SOCS

Hoofdstuk 1
Computerarchitectuur

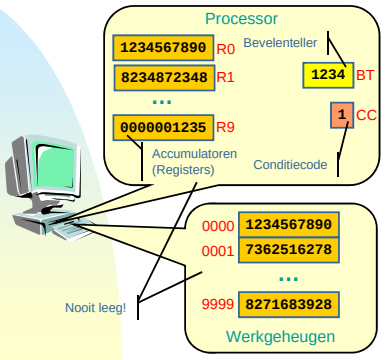
1

Inhoud

1. Basisstructuur
2. Inleiding tot C
3. Modelcomputer DRAMA
4. Programma´s voor DRAMA

2

DRAMA-computer



3

DRAMA: Bevelenset

- Bevel = (meestal) 3 delen

  FUNCODE ACC,OPERAND

  - Functiecode (3 letters): vb. HIA
    - Eventueel interpretatieveld: vb. HIA.w
  - Naam van Accumulator: vb. R0
  - Operand:
    - Geheugenadres: vb. 1209
    - Geheel getal vb. –15
    - Naam van Accumulator vb. R7

4

DRAMA: Bevelenset

- Reeds gezien:
  - HIA
  - BIG
  - OPT, AFT, VER, DEL, MOD
  - LEZ, DRU, DRS

5

DRAMA: Bevelenset

- Conditieregister
  - Resultaat van vorige bewerking
    0 (==0), 1 (>0), 2 (<0)
  - Alle transportbevelen, rekenkundige bevelen, in/uitvoer, vergelijk-bevel

  - VGL = Vergelijk *Kladregister*
    - VGL R7,123 | KR ← R7 – GeheugenReg[123]
    - VGL.w R6,10 | KR ← R6 – 10
    - VGL R8,R7 | KR ← R8 – R7

6

DRAMA: Bevelenset

- Organisatorische bevelen
  - Uitvoeringsvolgorde, …
  - SPR = onvoorwaardelijke Sprong
    - SPR 345 | Ga verder bij bevel met volgnr 345
  - VSP = Voorwaardelijke Sprong
    - VSP vw,123 | Ga verder bij bevel met volgnr 123 als vw voldaan | zoniet: volgend bevel
  - STP = Stop
    - STP | Stop uitvoering

7

DRAMA: Bevelenset

- Organisatorische bevelen
  - Voorwaarden

Voorwaarde		Voldaan als CC
NUL		0
POS		1
NEG		2
NNUL		
NPOS		
NNEG		

8

DRAMA: Bevelenset

- Organisatorische bevelen
  - Voorwaarden

Voorwaarde		Voldaan als CC
NUL		0
POS		1
NEG		2
NNUL		1 of 2
NPOS		0 of 2
NNEG		0 of 1

9

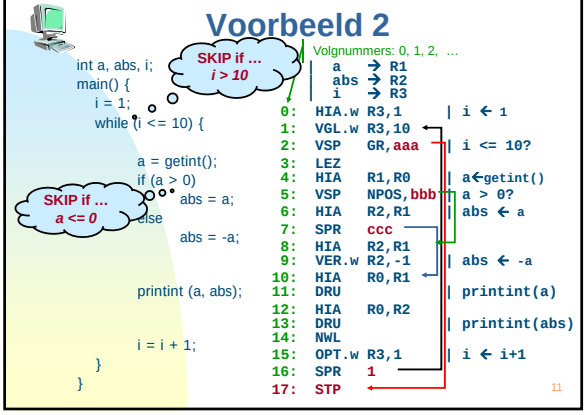
DRAMA: Bevelenset

- Organisatorische bevelen
  - Voorwaarden

Voorwaarde	Voorwaarde	Voldaan als CC
NUL	GEL	0
POS	GR	1
NEG	KL	2
NNUL	NGEL	1 of 2
NPOS	KLG	0 of 2
NNEG	GRG	0 of 1

10

Voorbeeld 2



11

Voorbeeld 2

12

Volgordebesturing

```
int a;
main() {
   a = getint();
   while (a != 0) {
         if (a < 0)
               a = -a;
         printint (a);
         a = getint();
   }
}
```

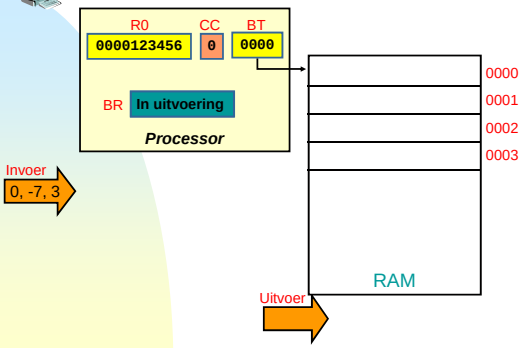
volgnummer = geheugenadres

```
|  a   → R0
0:  LEZ
1:  VSP   NUL,7
2:  VSP   NNEG,4
3:  VER.w R0,-1
4:  DRU
5:  LEZ
6:  SPR   1
7:  STP
```

RAM

13

Volgordebesturing



14

Volgordebesturing

- Naar simulatie …

15

Volgordebesturing

- Uitvoering van een programma:
  - Opeenvolging van bevelencycli:
    - Ophalen van het volgende bevel (BT)
      BT ← BT + 1
    - Analyse van het bevel
    - Uitvoeren van het bevel
      SPR (VSP) → BT nieuwe waarde

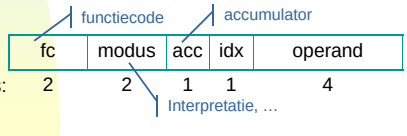
16

Interne voorstelling van bevelen

- Centraal geheugen → getallen
  - Bevel → getalvorm
  - Uitwendige machinetaal

    OPT.w R2,1012
  - Inwendige machinetaal

    2111201012



17

Interne voorstelling van bevelen

- Functiecodes

Bevel	Functiecode
HIA	11
BIG	12
OPT	21
AFT	22
VER	23
…	…

18

Interne voorstelling van bevelen

- Moduscijfers

fc	modus	acc	idx	operand

- Afhankelijk van de interpretatie
- Verzuimwaarde: .d *(behalve bij reg-reg operaties)*
  `HIA R1,123` → `HIA.d R1,123`

Inter-pretatie	Eerste modus-cijfer	Voor bevelen:
.w	1	Alle bevelen (als .w mogelijk)
.d	3	HIA, OPT, AFT, …, MOD, VGL
.d	2	BIG, SPR, VSP

19

Interne voorstelling van bevelen

- Accumulatorcijfer

fc	modus	acc	idx	operand

- R0 → 0, R1 → 1, …
- Voor VSP wordt daar de voorwaarde bijgehouden

Voorwaarde	Voorstelling
NUL	1
NNUL	8
POS	6
NPOS	2
NEG	7
…	…

20

Interne voorstelling van bevelen

- Index-veld

fc	modus	acc	idx	operand

- Register operand
- Voorbeeld:

```
HIA R1,R2
```

21

Interne voorstelling van bevelen

- Operandveld

fc	modus	acc	idx	operand

- Adres: vier cijfers van het adres
- Constante (.w): 10-complement
  0 → 4999 `0000` → `4999`
  -5000 → -1 `5000` → `9999`
- Register: NIET IN OPERAND-VELD (wel in idx-veld)

22

Interne voorstelling van de bevelen

`LEZ`	`7199999999`
`VSP NUL,7`	`3321100007`
`VSP NNEG,4`	`3321200004`
`VER.w R0,-1`	`2311009999`
`DRU`	`7299999999`
`LEZ`	`7199999999`
`SPR 1`	`3221900001`
`STP`	`9999999999`

23

C Arrays

```
int register[10];
...
/* R4 ← 1234 */
register[4] = 1234;
/* R2 ← R2 * R6 */
register[2] *= register[6];
```

Declaratie
- `int register[10];`
- geeft 10 elementen met indices 0 .. 9
- type van elk element = int

Commentaar
- `/* … */`
- `// …`

0 1 2 3 4 5 6 7 8 9

Selectie
- register[4]
- register[idx]
- register[2 * idx + 4]

24

Volgordebesturing

- C-programma dat volgordebesturing simuleert
  - Variabelen van de processor:

```
int BT;            /* bevelenteller */
int stop;          /* logische waarde */
int BR;            /* bevelenregister */
int CC;            /* conditiecode */
int fccode,modus,acc,idx,operand; /*velden*/
int waarde;        /* hulpregister */
int registers[10]; /* rij van 10 registers */
```

  - Geheugen:

```
int geheugen[10000]; /* 10.000 geheugenregisters */
```

  - Functies:
    - `expandeer(getal)`: 4 cijfers → 10 cijfers
      4999 → 0000004999 5800 → 9999995800
    - `teken(getal)`: geeft 0, 1, 2 terug als het getal nul, pos. of neg is

25

Volgordebesturing

```
BT = 0;
stop = 0;
while (! stop) {
  /* haal bevel op */
  BR = geheugen[BT];
  BT = BT + 1;
  /* analyseer bevel */
  fccode = BR / 100000000;
  modus = (BR%100000000)
          /1000000;
  acc = …;
  idx = …;
  operand = expandeer(BR
      % 10000);
```

```
/* voer bevel uit */
switch (fccode) {
 case 11: /* HIA */
  if (modus_regOpd(modus))
    waarde = registers[idx];
  else {
    if (modus_adres (modus))
      waarde = geheugen[
        operand % 10000];
    else
      waarde = operand;
  }
  registers[acc] = waarde;
  CC = teken(registers[acc])
  break;
```

26

Volgordebesturing

```
BT = 0;
stop = 0;
while (! stop) {
  /* haal bevel op */
  BR = geheugen[BT];
  BT = BT + 1;
  /* analyseer bevel */
  fccode = BR / 100000000;
  modus = (BR%100000000)
          /1000000;
  acc = …;
  idx = …;
  operand = expandeer(BR
      % 10000);
```

```
/* voer bevel uit */
switch (fccode) {
 case 11: /* HIA */
 case 12: /* BIG */
  geheugen[operand%10000] =
        registers[acc];
  CC = teken(registers[acc])
 break;
```

27

Volgordebesturing

```
BT = 0;
stop = 0;
while (! stop) {
  /* haal bevel op */
  BR = geheugen[BT];
  BT = BT + 1;
  /* analyseer bevel */
  fccode = BR / 100000000;
  modus = (BR%100000000)
          /1000000;
  acc = …;
  idx = …;
  operand = expandeer(BR
      % 10000);
```

```
/* voer bevel uit */
switch (fccode) {
 case 11: /* HIA */
 case 12: /* BIG */
 case 21: /* OPT */
  if (modus_regOpd(modus))
    waarde = registers[idx];
  else {
    if (modus_adres (modus))
      waarde = geheugen[
        operand % 10000];
    else
      waarde = operand;
  }
  registers[acc] += waarde;
  CC = teken(registers[acc])
  break;
```

28

Volgordebesturing

```
BT = 0;
stop = 0;
while (! stop) {
  /* haal bevel op */
  BR = geheugen[BT];
  BT = BT + 1;
  /* analyseer bevel */
  fccode = BR / 1000000000;
  modus = (BR%100000000)
          /1000000;
  acc = …;
  idx = …;
  operand = expandeer(BR
      % 10000);
```

```
/* voer bevel uit */
switch (fccode) {
 case 11: /* HIA */
 case 12: /* BIG */
 case 21: /* OPT */
 …
 case 32: /* SPR */
  BT = operand % 10000;
  break;
 case 99: /* STP */
  stop = 1;
 }
}
```

29

Assembleertaal

- Nadelen van uitwendige machinetaal:
  - Programmeur moet bevelen tellen 
  - Tussenvoegen bevelen: adressen aanpassen
  - Programmeur moet zelf geheugenplaatsen toekennen aan variabelen

30

Volgordebesturing

```
int a;
main() {
   a = getint();
   while (a != 0) {
         if (a < 0)
               a = -a;
         a += 10;
         printint (a);
         a = getint();
   }
}
```

```
|  a   → R0

0:  LEZ
1:  VSP   NUL,8
2:  VSP   NNEG,4
3:  VER.w R0,-1
4:  OPT.w R0,10
5:  DRU
6:  LEZ
7:  SPR   1
8:  STP
```

31

Assembleertaal

- Oplossing:
  - Symbolische adressen
  - Etiket (symbolisch adres) → DEFINITIE
    `NAAM:` …
  - Verwijzing (als operand) → REFERENTIE
    … `NAAM`

32

Assembleertaal

- Bevelen:

```
lus:   HIA   R2,123
       …
       SPR   lus
       …
       VSP   GEL,end
       …
       …
end:   HIA   R1,123
```

Achterwaartse referentie

Voorwaartse referentie

- Variabelen:
  - **RESGR** = Reserveer GeheugenRegister
  - ≠ machinebevel
  - = vertalerdirectief

```
       HIA R2,a
       …
a:     RESGR 1
```

geheugen-registers

33

Assembleertaal

- Constante:
  - Niet behorend tot interval [-5000,4999]

```
       …
miljoen: 1000000
       …
       OPT R1,miljoen
```

34

Voorbeeld

```
int a;
main() {
   a = getint();
   while (a != 0) {
         if (a < 0)
               a = -a;
         printint (a);
         a = getint();
   }
}
```

SKIP if *a == 0*

SKIP if *a >= 0*

```
       | a in geheugen/R0
       LEZ
       BIG   R0,a
while: VSP   NUL,endwh
       VSP   NNEG,endif
       VER.w R0,-1
       BIG   R0,a
endif: DRU
       LEZ
       BIG   R0,a
       SPR   while
endwh: STP
a:     RESGR 1
       EINDPR
```

Vertaler-directief: einde programma

35

Van C naar DRAMA

- Omzetting door vertaler
  - Correct algoritme (in hogere programmeertaal: C)
  - Omzetting C → DRAMA
  - Optimalisatie in laatste instantie

36

Van C naar DRAMA

- Variabelen
  - In geheugen
    `int a, b, c;` →
    ```
    a: RESGR 1
    b: RESGR 1
    c: RESGR 1
    ```
  - In registers
    - slechts een beperkt aantal

37

Van C naar DRAMA

- Toekenning
  - `a = b;` →
    ```
    HIA Rr,b
    BIG Rr,a
    ```
  - `a = exp;` → *… code voor exp in Rr …*
    ```
    BIG Rr,a
    ```
  - In registers
    - slechts een beperkt aantal

38

C labels en goto …

```
init: a = 1;
test: while (a > 10) {
   ...
}
goto test;
```

Label/Etiket
- symbolische naam gevolgd door :
- mag voor elke C-opdracht

goto

goto XXX;
- ga verder bij opdracht met label XXX

39

Van C naar DRAMA

- If-opdracht

SKIP if *! voorwaarde*

```
if (voorwaarde)
      S1;
else
      S2;
```

→

```
if (! voorwaarde)
        goto elsd;
      S1;
      goto naif;
elsd: S2;
naif: …
```

```
      VSP niet-voorwaarde,elsd
      … code voor S1 …
      SPR naif
elsd:
      … code voor S2 …
naif: …
```

40

Van C naar DRAMA

- While-opdracht

```
while (voorwaarde)
      S;
```

→

```
test:  if (! voorwaarde)
           goto endwh;
       S;
       goto test;
endwh: …
```

SKIP if *! voorwaarde*

```
test:  VSP niet-voorwaarde,endwh
       … code voor S …
       SPR test
endwh: …
```

41

Samenvatting

- Computer → Logische bouwstenen
  - CPU, RAM, in-/uitvoerorganen, hulpgeheugens
  - Eenvoudige instructies
    - Inwendige (machinetaal) vs Uitwendige (LPT) taal
    - Complexe programma's
  - Controleorgaan (CPU)
    - Volgordebesturing
- Assembleertaal → Ondersteuning programmeur
  - Symbolische adressen
  - Compilator
    - Hogere programmeertaal → Lagere programmeertaal

42

Opgave

Gegeven: Labels A en B verwijzen naar 2 geheugenplaatsen

Gevraagd: Schrijf een programma dat de inhoud van beide omwisselt, terwijl alleen R0 gebruikt wordt en geen enkele andere geheugenplaats

```
HIA R0,A          HIA R1,A
BIG R0,tmp        HIA R0,B
HIA R0,B          BIG R0,A
BIG R0,A          BIG R1,B
HIA R0,tmp
BIG R0,B
```

Tip: x en y
x-y en y → x, y

43

Cursustekst

- Hoofdstuk 1: pag. 1 → pag. 47

44

Volgordebesturing

R0: 0000123456 | CC: 0 | BT: 0000

BR:

Processor

Invoer: 0, -7, 3

```
    |  a    → R0
0:  LEZ
1:  VSP   NUL,7
2:  VSP   NNEG,4
3:  VER.w R0,-1
4:  DRU
5:  LEZ
6:  SPR   1
7:  STP
```

Uitvoer

45

Volgordebesturing

R0: 0000123456 | CC: 0 | BT: 0000

BR: LEZ

Processor

Invoer: 0, -7, 3

```
    |  a    → R0
0:  LEZ
1:  VSP   NUL,7
2:  VSP   NNEG,4
3:  VER.w R0,-1
4:  DRU
5:  LEZ
6:  SPR   1
7:  STP
```

Uitvoer

46

Volgordebesturing

R0: 0000123456 | CC: 0 | BT: 0001

BR: LEZ

Processor

Invoer: 0, -7, 3

```
    |  a    → R0
0:  LEZ
1:  VSP   NUL,7
2:  VSP   NNEG,4
3:  VER.w R0,-1
4:  DRU
5:  LEZ
6:  SPR   1
7:  STP
```

Uitvoer

47

Volgordebesturing

R0: 0000000003 | CC: 1 | BT: 0001

BR: LEZ

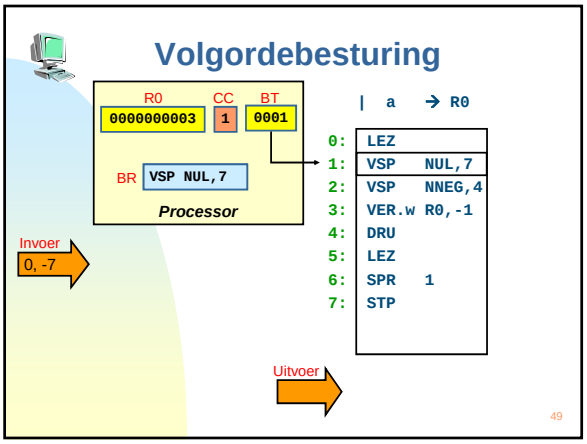
Processor

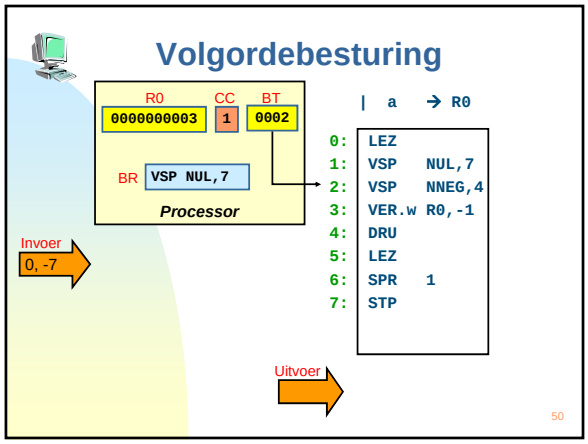
Invoer: 0, -7

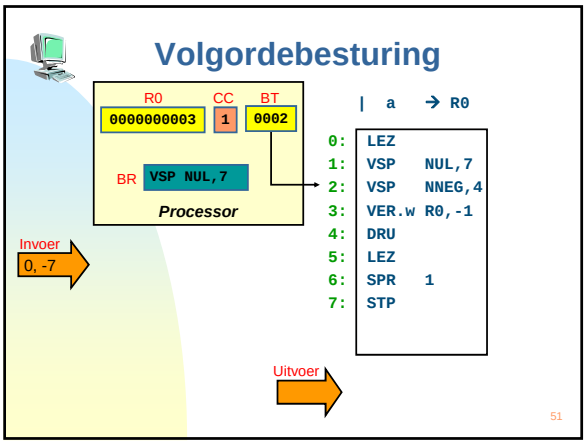
```
    |  a    → R0
0:  LEZ
1:  VSP   NUL,7
2:  VSP   NNEG,4
3:  VER.w R0,-1
4:  DRU
5:  LEZ
6:  SPR   1
7:  STP
```

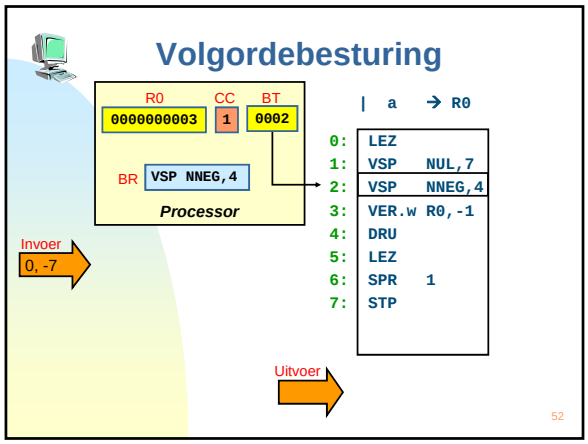
Uitvoer

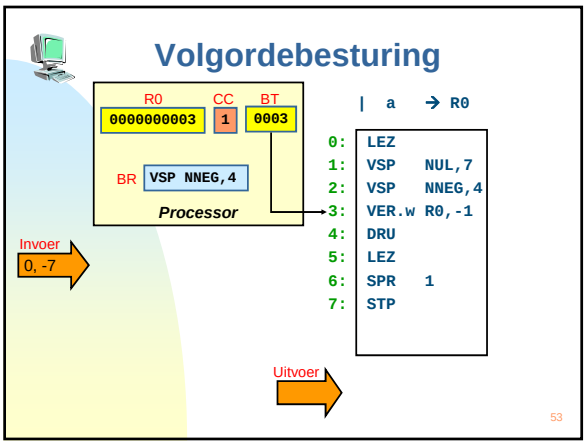
48

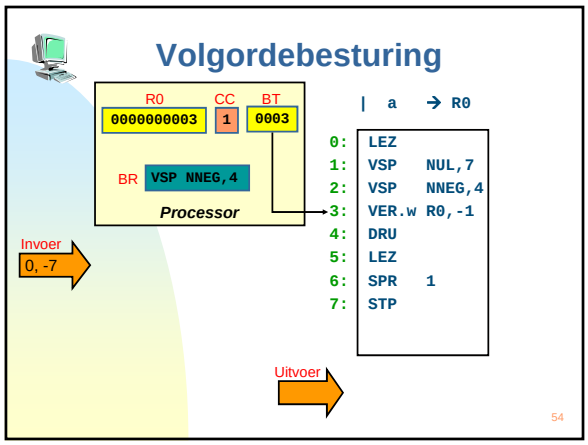

Volgordebesturing
R0
0000000003
CC
1
BT
0001
BR
VSP NUL,7
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
49

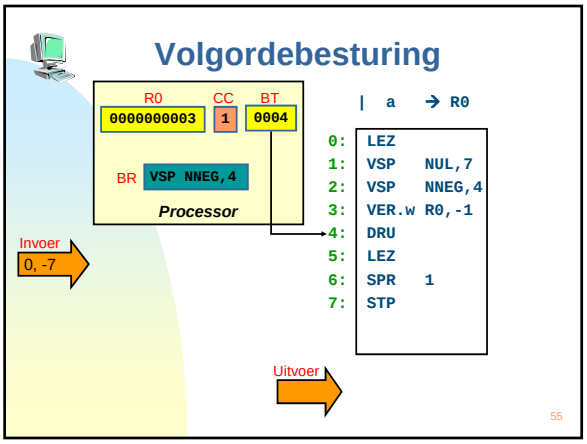

Volgordebesturing
R0
0000000003
CC
1
BT
0002
BR
VSP NUL,7
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
50

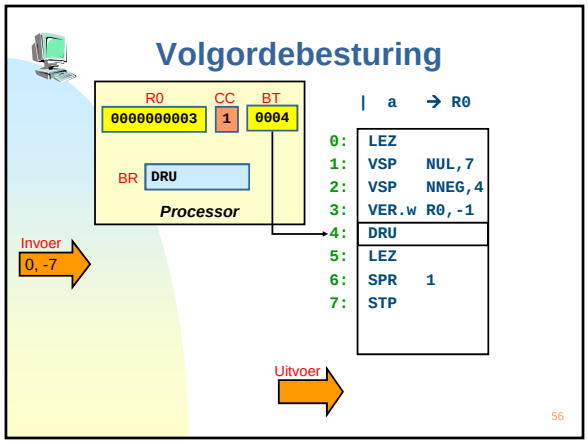

Volgordebesturing
R0
0000000003
CC
1
BT
0002
BR
VSP NUL,7
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
51

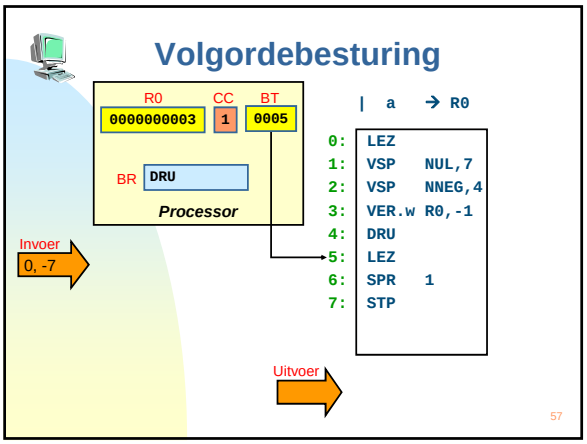

Volgordebesturing
R0
0000000003
CC
1
BT
0002
BR
VSP NNEG,4
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
52

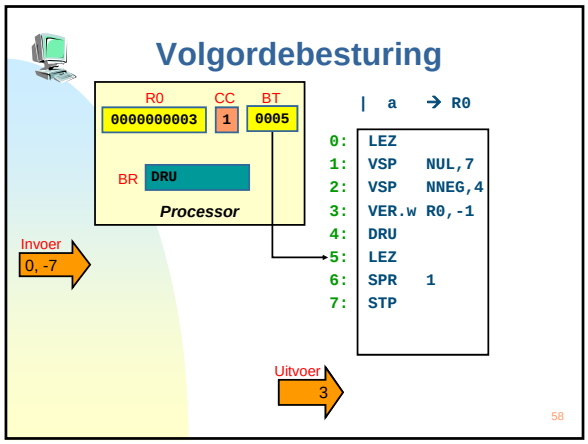

Volgordebesturing
R0
0000000003
CC
1
BT
0003
BR
VSP NNEG,4
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
53

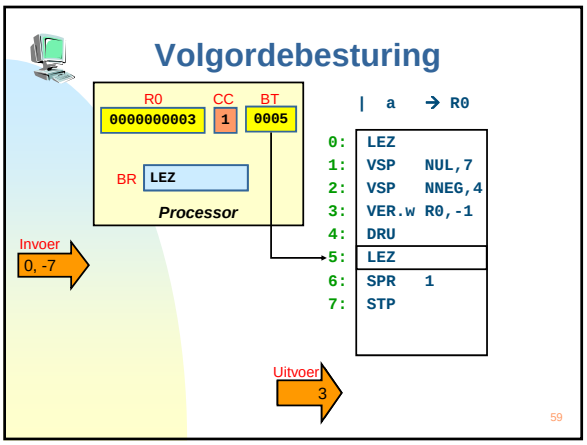

Volgordebesturing
R0
0000000003
CC
1
BT
0003
BR
VSP NNEG,4
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
54

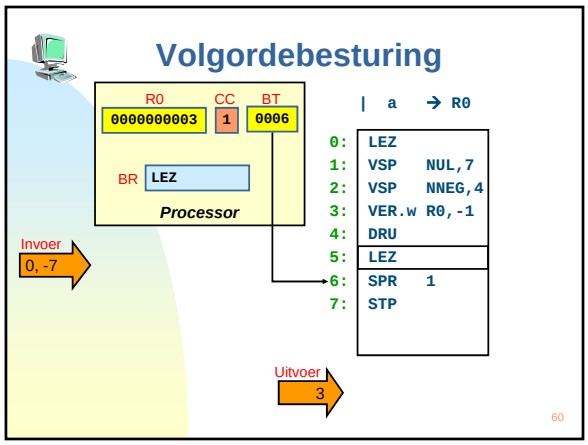

Volgordebesturing
R0
0000000003
CC
1
BT
0004
BR
VSP NNEG,4
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
55

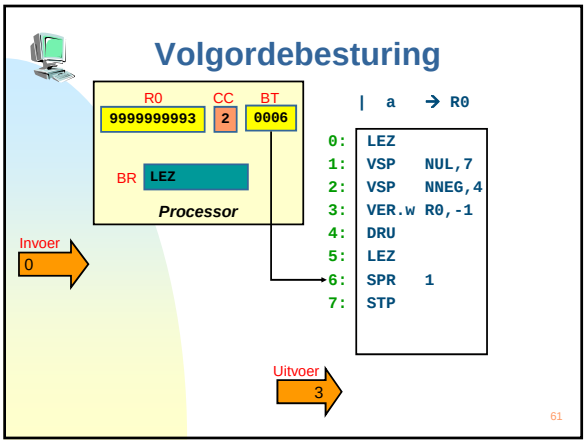

Volgordebesturing
R0
0000000003
CC
1
BT
0004
BR
DRU
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
56

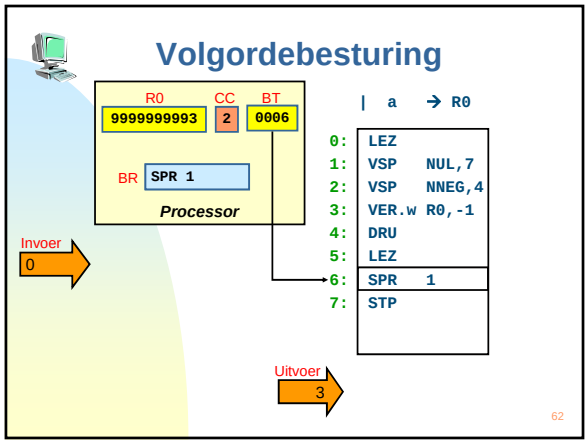

Volgordebesturing
R0
0000000003
CC
1
BT
0005
BR
DRU
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
57

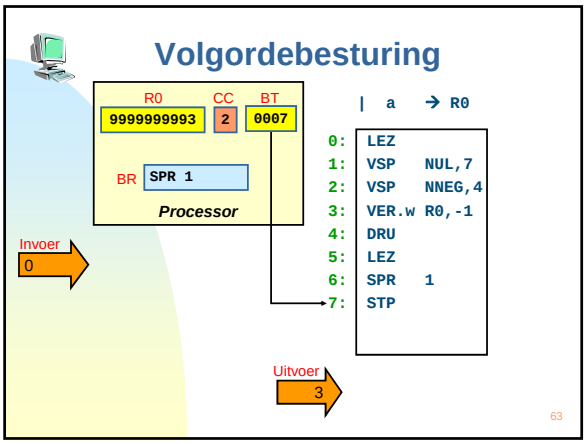

Volgordebesturing
R0
0000000003
CC
1
BT
0005
BR
DRU
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
58

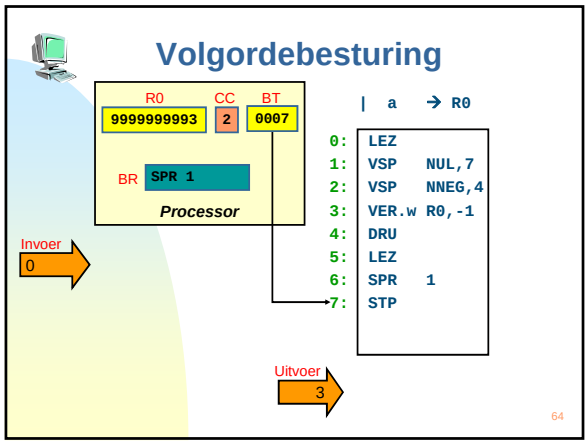

Volgordebesturing
R0
0000000003
CC
1
BT
0005
BR
LEZ
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
59

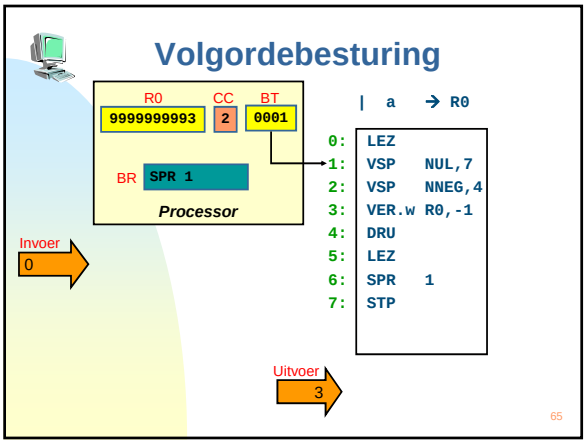

Volgordebesturing
R0
0000000003
CC
1
BT
0006
BR
LEZ
Processor
Invoer
0, -7
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
60

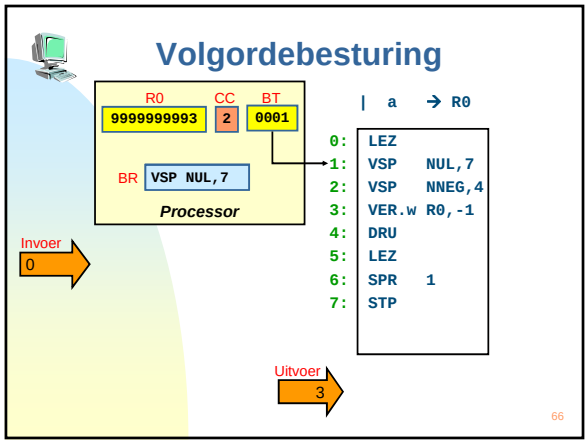

Volgordebesturing
R0
9999999993
CC
2
BT
0006
BR
LEZ
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
61

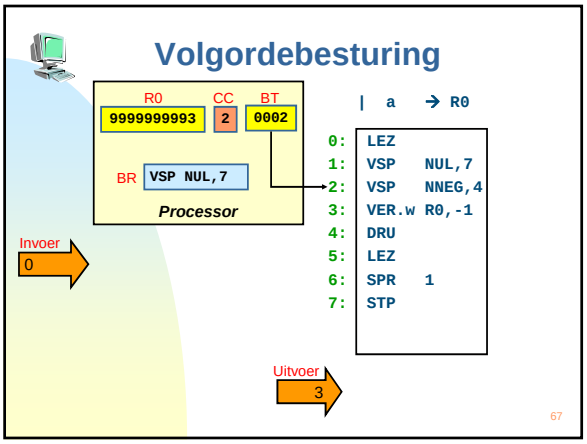

Volgordebesturing
R0
9999999993
CC
2
BT
0006
BR
SPR 1
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
62

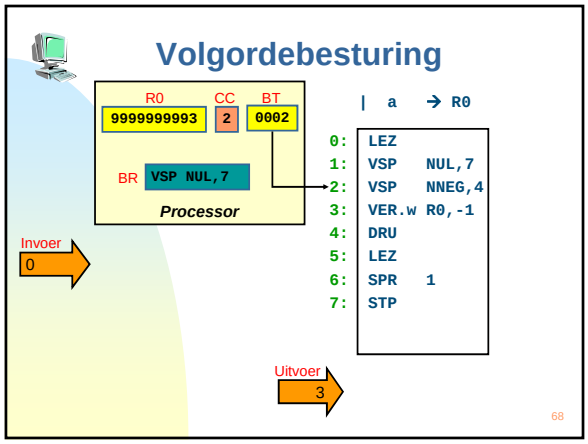

Volgordebesturing
R0
9999999993
CC
2
BT
0007
BR
SPR 1
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
63

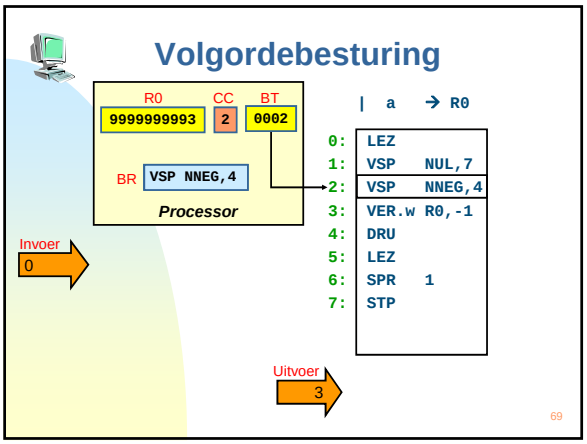

Volgordebesturing
R0
9999999993
CC
2
BT
0007
BR
SPR 1
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
64

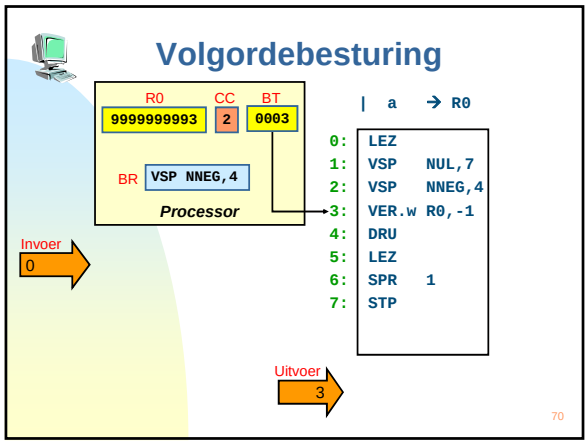

Volgordebesturing
R0
9999999993
CC
2
BT
0001
BR
SPR 1
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
65

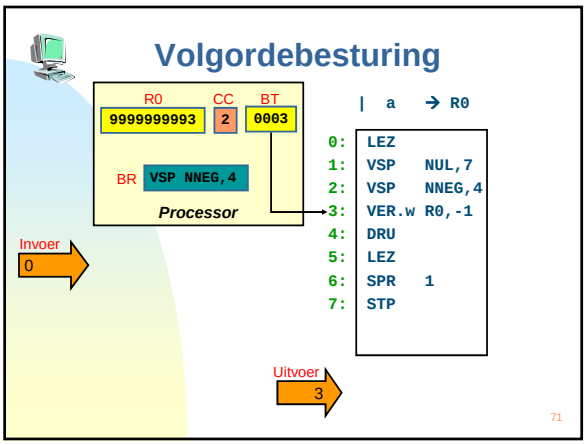

Volgordebesturing
R0
9999999993
CC
2
BT
0001
BR
VSP NUL,7
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
66

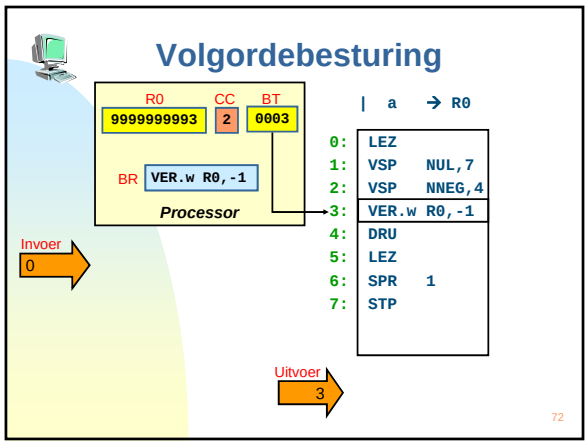

Volgordebesturing
R0
9999999993
CC
2
BT
0002
BR
VSP NUL,7
Processor
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Invoer
0
Uitvoer
3
67

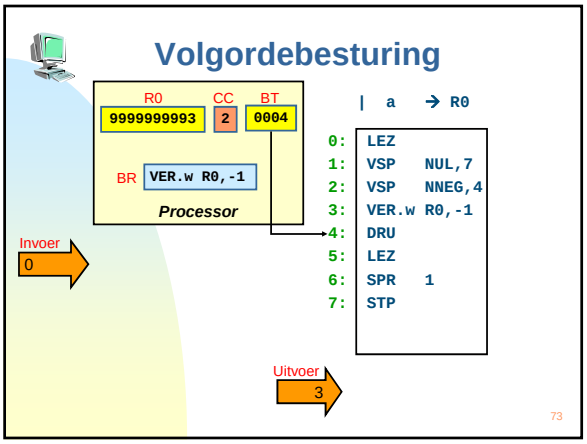

Volgordebesturing
R0
9999999993
CC
2
BT
0002
BR
VSP NUL,7
Processor
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Invoer
0
Uitvoer
3
68

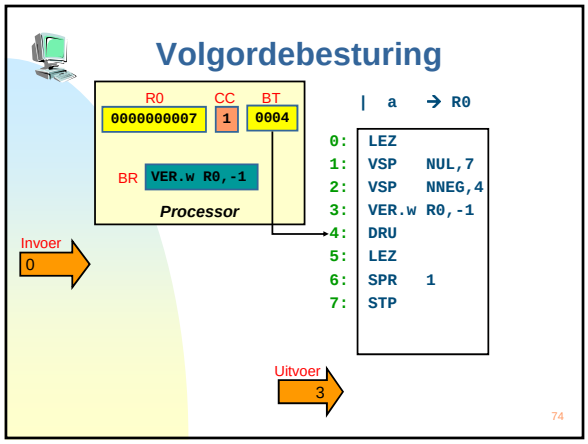

Volgordebesturing
R0
9999999993
CC
2
BT
0002
BR
VSP NNEG,4
Processor
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Invoer
0
Uitvoer
3
69

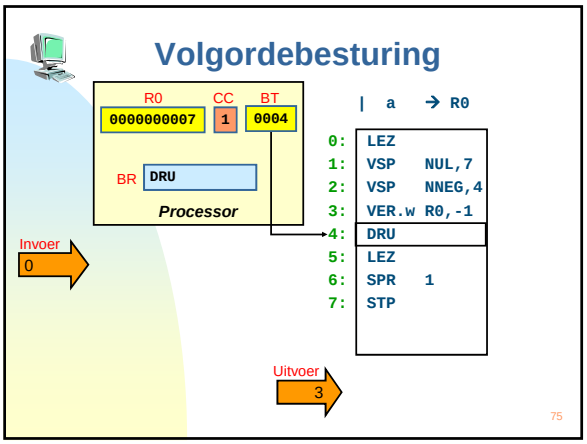

Volgordebesturing
R0
9999999993
CC
2
BT
0003
BR
VSP NNEG,4
Processor
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Invoer
0
Uitvoer
3
70

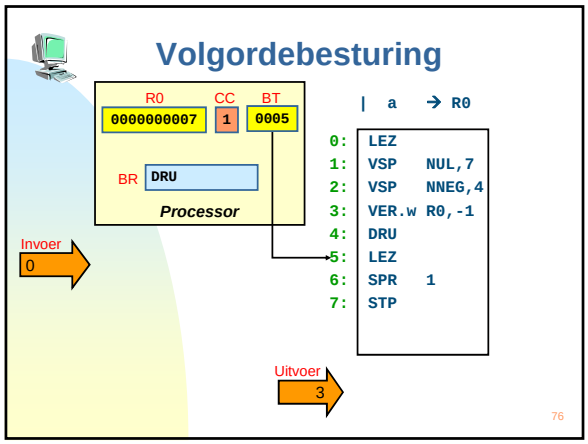

Volgordebesturing
R0
9999999993
CC
2
BT
0003
BR
VSP NNEG,4
Processor
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Invoer
0
Uitvoer
3
71

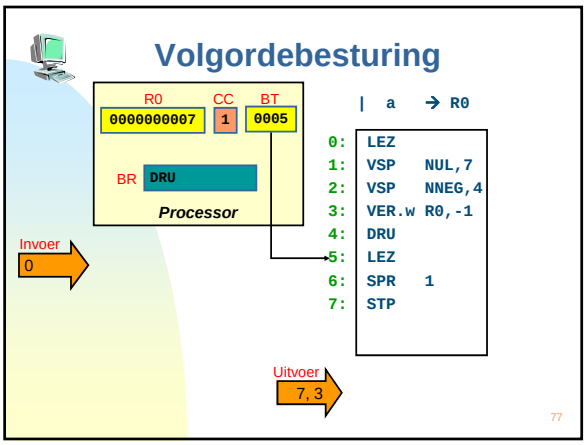

Volgordebesturing
R0
9999999993
CC
2
BT
0003
BR
VER.w R0,-1
Processor
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Invoer
0
Uitvoer
3
72

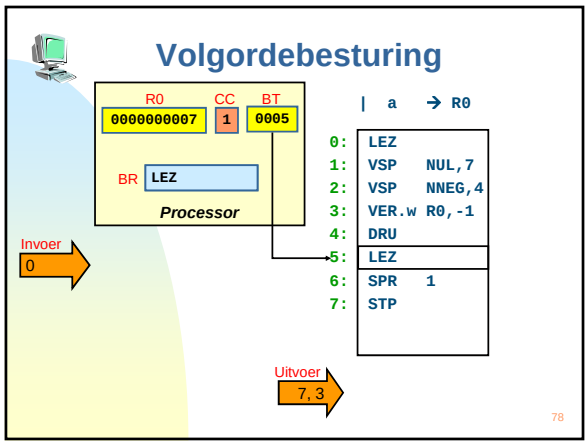

Volgordebesturing
R0
9999999993
CC
2
BT
0004
BR
VER.w R0,-1
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
73

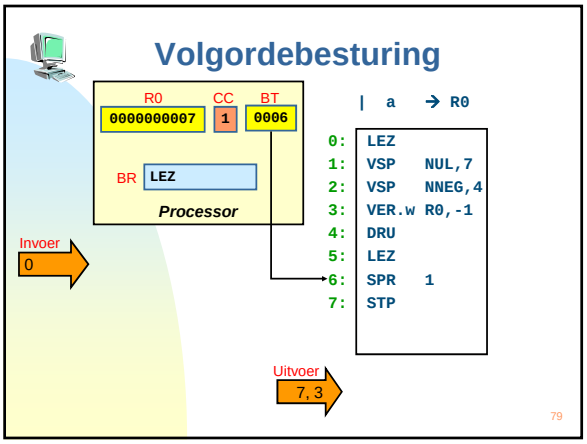

Volgordebesturing
R0
0000000007
CC
1
BT
0004
BR
VER.w R0,-1
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
74

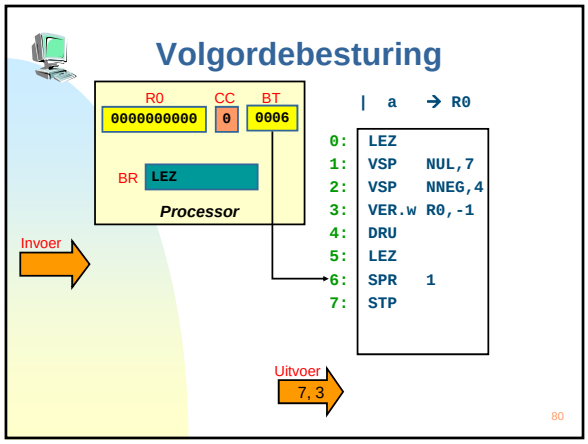

Volgordebesturing
R0
0000000007
CC
1
BT
0004
BR
DRU
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
75

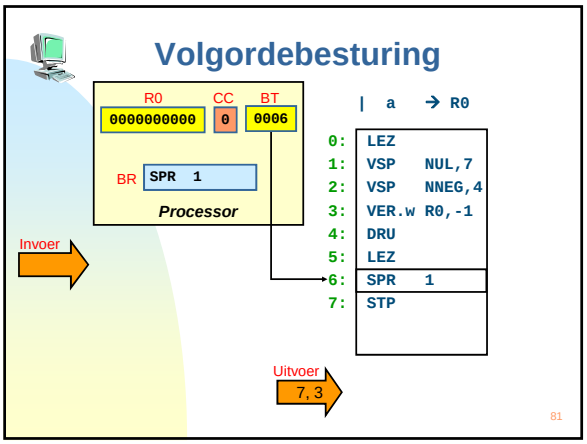

Volgordebesturing
R0
0000000007
CC
1
BT
0005
BR
DRU
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
3
76

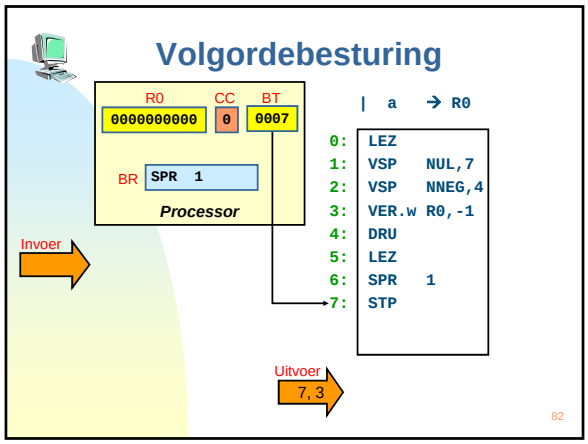

Volgordebesturing
R0
0000000007
CC
1
BT
0005
BR
DRU
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
77

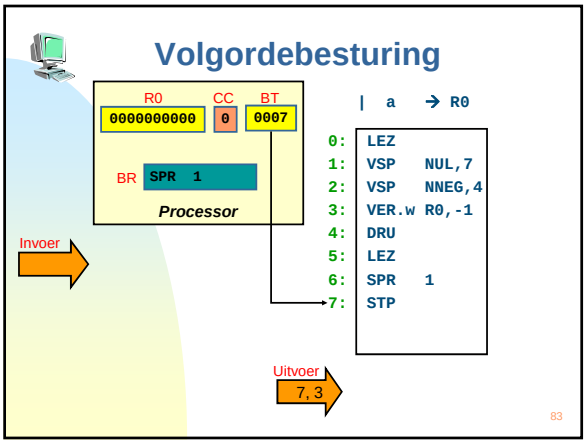

Volgordebesturing
R0
0000000007
CC
1
BT
0005
BR
LEZ
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
78

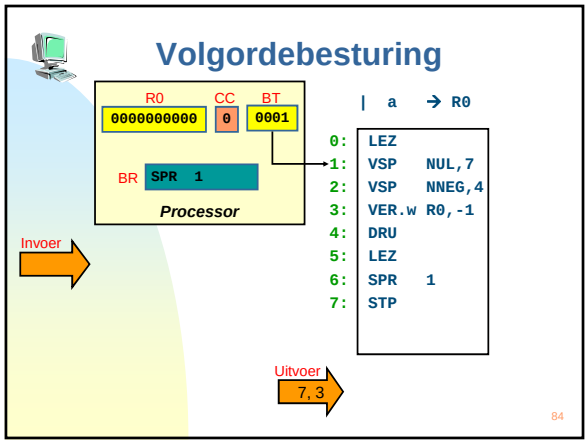

Volgordebesturing
R0
0000000007
CC
1
BT
0006
BR
LEZ
Processor
Invoer
0
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
79

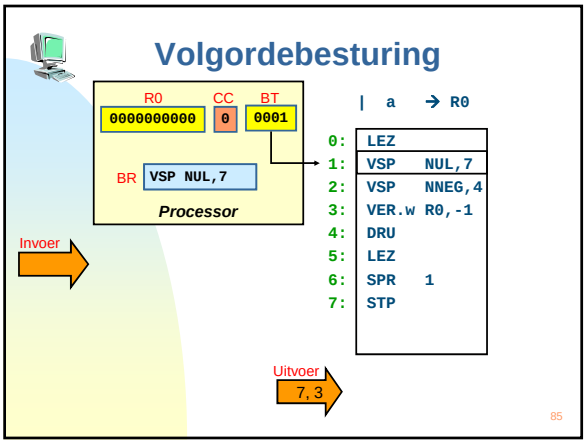

Volgordebesturing
R0
0000000000
CC
0
BT
0006
BR
LEZ
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
80

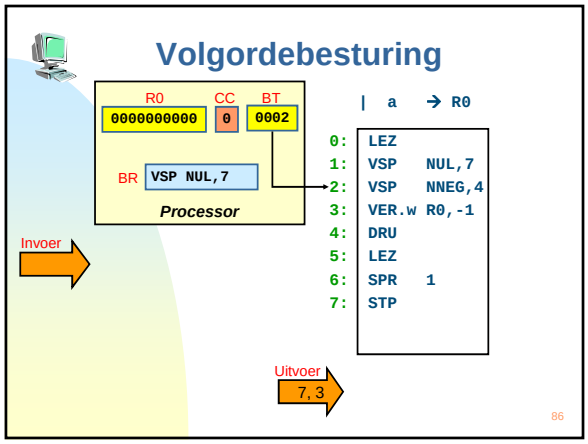

Volgordebesturing
R0
0000000000
CC
0
BT
0006
BR
SPR 1
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
81

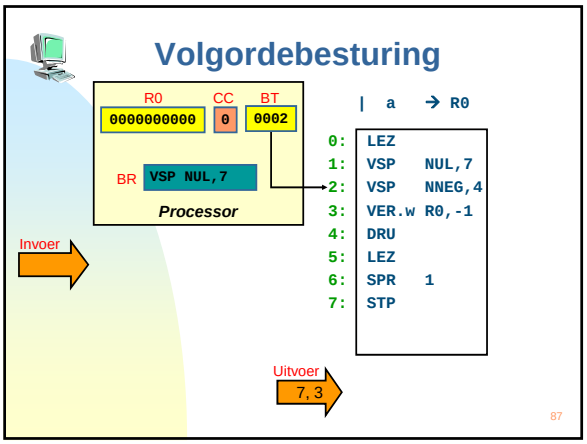

Volgordebesturing
R0
0000000000
CC
0
BT
0007
BR
SPR 1
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
82

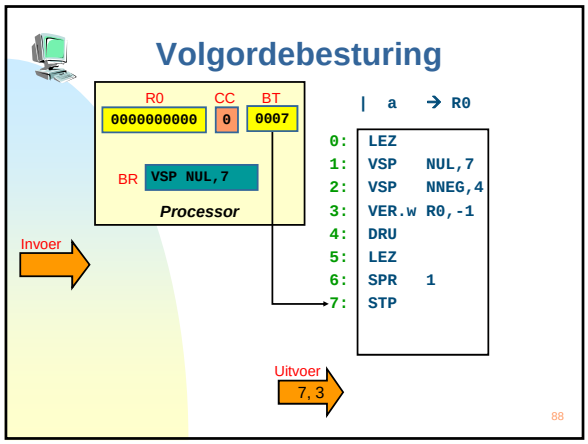

Volgordebesturing
R0
0000000000
CC
0
BT
0007
BR
SPR 1
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
83

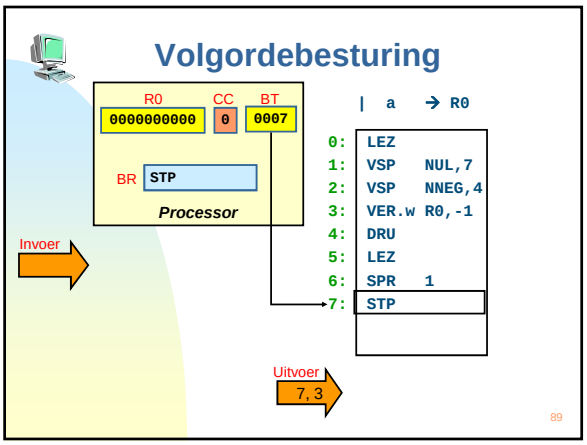

Volgordebesturing
R0
0000000000
CC
0
BT
0001
BR
SPR 1
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
84

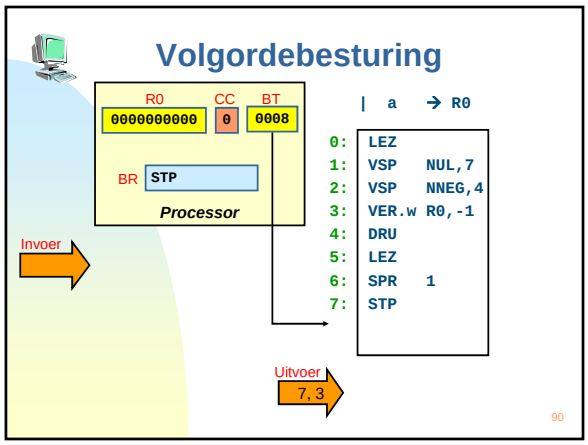

Volgordebesturing
R0 CC BT
0000000000 0 0001
BR VSP NUL,7
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
85

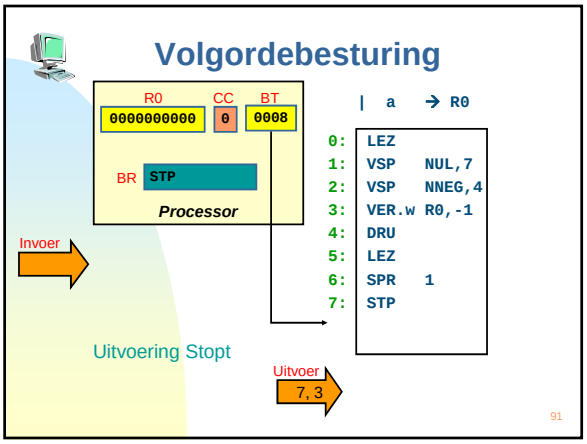

Volgordebesturing
R0 CC BT
0000000000 0 0002
BR VSP NUL,7
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
86

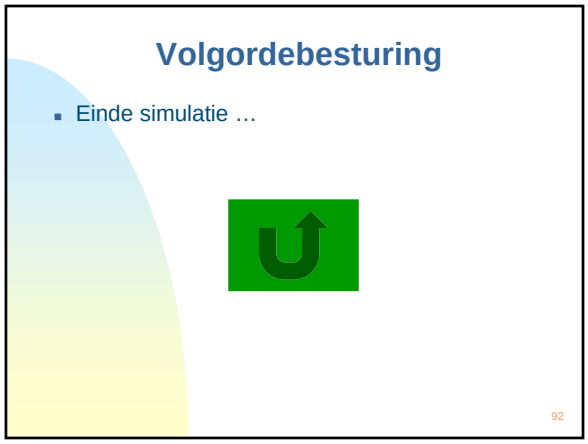

Volgordebesturing
R0 CC BT
0000000000 0 0002
BR VSP NUL,7
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
87


Volgordebesturing
R0 CC BT
0000000000 0 0007
BR VSP NUL,7
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
88


Volgordebesturing
R0 CC BT
0000000000 0 0007
BR STP
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
89


Volgordebesturing
R0 CC BT
0000000000 0 0008
BR STP
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoer
7, 3
90


Volgordebesturing
R0
CC
BT
0000000000
0
0008
BR
STP
Processor
Invoer
| a → R0
0: LEZ
1: VSP NUL,7
2: VSP NNEG,4
3: VER.w R0,-1
4: DRU
5: LEZ
6: SPR 1
7: STP
Uitvoering Stopt
Uitvoer
7, 3
91


Volgordebesturing
Einde simulatie …
92