Name: Alexandra Burke
Section/Time: CS 220 1319
Date: 7/12/23

The A-instruction: symbolic and binary syntax

Semantics: Set the A register to *value*

Symbolic syntax:

@*value*

Where *value* is either:

- a non-negative decimal constant ≤ 32767 ($=2^{15}-1$) or
- a symbol referring to such a constant (later)

Example:

@21

Effect: sets the A register to 21

Binary syntax: 0vvvvvvvvvvvvvvv
For example: @21 (symbolic syntax – assembly code)
0000000000010101 (binary syntax – machine code)

The C-instruction: symbolic and binary syntax

Symbolic syntax: *dest* = *comp* ; *jump*

Binary syntax: 1 1 1 a c1 c2 c3 c4 c5 c6 d1 d2 d3 j1 j2 j3

comp		c1	c2	c3	c4	c5	c6
0		1	0	1	0	1	0
1		1	1	1	1	1	1
-1		1	1	1	0	1	0
D		0	0	1	1	0	0
A	M	1	1	0	0	0	0
!D		0	0	1	1	0	1
!A	!M	1	1	0	0	0	1
-D		0	0	1	1	1	1
-A	-M	1	1	0	0	1	1
D+1		0	1	1	1	1	1
A+1	M+1	1	1	0	1	1	1
D-1		0	0	1	1	1	0
A-1	M-1	1	1	0	0	1	0
D+A	D+M	0	0	0	0	1	0
D-A	D-M	0	1	0	0	1	1
A-D	M-D	0	0	0	1	1	1
D&A	D&M	0	0	0	0	0	0
D|A	D|M	0	1	0	1	0	1
a=0	a=1						

dest	d1	d2	d3	effect: the value is stored in:
null	0	0	0	The value is not stored
M	0	0	1	RAM[A]
D	0	1	0	D register
MD	0	1	1	RAM[A] and D register
A	1	0	0	A register
AM	1	0	1	A register and RAM[A]
AD	1	1	0	A register and D register
AMD	1	1	1	A register, RAM[A], and D register

jump	j1	j2	j3	effect:
null	0	0	0	no jump
JGT	0	0	1	if out > 0 jump
JEQ	0	1	0	if out = 0 jump
JGE	0	1	1	if out ≥ 0 jump
JLT	1	0	0	if out < 0 jump
JNE	1	0	1	if out ≠ 0 jump
JLE	1	1	0	if out ≤ 0 jump
JMP	1	1	1	Unconditional jump

Directions: Complete the translation from Assembly to Machine code for the following instructions.

1) @1

parts																
values																

2) @SCREEN //@________

parts																
values																

3) @KBD //@________

parts																
values																

4) @R5 //@________

parts																
values																

5) A=-1

parts																
values																

6) A=D

parts																
values																

7) A=M

parts																
values																

8) A=D&M

parts																
values																

9) D=A

parts																
values																

10) D=M

parts																
values																

11) D=D|A

parts																
values																

12) D=!D

parts																
values																

13) M=A

parts																
values																

14) M=D

parts																
values																

15) M=D-1

parts																
values																

16) M=D+A

parts																
values																

17) MD=M-D

parts																
values																

18) AD=M+1

parts																
values																

19) AMD=!M

parts																
values																

20) 0;JMP //0 is ________

parts																
values																

21) D;JGT

parts																
values																

22) D;JEQ

parts																
values																

23) D;JLE

parts																
values																

24) D;JNE

parts																
values																

Lab #6 Computer Architecture - Memory

Name: Alexandra Burke
Section/Time: ______________________
Date: ______________________

❑ Write in the decimal values for each address below, and convert to binary:

NO CALCULATOR! Use back as scratch paper

RAM16K minimum address (first address) = (0)$_{10}$														
[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

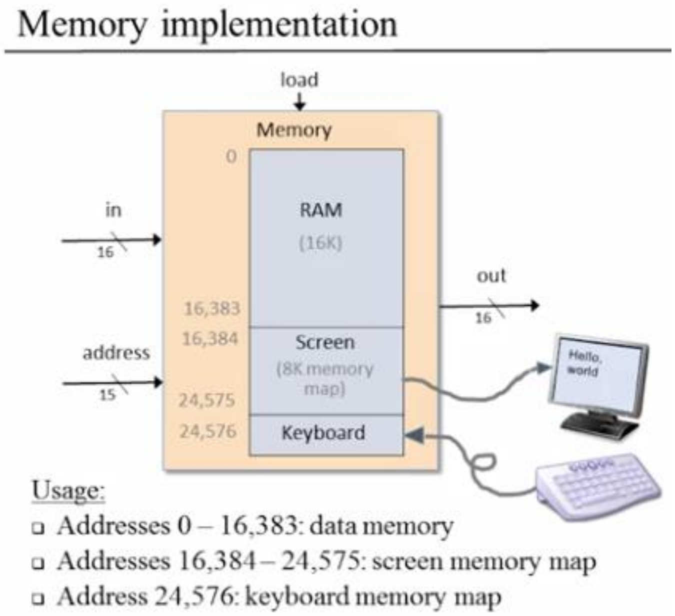
RAM16K maximum address (last address) = ($\boxed{16383}$)$_{10}$														
[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

SCREEN Pointer address (start of screen memory map) = (24575)$_{10}$														
[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]

Last possible address for screen memory map = ( )$_{10}$														
[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]

KBD Pointer address = (24575)$_{10}$														
[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]

Lab #6 Computer Architecture - Memory



- ❑ When interacting with the Memory chip, how can it differentiate between an address for the screen memory map, the keyboard register, and RAM16?

```
it can differentiate
between these by
determining the most
significant bits.
```

- ❑ What are the bus sizes for each chip in Memory.hdl?

```
load = 1, in = 16, out = =16, address = 15
```

- ❑ What does the **`load`** pin accomplish?

```
it is to load value or
data into a  register or
memory location and
writes the output.
```

- ❑ How is it used for the Memory chip?
  - ❑ Play through scenarios for the value of load and what SHOULD happen to each part of memory. How can the chip differentiate where load should go?

```
-load pin is used as a
control signal for the memory
chip for the write operation.

-if load = 1, val is loaded
into memory location
specified by address.
```

Lab #6 Computer Architecture - Memory

- ❑ Name the chip that will help you do the above: __MUX – multiplexer____

- ❑ The way keyboard input works is that the address **must be exactly 24,576** to output the value in the Keyboard register, for all other values it must output 0. How could you accomplish this with the chips we've built in previous topics?

- ❑ Draw and connect c~~hips that would ensure the~~ value above (should output 1 if true, 0 if false):