Tentamen

Datorteknik Y, TSEA28

Datum	2015-08-18
Lokal	TERE, TER4
Tid	14-18
Kurskod	TSEA28
Provkod	TEN1
Kursnamn *Provnamn*	Datorteknik Y Skriftlig tentamen
Institution	ISY
Antal frågor	7
Antal sidor (inklusive denna sida)	6
Kursansvarig	Kent Palmkvist, 1347, kentp@isy.liu.se
Lärare som besöker skrivsalen *Telefon under skrivtiden*	Kent Palmkvist 1347, 013-281347
Besöker skrivsalen	Cirka kl 15 och 17
Kursadministratör	Gunnel Hässler
Tillåtna hjälpmedel	Inga
Betygsgränser	Preliminärt 0-20: U, 21-30: 3, 31-40: 4, 41-50: 5
Visning	Onsdag 2 September kl 12.30 – 14.00 i kursansvarigs kontor

Viktig information

- 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 (8p)

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

Jämfört med den modell ni sett tidigare har följande förändring gjorts: Mikroprogrammeringsordet 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).

Datorn har just nu följande instruktioner implementerade:

Opcode	Instruktion	Betydelse	Adresserings moder (M)	Påverkar flaggor
0010	JNE ADR	PC := ADR om Z = 0 annars PC := PC+1	- (ange 00)	-
1000	LOAD GRx,M,ADR	GRx := PM(A)	00,01,10	-
1001	STORE GRx,M,ADR	PM(A) := GRx	00,10	-

Mikrokodsminnet innehåller (uPC++ och R:=0 antas om inget annat anges).

addr	Mikrokod	Kommentar
0	ASR := PC	
1	IR := PM, PC := PC+1	
2	uPC := K2(M)	
3	ASR := IR, uPC := K1(OP)	; direktadressering (M=00)
4	ASR := PC, PC := PC+1, uPC := K1(OP)	; omedelbar adressering (M=01)
5	ASR := IR	
6	ASR := PM, uPC := K1(OP)	; indirekt adressering (M=02)
7	GRx := PM, S := 0, uPC := 0	; LOAD
8	PM := Grx, S := 0, uPC := 0	; STORE
9	uPC := 0 om Z = 1, annars uPC++	; JNE addr
10	PC := IR, uPC := 0	

(a) (1p) Ange innehåll i minnet K2

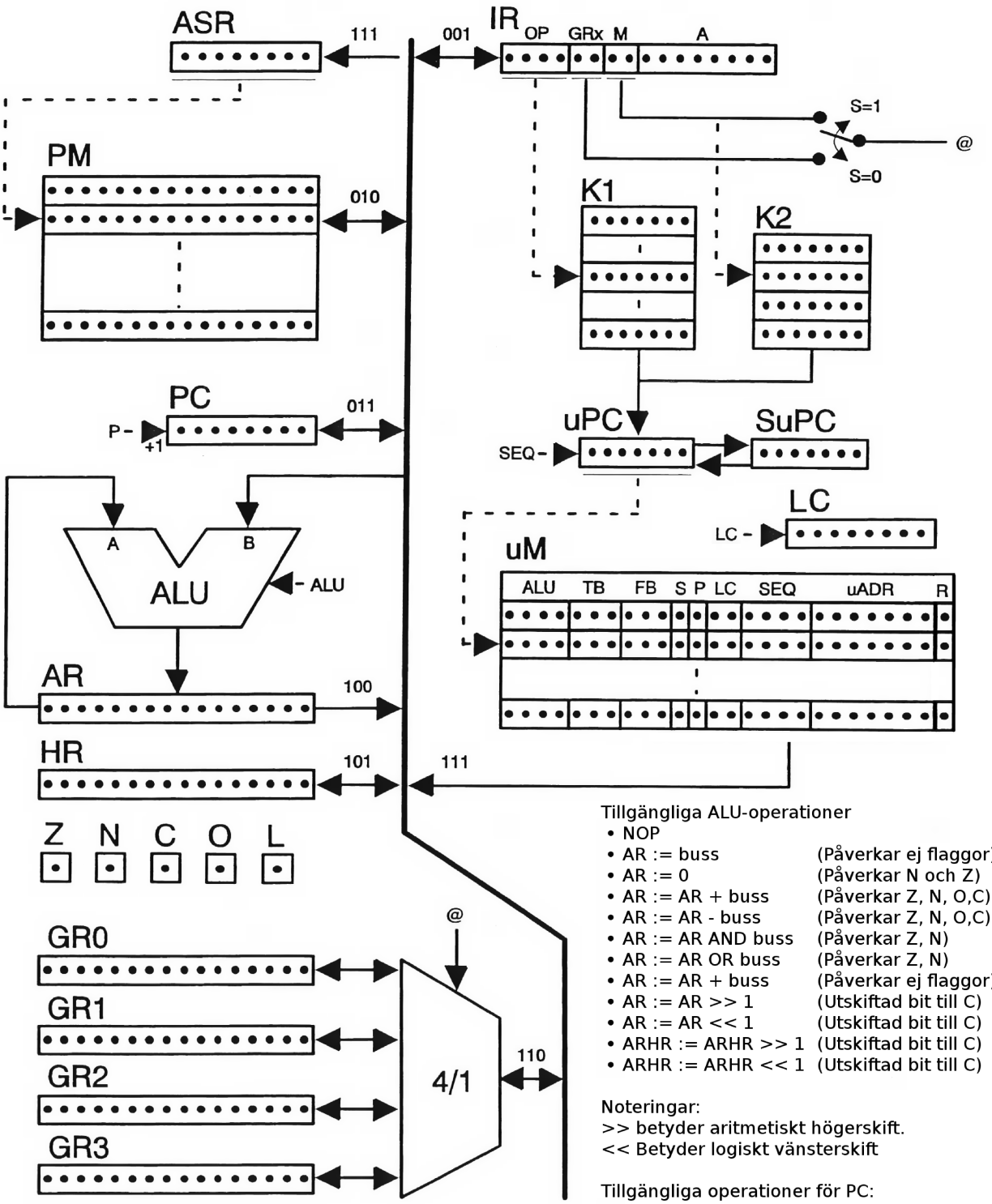
(b) (5p) Ändra/lägg till innehåll i mikrokoden så att instruktionen DECR läggs till.

Opcode	Instruktion	Betydelse	Adresseringsmoder	Påverkar flaggor
0100	DECR GRx	GRx := GRx - 1	- (ange 00)	Z,N,O,C

Exempel: GR2 innehåller värdet 42. Efter DECR GR2 skall GR2 innehålla 41 och flaggorna skall vara satta till 0.

(c) (1p) Ange det hexdecimala värdet för maskininstruktionen DECR GR1.

(d) (1p) Hur stort är minnet K2? Ange antal addresser och antal bitar i varje address.



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)

Fråga 2: Allmän teori (10p)

(a) (2p) Vad är skillnaden mellan ett FLASH minne och RAM minne?

(b) (2p) Ange två extrafunktioner som en modern cache har för att öka sannolikheten för en cacheträff respektive minska svarstid vid cachemiss.

(c) (2p) Vad har I-flaggorna (I2,I1,I0) för funktion i M68000 processorn?

(d) (2p) Hur fungerar virtuellt minne?

(e) (2p) Vad är skiljer en associativ cache från en direktmappad cache?

Fråga 3: Assemblerprogrammering (10p)

Skriv en subrutin som skriver ut värdet av de 16 minst signifikanta bitarna i D0 i hexadecimal form. Tillgängligt finns en subrutin på adress $4010 som skriver ut ett ASCII-tecken i register D0 (lägst 8 bitarna), och som inte påverkar några register.

ASCII-kod för tecknen '0' till '9' är $30 till $39, och 'A' till 'F' har ASCII-kod $41 till $46.

Exempel: D0 = $C1023A56 ska ge utskrift av de 4 ASCII-tecknen $33, $41, $35, $36.

Fråga 4: Avbrott (8p)

En reklamskylt styrs av en M68000 processor där port A och port B (8 bitar var) på en ansluten PIA-krets styr alla lampor som sitter runt skylten. För att få ljuset att rotera ansluts en klocka som ger ett avbrott var 200:e ms via PIA-kretsen. Rotationen ska skifta alla bitar ett steg åt vänster, och mest signifikant bit hos port B ska flyttas till minst signifikant bit i port A, och Processorn används även till andra uppgifter kopplade till styrning av hur starkt en lampa ska lysa beroende på hur ljust det är i omgivningen.

Skriv en avbrottsrutin som roterar alla bitarna i de båda portarna på PIA-kretsen. Dataregistret för port A har adress $10080 och för port B adress $10082.

Exempel: Antag port A har värde $6E port B har värde $84. Efter ett avbrott ska då port A ha värdet $DD och port B ha värdet $08.

Fråga 5: Aritmetik (4p)

(a) (2p) Antag operationen ADD.B D0,D1 beräknas då D0 = $BA och D1 = $6C beräknas (addition av två 8-bitars tal). Ange vad beräkningen ger för resultat (i binär form) och resulterande värden hos flaggorna C, V och Z.

(b) (2p) Beskriv talet -12 (decimalt) på 8-bitars 2-komplementsform.

Fråga 6: Cache (5p)

En processor har en så kallad level 1 cache på 32 Kbyte (32768 bytes). Denna cache är två-vägs gruppassociativ med en cachelinelängd av 16 bytes, där ersättningsalgoritmen behåller det värde som senast lästs. Ett program som körs på processorn visar följande adresseringssekvens (läser ett 16-bitars ord i varje minnesåtkomst):

Minnesåtkomst nummer 1-4	$004004, $004000, $004040, $004044
Minnesåtkomst nummer 5-8	$008080, $008008, $068000, $068008
Minnesåtkomst nummer 9-12	$006502, $068020, $056000, $056002
Minnesåtkomst nummer 13-16	$004004, $008080, $068000, $056000

(a) (3p) Ange för varje adress i sekvensen om det blir en cacheträff eller cachemiss. Antag att cachen från början är helt tom.

(b) (2p) Hur många cachemissar fås om cacheminnet är dubbelt så stort? Antag att cacheline fortfarande är 16 bytes och att det är en två-vägs gruppassociativ.

Fråga 7: Programförståelse (5p)

(a) (2p) Antag stacken innan subrutinanropet till $1000 är satt till $7000. Vilken adress pekar stackpekaren på när instruktionen på adress $102C utförs?

(b) (3p) Vilket värde får register D0 efter att nedanstående program anropats som en subrutin (med startadress $1000)?

```
ADRESS    MASKINKOD            INSTRUKTION
-----------------------------------------------
001000    4280                 CLR.L   D0
001002    207C00001032         MOVE.L  #$1032,A0
001008    123C0030             MOVE.B  #$40,D1
00100C    610E                 BSR.S   $00101C
00100E    207C00001038         MOVE.L  #$1038,A0
001014    123C0020             MOVE.B  #$20,D1
001018    6102                 BSR.S   $00101C
00101A    4E75                 RTS
00101C    D058                 ADD.W   (A0)+,D0
00101E    6108                 BSR.S   $001028
001020    04010010             SUB.B   #$10,D1
001024    66F6                 BNE.S   $00101C
001026    4E75                 RTS
001028    0C501234             CMP.W   #$1234,(A0)
00102C    6602                 BNE.S   $001030
00102E    221F                 MOVE.L  (A7)+,D1
001030    4E75                 RTS
001032    0001                 DC.W    $0001
001034    0002                 DC.W    $0002
001036    1234                 DC.W    $1234
001038    0010                 DC.W    $0010
00103A    0020                 DC.W    $0020
```

Kort repetition av M68000

Mnemonic	Kort förklaring	Mnemonic	Kort förklaring
ADD	Addition	EXT	Sign extend
ADDX	Add with X flag	EXTB	Sign extend a byte to 32 bit
AND	Logic and	JSR	Jump to subroutine
ASL	Arithmetic shift left	LEA	Load effective address
ASR	Arithmetic shift right	LSL	Logic shift left
BCC	Branch on carry clear	LSR	Logic shift right
BCS	Branch on carry set	MOVE	Move
BEQ	Branch on equal	MULS	Signed multiplication
BGT	Branch on greater than	MULU	Unsigned multiplication
BLT	Branch on less than	NEG	Negate
BNE	Branch on not equal	NOP	No operation
BRA	Branch always	NOT	Bitwise logic invert
BSR	Branch to subroutine	OR	Logic OR
CLR	Clear	ROL	Rotate left
CMP	Compare (Destination - Source)	ROR	Rotate right
DIVS	Signed division	RTE	Return from exception
DIVU	Unsigned division	RTS	Return from subroutine
EOR	Logic XOR	SUB	Subtract
EXG	Exchange	TST	Set integer condition codes

```
; Exempel på M68000 kod som kopierar 200 bytes från $2000 till $3000
    MOVE.L #$2000,A0
    MOVE.L #$3000,A1
    MOVE.B #50,D0
loop
    MOVE.L (A0)+,(A1)+
    ADD.B #-1,D0
    BNE loop
```