

 Friedrich-Ebert-Schule Esslingen	MIKROCONTROLLER	Name:
2.3.5.1	Adressierungsarten	Datum:

Für den Zugriff auf Speicherstellen, Register und On-Chip-Peripherie kennt der 8051-Controller fünf verschiedene Adressierungsarten. Sie unterscheiden sich in der Anzahl benötigter Befehls-Bytes, der Befehlsausführungszeit und den möglichen Zugriffszielen:

1. Direkte Adressierung

Der Operand ist eine Speicheradresse im internen RAM. Diese Adresse wird als Hex-Code oder symbolisch angegeben (siehe oben).

Beispiel: `mov 48h,0A0h ; Inhalt von Port2 (A0h) ins int. RAM`
`mov 48h,P2 ; dito. mit symbolischer Adresse`

Für die direkte Adressierung wird je Operand ein Byte im Programmspeicher benötigt.

2. Register-Adressierung

Bei der Register-Adressierung ist der Operand eines der Register: R0...R7, A, B, DPTR (16 Bit), CY (1 Bit).

Beispiel: `mov A,R4 ; Kopiert den Inhalt des Registers R4 in den`
`; Akku`

Da diese speziellen Register im 8051 sehr häufig benötigt werden, sind Sie direkt in den Opcode integriert. Für den Befehl **mov A,Rn** gibt es daher insgesamt 8 Opcodes, je nachdem welches der Register R0 ... R7 als Operand verwendet wird.

Befehl	Opcode
mov A,R0	E8h
mov A,R1	E9h
mov A,R2	EAh
mov A,R3	EBh
mov A,R4	ECh
mov A,R5	EDh
mov A,R6	EEh
mov A,R7	EFh

Da sich alle Register des 8051 organisatorisch im SFR oder internen RAM wiederfinden, kann jedes dieser Register natürlich auch mit seiner Speicheradresse (direkt) angesprochen werden. Obiges Beispiel lässt sich also auch so formulieren:

`mov 0E0h,04h ; E0 = Akku, 04 = R4`

Der Unterschied der beiden Adressierungsarten zeigt sich, wenn man den Speicherbedarf und die Ausführungszeit der Befehle vergleicht. Das erste Beispiel belegt nur 1 Byte Programmspeicher und ist in einem Maschinenzyklus bearbeitet. Mit direkter Adressierung werden drei Bytes und zwei Maschinenzyklen benötigt.

3. Unmittelbare (Immediate) Adressierung

Bei dieser Adressierungsart besteht der Operand aus einer Konstanten im Programmspeicher. Da der Programmspeicher nur bei der Programmierung, nicht aber während der Laufzeit geändert werden kann, ist eine solche Konstante natürlich nur als Quelloperand zulässig. Der Assembler erlaubt verschiedene Darstellungsarten für Konstanten. Diese werden aber immer durch das #-Symbol (Raute, Doppelkreuz oder Gartenzaun) gekennzeichnet. Die Zahl 65 kann somit in mehreren äquivalenten Formen dargestellt werden.

Zur Adressierung können auch 16 Bit Adresskonstanten definiert werden (normalerweise Hex):

Konstante	Codierung	Grösse
#1F00h	Adresskonstante	Word

Konstante	Codierung	Grösse
#65	Dezimal	Byte
#65d	Dezimal	Byte
#41h	Hex	Byte
#01000001b	Binär	Byte
#101o	Oktal	Byte
#'A'	ASCII	Byte

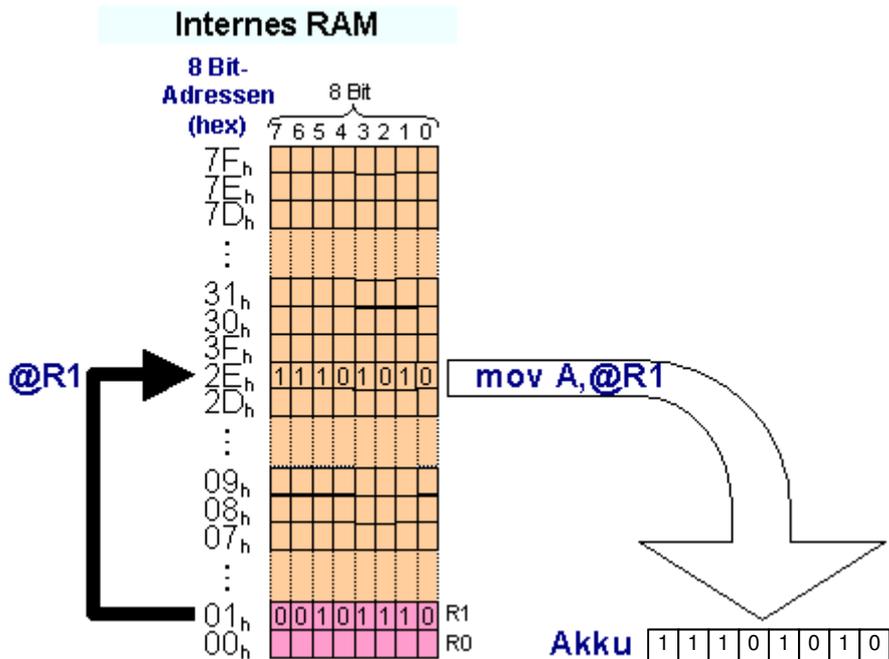
Beispiel: `mov A,#0d ; Löscht den Akku!`
`mov DPTR,#1F00h ; setzt den Datenzeiger auf die`
`; Adresse 1F00h im Programmspeicher`
`mov P1,#01110111b ; Die Bits von Port 1 werden entsprechend`
`; der Bitmaske belegt`

 Friedrich-Ebert-Schule Esslingen	MIKROCONTROLLER	Name:
2.3.5.2	Adressierungsarten	Datum:

4. Registerindirekte Adressierung

Hier wird die Adresse des Operanden nicht direkt im Befehl angegeben, sondern indirekt aus einem Hilfsregister gelesen. Als Hilfsregister dienen dabei **R0** und **R1** für Zugriffe ins interne RAM, sowie der Datenzeiger **DPTR** für Zugriffe ins eXterne RAM.

Beispiel: `mov R1, #2Eh` ; R1 mit der Adresskonstanten 2Eh laden!
`mov A, @R1` ; Kopiert den Inhalt von 2Eh in den Akku!



Die Registerindirekte Adressierung ermöglicht, dass der gleiche Code mit unterschiedlichen Ziel- oder Quelloperanden ausgeführt werden kann. Es muss jeweils nur die gewünschte Adresse nach R0, bzw. R1 geladen werden. Außerdem ist dies die einzige Adressierungsmöglichkeit für Zugriffe in den oberen RAM-Bereich (Adressen 80h ... FFh).

5. Indizierte Adressierung (Basisadresse + Index)

Nur durch indizierte Adressierung kann auf Datentabellen, Datenfelder oder Textkonstanten im **Programmspeicher** zugegriffen werden. Im Programmspeicher ist natürlich nur ein lesender Zugriff möglich. Als Basisadresse wird der Datenzeiger (**DPTR**) oder der Programmzähler (**PC**) verwendet. Als Index dient jeweils der Akkumulator. Das Holen eines Tabellenwertes aus dem Programmspeicher zeigt

folgendes **Beispiel:**

```
mov DPTR, #02B3h
mov A, #02h
movc A, @A+DPTR
```

