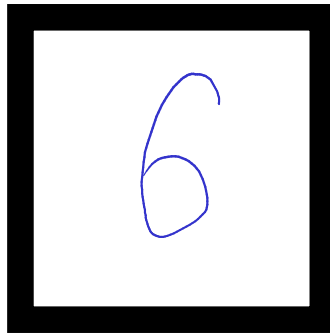


- Übungsblatt 3 -

Tutoriumsnummer




Name, Vorname: Slavov, Velislav

Matrikelnummer: 2385786

Studiengang: Informatik BSc



Name des Tutors: Jonas Heinle


A1) 4) "Velislav Slavov" verschlüsselt ist wieder "Velislav Slavov" 


A2) 1. Ausgabe: 17541.2324218750

Ich gehe davon aus, dass der Author entweder Nullen oder Dreier nach dem .233 erwartet hat





2. Bei mir wird ein Segmentation Fault erläutert   
und auch beim compilieren bekomme ich ein Warning.  
Scheinbar kann man die Adresse einer lokalen Variable nicht zurückgeben. 

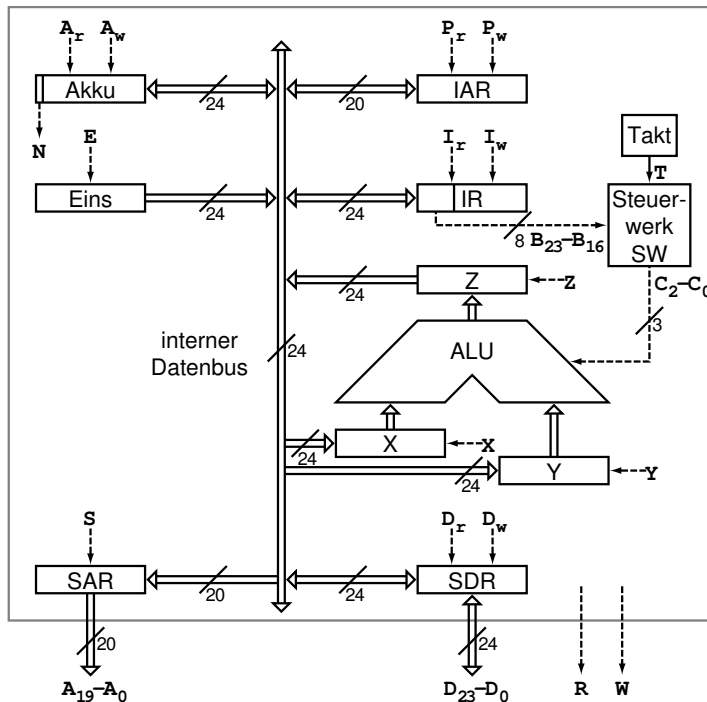
3. Da  $i$  ein unsigned int ist, wird  $i-1$  bei der letzten Iteration der for Schleife auf  $-1$  gesetzt, was zu einem Fehler führt.  
Kein array Index  $-1$  möglich! 

A 4) 1. So muss die Speicheradressierung nicht explizit angegeben werden. Also wir nehmen einfach die Bits nach dem Opcode und das ist der Wert. 

3.

- IAR = Instruktionsadresse register: Speichert die Adresse der nächsten Instruktion
- IR = Instruktionsregister: Speichert die Bits der nächsten Instruktion (inklusive Parametern)
- SAR = Speicheradressregister: Speichert eine Speicheradresse auf die später zugegriffen wird
- SDR = Speicherdatenregister: Speichert die Daten die vom Speicher gelesen wurden (bzw in Speicher geschrieben werden) 

4. Z.B. braucht die Addition die IAR, SAR & SDR nicht, da alle Infos schon im IR stehen. 

**Architektur der MIMA****Register**

Akku: Akkumulator  
 X: 1. ALU Operand  
 Y: 2. ALU Operand  
 Z: ALU Ergebnis  
 Eins: Konstante 1  
 IAR: Instruktionsadreßregister  
 IR: Instruktionsregister  
 SAR: Speicheradreßregister  
 SDR: Speicherdatenregister

**Steuersignale vom SW**

– für den internen Datenbus

$A_r$ : Akku liest  
 $A_w$ : Akku schreibt  
 $x$ : X-Register liest  
 $y$ : Y-Register liest  
 $z$ : Z-Register schreibt  
 $E$ : Eins-Register schreibt  
 $P_r$ : IAR liest  
 $P_w$ : IAR schreibt  
 $I_r$ : IR liest  
 $I_w$ : IR schreibt  
 $D_r$ : SDR liest  
 $D_w$ : SDR schreibt  
 $s$ : SAR liest

– für die ALU

$c_2-c_0$ : Operation auswählen

– für den Speicher

$R$ : Leseanforderung  
 $W$ : Schreibsanforderung

**Meldesignale zum SW**

$T$ : Takteingang  
 $N$ : Vorzeichen des Akku  
 $B_{23}-B_{16}$ : OpCode-Feld im IR

$c_2 c_1 c_0$	ALU Operation
0 0 0	tue nichts ( d.h. $Z \rightarrow Z$ )
0 0 1	$X + Y \rightarrow Z$
0 1 0	rotiere X nach rechts $\rightarrow Z$
0 1 1	$X \text{ AND } Y \rightarrow Z$
1 0 0	$X \text{ OR } Y \rightarrow Z$
1 0 1	$X \text{ XOR } Y \rightarrow Z$
1 1 0	Eins-Komplement von X $\rightarrow Z$
1 1 1	falls $X = Y$ , -1 $\rightarrow Z$ , sonst 0 $\rightarrow Z$

OpCode	Mnemonik	Beschreibung
0	LDC c	$c \rightarrow \text{Akku}$
1	LDV a	$\langle a \rangle \rightarrow \text{Akku}$
2	STV a	$\text{Akku} \rightarrow \langle a \rangle$
3	ADD a	$\text{Akku} + \langle a \rangle \rightarrow \text{Akku}$
4	AND a	$\text{Akku AND } \langle a \rangle \rightarrow \text{Akku}$
5	OR a	$\text{Akku OR } \langle a \rangle \rightarrow \text{Akku}$
6	XOR a	$\text{Akku XOR } \langle a \rangle \rightarrow \text{Akku}$
7	EQL a	falls $\text{Akku} = \langle a \rangle$ : -1 $\rightarrow \text{Akku}$ sonst : 0 $\rightarrow \text{Akku}$
8	JMP a	$a \rightarrow \text{IAR}$
9	JMN a	falls $\text{Akku} < 0$ : $a \rightarrow \text{IAR}$
F0	HALT	stoppt die MIMA
F1	NOT	bilde Eins-Komplement von Akku $\rightarrow \text{Akku}$
F2	RAR	rotiere Akku eins nach rechts $\rightarrow \text{Akku}$

**Befehlsformate**