Name	

Matrikel-nummer	

Dauer der Prüfung: 120 Minuten.
Lösen Sie die Aufgaben direkt auf diesen Aufgabenblättern.
Benötigte Punkte um Prüfung zu bestehen: 55

Gesamtpunktzahl: 110 Punkte. ***Viel Erfolg!***

Aufgabe 1 (10 Punkte)

Beantworten Sie die folgenden Aussagen (Richtige Antwort: +1 Punkt; Keine Antwort: 0 Punkt; Falsche Antwort: -1 Punkt):

a) Die CPU Performance kann verbessert werden, indem entweder die Länge der Taktzyklen verkürzt oder die Anzahl der Taktzyklen, welche für das Programm benötigt werden, verringert wird. Ja ☐ Nein ☐

b) Bei einem endlichem Automaten (Finite-State Machine (FSM)) kann die Zustandsübergangsfunktion (next state function) nur durch die Eingangssignale bestimmt werden. Ja ☐ Nein ☐

c) Durch das Hinzufügen von mehr Stufen zu einem Pipeline Prozessor erhöht sich normalerweise die Zahl von Control und Data Hazards. Ja ☐ Nein ☐

d) Mittels Forwarding kann eine CPI von 1 erreicht werden, selbst wenn es Abhängigkeiten der Daten gibt. Ja ☐ Nein ☐

e) Superpipelined Prozessoren haben eine kürzere Instruction Latency als Superscalare Prozessoren Ja ☐ Nein ☐

f) Die maximale Busgeschwindigkeit hängt nicht von der Anzahl der Teilnehmer am Bus ab. Ja ☐ Nein ☐

g) Wenn für die ALU und Branch Befehle weniger Stufen verwendet werden, und diese früher fertig sind als die anderen Befehle, wird die Performance der Pipeline nicht verbessert. Ja ☐ Nein ☐

h) Das Decodieren von Befehlen beinhaltet das Senden des abgerufenen Opcodes des Befehls zur Control Unit. Ja ☐ Nein ☐

i) Während eines einzelnen Taktzyklus können nicht mehrere Exceptions gleichzeitig auftreten Ja ☐ Nein ☐

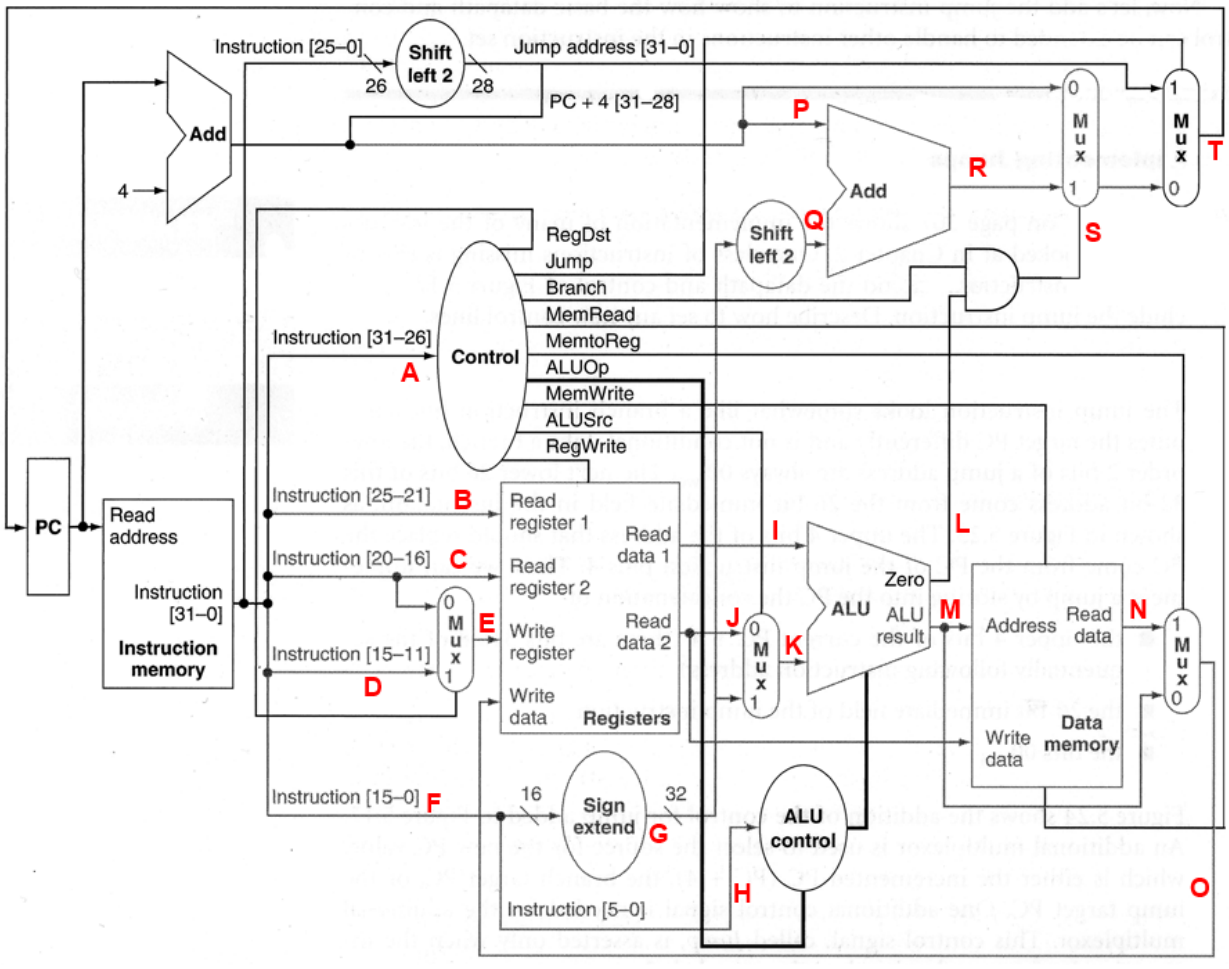
j) Ein Grund für Cache Misses ist, dass mehrere Speicher-positionen auf die gleiche Cacheposition gemappt werden. Ja ☐ Nein ☐

Aufgabe 2 (15 Punkte)

Gegeben sei folgende Single-Cycle Implementierung der MIPS ISA. Nehmen Sie an, ein Programm führe an der Instruktionsspeicher-Adresse `0x0000'1004` den Befehl `lw $r21, 0xC($r20)` aus. Folgender Zustand des Registerfiles und des Datamemory sei gegeben:

Register file	
Register	**Content**
PC	0x0000'1004
$r20	0x1004'0000
$r21	0x0000'0004
$r22	0x1004'000E
$r23	0x0000'0008

Data memory	
Address	**Content**
0x1004'0000	0x2FFF'0000
0x1004'0004	0x2FFF'0001
0x1004'0008	0x2FFF'0002
0x1004'000C	0x2FFF'0003
0x1004'0010	0x2FFF'0004
0x1004'0014	0x2FFF'0005



Bemerke: Für die folgenden Aufgaben wird `0x` als Präfix für hexadezimale Werte verwendet.

Beantworten Sie folgende Fragen basierend auf obiger Implementation und Register-/ Speicherzustände

a) (3 Punkte) Wie ist der Befehl `lw $r21, 0xC($r20)` codiert? Geben Sie die Binärdarstellung des Befehls an und dokumentieren Sie Ihren Lösungsweg. Der Opcode von `lw` ist `0x23`.
(Tipp: `lw` ist ein I-Format Befehl, die Syntax eines `lw` Befehls ist `lw $rt, offset($rs)`)

Antwort:

b) (1 Punkt) Was berechnet die ALU im gegebenen Befehl?

Antwort:

c) (3 Punkte) Welche Werte haben die Kontrollsignale RegDst, Jump, Branch, MemRead, MemtoReg, MemWrite, ALUsrc und RegWrite? Geben Sie auch (allfällige) Don't Care Fälle an (mit X).

Antwort:

d) (8 Punkte) Geben Sie für jeden im Diagramm dargestellten Buchstaben (A...T) das dazugehörige anliegende Signal an. Sie können binäre, hexadezimale und dezimale Werte verwenden. Alle (allfälligen) undefinierten Signale mit `X` kennzeichnen.
Falls Sie a) nicht lösen konnten, gehen Sie von einer Binärdarstellung der Instruktion von „`1000'1110'1101'0111'0000'0000'0000'0110`“ aus und decodieren diese zuerst (Beachten Sie, dass dies *nicht* die Lösung zu Teil a) ist).

Antwort:

Aufgabe 3 (10 Punkte)

a) (1 Punkt) Was sind die Vorteile des hierarchischen Aufbaus des Speichers?

Antwort:

b) (2 Punkte) Was bedeuten DRAM und SRAM? Warum werden beide verwendet (was sind die jeweiligen Vorteile?)?

Antwort:

c) (7 Punkte) Angenommen wir haben einen Prozessor mit einer Ideal CPI von 1.5. Die Taktgeschwindigkeit betrage 2 GHz. Ein Hauptspeicherzugriff benötigt 50 ns. Der Anteil an Load/Store Befehlen betrage 20%. Die Missrate für den Instruction Cache betrage 1%, für den Data Cache 5%.

i. (4 Punkte) Wie gross ist die Gesamt-CPI?

Antwort:

ii. (1 Punkt) Wie gross ist der Anteil (als Bruch) an Memory Stalls an der gesamten Ausführungszeit?

Antwort:

iii. (2 Punkte) Nehmen Sie an, es wird zusätzlich ein Level 2 Cache hinzugefügt mit einer Zugriffszeit von 20 Takten für Treffer und Fehlzugriff. Die Fehlerzugriffsrate auf den Hauptspeicher wird dabei auf 0.5% reduziert. Welches ist die Gesamt CPI dieses System (bei einer Ideal CPI von 1.5)?

Antwort:

Aufgabe 4 (20 Punkte)

Gegeben sei folgender MIPS Code:

```
Loop: lw   $t0, 0($s1)
      lw   $t1, 200($s1)
      addu $t2, $t0, $t1
      sw   $t2, 0($s1)
      addi $s1, $s1, -4
      bne  $s1, Loop
```

a) (3 Punkte) Wie viele Taktzyklen werden benötigt, um dieses Codefragment auf einer regulären (nicht pipeline Multicycle) Architektur auszuführen (nehmen Sie Single Loop an)? Erklären Sie Ihre Antwort.

Antwort:

b)

i. (10 Punkte) Wie viele Taktzyklen werden benötigt, um dieses Codefragment auf einer einfachen Pipeline mit normalem Forwarding und Bypassing auszuführen, wenn das Ergebnis des Branchbefehls (neuer PC) am Ende der ID-Stufe verfügbar ist (nehmen Sie Single Loop an)? Welche Arten von Hazards treten auf? Stellen Sie die Zeiten schematisch dar (inklusive Stalls).

Antwort:

ii. (7 Punkte) Ändern Sie die Reihenfolge der Befehle und die Position des Looplabels, so dass die minimale Anzahl Taktzyklen benötigt wird. Sie dürfen nicht die Namen der Register oder die Opcodekonfigurationen ändern.

Antwort:

Aufgabe 5 (15 Punkte)

a) (12 Punkte) Sei `m` ein 3x1-Array von Integer Zahlen. Gegeben sei folgendes Programmfragment:

```
for(int i = 0; i<6; i++) {
    m[i] = 3;
    if(i == 1) {
          m[i] = 2 * m[i];
    }
}
```

Geben Sie den Assemblercode für dieses Programm an. Die Startadresse von `m` sei im Register `$s1` gespeichert. Beachten Sie, dass die Startadresse nicht verloren gehen darf.

Antwort:

b) (2 Punkte) Nehmen Sie an, ein Prozessor macht Branch Prediction beim IF des Programms aus Aufgabe a).
Wie oft ist die Vorhersage falsch (zu Beginn wird *predict not taken* (0 bzw. 00) angenommen)

i. bei einem 1 bit Prädiktor,

Antwort:

ii. bei einem 2 bit Prädiktor?

Antwort:

c) (1 Punkt) Erkläre die Ergebnisse aus Teil b).

Antwort:

Aufgabe 6 (20 Punkte)

a) (4 Punkte) Was ist die Ausgabe des folgenden Programmes? Gehen Sie von einer 32-bit Maschine aus, Bytereihenfolge Little-Endian.

```
#include <stdio.h>

typedef struct {
      char a[2];
      unsigned short int b;
      unsigned int c;
} MyStruct;

typedef union {
      char a[2];
      unsigned short int b;
      unsigned int c;
} MyUnion;

int main(void) {
      MyStruct s;
      MyUnion u;

      printf("%d\n", sizeof(MyStruct));
      printf("%d\n", sizeof(MyUnion));

      s.b = 0xABCD;
      s.c = 0x12345678;
      u.b = 0xABCD;
      u.c = 0x12345678;

      printf("%x\n", s.b);
      printf("%x\n", u.b);

      return 0;
}
```

Antwort

Antwort

Schreiben Sie die Ausgaben direkt hinter die jeweiligen Zeilen im obigen Programm.

b) (4 Punkte) Gegeben sei folgendes Programmfragment:

```
void square(int x, int result) {
      result = x*x;

}

int main(void) {
      int a = 3;
      int r = 2;

      square(a, r);
      printf("%d", r);

      return 0;
}
```

i. (1 Punkt) Was ist die Ausgabe des Programmes und warum?

Antwort:

ii. (1 Punkt) Wie sieht der Stack nach der Zeile „`result = x*x;`“ aus?

Antwort:

iii. (2 Punkte) Wie muss das Programm verändert werden, damit der Funktionsaufruf `square(x, r)` das Quadrat von `x` in der Variablen `r` speichert?

Antwort:

c) (6 Punkte) Geben Sie für jede Zuweisung im folgenden Programm den Wert der linken Seite des Ausdrucks an. Nehmen Sie an, dass alle Zuweisungen nacheinander ausgeführt werden und dass die Adresse des `blocks` Arrays `1004` sei.

```
int main(void) {
   char blocks[3] = {'A','C','E'};
   char *ptr = &blocks[0];
   char temp;

   temp = blocks[0];

   temp = *(blocks + 2);
   temp = *(ptr + 1);
   temp = *ptr;

   ptr = blocks + 1;
   temp = *ptr;
   temp = *(ptr + 1);

   ptr = blocks;
   temp = *++ptr;
   temp = ++*ptr;
   temp = *ptr++;
   temp = *ptr;

   return 0;
}
```

Antwort

Schreiben Sie die Zuweisungen direkt hinter die jeweiligen Zeilen im obigen Programm.

d) (2 Punkte) Was ist die Ausgabe des folgenden Codes? Geben Sie allfällige Verbesserungen an.

```
#include <stdio.h>
#define AREA(a, b) a*b

int main(void) {
      int a = 1; int b = 3;
      int c = 2; int d = 4;

      printf("%d\n", AREA(a+c, b+d));
      return 0;
}
```

Antwort

Schreiben Sie die Ausgabe direkt hinter die entsprechende Zeile und markieren Sie auch allfällige Änderungen direkt im obigen Programm.

e) (4 Punkte) Was ist die Ausgabe des folgenden Programmes?

```
#include <stdio.h>
#define SIZE 4

int foo(a, b) { return a*b; }
int bar(a, b) { return a+b; }

int iterate(int* a, int (*op)(int, int)) {
      int i;
      int c = a[0];

      for (i = 1; i < SIZE; i++) {
            c = op(c, a[i]);
      }

      return c;
}

int main(void) {
      int a[SIZE] = {1,2,3,4};
      int b[SIZE] = {5,6,7,8};

      printf("%d\n", iterate(a, &foo));
      printf("%d\n", iterate(b, &bar));

      return 0;
}
```

Antwort

Schreiben Sie die Ausgaben direkt hinter die jeweiligen Zeilen im obigen Programm.

Aufgabe 7 (20 Punkte)

Beantworten Sie die folgenden Aussagen:

a) (2 Punkte) Wie funktioniert die Subtraktion mittels 2-Komplement? Zeigen Sie wie die äquivalente binäre Operation von 5 – 6 = -1 als 4 Bit 2-Komplement dargestellt werden kann.

Antwort:

b) (1 Punkt) Was ist der Unterschied zwischen RISC und CISC Prozessoren bei der Codierung der Befehle?

Antwort:

c) (2 Punkte) Welche MIPS Operand Addressing Modes gibt es? Erklären Sie die Unterschiede.

Antwort:

d) (1 Punkt) Was ist das Befehlsformat und wie wird die Sprungadresse berechnet bei einem unbedingten Sprungbefehl jump (j)?

Antwort:

e) (2 Punkte) Was ist das Instruction-Level Parallelism (ILP) eines Programms?

Antwort:

f) (2 Punkte) Welches sind die drei Hauptkategorien von Cache Misses?

Antwort:

g) (3 Punkte) Gegeben sei ein Programm mit einer bedingten Verzweigung. Die Ausgänge für die Verzweigungen bei der Ausführung eines Progamms seien: T-T-T-N-T-T-T-N-T (T bedeutet Taken und N bedeutet Not Taken). Berechne die Vorhersagen und die Richtigkeit für die folgenden Branch Predictions.
   i. (1 Punkt) 1-Bit Prädiktor, initialisiert mit Predict Taken.

Antwort:

   ii. (1 Punkt) 2-Bit Prädiktor, initialisiert mit Predict Taken.

Antwort:

   iii. (1 Punkt) Erklären Sie Ihre Antworten.

Antwort:

h) (1 Punkt) Wozu gibt es das Mehrzweckregister $sp?

Antwort:

i) (2 Punkte) Was ist ein Buffer Overflow? Können Sie ein Beispiel geben, wann dies auftreten kann?

Antwort:

j) (2 Punkte) Welche Funktionen werden für Dynamic Memory Allocation verwendet?

Antwort:

k) (2 Punkte) Was ist der Unterschied zwischen einem Direct Mapped Cache und einem Fully Associative Cache?

Antwort: