Project 4.A – Machine Code & CPU Operation

Due Friday, April 6 @ 11:59:59pm via Blackboard

# Machine Code Exercise

1. The table below contains a program written in *Hack* machine language. Decode the machine instructions into *Hack* assembly code. Write the assembly code for each instruction in the corresponding blank table cell.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Line | Machine Code | | | | | | | | | | | | | | | | Assembly Code |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  |
| 2 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |  |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |  |
| 4 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |  |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |  |
| 6 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |  |

1. Explain precisely what the last instruction does.
2. Explain briefly what the program does.
3. How many *A*­-instructions does the program have? \_\_\_\_\_\_\_\_

How many C­-instructions does the program have? \_\_\_\_\_\_\_\_

1. What is the effect of changing line 6 to 1110010011011000?

# CPU Operation

The *Hack* CPU contains an ALU, a Program Counter (PC), an address register (A), a data register (D). The *Hack* CPU has access to 24577 general purpose memory registers RAM[0]-RAM[24576].

*Hack* CPU instructions are in a temporary read-only memory space called ROM. CPU execution starts at ROM address 0. The Program Counter points to the ROM Address of the next instruction to be executed. The program counter either: a) increments by one after each instruction, b) points to the address in A when a *jump* condition is met.

Consider the following fragment of a Hack assembly language program.

|  |  |
| --- | --- |
| ***ROM Address*** | **Code** |
| 0 | @5 |
| 1 | D=M |
| 2 | @6 |
| 3 | D;JNE |
| 4 | @1 |
| 5 | M=M+1 |
| 6 | @2 |
| 7 | M=D+M |

1. Explain briefly what this program does.
2. Fill in the values of the program counter and registers AFTER each step of CPU execution but prior to executing the next instruction. Step 0 represents the initial state of the CPU and memory.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Step | PC | A | D | RAM[5] | RAM[1] | RAM[2] |
| 0 | 0 | 0 | 0 | 42 | 5 | 234 |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |