LMC - 3 Eksamensoppgave

```
start   INP             00 INP
        BRZ slutt       01 BRZ 13
        STA in          02 STA 27
        LDA laster      03 LDA 28
        ADD in          04 ADD 27
        STA loop        05 STA 06
loop    HLT             06 HLT
        BRZ start       07 BRZ 00
        OTC             08 OTC
        LDA loop        09 LDA 06
        ADD en          10 ADD 26
        STA loop        11 STA 06
        BRA loop        12 BRA 06
slutt   HLT             13 HLT
lmc     DAT 76 // L     14 DAT 76
        DAT 77 // M     15 DAT 77
        DAT 67 // C     16 DAT 67
        DAT 0 // NULL    17 DAT 00
jul     DAT 74 // J     18 DAT 74
        DAT 117 // u     19 DAT 117
        DAT 108 // l     20 DAT 108
        DAT 0 // NULL    21 DAT 00
ifi     DAT 73 // I     22 DAT 73
        DAT 70 // F     23 DAT 70
        DAT 73 // I     24 DAT 73
        DAT 0 // NULL    25 DAT 00
en      DAT 1           26 DAT 01
in      DAT             27 DAT 00
laster  LDA 0           28 LDA 00
```

Koden over skriver ut én og én karakter fra ønsket adresse inntil en "null" leses fra minnet. Etter å ha tastet inn "14", taster brukeren inn "18" og så "0".

1) **Hvilken tekst skrives ut? [Velg alternativ] ("IFI" og ikke noe mer.,"LMC","Jul", og ikke noe mer., Programmet kræsjer.,"IFI","Jul", og ikke noe mer.)**
2) **Hvilken maskinkode finner vi på etikett loop etter at den sjette instruksjonen har blitt eksekvert?[Velg alternativ] (500, 914, 518, 906, 300, 514)**

Del 1:Hva prøver programmet å gjøre?

Analyserer først hva programmet gjør, sånn som vi har gjort i gruppetimen. Minnet starter egentlig på 0, men siden jeg skriver dette i google docs blir det 1:)

1. start INP //les et tall fra brukeren
2. BRZ slutt //hvis tallet er 0 så sendes vi videre til slutt som avslutter programmet
3. STA in //lagre tallet i variabelen "in"
4. LDA laster //laster verdien 500 (maskinkoden for "LDA 0")
5. ADD in //ACC = 500 + in → 500 + (brukerens tall) 14, 18 eller 0
6. STA loop //skriv denne maskinkoden inn på adressen "loop"
7. loop HLT //overskrevet
8. BRZ start // hvis ACC = 0 → Hopp til start og be om et nytt tall
9. OTC //skriver ut tegnet i ACC som et ASCII-tegn i output
10. LDA loop //last inn instruksjonen som ligger på adressen loop
11. ADD en // ACCinstruksjonen + 1 som øker operand-delen med 1, som vil si at den peker på neste tegn i teksten
12. STA loop //lagrer den oppdaterte instruksjonen tilbake på adressen 'loop'
13. BRA loop // hopper tilbake til loop og fortsetter å skrive ut neste tegn
14. slutt HLT //avslutter programmet
15. lmc DAT 76 // 'L' - ASCII verdi
16. DAT 77 // 'M' - ASCII verdi
17. DAT 67 // 'C' -ASCII verdi
18. DAT 0 // NULL
19. jul DAT 74 // 'J' - ASCII verdi
20. DAT 117 //'u' - ASCII verdi
21. DAT 108 // 'l' - ASCII verdi
22. DAT 0 // NULL
23. ifi DAT 73 // 'I' - ASCII verdi
24. DAT 70 // 'F' - ASCII verdi
25. DAT 73 // 'I' - ASCII verdi
26. DAT 0 // NULL
27. en DAT 1 // 1

28. in   DAT 0 // 0
29. laster LDA 0   // maskinkode 500

Hvorfor stopper ikke programmet på HLT ved loop?

Siden dette ble et spørsmål i timen: Opprinnelig ligger HLT på minnet 06, men rett før vi hopper inn i løkken gjør vi følgende,

1. LDA laster   // ACC = 500 (LDA 0)
2. ADD in   // ACC = 500 + (brukerinput 14, 18 eller 0)
3. STA loop   // skriver ACC til adresse 06

Hva betyr det? Vel adresse 06 “loop” overskrives med en ny instruksjon LDA <input>.

HLT-en vi ser i koden er altså bare en startverdi, den blir aldri brukt. Derfor stopper ikke programmet på HLT i loop, den HLT-en er byttet ut med en LDA-instruksjon før løkka kjører.

Kjøring av programmet

Brukeren skriver 14.

Merk: tallene etter instruksjonskoden er adressen i minnet, hvis det er forvirrende:) Og ACC er arbeidsregisteret prosessoren har.

1. INP → ACC = 14
1. BRZ 13 → 14 ≠ 0, ingen hopp
2. STA 27 → in = 14
3. LDA 28 → ACC = innholdet på adresse 28 = LDA 00 = 500
4. ADD 27 → ACC = 500 + 14 = 514
5. STA 06 → minne[06] = 514 = LDA 14

Adressen loop (06 i minnet) inneholder nå LDA 14:)

Så hva skjer nå? Vel LMC går inn i løkken

1. LDA 14 → ACC = minne[14] = 76 = 'L' 

2. BRZ 00 → 76 ≠ 0, ingen hopp
3. OTC → skriver ut tegnet med verdi 76 → 'L'
4. LDA 06 → ACC = innholdet på adresse 06 = 514 (LDA 14)

5. ADD 26 → minne[26] = 1 → ACC = 514 + 1 = 515 (LDA 15)
6. STA 06 → lagre 515 på adresse 06 → nå står det LDA 15 på loop
7. BRA 06 → hopp til adresse 06 og gjenta

Neste runde:

- LDA 15 → leser 77 = 'M' → skriver 'M'
- øker igjen til LDA 16
- så LDA 16 → 67 = 'C' → skriver 'C'
- øker igjen til LDA 17
- LDA 17 → 0 (NULL)
- nå vil BRZ 00 hoppe tilbake til start

Første kjøring skriver altså: "LMC"

Brukeren skriver 18.

1. INP → ACC = 18
2. BRZ 13 → 18 ≠ 0
3. STA 27 → in = 18
4. LDA 28 → ACC = 500
5. ADD 27 → ACC = 500 + 18 = 518 = LDA 18
6. STA 06 → minne[06] = 518 = LDA 18

Løkka gjør nå samme prosedyre som i første kjøring, men starter på adresse 18:

- 18 → 74 → 'J'
- 19 →117 → 'u'
- 20 →108 → 'l'
- 21 → 0 → BRZ til start

Andre kjøring skriver: "Jul"

Brukeren skriver 0

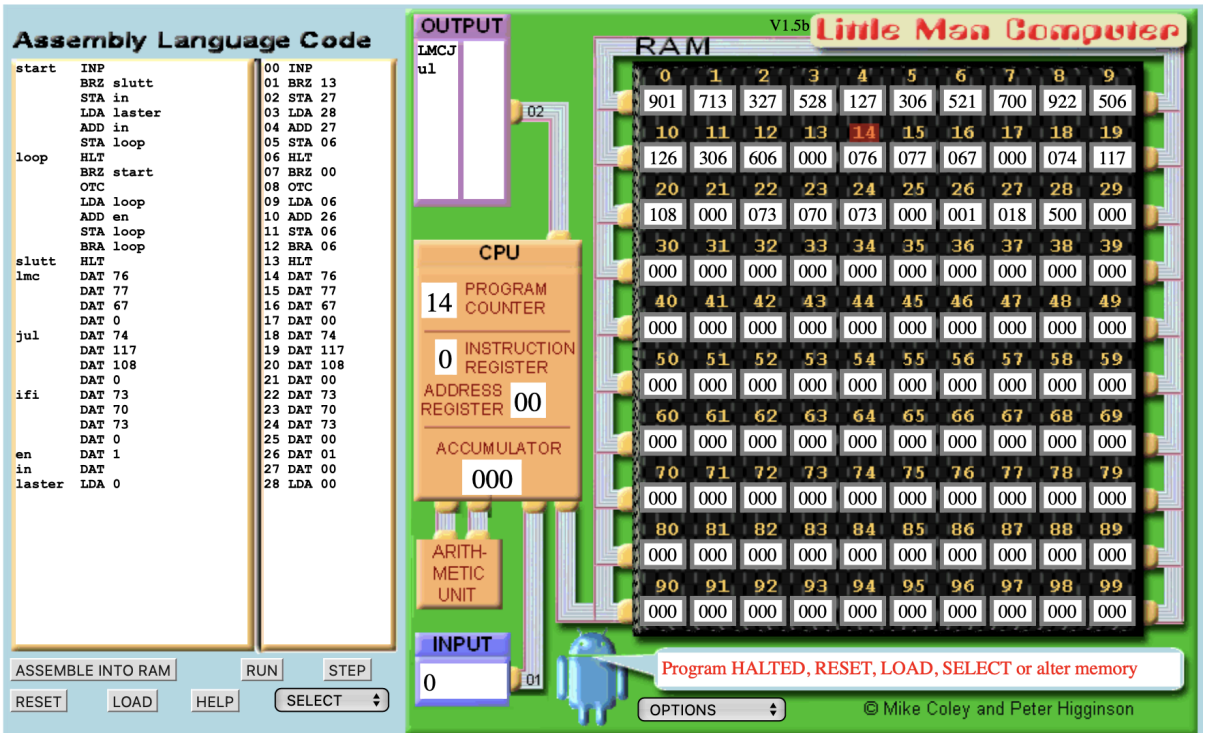
1. INP → ACC = 0

2. BRZ slutt (BRZ 13) → ACC = 0 → hopp til slutt (adresse 13)
3. slutt: HLT → programmet stopper.

Programmet skriver ikke ut noe.

Svar på oppgaven: Så svaret på oppgave 1 vil da være "LMC", "Jul", og ikke noe mer.

Prøv å kjør dette i LMC og se hva som skjer for hvert steg. Sett heller LMC-en i saktemodus så du ser hvordan programmet endrer verdiene i minnet basert på instruksjonen og daten:)



DEL 2: Hvilken maskinkode står på loop etter sjette instruksjon?

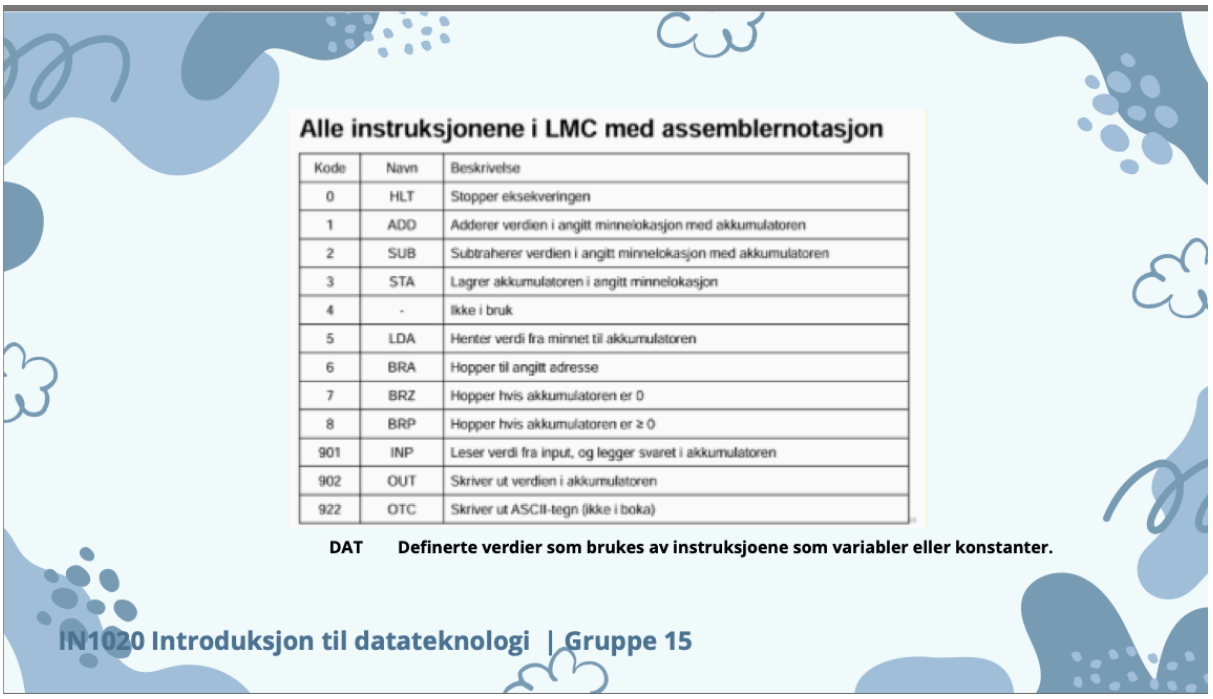
Vi ser på bare den første kjøringen, med input 14 i del 1, og følger de seks første instruksjonene én gang til, med ACC:

1. INP → ACC = 14
2. BRZ 13 → ingen hopp
3. STA 27 → minne[27] = 14
4. LDA 28 → ACC = minne[28] = LDA 00 = 500
5. ADD 27 → ACC = 500 + 14 = 514

6. STA 06 → minne[06] = 514

Instruksjon nummer 6 (den sjette som har blitt utført) er altså STA 06, og den lagrer verdien 514 på adressen loop. 514 i LMC betyr: instruksjonskode 5 → LDA, data 14 → adresse 14 → LDA 14.

Se gruppetime sliden min fra "Uke 8 - LMC" for instruksjonskodene (la inn et bilde under også), eller hvis du ikke helt har forstått hvordan minnet opererer (minneinnhold: første, andre og tredje siffer).



Svar: Maskinkoden på variablen 'loop' etter at den sjette instruksjonen er utført, er: 514