hodinového signálu.

- NE - nejdelsí je dv

⑭ CISLA s plouvacím rádotvárcem IEEE754 musí být vždy uložena v normalizovaném tvare (tzn. min. techn. je vždy jednouha).

- NE

⑮ Přenos DMA je možné využít pro některé operace dat, můžete použít mechanismus přerušení a dispečlen zaťaze (overhead), kde je potřeba pro inicializaci DMA.

- NE

16) float pow2(int x) ... realizuje hodnotu  $y = 2^x$

```

    float return1 = 1;
    if (x > 0)
    {
        for (int i = 0; i < x; ++i)
        {
            return1 *= 2;
        }
    }
    else if (x < 0)
    {
        for (int i = 0; i < x; ++i)
        {
            return1 /= 2;
        }
    }
    return return1;
}

```

```

    addi $1, zero, 0
    addi $1, a0, 127
    sll    $1, $1, 23

```

### ASSEMBLER

...

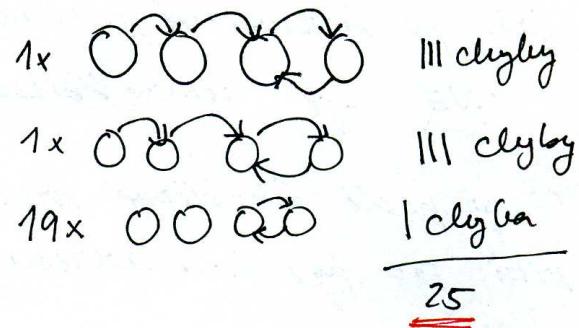
```

nebo float pow2(int x)
{
    x = (x + 127) << 23
    return * [(float *) &x];
}

```

20) celkový počet špatných predikcií  
v součtu pro všechny instrukce  
2-bitový predictor  
• strongly not taken (vyčíslit hodnotu)

- 25



17) 5-stupňový zřetezený procesor (forwarding)

	1	2	3	4	5	6	7	8	9	10
sub \$4 \$1 t1	IF	ID	EX	M	WB					
lw t2 0x4(\$4)	IF	ID	EX	M	WB					
addi \$2 \$2 0x3	IF	Shall	ID	EX	MB	WB				
add \$4 \$4 0x4			IF	ID	EX	MB	WB			
sw \$2 0(\$4)				IF	ID	EX	M	WB		

18) Ve kterém cyklu bude poslední instrukce v IF a když ve WB

- IF = 6 - WB = 10

19) uvedený program

- hodnota  $t_4$  ? (4 mistné číslo)

184410 4418XFAA

$s_4 = 0 \quad s_0 = 1$   
 $s_1 = 0 \quad t_1 = 21$   
 $t_2 = 7$

for ( $t_3 \neq 0$  deťej)  
for ( $t_3 \neq 0$  deťej)

$s_4 += 7$   
 $s_1 -= 1$   
if ( $s_1 < t_2 \rightarrow t_3 = 1$   
else  $t_3 = 0$ )

$s_0 += 1$   
if ( $s_0 < t_1 \rightarrow t_3 = 1$   
else  $t_3 = 0$ )

} 49 hodnota  
7x } 20x

=> 980

26.6 - 78

① bne

8: 1000  
3: 0011

0001010100000011  
opcode rs rt  
0x1503

② -3

0011  
1100  
0001  
1101  
13 → FFFFD

③ Frequency: 16 Hz

A: 4 (10%)  
B: 1 (20%)  
C: 4 (20%)  
D: 4 (50%)

3,4 cyklu / instrukci

$16 \text{ Hz} = 1 \cdot 10^9 \text{ Hz} \Rightarrow 1 \text{ cyklus za } 1 \cdot 10^{-9} \text{ s}$

$$\frac{3,4 \cdot 10^9}{1 \cdot 10^{-9}} = \underline{\underline{3,4 \cdot 10^{18}}}$$

④  $A = 0b01011110$   
 $B = 0b11010000$   
100101110

⑤ Z: 0  
C: 1  
OV: 0

⑥ nebrde schope vyzkovat spravne

- Write DataE → "0"
- vše co se zapisuje do memory → sw

⑦ • ALUoutW → "0"  
→ vše co se zapisuje do registru → add, addi, sub

(lze nejelikot plne  
z ReadDataW)

⑧

	1	2	3	4	5	6
add \$2 \$1 \$2	IF	ID	EX	M	WB	
sub \$4 \$0 \$1		IF	ID	EX	\$1	WB
beq \$2 \$2 \$2		IF	ID	EX	M	WB

	1	2	3	4	5	6
add \$0, \$1, \$1	IF	ID	EX	M	WB	
sub \$2, \$0, \$2		IF	ID	EX	X	
beq \$0, \$2, \$1						

$$⑨ SV / 1 \times \left(\frac{5}{6}\right)$$

IF: 5ms

ID: 8ms

$$EX: 20ms \rightarrow \frac{1}{20 \cdot 10^6} \cdot \frac{5}{6} = 41666666,67 \rightarrow 41,6 \underline{\underline{= 42}}$$

MEM: 8ms

WB: 1ms

⑩ Data uložená v L1 datové cache paměti dvojicích moderních procesorů, jenž vždy stejná jdele dat uložená na odpovídající adresu ve fyz. paměti

-NE

⑪ Architektura AMD64 používá 64-bitový formát virtuální adresy, ale po přehledu využívá pouze 48 bitů

-ANO

⑫ NE ⑬ Je možné instrukci aritmetického posunu upravit tak, aby mohla být použita pro log. posun (doplňková aritmetika)

-NE

• aritmeticky:  $\gg \underline{1} (-) 0 (+) \ll \underline{0}$

• logicky:  $\ll \underline{0} \gg \underline{0}$

⑭ mikroprogramový řadič realizuje každou programátorskou viditelnou instrukci pomocí vlastního mikroprogramu uloženého v řadiči paměti řadiči

-ANO

⑮ valid page fault  $\rightarrow$  rámec  
invalid page fault  $\rightarrow$  program je obvykle ukončen (segmentation fault)

⑯ ✓

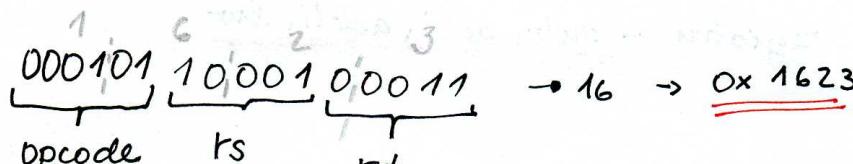
⑰ addi \$1, zero, 0  
st1 \$1, \$1, ,23  
dddi \$1, \$1, , -127

(25.6) (15)

① bin (v hexadecimálním formátu)

② ↳ I instrukce

horní instrukce ( $16 \rightarrow 32$ )



$\rightarrow 16 \rightarrow \underline{\underline{0x1623}}$

11 334

dolní polovina instrukce ( $0 \rightarrow 15$ )

$$PC + 4 + 4 \times C = -8 + PC \quad (2 \text{ řádky nahoru} \Rightarrow -4-4)$$

$$4C = -12$$

$$\underline{C = -3}$$

$$3 : 0011$$

$$-3 : 1100$$

$$\underline{0001}$$

$$\underline{\underline{1101}}$$

$$\rightarrow \begin{array}{cccccc} 15 & 15 & 15 & 15 \\ 1111 & 1111 & 1111 & 1101 \end{array} \\ \underline{\underline{0xFFFF}}$$

(3)

A - 10%

B - 30%

C - 30%

D - 30%

Frekvence

1,5 GHz

CPI - A

3

CPI - B

5

CPI - C

4

CPI - D

5

↳ počet cyklů za sekundu

Jaká je rychlosť procesoru v MIPS

$$\left( \frac{1}{1,5 \cdot 10^9} = 6,66 \cdot 10^{-10} \right) \text{ (1 cyklus)}$$

$$\frac{1,5 \cdot 10^9}{4,5} = 333333333,3$$

instrukce za sek

$$= \underline{\underline{333,3 \text{ MIPS}}}$$

100 instrukcí

10 instrukcí A (30 cyklů)	}
30 instrukcí B (150 cyklů)	
30 instrukcí C (120 cyklů)	
30 instrukcí D (150 cyklů)	

4,5 cyklů/instrukci

(4)  $A = 0b11110101$  $B = 0b10001100$ 110000001

(5)

Z : 0

C : 1

OV: 0 → polohu je carry v předposledním a posledním, mení to overflow

0	1	
0	1	
1	0	<u>overflow</u>

⑥ jakej instrukce nebude procesor schopen vykonavat správně

- ALUOutW je zablokovany na hodnotě "0" (BIT 0)

→ to co ujde do memory: sub, addi, add

⑦ - RegWriteE je zablokovany na hodnotě "0"

- může zapisovat do registru - sub, add, addi, lw

⑧

rozložit  
↓ ✓

	AE BE	1	2	3	4	5	6	7	8
add	s2s1s2	IF	ID	EX	M	WB			
sub	s4s0s1	IF	ID	EX	M	WB			
beq	s2s2s2	IF	stall	ID	EX	M	WB		

refungiye u s2

⑨ jaká je průměrná propustnost bublina 5V / 1x

IF - 7ms

ID - 6 ms

Ex - 35ms

MEM - 6 ms

WB - 4ms

$$\frac{1}{35 \cdot 10^9} \cdot 5 = 23809523,81 = \underline{\underline{23,8}}$$

frekvence

(nolice ms/s)

⑩ RAID úrovne 1 nepracuje s redundancí (nadbytnost)

- NE → rozložíruje si hodnoty (mirroring)

⑪ Data, která jsou uložena v L1 datové cache paměti dnesních moderních procesorů, jsou vždy stejná jako data uložená na odpovídající adrese ve fyzické paměti.

- NE

⑫ Dle konceptu von Neumanna se v počítání programy a výsledky ukládají do téže paměti - což je významující rozdílem v porovnání s tzv. Harvardskou architekturou.

- ANO

⑬ Paměťová buňka DRAM je rychlejší než SRAM

- NE

⑭ V případě použití programového kanálu s přerušením procesor jde pouze periodicky sledovat stavový bit / registr daného I/O zařízení což je v protikladu s metodou "polling-u", kdy procesor musí neustále ne snadně sledovat dané zařízení a reagovat na jeho stav. - NE

- ⑯ Nezávisle na organizaci cache, pokud nastane valid page fault (adresa je součástí virtuálního adresního prostoru), pak typicky dochází k mactení (z disku do hiberní paměti) jednoho:  
→ macte celý námc (page)

- ⑰
- IRDY #
  - TRDY # } dole
  - CLK ... náležitá frekvence
  - Frame# ... mimoře (po článkoch)

⑱ log 2

$$[(x \gg 23) - 127]$$

addi \$1, zero, 0  
srl \$1, \$1, 23  
addi \$1, \$1, -127

zde je fce

(15.6) - ①

① MemToReg W "0"  
· lwr \$s, C(\$s)② MemToReg W "1"  
· addi, add, sub

③

add	SO	AE DE	S1 S1
Sub	S2	AE BE	SO S2
beq	\$0	AE BE	S2 n1

1	2	3	4	5	6
IF	ID	Ex	A	WB	
		clzha			

add SO S1 S2  
Sub S4 SO S1 the same shit

④ 8v / 1x

$$\text{IF: } 5\text{ms} \quad \frac{1}{15 \cdot 10^9} \cdot \frac{8}{9} = 59259759,2 \text{ ns} \quad \cancel{= 59}$$

$$\text{ID: } 8\text{ms}$$

$$\text{Ex: } 15\text{ms}$$

$$\text{Mem: } 8\text{ns}$$

$$\text{WB: } 1\text{ns}$$

⑤ bet forwardingu

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
slt to a0 a1	IF	ID	Ex	M	WB											
addi t0 t0 -1	IF	(S)	(S)	ID	Ex	M	WB									
not t1 t0				(S)	(S)	ID	Ex	M	WB							
and t0 a0 t0						ID	Ex	N	WB							
and \$1 a1 t1							ID	Ex	M	WB						
or v0 t0 t1								ID	Ex	M	WB					

⑤ ⑦ ⑥ ② ⑦ ⑥ ⑧ ① ⑨ ②

$$\text{⑩ and: } \begin{array}{r|l} 1000 & 1000 \\ 0001 & 0001 \\ \hline 0000 & 0001 \end{array} \quad \text{not: } \begin{array}{r} 0000 \\ \downarrow \\ 1111 \end{array} \quad \text{or: } \begin{array}{r} 1000 \\ 0001 \\ \hline 1001 \end{array} \quad \text{maximum}$$

⑪: první mapování datového cache : 512 B

$$= 0 \times 015C$$

· delší blok: 32/16 = 2 16 B

· delší posl.: 173 slov

· cache má základní prázdno

· jedno slovo

· delší slova: 32 Bitů [4B]

· 10 průchodek

$$\Rightarrow \frac{512}{16} = 32$$

$$\frac{\text{velikost cache}}{\text{velikost bloku}} = \frac{512}{16} = 32$$

⑫ 0x15C

$$\begin{array}{r} 0001 \\ \hline 0101 \end{array}$$

$$\begin{array}{r} 1100 \\ \hline 21 \end{array}$$

cache block offset  $\rightarrow$  sloučit CISLO

jedny bloky

(13) pocet hitů  
- první přechod cyklem

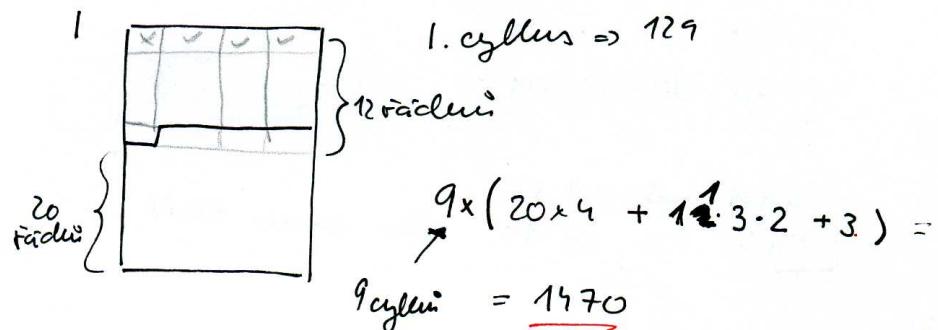
- 32 bloků
- alhern 173 slov  $\Rightarrow \frac{173}{4} = 43$  × nastavené (43,25)

$$1 \text{ blok} \rightarrow \underbrace{\text{mis}}_{\text{hit}} | \overbrace{\text{hit}} | \overbrace{\text{hit}} | \overbrace{\text{hit}}$$

$$\left. \begin{array}{l} 43 \cdot 3 = 129 \text{ hitů} \\ + 0 \text{ hitů} \end{array} \right\} (0,25)$$

(14) celkový pocet hitů

- 10 cyklů (11180,5560)



(15) 1-bitový predictor

- taken  $\Rightarrow \underline{?}$

(16) in/out instrukce - NE

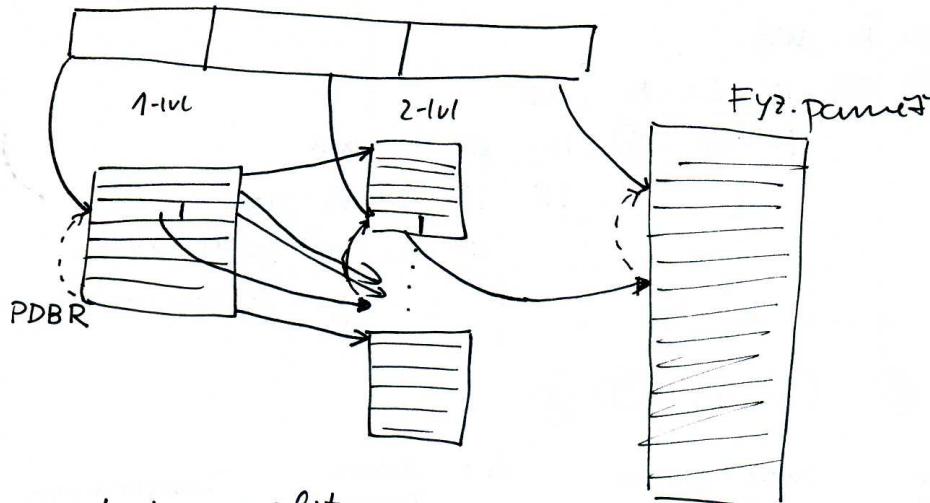
(17) dyn. predictor × rotační predictor  
- NE

(18) predečení IEEE754  $\rightarrow$  NaN - NE

(19) NE - assembler a stack

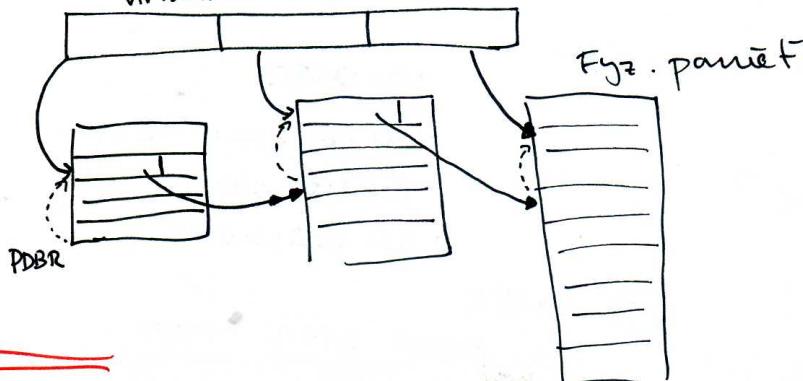
(20) adresový dešifrátor připojený na PCI sběrnice - NE

(21) virtuální adresy



repeat to profit

virtuální adresa



(15.6) -4

① ForwardBD je prerusen

beq③ ForwardAE 00,01,10  $\rightarrow$  00,01,00

AE RE		IF	ID	EX	M	WB
beg \$0 s2	n	IF	ID	EX	M	WB
add \$0 s1 s1		IF	ID	EX	M	WB
sub \$2 s2				chyla		

AE RE		IF	ID	EX	M	WB
sub \$0 s1 s2		IF	ID	EX	M	WB
add \$3 s4 s0		IF	ID	EX	M	WB
addi \$7 s0, 2		IF	ID	EX	M	WB

AE RE		IF	ID	EX	M	WB
sub \$2 s1 s2		IF	ID	EX	M	WB
addi \$0, \$0, 1				IF	ID	EX M
add \$1, s1, s3					IF	OK

AE RE		IF	ID	EX	M	WB
sub \$2, s3, s2		IF	ID	EX	M	WB
sub \$4, \$0, s1		IF	ID	EX	M	WB
add \$2, \$7, s2		IF	ID	EX	M	WB

ziskacim z BE

④ GU / 1x

$$\text{Ex: } 25 \text{ ms} \Rightarrow \frac{1}{25 \cdot 10^9} \cdot \frac{6}{7} = 34285714,29 \approx 34$$

⑤ bez forwardingu

10	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0 A	sub t0 a0 a1	IF	ID	EX	M	WB								
0 B	addi t2 zero 0	IF	ID	EX	M	WB								
1 C	sra t1 t0 31	IF	(S)	ID	EX	M	WB							
2 D	and t1 t1 t0		IF	(S)	(S)	ID	EX	M	WB					
0 E	addi v1 zero 0				IF	ID	EX	M	WB					
1 F	add v0 a1 t1					IF	(S)	ID	EX	M	WB			

- $t_0 = a_0 - a_1$
- $t_2 = 0$
- $t_1 = t_0 \gg 31$
- $t_1 = t_1 \& t_0$
- $v_1 = 0$
- $v_0 = a_1 + t_1$

5	$a_0 = 10$	$a_1 = 5$
0		
0		
0000		
0101		
0000		
5		

minimum

⑩ velikost cache: 1024 B

délka bloku: 64 B

délka slova: 32 bitů (4 B)

adresa: 0xBC

délka pole: 277 slov

velikost prvního pole: 1 slovo

Příchozí

Přímo mapování -  $n=1$

⑪

$$\frac{\text{velikost cache}}{\text{velikost bloku}} = \frac{1024}{64} = 16$$

$n$

$$\frac{1024}{64} = 16$$

⑫ 0xBC

1001 1100

2<sup>4</sup> [délka slova]

slov v bloku

[délka bloku]

[délka slova]

$$= 2$$

⑬ první příchozí

0xBC

$\downarrow$  1111, 16. slovo v bloku (1 miss)

$$\frac{277 \text{ slov}}{16 \text{ slovy v bloku}} = \text{kolik nejvýším setin} \Rightarrow \frac{17,3125}{0}$$

$$\frac{276}{16} = 17,25$$

-1 miss

• 17 → 1 miss 15 hits

• 0,25 → 3 hits  $(16 \cdot 0,25) = 4$  (1 miss 3 hits)

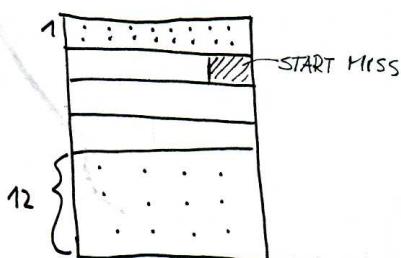
$$17 \times 15 + 3 = 258$$

⑭ celkový počet hits

$$258 + \left[ 13 \cdot 16 + (15 \cdot 4 + 3) \right] = 2155$$

alternativa  $(17-16) \cdot 2 \cdot 7 + 19 = 61$

277.



⑮ 8 (not taken) ANS

⑯ AND

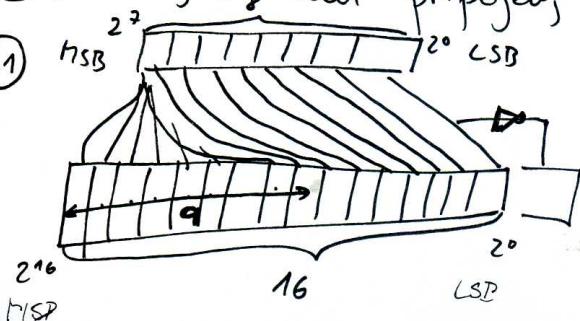
⑰ NE

⑱ RAID 6 rozkládá paritní informace napříč jednotlivými diskami  
- ANS

⑲ DMA data neprocházejí skrz procesor - ANS

⑳ adresový dešifrátor připojuje na PCI sběrnici - NE

㉑



15.6 - 7

① ALU OUT W → "0"  
• addi, add, sub

② sign Ext mornich 16 bitů má výstupu vždy  
ma 0.  
• sw, lw, addi, beq → všechno kde je c

③ Forward AE → "0"

add \$2 \$1 \$2	IF	ID	EX	n	WB
sub \$4 \$0 \$1	IF	ID	EX	n	WB
beq \$2 \$2 n2	IF	ID	EX	n	WB
			cycle		
sw \$0 \$1 \$2	IF	ID	EX	n	WB
add \$3 \$4 \$0	IF	ID	EX	n	WB
lw \$1 \$4(\$2)	IF	ID	EX	n	WB
			→ OK		

② sw \$0,4(\$1)	IF	ID	EX	n	WB
sub \$2,\$0,\$2	IF	ID	EX	n	WB
beq \$0,\$2,n1	IF	ID	EX	n	WB

add \$4,\$0,\$1	IF	ID	EX	n	WB
sub \$0,\$1,\$0	IF	ID	EX	n	WB
add \$1,\$1,\$3	IF	ID	EX	n	WB

④ 8V / 1x

$$\frac{1}{35 \cdot 10^9} \cdot \frac{8}{9} = 25$$

\* maximum

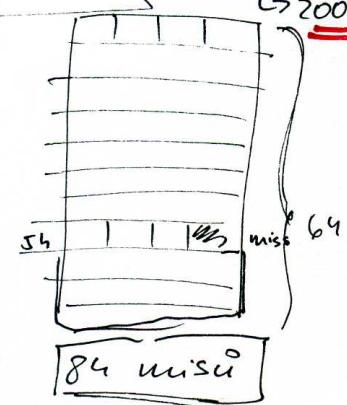
⑤ not forwarding

- ① slt t0,a0,a1
- ② addi t0,t0,-1
- ② not t1,t0
- ③ and t0,a0,t0
- ④ and t1,a1,t1
- ⑤ or v0,t0,t1

1	2	3	4	5	6	7	8	9	10
IF	ID	EX	n	WB					
IF	-	-	ID	EX	n	WB			
IF	-	-	ID	EX	n	WB			
IF	ID	EX	n	WB					
IF	-	-	ID	EX	n	WB			
IF	-	-	ID	EX	n	WB			
IF	-	-	ID	EX	n	WB			
IF	-	-	ID	EX	n	WB			
IF	-	-	ID	EX	n	WB			

11	12	13	14	15	16	17
WB						
M	WB					
ID	EX	WB	WB			
IF	-	ID	ID	EX	WB	WB

$$(84 - 61) \cdot 2 \cdot 6 + 84 = 324 \text{ missu}$$



⑨ 1024B

délka Bloku 16B

délka slova 32bitů (4B)

0x36C

333 slov

7 cyklů

• setři → 64

•  $0x36C$   
 $\begin{array}{r} 0011 \\ \diagdown \quad \diagup \\ 0110 \quad 1100 \\ \diagup \quad \diagdown \\ 3L16 \quad 42 \quad 2 \end{array} \rightarrow 54$

$$333 : 4 = 83,25$$

+

249 missu

84 missu

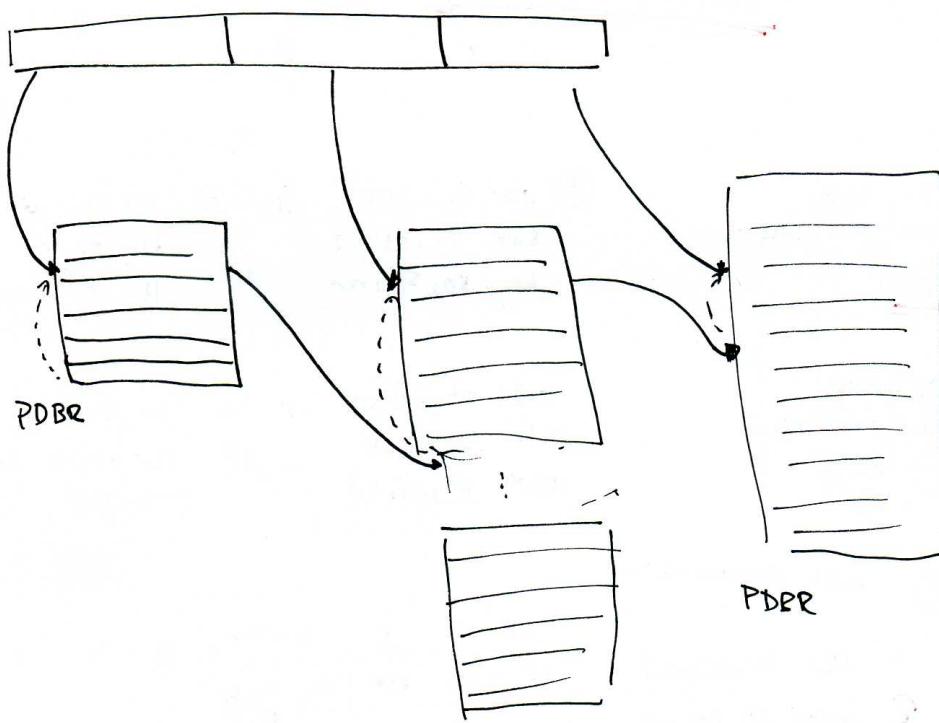
• 2007

• 1. cyklus

• 4 slova v bloku

(16) 1-bitary predictor  
• taken  $\Rightarrow$  31

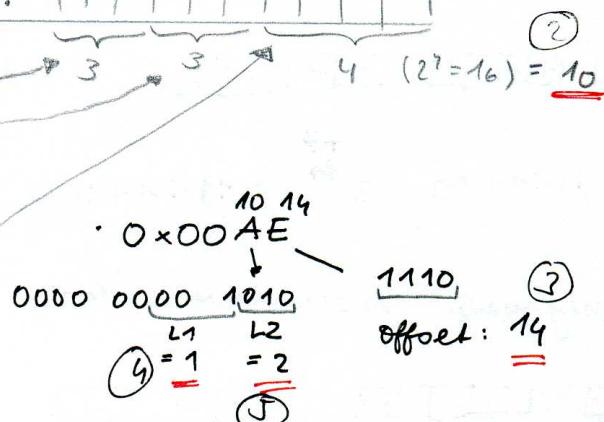
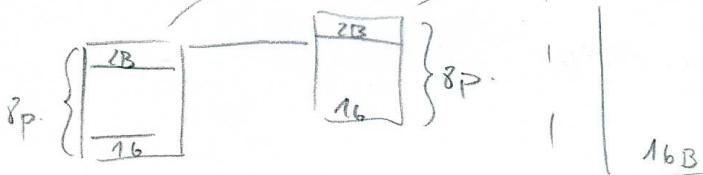
(17)



(6.6) - (16)

## ①. little-endian

- 16 bitový procesor
- stránka 16 B
- 2-úrovnové
- 2 B



- 40. → 1. poloha = 80 00 → 0080
- 80. → 2. poloha = 10 00 → 0070
- 10. → 14. poloha = 47 9B → 9B47 DATA  $\begin{matrix} 7 \\ 6 \end{matrix} \quad \begin{matrix} 4 \\ 4 \end{matrix}$

(8) velikost cache: 512 B

délka bloku: 32 B

stupen asociativity: 1

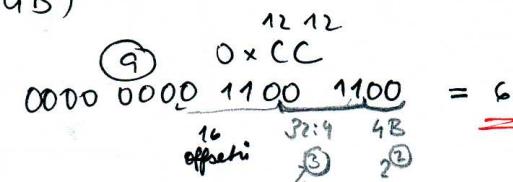
cache je na začátku prázdná

194 slov

4 cykly

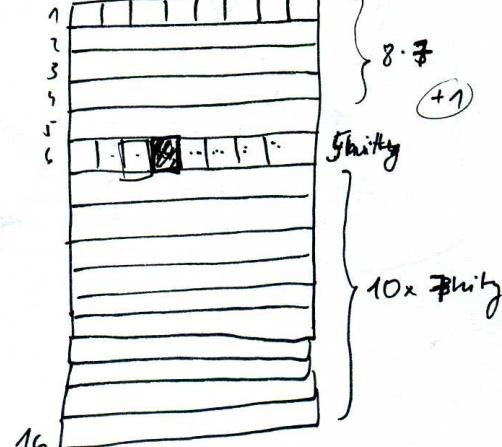
délka slova: 32 bitů (4 B)

(8) počet setů: 16



(11) počet missů

$$194 - 169 = 25$$



$$23 \cdot 7 + 5 + 3 = 169$$

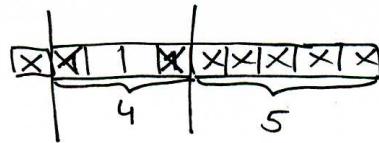
(12) celkový počet hitů

169 + 3 · (7 · 8) + 3 (16 · 7 + 8) = 697

obrázek

→ celkem 79 missů

(14)



$$-7.91 \Rightarrow 0111.11101$$

$$2^{\frac{1}{2}} \Rightarrow 9 : \text{bias} = 7$$

(15) největší možná zobrazitelná číslo

$$(2^4 : 2 - 1)$$

$$\square \boxed{1111} \quad \boxed{11111111} \quad +\infty \quad 111.. \quad 0000..$$

$$(16) 1,11111 \cdot 2^{\frac{+7}{4}} = 11111100 = \underline{\underline{252}}$$

(17) nejménší menšího dílučku v norm. čísle

$$\square \boxed{0000} \times \boxed{000000}$$

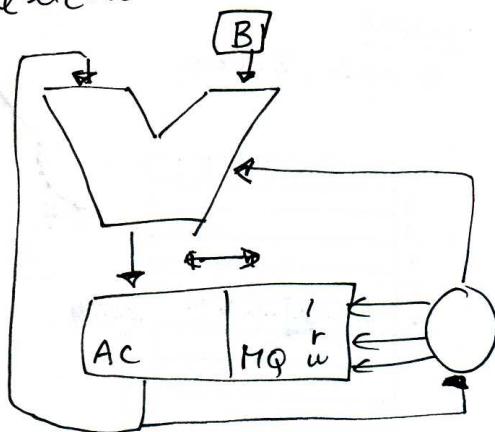
$$(18) 1,00000 \cdot 2^{-6} = ,0000010 \doteq \underline{\underline{0,016}}$$

$$(19) \text{not taken } = \underline{\underline{8}} \quad \text{správus}: 32 - 8 = \underline{\underline{24}}$$

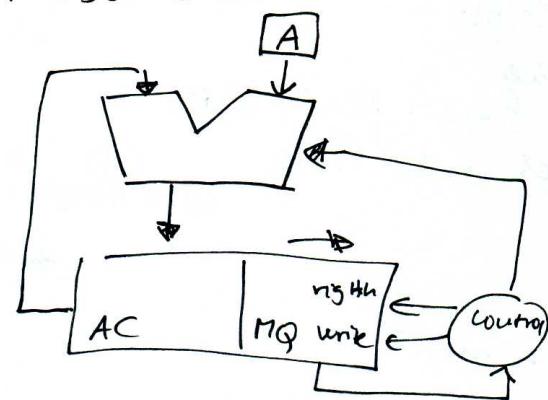
(20) V BAR registru c. 0 je uložena fyzická adresa ...

-NE

de lichia



na souběžen



(6.6) - 17

① little endian

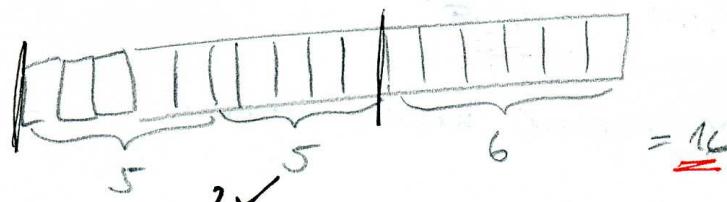
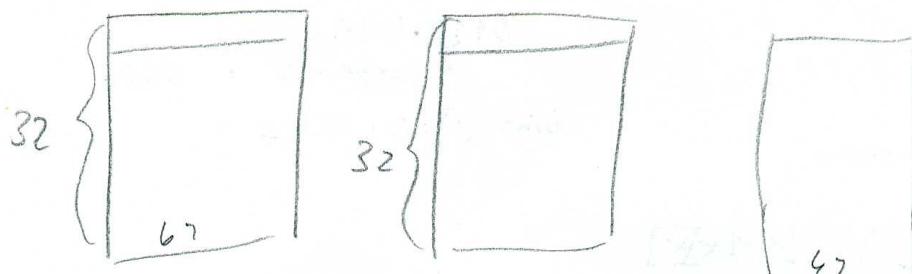
16 bitů

64 B

2 úrovniové

2 Bytes

0x0040



offset

$0 \times 9B9C^{12}$

19

14

28

• množství dat?

19  $\Rightarrow$  80 00  $\Rightarrow$  0080

14  $\Rightarrow$  00 00  $\Rightarrow$  0000

28  $\Rightarrow$  ~~03 80 12 59 00~~  $\Rightarrow$  ~~F3AB~~

• adresa  $\Rightarrow$  0x001C

(8)

2048 B

délka Bloku: 64 B

primo mapped

slovo: 32 bitů [4]

0x33C

692 slov

7 cílů

• celkový počet bitů

$647 + 6$  [

• setz: 32

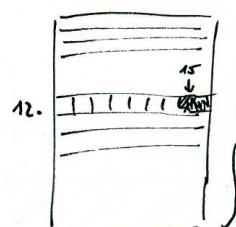
•  $0 \times 33C^{12}$

0011 0011 1100

16 slov / 5104

• 1. průchod

$$\frac{691 \text{ slov}}{16} = 43,18$$



• missů = 45

$$43 \cdot 15 = \frac{645 + 2}{16} = 647 \text{ bitů}$$

(9)



-6,43



$$\begin{array}{r} 0110.0110 \\ \hline 2 \end{array} \Rightarrow e^x = 5 : 0101$$

$$\text{bias}(2^3 : 2 - 1) = 3$$

(10) největší možné



$$1,1111 \cdot 2^3 = 1111,1 = \underline{\underline{15,5}}$$

(11) nejménší můžoucí hodnoty normalizované

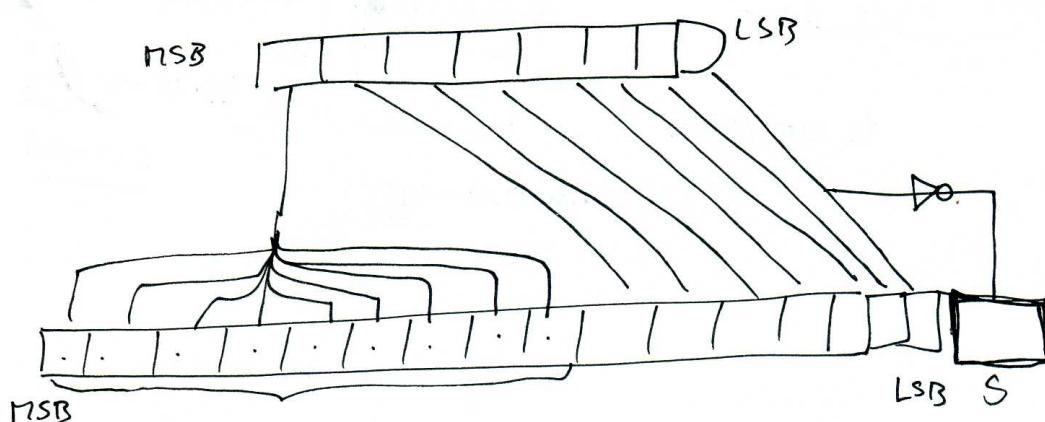


$$1,0000 \cdot 2^2 = \underline{\underline{0,25}}$$

(12) 2-bitový - strongly TAKEN

2 cely  $\cdot$  celkový počet  $\geq$ 

(20)



suale  $\rightarrow$  true  
 (false  $\rightarrow$  false)

26.6 - 79

① addi (18)

• horní polovina instrukce

→  $\text{opcode}_{(6)}$ ,  $rs(5)$ ,  $rt(5)$

$$\begin{array}{ccccccc} & 2 & 2 & 1 & 0 \\ \underline{001000} & , \underline{10000} & , \underline{10000} & & & & \\ \text{opcode} & rs & rt & & & & \end{array}$$

16: 1 0000

2210 ✓

②

• dolní polovina

$\cdot -8 \rightarrow \begin{array}{r} 1000 \\ 0111 \\ \hline 0001 \end{array}$

$\begin{array}{c} 1111111 \\ \hline \boxed{1000} \end{array}$

FFF      8

③

A: 30%  $\rightarrow 5$   
 B: 20%  $\rightarrow 3$   
 C: 30%  $\rightarrow 3$   
 D: 20%  $\rightarrow 2$

$\left. \begin{array}{l} \\ \\ \end{array} \right\} 3,4 \text{ cyklu/instrukci}$

$$(0,3 \cdot 5 + 0,2 \cdot 3 + 0,3 \cdot 3 + 0,2 \cdot 2)$$

$$\text{Frequency: } 1,5 \text{ MHz} = 1,5 \cdot 10^9 \text{ Hz} \rightarrow \frac{1}{1,5 \cdot 10^9} = 6,667 \cdot 10^{-10} \text{ cykly/sec}$$

~~vystřelec~~ ~~3,4 / 10^9~~  
~~6,667 \cdot 10^{-10} c/s~~

13 instrukcí      3,4 --- 1 in  
 1... - x  $\rightarrow \frac{5}{17}$  1 cyklus ... whole instruction

$$\frac{5}{17} \cdot 1,5 \cdot 10^9 = 441176470,6 \text{ instrukc/s}$$

↳ million  $\rightarrow \underline{\underline{441}}$

④

$$\begin{array}{r} A=0b10110011 \\ B=0b10001100 \\ \hline 100111111 \end{array}$$

- ⑤  
 • carry  $\Rightarrow 1$   
 • overflow = 1  
 • zero  $\Rightarrow 0$

⑥ Write Data E zařazenou "0"

sw

⑦ MemTOReg W "0"

lw

⑧ Forward AE  $\rightarrow$  0

<u>sw \$0, 4(\$1)</u>	IF	ID	EX	M	WB
<u>sub \$2,\$0,\$2</u>	IF	ID	EX	M	WB
<u>beq \$0,\$2,n1</u>	IF	ID	EX	M	WB ✓

<u>add \$4,\$0,\$1</u>	IF	ID	EX	M	WB
<u>sub \$0,\$1,\$0</u>	IF	ID	EX	M	WB
<u>add \$1,\$1,\$3</u>	IF	ID	EX	M	WB ✓

<u>add \$2,\$1,\$2</u>	IF	ID	EX	M	WB
<u>sub \$4,\$0,\$1</u>	IF	ID	EX	M	WB
<u>beq \$2,\$2,<sup>AE</sup>n2</u>			<u>clyba</u>		

<u>sub \$0,\$1,\$2</u>	IF	ID	EX	M	WB
<u>add \$3,\$4,\$0</u>	IF	ID	EX	M	WB
<u>lw \$1, 4(\$2)</u>	IF	ID	EX	M	WB ✓

⑨  $7v / 1x$

$$\text{Ex: } 20 \text{ ns} \quad \frac{1}{20 \cdot 10^{-9}} \cdot \frac{2}{5} = 43750000 = \underline{\underline{43}}$$

⑩ ... ⑯

⑯ zählerlost na diesel ✓

⑰  $(x \gg 23) - 127$

addi	\$1, zero, 0
srl	\$1, 20, 23
addi	\$1, \$1, -127

(26.6) - daloří

① bne

① horní Polovina

1	5	8	3
000101	01100	10011	
opcode	rs	rt	
12:		3: 0011	
<u>1100</u>			

0x 15B3

② dolní polovina

③  $\rightarrow$  FFF13  
c

③ A: 20% - 4  
B: 30% - 4  
C: 20% - 2  
D: 30% - 3 } 3,3 zlulu/instrukci

$$2 \cdot 64 \cdot 2 = 2 \cdot 10^9 \Rightarrow$$

$$\frac{3,3 \cdot 10^9}{2 \cdot 10^9} = 1,65$$

④ A=0b10010110  
B=0b11000100

101011010

⑤ zero 0  
carry 1  
OV 1

⑥ ALU out W "0"  
'add, addi, sub'

⑦ Mem to Reg W "0"  
- LRU

⑧ Forward AE  $\Rightarrow$  '0'

beq \$0 \$2 n1	IF	ID	EX	n	WB
add \$0 \$1 \$1	IF	ID	EX	m	WB
sub \$2 \$0 \$2	IF	<del>EX</del>	<del>n</del>	<del>WB</del>	
					<u>carry</u>
sub \$2 \$1 \$2	IF	ID	EX	n	WB
addi \$0 \$0 1	IF	ID	EX	n	WB
add \$1 \$1 \$3	IF	ID	EX	m	WB

sub \$2 \$3 \$1	IF	ID	EX	n	WB
sub \$4 \$0 \$1	IF	ID	EX	m	WB
add \$2 \$2 \$2	IF	ID	EX	n	WB
					<u>carry</u>
sub \$0 \$1 \$2	IF	ID	EX	n	WB
add \$3 \$4 \$0	IF	ID	EX	m	WB
addi \$1 \$0 2	IF	ID	EX	n	WB
					<u>mem</u> <u>carry</u>

⑨ 6 / 11 x

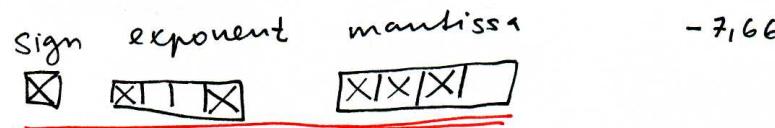
$$\frac{1}{25 \cdot 10^{-5}} \cdot \frac{6}{7} = \underline{\underline{34}}$$

⑩

~~oh~~

19.6

① 9-bitů



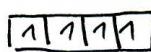
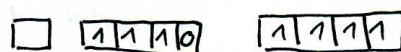
$$-7,66 = 0111,10101000_2$$

$$\text{bias} = 7 \rightarrow \boxed{9} \quad \boxed{1001}$$

$$\underline{2^4 : 2 - 1}$$

② největší možné reálné číslo

nekompatibilní  
ex ... 111...  
m... 000...



$$1,1111 \cdot 2^{14-7} \rightarrow 11111000 = \underline{\underline{248}}$$

④ nejménší menšího hodnoty normalizované číslo

<input type="checkbox"/>
<input type="checkbox"/> <input type="checkbox"/> <input checked="" type="checkbox"/>
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

$$\underline{1 \cdot 2^{-6}}$$

$$1 \cdot 2^{-6} = 0,015625 = \underline{\underline{0,016}}$$

⑥ lw \$t, C(\$s)      ⑦ add \$d,\$s,\$t      ⑧ sw \$t,C(\$s)  
 • A,B,C,F      • A,B,C,E,G

A, D, F, G  
 ↘  
 z mem      offset  
 potřeba na ALU      psani do registeru

⑨ sll \$d,\$s,\$t  
 • A,B,F,C

⑩ VO po skončení programu

S0 : a4ed000...

S0 : a4ed3d6a

S1 : 3b18

ma 4A78: a4ed3d6a

S1 : 3b3e →  $\begin{array}{r} 3b18 \\ 0026 \\ \hline 3b3e \end{array}$ 

S2 : 00d2

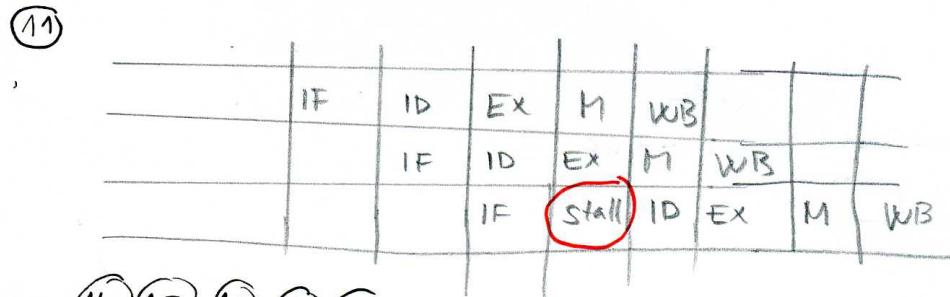
S3 : -001a

S4 : 3b24 →  $\begin{array}{r} 3b3e \\ -0014 \\ \hline 3b24 \end{array}$ 

ma 4A78: d2ed3d6a

VO : to co je ma 4A78 → d2ed3d6a

- lui S0, 0xa4ed
- ori S0, S0, 0x3d6a
- addi S1, zero, 0x3b18
- sw S0, 0x0f60(S1)
- addi S1, S1, 0x0026
- addi S2, zero, 0x00d2
- addi S3, zero, -0x001a
- addl S4, S1, S3
- sb S2, 0x0f34(S1)
- lw VO, 0x0f54(S4)



(12) zdvá stall  
-instrukce mod tím

(13) + 14 + 15 + 16 + 17 + 18

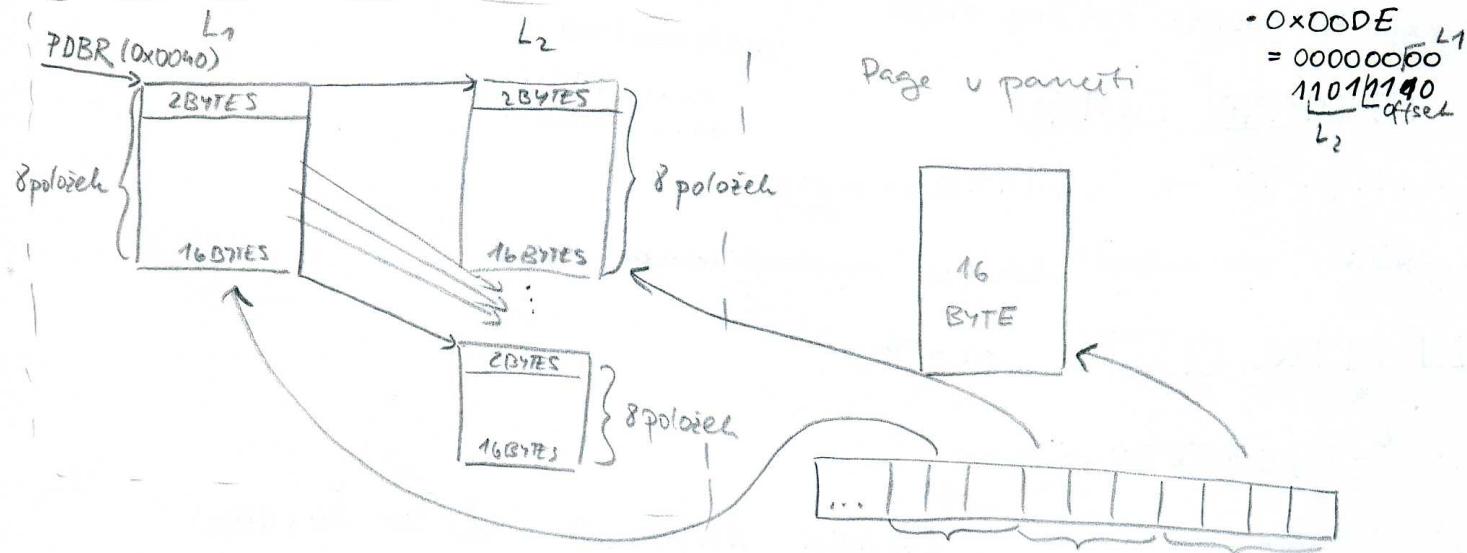
• 16-bitový

• little endian

• 16 byte ... velikost stránky

• 1 položka (2-vírovoucí tabulek) - 2 byty

• velikost adresy je částí tabulek, takže v tabulek o délce n m stranu odpovídá právě  $n$  stranice



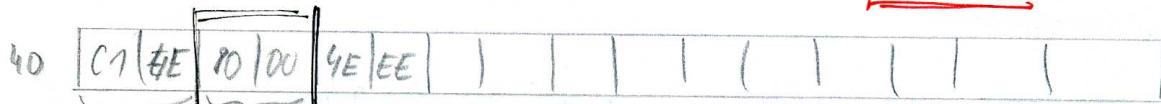
L1 → 40: C1, 4E celý rádce.

• virtuální adresa

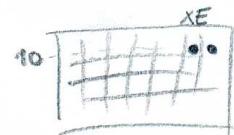
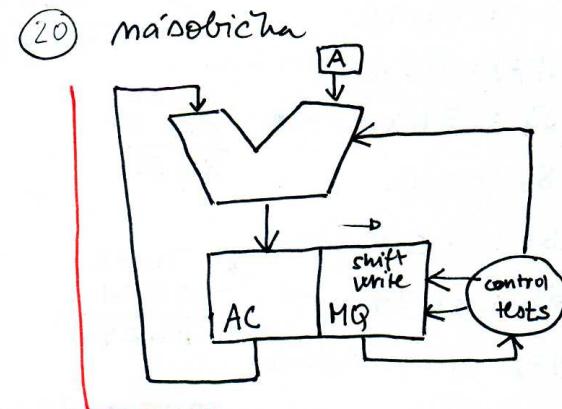
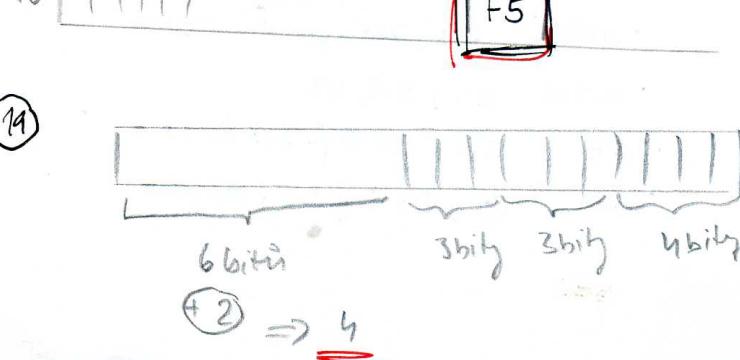
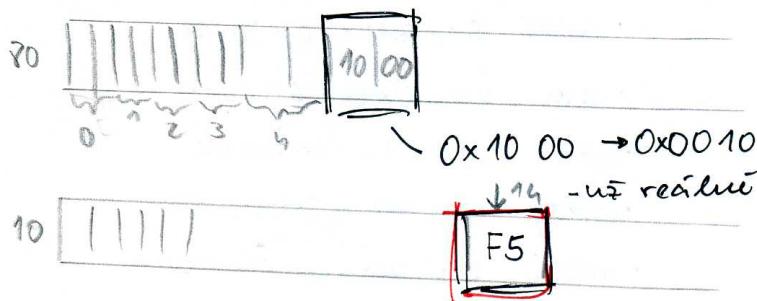
0x00DE → 0000 0000 ~~1101~~ 1101 L1 L2 offset

3bit 3bit 4bit = 10 byte  
 $2^3 = 8$  p.       $2^3 = 8$  p.       $2^4 = 16$  byte

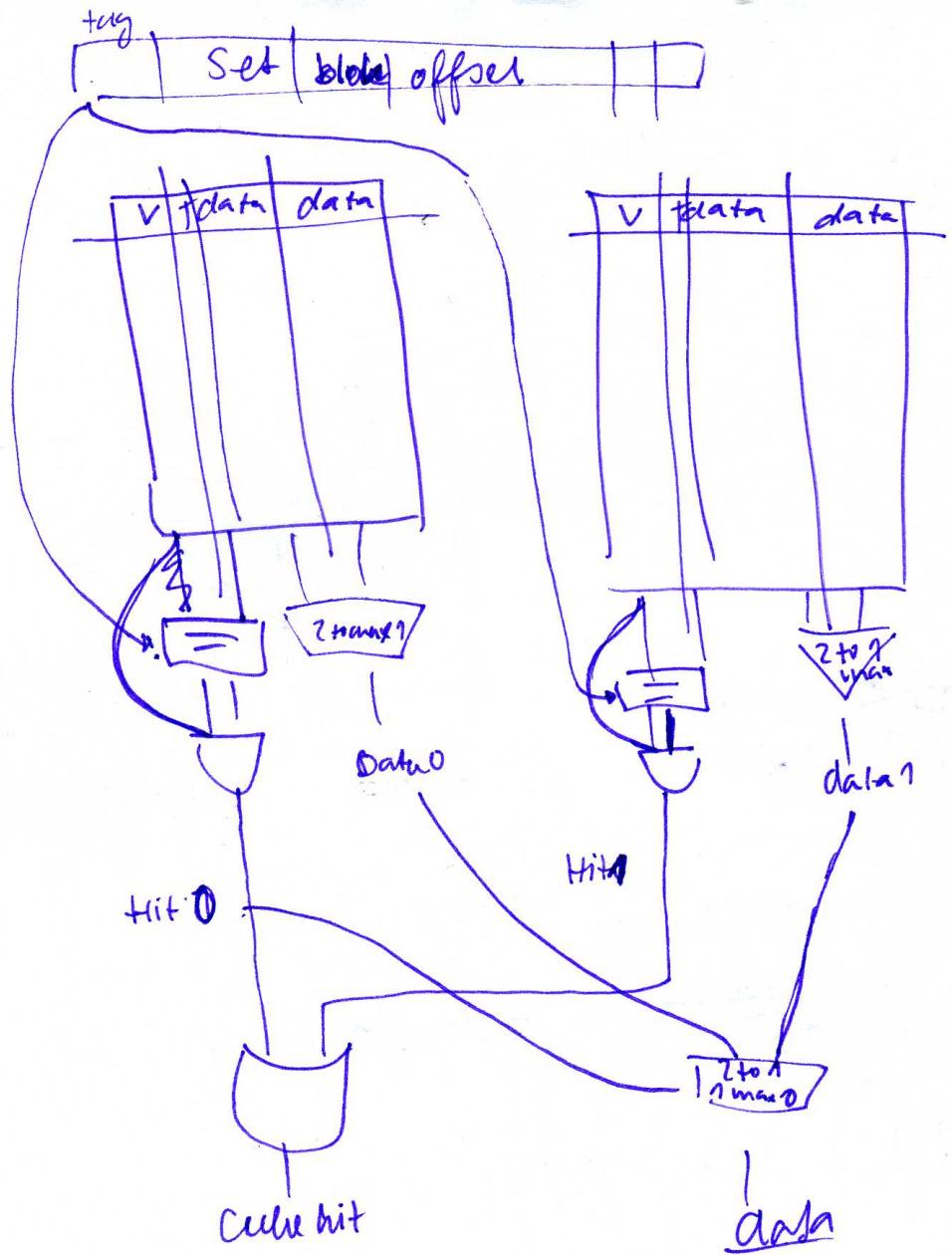
offset: 14  
L2: 5  
L1: 1



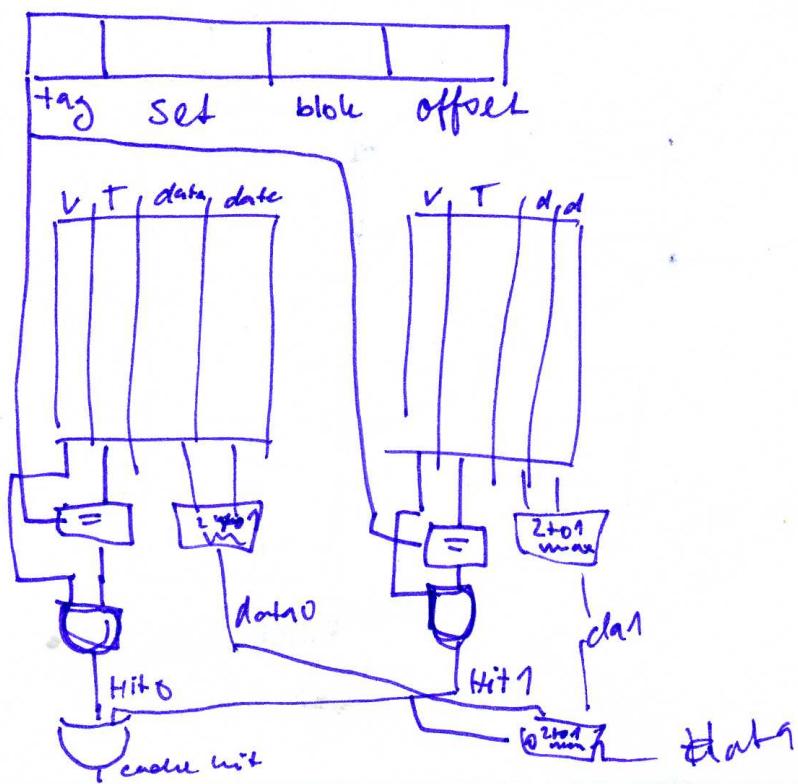
OP 1.p      0x80 00 → little endian ⇒ 0x0080 ... jdeme na adresu

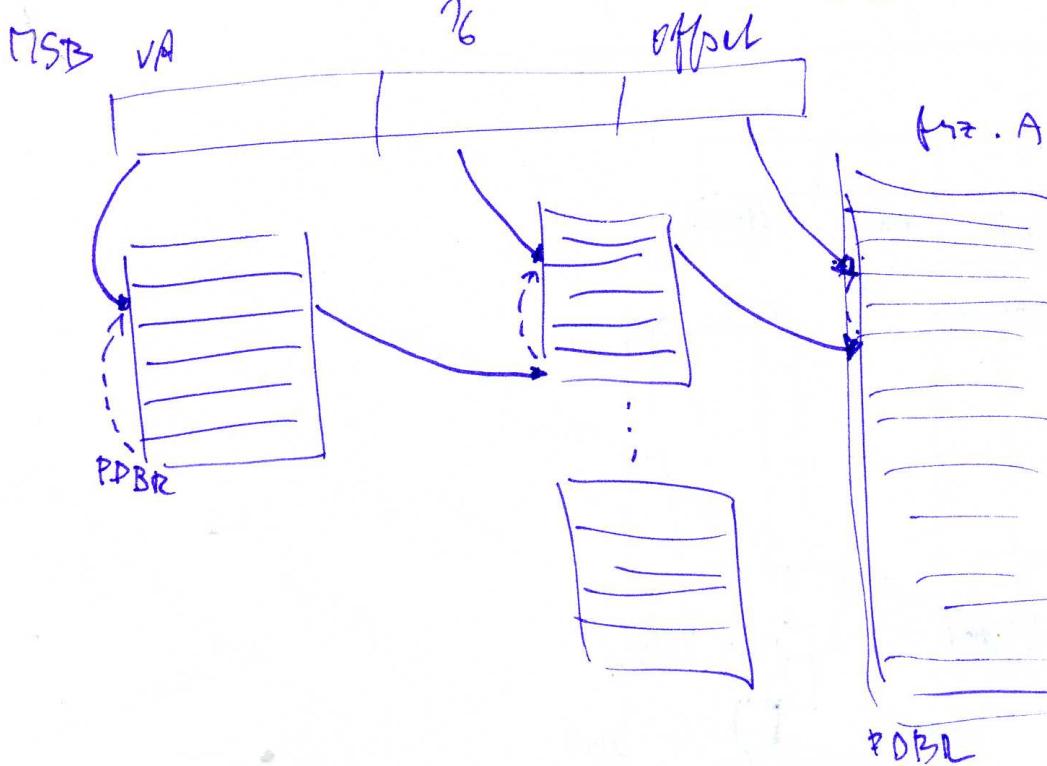
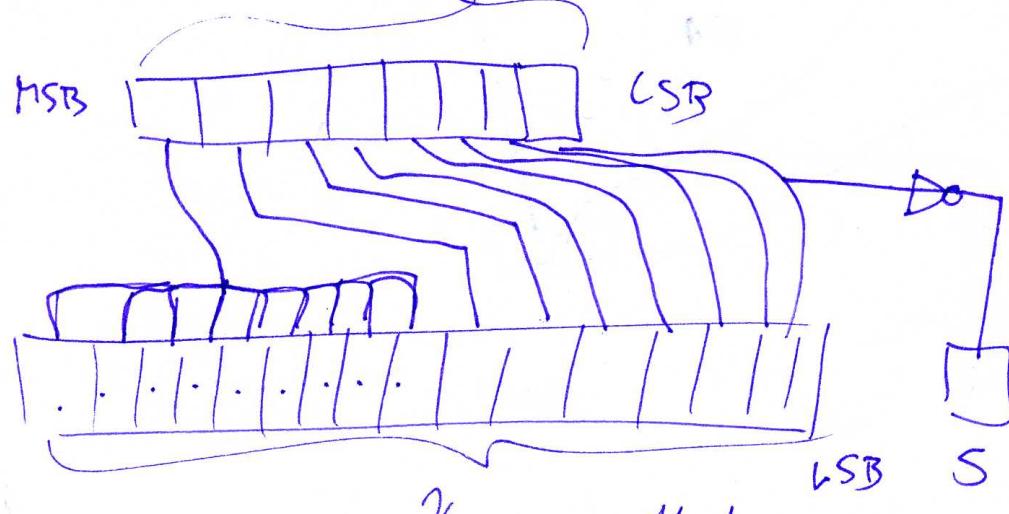
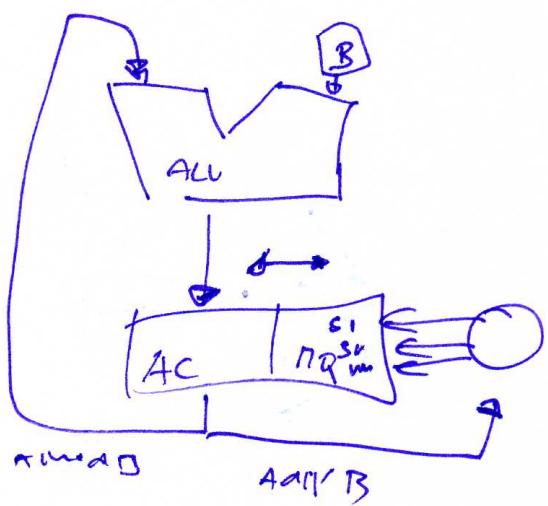
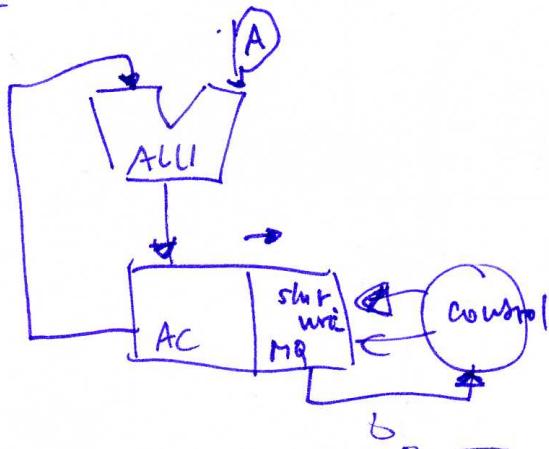
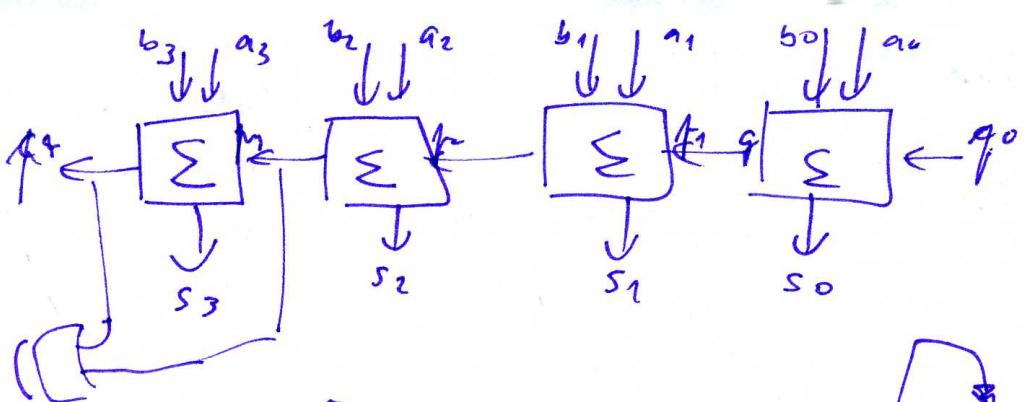


OX00'AE  
→ P42 ADRESA



offset 2  
block 3  
set 4



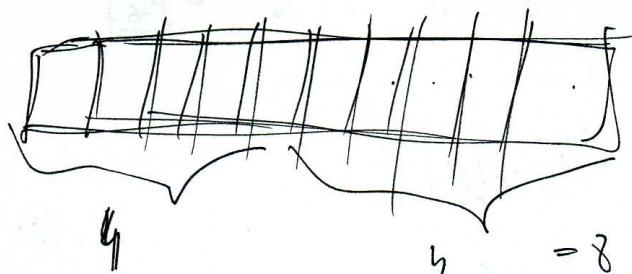
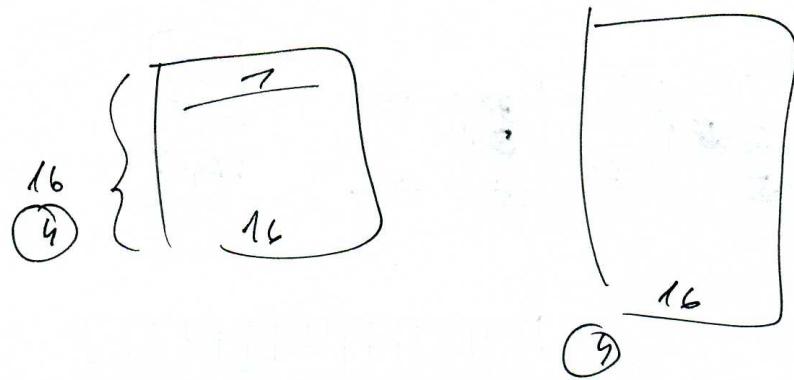


8-bit long

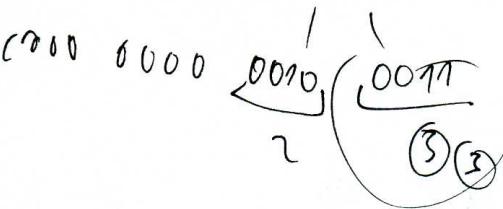
16 B

1 byte

7 miss



0x0023



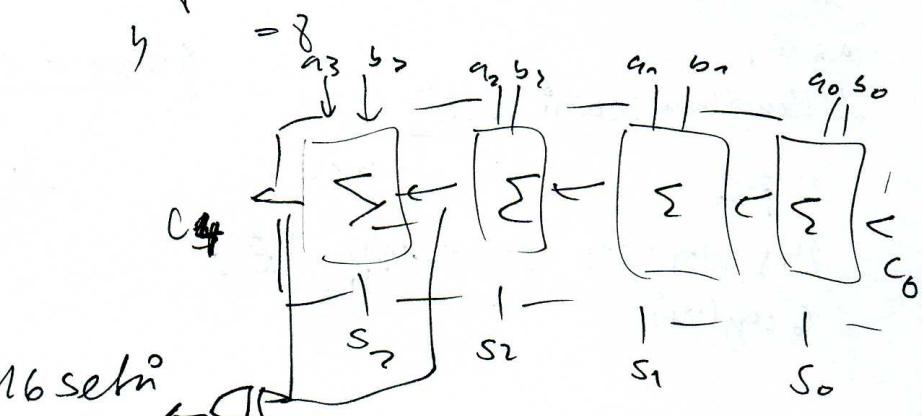
756 B

d. b 16B

1

32 bits (4B)

0x4C → 10 slots



5 x

4 slots

24.

$$72 + 2 = 74$$

0000 0x4C 11  
0000 0100 1100

18+12

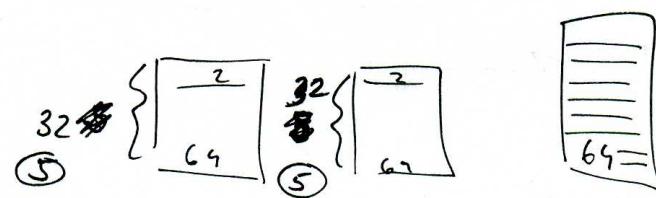
18+12

18+12

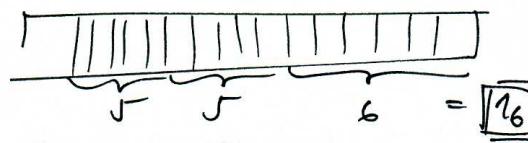
$$(16 - 26) \cdot 2 \cdot 4 + 2$$

26 miss

- ① . 16-bitový
- little endian
- 64 B
- 2-vrátne
- 2<sup>8</sup>



$0x2D18$   
 $\underline{\hspace{5mm} \hspace{5mm} \hspace{5mm}}$   
 $\underline{\hspace{5mm} \hspace{5mm} \hspace{5mm}}$   
 $\underline{\hspace{5mm} \hspace{5mm} \hspace{5mm}}$   
 $5 \quad 20 \quad 2$



PP90

Velikost cache 512 B

16 setů

délka bloku 32 B

$$18 \cdot 7 = \underline{\underline{126}}$$

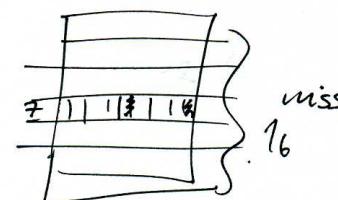
délka slova 32bitů (9B)

8 sloupců v 1 bloku

as = 7

1. cyklus

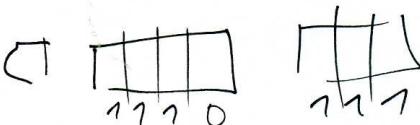
$0x\text{FC}$   
 $145 \text{ slou} \underline{0000} \quad 0000 \underline{1111} \quad 1100$   
6 cyklov  
7 (7)



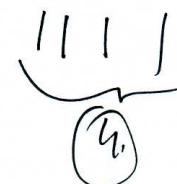
$$(19-16) \cdot 7 \cdot 5 + 19$$

1 miss

99 missů



bias = 7



$$1,111 \cdot 2^7 = 1,111 \underline{0000} = \underline{\underline{240}}$$

26 bitový

SNT

