T-61.123 TIETOKONEEN ARKKITEHTUURI

Tentti 10.1.2005, laatija Seppo Haltsonen

Merkitse jokaiseen tenttipaperiin:
- nimi ja opintokirjan numero
- osasto ja vuosikurssi
- harjoitustyön suorittamisvuosi, jos tehty

1. a) Selitä, mitä käsitteet tietokoneen arkkitehtuuri (computer architecture) ja tietokoneen organisaatio (computer organisation) tarkoittavat. Mitä tietokoneen osia ja ominaisuuksia kumpikin käsite sisältää?

   b) Mistä lohkoista tietokoneen keskusyksikkö koostuu? Mitkä ovat kunkin lohkon tehtävät?

2. a) Tietokoneen kellotaajuus on 500 MHz. Eräs ohjelma voidaan laatia kahdella eri tavalla. Ohjelman ensimmäisessä versiossa suoritetaan 14 miljoonaa käskyä, jotka vievät 20 miljoonaa kellojaksoa. Toisessa versiossa suoritetaan 24 miljoonaa käskyä, jotka vievät 30 miljoonaa kellojaksoa. Laske ohjelmien suoritusajat, CPI-arvot ja MIPS-arvot. Mikäli havaitset tuloksissa jotain odottamatonta, kerro mitä havaitsit ja selitä, mistä havaitsemasi seikka johtuu.

   b) Mitä tarkoittaa viivästetty haarautuminen? Kirjoita seuraava ohjelma muotoon, jossa sitä hyödynnetään.

```
Loop: lw   $2, 100($3)
      addi $3, $3, 4
      beq  $3, $2, Loop
```

3. Mitä ovat keskeytykset (interrupt) ja poikkeukset (exception)? Miten MIPS käsittelee niitä?

4. a) Mitä tarkoittaa käsite forwarding? Mihin tarkoitukseen ja miten sitä käytetään MIPSissä? Mitä tehdään silloin, kun forwarding ei ratkaise ongelmaa?

   b) Mihin liittyvät käsitteet Write through ja Write back? Selosta niiden kuvaamat toiminnot. Vertaile niitä toisiinsa.

5. Selitä yksityiskohtaisesti suoraan kuvatun välimuistin (direct mapped cache memory) toteutus ja toiminta.

MIPS R3000 Fixed-Point Instruction Set Summary

Category	Instruction	Example	Meaning	Comments
Arithmetic	add	add $1,$2,$3	$1 = $2 + $3	3 operands; exception possible
	subtract	sub $1,$2,$3	$1 = $2 - $3	3 operands; exception possible
	add immediate	addi $1,$2,100	$1 = $2 + 100	+ constant; exception possible
	add unsigned	addu $1,$2,$3	$1 = $2 + $3	3 operands; exception not possible
	subtract unsigned	subu $1,$2,$3	$1 = $2 - $3	3 operands; exception not possible
	add immediate unsigned	addiu $1,$2,100	$1 = $2 + 100	+ constant; exception not possible
Logical	and	and $1,$2,$3	$1 = $2 & $3	3 register operands; Logical AND
	or	or $1,$2,$3	$1 = $2 | $3	3 register operands; Logical OR
	and immediate	andi $1,$2,100	$1 = $2 & 100	Logical AND register, constant
	or immediate	ori $1,$2,100	$1 = $2 | 100	Logical OR register, constant
	shift left logical	sll $1,$2,10	$1 = $2 << 10	Shift left by constant
	shift right logical	srl $1,$2,10	$1 = $2 >> 10	Shift right by constant
Data transfer	load word	lw $1,(100)$2	$1 = Memory[$2+100]	Data from memory to register
	store word	sw $1,(100)$2	Memory[$2+100] = $1	Data from register to memory
	load upper immediate	lui $1,100	$1 = 100 * 2^{16}	Load constant in upper 16bits
Conditional branch	branch on equal	beq $1,$2,100	if ($1 == $2) go to PC+4+100	Equal test; PC relative branch
	branch on not equal	bne $1,$2,100	if ($1 != $2) go to PC+4+100	Not equal test; PC relative
	set on less than	slt $1,$2,$3	if ($2 < $3) $1 = 1; else $1 = 0	Compare less than; 2`s complement
	set less than immediate	slti $1,$2,100	if ($2 < 100) $1 = 1; else $1 = 0	Compare < constant; 2`s complement
	set less than unsigned	sltu $1,$2,$3	if ($2 < $3) $1 = 1; else $1 = 0	Compare less than; natural number
	set less than immediate unsigned	sltiu $1,$2,100	if ($2 < 100) $1 = 1; else $1 = 0	Compare constant; natural number
Unconditional jump	jump	j 10000	goto 10000	Jump to target address
	jump register	jr $31	goto $31	For switch, procedure return
	jump and link	jal 10000	$31 = PC + 4;go to 10000	For procedure call