TSEA83 : Datorkonstruktion
Fö2

Mikroprogrammering 1

LINKÖPINGS UNIVERSITET

Fö2 : Agenda

- Att bygga en CPU
  -Mikroprogrammering vs Pipelining
- Mikroprogrammering
  -Grundläggande byggstenar
- Mikromaskinen
  -”Olle Roos”-datorn
- Mikrokod
  -Ett mindre program
  -Ytterligare exempel för olika instruktioner och adresseringsmoder

Att bygga en CPU

Mikroprogrammering vs Pipelining

Att bygga en CPU

Mikroprogrammerad dator (fö2+3)
Björn Lindskog-datorn (lab1)

- Variabel exekveringstid
- Variabelt format
- Inget överlapp
- Central styrenhet, som är mikroprogrammerad
- Flera adresseringsmoder/instruktion
- 1 ackumulator
- Nästan alla instruktioner har operand i minnet:
  LDA Adr ; A=M(Adr)
  ADDA Adr ; A=A+M(Adr)

Pipelinad dator (fö4,lab2)

- Alla instruktioner tar 5 CK
- Alla instruktioner har samma format
- Pipelining/överlapp ger 1 färdig instruktion/CK
- Flera avkodare (inget mprog)
- 1 a-mod/instruktion
- 32 register
- Endast LD/ST har operand i minnet:
  LD Rd,(Ra) ;Rd=M(Ra)
  ADD Rd,Ra,Rb ;Rd=Ra+Rb

Att bygga en CPU

Mikroprogrammerad dator (fö2+3)
Björn Lindskog-datorn (lab1)

- Typisk CISC
- Programmering på 2 nivåer asm och mikro
- Enkel controller:
  - garageportsöppnare
  - del av dator
- + Man kan göra avancerade instruktioner: **sortera**
- **-** Det blir många klockcykler / instr.

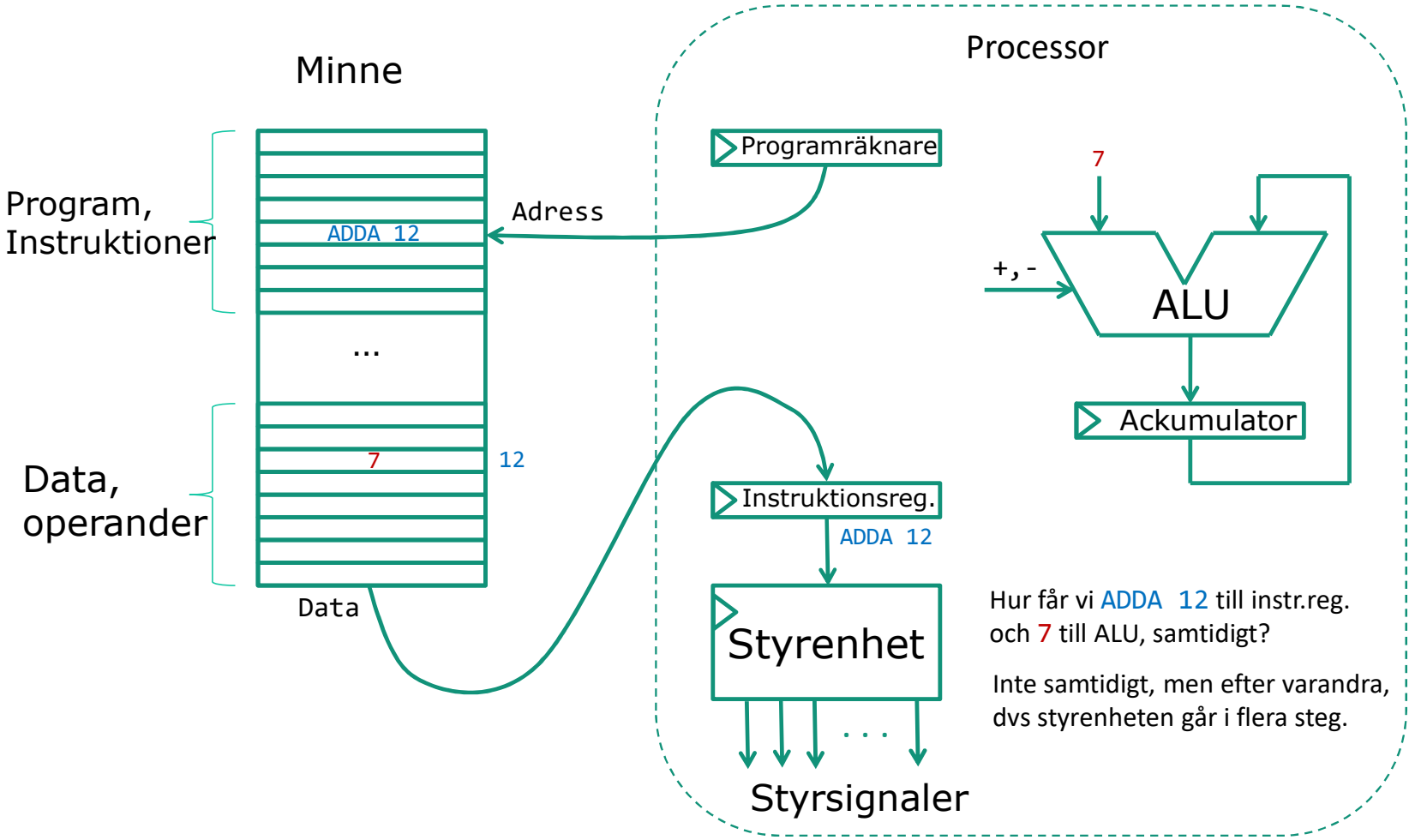
Pipelinad dator (fö4,lab2)

- Typisk RISC
- Programmering på 1 nivå: asm
- Enkel CPU
  - enkel mobil
- Finns bara enkla instr.
- Snabb

Mikroprogrammering

Grundläggande byggstenar

Ritning 1

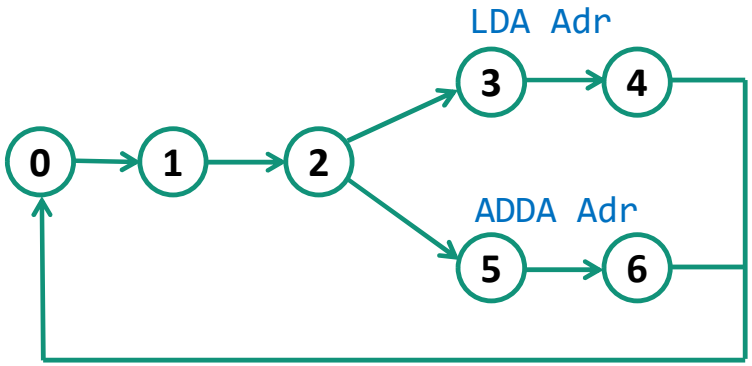


Hur får vi ADDA 12 till instr.reg.
och 7 till ALU, samtidigt?

Inte samtidigt, men efter varandra,
dvs styrenheten går i flera steg.

Mikroprogrammering

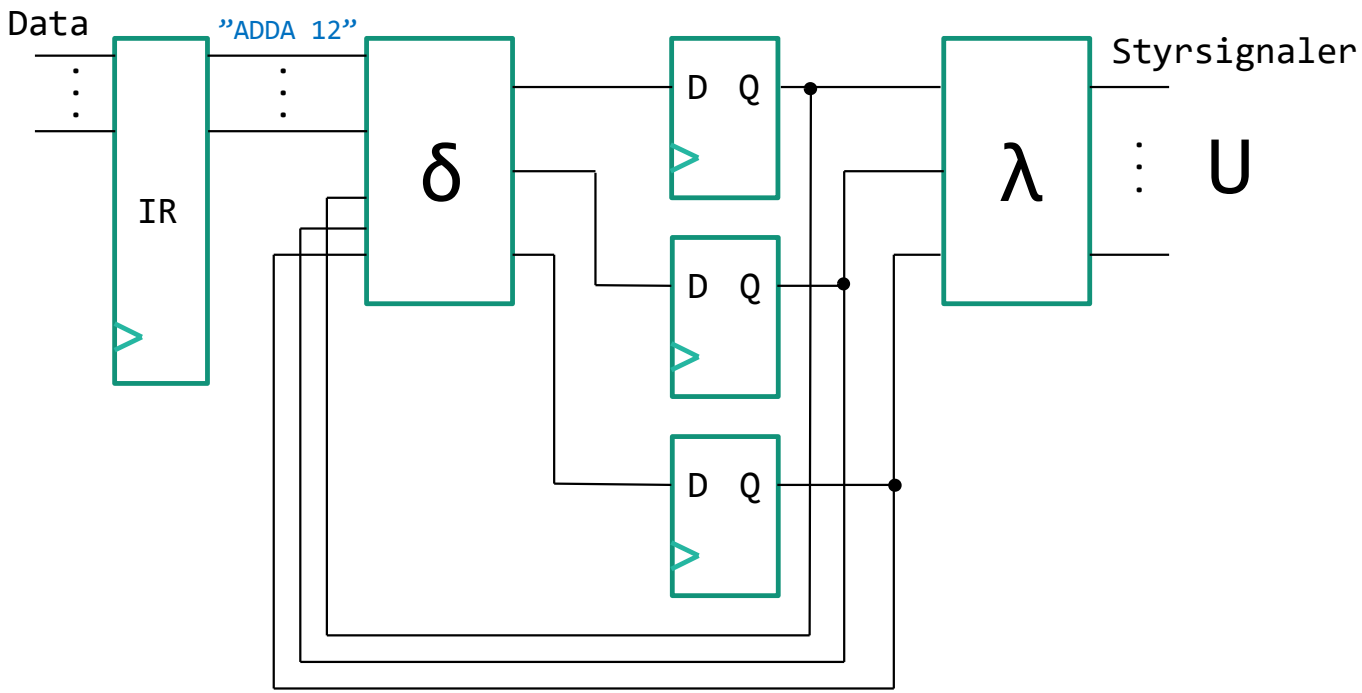
- Vi ska bygga en liten dator med enkla komponenter
- Styrenheten (SekvensNät) visar sig vara svårast. Hur gör man för att konstruera ett SN med 100+ tillstånd?
- Mikroprogrammering är en vidareutveckling (och faktiskt begränsning) av tekniken att bygga sekvensnät med ROM
- Idé: byt tillståndsvipporna mot en universalräknare



Tänkbara händelser vid olika steg:
Steg 0-2: Hämta instruktion, samt beräkna adress
Steg 3-4: Utför en instruktion
Steg 5-6: Utför en annan instruktion

Mikroprogrammering

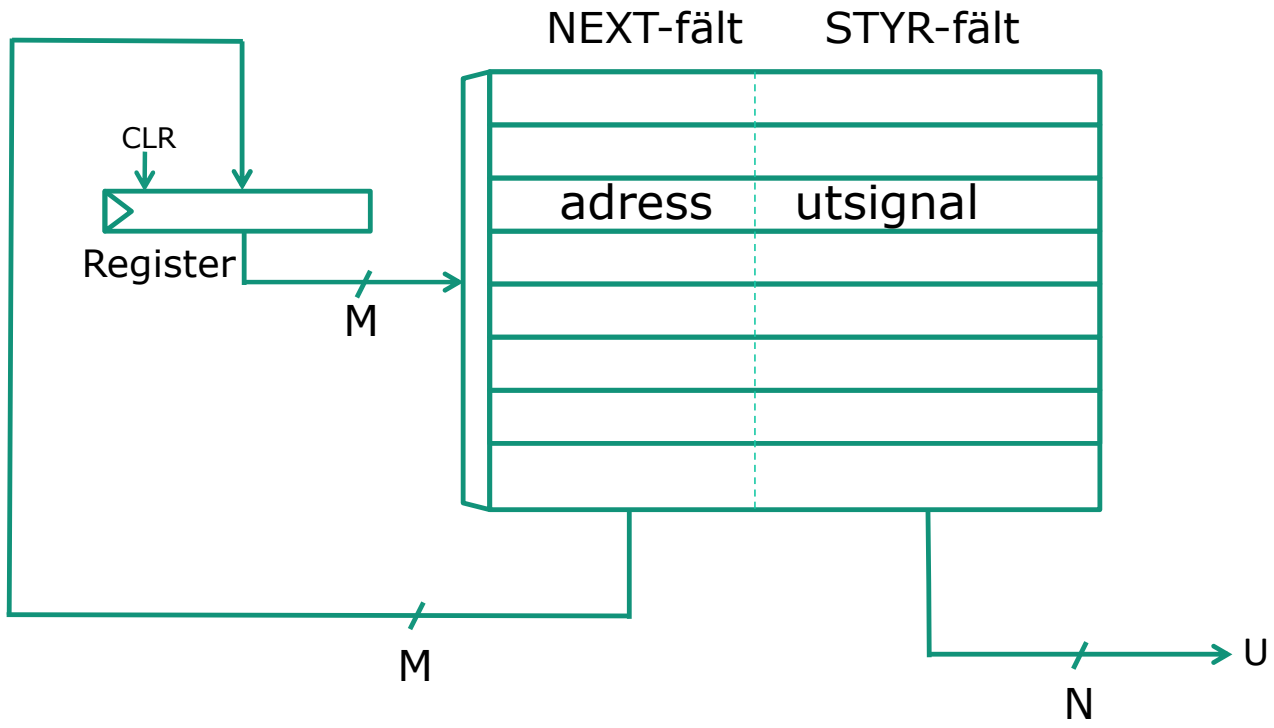
Vi skulle kunna bygga sekvensnätet på ”vanligt” sätt:



Men det blir väldigt jobbigt för 100+ tillstånd,
dessutom omständligt att konfigurera om vid behov.

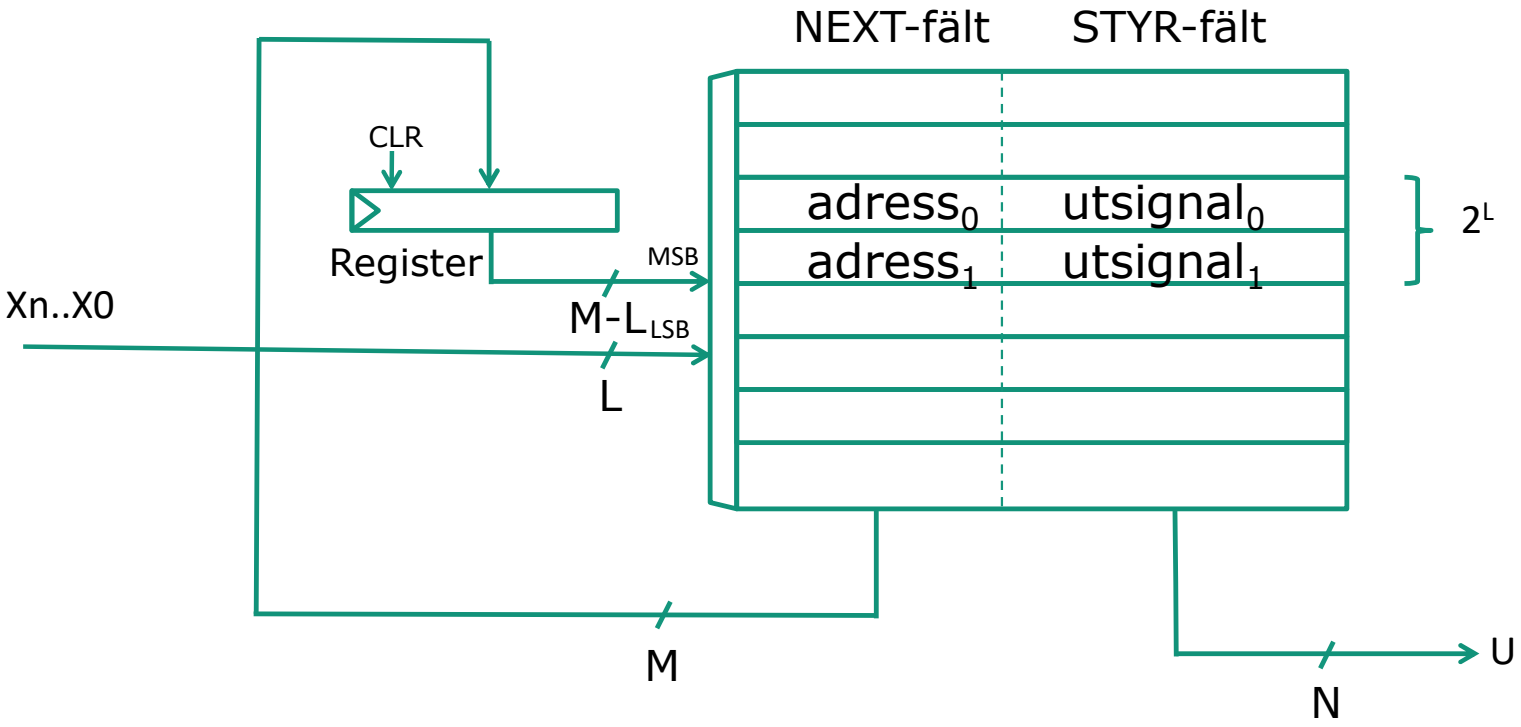
Ritning 2

Autonom styrenhet med ROM/Register



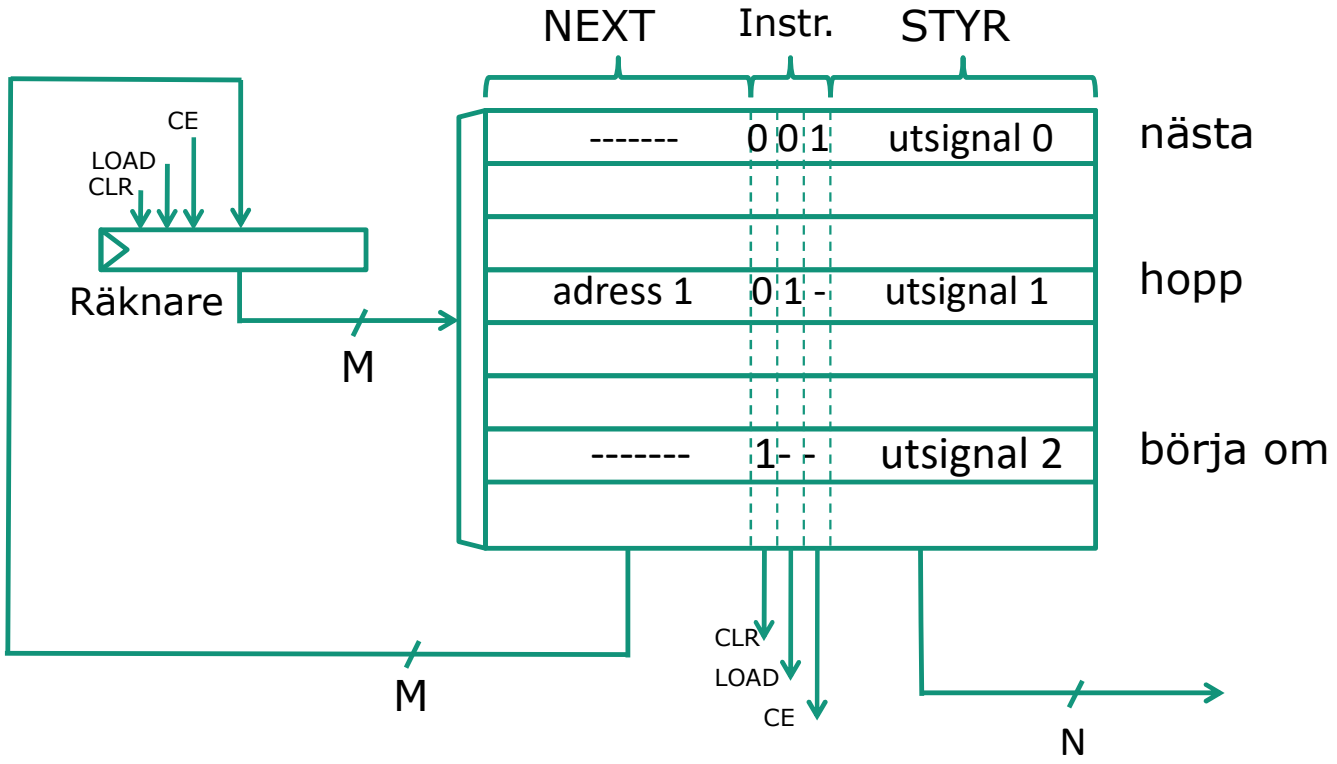
Variant 2

Autonom styrenhet med ROM/Register

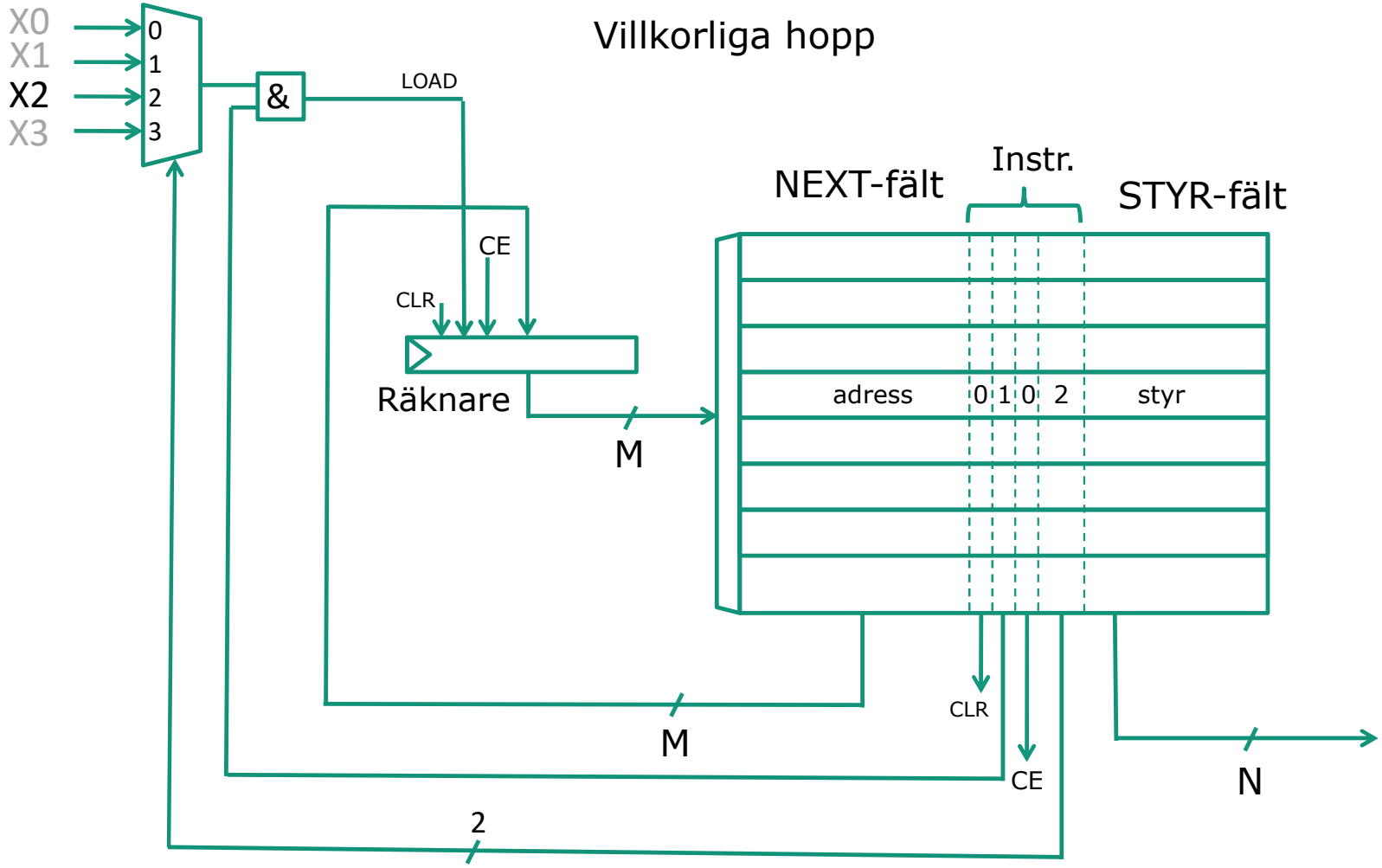


Ritning 3

Autonom styrenhet med ROM/Räknare

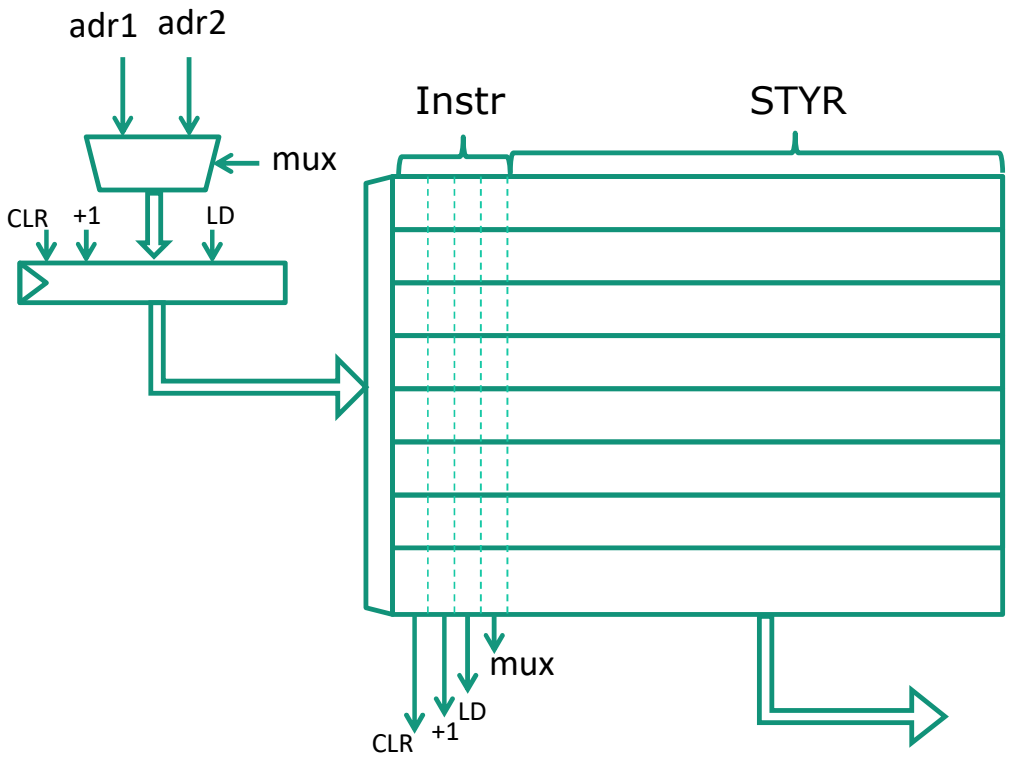


Ritning 4

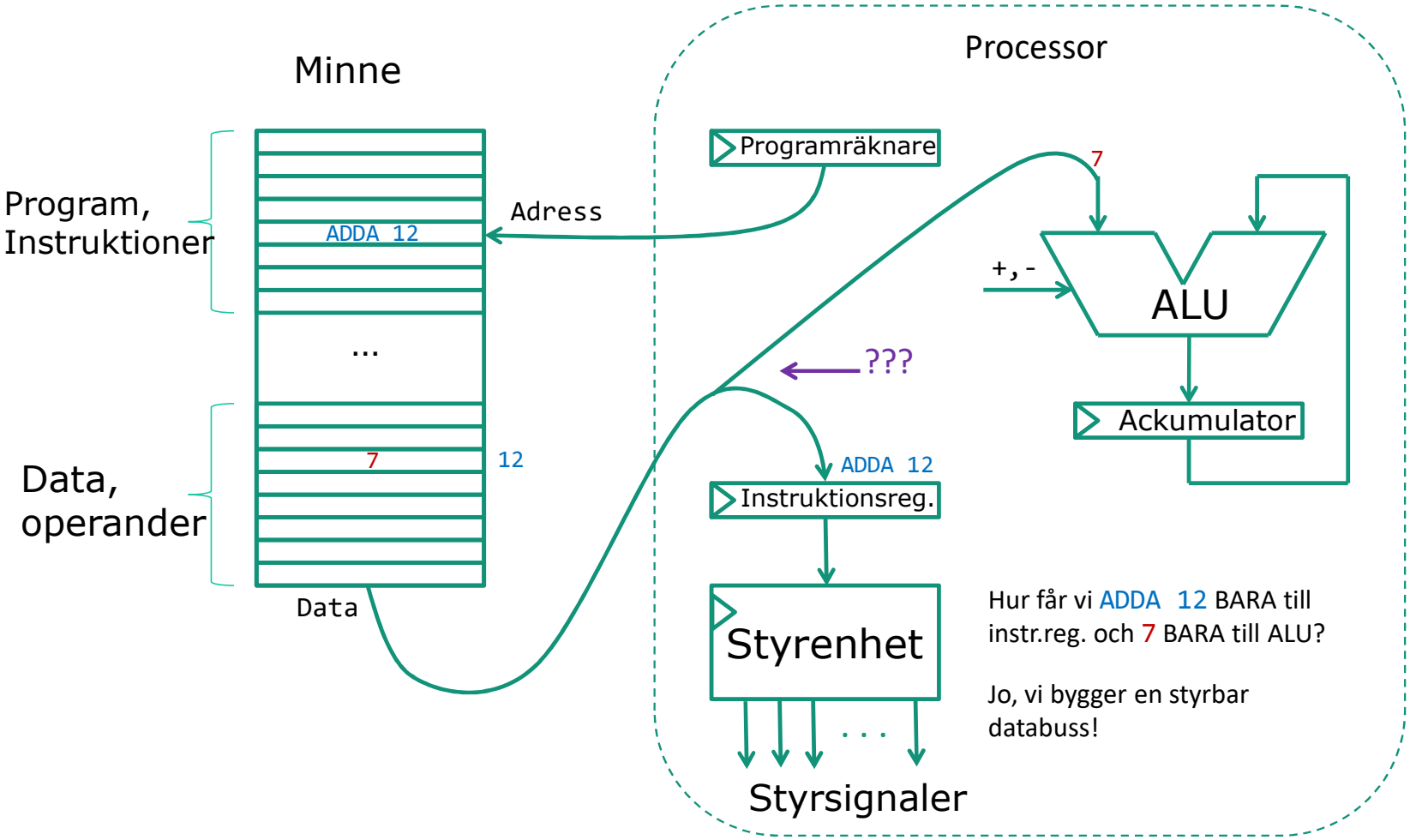

Villkorliga hopp
X0
X1
X2
X3
0
1
2
3
&
LOAD
CE
CLR
Räknare
M
NEXT-fält
Instr.
STYR-fält
adress
0 1 0 2
styr
M
CLR
CE
N
2

En variant

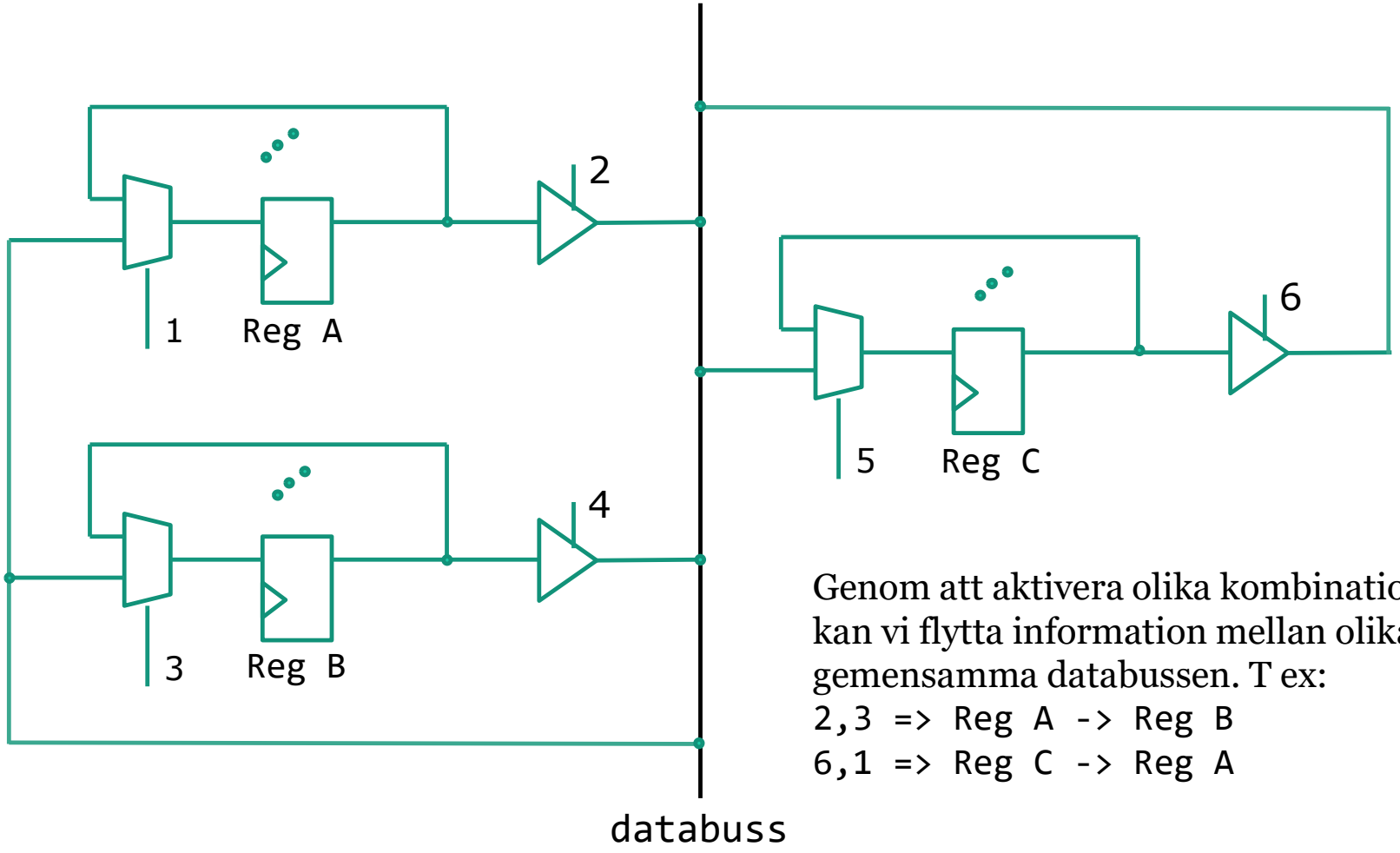
Autonom styrenhet utan adressfält



Databussen då?



Ritning 5



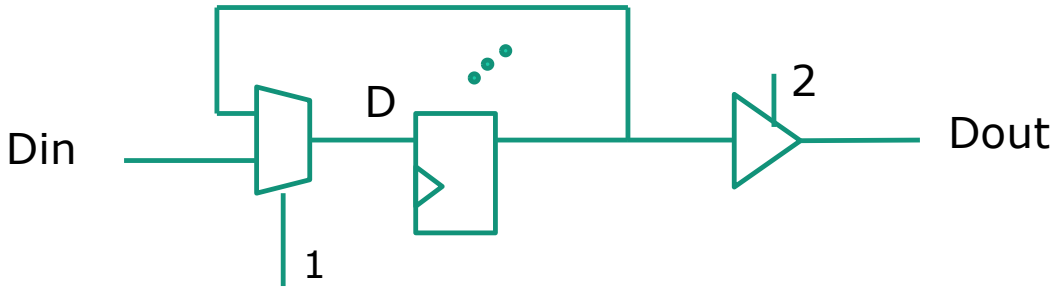
Genom att aktivera olika kombinationer av styrsignaler kan vi flytta information mellan olika register via den gemensamma databussen. T ex:

```
2,3 => Reg A -> Reg B
6,1 => Reg C -> Reg A
```

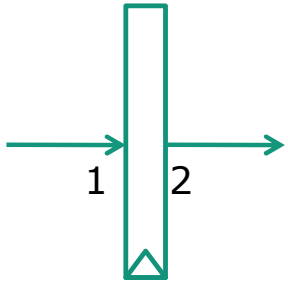
Ritning 5

Register

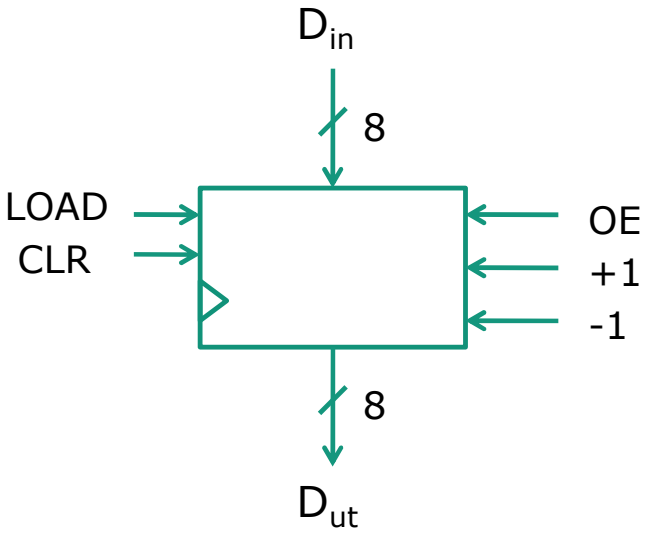
Det som egentligen ser ut så här =>



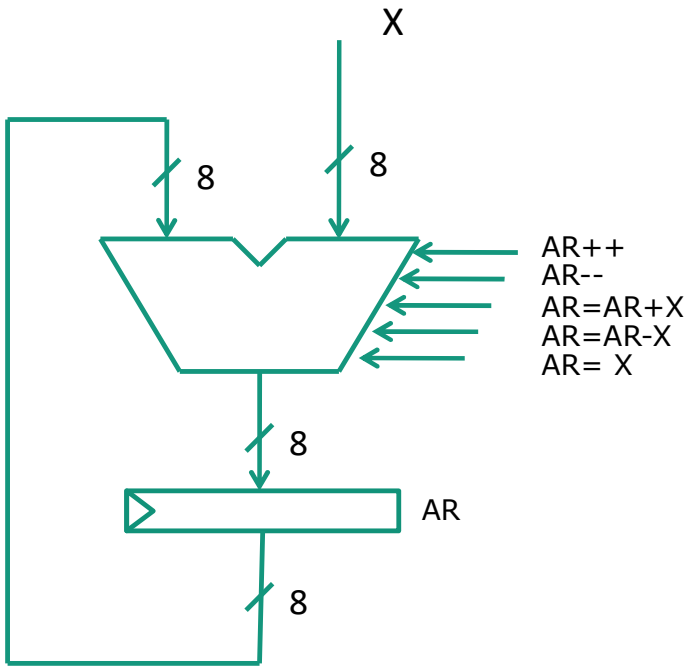
Förenklar vi genom att rita så här =>



Ritning 6 och 7

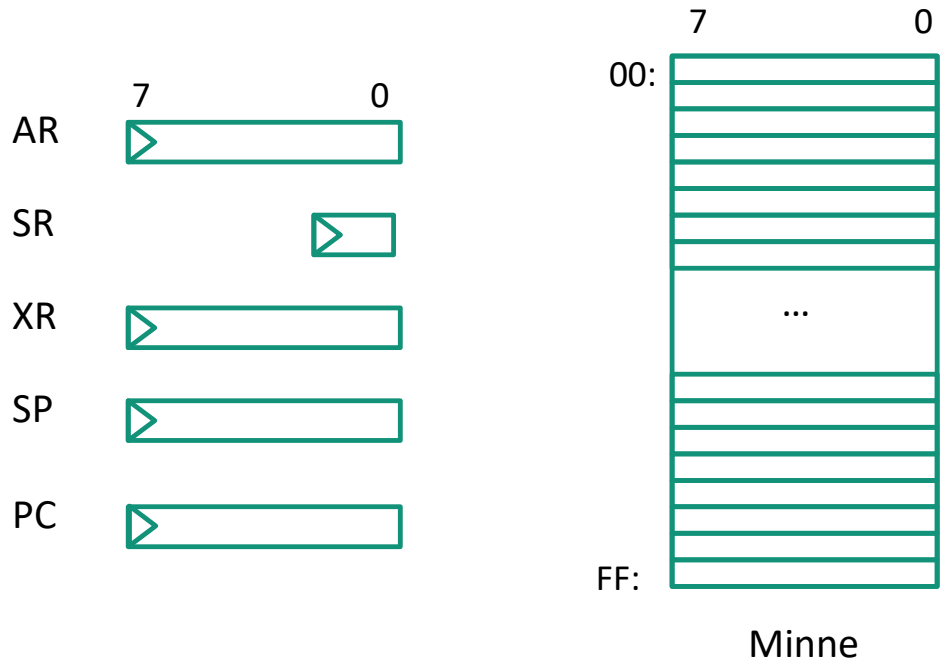


universalräknare



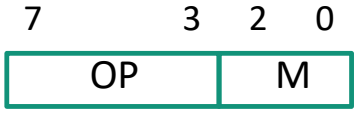
ALU/ackumulator

Ritning 8 - Programmerarmodell



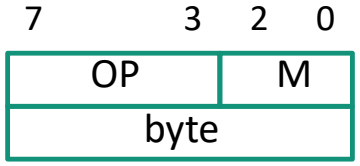
Endast 2 flaggor: Z,N

Ritning 9 - Instruktionsformat



Opkod
32 instruktioner

Märkfält
8 adresseringsmoder



Adress/data

Adresseringsmoder

M	Mod	Exempel	EA
000	Absolut	`LDA` *addr*	*addr*
001	Indirekt	`LDA` (*addr*)	$M(addr)$
010	Indexerad	`LDA` *disp*`,(XR)`	$XR + disp$
011	Relativ	`JMP` *disp*	$PC + 2 + disp$
100	Omedelbar	`LDA #`*n*	$PC + 1$
101	Underförstådd	`INCA/INC`	—

```
M(addr)->AR
M(M(addr))->AR
M(XR+disp)->AR
PC+2+disp->PC
n->AR
```

Exempelvis:

LDA 3

0:	LDA	000
1:	3	
2:		
3:	5	

Absolut
EA=3 (Effektiv Adress)
Operanden = 5

Instruktioner

Instruktion	Verkan	Status			
		N	Z	C	V
LDA *addr*	$AR := M(addr)$	*	*	-	0
STA *addr*	$M(addr) := AR$	-	-	-	0
ADD *addr*	$AR := AR + M(addr)$	*	*	*	*
SUB *addr*	$AR := AR - M(addr)$	*	*	*	*
INCA	$AR := AR + 1$	*	*	*	*
DEC	$AR := AR - 1$	*	*	*	*
CMP *addr*	$AR - M(addr)$	*	*	*	*
CLRA	$AR := 0$	0	1	0	0
ASRA	$AR := AR/2$	*	*	*	-
ASLA	$AR := AR \cdot 2$	*	*	*	*
LSRA	logiskt högerskift av AR	0	*	*	-
AND *addr*	$AR := AR$ and $M(addr)$	*	*	-	0
OR *addr*	$AR := AR$ or $M(addr)$	*	*	-	0
JMP *addr*	$PC := addr$	-	-	-	-
JMPN *addr*	$PC := addr$ om $N = 1$	-	-	-	-
JMPZ *addr*	$PC := addr$ om $Z = 1$	-	-	-	-
JMPC *addr*	$PC := addr$ om $C = 1$	-	-	-	-
JMPV *addr*	$PC := addr$ om $V = 1$	-	-	-	-
IN	$AR := IN$	*	*	-	0
OUT	$UT := AR$	-	-	-	0

Mikromaskinen

”Olle-roos”-datorn

Mikromaskinen

”Olle Roos – datorn”

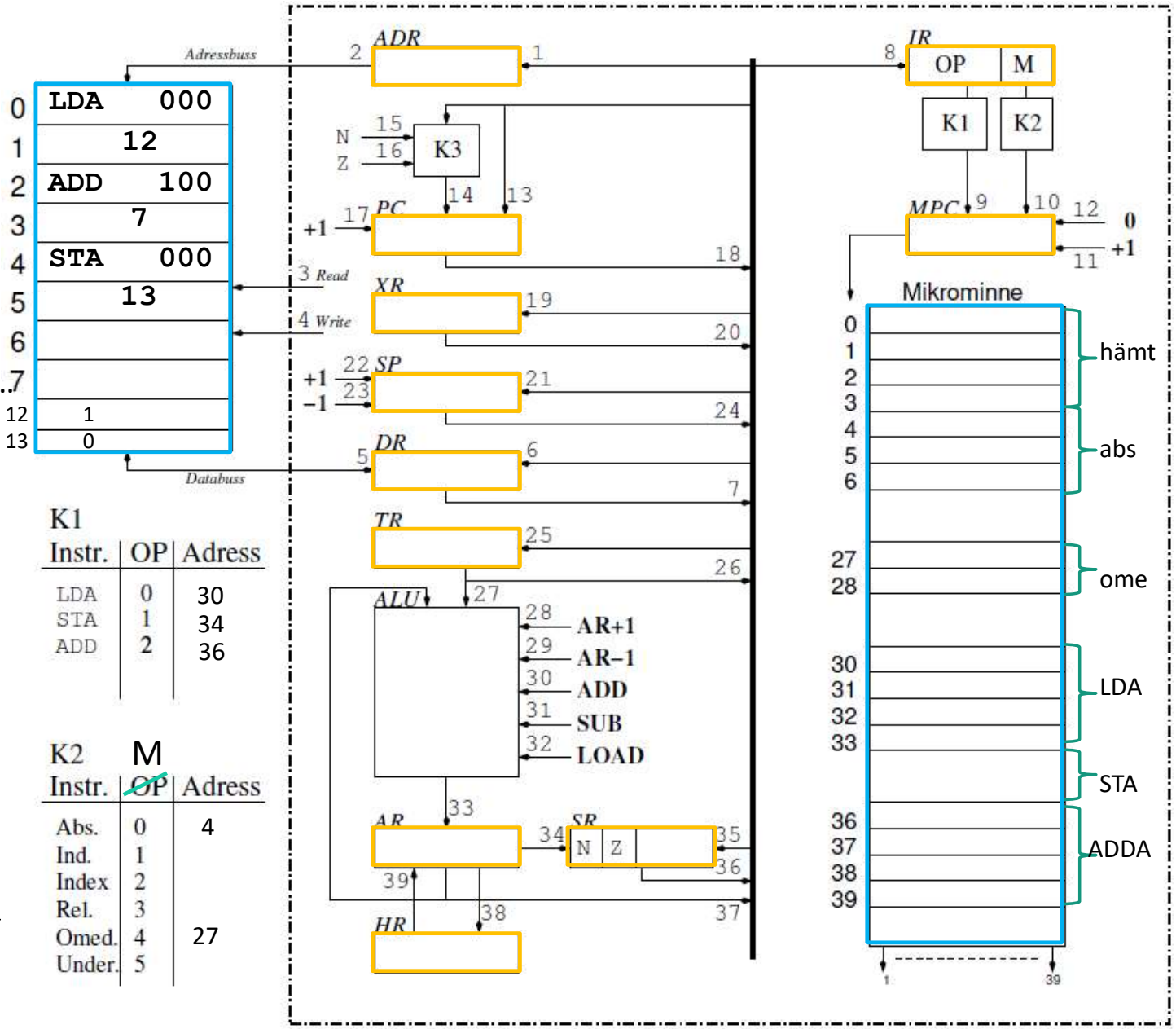
= register

= minne

= kombinatorik

Tabellerna K1, K2 och K3 kan även implementeras som minnen.

RESET



K1

Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 M

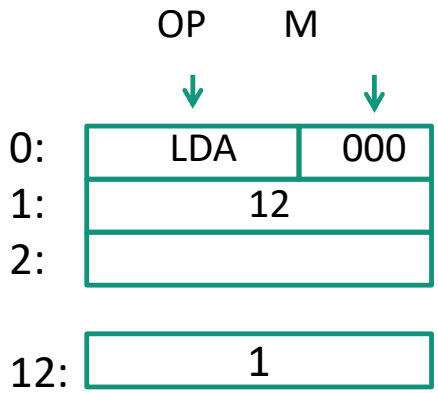
Instr.	~~OP~~	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

Normal arbetsgång - översikt

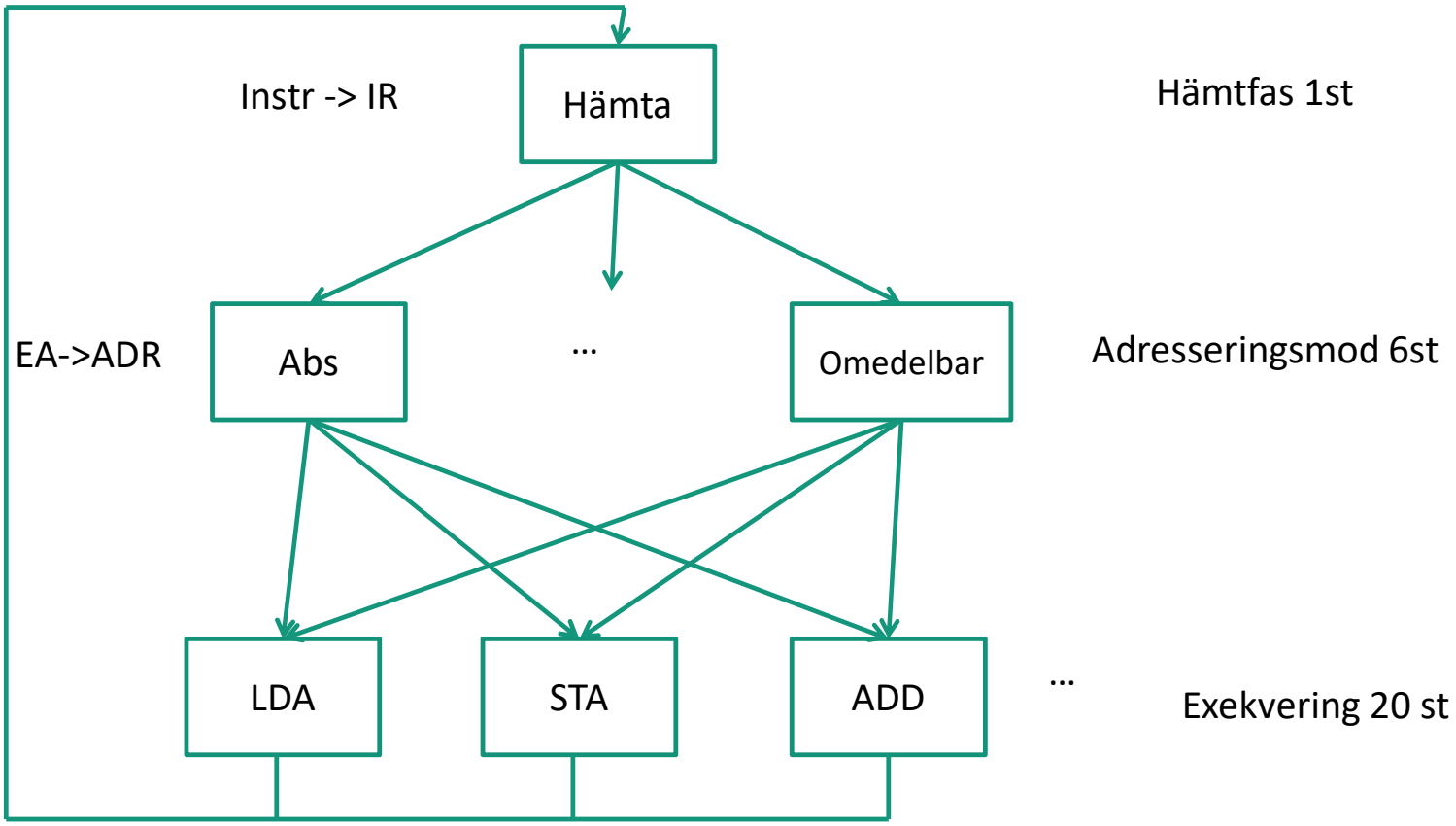
För varje instruktion {

1. **Hämtfas =>** Samma för alla instruktioner
   1. Hämta instruktionen till IR
   2. PC++
   3. Hoppa till rätt ...
2. **Adresseringsmodsfas** Beroende på M sker olika saker
   1. Vanligen: Hämta byten, PC++
   2. EA till ADR
   3. Hoppa till rätt ...
3. **Exekveringsfas =>** Beroende på OP sker olika saker
   1. Vanligen: Hämta operanden
   2. Resultatet till AR och uppdatera SR
   3. Hoppa till Hämtfas

}



Organisation av mikroprogram

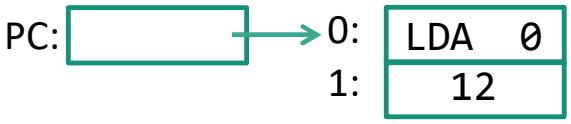


Mikrokod

-Ett mindre program
-Ytterligare exempel för olika instruktioner och adresseringsmoder

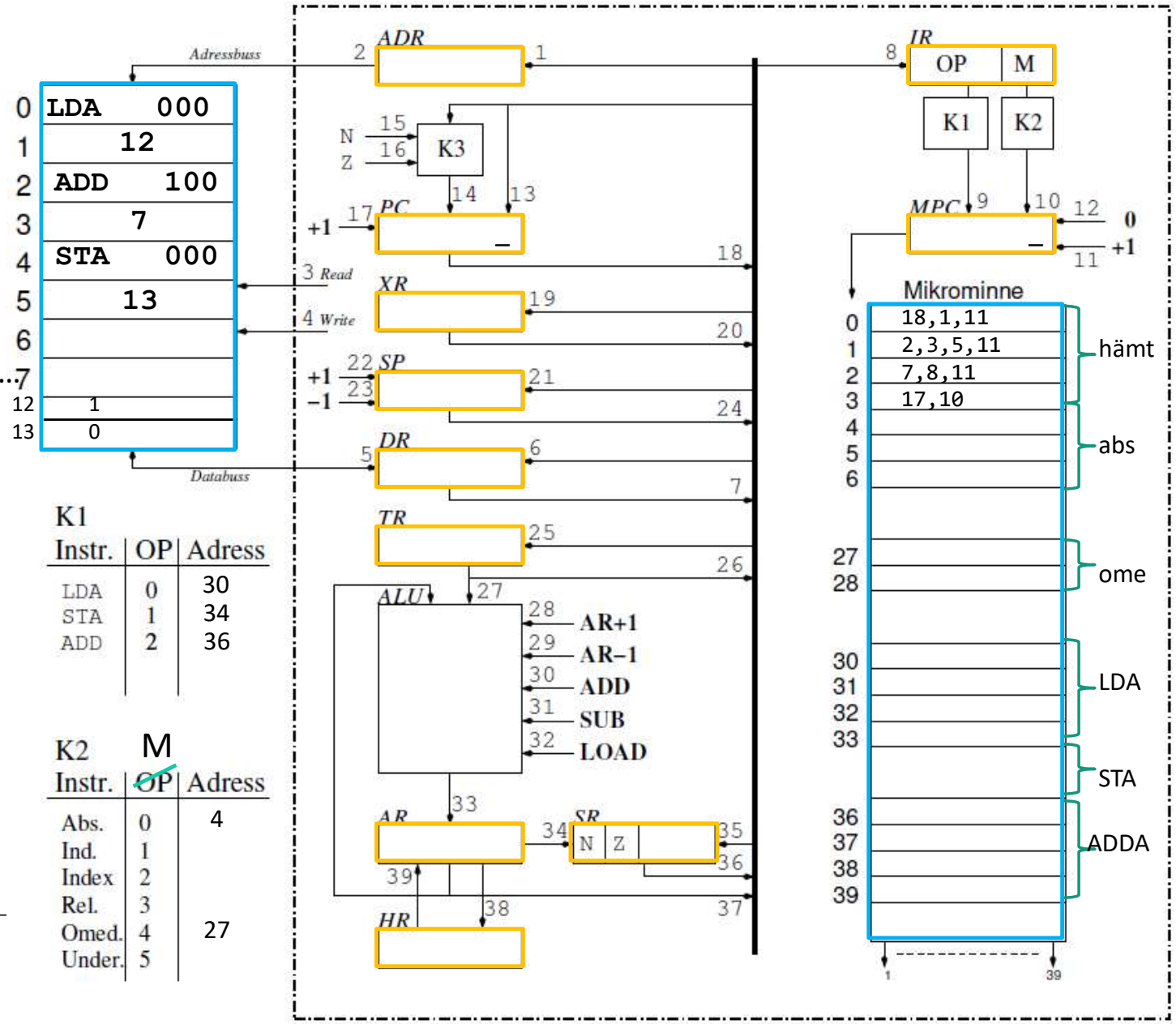
Steg 1 : Hämtfas

M(PC) -> IR




```
0: pc->adr,mpc++                18,1,11
1: adr->minne,data->dr,mpc++    2,3,5,11
2: dr->ir,mpc++                 7,8,11
3: PC++,K2->mpc                 17,10
```

Vid reset nollställs PC och MPC



K1

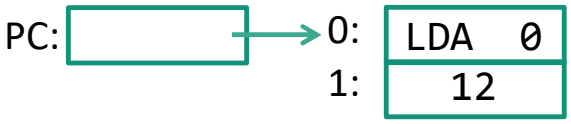
Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 M

Instr.	~~OP~~	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

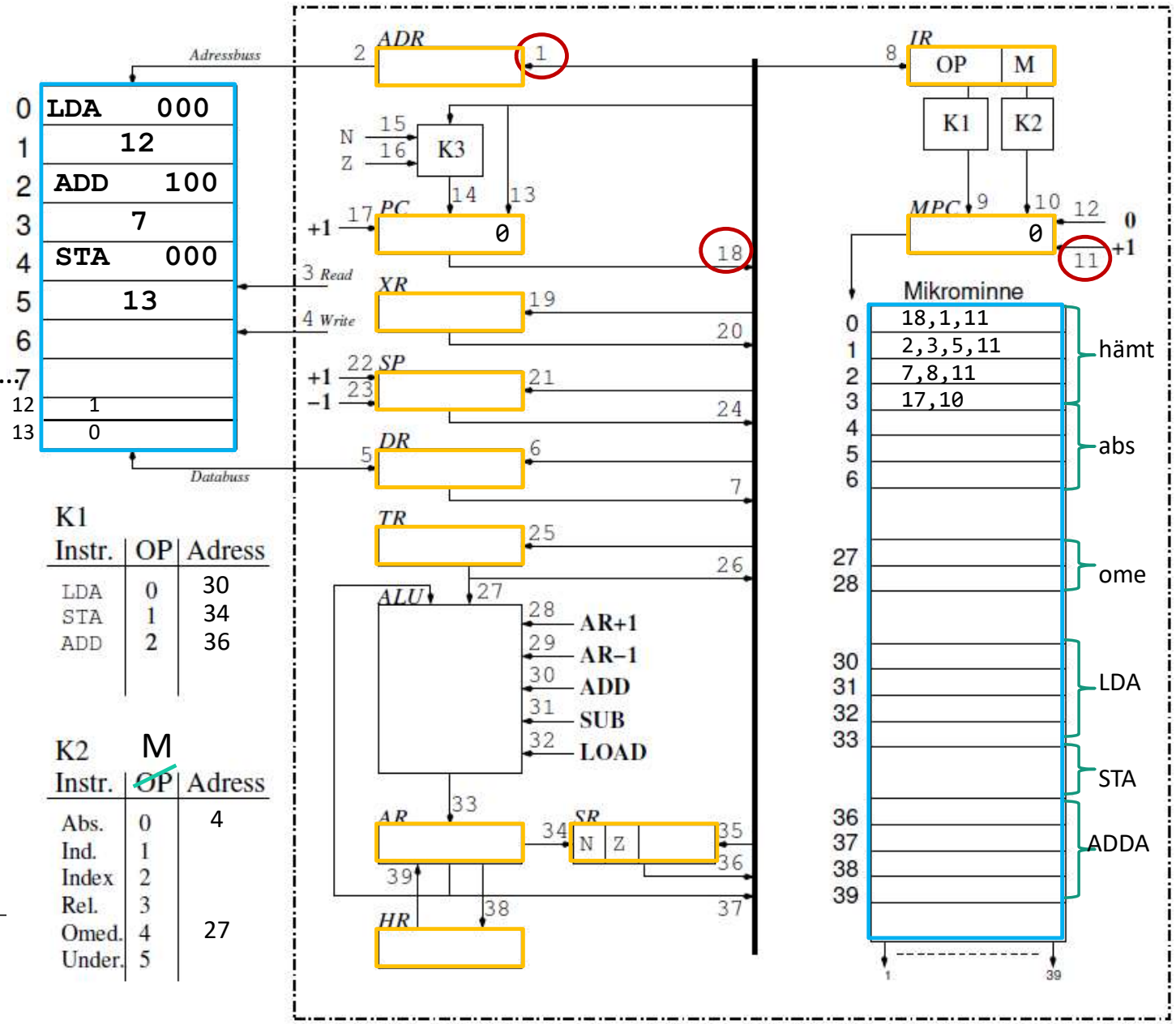
Steg 1 : Hämtfas

M(PC) -> IR




```
0: pc->adr,mpc++                18,1,11
1: adr->minne,data->dr,mpc++    2,3,5,11
2: dr->ir,mpc++                 7,8,11
3: PC++,K2->mpc                 17,10
```

Rad `0` i Mikrominnet adresseras.
Effekterna av aktiverade styrsignaler sker vid nästkommande klockflank.



K1

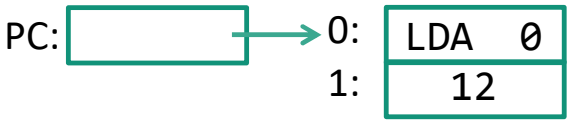
Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 M

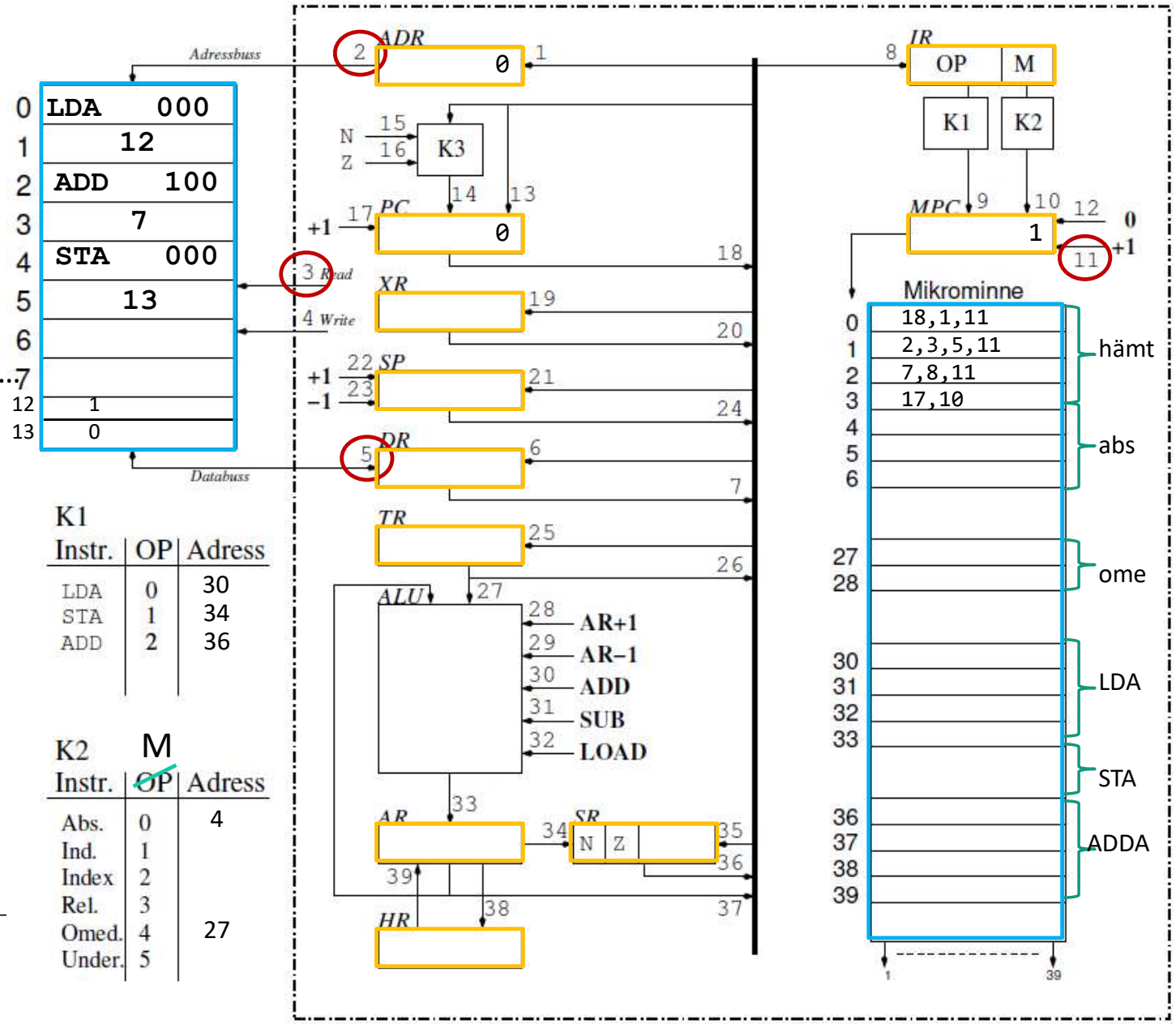
Instr.	~~OP~~	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

Steg 1 : Hämtfas

M(PC) -> IR




```
0: pc->adr,mpc++                18,1,11
1: adr->minne,data->dr,mpc++    2,3,5,11
2: dr->ir,mpc++                 7,8,11
3: PC++,K2->mpc                 17,10
```



K1

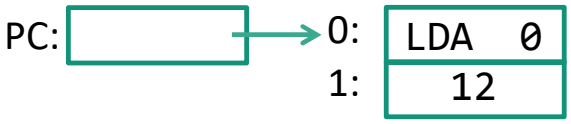
Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 M

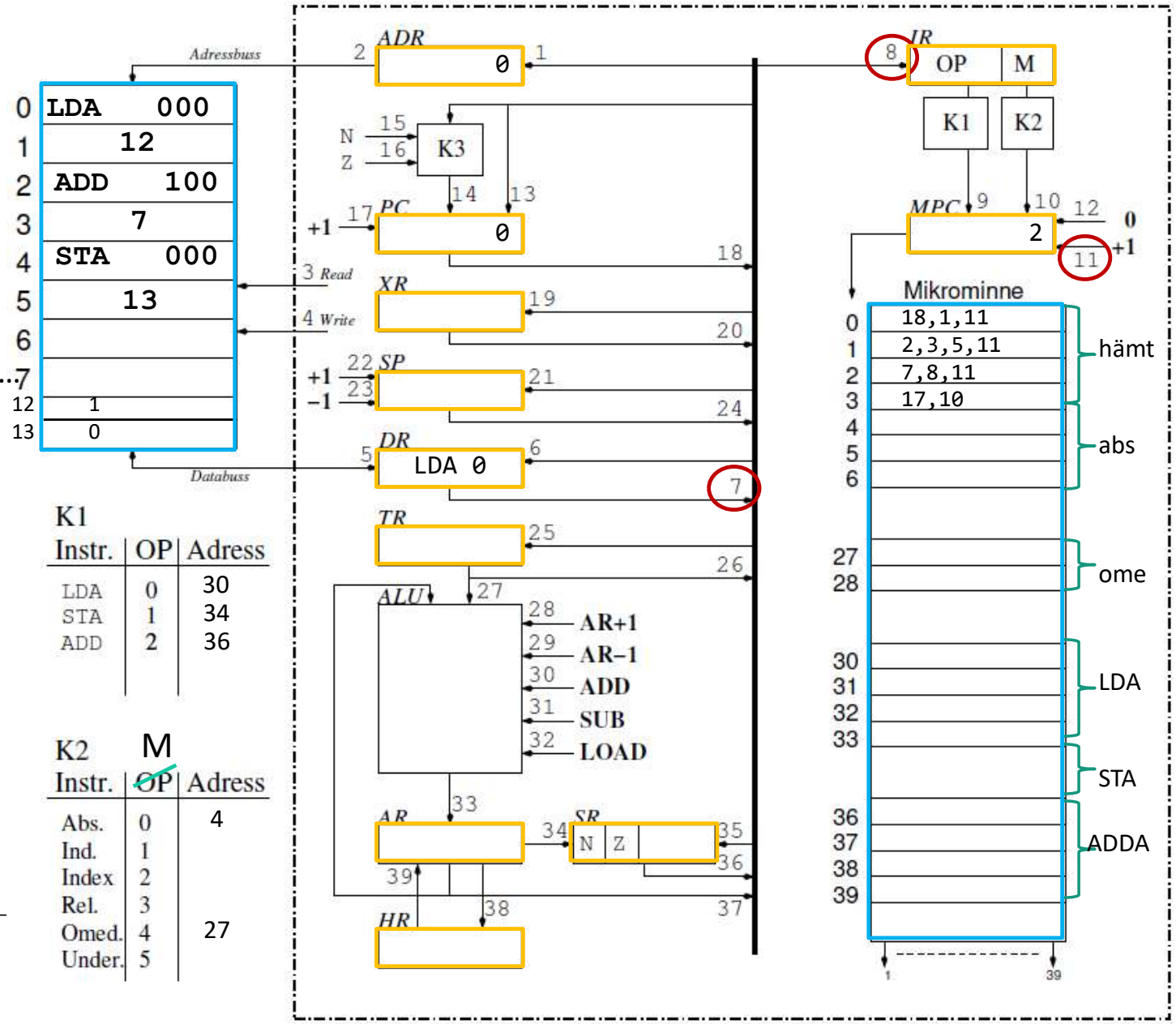
Instr.	~~OP~~	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

Steg 1 : Hämtfas

M(PC) -> IR




```
0: pc->adr,mpc++                18,1,11
1: adr->minne,data->dr,mpc++    2,3,5,11
2: dr->ir,mpc++                 7,8,11
3: PC++,K2->mpc                 17,10
```



K1

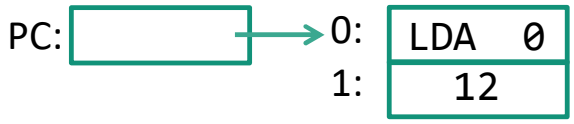
Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 M

Instr.	~~OP~~	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

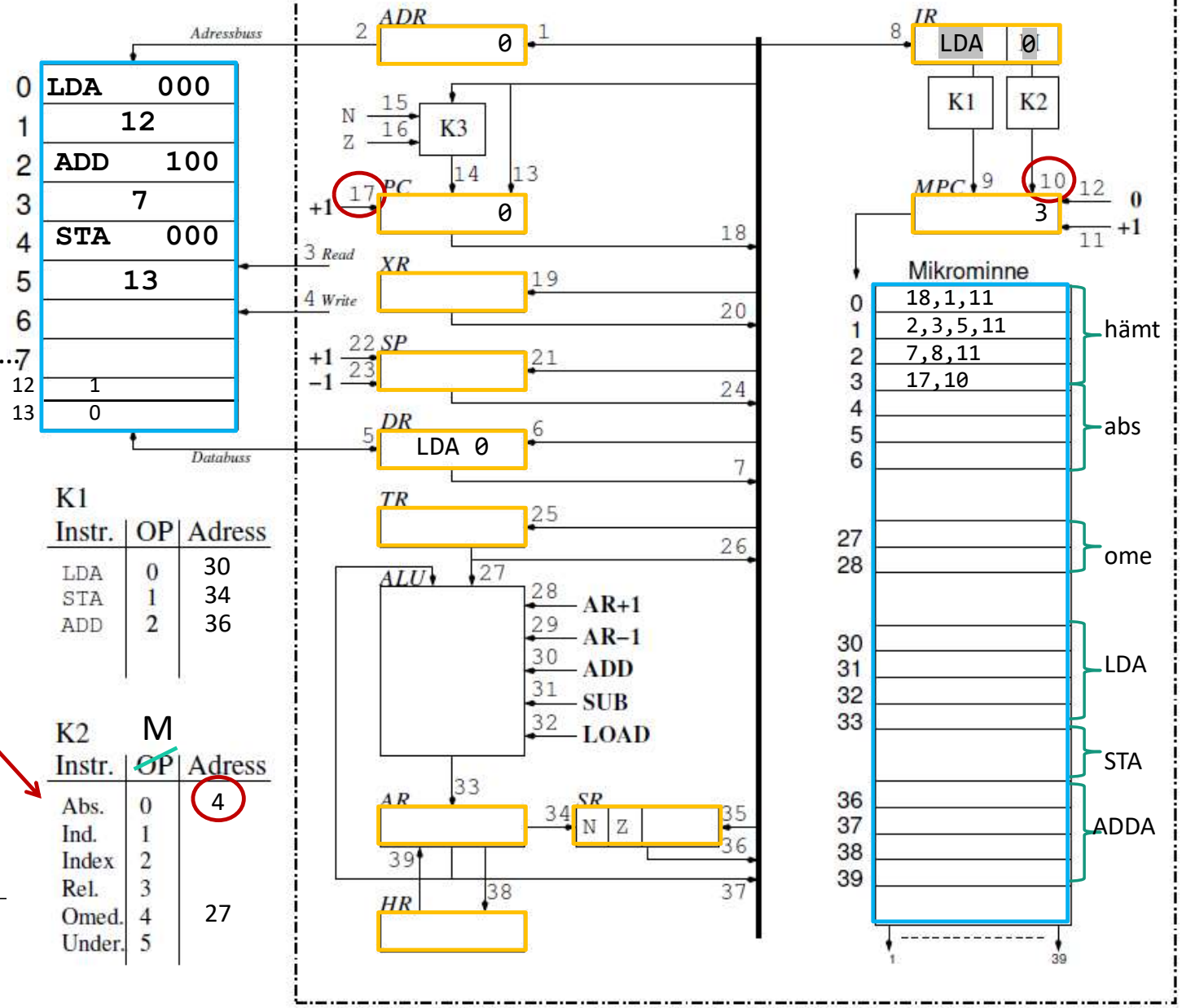
Steg 1 : Hämtfas

M(PC) -> IR




```
0: pc->adr,mpc++                18,1,11
1: adr->minne,data->dr,mpc++    2,3,5,11
2: dr->ir,mpc++                 7,8,11
3: PC++,K2->mpc                 17,10
```

Signal 10 aktiverar hopp till adresseringsmodsfas (Abs)



K1

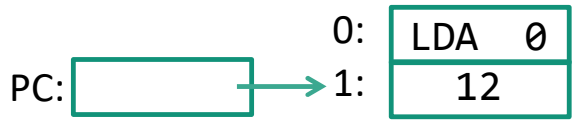
Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 M

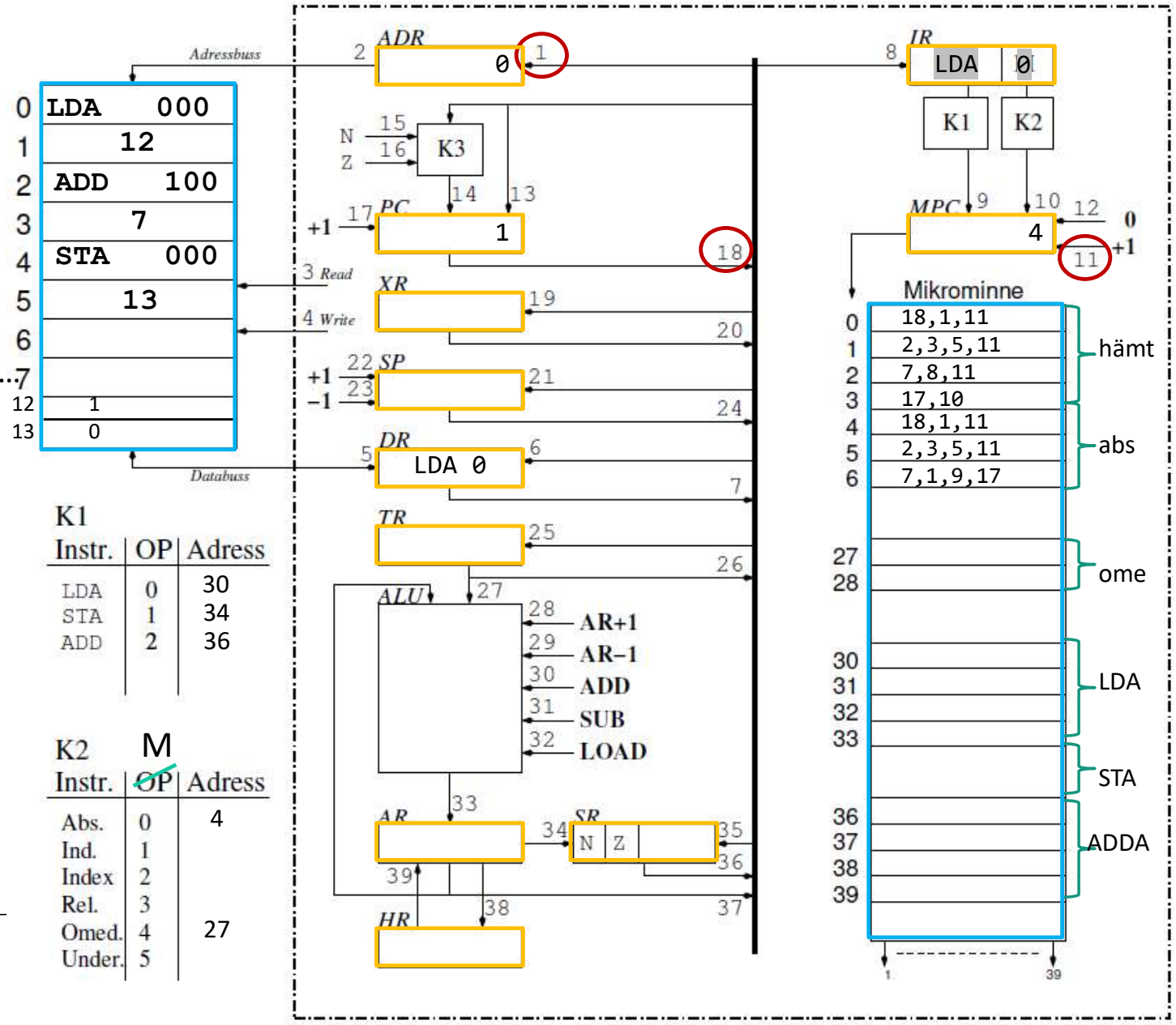
Instr.	~~OP~~	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

Steg 2 : A-modfas (Abs)

M(PC) -> ADR




```
4: pc->adr,mpc++                18,1,11
5: adr->minne,data->dr,mpc++    2,3,5,11
6: dr->adr,K1->mpc,PC++         7,1,9,17
```



K1

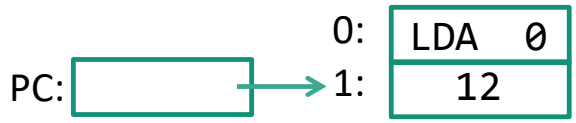
Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 M

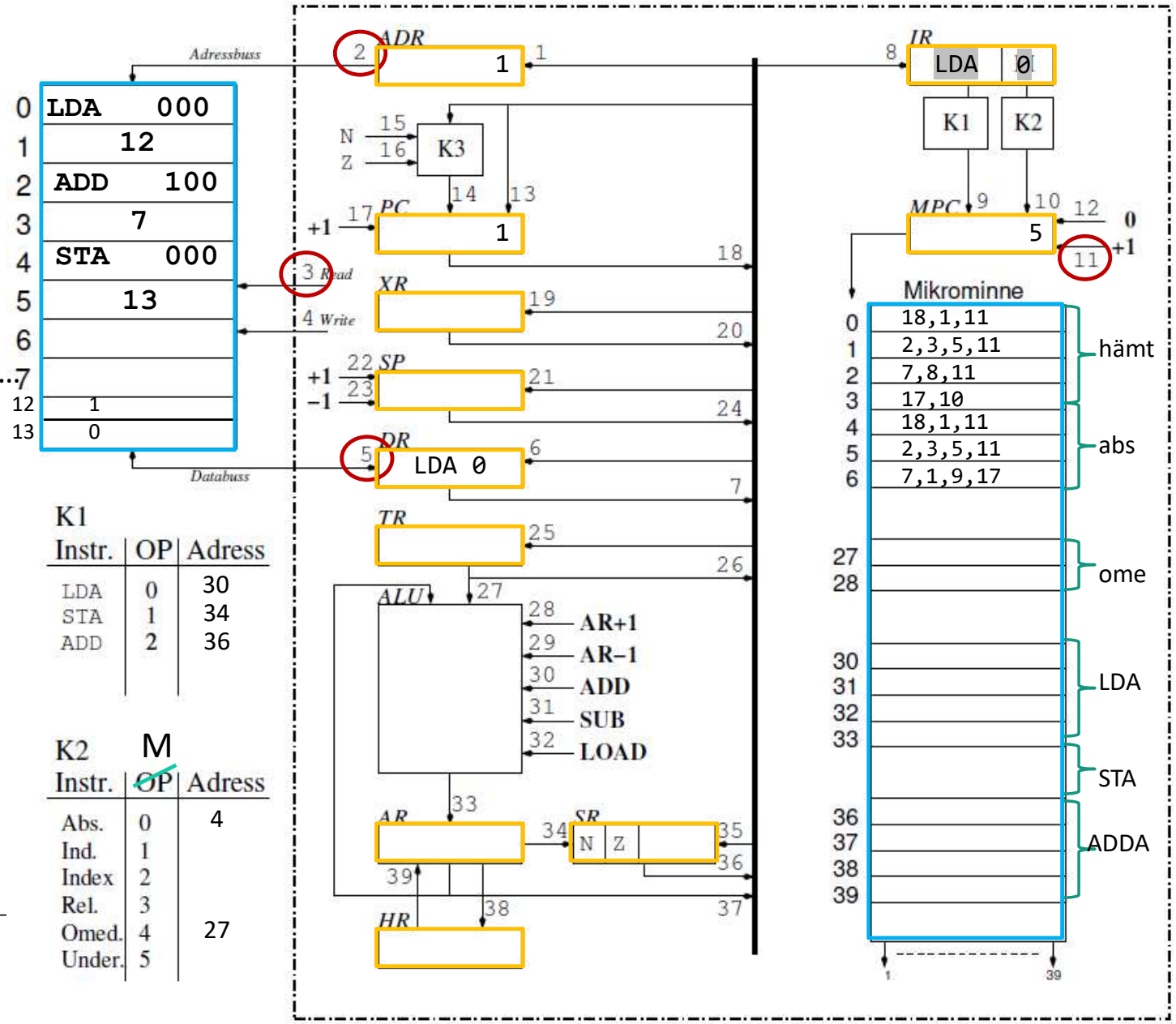
Instr.	~~OP~~	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

Steg 2 : A-modfas (Abs)

M(PC) -> ADR




```
4: pc->adr,mpc++                  18,1,11
5: adr->minne,data->dr,mpc++      2,3,5,11
6: dr->adr,K1->mpc,PC++           7,1,9,17
```



K1

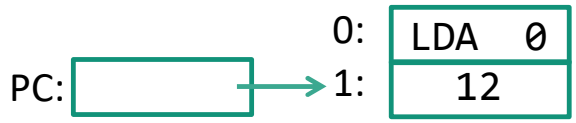
Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 M

Instr.	~~OP~~	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

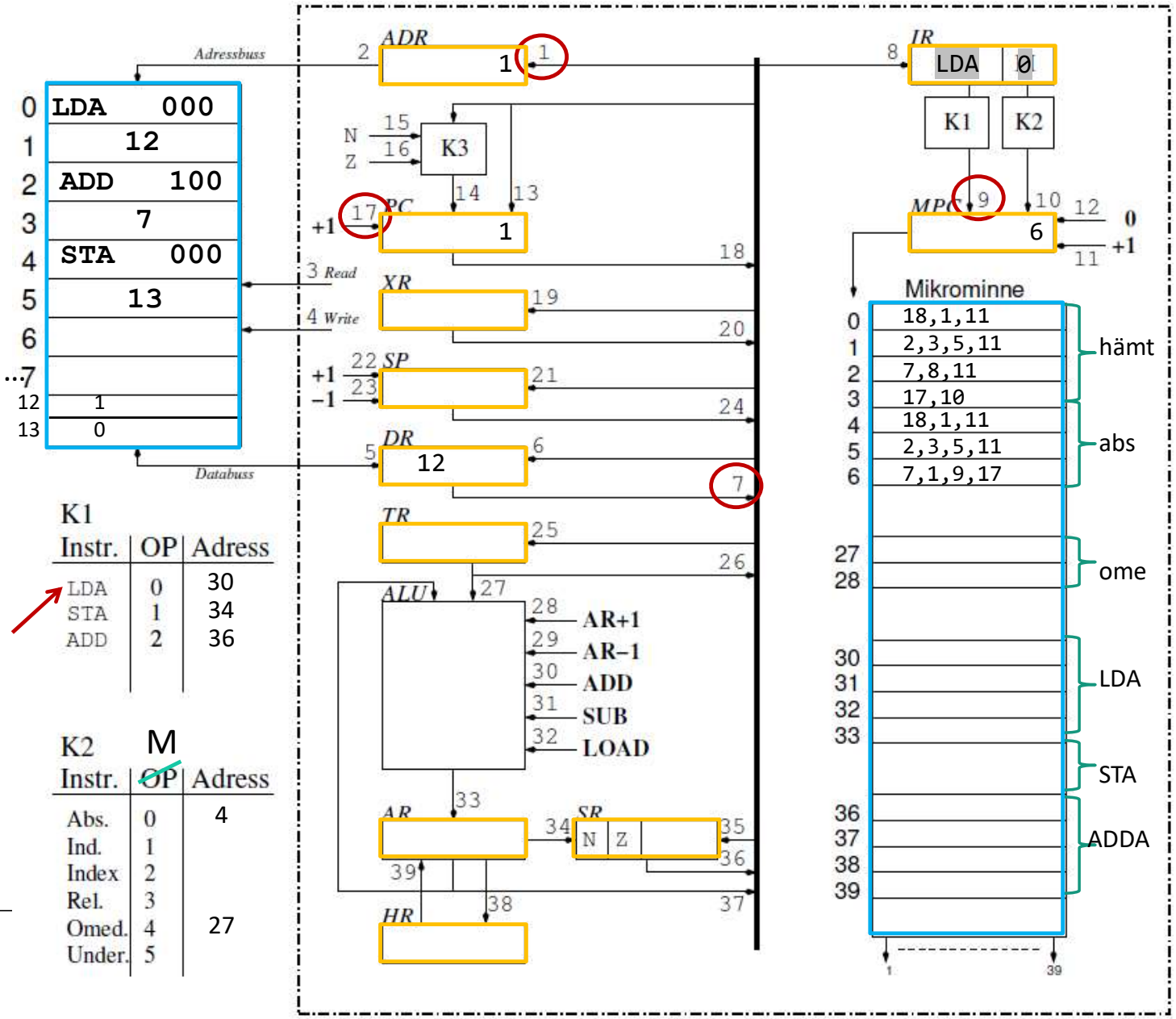
Steg 2 : A-modfas (Abs)

M(PC) -> ADR




```
4: pc->adr,mpc++                18,1,11
5: adr->minne,data->dr,mpc++    2,3,5,11
6: dr->adr,K1->mpc,PC++         7,1,9,17
```

Signal 9 aktiverar hopp till exekveringsfas (LDA)



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	

Steg 3 : Exe-fas (LDA)

AR = M(ADR)

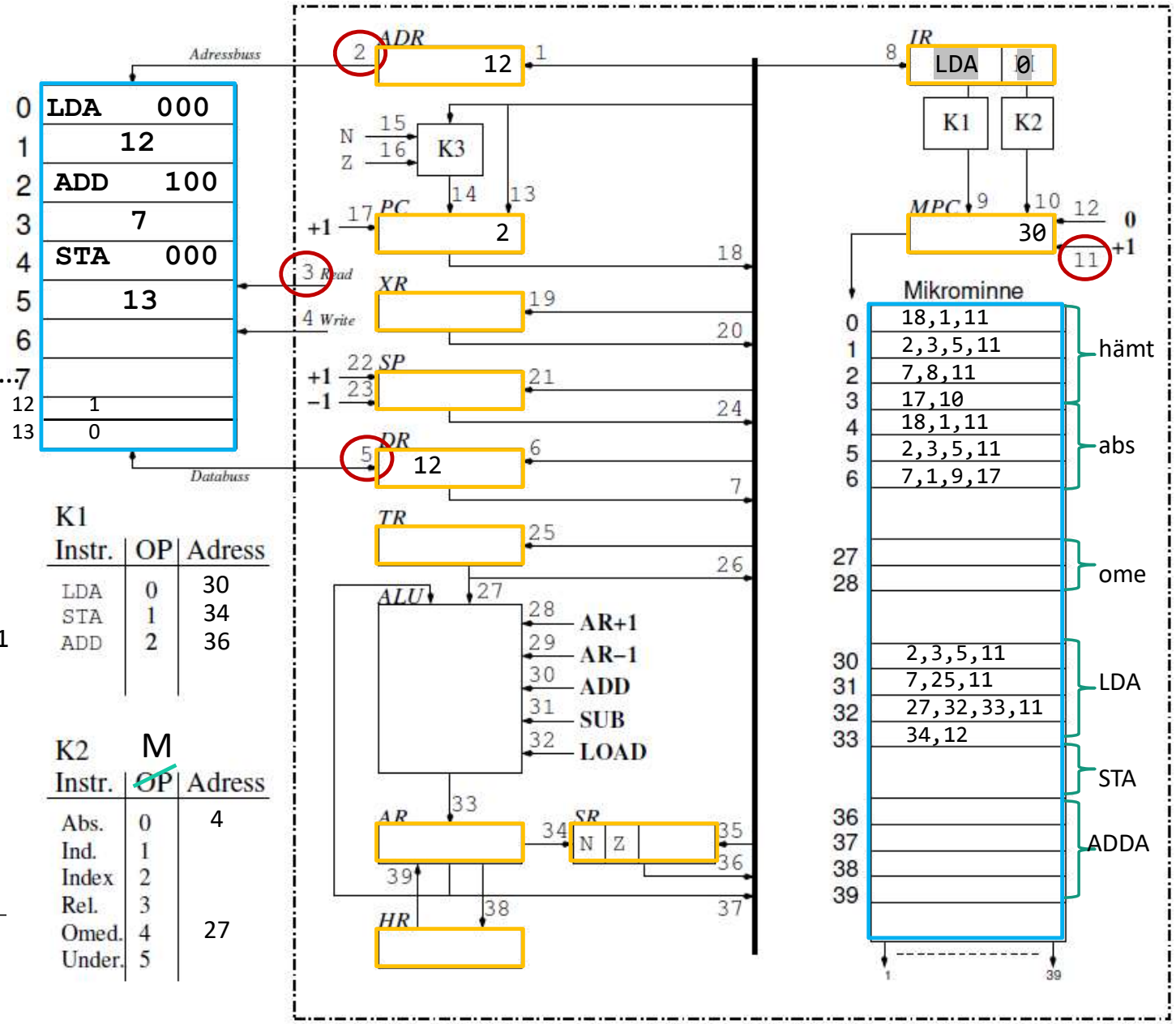
AR:

ADR: 12

12: 1

13: 0

```
30: adr->minne,data->dr,mpc++ 2,3,5,11
31: dr->tr,mpc++              7,25,11
32: tr->ar,mpc++              27,32,33,11
33: status, 0->mpc            34,12
```



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	

Steg 3 : Exe-fas (LDA)

AR = M(ADR)

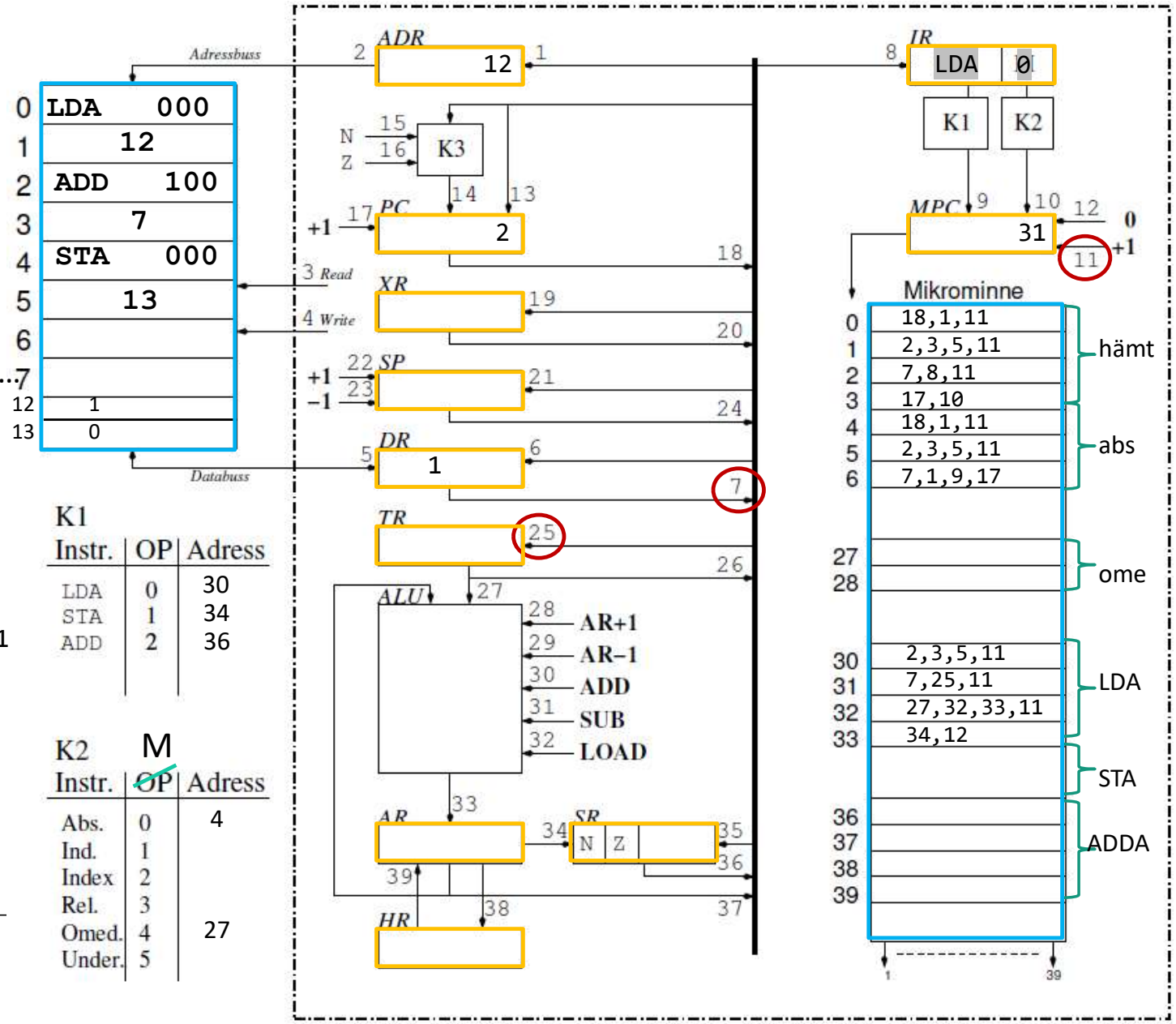
AR:

ADR: 12

12: 1

13: 0

```
30: adr->minne,data->dr,mpc++ 2,3,5,11
31: dr->tr,mpc++              7,25,11
32: tr->ar,mpc++              27,32,33,11
33: status, 0->mpc            34,12
```



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	

Steg 3 : Exe-fas (LDA)

AR = M(ADR)

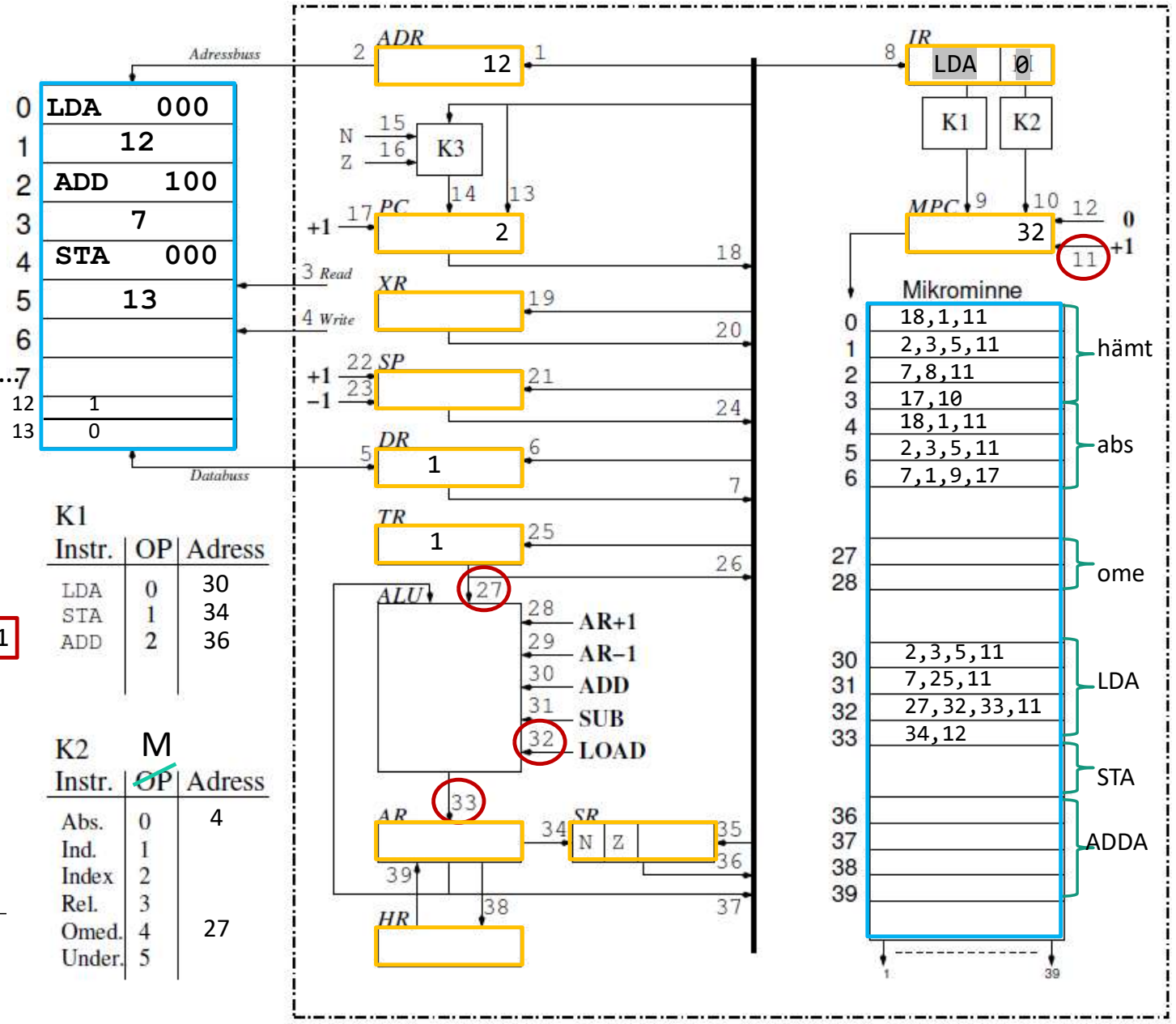
AR:

ADR: 12

12: 1

13: 0

```
30: adr->minne,data->dr,mpc++ 2,3,5,11
31: dr->tr,mpc++              7,25,11
32: tr->ar,mpc++              27,32,33,11
33: status, 0->mpc            34,12
```



K1

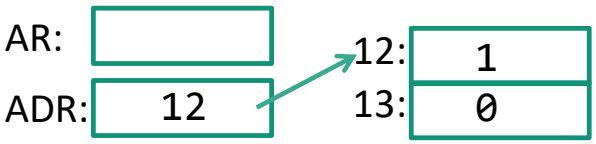
Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 M

Instr.	~~OP~~	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

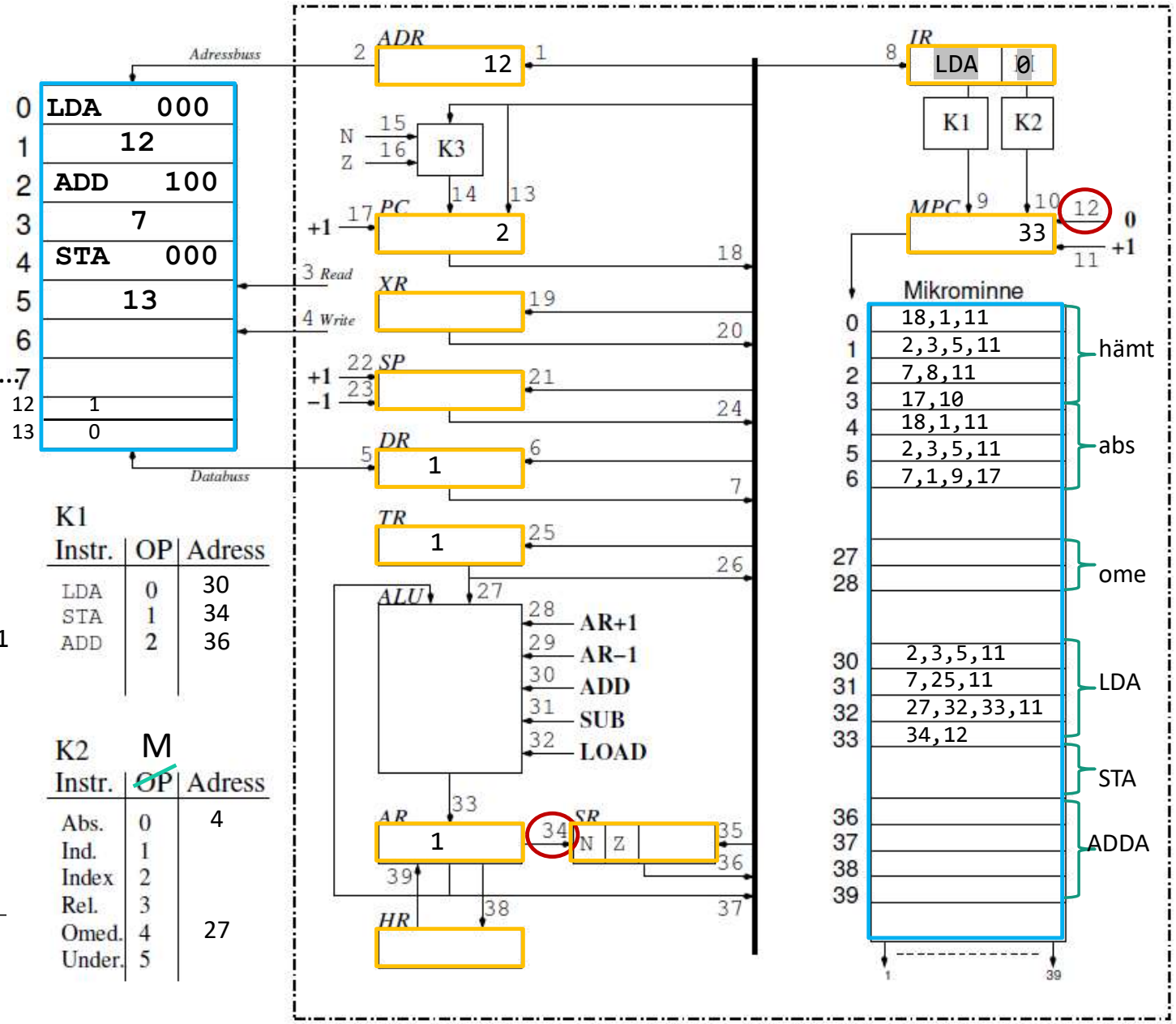
Steg 3 : Exe-fas (LDA)

AR = M(ADR)




```
30: adr->minne,data->dr,mpc++ 2,3,5,11
31: dr->tr,mpc++              7,25,11
32: tr->ar,mpc++              27,32,33,11
33: status, 0->mpc            34,12
```

Signal 12 aktiverar nollställning av MPC, för nästa hämtfas



K1

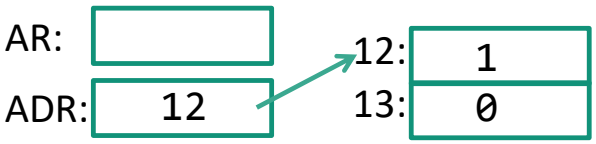
Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 M

Instr.	~~OP~~	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

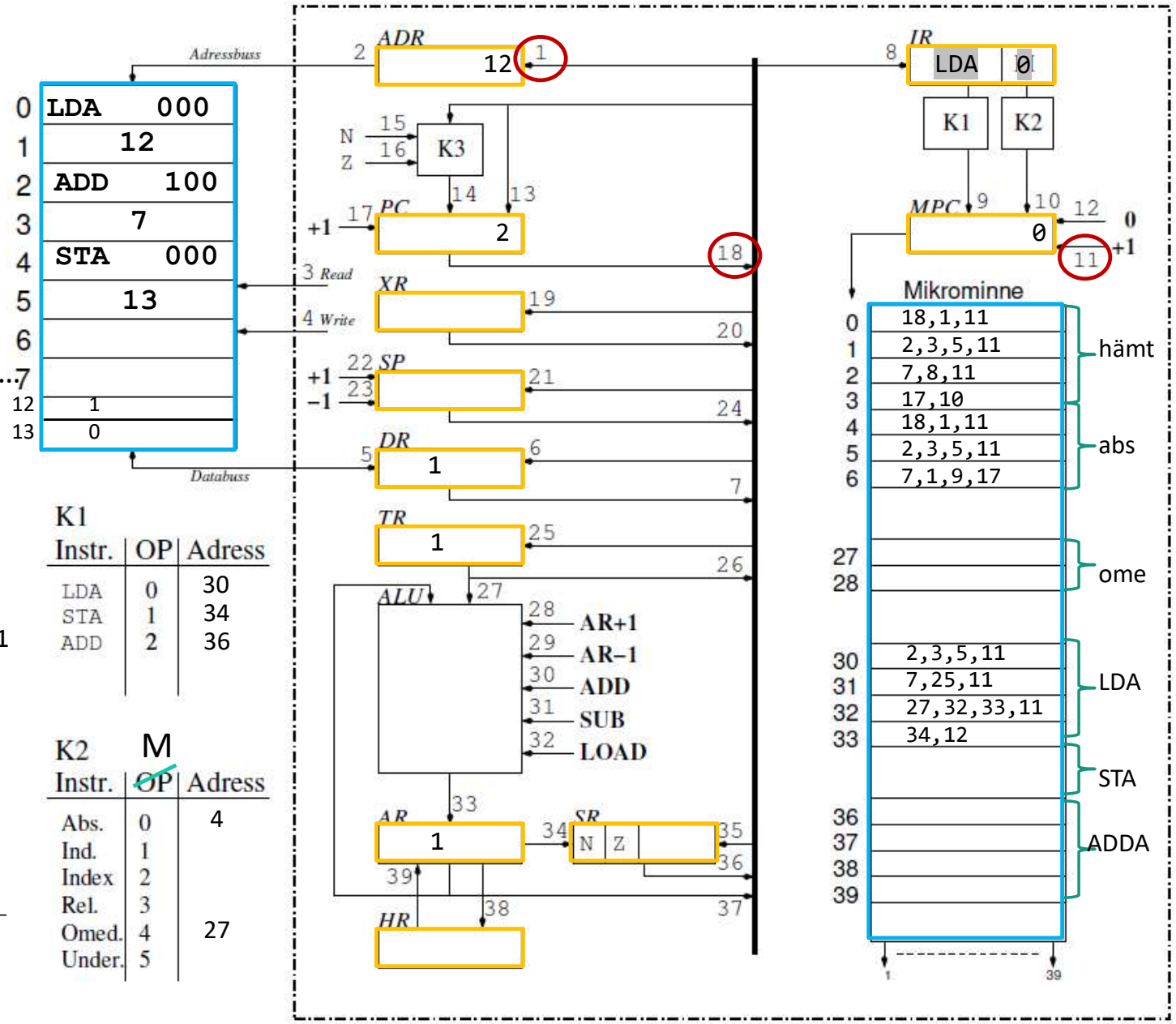
Steg 3 : Exe-fas (LDA)

AR = M(ADR)




```
30: adr->minne,data->dr,mpc++ 2,3,5,11
31: dr->tr,mpc++              7,25,11
32: tr->ar,mpc++              27,32,33,11
33: status, 0->mpc            34,12
```

Signal 12 aktiverar
nollställning av MPC,
för nästa hämtfas, så där!



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	

Mikrokod för ADD #7

Mikrokod för ADD #7

Steg 1 : H-fas, som förut

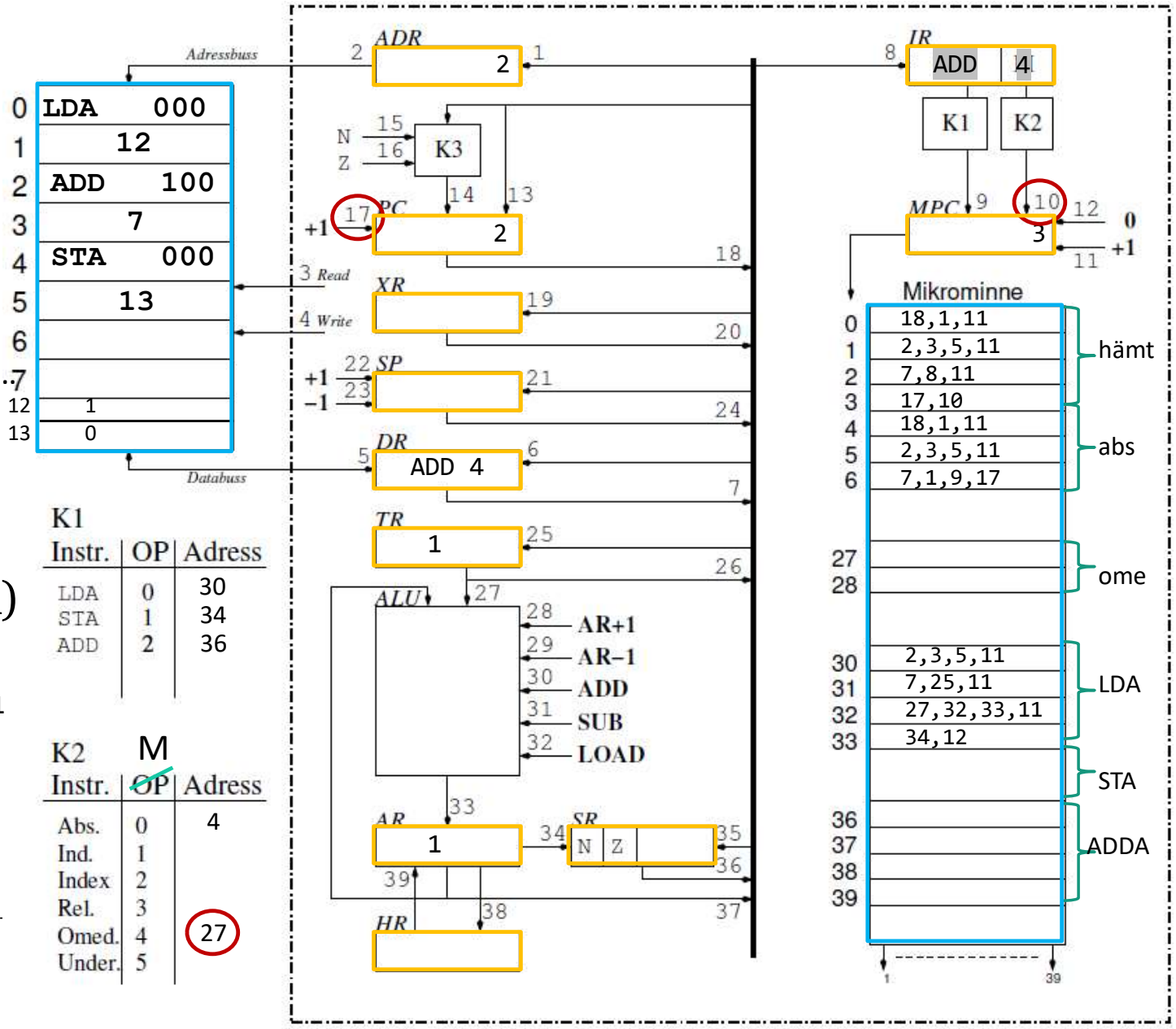
```
...
3: PC++,K2->mpc          17,10
```

Steg 2 : A-fas, Omedelbar

```
27: PC->adr,mpc++        18,1,11
28: PC++,K1->mpc         17,9
```

Steg 3 : Exe, AR=AR+M(ADR)

```
36: data->dr,mpc++       2,3,5,11
37: dr->tr,mpc++         7,25,11
38: ar+tr->ar, mpc++     27,33,30,11
39: status, 0->mpc       34,12
```



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	

Mikrokod för ADD #7

Steg 1 : H-fas, som förut

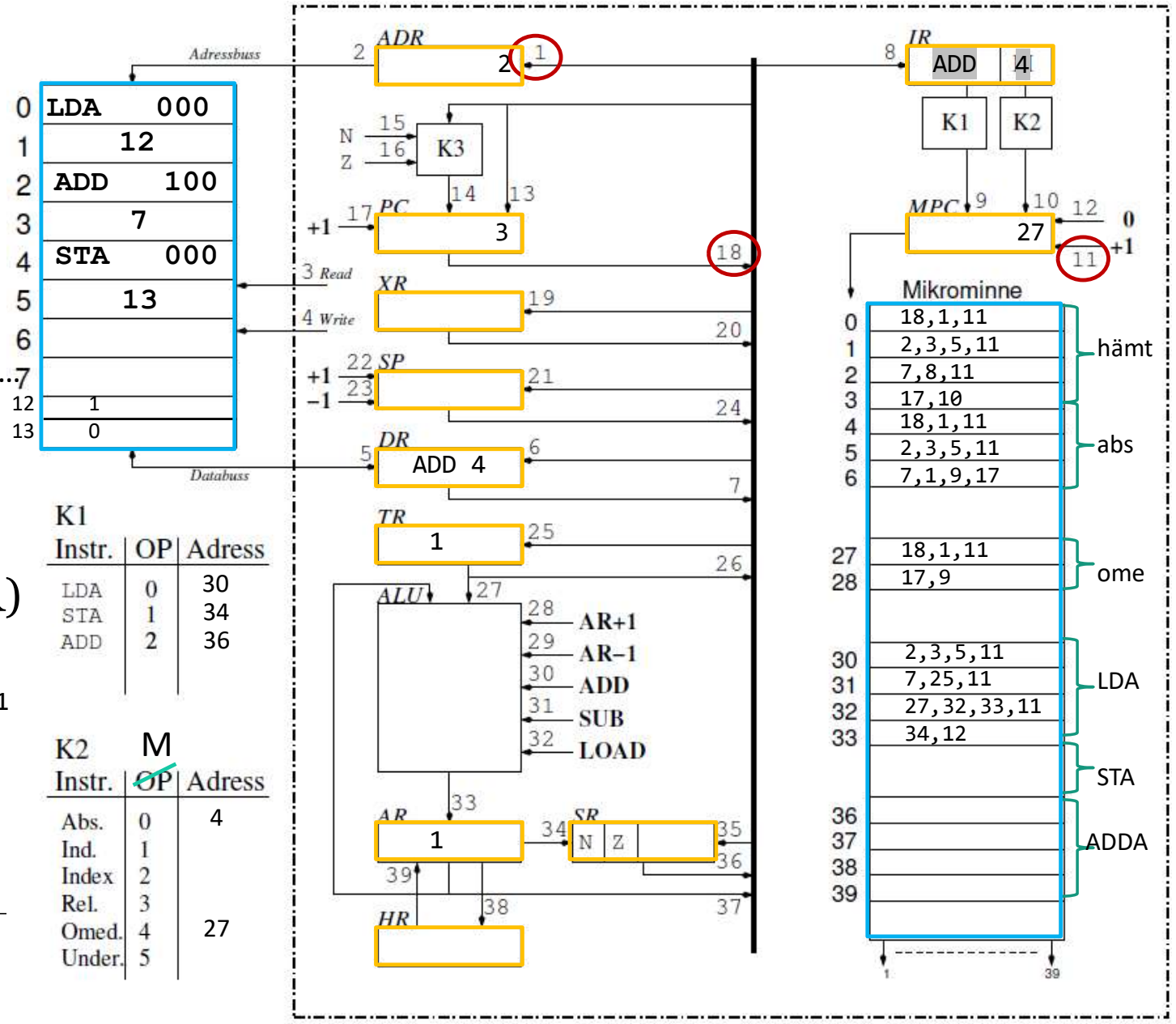
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Omedelbar

```
27: PC->adr,mpc++                  18,1,11
28: PC++,K1->mpc                   17,9
```

Steg 3 : Exe, AR=AR+M(ADR)

```
36: data->dr,mpc++                 2,3,5,11
37: dr->tr,mpc++                   7,25,11
38: ar+tr->ar, mpc++               27,33,30,11
39: status, 0->mpc                 34,12
```



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	

Mikrokod för ADD #7

Steg 1 : H-fas, som förut

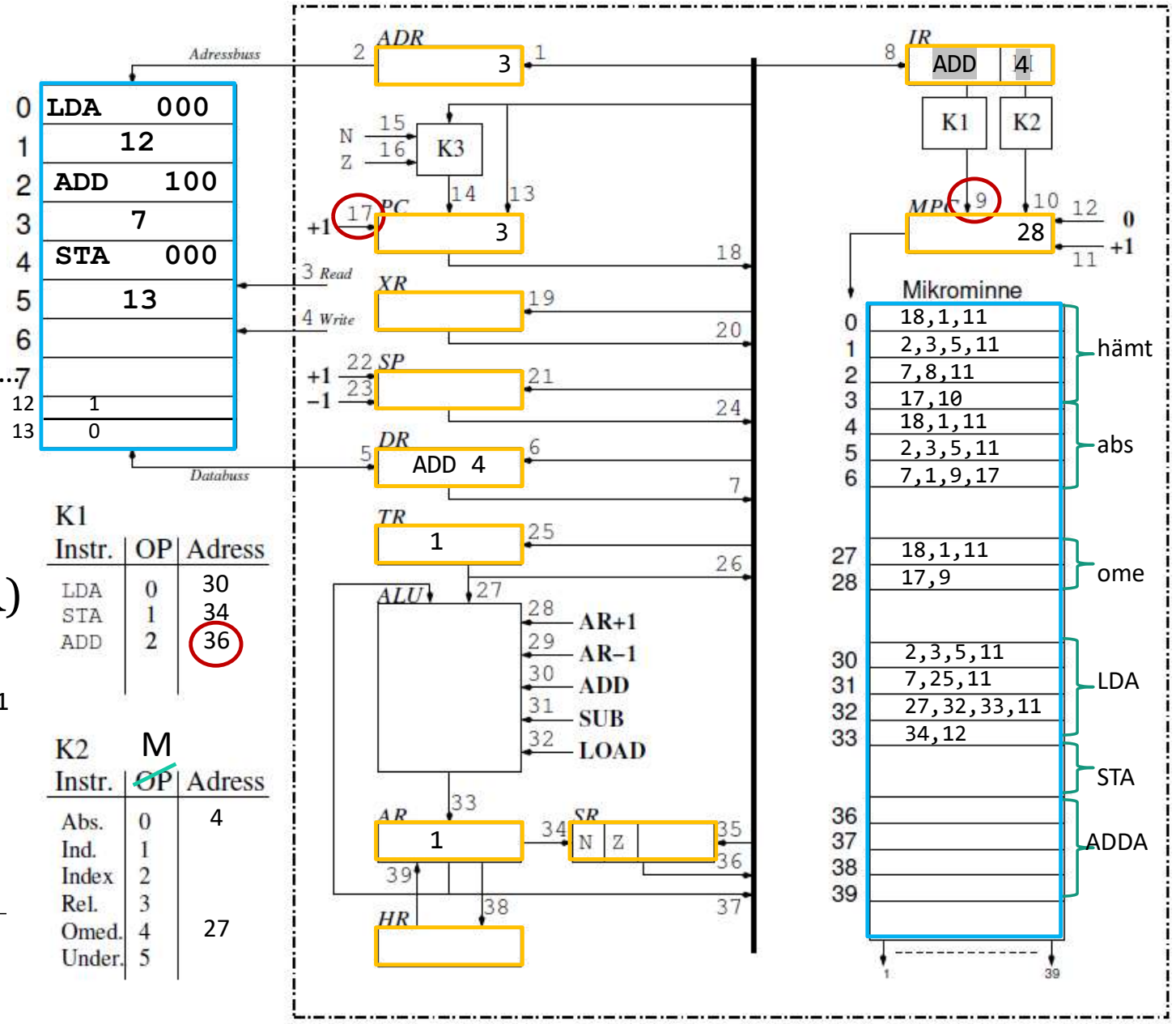
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Omedelbar

```
27: PC->adr,mpc++                  18,1,11
28: PC++,K1->mpc                   17,9
```

Steg 3 : Exe, AR=AR+M(ADR)

```
36: data->dr,mpc++                 2,3,5,11
37: dr->tr,mpc++                   7,25,11
38: ar+tr->ar, mpc++               27,33,30,11
39: status, 0->mpc                 34,12
```



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	

Mikrokod för ADD #7

Steg 1 : H-fas, som förut

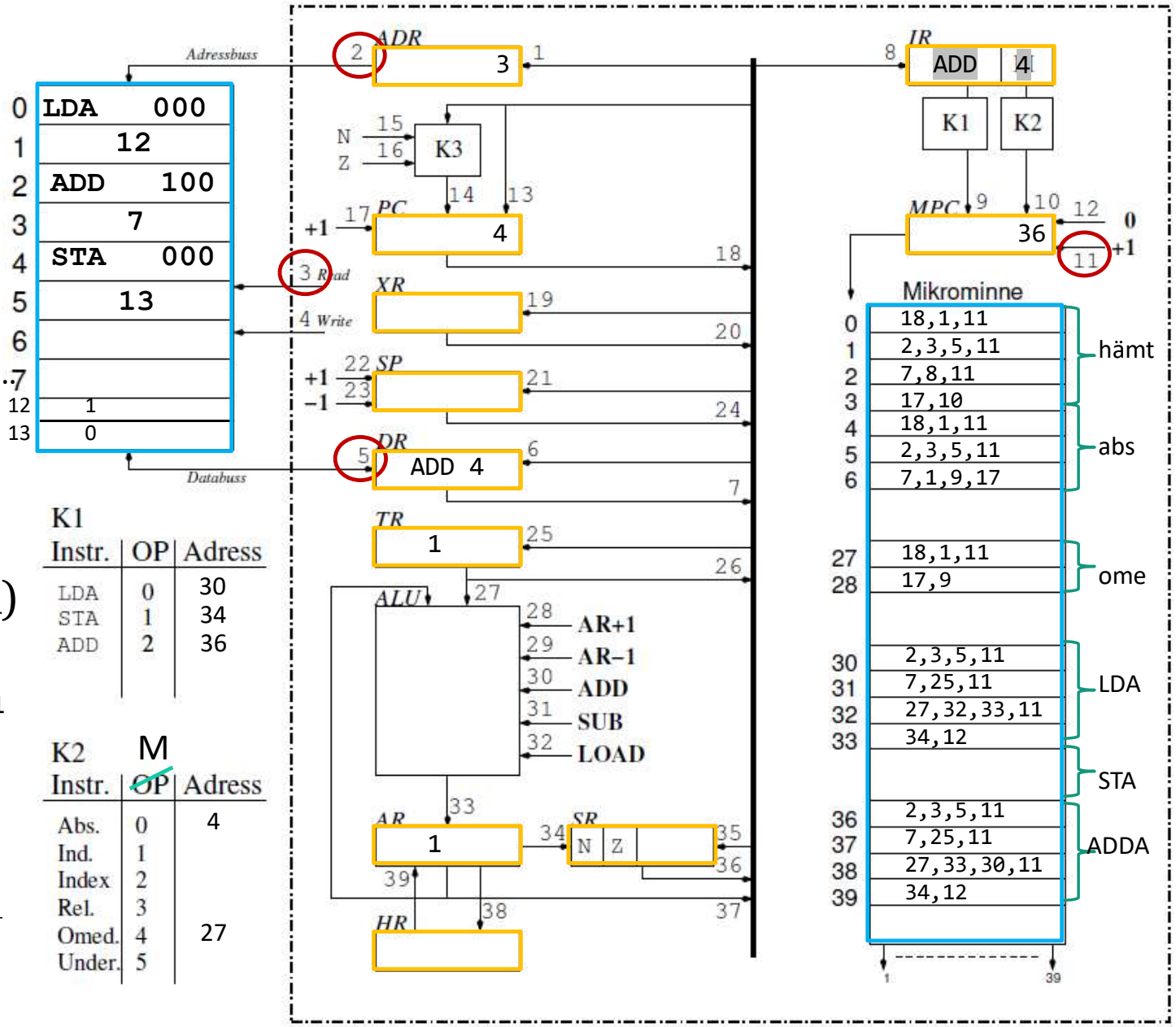
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Omedelbar

```
27: PC->adr,mpc++                  18,1,11
28: PC++,K1->mpc                   17,9
```

Steg 3 : Exe, AR=AR+M(ADR)

```
36: data->dr,mpc++                 2,3,5,11
37: dr->tr,mpc++                   7,25,11
38: ar+tr->ar, mpc++               27,33,30,11
39: status, 0->mpc                 34,12
```



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	

Mikrokod för ADD #7

Steg 1 : H-fas, som förut

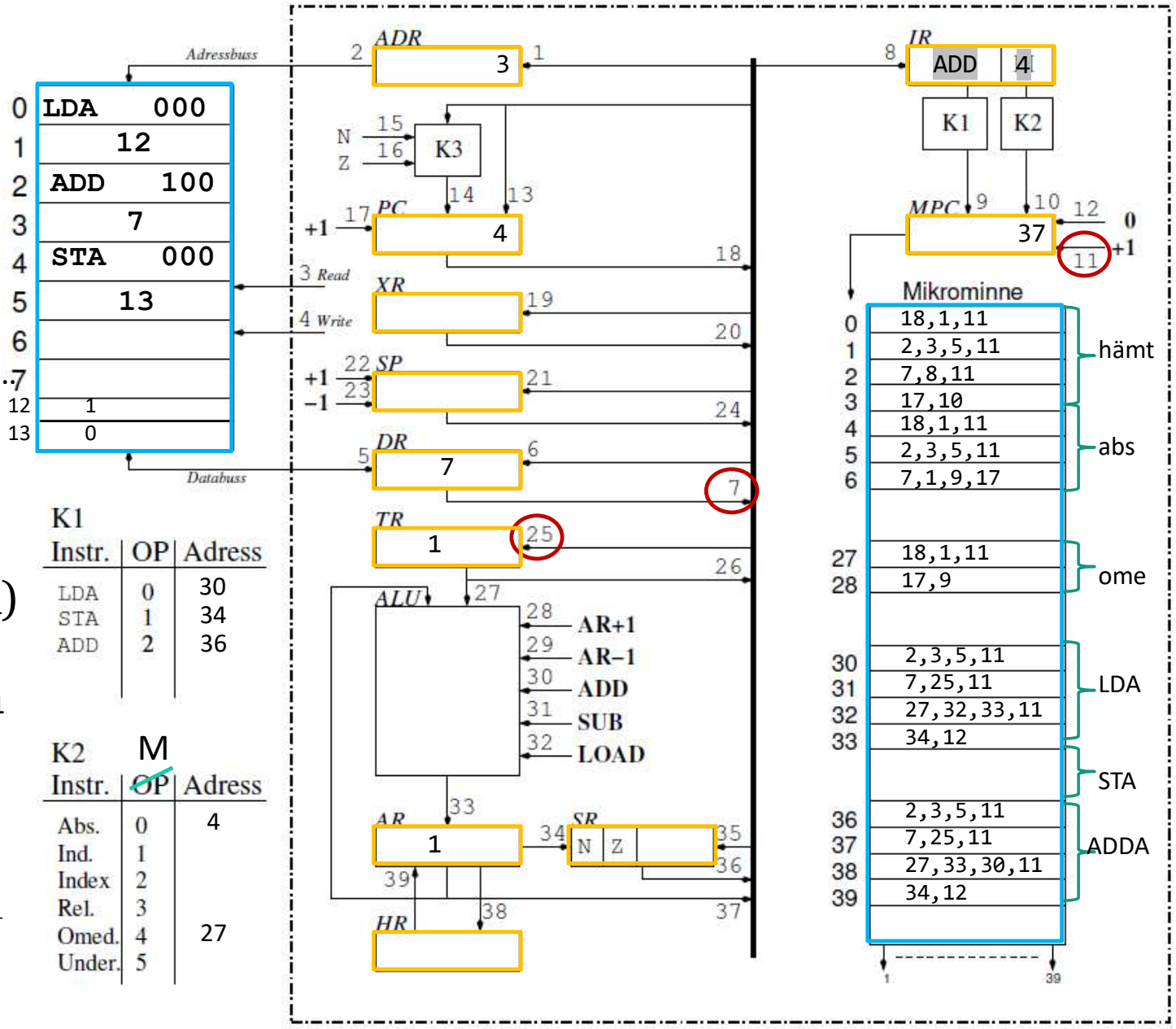
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Omedelbar

```
27: PC->adr,mpc++                  18,1,11
28: PC++,K1->mpc                   17,9
```

Steg 3 : Exe, AR=AR+M(ADR)

```
36: data->dr,mpc++                 2,3,5,11
37: dr->tr,mpc++                   7,25,11
38: ar+tr->ar, mpc++               27,33,30,11
39: status, 0->mpc                 34,12
```



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	

Mikrokod för ADD #7

Steg 1 : H-fas, som förut

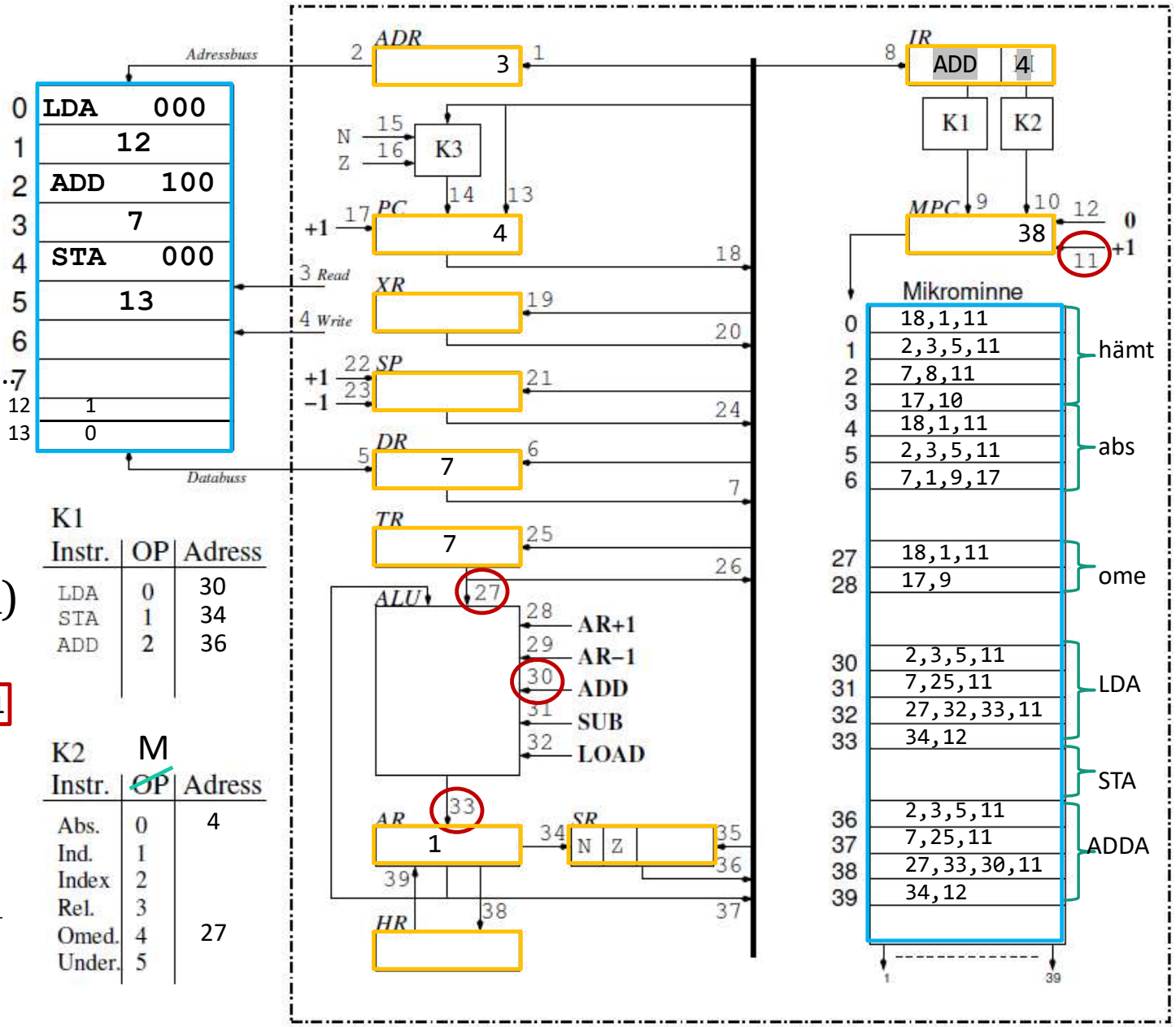
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Omedelbar

```
27: PC->adr,mpc++                  18,1,11
28: PC++,K1->mpc                   17,9
```

Steg 3 : Exe, AR=AR+M(ADR)

```
36: data->dr,mpc++                 2,3,5,11
37: dr->tr,mpc++                   7,25,11
38: ar+tr->ar, mpc++               27,33,30,11
39: status, 0->mpc                 34,12
```



K1 Instr.	OP	Adress
LDA	0	30
STA	1	34
ADD	2	36

K2 Instr.	M OP	Adress
Abs.	0	4
Ind.	1	
Index	2	
Rel.	3	
Omed.	4	27
Under.	5	

Mikrokod för ADD #7

Steg 1 : H-fas, som förut

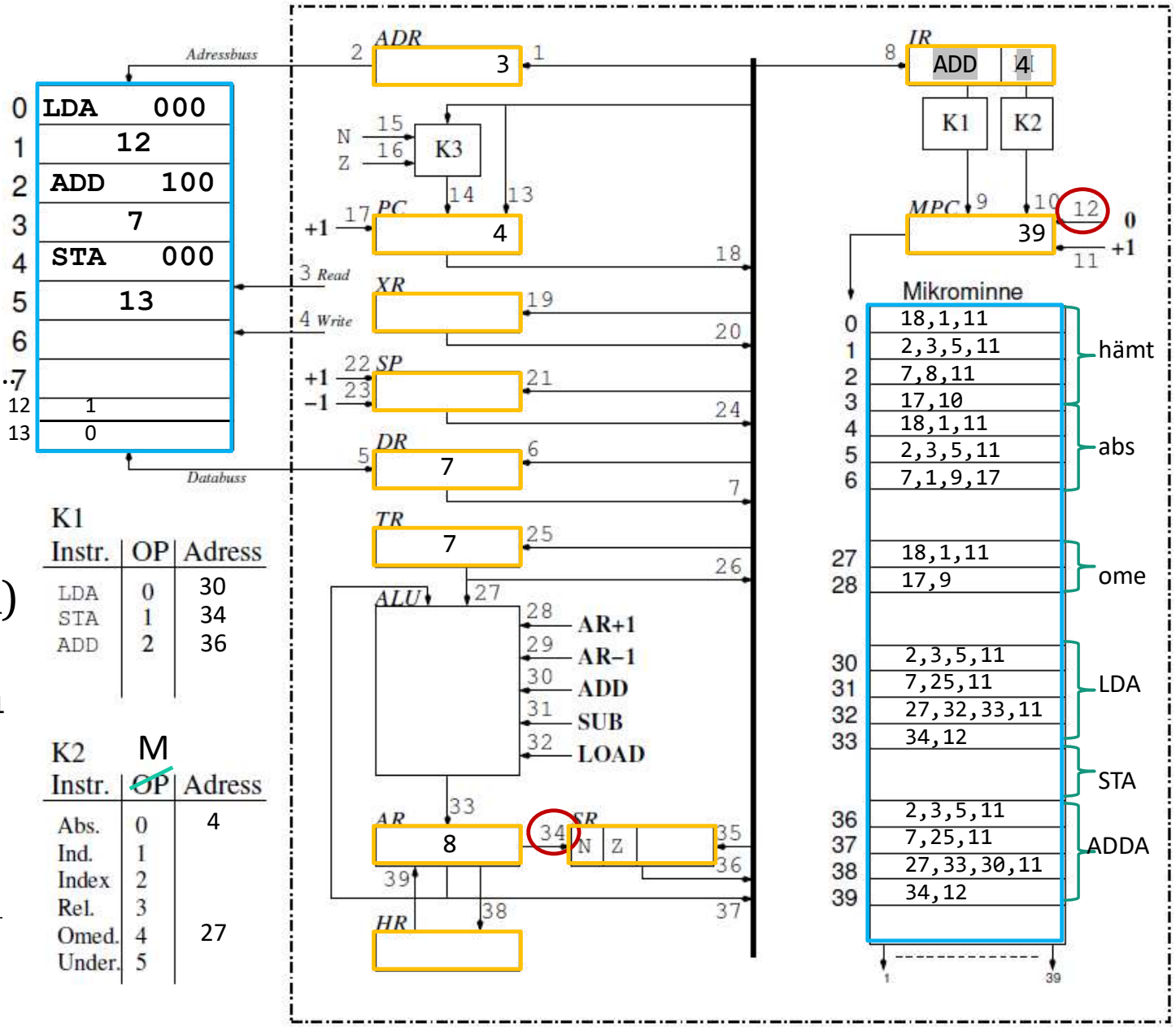
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Omedelbar

```
27: PC->adr,mpc++                  18,1,11
28: PC++,K1->mpc                   17,9
```

Steg 3 : Exe, AR=AR+M(ADR)

```
36: data->dr,mpc++                 2,3,5,11
37: dr->tr,mpc++                   7,25,11
38: ar+tr->ar, mpc++               27,33,30,11
39: status, 0->mpc                 34,12
```



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	

Mikrokod för ADD #7

Steg 1 : H-fas, som förut

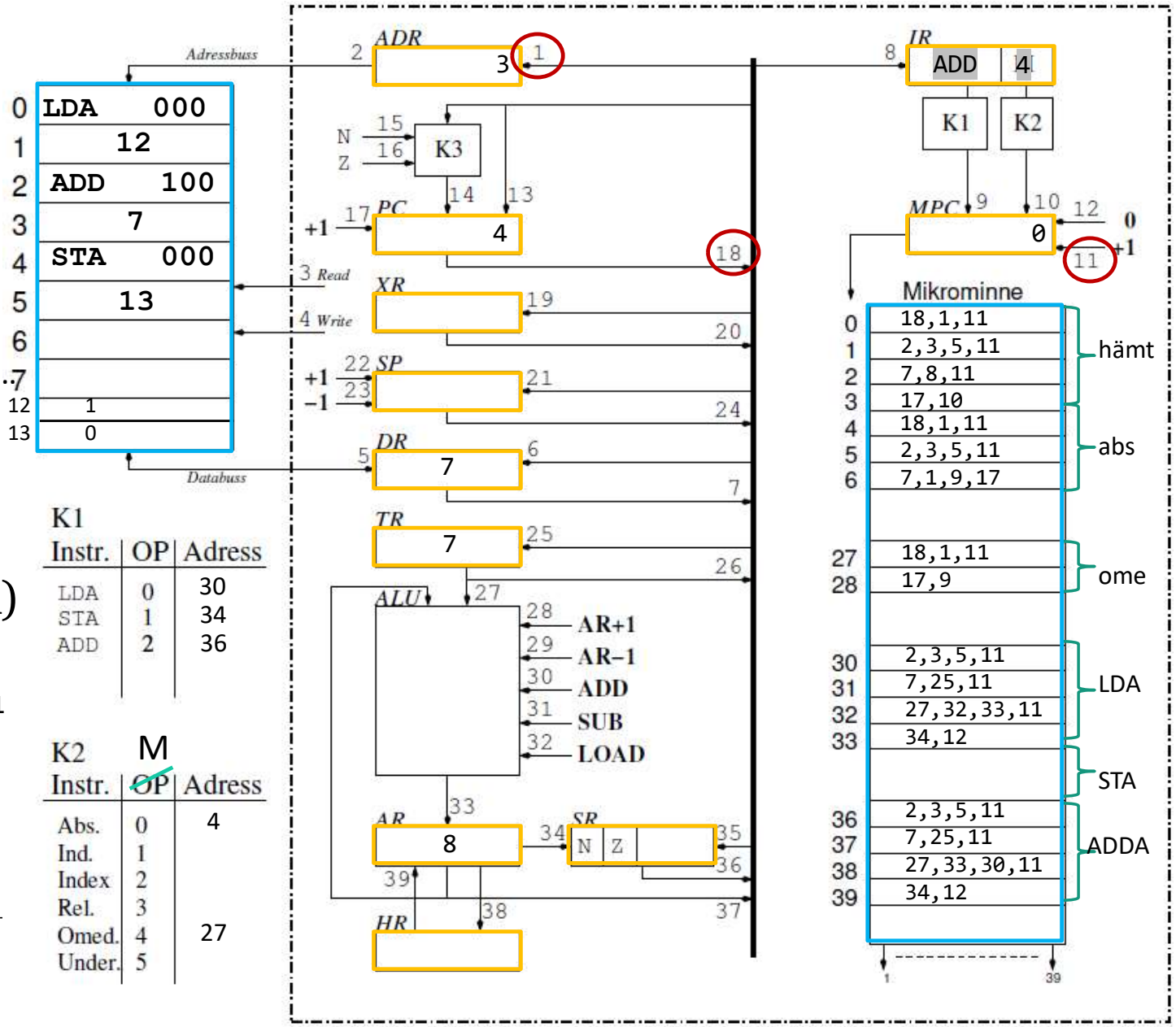
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Omedelbar

```
27: PC->adr,mpc++                  18,1,11
28: PC++,K1->mpc                   17,9
```

Steg 3 : Exe, AR=AR+M(ADR)

```
36: data->dr,mpc++                 2,3,5,11
37: dr->tr,mpc++                   7,25,11
38: ar+tr->ar, mpc++               27,33,30,11
39: status, 0->mpc                 34,12
```



Mikrokod för STA (13)

Mikrokod för STA 13

Steg 1 : H-fas, som förut

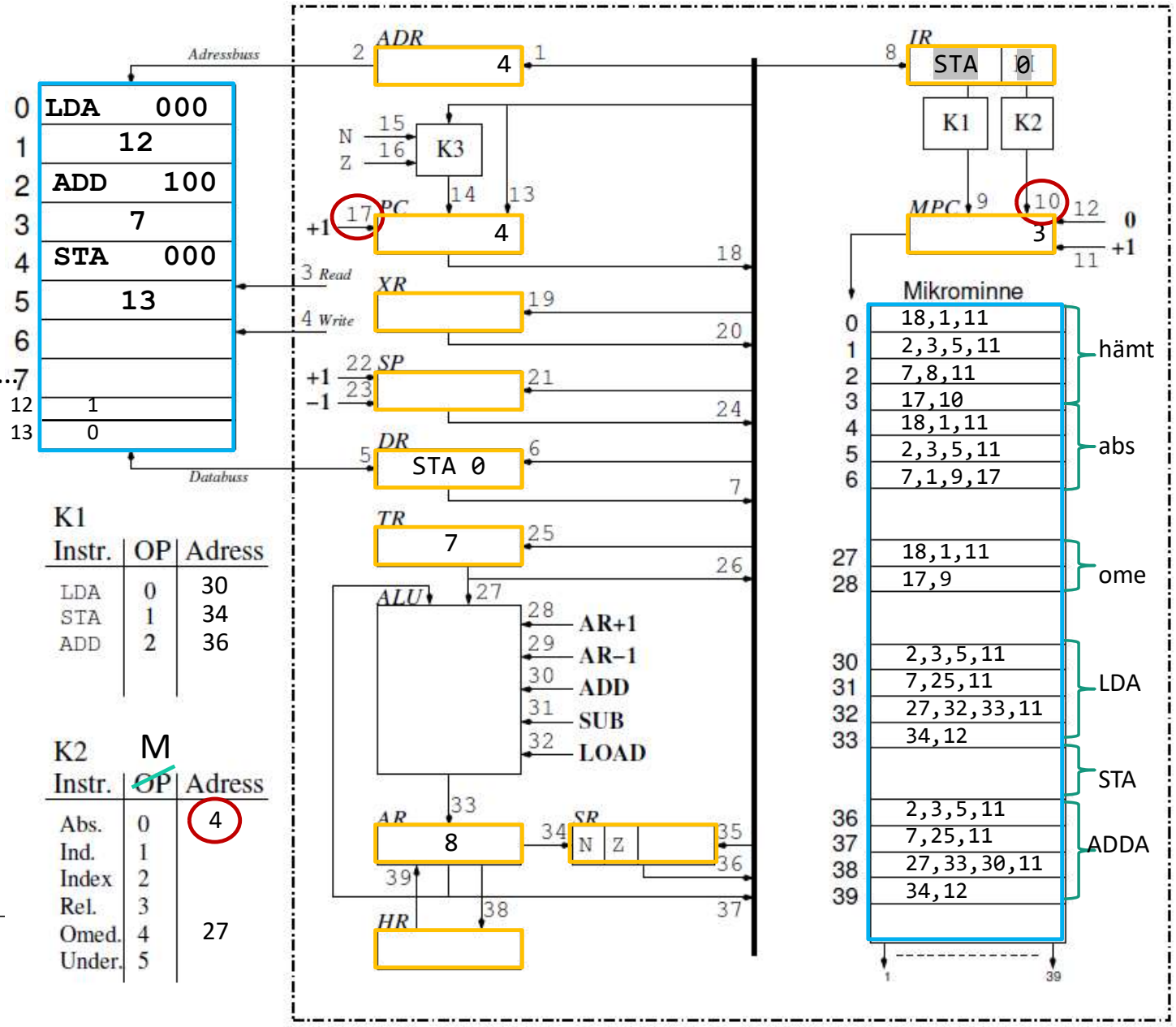
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Absolut

```
...
6: dr->adr,K1->mpc,PC++            7,1,9,17
```

Steg 3 : Exe, AR->M(ADR)

```
34: ar->dr,mpc++                   37,6,11
35: dr->M,0->mpc                   2,4,5,12
```



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	

Mikrokod för STA 13

Steg 1 : H-fas, som förut

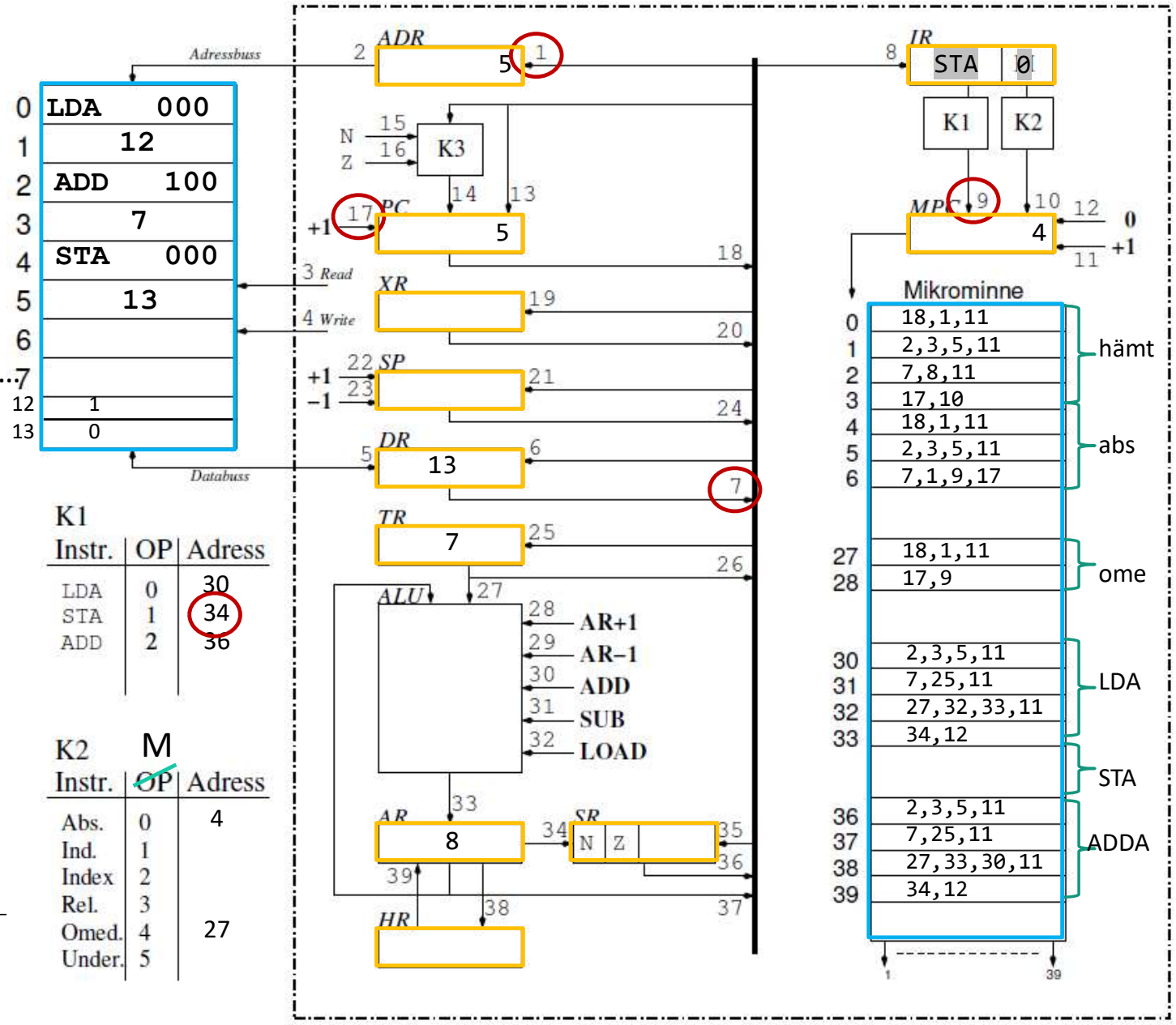
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Absolut

```
...
6: dr->adr,K1->mpc,PC++            7,1,9,17
```

Steg 3 : Exe, AR->M(ADR)

```
34: ar->dr,mpc++                   37,6,11
35: dr->M,0->mpc                   2,4,5,12
```



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	

Mikrokod för STA 13

Steg 1 : H-fas, som förut

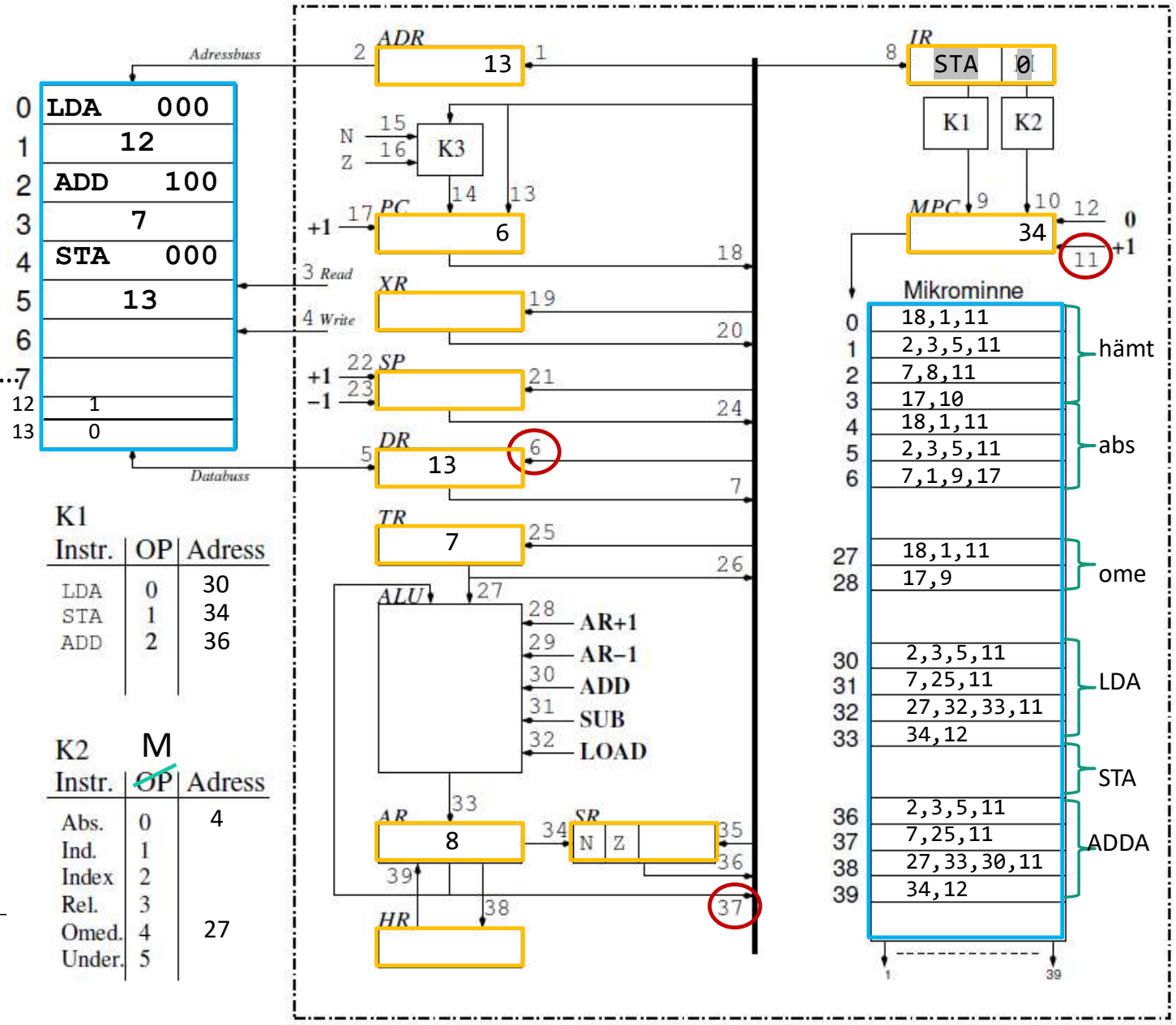
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Absolut

```
...
6: dr->adr,K1->mpc,PC++            7,1,9,17
```

Steg 3 : Exe, AR->M(ADR)

```
34: ar->dr,mpc++                   37,6,11
35: dr->M,0->mpc                   2,4,5,12
```



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	

Mikrokod för STA 13

Steg 1 : H-fas, som förut

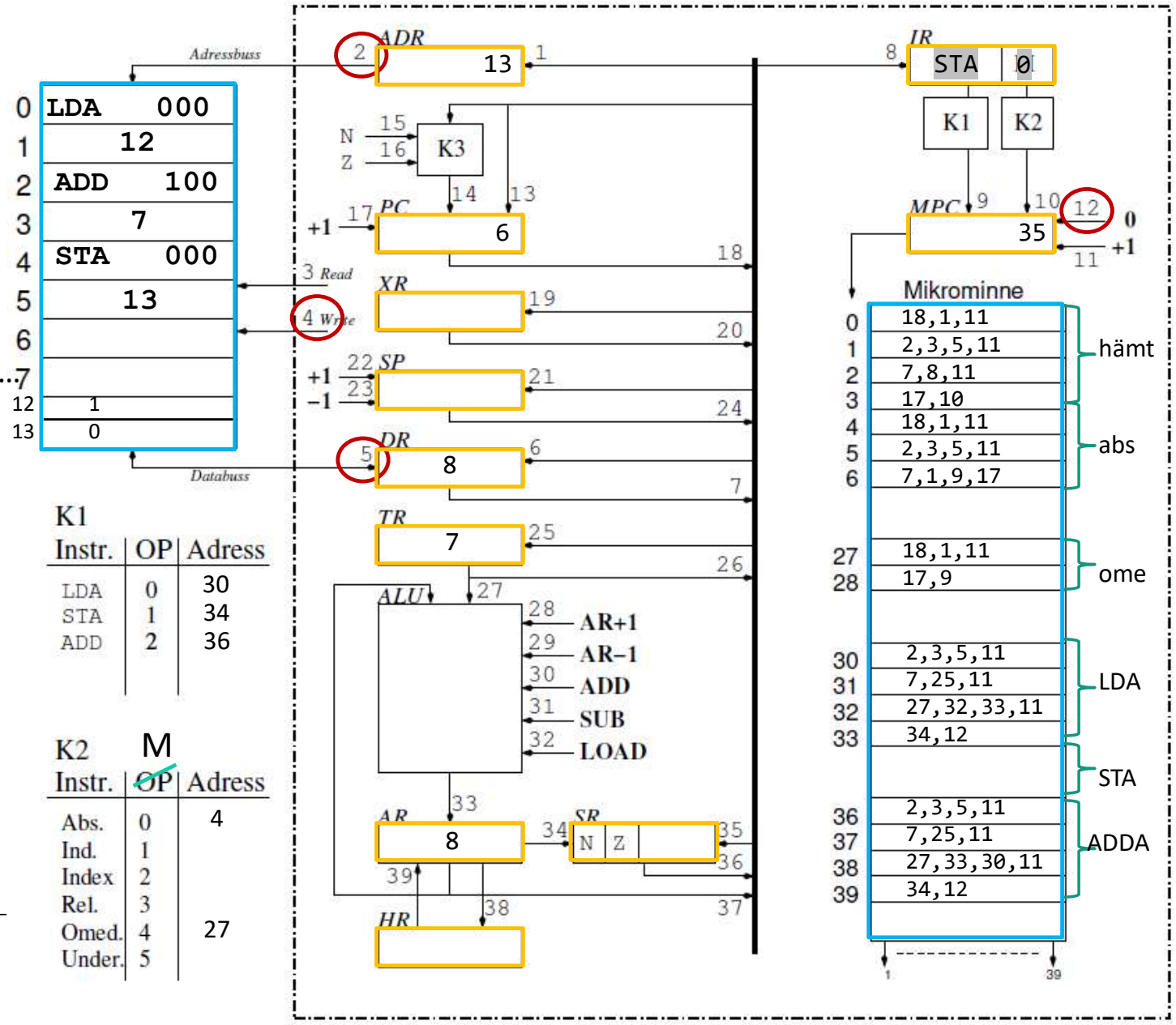
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Absolut

```
...
6: dr->adr,K1->mpc,PC++            7,1,9,17
```

Steg 3 : Exe, AR->M(ADR)

```
34: ar->dr,mpc++                   37,6,11
35: dr->M,0->mpc                   2,4,5,12
```



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	

Mikrokod för STA 13

Steg 1 : H-fas, som förut

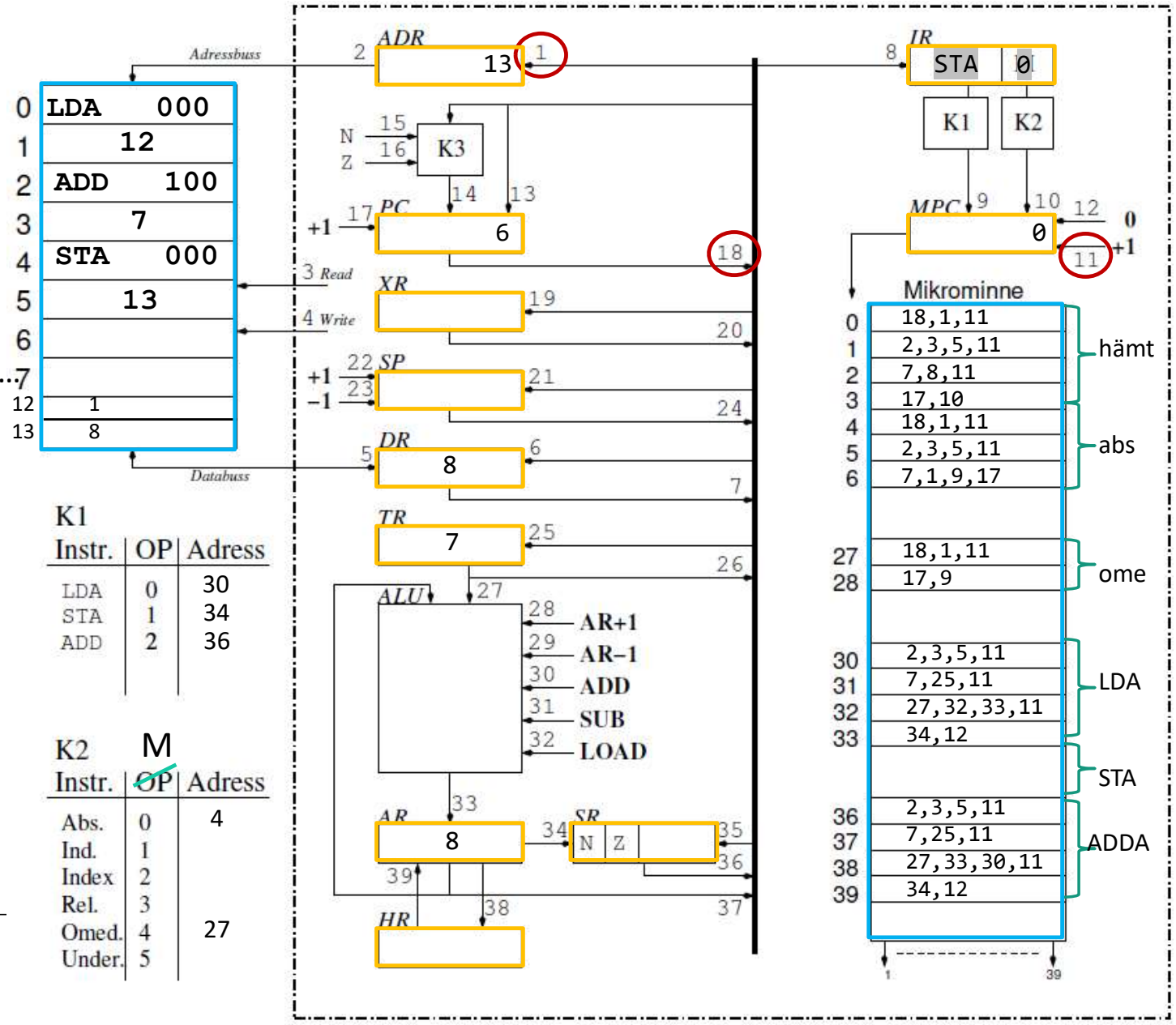
```
...
3: PC++,K2->mpc                    17,10
```

Steg 2 : A-fas, Absolut

```
...
6: dr->adr,K1->mpc,PC++            7,1,9,17
```

Steg 3 : Exe, AR->M(ADR)

```
34: ar->dr,mpc++                   37,6,11
35: dr->M,0->mpc                   2,4,5,12
```



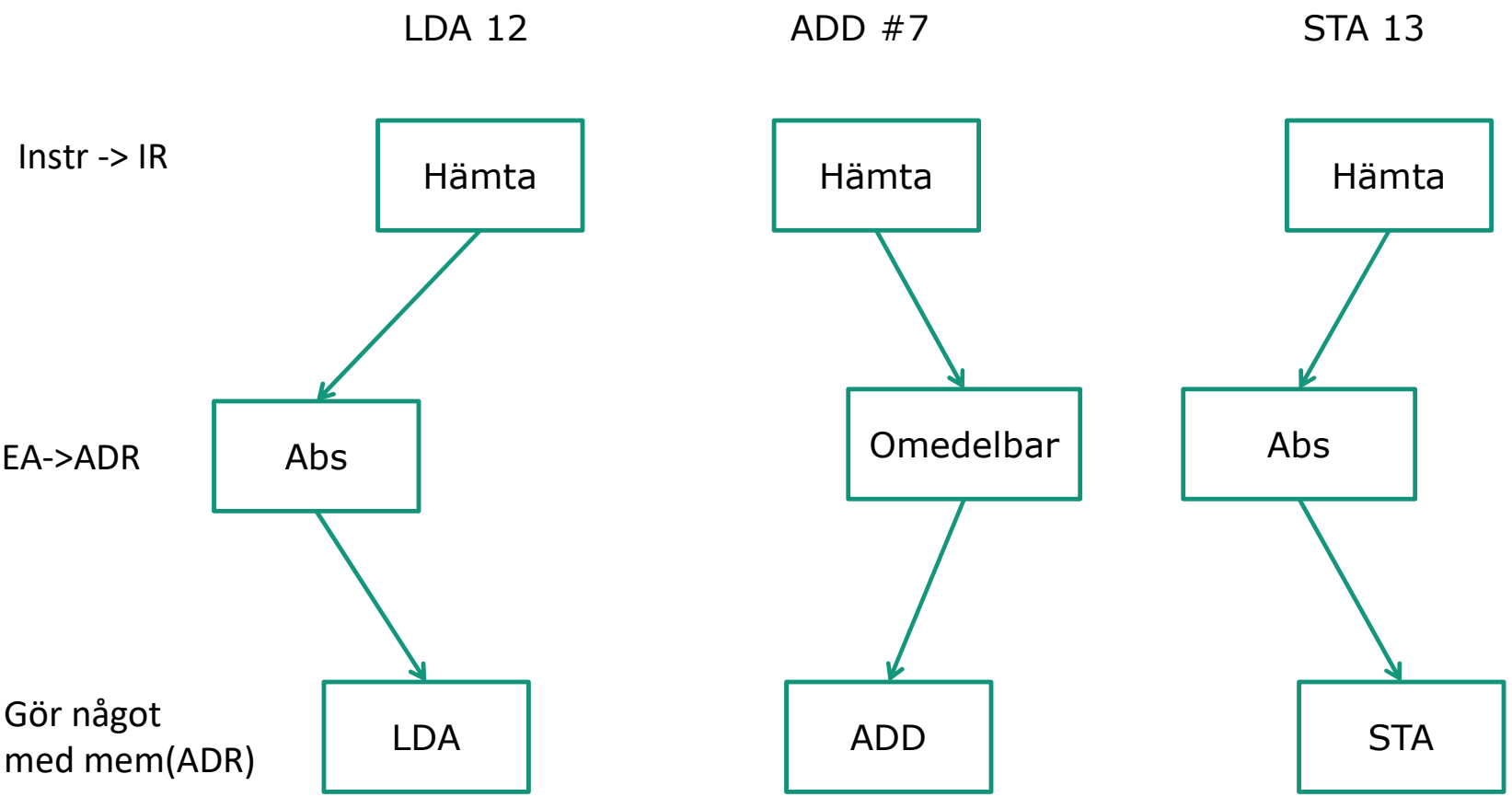
K1

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	

Sammanfattning



Mikrokod för LDA 3(X)

Indexerad adressering

Mikrokod för LDA 3(X)

`M(XR+3) -> AR`

Steg 1 : H-fas

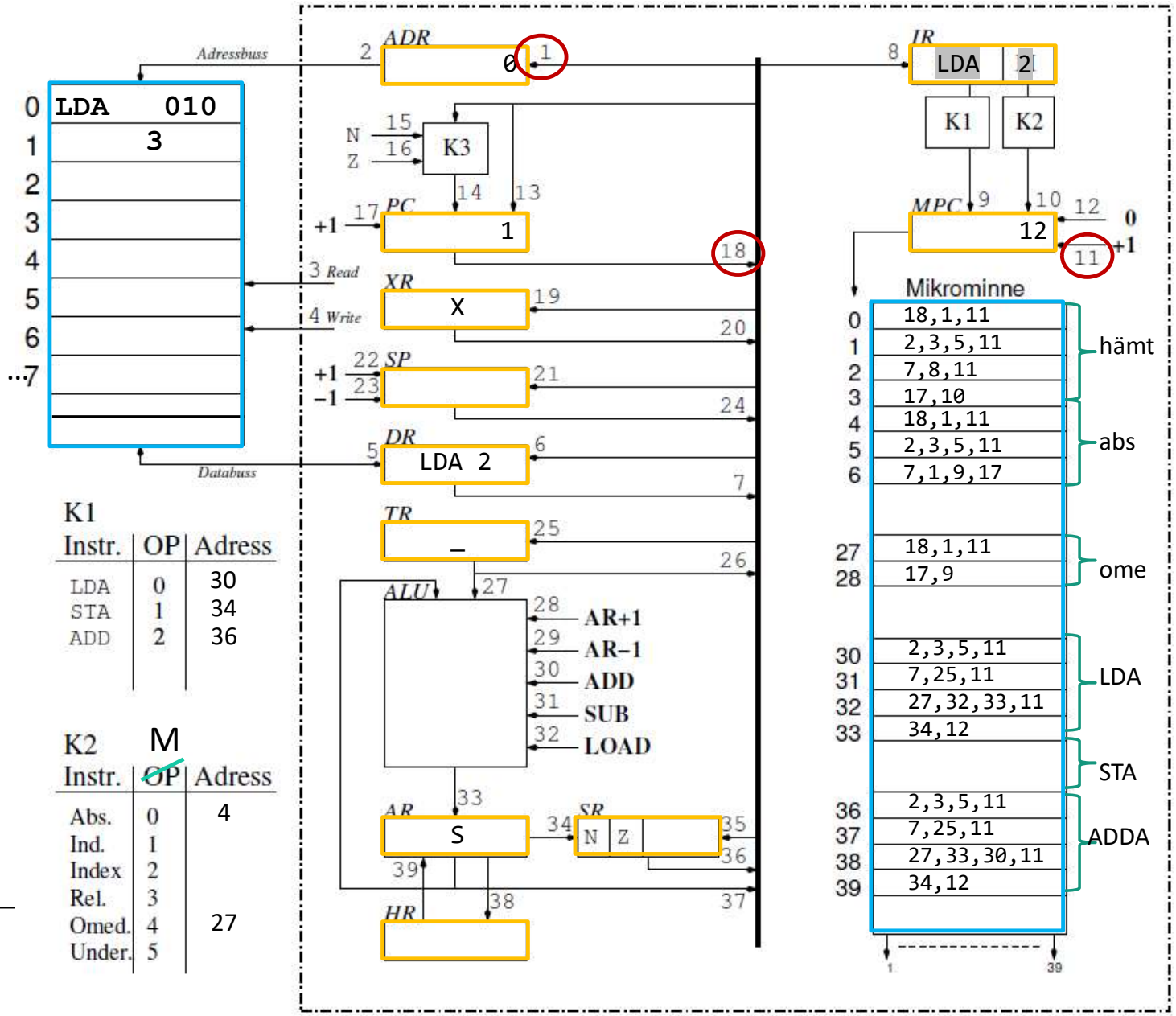
`Finns redan`

Steg 2 : A-fas, Indexerad

```
12: PC->ADR, PC++, MPC++
13: M->DR, XR->TR, MPC++
14: DR->TR, TR->AR, AR->HR, MPC++
15: AR+TR->AR, MPC++
16: HR->AR, AR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för LDA 3(X)

`M(XR+3) -> AR`

Steg 1 : H-fas

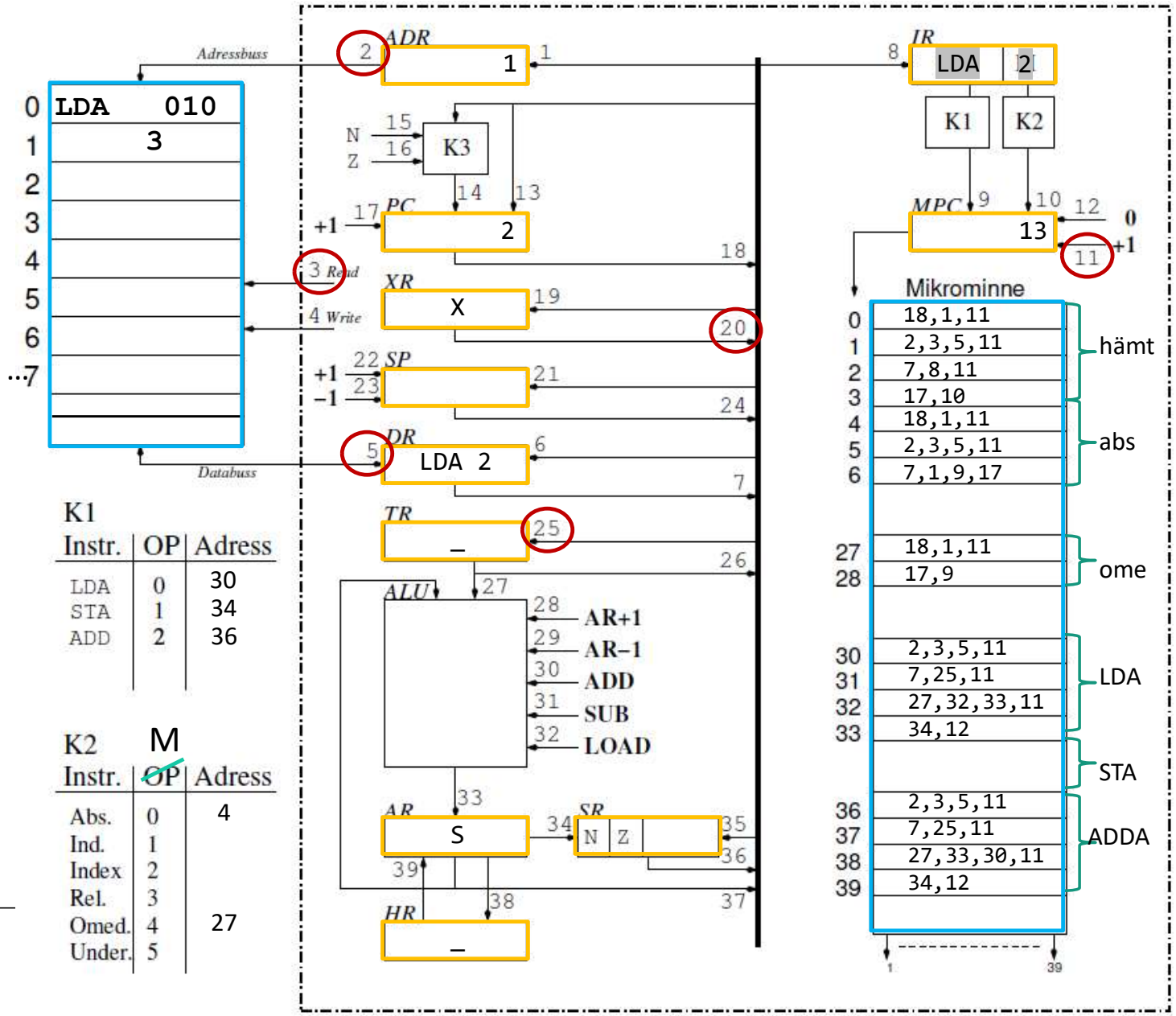
`Finns redan`

Steg 2 : A-fas, Indexerad

```
12: PC->ADR, PC++, MPC++
13: M->DR, XR->TR, MPC++
14: DR->TR, TR->AR, AR->HR, MPC++
15: AR+TR->AR, MPC++
16: HR->AR, AR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för LDA 3(X)

`M(XR+3) -> AR`

Steg 1 : H-fas

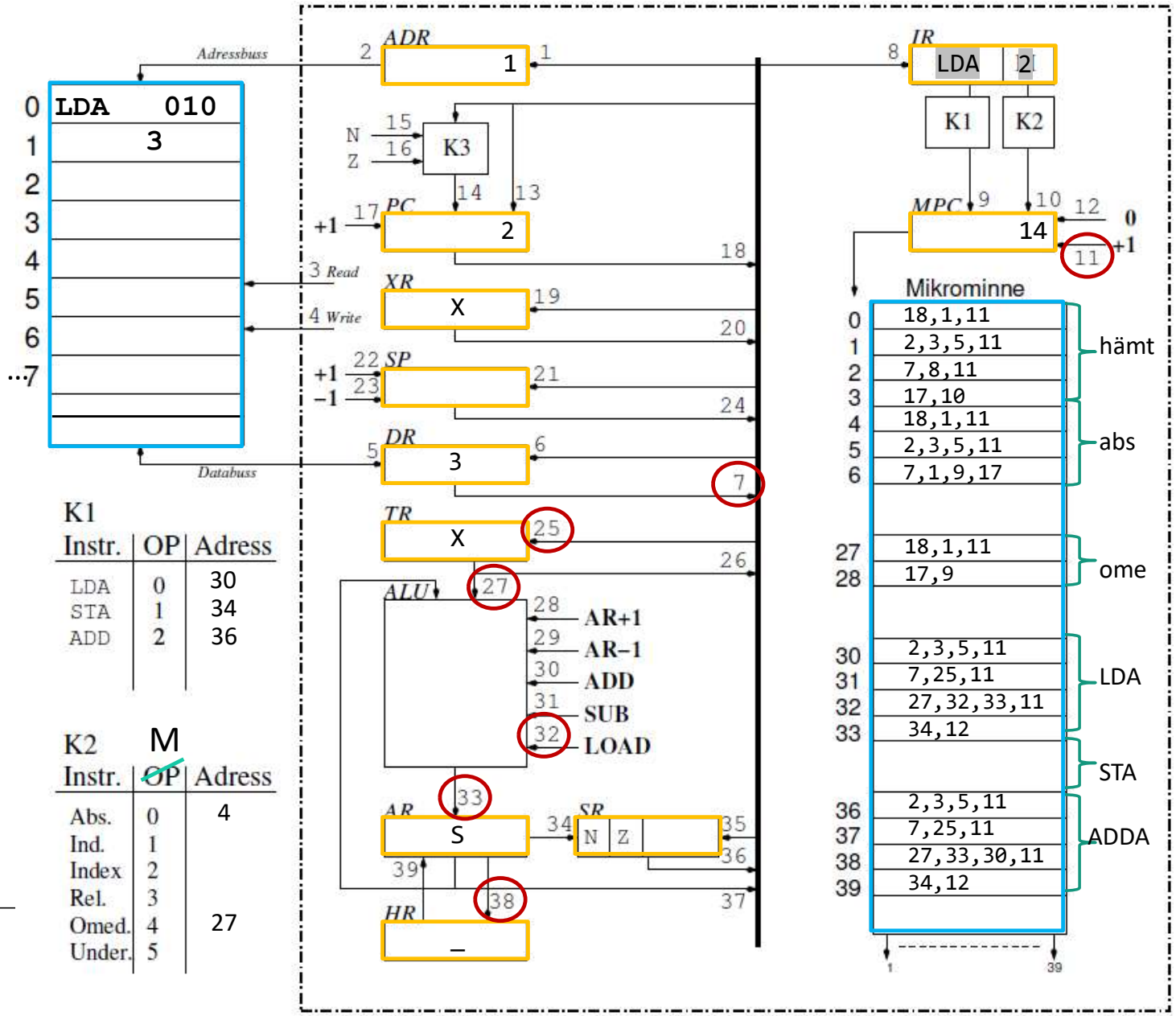
`Finns redan`

Steg 2 : A-fas, Indexerad

```
12: PC->ADR, PC++, MPC++
13: M->DR, XR->TR, MPC++
14: DR->TR, TR->AR, AR->HR, MPC++
15: AR+TR->AR, MPC++
16: HR->AR, AR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för LDA 3(X)

`M(XR+3) -> AR`

Steg 1 : H-fas

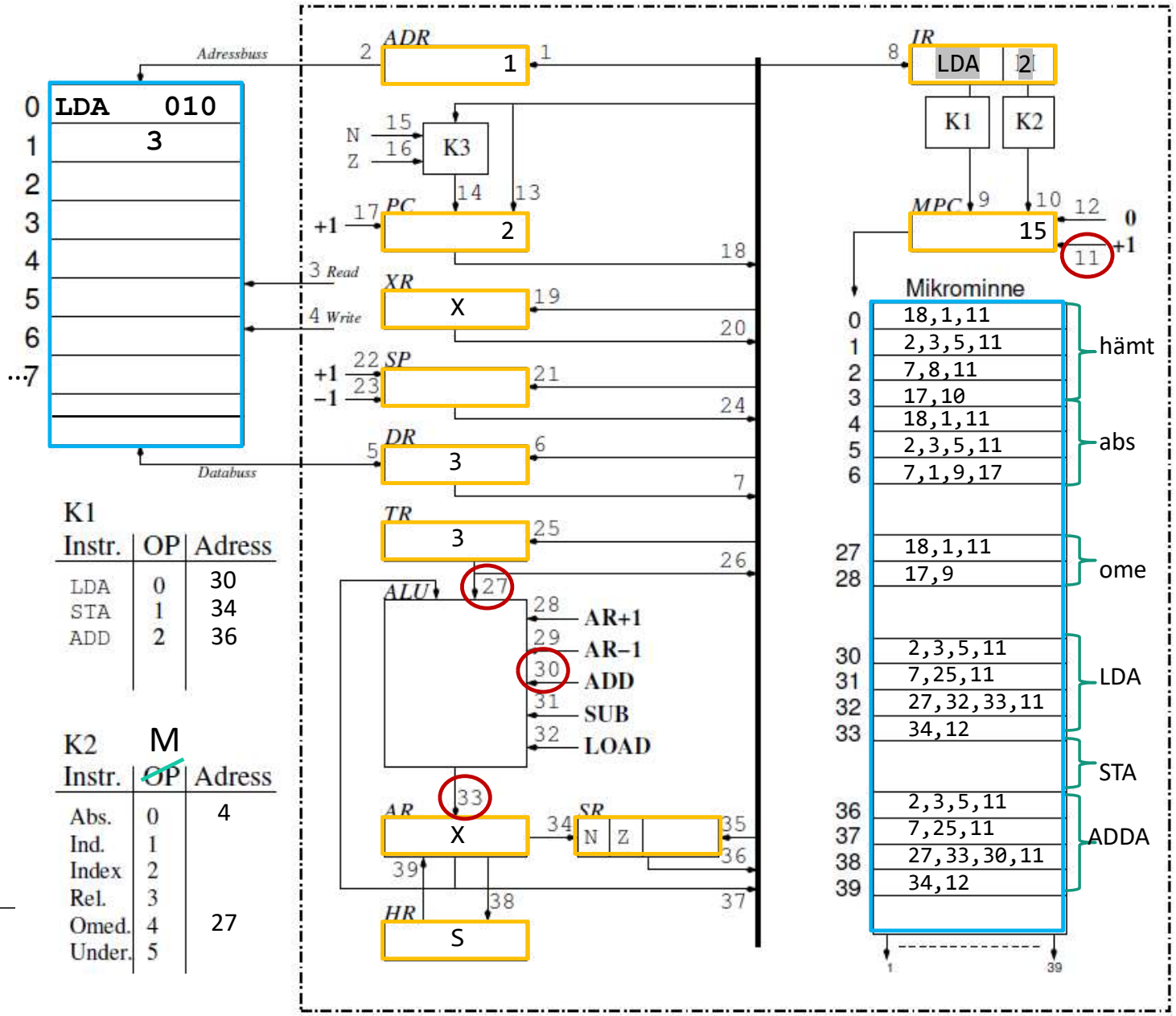
`Finns redan`

Steg 2 : A-fas, Indexerad

```
12: PC->ADR, PC++, MPC++
13: M->DR, XR->TR, MPC++
14: DR->TR, TR->AR, AR->HR, MPC++
15: AR+TR->AR, MPC++
16: HR->AR, AR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för LDA 3(X)

`M(XR+3) -> AR`

Steg 1 : H-fas

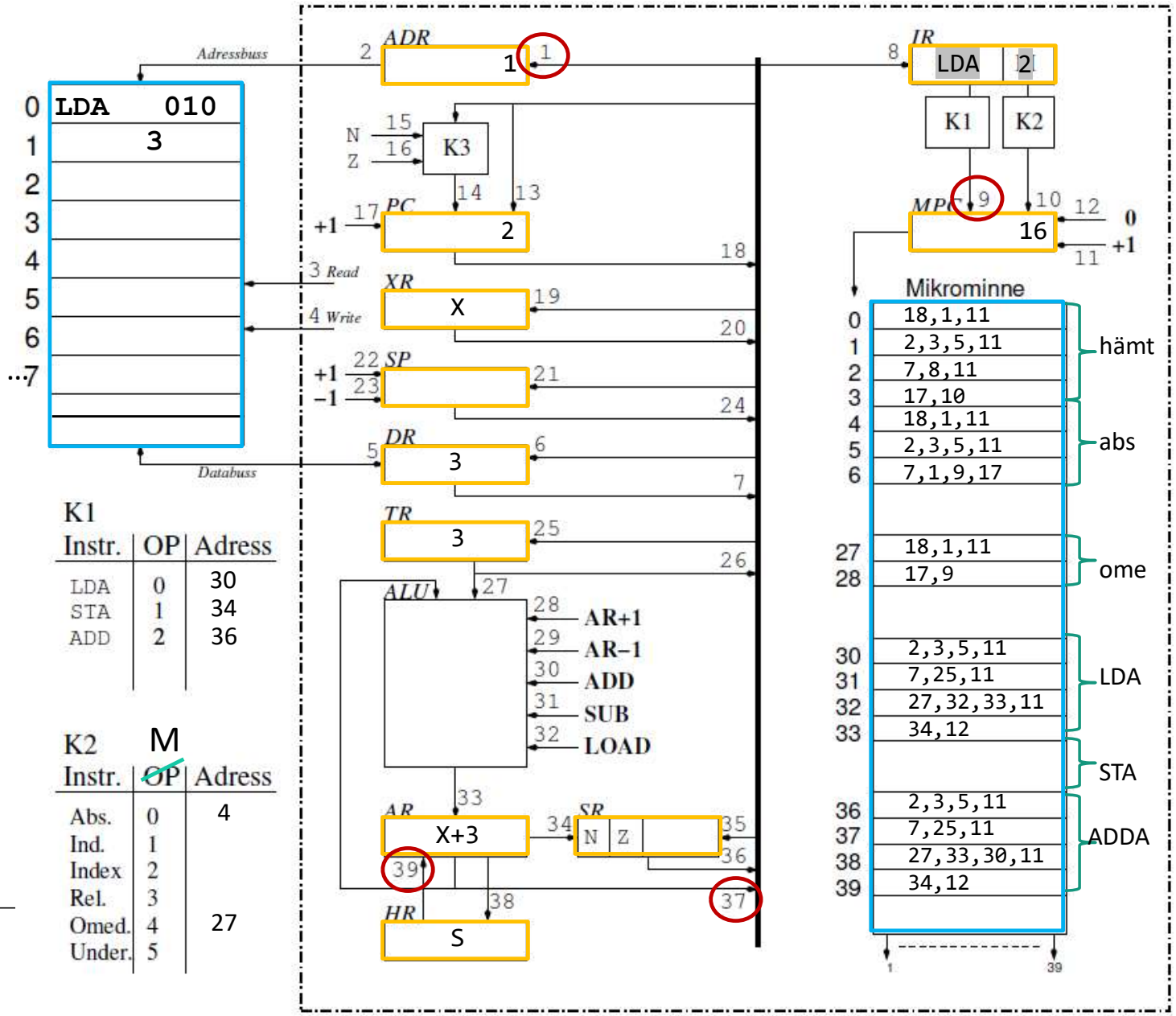
`Finns redan`

Steg 2 : A-fas, Indexerad

```
12: PC->ADR, PC++, MPC++
13: M->DR, XR->TR, MPC++
14: DR->TR, TR->AR, AR->HR, MPC++
15: AR+TR->AR, MPC++
16: HR->AR, AR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för LDA 3(X)

`M(XR+3) -> AR`

Steg 1 : H-fas

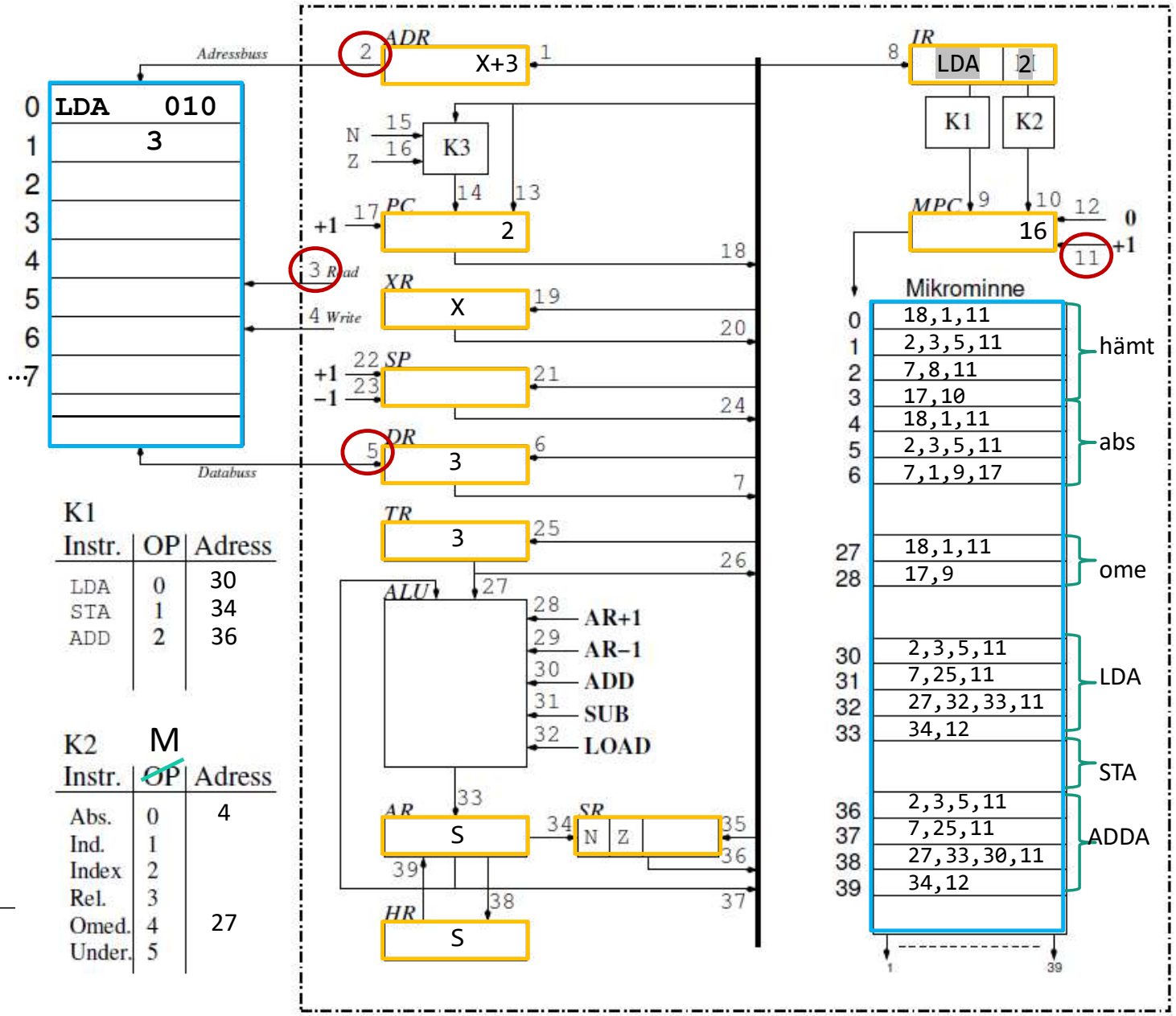
`Finns redan`

Steg 2 : A-fas, Indexerad

```
12: PC->ADR, PC++, MPC++
13: M->DR, XR->TR, MPC++
14: DR->TR, TR->AR, AR->HR, MPC++
15: AR+TR->AR, MPC++
16: HR->AR, AR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för INCA

Underförstådd adressering

Mikrokod för INCA

AR+1 -> AR

Steg 1 : H-fas

Finns redan

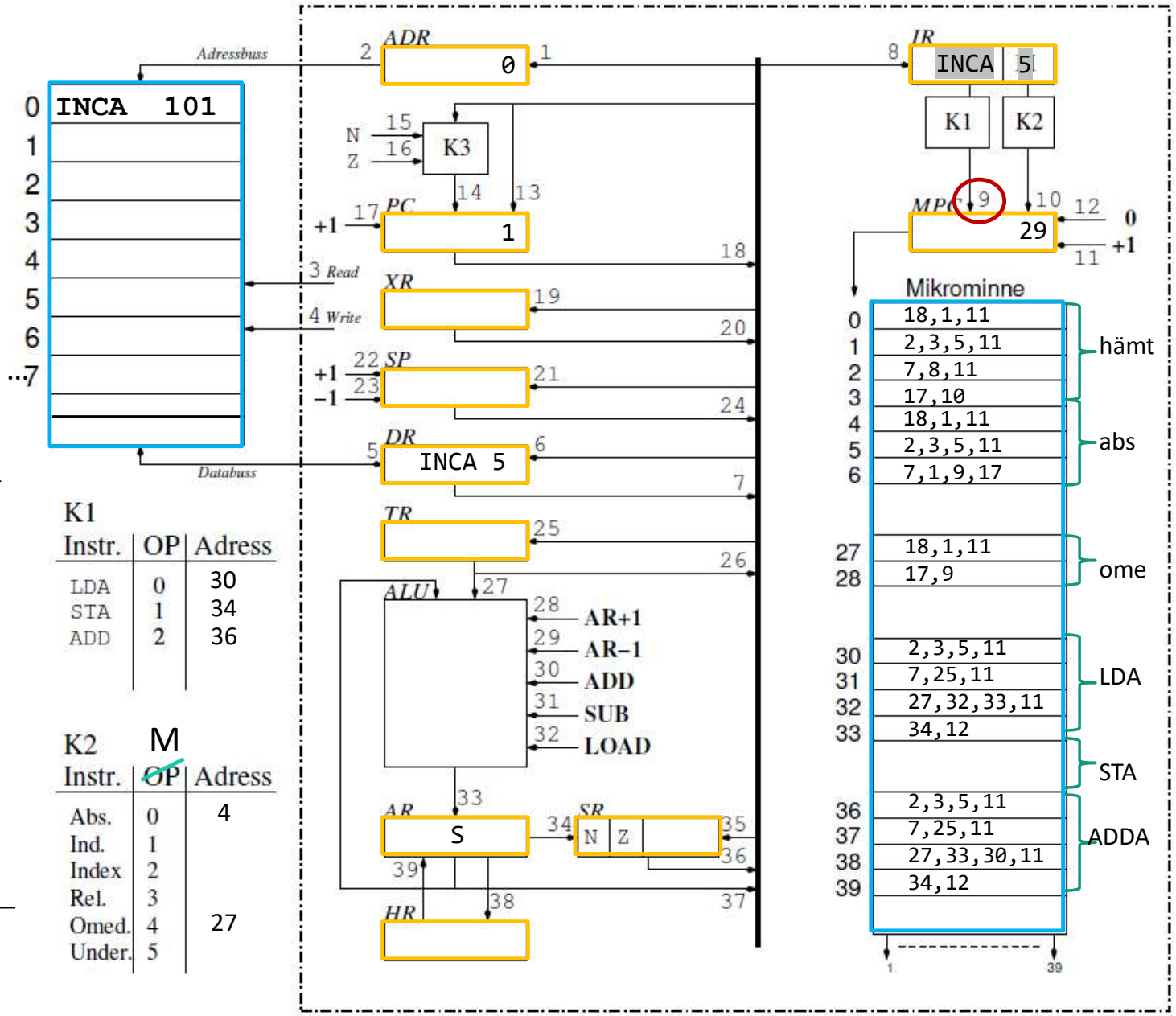
Steg 2 : A-fas, Underförstådd

29: K1->MPC

Steg 3 : Exe, INCA

44: AR+1->AR, MPC++

45: status, 0->MPC



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	

Mikrokod för INCA

AR+1 -> AR

Steg 1 : H-fas

Finns redan

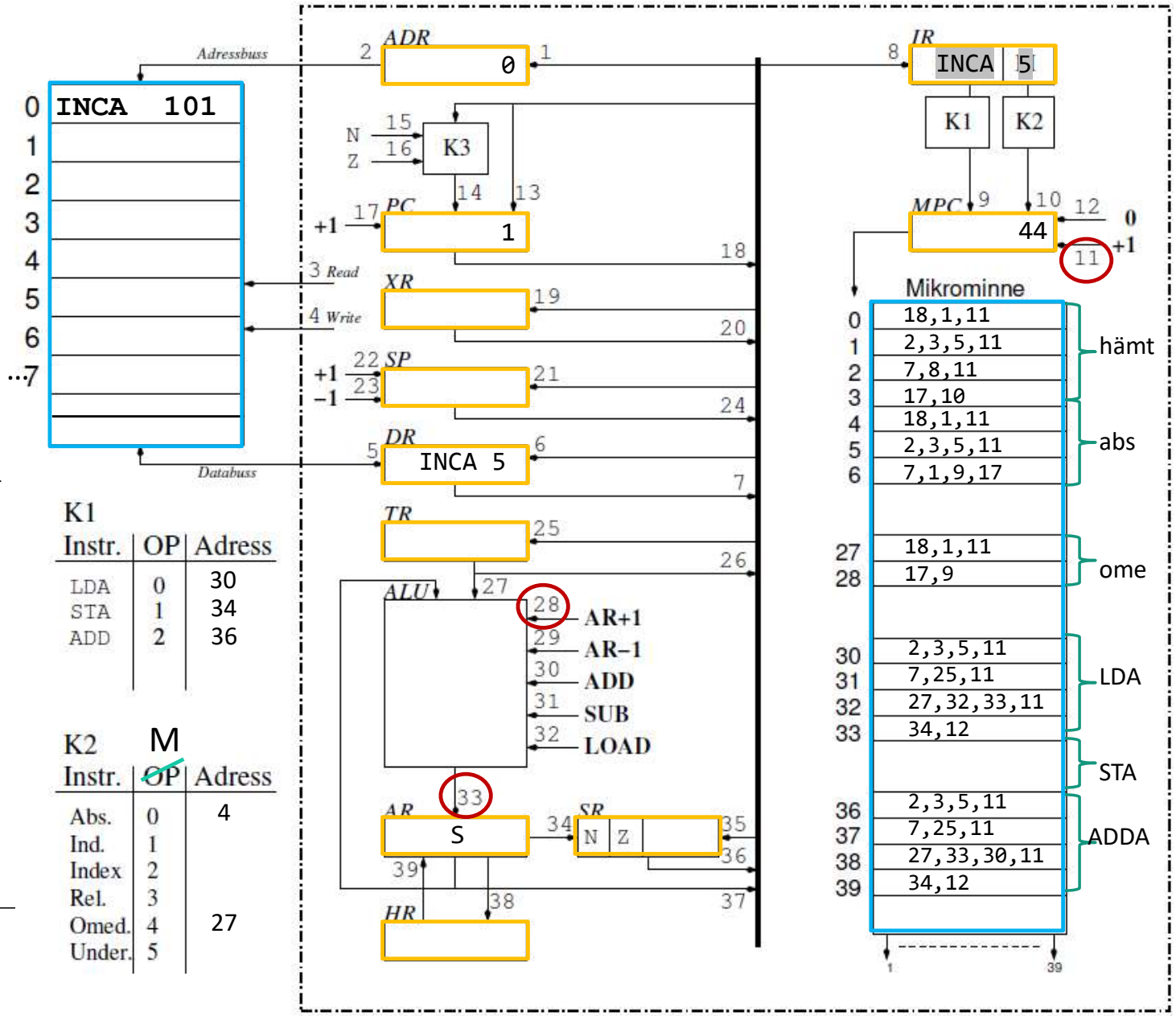
Steg 2 : A-fas, Underförstådd

29: K1->MPC

Steg 3 : Exe, INCA

44: AR+1->AR, MPC++

45: status, 0->MPC



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	

Mikrokod för INCA

AR+1 -> AR

Steg 1 : H-fas

Finns redan

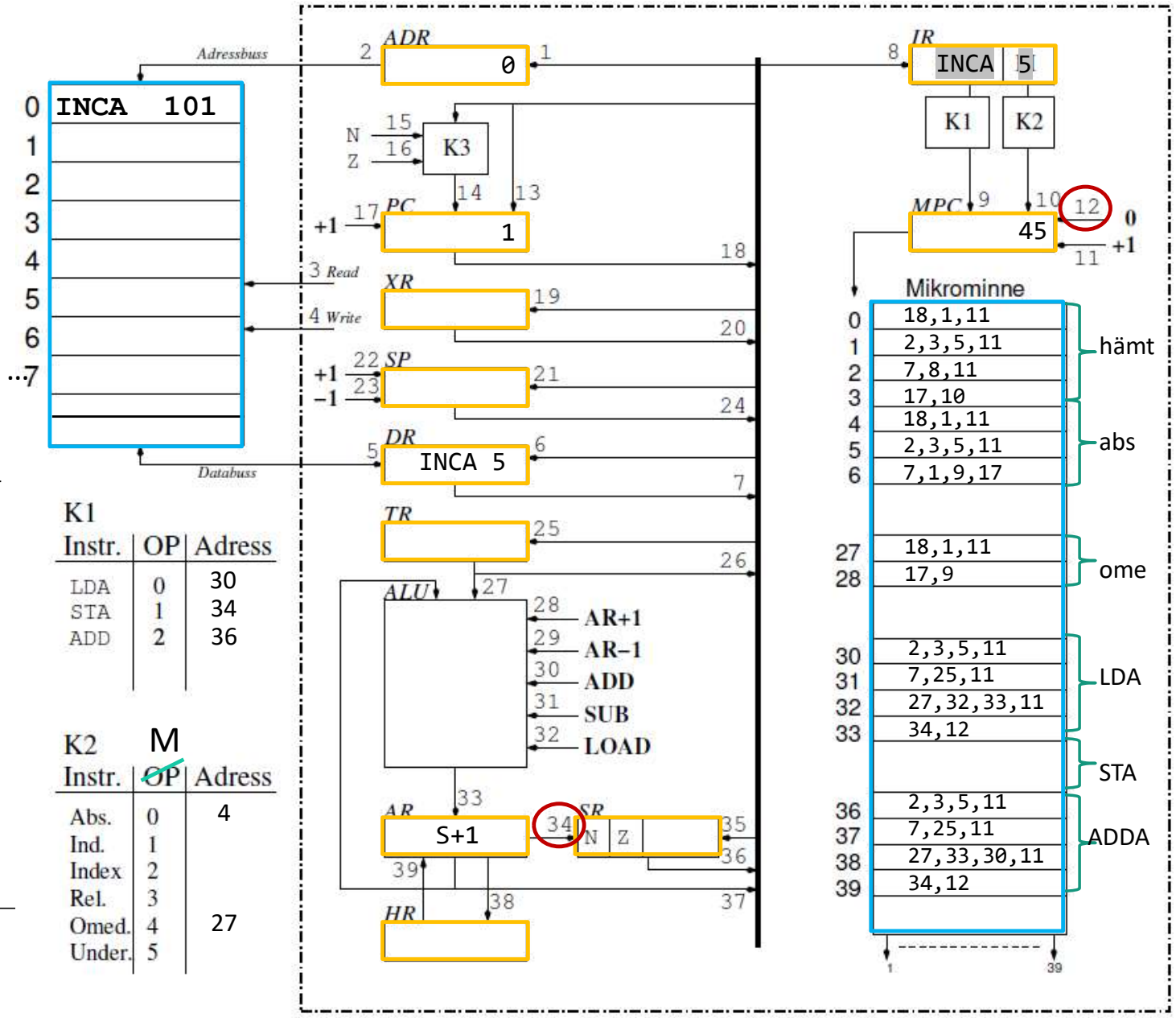
Steg 2 : A-fas, Underförstådd

29: K1->MPC

Steg 3 : Exe, INCA

44: AR+1->AR, MPC++

45: status, 0->MPC



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	

Mikrokod för LDA (3)

Indirekt adressering

Mikrokod för LDA (3)

`M(M(3)) -> AR`

Steg 1 : H-fas

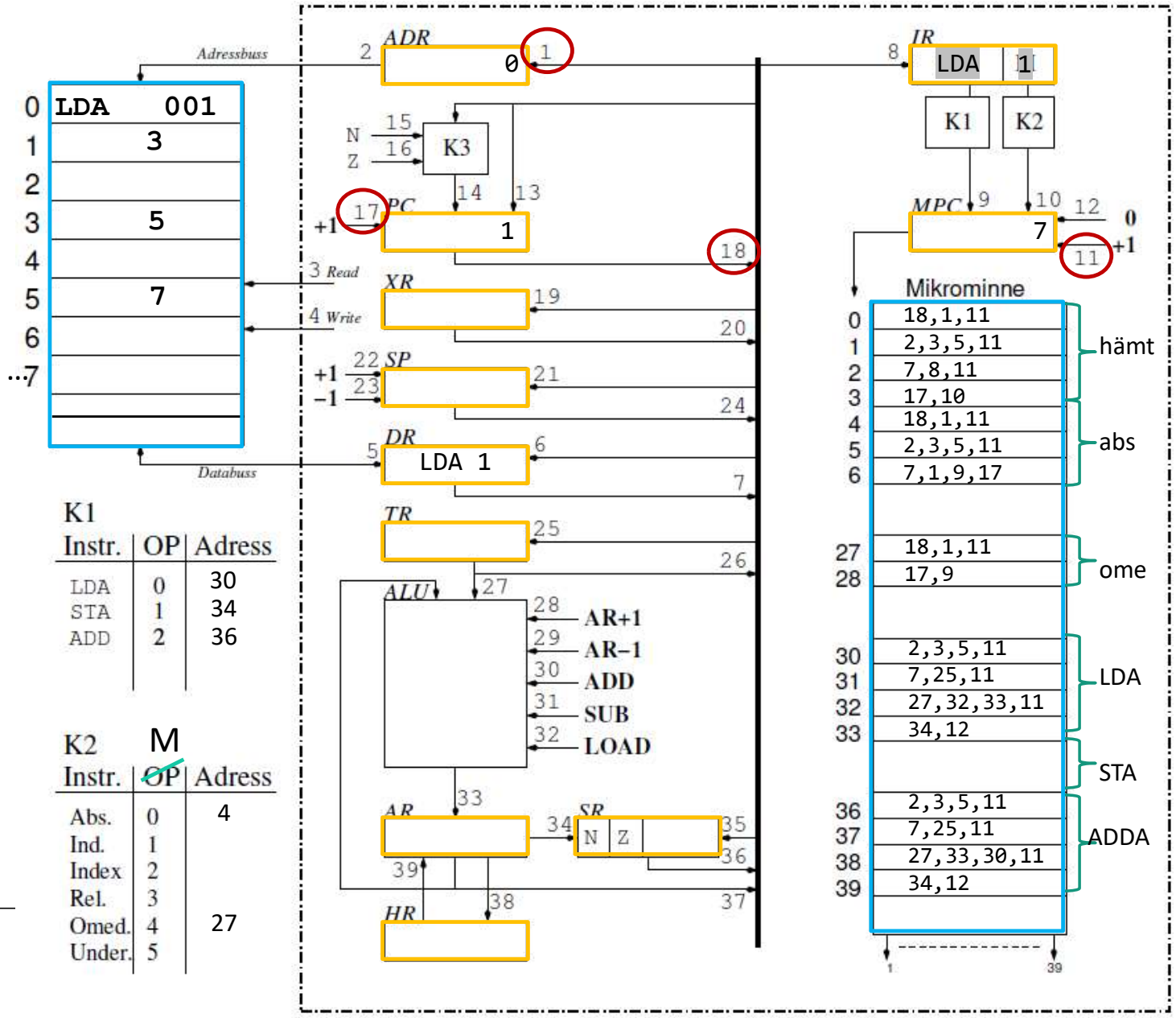
`Finns redan`

Steg 2 : A-fas, Indirekt

```
 7: PC->ADR, PC++, MPC++
 8: M->DR, MPC++
 9: DR->ADR, MPC++
10: M->DR, MPC++
11: DR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för LDA (3)

`M(M(3)) -> AR`

Steg 1 : H-fas

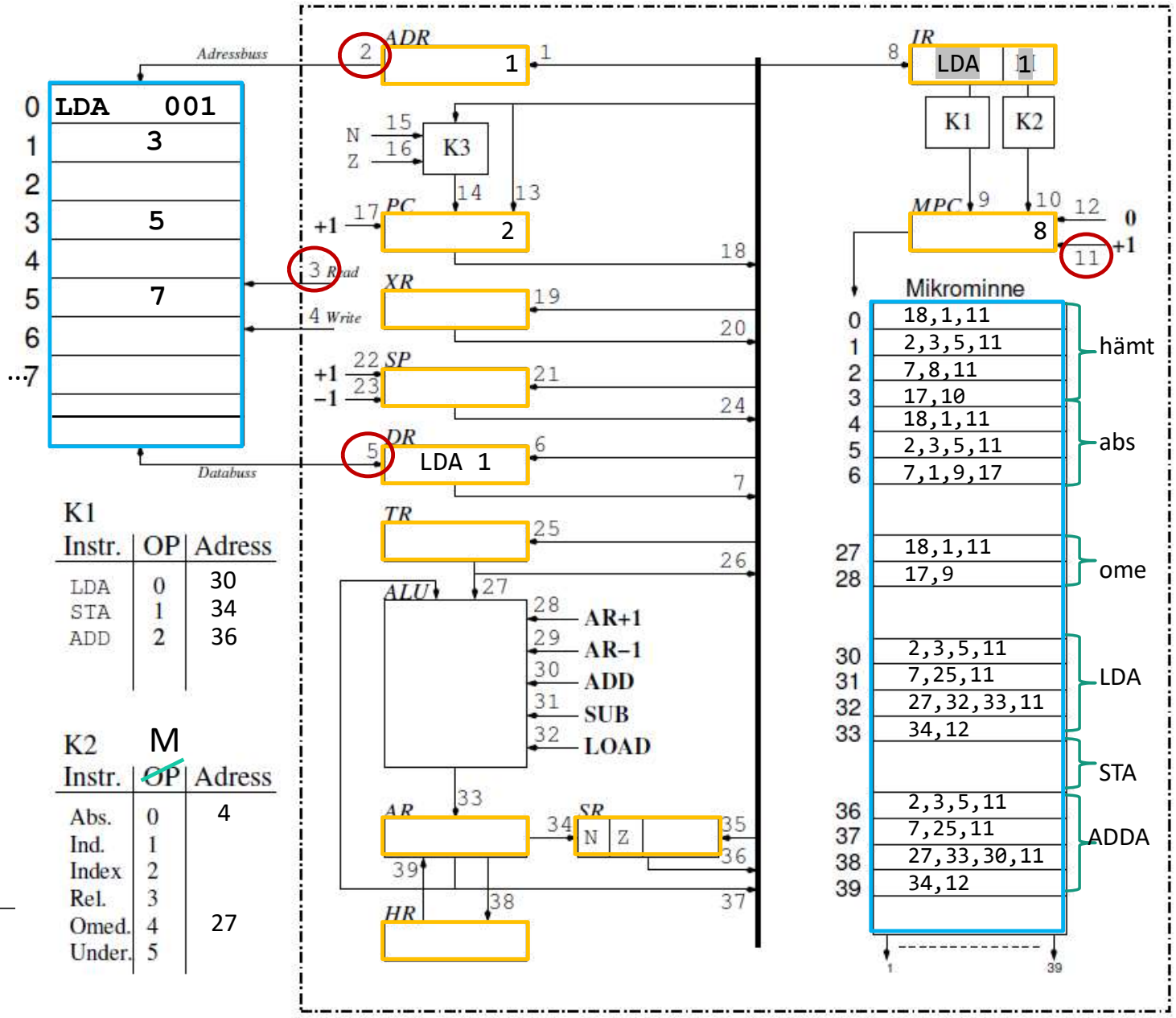
`Finns redan`

Steg 2 : A-fas, Indirekt

```
 7: PC->ADR, PC++, MPC++
 8: M->DR, MPC++
 9: DR->ADR, MPC++
10: M->DR, MPC++
11: DR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för LDA (3)

`M(M(3)) -> AR`

Steg 1 : H-fas

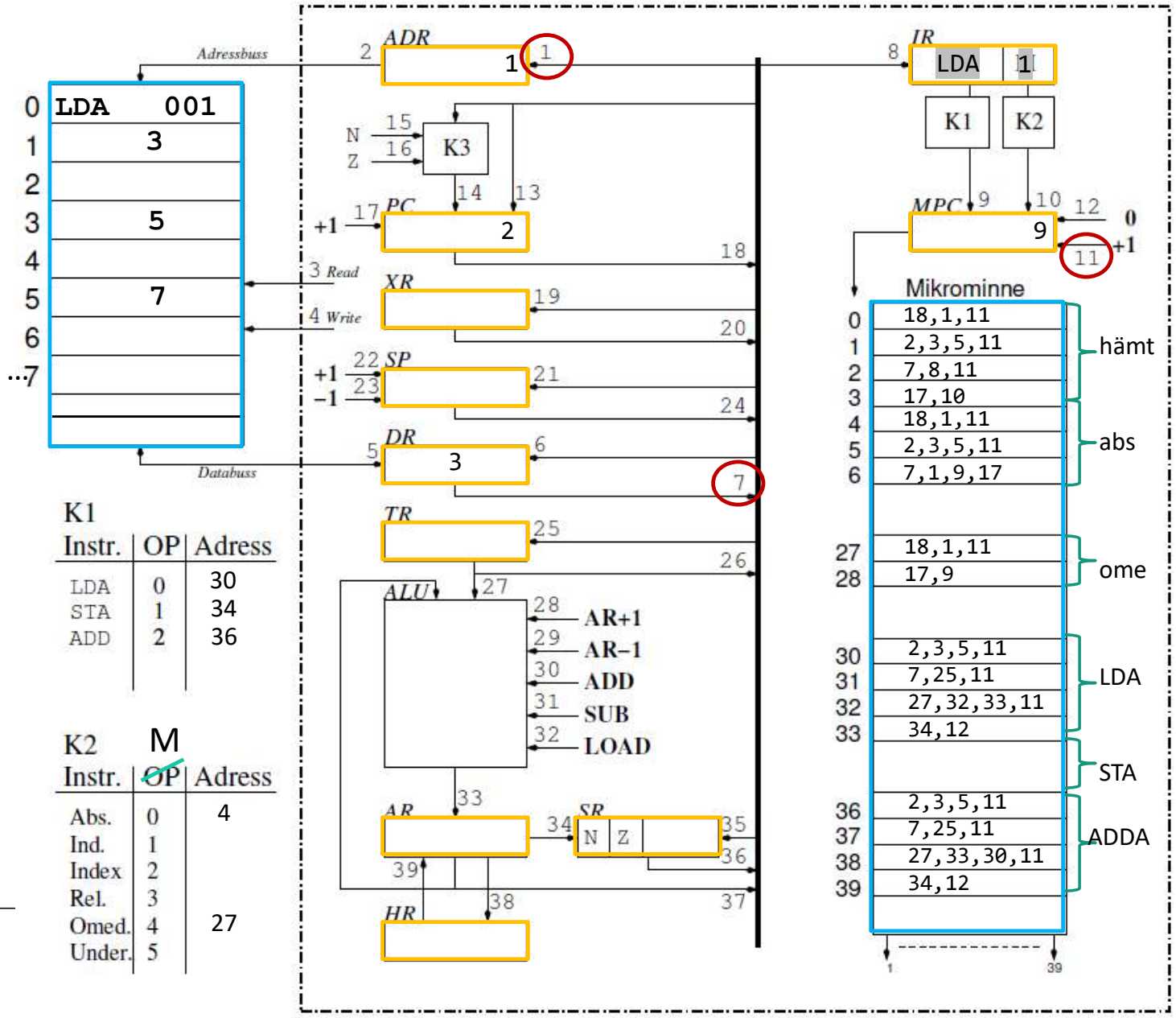
`Finns redan`

Steg 2 : A-fas, Indirekt

```
 7: PC->ADR, PC++, MPC++
 8: M->DR, MPC++
 9: DR->ADR, MPC++
10: M->DR, MPC++
11: DR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för LDA (3)

`M(M(3)) -> AR`

Steg 1 : H-fas

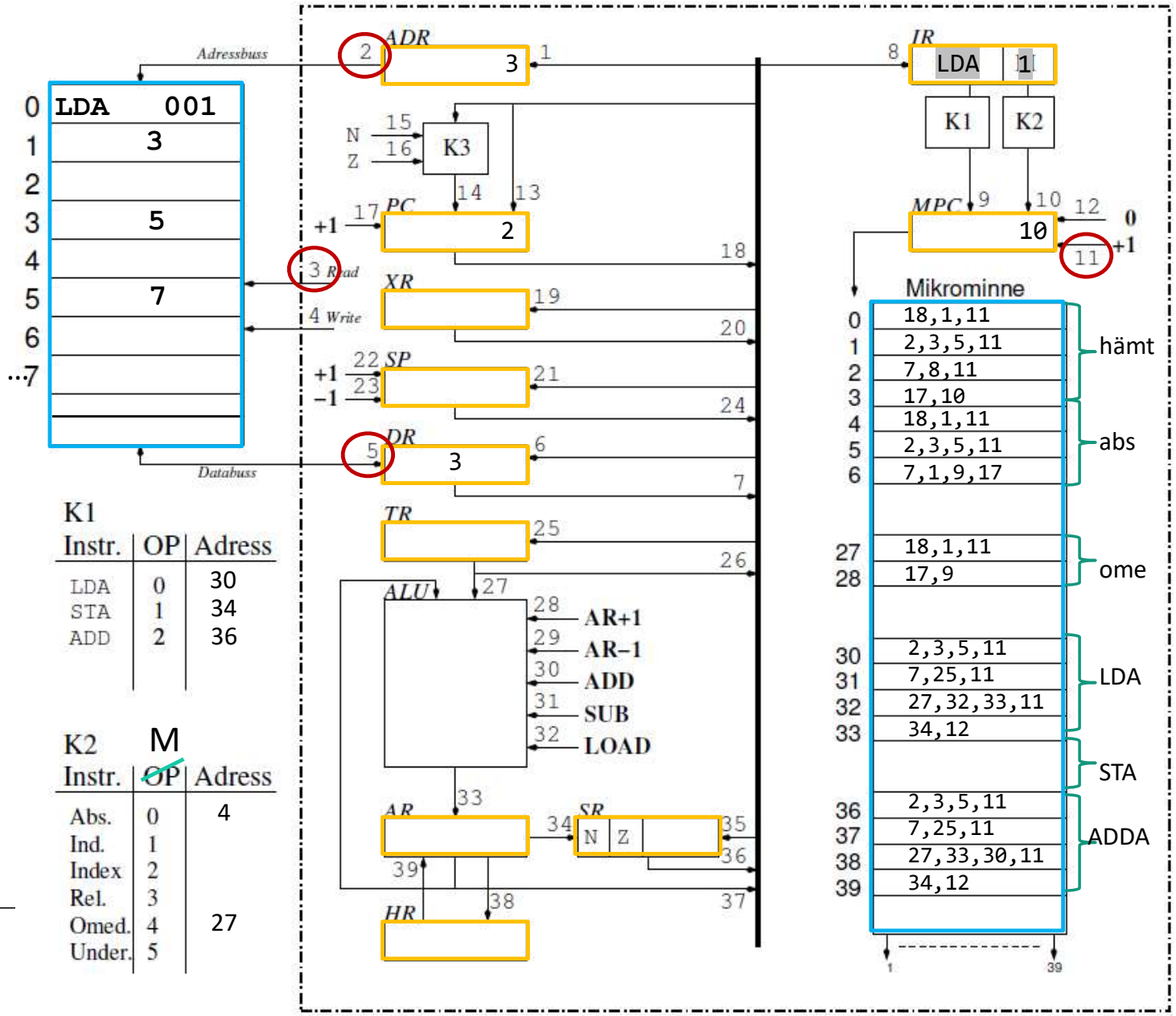
`Finns redan`

Steg 2 : A-fas, Indirekt

```
 7: PC->ADR, PC++, MPC++
 8: M->DR, MPC++
 9: DR->ADR, MPC++
10: M->DR, MPC++
11: DR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för LDA (3)

`M(M(3)) -> AR`

Steg 1 : H-fas

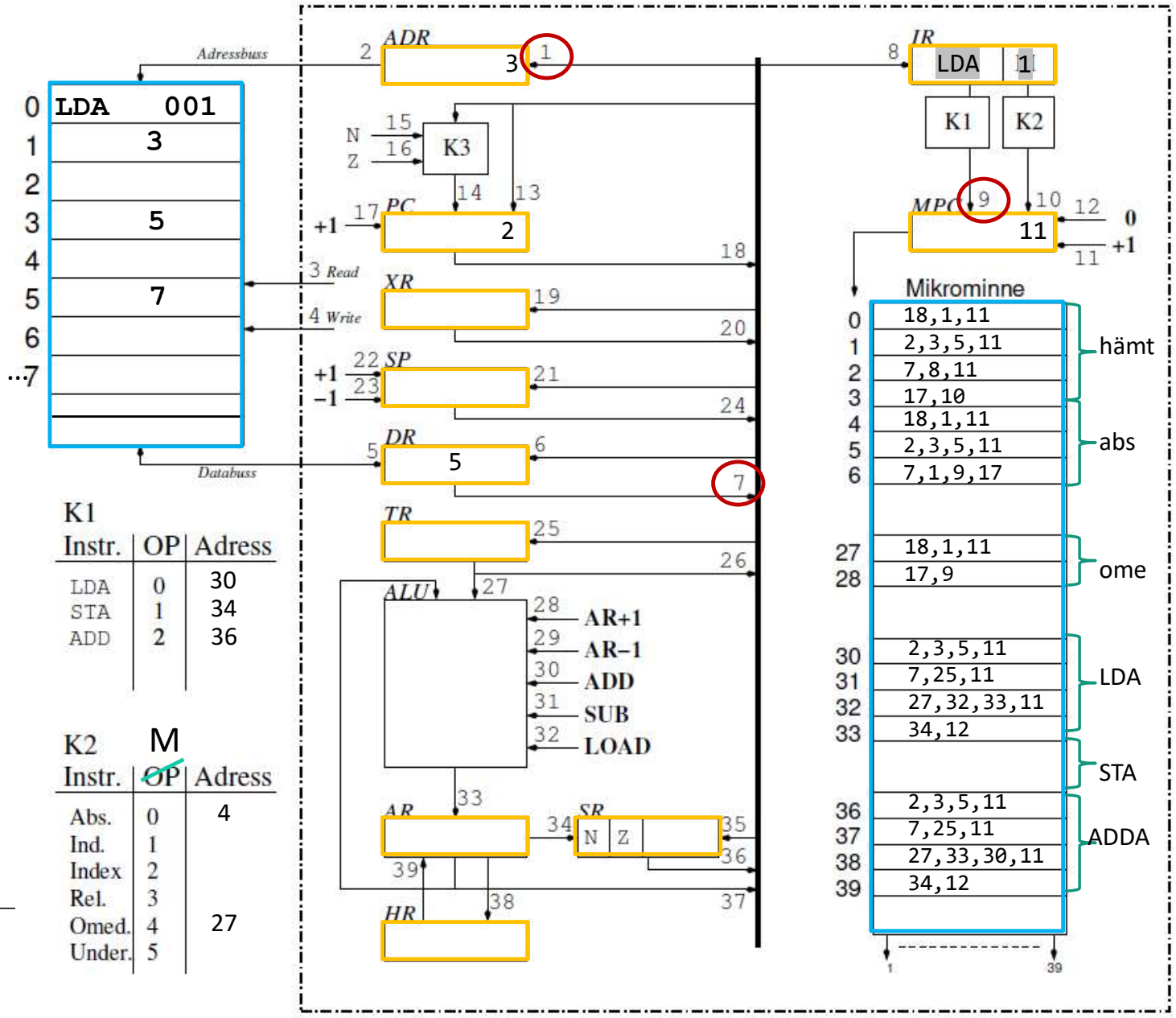
`Finns redan`

Steg 2 : A-fas, Indirekt

```
 7: PC->ADR, PC++, MPC++
 8: M->DR, MPC++
 9: DR->ADR, MPC++
10: M->DR, MPC++
11: DR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Mikrokod för LDA (3)

`M(M(3)) -> AR`

Steg 1 : H-fas

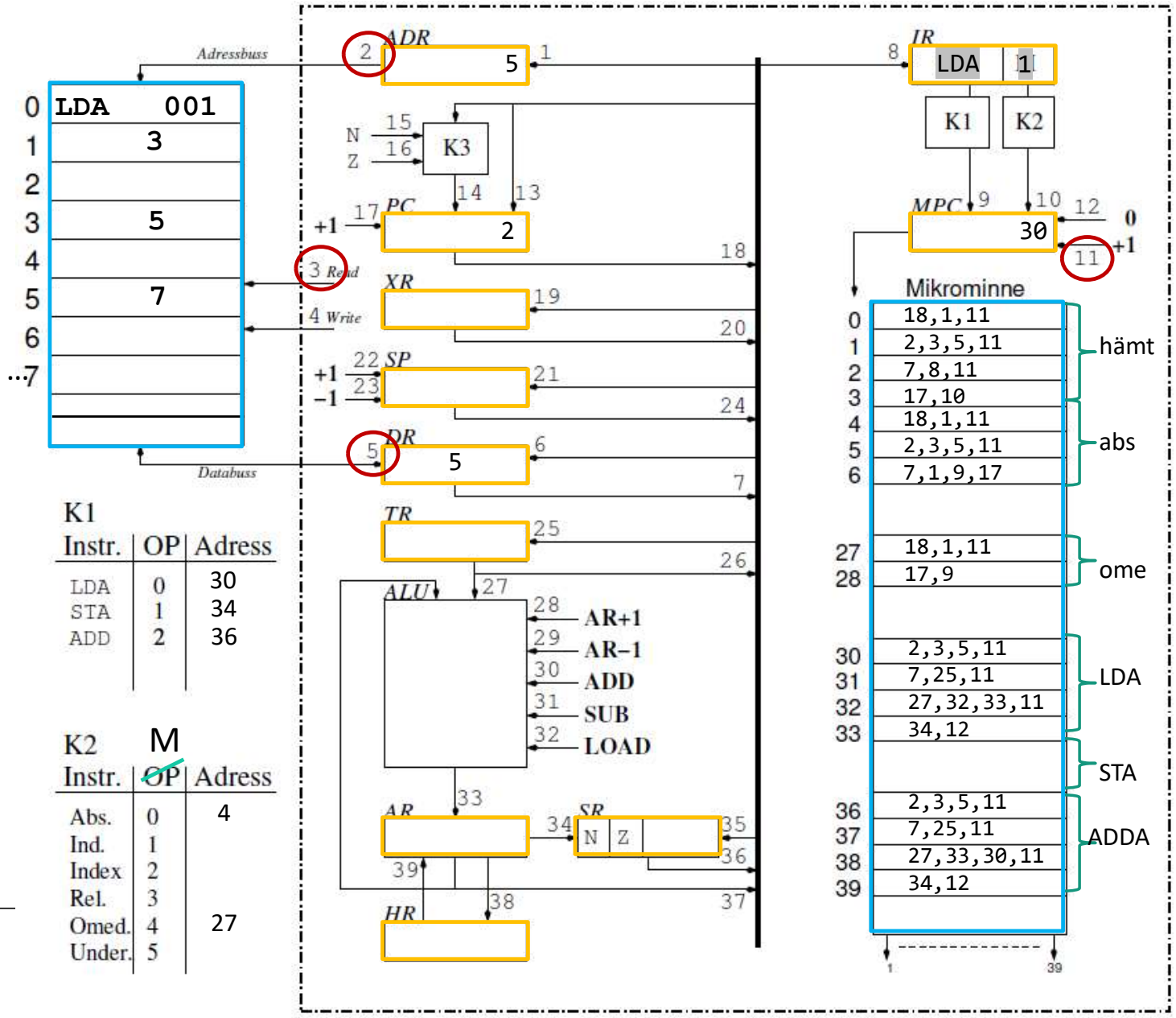
`Finns redan`

Steg 2 : A-fas, Indirekt

```
 7: PC->ADR, PC++, MPC++
 8: M->DR, MPC++
 9: DR->ADR, MPC++
10: M->DR, MPC++
11: DR->ADR, K1->MPC
```

Steg 3 : Exe, LDA

`Finns redan`



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	

Anders Nilsson

www.liu.se

LINKÖPINGS UNIVERSITET