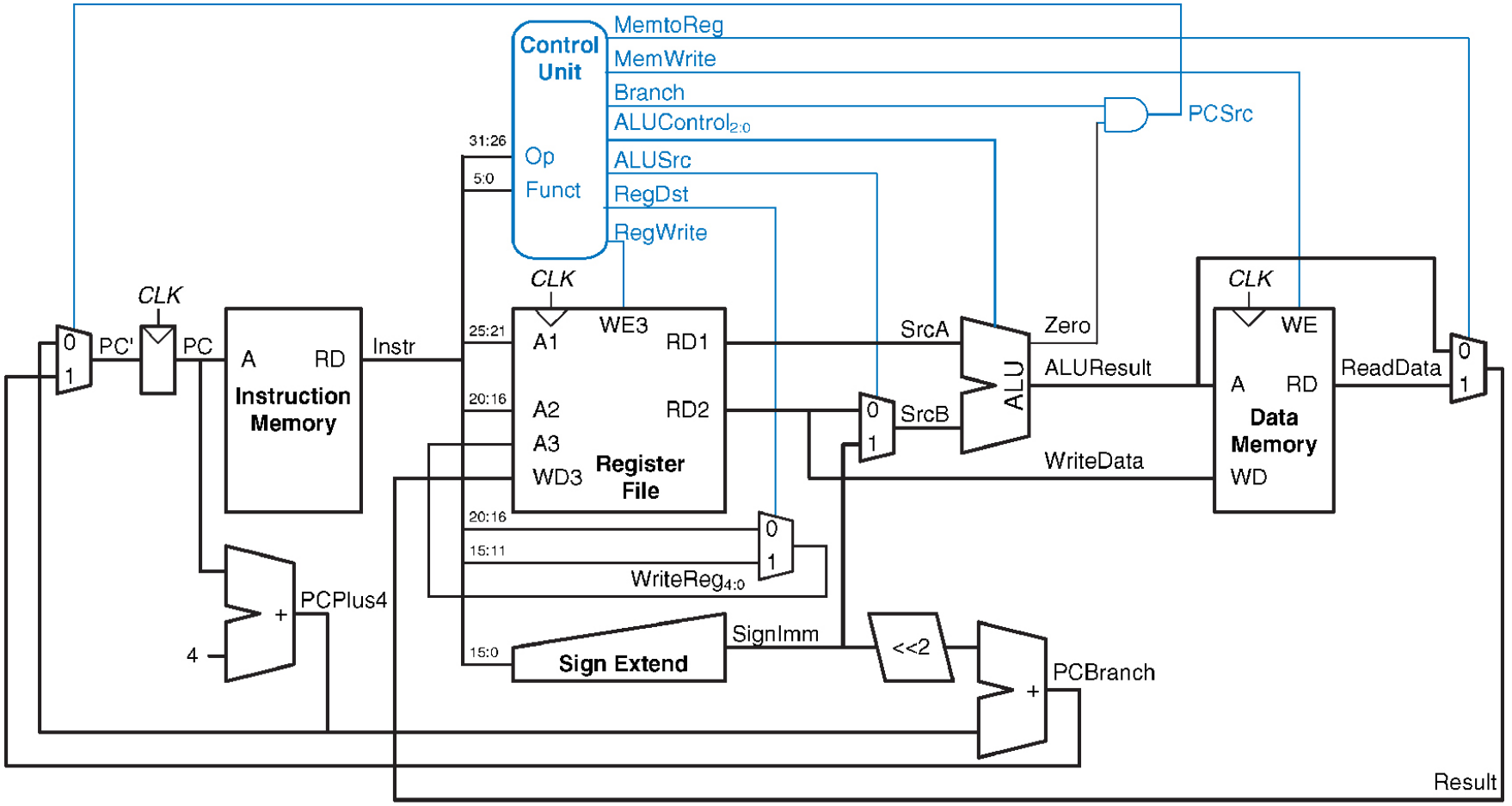
Technische Grundlagen der Informatik

Vorrechenübung 11.02.10

TECHNISCHE UNIVERSITÄT DARMSTADT

embedded systems + applications

Aufgabe 1: MIPS-Eintakt



Aufgabe 1: Erweiterung

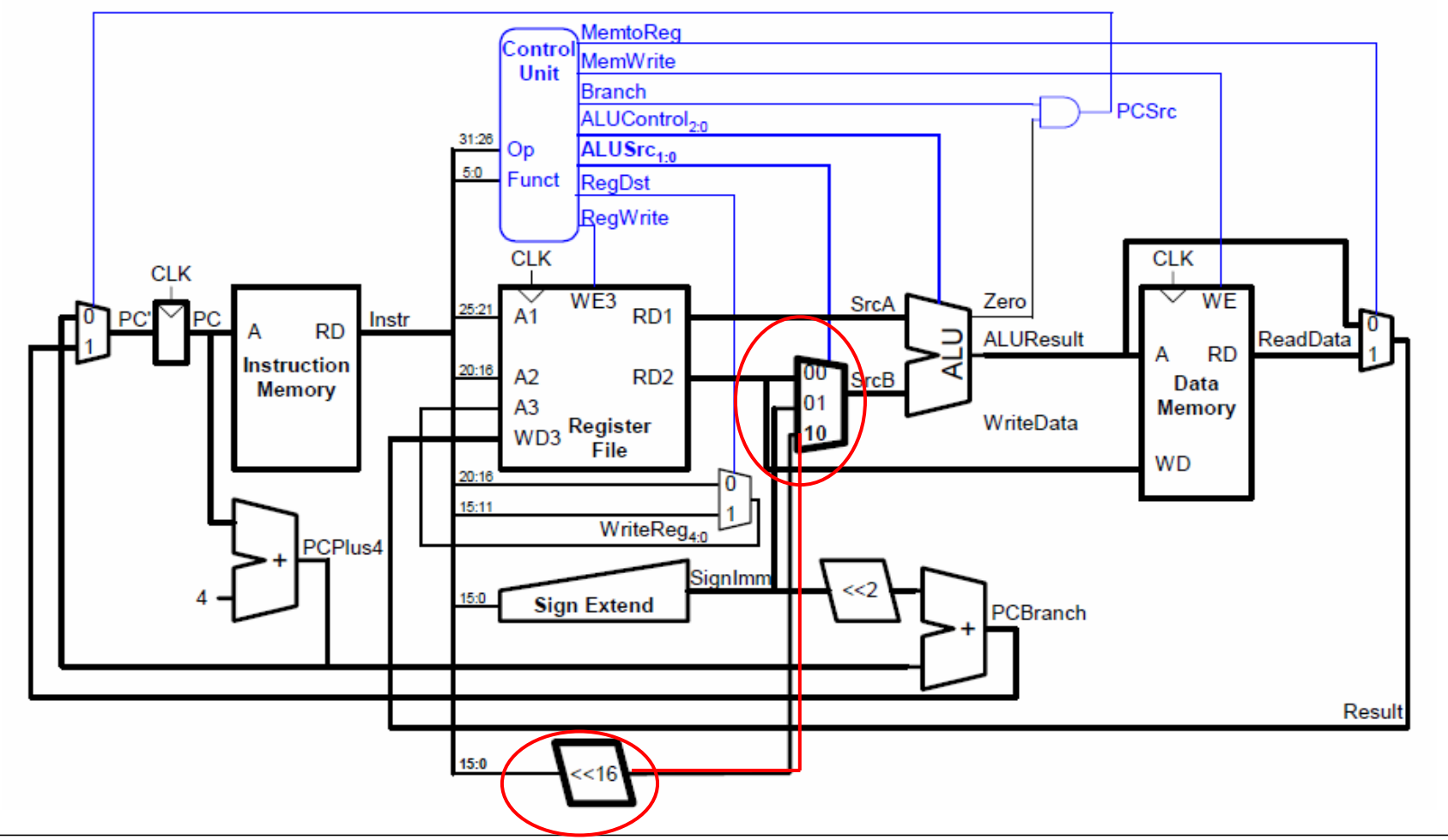
TECHNISCHE UNIVERSITÄT DARMSTADT

- Erweitern Sie den Prozessor, so dass er den Befehl lui verarbeiten kann.
- Erweitern Sie hierzu das Schaltbild und geben Sie die Belegung der Steuersignale an.
- Befehl lui (load upper immidiate): rt = imm*2^{16}

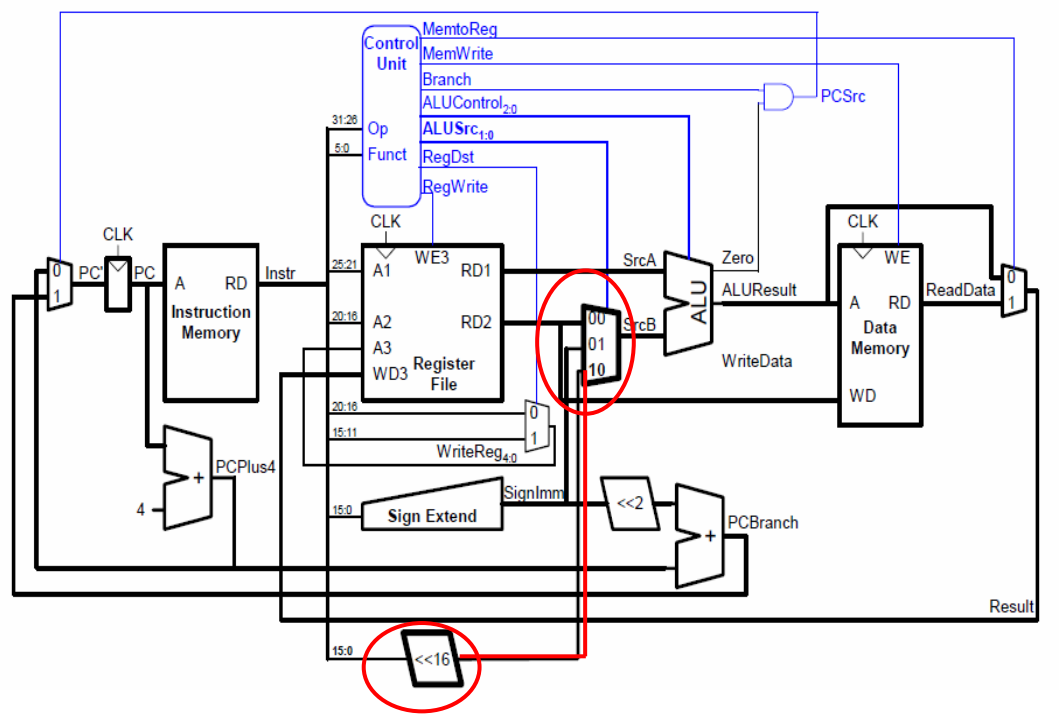
001111	0	rt	16 Bit imm

embedded systems + applications

Aufgabe 1: Datenpfad

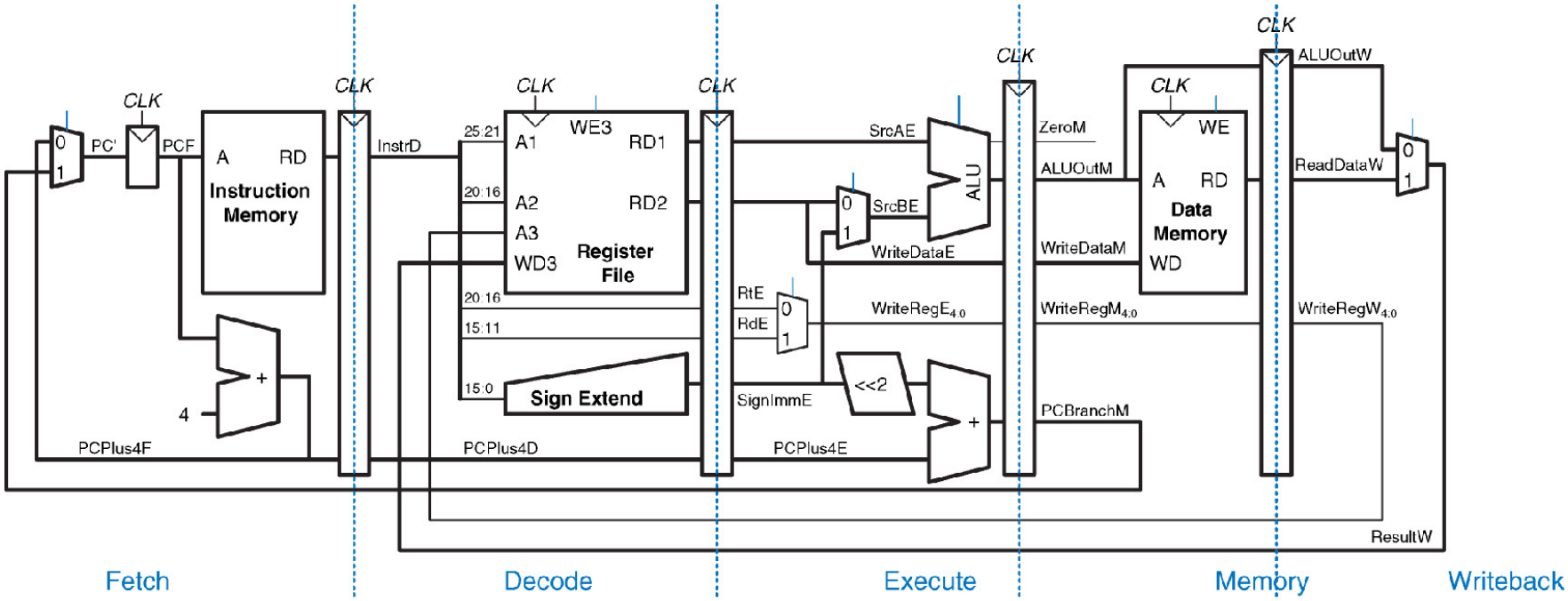


Aufgabe 1: Steuersignale



- MemtoReg = 0
- MemWrite = 0
- Branch = 0
- ALUControl = add
- ALUSrc = 10
- RegDst = 0
- RegWrite = 1

Aufgabe 2: Pipelining



Aufgabe 2: Pipelining

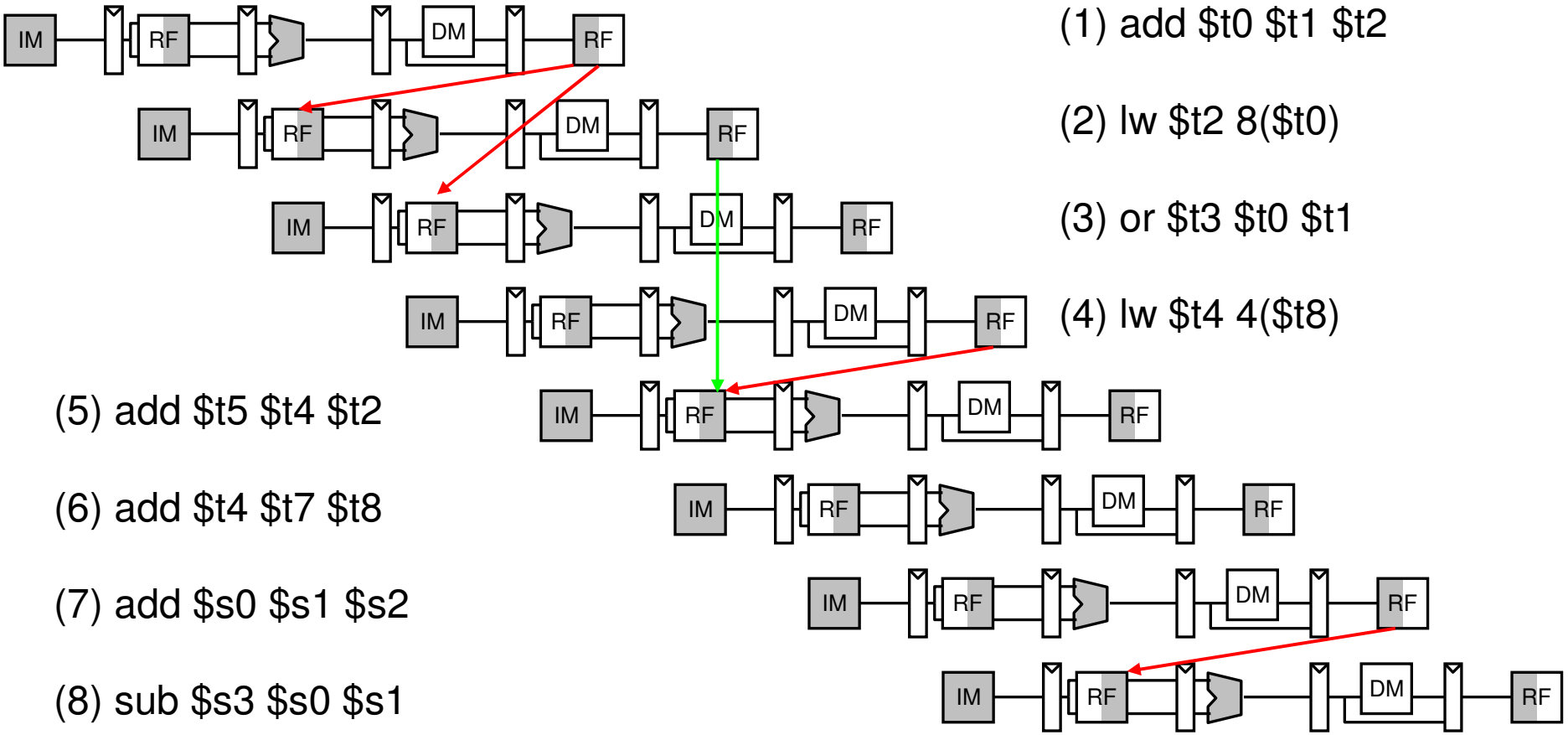
TECHNISCHE UNIVERSITÄT DARMSTADT

- Analysieren Sie die folgende MIPS-Befehlsfolge. Sie soll auf einem MIPS-Prozessor mit Pipelining ohne Forwarding ausgeführt werden. Markieren Sie im gegebenen Diagramm Stellen, an denen Probleme auftreten können und benennen Sie die Probleme.

- add $t0 $t1 $t2
- lw $t2 8($t0)
- or $t3 $t0 $t1
- lw $t4 4($t8)
- add $t5 $t4 $t2
- add $t4 $t7 $t8
- add $s0 $s1 $s2
- sub $s3 $s0 $s1

embedded systems + applications

Hazards - Datenabhängigkeiten

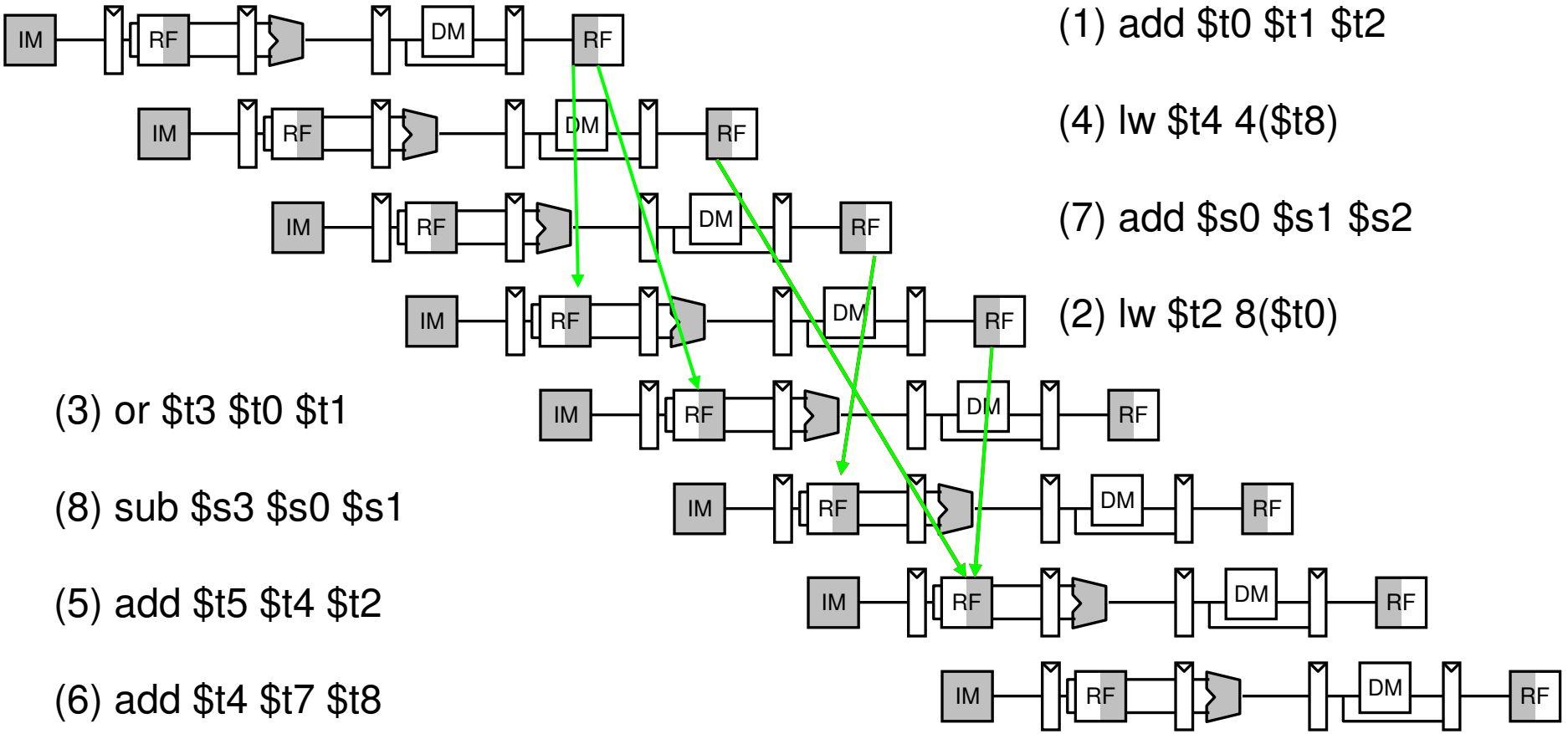


Hazards - Behebung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Umsortieren der Befehle (wenn möglich)
- NOPS einfügen

Hazards - Behebung



Verilog

TECHNISCHE UNIVERSITÄT DARMSTADT

- Schreiben Sie ein Modul, welches den gcd von zwei 16-Bit Zahlen berechnet
- Eingang: Ain, Bin, start
- Ausgang: result, done

```
module gcd (input clk,
            input start,
            input [15:0] Ain,Bin,
            output reg[15:0] result,
            output reg done);
```


embedded systems + applications

Berechnung des gcd

TECHNISCHE UNIVERSITÄT DARMSTADT

- Bei der Berechnung verfährt man nach Euklid wie folgt:

- 1. Ist a < b, so vertausche b und a
- 2. Berechne r = a - b
- 3. Setze a = b, b = r
- 4. Ist r != 0 weiter mit Schritt 1

- Nach Ablauf des Verfahrens hat man mit a den gcd(a; b) gefunden.

Implementierung in Verilog

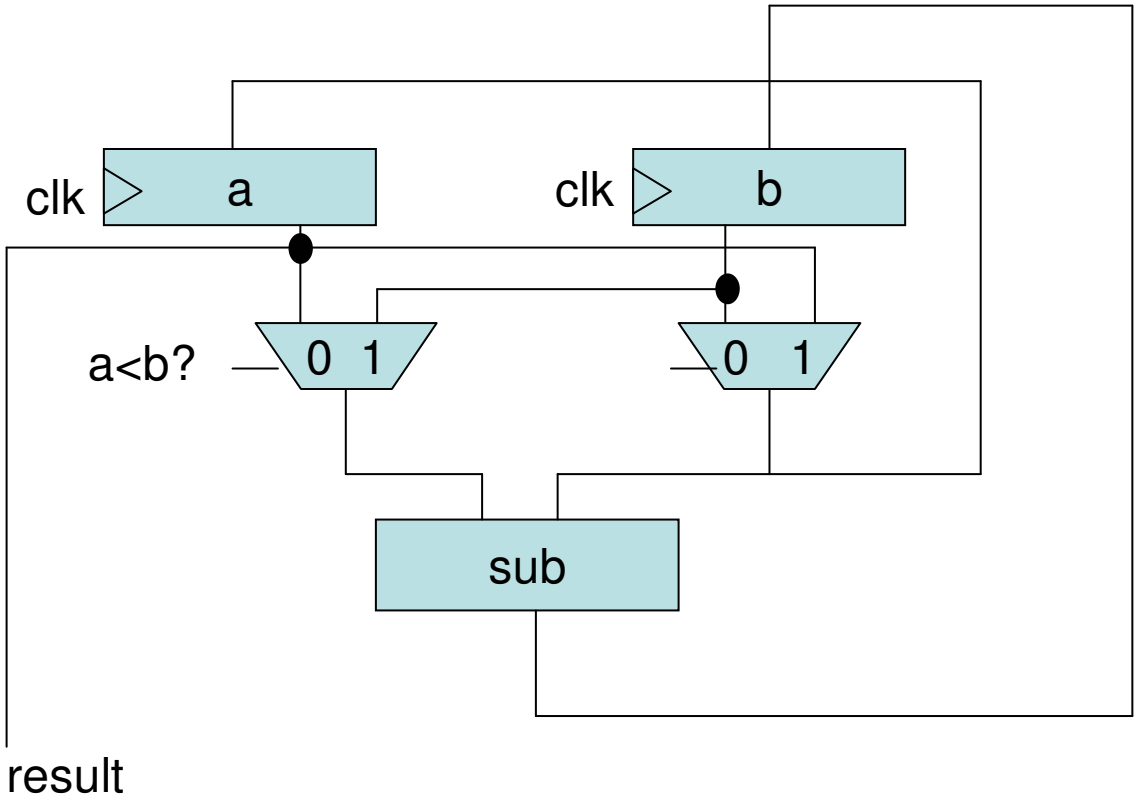
TECHNISCHE UNIVERSITÄT DARMSTADT

- Im Praktikum oft gesehen
- While (r !=0) ….

- Variable (bzw. unendliche) Anzahl an Schleifendurchläufen – eine solche Hardware gibt es nicht!!!!

- Sequentielle Implementierung: In jedem Takt wird ein Durchlauf abgearbeitet

Aufbau



Verilog-Code

TECHNISCHE UNIVERSITÄT DARMSTADT

```
reg [15:0] a, b, tmp_a, tmp_b;

//Getakteter Teil
always @ (posedge clk)
begin
if (start)  //Werte einlesen
begin
  a <= Ain; b <= Bin; done <= 0;
end
else
  if (b == 0)     //Abbruch?
  begin
    result <= a;
    done <= 1;
  end
  else            //Berechnung
    begin
      a<=tmp_b;
      b<=tmp_a-tmp_b;
    end
end
```

```
//ungetakteter Teil
//realisiert Vertauschen,
//wird als Multiplexer realisiert
always@(*)
begin
  if(a>b) begin
    tmp_a = a;
    tmp_b = b;
  end
  else
  begin
    tmp_a = b;
    tmp_b = a;
  end
end


endmodule
```


embedded systems + applications

Fragen?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Viel Erfolg bei der Klausur!

- Raumeinteilung:

S101/A01	Bsc INF 09 A-D
S101/A03	Bsc INF 09 E-G
S101/A04	Bsc INF 09 H-J
S311/0012	Bsc INF 09 K-Me
S206/030	Bsc INF 09 Mi - Sp
S202/C205	Bsc INF 09 St-Z
S101/A1	Bsc INF 07/04, Bsc Education, alle anderen