HOCHSCHULE ESSLINGEN FAKULTÄT INFORMATIONSTECHNIK

Sample Exam G		Blatt Nr.:	**1 von 8**
Studiengang:	**Kommunikationstechnik KTB** **Softwaretechnik SWB** **Technische Informatik TIB** **Ingenieurpädagogik IEP**	Semester:	IT4
Prüfungsfach:	**Computerarchitektur**	Fachnummer:	**4022, 1054003**
Hilfsmittel:	**Vorlesungs- und Labormanuskript,** **Fachliteratur, Taschenrechner** **Lecture manuscript, books, pocket calculator**	Dauer: Dozent: :	**90 min** Zimmermann

Insert your name here: Total: 90 Points

Given name (Vorname):	Last name (Nachname):	Student ID (Matrikelnummer):
Lösungsvorschlag (ohne Gewähr für Richtigkeit oder Vollständigkeit)		

Please use the free space on these sheets for your solution. Solutions may be in English or German. If space is not sufficient, please use the backside or additional sheets.

Bitte tragen Sie Ihre Lösungen in Deutsch oder English in die Aufgabenblätter ein. Sollte der vorgesehene Platz nicht ausreichen, verwenden Sie bitte die Rückseite bzw. Zusatzblätter. Kennzeichnen Sie eindeutig, auf welche Fragen sich die Zusatzblätter beziehen.

Viel Erfolg - Good luck!

Problem 1: Miscellaneous (Σ 10 points)

1.1 (4 points)

What is a “Programming Model”? What does the term “CPU family” imply?

Programming model:
- Register set
- Instruction set
- Adressing modes.

A „CPU family“ is a group of CPUs which use the same programming model.

1.2 (4 points)

What is the contents of variables `S1,S2,S3 and S4`, after the following C code has been executed: `char a = 0x7E, b = 0x81, c=0x7F, S1, S2, S3, S4;`

```
S1 = !a    =    0
S2 = b & c =    0x01
S3 = c + 1 =    0x80
S4 = a ^ a =    0
```

1.3 (2 points)

In which memory area (memory type) does the HCS12 C-compiler place the initialization values of global C variables?

In ROM

Sample Exam G		Blatt Nr.:	**2 von 8**
Studiengang:	**Kommunikationstechnik KTB** **Softwaretechnik SWB** **Technische Informatik TIB** **Ingenieurpädagogik IEP**	Semester:	IT4
Prüfungsfach:	**Computerarchitektur**	Fachnummer:	**4022, 1054003**
Hilfsmittel:	**Vorlesungs- und Labormanuskript,** **Fachliteratur, Taschenrechner** **Lecture manuscript, books, pocket calculator**	Dauer: Dozent: :	**90 min** Zimmermann

Problem 2: Addressing Modes (Σ 30 points)

2.1 (8 points)

Specify the addressing mode for all operands of the following HCS12 instructions:

Instruction	1st operand	2nd operand
`EXG X, D`	Explicit Register Addressing for Y	Explicit Register Addressing for D
`LDY #var2`	Implicit Register Addressing for Y	Immediate Addressing
`STD 4, -X`	- " - for D	Register-Indirect Addressng with Pre-Decrement
`DEX`	- " - for X	(Implicit Constante 1 in Decrement-Instruction)
`MOVW var1,4,Y`	Direct Addressing for var1	Register-indirecte Addressing with Index (Offset)

2.2

A HCS12 assembler program defines the following global variables and constants:

```
.const:         SECTION
                ORG     $D800

con1:           DC.W    $3210, $7654, $BA98, $FEDC
con2:           DC.B    $01, $23, $45, $67, $89, $AB, $CD, $EF
con3:           DC.L    $10203040, $50607080

.data:  SECTION
                ORG     $2800

v1:             DS.W    1
v2:             DS.B    2
```

For each row in the following table specify the contents of CPU registers D, X and Y and of variables v1 and v2, after the instructions in the left field of the row have been executed. Fields, which do not change values, may be left empty. Mark unknown values as “???” if necessary.

(22 points)

Sample Exam G		Blatt Nr.:	**3 von 8**
Studiengang:	**Kommunikationstechnik KTB** **Softwaretechnik SWB** **Technische Informatik TIB** **Ingenieurpädagogik IEP**	Semester:	IT4
Prüfungsfach:	**Computerarchitektur**	Fachnummer:	**4022, 1054003**
Hilfsmittel:	**Vorlesungs- und Labormanuskript,** **Fachliteratur, Taschenrechner** **Lecture manuscript, books, pocket calculator**	Dauer: Dozent: :	**90 min** Zimmermann

HCS12 instructions	D	X	Y	v1	v2[0]	v2[1]
`Initial values`	`$1122`	`$3344`	`$5566`	`$7788`	`$99`	`$AA`

`LDD con1`
`LDX con2`
`LDY con3`
`MOVW #$2456, v1`
`MOVW #v1, v2`
	`$3210`	`$0123`	`$1020`	`$2456`	`$28`	`$00`

`STX 2,-SP`
`STY 2,-SP`
`STAA 1,-SP`
`STAB 1,-SP`
`PULX`
`PULD`
`LDY 2,SP+`
	`$1020`	`$1032`	`$0123`	=	=	=

`LDX #con1`
`LDY #con2`
`LDAA 4,+X`
`LDAB 1, Y-`
	`$BA01`	`$D804`	`$D807`	=	=	=

`LDX #v2` `LDD [0,X]`	`$2456`	`$2802`	=	=	=	=

`LDD #$789A`
`TFR A, X`
`TFR B, Y`
	`$789A`	`$0078`	`$FF9A`	=	=	=

`LDD #$2800`
`LDX $D800`
`LDY con3+2`
	`$2800`	`$3210`	`$3040`	=	=	=

`MOVB #0, v2`
`LDD #$AABB`
`STD v1+2`
	`$AABB`	=	=	=	`$AA`	`$BB`

`LDX #$2802`
`LDD #$1122`
`STD 2, -X`
`LDY 2, X+`
	`$1122`	`$2802`	`$1122`	`$1122`	=	=

Sample Exam G		Blatt Nr.:	**4 von 8**
Studiengang:	**Kommunikationstechnik KTB** **Softwaretechnik SWB** **Technische Informatik TIB** **Ingenieurpädagogik IEP**	Semester:	IT4
Prüfungsfach:	**Computerarchitektur**	Fachnummer:	**4022, 1054003**
Hilfsmittel:	**Vorlesungs- und Labormanuskript,** **Fachliteratur, Taschenrechner** **Lecture manuscript, books, pocket calculator**	Dauer: Dozent: :	**90 min** Zimmermann

Problem 3: Code Analysis (Σ 30 points)

The following C program deals with strings:

```
int   subA(char *pString);
char* subB(char *pString, char c);

char *text = "LittleString";
char *pP;
int  n = 0;

void main(void)
{   EnableInterrupts;
    n  = subA(text);            // Line 1
    pP = subB(text, 'e');       // Line 2
}
```

Subroutines **subA()** and **subB()** are coded in HCS12 assembler:

```
subA:   TFR     D, X        ; &text[0] → X
        LDD     #0          ; D = 0 (counter variable)
L0:     TST     1,X+        ; if end of string reached
        BEQ     L1          ;    return D
        ADDD    #1          ; else
        BRA     L0          ;    D++ and goto L0
L1:     RTS

subB:   PSHX                ; save register
        LDX     4, SP       ; parameter &text[0] → X
L3:     LDAA    0,X     ; // Line 3
        CBA                 ; character c found in text[]?
        BEQ     L4          ; if yes we're done
        TST     1,+X        ; if no end of string text[] reached?
        BNE     L3          ; if no loop again (at L3)
        LDX     #0          ; if yes, prepare to return X=0
L4:     TFR     X, D        ; return X (pointer to character or 0)
        PULX                ; restore register
        RTS
```

3.1

What is the value of variable **n** after line 1 in **main()** has been executed?
What is the purpose of subroutine **subA**?

(6 points)

n = 12

Purpose of subA(): compute length of string (without trailing 0)

Sample Exam G		Blatt Nr.:	**5 von 8**
Studiengang:	**Kommunikationstechnik KTB** **Softwaretechnik SWB** **Technische Informatik TIB** **Ingenieurpädagogik IEP**	Semester:	IT4
Prüfungsfach:	**Computerarchitektur**	Fachnummer:	**4022, 1054003**
Hilfsmittel:	**Vorlesungs- und Labormanuskript,** **Fachliteratur, Taschenrechner** **Lecture manuscript, books, pocket calculator**	Dauer: Dozent: :	**90 min** Zimmermann

3.2

Into which HCS12 assembler instructions does the C compiler translate the subroutine call

```
pP = subB(text, 'e');        // Line 2
```

(6 points)

```
LDD   text1       <- NOTE: NO HASH SIGN, passes value of pointer
PSHD
LDAB  #'e'        <- Note: Here HASH SIGN, 'e' is a constant
JSR   subB
STD   pP
LEAS  2,SP
```

3.3

What is the contents of registers A and B, when the program completes executing line 3 in **`subB()`** for the first time?

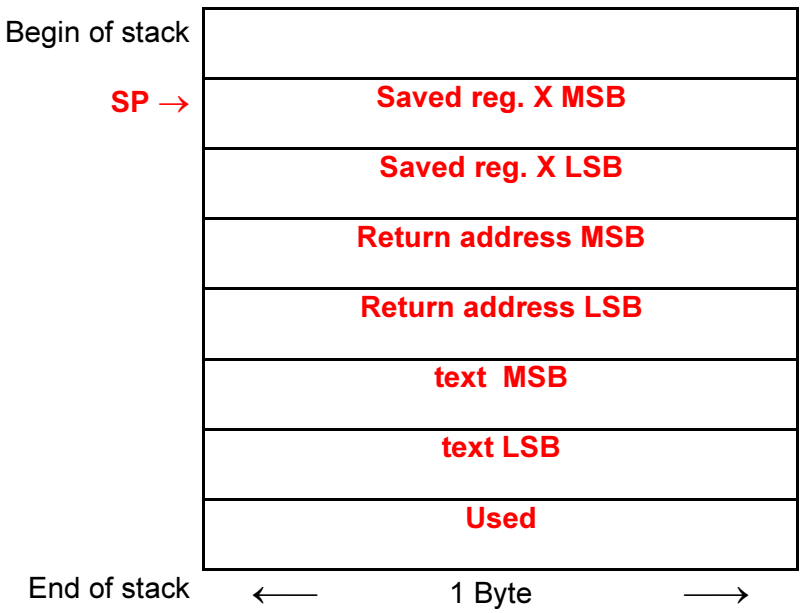
(4 points)

A = 'L' (=0x4C)

B = 'e' (=0x65)

Specify the state of the stack, when the CPU reaches line 3 in **`subB()`**:

(5 points)



Sample Exam G		Blatt Nr.:	**6 von 8**
Studiengang:	**Kommunikationstechnik KTB** **Softwaretechnik SWB** **Technische Informatik TIB** **Ingenieurpädagogik IEP**	Semester:	IT4
Prüfungsfach:	**Computerarchitektur**	Fachnummer:	**4022, 1054003**
Hilfsmittel:	**Vorlesungs- und Labormanuskript,** **Fachliteratur, Taschenrechner** **Lecture manuscript, books, pocket calculator**	Dauer: Dozent: :	**90 min** Zimmermann

3.4
Where does pointer `pP` point to, when `subB()` returns in line 2?

(3 points)

pP points to character 'e' in string text

3.5
What is the value of pointer `pP` after execution of line 2, when the subroutine was called as follows: `pP = subB(text, 'Z');  // Line 2 modified`

(2 points)

pP = 0

3.6
What is the purpose of subroutine `subB()`?

(2 points)

Searches the character c in the string and returns a pointer to the first occurrence of the specified character in the string.
Returns 0, if character not found in string

3.7
Why did subroutine `subB()` not save and restore register D?

(2 points)

Register D is used for the return value. So it does not make sense to save it, because it need not be restored anyhow.

Sample Exam G		Blatt Nr.:	**7 von 8**
Studiengang:	**Kommunikationstechnik KTB** **Softwaretechnik SWB** **Technische Informatik TIB** **Ingenieurpädagogik IEP**	Semester:	IT4
Prüfungsfach:	**Computerarchitektur**	Fachnummer:	**4022, 1054003**
Hilfsmittel:	**Vorlesungs- und Labormanuskript,** **Fachliteratur, Taschenrechner** **Lecture manuscript, books, pocket calculator**	Dauer: Dozent::	**90 min** Zimmermann

Problem 4: Dragon12 Electronic Music (Σ 20 points)

The C-program on page 8 generates sound with the Dragon12's beeper, which is connected to timer channel 5. The sound frequency can be modified via the potentiometer on the board, which is connected to the analog to digital converter ATD0's channel 7.

4.1

With which frequency will the interrupt service routine **`timerHandler()`** be called, when the timer is initialized via **`timerInit()`**? You may assume **`deltaTicks = 0`** here.

(4 points)

Timer clock period 0.667us = 667ns (TSCR2=4 → manuscript pg. 3.52)

Event period = interrupt period 1704 * 667ns = 1.137ms

→ Interrupt frequency 1/1.137ms = 880 Hz

4.2

Add the required C-code for functions **`adcInit()`** and **`adcGet()`**. The ADC shall be initialized in **`adcInit()`**, but no conversion started. **`adcGet()`** shall start the conversion, wait till the conversion completes and return the conversion result. Detailed requirements are:

- Single measurement on channel 7
- 10 bit resolution, result right-adjusted
- Fastest conversion time possible. (10 points)

4.3

When the program is running, what does the user have to do, to actually hear the beeper sound playing?

(3 points)

Press button H.0.

4.4

Assume the analog voltage to vary between 0 and 5 V. What influence does the potentiometer setting have?

(3 points)

The higher the analog voltage, the higher the sound frequency.
0V → 1.137ms x 2 → 440Hz
5V → 0.454ms x 2 → 1100Hz

Sample Exam G		Blatt Nr.:	**8 von 8**
Studiengang:	**Kommunikationstechnik KTB** **Softwaretechnik SWB** **Technische Informatik TIB** **Ingenieurpädagogik IEP**	Semester:	IT4
Prüfungsfach:	**Computerarchitektur**	Fachnummer:	**4022, 1054003**
Hilfsmittel:	**Vorlesungs- und Labormanuskript,** **Fachliteratur, Taschenrechner** **Lecture manuscript, books, pocket calculator**	Dauer: Dozent: :	**90 min** Zimmermann

```
// Dragon12 Electronic Music

unsigned int deltaTicks = 0;
unsigned int baseSound  = 1704;

void timerInit(void)
{       TSCR1 = 0x80;
        TSCR2 = 4;
        TIOS  = 0x20;
        TCTL1 = 0b00000100;
        TC5   = TCNT + baseSound;
        TIE   = 0x20;
}

void interrupt 13 timerHandler(void)
{       TC5   = TC5 + baseSound - deltaTicks;
        TFLG1 = 0x20;
}

void adcInit(void)
{       ATD0CTL2 = 0xC0;                                    (5 points)
        ATD0CTL3 = 0x08;
        ATD0CTL4 = 0x05;
        . . .


}

unsigned int adcGet(void)
{       ATD0CTL5 = 0x87;                                    (5 points)
        while ((ATD0STAT0 & 0x80) == 0) { };
        return ATD0DR0;
        . . .


}

void main(void)
{   EnableInterrupts;

    adcInit();
    timerInit();

    for (;;)
    {   if ((PTH & 0x01) == 0)
        {  TCTL1 = 0b00000100;
           deltaTicks = adcGet();
        } else
        {  TCTL1 = 0b00000000;
        }
    }
}
```