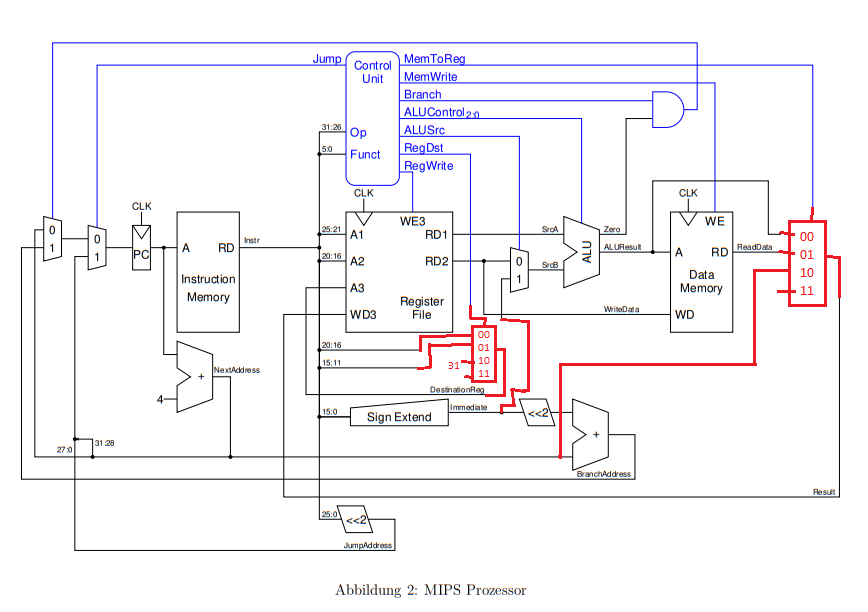
Erweiterung des Prozessors

# Erweiterungen

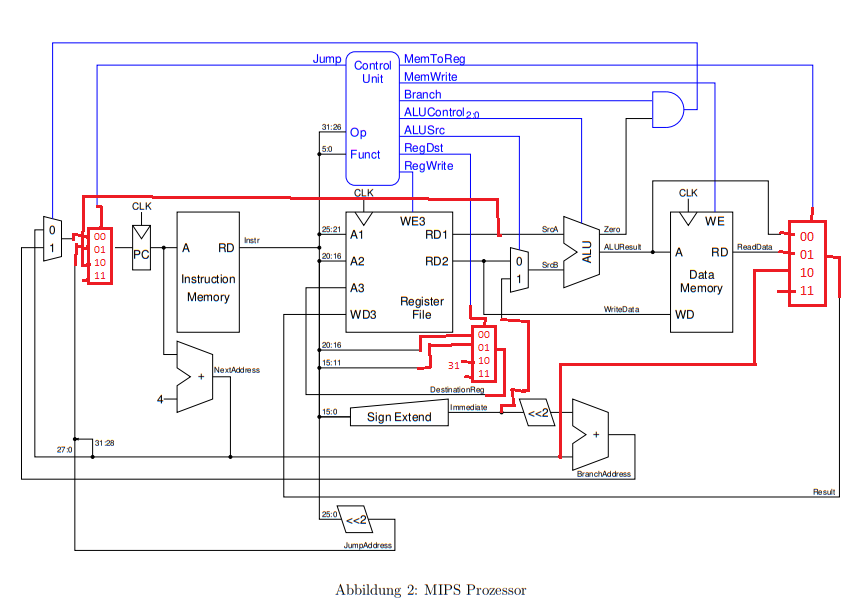
|  |  |  |  |
| --- | --- | --- | --- |
| **OpCode** | **Name** | **Beschreibung** | **Operation** |
| 0011 (3) | jal label | jump and link | $ra = PC + 4, PC = JTA |

* 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.



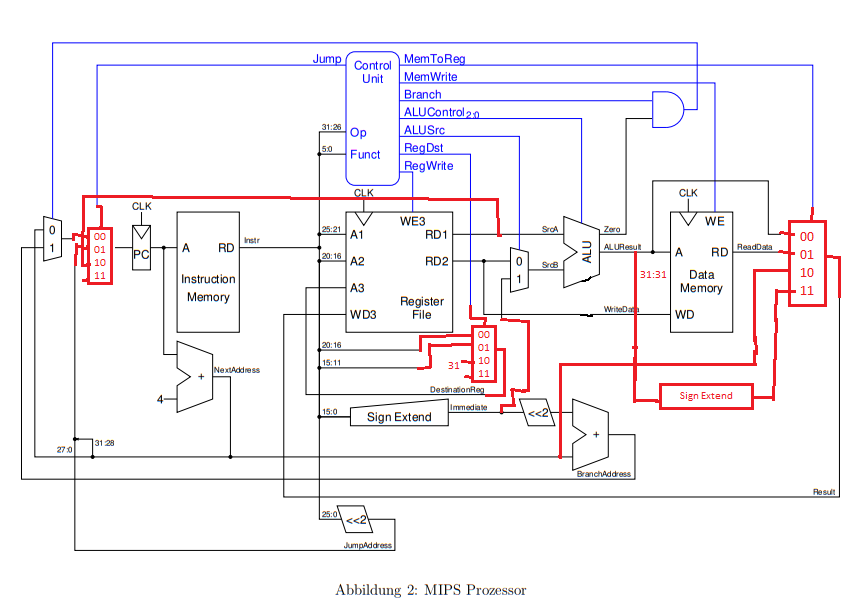
|  |  |  |  |
| --- | --- | --- | --- |
| **OpCode** | **Name** | **Beschreibung** | **Operation** |
| 001000 (8) | jr rs | jump register | PC = [rs] |

* 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.



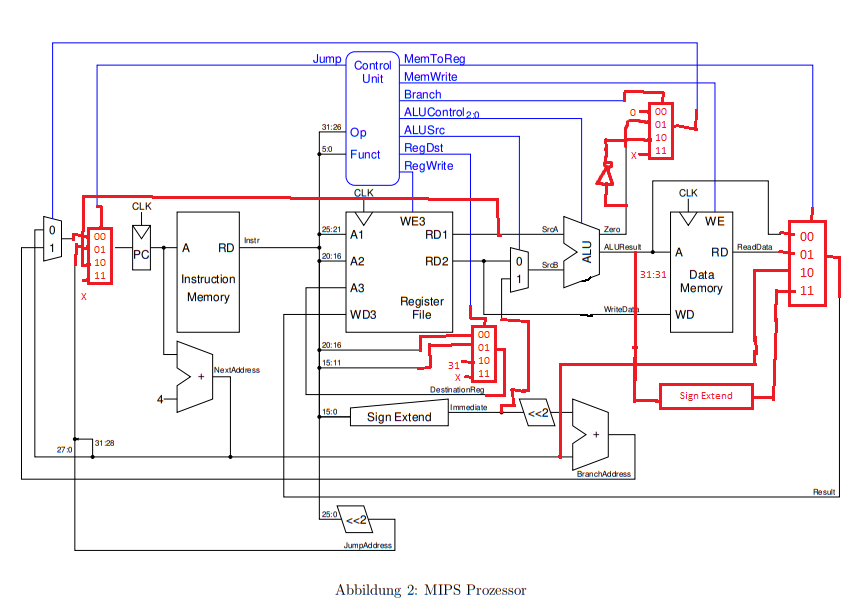
|  |  |  |  |
| --- | --- | --- | --- |
| **OpCode** | **Name** | **Beschreibung** | **Operation** |
| 001010 (10) | slti rt, rs, imm | set less than immediate | [rs] < SignImm ? [rt] = l : [rt] = 0 |

* ControlUnit ALUControl wird auf Subtrahieren gesetzt.
* Mittels SignExtend liegt an der ALU SrcA rs an und an SrcB SignImm.
* Falls das Ergebnis bei der ALU negativ ist, ist das höchste Bit 1, dieses soll an der Mux anliegen mit einer Erweiterung auf 32 bit.
* ControlUnit MemToReg steuert das Schreiben dieses Wertes in das Ziel rt.



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

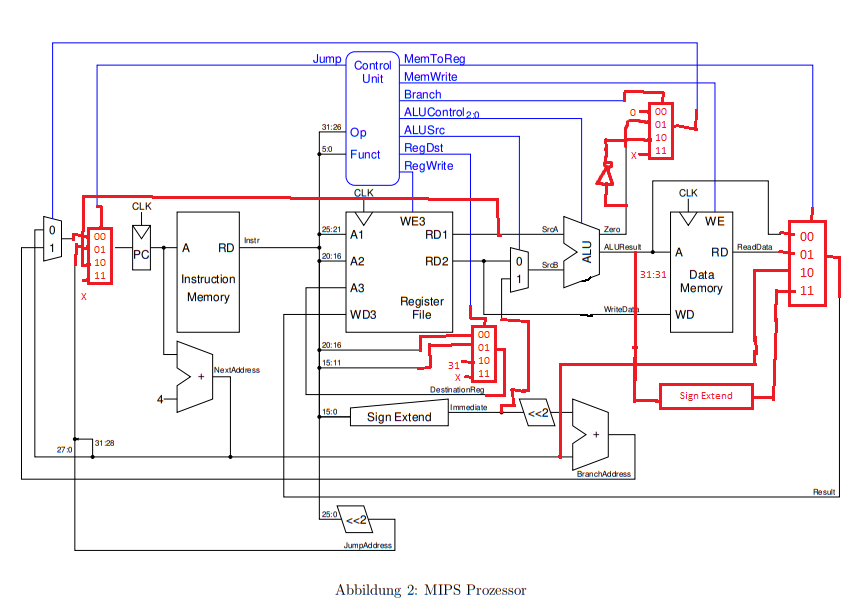
* Branch ist schon implementiert durch be.
* Ersetzen des Logischen NAND durch einen Mux.
* ControlUnit Branch steuert ob nicht Zero oder Zero geschrieben wird.



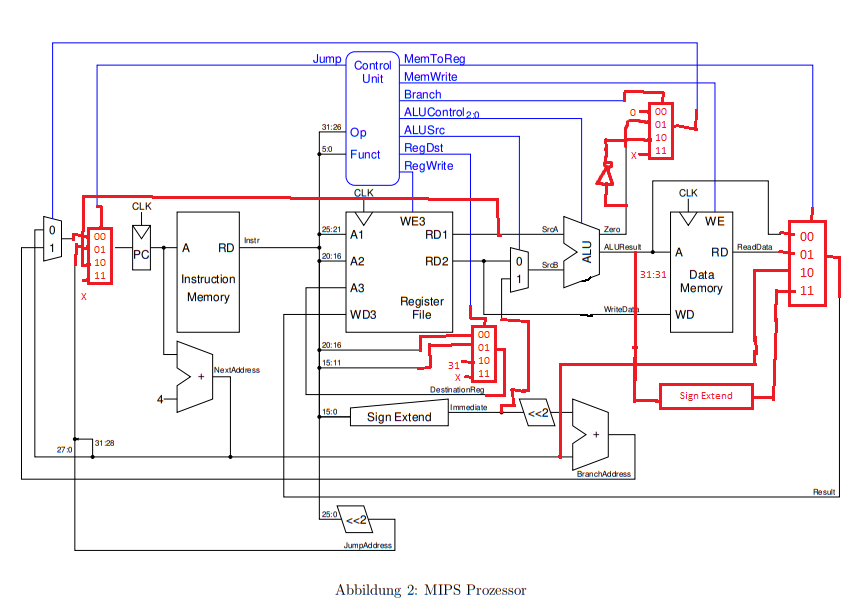
|  |  |  |  |
| --- | --- | --- | --- |
| **OpCode** | **Name** | **Beschreibung** | **Operation** |
| 100000 (32) | lb rt, imm(rs) | load byte | [rt] = SignExt ([Address]7:0) |

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

* Für lb und wb sollten keine Änderungen mehr gemacht werden müssen. Über die Write und ReadData Pfade zu der DataMemory sollte alles abgedeckt sein.



# Finaler Prozessor



# Kontrollsignale

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **00** | **01** | **10** | **11** |
| MemToReg | ALUResult | ReadData | NextAddress | 1 or 0 in 32 Bit |
| RegDst | Instr 20:16 | Instr 15:11 | 31 (ra) | X |
| Branch | 0 | ALU Zero | Not ALU Zero | X |
| Jump | MUX Links außen | Jump Address | RD1 | X |