



Introduction to Information Systems

- Understanding the digital world

3 Inside the CPU

Liang Zhao

ILA, Doshisha University

12001102, Fall, 2024



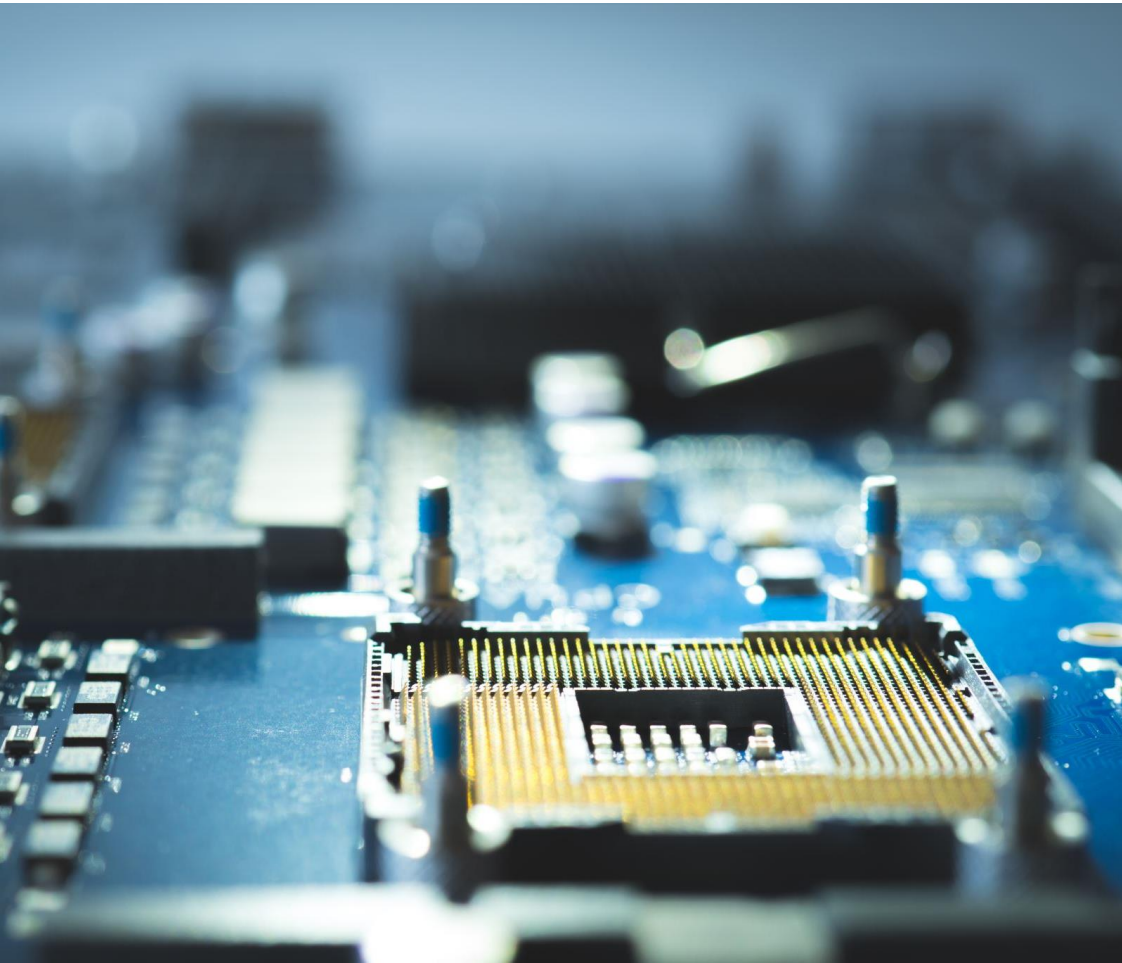
<https://aw.gsais.kyoto-u.ac.jp/liang/lectures>



Today's schedule

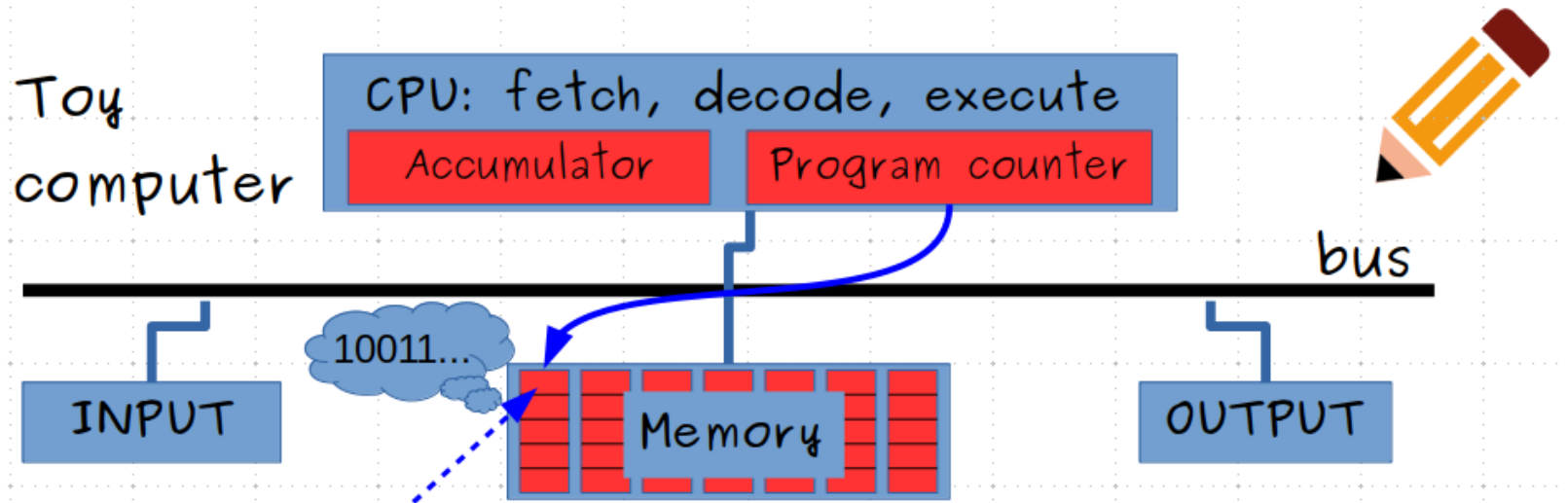
- **Review of Mini test #2 (5')**
- **Mini test #3 (25')**
- **CPU: Review of Chapter 3 (10')**
- **Coding with the Toy Machine (50')**

CPU (Central Processing Unit)



CPU performs:

- Arithmetic: +, -, x, /, etc.. (like a calculator with more but limited functions)
- Fetch/store/operate data from/to/in the memory (RAM)
- Coordinate input, output and others
- Compare numbers and decide the next to do
- Note: Instructions and data are in the RAM.



Toy Machine & instructions

label	instruction	description
	get	get a number from keyboard into accumulator
L	print	print contents of accumulator
	load Val	load accumulator with Val (Val unchanged)
	store M	store contents of accumulator into memory location called M
	add Val	add Val to contents of accumulator (Val unchanged)
	sub Val	subtract Val from contents of accumulator (Val unchanged)
	goto L	go to instruction labeled L
	ifpos L	go to instruction labeled L if accumulator is \geq zero
	ifzero L	go to instruction labeled L if accumulator is zero
	stop	stop running
M	Num	before program runs, set this memory location (called M) to Num

Ex. 1: First program

- Go to <http://www.cs.princeton.edu/courses/archive/fall18/cos109/toysim.html>
- Input the program into the left box. Notice a space is required before the first letter (read the instructions).
- Click "Run" and input some **number** (e.g., 367) when asked, then check the output in the right box.
- In case of error, revise your program, spell, space, input, etc. (called debug).

COS 109 Toy Machine Simulator - Mozilla Firefox

COS 109 Toy Machine Simulator

(You must have Javascript enabled.) Type your program in the left window. Labels must start in the first column and operators like GET or ADD must start anywhere but the first column, i.e., there must be one or more spaces before them. The simulator does not distinguish upper case from lower case, and is not robust, so be sure to spell instructions correctly and format code carefully.

Push RUN to run your program. A dialog box will appear when a GET is executed, and output from PRINT will appear in the right window. The simulator will stop if you Cancel a GET or don't enter anything.

```
1 GET
2 PRINT
3 STOP
4
5
6
7
8
9
10
11
12
13
14
15
```

Accumulator:

Run Clear output

Syntax reminder

```
get      get a number from keyboard into accumulator
print    print contents of accumulator
```

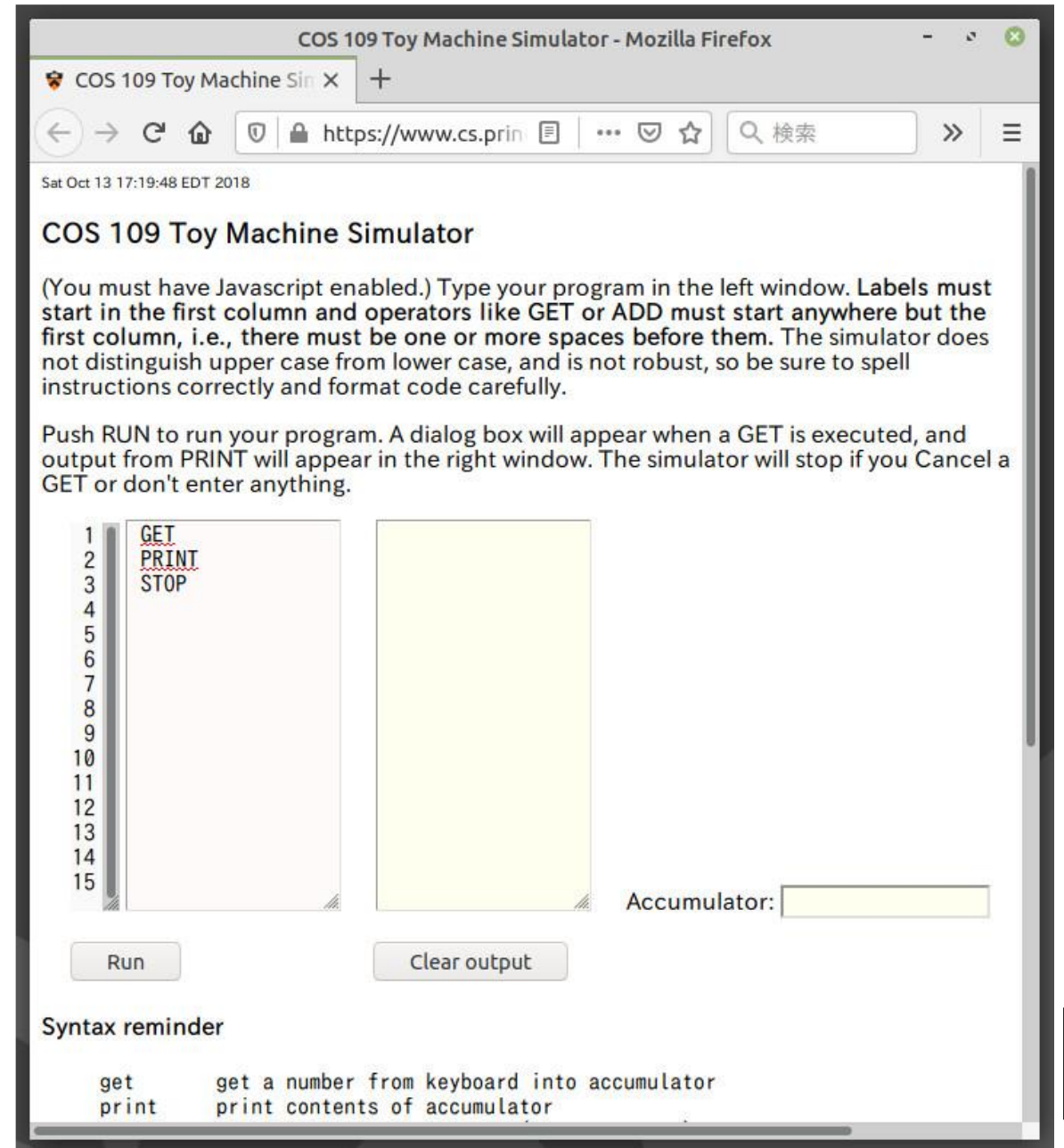
Ex. 1: Explained

1. `_GET` -> read some number ¹
2. `_PRINT` -> print it to the output ²
3. `_STOP` -> end the program

¹ The number is store in the accumulator.

² It prints contents of the accumulator.

Notice: "`_`" shows an invisible space.



The screenshot shows a web browser window titled "COS 109 Toy Machine Simulator - Mozilla Firefox". The address bar shows the URL "https://www.cs.prin...". The page content includes the title "COS 109 Toy Machine Simulator" and a paragraph of instructions: "(You must have Javascript enabled.) Type your program in the left window. Labels must start in the first column and operators like GET or ADD must start anywhere but the first column, i.e., there must be one or more spaces before them. The simulator does not distinguish upper case from lower case, and is not robust, so be sure to spell instructions correctly and format code carefully." Below this is another paragraph: "Push RUN to run your program. A dialog box will appear when a GET is executed, and output from PRINT will appear in the right window. The simulator will stop if you Cancel a GET or don't enter anything." The interface features a code editor on the left with a line number column (1-15) and the code: `_GET`, `_PRINT`, and `_STOP`. To the right of the code editor is a yellow output window. Below the code editor is a "Run" button. To the right of the output window is a "Clear output" button. Further right is an "Accumulator:" label followed by a yellow input field. At the bottom, a "Syntax reminder" section lists: `get` get a number from keyboard into accumulator and `print` print contents of accumulator.

Ex. 2: Run the next program and find what it does.

1. `_GET`
2. `_STORE M`
3. `_ADD M`
4. `_PRINT`
5. `_STOP`
6. `M`

Note: `STORE M` copies the value in the accumulator into a space named `M`, whereas `ADD M` adds the two values in `M` and in the accumulator and puts the result into the accumulator.

Ex. 3: Run the next program and find what it does.

1. `_GET`
2. `_STORE A`
3. `_GET`
4. `_ADD A`
5. `_PRINT`
6. `_STOP`
7. `A`

Note: `GET` reads a number into the accumulator (and overwrite the old content).

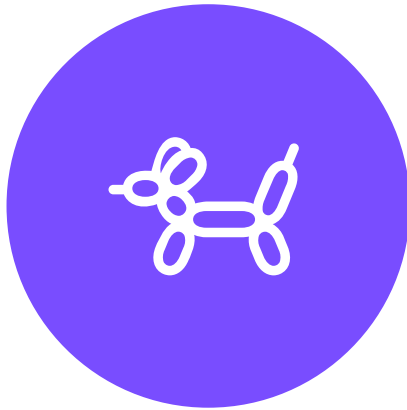
Coding task

Write a program for the Toy Machine that **reads a number A from the user** and calculates $3 \times A$ (that is, $A + A + A$), then prints it out.

Optional task (1 bonus point)

Write a program that reads an arbitrary number A from the user, calculates and prints $100 \times A$ (100 times of A). A **smart** program is expected :-). You can try it after the lecture and submit it to me by email before Oct 14th.

Homework



WATCH TWO VIDEOS



RE-READ CHAPTER 3

YouTube -> Crash course -> Computer Science (You are not expected to understand everything)

#8 <https://www.youtube.com/watch?v=zlTgXvg6r3k> & #9 <https://www.youtube.com/watch?v=rtAIC5JIU40>