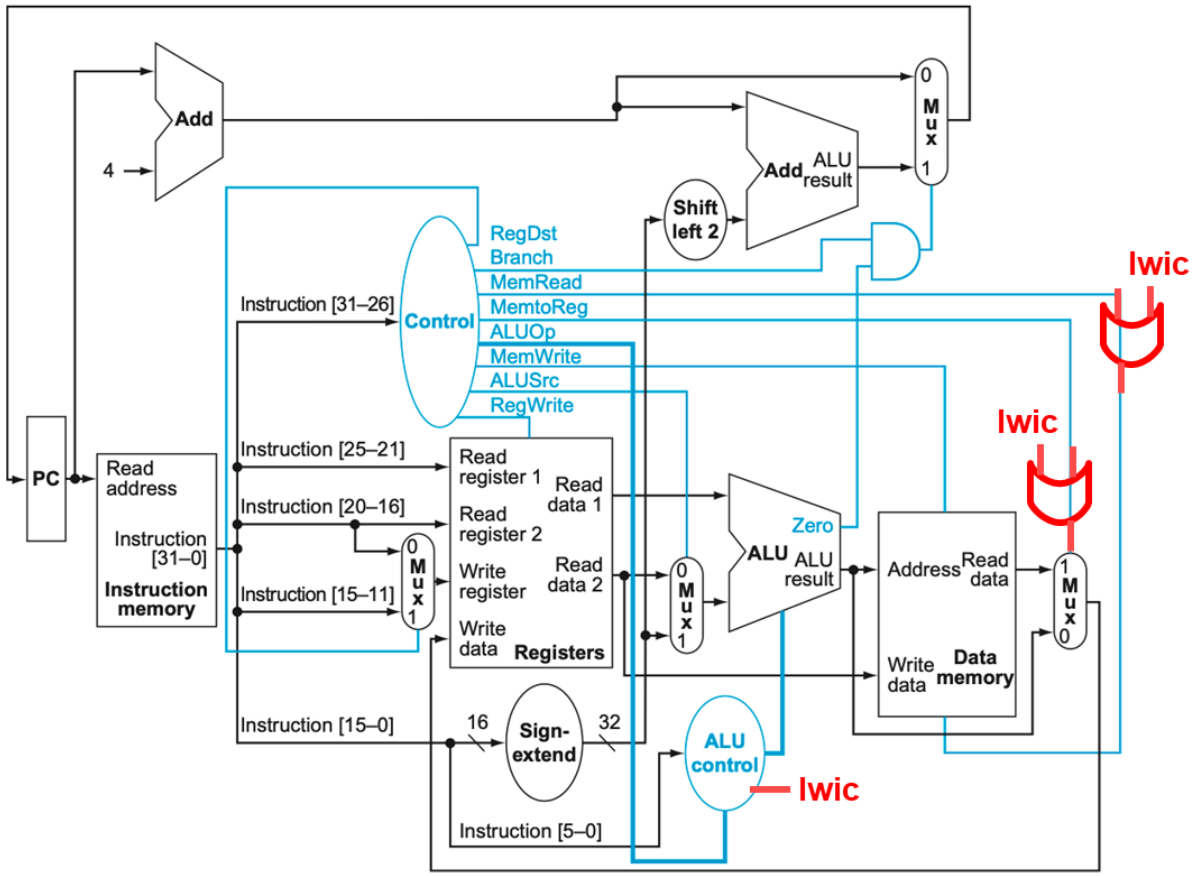
4.10

The additional registers will allow us to remove 12% of the loads and stores, or (0.12)*(0.25 + 0.1) = 4.2% of all instructions. Thus, the time to run n instructions will decrease from 950*n to 960*.958*n = 919.68*n. That corresponds to a speedup of 950/919.68 = 1.03.

The cost of the original CPU is 4496; the cost of the improved CPU is 4696. PC: 5, I-Mem: 1000, Register file: 200, ALU: 100, D-Mem: 2000, Sign Extend: 100, Controls: 1000, adders: 30*2, muxes: 3*10, single gates: 1*1. For a 3% increase in performance, the cost of the CPU increases by about 4.4%.

From a strictly mathematical standpoint, it does not make sense to add more registers because the new CPU costs more per unit of performance. However, that simple calculation does not account for the utility of the performance. For example, in a real-time system, a 3% performance may make the difference between meeting or missing deadlines. In which case, the improvement would be well worth the 4.4% additional cost.

4.11



No change to the Main Controller

ALU Controller

opcode	ALUOp	Operation	funct	ALU function	ALU control	lwic
lw	00	load word	XXXXXX	add	0010	0
sw	00	store word	XXXXXX	add	0010	0
beq	01	branch equal	XXXXXX	subtract	0110	0
R-type	10	add	100000	add	0010	0
R-type	10	subtract	100010	subtract	0110	0
R-type	10	AND	100100	AND	0000	0
R-type	10	OR	100101	OR	0001	0
R-type	10	set-on-less-than	101010	set-on-less-than	0111	0
R-type	**10**	**lwi**	**UNIQUE**	**add**	**0010**	**1**

4.12

Add a proposed `swap rs, rt` instruction to MIPS.

```
Reg[rt] = Reg[rs]; Reg[rs] = Reg[rt]
```

Outline of a possible solution is as follows:

- The register file needs to be modified so that it can write to two registers in the same cycle: need to add an extra write_register_2 port and an extra write_data_2 port.
- Connect read_data_1 to the original write_data port, and use mux to block out the connection in the original datapath.
- Connect read_data_2 to the new write_data_2 port, and pass the input to read_register_1 also to write_register_2.

Main Controller

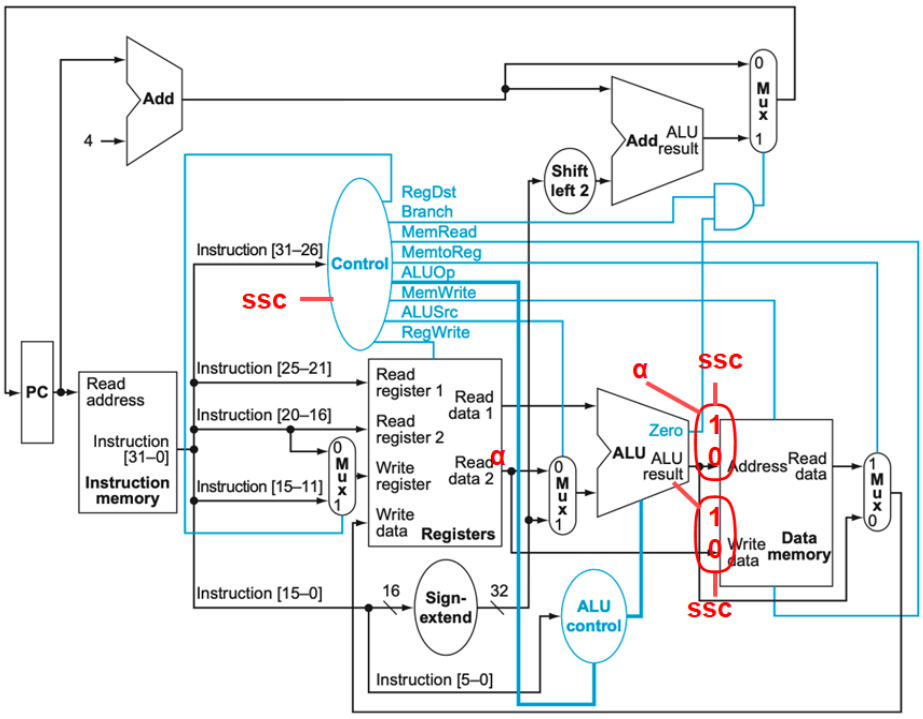
Signal Name	R-format	lw	sw	Beq	**swap**
Op5	0	1	1	0	**U**
Op4	0	0	0	0	**N**
Op3	0	0	1	0	**I**
Op2	0	0	0	1	**Q**
Op1	0	1	1	0	**U**
Op0	0	1	1	0	**E**
RegDst	1	0	X	X	**0**
ALUSrc	0	1	1	0	**X**
MemtoReg	0	1	X	X	**X**
RegWrite	1	1	0	0	**1**
MemRead	0	1	0	0	**0**
MemWrite	0	0	1	0	**0**
Branch	0	0	0	1	**0**
ALUOp1	1	0	0	0	**X**
ALUOp0	0	0	0	1	**X**
swapc	**0**	**0**	**0**	**0**	**1**

ALU is not used, no need to change the ALU Controller

4.13

Add a proposed `ss rt, rs, imm` (Store Sum) instruction to MIPS

`Mem[Reg[rt]]=Reg[rs]+SignExtend(immediate)`



Main Controller

Signal Name	R-format	lw	sw	Beq	ss
Op5	0	1	1	0	**U**
Op4	0	0	0	0	**N**
Op3	0	0	1	0	**I**
Op2	0	0	0	1	**Q**
Op1	0	1	1	0	**U**
Op0	0	1	1	0	**E**
RegDst	1	0	X	X	**X**
ALUSrc	0	1	1	0	**1**
MemtoReg	0	1	X	X	**X**
RegWrite	1	1	0	0	**0**
MemRead	0	1	0	0	**0**
MemWrite	0	0	1	0	**1**
Branch	0	0	0	1	**0**
ALUOp1	1	0	0	0	**0**
ALUOp0	0	0	0	1	**0**
ssc	**0**	**0**	**0**	**0**	**1**

No change to the ALU Controller