Informationssysteme
Semesterwoche 3

Teil 1: Zahlen und Logik

A) Aufgaben zu den ganzen Zahlen

1. Konvertieren Sie die folgenden Zahlen in die Binärform:

1984

/ 2 = 992	0	(LSB)
/ 2 = 496	0	
/ 2 = 248	0	
/ 2 = 124	0	
/ 2 = 62	0	
/ 2 = 31	0	
/ 2 = 15	1	
/ 2 = 7	1	
/ 2 = 3	1	
/ 2 = 1	1	
/ 2 = 0	1	(MSB)

= 11111000000

4000

/ 2 = 2000	0
/ 2 = 1000	0
/ 2 = 500	0
/ 2 = 250	0
/ 2 = 125	0
/ 2 = 62	1
/ 2 = 31	0
/ 2 = 15	1
/ 2 = 7	1
/ 2 = 3	1
/ 2 = 1	1
/ 2 = 0	1

= 111110100000

8192 = 1000000000000

2. Wie lautet 1001101001 (binär) dezimal, oktal, hexadezimal?

Binär → Dezimal
1001101001 = 1*2^0 + 0 + 0 + 1*2^3 + 0 +1*2^5 +1*2^6 + 0 + 0 +1*2^9 = 1 + 8 + 32 + 64 + 512 = 617

Binär → Oktal
1 001 101 001 = 1 5 1 1

Binär → Hex
0010 0110 1001 = 2 6 9

3. Welche der folgenden Darstellungen sind gültige hexadezimale Zahlen?
BED, CAB, DEAD, DECADE, ACCEDED, BAG, DAD.
→ Gültige Zeichen des hexadezimalen Zahlenbereichs sind 0-9 und A-F.
→ Alle ausser BAG sind demzufolge hexadezimale Zahlen.

4. Stellen Sie die Dezimalzahl 100 unter Benutzung der Basiszahlen von 2 und 8 dar.
→ 100(dez) = 10^2

100 / 2 = 50 0
50 / 2 = 25 0
25 / 2 = 12 1
12 / 2 = 6 0
6 / 2 = 3 0
3 / 2 = 1 1
1 / 2 = 0 1

100 (dez) = 1100100 (bin)
1100100 = 2^2 + 2^5 + 2^6
001 100 100 (bin) = 8^4 + 8^4 + 8^1

5. * Wie viele verschiedene positive Ganzzahlen können in k Ziffern mit Hilfe von r Basiszahlen ausgedrückt werden?
→ Anz.Pos.Ganzzahlen = | r ^ k |

B) Aufgaben zu den Gleitkommazahlen

1) Konvertieren Sie die folgenden Zahlen in das einfachgenaue IEEE-Format. Geben Sie die Ergebnisse als acht hexadezimale Ziffern aus.

---

a) 9

9 / 2 = 4 1 (Least-Significant Bit)
4 / 2 = 2 0
2 / 2 = 1 0
1 / 2 = 0 1 (Most-Significant Bit)

9(dez) = 1001.00000000000000000000 (bin)

Normalisieren
1.00100000000000000000000

Exponent ist +3
127+3 = 130 (dez) = 10000010

Vorzeichen
0 (Positiv)

= 0 10000010 00100000000000000000000 (bin)
= 41100000 (hex)

---

b) 5/32 = 0.15625

0.15625 * 2 = 0.3125 0 (MSB)
0.3125 * 2 = 0.625 0
0.625 * 2 = 1.25 1 (LSB)

0.15625 (dez) = 1.25 * 2^(-3)

Exponent bestimmen:
Die 2er Potenz ist -3
(-3) + 127 = 124 = 1111100 (Excess-127 Format)

Mantisse approximieren (was in diesem Fall ziemlich präzise möglich ist):
0.25 = 2^ -2 = 0.01(bin)

Vorzeichen-Bit = 0 (Positiv)

= 0 1111100 01000000000000000000 (bin)
= 3E200000 (hex)

---

c) -5/32 = -0.15625

Vorzeichen-Bit = 1 (Negativ)

= 1 1111100 01000000000000000000 (bin)
= 7E200000 (hex)

---

d) 6.125

1. Umwandlung der Dezimalzahl in eine duale Festkommazahl ohne Vorzeichen

6 / 2 = 3	0	(LSB)
3 / 2 = 1	1	
1 / 2 = 0	1	(MSB)

= 110 (bin)

0.125 * 2 = 0.25	0	(MSB)
0.25 * 2 = 0.5	0	
0.5 * 2 = 1	1	(LSB)

= 0.001 (bin)

2. Normalisieren
110.001 00000000000000000000 * 2^0 = 1.10001000000000000000000000 * 2^2

3. Exponent bestimmen
= 2
127+2 = 129 = 10000001(bin)

4. Vorzeichen
Positiv = 0

5. Komponenten zusammensetzen
= 0 10000001 10001000000000000000000 (bin)
= 103100000 (hex)

2) Konvertieren Sie die folgenden einfachgenauen IEEE-Gleitkommazahlen von Hexadezimal in Dezimal:

a) 42E48000H
42E48000(hex) = 0 10000101 11001001000000000000000(bin)

1. Vorzeichen
0 = Positiv

2. Exponent berechnen
10000101 (bin) = 133(dez)
133 – 127 = 6

3. Umwandlung in eine Dezimalzahl (näherungsweise)
0.11001001000000000000000= 1*2^-1 + 1*2^-2 + 1*2^-5 + 1*2^-8 = 201/256 = 0.7851…
4. Einzelne Teile zusammensetzen
(+1) * (1 + 201/256) * 2^6 = 114.25

b) 3F880000H
3F880000(hex) = 0 11111110 00100000000000000000

Vorzeichen
0 = Positiv

Exponent
11111110 = 254
254-127 = 127

Umwandlung in Dezimalzahl
0.00100000000000000000= 2^-3 = 0.125(dez)

Zusammensetzen
(+1) * (1+0.125) * 2^(127) = **???**

```
1.0625
```

c) 00800000H

d) C7F00000H

= 1 10001111 11100000000000000000000

1. Vorzeichen
= -1 (Negativ)

2. Exponent
10001111 = 143
143-127 = 16

3. Umwandlung in Dezimalzahl
0.11100000000000000000000 * 2^16 = 2^-1 + 2^-2 + 2^-3 = 7/8 = 0.875

4. Zusammensetzen

-1 * (1+0.875) *2^16 = -122880

3) In einer bestimmten Situation kann eine Operation auf zwei Gleitkommazahlen eine drastische Reduzierung der Anzahl der signifikanten Bits im Ergebnis verursachen. Warum ist das der Fall?

C) Aufgaben zur Bool'schen Algebra

Aufgabe 1 Hamburger oder Hot-Dog

Ein Logiker fährt in ein Drive-in-Restaurant und sagt: "Ich möchte einen Hamburger oder einen Hot-Dog und Pommes Frites." Der Koch hat die Schule zu häufig geschwänzt und weiss nicht (oder kümmert sich nicht darum), ob "UND" oder "ODER" Vorrang hat. Für ihn ist das eine so gut wie das andere. Welche der folgenden Fälle sind gültige Interpretationen der Bestellung? (Im Englischen hat das Wort "OR" ["ODER"] die Bedeutung von "EXCLUSIVE OR" [ausschliessliches oder].)

a. Nur ein Hamburger.
b. Nur ein Hot-Dog.
c. Nur Pommes Frites.
d. Ein Hot-Dog und Pommes Frites.
e. Ein Hamburger und Pommes Frites.
f. Ein Hot-Dog und ein Hamburger.
g. Alle drei.
h. Nichts - der Logiker bleibt zwar klug, fährt aber hungrig von dannen.

Aufgabe 2 Missionar

Ein Missionar hat sich in Südkalifornien verirrt und stoppt an einer Kreuzung. Er weiss, dass zwei Motorradbanden die Gegend unsicher machen. Eine davon sagt immer die Wahrheit, und die andere lügt immer. Er möchte wissen, welche Strasse nach Disneyland führt. Welche Frage sollte er stellen?
--> Wahrheitstabelle mit 4 Kombinationen

Aufgabe 3 * Bool'sche Funktionen

Es gibt vier boolesche Funktionen einer einzigen Variablen und 16 Funktionen von zwei Variablen. Wie viele Funktionen von drei Variablen gibt es? Wie viele gibt es von n Variablen?

Fall n=0
2^2^0 = 2^1 = 2

Fall n=1
2^2^1 = 2^2 = 4

Fall n=2
2^2^2 = 2^4 = 16

Fall n>2
2^2^n = …

http://de.wikipedia.org/wiki/Boolesche_Funktion

Aufgabe 4 Wahrheitstabelle

Benutzen Sie eine Wahrheitstabelle, um folgendes zu zeigen:
P = (P AND Q) OR (P AND NOT Q).

P	Q	AND	AND NOT	(P AND Q) OR (P AND NOT Q)
1	1	1	0	0

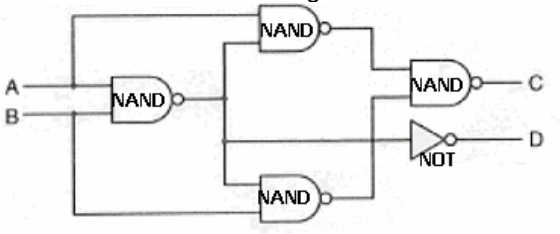
1	0	0	1	0
0	1	0	0	1
0	0	0	0	1

P = 1 OR 0
P = 0 OR 1
…
Sämtliche Fälle treffen zu, da AND und NAND gegensätzliche Funktionen sind.
http://de.wikipedia.org/wiki/Wahrheitstabelle

Aufgabe 5 AND aus NAND
Zeigen Sie, wie die AND-Funktion aus zwei NAND - Gates gebildet werden kann.
P = (P NAND (P NAND Q))
http://de.wikipedia.org/wiki/NAND-Gatter

Aufgabe 6 De Morgan
Benutzen Sie die De Morganschen Gleichungen, um das Komplement (=Negierung) von AB zu ermitteln.
$\overline{AB} = \overline{A}+\overline{B}$
http://de.wikipedia.org/wiki/De_Morgansche_Gesetze

Aufgabe 7 * Schaltungsanalyse
Was macht diese Schaltung?

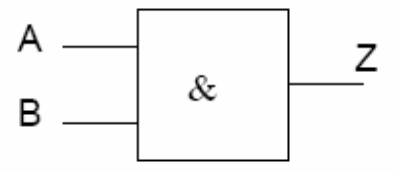


A=1, B=1 --> C=1, D=1
A=0, B=0 --> C=0, D=0
A=1, B=0 --> C=1, D=0

Diese Schaltung ist ein Halbaddierer. C ist die Summe, D ist der Übertrag (Carry).

Aufgabe 8 Logik-Bausteine
Ergänzen Sie die fehlenden Namen und die zugehörigen Wahrheitstabellen.
Geben Sie ein Schalteräquivalent an.

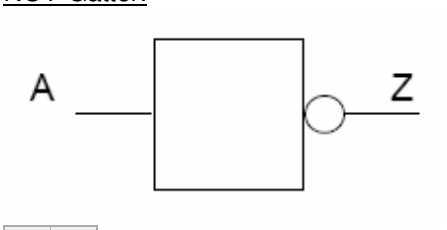
AND-Gatter:



A	B	Z
0	0	0
0	1	0

1	0	0
1	1	1

NOT-Gatter:



A	Z
0	1
1	0

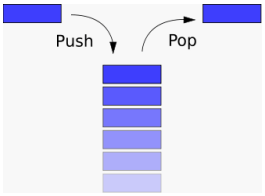
http://de.wikipedia.org/wiki/Logikgatter

Teil 2: Rechner-Architekturen

1.) Beantworten Sie bitte folgende Repetitionsfragen

1. Beschreiben Sie in eigenen Worten und mit einer Skizze die Schichtung einer Multilevel Maschine.
2. Welche Grundlegenden Operationsarten gibt es bei den Maschinenbefehlen?
3. Skizzieren Sie einen Drei-Adress Befehl.
4. Beschreiben Sie mit einer Skizze und ein paar Stichworten, was folgende zwei Adressierungsarten bedeuten: absolute Adressierung und unmittelbare Adressierung
5. Wozu dient der Instruction Pointer? Wie nennt man ihn auch noch?
Instruction Pointer = Program Counter (PC) = Befehlszähler. Er beinhaltet jeweils die Adresse des derzeitigen/nächsten Befehls.
http://de.wikipedia.org/wiki/Befehlsz%C3%A4hler

6. Wozu kann ein Stack verwendet werden?
→ Ein Stack ist ein Stapelspeicher, welcher nach dem Prinzip "first in – last out" abgearbeitet wird.



http://de.wikipedia.org/wiki/Stapelspeicher

7. Wozu wird ein Cache verwendet?
→ Cache ist ein schneller Zwischenspeicher für Informationen, welche ständig/wiederholt abgerufen werden müssen. Die Grösse eines Caches ist meistens stark beschränkt. (Je schneller, je weniger ist vorhanden).
http://de.wikipedia.org/wiki/Cache

8. Wozu dient Mikroprogrammierung?
→ Mikroprogramme können als Firmware einer CPU verstanden werden. Sie enthalten Listen mit Steuersignalen die das Verhalten der einzelnen Prozessorelemente regeln.
http://de.wikipedia.org/wiki/Mikroprogrammierung

2.) Mikrocontroller

Lesen Sie das Datenblatt des Motorola M68HC05 Mikrocontrollers und beantworten nachstehende Fragen. Das Datenblatt finden sie auch im ILIAS. Die Fragen beziehen sich zum einen Teil auf Begriffe, die Sie im Unterricht gehört haben und die sie hier an einem konkreten Beispiel wieder finden. Zum anderen sollen Sie auch ganz Neues lernen und einen ersten Eindruck davon bekommen, was ein Mikrocontroller ist und kann. Und drittens üben Sie dabei den Umgang mit Original-Dokumentation. In grossen Dokumenten die relevante Information zu suchen und auch zu finden :-) ist eine für den Ingenieur und die Ingenieurin wichtige Fähigkeit.
Seien Sie nicht beunruhigt, wenn Sie Einiges noch nicht verstehen. Sie haben im Unterricht in den Übungsbesprechungen Gelegenheit, Fragen zu stellen.
Falls Sie sich das jetzt schon zutrauen, verwenden Sie für diese Aufgabe das vollständige Datenblatt mit einem Umfang von 332 Seiten. Der Filename ist M68HC05TB.pdf. Falls Sie ein so grosses Dokument im Moment noch abschreckt, verwenden Sie die Zusammenfassung, die im File M68HC05TB-Excerpt.pdf steht und nur 19 Seiten umfasst. Alle "obligatorischen" Fragen lassen sich damit beantworten. Für die fakultativen ("*") Aufgaben ist teilweise die Vollversion des Datenblatts nötig Notieren Sie bei Ihren Antworten die entsprechende Seitennummer des Datenblatts, das hilft bei der Besprechung Ihrer Arbeit. Die Seitenummern im Auszug aus dem Datenblatt stimmen mit der Vollversion überein.

Aufgabe 1 Mikrocontroller

Was ist ein Mikrocontroller?
→ Als Mikrocontroller (auch µController, µC, MCU) werden Halbleiterchips bezeichnet, die mit dem Prozessor wesentliche Peripheriefunktionen auf einem Chip vereinen. In vielen Fällen befindet sich der Arbeits- und Programmspeicher ebenfalls teilweise oder komplett auf dem gleichen Chip. Ein Mikrocontroller ist praktisch ein Ein-Chip-Computersystem.
Auf modernen Mikrocontrollern finden sich häufig auch komplexe Peripheriefunktionen wie z.B. USB, RS232, Ethernet-Schnittstellen, PWM-Ausgänge, LCD-Controller und -treiber und Analog-Digital-Wandler. Einige Mikrocontroller verfügen auch über programmierbare digitale und/oder analoge Funktionsblöcke.
http://de.wikipedia.org/wiki/Mikrocontroller

Aufgabe 2 Architektur

Um welche Architektur handelt es sich bei diesem Mikrocontroller?
→ 8-Bit-Prozessor der Von-Neumann-Architektur. Daten- und Programmspeicher im selben Speicher. (S.61)

Aufgabe 3 Mnemonics

(a) Wie lautet die binäre Darstellung der Mnemonischen Codes RTS?
→ RTS hat den hexadezimalen Wert 81. Dies entspricht dem Binärwert 1000 0001 (S.9, 132, 77)

(b) Welche Funktion führt dieses "Mnemonic" aus?
→ Eine Subroutine, welche zum Hauptprogramm zurückkehren möchte. (RTS = Return from subroutine).

(c) Mit welchen Adressierungsarten kann ADD verwendet werden? Geben Sie dazu zwei Beispiele.
→ ADD („Add without carry“) kann laut Tabelle mit folgenden Adressierungsarten verwendet werden:

IMM*	In immediate addressing mode, the operand for the instruction is the byte immediately after the opcode.
DIR*	In direct addressing mode, the low order eight bits of the address of the operand are located in the next byte of memory after the opcode and the high order byte of the address is assumed to be $00. This mode is more efficient than the extended addressing mode because the high order address byte is not explicitly included in the program.
EXT	In extended addressing mode, the 16-bit address of the operand is located in the next two memory bytes after the instruction opcode.
IX* IX1 IX2	In indexed addressing mode, the current value of the index register is added to a 0-, 1-, or 2-byte offset in the next 0, 1, or 2 memory locations after the opcode to form a pointer to the address of the operand in memory.

(*prüfungsrelevant; S.132, 223)

Aufgabe 4 Register

Wie viele Register weist der M68HC05 auf? Benennen Sie alle Register mit Ihrer Abkürzung.
Der M68HC05 hat fünf CPU Register:
→ Akkumulator (A) als generelles 8-bit Register
→ Indexregister (X) als 8-bit Pointer-Register
→ Stack Pointer (SP)
→ Program Counter (PC)
→ Statusregister (Condition Code Register; CCR)

Aufgabe 5 Index Register

Bei Aufgabe 5 sollten Sie auch auf ein Register mit dem Namen "Index Register" gestossen sein.
Wozu könnte dieses Register nützlich sein?
→ Das Index Register wird im Indexed Addressing Mode (IX) als Zusatz zum 8-bit Akkumulator als generelles Register gebraucht.

→ Anwendung: Indizieren eines Arrays, Schleifenzähler, usw…
(S.71)

Aufgabe 6 Instruction Set

Wie lautet der binäre Code
(a) des Befehls RSP
→ RSP („Reset Stack Pointer“) hat den hexadezimalen Wert 9C, was dem binären Wert 1001 1101.
(S.132)

(b) des Befehls LDA? Was fällt Ihnen dabei auf?
→ LDA entspricht hexadezimal A6 (1010 0110)… gültige Lösung: A6-F6 (je nach Adressierungsart).

Aufgabe 7 Programmanalyse

(a) Im folgenden Programmabschnitt wird ein Wert in den Akkumulator A eingelesen. Aus welcher Adresse wird der Wert gelesen? Wie heisst die Adressierungsart?

```
0003          SAM   EQU   $03          SAM equal an 8-bit value
1400          LARRY EQU   $1400        LARRY equal a 16-bit value
0100          ORG   $100               Set program starting point
0100 ae 02    TOP   LDX   #$02         Initialize index register
0102 a6 05    LDA   #$05               Read value into A
```

→ 0103

(b) Im folgenden Programmabschnitt wird ein Wert in den Akkumulator A eingelesen. Aus welcher Adresse wird der Wert gelesen? Wie heisst die Adressierungsart?
0003 SAM EQU $03 SAM equal an 8-bit value
1400 LARRY EQU $1400 LARRY equal a 16-bit value
0100 ORG $100 Set program starting point
0100 ae 02 TOP LDX #$02 Initialize index register
0102 b6 05 LDA $05 Read value into A

→ AUS DER ADRESSE 05 (direkte Adressierung)

Aufgabe 8 * Programmanalyse

Folgendes Programm wird von START bis END abgearbeitet. Welche Zahl steht am Schluss in der Spicherzelle $00FF?

```
0100 9C START RSP Reset SP to $00FF
0101 cd 02 00 JSR SUB Call SUB
0104 cd 02 00 JSR SUB Call SUB again
0107 9d END NOP Done
……
……
0200 81 SUB RTS Just Return
```

Aufgabe 9 * Subroutinen Verschachtelung

Wie gross ist die maximale Schachtelungstiefe für Subroutinen? Geben Sie allfällige Randbedingungen genau an.
→ 32 Verschachtelungen. Dies ist beschränkt durch die Stackgrösse von 64byte. (Jeder Befehl nutzt 2byte --> 32x2byte =64byte).