

Computer systems 2017 -2018 Assembler

HOGESCHOOL PXL



Assembler, benodigdheden

VIRTUAL MACHINE WINDOWS 3.1 (te vinden op blackboard)



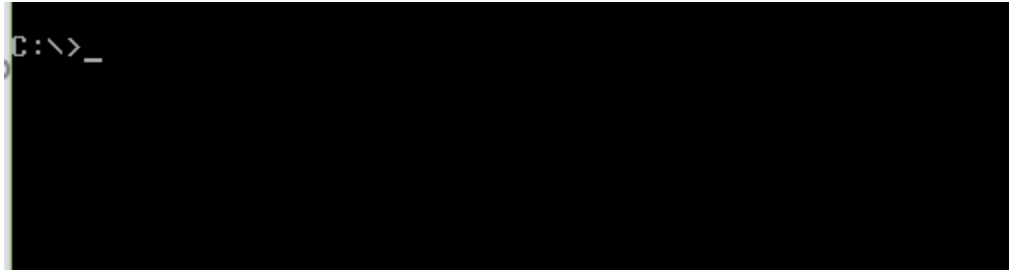
Voorbeeld VIRTUAL MACHINE WINDOWS 3.1

A screenshot of a Windows 3.1 virtual machine window running in Oracle VM VirtualBox. The window title is 'Win3.1 [Draaiend] - Oracle VM VirtualBox'. The menu bar includes 'Bestand', 'Machine', 'Weergeven', 'Invoer', 'Apparaten', and 'Hulp'. The command prompt shows the following text:

```
Starting MS-DOS...  
  
HIMEM is testing extended memory...done.  
  
Unrecognized command in CONFIG.SYS  
Error in CONFIG.SYS line 6  
  
C:\>C:\DOS\SMARTDRV.EXE /X  
C:\>_
```

Assembler, Start debug

VIRTUAL MACHINE WINDOWS 3.1 opstartscherm:



Start debug: “type debug”



“DEBUG start het programma debug!”

Assembler, De eerste instructies in DEBUG

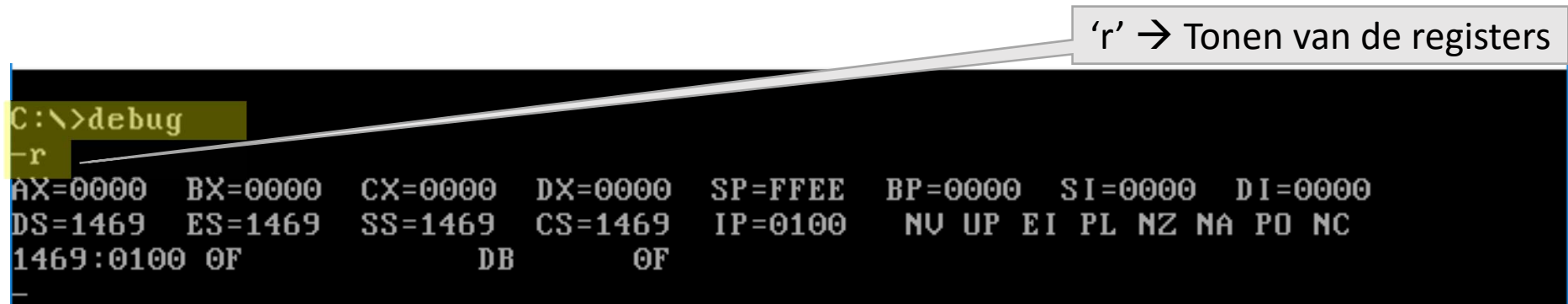
"r" = Tonen van registers

"e" = enter commando

"d" = dump commando

Assembler, De eerste instructies in DEBUG

“r” = Tonen van registers



```
C:\>debug
-r
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=1469  ES=1469  SS=1469  CS=1469  IP=0100  NV UP EI PL NZ NA PO NC
1469:0100 0F                                DB          0F
_
```

Opmerking:

- De instructie 'r' in debug toont:
 - de actuele stand van de registers.
 - de flag-registers.
 - de actuele stand van de instructiepointer
 - de volgende instructie in machinetaal
 - de volgende instructie in assembly taal
- De waarden in de registers zijn allen HEXADECIMAAL!

Assembler, De eerste instructies in DEBUG

“r” = Tonen van registers

Het register AX heeft
momenteel de waarde
'0000'

```
C:\>debug
```

```
-r
```

```
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000  
DS=1469  ES=1469  SS=1469  CS=1469  IP=0100  NV UP EI PL NZ NA PO NC  
1469:0100 0F DB 0F
```

De actuele waarde van de
instructiepointer is
1469:0100

De 'data' op adres
1469:0100 is 0F
Dit is in machinetaal de
volgende instructie.
(=gegevens op het
adres van de
instructiepointer)

De actuele waarde van de
instructie pointer = 100

De volgende instructie
vertaald naar assembly
taal is 'DB 0F'

Assembler, De eerste instructies in DEBUG

“e” = ‘enter’ commando

```
-e 100  
1469:0100 0F.00  
-e 100 0A 0B 0C 0D 0E 0F
```

Via het ‘enter’ commando kunnen we geheugen cellen manipuleren.

“Het enter commando zal data op een geheugenlocatie wegschrijven”

Voorbeeld: ‘e100’

→ De actuele waarde van adres 100 wordt getoond.

→ Vervolgens kan de waarde worden veranderd.

Opmerking:

Elke geheugenlocatie is 1 byte groot → 2 hexadecimale tekens

Assembler, De eerste instructies in DEBUG

“e” = ‘enter’ commando

De actuele data op adres
1469:0100 is ‘OF’

De data op adres 1469:0100
wordt overschreven met 00

```
-e 100  
1469:0100 0F.00  
-e 100 0A 0B 0C 0D 0E 0F
```

Startend vanaf adres 100 wordt de data ‘0A 0B 0C
0D 0E 0F’ weggeschreven

Assembler, De eerste instructies in DEBUG

“d” = ‘dump’ instructie

Via een dump wordt de data van een geheugenbereik op het scherm getoond.

```
-d 100
1469:0100  0A 0B 0C 0D 0E 0F AE 47-61 03 1F 8B C3 48 12 B1  .....Ga....H..
1469:0110  04 8B C6 F7 0A 0A D0 D3-48 DA 2B D0 34 00 58 14  .....H.+ .4.X.
1469:0120  00 DB D2 D3 E0 03 F0 8E-DA 8B C7 16 C2 B6 01 16  .....
1469:0130  C0 16 F8 8E C2 AC 8A D0-00 00 4E AD 8B C8 46 8A  .....N...F.
1469:0140  C2 24 FE 3C B0 75 05 AC-F3 AA A0 0A EB 06 3C B2  .$.<.u.....<.
1469:0150  75 6D 6D 13 A8 01 50 14-74 B1 BE 32 01 8D 8B 1E  umm...P.t..2....
1469:0160  8E FC 12 A8 33 D2 29 E3-13 8B C2 03 C3 69 02 00  ....3.).....i..
1469:0170  0B F8 83 FF FF 74 11 26-01 1D E2 F3 81 00 94 FA  ....t.&.....
-
-
```

Assembler, DEBUG - machinetaal

“d” = ‘dump’ instructie

Voorbeeld ‘d100’ toont het adres bereik vanaf adres 100.

De inhoud van de adressen worden getoond in hexadecimale waarde en met de overeenkomstige ascii waarden.

Geheugenlocatie 100	Hexadecimale waarden	ASCII symbolen (overeenkomstig met de ascii waarde)
<code>-d 100</code>		
1469:0100	0A 0B 0C 0D 0E 0F AE 47-61 03 1F 8B C3 48 12 B1Ga....H..
1469:0110	04 8B C6 F7 0A 0A D0 D3-48 DA 2B D0 34 00 58 14H.+ .4.X.
1469:0120	00 DB D2 D3 E0 03 F0 8E-DA 8B C7 16 C2 B6 01 16
1469:0130	C0 16 F8 8E C2 AC 8A D0-00 00 4E AD 8B C8 46 8AN...F.
1469:0140	C2 24 FE 3C B0 75 05 AC-F3 AA A0 0A EB 06 3C B2	.\$.<.u.....<.
1469:0150	75 6D 6D 13 A8 01 50 14-74 B1 BE 32 01 8D 8B 1E	umm...P.t..2....
1469:0160	8E FC 12 A8 33 D2 29 E3-13 8B C2 03 C3 69 02 003.).....i..
1469:0170	0B F8 83 FF FF 74 11 26-01 1D E2 F3 81 00 94 FAt.&.....
-		
-		

Opmerking: In het rood zie je de gegevens ingegeven via het e-commando

Assembler, DEBUG - machinetaal

Een voorbeeld programma

Machinetaal	Betekenis	Uitvoering
01 D8	Optelling	$AX = AX + BX$
29 D8	Aftrekking	$AX = AX - BX$
F7 E3	Vermenigvuldiging	$AX = AX * BX$
F7 F3	Deling	$AX = A/BX$

Invoer in debug:

e 100 01 D8 29 D8 F7 E3 F7 F3

Controle van geheugen (dump op het scherm):

d 100

```
C:\>DEBUG
-e 100 01 D8 29 D8 F7 E3 F7 F3
-
-d 100
1469:0100  01 D8 29 D8 F7 E3 F7 F3-61 03 1F 8B C3 48 12 B1  ..).a...H..
1469:0110  04 8B C6 F7 0A 0A D0 D3-48 DA 2B D0 34 00 58 14  .....H.+4.X.
1469:0120  00 DB D2 D3 E0 03 F0 8E-DA 8B C7 16 C2 B6 01 16  .....
1469:0130  C0 16 F8 8E C2 AC 8A D0-00 00 4E AD 8B C8 46 8A  .....N...F.
1469:0140  C2 24 FE 3C B0 75 05 AC-F3 AA A0 0A EB 06 3C B2  .$.<.u.....<.
1469:0150  75 6D 6D 13 A8 01 50 14-74 B1 BE 32 01 8D 8B 1E  umm...P.t..2...
1469:0160  8E FC 12 A8 33 D2 29 E3-13 8B C2 03 C3 69 02 00  ....3.).....i..
1469:0170  0B F8 83 FF FF 74 11 26-01 1D E2 F3 81 00 94 FA  ....t.&.....
-

```

Assembler, DEBUG - machinetaal

Een voorbeeld programma

Dit is een voorbeeld van machinetaal...

Machinetaal	Betekenis	Uitvoering
01 D8	Optelling	$AX = AX + BX$
29 D8	Aftrekking	$AX = AX - BX$
F7 E3	Vermenigvuldiging	$AX = AX * BX$
F7 F3	Deling	$AX = A/BX$

Dit is de data die wordt gezien door de processor. De instructie pointer wijst deze data aan...
Bijvoorbeeld: op adreslocatie 100 & 101 staat 010 & D8.

Bijvoorbeeld: wanneer de processor 01 D8 ziet zal hij dit vertalen als “een optelling van het AX register met het BX register”

Dit is de instructie die tijdens het programma zal worden uitgevoerd wanneer de instructie pointer het adres 0100 aanduid.

Assembler, DEBUG - machinetaal

In dit voorbeeld wordt via het enter commando de machinetaal manueel ingevoerd. Dit zal later gebeuren aan de hand van assembler taal!

Machinetaal is een reeks van hexadecimale karakters.

Invoer in debug:

E 100 01 D8 29 D8 F7 E3 F7 F3

Controle van geheugen (dump op het scherm):

d 100

Via de dump instructie (d100) wordt de data getoond vanaf locatie 0100

```
C:\>DEBUG
-e 100 01 D8 29 D8 F7 E3 F7 F3
-d 100
1469:0100 01 D8 29 D8 F7 E3 F7 F3-61 03 1F 8B C3 48 12 B1 ..).....a....H..
1469:0110 04 8B C6 F7 0A 0A D0 D3-48 DA 2B D0 34 00 58 14 .....H.+..4.X.
1469:0120 00 DB D2 D3 E0 03 F0 8E-DA 8B C7 16 C2 B6 01 16 .....
1469:0130 C0 16 F8 8E C2 AC 8A D0-00 00 4E AD 8B C8 46 8A .....N...F.
1469:0140 C2 24 FE 3C B0 75 05 AC-F3 AA A0 0A EB 06 3C B2 .$.<.u.....<.
1469:0150 75 6D 6D 13 A8 01 50 14-74 B1 BE 32 01 8D 8B 1E umm...P.t..2....
1469:0160 8E FC 12 A8 33 D2 29 E3-13 8B C2 03 C3 69 02 00 ....3.).....i..
1469:0170 0B F8 83 FF FF 74 11 26-01 1D E2 F3 81 00 94 FA .....t.&.....
-
```

Assembler, DEBUG - machinetaal

'r' toont de actuele stand van de registers

```
-r  
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000  
DS=1469 ES=1469 SS=1469 CS=1469 IP=0100  NV UP EI PL NZ NA PO NC  
1469:0100 01D8      ADD     AX,BX  
-
```

Volgende instructie in assembly taal (= ADD AX, BX)

Volgende instructie in machinetaal (=01D8)

Waarde van de instructiepointer (plaats waard de instructie wordt opgehaald)

Assembler, DEBUG - machinetaal

Registers "manueel vullen"

- **"r ax"**
- Geef een waarde → HEXADECIMAAL !!!
- Geef een waarde aan AX en BX
- Controleer de waarde via het commando "r" (opvraging van de registers)

r ax → toont de actuele waarde van het register AX
(bijvoorbeeld 0000)

Na ':' kan de waarde manueel worden gewijzigd.
(bijvoorbeeld naar 10 (=decimaal 16!))

```
-r ax
AX 0000
:10
```

```
-r
AX=0010 BX=0000 CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=1469 ES=1469 SS=1469 CS=1469 IP=0100  NV UP EI PL NZ NA PO NC
1469:0100 01D8          ADD     AX,BX
```

Opmerking: herhaal deze stappen met BX

Assembler, DEBUG - machinetaal

```
-r
AX=0010  BX=0005  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=1469  ES=1469  SS=1469  CS=1469  IP=0100  NV UP EI PL NZ NA PO NC
1469:0100 01D8          ADD     AX,BX
```

'r' Toont de huidige waarde van de registers

- IP (Instructie Pointer) = 0100
- 01 D8 is de data op adres 0100 (De instructie pointer verwijst naar deze data)
- Machinetaal voor deze instructie = 01D8, en staat op adres 100.
- Uit te voeren instructie = ADD AX, BX

Assembler, DEBUG - machinetaal

't' = trace

```
-r
AX=0010 BX=0005 CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=1469 ES=1469 SS=1469 CS=1469 IP=0100  NV UP EI PL NZ NA PO NC
1469:0100 01D8          ADD     AX,BX
-t
AX=0015 BX=0005 CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=1469 ES=1469 SS=1469 CS=1469 IP=0102  NV UP EI PL NZ NA PO NC
1469:0102 29D8          SUB     AX,BX
-
```

't' (=Trace) voert de instructie uit:

- ADD AX, BX wordt uitgevoerd.
- Nieuwe waarde in AX is 0015 (➔ $ax+bx = 10+5$).
- IP is verhoogt (zodat de volgende instructie staat klaar).
- Volgende instructie = SUB AX, BX (=29D8 in machinetaal ...).