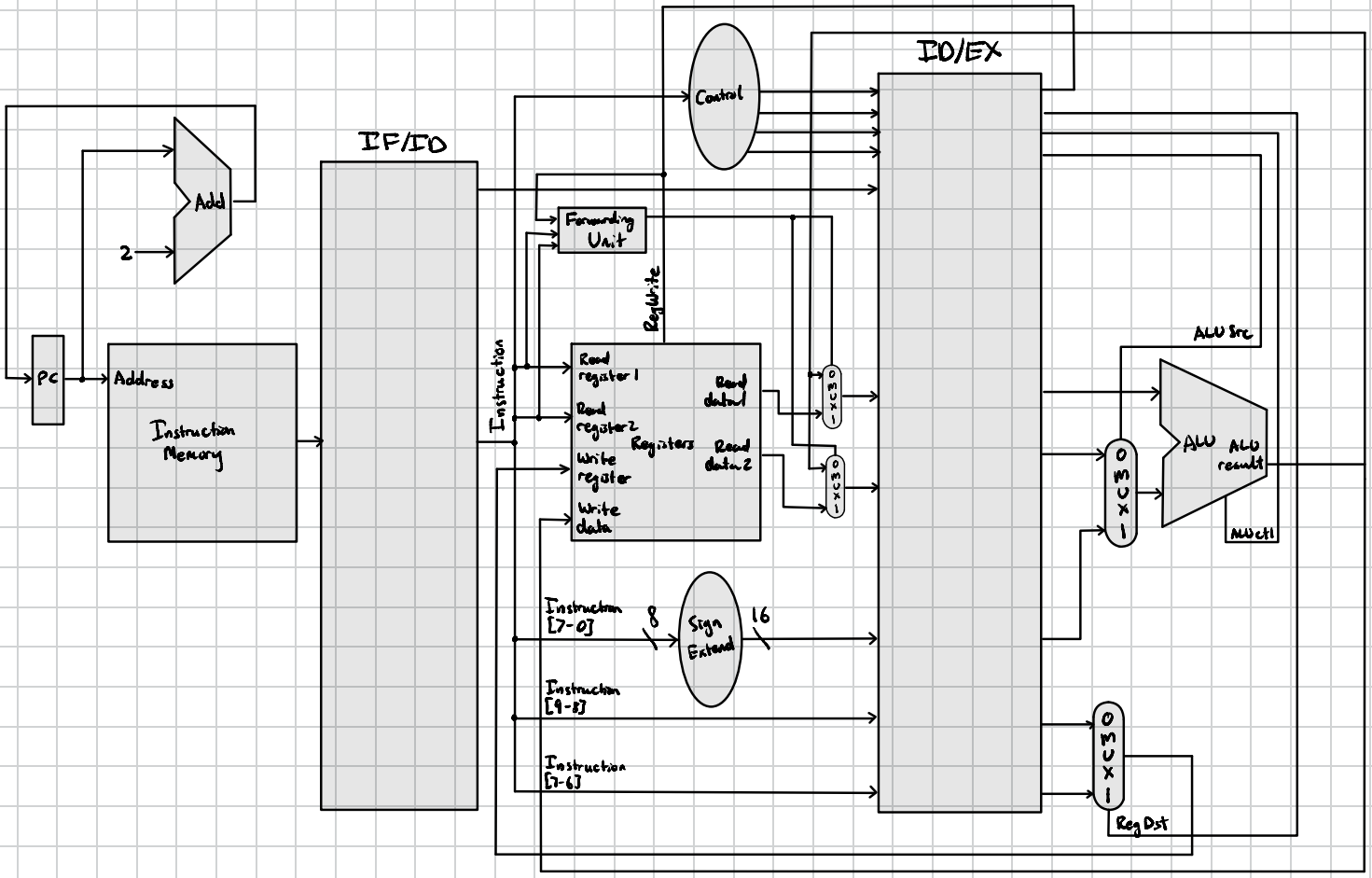
Rumman Shahzad
Shah Qureshi

Progress Report #3 – 4/11/2024

Objective: 3-stage pipelined datapath supporting R-type & addi instructions (16-bit MIPS) w/forwarding included.



Reg Array Implementation:
- PC
- Instruction memory
- Pipeline Registers

Behavioral:
- Main Control
- Register File
- Sign Extend/Adder
- Forwarding

Gate level:
- ALU
- Multiplexers

* HA/FA logic diagrams are not shown

Instruction Set Architecture/Control Unit Truth Table:

	Instr	Opcode	RegDst	ALUSrc	RegWrite	ALUctl
R-type	add	0000	1	0	1	0010
	sub	0001	1	0	1	0110
	and	0010	1	0	1	0000
	or	0011	1	0	1	0001
	nor	0100	1	0	1	1100
	nand	0101	1	0	1	1101
	slt	0110	1	0	1	0111
I-type	addi	0111	0	1	1	0010

Instruction Formats:

R-format: add, sub, and, or, nor, nand, slt

op	rs	rt	rd	unused
4	2	2	2	6

⟷ instr rd, rs, rt

I-format: addi

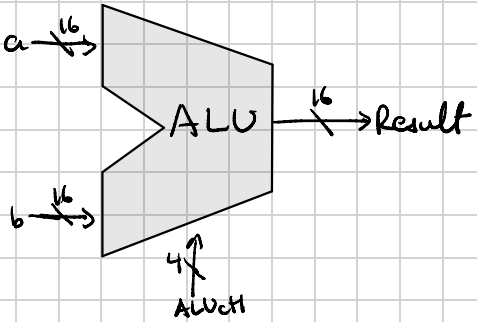
op	rs	rt	addr/value
4	2	2	8

⟷ addi rt, rs, value

Rumman Shahzad
Shah Qureshi

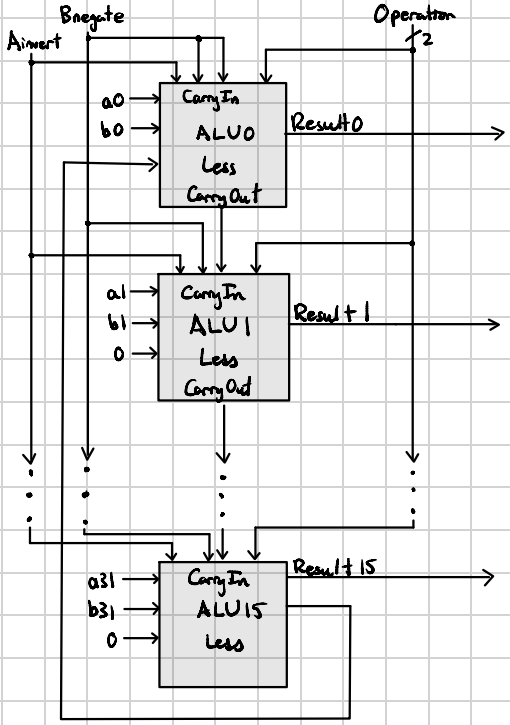
16-bit ALU:

* No Zero or Overflow *



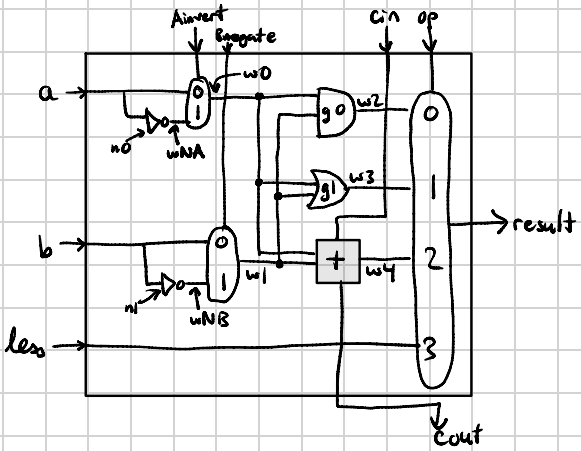
ALU Control Lines:

Ainvert	Bnegate	Operation	Instr.
ALUctl[3]	ALUctl[2]	ALUctl[1:0]	
0	0	10	add
0	1	10	sub
0	0	00	and
0	0	01	or
1	1	00	nor
1	1	01	nand
1	1	11	slt
0	0	10	addi



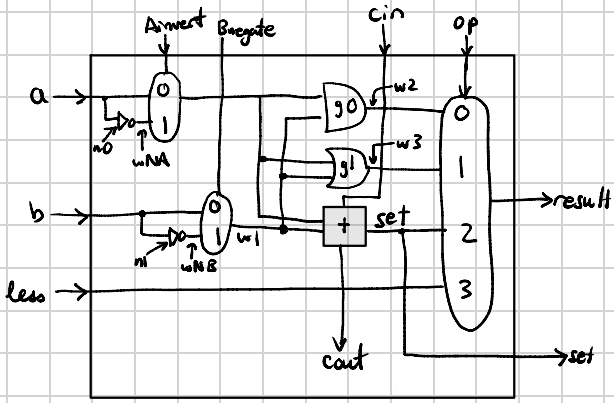
ALU0 to ALU14:

1-bit ALU, cascaded.

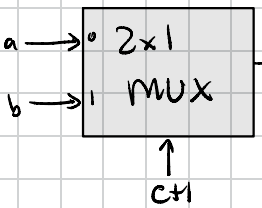


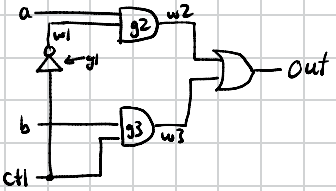
ALU15:

1-bit ALU, last of the cascaded chain of ALU's. Slightly diff.



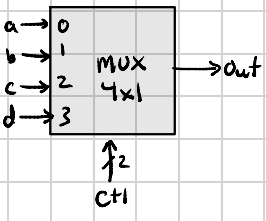
1-bit 2x1 Multiplexer:

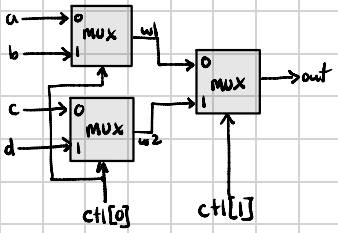




1-bit 4x1 Multiplexer:

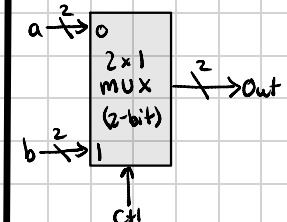
Implemented by cascading 2x1 multiplexers.

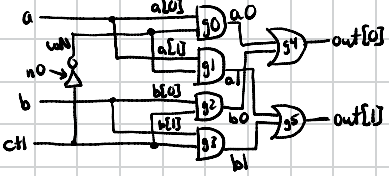




2-bit 2x1 Multiplexer:

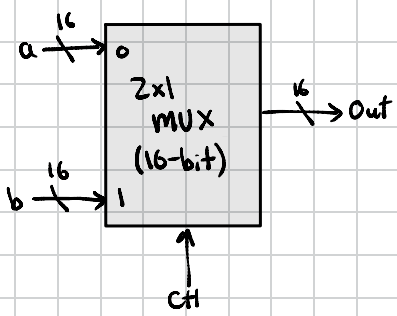
Multiplexes 2 inputs of 2-bit data.

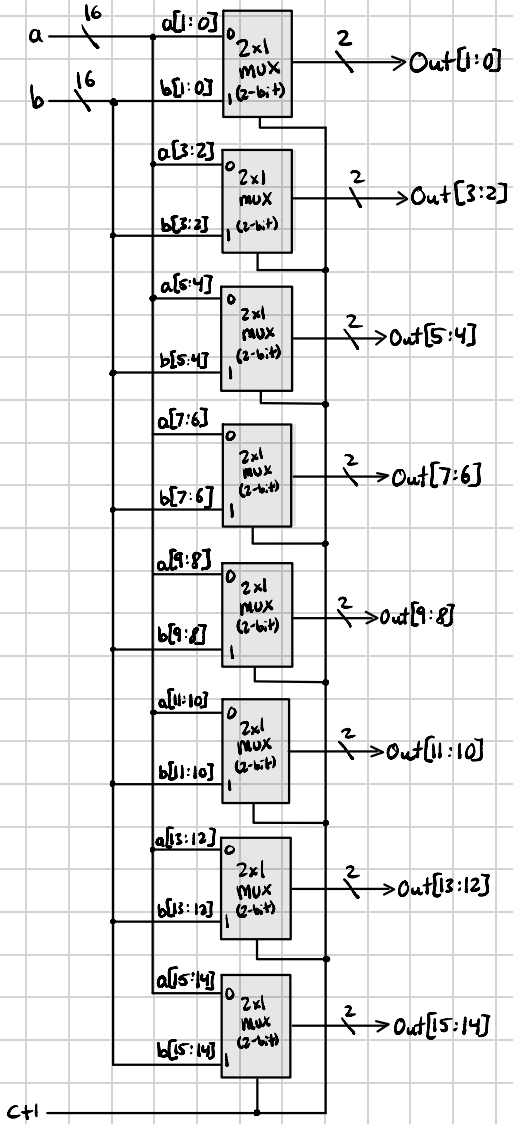




16-bit 2x1 Multiplexer: Multiplexes 2 inputs of 16-bit data.

↳ Implemented by cascading 8 2-bit 2x1 multiplexers, as shown below.





Objective: 3-stage pipelined datapath for R-type & addi