Bitadressierung im 8051-Microcontroller

Walter Riemer

Der 8051 Mikrocontroller ist nach wie vor eine Säule unseres Unterrichts; der Autor dieses Berichts befasst sich schon seit vielen Jahren mit ihm und hat auch schon vor Jahren ein Skriptum verfasst, welches in der Lehrmittelstelle zum freien Kopieren aufliegt und fallweise auf den neuesten Stand gebracht wurde. Auch viele Lehrerkollegen verwenden das Skriptum im Unterricht.

Im Zusammenhang mit der Bitadressierung heißt es in diesem Skriptum:

Im bitadressierbaren Bereich wird mit der Bitnummer (0 bis 127 vom niedersten Bit auf Byte 20h aufwärts) adressiert, aber ein Byte kann auch mit seiner Adresse (zum Beispiel 20h) als Ganzes angesprochen werden.

An anderer Stelle heißt es:

Bitadressierung (badr): Bits im bitadressierbaren internen Speicher werden mit ihrer Bitadresse (Nummer von 0 bis 127) oder einem mittels EQU-Direktive angegeben Namen adressiert.

BEISPIELE:

```
Alarm1  EQU  27h
  SETB  27h
  SETB  Alarm1
```

Die beiden Befehle sind identisch.

Das 8051-Fachbuch Nummer Eins: Köhn/Schultes: 8051-Prozessoren, belehrt uns auf Seite 122, Abschnitt 4.11.3 zum Thema "Setzen und Löschen von Bits":

```
SETB badr  ; mit 0 <= badr <= 255
```

und auf Seite 119, Abschnitt 4.11:

Um eine Einzelbit-Stelle ansprechen zu können, benötigt der Rechner ebenfalls eine Adresse. Die Bitstellen werden - wie von den Bytes bekannt - einfach durchnummeriert. Im Bereich der Speicherzellen 20H ... 2Fh sind die Bitstellen von 0 ... 127 (7FH) fortlaufend nummeriert.

Besonders beachtenswert ist der vorstehend letzte Satz. Aus ihm würde man den Schluß ziehen, dass etwa das Bit 27h im bitadressierbaren Bereich (der mit Bit 7 des Bytes 20h beginnt) identisch mit dem Bit 24h.0 sein sollte, das ist das niedrigwertigste Bit im Byte mit der Adresse 24h. Aus dem besprochenen letzten Satz und dem davorstehenden vorletzten Satz wird man den Schluß ziehen, dass man dieses Bit entweder mit seiner "fortlaufenden" Bit-Nummer (27h = 39d) oder mit seiner Byteadresse (24h) samt Bitnummer adressieren kann; sofern die Bitnummer in einem Byte von links nach rechts liefe, wäre die Bitadresse also 24h.7 .

So weit, so logisch.

Tatsächlich übersetzt der Assembler ASM51 den Befehl SETB 27h in den Object Code D2 27 , ebenso auch den Befehl SETB 24h.7 . Die Welt ist also durchaus in Ordnung:

```
LOC  OBJ   LINE  SOURCE
0000 D227  1     SETB    24h.7
           2     END     ;
```

Auch der Source-Code-Debugger DScope51 spielt brav mit. Der Befehl wird mittels ASM-Kommandos eingegeben:

```
asm 0
```
Assemblieren ab Adresse 0

```
0000H       setb 27h
```

```
0002H       NOP
```
Eingabe des Befehls, Beenden des Assemblierens

Tatsächlich entsteht der gleiche Object Code wie ihn der Assembler generiert:

```
>d c:0,1
```
Display Code-Speicher ab Adresse 0 bis Adresse 1

```
C:0000  D2 27
>
```
Inhalt dieser beiden Bytes

Im Language-Fenster wird auch ordnungsgemäß der Quellcode in der Byte.bit-Adreßschreibweise angezeigt:

```
0000H       SETB      24H.7
0002H       NOP
```

Wie es sich nach einem Reset gehört, sind die bitadressierbaren Bits alle Null:

```
>d i:20h,2fh
```
Display internes RAM, ganzen bitadressierbaren Bereich

```
I:0020  00 00 00 00 00 00 00 00-00
00 00 00 00 00 00 00  ............
```

Man wird nach dem oben Gesagten damit rechnen, dass das vierzigste Bit (also das Bit 39, da die Zählung ja mit Null beginnt) auf 1 gesetzt wird. Es muss dies im Byte 24h das äußerst-rechte (niedrigwertigste) Bit sein, dessen rechtes Halbbyte oben hervorgehoben wird.

Der Program Counter steht ohnehin noch auf Null, man kann also den Befehl mit Trace-Kommando getrost ausführen:

```
>t
```

Danach wird man wieder in den bitadressierbaren Bereich hineinschauen und findet statt des erwarteten Werts 01 auf dem fraglichen Byte den Wert 80:

```
>d i:20h,2fh I:0020  00 00 00 00 80 00
00 00-00 00 00 00 00 00 00 00
............
```

Also: Die Bitzählung im bitadressierbaren Bereich funktioniert so:

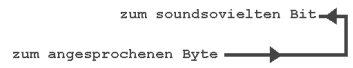
Bitnummer / 8 gibt die Byteadresse, im Beispiel 39/8=4, 7 Rest. Rest (7) ist die Bitnummer im so bestimmten Byte, von rechts nach links!!

Die Bitnummer ist also jene, die den Stellenwert bestimmt: Ganz rechts ist Bit 0 (mit dem Stellenwert 2 hoch 0 = 1, links daneben ist das Bit mit dem Stellenwert 2 hoch 1 = 2, usw. Ganz links ist das Bit 7 mit dem Stellenwert 2 hoch 7 = 64.

Nun hätte das ja alles seine Logik, wenn nicht die Aussage im Raum stünde:

Bits im bitadressierbaren internen Speicher werden mit ihrer Bitadresse (Nummer von 0 bis 127) adressiert.

Man neigt logischerweise dazu, diese Bitzählung durchgehend von links nach rechts vorzunehmen, muss aber zu Kenntnis nehmen, dass man nur byteweise von links nach rechts zu zählen hat, im angesprochenen Byte aber dann von rechts nach links:



Wenn man diese Besonderheit nicht weiß, interpretiert man zweifellos die Bitadressierung im bitadressierbaren Bereich des internen RAM mittels durchgehender Bitzählung falsch.

Das Assembler-Manual geht auf die Bitadressierung in der Form `SETB 27h` überhaupt nicht ein (wohl mit gutem Grund) und führt nur ein Beispiel mit Byte.bit-Adressierung an, nämlich `SETB 41.5` .

Das oben erwähnte Buch enthält auch die kluge Aussage:

PHYTEC Starterkit C167CR

Flash-Tools

Hermann Krammer

Der 256-kByte-Flash-Speicher auf dem PHYTEC-Board besteht aus 2 Flash-Bausteinen Am29F010 (128 K x 8 Bit), die somit einen 256-kByte-Speicher mit der Organisation 128k x 16 Bit ergeben. Dieser Flash-Speicher wird in 8 Sektoren mit den Offset-Adressen 0000h, 8000h, 10000h bis 38000h unterteilt. Für gewisse Anwendungen ist es wünschenswert, dass das Anwenderprogramm selbst Daten im Flash-Speicher ablegen kann. Diesem Zweck dienen die in `AMDFLASH.C` gesammelten Tools, die im Zuge eines Projekts an der HTL Braunau entwickelt wurden.

Da während des Löschens und Beschreibens des Flash-Speichers die dazu nötigen Routinen nicht im Flash laufen können, wählten wir folgende Vorgangsweise:

Die Routinen aus `AMDFLASH.C` müssen zunächst im Segment 0 installiert werden. Das externe 64-kByte-RAM wird sozusagen als Schattenspeicher für das Segment 0 eingesetzt und steht daher dem Anwender nicht zur Verfügung. Vor Anwendung der eigentlichen Tools muss `SwitchToRAM()` aufgerufen werden. In dieser Routine wird zunächst der Flash-Bereich 00:0000 bis 00:DFFF ins RAM kopiert, und wird das RAM ins Segment 0 eingeblendet. Das Flash ist über die Segmente 4 bis 7 (Memory image) erreichbar. Der Datenaustausch in sämtlichen Routinen erfolgt grundsätzlich mit 16 bit. Nach Beschreiben des Flash-Speichers kann mit `SwitchToFlash()` wieder auf die ursprüngliche Konfiguration umgeschaltet werden.

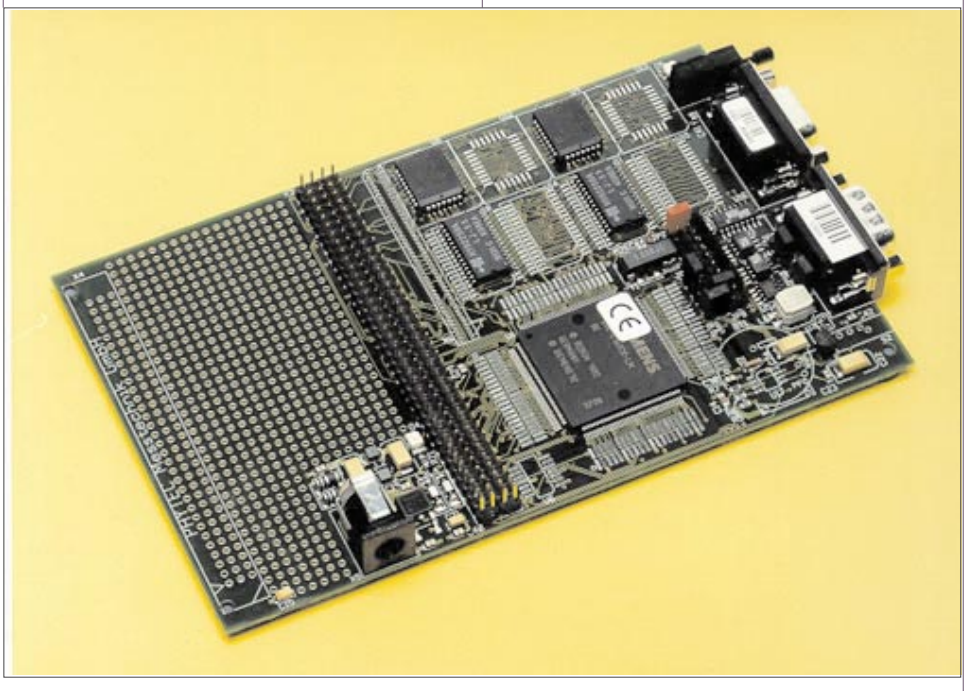
Der Sourcecode der Routinen steht als `FLASH.ZIP` in der Homepage der HTL Braunau zur Verfügung:

http://www.asn-linz.ac.at/schule/htlbraunau/lehrer/krammer/index.htm

```
// AMDFLASH.H
// Flash-Utilities for AMD Am29F010 on PHYTEC Board U8/U9
// 2 x 128 kBytes, CS0 (00:0000, 04:0000, 08:0000, ...
//     if not covered by CS1, CS2, CS3
// This module and the calling program must be located in
//     Segment 0
// External RAM (64kBytes) is used as shadow memory of
//     Segment 0

// Routines cannot run in the FLASH. Therefore switch to RAM
void SwitchToRAM(void);
void SwitchToFlash(void);

uword GetManufacturerID(void);  uword GetDeviceID(void);

// sector_number 0..7
uword SectorProtectVerify(uword sector_number);

// returns  0x0000 .. ok;  0x0080, 0x8000, 0x8080 .. Errors
uword ProgramWord(ulong addr, uword val);

// returns  0x0000 .. ok;  0x0080, 0x8000, 0x8080 .. Errors
uword SectorErase(ulong addr);

// returns  0x0000 .. ok;  0x0080, 0x8000, 0x8080 .. Errors
uword ChipErase(void);
```

Literatur

- Am29F010 Datenblatt, AMD Publication #16736, February 1998
- User´s Manual C167 Derivatives, Siemens AG München 1996
- KitCON-167 Hardware-Manual, PHYTEC Messtechnik GmbH Mainz 1997



➢ *Im Assembler ASM51 können Bit- wie Byteadressen zur besseren Lesbarkeit auch symbolisch angegeben werden. ASM51 erlaubt z.B. die Angabe einer Bitstelle im Byte mit*

```
Byteadresse.Bitnummer, also z.B. ACC.3
```

Ein Hinweis oder ein Beispiel auf die Form `SETB 27h` findet sich (wohl mit gutem Grund!) nirgends.

Kurzum: Die durchgehende Bitzählung ist bedeutungslos und irreführend. Diesbezügliche Aussagen sollten (aus Büchern und Skripten) ersatzlos gestrichen werden. Auffallend ist allerdings, dass dieser Sachverhalt in vielen Jahren anscheinend noch niemandem aufgefallen ist!

Auch Bits im bitadressierbaren Bereich sollten also nur in der Form

```
Byteadresse.Bitnummer
```

adressiert werden.

Hermann Krammer h.krammer@mail.asn-linz.ac.at