Technische Grundlagen der Informatik

Vorrechenübung 12.2.14

TECHNISCHE UNIVERSITÄT DARMSTADT

Embedded Systems & Applications

Übersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- Klausurregeln/Allgemeines
- Stoffübersicht
  - Auswahl einiger Folien für Vorrechenübung
- Vorrechenübung
  - MIPS-Eintakt-Befehlserweiterung
  - MIPS-Mehrtakt-Befehlserweiterung
  - MIPS-Pipeline/Hazards

Embedded Systems & Applications

Klausurregeln

TECHNISCHE UNIVERSITÄT DARMSTADT

- Freitag den 14.02.14 um 18:00 Uhr
- Dauer: 90min
- Raumverteilung wird im Moodle bekanntgegeben
  - Ungleich der Raumverteilung vom letzten Mal
- Seien Sie pünktlich da!
- Wir gehen die Klausur gemeinsam durch, danach startet die Bearbeitungszeit

Embedded Systems & Applications

Klausurregeln - Fachliches

TECHNISCHE UNIVERSITÄT DARMSTADT

- Relevanter Stoff: Hauptsächlich Kapitel 5-7, Übungen 8-12
- Aber: vorheriger Stoff wird als bekannt vorausgesetzt!

- KEINE Hilfsmittel (Ausnahme: Wörterbuch für ausländische Studierende) – außer Stift und Lineal

- Verilog/MIPS-Syntaxblatt wird der Klausur beiliegen

- Kommentare in Programmen werden ebenfalls bewertet!

- Bewertet wird nicht nur die Lösung, auch der Lösungsweg

Klausurregeln - Formales

TECHNISCHE UNIVERSITÄT DARMSTADT

- Dokumentenechter Stift, blau oder schwarz, KEIN rot oder grün, auf KEINEN FALL Bleistift (auch nicht zum Vorzeichnen)
- Studentenausweis und Lichtbildausweis
- Kein eigenes Papier, bei Bedarf von uns
- Durchgestrichenes DÜRFEN wir nicht bewerten – auch wenn es korrekt ist
- Essen und Trinken erlaubt, aber bitte Rücksicht

Embedded Systems & Applications

Klausurregeln – Bewertung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Bsc INF PO 2009
- In der ersten Teilklausur 90 Punkte
- In der zweiten Teilklausur 90 Punkte
- Im Praktikum 45 Punkte
- Punkte werden einfach addiert (225 möglich)
- Mit 112,5 Punkten (50%) auf jeden Fall bestanden

Stoffübersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- Kurze Übersicht der letzten drei Kapitel
- Wiederholung einzelner Folien für die Vorrechenübung
- Nochmal: sämtlicher Stoff aus Vorlesung und Übungen kann abgefragt werden

Embedded Systems & Applications

Stoffübersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- Kapitel 5
  - Addiererschaltungen
  - ALU
  - Multiplikation
  - Division
  - Gleitkommadarstellung
  - Festkommadarstellung
  - Register
  - Speicher
  - PLA/FPGA

Embedded Systems & Applications

Stoffübersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- Kapitel 6
  - MIPS-Assembler
  - Befehle
  - Registerfeld
  - Speicherverwaltung
  - Befehlsformate
  - Maschinensprache
  - Prozeduren
  - Stackverwaltung
  - Adressierungsarten
  - Ausnahmebehandlung

Stoffübersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- Maschinensprache
  - Computer verstehen nur 0'en und 1'en
  - Maschinensprache: Binärdarstellung von Befehlen
  - 32b Befehle
    - Regularität vereinfacht Entwurf: Daten und Befehle sind beides 32b Worte
  - Drei Befehlsformate
    - R-Typ: Operanden sind nur Register
    - I-Typ: Register und ein Direktwert
    - J-Typ: für Programmsprünge (kommt noch)

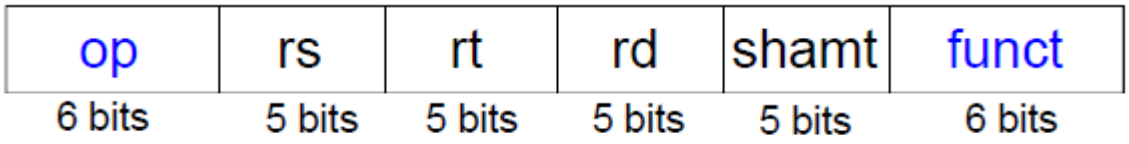
Embedded Systems & Applications

Stoffübersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- Befehlsformat R-Typ
  - *Register Typ*
  - 3 Registeroperanden
    - `rs, rt:` Quellregister
    - `rd:` Zielregister
  - Andere Angaben in binärkodiertem Befehl:
    - `op:` *Operations-Code* oder *Opcode* (ist 0 für Befehle vom R-Typ)
    - `funct:` Auswahl der genauen *Funktion*
      Opcode und Funktion zusammen bestimmen die auszuführende Operation
    - `shamt:` Schiebeweite für Shift-Befehle, sonst 0

R-Typ



Embedded Systems & Applications

Stoffübersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- Beispiel für Befehle vom R-Typ

Assemblersprache

```
add $s0, $s1, $s2
sub $t0, $t3, $t5
```

Felder in Befehlswort

op	rs	rt	rd	shamt	funct
0	17	18	16	0	32
0	11	13	8	0	34
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

Maschinensprache

op	rs	rt	rd	shamt	funct	
000000	10001	10010	10000	00000	100000	(0x02328020)
000000	01011	01101	01000	00000	100010	(0x016D4022)
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	

Beachte andere Reihenfolge der Register in Assembler-Sprache:

```
add rd, rs, rt
```


Embedded Systems & Applications

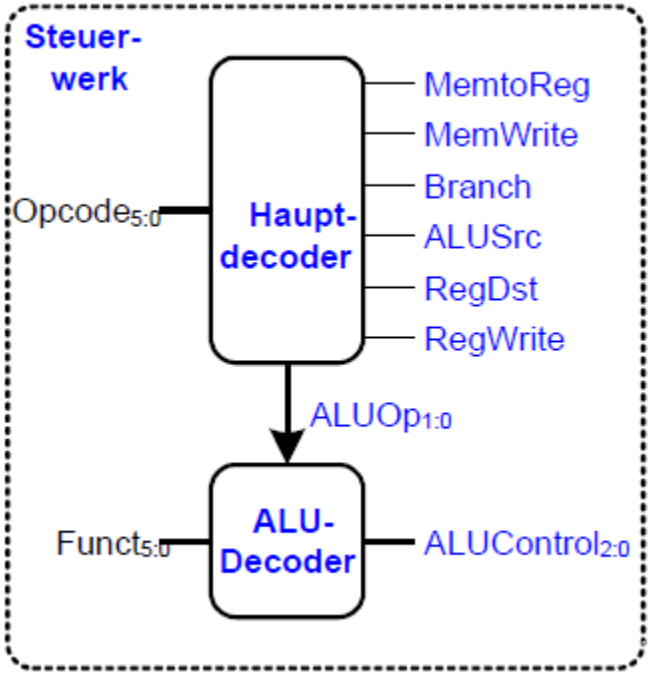
Stoffübersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- Kapitel 7
  - Eintaktprozessor
    - Datenpfad
    - Steuerwerk
  - Mehrtaktprozessor
    - Datenpfad
    - Steuerwerk
  - Pipelining
  - Hazards
  - Ausnahmebehandlung

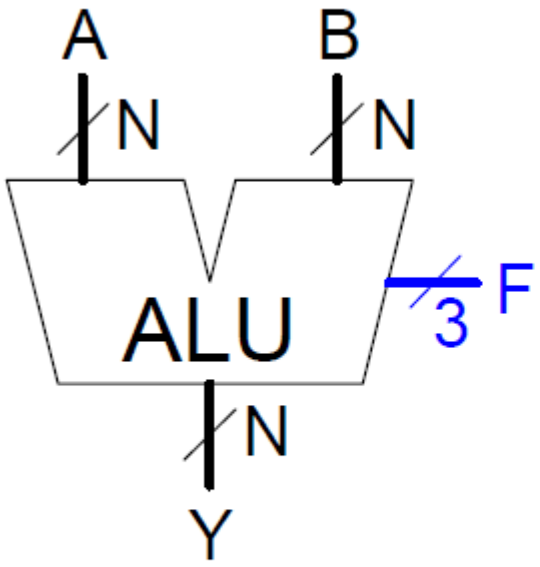
Stoffübersicht

- Steuerwerk



Stoffübersicht

- Zur Erinnerung: ALU



$F_{2:0}$	Funktion
000	A & B
001	A | B
010	A + B
011	unbenutzt
100	A & ~B
101	A | ~B
110	A - B
111	SLT

Stoffübersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- Steuerwerk: ALU-Dekoder

$ALUOp_{1:0}$	Bedeutung
00	Addiere
01	Subtrahiere
10	Werte Funct-Feld aus
11	unbenutzt

$ALUOp_{1:0}$	Funct	$ALUControl_{2:0}$
00	X	010 (Add)
X1	X	110 (Subtract)
1X	100000 (`add`)	010 (Add)
1X	100010 (`sub`)	110 (Subtract)
1X	100100 (`and`)	000 (And)
1X	100101 (`or`)	001 (Or)
1X	101010 (`slt`)	111 (SLT)

Embedded Systems & Applications

Stoffübersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

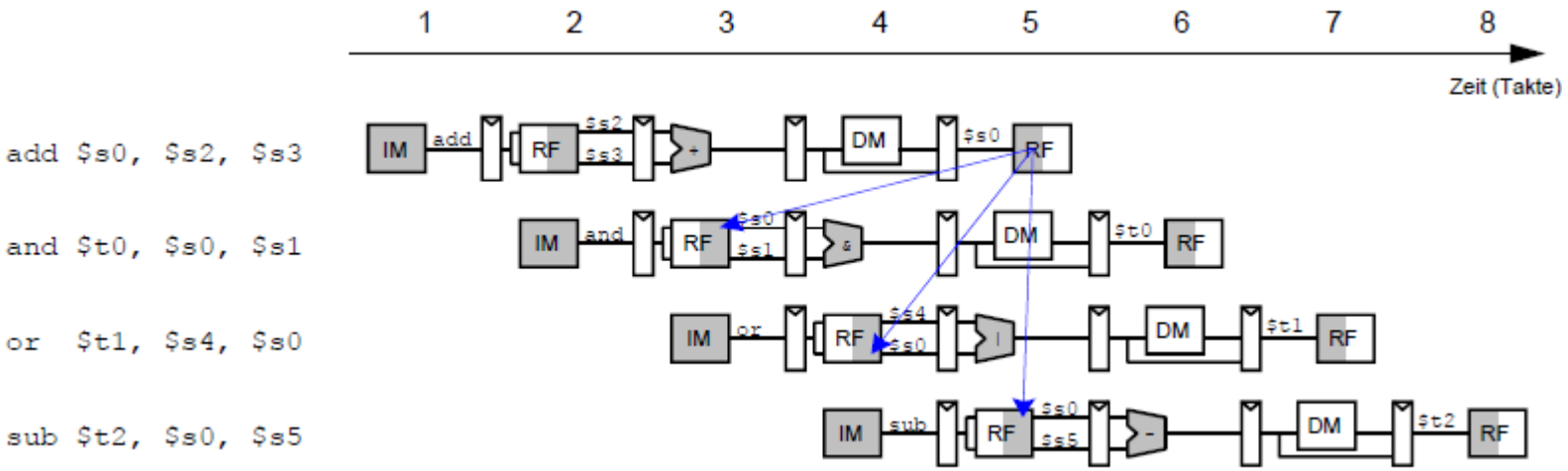
- Hazards
  - Treten auf wenn eine
    - Instruktion vom Ergebnis einer vorhergehenden abhängt
    - … diese aber noch kein Ergebnis geliefert hat
  - Arten von Hazards
    - **Data Hazard:** z.B. Neuer Wert von Register noch nicht in Registerfeld eingetragen
    - **Control Hazard:** Unklar welche Instruktion als nächstes ausgeführt werden muss
      - Tritt bei Verzweigungen auf

Embedded Systems & Applications

Stoffübersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- Data Hazard

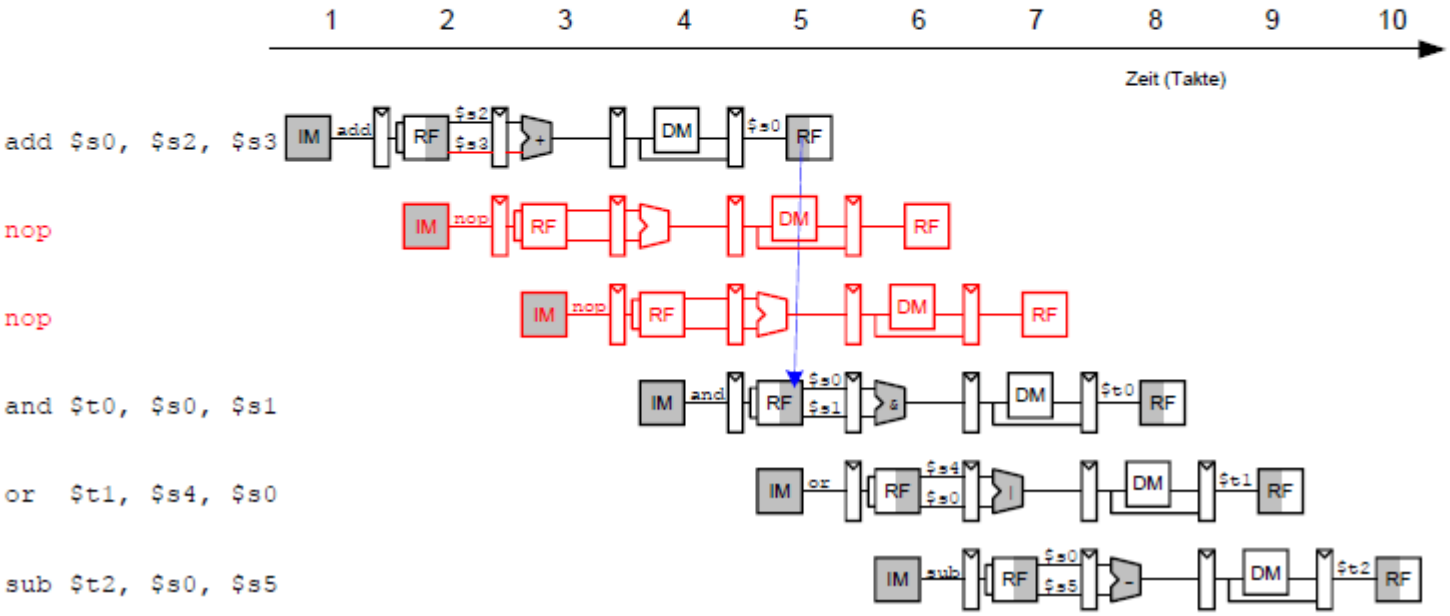


Hier: Read-after-Write Hazard (RAW)
- $s0 „muss vor Lesen geschrieben werden“

Embedded Systems & Applications

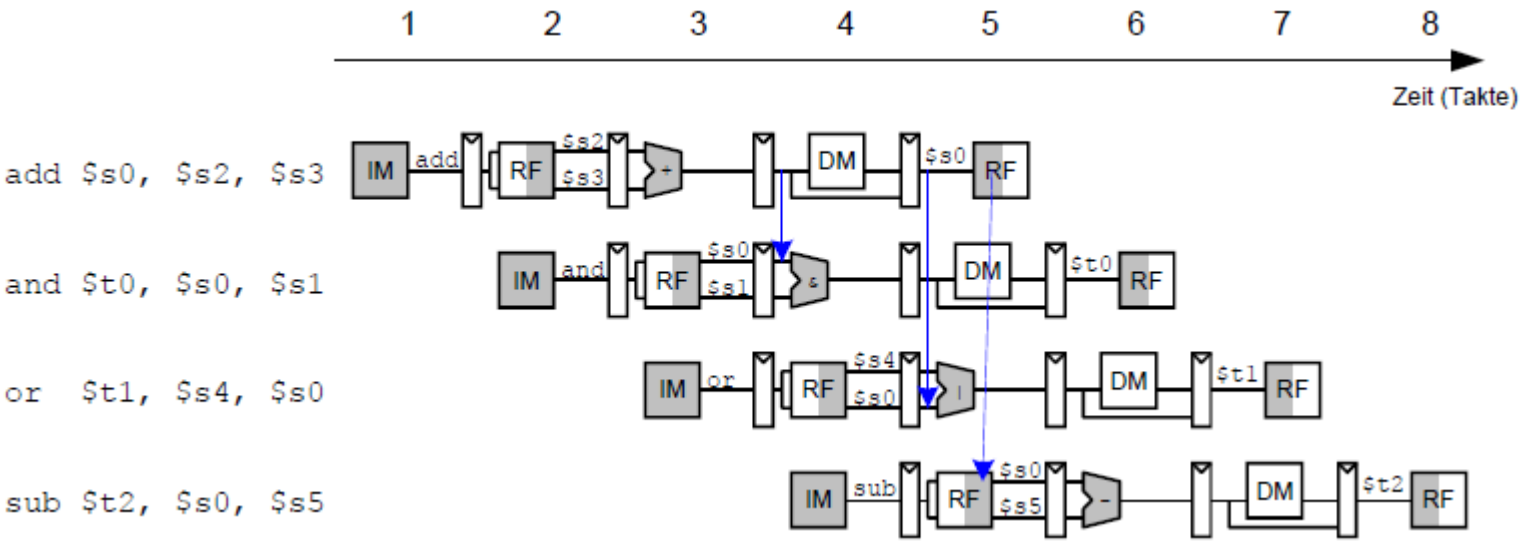
Stoffübersicht

- Beseitigung zur Compile-Zeit
  - Füge ausreichend viele `nop`s ein bis Ergebnis bereitsteht
  - Oder schiebe unabhängige Instruktionen nach vorne (statt `nop`s)



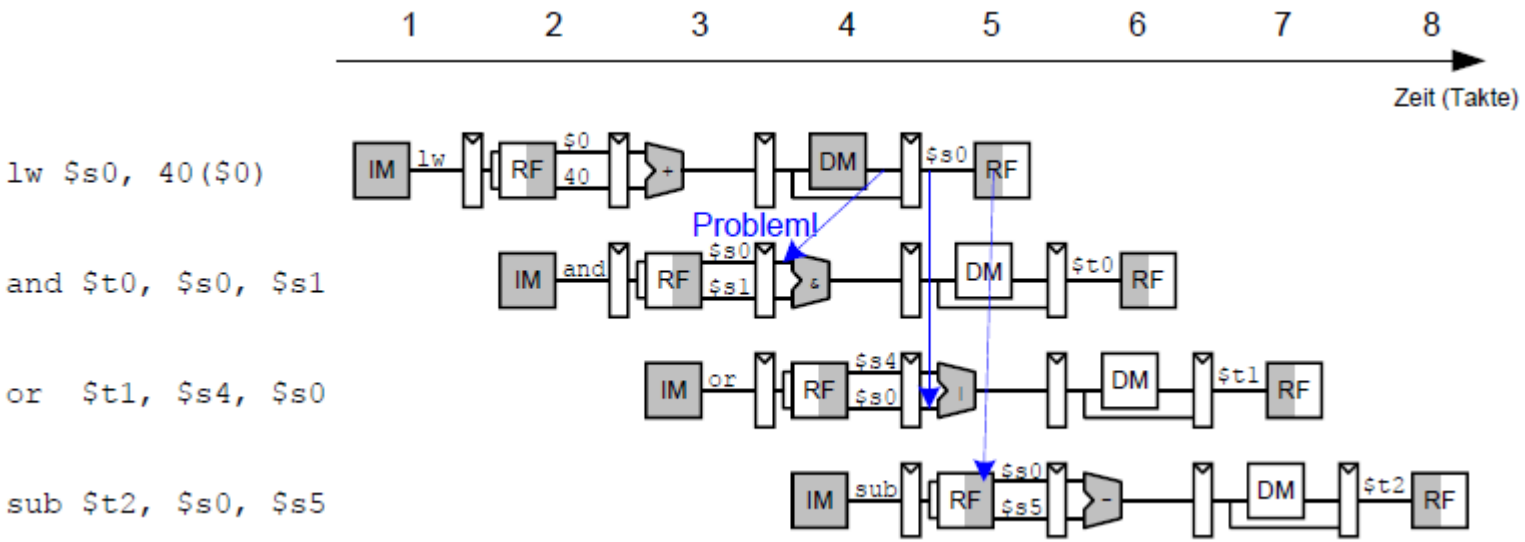
Stoffübersicht

- Data Forwarding: “Abkürzung“ einbauen



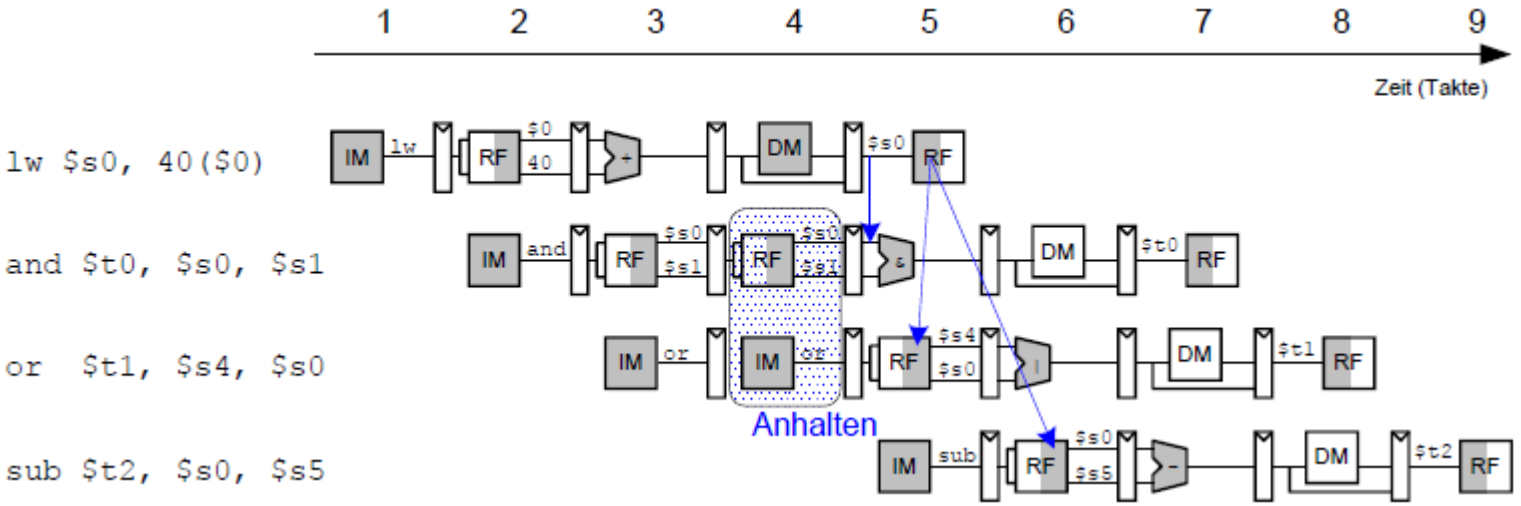
Stoffübersicht

- Anhalten des Prozessors (*stalling*)



Stoffübersicht

- Anhalten des Prozessors (*stalling*)



Aufgaben aus Klausur WS11/12

TECHNISCHE UNIVERSITÄT DARMSTADT

- Deckt nicht vollständige Klausur ab
- Auswahl aufgrund Aktualität (die letzten in der Vorlesung behandelten Themen)
- Und nochmal: sämtlicher Stoff aus Vorlesung und Übungen kann abgefragt werden

MIPS-Eintakt-Befehlserweiterung

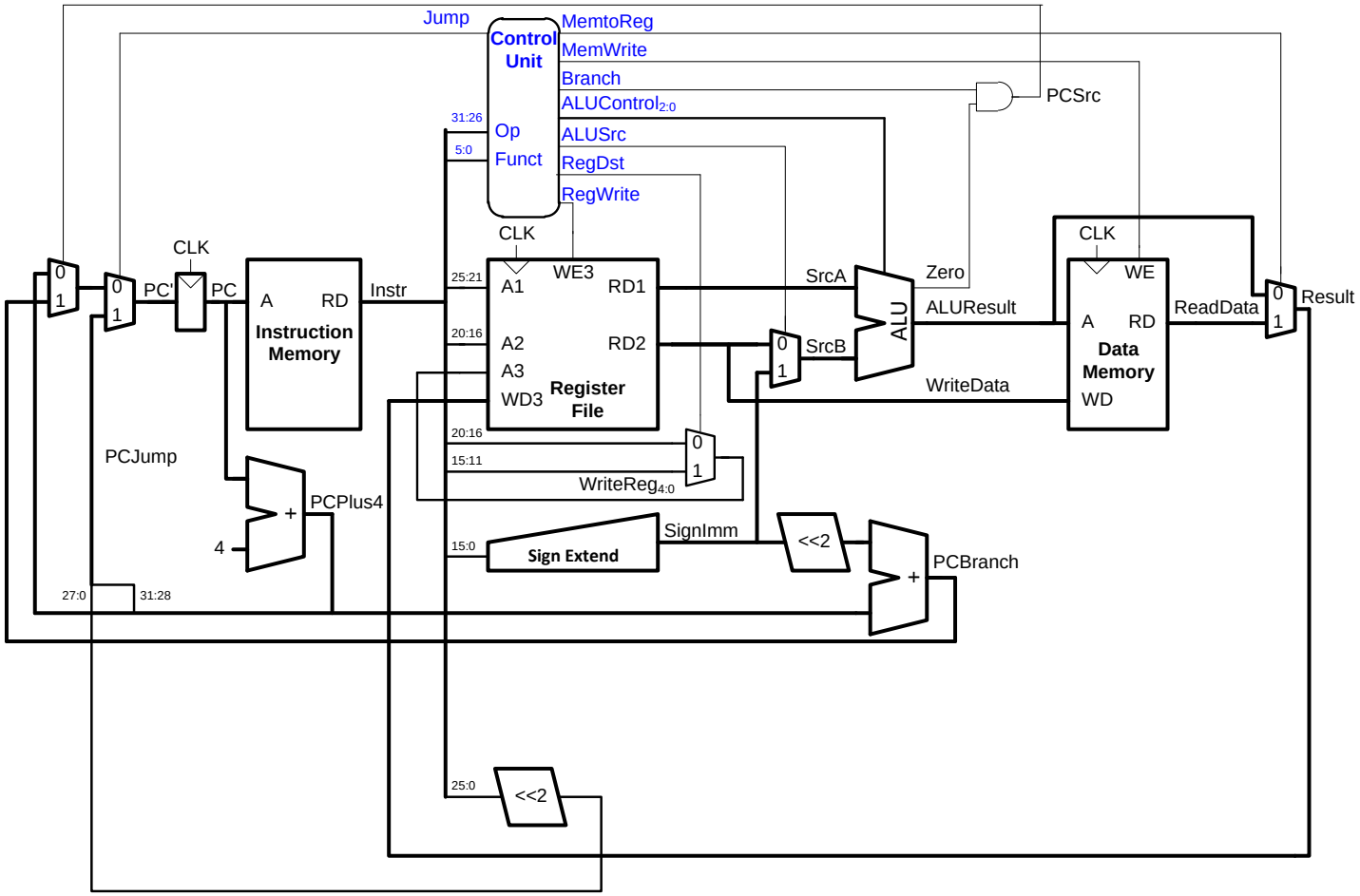
TECHNISCHE UNIVERSITÄT DARMSTADT

- In dieser Aufgabe soll die Eintaktimplementierung des MIPS-Prozessors um eine Unterstützung für die Befehle
  - mult *R*1, *R*2
  - mfhi *R*1
  - mflo *R*1

  erweitert werden. Zur Bedeutung siehe auch das MIPS-Syntaxblatt am Ende der Klausur.

- Erweitern Sie den Datenpfad auf der nächsten Seite so, dass die Befehle unterstützt werden. Sie haben hierzu eine Multiplizierer-Einheit zur Verfügung, die kombinatorisch das Ergebnis berechnet. Des weiteren stehen Ihnen Register in beliebiger Breite sowie Multiplexer zur Verfügung und Sie dürfen das Steuerwerk um beliebig viele Steuersignale erweitern.

MIPS-Eintakt-Befehlserweiterung

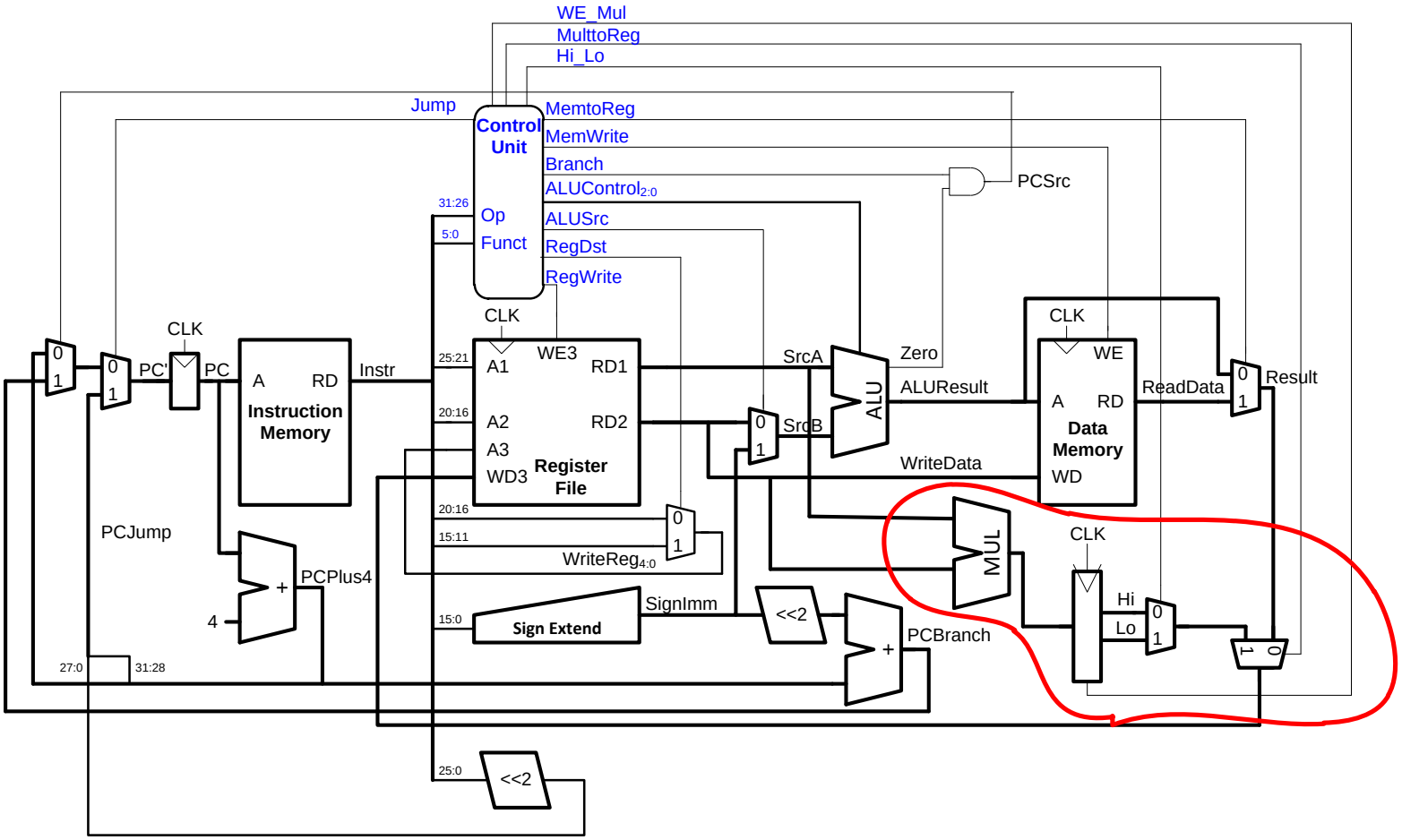


MIPS-Eintakt-Befehlserweiterung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Für die Multiplikation muss die Multiplizierer-Einheit sowie die beiden Register lo und hi eingebunden werden.
- Zur Auswahl muss ein Multiplexer eingebaut werden, der zwischen hi und lo umschalten kann
- Zusätzlich muss noch ein Multiplexer in den Datenpfad eingebunden werden, um das Ergebnis in den normalen Registersatz zu transferieren

MIPS-Eintakt-Befehlserweiterung



MIPS-Mehrtakt-Befehlserweiterung

TECHNISCHE UNIVERSITÄT DARMSTADT

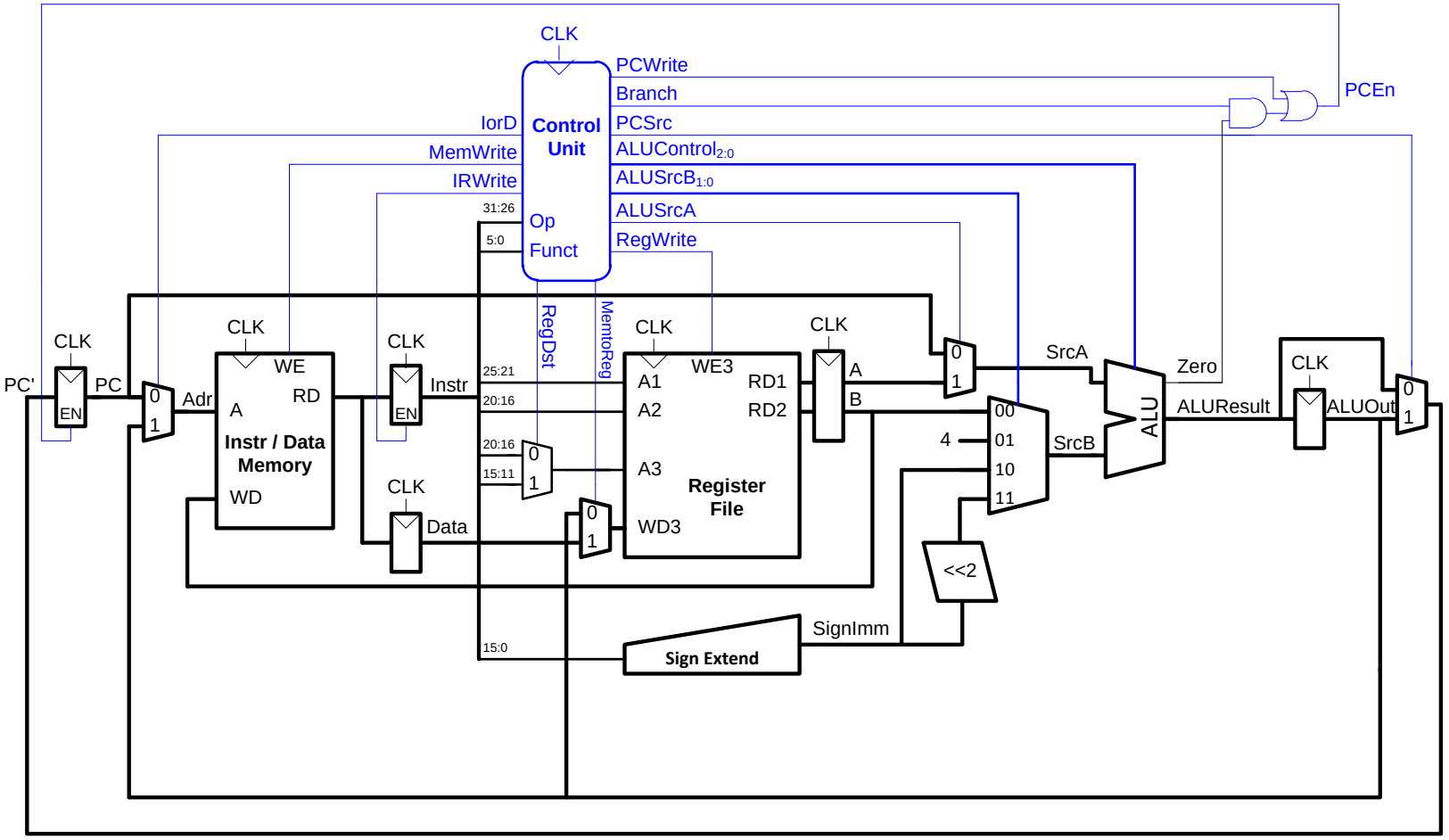
- Der Mehrtakt-MIPS Prozessor soll um einen Befehl min R1,R2,R3 erweitert werden, der die Register *R*2 und *R*3 ausliest und den Inhalt des Registers mit dem kleineren Wert in Register *R*1 speichert.

- a) Welches Befehlsformat wählen Sie? Begründen Sie Ihre Antwort.
- b) Erweitern Sie den Datenpfad des Prozessors auf der nächsten Seite, so dass der Befehl ausgeführt werden kann.

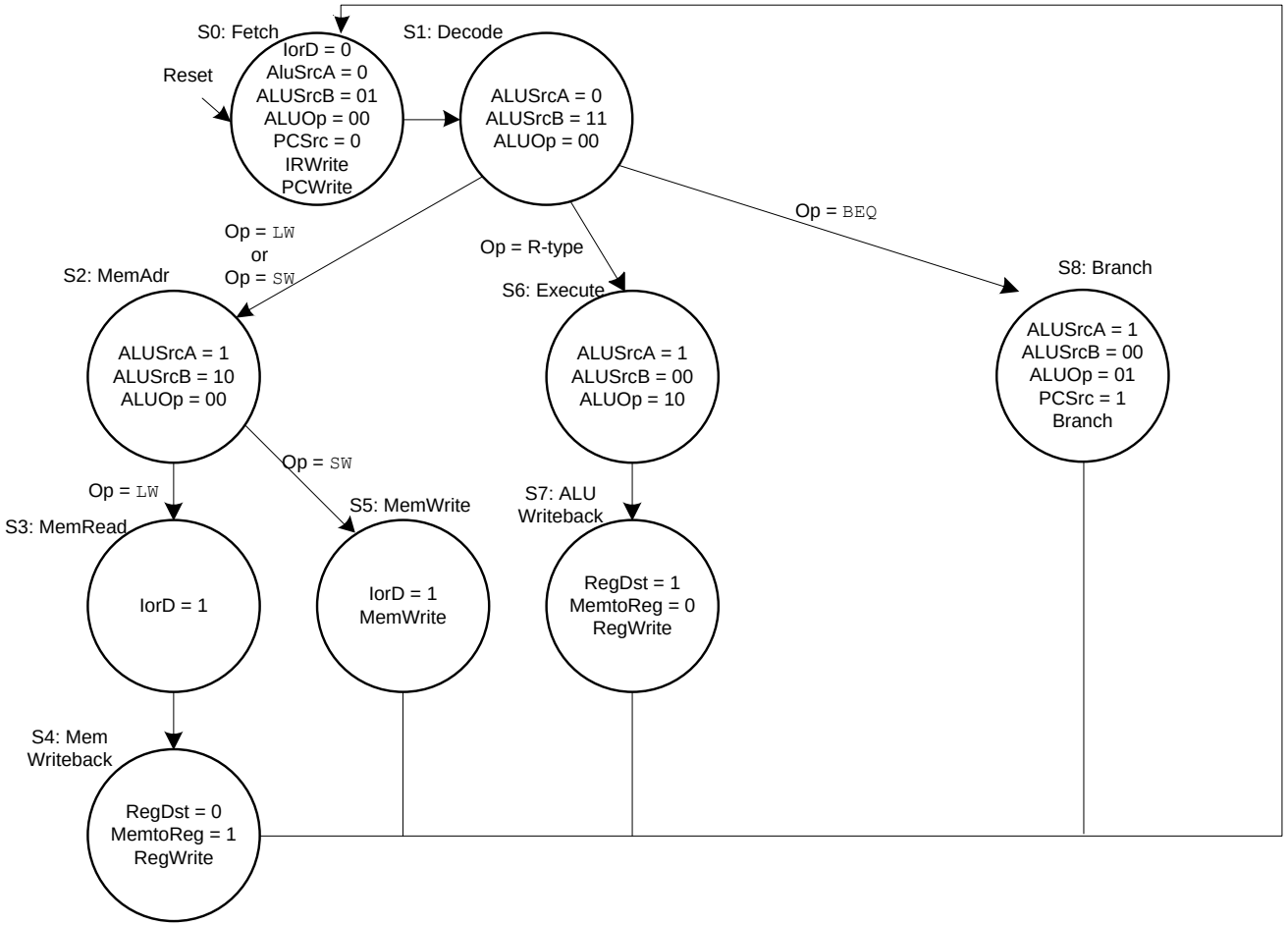
  Ihnen stehen Register in beliebiger Breite, Multiplexer und Shifter zur Verfügung.
- c) Erweitern Sie die Steuerwerk-FSM, so dass der Befehl ausgeführt werden kann. Verwenden Sie möglichst wenig neue Zustände.

Embedded Systems & Applications

MIPS-Mehrtakt-Befehlserweiterung



MIPS-Mehrtakt-Befehlserweiterung



MIPS-Mehrtakt-Befehlserweiterung

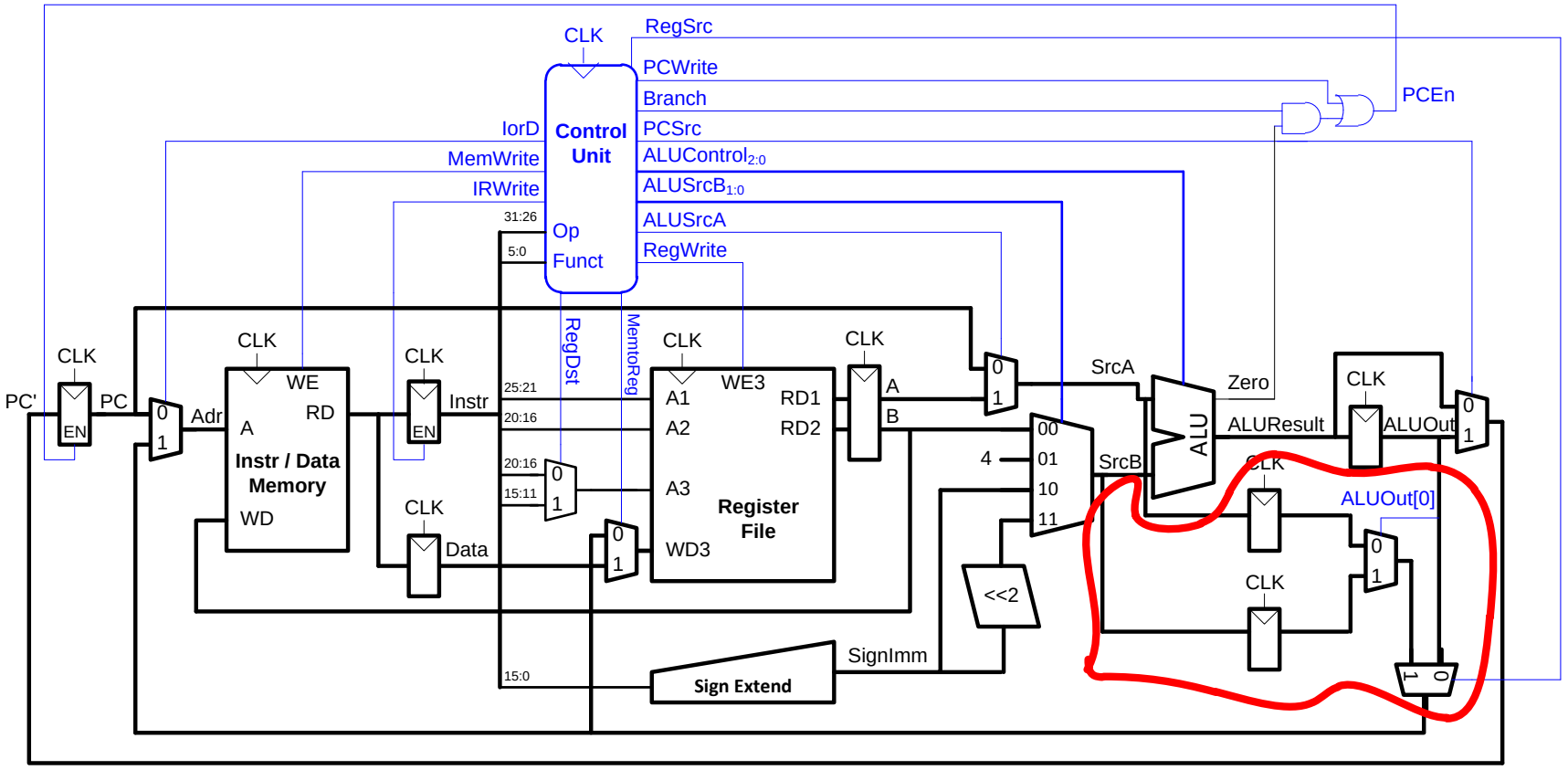
TECHNISCHE UNIVERSITÄT DARMSTADT

- **a) Lösung:**

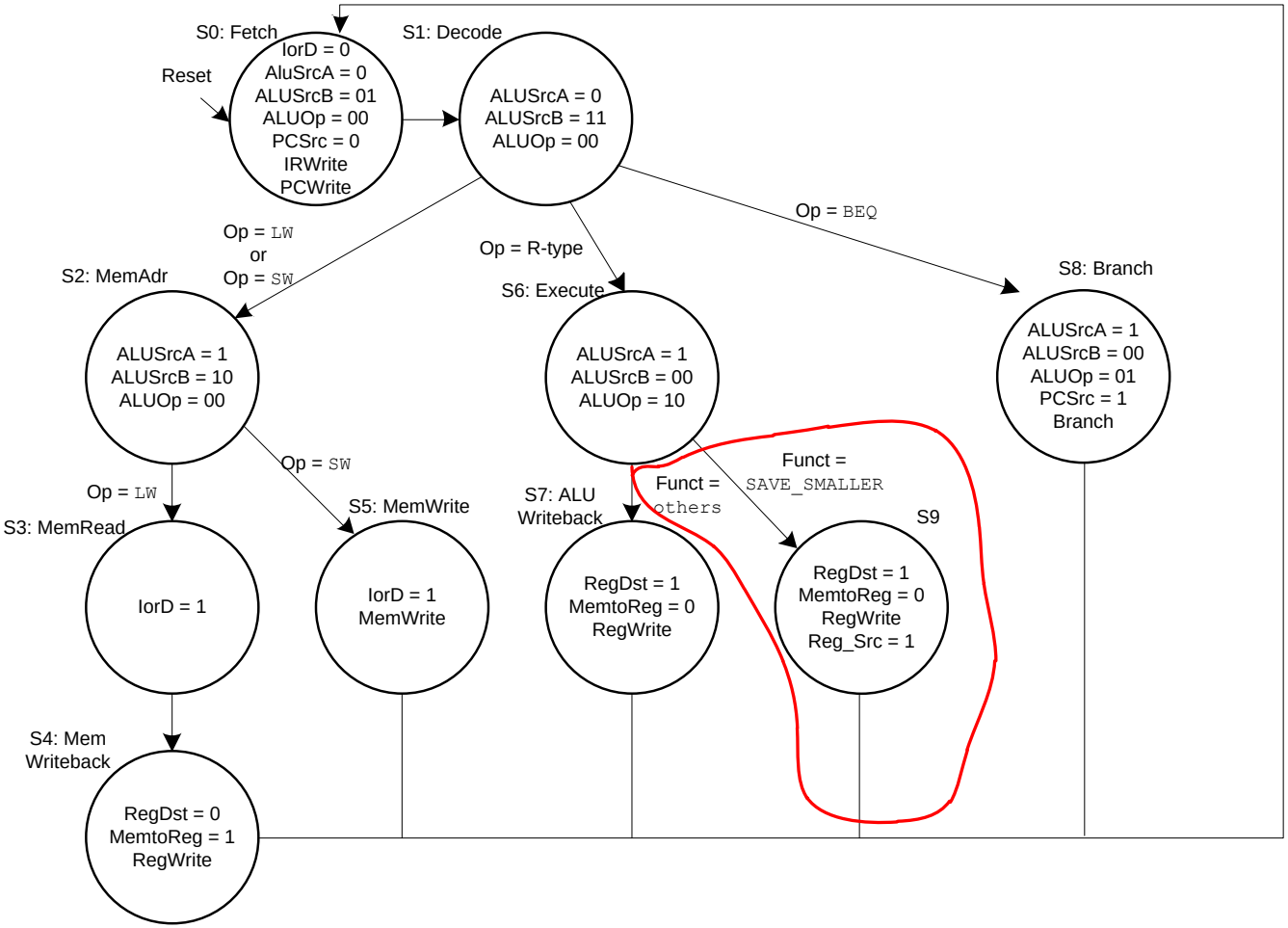
- Da 3 Register im Befehl vorhanden sind, muss das R-Typ Format verwendet werden.

Embedded Systems & Applications

MIPS-Mehrtakt-Befehlserweiterung

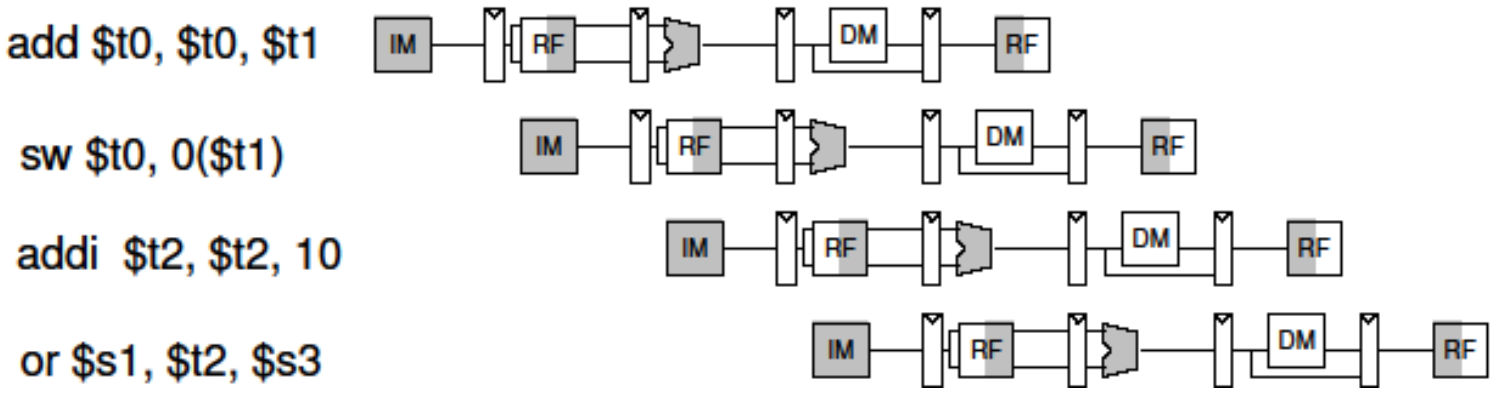


MIPS-Mehrtakt-Befehlserweiterung

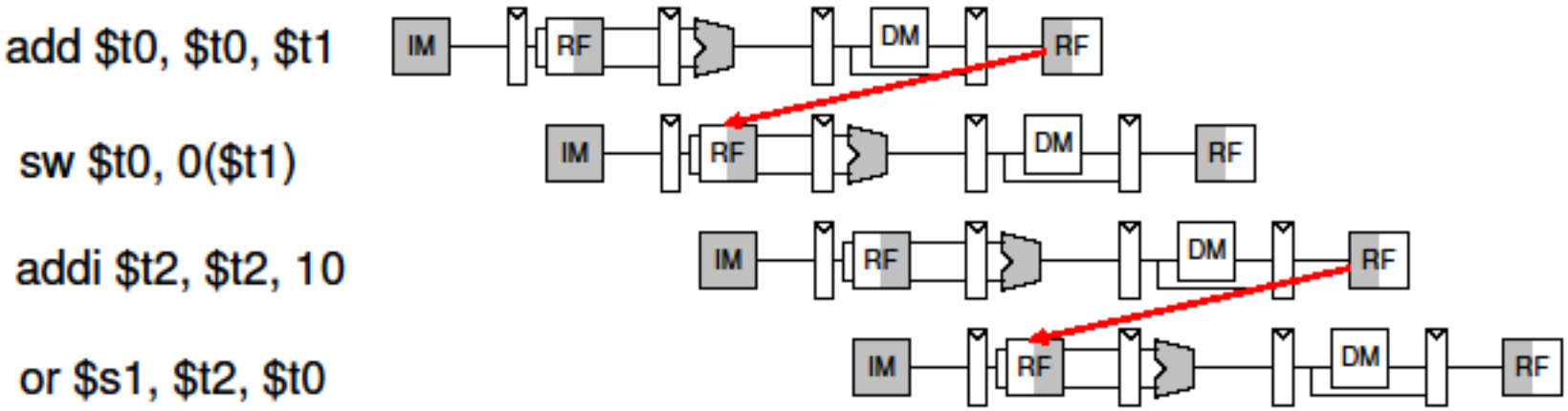


Hazards

- Gegeben ist das folgende Programmstück in MIPS-Assembler. An welchen Stellen können Hazards auftreten? Markieren Sie die Stellen im Diagramm.
- Geben Sie ein äquivalentes MIPS-Programm an, welches auf einer MIPS-CPU ohne Forwarding keine Hazards enthält.
- Die Dauer der Ausführung soll nach Ihren Änderung minimal sein.



Hazards



add $t0, $t0, $t1
addi $t2, $t2, 10
NOP
sw $t0, 0($t1)
or $s1, $t2, $s3

Viel Erfolg!

TECHNISCHE UNIVERSITÄT DARMSTADT

- Viel Erfolg bei der 2. Klausur, im Praktikum und im weiteren Studium!

- Noch Fragen: jetzt stellen, Sprechstunden oder das Forum nutzen

Embedded Systems & Applications