

# The Hack Computer Platform

The screenshot shows the CPU Emulator (1.4b3) interface. The ROM window displays assembly code, with instruction 15 highlighted. The RAM window shows memory addresses and values. The registers window shows PC at 15 and A at 17. The ALU window shows D Input at 25 and M/A Input at 17184, resulting in an ALU output of 17184. Callouts identify the screen, data memory, keyboard enabler, registers, and ALU.

ROM	Asm	RAM
0	@0	0 50
1	D=M	1 0
2	@23	2 0
3	D; JLE	3 0
4	@16	4 0
5	M=D	5 0
6	@16384	6 0
7	D=A	7 0
8	@17	8 0
9	M=D	9 0
10	@17	10 0
11	A=M	11 0
12	M=-1	12 0
13	@17	13 0
14	D=M	14 0
15	@32	15 0
16	D=D+A	16 25
17	@17	17 17184
18	M=D	18 0
19	@16	19 0
20	MD=M-1	20 0
21	@10	21 0
22	D; JGT	22 0
23	@23	23 0
24	0; JMP	24 0
25		25 0
26		26 0
27		27 0
28		28 0

Registers: PC = 15, A = 17

ALU: D Input = 25, M/A Input = 17184, ALU output = 17184

# Instruction memory

The screenshot shows the CPU Emulator interface with the following components and callouts:

- ROM Asm:** A list of instructions. The instruction at address 15, `@32`, is highlighted in yellow. A red circle highlights the dropdown menu above it.
- RAM:** A list of memory addresses and values. Address 17 contains the value 17184.
- PC (Program Counter):** A box at the bottom left showing the value 15, with a red arrow pointing to it.
- ALU:** A component with inputs and outputs. The D Input is 25, and the M/A Input is 17184. The ALU output is 17184.

Callouts provide additional information:

- The loaded code can be viewed either in binary, or in symbolic notation (present view)** (points to the ROM Asm dropdown menu).
- Instruction memory (32K): Holds a machine language program** (points to the RAM area).
- Next instruction is highlighted** (points to the highlighted instruction at address 15).
- Program counter (PC) (16-bit): Selects the next instruction.** (points to the PC box).

# Data memory (RAM)

**CPU Emulator (1.4b3) - G:\examples\Rect.asm**

File View Run Help

Slow Fast Animate: Program flow View: Screen Format: Decimal

ROM	Asm
0	@0
1	D=M
2	@23
3	D; JLE
4	@16
5	M=D
6	@16384
7	D=A
8	@17
9	M=D
10	@17
11	A=M
12	M=-1
13	@17
14	D=M
15	@32
16	D=D+A
17	@17
18	M=D
19	@16
20	MD=M-1
21	@10
22	D; JGT
23	@23
24	0; JMP
25	
26	
27	
28	

RAM	
0	50
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	25
17	17184
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC 15 A 17

**Data memory (32K RAM), used for:**

- General-purpose data storage (variables, arrays, objects, etc.)
- Screen memory map
- Keyboard memory map

**Address (A) register, used to:**

- Select the current RAM location

OR

- Set the Program Counter (PC) for jumps (relevant only if the current instruction includes a jump directive).

# Registers

The screenshot shows a CPU Emulator window titled "CPU Emulator (1.4b3) - G:\examples\Rect.asm". The interface includes a menu bar (File, View, Run, Help), a toolbar with navigation and execution controls, and two main panels: ROM (Assembly) and RAM (Memory).

**ROM (Assembly):**

Address	Instruction
0	@0
1	D=M
2	@23
3	D; JLE
4	@16
5	M=D
6	@16384
7	D=A
8	@17
9	M=D
10	@17
11	A=M
12	M=-1
13	@17
14	D=M
15	@32
16	D=D+A
17	@17
18	M=D
19	@16
20	MD=M-1
21	@10
22	D; JGT
23	@23
24	0; JMP
25	
26	
27	
28	

**RAM (Memory):**

Address	Value
0	50
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
12	0
13	0
14	0
15	0
16	25
17	17184
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

**Registers and ALU:**

- PC (Program Counter):** 15
- A (Address Register):** 17
- D (Data Register):** 17184
- ALU:** D Input: 25, M/A Input: 17184, ALU output: 17184

**Annotations:**

- A yellow callout box titled "Registers (all 16-bit):" lists:
  - D:** Data register
  - A:** Address register
  - M:** Stands for the memory register whose address is the current value of the Address register
- An orange box labeled "M (=RAM[A])" points to the value 17184 in the RAM table at address 17.
- An orange box labeled "D" points to the value 17184 in the Data Register.
- An orange box labeled "A" points to the value 17 in the Address Register.

# Arithmetic/Logic Unit

The screenshot shows the CPU Emulator interface with the following components and annotations:

- ROM Asm:** A list of instructions. Instruction 14, `D=M`, is highlighted in red and labeled "Current instruction".
- RAM:** A memory table where address 17 contains the value 17184. This value is labeled "M (=RAM[A])".
- Registers:** The PC register is at 15, and the A register is at 17. The D register is shown with the value 17184.
- ALU:** A diagram showing the ALU operation. The D Input is 25, and the M/A Input is 17184. The ALU output is 17184.

**Arithmetic logic unit (ALU)**

- The ALU can compute various arithmetic and logical functions (let's call them  $f$ ) on subsets of the three registers  $\{M,A,D\}$
- All ALU instructions are of the form  $\{M,A,D\} = f(\{M,A,D\})$  (e.g.  $M=M-1$ ,  $MD=D+A$ ,  $A=0$ , etc.)
- The ALU operation (LHS destination, function, RHS operands) is specified by the current instruction.

# Loading a program

The screenshot shows the CPU Emulator (1.4b1) interface. The 'Load ROM' dialog box is open, showing a file list with 'Rect.hack' selected. The 'Load ROM' button is circled in red. A yellow callout bubble points to the dialog with the text: 'Navigate to a directory and select a .hack or .asm file.'

ROM	Asm
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

RAM	
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC: 0

# Loading a program

The screenshot shows a CPU Emulator window titled "CPU Emulator - D:\hack\instructor\Examples\rect\rect.bin". The interface includes a menu bar (File, View, Run, Help), a toolbar with navigation and execution icons, and control options for "Animate" (Program flow), "View" (None), and "Format" (Decimal). The main area is divided into several sections:

- ROM:** A table showing memory addresses 0 to 28 with their corresponding binary values. The first row (address 0) contains the value 0000000000000000.
- RAM:** A table showing memory addresses 0 to 28, all currently containing the value 0.
- PC (Program Counter):** A register showing the value 0.
- A (Accumulator):** A register showing the value 0.
- Keyboard:** A virtual keyboard icon and a text input field.
- D (Data Register):** A register showing the value 0.
- ALU:** A diagram of an ALU with two inputs: "D Input" and "M/A Input", both showing 0. The output is "ALU output", also showing 0.

A yellow callout bubble with a pointer to the ROM table contains the text: "Can switch from binary to symbolic representation".

# Running a program

The screenshot shows a CPU Emulator window titled "CPU Emulator - D:\hack\instructor\Examples\rect\rect.bin". The interface includes a menu bar (File, View, Run, Help), a toolbar with navigation buttons (back, forward, stop, home, end), a speed control slider (Slow to Fast), and dropdown menus for Animate (Program flow), View (None), and Format (Decimal). The main area is divided into ROM Asm, RAM, and a CPU window. The ROM Asm window shows assembly code, and the RAM window shows memory addresses and values. The CPU window shows registers PC and A. A keyboard icon is visible at the bottom.

2. Click the "run" button.

1. Enter a number, say 50.

3. To speed up execution, use the speed control slider

4. Watch here

**Program's description:** Draws a rectangle at the top left corner of the screen. The rectangle's width is 16 pixels, and its length is determined by the current contents of RAM[0].

**Note:** There is no need to understand the program's code in order to understand what's going on.

# Running a program

The screenshot shows a CPU Emulator window titled "CPU Emulator - D:\hack\Programs\rect.bin". The interface includes a menu bar (File, View, Run, Help), a toolbar with navigation buttons (back, forward, stop, etc.), a speed control slider (Slow to Fast), and dropdown menus for Animate (Program flow), View (None), and Format (Decimal). The main area is divided into ROM and RAM memory windows, a keyboard input area, and a large display area. The ROM window shows assembly code, with line 12 highlighted. The RAM window shows memory addresses, with address 17 highlighted containing the value 17536. The display area shows a small black rectangle in the top-left corner. Callouts provide instructions: 1. Enter a number, say 50. 2. Click the "run" button. 3. To speed up execution, use the speed control slider. 4. Watch here. A large callout at the bottom right provides a program description and a note.

2. Click the "run" button.

1. Enter a number, say 50.

3. To speed up execution, use the speed control slider

4. Watch here

**Program's description:** Draws a rectangle at the top left corner of the screen. The rectangle's width is 16 pixels, and its length is determined by the current contents of RAM[0].

**Note:** There is no need to understand the program's code in order to understand what's going on.