Tentamen

Datorteknik Y, TSEA28

Datum	2016-10-18
Lokal	TER1
Tid	8-12
Kurskod	TSEA28
Provkod	TEN1
Kursnamn *Provnamn*	Datorteknik Y Skriftlig tentamen
Institution	ISY
Antal frågor	7
Antal sidor (inklusive denna sida)	4
Kursansvarig	Kent Palmkvist, 1347, kentp@isy.liu.se
Lärare som besöker skrivsalen *Telefon under skrivtiden*	Kent Palmkvist 1347, 013-281347
Besöker skrivsalen	Cirka kl 9 och 11
Kursadministratör	Gunnel Hässler
Tillåtna hjälpmedel	Inga
Betygsgränser	Preliminärt 0-20: U, 21-30: 3, 31-40: 4, 41-50: 5
Visning	Onsdag 2 November 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 (10p)

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).

Mikrokodsminnnet innehåller (uPC=uPC+1 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=10)
7	PM := GRx, S := 0, uPC := 0	; STORE
8	uPC := 0 om Z = 1, annars uPC++	; JNE addr
9	PC := IR, uPC := 0	
10	GRx := PM, S=0, uPC := 0	; LOAD

(a) (6p) Skriv mikrokod för instruktionen RTS. Register GR3 ska användas som stackpekare, och stacken ska växa på samma sätt som för tutor (GR3 pekar på senaste värdet lagt på stacken, nytt värde som läggs på stacken hamnar på närmast lägre adress i minnet).

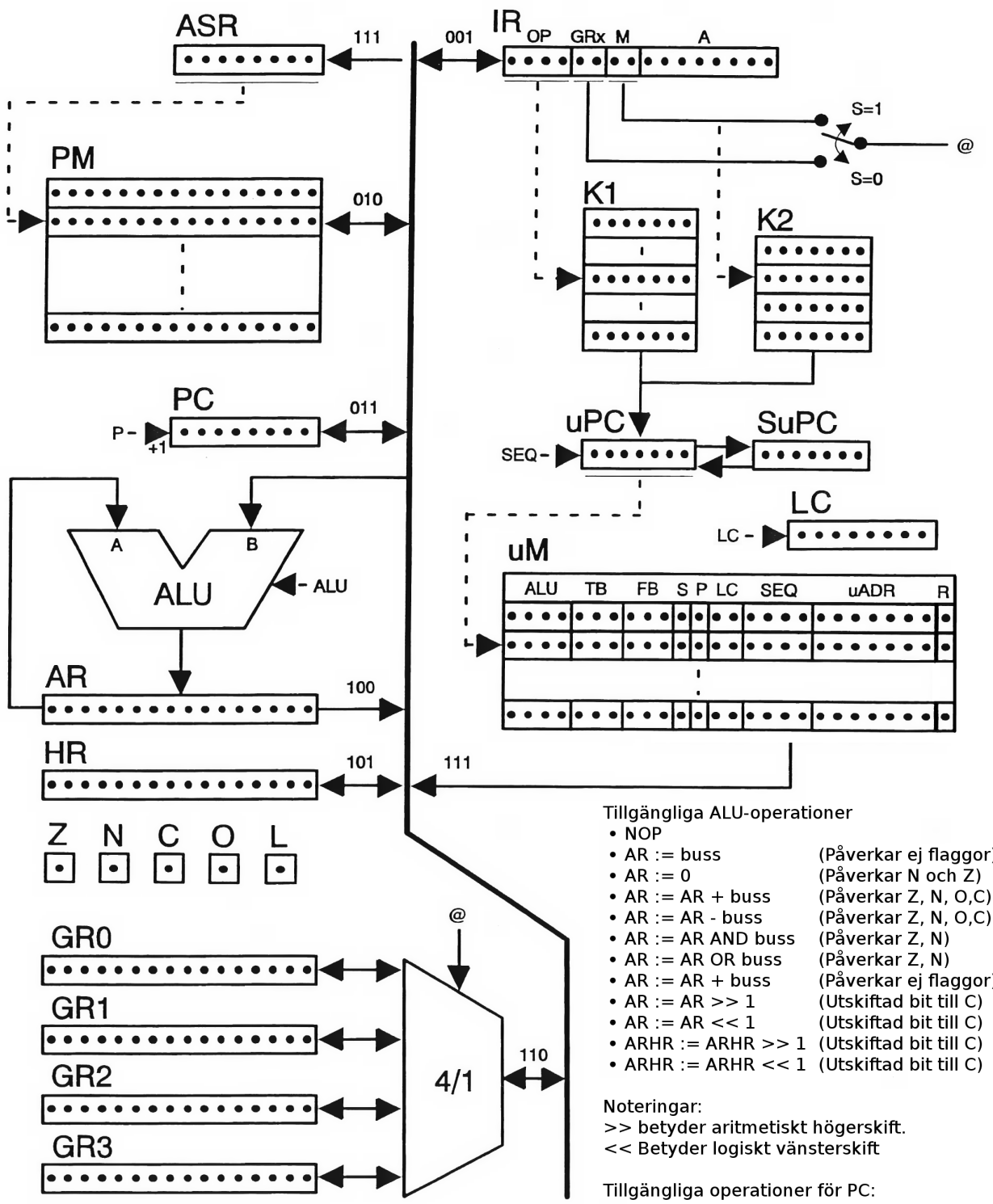
Inkludera även address för mikrokodsinstruktionerna.

Opcode	Instruktion	Betydelse	Adresseringsmoder	Påverkar flaggor
0001	RTS	PC := PM(GR3), GR3:=GR3+1	- (ange 00)	-

Exempel: GR3 = $37, PM($37) = $11. Efter RTS ska PC:= $11, GR3 = $38

(b) (3p) Opcode för JNE är 0101, opcode för LOAD är 1100, opcode för JNE är 1101. Ange det binära innehåller i K1 (inkluderat RTS). Ange värdet '-' för ej definierade bitar i K1. Mikrominnet har 128 adresser.

(c) (1p) Hur många klockcykler (dvs steg) tar det att utföra en hel RTS-instruktion?



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) Ange två halvledarminnen som förlorar innehållet när spänningen stängs av.

(b) (2p) Ange vilken typ av halvledarminne används som cacheminne i en processor, och motivera varför denna typ används.

(c) (2p) Vilka kombination av flagvärden visar att summering av två stycken tvåkomplementstal gav ett negativt resultat?

(d) (2p) Vad är skillnaden mellan en SIMD-processor och en generell processor (t ex M68000).

(e) (2p) Varför är villkorliga hopp svåra att hantera i en pipelinad RISC-processor? Nämn ett sätt att minska inverkan av dom.

Fråga 3: Assemblerprogrammering (7p)

I minnet ligger ett datapaket som mottagits från ett nätverkskort.

(a) (6p) Skriv en subrutin som beräknar checksumman för datapaketet. Indata till subrutinen är paketets startadress i A0, och längden (antal byte) hos paketet anges i D0. Returnera i D0 checksumman (xor av alla byte i paketet).

Exempel: A0 = $00004022, D0 = $06, minnet innehåller

Adress	Värde	Adress	Värde
$00004020	$123B5678	$00004028	$015123B0
$00004024	$11A12222	$0000403C	$CD121212

Efter anrop till subrutinen ska D0 innehålla $9E (dvs $56 xor $78 xor $11 xor $A1 xor $22 xor $22).

(b) (1p) Vad blir resultatet om subrutinen anropas med A0 = $00004028 och D0 = $05 när minnets innehåll är som i (a) ovan?

Fråga 4: Avbrott (8p)

I en styrenhet ska ett tutor-system användas för att styra nivån i en vattentank. En sensor som är ansluten till en port på adress $10040 visar med hjälp av 32 bitar hur mycket vatten det finns i tanken. Varje bit i det 32-bitars värdet anger om vattnet nått en viss nivå i tanken. Om tanken är tom är värdet $00000000, och om mängden vatten ökas så ökar värdet enligt sekvensen $00000001, $00000003, $00000007, $0000000F,$0000001F ... $7FFFFFFF, $FFFFFFFF, dvs det finns 33 olika möjliga värden.

Tyvärr kan det bli lite vågor i tanken, så ibland ger sensorn värden såsom $0000017F (dvs en nolla i raden av 1:or). Vid dessa fall ska rutinen ange antal 1:or som finns i sensorvärdet, dvs i exemplet blir det 8.

Skriv en avbrottsrutin som triggas av en timer. Denna avbrottsrutin ska läsa av porten på adress $10040 och översätta värdet till ett värde mellan 0 och 32 vilket avbrottsrutinen ska spara på adress $5000. Avbrottet nollställs när adress $10040 läses.

Exempel: Om $10040=$000005FF ska avbrottsrutinen skriva värdet 10 på adress $5000.

Fråga 5: Stack, programförståelse (4p)

I ett datorvirus hittas subrutinen nedan.

```
ADRESS  MASKINKOD       ASSEMBLERKOD
1000    203C 0000 1012  Subrutin:  move.l #$1012,D0
1006    2F00                       move.l D0,-(A7)
1008    203C 0000 102c             move.l #$102c,D0
100e    2F00                       move.l D0,-(A7)
1010    4E75                       rts
1012    203C 0000 101a             move.l #$101a,D0
1018    4E75                       rts
101a    4EBA FFF6                  jsr $1012
101e    203C 0000 102c             move.l #$102c,D0
1024    2F3C 0000 1034             move.l #$1034,-(A7)
102a    4E75                       rts
102c    201F                       move.l (a7)+,D0
102e    4EBA FFDE                  jsr $100e
1032    4E75                       rts
1034    203C 0000 1034             move.l #$1034,D0
103a    4E75                       rts
```

(a) (2p) Vilket värde har register D0 när subrutinen är klar?

(b) (2p) Hur många instruktioner utförs i subrutinen?

Fråga 6: Cache (6p)

En processor med 32-bitars adress har en cache på 16 Kbyte (16384 bytes, dvs 2^14 bytes). Denna cache är en 4-vägs gruppassociativ cache med cachelinelängd av 8 byte. Ersättningsalgoritmen behåller det värde som lästs senast. Ett beräkningsprogram som körs i processorn läser minnet på adresserna enligt sekvensen: $1000+$100*n, dvs $1000, $1100,$1200 etc. Antag att cache från början är tom.

(a) (2p) Hur många cachelines finns i cacheminnet totalt?

(b) (2p) Hur många bitar av adressen används som tag i cacheminnet?

(c) (2p) Hur många läsningar hinner beräkningsprogrammet göra innan gamla värden börjar tas bort ur cachen?

Fråga 7: Aritmetik (5p)

(a) (2p) Ange den binära representationen av det 8-bitars 2-komplementstal som har värdet -7.

(b) (3p) Beräkna produkten (i 2-komplementsform) av de två 6-bitars 2-komplementstalen 110011 och 001100.

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
```