

ECS 154B Discussion 3

January 21st, 2022

Goals

- Assignment 2 FAQ.
- Datapath examples for different instruction types.
- Overview of assignment 3.

Logistics

- Assignment 2 new due date is on January 26th
- Assignment 3 is *going to be* released this week
 - Prompt: https://jlpteaching.github.io/comparch/modules/dino_cpu/assignment3/
 - Repo: <https://github.com/jlpteaching/dinocpu-wq22>
 - Due on February 6th (both parts)
 - Will need to update your repo to get the tests for lab 3

Assignment 2: Memory Ports

- **maskmode:** size of data
 - 0 -> byte (8 bits)
 - 1 -> halfword (16 bits)
 - 2 -> word (32 bits)
 - 3 -> doubleword (64 bits)
- **sext:** sign extending data
 - Used for load instructions
 - True if the loaded data should be sign extended
 - Necessary since each load instruction has an unsigned variant.
 - E.g.: `lb` and `lbu`

Data Memory
address readdata
memread
memwrite
valid
maskmode
sext
writedata

Assignment 2: Shift Instructions (I-type)

- Only 6 bits are used for shift amount (5 bits for word variants).
- imm[11:6]: used to differentiate SRLI and SRAI

31	26	25	24	20 19	15 14	12 11	7 6	0
imm[11:6]	imm[5]	imm[4:0]		rs1	funct3	rd	opcode	
6	1	5	5	3	5	7		
000000	shamt[5]	shamt[4:0]		src	SLLI	dest	OP-IMM	
000000	shamt[5]	shamt[4:0]	(15:20)	src	SRLI 101	dest	OP-IMM	
<u>010000</u>	<u>shamt[5]</u>	<u>shamt[4:0]</u>	<u>(15:20)</u>	src	SRAI 101	dest	OP-IMM	
000000	0	shamt[4:0]		src	SLLIW	dest	OP-IMM-32	
000000	0	shamt[4:0]		src	SRLIW	dest	OP-IMM-32	
010000	0	shamt[4:0]		src	SRAIW	dest	OP-IMM-32	

W/

Assignment 2: Converting between signed and unsigned

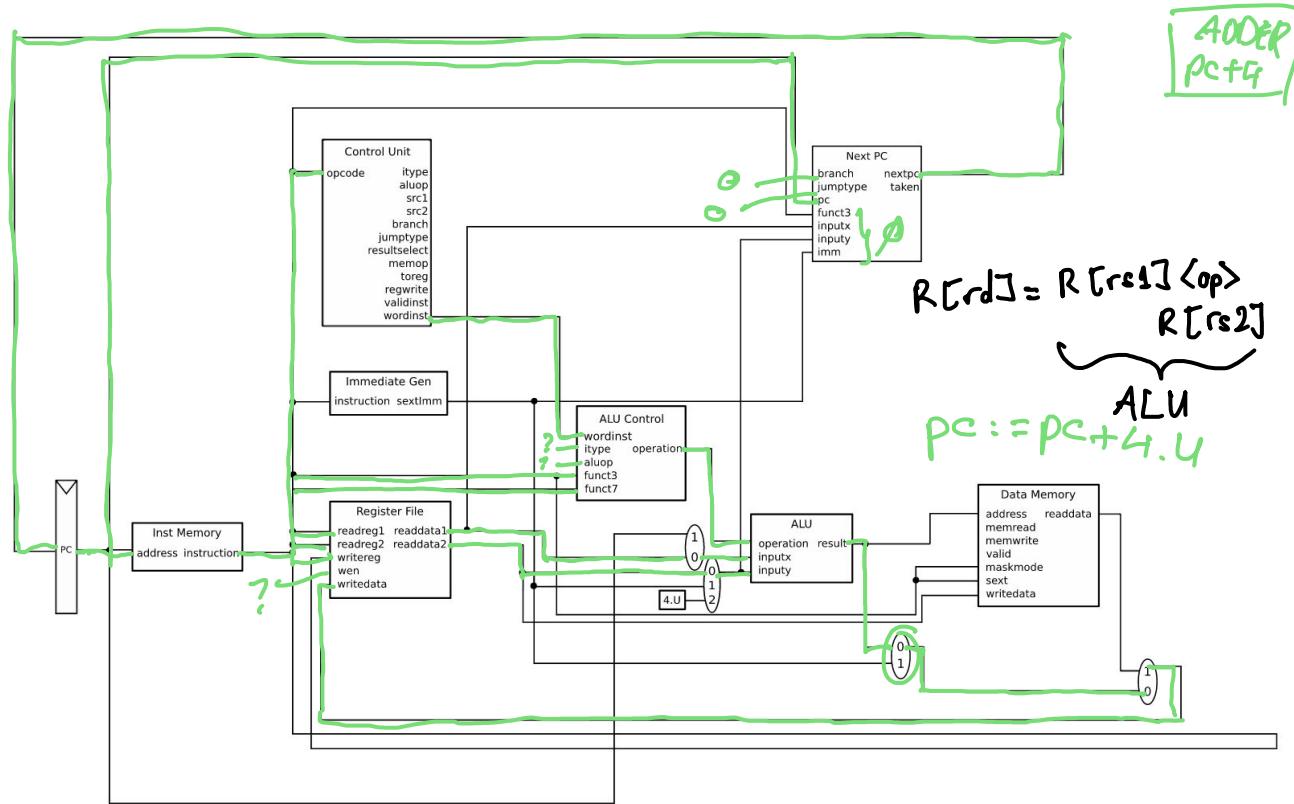
- `.asUInt`: convert to an unsigned integer
- `.asSInt`: convert to a signed integer

Val $va = 4.4(4.w)$

when ($va.\text{asSInt} == (-1, \underbrace{s}_{\text{signed int}}) \{ ... \}$)

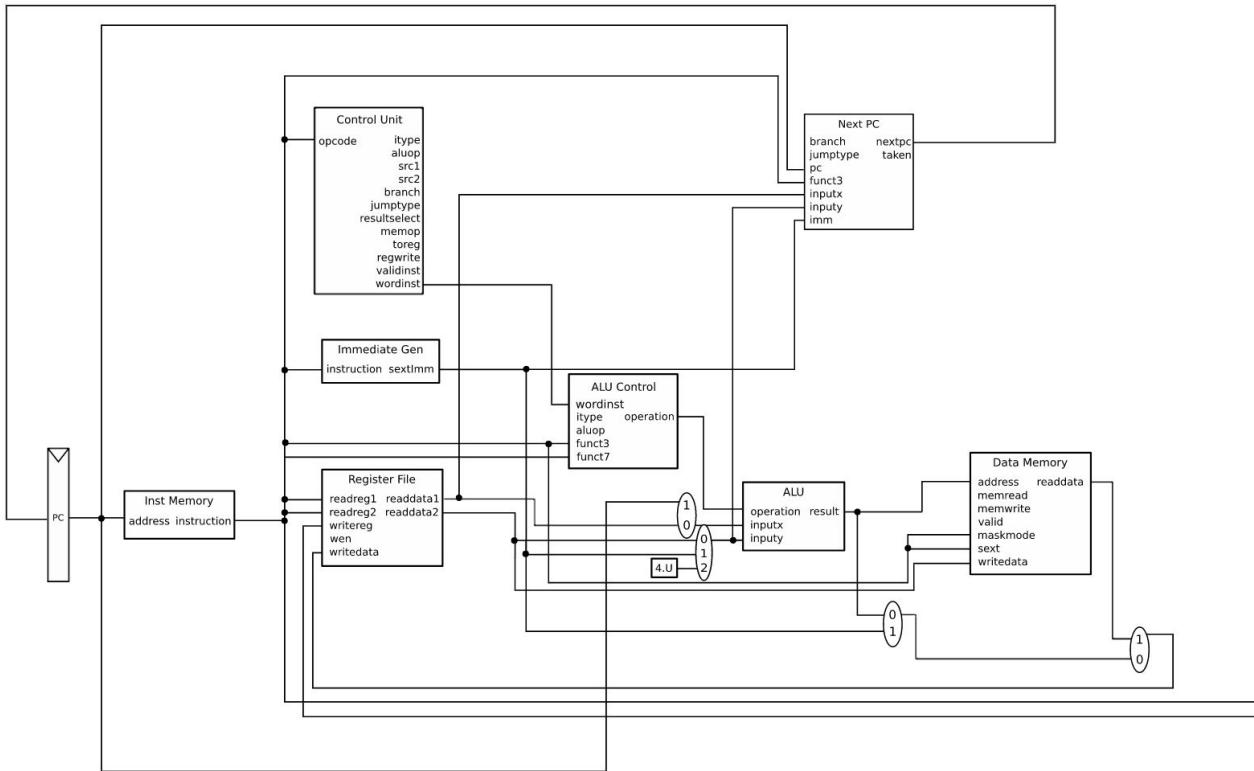
→ components / alu.scala

R-type Data Path

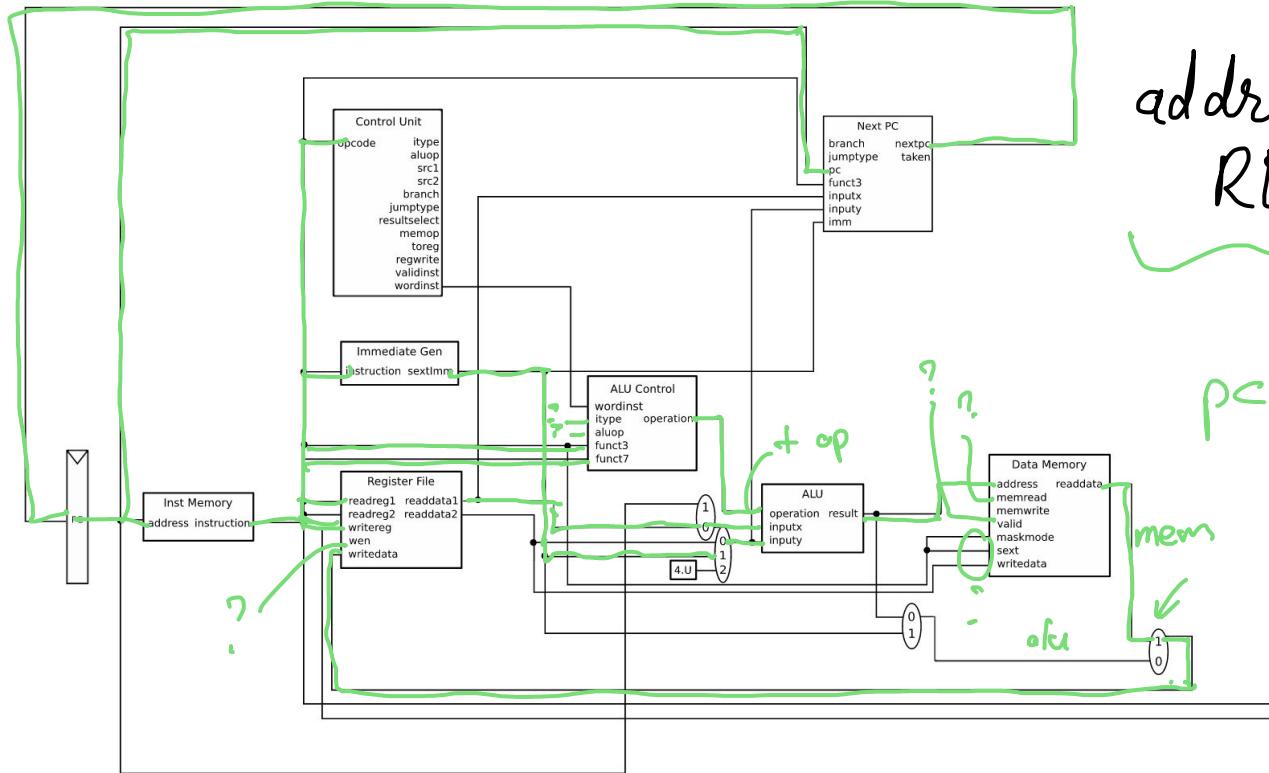


I-type Data Path

(Discussion 2)



Load Instructions Data Path



$$\underline{R[rd]} = \text{Mem}[R[rs1] + imm]$$

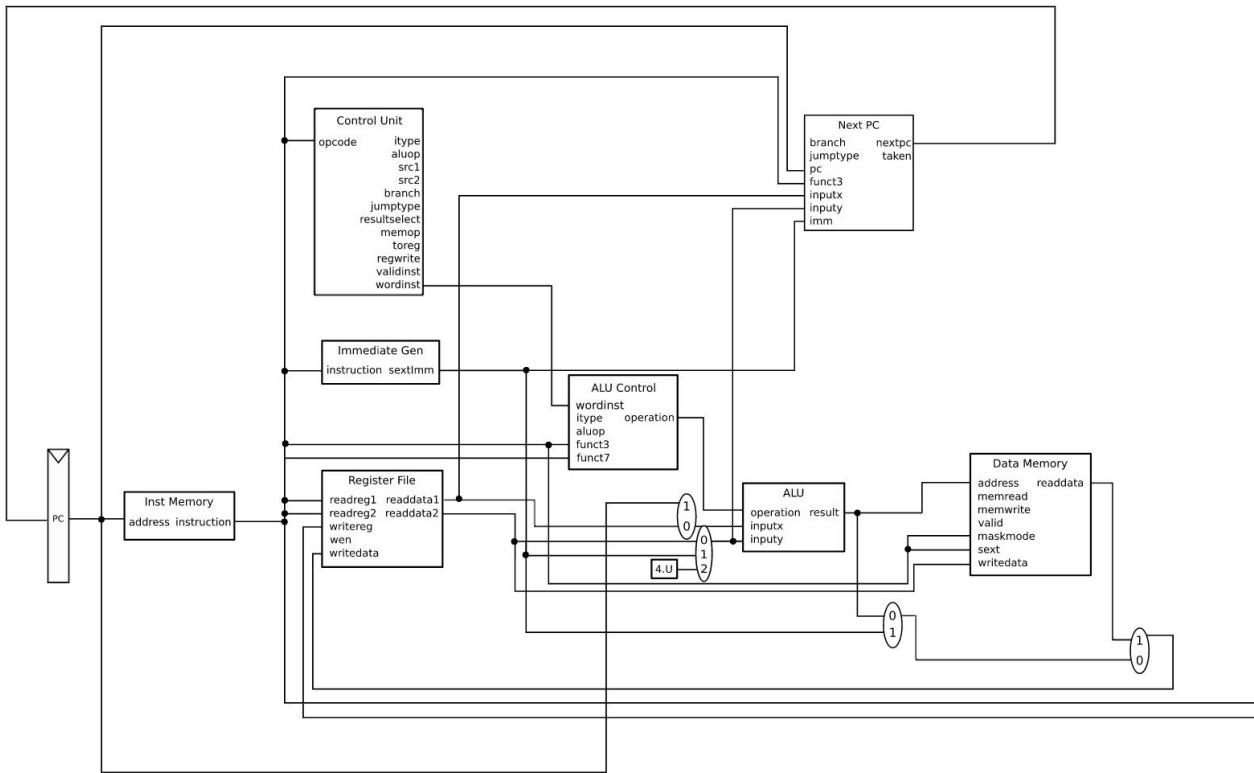
address = +
RE[rs1] + imm

ALU
PC = PC + 4

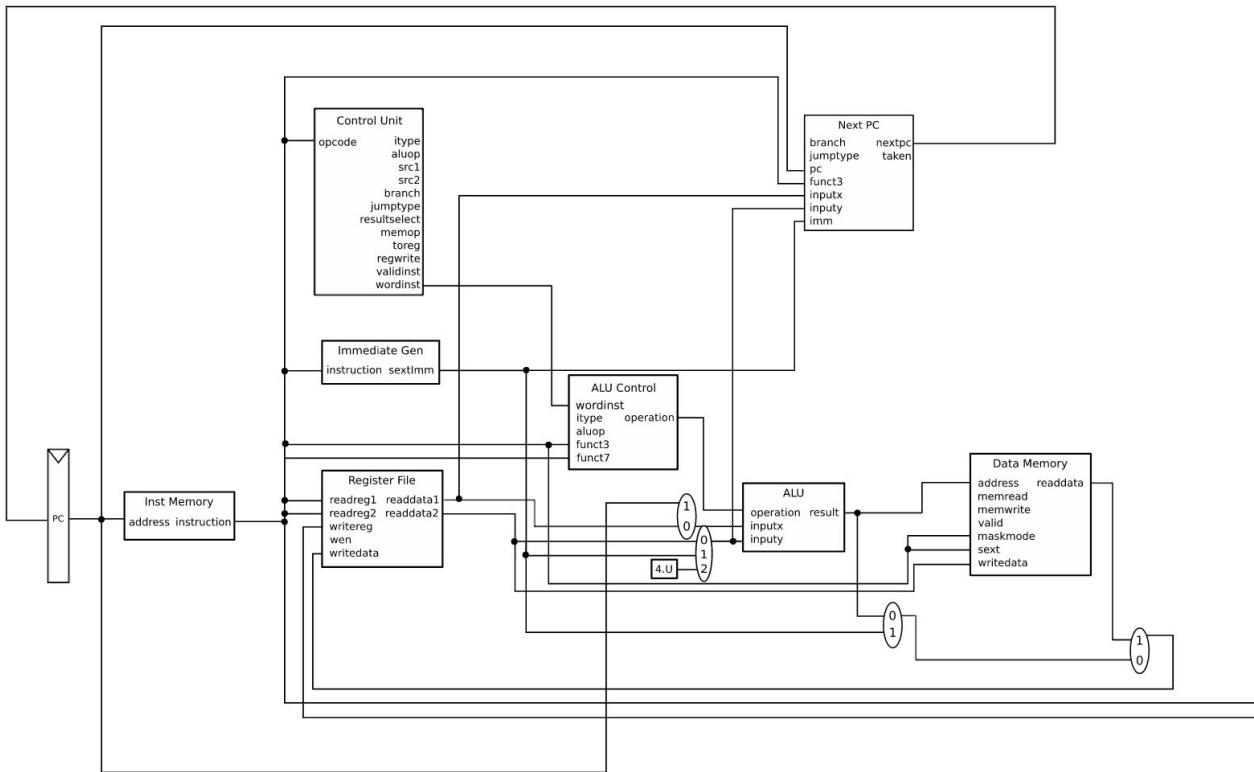
mem

alu

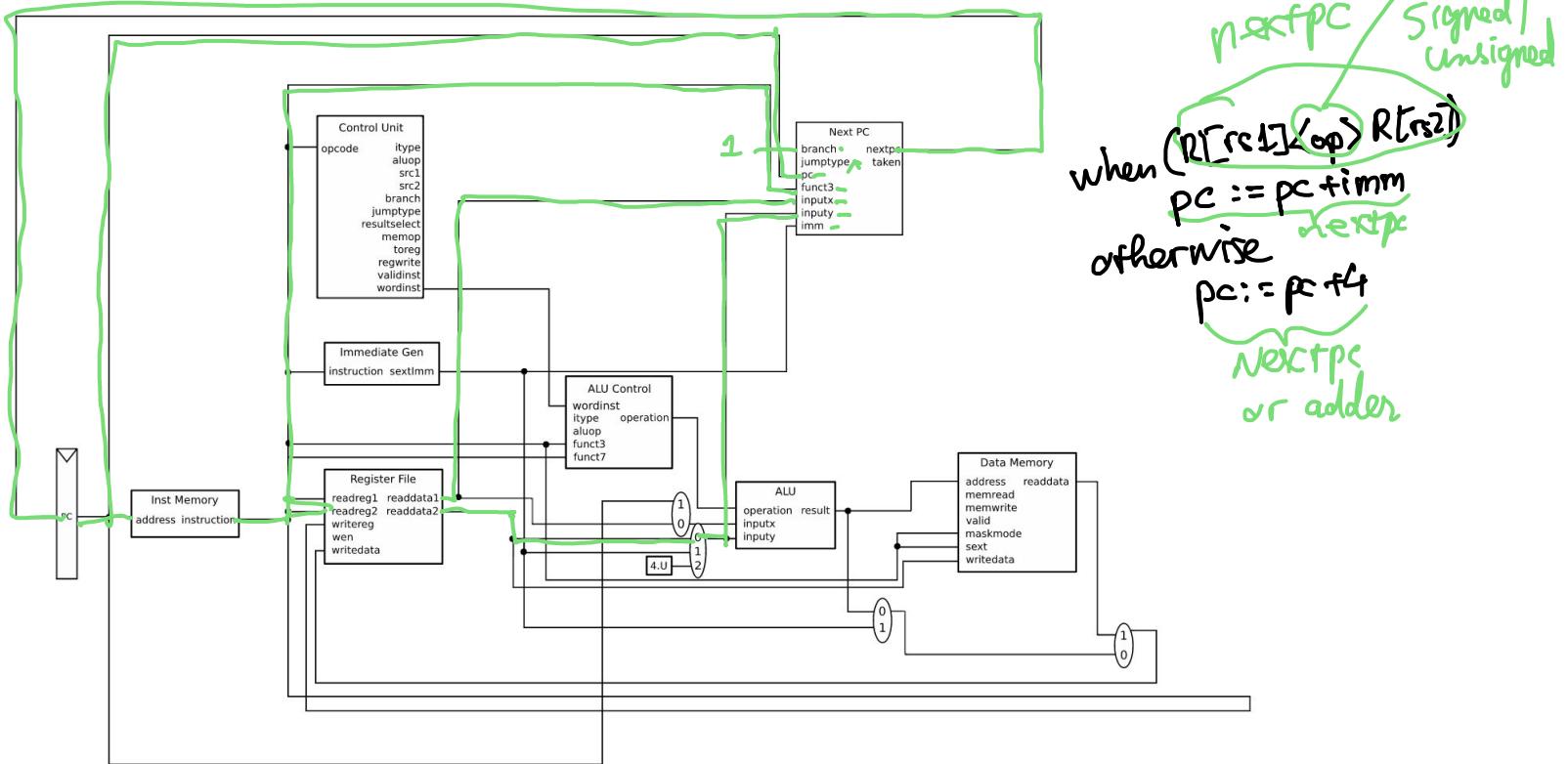
Store Instructions Data Path



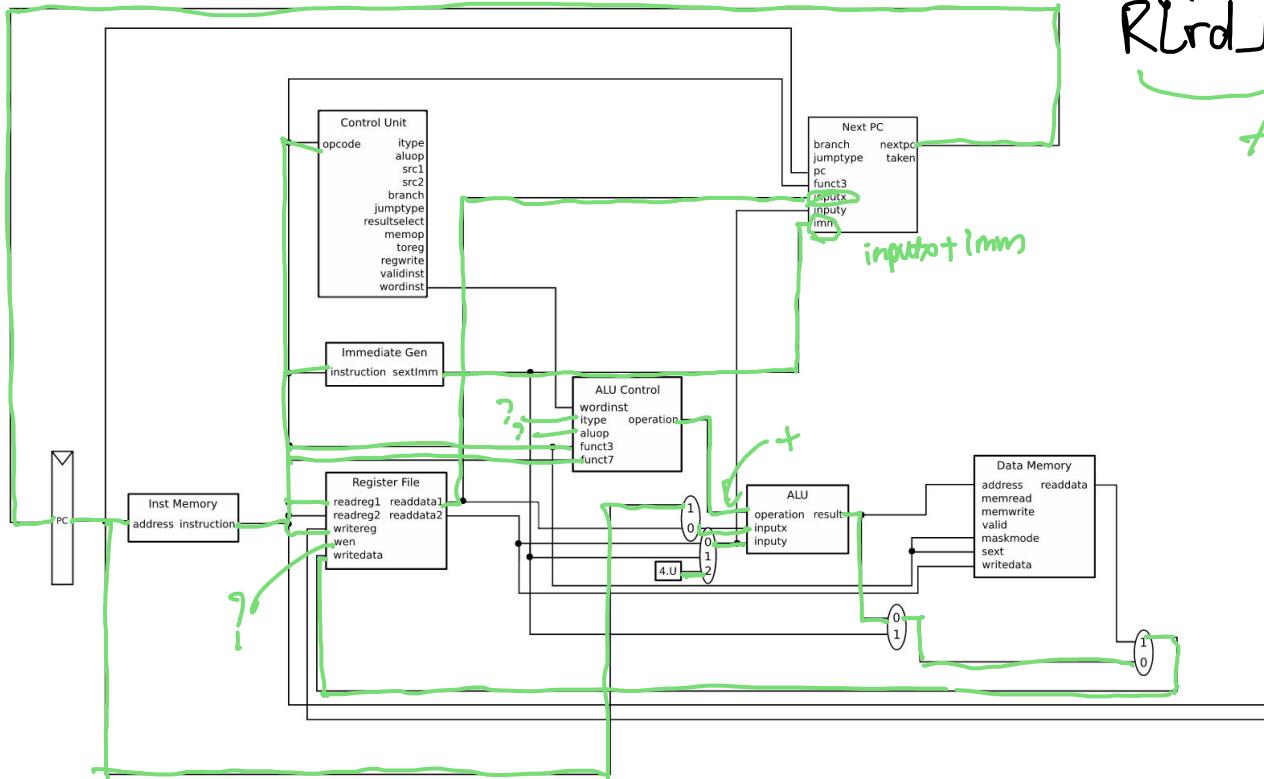
U-type Data Path



B-type Data Path



Jump Instructions Data Path



$$\begin{aligned} \text{jal} \\ \text{jalr} \\ \text{nextpc} \\ \{ \text{pc} = R[\text{rs1}] + \text{imm} \} \\ R[\text{rd}] = \text{pc} + 4 \\ \text{ALU} \end{aligned}$$

Annotations for thejal and jalr instructions:

- jal:** A green oval highlights the jal instruction.
- jalr:** A green oval highlights the jalr instruction.
- nextpc:** A green label indicates the output of the ALU is the next PC.
- plus:** A green label indicates the operation is addition (inputx + imm).
- ALU:** A green label identifies the ALU unit.

preserving the 5-stage pipelined CPU: F, D, E, N, WB

Assignment 3: Pipelined CPU

