Name:	Punkte:/31P
Datum:	Note:
1) Theorie Rechnerarchitektur  1.a) 0-Adressarchitektur  Eine gegebene Stackmaschine unterstützt die Befehle push, add die Berechnung (5+2)*3+1 auszuführen?	/3P d und mult. Welche Befehle werden ausgeführt, um
push 3 push 5 push 2 add mult add	
1.b) CISC vs. RISC	/4P
Ordne die folgenden (typischen) Eigenschaften der jeweiligen Ar Befehlsausführung meist in einem Takt Steuerwerk wird oft mittels Mikrocode realisiert Es gibt meist nur ein oder zwei allgemeine Register Für Pipelining optimiert Gilt als "modernere" Architektur Datentransfer fast nur über Load-Store Befehle Atmel AVR ist ein typischer Die Programme sind im allgemeinen kleiner 1.c) Steuerwerk und Datenpfad Ordne die folgenden Eigenschaften dem Steuerwerk oder dem Dakkumulator Mikrocode Carry-Flag Datenbus Fetch Instruction Harvard-Architektur Programmspeicher Rechenwerk	CISC RISC

2) Praxis Assemblerprogrammierung

```
.include "m16def.inc"
      clr XH
      ldi XL, 0x60
strupper:
      ld R16, X+
      cpi R16, 0x00
      breq end
      cpi R16, 0x61
      brlo strupper
      cpi R16, 0x7B
      brsh strupper
      subi R16, 0x20
      dec\ XL
      st X+, R16
      rjmp strupper
end:
      rjmp end
```

## Speicherauszug:

0x5F 0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F 0x70 0x71 0x72 0x37 0x48 0x69 0x20 0x34 0x61 0x68 0x65 0x6c 0x69 0x00 0x20 0x44 0x5A 0x42 0x6C 0x63 0x52 0x51 0x32

2.a) Wie groß ist das Programm in Bytes?

/1P

28 Bytes

2.b) Wie viele Takte benötigt das Programm bis zum Erreichen des Labels end?

\_\_\_/3P

Hinweis: Notiere die Zwischenschritte der Berechnung!

## 112 Takte

2.c) Was steht im Speicher nach der Ausführung?

\_\_\_/3P

0x5F	0x60	0x61	0x62	0x63	0x64	0x65	0x66	0x6/	0x68	0x69	0x6A	0x6B	0x6C	0x6D	0x6E	0x6F	0x/0	0x/1	0x/2
	0x48	0×49	0×20	0x34	0×41	0x48	0x45	0x4c	0×49										

2.d) Welche Werte haben die Register nach der Ausführung?

\_\_\_/1P

R16	X				
0x00	0x006A				

2.e) Fragen zum Programm

\_\_/5P

Das Programm zählt die Anzahl der 0x00 Bytes

Das Programm nutzt den Stack

Es wird nur das Register R16 verwendet

Es werden Daten kopiert

Die Laufzeit ist abhängig von den Daten im Speicher

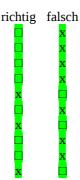
Am Ende der Routine ist der X Zeiger wieder am ursprünglichen Wert

Es werden nur bestimmte Bytes im Speicher verändert

Im Speicher werden 16 Bit Daten im Speicher verarbeitet

Es kann zu einem Überlauf in der ALU kommen

Es können nicht mehr als 255 Bytes an Daten bearbeitet werden

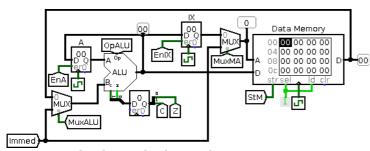


## 2.f) Assembler Aufgabe

/3P

Im Register R16:R17 (high:low) steht ein 16 Bit Wert. Dieser soll mit 5 multipliziert werden. Bringe die Befehle in die richtige Reihenfolge.

- <mark>1</mark> mov R19, R17
- 3 lsl R19
- <mark>8</mark> adc R16, R18
- <mark>4</mark> rol R18
- <mark>6</mark> rol R18
- <mark>2</mark> mo∨ R18, R16
- <mark>5</mark> lsl R19
- <mark>7</mark> add R17, R19
- 3) Praxis Rechnerarchitektur



ALU	Operation	Beschreibung
000	Result=A	Legt Operand A auf den Ausgang
001	Result=B	Legt Operand B auf den Ausgang
010	Result=A+B	Addiert A und B
011	Result=A-B	Subtrahiert B von A
100	Result=A AND B	Bitweise UND Verknüpfung
101	Result=A OR B	Bitweise OR Verknüpfung
110	Result=A EOR B	Bitweise Exclusive-OR
111	Result=A>>1	Logisches Rechtsschieben von A

3.a) Ordne die Beschreibung richtig zu

/4P

- 1. nop Führt keine Operation aus (No Operation)
- 2. ld IX, (IX) Lädt Register IX mit dem Wert an der Adresse IX
- 3. or A, Imm. Ver-ODER-t Register A mit Konstante
- 4. st (IX), Imm. Speichert die Konstante Imm. an der Adresse IX
- 5. st (IX), A Speichert den Wert von Register A an der Adresse IX
- 6. ld IX,A Lädt Register IX mit dem Wert aus Register A
- 7. and A, (IX) Ver-UND-et Register A mit dem Wert an Adresse IX
- 8. ld A, (Imm.) Lädt Register A mit dem Wert an der Adresse Imm.

Nummer	EnA	EnIX	StM	MuxALU	MuxMA	OpALU
<mark>6</mark>	0	1	0	X	X	000
3	1	0	0	1	X	101
8	1	0	0	0	1	001
5	0	0	1	X	0	000
1	0	0	0	X	X	X
2	0	1	0	0	0	001
7	1	0	0	0	0	100
4	0	0	1	1	0	001