Prüfung: µ-Computertechnik - Bachelor
Termin: Montag, 18.07.2011; 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	10	
2	20	
3	20	
Gesamt	50	
	Note	

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

Viel Erfolg!

Bemerkungen: Bitte erstellen Sie die Lösungen auf der eigenen Festplatte im Ordner SS11_NAME! Am Ende der Klausur belassen Sie ihren Rechner am Platz und verlassen den Raum.
Zum Kopieren ihrer Lösung werden Sie jeweils mit Namen im Anschluss an die Klausur aufgerufen.
Schreiben Sie in jeden Programmkopf ihren Namen! Bei nicht vorhandenem Namen wird die Lösung NICHT gewertet.

WICHTIG!
Alle Programme sind für die VC_2-Hardware zu schreiben. Der Sysclk liegt bei 3 MHz. Speichern Sie jeweils die dazugehörige Configuration Wizard Datei!!

Programm „HZ_250“ (10 Punkte)

Die LED_1 an Port 3.2 soll 250-mal pro Sekunde geschaltet werden. Verwenden Sie Timer 0 im Autoreload-Betrieb. TIPP: Verwenden Sie den richtigen „Prescaled Clock“. Mit T1 startet das Programm.

a) Mit welcher Zahl laden Sie das Autoreloadregister?
b) Schreiben Sie das Programm

```
$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 (HZ_250.A51)
;------------------------------------------------------------------------
;Programmbeschreibung
;------------------------------------------------------------------------
;Programm:
;;Mit T1 startet das Programm
;An Port 3.4 liegt LED 3
;LED 3 wird 250 mal pro Sekunde umgeschaltet.
;Sysclk ist mit 3 MHz festgelegt
;Timer 0 im Autoreload-Modus mit Sysclk / 48 --> 62500 Hz / 250 = 250 Hz --
> 4ms
;Reload-Wert 256-250=6
;
;
;
;Erstellt am: Sonntag, 17. Juli 2011 12:35:57
;Programmiert: Prof. Juergen Walter
;
;Verwendete Einspruenge: keine
;
;Verwendete Unterprogramme: keine
;
;
;Verwendete Register und Variable:
;Registerbank(0)
;R2
;
;Kommentar:
;
;Aenderungen:
;Geaendert am: Sonntag, 17. Juli 2011 12:35:57
;
;
;------------------------------------------------------------------------
;Initialisierungsteil fuer allgemeine Konstanten
;------------------------------------------------------------------------


CSEG AT 0H              ;Legt absolute Codesegmentadresse auf 0h
```

```
call Init_Device        ;Aufruf zur Initialisierung der Controller
Funktionen
jmp INIT
;
;-------------------------------------------------------------------------
;Interrupt-Vektoren
;-------------------------------------------------------------------------
ORG 000BH
call ISR_T0       ;Interrupt Service Routine Timer 0
reti

;-------------------------------------------------------------------------
;Initialisierungsteil fuer On-Chip Peripherie
;-------------------------------------------------------------------------
ORG 100H                ;Programmstart bei 100H
INIT:

;-------------------------------------------------------------------------
;Programmschleife
;-------------------------------------------------------------------------

ABFRAGE:
jnb P1.1,START                ;Start des Programmes
jmp ABFRAGE

START:
setb TR0                      ;Timer laeuft
jmp ABFRAGE

ISR_T0:
cpl P3.2                      ;umschalten
ret



;-------------------------------------------------------------------------
; Peripheral specific initialization functions,
; Called from the Init_Device label
;-------------------------------------------------------------------------
PCA_Init:
    anl  PCA0MD,    #0BFh
    mov  PCA0MD,    #000h
    ret

Timer_Init:
    mov  TMOD,      #002h
    mov  CKCON,     #002h
    mov  TH0,       #006h
    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  -  TX0 (UART0), Push-Pull,  Digital
    ; P0.5  -  RX0 (UART0), Push-Pull,  Digital
    ; P0.6  -  Skipped,     Push-Pull,  Digital
    ; P0.7  -  Skipped,     Push-Pull,  Digital
```

```
    ; P1.0  -  CEX0  (PCA), Push-Pull,  Digital
    ; P1.1  -  CEX1  (PCA), Push-Pull,  Digital
    ; P1.2  -  CEX2  (PCA), Push-Pull,  Digital
    ; P1.3  -  CEX3  (PCA), Push-Pull,  Digital
    ; P1.4  -  CEX4  (PCA), Push-Pull,  Digital
    ; P1.5  -  Skipped,     Push-Pull,  Digital
    ; P1.6  -  Skipped,     Push-Pull,  Digital
    ; P1.7  -  Skipped,     Push-Pull,  Digital

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

    ; P3.0  -  Skipped,     Push-Pull,  Digital
    ; P3.1  -  Skipped,     Push-Pull,  Digital
    ; P3.2  -  Skipped,     Push-Pull,  Digital
    ; P3.3  -  Skipped,     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,    #0CFh
    mov  P1SKIP,    #0E0h
    mov  P2SKIP,    #0FFh
    mov  P3SKIP,    #00Fh
    mov  XBR0,      #001h
    mov  XBR1,      #045h
    ret

Oscillator_Init:
    mov  OSCICN,    #081h
    ret

Interrupts_Init:
    mov  IT01CF,    #010h
    mov  IE,        #082h
    ret

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

end
```

Vorgehensweise Bewertung:

1. Programm kompilieren – falls Fehler 0-Punkte
2. Remote Debug überprüfen
3. Unter Debug ausführen
4. Läuft → 10 Punkte
5. Falls Fehlfunktion→jeweils 1 Punkt
   1. IE
   2. OSCICN
   3. Timer_Init
   4. Cpl 3.2
   5. TR0
   6. Jnb P1.1
   7. Org 000B
6. Falls Funktion aber nicht mit Vorgabe -2P

Programm „DUTY_VARIATION“ (20 Punkte)

Mit T1 startet das Programm.
An Port 3.2 liegt LED 1.
Die Helligkeit von LED 1 wird mit PCA0 von ca. 0% ~ 0 bis ca. 99% ~ 65000 Wert des Compare-Registers innerhalb von ca. 5s gesteuert.
Verwenden Sie hierzu die mit Programm „HZ_250“ erzeugte Frequenz von 250 Hz.
Das Programm hat keine Abbruchbedingung.
Tipp: Legen Sie den CEX0-Ausgang direkt auf P3.2

```
$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 (DUTY_VARIATION.A51)
;------------------------------------------------------------------------
;Programmbeschreibung
;------------------------------------------------------------------------
;Programm:
;Mit T1 startet das Programm
;An Port 3.4 liegt LED 3
;
;Sysclk ist mit 3 MHz festgelegt
;Timer 0 im Autoreload-Modus mit Sysclk / 48 --> 62500 Hz / 250 = 250 Hz --
> 4ms
;Reload-Wert 256-250=6
;
;LED 3 wird mit PCA0 von ca.  0% ~ 0 bis ca. 99% ~ 65000 gesteuert
;65000/5=13000 Schritte pro Sekunde → 13000/250=52 →
;250 mal/s muss 52 addiert werden.
;
;Erstellt am: Sonntag, 17. Juli 2011 12:35:57
;Programmiert: Prof. Juergen Walter
;
;Verwendete Einspruenge: keine
;
;Verwendete Unterprogramme: keine
;
;
```

```
;Verwendete Register und Variable:
;Registerbank(0)
;R2
;
;Kommentar:
;
;Aenderungen:
;Geaendert am: Sonntag, 17. Juli 2011 12:35:57
;
;
;-------------------------------------------------------------------------
;Initialisierungsteil fuer allgemeine Konstanten
;-------------------------------------------------------------------------
ADD_CONST EQU 52        ;Additionskonstante - Inkremente für PCA0CP

CSEG AT 0H              ;Legt absolute Codesegmentadresse auf 0h
jmp INIT
;
;-------------------------------------------------------------------------
;Interrupt-Vektoren
;-------------------------------------------------------------------------
ORG 000BH
call ISR_T0       ;Interrupt Service Routine Timer 0
reti

;-------------------------------------------------------------------------
;Initialisierungsteil fuer On-Chip Peripherie
;-------------------------------------------------------------------------
ORG 100H                ;Programmstart bei 100H
INIT:
call Init_Device        ;Aufruf zur Initialisierung der Controller
Funktionen
mov PCA0CPM0,#0CBh            ;PWM variieren                    ;
;-------------------------------------------------------------------------
;Programmschleife
;-------------------------------------------------------------------------

ABFRAGE:
jnb P1.1,START                ;Start des Programmes
jmp ABFRAGE

START:
setb TR0                      ;Timer laeuft
jmp ABFRAGE

ISR_T0:
call ADDITION_PCA0_CP0

ret

ADDITION_PCA0_CP0:
mov A,PCA0CPL0                ;Unteres Byte holen
clr C                         ;Carry = 0
add A,#ADD_CONST              ;+52
mov PCA0CPL0,A                ;Wieder zurueckschreiben
mov A,PCA0CPH0
addc A,#0                     ;Addition C
mov PCA0CPH0,A                ;

ret
```

```
                                        ;

;-------------------------------------------------------------------------
; Peripheral specific initialization functions,
; Called from the Init_Device label
;-------------------------------------------------------------------------
PCA_Init:
    mov  PCA0CN,    #040h
    anl  PCA0MD,    #0BFh
    mov  PCA0MD,    #008h
    mov  PCA0CPM0,  #0C2h
    ret

Timer_Init:
    mov  TMOD,      #002h
    mov  CKCON,     #002h
    mov  TH0,       #006h
    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  -  Skipped,     Push-Pull,  Digital
    ; P1.2  -  Skipped,     Push-Pull,  Digital
    ; P1.3  -  Skipped,     Push-Pull,  Digital
    ; P1.4  -  Skipped,     Push-Pull,  Digital
    ; P1.5  -  Skipped,     Push-Pull,  Digital
    ; P1.6  -  Skipped,     Push-Pull,  Digital
    ; P1.7  -  Skipped,     Push-Pull,  Digital

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

    ; P3.0  -  Skipped,     Push-Pull,  Digital
    ; P3.1  -  Skipped,     Push-Pull,  Digital
    ; P3.2  -  CEX0  (PCA), 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,    #0FFh
    mov  P2SKIP,    #0FFh
    mov  P3SKIP,    #003h
    mov  XBR1,      #041h
    ret

Oscillator_Init:
    mov  OSCICN,    #081h
    ret

Interrupts_Init:
    mov  IT01CF,    #010h
    mov  IE,        #082h
    ret

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

end
```

Vorgehensweise Bewertung:

1. Programm kompilieren – falls Fehler 0-Punkte
2. Remote Debug überprüfen
3. Unter Debug ausführen
4. Läuft → 20 Punkte
5. Falls Fehlfunktion→
   1. 16-Bit Addition 5 Punkte
   2. Schrittweite 52 – 2 Punkte
   3. Timer_Init
6. Falls Funktion aber nicht mit Vorgabe -2P

Programm „DUTY_VAR_AUF_AB“ (20 Punkte)

Erweitern Sie das Programm „DUTY_VARIATION“. Zuerst zählen Sie mit der PCA-Einheit bis ungefähr 65000 und anschließend wieder von 65000 bis ca. 0. Wiederholen Sie das Ganze bis in alle Ewigkeit…;-)
TIPP: Es genügt die Abfrage des PCA0CPH0 mit dem Befehl cjne A,#const8,rel für die beiden Fälle: „Aufwärts“ – „Abwärts“.

```
$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 (DUTY_VAR_AUF_AB.A51)
;-------------------------------------------------------------------------
;Programmbeschreibung
;-------------------------------------------------------------------------
;Programm:
;Mit T1 startet das Programm
```

```
;An Port 3.4 liegt LED 3
;
;Sysclk ist mit 3 MHz festgelegt
;Timer 0 im Autoreload-Modus mit Sysclk / 48 --> 62500 Hz / 250 = 250 Hz --
> 4ms
;Reload-Wert 256-250=6
;
;LED 1 wird mit PCA0 von ca.  0% ~ 0 bis ca. 99% ~ 65000 gesteuert
;F0=0 Addition - Heller
;F0=1 Subtraktion - Dunkler
;Abfrage PCA0CPH0
;Erstellt am: Sonntag, 17. Juli 2011 12:35:57
;Programmiert: Prof. Juergen Walter
;
;Verwendete Einspruenge: keine
;
;Verwendete Unterprogramme: keine
;
;
;Verwendete Register und Variable:
;Registerbank(0)
;R2
;
;Kommentar:
;
;Aenderungen:
;Geaendert am: Sonntag, 17. Juli 2011 12:35:57
;
;
;-------------------------------------------------------------------------
;Initialisierungsteil fuer allgemeine Konstanten
;-------------------------------------------------------------------------
ADD_CONST EQU 52        ;Additionskonstante - Inkremente für PCA0CP

CSEG AT 0H              ;Legt absolute Codesegmentadresse auf 0h
jmp INIT
;
;-------------------------------------------------------------------------
;Interrupt-Vektoren
;-------------------------------------------------------------------------
ORG 000BH
call ISR_T0       ;Interrupt Service Routine Timer 0
reti

;-------------------------------------------------------------------------
;Initialisierungsteil fuer On-Chip Peripherie
;-------------------------------------------------------------------------
ORG 100H                ;Programmstart bei 100H
INIT:
call Init_Device        ;Aufruf zur Initialisierung der Controller
Funktionen
mov PCA0CPM0,#0CBh            ;PWM variieren                    ;
;-------------------------------------------------------------------------
;Programmschleife
;-------------------------------------------------------------------------

ABFRAGE:
jnb P1.1,START                ;Start des Programmes
jmp ABFRAGE
```

```
START:
setb TR0                        ;Timer laeuft
clr F0
jmp ABFRAGE

ISR_T0:
jb F0,SUB_PCA0_CP0
call ADDITION_PCA0_CP0
jnb F0,ISR_T0_ENDE
call SUB_PCA0_CP0
ISR_T0_ENDE:
ret

ADDITION_PCA0_CP0:
mov A,PCA0CPL0                  ;Unteres Byte holen
clr C                           ;Carry = 0
add A,#ADD_CONST                ;+52
mov PCA0CPL0,A                  ;Wieder zurueckschreiben
mov A,PCA0CPH0
addc A,#0                       ;Addition C
mov PCA0CPH0,A                  ;
cjne A,#0FEh,UNGLEICH_FE;Abfrage auf FE
setb F0                         ;Auf Subtraktion umschalten
UNGLEICH_FE:
ret

SUB_PCA0_CP0:
mov A,PCA0CPL0                  ;Unteres Byte holen
clr C                           ;Carry = 0
subb A,#ADD_CONST               ;-52
mov PCA0CPL0,A                  ;Wieder zurueckschreiben
mov A,PCA0CPH0
jnc KEIN_SUB_UEB
dec A
KEIN_SUB_UEB:
mov PCA0CPH0,A                  ;
cjne A,#0,UNGLEICH_0_SUB;Abfrage auf 0
clr F0                          ;Auf Addition umschalten
UNGLEICH_0_SUB:
ret
                        ;

;-------------------------------------------------------------------------
; Peripheral specific initialization functions,
; Called from the Init_Device label
;-------------------------------------------------------------------------
PCA_Init:
    mov  PCA0CN,    #040h
    anl  PCA0MD,    #0BFh
    mov  PCA0MD,    #008h
    mov  PCA0CPM0,  #0C2h
    ret

Timer_Init:
    mov  TMOD,      #002h
    mov  CKCON,     #002h
    mov  TH0,       #006h
    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  -  Skipped,     Push-Pull,  Digital
    ; P1.2  -  Skipped,     Push-Pull,  Digital
    ; P1.3  -  Skipped,     Push-Pull,  Digital
    ; P1.4  -  Skipped,     Push-Pull,  Digital
    ; P1.5  -  Skipped,     Push-Pull,  Digital
    ; P1.6  -  Skipped,     Push-Pull,  Digital
    ; P1.7  -  Skipped,     Push-Pull,  Digital

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

    ; P3.0  -  Skipped,     Push-Pull,  Digital
    ; P3.1  -  Skipped,     Push-Pull,  Digital
    ; P3.2  -  CEX0  (PCA), 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,    #0FFh
    mov  P2SKIP,    #0FFh
    mov  P3SKIP,    #003h
    mov  XBR1,      #041h
    ret

Oscillator_Init:
    mov  OSCICN,    #081h
    ret

Interrupts_Init:
    mov  IT01CF,    #010h
    mov  IE,        #082h
    ret

; Initialization function for device,
; Call Init_Device from your main program
Init_Device:
    lcall PCA_Init
```

```
    lcall Timer_Init
    lcall Port_IO_Init
    lcall Oscillator_Init
    lcall Interrupts_Init
    ret

end
```

Vorgehensweise Bewertung:

1. Programm kompilieren – falls Fehler 0-Punkte
2. Remote Debug überprüfen
3. Unter Debug ausführen
4. Läuft → 20 Punkte
5. Falls Fehlfunktion→
   1. 16-Bit Subtraktion 5 Punkte
   2. Timer_Init
6. Falls Funktion aber nicht mit Vorgabe -2P

Häufigste ärgerliche Fehler:

1. Programm nicht kompiliert
2. Compiler zeigt Fehler