Lösningar till tentamen i kursen EDA330

Datorsystemteknik D

27/5 2000

Följande är skisser till lösningar av uppgifterna. Full poäng på en uppgift kräver i de flesta fall en något fylligare motivering. I en del fall är alternativa lösningar möjliga.

1.

a. Blockoffset kräver 5 bitar. Cacheminnet innehåller totalt 32 KB/32 B = 1024 block fördelade med två per set. Alltså finns det 1024/2 = 512 set, och index kräver därför 9 bitar. Därmed återstår 32-9-5 = **18 bitar** för tag.

b. **Fysisk placering av sidan, protection bits, replacement bits, dirty bit**.

c. **Antalet bitar som krävs för blockoffset plus index vid cacheuppslagning ska vara mindre än eller lika med det antal bitar som utgör sidoffset i det virtuella minnet**. **Med cacheminnet i deluppgift a innebär detta att sidstorleken i virtuella minnet måste vara större än eller lika med 16 KB** (16 K = 2^{14} = 2^{5+9}).

d. **Primärminnesadress, blockstorlek, och skivminnesadress från processor till I/O- och DMA-enhet. Data från skivminne till I/O- och DMA-enhet. Data från I/O- och DMA-enhet till primärminne. Avbrott från I/O- och DMA-enhet till processor**.

2.

a. **Se kursboken**.

b. **ALU ctrl bestämmer vilken operation som ALU ska utföra baserat på funct-fältet i instruktionen och styrsignalen ALUOp. Om ALUOp anger addition eller subtraktion så utförs den operationen, annars utförs den operation som anges i funct-fältet. BD bestämmer vilket värde PC ska laddas med baserat på nollflaggan från ALU och Branch-signalen. Vid ej hopp laddas nästa PC med PC+4. Vid ovillkorligt hopp laddas PC med den absoluta hoppadress som ingår i jump-instruktioner. Vid hopp om lika laddas PC med resultatet av hoppadressberäkningen i EX om ALU-resultatet är noll, annars med PC+4. Vid hopp om ej lika laddas PC med resultatet av hoppadressberäkningen i EX om ALU-resultatet inte är noll, annars med PC+4**.

c.

d. **Se kursboken**.

3. (OBS! I ursprungliga uppgiftstexten stod att vart femte block skrevs tillbaka till primärminnet. Detta skulle varit 60%, vilket används nedan. Svar baserade på ursprungliga uppgiftstexten godkänns naturligtvis.)

   a. Utnyttja T = I*CPI*Tc. Tc är känd(5 ns) från klockfrekvensen. I måste lösas ut ur en eller flera ekvationer. Dela upp CPI i bidrag från instruktionscachemissar CPI_I, datacachemissar CPI_D, felaktiga hoppgissningar CPI_J, och övrigt CPI_0 som alltså är det sökta CPI-värdet i idealfallet. CPI_I = miss rate för I * miss penalty för I. Miss rate för I är enligt uppgiften 1%. Miss penalty är den tid räknat i hela processorcykler det tar att läsa ett block från primärminnet, vilket kan beräknas till 55 ns / 5 ns = 11 cykler. CPI_I är alltså 0,11. Vi räknar på motsvarande sätt för datacache, men där varierar miss rate mellan de olika implementeringsfallen och i miss penalty måste hänsyn tas till att vart femte block som kastas ut vid en miss också måste skrivas till primärminnet (vilket tar lika lång tid som en läsning). Miss penalty för D blir alltså (1+0,6)*55 ns / 5 ns. Vi ska också ta hänsyn till att det bara är var fjärde instruktion som läser eller skriver minnesdata. Totalt blir då CPI_D miss $rate_D$ * 4,4. CPIJ kan räknas ut som mispredict rate * 1/6 * 3 processorcykler = 0,5 * mispredict rate, eftersom var sjätte instruktion är en hoppinstruktion och en felaktig gissning gör att det tar tre cykler innan rätt efterföljande instruktion nått EX. Mispredict rate är i assume not taken-fallet lika med $P_{taken,}$ den sökta sannolikheten att ett hopp tas. I assume taken-fallet är mispredict rate ($1-P_{taken}$). Vi kan nu sätta upp följande ekvation för CPU-tiden för T1: T = I * Tc * (CPI_0 + CPII + CPID + CPIJ) = I * $5*10^{-9}$ * (CPI_0 + 0,11 + 4,4*miss rateD + 0,5 * mispredict rate). Med insättning av T, miss rateD, och mispredict rate för de tre fallen I0, I1, och I2 fås tre ekvationer ur vilka I, CPI_0 och P_{taken} kan lösas ut. Resultatet blir I = $2*10^6$, **andelen hopp som tas = 60%, och ideal CPI = 1,2**.

   b. I denna deluppgift används resultaten från förra deluppgiften, men hänsyn måste tas till att processorns klockcykeltid påverkas. Den kritiska vägen i fallen I1, I2, och I3 blir 5,5 ns (4,5 ns + 1 ns, respektive 4,8 ns + 0,7 ns), vilket innebär att Tc också måste ändras till 5,5 ns för I1-I3. Det i sin tur innebär att miss penalty för cacheminnena ändras. För instruktioner blir miss penalty 55 ns / 5,5 ns = 10 processorcykler. CPII blir nu alltså 0,1. CPID blir miss rateD * 4, dvs 0,4 för I0 och I2 och 0,2 för I1 och I3. CPIJ blir för I0 och I1 0,5*0,6 = 0,3, och för I2 och I3 0,5*0,4 = 0,2. Med hänsyn taget till detta kan CPU-tiden i fallen I1-I3 beräknas. För I1 blir tiden $2*10^6$ * $5,5*10^{-9}$ * (1,2 + 0,1 + 0,2 + 0,3) = 11 * 10^{-3} * 1,8 = 19,8 ms. För I2 blir tiden $2*10^6$ * $5,5*10^{-9}$ * (1,2 + 0,1 + 0,4 + 0,2) = 11 * 10^{-3} * 1,9 = 20,9 ms. För I3 blir tiden $2*10^6$ * $5,5*10^{-9}$ * (1,2 + 0,1 + 0,2 + 0,2) = 11 * 10^{-3} * 1,7 = 18,7 ms. **I3 blir bäst med körningstiden 18,7 ms**.

4.

a. Först måste tiden att få tillgång till bussen för en blocköverföring till instruktionscache beräknas. 60% av tiden är bussen ledig och det tar noll cykler. Övrig tid måste en blocköverföring av fyra ord avslutas först. En sådan överföring tar (1 cykel för att skicka adress + 75 ns (hämta två ord)/15 ns (busscykeltid) cykler + max(2 cykler (överför två ord på bussen), 25 ns/15 ns (hämta nästa två ord)) + 2 cykler (överför två ord på bussen)) = 1+5+2+2 busscykler = 10 busscykler. I genomsnitt återstår halva överföringen, så statistiskt tar det 0,6*0 + 0,4*10/2 = 2 cykler att få tillgång till bussen för en instruktionshämtning.

Hämtningstid för instruktionsblock om 2 ord = (2 cykler (få bussen) + 1 cykel (skicka adress) + 5 cykler (hämta första två ord) + 2 cykler (överför två ord på bussen)) * 15 ns (busscykeltid) = 150 ns = 150 ns/(5 ns/processorcykel) = 30 processorcykler = miss penalty vid 2 ord/block.

Hämtningstid för instruktionsblock om 4 ord = (2 cykler (få bussen) + 1 cykel (skicka adress) + 5 cykler (hämta första två ord) + 2 cykler (överför två ord, hämta nästa två) + 2 cykler (överför två ord)) * 15 ns (busscykeltid) = 180 ns = 180 ns/(5 ns/processorcykel) = 36 processorcykler = miss penalty vid 4 ord/block.

På motsvarande sätt blir miss penalty vid 8 ord/block 48 processorcykler. I fallet 16 ord/block kan hela blocket inte överföras med en minnesåtkomst, utan flera busstransaktioner krävs. Det bästa är då att göra två överföringar om 8 ord. Eftersom instruktionsöverföringar alltid har högst prioritet på bussen kan dessa ske direkt efter varandra, så miss penalty vid 16 ord/block blir 2*48 processorcykler - (2 busscykler * 3 processorcykler/busscykel) = 90 processorcykler.

Antalet stoppcykler per instruktionshämtning kan räknas ut som miss rate * miss penalty. Vid 2 ord/block blir antalet stoppcykler 4%*30 = 1,2. Vid 4 ord/block blir antalet stoppcykler 2%*36 = 0,72. Vid 8 ord/block blir antalet stoppcykler 1%*48 = 0,48. Vid 16 ord/block blir antalet stoppcykler 0,8%*90 = 0,72. Minst antal stoppcykler fås alltså vid **8 ord/block**.

b. Under en tid T hämtas T*f/CPI (f = processorfrekvens = 200 MHz) instruktioner. CPI = CPI_0 + stoppcykler/instruktionshämtning, där CPI_0 är CPI utan inverkan av instruktionshämtningsmissar. Enligt uppgiften är CPI_0 = 1,. Stoppcykler/instruktionshämtning fås från förra deluppgiften.

En instruktionsmiss belastar bussen under alla busscykler beräknade ovan utom två då bussen reserveras. Belastning på bussen (andel av tiden som bussen reserveras) under en tid T blir:

antal hämtade instruktioner under tiden T * missannolikhet * busstid/miss / T = 200 MHz/(1,6 + stoppcykler/instruktionshämtning) * missannolikhet * busstid/miss

Belastning vid 2 ord/block = 200 MHz/(1,6 + 1,2) * 4% * 8/(67 MHz) = 4%*8*3/(1,6 + 1,2) = 34%.

Belastning vid 4 ord/block = 2%*10*3/(1,6 + 0,72) = 26%.

Belastning vid 8 ord/block = 1%*14*3/(1,6 + 0,48) = 20%.

Belastning vid 16 ord/block = 0,8%*28*3/(1,6 + 0,72) = 29%.

Med **8 ord/block** fås minst andel av tiden då bussen är reserverad för läsning av instruktioner.

c. Vi räknar om antalet stoppcykler för 4 ord/block från deluppgift a enligt: Väntetid att få bussen blir 0,4*1/2*(1 cykel för att skicka adress + 75 ns (hämta två ord)/10 ns (busscykeltid) cykler + max(2 cykler (överför två ord på bussen), 25 ns/10 ns (hämta nästa två ord)) + 2 cykler (överför två ord på bussen)) = 0,2*(1+8+3+2) = 2,8 busscykler. Observera att antalet busscykler för olika operationer måste vara ett helt antal. Hämtningstid för instruktionsblock om 4 ord bli alltså 2,8+14 = 16,8 busscykler = 168 ns = 168 ns/(5 ns/processorcykel) = 34 processorcykler = miss penalty vid 4 ord/block. Antal stoppcykler blir alltså 2%*34 = 0,68. Därigenom ökar prestanda med en faktor (1,6+0,72)/(1,6+0,68) = 1,02. Bussbelastning vid 4 ord/block = 200 MHz/(1,6 + 0,68) * 2% * 14/(100 MHz) = 2%*14*2/(1,6 + 0,68) = 25%.

Processorprestanda ökar med ungefär 2%, och bussbelastningen minskar med ungefär 1%.

5.

Deluppgift	1	X	2
a	1		
b			2
c			2
d		X	
e			2
f	1		
g			2
h		X	
i			2
j	1		
k			2
l		X	