MPS1-Praktikum - Versuch 2

Die Lösungen aller drei Aufgaben sind für das Testat mit einem in der Vorbereitung erstellten **PAP / Struktogramm** und einer **ausgedruckten Übersetzungsliste** zu dokumentieren!!!

1. Aufgabe: Lesen und Schreiben im externen Programm- und Datenspeicher

Für den **externen Programmspeicher** ist ab Adresse 0000 ein Programm für das Lesen und Beschreiben des gemischten Programm / Daten-Speichers (Adressen 0000 h - 7FFF h) zu erstellen.
Das Low-Byte der Adresse werde an den Kippschaltern des Portes P5 und das High-Adreßbyte an Port C eingestellt

Der Inhalt der adressierten Speicherzelle werde immer in einer Schleife auf Port P1 (P1-LEDs) ausgegeben
Die P1-Kippschalter müssen dazu alle geöffnet sein.!!!

Wenn gilt, P3.2 = 0 dann soll das Kippschalterbitmuster von Port P4 ***einmal*** in die adressierte Speicherzelle geschrieben werden.
Dieses **einmalige Schreiben** soll ein unkontrolliertes Schreiben in den Speicher beim Einstellen der Adressen vermeiden.
Eine andere Lösung des Vermeidens von unkontrolliertem Schreiben besteht darin, die manuelle Veränderung der Adresse nur bei P3.2 = 1 durchzuführen (unkontrolliertes Lesen ist unkritisch), was allerdings vom Anwender erhöhte Aufmerksamkeit erfordert

Das Programm arbeite in Endlosschleife, soll also nur durch RESET abgebrochen werden können.

a) Geben Sie zur Kontrolle einen Teil Ihres eigenen Programmes (ab Adresse 0000 h) byteweise auf die P1-LEDs aus.

b) Schreiben Sie mit diesem Programm in die externen Speicherzellen von Adresse 0040 h bis 0044 h das jeweils zugehörige Low-Adress-Byte. Kontrollieren Sie mittels DX - Befehl.

c) Ändern Sie Ihr obiges Programm durch das Vorschalten folgender zwei Programmzeilen ab:

```
          ORG   0000h
LISTE:    DB    0,1,2,3,4,5,6,7,8,9
          DB    10,11,12,13,14,15
          MOV   ----          ; erster Programmbefehl
          -----  ----         ; des ursprünglichen Programmes
```

Was wird durch den **Assembler-Pseudobefehl DB** bewirkt ?. (auch Direktive oder Assembler-Anweisung)

Was passiert, wenn Sie versuchsweise dieses modifizierte Programm mit G 0000 starten ?.
Versuchen Sie mit Hilfe von **Disassembler (Befehl U)** und **Einzelschrittmodus (Trace-Befehl T)** den Ablauf im Detail zu erklären
Wie müssen Sie nun für eine korrekte Funktion Ihr Programm starten ?.

Kontrollieren Sie mit Ihrem Programm den externen Speicherinhalt ab Adresse 0000h bis 000Fh .

2. Aufgabe: Interner, nur direkt adressierbarer Datenspeicher bzw SFR-Bereich *(Skript Seite 11 und 72)*

a)Es soll der Inhalt der internen, **nur direkt adressierbaren Spezial-Funktions-Register** in einer Endloschleife auf Port P5 (P5-LEDs) ausgegeben werden.
Die Direkt-Adresse 80 - FF h des gewünschten SFRs werde an den Kippschaltern von Port P4 eingestellt.

Hinweis: Die Direkt-Adresse muß dazu **direkt** in das entsprechende **Operanden-Byte des Ausgabebefehles MOV dadr1, dadr2** im Maschinenprogramm geschrieben werden.
Dies ergibt dann ein sogn. **selbstmodifizierendes Programm**, das nur in einem Schreib-Lese-Speicher RAM ablauffähig ist.
Diese **Programm-Selbstmodifikation stellt eine „verbotene" Programmiertechnik dar !!!** – allerdings gibt es bei diesem Prozessor keine andere Lösungsmöglichkeit
Beachten Sie, daß die Anordnung der drei Bytes des Befehls **MOV dadr1,dadr2** im Programmspeicher folgende ist:

```
0200  85      ( OpCode MOV ..)
0201  dadr2
0202  dadr1
```

Vervollständigen Sie das angegebene, rudimentäre Programm mit zwei Befehlen und einem Sprungziel.

```
        ORG   300h
          -     -                    ; P4 für Einlesen vorbereiten
        MOV   DPTR,#LABEL+1          ; Assembler berechnet Adresse
          -     -
        MOVX  @DPTR,A
LABEL:  MOV    P5,00                 ; Direkt-Adr. 00 nur Platzhalter
        SJMP   -

        END
```

Prüfen Sie das Programm, indem Sie die Adresse von **SFR P1** einstellen und mit den Kippschaltern P1.x den Inhalt verändern.
Stellen Sie die **Adressen E0 h,**

82 h

und **83 h**

ein und erklären Sie das auf P5 angezeigte Ergebnis.

b) Erstellen Sie ein weiteres Programm, das ein **Beschreiben der SFRs** (Adresse v. P4) mit dem Kippschalter - bitmuster von Port P5 nach P1.0 = 0 möglich macht (nur Schreiben gefordert !!).

Nach P1.1 = 0 soll **unabhängig von P1.0 unbedingt** in das Betriebssystem zurückgesprungen werden.

Testen Sie das Programm durch Schreiben von Bitmustern in das **niederwertige Halbbyte (Low-Nibble) von P1 (P1.0 - P1.3).** Spielen Sie die Bitmuster **xxxx0011, xxxx1111, xxxx0010, xxxx0001 und xxxx0000** durch und erklären Sie die beobachteten Effekte.

Überprüfen Sie nach einem **Rücksprung durch P1.1** in das Betriebssystem mit dem DD - Befehl das Ergebnis des Beschreibens der SFRs CCL1 mit dem Testbitmuster AA H.

Testen Sie genauso das Beschreiben des Akkus ACC.
Wenn Sie das Testbitmuster nach dem P1.1-Rücksprung nicht mehr im Akku vorfinden, dann untersuchen Sie mit Hilfe des TRACE-Befehles wo der Akku (evtl. auch außerhalb ihres Programmes) verändert wird.

Prüfen Sie, ob Sie auch nach Abbruch des Programmes mit RESET den eingegebenen Inhalt in den SFRs finden und erklären Sie das Ergebnis ?

Aufgabe 3: Lauflicht *(Skript Seite 44)*

Auf den 16 LEDs von Port C und Port P5 soll ein nach **links laufendes Licht (1 Bit)** erzeugt werden. (<- PC-P5 <-) Dieses Licht soll zyklisch umlaufen, also einen 16 Bit Ringzähler bilden.

Mit **P1.0 = 0** soll das Lauflicht gestartet, (danach hat P1.0 keinen Einfluß mehr)
solange **P1.1 = 0** gilt, soll das Lauflicht angehalten
und mit **P1.2 = 0** soll das Programm **unbedingt** abgebrochen und in das Betriebsystem zurückgesprungen werden. (unbedingt bedeutet nicht sofort!! -> noch keine Interrupts verwenden)

Die Lichtwechsel sollen **jede Sekunde** stattfinden. Für die dafür nötige Zeitverzögerung können Sie das unten angegebene Unterprogramm ZEITVERZ verwenden, wobei Sie die Initialisierungskonstanten der drei Register R0, R1 und R2 noch berechnen müssen.

Hinweis: Es kann nur im Akku rotiert werden und die Übergabe zwischen Low-Byte (P5) und High-Byte (Port C) muß über das **Carry-Bit CY** erfolgen (RLC A).
Die einfachste Lösung besteht darin, die Bitmuster von P5 und Port C nacheinander in den Akku zuladen, mit RLC A den Akkuinhalt jeweils um 1 Bit nach links zu schieben und dann den Akkuinhalt nach P5 oder Port C zurückzuschreiben z.B.: .

```
MOV   A,P5
RLC   A
MOV   P5,A
MOVX  A, @DPTR
RLC   A
MOVX  @DPTR, A      ; Übergang von PC.7 auf P5.0 beachten !!
```

Port C des 8255 wird hierfür auf Ausgabe initialisiert und das bedeutet, daß beim Lesen von Port C der Inhalt des Ausgaberegisters (der internen D-Flipflops) und nicht die anliegenden Portpegel gelesen werden.

Die Verzögerungszeit des Unterprogramms ZEITVERZ ist durch die Berechnung der drei 8Bit-Konstanten **8konstx** so festzulegen, daß die Anzahl der Maschinenzyklen (1 µsec) der drei geschachtelten Schleifen 10^6 bzw. 1 Sekunde ergibt.

```
ZEITVERZ:  MOV   R0, #8konst0
EINS:      MOV   R1, #8konst1
ZWEI:      MOV   R2, #8konst2
DREI:      DJNZ  R2, DREI
           DJNZ  R1, ZWEI
           DJNZ  R0, EINS
           RET
```