| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Register** | R0, R1, R2, R3 | 16-Bit Register allgemein | | | | | R0 teilw. mit Sonderfuntion |
| | STAGE_CNT | 8 Bit Schleifenzähler | | | | | |
| **ALU – Instructions** | | | **EQ** | **OV** | OP-Cyc | Fetch-Cyc | |
| ADD | Rd, Rs1, Rs2 | Rd = Rs1 + Rs2 | x | x | 2 | 4 | |
| ADD | Rd, Rs1, Imm | Rd = Rs1 + u16 | x | x | 2 | 4 | |
| SUB | Rd, Rs1, Rs2 | Rd = Rs1 - Rs2 | x | x | 2 | 4 | |
| SUB | Rd, Rs1, Imm | Rd = Rs1 - u16 | x | x | 2 | 4 | |
| | | | | | | | |
| AND | Rd, Rs1, Rs2 | Rd = Rs1 & Rs2 | x | | 2 | 4 | |
| AND | Rd, Rs1, Imm | Rd = Rs1 & u16 | x | | 2 | 4 | |
| OR | Rd, Rs1, Rs2 | Rd = Rs1 \| Rs2 | x | | 2 | 4 | |
| OR | Rd, Rs1, Imm | Rd = Rs1 \| u16 | x | | 2 | 4 | |
| | | | | | | | |
| LSH | Rd, Rs1, Rs2 | Rd = Rs1 << (Rs2 & 0x000F) | x | | 2 | 4 | Schiftweite wird maskiert |
| LSH | Rd, Rs1, Imm | Rd = Rs1 << (u16 & 0x000F) | x | | 2 | 4 | dto |
| RSH | Rd, Rs1, Rs2 | Rd = Rs1 >> (Rs2 & 0x000F) | x | | 2 | 4 | dto |
| RSH | Rd, Rs1, Imm | Rd = Rs1 >> (u16 & 0x000F) | x | | 2 | 4 | dto |
| | | | | | | | |
| MOVE | Rd, Rs | Rd = Rs | x | | 2 | 4 | |
| MOVE | Rd, Imm | Rd = u16 | x | | 2 | 4 | |
| | | | | | | | |
| **Memory Data (32Bit)** | | | | | | | |
| ST | Rs, Rd, Imm | Mem[Rd + offset] = Rs | | | 4 | 4 | Offset in Bytes [0 .. 0x7FF) |
| LD | Rd, Rs, Imm | Rd = Mem[Rs + offset] | | | 4 | 4 | |
| | | | | | | | |
| **Sprung-Befehle zu absoluter Addr** | | | | | | | |
| JUMP | Rd | Jump → Addr(Rd) | | | 2 | 2 | |
| JUMP | label | Jump → Addr(label) | | | 2 | 2 | |
| JUMP | Rd, flag | if (flag) Jump → Addr(Rd) | * | * | 2 | 2 | |
| JUMP | label, flag | if (flag) Jump → Addr(label) | * | * | 2 | 2 | |
| | | | | | | | |
| **Sprung-Befehle zu relativer Addr (I7 => +/- 64 Befehle)** | | | | | | | |
| JUMPR | label, Imm, GE | if (R0>=u16) Jump label | | | 2 | 2 | Sprungdistanz zu Label |
| JUMPR | label, Imm, LT | if (R0 < u16) Jump label | | | 2 | 2 | errechnet Assembler |
| | | | | | | | |
| JUMPS | label, I8, EQ | if (scnt==I8) Jump label | | | 4 | 4 | |
| JUMPS | label, I8, LT | if (scnt<I8) Jump label | | | 2 | 2 | |
| JUMPS | label, I8, LE | if (scnt<=I8) Jump label | | | 2 | 2 | |
| JUMPS | label, I8, GT | if (scnt>I8) Jump label | | | 4 | 4 | |
| JUMPS | label, I8, GE | if (scnt>=I8) Jump label | | | 2 | 2 | |
| | | | | | | | |
| **Stage-Counter** | | | | | | | |
| STAGE_RST | | scnt = 0 | | | 2 | 4 | |
| STAGE_INC | Imm | scnt = scnt + I8 | | | 2 | 4 | |
| STAGE_DEC | Imm | scnt = scnt - i8 | | | 2 | 4 | |
| | | | | | | | |
| **Ablaufsteuerung** | | | | | | | |
| HALT | | Halt | | | 2 | | Starts ULP Wakeup-Timer if enabled |
| WAKE | | Wake SoC or Interrupt | | | 2 | 4 | |
| SLEEP | [0..4] | Select Sleep-Register | | | 2 | 4 | |
| NOP | | No Operation (750ns) | | | 2 | 4 | |
| WAIT | Imm | Wait (u16 * 125ns) | | | 2 | 4 | |