PROSESSER OG LMC

Av Elise

PLAN

- LMC - Hjelper oss å forså hvordan ekte CPU-er fungerer (miniatyr)
- CPU - hjernen til datamaskin
- Prøve selv og live koding (forstå basis)

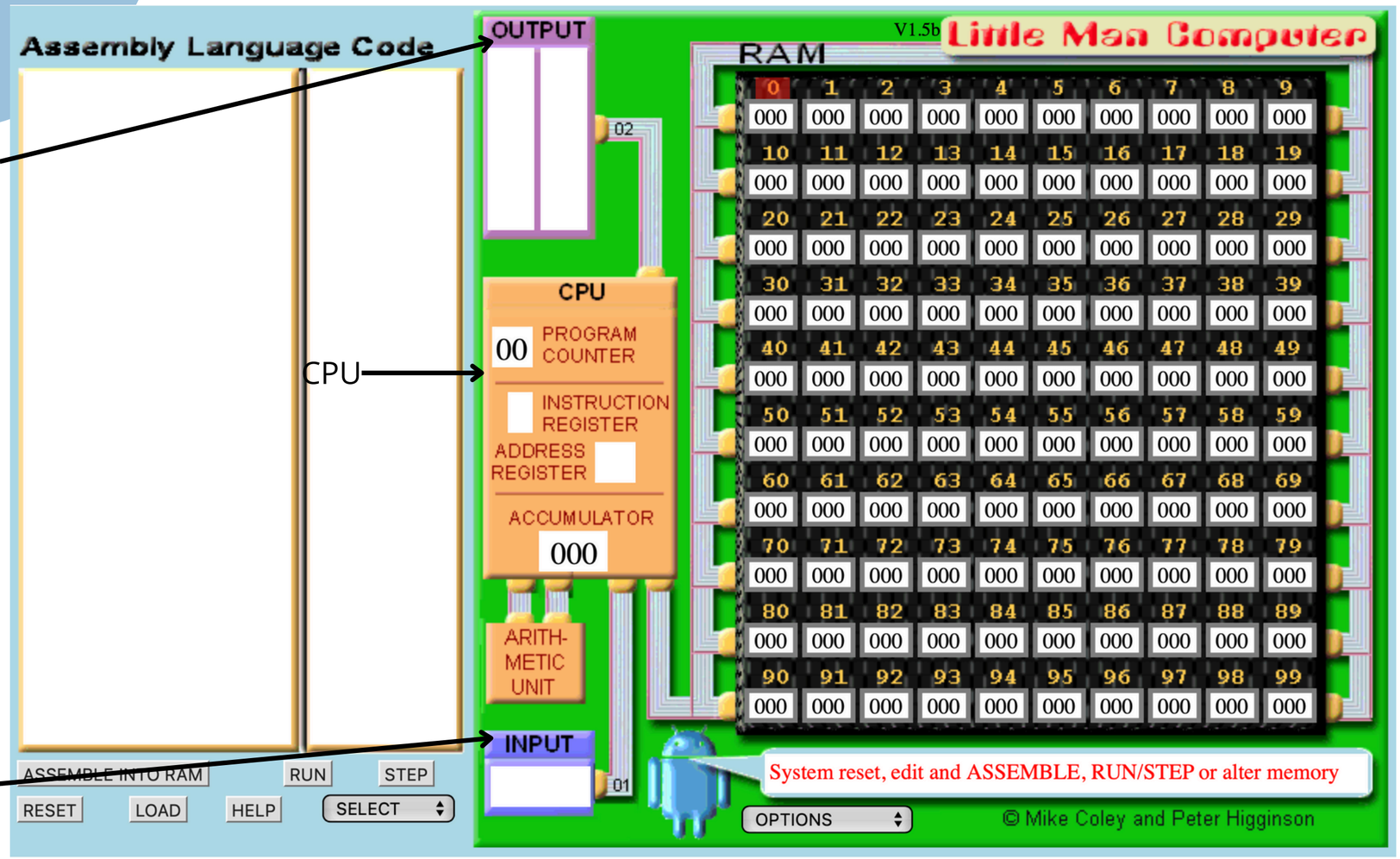
PRAKTISK INFO

- Ordinær oblig-løp: frist 24/10
  - Kan leveres i grupper! Registrer gruppen i devilry
  - Kan ikke levere blankt denne gangen!
  - Kriterier: Gjør en ordentlig innsats, men trenger ikke å få alt riktig!
  - Prøv å ikke bruk KI❤️
- IN1020 universet: Håper alle har kommet godt i gang! Programmeringsoppgaven ligger ute på satellitten:)
  - Frist for å åpne sattelittoppgaven: 21.10
  - Frist for å fullføre satelittoppgaven: 27.10

LMC – LITTLE MAN COMPUTER

- En meget enkel, men teoretisk korrekt, datamaskin
- Brukes til å forstå hvordan ekte prosessorer fungerer
- LMC kan bare håndtere et lite utvalg av operasjoner / oppgaver
  - Input / output
  - Lagre og hente data
  - enkle beregninger

Konsept: “En liten postmann som henter intruksjoner fra minnet, utfører dem en etter en, og leverer resultater tilbake.”



Output kommer her

CPU

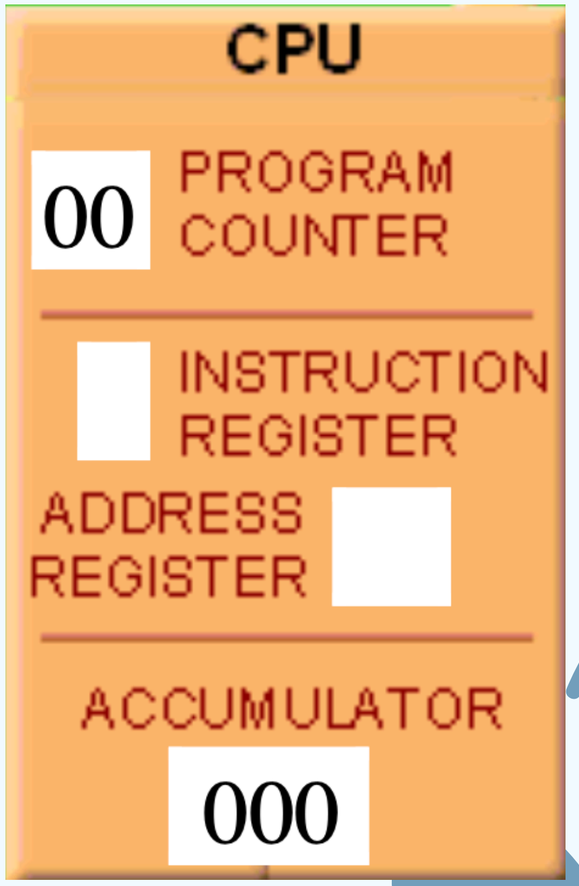
Input in her

Minnet - Von Neumann arkitektur

Inneholder både program og data
0 - 99 plasser (100 posthyller)

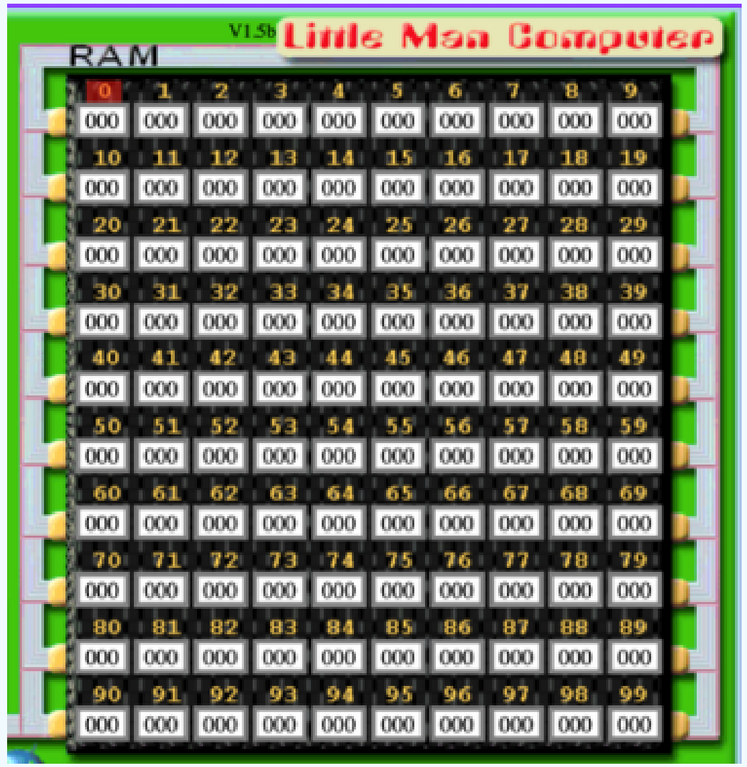
CPU – CENTRAL PROSESSING UNIT

- **Hjernen** i datamaskinen, den som utfører instruksjonene i et program
- ***Fire operasjoner:***
  - **Program Counter:** forteller den lille mannen hvilke postkasse hans neste instruksjon er i. Øker for hver instruksjon utført
  - **Instruction register:** Inneholder den gjeldene instruksjonen som utføres.
  - **Adress Register:** Lagrer adressen til data som brukes eller som instruksjonen skal behandles
  - **Accumulator:** Holder verdien midlertidig under programkjøring, simpel kalkulator som kan håndtere 3 sifrer



MINNET

- Tenk på det som posthyller
- Vi henter og lagrer data i minnet
- Svært lite og enkelt. Har bare 100 minneplasser (adresser 0-99)
- **Minnelokasjon:** adressen minneinnhold er tildelt
- **Minneinnhold:** beregnet til maks 3 siffer (ikke negative)
  - Første siffer: instruksjonskoden, sier hvilken operasjon LMC må gjøre
  - Andre og tredje siffer: Adressen i minnet, hvor LMC skal operere (minnelokasjon)
  - Eksempel → Neste side



MINNET

Minnelokasjon 00:

- Minneinnhold 542 = LDA 42
- Henter verdien som ligger i adresse 42

Minnelokasjon 01:

- Minneinnhold 199 = ADD 99
- Legger til verdien i adresse 99

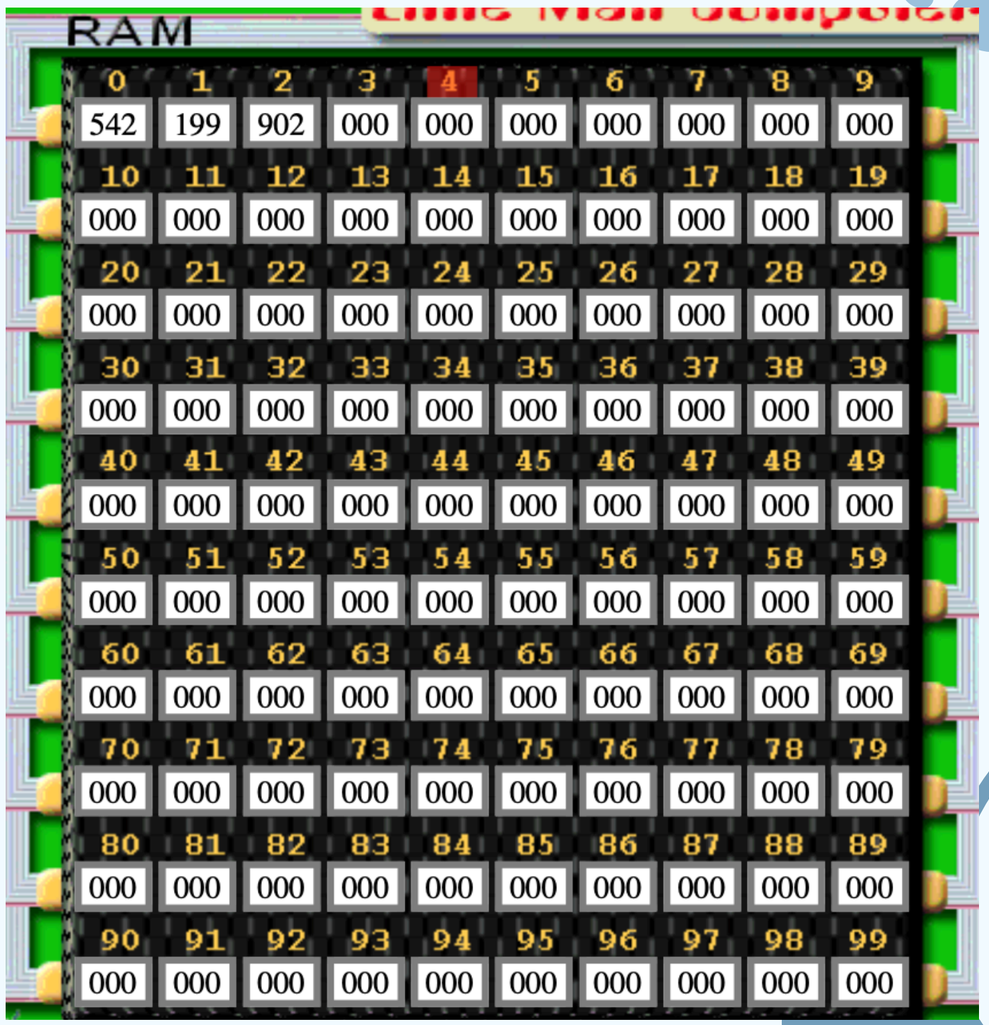
Minnelokasjon 02:

- Minneinnhold 902 = OUT
- Skriver ut resultatet

Minnelokasjon 03:

- Minneinnhold 000 = HLT
- Stopper programmet

```
LDA  42
ADD  99
OUT
HLT
```



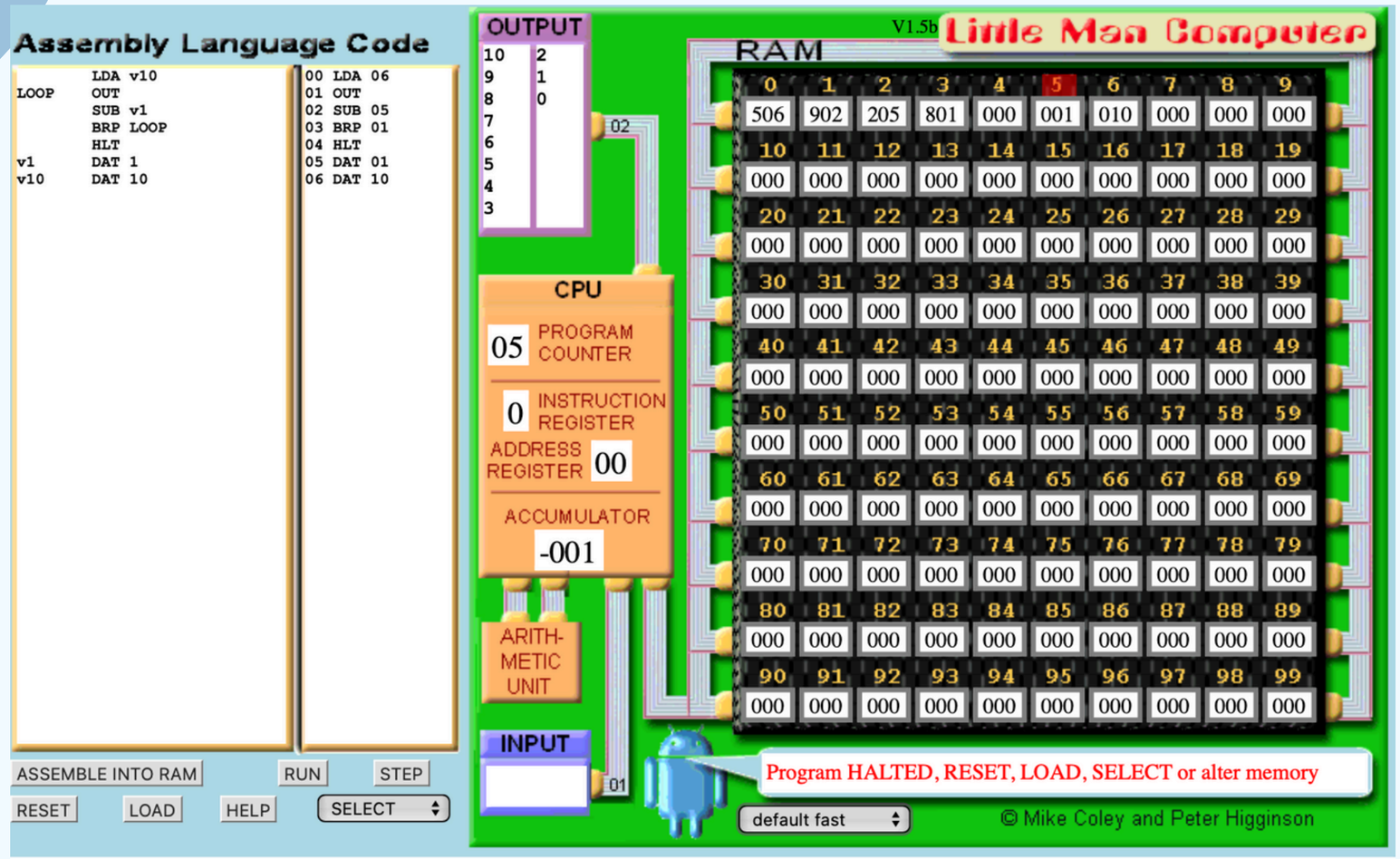
Alle instruksjonene i LMC med assemblernotasjon

Kode	Navn	Beskrivelse
0	HLT	Stopper eksekveringen
1	ADD	Adderer verdien i angitt minnelokasjon med akkumulatoren
2	SUB	Subtraherer verdien i angitt minnelokasjon med akkumulatoren
3	STA	Lagrer akkumulatoren i angitt minnelokasjon
4	-	Ikke i bruk
5	LDA	Henter verdi fra minnet til akkumulatoren
6	BRA	Hopper til angitt adresse
7	BRZ	Hopper hvis akkumulatoren er 0
8	BRP	Hopper hvis akkumulatoren er ≥ 0
901	INP	Leser verdi fra input, og legger svaret i akkumulatoren
902	OUT	Skriver ut verdien i akkumulatoren
922	OTC	Skriver ut ASCII-tegn (ikke i boka)

DAT **Definerte verdier som brukes av instruksjoene som variabler eller konstanter.**

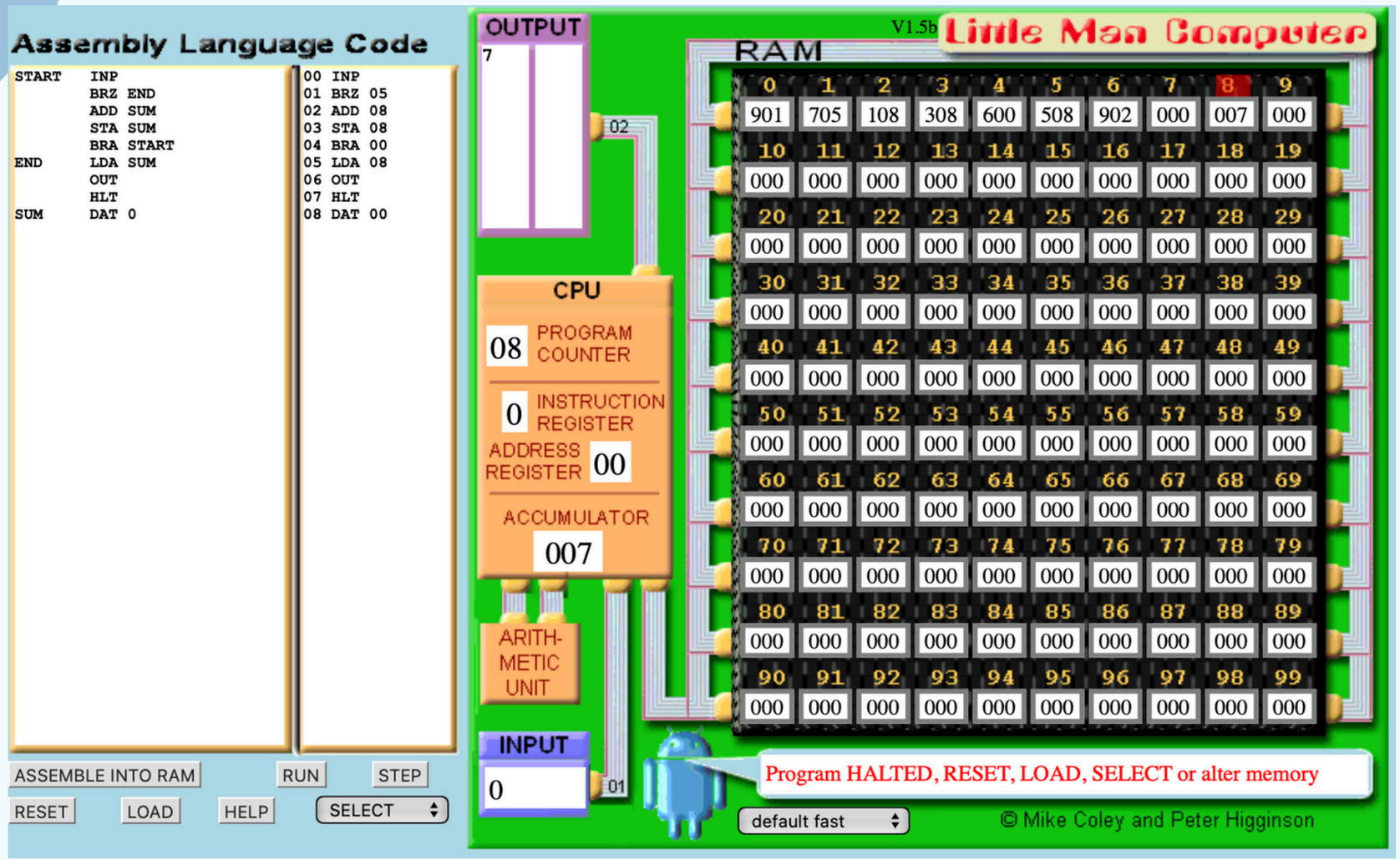
Oppgave 1

Lage et program i Little Man Computer som teller ned fra 10 til 1 og skriver ut hvert tall.


Assembly Language Code
LDA v10
LOOP OUT
SUB v1
BRP LOOP
HLT
v1 DAT 1
v10 DAT 10
00 LDA 06
01 OUT
02 SUB 05
03 BRP 01
04 HLT
05 DAT 01
06 DAT 10
ASSEMBLE INTO RAM
RUN
STEP
RESET
LOAD
HELP
SELECT
OUTPUT
10 2
9 1
8 0
7
6
5
4
3
02
CPU
05 PROGRAM COUNTER
0 INSTRUCTION REGISTER
ADDRESS REGISTER 00
ACCUMULATOR
-001
ARITH-METIC UNIT
INPUT
01
V1.5b
Little Man Computer
RAM
0 1 2 3 4 5 6 7 8 9
506 902 205 801 000 001 010 000 000 000
10 11 12 13 14 15 16 17 18 19
000 000 000 000 000 000 000 000 000 000
20 21 22 23 24 25 26 27 28 29
000 000 000 000 000 000 000 000 000 000
30 31 32 33 34 35 36 37 38 39
000 000 000 000 000 000 000 000 000 000
40 41 42 43 44 45 46 47 48 49
000 000 000 000 000 000 000 000 000 000
50 51 52 53 54 55 56 57 58 59
000 000 000 000 000 000 000 000 000 000
60 61 62 63 64 65 66 67 68 69
000 000 000 000 000 000 000 000 000 000
70 71 72 73 74 75 76 77 78 79
000 000 000 000 000 000 000 000 000 000
80 81 82 83 84 85 86 87 88 89
000 000 000 000 000 000 000 000 000 000
90 91 92 93 94 95 96 97 98 99
000 000 000 000 000 000 000 000 000 000
Program HALTED, RESET, LOAD, SELECT or alter memory
default fast
© Mike Coley and Peter Higginson

Oppgave 2

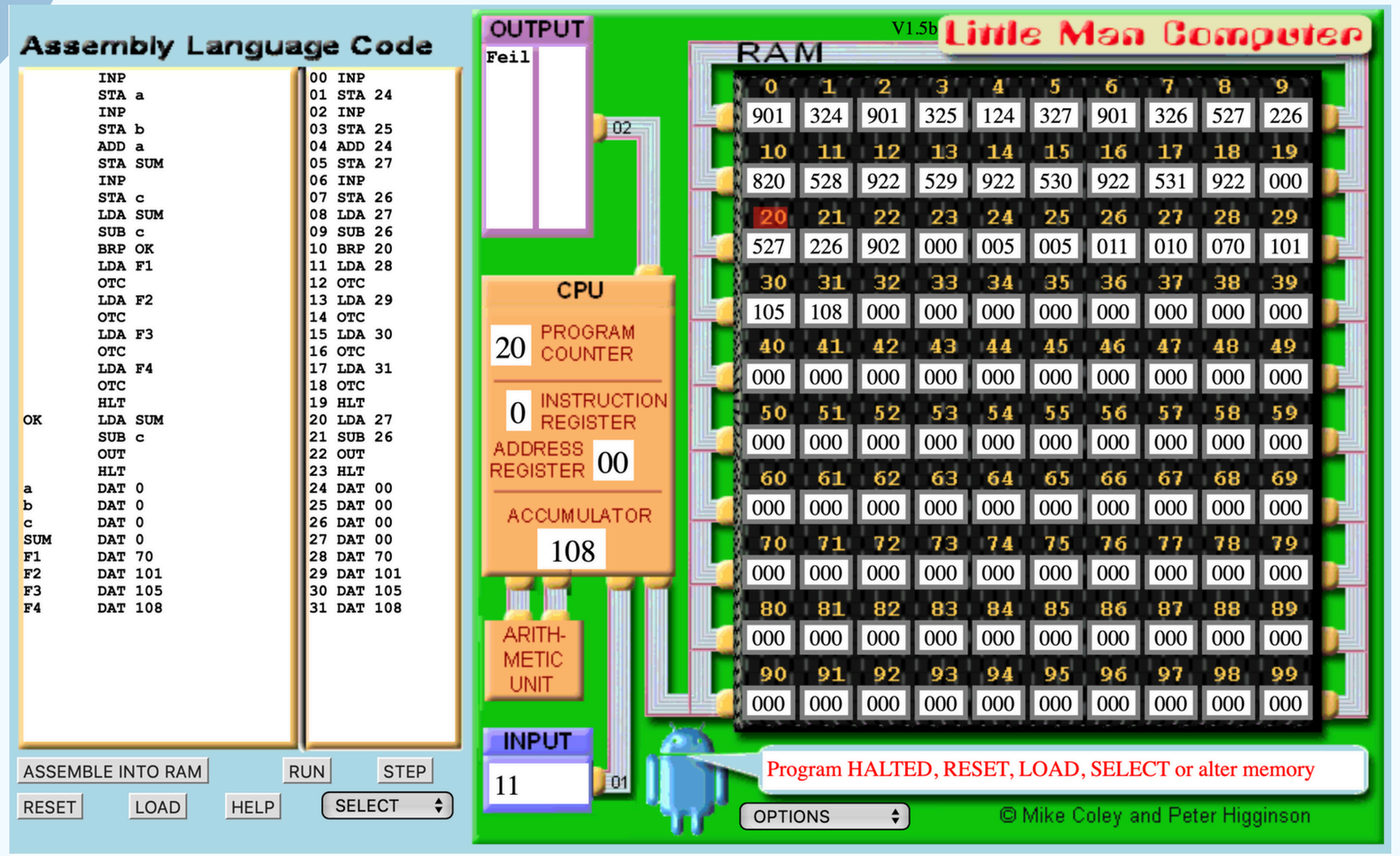
Lag et program i Little Man Computer som leser inn tall fra brukeren, summerer dem, og skriver ut totalen når brukeren taster inn 0.


Assembly Language Code
START INP
BRZ END
ADD SUM
STA SUM
BRA START
END LDA SUM
OUT
HLT
SUM DAT 0
00 INP
01 BRZ 05
02 ADD 08
03 STA 08
04 BRA 00
05 LDA 08
06 OUT
07 HLT
08 DAT 00
ASSEMBLE INTO RAM
RUN
STEP
RESET
LOAD
HELP
SELECT
OUTPUT
7
02
CPU
08 PROGRAM COUNTER
0 INSTRUCTION REGISTER
ADDRESS REGISTER 00
ACCUMULATOR
007
ARITH-METIC UNIT
INPUT
0
01
V1.5b
Little Man Computer
RAM
0 1 2 3 4 5 6 7 8 9
901 705 108 308 600 508 902 000 007 000
10 11 12 13 14 15 16 17 18 19
000 000 000 000 000 000 000 000 000 000
20 21 22 23 24 25 26 27 28 29
000 000 000 000 000 000 000 000 000 000
30 31 32 33 34 35 36 37 38 39
000 000 000 000 000 000 000 000 000 000
40 41 42 43 44 45 46 47 48 49
000 000 000 000 000 000 000 000 000 000
50 51 52 53 54 55 56 57 58 59
000 000 000 000 000 000 000 000 000 000
60 61 62 63 64 65 66 67 68 69
000 000 000 000 000 000 000 000 000 000
70 71 72 73 74 75 76 77 78 79
000 000 000 000 000 000 000 000 000 000
80 81 82 83 84 85 86 87 88 89
000 000 000 000 000 000 000 000 000 000
90 91 92 93 94 95 96 97 98 99
000 000 000 000 000 000 000 000 000 000
Program HALTED, RESET, LOAD, SELECT or alter memory
default fast
© Mike Coley and Peter Higginson

Oppgave 3

Lag et program som:

- Tar to tall (a og b) leser inn og summeres, lagres som SUM.
- Et tredje tall (c) leses inn og trekkes fra SUM.
- Hvis resultatet blir negativt, skriv ut en feilmelding (f.eks. “Feil”) ved hjelp av ASCII (OTC).
- Ellers skriv ut resultatet som et tall (OUT).


Assembly Language Code
INP
STA a
INP
STA b
ADD a
STA SUM
INP
STA c
LDA SUM
SUB c
BRP OK
LDA F1
OTC
LDA F2
OTC
LDA F3
OTC
LDA F4
OTC
HLT
OK LDA SUM
SUB c
OUT
HLT
a DAT 0
b DAT 0
c DAT 0
SUM DAT 0
F1 DAT 70
F2 DAT 101
F3 DAT 105
F4 DAT 108
00 INP
01 STA 24
02 INP
03 STA 25
04 ADD 24
05 STA 27
06 INP
07 STA 26
08 LDA 27
09 SUB 26
10 BRP 20
11 LDA 28
12 OTC
13 LDA 29
14 OTC
15 LDA 30
16 OTC
17 LDA 31
18 OTC
19 HLT
20 LDA 27
21 SUB 26
22 OUT
23 HLT
24 DAT 00
25 DAT 00
26 DAT 00
27 DAT 00
28 DAT 70
29 DAT 101
30 DAT 105
31 DAT 108
ASSEMBLE INTO RAM
RUN
STEP
RESET
LOAD
HELP
SELECT
OUTPUT
Feil
02
CPU
20 PROGRAM COUNTER
0 INSTRUCTION REGISTER
ADDRESS REGISTER 00
ACCUMULATOR
108
ARITH-METIC UNIT
INPUT
11
01
V1.5b
Little Man Computer
RAM
0 1 2 3 4 5 6 7 8 9
901 324 901 325 124 327 901 326 527 226
10 11 12 13 14 15 16 17 18 19
820 528 922 529 922 530 922 531 922 000
20 21 22 23 24 25 26 27 28 29
527 226 902 000 005 005 011 010 070 101
30 31 32 33 34 35 36 37 38 39
105 108 000 000 000 000 000 000 000 000
40 41 42 43 44 45 46 47 48 49
000 000 000 000 000 000 000 000 000 000
50 51 52 53 54 55 56 57 58 59
000 000 000 000 000 000 000 000 000 000
60 61 62 63 64 65 66 67 68 69
000 000 000 000 000 000 000 000 000 000
70 71 72 73 74 75 76 77 78 79
000 000 000 000 000 000 000 000 000 000
80 81 82 83 84 85 86 87 88 89
000 000 000 000 000 000 000 000 000 000
90 91 92 93 94 95 96 97 98 99
000 000 000 000 000 000 000 000 000 000
Program HALTED, RESET, LOAD, SELECT or alter memory
OPTIONS
© Mike Coley and Peter Higginson

Hva skriver LMC ut dersom bruker gir input **5** og så **1**?

```
          INP
          STA a
          INP
          ADD a
          SUB v
          OUT
          HLT
a         DAT
v         DAT 2
```

Velg ett alternativ:

- 5
- 4
- 2
- -2
- -1

LMC - Aritmetikk

Hva skriver LMC ut dersom bruker gir input **5** og så **1**?

```
          INP
          STA a
          INP
          ADD a
          SUB v
          OUT
          HLT
a         DAT
v         DAT 2
```

Velg ett alternativ:

- -2
- 5
- -1
- 4 ✔
- 2

LMC - Programflyt

```
          INP
          BRP fin
          STA v
          SUB v
          SUB v
fin       OUT
          HLT
v         DAT
```

Programmet ovenfor skriver ut **2**. Hva kan bruker ha gitt som **input** til LMC?
Velg ett eller flere alternativer

- [ ] -2
- [ ] 3 og så 1
- [ ] 2
- [ ] -1
- [ ] 5
- [ ] 4

```
          INP
          BRP fin
          STA v
          SUB v
          SUB v
fin       OUT
          HLT
v         DAT
```

Programmet ovenfor skriver ut **2**. Hva kan bruker ha gitt som **input** til LMC?
Velg ett eller flere alternativer

- [ ] 5
- [x] 2
- [x] -2
- [ ] -1
- [ ] 4
- [ ] 3 og så 1

OBLIG OG UKESOPPGAVER

IN1020 Introduksjon til datateknologi | Gruppe 15

TAKK FOR IDAG!