Login:

31.5.2012

Zkouškový test z předmětu A0B36APO

Jméno a příjmení Já
Jméno cvičícího On

1. (1b) Jakou hodnotu reprezentuje následující binární číslo, uložené dle formátu IEEE754?
„11111111 10000000 00000000 00000001"
Předpokládejte 32 bitovou reprezentaci s 8 bitovým exponentem.

a) ○ 0
b) ✗ NaN
c) ○ -0
d) ○ +∞
e) ○ 5
f) ○ 0.5
g) ○ -0.5
h) ○ 1e64

Uvažujte vykonání následovné části programu při použití přímo mapované instrukční cache o velikosti 8 slov, **velikost bloku 1 slovo**. Předpokládejte paměť slovně zarovnanou, velikost slova 4B, tj. dva nejnižší bity adresy slova jsou 00. Cache je na počátku prázdná. Ukončení programu předpokládejte dosažením návěstí "done". Přitom instrukce na této adrese není načtena. Uvažujte, že instrukce bezprostředně za skokovu instrukcí se vykoná vždy. Registry i instrukce jsou 32-bitové. Program je vykonán na procesoru MIPS.

```
0x0000          addi t0, $0, 5
0x0004    loop: beq  t0, $0, done
0x0008          nop
0x000C          lw   t1, 0x4($0)
0x0010          lw   t2, 0x24($0)
0x0014          addi t0, t0, -1
0x0018          j    loop
0x001C          nop
0x0020    done:
```

2. (1b) Určete obsah registru t0 po ukončení programu: ..0.....
3. (1b) Určete Hit Count pro velikost bloku 1 slovo: ..30...
4. (1b) Určete Miss Count pro velikost bloku 1 slovo: ..8...

Jak se změní situace, pokud při vykonání výše uvedeného programu, použijeme **stejně velkou přímo mapovanou cache s velikostí bloku 4 slova?**

5. (1b) Určete Hit Count pro velikost bloku 4 slova: ..26...
6. (1b) Určete Miss Count pro velikost bloku 4 slova: ..2.....

7. (1b) Kolik cyklů bude trvat násobení dvou čísel a a b reprezentovaných n_a a n_b bity, pokud je násobička realizovaná dvěma posuvnými registry a sčítačkou za předpokladu, že je možné provést potřebné sčítání i posuny v jednom cyklu.

a) ○ $n_a + n_b$
b) ○ $n_a \cdot n_b$
c) ✗ n_a nebo n_b
d) ○ $n_a + n_b + 1$
e) ○ podle pořadí násobení bude odpovídat hodnotě čísla a nebo b
f) ○ $n_a \cdot n_b + 1$
g) ○ menší z čísel a a b

Login: 31.5.2012

8. (1b) Uveďte obvyklá pole obsažená ve formátu instrukcí !

Operační znak a operandy.

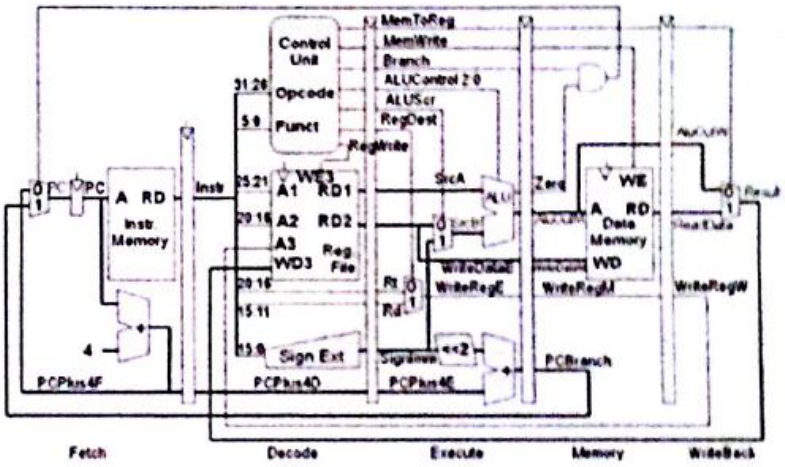
9. (2b) Určete zobrazitelné rozsahy pro tři kódování celých čísel se znaménkem při využití *n* bitů

Označení kódu	Minimální hodnota	Maximální hodnota
přímý kód	-2^(n-1) - 1	2^(n-1) - 1
doplňkový kód	-1/2 * 2^n	1/2 * 2^n - 1
aditivní kód	-1/2 * 2^n	1/2 * 2^n

10. (1b) Jaké druhy obsluhy přerušení/výjímek z pohledu běhu programu exitují:

synchronní, asynchronní (vnitřní a vnější)

Uvažujte rozdělení provádění instrukcí do jednotlivých stupňů tak, jak bylo probírané na modelovém procesoru MIPS (viz obrázek vpravo). Hazardy v datové cestě zatím nejsou řešeny, ale budou řešeny přeposíláním.



11. (1b) Ze kterého(ých) stupňě(ů) se budou mezivýsledky přeposílat?

WB,MEM

12. (1b) Na vstupy, které(ých) jednotky(ek) budou mezivýsledky přivádědné?

vstup do multiplexoru ALU

Uvažujte datovou vyrovnávací paměť (cache) se stupněm asociativity 2. Ve kterých z uvedených případů může dojít ke kolizi (soupeření o jednu řádku cache) při přístupech do paměti (pole) v následujících fragmentech kódu. Uvažujte procházení vektorů ve smyčce (řídicí proměnná for-cyklu je `i`). Cache je před začátkem cyklu prázdná.

13. (±1b) Sčítání dvou vektorů (polí čísel) s uložením výsledku do dalšího vektoru (pole)

```
y[i] = a[i] + b[i];
```

a) ✗ Může dojít ke kolizi b) ○ Nemůže dojít ke kolizi

14. (±1b) Kumulativní součet hodnot prvků z jednoho vektoru a uložení do jiného vektoru

```
y[i] = i > 0 ? y[i-1] + a[i] : a[0];
```

Login: 31.5.2012

a) Může dojít ke kolizi b) X Nemůže dojít ke kolizi

Předpokládejte, že jednotka pro správu paměti 32-bit procesoru s 32-bit virtuální i fyzickou adresou používá pro mapování fyzické paměti do virtuálních adresních prostorů systém stránkování. Velikost jedné stránky je 4 kB. (Nápověda: Položka tabulky má 32 bitů.)

15. (1b) Kolik úrovní tabulek bude využito, pokud je požadované, aby se část stránkovací tabulky každé úrovně vešla právě do jedné stránky paměti (tato shoda velmi zjednodušuje alokaci fyzické paměti potřebné pro vlastní stránkové tabulky):

.....2.....

16. (2b) Kolik bitů z adresy bude využito pro jednotlivé použité úrovně stránkovacích tabulek a kolik pro offset ve stránce:

Nejvyšší úroveň - Root Level Table Index [bitů]	1. Vnitřní úroveň - Pointer Level Table Index 1 [bitů]	2. Vnitřní úroveň - Pointer Level Table Index 2 [bitů]	Úroveň položek mapujících jednotlivé stránky – Page Level table Index [bitů]	Offset ve stránce [bitů]
10	0	0	10	12

Jednotlivé fáze vykonávání instrukce mají zpoždění (dobu nutnou na vykonání) podle následující tabulky.

IF	ID	EX	MEM	WB
400ps	500ps	450ps	500ps	150ps

Určete délku hodinového taktu pro:

17. (0,5b) procesor s pipeline: 500ps

(0,5b) jednocyklový procesor bez pipeline: ...1600ps....

(1b) a určete horní odhad zrychlení (Speed-up), které lze dosáhnout použitím zřetězeného procesoru (uveďte vztah pro vykonání *N* instrukcí, uvažujte plnění pipeline):

$$\frac{n \cdot \sum C_i}{k C + (n-1) C}$$

n .. počet instrukcí
k ... k stupňová pipeline

18. (1b) Krátce popište co znamená spekulativní vykonávání kódu!

Procesor vykonává kód, o kterém neví jestli se opravdu má vykonávat. Když byl skok predikován dobře je kód potvrzen commit, naopak je kód ~~vrácen~~ neprovést a vrácen zpět write back.

19. (1b) Jaké existují druhy predikce skoků a kdy se vyhodnocují?

- statická
- dynamická

→ tf v době kompilace

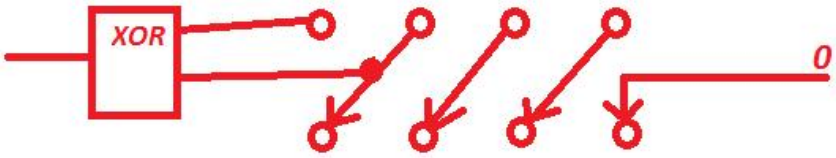
Vyhodnocují se v případě, že procesor dojde na instrukci skoku.

20. (1b) Vysvětlete pojem "podtečení" v souvislosti s reprezentací čísel IEEE-754!

Číslo je tak malé, že se nedá zapsat pomocí normalizovaných čísel. Většině podtečení se dá zabránit použitím denormalizovaných čísel. jen do určité míry pak → 0

Login: 31.5.2012

21. (2b) Nakreslete schéma obvodů pro aritmetický posun čtyřbitového čísla v doplňkovém kódu o jeden bit vlevo. Realizujte detekci přetečení.



22. (1b) Sběrnice PCI
a) X dovoluje obousměrnou komunikaci "každého s každým"
b) ○ v jednom okamžiku umožňuje libovolné množství datových přenosů
c) ○ v jednom okamžiku umožňuje pouze jeden datový přenos
d) ○ reprezentuje propojovací síť typu "křížový přepínač"

23. (2b) Transakce na 3.3V 32-bitové sběrnici PCI s taktovací frekvencí 66 MHz přenese v ideálním případě datovou strukturu

```
struct vektor {
  float data[4];
};
```

do paměti během:

a) ○ 1 taktu
b) ○ 2 taktů
c) ○ 4 taktů
d) X 5 taktů
e) ○ 9 taktů
f) ○ 128 taktů
g) ○ 264 taktů
h) ○ 66 600 000 taktů
i) ○ nelze určit

24. (2b) Kolik přístupů do paměti (počet zápisů + počet čtení) ušetří pro volání jednoduché funkce **fa()** využití návratového (link) registru (**ra/lr**) používaného pro architektury RISC oproti klasickému procesoru CISC s automatickým ukládáním návratové hodnoty na zásobník při realizaci instrukce CALL? Kolik přístupů se pak ušetří při volání a vykonávání funkce **fb()**, která **fa()** volá 3-krát a i přístupy v těchto voláních započítáme? Předpokládejte, že veškeré parametry a návratové hodnoty je možné předat přes registry.

a) ○ při volání **fa()** se ušetří 2, při **fb()** 2
b) ○ při volání **fa()** se ušetří 4, při **fb()** 0
c) ○ při volání **fa()** se ušetří 2, při **fb()** 8
d) X při volání **fa()** se ušetří 2, při **fb()** 6
e) ○ při volání **fa()** se ušetří 1, při **fb()** 4
f) ○ při volání **fa()** se ušetří 2, při **fb()** 4