Prüfung: µ-Computertechnik - Bachelor
Termin: Montag, 19.07.2010; 08:30-10:00
Prüfer: Prof. Walter
Hilfsmittel: beliebig, keine Kommunikationsmittel

Name: _____

Vorname: _____

Studiengang: _____

Labor: _____

USB-Stick: _____

Bitte überprüfen Sie, ob alle Protokolle des Labors in den Webseiten sind, inkl. Protokoll der Präsentation
bitte keine rote Farbe verwenden

(nicht ausfüllen)!

Aufgabe	mögl. Punkte	erreichte Punkte
1	20	
2	30	
3	0	
Gesamt	50	
	Note	

Bearbeiten Sie die Aufgaben nur, falls Sie keine gesundheitlichen Beschwerden haben.

Viel Erfolg!

Bemerkungen: Leeren Sie bei Prüfungsbeginn den Stick. Bitte erstellen Sie die Lösungen auf der eigenen Festplatte und kopieren diese anschließend auf den Stick, bzw. auf die zu Verfügung gestellten Datenträger.
Schreiben Sie in jeden Programmkopf ihren Namen! Bei nicht vorhandenem Namen wird die Lösung NICHT gewertet.

WICHTIG!
Die nachfolgenden Programme müssen mit der PCA0-Einheit per Interrupt gelöst werden. Alle anderen Timer werden für Programmerweiterungen benutzt. Die Oszillatorfrequenz ist 3 MHz. Das cwg-File ist im Projektordner SCHNAPS_xx zu speichern.

Erstellen Sie auf ihrer Festplatte einen Ordner mit: NACHNAME_VORNAME
Kopieren Sie in diesen Ordner die Projektordner SCHNAPS_01 und SCHNAPS_02

Ein Servomotor darf nicht verwendet werden. Das Programm wird für die VC_2-Karte entwickelt und auch dort ausgeführt.

SCHNAPS_01.A51 (20 Punkte)

Schreiben Sie das Programm „SCHNAPS_01.A51" mit folgendem Ablauf:
Nach dem Einschalten wird die Position POS_0 angefahren. Jetzt können vier leere Schnapsgläser in die Aussparungen eingesetzt werden. Bei Tastendruck T3 wird die Tabelle jeweils zyklisch bis POS_3 bearbeitet. Während der Wartezeit wird das Schnapsglas gefüllt. Bem.: Für den Test wurde die Wartezeit verkürzt gewählt.

Position	CPn	DIFF_n	LED 2	LED 1	Wartezeit
POS_0	0EAFFh		Aus	aus	ca. 1,5s
POS_1	0EEDDh	990	Aus	ein	ca. 1,5s
POS_2	0F207h	810	Ein	aus	ca. 1,5s
POS_3	0F527h	800	Ein	Ein	ca. 1,5s

```
$NOMOD51                ;der Modus fuer 8051 wird abgeschaltet
$debug
$nolist                 ;es wird kein Listing fuer reg535 erstellt
$include(C8051F340.inc) ;die C8051F340-spezifischen Daten
$list                   ;es wird ein Listing erstellt
$title (SCHNAPS_01.A51)
;------------------------------------------------------------------------
;Programmbeschreibung
;------------------------------------------------------------------------
;Programm: SCHNAPS_01 entspricht VIER-STELLUNGEN
;faehrt vier Stellungen an:                L2 L1
;POS_0      0EAFF 60159            Ruhestellung  aus aus
;POS_1      0EEDD 61149 990                      aus ein
;POS_2      0F207 61959 810                      ein aus
;POS_3      0F527 62759 800                      ein ein
;
;     46 * 21,45ms ~ 1s
;     69 * 21,45ms ~ 1,5s
;
;Erstellt am: Montag,16. Juli 2010 09:09:04
;Programmiert: Juergen Walter
;
;Verwendete Einspruenge: keine
;
;Verwendete Unterprogramme: keine
;
;
;Verwendete Register und Variable:
;Registerbank(0)
;R2
;
```

```
;Kommentar:
;
;Aenderungen:
;Geaendert am: Montag, 28. Juni 2010 09:09:04
;
;
;------------------------------------------------------------------------
;Initialisierungsteil fuer allgemeine Konstanten
;------------------------------------------------------------------------
SEKUNDEN    EQU         69    ;Wartezeit 1s ueber PCA Interrupts
POS_00L     EQU         0FFh  ;Stellung 00 LOW-Byte
POS_00H     EQU         0EAh  ;Stellung 00 High-Byte
POS_01L     EQU         0DDh  ;Stellung 01 Low-Byte
POS_01H     EQU         0EEh  ;Stellung 01 High-Byte
POS_02L     EQU         007h  ;Stellung 02 Low-Byte
POS_02H     EQU         0F2h  ;Stellung 02 High-Byte
POS_03L     EQU         027h  ;Stellung 03 Low-Byte
POS_03H     EQU         0F5h  ;Stellung 03 High-Byte


CSEG AT 0H                    ;Legt absolute Codesegmentadresse auf 0h
jmp INIT
;
;------------------------------------------------------------------------
;Interrupt-Vektoren
;------------------------------------------------------------------------
ORG 05Bh
clr CF                        ; Interrupt erkannt
call ISR_PCA0                 ; PCA Interrupt Servie Routine
reti

;------------------------------------------------------------------------
;Initialisierungsteil fuer On-Chip Peripherie
;------------------------------------------------------------------------
ORG 100H                      ;Programmstart bei 100H
INIT:
call Init_Device              ;Aufruf zur Initialisierung der Controller
Funktionen

;------------------------------------------------------------------------
;Programmschleife
;------------------------------------------------------------------------
ABFRAGE:
jnb P1.3,VIER_STELLUNGEN      ; T1--> Vier Stellungen
jmp ABFRAGE

VIER_STELLUNGEN:
mov PCA0CPL0,#POS_00L         ;Stellung 0
mov PCA0CPH0,#POS_00H
setb P3.2                     ;L1 aus
setb P3.3                     ;L2 aus
call WARTEN_nS
mov PCA0CPL0,#POS_01L         ;Stellung 1
mov PCA0CPH0,#POS_01H
clr P3.2                      ;L1 ein
setb P3.3                     ;L2 aus
call WARTEN_nS
mov PCA0CPL0,#POS_02L         ;Stellung 2
mov PCA0CPH0,#POS_02H
setb P3.2                     ;L1 aus
```

```
clr P3.3                        ;L2 ein
call WARTEN_nS
mov PCA0CPL0,#POS_03L           ;Stellung 3
mov PCA0CPH0,#POS_03H
clr P3.2                        ;L1 ein
clr P3.3                        ;L2 ein
call WARTEN_nS

jmp ABFRAGE


WARTEN_nS:
clr F0                          ;F0 =0 Sekunde nicht vorbei
mov R2,#SEKUNDEN                ;R2 laden
mov EIE1,#010h                  ;PCA0-Interrupts ermoeglichen
jnb F0,$                        ;warten bis F0=1 --> ausserhalb der Interrupt
                                ;Service Routine - weiterer Interrupt gleicher
                                ;Prioritaet kann wirksam werden
ret

ISR_PCA0:
djnz R2,ISR_PCA0_ENDE           ;Sekunde noch nicht vorbei
mov EIE1,#00h                   ;PCA0-Interrupts sperren
setb F0                         ;genuegend Interrupts
ISR_PCA0_ENDE:
ret


;------------------------------------------------------------------------
; Peripheral specific initialization functions,
; Called from the Init_Device label
;------------------------------------------------------------------------

PCA_Init:
    mov  PCA0CN,    #040h
    anl  PCA0MD,    #0BFh
    mov  PCA0MD,    #009h
    mov  PCA0CPM0,  #0C2h
    mov  PCA0CPL0,  #0FFh
    mov  PCA0CPH0,  #0EAh
    ret

Port_IO_Init:
    ; P0.0  -  Skipped,     Push-Pull,  Digital
    ; P0.1  -  Skipped,     Push-Pull,  Digital
    ; P0.2  -  Skipped,     Push-Pull,  Digital
    ; P0.3  -  Skipped,     Push-Pull,  Digital
    ; P0.4  -  Skipped,     Push-Pull,  Digital
    ; P0.5  -  Skipped,     Push-Pull,  Digital
    ; P0.6  -  Skipped,     Push-Pull,  Digital
    ; P0.7  -  Skipped,     Push-Pull,  Digital

    ; P1.0  -  Skipped,     Push-Pull,  Digital
    ; P1.1  -  CEX0  (PCA), Push-Pull,  Digital
    ; P1.2  -  Unassigned,  Push-Pull,  Digital
    ; P1.3  -  Skipped,     Push-Pull,  Digital
    ; P1.4  -  Unassigned,  Push-Pull,  Digital
    ; P1.5  -  Unassigned,  Push-Pull,  Digital
    ; P1.6  -  Unassigned,  Push-Pull,  Digital
    ; P1.7  -  Unassigned,  Push-Pull,  Digital
```

```
    ; P2.0  -  Unassigned,  Push-Pull,  Digital
    ; P2.1  -  Unassigned,  Push-Pull,  Digital
    ; P2.2  -  Unassigned,  Push-Pull,  Digital
    ; P2.3  -  Unassigned,  Push-Pull,  Digital
    ; P2.4  -  Unassigned,  Push-Pull,  Digital
    ; P2.5  -  Unassigned,  Push-Pull,  Digital
    ; P2.6  -  Unassigned,  Push-Pull,  Digital
    ; P2.7  -  Unassigned,  Push-Pull,  Digital

    ; P3.0  -  Unassigned,  Push-Pull,  Digital
    ; P3.1  -  Unassigned,  Push-Pull,  Digital
    ; P3.2  -  Unassigned,  Push-Pull,  Digital
    ; P3.3  -  Unassigned,  Push-Pull,  Digital
    ; P3.4  -  Unassigned,  Push-Pull,  Digital
    ; P3.5  -  Unassigned,  Push-Pull,  Digital
    ; P3.6  -  Unassigned,  Push-Pull,  Digital
    ; P3.7  -  Unassigned,  Push-Pull,  Digital

    mov  P0MDOUT,   #0FFh
    mov  P1MDOUT,   #0FFh
    mov  P2MDOUT,   #0FFh
    mov  P3MDOUT,   #0FFh
    mov  P0SKIP,    #0FFh
    mov  P1SKIP,    #009h
    mov  XBR1,      #041h
    ret

Oscillator_Init:
    mov  OSCICN,    #081h
    ret

Interrupts_Init:
    mov  IT01CF,    #010h
    mov  IE,        #080h
    ret

; Initialization function for device,
; Call Init_Device from your main program
Init_Device:
    lcall PCA_Init
    lcall Port_IO_Init
    lcall Oscillator_Init
    lcall Interrupts_Init
    ret

end
```

SCHNAPS_02.A51 (30 Punkte)

Durch die Fahrt mit maximaler Geschwindigkeit auf die einzelnen Positionen wird immer wieder Schnaps verschüttet. Schreiben Sie das Programm SCHNAPS_02.A51 so, dass POS_1 bis POS_3 mit Geschwindigkeit von ca. 233 PCA0-Timerschritten/s;
(1/(2 * 21,45ms) *10 Schritte) angefahren werden. POS_0 wird nach der Entnahme der vollen Gläser mit maximaler Schwindigkeit angefahren.

Hilfe:
Es gibt zwei Zustände im Programm SCHNAPS_02:

1. F1=0 → Warten: F0=0 - Wartezeit noch nicht vorbei. F0 = 1 Wartezeit vorbei
2. F1=1 → Langsam fahren:

Durch das Flag F1 können Sie in der ISR_PCA0 jeweils auf einen der beiden Fälle eingehen.
Benutzen Sie die Register:

1. R2 Zaehlregister - Warten
2. R3 Zaehlregister - PCA0-Interrupts langsame Fahrt
3. R4 Zaehlregister – Differenzen:

Übergeben Sie in Register 4 die Anzahl der Differenzen DIFF_n/10 an die ISR_PCA0. Das Unterprogramm FAHRT_GU (Fahrt gegen Uhrzeigersinn) können Sie dann jeweils für die Anfahrt der einzelnen Positionen aufrufen.

```
$NOMOD51                ;der Modus fuer 8051 wird abgeschaltet
$debug
$nolist                 ;es wird kein Listing fuer reg535 erstellt
$include(C8051F340.inc) ;die C8051F340-spezifischen Daten
$list                   ;es wird ein Listing erstellt
$title (SCHNAPS_01.A51)
;------------------------------------------------------------------------
;Programmbeschreibung
;------------------------------------------------------------------------
;Programm: SCHNAPS_01 entspricht VIER-STELLUNGEN
;faehrt vier Stellungen an:              L2 L1
;POS_0      0EAFF 60159       Ruhestellung  aus aus
;POS_1      0EEDD 61149 990                    aus ein
;POS_2      0F207 61959 810                    ein aus
;POS_3      0F527 62759 800                    ein ein
;
; R2 Wartezeitberechnung R2
;           46 * 21,45ms ~ 1s
;         184 * 21,45ms ~ 4s
; R3 Anzahl von Interrupts bis Addition wirksam
; R4 Anzahl der POS_DIFF Schritte
;
;Erstellt am: Montag,16. Juli 2010 09:09:04
;Programmiert: Juergen Walter
;
;Verwendete Einspruenge: keine
;
;Verwendete Unterprogramme: keine
;
;
;Verwendete Register und Variable:
```

```
;Registerbank(0)
;R2 Zaehlregister Warten
;
;Kommentar:
;
;Aenderungen:
;Geaendert am: Montag, 28. Juni 2010 09:09:04
;
;
;------------------------------------------------------------------------
;Initialisierungsteil fuer allgemeine Konstanten
;------------------------------------------------------------------------
SEKUNDEN    EQU   184         ;Wartezeit 4s ueber PCA Interrupts
POS_00L     EQU   0FFh  ;Stellung 00 LOW-Byte
POS_00H     EQU   0EAh  ;Stellung 00 High-Byte
POS_01L     EQU   0DDh  ;Stellung 01 Low-Byte
POS_01H     EQU   0EEh  ;Stellung 01 High-Byte
POS_02L     EQU   007h  ;Stellung 02 Low-Byte
POS_02H     EQU   0F2h  ;Stellung 02 High-Byte
POS_03L     EQU   027h  ;Stellung 03 Low-Byte
POS_03H     EQU   0F5h  ;Stellung 03 High-Byte

DIFF_1      EQU   99    ;99*10
DIFF_2      EQU   81    ;81*10
DIFF_3      EQU   80    ;80*10

C_ADD EQU 10                ;Addition der Schrittweite Comparewert
PCA0_INT EQU 2              ;Anzahl der Interrupts bis Aktion
2*21,45ms=42.9ms
                            ;ca 23,3 Hz 233 Schritte/s 2600/233Hz= 11,2s

CSEG AT 0H                  ;Legt absolute Codesegmentadresse auf 0h
jmp INIT
;
;------------------------------------------------------------------------
;Interrupt-Vektoren
;------------------------------------------------------------------------
ORG 05Bh
clr CF                      ; Interrupt erkannt
call ISR_PCA0               ; PCA Interrupt Servie Routine
reti

;------------------------------------------------------------------------
;Initialisierungsteil fuer On-Chip Peripherie
;------------------------------------------------------------------------
ORG 100H                    ;Programmstart bei 100H
INIT:
call Init_Device            ;Aufruf zur Initialisierung der Controller
Funktionen

;------------------------------------------------------------------------
;Programmschleife
;------------------------------------------------------------------------
ABFRAGE:
jnb P1.3,VIER_STELLUNGEN    ; T1--> Vier Stellungen
jmp ABFRAGE

VIER_STELLUNGEN:
mov PCA0CPL0,#POS_00L       ;Stellung 0
mov PCA0CPH0,#POS_00H
```

```
setb P3.2                       ;L1 aus
setb P3.3                       ;L2 aus
call WARTEN_nS                  ;F1=1 Fahrt F1=0 Warten
mov R4,#DIFF_1
call FAHRT_GU
clr P3.2                        ;L1 ein
setb P3.3                       ;L2 aus
call WARTEN_nS
mov R4,#DIFF_2
call FAHRT_GU
setb P3.2                       ;L1 aus
clr P3.3                        ;L2 ein
call WARTEN_nS
mov R4,#DIFF_3
call FAHRT_GU
clr P3.2                        ;L1 ein
clr P3.3                        ;L2 ein
call WARTEN_nS
jmp ABFRAGE


WARTEN_nS:
clr F1                          ;F1=1 Fahrt F1=0 Warten
clr F0                          ;F0 =0 Sekunde nicht vorbei
mov R2,#SEKUNDEN                ;R2 laden
mov EIE1,#010h                  ;PCA0-Interrupts ermoeglichen
jnb F0,$                        ;warten bis F0=1 --> ausserhalb der Interrupt
                                ;Service Routine - weiterer Interrupt gleicher
                                ;Prioritaet kann wirksam werden
ret

FAHRT_GU:
setb F1                         ;F1=1 Fahrt F1=0 Warten
mov EIE1,#010h                  ;PCA0-Interrupts ermoeglichen
jb F1,$                         ;Warten bis Fahrt vorbei
ret



ISR_PCA0:
jnb F1,ZUSTAND_WARTEN           ;F1=0 -->Springe in Zustand warten

;-------------------------------------------------------------------------
; LANGSAME FAHRT
;-------------------------------------------------------------------------


djnz R3,ISR_PCA0_ENDE           ;Sind bereits n Interrupts aufgetreten?
mov R3,#PCA0_INT                ;Wiederladen des Zaehlregisters
djnz R4,ADDITION_PCA0CP0        ;R4 fuer Anzahl der 10-er Schritte
clr F1                          ;F1=0 --> Zustand warten wird aufgerufen
mov EIE1,#00h                   ;PCA0-Interrupts sperren
ret                             ;Fahrt ENDE

ADDITION_PCA0CP0:
mov A,PCA0CPL0                  ;Compare 0 Low Byte in Akku
clr C                           ;Carry loeschen
add A,#C_ADD
mov PCA0CPL0,A                  ;Low Byte wieder zurueckschreiben
mov A,PCA0CPH0                  ;High Byte Compare holen
```

```
jnc KEINUEBERTRAG_1
inc A
KEINUEBERTRAG_1:
mov PCA0CPH0,A                ;High Byte Compare schreiben

;------------------------------------------------------------------------
; WARTEN
;------------------------------------------------------------------------


ZUSTAND_WARTEN:
jb F1,ISR_PCA0_ENDE           ;Fahrt - F1=1
djnz R2,ISR_PCA0_ENDE         ;Sekunde noch nicht vorbei
                              ;R2 fuer Wartezeit
mov EIE1,#00h                 ;PCA0-Interrupts sperren
setb F0                       ;genuegend Interrupts
ISR_PCA0_ENDE:
ret


;------------------------------------------------------------------------
; Peripheral specific initialization functions,
; Called from the Init_Device label
;------------------------------------------------------------------------

PCA_Init:
    mov  PCA0CN,    #040h
    anl  PCA0MD,    #0BFh
    mov  PCA0MD,    #009h
    mov  PCA0CPM0,  #0C2h
    mov  PCA0CPL0,  #0FFh
    mov  PCA0CPH0,  #0EAh
    ret

Port_IO_Init:
    ; P0.0  -  Skipped,     Push-Pull,  Digital
    ; P0.1  -  Skipped,     Push-Pull,  Digital
    ; P0.2  -  Skipped,     Push-Pull,  Digital
    ; P0.3  -  Skipped,     Push-Pull,  Digital
    ; P0.4  -  Skipped,     Push-Pull,  Digital
    ; P0.5  -  Skipped,     Push-Pull,  Digital
    ; P0.6  -  Skipped,     Push-Pull,  Digital
    ; P0.7  -  Skipped,     Push-Pull,  Digital

    ; P1.0  -  Skipped,     Push-Pull,  Digital
    ; P1.1  -  CEX0  (PCA), Push-Pull,  Digital
    ; P1.2  -  Unassigned,  Push-Pull,  Digital
    ; P1.3  -  Skipped,     Push-Pull,  Digital
    ; P1.4  -  Unassigned,  Push-Pull,  Digital
    ; P1.5  -  Unassigned,  Push-Pull,  Digital
    ; P1.6  -  Unassigned,  Push-Pull,  Digital
    ; P1.7  -  Unassigned,  Push-Pull,  Digital

    ; P2.0  -  Unassigned,  Push-Pull,  Digital
    ; P2.1  -  Unassigned,  Push-Pull,  Digital
    ; P2.2  -  Unassigned,  Push-Pull,  Digital
    ; P2.3  -  Unassigned,  Push-Pull,  Digital
    ; P2.4  -  Unassigned,  Push-Pull,  Digital
    ; P2.5  -  Unassigned,  Push-Pull,  Digital
    ; P2.6  -  Unassigned,  Push-Pull,  Digital
```

```
    ; P2.7  -  Unassigned,  Push-Pull,  Digital

    ; P3.0  -  Unassigned,  Push-Pull,  Digital
    ; P3.1  -  Unassigned,  Push-Pull,  Digital
    ; P3.2  -  Unassigned,  Push-Pull,  Digital
    ; P3.3  -  Unassigned,  Push-Pull,  Digital
    ; P3.4  -  Unassigned,  Push-Pull,  Digital
    ; P3.5  -  Unassigned,  Push-Pull,  Digital
    ; P3.6  -  Unassigned,  Push-Pull,  Digital
    ; P3.7  -  Unassigned,  Push-Pull,  Digital

    mov  P0MDOUT,   #0FFh
    mov  P1MDOUT,   #0FFh
    mov  P2MDOUT,   #0FFh
    mov  P3MDOUT,   #0FFh
    mov  P0SKIP,    #0FFh
    mov  P1SKIP,    #009h
    mov  XBR1,      #041h
    ret

Oscillator_Init:
    mov  OSCICN,    #081h
    ret

Interrupts_Init:
    mov  IT01CF,    #010h
    mov  IE,        #080h
    ret

; Initialization function for device,
; Call Init_Device from your main program
Init_Device:
    lcall PCA_Init
    lcall Port_IO_Init
    lcall Oscillator_Init
    lcall Interrupts_Init
    ret

end
```