

# Oppgave 1

$$\text{ARKESSTID} = C + (1-w) \cdot m$$

a) Utgangspunkt:  
 $0,05\mu s + 0,2 \cdot 1 = 0,25\mu s.$

①  $0,05\mu s + 0,05\mu s = 0,1\mu s.$

②  $0,05\mu s + 0,2 \cdot 0,15\mu s = 0,15\mu s.$

Alternativ 7 gir best ytelsesforbedring.

b) Dynamisk minne er ikke like raskt, men det bruker lite avtal sammenlignet med statisk RAM. Bruker også mindre strøm enn SRAM.

c) Moores lov sier at i nyae tid er det slik at ca. hver 18. mån døbles antall transistorer på chipper. Eksponensiell vekst.

d) Nei, de kan være forskjellige.

e)

①

Frekvens:

$$\frac{1}{16\text{ns}} \cdot 1000 = 62,5 \text{ MHz}$$

②

Frekvens:

$$\frac{1}{1\text{ns}} \cdot 100 = 100 \text{ MHz}$$

③

Frekvens:

$$\frac{1}{8\text{ns}} \cdot 1000 = 125 \text{ MHz}$$

Steg ③ har høyest ILP.

## Oppgave 2.

a)

i)

ROM:

000 X X V V X X X X X X X X

[0000, 1FFF]

Ikke relativt stønnes-forhold.



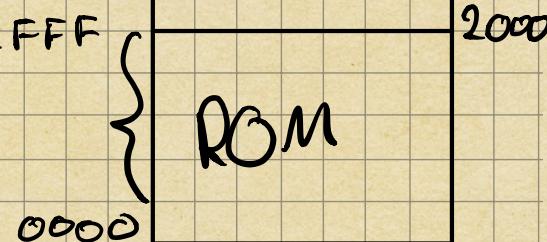
Termostat:

[FFFF]

Display:

1 1 1 1 1 1 1 1 X X X X 1FFF

[FFFO, FFFE]



## RAM:

$\checkmark \times \mid \times$     $\times \vee \times \times$     $\times \times \times 0$     $\times \times \times \times$   
 [2000, FFEEF]

ii)  $2^{\text{16}}$  - 65536 bit

$$\frac{65536 \text{ bit}}{8} = \underline{8192 \text{ byte}}$$

b)

ii) 15 adresser fra FFFO til FFEE

$$(FFEE - FFEO + 1) = \underline{15}$$

iii)

Ingen overlapp

c)

ii) Nei. Det er ikke meir plass i minna. Alle adressi er brukti opp

Nåla diktatoren!

iii)

Ja. Det er ingen Interrupt Request-linje. Oppdatering skjer kun ved polling.

## Oppgave 3

a)

Det niente bit'et i MPC angir hvorvidt man skal sette et hopp eller ikke.

Bit'et gjør at man kan velge mellom to programstier.

b)

1. ALU: B, C: H, MEM: 000, B: TOS

2. ALU: A+B, C: H, MEM: 000, B: LV

3. ALU: A+B, C: TOS, MEM: 000, B: OPC

Ork ikke skrive Bit-pattern. //

c)

1: ALU: 010100 (B), C: 100000000 (H), MEM: 000, B: 0110 (cn)

2: ALU: 001100 (AND), C: 001000000 (TOS), MEM: 000, B: 0100 (SP)

$$TOS = CPP \ AND \ SP$$

$$\begin{array}{r} 0x5555 \\ \times 4AAA \\ \hline 0xAAAA \ SP \end{array} \quad LV$$

$$N=0, \ Z=1$$

$$= \underline{\underline{0x0000}}$$

## Oppgave 4

a) Dette er en RISC-prosessor fordi det like lange instruksjoner, og de er meget enkle. Det er oppgitt at det er mange generelle register. I tillegg er det load/store-arkitektur.

- b)
- i) En-adress-instruksjon. Bruker kun ~~adressepunktet~~ til den den skal lagre verdien.
  - ii) MOVC bruker immediate addressing konstant/data gitt som operand.
  - iii) Register addressing.

c) Ja. Vi har 8 bit for å lage adresser.

16 adresser krever kom 4 bit ( $2^4 = 16$ ),  
så vi kan få potensiert  $2^8 = 256$  adresser.

d) Ja. Har branching, YO og ALU-funksjoner.

d) STORE R<sub>1</sub>, R<sub>3</sub>

$$[0xAAAA\ AAAAA] \approx R_1 = 0x0000\ AAAA$$

STORE R<sub>2</sub>, R<sub>4</sub>

$$[0x5555\ 5555] = R_2 = 0x0000\ 5555$$

INV R<sub>3</sub>, R<sub>3</sub>

$$R_3 = 0x5555\ 5555$$

INV R<sub>4</sub>, R<sub>4</sub>

$$R_4 = 0x\ AAAAA\ AAAAA$$

LOAD R<sub>1</sub>, R<sub>3</sub>

$$R_1 = [R_3] = 0x0000\ 5555 \quad \text{F}$$

LOAD R<sub>2</sub>, R<sub>4</sub>

$$R_2 = [R_4] = 0x0000\ AAAAA \quad \text{F}$$

i) STORE R<sub>1</sub>, R<sub>3</sub>

$$[0x\ AAAAA\ AAAAA] = 0x0000\ AAAAA$$

STORE R<sub>2</sub>, R<sub>1</sub>

$$[0x5555 \ 5555] = 0x0000 \ 5555$$

CP R<sub>1</sub>, R<sub>2</sub>

$$R_1 = 0x0000 \ 5555$$

CP R<sub>2</sub>, R<sub>1</sub>

$$R_2 = 0x0000 \ 5555$$

Det samme, idt  
liket som resultat  
over