



Norwegian University of Science and Technology
Engineering
The Department of Computer and Information Science

TDT4160

DATAMASKINER GRUNNKURS

EKSAMEN

6. AUGUST, 2015, 09:00–13:00

Kontakt under eksamen:

Odd Rune Lykkebø 473 24 556

Tillatte hjelpemidler:

D.

Ingen trykte eller håndskrivne hjelpemiddel er tillat.

Enkel godkjent kalkulator er tillat.

Målform:

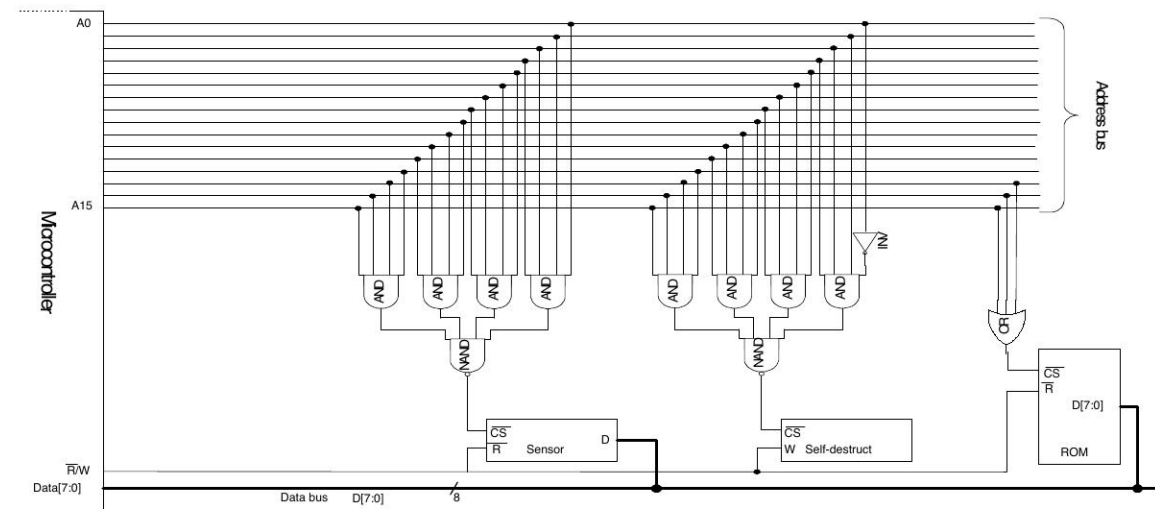
Bokmål

OPPGAVE 1: OPPSTART, LITT AV HVERT (25 %)

- a. Forklar kort hva som ligger i begrepet lokalitet (locality) i sammenheng med hurtigbuffer (cache).
- b. Hva er gjennomsnittlig aksesstid for et minnesystem med ett nivå hurtigbuffer (cache) der hovedminnet har en aksesstid på $1\mu s$, hurtigbuffer har en aksesstid på $0,05\mu s$ og trefferholdstall for systemet (hit ratio) er 90 %?
- c. Forklar kort forskjellen på asynkron og synkron bussoverføring?
- d. Hva kjennetegner en prosessor med Harvard-arkitektur? Forklar kort.
- e. Forklar kort hva begrepet ILP er og hvordan dybden på samlebånd (pipelines) påvirker ILP.

OPPGAVE 2: DIGITALT LOGISK NIVÅ (25 % (15 % PÅ A, 5 % PÅ B OG C))

I en rakett benyttes det en mikrokontroller for å overvåke, og eventuelt avbryte oppskytingen. Figur 1 viser det eksterne bussgrensesnittet for mikrokontrolleren. Det er en ROM-brikke for program. En sensor som overvåker kursen og en enhet for å avbryte oppskytingen. Alle enhetene benytter et aktivt lavt (logisk "0") CS (Chip Select)-signal. Det er 1K byte RAM tilgjengelig internt i mikrokontrolleren.



Figur 1: Address decoding.

- a.
 - i) Tegn minnekart for systemet.
 - ii) Hva er det største programmet (i byte) som får plass i ROM.
- b. Etter flere litt uheldige episoder blir det bestemt å endre programmet som ligger i ROM. Det viser seg da at det er behov for mer RAM for å forbedre systemet. Hvor mye RAM er det mulig å utvide med?
- c. Ut i fra den informasjonen du har tilgjengelig:
Hva slags signal(er) er nødvendig for å avbryte oppskytingen? Forklar kort.

OPPGAVE 3: MIKROARKITEKTUR OG MIKROINSTRUKSJONER (25 % (5 % PÅ A, 10 % PÅ B OG C))

Bruk vedlagte diagram i figur 3, figur 4, figur 5 og figur 6 for IJVM til å løse oppgavene.

- a. i) Er det mulig å skrive resultatet fra en ALU-operasjon direkte til registeret MBR? Begrunn svaret.
ii) Når i klokkeperioden blir registerinnhold oppdatert i IJVM? Forklar kort.
- b. Lag mikroinstruksjon(er) for følgende IJVM-operasjon: $CPP = CPP + LV$.
Du behøver ikke ta hensyn til Addr- og J-feltene. Oppgi korrekt bit-verdi for ALU-, C-, Mem- og B-feltene. – Se figur 4.
- c. IJVM-registrene i figur 3 er satt til følgende verdier:

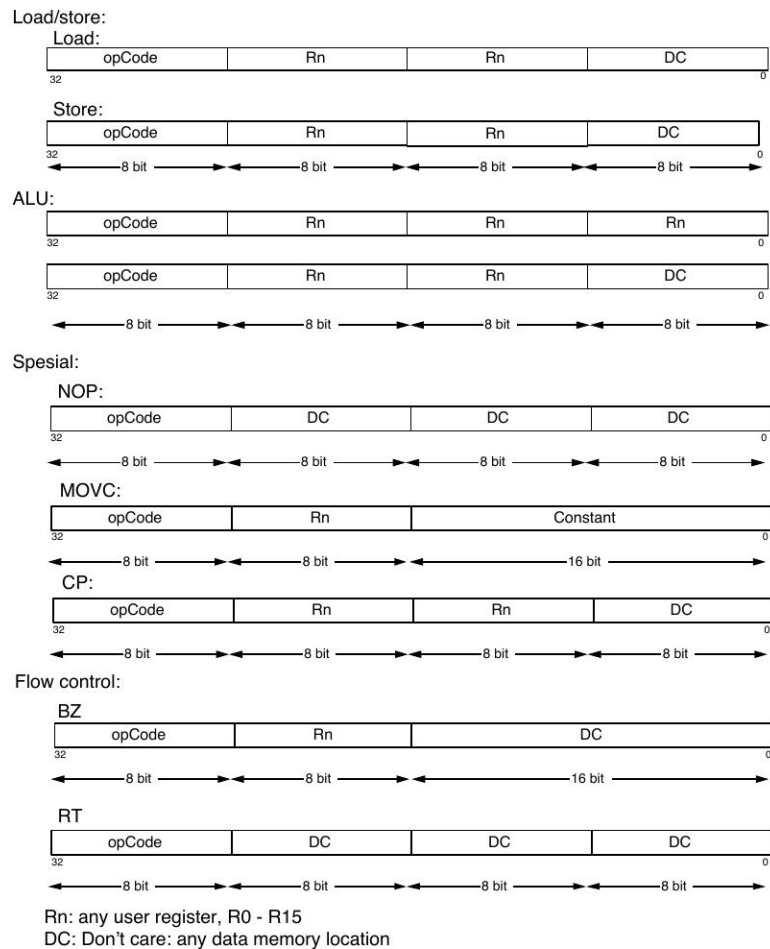
"SP": hex(0001),
"LV": hex(0505),
"CPP": hex(AAAA),
"TOS": hex(5555),
"OPC": hex(0000),
"H": hex(FF0A).

Hva er innholdet i H-registeret og TOS-registeret etter at følgende mikroinstruksjoner har blitt utført? Oppgi svaret i hex-format.

- 1: ALU: 010100, C: 100000000, Mem: 000 og B: 0110
- 2: ALU: 011100, C: 101000000, Mem: 000 og B: 0111
- 3: ALU: 001100, C: 001000000, Mem: 000 og B: 1000

OPPGAVE 4: INSTRUKSJONSSETT ARKITEKTUR (ISA)(25 %)

En svært enkel prosessor har en load, en store, 8 ALU-instruksjoner og noen spesialinstruksjoner, inkludert NOP-instruksjonen og to flytkontrollinstruksjoner (flow control instructions).. Instruksjonsformatet for instruksjonene er vist i figur 2. Alle register og busser er 32-bit. Prosessoren har en Harvard architecture.



Figur 2: Instruction formats.

Instructions set:

LOAD: Load data from memory.

load Rn, Rn Load register Rn from memory location in Rn.

STORE: Store data in memory.

store Rn, Rn Store register Rn in memory location in Rn.

ALU: Data manipulation, register-register operations.

ADD Rn, Rn, Rn ADD, $Rn = Rn + Rn$. Set Z-flag if result =0.

NAND Rn, Rn, Rn Bitwise NAND, $Rn = \overline{Rn} \cdot \overline{Rn}$. Set Z-flag if result =0.

OR Rn, Rn, Rn Bitwise OR, $Rn = Rn + Rn$. Set Z-flag if result =0.

INV Rn, Rn Bitwise invert, $Rn = \overline{Rn}$. Set Z-flag if result =0.

INC Rn, Rn Increment, $Rn = Rn + 1$. Set Z-flag if result =0.

DEC Rn, Rn Decrement, $Rn = Rn - 1$. Set Z-flag if result =0.

MUL Rn, Rn, Rn Multiplication, $Rn = Rn * Rn$. Set Z-flag if result =0.

CMP, Rn, Rn Compare, Set Z-flag if $Rn = Rn$

Special: Misc.

CP Rn, Rn Copy, $Rn < -Rn$

NOP Waste of time, 1 clk cycle.

MOVC Rn, constant Put a constant in register $Rn = C$.

Flow control: Branch.

BZ, Rn Conditional branch on zero, $PC = Rn$.

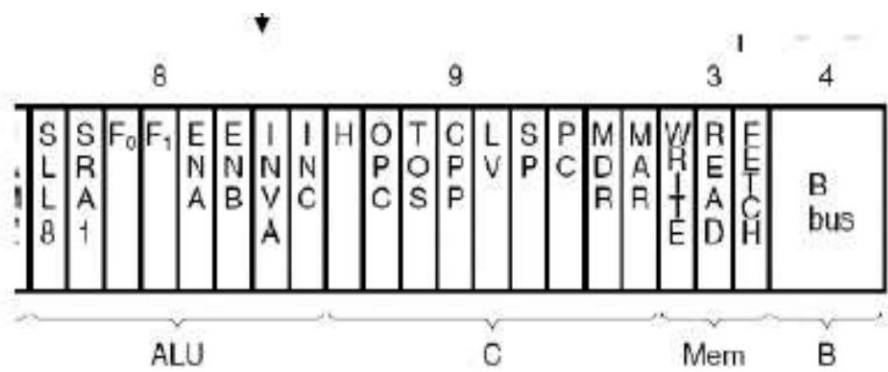
RT Return, return from branch.

Rn: Any user register.

DC: Don't care.

- a. Er dette en generell prosessor? Hvorfor/hvorfor ikke?
- b. Hvilken type instruksjon er NOP og RT?
- c. Hvilken type adresseringsmodi (addressing modes) bruker ALU-instruksjonene?
- d. Hva er den maksimale størrelsen på konstanten som MOVC kan benytte?
- e. Vil følgende psaudokode føre til at Z-flagget blir aktivert slik at betingelsene for et hopp er tilstede? Begrunn svaret.
 - MOVC R1, 0xAA 55;
 - MOVC R2, 0x55 AA;;
 - CP R3, R2;
 - INV R3, R3;
 - CMP R1, R3;
 - BZ R2;

IJVM appendix



B bus registers

0 = MDR	5 = LV
1 = PC	6 = CPP
2 = MBR	7 = TOS
3 = MBRU	8 = OPC
4 = SP	9-15 none

Figur 4: Microinstruction format (IJVM).

F_0	F_1	ENA	ENB	INVA	INC	Function
0	1	1	0	0	0	A
0	1	0	1	0	0	B
0	1	1	0	1	0	$\bar{A}$
1	0	1	1	0	0	$\bar{B}$
1	1	1	1	0	0	A + B
1	1	1	1	0	1	A + B + 1
1	1	1	0	0	1	A + 1
1	1	0	1	0	1	B + 1
1	1	1	1	1	1	B - A
1	1	0	1	1	0	B - 1
1	1	1	0	1	1	-A
0	0	1	1	0	0	A AND B
0	1	1	1	0	0	A OR B
0	1	0	0	0	0	0
1	1	0	0	0	1	1
1	1	0	0	1	0	-1

SLR1 SLL8 Function
 0 0 No shift
 0 1 Shift 8 bit left
 1 0 Shift 1 bit right

Figur 5: ALU functions (IJVM).

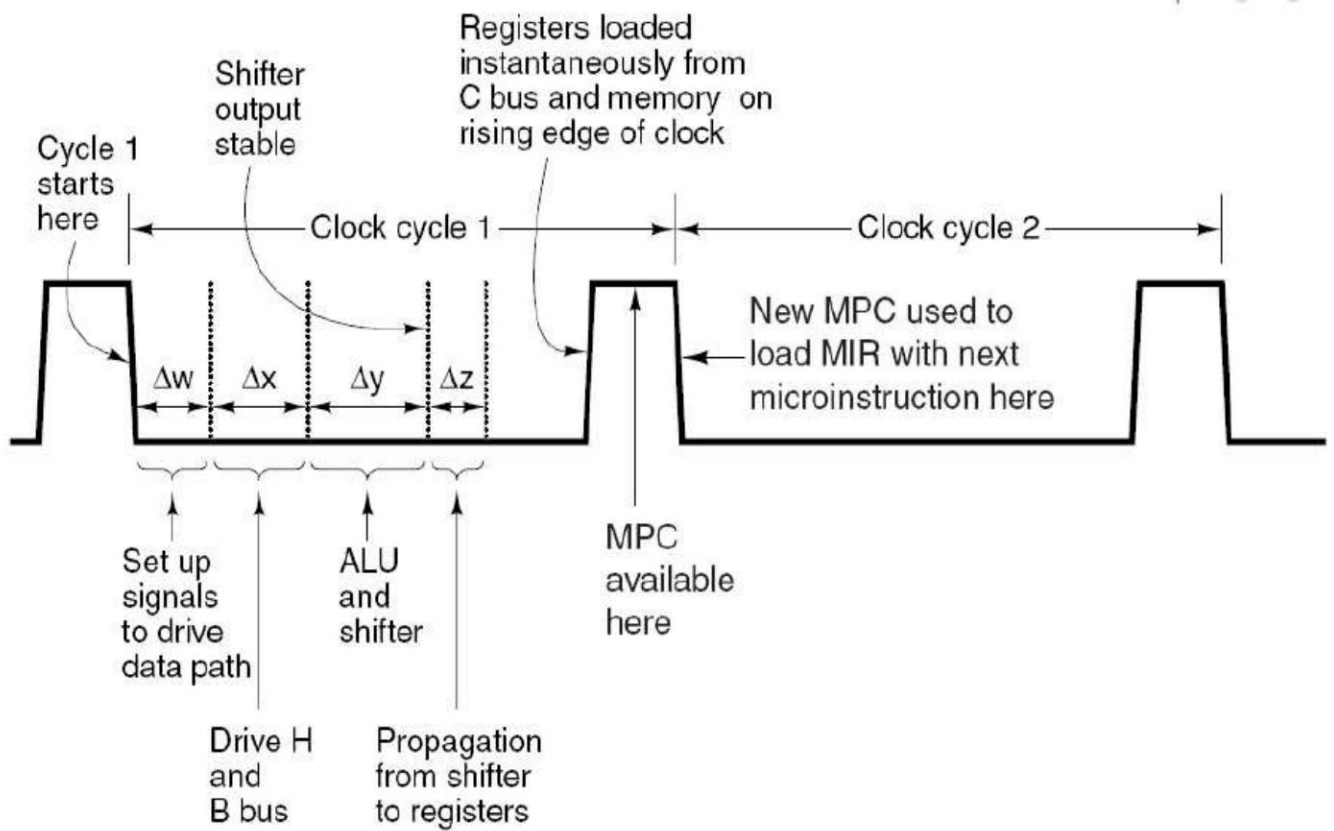


Figure 6: Timing diagram (IJVM).

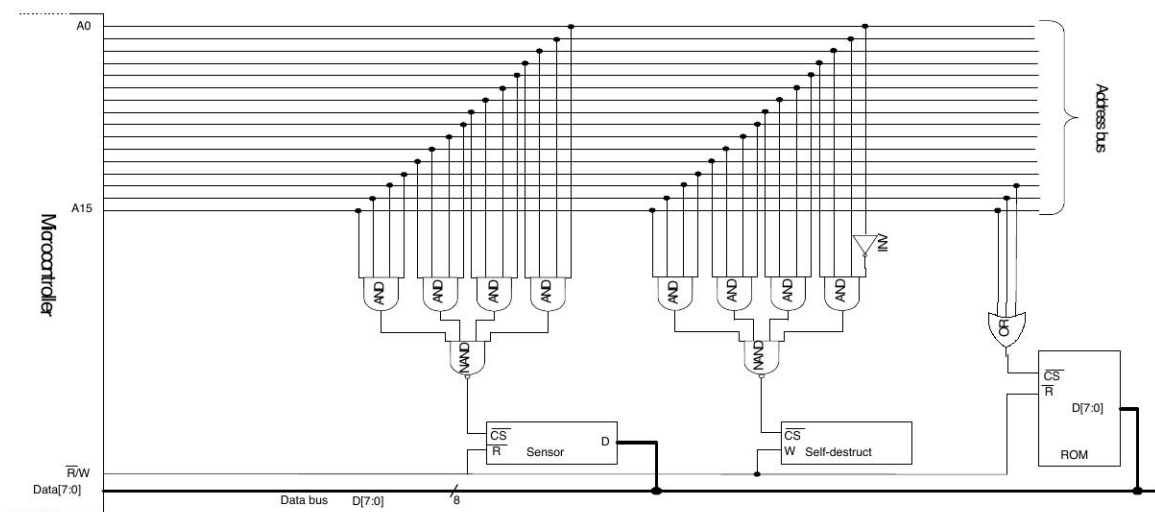


Figure 7: Address decoding.