1.

(a)ld and sd

(b)add and sub

(c)add

2.

(a)

Branch:

000100 00001 00000 0000 0000 0000 0011

Op rs rt address

MemRead:

MemRead and Memwrite should be set to 1 if the data memory is to read or write resp, otherwise 0.

ALUOp:

R-type of instructions must access or perform registers and ALU

| ALUOp | func |
|---|---|
| 000 | and |
| 001 | or |
| 010 | add |
| 110 | sub |
| 111 | s/t |

MemWrite:

Data memory contents designated by address input are present at the write data input.

ALUSrc:

The second ALU op is taken from the second register file output. The second ALU op is the sign extended, lower 16 bits of the instruction.

RegWrite:

The value present at the write data input is output from the ALU and the value is register writedata input is taken from data memory

MemtoReg:

32bit values and specifier is 5 bits long and we can read from two registers at a time.

(b)

| ALU Control Input | func |
|---|---|
| 000 | and |
| 001 | or |
| 010 | add |
| 110 | sub |

| | |
|---|---|
| 111 | s/t |

| ALUop Input | operation |
|---|---|
| 00 | load/store |
| 01 | beq |
| 10 | determined by opcode |

3.

(a)

Latency of an add instruction =I-Mem/D-Mem + Register File + Mux + ALU +Adder + PC read + Register set up + sign extension = 230+130 + 20+180 + 130+ 20+10+30 = 750ps

(b)

Latency of an ld instruction =I-Mem/D-Mem + Register File +Adder + PC read + Register set up = 230+130 +130+ 20+10 = 520ps

(c)

Latency of an sd instruction =I-Mem/D-Mem + Register File +Adder + PC read + Register set up = 230+130 +130+ 20+10 = 520 ps

4

(a)

No additional logic blocks or wires are needed.

Sending the code for slt rd, rs1, rs2 causes the ALU to set the operation code register to 0110 for add-with-carry. The carry-bit is thus set unbeknownst to the programmer and will be ignored. The processor then executes an add instruction, using A as a counter operand; adding "1", which is represented in binary by a "0", produces no carry because A is a one less than B (1000011 versus 1000000). Normally, this would cause an overflow condition in add-with-carry, but here it has no effect since C is not used.

(b)

Yes. The C-unit needs an additional "ct" input. This input should come from the same line through which rs2 is read in cycle 1, i.e., the rs1 line.

But what if a third instruction is added, say setg? To handle this case, the "Control" unit will have to have three inputs: ct, rs1, and rs2.

(c)

Yes. The reason is that the code sequence "rs1 = slt rd, rs1, rs2" does not make any

sense for a processor implemented using just one register and two-cycle ALU. What one expects to see is a different code sequence: "r0 = slt rd, r0, r2", where "r0" is the result of an earlier slt operation on some other register. However there is no such code sequence within ALU can support such instruction. The ALU always sends out its result to the next stage of execution.

5.

(a)

Clock period of pipelined processor: maximum latency among all the latencies =450ps

Clock period of single cycle processor: sum of latencies of all the stages=300+400+450+350+250=1750ps

(b)

Sd instruction in pipelined processor will take upto MEM stages. Therefore latency of sd instruction in pipelined processor=4*clock period=4*450=1800ps

And latency of sd instruction in single cycle processor =latency of one clock cycle=1750ps

(c)

In order to get minimum clock cycle period, we need to split the stage with maximum latency, therefore we need to split EX stage. Split into to halves, EX1=225ps and EX2=225ps

Therefore, new clock period=maximum latency among all the latencies=400ps

6.

Clock rate:0.5*10^9 cycles/sec

Cycle time:2ns

(a)

Num of instructions: S

Num of stages in pipeline:4

Total num of cycles required: 4+(S-1)*1=S+3

Total execution time: (total num of cycles)*(cycle time) = (S+3)*2ns

Given: (S+3)*2=50 =>(S+3)=25 => S=22

Num of cycles required to execute 4S=2*22=88 instructions: 4+(88-1)*1=91cycles

Total execution time: (total num of cycles)*(cycle time)=91*2= 182ns

(b)

Total num of cycles required for S instructions in N stage pipeline: N+(S-1)*1=N+S-1

Total execution time: (total num of cycles)*(cycle time)=(N+S-1)*2ns

Given: (N+S-1)*2=100 ------(1)

Also

Total num of cycles required for 4S instructions in N stage pipeline: N+(4S-1)*1=N+4S-1

Total execution time: (total num of cycles)*(cycle time)= (N+4S-1)*2ns

Given: (N+4S-1)*2 = 340-------(2)

From(1)and(2)

We get S=40 and N=11

8.

(a)

The accuracy of the always-taken predictor for this sequence of branch outcomes is 100%. The accuracy of the always-not-taken predictor for this sequence of branch outcomes is 0%.

(b)

The accuracy of the 1-bit dynamic predictor for this sequence of branch outcomes is 50%.

(c)

The accuracy of the 2-bit dynamic predictor for this sequence of branch outcomes is 75%.

9.

IF: First instruction of the exception handler

ID: NOP

EX: NOP

MEM: sub x6, x31, x28

EX: add x10, x28, x29