Prof. Komar **SS 2002**
KoSI 4 **Name:** **Matr.Nr.:**

Mikrocontroller in der Telekommunikation

Skripte, Umdrucke, Kopien und handschriftliche Aufzeichnungen sind zugelassen.
Alle Lösungen sollen auf dem Übungsgerät MVUS 80535 ablauffähig sein.
Alle für die korrekte Funktion der Programme benötigten Initialisierungen sind anzugeben, d.h. bestimmte RESET- oder Betriebssysteminitialisierungen dürfen nicht vorausgesetzt werden.
Kennzeichnen Sie alle Lösungsblätter mit **Name, Matr.Nr. und Aufgabennummer** !!!!
Dieses Aufgabenblatt mitabgeben !!!

Aufgabe	1 + PAP	2	3	gesamt	Note
mögliche Punkte	94 + 36	75	75	280	
Ergebnis					

Aufgabe 1
Nach einer **positiven Flanke an P1.0** soll **schnellstmöglich** über einen Zeitraum von **10 Sekunden** jede **negative P1.0-Flanke** in einem **zweistelligen Dezimalzähler 00 - 99** gezählt werden.
Nach 10 Sekunden bleiben P1.0-Flanken wirkungslos, bis durch eine **positive Flanke von P1.1** die **Zählbereitschaft aktiviert** wird und nach einer positiven P1.0-Flanke der BCD-Zähler erneut für weiter 10 -Sekunden auf dem **vorherigen Zählerstand** weiterzählt.
Die **aktive Zählphase** von 10 Sekunden soll durch **P1.7 = 1** signalisiert werden und ist mit **Timer2** zu realisieren
Der **aktuelle Zählerstand** werde **fortlaufend** und **konsistent** auf dem **LCD-Display** in der **oberen rechten Ecke** dargestellt.
Nach Ausgabe des **maximalen Zählerstandes 99** auf das LCD-Display soll das Programm **korrekt beendet** und in das Betriebssystem zurückgesprungen werden.

Geben Sie einen **Programmablaufplan** an.

Aufgabe 2
Es ist eine **Unterprogramm MUL8DUAL** zu schreiben, das die beiden auf dem Stapel übergebenen **vorzeichenbehafteten** 8-Bit Dualzahlen multipliziert und das **16-Bit-Ergebnis** auch auf dem Stapel zurückliefert.

```
PUSH   Operand 1
PUSH   Operand 2
LCALL  MUL8DUAL
POP    High-Ergebnis-Byte
POP    Low-Ergebnis-Byte
```

Durch das UP sollen keine Registerinhalte verändert werden.
Falls nötig, kann eine eventuelle Zahlenbereichsüberschreitung des 16-Bit-Ergebnisses durch CY=1 signalisiert werden.

Aufgabe 3
Am MVUS-System soll nach einem **RESET** das unten angegebene Programm mit G 0000 gestartet werden.
Disassemblieren Sie und **kommentieren** Sie stichwortartig (ganz knapp), was das Programm macht.

Welche Bitmuster erscheinen an welchem Port der MVUS-Anzeigebox wenn an Port P1 und P4 folgende Bitmuster eingestellt sind:

Port P4 -> 8Fh **Port P1 -> 80h** **=>**

Wo verändert sich was, wenn nach dem Start des Programm das Bitmuster an Port P4 verändert wird ??

Wie kann das Programm **beendet** werden bzw. wie beendet sich das Programm ?

Dokumentieren Sie Ihre Lösung nachfolgend auf dieser Seite

```
ORG    0000h
SETB   EX3
SETB   EAL
SETB   IEX3
LJMP   0FA00h

ORG    0053h
DB     0E5h
DB     90h
DB     90h
DB     00h
DB     06h
DB     0F0h
DB     85h
DB     0E8h
DB     0F8h
DB     0C2h
DB     0AFh
DB     32h

END
```

Prof. Komar **WS 2001**

KOSI4 **Name:** **Matr.Nr.:**

Mikrocontrollertechnik in der Telekommunikation

Skripte, Umdrucke, Kopien und handschriftliche Aufzeichnungen sind zugelassen.
Alle Lösungen sollen auf dem Übungsgerät MVUS 80535 ablauffähig sein.
Alle für die korrekte Funktion der Programme benötigten Initialisierungen sind anzugeben, d.h. bestimmte RESET- oder Betriebssysteminitialisierungen dürfen nicht vorausgesetzt werden.

Aufgabe	1a + PAP	2	3	gesamt	Note
mögliche Punkte	90 + 30	60	45	225	
Ergebnis					

Aufgabe 1
An **P1.0** ist ein **TTL-Rechtecksignal** angeschlossen, dessen **Pulsdauer** von der positiven bis zur negativen Flanke möglichst genau (per Interrupt-Anforderung) fortdauernd gemessen werden soll.
Die maximale mögliche Pulsdauer des ankommenden Rechtecksignals beträgt 99 Milli-Sekunden.

Der aktuelle Meßwert soll fortlaufend auf dem **Bildschirm dezimal** und **konsistent** in konstanter Position in **Millisekunden** von **00 - 99** angezeigt werden. Eine positive Flanke starte den Meßvorgang neu.

Das Programm soll nur durch eine positive und darauffolgende negative Flanke an P1.1 abgebrochen werden, und zwar völlig unabhängig von der Schalterstellung von P1.1 beim Start des Programms.

Das Terminal kommuniziere mit dem 80535 über die serielle Schnittstelle mit
100 Baud, keine Parität, 8 Datenbit, 1 Stoppbit.
Geben Sie einen **Programmablaufplan** an.

Aufgabe 2
Es ist eine **Interrupt-Service-Routine** zu schreiben, in die nach einer **negativen Flanke von P3.2** gesprungen wird.
In dieser **ISR sollen zunächst binär ausgegeben** werden :

- die **Adresse** des **nächsten abzuarbeitenden Befehles** des unterbrochenen Programmes auf **Port A- P1**,

- die **Adresse der Stapelspitze** (Top of Stack) vor der Programmunterbrechung durch P3.2 auf **Port P4**

- der **Akkuinhalt** vor der Programmunterbrechung auf **Port P5**

Solange **P 3. 2 = 0** gilt, soll die ISR nicht verlassen werden.

Zusätzlich sind alle für die **Funktion dieser ISR** nötigen **Initialisierungen** in einer Programmsequenz (kein ablauffähiges Hauptprogramm) zusammengefasst anzugeben.

Aufgabe 3
Am MVUS-System soll nach einem **RESET** das unten, im Maschinencode angegebene Programm, mit G 1000 gestartet werden.
Disassemblieren und kommentieren Sie stichwortartig (ganz knapp) das Programm.

Welches Bitmuster erhalten Sie an welchem Port der MVUS-Anzeigebox wenn Sie an Port P4 und P5 folgende Bitmuster einstellen:

Port P4 -> 90h **Port P5 -> 0Fh =>** [ ]

Wie kann das Programm **beendet** werden. ?

(Dokumentieren Sie Ihre Lösung nachfolgend auf dieser Seite)

```
ORG  1000h
DB   90h
DB   10h
DB   08h
DB   E5h
DB   E8h
DB   F0h
DB   85h
DB   F8h
DB   00h
DB   80h
DB   F8h
```

Prof. Komar **Name:** **Matr.Nr.:**

Leistungsnachweis: Mikrocontroller in der Telekommunikation SS 2001

Skripte, Umdrucke, Kopien und handschriftliche Aufzeichnungen sind zugelassen.
Alle Lösungen sollen auf dem Übungsgerät MVUS 80535 ablauffähig sein.
Alle für die korrekte Funktion der Programme benötigten Initialisierungen sind anzugeben, d.h. bestimmte RESET- oder Betriebssysteminitialisierungen dürfen nicht vorausgesetzt werden.

Aufgabe 1
Nach einer **positiven Flanke** an P1.0 sollen über einen Zeitraum von 10 Sekunden (angezeigt mit P1.7 = 1) **interruptgetrieben** Daten von der seriellen Schnittstelle empfangen werden
In diesen 10 Sekunden sollen weitere positive Flanken an P1.0 keine Auswirkungen haben.
Die Übertragungsrate der seriellen Schnittstelle betrage **110 Bd mit ungerader Paritätssicherung**.

Jedes empfangene Bitmuster, das einem Code für ein **darstellbares ASCII-Zeichen** entspricht, werde auf dem LCD-Display, beginnend links oben, ausgegeben.
Fehlerhaft empfangene Bitmuster werden auf dem LCD-Display durch ein ! markiert.

Nach 10 Sekunden werde der serielle Empfang gesperrt (P1.7 = 0)
Bei einer erneuten positiven Flanke an P1.0 werde der serielle Empfang wieder freigegeben und auf der vorherigen LCD-Position weiter ausgegeben.

In einer **Arbeitsschleife im Hauptprogramm** soll überprüft werden, ob das **LCD-Display vollgeschrieben** ist, und wenn das der Fall ist, soll das Programm beendet werden.

Dokumentieren Sie Ihren Entwurf mit einem **Programmablaufplan.**

Aufgabe 2
Das Unterprogramm **SUB16** soll zwei **vorzeichenbehaftete 16-Bit-Dualzahlen** subtrahieren.
Die beiden 16-Bit-Operanden werden in der Zweier-Komplementdarstellung im DPTR (OP1) und auf dem Stapel (OP2) an das Unterprogramm übergeben

OP1 - OP2 = ERG

und das Ergebnis im DPTR (ERG) zurückgeliefert.
Der Operand 2 OP2 wird in folgender Form vor dem Aufruf des Unterprogramms auf den Stapel gelegt: (die beiden DEC-Befehle sollen den Stapel ausgleichen)

```
PUSH   OP2-LowByte
PUSH   OP2-HighByte
LCALL  SUB16
DEC    SP
DEC    SP  ; Stapelausgleich
```

Wenn das Ergebnis ERG den 16-Bit Zahlenbereich überschreitet, so soll ein **Überlauf** durch das gesetzte Flag **F0 = PSW.5** und ein **Unterlauf** durch das gesetzte Flag **F1 = PSW.1** angezeigt werden.

F0 = 0 F1 = 0 keine Zahlenbereichsüberschreitung
F0 = 1 F1 = 0 Überlauf
F0 = 0 F1 = 1 Unterlauf

Bis auf den DPTR und die beiden Flags F0 und F1 sollen durch das UP keine Registerinhalte verändert werden.

Aufgabe 3

Das **Unterprogramm EBCDIC** soll die Hex-Zahl des Bitmusters im **höherwertigen Akkuhalbbyte** in den 8-Bit-EBCDIC-Code umcodieren und im Akku zurückliefern

Die Ein- und Ausgabe erfolge also im Akku. Außer dem Akku sollen durch das UP keine anderen Registerinhalte verändert werden.

Das UP ist mitsamt eventuell zugehöriger Daten (Tabellen usw.) für den **externen Programmspeicher** zu entwerfen und soll **relozierbar**, d.h. ohne Neuassemblierung überall im externen Programmspeicher ablauffähig sein.

Ziffer	**EBCDIC-Code**
0	1111 0000
1	1111 0001
2	1111 0010
3	1111 0011
4	1111 0100
5	1111 0101
6	1111 0110
7	1111 0111
8	1111 1000
9	1111 1001
A	1100 0001
B	1100 0010
C	1100 0011
D	1100 0100
E	1100 0101
F	1100 0110

Sem.Gem.: **Tisch Nr.:** **Name:**

Aufgabenblatt mitabgeben !!

Prof. Komar **Leistungsnachweis Mikrocomputertechnik** **WS 2000**

Skripte, Umdrucke, Kopien und handschriftliche Aufzeichnungen sind zugelassen.
Alle Lösungen sollen auf dem Übungsgerät MVUS 80535 ablauffähig sein.
Alle für die korrekte Funktion der Programme benötigten Initialisierungen sind anzugeben, d.h. bestimmte RESET- oder Betriebssysteminitialisierungen dürfen nicht vorausgesetzt werden.

Aufgabe 1:

Es ist ein **Unterprogramm UPADRES** zu entwickeln, das auf den LEDs des Übungsboards die **Rücksprungadresse** und die **Adresse der Stapelspitze (Top of Stack)** vor dem LCALL-Aufruf des Unterprogramms ausgibt.
Desweiteren soll der **Inhalt des Datapointers DPTR** vor dem Sprung zum Unterprogramm ausgegeben werden.
Das UP soll keine Registerinhalte verändern.

Geben Sie an, welche Werte von Ihrem Programm auf welchen Ports ausgegeben werden (sollen).

Aufgabe 2:

Über die **serielle Schnittstelle empfangene und mit gerader Parität gesicherte Bitmuster** werden nach der Überprüfung der Paritätssicherung auf das **LCD-Display**, beginnend links oben, ausgegeben.
Ein fehlerhaft empfangenes Zeichen werde nicht auf dem LCD-Display ausgegeben.
Wenn das LCD-Display vollgeschrieben ist, so soll der **serielle Empfang gesperrt** und der gesamte, dargestellte Inhalt auf dem LCD-Displays, beginnend links oben, **interrupt getrieben** auf die serielle Schnittstelle ausgegeben werden.
Anschliessend soll das Programm korrekt beendet und in das Betriebssystem zurückgekehrt werden.
Die serielle Schnittstelle arbeite mit **50 Bd,** und **geradem Paritätsbit**.

Dokumentieren Sie Ihren Entwurf mit einem **Programmablaufplan.**

Hinweise für einfache Programmstruktur: In der ersten Phase wird über die serielle Schnittstelle nur empfangen, in der zweiten nur gesendet.
Programm-Abbruch möglichst in Hauptprogramm-Arbeitsschleife realisieren.

Aufgabe 3:
Das nachfolgende Programm werde nach **RESET mit g 1000** gestartet. Beschreiben Sie stichwortartig die **Funktion** dieses Programms.
An welchem Port wird durch dieses Programm welches Bitmuster ausgegeben ?
Wie kann das Programm beendet werden ?

```
ORG   1000 h
MOV  R0, #0FF h
MOV   @R0, 00
DB     0D8h
DB     0FCh
DB     0E5h
DB     00h
DB     88h
DB     82h
DB     78h
DB     80h
DB     86h
DB     83h
DB     73h
END
```