

Mikrocontroller-Grundlagen

Teil 6

Die Registerstruktur zur MCS-51-Familie beschreibt der vorliegende Artikel.

2.18. Spezielle Speicher

Eine besondere Funktion innerhalb des direkt adressierbaren RAMs haben der Akkumulator, das Programm-Statuswort, der Stack-Pointer sowie die Register. Für den Akkumulator, die Register und das Programm-Statuswort sind spezielle Prozessorbefehle implementiert, so daß damit sehr effiziente Zugriffe ermöglicht werden.

2.18.1 Akkumulator

Eine der wichtigsten Speicherzellen des MCS-51-Mikrocontrollers ist der Akkumulator (A oder ACC genannt). Sämtliche Arithmetik-Operationen wie Additionen, Subtraktionen, Multiplikationen und Divisionen sowie viele Speicherkopierfunktionen arbeiten mit diesem Register zusammen. Des weiteren kann der Akku als Arbeitsregister verwendet werden, um beispielsweise Informationen zwischenspeichern.

Wie im weiteren Verlauf der Artikelserie noch zu sehen sein wird, gibt es Befehle, die immer mit dem Akku arbeiten. Der Akkumulator ist über unterschiedliche Adressierungsarten ansprechbar. In den meisten Fällen wird dies direkt über den

entsprechenden Befehl vorgenommen. Er kann aber auch direkt als SFR (Special Function Register) oder auch bitweise über einen Bitverarbeitungsbehl angesprochen werden.

Tabelle 11 zeigt die Bitadressen der einzelnen Akkumulatorbits und die Byteadresse für die direkte Adressierung.

Tabelle 12 zeigt eine Übersicht über den Inhalt des Programm-Statuswortes. Durch die direkte Bit-Adressierung läßt sich somit jedes einzelne Bit beeinflussen.

Tabelle 13 zeigt die in Abhängigkeit von RS 0 und RS 1 selektierte Registerbank.

Das CY (Carry, Übertrag)-Flag, welches oft auch nur mit „C“ bezeichnet ist,

Tabelle 11: Bit- und Byte-Adressen des Akkumulators

Bezeichnung	ACC.7	ACC.6	ACC.5	ACC.4	ACC.3	ACC.2	ACC.1	ACC.0
Bitadresse	E7H	E6H	E5H	E4H	E3H	E2H	E1H	E0H
Byteadresse	0E0H							

Tabelle 12: Bit- und Byte-Adressen des Programm-Statuswortes

Bedeutung	CY	AC	F0	RS1	RS0	OV	-	P
Bitadresse	D7H	D6H	D5H	D4H	D3H	D2H	D1H	D0H
Byteadresse	0D0H							

2.18.2 Programm-Statuswort

Das Programm-Statuswort (PSW) ist wie der Akkumulator ein bitadressierbares Register. Dieses Register beinhaltet neben den Flags, die beispielsweise Auskunft darüber geben, ob bei dem letzten Arithmetik-Befehl ein Überlauf aufgetreten ist, 2 Bit für die Registerbankselektierung.

Tabelle 13: Selektierung der 4 Registerbanken der MCS-51-Familie

RS 1	RS 0	Register-Bank	RAM-Adresse
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

Tabelle 14: Zuordnung der 4 Registerbänke im unteren RAM-Bereich

Registerbank	Adresse							
	R0	R1	R2	R3	R4	R5	R6	R7
0	00H	01H	02H	03H	04H	05H	06H	07H
1	08H	09H	0AH	0BH	0CH	0DH	0EH	0FH
2	10H	11H	12H	13H	14H	15H	16H	17H
3	18H	19H	1AH	1BH	1CH	1DH	1EH	1FH

nutzt, um einen Über- bzw. Untertrag zu kennzeichnen. Vielfach wird es auch als Zwischenspeicher für Bit-Werte genutzt.

Das AC (Auxiliary Carry)-Flag findet in der BCD-Arithmetik Verwendung. Es wird gesetzt, wenn bei einer Addition der Inhalt des niederwertigen Nibbels (4 Bit) des Akkumulators größer als 9 oder bei einer Subtraktion ein negativer Übertrag aus dem höherwertigen Nibbel des Akkumulators vorliegt. F0 (Flag 0) ist für allgemeine Anwendungen frei verwendbar und wird durch keinerlei Arithmetikoperationen beeinflusst.

RS0 und RS1 selektieren die gerade zu verwendende Registerbank und lassen sich unabhängig von den restlichen Flags setzen bzw. löschen.

Das OV (Overflow, Überlauf)-Flag wird für die vorzeichenbehaftete Arithmetik sowie bei der Division und Multiplikation benötigt. Es wird gesetzt, wenn bei einer Addition oder Subtraktion der vorzeichenbehaftete Zahlenbereich (-128 bis +127) verlassen wird, bei einer Division durch 0 oder wenn das Ergebnis einer Multiplikation größer als FFH ist.

Das P (Parity)-Flag gibt stets die Parität des Akkumulators wieder. Es wird hardwaremäßig bei jeder Veränderung des Akkumulators aktualisiert. Es ist auf logisch 1 gesetzt, wenn die Zahl der gesetzten Bits im Akkumulator ungerade ist, anderenfalls ist es gelöscht. Dieses Bit läßt sich ausschließlich lesen, ein Überschreiben beeinflusst den Wert dieses Flags nicht.

2.18.3 Register

Die MCS-51-Familie verfügt über 8 unabhängige Register R 0 bis R 7, die vom Prozessor durch leistungsfähige Befehle unterstützt werden. Der Registersatz ist vierfach vorhanden. Die Selektierung des Registersatzes erfolgt über RS0 und RS1, die im Programm-Statuswort (PSW) untergebracht sind.

Tabelle 14 zeigt die Zuordnung und Platzierung der Register R 0 bis R 7 zu den Registerbänken 0 bis 3 im internen RAM-Speicher.

2.18.4 Der Stack-Pointer

Der normale Programmablauf im MCS-51-Mikroprozessor ist linear, d. h. es wird nacheinander Befehl für Befehl abgearbeitet und dabei der PC (Program-Counter,

Programmzähler) kontinuierlich erhöht. Um nun beispielsweise für immer wiederkehrende Programmteile diese nicht immer neu schreiben bzw. kopieren zu müssen, wurde die sogenannte Unterprogrammtechnik entwickelt. Hierzu wird über einen speziellen Befehl ein Unterprogramm angesprungen.

Bei der Ausführung eines Unterprogrammaufrufs wird die Rücksprung-Adresse auf den Stack (Stapelspeicher) gerettet. Zum Abschluß des Unterprogramms wird über einen speziellen Assembler-Befehl veranlaßt, daß die zuvor gespeicherte Rücksprungadresse wieder vom Stack heruntergenommen und zum Programmablauf an der ursprünglichen Stelle fortgesetzt wird. Durch diese Programmtechnik muß dem Unterprogramm zur Zeit des Rücksprungs nicht bekannt sein, von wo aus es aufgerufen wurde, bzw. wo die Programmabarbeitung nach Beendigung wieder fortgesetzt wird.

Der Stack-Bereich liegt im internen RAM von 0 bis 127 bzw. 0 bis 255 (8032/52) und kann durch Laden des Stack-Pointer (SP), der auch direkt adressierbar ist, an eine beliebige Stelle gelegt werden. Nach dem Zurücksetzen (Reset) des Prozessors ist dieser Zeiger auf 07H gemäß Abbildung 46 initialisiert. Der Stack-Pointer zeigt immer auf die zuletzt benutzte Speicherzelle.

Bei der Ablage von Daten bzw. Adressen auf dem Stack wird zunächst der Stack-Pointer um 1 erhöht und der gewünschte Wert in die dann adressierte Speicherzelle hineingeschrieben. Der Stack-Bereich wächst somit von niedrigeren Adressen zu höheren.

Der Anfangswert des Stack-Pointers kann durch Neuladen des SP-Registers auf

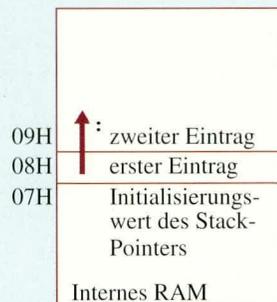


Bild 46: Vorinitialisierung des Stack-Pointers für das interne RAM

einen beliebigen 8-Bit-Wert gesetzt werden. Das Neuladen des Stack-Pointers nach dem Programmstart ist sinnvoll, da der Initialisierungswert sich mit der Registerbank 1 überschneidet und somit zu Kollisionen in der Speicherbenutzung führen könnte.

Die Programm-Variablen sind meistens im unteren Teil (niedrige Adressen) des internen RAMs untergebracht. Üblicherweise wird dann der Stack-Pointer so initialisiert, daß der interne RAM-Bereich für den Stack nach den Variablen genutzt wird. Zu beachten ist dabei allerdings, daß der Stack-Pointer nicht über die interne RAM-Grenze hinauswachsen darf (7FH bzw. FFH), da bei zu tief geschachtelter Unterprogrammstruktur o. ä. der Stack-Pointer überläuft und somit ein Programmabsturz voraussichtlich die Folge wäre.

2.19 Adressierungsarten

Die Register- und Speicherzellen der MCS-51-Familie lassen sich über unterschiedliche Adressierungsarten ansprechen. Neben der registerdirekten und unmittelbaren Adressierung ist noch die indirekte Adressierung des Code-Speichers oder Datenspeichers über ein 16-Bit-Index-Register vorgesehen.

2.19.1 Register-Adressierung

Die Register-Adressierung erlaubt den Zugriff auf 8 Register (R0 bis R7) der aktuellen Registerbank. Beispielsweise kopiert der Befehl <MOV A, R0> den Inhalt des Registers 0 in den Akkumulator. Je nach gewählter Registerbank wird der Wert für Register 0 aus den Speicherzellen 00H, 08H, 10H oder 18H gelesen und in den Akkumulator übernommen.

2.19.2 Direkte Adressierung

Die direkte Adressierung stellt die einzige Methode dar, auf die speziellen Funktionsregister zuzugreifen. Weiterhin sind die ersten 128 Byte des internen RAMs ebenfalls direkt adressierbar. Beispielsweise kopiert der Befehl <MOV A, PSW> den Inhalt des direkt adressierbaren Speichers PSW (D 0 H) in den Akkumulator.

2.19.3 Registerindirekte Adressierung

Bei der registerindirekten Adressierung wird der Inhalt von R 0 oder R 1 als Zeiger auf einen Block von maximal 256 Byte verwendet. Je nach Befehlsart werden entweder auf die ersten 128 Byte (beim 8032/52 256 Bytes) des internen RAM oder die unteren 256 Byte des externen Datenspeichers, der über die \overline{WR} oder \overline{RD} -Steuerleitung selektiert wird, zugegriffen. Auf die speziellen Funktionsregister kann

mit dieser Adressierungsart allerdings nicht zugegriffen werden.

Beispielsweise kopiert der Befehl <MOV A, @ RO> den Inhalt der durch Register R0 adressierten Speicherzelle in den Akkumulator. Für den Zugriff auf den gesamten externen 64 kByte Datenspeicher-Bereich reicht das 8-Bit-Register R 0 bzw. R 1 nicht aus. Für die Adressierungen sind zwei 8-Bit-Register, die dann als ein 16-Bit-Indexregister verwendet werden (DPL + DPH = DPTR), vorgesehen.

2.19.4 Unmittelbare Adressierung

Die unmittelbare Adressierung erlaubt es, konstante Werte im Programmablauf zu übernehmen. Diesen Konstanten wird grundsätzlich eine „#“ vorangestellt, um eine Verwechslung mit den speziellen Funktionsregistern auszuschließen. Beispielsweise wird durch den Befehl <MOV A, #010H> der Akkumulator mit dem konstanten Wert „16“ geladen.

2.19.5 Indirekte Adressierung durch Indexregister

Diese Adressierungsart erleichtert den Zugriff auf die im Programmspeicher stehende Tabellen. Die Speicheradresse ergibt sich aus der Summe des Akkumulators und des 16-Bit-Daten-Pointers (DPTR) oder des 16-Bit-Programm-Counters (PC, aktueller Programmzählerstand). Der Zugriff erfolgt mit den Befehlen <MOVC A, @A + DPTR> oder <MOVC A, @A + PC>, die wir im weiteren Verlauf der Artikelserie noch genauer behandeln.

2.20 Mnemonik

Ein Befehl der vom Mikrocontroller abgearbeitet wird, besteht im Falle eines 8-Bit-Mikrocontrollers aus einer Folge von acht logischen Nullen oder Einsen. Daraus ergeben sich 256 Kombinationsmöglichkeiten. Eine solche Kombination gelangt mit jedem Befehlszugriff in den Befehlsdecoder der CPU (Central Processing Unit), die daraufhin den Prozessor zu ganz bestimmten Aktionen veranlaßt.

Mit Ausnahme der Sprungbefehle wird der Programmzähler (PC, Program-Counter) nach der Abarbeitung jedes Befehls, um 1 erhöht und der nächste Befehl abgearbeitet.

Bei Sprungbefehlen wird entsprechend die neue Zieladresse in den PC geladen und der folgende Befehl ab der neuen ROM-Adresse eingelesen.

Nach dem Zurücksetzen (Reset) des Mikroprozessors ist der PC auf 0 gesetzt und

bewirkt somit, daß der erste Befehl ab der Adresse 0 gelesen wird.

Um eine bessere Übersicht zu erreichen, teilt man eine 8stellige Dualzahl in einen 4stelligen unteren Teil (Low-Nibbel) und einen 4stelligen oberen Teil (High-Nibbel) auf (hexadezimal Darstellung). Da nun aber diese Hexadezimalzahlen ebenfalls für den Programmierer sehr schwer zuzuordnen und zu behalten sind, wurden für die einzelnen Befehle kurze Buchstabenkombinationen, die das wesentliche eines Codes in Erinnerung rufen, eingeführt. Diese sogenannten Mnemoniks lassen sich wesentlich einfacher erlernen und vermindern außerdem die Fehleranfälligkeit beim Programmieren.

Da der Mikroprozessor natürlich diese Mnemoniks nicht direkt versteht, müssen

Unterschiedliche Adressierungsarten ermöglichen den universellen Einsatz der MCS-51-Familie

diese in maschinenlesbare Form übersetzt werden. Hierzu wird ein sogenannter Assembler benötigt, der die Übersetzung in die entsprechenden Maschinenbefehle übernimmt, worauf wir im Verlauf der Artikelserie aber noch näher eingehen.

2.21. Testprogramme

Um die Funktionsweise der nachfolgend beschriebenen Befehle auch praktisch zu verdeutlichen, wurde von ELV ein Test-EPROM mit der Bezeichnung ELV9477 erstellt, mit dessen Hilfe die Funktionsweise unterschiedlicher Befehle praktisch nachvollzogen werden kann. Zum Betrieb wird die in Kapitel 2.17 vorgestellte Testschaltung, bestehend aus einer Schalter- und Leuchtdiodenplatine sowie einer Programm-Auswahlschaltung, die mit zwei 10fach-Schaltern bestückt ist, benötigt.

Zur Verdeutlichung der Funktionsweise der einzelnen Befehle ist jeweils eine Testprogramm-Nummer angegeben, die über S 1 und S 2 eingestellt wird. S 2 ist demnach auf die Zehnerstelle der angegebenen Nummer und S 1 auf die Einerstelle einzustellen.

Nach dem Zurücksetzen des Mikroprozessors fragt dieser programmgesteuert die Schalterstellung der beiden Drehschalter ab und verzweigt sich in die dazugehörigen Testprogramme.

Zum Wechseln der Testprogramme ist zunächst über die Drehschalter die neue Testprogramm-Nummer einzustellen und anschließend die Reset-Taste kurz zu betätigen, damit der Prozessor mit einem neuen Programmstart das neu angewählte Testprogramm ausführen kann.

2.22 Abkürzungen

Für das Verständnis der unterschiedlichen Prozessorbefehle werden im Verlauf der weiteren Beschreibung unterschiedliche Abkürzungen für die Register, Konstanten und Adressierungsarten verwendet, die hier kurz erläutert werden sollen. So bedeuten:

- Rr Ein Register R 0 bis R 7 der gerade selektierten Registerbank
- direct 8-Bit-Adresse eines internen Datenspeichersplatzes. Liegt die Zahl im Bereich 0 bis 127, so wird damit ein Speicherplatz des internen RAMs angesprochen, während anderweitig (128 bis 255) ein spezielles Funktionsregister (SFR) wie z. B. ein EA-Port, Status- oder Steuerregister angesprochen wird.
- @ Ri 8-Bit-Speicherplatz des internen RAMs (0 bis 127 / beim 8032/52 : 0 bis 255), der durch Register R 0 oder R 1 indirekt adressiert wird.
- #data 8-Bit-Konstante, die direkt als Parameter zu einem Befehl zugehörig ist.
- #data16 16-Bit-Konstante, die zu einem Befehl zugehörig ist.
- addr16 16-Bit-Zieladresse, die an einem Speicherplatz innerhalb des 64 k-Programmspeicher-Adreßbereiches verzweigt.
- adr11 11-Bit-Zieladresse, die sich innerhalb der aktuellen 2kByte-Seite des Programmspeichers befindet.
- rel Relative Sprungadresse, die einen Sprung relativ zum aktuellen PC-Stand um -128 bis +127 Byte relativ zum ersten Byte des folgenden Befehls erlaubt.
- bit Adressiert ein direkt adressierbares Bit im internen RAM bzw. speziellen Funktionsregister. Liegt der Wert im Bereich 0 bis 127, so wird damit 1 Bit im internen RAM-Speicher bei der Byte-Adresse 20H bis 2FH angesprochen. Anderenfalls (128 bis 255) handelt es sich jeweils um 1 Bit in einem der bitadressierbaren speziellen Funktionsregister (z. B. einzelnen Port-Pins).

Im nächsten Teil dieser Artikelserie wenden wir uns dem Befehlsatz der MCS-51-Familie zu. 