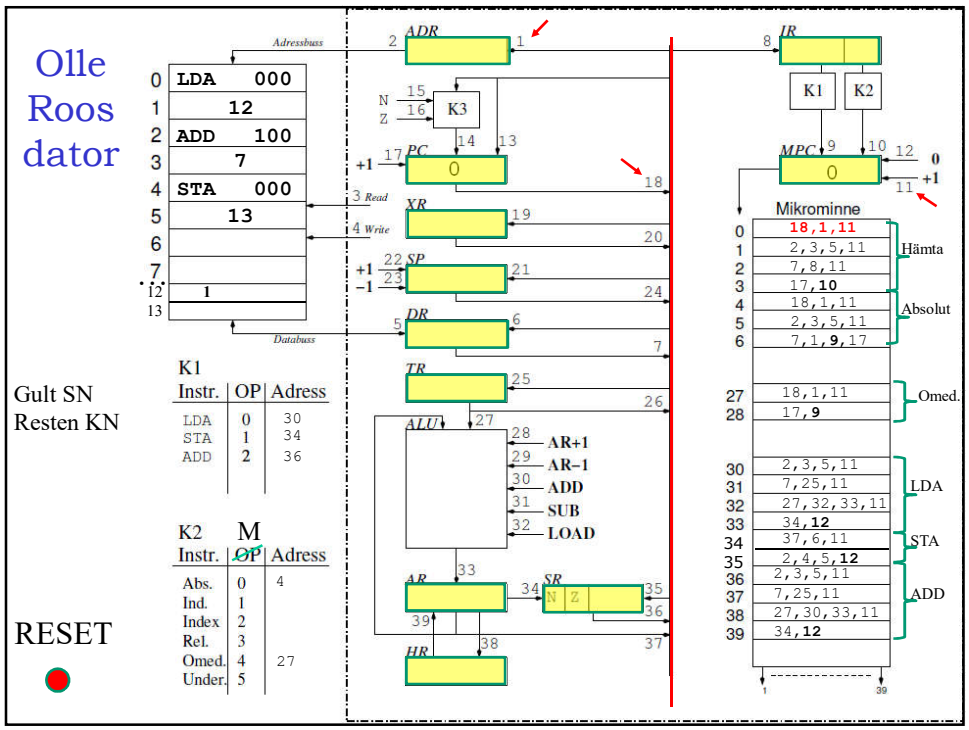
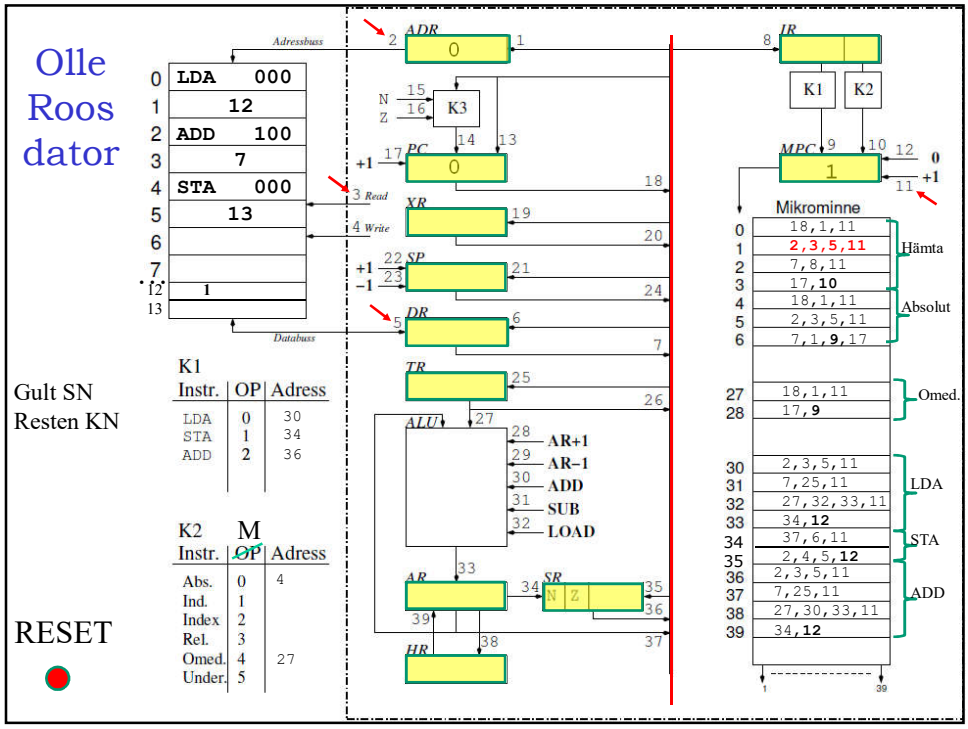
3. Mikroprogrammering II

- lite repetition
- in/ut-matning
- avbrott på OR-datorn
- hoppinstruktion
- labben

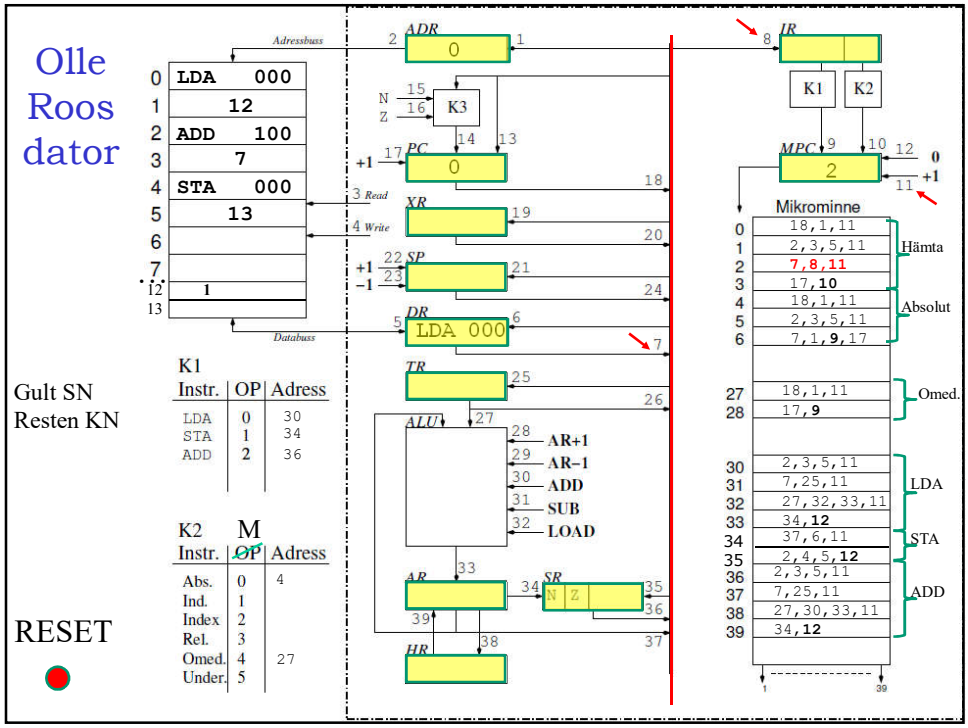
1



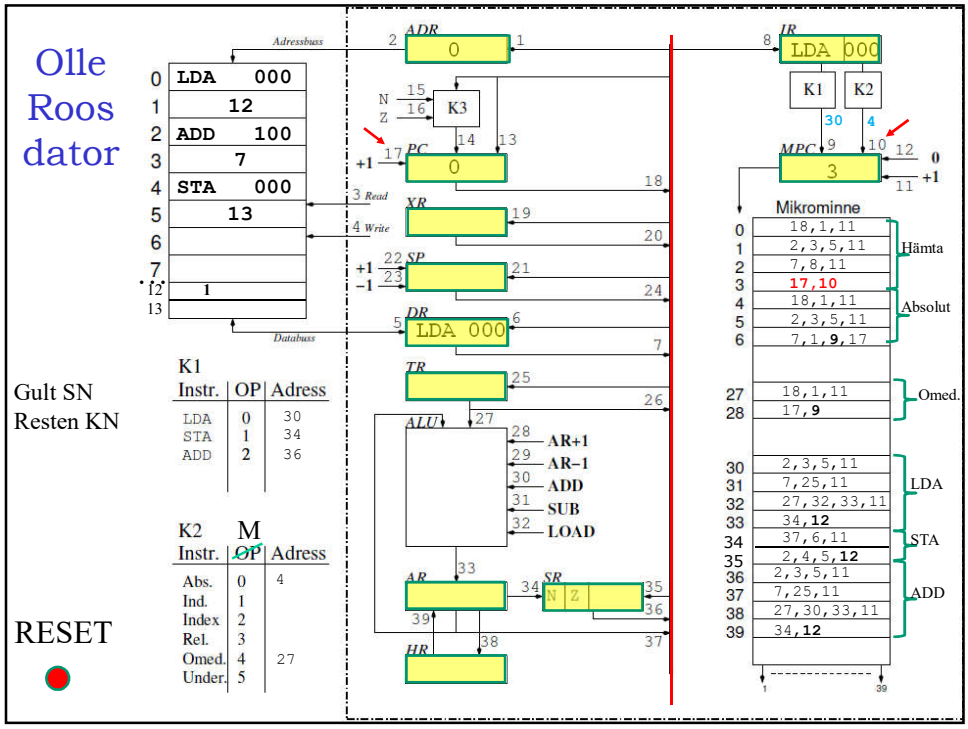
2


Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR
PC
XR
SP
DR
TR
ALU
AR
SR
HR
IR
K1
K2
K3
MPC
Mikrominne
Hämta
Absolut
Omed.
LDA
STA
ADD
AR+1
AR−1
ADD
SUB
LOAD
Read
Write

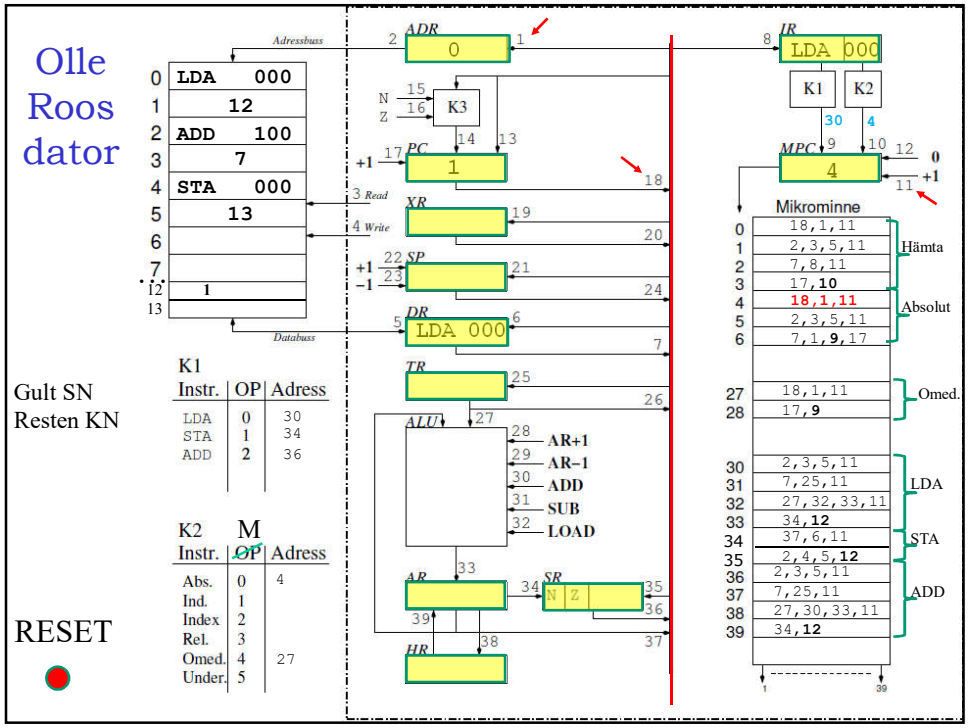
3


Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR
PC
XR
SP
DR
LDA 000
TR
ALU
AR
SR
HR
IR
K1
K2
K3
MPC
Mikrominne
Hämta
Absolut
Omed.
LDA
STA
ADD
AR+1
AR−1
ADD
SUB
LOAD
Read
Write

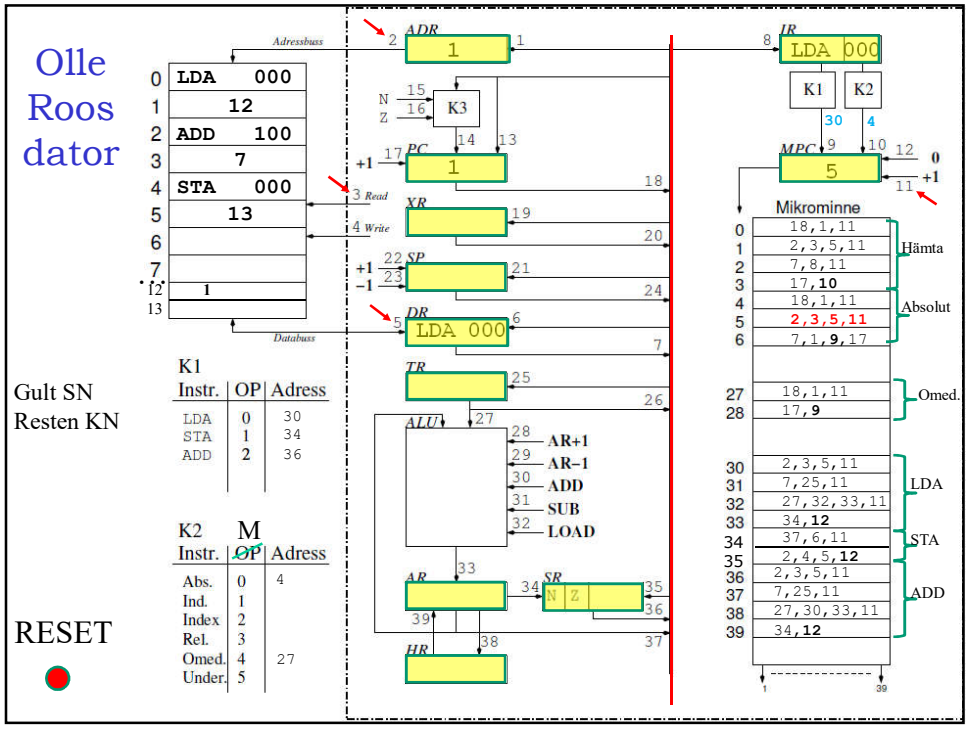
4


Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR 0
PC 0
XR
SP
DR LDA 000
TR
ALU
AR
SR N Z
HR
K3
N
Z
3 Read
4 Write
+1
-1
AR+1
AR−1
ADD
SUB
LOAD
IR LDA 000
K1
K2
MPC 3
Mikrominne
0 18,1,11
1 2,3,5,11
2 7,8,11
3 17,10
4 18,1,11
5 2,3,5,11
6 7,1,9,17
27 18,1,11
28 17,9
30 2,3,5,11
31 7,25,11
32 27,32,33,11
33 34,12
34 37,6,11
35 2,4,5,12
36 2,3,5,11
37 7,25,11
38 27,30,33,11
39 34,12
Hämta
Absolut
Omed.
LDA
STA
ADD

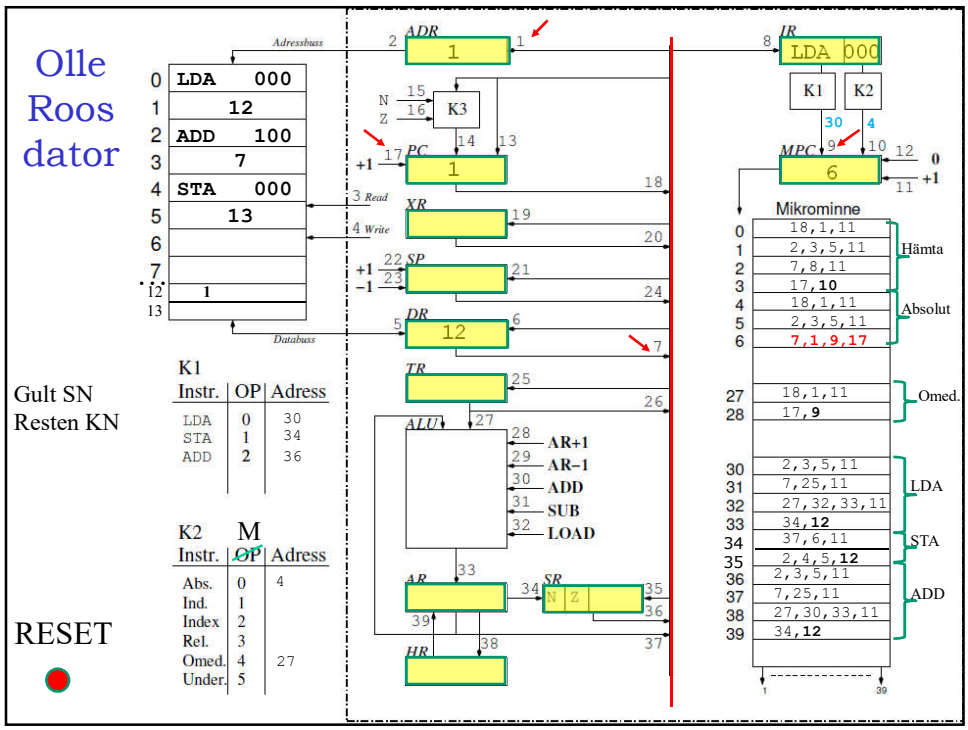
5


Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR 0
PC 1
XR
SP
DR LDA 000
TR
ALU
AR
SR N Z
HR
K3
N
Z
3 Read
4 Write
+1
-1
AR+1
AR−1
ADD
SUB
LOAD
IR LDA 000
K1
K2
MPC 4
Mikrominne
0 18,1,11
1 2,3,5,11
2 7,8,11
3 17,10
4 18,1,11
5 2,3,5,11
6 7,1,9,17
27 18,1,11
28 17,9
30 2,3,5,11
31 7,25,11
32 27,32,33,11
33 34,12
34 37,6,11
35 2,4,5,12
36 2,3,5,11
37 7,25,11
38 27,30,33,11
39 34,12
Hämta
Absolut
Omed.
LDA
STA
ADD

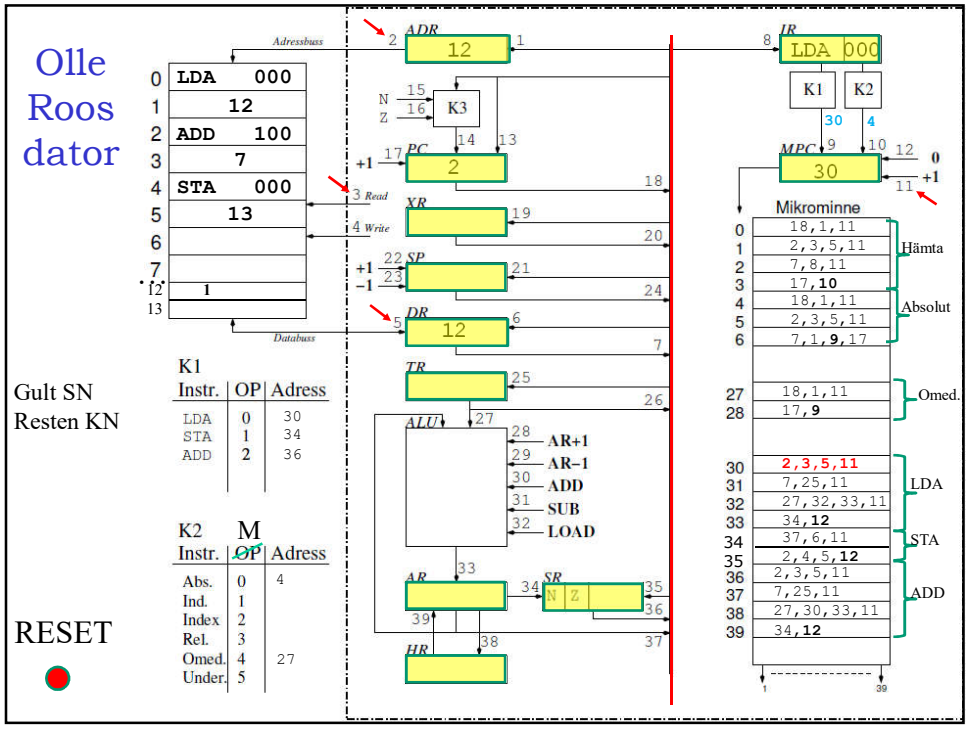
6


Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR
PC
XR
SP
DR
TR
ALU
AR
SR
HR
IR
MPC
K1
K2
K3
N
Z
Read
Write
AR+1
AR−1
ADD
SUB
LOAD
Mikrominne
Hämta
Absolut
Omed.
LDA
STA
ADD

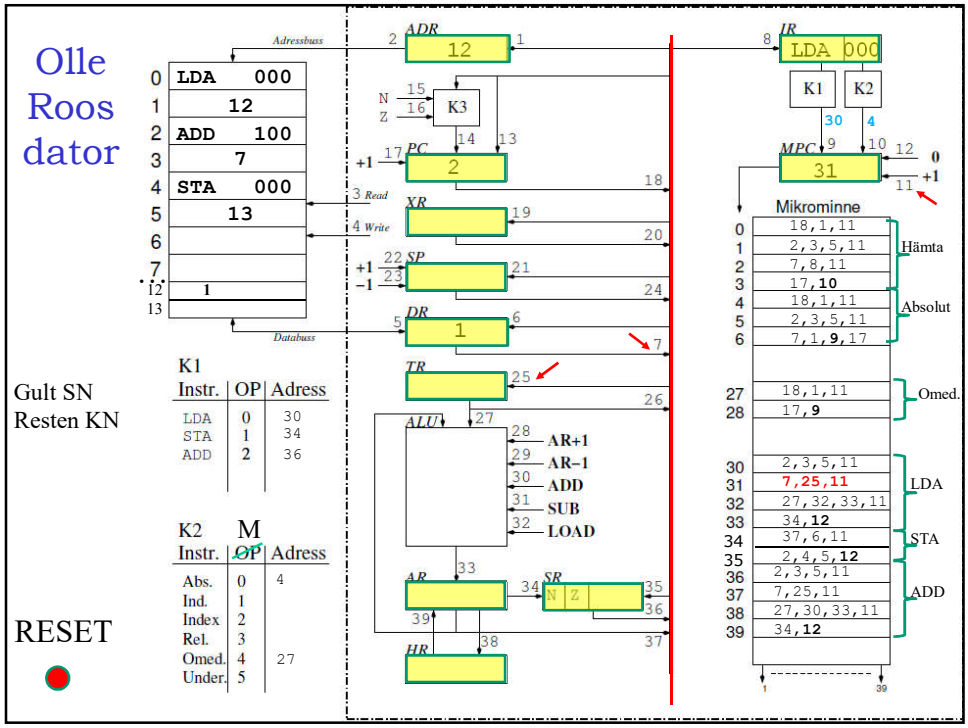
7


Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR
PC
XR
SP
DR
TR
ALU
AR
SR
HR
IR
MPC
K1
K2
K3
N
Z
Read
Write
AR+1
AR−1
ADD
SUB
LOAD
Mikrominne
Hämta
Absolut
Omed.
LDA
STA
ADD

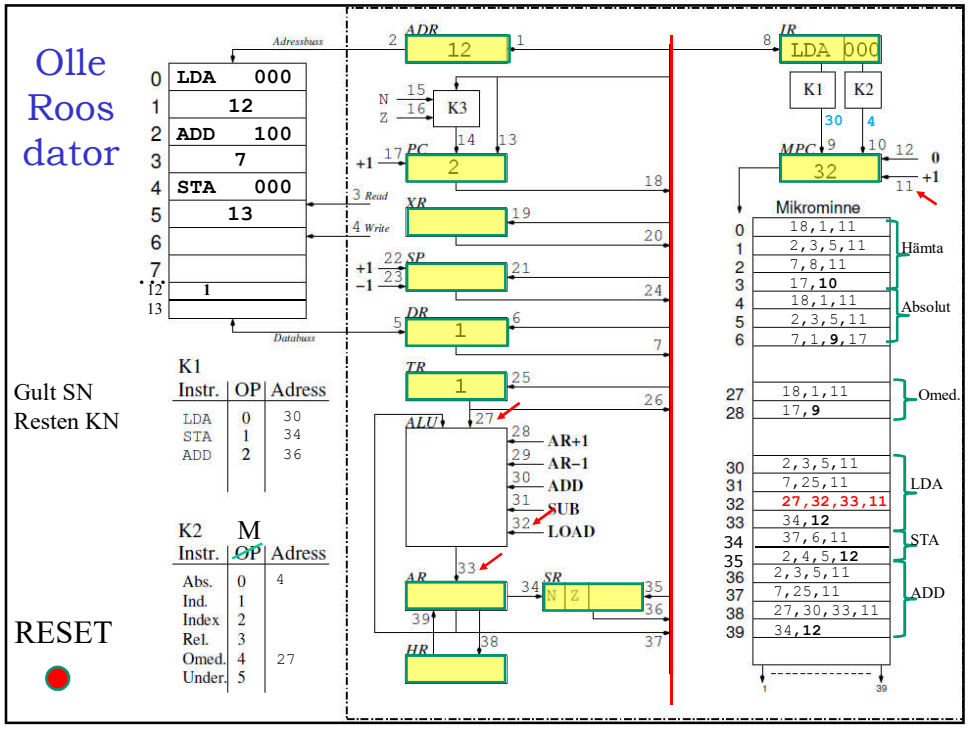
8


Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR 12
PC 2
XR
SP
DR 12
TR
ALU
AR
SR N Z
HR
K3
3 Read
4 Write
AR+1
AR−1
ADD
SUB
LOAD
IR LDA 000
K1 K2
MPC 30
Mikrominne
0 18,1,11
1 2,3,5,11
2 7,8,11
3 17,10
4 18,1,11
5 2,3,5,11
6 7,1,9,17
27 18,1,11
28 17,9
30 2,3,5,11
31 7,25,11
32 27,32,33,11
33 34,12
34 37,6,11
35 2,4,5,12
36 2,3,5,11
37 7,25,11
38 27,30,33,11
39 34,12
Hämta
Absolut
Omed.
LDA
STA
ADD

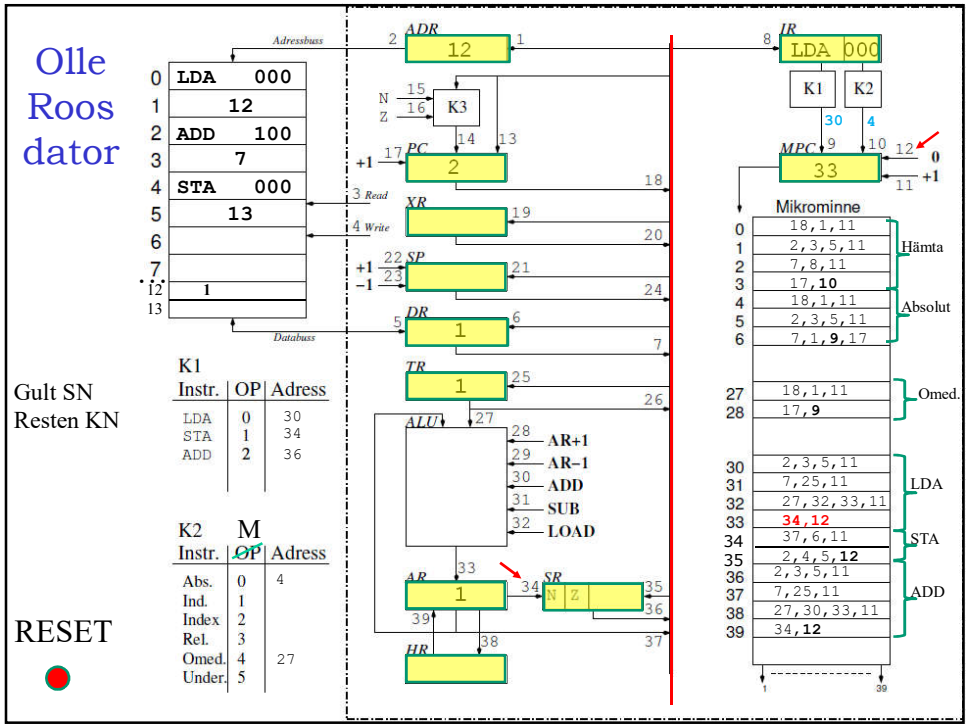
9


Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR 12
PC 2
XR
SP
DR 1
TR
ALU
AR
SR N Z
HR
K3
3 Read
4 Write
AR+1
AR−1
ADD
SUB
LOAD
IR LDA 000
K1 K2
MPC 31
Mikrominne
0 18,1,11
1 2,3,5,11
2 7,8,11
3 17,10
4 18,1,11
5 2,3,5,11
6 7,1,9,17
27 18,1,11
28 17,9
30 2,3,5,11
31 7,25,11
32 27,32,33,11
33 34,12
34 37,6,11
35 2,4,5,12
36 2,3,5,11
37 7,25,11
38 27,30,33,11
39 34,12
Hämta
Absolut
Omed.
LDA
STA
ADD

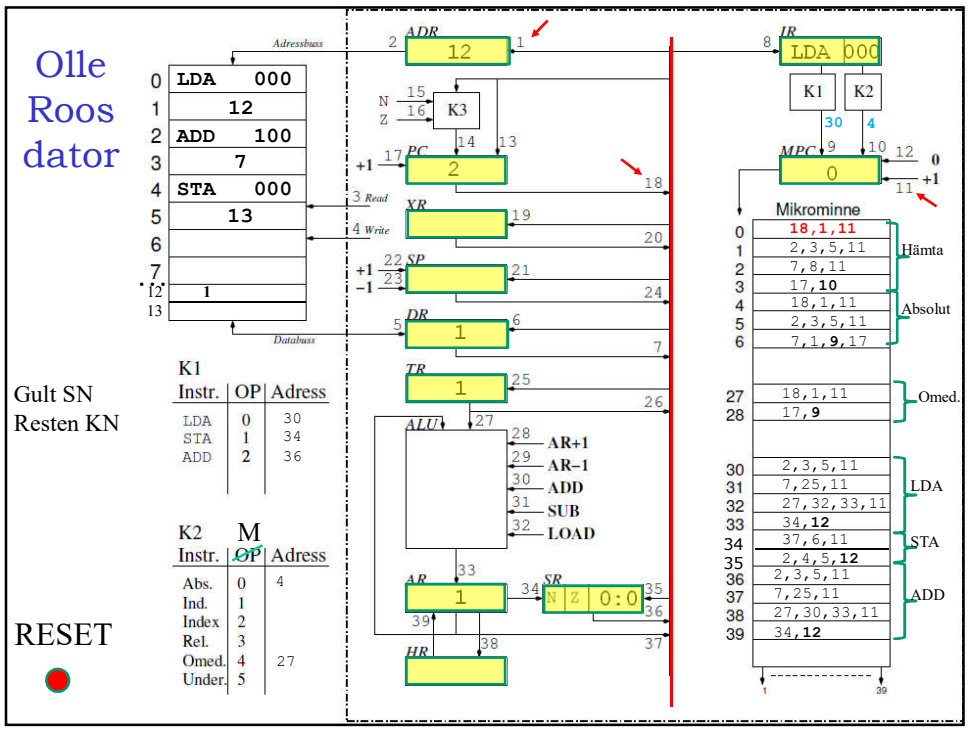
10


Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR 12
PC 2
XR
SP
DR 1
TR 1
ALU
AR
SR N Z
HR
K3
3 Read
4 Write
AR+1
AR−1
ADD
SUB
LOAD
IR LDA 000
K1 K2
MPC 32
Mikrominne
0 18,1,11
1 2,3,5,11
2 7,8,11
3 17,10
4 18,1,11
5 2,3,5,11
6 7,1,9,17
27 18,1,11
28 17,9
30 2,3,5,11
31 7,25,11
32 27,32,33,11
33 34,12
34 37,6,11
35 2,4,5,12
36 2,3,5,11
37 7,25,11
38 27,30,33,11
39 34,12
Hämta
Absolut
Omed.
LDA
STA
ADD

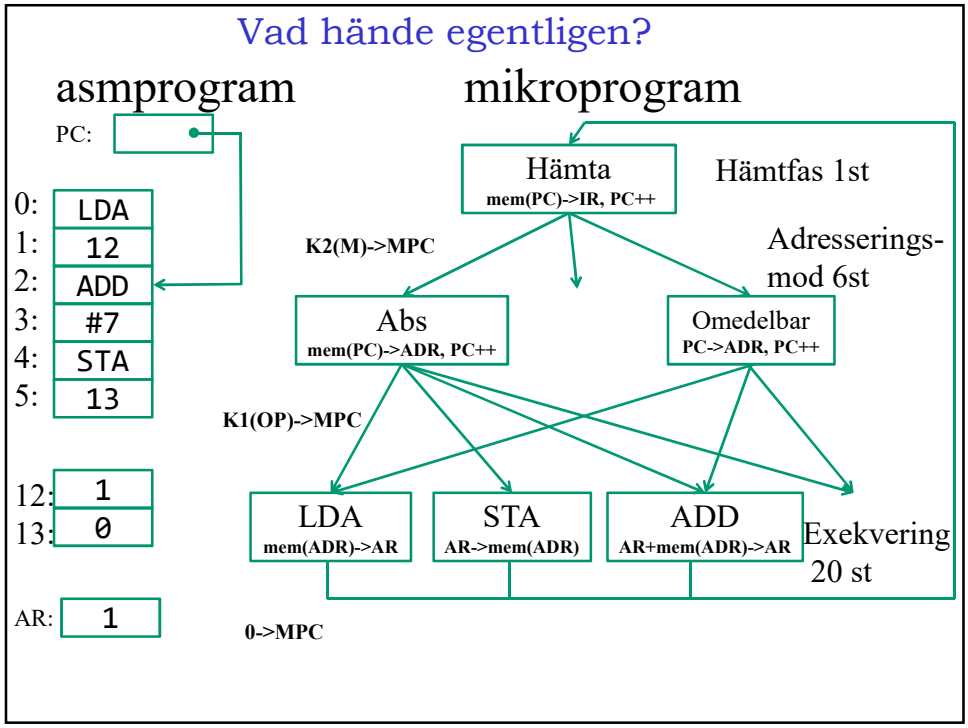
11


Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR 12
PC 2
XR
SP
DR 1
TR 1
ALU
AR 1
SR N Z
HR
K3
3 Read
4 Write
AR+1
AR−1
ADD
SUB
LOAD
IR LDA 000
K1 K2
MPC 33
Mikrominne
0 18,1,11
1 2,3,5,11
2 7,8,11
3 17,10
4 18,1,11
5 2,3,5,11
6 7,1,9,17
27 18,1,11
28 17,9
30 2,3,5,11
31 7,25,11
32 27,32,33,11
33 34,12
34 37,6,11
35 2,4,5,12
36 2,3,5,11
37 7,25,11
38 27,30,33,11
39 34,12
Hämta
Absolut
Omed.
LDA
STA
ADD

12

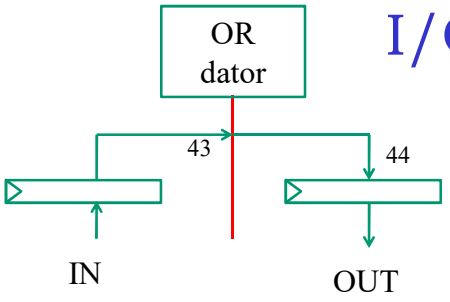

Olle
Roos
dator
Gult SN
Resten KN
RESET
Adressbuss
Databuss
0 LDA 000
1 12
2 ADD 100
3 7
4 STA 000
5 13
12 1
13
K1
Instr. OP Adress
LDA 0 30
STA 1 34
ADD 2 36
K2 M
Instr. OP Adress
Abs. 0 4
Ind. 1
Index 2
Rel. 3
Omed. 4 27
Under. 5
ADR
12
PC
2
XR
SP
DR
1
TR
1
ALU
AR+1
AR−1
ADD
SUB
LOAD
AR
1
SR
N Z 0:0
HR
IR
LDA 000
K1
K2
MPC
0
Mikrominne
0 18,1,11
1 2,3,5,11
2 7,8,11
3 17,10
4 18,1,11
5 2,3,5,11
6 7,1,9,17
27 18,1,11
28 17,9
30 2,3,5,11
31 7,25,11
32 27,32,33,11
33 34,12
34 37,6,11
35 2,4,5,12
36 2,3,5,11
37 7,25,11
38 27,30,33,11
39 34,12
Hämta
Absolut
Omed.
LDA
STA
ADD

13


Vad hände egentligen?
asmprogram
mikroprogram
PC:
0: LDA
1: 12
2: ADD
3: #7
4: STA
5: 13
12: 1
13: 0
AR: 1
Hämta
mem(PC)->IR, PC++
Hämtfas 1st
K2(M)->MPC
Adresserings-
mod 6st
Abs
mem(PC)->ADR, PC++
Omedelbar
PC->ADR, PC++
K1(OP)->MPC
LDA
mem(ADR)->AR
STA
AR->mem(ADR)
ADD
AR+mem(ADR)->AR
Exekvering
20 st
0->MPC

14

I/O



Instruktion	Betydelse
IN	$AR := IN$
OUT	$OUT := AR$

Adresseringsmod: underförstådd K2(5) = 29

```
29: K1->mpc          9
```

IN

```
50: in->tr, mpc++    43,25,11
51: tr->ar, mpc++    27,32,33,11
52: status, 0->mpc   34,12
```

OUT

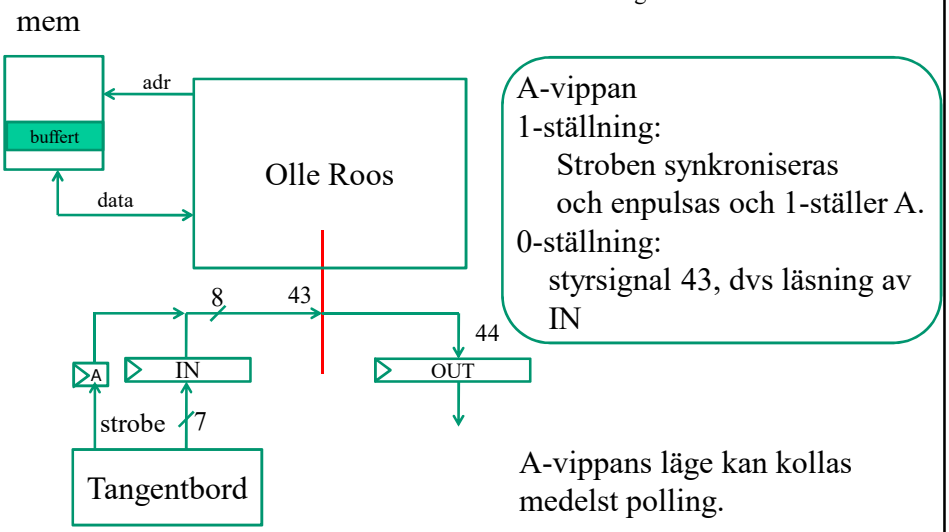
```
53: ar->out, 0->mpc  37,44,12
```

15

In-matning

(tangenttryckningar –> buffert)

Problem:
Hur vet vi när det finns ett nytt värde i IN-registret?



A-vippan
1-ställning:
Stroben synkroniseras och enpulsas och 1-ställer A.
0-ställning:
styrsignal 43, dvs läsning av IN

A-vippans läge kan kollas medelst polling.

16

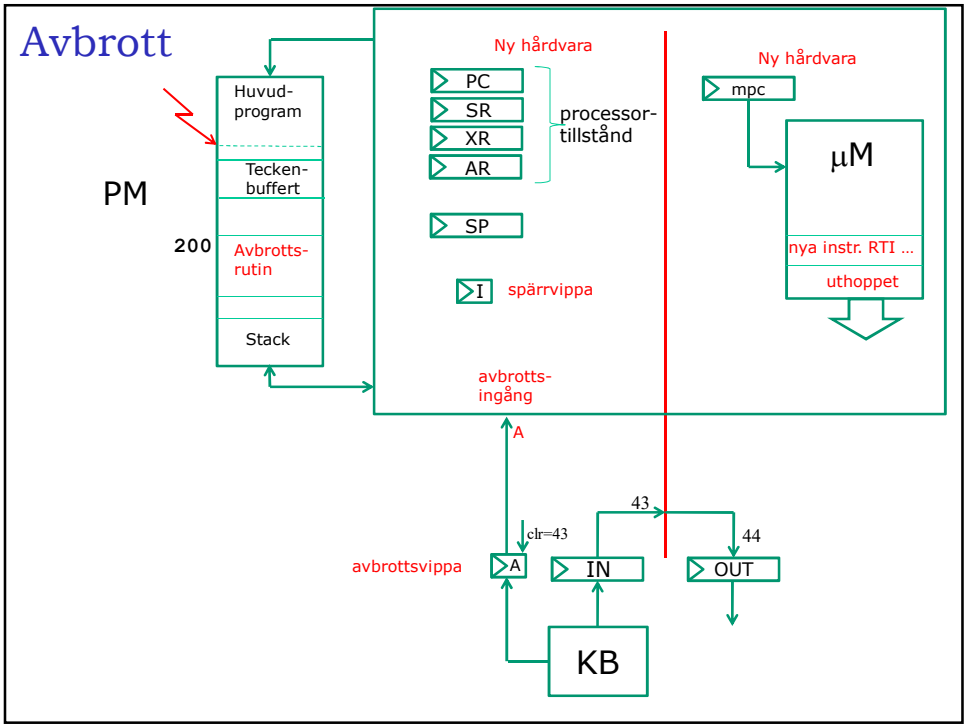
Olika metoder för I/O

1) Programmet väntar på att stroben ska bli hög,
(testa om IN är negativ)
läser tecknet och placerar i minnet.
Programstyrd I/O, polling, busy waiting.

2) Programmet behöver inte alls vänta på stroben.
När stroben går hög startar en avbrottsrutin, som
läser in tecknet och placerar i minnet. **Avbrott.**

3) I/O kretsen skriver själv (genom att ta över lämpliga bussar)
i minnet. **DMA = direkt minnesaccess.** Programmet behöver
bara uppmärksammas när return har kommit in.
Kan ske genom att koppla bort CPUn från bussarna eller
genom att utnyttja lediga minnescykler.

17



18

Avbrott

1. Tryck på en tangent => 1->A
2. Gör klart pågående instruktion
3. Om I=0 så uthopp:
   1. Spara reg. på stacken
      PC, SR, XR, AR
   2. Förhindra fler avbrott: 1->I.
   3. Hoppa till avbrottsrutinen
4. I avbrottsrutinen:
   1. Läs in tecknet till minnesbuffer och 0->A
   2. Återhopp RTI (återställ reg., 0->I)

3 nya instruktioner:
EI: 0->I (Enable Interrupt)
DI: 1->I (Disable Interrupt)
RTI: återhopp

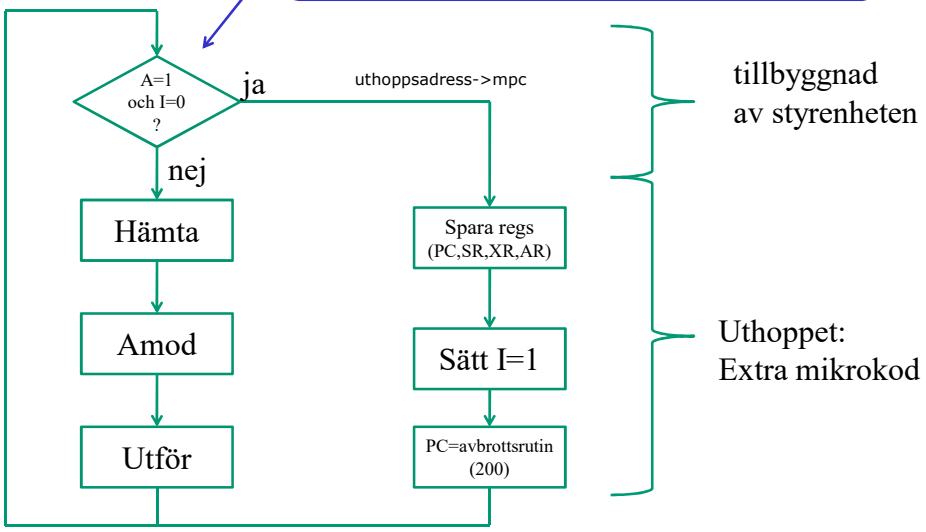
Mikrokod för uthopp

Förbättrad mpc

19

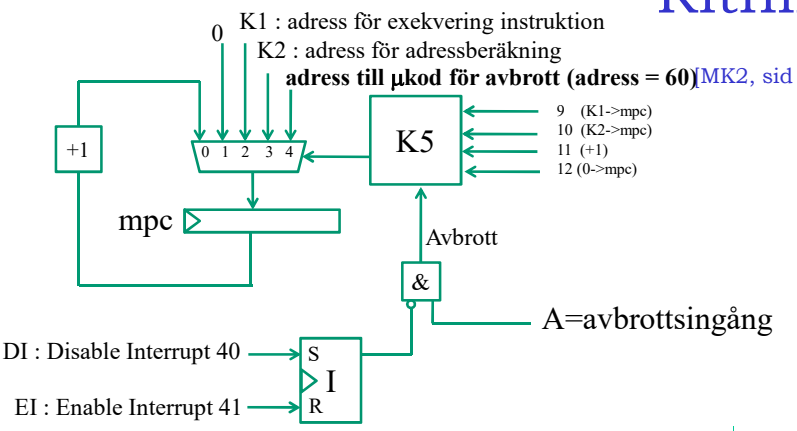
Ritning [MK2, sid 119]

Styrenheten kollar om det är avbrott mellan två instruktioner



20

Ritning



K5:	9	10	11	12	Avbr	mux-ingång
	1	0	0	0	-	2
	0	1	0	0	-	3
	0	0	1	0	-	0
	0	0	0	1	0	1
	0	0	0	1	1	4

Adress (200) till avbrottsrutin

21

Ritning [MK2, sid 120-121]

```
60: sp->adr, mpc++          24,1,11
61: pc->dr,mpc++            18,6,11
62: skriv, sp--,mpc++       2,4,5,23,11
```
PC

```
63: sp->adr, mpc++          24,1,11
64: xr->dr, mpc++           20,6,11
65: skriv, sp--,mpc++       2,4,5,23,11
```
XR

```
66: sp->adr, mpc++          24,1,11
67: ar->dr, mpc++           37,6,11
68: skriv, sp--,mpc++       2,4,5,23,11
```
AR

```
69: sp->adr, mpc++          24,1,11
70: sr->dr, mpc++           36,6,11
71: skriv, sp--,mpc++       2,4,5,23,11
72: 1->I,mpc++              40,11
73: 200->PC, mpc=0  42,12
```
SR

71,72,73 kan slås ihop

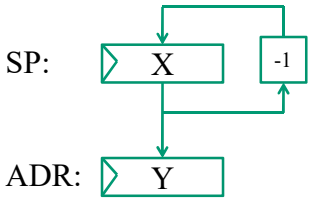
22

Något kortare variant

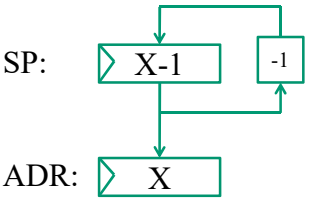
```
60: sp->adr, mpc++          60: sp->adr, sp--, mpc++
61: pc->dr, mpc++           61: pc->dr, mpc++
62: skriv, sp--, mpc++      62: skriv, sp->adr, mpc++
63: sp->adr, mpc++
```

Ny rad 60

-Före klockflank:



-Efter klockflank:

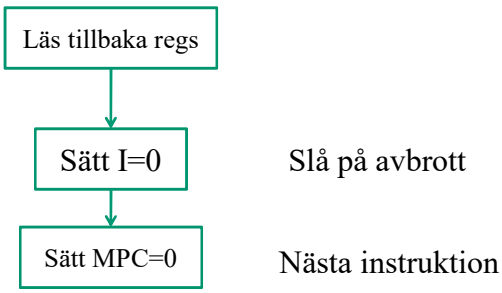


23

Ritning [MK2, sid 119]

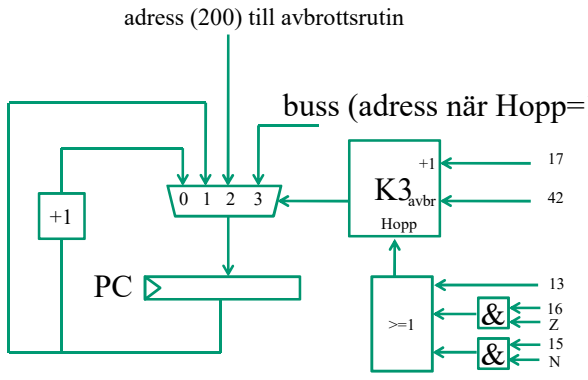
Ny instruktion: **RTI**

=> Vanlig instruktion, mikrokod för exekveringsfasen



24

Hopp



K3

Hopp	+1	avbr	mux-ing.	PC
1	0	0	3	buss
0	1	0	0	PC+1
0	0	1	2	avbr.rutin
0	0	0	1	PC
…				

25

Mikrokod för JMPN D

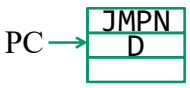
Om N=1
PC+2+D -> PC
annars
PC+2 -> PC

K2(3) = 19

självrelativ a-mod (utförs oavsett värde på N-flaggan)

```
19: PC->ADR, PC++, MPC++
20: PC->TR, minne->DR, MPC++
21: DR->TR, TR->AR, AR->HR, MPC++
22: AR+TR->AR, MPC++
23: HR->AR, AR->TR, K1->MPC
```

Före självrel. a-mod:



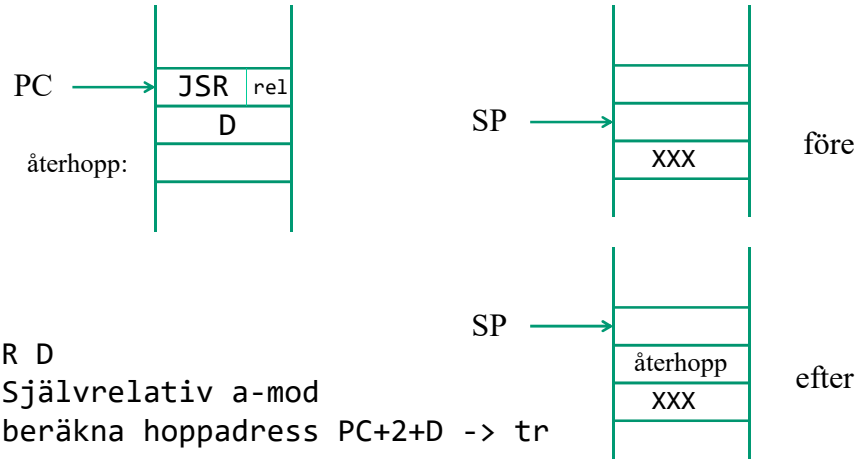
K1(17)=78

Exekvering för JMPN (TR->PC om N=1)

```
78: TR->PC(N), 0->MPC              26,15,12
```

26

Subrutinhopp JSR D



JSR D
- Självrelativ a-mod
  beräkna hoppadress PC+2+D -> tr
- exe
  PC+2 -> mem(SP), SP--
  tr -> PC

27

Subrutinåterhopp RTS

* underförstådd a-mod

* exe

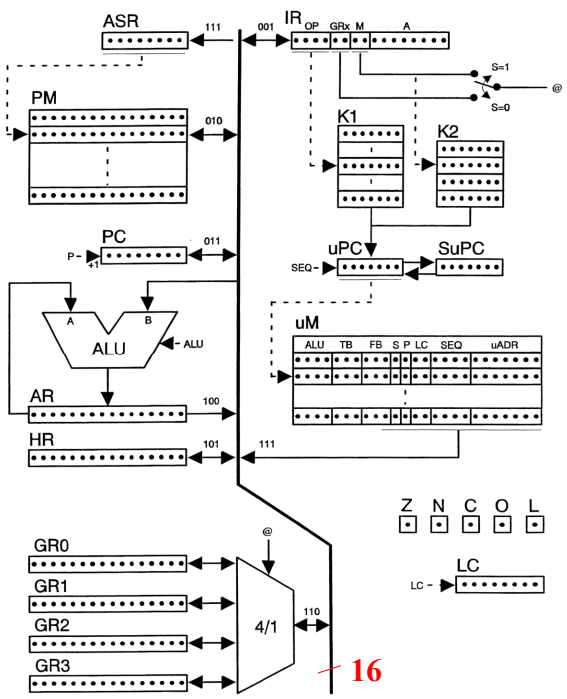
```
SP++              ; justera SP
M(SP) -> PC       ; hoppa tillbaka
```

28

Björn Lindskogs dator

- 1,2,3 ska göras
- Frivillig tävling
- 16-bitars maskin (ASR,PC 8 bitar)
- 4 generella reg.
- AR,HR,ASR arbetsreg.
- Avancerad styrenhet
  - Hopp
  - Villkorliga hopp
  - Loopar (LC,L)
  - Subrutiner (SuPC)
  - Konstanter
- GR3 indexreg

OP	GRx	M	ADR



29