Answer Key for The Exam

OPPGÅVE 1: DIGITALTLOGISKNIVÅ (25% (10% PÅ A OG B; 5% PÅ C))

a. I figur 1 er EPROM og RAM kopla til ein felles buss. Finn adresseområde for EPROM og RAM og teikn minnekart. Alle einingane har aktivt lågt (logisk "0") CS (Chip Select) signal.

Answer: EPROM2: hex(0000) - hex(3FFF)
EPROM1: hex(4000) - hex(7FFF)

RAM1: hex(8000) - hex (BFFF)
RAM2: hex(C000) - hex (FFFF)

Minnekart, sjå figur 11.

(EPROM2: 00xx xxxx xxxx xxxx, 0000 - 3FFF)
(EPROM1: 01xx xxxx xxxx xxxx, 4000 - 7FFF)
(RAM1: 10xx xxxx xxxx xxxx, 8000 - BFFF)
(RAM2: 11xx xxxx xxxx xxxx, C000 - FFFF)

b. I eit forsøk på optimalisering endrast systemet til å nytte dobbel så store minnebrikker (EPROM og RAM). Det nye systemet er vist i figure 2. Teikn minnekart for det nye systemet.

Answer: Uendra minnekart, men minne er delt på to brikkar i staden for fire, sjå figur 13.

c. Har optimaliserings forsøke nokon innverknad på utføring av program som opphavleg var skrevet (og kompilert) for systemet vist i figur 1? Grunngi svaret.

Answer: Ingen endring. Alle minneadresser i koden er fortsatt gyldige.

OPPGÅVE 2: MIKROARKITEKTUR OG MIKROINSTRUKSJONAR (30% (5% PÅ A OG B; 10% PÅ C OG D))

Bruk vedlagte diagram og tabellar for IJVM til å løyse oppgåvene.

a. Register "PC" har verdien hex(F0F0F00000) og MPC-registeret verdien hex(08). Kva brukast desse registera til? Kva angir verdiane?

   Answer: PC (Program Counter) inneheld minne adressa til instruksjon som skal utførast, peikar på instruksjon (opcode) som ligg på den minnelokasjon som innhaldet peikar på. MPC Micro Program Counter innheld ein opcode som brukast som peikar (adr) til microinstruksjon i control store.

b. Kva er minimum mengd mikroinstruksjonar ein må bruke for å kopiere verdien i H-registeret til alle desse registera: OPC, TOS, CPP, LV, SP, PC, MBR og MAR? Grunngi svaret.

   Answer: Ein mikroinstruksjon. Det er mogleg å setje alle bita i C-felte i mikroinst. til "1" (H-bit er egentlig DC, om H-bit er 1 eller 0 spelar ingen logisk rolle, resultatet er det same). Alle register kan utføre "LOAD" frå C-bussen i paralell.

c. Lag mikroinstruksjon(ar) for følgjande IJVM-operasjon:
   MAR = MDR + PC.

   Sjå vekk frå Addr- og J-felte i mikroinstruksjonsformatet. Angi korrekte bit for ALU, C, Mem og B gitt i Figur 7.

   Answer: 1: Laste MDR inn i H
   ALU: 010100 (B) C: 100000000 (H) Mem: 000 (ingen mem opprasjon) B: 0000 (0 MDR)

   2 addere H + PC skriv til MAR.
   ALU: 111100 (A+B) C: 000000001 (MAR) Mem: 000 (ingen mem opprasjon) B: 0001 (1 PC)

   eller 1: PC -> H
   2: MDR + H

d. 1 Lag mikroinstruksjon(ar) som kan teste om innehalde i OPC $\geq$ MDR.
   2 Kva signal i mikroarkitekturen angir resultatet av testen?

   Sjå vekk frå Addr- og J-felta i mikroinstruksjonsformatet. Angi korrekte bit for ALU, C, Mem og B gitt i Figur 7.

Answer: Her er det fleire løysingar, men alle krever at ein skjønnar IJVM godt (eller er flink og klarar å utnytte vedlegg). Huks at det er lov å oppgi forutsetjingar. Løysingane vist er ikkje dei einaste moglege, men eksempel på korleis det kan gjerast.

Løysing X:
Sidan ein skal sjå vekk frå J feltet er enklaste løysing å forutsetje positive tal. Løysinga vert då:

1)
Laste MDR inn i H
ALU: 010100 (B) C: 100000000 (H) Mem: 000 (ingen mem opprasjon) B: 0000 (0 MDR)

OPC - H
ALU: 111111 (B - A) C: 000000000 (Ingen) Mem: 000 (ingen mem opprasjon) B: 1000 (8 OPC)

2)
N-flagg aktivt viss OPC - MDR er negativt, OPC >= MDR.
Kan også løyse det ved å bytte om OPC og MDR for å få N-flagge til å gi motsatt indikasjon (N = 1 ved MDR >= OPC. Begge vert rekna som korrekt viss ein angir korleis ein tolkar resultatet.

Løysing Y:
Her er ei anna mogleg løysing, meir komplisert, kan også handtere negative verdiar. Her utnyttar eg at ALU brukar toerskomplement for trekke frå (kan vite det sidan sub i funksjonstabellen består av: B inngang og (INVA og INC A), dette betyr også at negative tal er lagra på toerskomplimentsform. Merk at det er mange måtar å få dette til på, dette er eit eksempel på ei mogleg løysing.

1)
Fyrst utnyttar eg at MSD i IJVM er eit fortegnsbit, seier ikkje noko om storleiken. Eg brukar dette bite som eit vanleg bit. endrar då talområde frå stort minus tal til stort plusstal til hex(00000000) - hex(FFFFFFFF) (0 til større plusstal).

Kan då flytte alle tal opp i det konstruerte "positive" talområde for testen ved å addere (7FFFFFFF) med variablane i OPC og MDR.

Kan då bruke samme avslutning som i løysing x.

Leggje hex(7FFFFFFF) i H-reg
Kan då gjere følgjande: Fyrst 0 i H
Microinst 1:
ALU: 010000 (0) C: 100000000 (H) Mem: 000 (ingen mem opprasjon) B: 1111 ()

Microinst 2:
hex(0) i H, ALU: invers A, shift 1 bit right, SLR1 =1 gir H = hex(7FFFFFFF)
ALU: 101100 (Ã) C: 100000000 (H) Mem: 000 (ingen mem opprasjon) B: 1111 () (SLR1: 1, SLL8: 0
Dette kan gjerast på andremåtar også.

Microinst 3:
OPC = OPC + H, justerar verdien i OPC
ALU: 111100 (A+B) C: 010000000 (OPC) Mem: 000 (ingen mem opprasjon) B: 1000 (8 OPC)

Microinst 4:
H = MDR + H, H = MDR + hex(7FFFFFFF), justerar verdien i MDR for test, innhald MDR uendra.
ALU: 111100 (A + B) C: 100000000 (H) Mem: 000 (ingen mem opprasjon) B: 0000 (0 MDR)

Microinst 5:
OPC - H
ALU: 111111 (B - A) C: 000000000 (Ingen) Mem: 000 (ingen mem opprasjon) B: 1000 (8 OPC)

Kan utvidast med å gjennopprette verdien i OPC, må då ta vare på resultatet frå microinst 5 før gjennoppretting av OPC slik at forteikne på resultatet kan testast i siste mikroinstruksjon (N gyldig for testen i siste mikroinstruksjon).

OPPGÅVE 3: ISA-NIVÅ OG AUKE YTING (30% (10% PÅ A OG C; 5% PÅ B OG D)))

I ein tenkt arkitektur har ein definert eit format for instruksjonar som vist i figur 3. Det er definert at det eksisterar 16 brukarregister. Feltet register (figur 3 b) angir kva brukarregister som nyttast.

a. Kva type adressering kan ein utifrå figuren anta at nyttast? Forklar korleis kvart modi nyttast.

   Answer: I denne oppgåva er det diverre ein liten typo, det skal stå: Feltet register (figur 3 c). Det vart opplyst muntleg på eksamen.

   Kan svare:
   Direkte adresering. Dette kan seiast heilt sikkert sidan det er einaste plassen det er definert noko adressefelt. Gir full utteljing
   Ein eller anna form for Register adresering (register indirect, indexed, etc). Dette kan være mulig, men ikkje gitt at det er brukt. Ikkje trekk for å nemne dette.

   Som er heilt rett. Men med lite trekk godtek eg også instruksjonsformat, dette er ikkje adressering men oppbygjing (definisjon) av instruksjonar:

   0-adresse instruksjonar
   1-adress instruksjonar
   register register instruksjonar
   Men dei må forklarast rett.

b. Kva er det maksimale adresserommet for denne arkitekturen? Grunngi svaret.

   Answer: 2^{16}. Det 16 bit store "address" felte er einaste informasjonen me har som angir noko om addresering.

c. Kan ein seie noko om ordbredda (mengd bit maskina opererar på)? Grunngi svaret.

   Answer: Nei. Det er ingen ting i informasjonen som seier noko om ordbredda. Mengd bit i register er ikkje oppgitt.

d. I ein versjon av maskina er det ikkje samleband (pipeline) i instruksjonsutføringa (datapath). Tidbruken for å køyre ein instruksjon for denne maskina er vist i figur 4(a). For å auke ytinga ønskjer ein å innføre samleband. Det vert laga ein versjon med eit samleband med tre steg. Samlebandet er vist i figur 4(b).

   Bruk figur 4(a) og 4(b) til å angi korleis innføring av samlebandet påverkar klokkefrekvens og ILP (Instruction Level Parallelism).

   Answer: Klokka kan setjast til tregaste trinn, og tre instruksjonar i paralell.

OPPGÅVE 4: DIVERSE (15%)

Finn rett svaralternativ for oppgåvene. Korrekte svar gir 3% utteljing, feil svar gir -0.5% og veit ikkje (ikkje svar/fleire svar) gir inga utteljing.

a. Er det mogleg å kopiere innehalde i MAR til PC? Sjå figur 6.

   1) Ja.
   2) Nei.

   Answer: 2

b. Kva påstand er **ikkje** korrekt for ein ein-brikke multiprosessor (CMP) .

   1) Kan ha felles hovudminne.
   2) Kan dele cache.
   3) Er av type homogen eller heterogen.
   4) Kan berre nytte ILP som parallelliserings strategi.

   Answer: 4 (Kan nytte prosessor level Parallelism og så)

c. Figur 5(a) har ein funksjon som gitt i sanningstabellen i figur5(b).

   1) Korrekt.
   2) Ikkje korrekt.

   Answer: 1
   Jadå det er ein fulladder.

d. Ved å auke mengda samlebandstrinn aukar ILP.

1) Korrekt.

2) Ikkje korrekt.

Answer: 1

e. Under følgjer ein rekke påstandar om datamaskinkomponentar. Kva påstand er korrekt?

1) Multipleksa adresse/data buss halverar adresserommet samanlikna med system med dedikerte adresse- og databusar.
2) Ein prosessor som har ein ALU som kan utføre NAND-operasjon, kan programmerast til å utføre alle logiske operasjonar.
3) PROM og flash-minne kan slettast og programmerast fleire gonger.
4) Ein dekodarkrets har fleire inngangar enn utgangar.

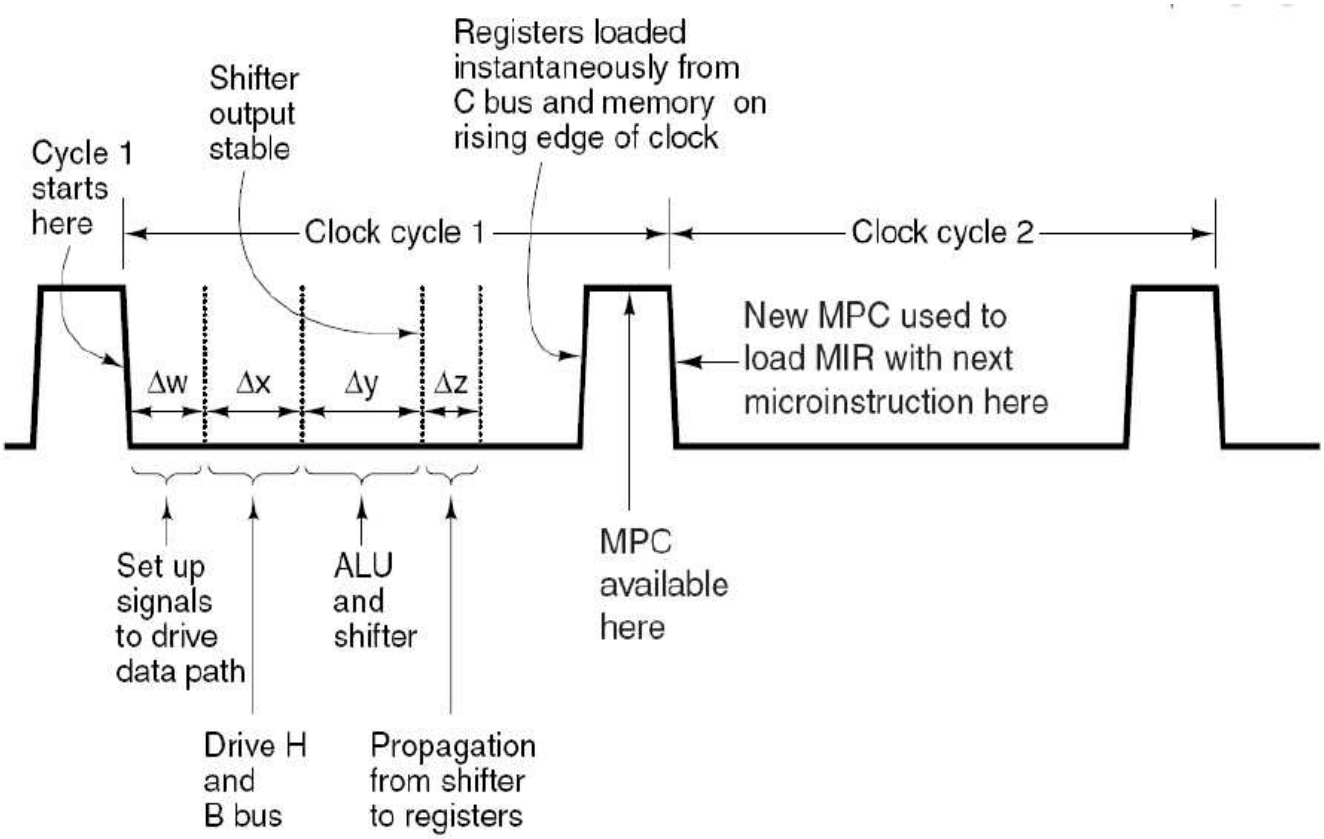
Answer: korrekt? 2

IJVM vedlegg

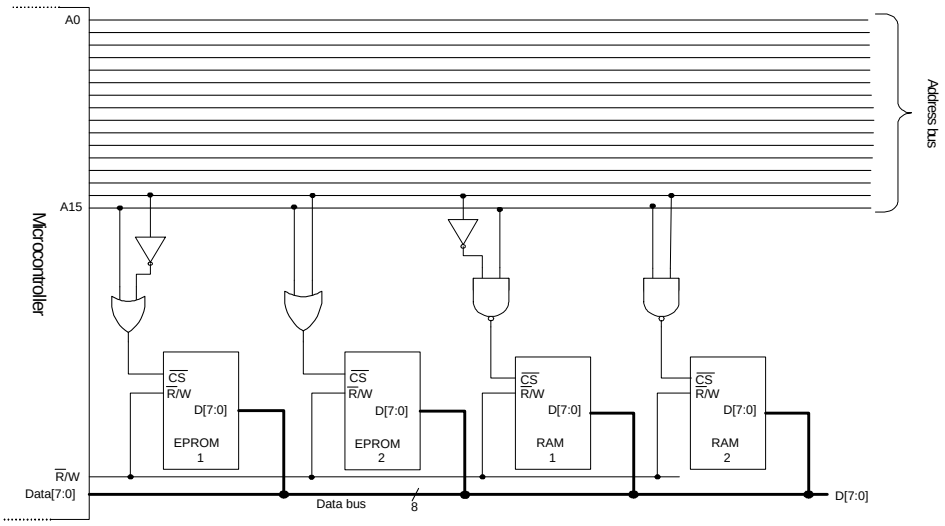
F_0	F_1	ENA	ENB	$\overline{INVA}$	INC	Function
0	1	1	0	0	0	A
0	1	0	1	0	0	B
0	1	1	0	1	0	$\overline{A}$
1	0	1	1	0	0	$\overline{B}$
1	1	1	1	0	0	A + B
1	1	1	1	0	1	A + B + 1
1	1	1	0	0	1	A + 1
1	1	0	1	0	1	B + 1
1	1	1	1	1	1	B – A
1	1	0	1	1	0	B – 1
1	1	1	0	1	1	–A
0	0	1	1	0	0	A AND B
0	1	1	1	0	0	A OR B
0	1	0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

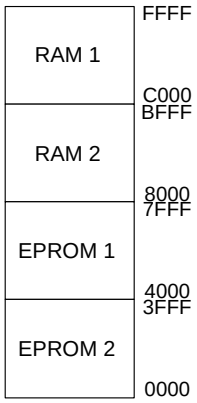
Figur 8: Funksjonstabell for ALU (IJVM).



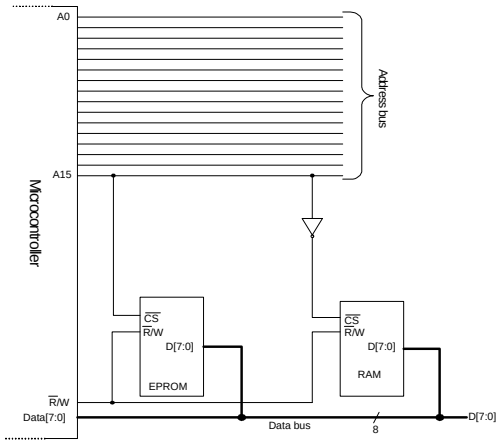
Figur 9: Timingdiagram (IJVM).



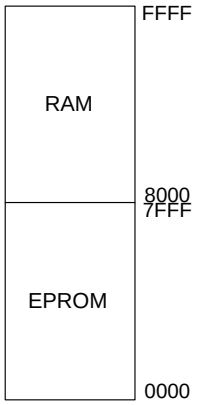
Figur 10: Adressedekoding.



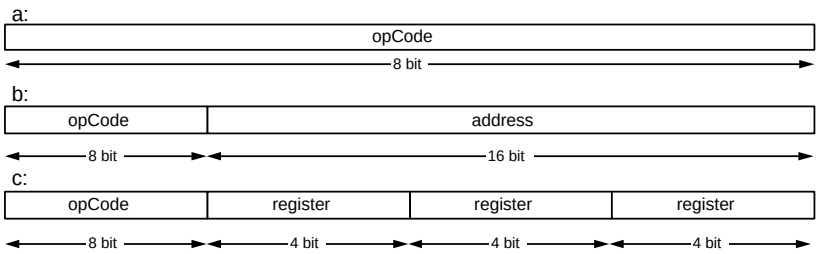
Figur 11: Minnekart.



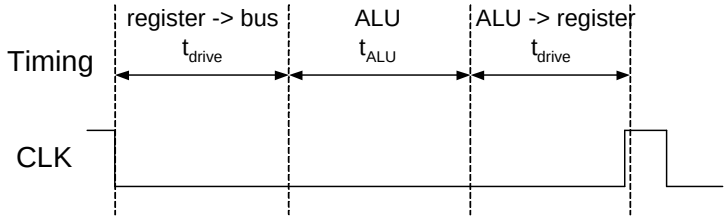
Figur 12: Adressedekoding.



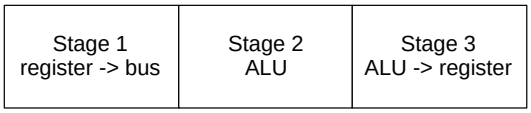
Figur 13: Minnekart.



Figur 14: Instruksjonar.

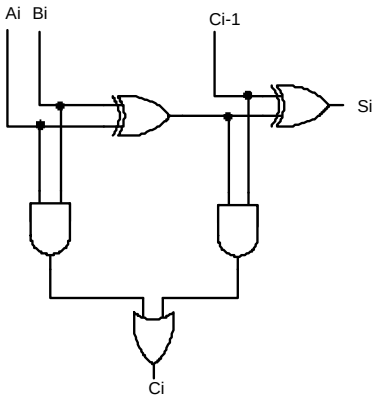


(a) Datapath tidsbruk utan samleband.



(b) Samleband med tre steg.

Figur 15:

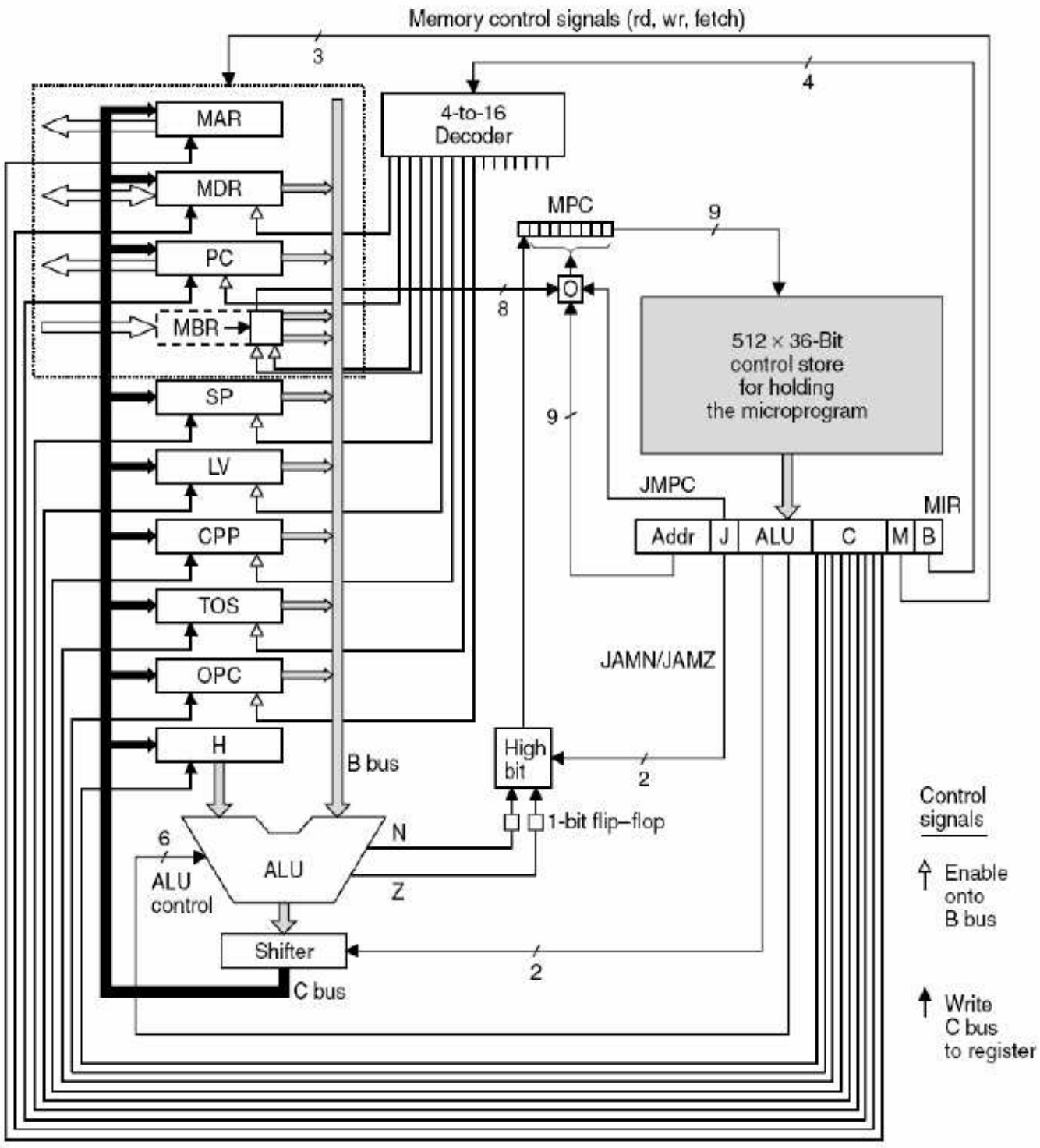


(a) Mystisk dings.

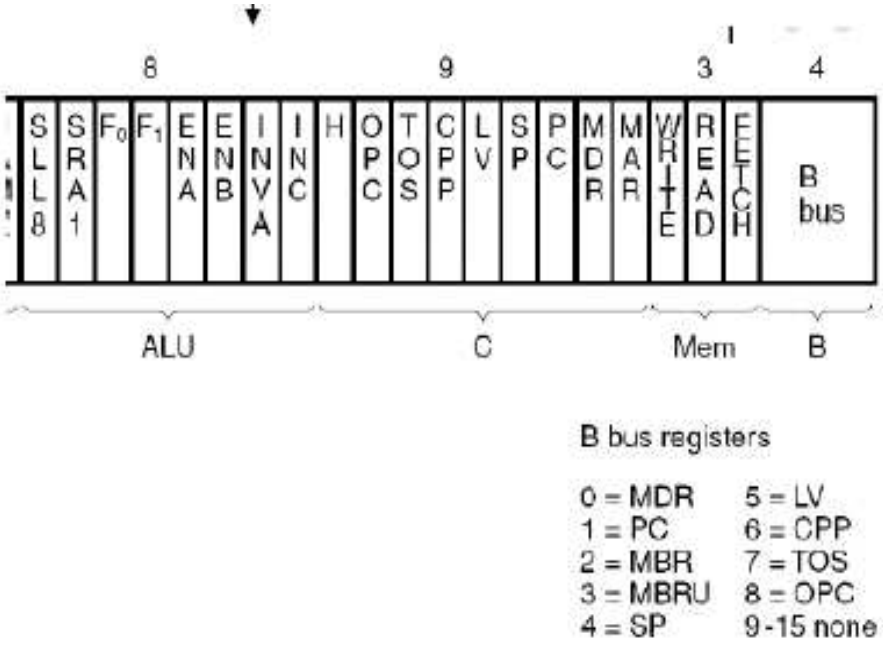
Ai	Bi	Ci-1	Si	Ci
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

(b) Sanningstabell for mystisk dings

Figur 16:



Figur 17: Blokkdiagram (IJVM).

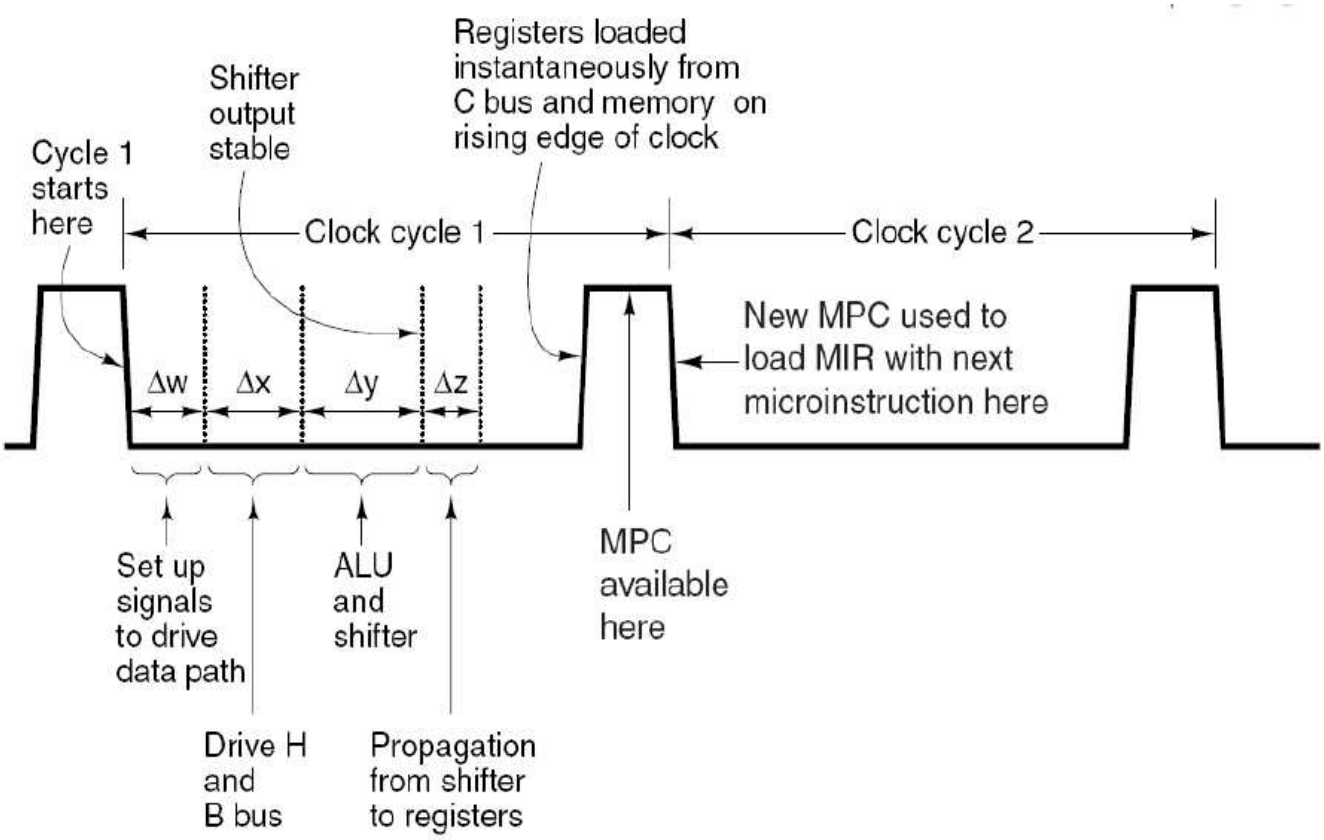


Figur 18: Mikroinstruksjonsformat (IJVM).

F_0	F_1	ENA	ENB	$\overline{INVA}$	INC	Function
0	1	1	0	0	0	A
0	1	0	1	0	0	B
0	1	1	0	1	0	$\overline{A}$
1	0	1	1	0	0	$\overline{B}$
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 19: Funksjonstabell for ALU (IJVM).



Figur 20: Timingdiagram (IJVM).