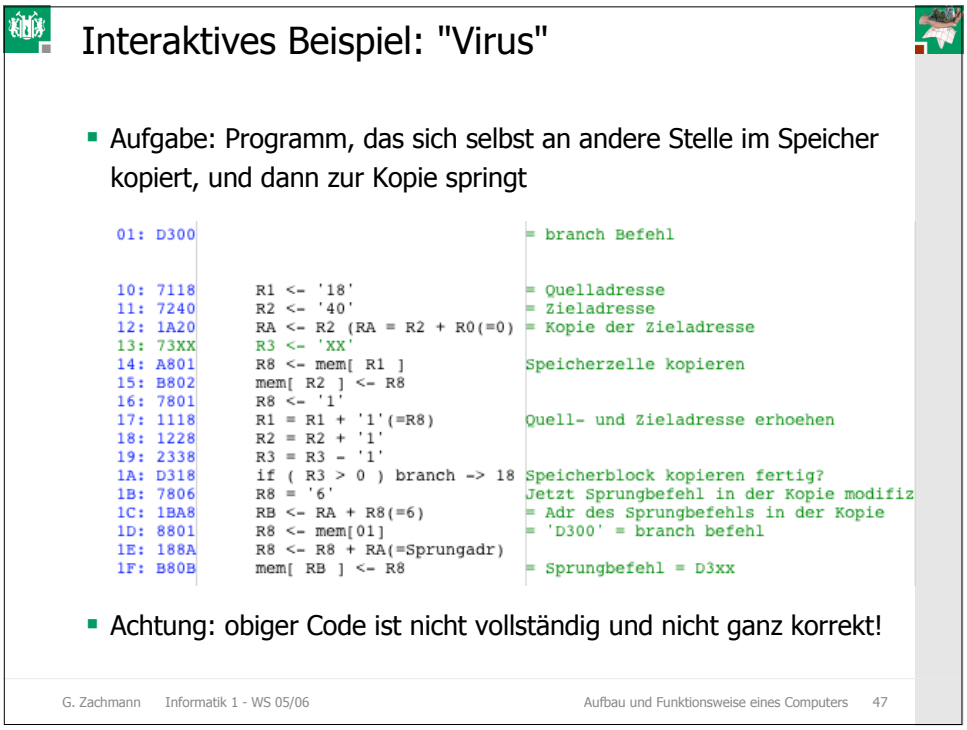


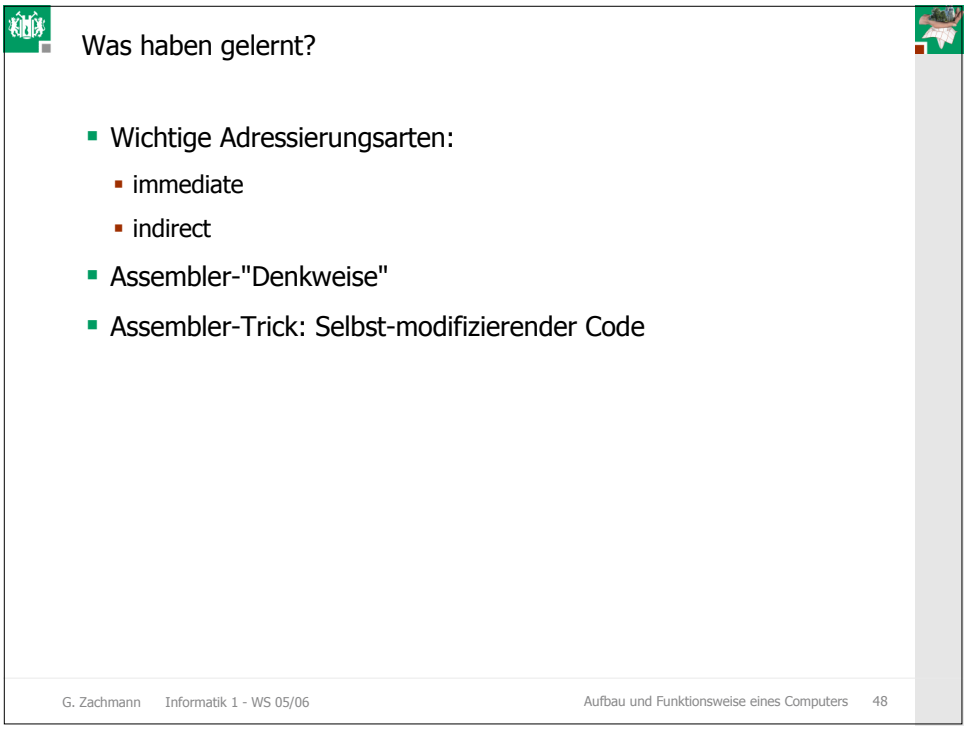
Interaktives Beispiel: "Virus"

- Aufgabe: Programm, das sich selbst an andere Stelle im Speicher kopiert, und dann zur Kopie springt

```
01: D300                                    = branch Befehl


10: 7118      R1 <- '18'                    = Quelladresse
11: 7240      R2 <- '40'                    = Zieladresse
12: 1A20      RA <- R2 (RA = R2 + R0(=0)    = Kopie der Zieladresse
13: 73XX      R3 <- 'XX'
14: A801      R8 <- mem[ R1 ]               Speicherzelle kopieren
15: B802      mem[ R2 ] <- R8
16: 7801      R8 <- '1'
17: 1118      R1 = R1 + '1'(=R8)            Quell- und Zieladresse erhoehen
18: 1228      R2 = R2 + '1'
19: 2338      R3 = R3 - '1'
1A: D318      if ( R3 > 0 ) branch -> 18    Speicherblock kopieren fertig?
1B: 7806      R8 = '6'                      Jetzt Sprungbefehl in der Kopie modifiz
1C: 1BA8      RB <- RA + R8(=6)             = Adr des Sprungbefehls in der Kopie
1D: 8801      R8 <- mem[01]                 = 'D300' = branch befehl
1E: 188A      R8 <- R8 + RA(=Sprungadr)
1F: B80B      mem[ RB ] <- R8               = Sprungbefehl = D3xx
```

- Achtung: obiger Code ist nicht vollständig und nicht ganz korrekt!



Was haben gelernt?

- Wichtige Adressierungsarten:
  - immediate
  - indirect
- Assembler-"Denkweise"
- Assembler-Trick: Selbst-modifizierender Code