Erweiterung des Prozessors

Erweiterungen

OpCode	Name	Beschreibung	Operation
0011 (3)	jal label	jump and link	$ra = PC + 4, PC = JTA

- Jump zur Addresse wird bereits durch die Jump Operation abgebildet.
- Der Programcounter +4 muss noch in Register $ra (Register 31 gespeichert werden)
- ControlUnit MemToReg wird auf 2 Bits erweitert, um das Schreiben in WD3 zu steuern mit dem der PC + 4 angelegt werden kann.
- ControlUnit RegDst wird auf 2 Bits erweitert, um das Schreiben in A3 zu steuern mit dem 31 Bit ($ra) angelegt werden kann.

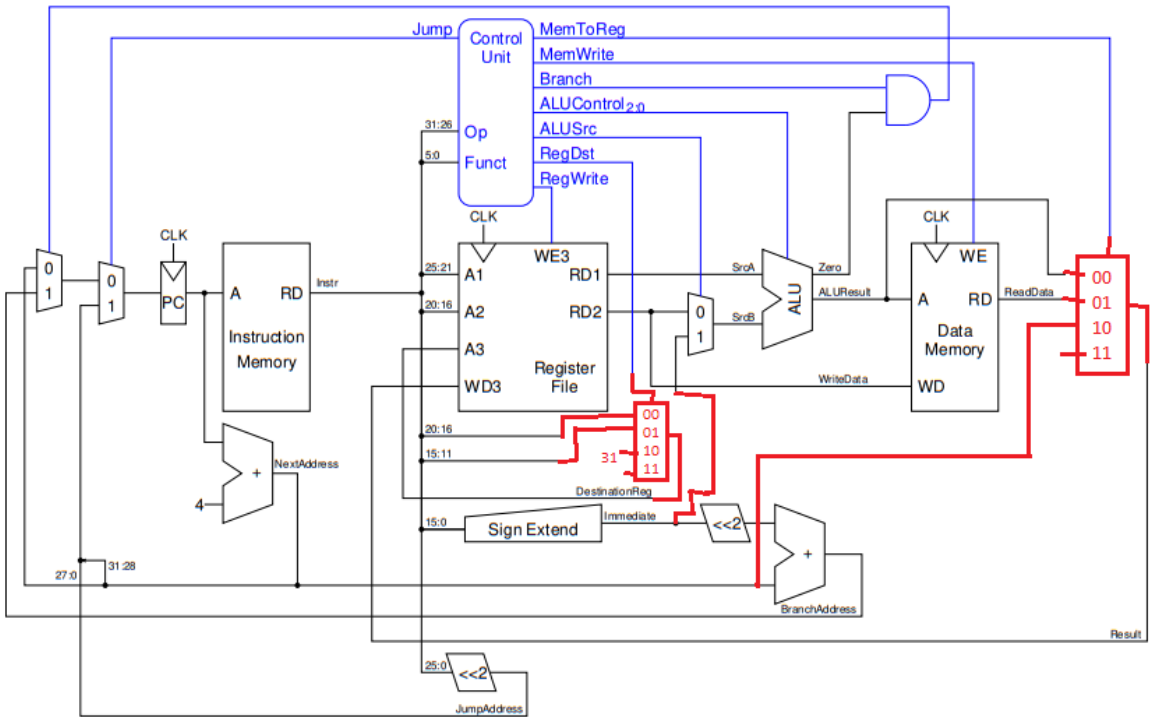

Abbildung 2: MIPS Prozessor

OpCode	Name	Beschreibung	Operation
001000 (8)	jr rs	jump register	PC = [rs]

- Jump zur Addresse wird bereits durch die Jump Operation abgebildet.
- Die Ziel Addresse befindet sich am Ausgang RD1 und muss in den PC geschrieben werden.
- ControlUnit Jump wird auf 2 Bits erweitert, um das Schreiben in PC zu steuern mit dem RD1 angelegt werden kann.

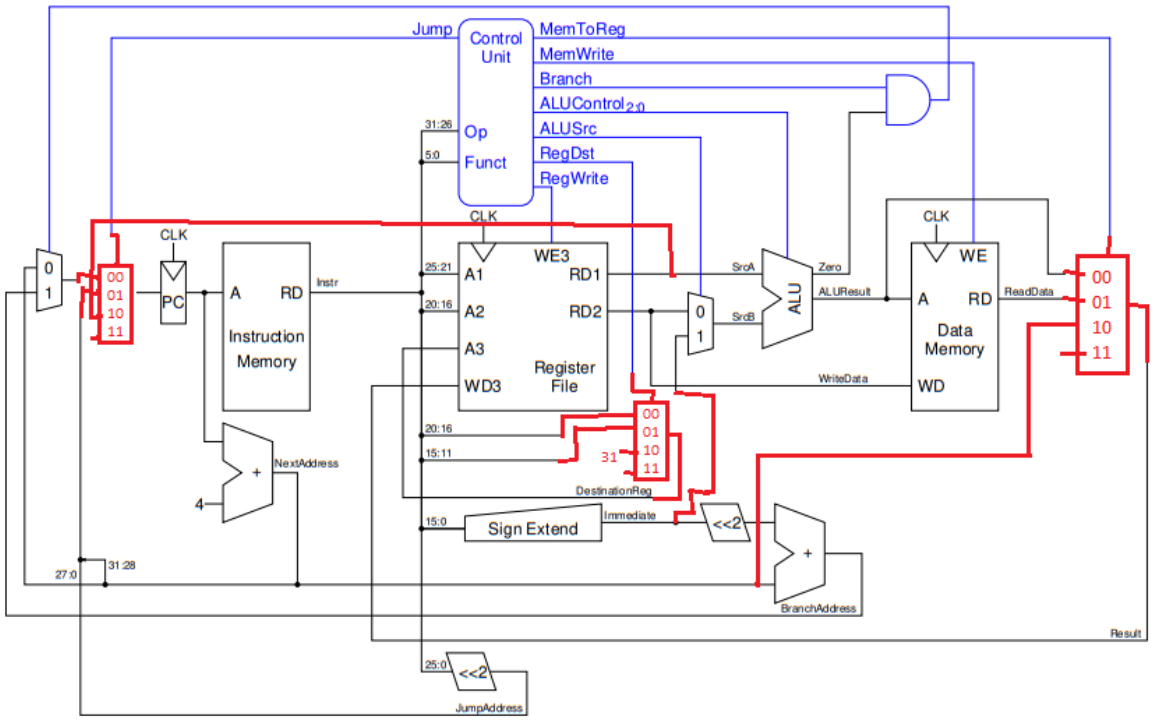

Abbildung 2: MIPS Prozessor

OpCode	Name	Beschreibung	Operation
001010 (10)	slti rt, rs, imm	set less than immediate	[rs] < SignImm ? [rt] = 1 : [rt] = 0

- ControlUnit ALUControl wird auf SET ON LESS THAN gesetzt.
- An der ALU SrcA liegt rs an. ALUSrc wird gesetzt, daher liegt an SrcB der Wert des Immediate an.
- ALU liefert bereits 1 oder 0, dies wird in das Ausgabe Register rt geschrieben. (MemToReg 00, RegWrite 1, RegDst 00)

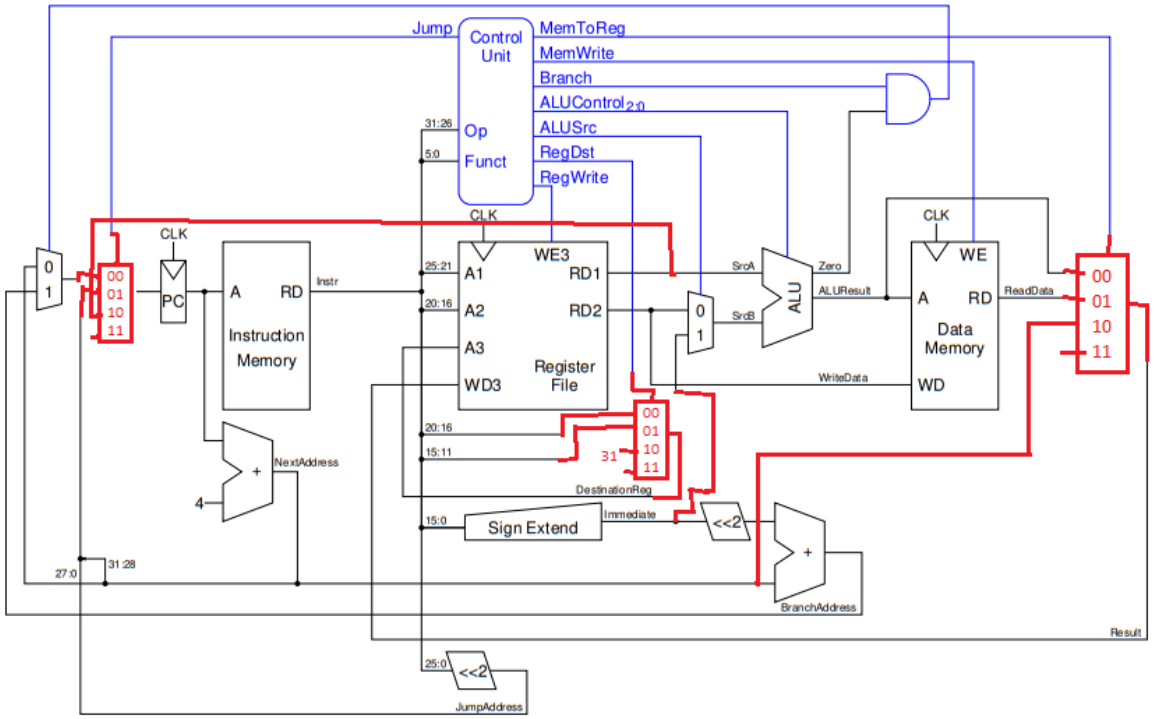

Abbildung 2: MIPS Prozessor

(Bild nicht verändert zum vorherigen)

OpCode	Name	Beschreibung	Operation
000101 (5)	bne rs, rt, label	branch if not equal	if ([rs] != [rt]) PC = BTA

- Branch ist schon implementiert durch be.
- Ersetzen des Logischen NAND durch einen Mux.
- ControlUnit Branch steuert ob nicht Zero oder Zero geschrieben wird.

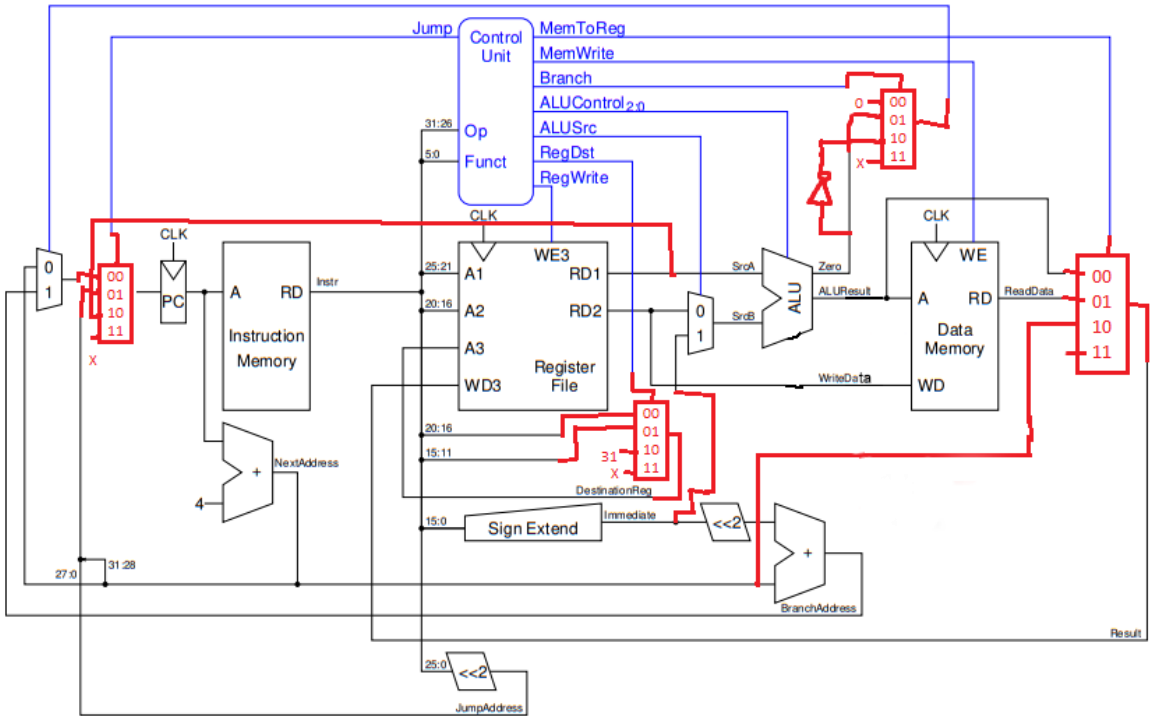

Abbildung 2: MIPS Prozessor

OpCode	Name	Beschreibung	Operation
100000 (32)	lb rt, imm(rs)	load byte	[rt] = SignExt ([Address]7:0)

OpCode	Name	Beschreibung	Operation
101000 (40)	sb rt, imm(rs)	store byte	[Address]7:0 = [rt]7:0

Hier werden mithilfe von Multiplexern nur ein einzelnes Byte zum schreiben bzw. lesen verwendet. Da die Memory Unit immer nur Wörter lädt, werden die zwei kleinsten Bits der Adresse als Kontrollsignal für den Multiplexer verwendet. (Grafik beinhaltet nur Load Byte, Store Byte funktioniert aber analog).

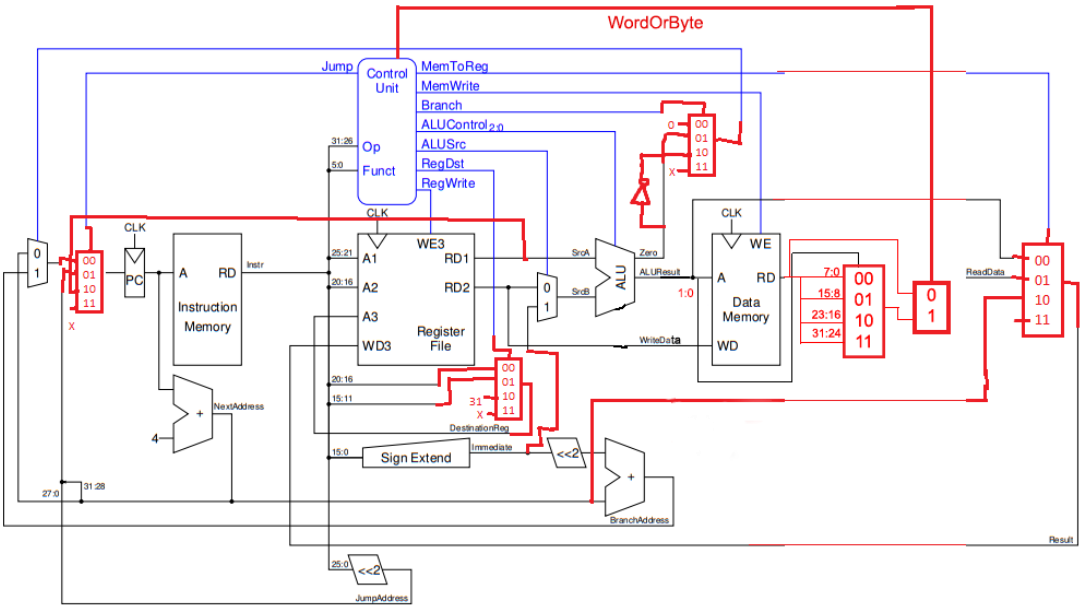

Abbildung 2: MIPS Prozessor

Finaler Prozessor

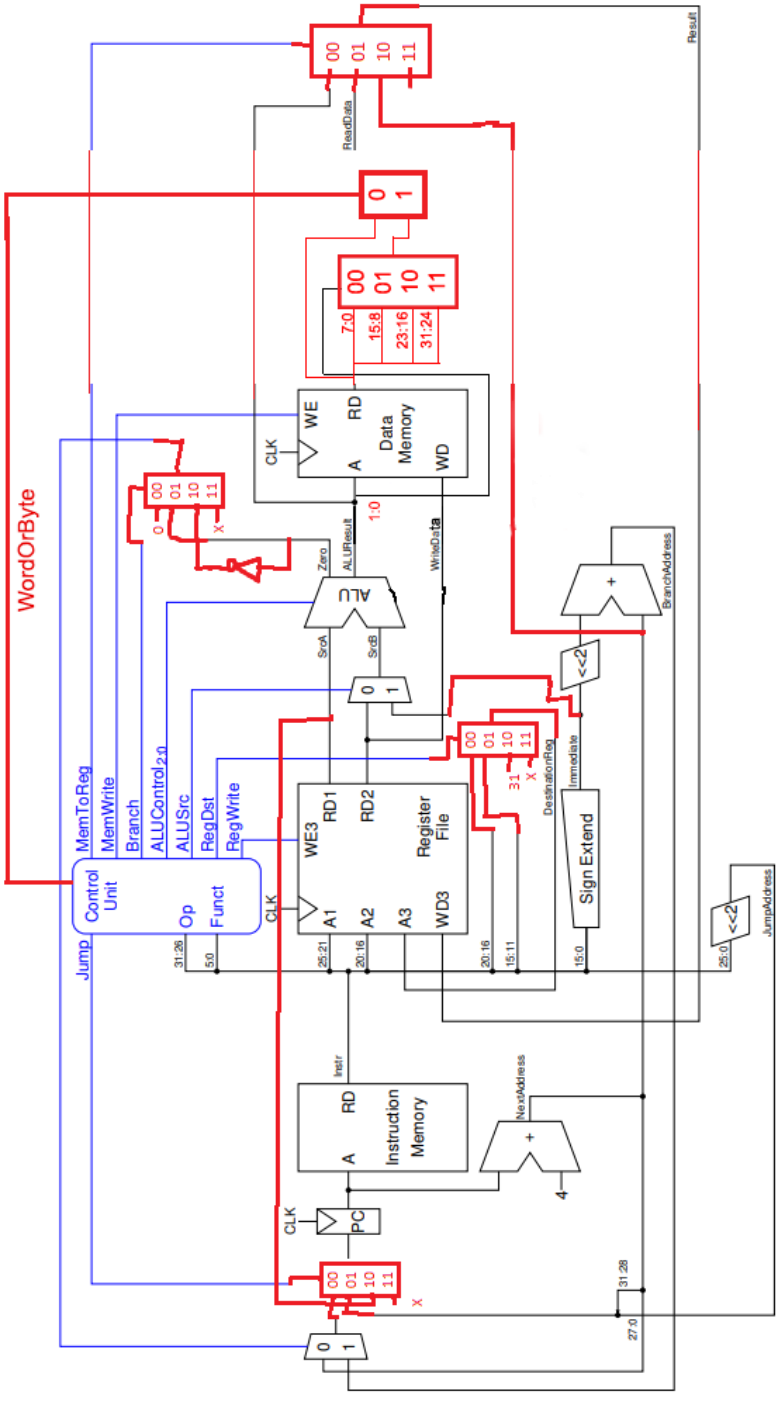

Abbildung 2: MIPS Prozessor

Hinzugefügte Multiplexer (4 to 1)

Kontrollsignal	**00**	**01**	**10**	**11**
MemToReg	ALUResult	ReadData	NextAddress	X
RegDst	Instr 20:16	Instr 15:11	31 ($ra)	X
Branch	0	ALU Zero	Not ALU Zero	X
Jump	Next Address (MUX links außen)	Jump Address	RD1	X
ALUResult 1:0	Byte 0	Byte 1	Byte 2	Byte 3
WordOrByte	Geladenes Wort	Geladenes Byte	---	---