

Name:

Punkte: ___/31P

Datum:

Note:

1) Theorie Rechnerarchitektur

1.a) Steuerwerk

___/2P

Bringe die 5 Teilschritte des Von-Neumann Zyklus in die richtige Reihenfolge (durch Nummerierung)

- ___ Execute
- ___ Write Back
- ___ Decode
- ___ Fetch Instruction
- ___ Fetch Operands

1.b) CISC vs. RISC

___/3P

Ordne die folgenden (typischen) Eigenschaften der jeweiligen Architektur zu

	CISC	RISC
Befehlsausführung meist in einem Takt	<input type="checkbox"/>	<input type="checkbox"/>
Steuerwerk kann mittels Mikrocode realisiert werden	<input type="checkbox"/>	<input type="checkbox"/>
Sehr viele allgemeine Register	<input type="checkbox"/>	<input type="checkbox"/>
Für Pipelining optimiert	<input type="checkbox"/>	<input type="checkbox"/>
Programme sind „kürzer“ (benötigen weniger Programmspeicher)	<input type="checkbox"/>	<input type="checkbox"/>
Datentransfer fast nur über Load-Store Befehle	<input type="checkbox"/>	<input type="checkbox"/>

1.c) 0-Adressarchitektur

___/2P

Welcher Wert liegt am Stack nach folgender Programmausführung? (Tip: Skizziere die Vorgänge am Stack)

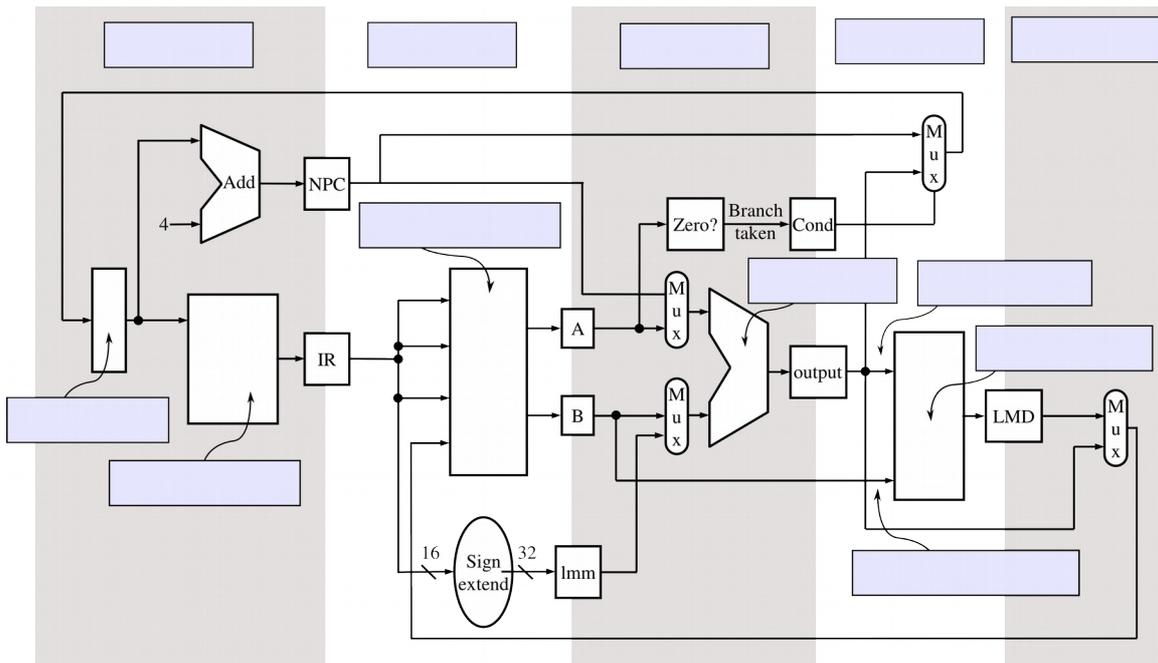
```

push 0x02
push 0x04
push 0x0C
add
mult
push 0x01
add
    
```

Ergebnis: _____

1.d) DLX Architektur

___/3P



Trage die fehlenden Namen der Komponenten ein. Mögliche Begriffe: ALU, Daten, Adresse, Steuersignale, Datenspeicher, Stack, Programmspeicher, Register, Pipeline, MEM, ID, IF, CPI, WB, EX

2) Praxis Assemblerprogrammierung

```
.include "m16def.inc"
clr XH
ldi XL, 0x67
clr YH
ldi YL, 0x60
ldi R16, 0x02
out SPH, R16
ldi R16, 0x00
out SPL, R16
rcall strcat
end:
rjmp end
strcat:
ld R16, X+
cpi R16, 0x00
brne strcat
dec XL
cp:
ld R16, Y+
st X+, R16
cpi R16, 0x00
brne cp
ret
```

Speicherauszug:

0x5F	0x60	0x61	0x62	0x63	0x64	0x65	0x66	0x67	0x68	0x69	0x6A	0x6B	0x6C	0x6D	0x6E	0x6F	0x70	0x71	0x72
0x37	0x35	0x41	0x48	0x45	0x4C	0x49	0x00	0x48	0x69	0x20	0x00	0x4A	0x5A	0x42	0x6C	0x63	0x52	0x51	0x32

2.a) Wie groß ist das Programm in Bytes? ___/1P

___ Bytes

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

___ Takte

2.c) Ausführung des Programms ___/3P

Was steht im Speicher nach der Ausführung?

0x5F 0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6A 0x6B 0x6C 0x6D 0x6E 0x6F 0x70 0x71 0x72

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

Welche Werte haben die Register nach der Ausführung?

R16	X	Y	SP

2.d) Fragen zum Programm ___/3P

Das Programm zählt die Anzahl der 0x00 Bytes

richtig falsch

Das Programm nutzt den Stack

Es wird nur das Register R16 verwendet

Es werden Daten kopiert

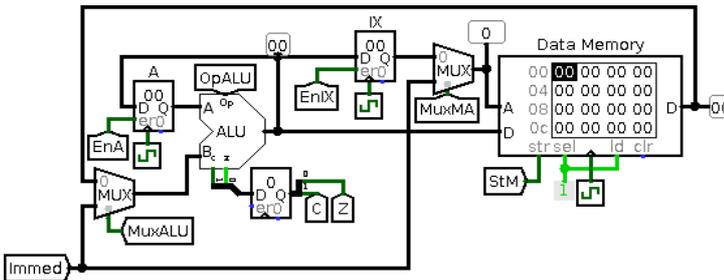
Die Laufzeit ist abhängig von den Daten im Speicher

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

2.e) Wie verändern sich die Register? ___/2P

	R16	R17	R18	R19
	0x80	0x01	0x82	0x02
add R16, R18				
adc R17, R19				
inc R19				
sub R17, R19				
eor 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) Setze die entsprechenden Steuersignale für den Datenpfad ___/4P

Befehl	EnA	EnIX	StM	MuxALU	MuxMA	OpALU	Beschreibung
ld IX, A							Lädt Register IX mit dem Wert aus Register A
or A, Imm.							Ver-ODER-t Register A mit Konstante
ld A, (Imm.)							Lädt Register A mit dem Wert an der Adresse Imm.
st (IX), A							Speichert Register A an der Adresse IX
nop							Führt keine Operation aus (No Operation)
ld IX, (IX)							Lädt Register IX mit dem Wert an der Adresse IX
and A, (IX)							Ver-UND-et Register A mit Wert an Adresse IX
st (IX), Imm.							Speichert die Konstante an Adresse IX

3.b) Fragen zum Logisim Prozessor ___/4P

- | | | |
|---|--------------------------|--------------------------|
| | richtig | falsch |
| Der Prozessor unterstützt indirekte Adressierung | <input type="checkbox"/> | <input type="checkbox"/> |
| Es gibt zwei Register zur allgemeinen Verwendung (GPR) | <input type="checkbox"/> | <input type="checkbox"/> |
| Es handelt sich um eine typische RISC Architektur | <input type="checkbox"/> | <input type="checkbox"/> |
| CPI ist konstant 2 | <input type="checkbox"/> | <input type="checkbox"/> |
| Man kann ein Programm schreiben, dass die Operation "links schieben" durchführt | <input type="checkbox"/> | <input type="checkbox"/> |
| Man kann hier den Von-Neuman-Flaschenhals mittels Caches vermeiden | <input type="checkbox"/> | <input type="checkbox"/> |
| Der Architektur fehlt ein eigener Stackpointer | <input type="checkbox"/> | <input type="checkbox"/> |
| Es handelt sich um eine Harvard Architektur | <input type="checkbox"/> | <input type="checkbox"/> |

3.c) DLX Pipeline: Zeichne die Datenhazards ein ___/2P

