6502 Programmieren - Teil 3#

von R. Wilde#

In der letzten Ausgabe des Magazins hatten wir die Maschinenbefehle des 6502 besprochen. Diesmal sehen wir uns deren Adressierungsarten an.

Hexadezimalzahlen werden wie üblich mit einem vorangestellten "$" gekennzeichnet.

Adressierungsarten :

Relativ
Accumulator
Implied
Immediate
Zeropage
Zeropage,X
Zeropage,Y
Absolut
Absolut,X
Absolut,Y
(Indirekt)
(Indirekt,X)
(Indirekt),Y

Relativ#

Diese Adressierungsart finden wir nur in Verzweigungsbefehlen (Branch). Die Adressierung bezieht sich hier auf die Adresse im Programmzähler (PC). Der Wert hinter dem BRANCH-Befehl gibt den Abstand von der augenblicklichen Adresse im Programmzähler (PC) an. Ist der Wert positiv (0 bis 127), wird nach vorne verzweigt. Ist der Wert negativ (128-255), wird rückwärts, vor den BRANCH-Befehl verzweigt.

Zwei Beispiele

```
1) Adresse Befehl
==> $4000 : F0 01 BPL $4003
    $4002 : 60 RTS
    $4003 : A9 40 LDA #$40

2) Adresse Befehl
    $5010 : A2 00 LDX #0
    $5012 : A0 10 LDY #$10
    $5014 : 60 RTS
    $5015 : AD 00 10 LDA $1000
==> $5018 : 30 F6 BMI $5010
```

1) Wert im PC nachdem der Prozessor den BRANCH-Befehl erkannt hat =$4002. Der Wert hinter dem Befehl = 1. Das Programm wird ab der Adresse $4002 + 1 = $4003 fortgeführt.

2) Wert im PC = $501A, Wert hinter dem BRANCH-Befehl = $F6 (246). Ergibt : $501A - ($100-$F6) = $5010

Accumulator#

Hier bezieht sich die Adressierung nur auf ein Register innerhalb des Prozessors, den ACCU. Diese Adressierungsart finden wir nur bei den Schiebebefehlen wie ASL, LSR, ROL und ROR.

Implied #

Implied steht hier für innerhalb. Auch hier bezieht sich die Adressierung nur auf Register innerhalb des Prozessors wie Status-Register (P) und Stackpointer (S).

Beispiele

```
CLC      Clear Carry-Flag
SED      Set Dezimal-Flag
PHA      Push ACCU on Stack
```

Immediate#

Direkte, unmittelbare Adressierung. Hier wird nicht auf Werte in Speicherstellen, sondern auf Werte, die direkt hinter dem entsprechenden Maschinenbefehl stehen, zugegriffen.

Beispiele

```
LDA #4     Lade ACCU mit dem Wert 4
ADC #1     Addiere zum ACCU den Wert 1
EOR #$80   Exklusiv ODER (ACCU mit dem Wert $80)
LDY #127   Lade Y-Register mit Wert 127
```

Zeropage#

Diese Adressierung bezieht sich auf die erste Speicherseite im Computer (Speicherstellen $0 bis $FF).

Beispiele

```
LDA $0     Lade ACCU mit dem Inhalt der Speicherstelle 0
LDX $80    Lade X-Register mit dem In- halt der Speicherstelle $80
STY $E0    Speichere Wert im Y-Register in die Speicherstelle $E0
```

Zeropage,X#

Hier wird zur Zeropage-Adresse der Wert im X-Register dazugezählt. Erst das Ergebnis bezeichnet die effektive Speicherstelle.

Beispiele

```
LDA $40,X     Ist im X-Register der Wert 6, so wird der ACCU mit dem Inhalt der
              Speicherstelle $40 + 6 = $46 geladen.
STY $90,X     Ist im X-Register der Wert $3E (62), so wird der Wert im Y-Register
              in die Speicherstelle $90 + $3E = $CE abgelegt.
```

Zeropage,Y#

Das ist die gleiche Adressierungsart wie zuvor. Nur dass an Stelle des Indexregisters X das Indexregister Y steht.

Absolut#

Das ist sicherlich die am meisten gebrauchte Art der Adressierung. Wie die Bezeichnung bereits sagt, wird hier die Adresse, die angesprochen werden soll, direkt angegeben.

Beispiele

```
LDA $89AB     Lade ACCU mit dem Inhalt der Speicherstelle $89AB
AND $2000     Führe eine UND-Verknüpfung des ACCU mit dem Inhalt der Speicherstelle $2
JMP $7000     Springe zur Adresse $7000
```

Absolut,X#

Hier wird wieder zur angegebenen Adresse der Inhalt des Indexregisters X zugezählt. Das Ergebnis bezeichnet die effektive Adresse.

Beispiele

```
STA $43F0,X     Ist der Inhalt des Index-Registers X = $20, wird der Wert im
                ACCU in der Speicherstelle $43F0+$20 = $4410 abgelegt
LDY $1000,X     Ist der Inhalt des X-Registers = $B3, wird der Inhalt der
                Speicherstelle $1000 + $B3 = $10B3 in das Y-Register geladen
```

Absolut,Y#

Auch hier gilt das gleiche wie zuvor bei der Adressierungsart "Absolut,X". Nur dass auch hier statt dem Register X das Register Y steht.

(Indirekt)#

Diese Adressierungsart finden wir nur bei dem JMP-Befehl. Die Adresse hinter dem Befehl gibt die Speicherstelle an, ab welcher die Adresse steht, zu der gesprungen werden soll. Deshalb heißt diese Adressierungsart auch "Indirekt".

Beispiel

```
JMP ($BFFE)     Steht in Adresse $BFFE und $BFFF
                (Die Adressangabe besteht hier immer aus 2 Bytes)
                $30 und $A5, dann wird zur Adresse $A530 gesprungen.
```

(Indirekt,X)#

Die Adresse, auf die zugegriffen werden soll, steht in den Speicherstellen, die folgendermaßen errechnet werden: Die hinter dem Befehl angegebene Adresse + Inhalt des X-Registers. Die Adresse darf sich nur auf die Zeropage ($00 bis $FF) beziehen.

Beispiel

```
LDA ($10,X)     Wenn im X-Register der Wert $50 steht, ergibt sich die Adresse $10+$50
                Ab $60 steht die Adresse (2 Bytes) der Speicherstelle, deren Inhalt in
```

(Indirekt),Y#

Das ist die komplexeste Art der Adressierung. Hier steht hinter dem Befehl eine Zeropage-Adresse (1 Byte) ab der die Adresse steht, zu der noch der Wert des Y-Registers gezählt werden muss. Dann hat man die effektive Adresse.

Beispiel

```
LDA ($E0),Y     In den Speicherstellen $E0/$E1 steht die absolute Adresse (2 Bytes),
                zu der noch der Inhalt des Indexregisters Y zugezählt wird.
                Das ergibt die Adresse der Speicherstelle, deren Inhalt in den ACCU ge
```

Bis zur nächsten Ausgabe Euer R.Wilde
CSM / 8.1988

Der Artikel entstammt der Kursreihe ?6502 Programmieren? des Compy Shop Diskettenmagazins.
Die Kursreihe besteht aus 14 Kursen, die im Laufe des Jahres 2011 in unregelmäßigen Abständen einzeln veröffentlicht werden, bzw. anschließend als Zusammenzug als ABBUC-Buch ?6502 Programmieren? erscheinen.
Koordination: Volkert Barr (volkert@nivoba.de)
Version 1.1 / 2011-01-15