**Architecture Manual: RISC-V 5-Stage Pipelined CPU and Verification**

**CPU Pipeline Stages**

1. **IF – Instruction Fetch**

During the IF stage, the following occurs:

* + Branch logic selects an instruction address and the instruction cache fetch begins.

1. **ID – Instruction Decode**

During the ID stage, the following occurs:

* + The instruction decoder decodes the instruction and checks for interlock conditions.
  + Any required operands are fetched from the register file.

1. **EX – Execute**

During the EX stage, one of the following occurs:

* + The arithmetic logic unit (ALU) performs the arithmetic or logical operation for register-to-register instructions.
  + The ALU calculates the data address for load and store instructions.
  + The ALU determines whether the branch condition is true and calculates the branch target address for branch instructions.

1. **MEM – Memory Access**

During the MEM stage, one of the following occurs:

* + The data memory (DMEM) is read from or written to for load and store instructions.
  + No operation is performed for register-to-register instructions.

1. **WB – Write Back**

For register-to-register instructions, the instruction result is written back to the register file during the WB stage. Branch instructions perform no operation during this stage.

**Pipeline Operation**

The operation of the pipeline is illustrated by the following examples that describe how typical instructions are executed. The instructions described are: ADD, JALR, BEQ, ECALL, LW, and SW. Each instruction is taken through the pipeline and the operations that occur in each relevant stage are described.

Add Instruction

ADD rd,rs1,rs2

|  |  |
| --- | --- |
| **Stage** | **Description** |
| **IF** | The program counter value is used to read the instruction word from instruction memory (IMEM). |
| **ID** | The instruction bits are decoded. The *rs1* and *rs2* fields of the 2-port register file are accessed and the register data is valid at the register file output. At the same time, bypass multiplexers select inputs from either the EX- or MEM-stage output in addition to the register file output, depending on the need for an operand bypass. |
| **EX** | The ALU controls are set to do an A+B operation. The operands flow into the ALU inputs, and the ALU operation is started. The result of the ALU operation is latched into the ALU output latch. |
| **MEM** | This stage is a NOP for this instruction. The data from the output of the EX stage (the ALU) is moved into the output latch of the MEM stage. |
| **WB** | The WB latch feeds the data to the inputs of the register file, which is addressed by the *rd* field. The file write flag is enabled. The data is then written into the register file. |

Jump and Link Register Instruction

JALR rd,rs1,imm

|  |  |
| --- | --- |
| **Stage** | **Description** |
| **IF** | Same as the IF stage for the ADD instruction. |
| **ID** | The instruction bits are decoded. The register addressed by the *rs1* field is read out of the register file. The 12-bit immediate is extracted and sign-extended. |
| **EX** | Refer to the EX stage for the ADD instruction. For JALR, the inputs to the ALU come from *GPR[rs1]* through the bypass multiplexer and the sign-extended immediate field. The computed result of the ALU operation represents the PC-relative branch target of the JALR instruction. The least significant bit of the result is set to zero. The result is clocked into the virtual PC latch.  The value of the PC incremented in the IF stage is the PC to which the program will eventually return from the jump destination. This value is placed in the Link output latch of the Instruction Address unit. |
| **MEM** | The PC+4 value is moved from the Link output latch to the output latch of the MEM pipeline stage. |
| **WB** | Same as the WB stage for the ADD instruction. |

Branch if Equal Instruction

BEQ rs1,rs2,imm

|  |  |
| --- | --- |
| **Stage** | **Description** |
| **IF** | Same as the IF stage for the ADD instruction. |
| **ID** | The instruction bits are decoded. The register file is addressed with the *rs1* and *rs2* fields and the contents of these registers are placed in the register file output latch. The 12-bit immediate is extracted and sign-extended. |
| **EX** | A check is performed to determine if each corresponding bit position of these two operands has equal values. If they are equal, the PC is set to PC+*target*, where *target* is the sign-extended offset field. If they are not equal, the PC is set to *PC+4*. |
| **MEM** | This stage is a NOP for this instruction. |
| **WB** | This stage is a NOP for this instruction. |

Environment Call Instruction

ECALL

|  |  |
| --- | --- |
| **Stage** | **Description** |
| **IF** | Same as the IF stage for the ADD instruction. |
| **ID** | The instruction bits are decoded. A Trap Exception is generated. The |
| **EX** | No arithmetic or address calculation is performed. The exception cause code is generated here. |
| **MEM** | The *PC* register is loaded with the value of the exception vector and the instructions following in the previous pipeline stages are killed. |
| **WB** | The exception code is set in the ExCode field in the *mcause* register. The PC value of this instruction is stored in the *mepc* register and *mtval* bits are updated appropriately according to the contents of the *EXL* bit of the *mstatus* register. |

Load Word Instruction

LW rd,rs1,imm

|  |  |
| --- | --- |
| **Stage** | **Description** |
| **IF** | Same as the IF stage for the ADD instruction. |
| **ID** | Same as the ID stage for the ADD instruction. |
| **EX** | Refer to the EX stage for the ADD instruction. For LW, the inputs to the ALU come from *GPR[rs1]* through the bypass multiplexer and from the sign-extended offset field. The result of the ALU operation that is latched into the ALU output latch represents the effective address of the operand. |
| **MEM** | The data memory is accessed, the data is stored and the read data is placed in the MEM output latch. |
| **WB** | This stage is a NOP for this instruction. |

Store Word Instruction

SW rs1,rs2,imm

|  |  |
| --- | --- |
| **Stage** | **Description** |
| **IF** | Same as the IF stage for the ADD instruction. |
| **ID** | Same as the ID stage for the ADD instruction. |
| **EX** | Refer to the LW instruction for a calculation of the effective address. From the ID output latch the *GPR[rs1]* is sent through the bypass multiplexer. The results of the ALU operations are latched into the output latches. |
| **MEM** | Same as the MEM stage for the LW instruction. |
| **WB** | The read data is written into the register file addressed by the *rd* field. |