Tentamen

Datorteknik Y, TSEA28

Datum	2024-05-28
Lokal	TER1, TER2, TERE
Tid	14-18
Utb. kod	TSEA28
Modul	DIT1
Kursnamn *Provnamn*	Datorteknik Y Digital tentamen
Institution	ISY
Antal frågor	6
Antal sidor (inklusive denna sida)	6
Kursansvarig	Kent Palmkvist, 1347, Kent.Palmkvist@liu.se
Lärare som besöker skrivsalen *Telefon under skrivtiden*	Kent Palmkvist 013-281347
Besöker skrivsalen	Ca kl 15 och kl 17 (+/- 30 minuter)
Kursadministratör	Ulrika Ericsson, 013 282379
Tillåtna hjälpmedel	Inga
Betygsgränser	Preliminärt 0-20: U, 21-30: 3, 31-40: 4, 41-50: 5

Viktig information

- Hexadecimala värden anges antingen som 0x1234 eller $1234
- För de uppgifter där du måste göra uträkningar är det viktigt att du redovisar alla relevanta mellansteg
- I de uppgifter där du ska skriva assembler- eller mikrokod är det viktigt att du kommenterar koden
- Om du i en viss uppgift ska förklara något, tänk på att du gärna får rita figurer för att göra din förklaring tydligare
- Uppgifterna i tentan är inte ordnade efter svårighetsgrad
- Skriv inga svar i detta häfte!

Lycka till!

Fråga 1: Mikroprogrammering (10p)

I figur 1 återfinns en bekant datormodell som du ska mikroprogrammera i denna uppgift.

Jämfört med den modell du sett tidigare har följande förändring gjorts:
Mikroprogrammerings-ordet har utökats med ytterligare en bit (kallad R). Om R=0 fungerar datorn precis som tidigare. Om R=1 sätts styrsignaler som styr multiplexern vid GR0-GR3 till 3 (dvs register GR3 väljs), oavsett vad IR innehåller.

Notera att det är viktigt att du skriver vilken additionsoperation du valt (den som påverkar flaggorna eller den som inte påverkar flaggorna). (Om du inte skriver något speciellt kommer jag att anta att du ej vill modifiera flaggorna).

Mikrokodsminnnet innehåller i denna datormodell bland annat

addr	Mikrokod	Kommentar
0	ASR := PC, uPC := uPC+1	
1	IR := PM, PC := PC+1, uPC := uPC+1	
2	uPC := K2(M)	
3	ASR := IR, uPC := K1(OP)	; direktadressering (M=00)
4	ASR := PC, uPC := uPC + 1	; omedelbar adressering (M=01)
5	PC := PC + 1, uPC := K1(OP)	
6	ASR := IR, uPC := uPC+1	; indirekt adressering (M=10)
7	ASR := PM, uPC := K1(OP)	
8	AR := IR, uPC := uPC+1	; indexerad adressering (M=11)
9	AR := GR3+AR, uPC := uPC+1, R:=1	
0xa	ASR := AR, uPC := K1(OP)	

(a) (8p) Skriv mikrokod för instruktionen ADD_FORCE_POS GRx, M, A. Instruktionen ska addera GRx med ett värde ur minnet enligt adresseringsmode M, där båda talen antas vara i 2-komplementsform. Svaret ska sparas i GRx. Om additionen generar ett negativt tal ska värdet 0 sparas i GRx istället. Flaggor får påverkas. Ange adresserna för mikrokodsraderna som absoluta numeriska värden.

Exempel: Antag GR2=0x0034, PM(0x35)=0x1234. Om instruktionen ADD_FORCE_POS GR2, M=00, A=0x35 körs resulterar det i GR2 = 0x1268.

Exempel: Antag GR1=0xc123, GR3=0x40, PM(0x40)=0x0123. Om instruktionen ADD_FORCE_POS GR1, M=11, A=0 körs resulterar det i GR1=0x00 då summan 0xc123+0x0123=0xc246 är negativ.

(b) (2p) Antag instruktionen ADD_FORCE_POS GR0, M=10, A=0x70 utförs om GR0=0x9123, PM(0x70)=0x0045, och PM(0x45)=0x8345. Hur många klockcykler krävs för att utföra instruktionen?

Fråga 2: Allmän teori (10p)

(a) (2p) Beskriv och ange namnet för en metod som undviker data-konflikter i en RISC-processor.

(b) (2p) Vad skiljer SRAM från DRAM?

(c) (2p) Vad är skillnaden mellan en processor av hardvard-typ jämfört med en processor av von-Neumann typ?

d) (2p) Antag en gruppassociativ cache har en viss storlek och 4 vägar. Hur ändras antal adressbitar som används som tag om associativiteten dubbleras respektive halveras (men cachen storlek inte ändras)?

(e) (2p) Antag en 4-vägs superskalär processor har klockfrekvensen 100 MHz, och en vanlig 5 stegs RISC-processor har klockfrekvensen 200 MHz. Vad är absolut max antal instruktioner per tidsenhet som respektive processor kan utföra?

Fråga 3: Assemblerprogrammering (10p)

Skriv en subrutin som letar igenom en tabell med 16-bitars värden efter det minsta värdet. Inte alla värden i tabellen ska ingå i sökandet, utan bara de som har bit 0 = 1 och bit 14 = 0. Bit 13 till och med bit 1 ska tolkas som ett 13-bitars 2-komplementstal när man letar efter minsta värdet.

Subrutinen anropas med r0 innehållande adress till tabellens start, och r1 innehåller antal värden i tabellen. När subrutinen returnerar ska r0 innehålla det minsta värdet.

Minnescellerna som innehåller tabellen får inte ändras av subrutinen.EA28-24-05-28.pdf

Exempel: r0 = 0x20001002, r1=0x00000006

Minnesinnehåll

Adress	Värde	Adress	Värde
0x20001000	0x1076	0x20001008	0x869B
0x20001002	0x2149	0x2000100A	0x1945
0x20001004	0xCF1E	0x2000100C	0x3F71
0x20001006	0x7021	0x2000100E	0x0002

De rader i tabellen som ska jämförs innehåller då värdena 0x2149, 0x869B, 0x1945 och 0x3F71. Motsvarande 16-bit tvåkomplementvärden är då (teckenförlängt och skiftat höger) 0xF0A4, 0x034D, 0x0CA2 samt 0xFFB8. Ordnat i minst först för 2-komplement blir det 0xF0A4 < 0xFFB8 < 0x034D < 0x04A2. Så subrutinen returnerar med r0 = 0xF0A4.

Fråga 4: Avbrott (8p)

Skriv en avbrottsrutin styr en värmeanläggning. Måltemperatur finns sparad som en byte på adress 0x20001000. Avbrottsrutinen ska läsa aktuell temperatur som en byte från adress 0x40001000. Adress 0x40001001 får inte läsas. Om temperaturen är större än (måltemperatur + 1) ska temperaturstyrningen minskas, och om temperaturen är lägre än (måltemperatur – 1) så ska temperaturstyrningen ökas.

Temperaturstyrningen ökas genom att subrutinen ctlincrease anropas. Temperaturstyrningen minskas genom att subrutinen ctldecrease anropas. Dessa båda subrutinen antas vara skrivna av någon annan (ska ej skrivas). Båda dessa subrutiner påverkar registret R5.

Inga andra avbrott får starta medan denna avbrottsrutin kör.

Exempel: Avbrottsrutinen läser in värdet 0x25 från adress 0x40001000. I minnet på adress 0x20001000 ligger värdet 0x23. Eftersom 0x25 > (0x23+1) anropas subrutinen ctldecrease.

Fråga 5: Aritmetik (6p)

(a) (2p) Antag register r3=0x12345678 och r4=0x71B52D41 och instruktionen ADD r3,r4 körs. Vad blir värdet i r3 och värdet på flaggorna Z, C, N, och V?

(b) (2p) Beräkna värdet i decimal form för 2-komplementsvärdet 11001_{2C}.

(c) (2p) Beräkna differensen $01101_{2C} - 101_{2C}$ (båda är 2-komplementstal med 6 respektive 4 bitars ordlängd). Beräkna subtraktionen A-B som summan A + (-B). Ange svaret som ett 8-bitars 2-komplementstal.

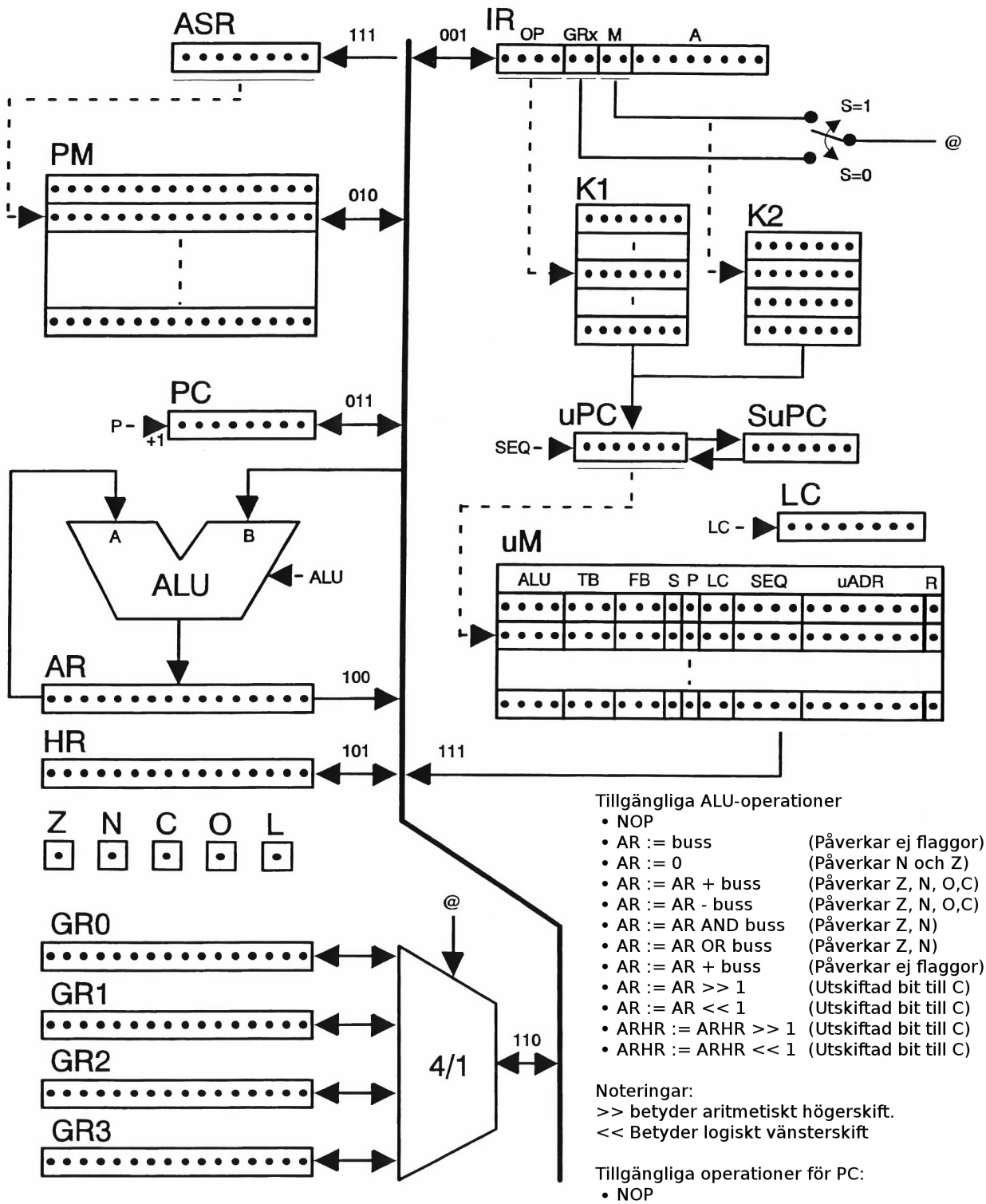
Fråga 6: Cache (6p)

En processor har en 8-vägs gruppassociativ cache. Adressbussen är 32 bitar lång. Varje cacheline är 32 byte lång. 21 adressbitar används som tag av cachen.

(a) (2p) Hur stor är denna cache? Ange svaret som ett numeriskt värde.

(b) (2p) Hur många adressbitar används som index?

(c) (2p) Antag cachen är tom. Hur länge kan adresser läsas enligt sekvensen 0x2456789C + n*0x040 (n = 0,1,2,3,..7.) göras innan tidigare inläst data kastas ut ur cachen?



Tillgängliga ALU-operationer
- NOP
- AR := buss (Påverkar ej flaggor)
- AR := 0 (Påverkar N och Z)
- AR := AR + buss (Påverkar Z, N, O,C)
- AR := AR - buss (Påverkar Z, N, O,C)
- AR := AR AND buss (Påverkar Z, N)
- AR := AR OR buss (Påverkar Z, N)
- AR := AR + buss (Påverkar ej flaggor)
- AR := AR >> 1 (Utskiftad bit till C)
- AR := AR << 1 (Utskiftad bit till C)
- ARHR := ARHR >> 1 (Utskiftad bit till C)
- ARHR := ARHR << 1 (Utskiftad bit till C)

Noteringar:
>> betyder aritmetiskt högerskift.
<< Betyder logiskt vänsterskift

Tillgängliga operationer för PC:
- NOP
- PC := PC + 1 (PC räknas upp med ett)
- PC := buss

Tillgängliga operationer för LC:
- NOP
- LC räknas ned med ett
- LC laddas från uADR

Tillgängliga operationer för uPC:
- uPC := uPC + 1 (uPC räknas upp med ett)
- uPC := K1(OP)
- uPC := K2(M)
- uPC := 0
- uPC := uADR
- uPC := uADR om (valfri) flagga är 1, annars uPC+1
- uPC := uADR om (valfri) flagga är 0, annars uPC+1

Under varje klockcykel kan även en av följande enheter skicka data från bussen: IR, PM, PC, AR, HR, GRx eller uM. En av följande enheter kan också ta emot data ifrån bussen varje klockcykel: IR, PM, PC, AR, HR, GRx eller ASR Viktigt: När uM (de 16 minst signifikanta bitarna) skickas ut till bussen kan enbart en ALU-operation och/eller en bussöverföring göras. uPC räknas också automatiskt upp med ett i detta fall.

Signalen S väljer om M eller GRx-fältet ska användas till att adressera GR0-GR3 (S=0 innebär att GRx-fältet adresserar GR0-GR3). Slutligen används signalen R=1 för att tvinga styrsignalen @ att bli 3.

Figur 1: En välkänd datormodell (Björn Lindskog 1981)

Kort repetition av ARM Cortex-M4

Mnemonic	Kort förklaring	Mnemonic	Kort förklaring
ADR	Address	CPSID	Disable interrupt
ADD, ADDS	Addition	CPSIE	Enable interrupt
ADDC,ADDCS	Addition with C flag	EOR,EORS	Logic XOR
AND,ANDS	Logic AND	LDR	Load register
ASR,ASRS	Arithmetic shift right	LDRB,LDRSB	Load register byte
B	Branch	LDRH,LDRSH	Load register halfword
BCC,BLO	Branch on carry clear	LSL,LSLS	Logic shift left
BCS,BHI	Branch on carry set	LSR,LSRS	Logic shift right
BEQ	Branch on equal	MOV,MOVS	Move
BGE	Branch on greater than or equal	MUL,MULS	Multiply
BGT	Branch on greater than	MVN	Move not
BL	Branch and link	NOP	No operation
BLT	Branch on less than	ORR,ORRS	Logic OR
BLE	Branch on less than or equal	POP	Pop
BLS	Branch on lower or same	PUSH	Push
BLT	Branch on less than	ROR	Rotate right
BMI	Branch on minus	SBC,SBCS	Subtraction with C flag
BNE	Branch on not equal	STR	Store register
BPL	Branch on plus	STRB	Store register byte
BVC	Branch on overflow clear	STRH	Store register halfword
BVS	Branch on overflow set	SUB,SUBS	Subtraction
BX	Branch and exchange	TST	Test
CMP	Compare (Destination - Source)		

```
; Exempel på ARM Cortex-M4 kod som kopierar 200 bytes från 0x2000 till 0x3000

main:  mov  r0,#(0x2000 & 0xffff)
       movt r0,#(0x2000 >> 16)
       mov  r1,#(0x3000 & 0xffff)
       movt r1,#(0x3000 >> 16)
       mov  r3,#50
loop:
       ldr  r4,[r0],#4
       str  r4,[r1],#4
       subs r3,r3,#1
       bne  loop
```