Each instruction influences the data path in a unique way. Our processor by default sets all control signals to 0 (except for PCen, and enRAM/enROM), and therefore, as the control signals are explained, only signals that will need to change state from 0 will be listed for each instruction.

addi:

addi is an I-Type instruction with an 18-bit, signed immediate value. Because the immediate value is interpreted as a two’s complement number, it can be used to add both positive and negative integer values. When the instruction is decoded, the Operation Code that goes to the Logic Controller is 4'b0101. The destination register is decoded from bits [27:23] of the instruction, and the source register, Rs, is decoded from bits [23:18] of the instruction. The immediate value is decoded from bits [17:0], but is fed into a sign extender (to preserve the two’s complement sign of the number) to be used in computation as a 32-bit number. From the Logic Controller, aluSrcb is set to a 0, which allows the immediate value to pass through the multiplexer determining the value of the second argument into the ALU. The Logic Controller also sets aluop to 3’b000 to perform addition. Thus the ALU utilizes the data stored in Rs and the immediate value to perform the add computation. The result of the ALU is passed through a mux controlled by memSrc, which is set to 1 to allow the ALU data to pass through. The data then passes through both the wbSrc and wbPSR muxes till it gets to the RaWriteMUX, where it also passes through. The Logic Controller sets regWriteEn high at the beginning of the instruction, and thus computation can be stored into the destination register.

*Control Signals:*

CFwrite = 1;

memSrc = 1;

wbSrc = 1;

regWriteEn = 1;

sub:

sub is an R-Type instruction. Because this is an R-Type instruction, the two arguments are registers that contain data to be subtracted. The Operation Code that goes to the Logic Controller is 4'b0000. The associated decoded Function Code comes from bits [3:0] of the instruction and are 4'b1001 for subtraction. The resulting control signals from the Logic Controller set the ALU to perform subtraction on two signed numbers. Three operands are considered for this instruction: Rdest, bits [27:23]; Rs, bits [22:18]; and Rt, bits [17:13].

*Control Signals:*

CFWrite = 1'b1;

wbSrc = 1'b1;

memSrc = 1'b1;

aluSrcb = 1'b1;

regWriteEn = 1'b1;

aluop = 3'b001;

multi:

multi is an I-Type instruction.

*Control Signals:*

wbSrc <= 1'b1;

memSrc <= 1'b1;

regWriteEn <= 1'b1;

aluop <= 3'b110;

cmpi :

cmpi is an I-Type instruction.

*Control Signals:*

LZNWrite <= 1'b1;

wbSrc <= 1'b1;

memSrc <= 1'b1;

aluop <= 3'b111;

andi :

andi is an I-Type instruction.

*Control Signals:*

wbSrc <= 1'b1;

memSrc <= 1'b1;

regWriteEn <= 1'b1;

aluop <= 3'b011;

ori :

ori is an I-Type instruction.

*Control Signals:*

wbSrc <= 1'b1;

memSrc <= 1'b1;

regWriteEn <= 1'b1;

aluop <= 3'b010;

xori:

xori is an I-Type instruction.

*Control Signals:*

wbSrc <= 1'b1;

memSrc <= 1'b1;

regWriteEn <= 1'b1;

aluop <= 3'b100;