HOCHSCHULE ESSLINGEN FAKULTÄT INFORMATIONSTECHNIK

Wintersemester 2008/09		Blatt Nr.:	**1 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

Aufgabe 1: Verständnisfragen (20 Punkte)

1.1 Erläutern Sie die verschiedenen Bereiche im Adressraum des im Labor verwendeten Microcontrollers MC9S12DP256 von Freescale. Geben Sie jeweils die Anfangs- und Endadresse des Bereichs an und die Funktion bzw. den Zweck.

Lösung zu Aufgabe 1.1:

$0000-$03FF: On-Chip-Peripherie
$0400-$0FFF: EEPROM, für nichtflüchtige Variablen
$1000-$3FFF: RAM, für Stack, Variablen
$4000-$FFFF: Flash-ROM für Code und Konstanten

1.2 Erläutern Sie die Funktionen der 6 niederwertigsten Bit des Condition Code Registers des Mikrocontrollers MC9S12DP256 von Freescale.

Lösung zu Aufgabe 1.2:

Half carry: Übertrag bei Operationen mit BCD-Zahlen
Interrupt mask: Sperren der On-Chip-Unterbrechungssignale
Negative: Ergebnis einer Operation war negativ
Zero: Ergebnis einer Operation war Null
Overflow: Übertrag bei 2er-Komplementzahlen
Carry: Übertrag bei Operation mit Betragszahlen

Bitte geben Sie alle Aufgabenblätter wieder ab!

Wintersemester 2008/09		Blatt Nr.:	**2 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

1.3 Der Mikrocontroller vom Typ HCS12 unterstützt verschiedene Adressierungsarten. Im Folgenden sind Beispiele gegeben (var 1 und const sind mit ds.w bzw. dc.w definiert). Schreiben Sie die genaue Bezeichnung der Adressierungsart jeweils hinter das Beispiel. Einen der Befehle gibt es so nicht. Markieren Sie diesen mit einem Kreuz.

Lösung zu Aufgabe 1.3:

`TFR D,X`	_____	**Registeradressierung**
`LDD const`	_____	**direkte Adressierung**
`STD #const`	_____	**X**
`MOVB 1,X,0,X`	_____	**indiziert**
`LDD [var1,X]`	_____	**indirekt indiziert**
`LDAA 1,Y+`	_____	**indirekt post-inkrement**
`LDX const,Y`	_____	**indiziert**

1.4 Erläutern Sie stichwortartig die Bedeutung der drei Register DDRB, PPSB und PERB des im Labor verwendeten Mikrocontrollers MC9S12DP256 von Freescale.

Lösung zu Aufgabe 1.4:

DDRB: Datenrichtungsregister; bestimmt, ob Port als Eingang oder Ausgang geschaltet ist.

PPSB: Port Polarity Select; bestimmt, ob ein Pullup- oder Pull-down-Widerstand am Eingang angelegt werden soll

PERB: Schaltet den Pullup- bzw. Pulldown-Widerstand ein und aus

Bitte geben Sie alle Aufgabenblätter wieder ab!

Wintersemester 2008/09		Blatt Nr.:	**3 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

1.5 Erläutern Sie die Bedingungen, die vorliegen müssen, damit eine Interrupt-Service-Routine, z. B. für Port H, wiederholt ausgeführt werden kann.

Lösung zu Aufgabe 1.5:

I-Bit im CCR muss 0 sein
Adresse der ISR in der Interrupt-Vektor-Tabelle
Port H muss für Interrupts konfiguriert sein (PIEHx = 1)
Interrupt Flag (PIFH) muss durch Schreiben einer 1 auf das Register zurückgesetzt werden
(und ein entsprechendes Ereignis muss an Port H anliegen)

1.6 Der im Labor verwendete Mikrocontroller MC9S12DP256 von Freescale besitzt ein so genanntes Real Time Interrupt Modul (RTI). Nehmen Sie an, dass der Oszillatortakt mit Hilfe eines Quarzes auf 8 MHz eingestellt ist. Sie sollen den RTI so einstellen, dass er möglichst genau alle 100 ms einen Interrupt auslöst.

Bestimmen Sie den optimalen Wert für das Register RTICTL. Geben Sie an, wie viele Sekunden Abweichung eine Uhr nach 24 Stunden hätte, die direkt durch den 100 ms Takt getrieben würde.

Lösung zu Aufgabe 1.6:

f_{RTI} = 10 Hz = f_{oscclk} / (2^{9+x} (Y+1))

Wähle x=7 : daraus ergibt sich Y = 11

RTICTL = 0111 1011 (7B)

Damit wird f_{RTI} = 10,17253 Hz, d.h. eine Abweichung von 1,7253% (zu schnell).

Damit nach 24 Stunden = 24 * 3600 s: ca. 25 Minuten Abweichung.

RTICTL (binär oder hex): _____

Abweichung nach 24 h: _____

Bitte geben Sie alle Aufgabenblätter wieder ab!

Wintersemester 2008/09		Blatt Nr.:	**4 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

Aufgabe 2: Programmanalyse (30 Punkte)

Das folgende Assemblerlisting stellt zwei Funktionen dar, die von einem C-Programm aufgerufen werden können. Das Programm wurde für den Codewarrior-Compiler/Assembler geschrieben. Die C-Prototyp-Definitionen sehen so aus:

```
int f1(char *a1, const char *a2);
void f2(void *a1, const void *a2, int a3);
```

Listing zu Aufgabe 2:

```
1    f1:  TFR   D,  X
2         LDY   +2, SP
3         CLRA
4         CLRB
5
6    la:  ADDD  #1
7         MOVB  1, X+, 1, Y+
8         TST   -1, X
9         BNE   la
10
11        RTS
12
13   f2:  LDY   +2, SP
14        LDX   +4, SP
15
16        CPD  #0
17        BEQ   lc
18
19   lb:  MOVB  1, X+, 1, Y+
20        SUBD  #1
21        BNE   lb
22
23   lc:  RTS
```

2.1 Die erste Funktion wird folgendermaßen aufgerufen: **`e = f1(0x1234, 0x2345).`** Welcher Wert steht nach Ausführung von Zeile 1 im X-Register?

Lösung zu Aufgabe 2.1:

X = $2345

Wintersemester 2008/09		Blatt Nr.:	**5 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

2.2 Die Funktion wird folgendermaßen aufgerufen: **`e = f1(0x1000, 0x2000).`** Sie sehen in der untenstehenden Tabelle, welche Werte an diesen Speicherstellen vor dem Aufruf der Funktion stehen. Schreiben Sie die Werte nach dem Aufruf (also nach Ausführung der Zeile 11) in die vorgesehene Tabellenspalte.

Werte und Lösung zu Aufgabe 2.2:

Adresse	Wert vor Aufruf	Wert nach Aufruf
1000h	1	**20**
1001h	2	**30**
1002h	3	**0**
1003h	4	**4**
…		
2000h	20	**20**
2001h	30	**30**
2002h	0	**0**
2003h	-4	**-4**
2004h	20	**20**

2.3 Welcher Wert steht im D-Register vor Aufruf der Zeile 11, wenn die Funktion gemäß Aufgabe 2.2 aufgerufen wurde (einschließlich Tabellenwerte)?

Lösung zu Aufgabe 2.3:

`D = 3`

2.4 Die zweite Funktion wird folgendermaßen aufgerufen: **`f2(0x1000, 0x2000, 4).`** Welcher Wert steht nach Ausführung von Zeile 14 im X-Register?

Lösung zu Aufgabe 2.4:

`X = $1000`

2.5 Tragen Sie die Werte nach Aufruf der Funktion f2 (also nach Ausführung von Zeile 23) gemäß Aufgabe 2.4 in die untenstehende Tabelle ein.

Bitte geben Sie alle Aufgabenblätter wieder ab!

Wintersemester 2008/09		Blatt Nr.:	**6 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

Werte und Lösung zu Aufgabe 2.5:

Adresse	Wert vor Aufruf	Wert nach Aufruf
1000h	1	**1**
1001h	2	**2**
1002h	3	**3**
1003h	4	**4**
…		
2000h	20	**1**
2001h	30	**2**
2002h	0	**3**
2003h	-4	**4**
2004h	20	**20**

2.6 Arbeitet die Funktion f2 noch sinnvoll, wenn a3 den Wert null hat oder negativ ist? Bitte begründen.

Lösung zu Aufgabe 2.6:

null ist okay, negativ nicht mehr (Zeile 17)

Wintersemester 2008/09		Blatt Nr.:	**7 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

Aufgabe 3: Adressierungsarten und Stack (25 Punkte):

3.1
In einem HCS12-Assemblerprogramm sind folgende globalen Variablen definiert:

```
.const:    SECTION
           ORG   $D000
tabelle1:  DC.B  $11, $22, $33, $44, $55, $66, $77, $88
tabelle2:  DC.W  $D002, $D004
```

Geben Sie den Inhalt der CPU-Register D, X und Y nach jedem Assemblerbefehl an, wenn das folgende Programm ausgeführt wird. Es reicht aus, wenn Sie bei jedem Befehl diejenigen Registerwerte eintragen, die sich jeweils ändern.

Assemblerbefehle	D	X	Y
	$0000	$0000	$0000
LDX #2		$2	
LDD tabelle1, X	$3344		
LDX tabelle1		$1122	
LDY #tabelle1			$D000
LDAA 1, Y+	$1144		$D001
LDAA 2, +Y	$4444		$D003
LDAA 1, -Y	$3344		$D002
LDX 3, Y		$6677	
LDX -1, Y		$2233	
LEAY 2, +Y			$D004
LDX #tabelle2		$D008	
LDD [2, X]	$5566		

Bitte geben Sie alle Aufgabenblätter wieder ab!

Wintersemester 2008/09		Blatt Nr.:	**8 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

3.2
In einem C-Programm seien die folgenden globalen Variablen definiert:

```
int valA, valB, valC;
int m;
```

Diese Variablen werden im folgenden Ausschnitt des C-Programms verwendet, das Sie „von Hand“ in die entsprechenden HCS12-Assemblerbefehle übersetzen sollen. Die Definition der globalen Variablen muss nicht übersetzt werden. Assemblerdirektiven wie XDEF, XREF, INCLUDE, SECTION usw. dürfen weggelassen werden.

a) Geben Sie den Assembler-Programmcode an:

Lösung zu Aufgabe 3.2 a:

C-Programm

```
//***** Hauptprogramm *****
void main(void)
{   . . .
    m = add3(valA, valB, valC);
    . . .
}



//***** Unterprogramm *****
int add3(int a, int b, intc)
{
    return a + b + c;
}
```

HCS12-Assembler-Programm

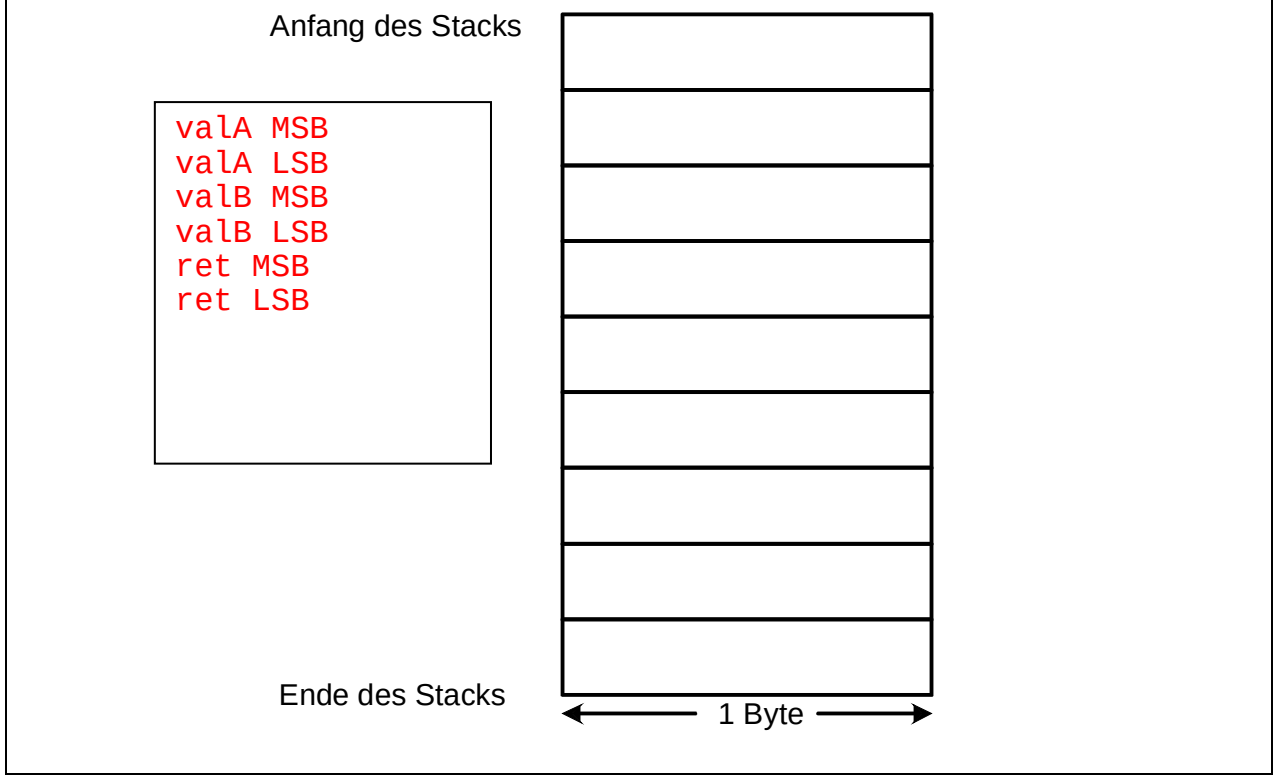
```
     LDD   valC
     LDX   valB
     LDY   valA
     PSHY
     PSHX
     JSR   add3
     LEAS  4,SP
     STD   m



add3:
     ADDD  4,SP
     ADDD  2,SP
     RTS
```

Bitte geben Sie alle Aufgabenblätter wieder ab!

Wintersemester 2008/09		Blatt Nr.:	**9 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

b) Tragen Sie in die folgende Tabelle den Zustand des Stacks direkt nach dem Aufruf des Unterprogramms „add3“ ein, und geben Sie an, auf welche Speicherzelle der Stack Pointer zu diesem Zeitpunkt zeigt.

Lösung zu Aufgabe 3.2b:



Bitte geben Sie alle Aufgabenblätter wieder ab!

Wintersemester 2008/09		Blatt Nr.:	**10 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

Aufgabe 4: HCS12-Peripheriebausteine (25 Punkte):

4.1 Schreiben Sie ein Assemblerprogramm „initSCI1“, das auf dem im Labor verwendeten Board mit 24 MHz Busfrequenz die serielle Schnittstelle SCI1 so konfiguriert, dass durch wiederholtes Ausgeben des hexadezimalen Zeichens „0xAA“ am zugehörigen Ausgang ein Rechtecksignal mit möglichst genau 440 Hz entstehen würde.

Initialisieren Sie die Schnittstelle so, dass sofort wenn der Transmitbuffer frei wird, ein Interrupt erzeugt wird.

Hinweis: Eine Schwingung wird gebildet durch einen 1-0- bzw. 0-1-Übergang, d.h. zwei Bit.

Lösung zu Frage 4.1:

```
2 Bit dauern 1/440 s, d.h. T_Bit = 1/880 s (Baudrate).
=> SCI1BD = 24 MHz / 16 * T_Bit = 1705
initSCI1:
     MOVB #1705, SCI1BD
     LDAA SCI1CR1
     ANDA #$02           ; keine Parität
     STAA SCI1CR1
     MOVB #%10001000, SCI1CR2     ; Transmit enable,
                                  ; Transmit Interrupt enable
     RTS
```

4.2 Schreiben Sie eine Interruptservice-Routine „isrSCI1“ für die oben initialisierte Schnittstelle, die bei jedem Auftreten des Transmit Interrupts das hexadezimale Zeichen „0xAA“ in den Transmitpuffer schreibt.

Lösung zu Frage 4.2:

Bitte geben Sie alle Aufgabenblätter wieder ab!

Wintersemester 2008/09		Blatt Nr.:	**11 von 11**
Studiengang:	**Kommunikationstechnik Softwaretechnik Technische Informatik**	Semester:	SWB4, TIB4, KTB4
Prüfungsfach:	**Computerarchitektur 3**	Fachnummer:	**4021**
Hilfsmittel:	**Vorlesungs- und Labormanuskript, Fachliteratur, Taschenrechner**	Dauer:	**90 min**

```
isrSCI1:
     LDAA SCI1SR1       ; Flag zurücksetzen und Interrupt prüfen
     BRCLR SCI1SR1, #$80, skip
     MOVB #$AA, SCI1DRL
skip:
     RTS
```

4.3 Schreiben Sie das Hauptprogramm, dass mit Hilfe der obigen Routinen die Schnittstelle initialisiert und die Erzeugung des Signals startet. Tragen Sie die Adresse der Interruptservice-Routine mit Hilfe von Pseudo-Assemblerbefehlen in die Interruptvektortabelle ein.

Lösung zu Frage 4.3:

```
.vect: SECTION
       ORG $FFD4
int21: DC.W isrSCI1
.init  SECTION
main:
     LDS #SEG_END_SSTACK
     CLI
     JSR  initSCI1
     LDAA SCI1SR1       ; Flag zurücksetzen und Interrupt prüfen
loop:
     BRA loop
```

Bitte geben Sie alle Aufgabenblätter wieder ab!