6502 Programmieren - Teil 9#

von Uwe Röder#

Herzlich Willkommen zum neunten Teil unseres Assembler-Kurses!!!

Diesen Monat werde ich Ihnen die drei Vergleichsbefehle CMP, CPX und CPY vorstellen, die sehr gut Ihr bisher angehäuftes Wissen über die Flags und die relativen Sprungbefehle ergänzen werden. Außerdem werde ich noch auf die Transportbefehle und den Stack eingehen.

Um es übrigens gleich vorweg zu sagen: Diesen Monat wird unsere Assembler-Ecke etwas kürzer als sonst ausfallen, da ich zur Zeit ziemlich viel zu tun habe. Falls Ihnen die Informationen dieses Artikels nicht ausreichen, um einen vollen Monat damit beschäftigt zu sein, empfehle ich Ihnen den Artikel über den DLI in dieser Ausgabe genau durchzuarbeiten. Auf der Diskette sind dazu drei Beispielprogramme im Bibo-Assemblerformat.

Nun aber zu den Vergleichsbefehlen.

Unser ATARI hat drei Vergleichsbefehle anzubieten: CMP, CPX und CPY.

In diesen Fällen wird immer der Inhalt einer Adresse oder eine Zahl mit dem Inhalt von Akku, X- oder Y-Register verglichen. Die Register- und Speicherinhalte werden dabei nicht verändert. Das Ergebnis der Vergleichsoperation können Sie an folgenden drei Flags erkennen: Negative, Zero, Carry.

Ist der Inhalt des Registers (A,X,Y) grösser als die zu vergleichenden Daten, so ist das Carry-Flag gesetzt. Zero- und Negative-Flag sind gelöscht.

Ist der Inhalt des Registers gleich den zu vergleichenden Daten, so sind Carry- und Zero-Flag gesetzt und das Negative-Flag gelöscht.

Und schließlich der letzte Fall. Wenn der Inhalt des Registers kleiner ist als die zu vergleichende Daten, so ist das Negative-Flag gesetzt und Carry- und Zero-Flag sind gelöscht.

Daraus ergeben sich für die Sprungbefehle folgende Bedeutungen:

BEQ - Sprung bei Gleichheit BNE - Sprung bei Ungleichheit

BCS - Sprung wenn Register >= Daten BCC - Sprung wenn Register < Daten

BPL - Sprung wenn Register >= Daten BMI - Sprung wenn Register < Daten

Der Befehl CMP, der einen Vergleich zwischen dem Akku und dem Inhalt einer Adresse oder einer Zahl tätigt, kann wie folgt angewendet werden:

```
CMP ADR       :ADR=16 bit Adresse
CMP adr       :adr=8 bit Adresse
CMP #NO       :NO=8 bit Zahl
CMP ADR,X
CMP ADR,Y     :ADR=16 bit Adresse
CMP (adr,X)
CMP (adr),Y
CMP adr,x     :adr=8 bit Adresse
```

Für die Befehle CPX und CPY gibt es nur drei Adressierungsarten:

```
CPX ADR       :ADR=16 bit Adresse
CPX adr       :adr=8 bit Adresse
CPX NO        :NO=8 bit Zahl
```

Jetzt, wo Sie die Vergleichsbefehle kennengelernt haben, sollten Sie eigentlich schon hervorragend programmieren können, denn programmieren in Assembler heißt nichts anderes zu tun als Speicherstellen zu verändern, um dann die Inhalte zu vergleichen und aufgrund der Ergebnisse wieder Speicherstellen zu verändern...

Nun ja. Ich werde dann doch lieber erst auf die Transportbefehle eingehen. Es gibt genau sechs Transportbefehle: TAX,TAY,TXA,TYA,TSX und TXS.

Die ersten vier Befehle sind noch ganz einfach zu erklären. Hier wird einfach der Inhalt eines Registers in ein anderes kopiert. Dabei ist es allerdings nicht möglich den Inhalt des X-Registers direkt in das Y-Register zu schreiben und umgekehrt. Die ersten vier Befehle haben somit folgende Bedeutung:

TAX
: Inhalt des Akku ins X-Register übertragen.

TXA
: Inhalt des X-Registers in den Akku übertragen.

TAY
: Inhalt des Akku in das Y-Register übertragen.

TYA
: Inhalt des Y-Register in den Akku übertragen.

Bei diesen Transportbefehlen wird immer nur der Registerinhalt des Registers verändert, in das hineinkopiert wird. Der Inhalt des 'Quell'-Registers bleibt erhalten.

Die Transportbefehle können das Zero- und das Negative-Flag verändern.

Die letzten beiden Befehle TSX und TXS haben eine besondere Bedeutung. Mit Ihnen kann man den Stapelzeiger ins X-Register bzw. das X-Register in den Stapelzeiger übertragen.

Unter dem Stapel oder STACK versteht man einen bestimmten Speicherbereich, der bei drei Sachen benötigt wird:

1. vorübergehende Datenspeicherung
2. Adressenspeicherung bei Unterprogrammen
3. Registerspeicherung bei Interrupts

Der Stapel belegt bei den 6502-Rechnern immer die Page 1, das heißt er nimmt die Adressen 256 ($100) - 511 ($1FF) ein.

Das besondere am Stapel ist seine Struktur. Man kann sie vergleichen mit einem Stapel Karten aus einem Kartenspiel. Bei diesem Karten-Stapel sind nun die Werte bestimmte Spielkarten.

Die Bedingung, die das Arbeiten mit dem Stapel manchmal etwas schwierig erscheinen lässt ist, dass man immer nur die oberste Karte wieder abheben kann. Das heißt, dass die Karte, die ich zuletzt abgelegt habe, die erste ist, die ich wiederholen kann. Um an die erste abgelegte Karte zu gelangen muss ich erst alle anderen Karten Stück für Stück abheben.

Beim Rechner ist dies wie folgt organisiert:

Der Stapelzeiger zeigt immer auf das nächste freie Byte des Stapels. Ganz zu Beginn müsste dies die Adresse $1FF sein. Der Stapelzeiger hat dann den Wert $FF, da er nur 8 veränderliche Bits umfasst. Ein neuntes unveränderliches und eigentlich auch 'unsichtbares' Bit hat immer den Wert 1.

Lege ich nun mit PHA den Inhalt des Akku auf den Stapel, so wird bei der Adresse $1FF der Inhalt des Akku abgelegt. Der Stapelzeiger wird um eins vermindert und zeigt auf die nächste freie Adresse, also $1FE.

Jedes Mal wenn ein neuer Wert auf den Stapel gelegt wird, passiert dieser Vorgang.

Mit dem Befehl PLA wird nun ein Byte vom Stapel in den Akku übertragen.

Die Adresse des Bytes im Stapel ist die des Stapelzeigers plus eins. Plus eins deshalb, weil der Stapelzeiger ja auf das erste freie Byte zeigt. Und das letzte abgelegte liegt ja eine Adresse höher.

Bei Interrupts werden nun zu Beginn der Interrupt-Routine die Werte der Register auf dem Stapel abgelegt und ganz am Ende der Routine wieder zurückgeholt.

Dieses Zurückholen von Daten ist sehr wichtig. Auf dem Stapel werden nämlich auch die Rücksprungadressen für Unterprogramme abgelegt. Wenn ich nun Daten auf dem Stapel ablege und diese nicht zurückhole bevor ein RTS, also ein Rücksprung aus einem Unterprogramm durchgeführt wird, so hält der Computer meine Daten für die Rücksprungadresse und springt fleißig irgendwo in den Speicher. Dies kann sehr leicht zu Abstürzen führen.

Mit den oben erwähnten Transportbefehlen TSX und TXS kann nun der Stapelzeiger mit Hilfe des X-Registers ausgelesen und verändert werden. Ich selbst habe diese Möglichkeit noch nie in einem meiner Programme benutzt und möchte Ihnen empfehlen dies erst dann zu tun wenn Sie absolut sicher im Programmieren sind und wenn Sie keine Möglichkeit sehen dies zu umgehen.

Damit möchte ich dann auch diesen Monat schließen. Nächsten Monat werde ich noch im Zusammenhang mit dem Stapel auf den JSR und den RTS Befehl eingehen und die letzten noch unbekannten Befehle erläutern.

Bis dahin viel Spaß am Assembler.
Ihr Uwe Röder
CSM / 4.1989

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 / 2011-01-23