154 Discussion 2

January 18th, 2023

Goals

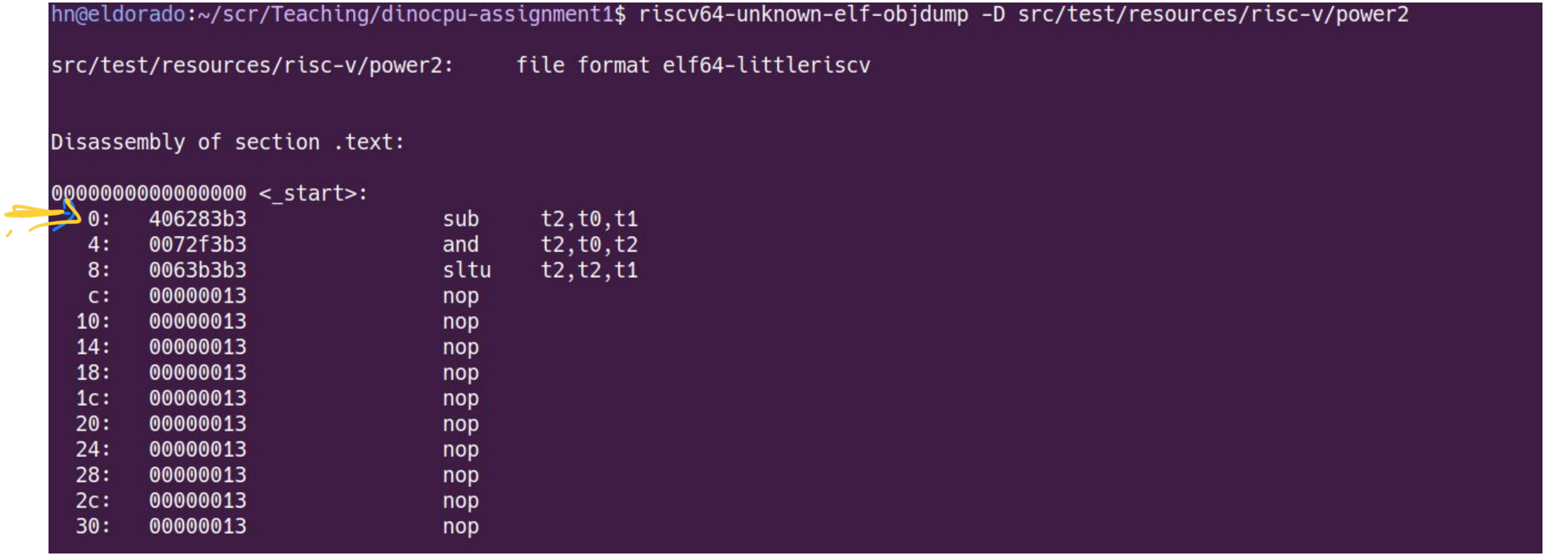
- Assignment 1 Part V: executing multiple cycles.
- Assignment 1 diagram.
- Assignment 2 Overview.
    - Instruction types.
    - Control flow / Data flow.

Logistics

- Assignment 3 will be released some time this week.
- Quiz grading,
  - score = max(original_quiz_score, late_quiz_score - 0.25 * not_attempted_original_quiz)
  - Unlimited attempts.

Assignment 1: executing multiple cycles

- Where is the current instruction to execute?
    - The value of PC is the address of the instruction.

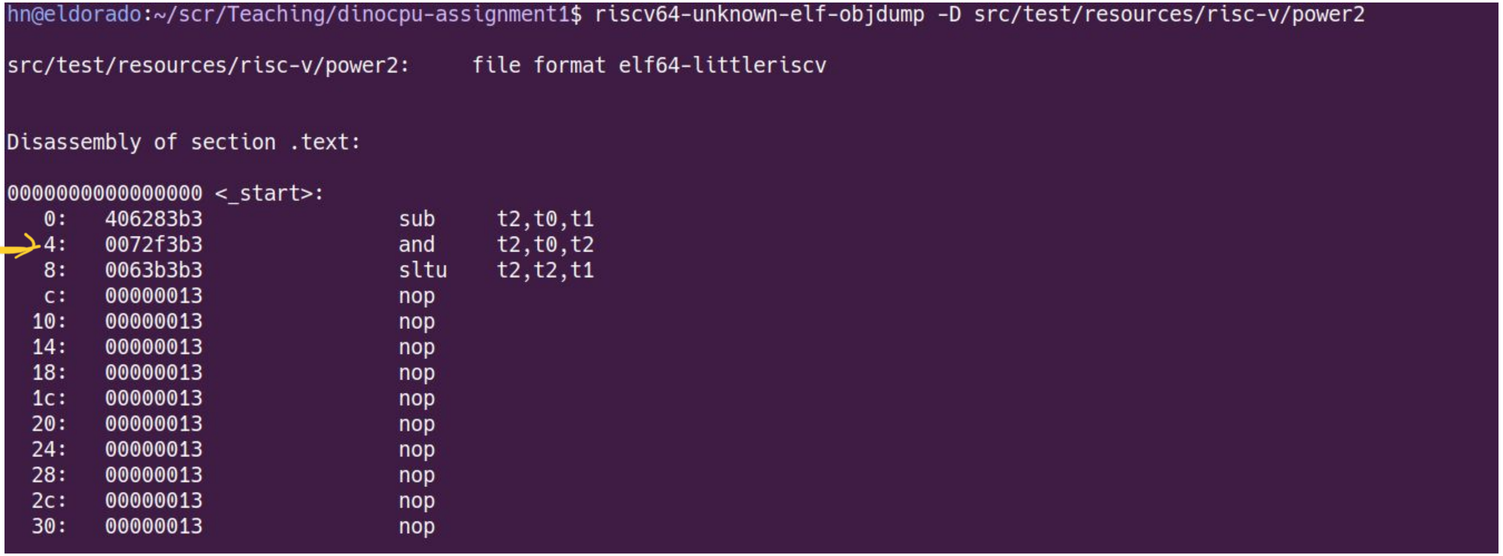


```
hn@eldorado:~/scr/Teaching/dinocpu-assignment1$ riscv64-unknown-elf-objdump -D src/test/resources/risc-v/power2

src/test/resources/risc-v/power2:     file format elf64-littleriscv


Disassembly of section .text:

0000000000000000 <_start>:
   0:   406283b3                sub     t2,t0,t1
   4:   0072f3b3                and     t2,t0,t2
   8:   0063b3b3                sltu    t2,t2,t1
   c:   00000013                nop
  10:   00000013                nop
  14:   00000013                nop
  18:   00000013                nop
  1c:   00000013                nop
  20:   00000013                nop
  24:   00000013                nop
  28:   00000013                nop
  2c:   00000013                nop
  30:   00000013                nop
```

Assignment 1: executing multiple cycles

instruction : 32-bit
(4 bytes)

- Where is the next instruction to execute?
  - If the current instruction does not alter the control flow, the next instruction is the one next to the current instruction.



```
hn@eldorado:~/scr/Teaching/dinocpu-assignment1$ riscv64-unknown-elf-objdump -D src/test/resources/risc-v/power2

src/test/resources/risc-v/power2:     file format elf64-littleriscv


Disassembly of section .text:

0000000000000000 <_start>:
   0:	406283b3          	sub	t2,t0,t1
   4:	0072f3b3          	and	t2,t0,t2
   8:	0063b3b3          	sltu	t2,t2,t1
   c:	00000013          	nop
  10:	00000013          	nop
  14:	00000013          	nop
  18:	00000013          	nop
  1c:	00000013          	nop
  20:	00000013          	nop
  24:	00000013          	nop
  28:	00000013          	nop
  2c:	00000013          	nop
  30:	00000013          	nop
```

Assignment 1: executing multiple cycles

- Why PC + 4? What is the assumption?
  - From ISA: One instruction is 32 bits (4 bytes) long.
  - In DINOCPU (and most common microarchitectures): Byte-addressable architecture.
- There are other addressing mode, e.g., word addressing mode.

Address (byte-addressable)

0 → first byte

1 → second byte

⋮

Address

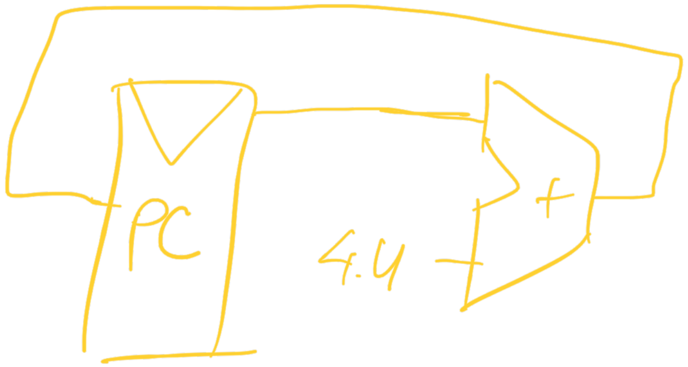
0 → first 32 bits

1 → second 32 bits

⋮

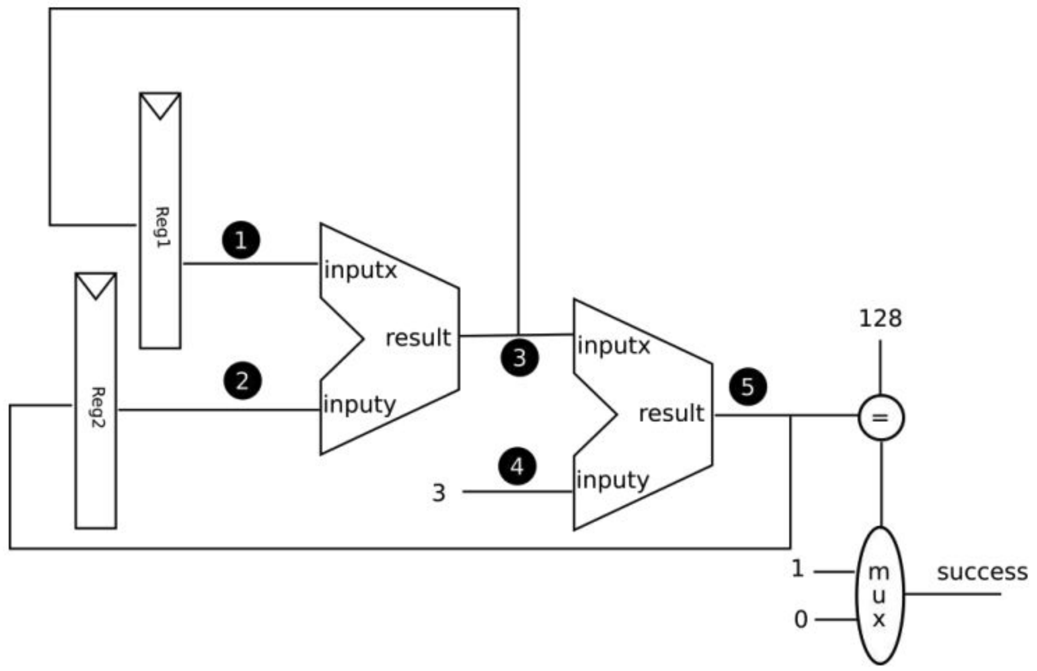
Assignment 1: executing multiple cycles

- What Chisel does when compiling the statement PC := PC + 4.U?
  - The compiler will generate an adder updating the PC register.



Assignment 1 Diagram

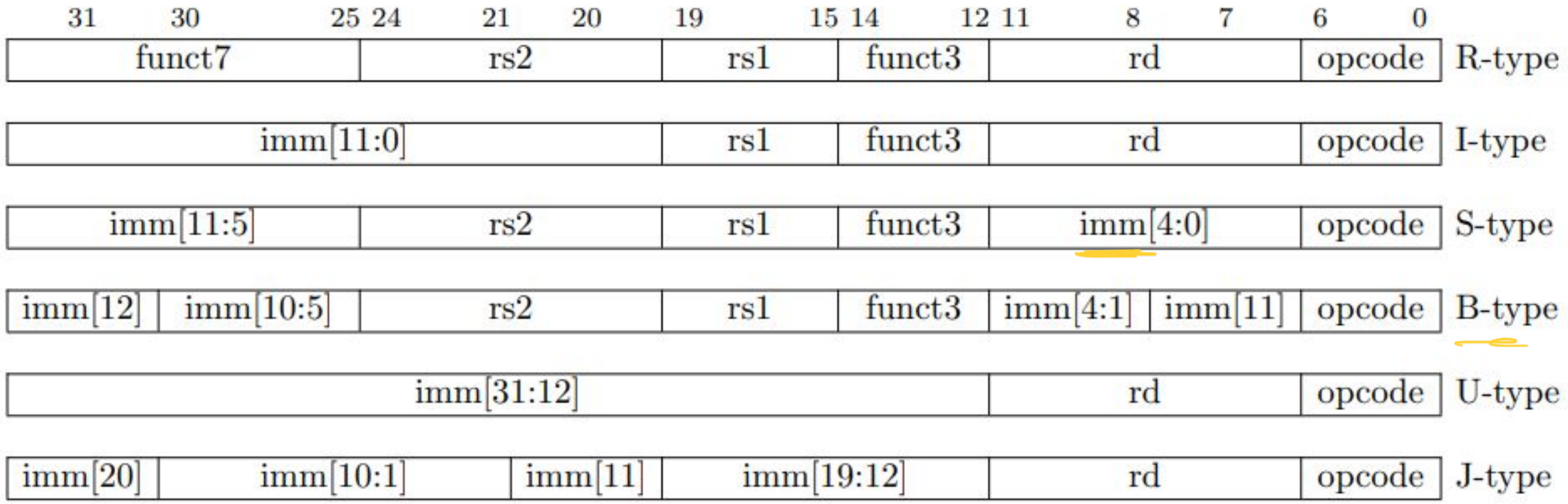
- Wires
  - Width
  - Subset of bits
- Components
  - Adder for updating PC
  - Comparator if needed
- Don’t have to be formal, but must be clear and readable.



B-type: if [rs1]<op>[rs2] == True: PC = PC+imm
otherwise: PC = PC+4

R: [rd] = [rs1] <op> [rs2]
I: [rd] = [rs1] <op> imm
S: mem[rs1 + imm] = [rs2]

Assignment 2: RV64IM Instruction Types



31	30 — 25	24 — 21	20	19 — 15	14 — 12	11 — 8	7	6 — 0	
funct7		rs2		rs1	funct3	rd		opcode	R-type
imm[11:0]				rs1	funct3	rd		opcode	I-type
imm[11:5]		rs2		rs1	funct3	imm[4:0]		opcode	S-type
imm[12]	imm[10:5]	rs2		rs1	funct3	imm[4:1]	imm[11]	opcode	B-type
imm[31:12]						rd		opcode	U-type
imm[20]	imm[10:1]		imm[11]	imm[19:12]		rd		opcode	J-type

Figure 2.3: RISC-V base instruction formats showing immediate variants.

Assignment 2: Control Unit

- The control unit orchestrates the control flow and the data flow of the instruction execution per instruction type.
  - Deciding the functionality of each component.

CU

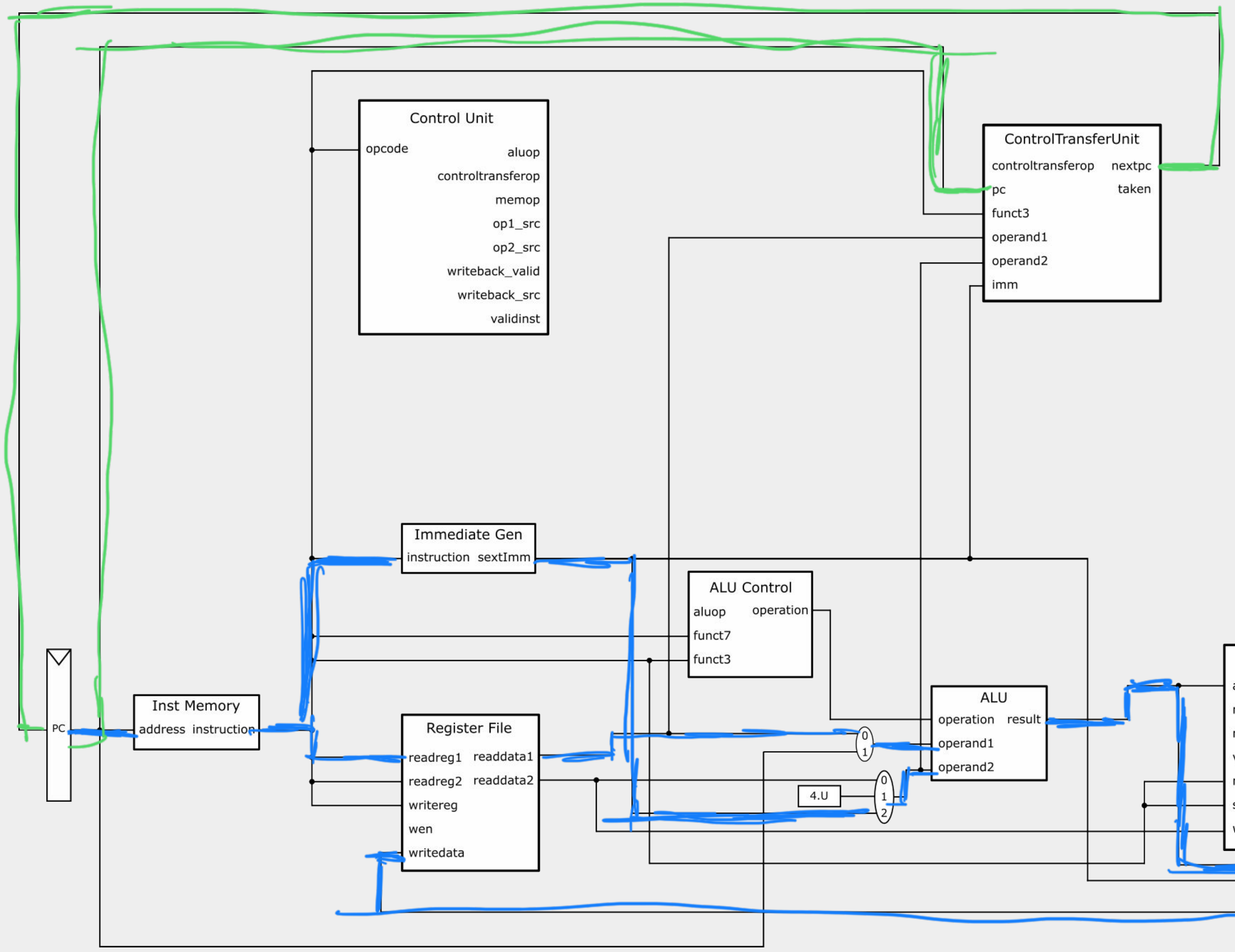
input: opcode $\rightarrow$ instruction type

Assignment 2: Control Flow

- Branch/Jump instructions are called control transfer instructions. I.e., they alter the control flow.
  - Jumps always change the next PC to a value other than PC+4. Used a lot in function calls.
  - Branches change the next PC to a value other than PC+4 if the branch condition is true.

Assignment 2: Data Flow

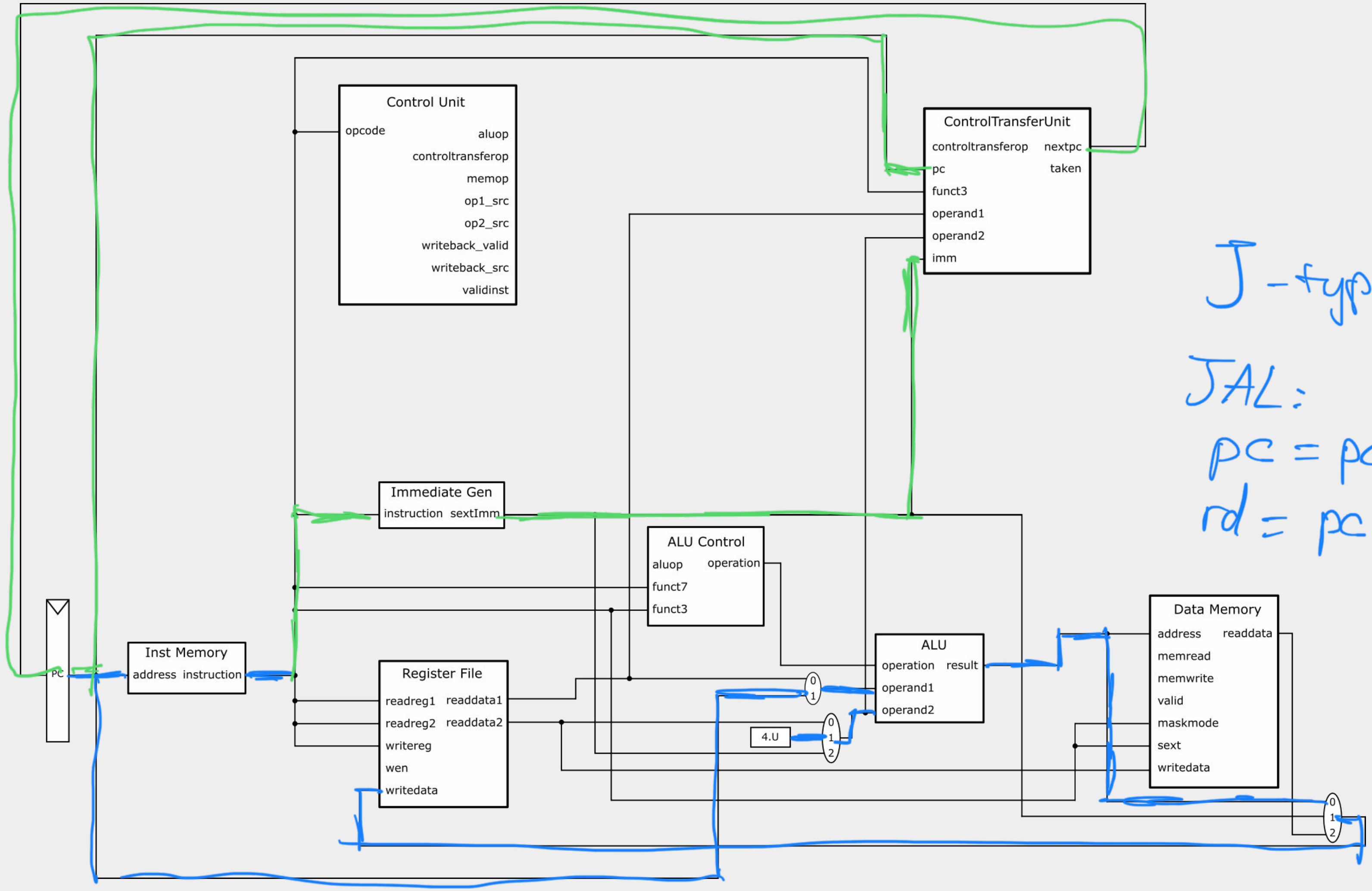
- Data flow in the single-cycle DINOCPU,
  - registers -> registers
  - registers -> memory
  - memory -> registers



I-type

$rd = rs1 \underset{\langle op \rangle}{\&} imm$

$pc = pc + 4$



J-type

JAL:

pc = pc + imm

rd = pc + 4

Week 2 Quiz

Iron Law:

$$\frac{\text{time}}{\text{program}} = \underbrace{\frac{\overset{\text{Dynamic}}{\text{Instructions}}}{\text{program}}}_{\text{alters when changing ISA}} \times \underbrace{\text{CPI}}_{\text{alters when using different march / applications}} \times \underset{(1/f)}{\frac{\text{time}}{\text{cycle}}}$$

Week 2 Quiz

Q.9.
Q.10 $\Bigg\}$ $\dfrac{\#\text{cycles}}{\#\text{instructions}}$

Q.14. average freq of Intel : ?

$\dfrac{\text{cycles}}{1s}$ $\dfrac{\#\text{cycles}}{\#\text{seconds}}$

Q.16 CPI = 1.2

# instructions

→ time?

freq × CPI × #insts

Week 2 Quiz

$P_{old} = P_{new}$

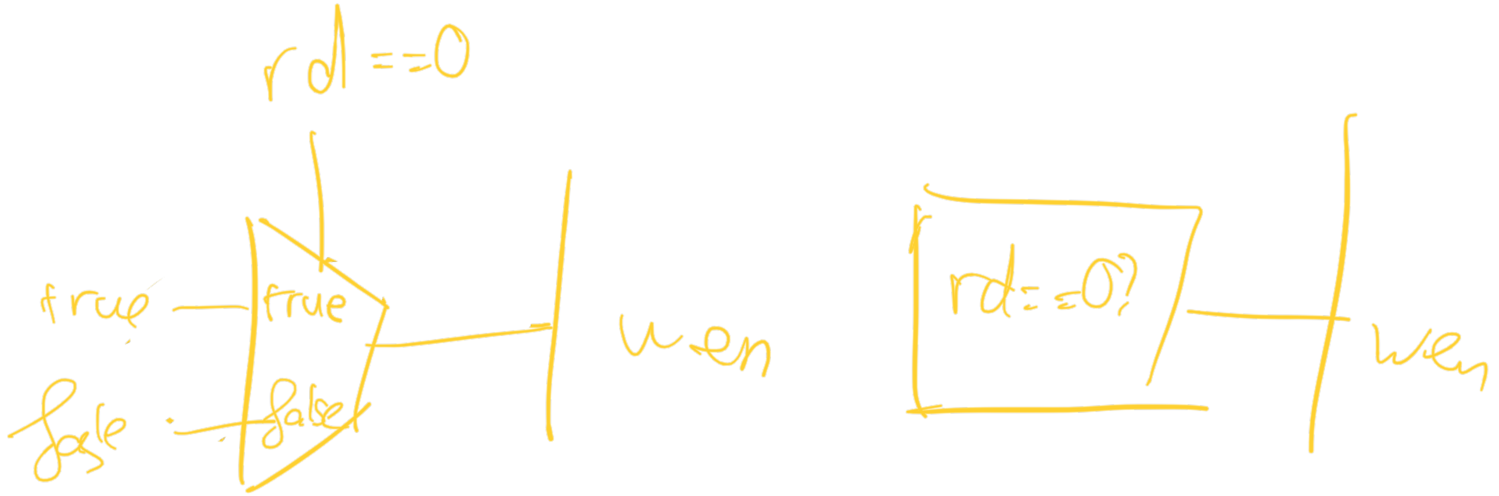
$P \sim \frac{C \cdot V^2}{\cancel{[illegible]}} \cdot f$

$\cancel{C} \cdot V_{old}^2 \cdot f_{old} = \cancel{C} \cdot V_{new}^2 \cdot f_{new}$

$\cancel{V_{old}^2} \cdot f_{old} = (0.9\, \cancel{V_{old}})^2 \cdot f_{new}$

$1.\ldots = \frac{1}{0.81} = \frac{1}{0.9^2} = \frac{f_{new}}{f_{old}}$

~~Week 2 Quiz~~ Diagram of Assignment 1



~~Week 2 Quiz~~ Assignment 1 note

→ The RISC-V ISA specs say that x0 (the register 0) is a hardwired zero.

→ Since you're not allowed to update the code of the register file, you should have extra logics in the cpu that register 0 will not be written to if rd is 0.

~~Week 2 Quiz~~ Iron Law note

$$\mathrm{CPI} = \sum_{\text{instruction type}} \left(\underbrace{\%\ \text{instruction type in a program}}_{\text{dependent on the program}} \times \underbrace{\text{time to execute that instruction type}}_{\text{dependent on the march}} \right)$$