


COSC175 (Systems I): Computer Organization & Design









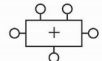
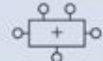










Professor Lillian Pentecost
Fall 2024



Warm-Up November 12

- Where we were
 - How are high-level languages translated into assembly instructions?
 - Demo w/ command line, GDB, and Makefile
- Where we are going
 - Roadmap for the rest of this course!!
- Logistics, Reminders
 - TA help 7-9PM on Sundays, Tuesdays, Thursdays in C107
 - Use to start review of whole semester!
 - LP Office hours M 9-10:30AM, Th 2:30-4PM
 - Weekly Exercises due Friday 5PM
 - A little bit different – take a look ahead of time!
 - Lab 6 due **Today 10PM**

Application Software		Application Software	
Operating Systems		Operating Systems	
Architecture		Architecture	
Micro-architecture		Micro-architecture	
Logic		Logic	
Digital Circuits		Digital Circuits	
Analog Circuits		Analog Circuits	
Devices		Devices	
Physics		Physics	

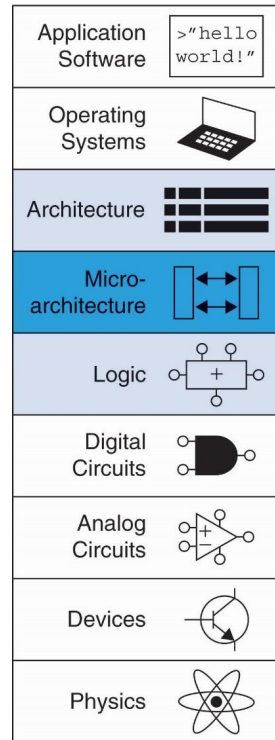
Roadmap for the rest of this course!!

Three big questions left to explore:

1. **How to implement and support an architecture (RISC-V) with a *microarchitecture*?**
 - a. Your last, multi-part lab will ask you to construct and test a RISC-V 32-bit processor
 - b. This will be a specific arrangement and combination of the building blocks we've seen all semester – registers and ALUs and FSMs and Memories and MUXes, all synchronized by a clock signal
2. **What are key additions and optimizations that make a computer more *efficient* and *usable* from a programmer's standpoint?**
 - a. Mostly, this means both HW and OS approaches to managing memory – **caches, virtual memory, paging**, and file systems (all of which you'll see more of in COSC-275)
 - b. This will incorporate additional analysis of C programs, more interesting weekly exercises
3. **What is a topic in computer organization and design that you'd like to learn more about?**
 - a. You will vote next week on an advanced topic to cover!

What is microarchitecture?

- **Microarchitecture:** how to implement an architecture in hardware
- We've studied RISC-V architecture from a *programmer's perspective*, now we try an example implementation
- **Processor:**
 - **Datapath:** functional blocks
 - **Control:** control signals



Microarchitecture

- **Multiple implementations** for a single architecture, for example:
 - **Single-cycle:** Each instruction executes in a single cycle
 - **Multicycle:** Each instruction is broken up into series of shorter steps
 - **Pipelined:** Each instruction broken up into series of steps & multiple instructions execute at once

Processor Performance

- **How can we compare different implementations? What is our goal?**

Processor Performance

- **How can we compare different implementations? What is our goal?**

One important goal:

minimize program execution time

Processor Performance

- **Program execution time**

Execution Time = (#instructions)(cycles/instruction)(seconds/cycle)

- **Definitions:**

- CPI: Cycles/instruction
- clock period: seconds/cycle
- IPC: instructions/cycle = IPC

- **Challenge is to satisfy constraints of:**

- Cost
- Power
- Performance

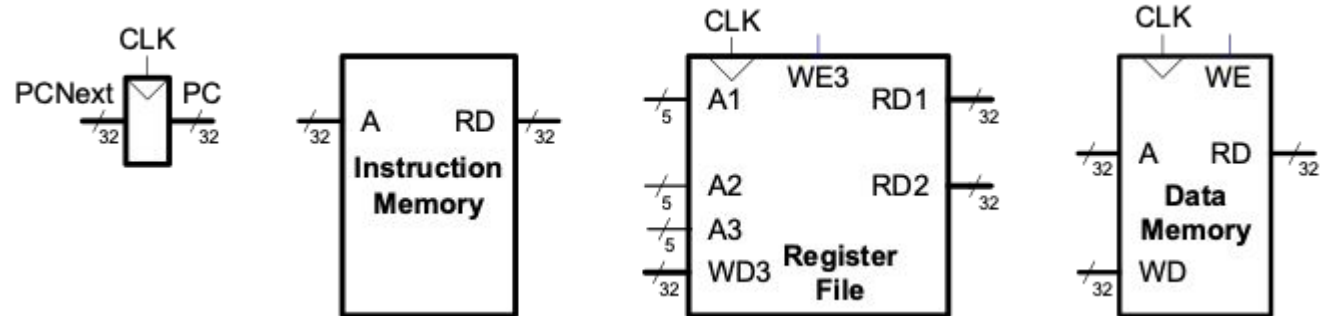
RISC-V Processor: first design

- Consider **subset** of RISC-V instructions:
 - R-type ALU instructions:
 - **add, sub, and, or, slt**
 - Memory instructions:
 - **lw, sw**
 - Branch instructions:
 - **beq**

Architectural State Elements

Determines everything about a processor:

- **Architectural state:**
 - 32 registers
 - PC
 - Memory (split into instructions, data)
- How to connect and control these elements?



Single-Cycle RISC-V Processor

- Datapath (the components)
- Control (given current instruction, send signals to datapath)
- For each instruction:
 - At clock edge, fetch next instruction
 - Decode based on instruction type
 - Execute the instruction
 - Increment / update PC
- Choose long enough clock cycle so that this always completes before the next rising edge

Example Program

- Design datapath
- View example program executing

Example Program:

Address	Instruction	Type	Fields					Machine Language	
0x1000	L7: lw x6, -4(x9)	I	imm _{11:0}	rs1	f3	rd	op		
			1111111111100	01001	010	00110	0000011	FFC4A303	
0x1004	sw x6, 8(x9)	S	imm _{11:5}	rs2	rs1	f3	imm _{4:0}	op	
			0000000	00110	01001	010	01000	0100011	0064A423
0x1008	or x4, x5, x6	R	funct7	rs2	rs1	f3	rd	op	
			0000000	00110	00101	110	00100	0110011	0062E233
0x100C	beq x4, x4, L7	B	imm _{12,10:5}	rs2	rs1	f3	imm _{4:1,11}	op	
			1111111	00100	00100	000	10101	1100011	FE420AE3

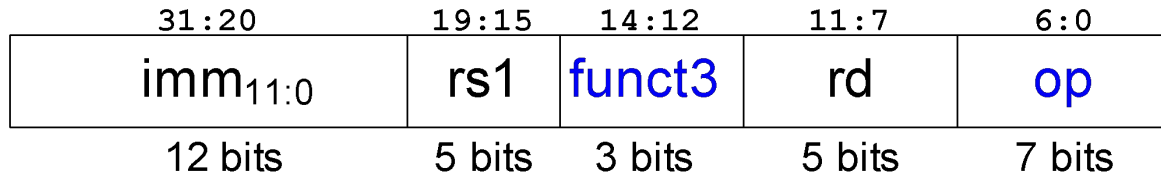
Single-Cycle RISC-V Processor

- **Datapath:** start with lw instruction

- **Example:** `lw x6, -4(x9)`

`lw rd, imm(rs1)`

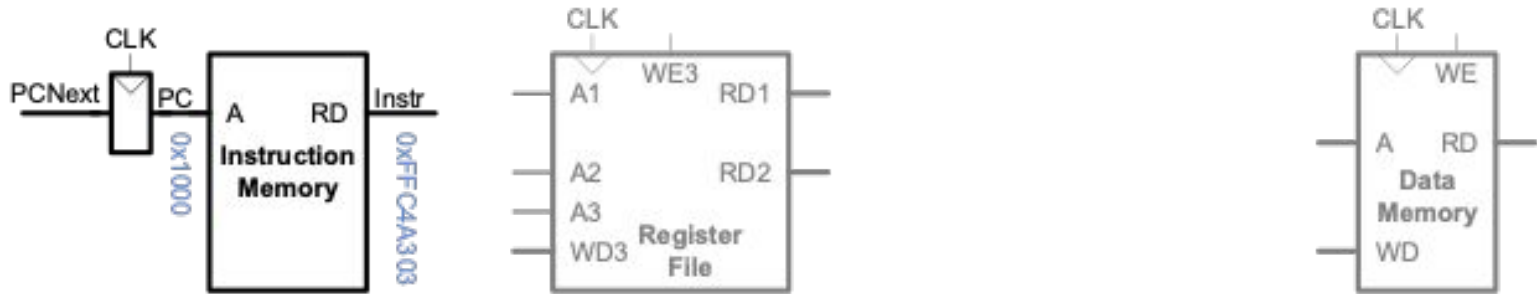
I-Type



Let's execute this instruction!!

Single-Cycle Datapath: lw fetch

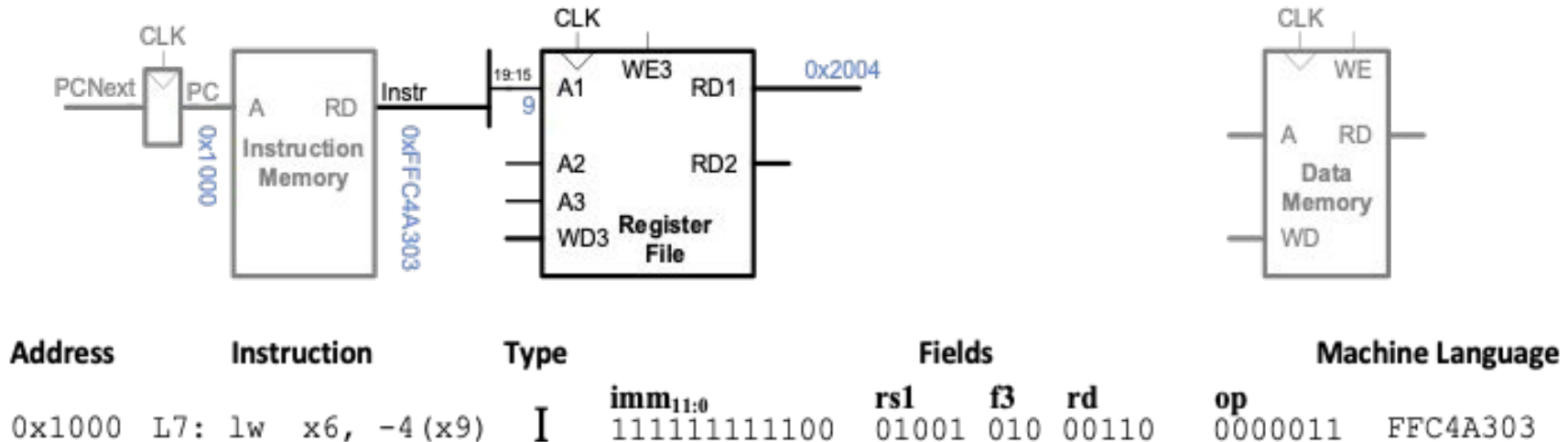
STEP 1: Fetch instruction



Address	Instruction	Type	Fields	Machine Language
0x1000	L7: lw x6, -4(x9)	I	<div><div>imm_{11:0}</div><div>111111111100</div><div>rs1</div><div>01001</div><div>f3</div><div>010</div><div>rd</div><div>00110</div></div>	<div><div>op</div><div>0000011</div><div>FFC4A303</div></div>

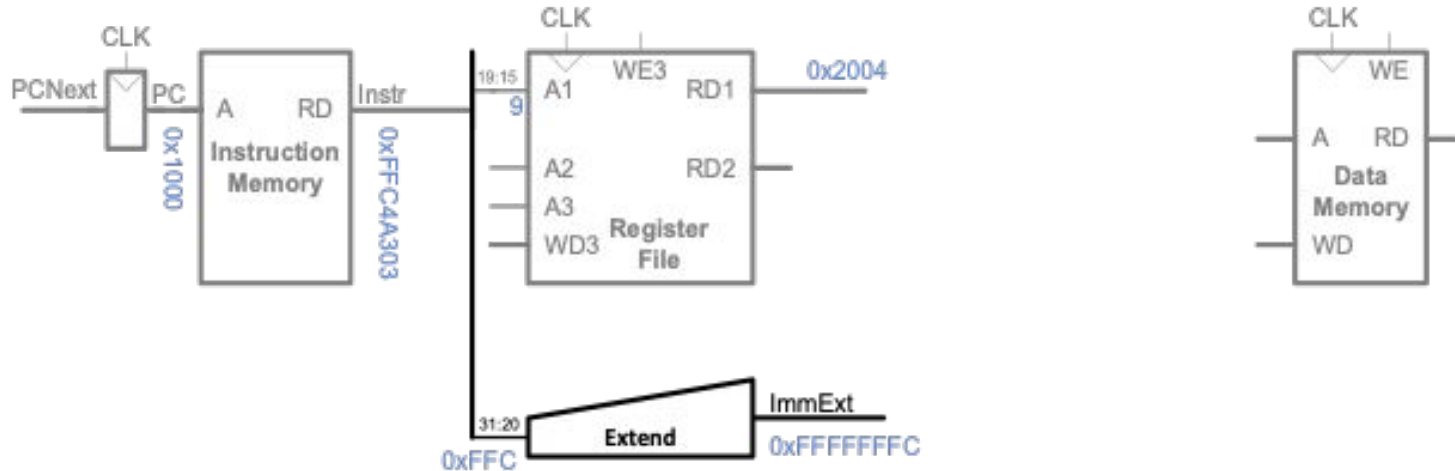
Single-Cycle Datapath: lw Reg Read

STEP 2: Read source operand (**rs1**) from RF



Single-Cycle Datapath: lw Immediate

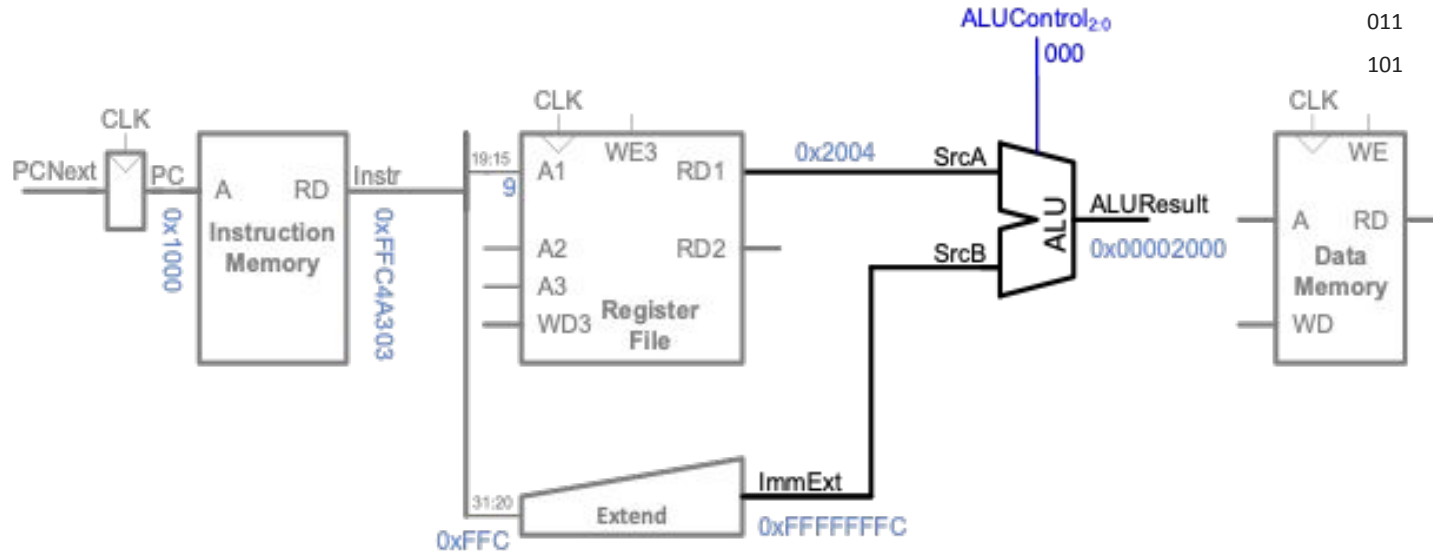
STEP 3: Extend the immediate



Address	Instruction	Type	Fields	Machine Language										
0x1000	L7: lw x6, -4(x9)	I	<table><tr><th>imm_{11:0}</th><th>rs1</th><th>f3</th><th>rd</th><th>op</th></tr><tr><td>111111111100</td><td>01001</td><td>010</td><td>00110</td><td>0000011</td></tr></table>	imm _{11:0}	rs1	f3	rd	op	111111111100	01001	010	00110	0000011	FFC4A303
imm _{11:0}	rs1	f3	rd	op										
111111111100	01001	010	00110	0000011										

Single-Cycle Datapath: lw Address

STEP 4: Compute the memory address

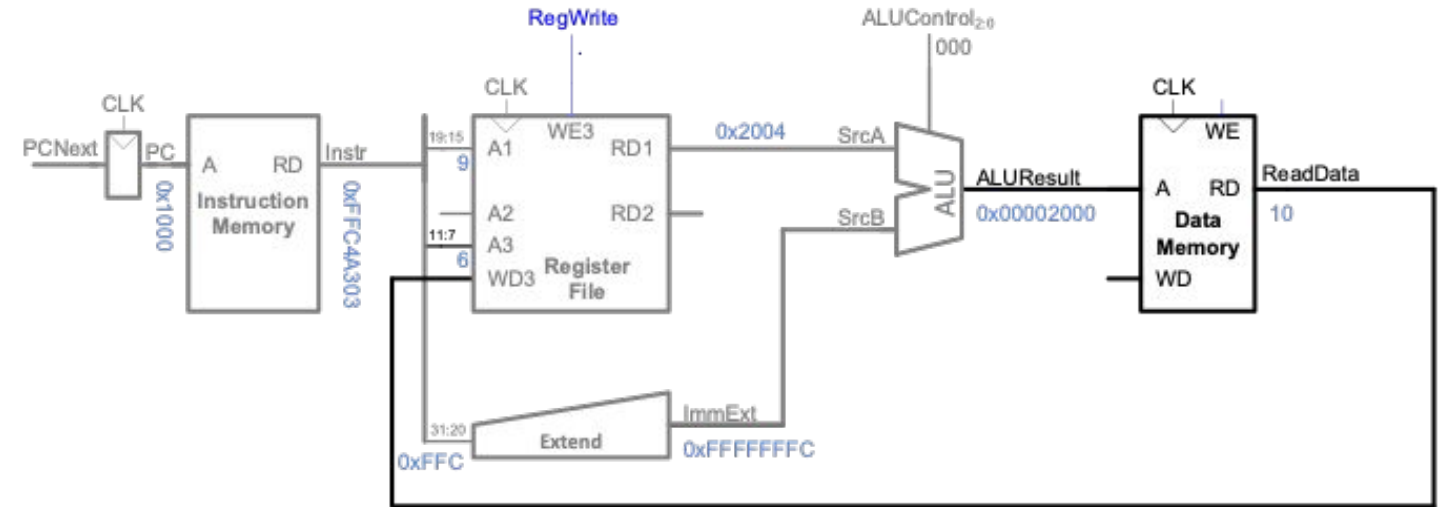


ALUControl _{2:0}	Function
000	add
001	subtract
010	and
011	or
101	SLT

Address	Instruction	Type	Fields			Machine Language	
0x1000	L7: lw x6, -4(x9)	I	imm _{11:0}	rs1	f3 rd	op	FFC4A303
			111111111100	01001	010 00110	0000011	

Single-Cycle Datapath: lw Mem Read

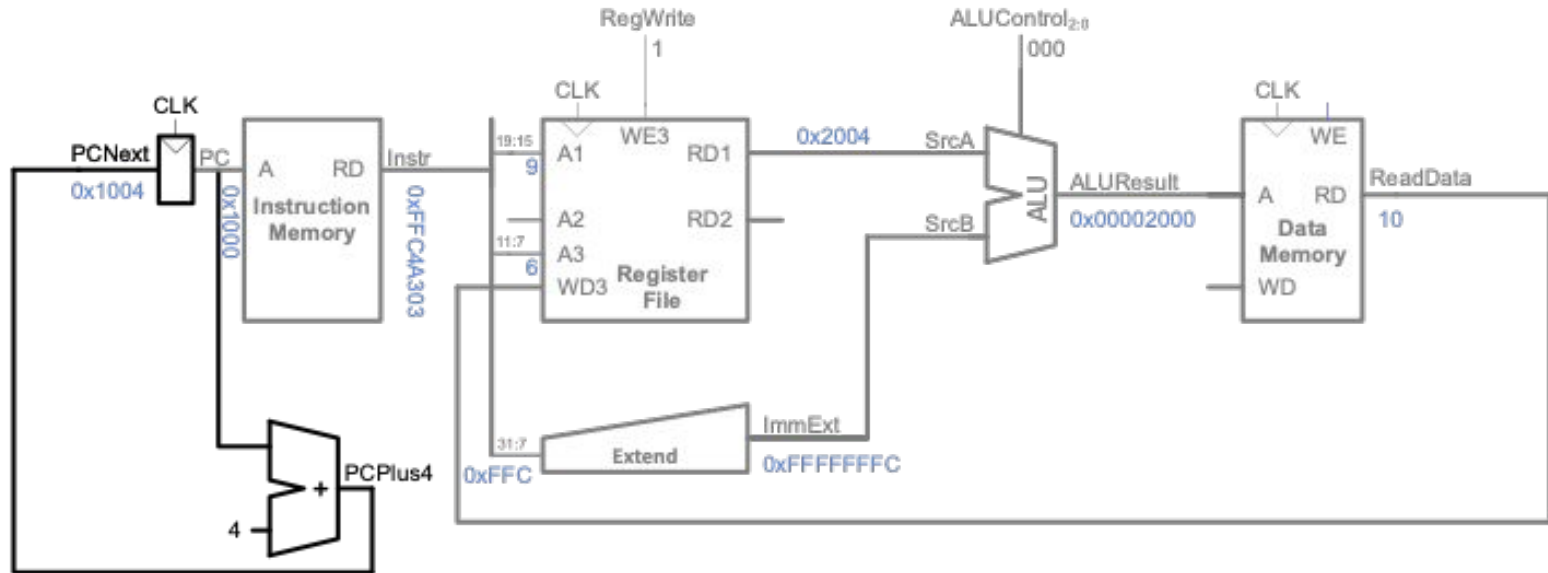
STEP 5: Read data from memory and write it back to register file



Address	Instruction	Type	Fields	Machine Language										
0x1000	L7: lw x6, -4(x9)	I	<table><tr><th>imm_{11:0}</th><th>rs1</th><th>f3</th><th>rd</th><th>op</th></tr><tr><td>111111111100</td><td>01001</td><td>010</td><td>00110</td><td>0000011</td></tr></table>	imm _{11:0}	rs1	f3	rd	op	111111111100	01001	010	00110	0000011	FFC4A303
imm _{11:0}	rs1	f3	rd	op										
111111111100	01001	010	00110	0000011										

Single-Cycle Datapath: PC Increment

STEP 6: Determine address of next instruction



Address	Instruction	Type	Fields				Machine Language	
			imm _{11:0}	rs1	f3	rd	op	
0x1000	L7: lw x6, -4(x9)	I	111111111100	01001	010	00110	0000011	FFC4A303

(Time Permitting) **Check-In:** What's my clock cycle?

- With a partner, on a notecard:
 - For the **lw** instruction we just walked through, what is the approximate delay required to complete the instruction?
 - Use the provided table of delays, and write down what questions you have about what to include in your calculation.
 - Do you expect our other instructions (**add**, **sub**, **beq**, etc.) to take more or less time to execute than **lw**?
 - Based on your finding, how long of a clock cycle would you recommend we use for our single-cycle processor?

Element	Parameter	Delay (ps)
Register clock-to-Q	t_{pcq_PC}	40
Multiplexer	t_{mux}	30
AND, OR gate	t_{AND-OR}	20
ALU	t_{ALU}	120
Decoder (Control Unit)	t_{dec}	25
Extend unit	t_{ext}	35
Memory read	t_{mem}	200
Register file read	t_{RFread}	100
Register file setup	$t_{RFsetup}$	60

Wrap-Up November 12



- Coming up next!
 - Constructing your processor!
 - Solidifying your understanding of the program memory address space
 - Keeping both the programmer AND microarchitectural perspective in mind
- Logistics, Reminders
 - TA help 7-9PM on Sundays, Tuesdays, Thursdays in C107
 - LP Office hours M 9-10:30AM, Th 2:30-4PM
 - Weekly Exercises due Friday 5PM
 - Lab 6 due **Today 10PM**
 - No pre-lab for tomorrow, but important weekly reading will be posted
- FEEDBACK
 - <https://forms.gle/5Aafcm3iJthX78jx6>