Name_____

EET 1131 Lab #14
Random Access Memory

7489 RAM Chip

The 7489 is a TTL random access memory chip. It is organized as a 16 x 4 RAM. To be sure you understand what this means, answer the following questions:

Since this is a 16 x 4 chip, how many addressable memory locations does it contain?_____

How many address bits are needed to address this many locations? _____

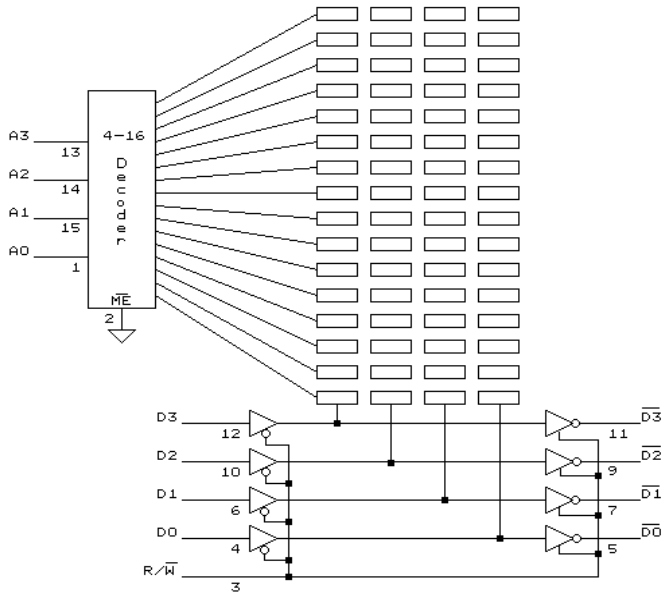
How many data bits are contained in each one these locations? _____

Therefore, what is the chip's bit capacity? (How many bits can the chip store?) _____

By today's standards, this chip's bit capacity is tiny; but it will serve to illustrate the general principles of how memory chips operate.

Functional Block Diagram

The block diagram shows what's inside this chip.



Note the following:

- The four data input pins are separate from the four data output pins; therefore the data pins are not bi-directional, as they are on some RAM chips.
- The data outputs are open-collector pins, so each one requires its own external pull-up resistor.
- The data outputs are also active-low, so each one requires an inverter for correct logic levels.

- The chip's mode of operation is determined by the logic level on the Memory Enable and Write Enable input pins, which are active-low. The Memory Enable pin enables or disables the entire chip. The Write Enable pin controls whether the chip is in Read mode or Write mode.

7489 Datasheet

1. On the course website you'll find a copy of the 7489's datasheet. **Using a straight edge**, copy the 7489's pin diagram and logic symbol below. Be sure to include each pin's name and number. Also be sure to include any markings that indicate special features of pins, such as active-low or open-collector.

   7489 pin diagram 7489 logic symbol

2. Use the 7489's logic symbol to answer the following questions:

   How many **input** pins does a 7489 have? _____

   Which one of the following statements is true?
   ___ All of the 7489's inputs are active-high.
   ___ All of the 7489's inputs are active-low.
   ___ Some of the 7489's inputs are active-high, and some are active-low.

   How many **output** pins does a 7489 have? _____

   Which one of the following statements is true?
   ___ All of the 7489's outputs are active-high.
   ___ All of the 7489's outputs are active-low.
   ___ Some of the 7489's outputs are active-high, and some are active-low.

Procedure

1) Breadboard a 7489 memory chip, as described below.
   a) Using wire of one color, connect the chip's four address input pins to the trainer's switches SW7 to SW4 (with the most significant bit on the left).
   b) Using wire of a second color, connect the chip's four data input pins to the trainer's switches SW3 to SW0 (again with the MSB on the left).
   c) Since the 7489's data outputs are open-collector outputs, connect a 10 kΩ pull-up resistor between each data output pin and V_{CC}. (Each output pin needs its own pull-up resistor.)
   d) Using wire of a third color, run each data output through an inverter (since the chip's data outputs are active-low) and then to an LED on the trainer.
   e) Permanently enable the chip by tying the Memory Enable pin to the correct level.
   f) Using wire of a fourth color, connect the Write Enable pin to the trainer's pushbutton P_A so that the chip will read when the button is not pressed and write when the button is pressed. (You'll need to decide which of the pushbutton's terminals to use: A or $\overline{A}$.)

2) The table below shows the data that we wish to write into the RAM at each location. (The dollar sign is a common way of indicating hex. The dollar sign means the same thing as a subscript 16. So, for example, we can write **$1C3** instead of **$1C3_{16}$**. They mean the same thing.) For some locations, the data is given in binary, while for other locations the data is given in hex. Fill in all blank spaces in the Binary and Hex columns. In other words, convert each binary data value to hex, and convert each hex data value to binary.

Data Table #1: Values to be Written into Memory

Hex Address	Data	
	Binary	Hex
$0	0111	
$1		$3
$2		$C
$3	1000	
$4	0010	
$5	1111	
$6		$A
$7	0001	
$8	1001	
$9		$D
$A	0100	
$B	1011	
$C	0101	
$D		$6
$E	1110	
$F	0000	

3) Set all four address pins LOW. By doing this, you have selected memory location $0.
4) Set the four data input pins equal to the value from the table that we wish to store in the selected address.
5) Press the pushbutton to force the read/write pin LOW, storing the data. Then release the pushbutton, letting the read/write pin return to a HIGH state.

6) Repeat Steps 3 through 5 for the fifteen other memory locations.
7) Go back and read all locations to make sure they accepted the data correctly, filling in the table below. The values you read here should be the same as the ones you wrote above. Change any that are not correct by rewriting the correct data.

Data Table #2: Values Read from the Memory

Hex Address	Contents
$0	
$1	
$2	
$3	
$4	
$5	
$6	
$7	
$8	
$9	
$A	
$B	
$C	
$D	
$E	
$F	

8) Call me over to check your work. _____
9) Turn the power off and back on. Read all locations again, filling in the table below.

Data Table #3: Values Read from the Memory after Powering Off & On

Hex Address	Contents
$0	
$1	
$2	
$3	
$4	
$5	
$6	
$7	
$8	
$9	
$A	
$B	
$C	
$D	
$E	
$F	

10) You're finished with this circuit and can take it apart.

Multisim Circuit #1: 2k x 8 RAM

Now let's use Multisim to simulate the sort of thing you did above, on two larger memory chips. The first memory chip is Multisim's 2k x 8 RAM. To find it in Multisim's component bin, go into the **Misc Digital** group, then into the **TIL** family, and you should see the **2K8RAM** listed as the first component. Place one of these on the Multisim workspace.

Before you wire up the chip, let's note that the creators of Multisim were mistaken to call this a 2k x 8 chip. To see why, answer the following questions:

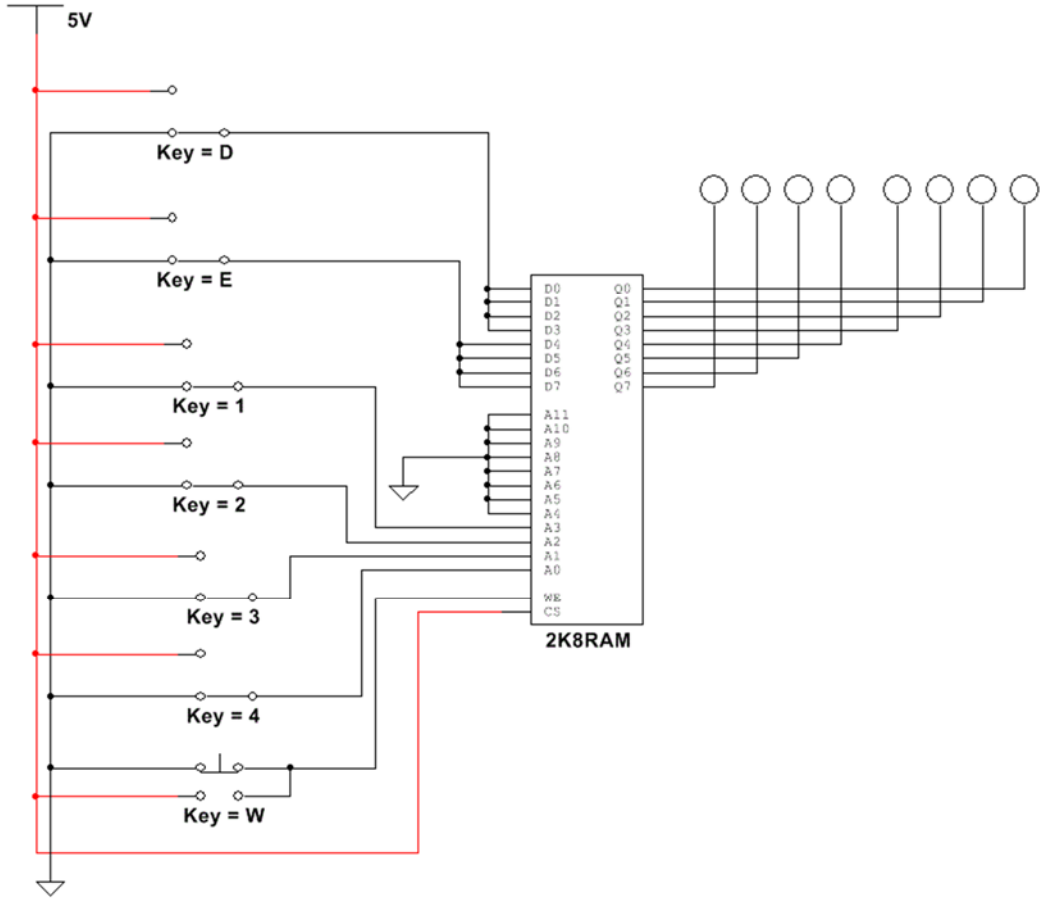
How many address pins does the chip have? _____

With that many address pins, how many memory locations can we address?_____

So what should this chip correctly be called, instead of 2k x 8? _____

What is the chip's bit capacity? _____

Now build (in Multisim) the circuit shown below.



Note the following points about this circuit:

- We're permanently enabling the chip by tying its CS (Chip Select) input HIGH.

- Although this chip contains thousands of memory locations, we're ignoring most of these by tying address pins A4 through A11 low. Since we're only using address pins A0 through A3, we'll only be able to access the chip's first _____ memory locations. (**Fill in the blank in the previous sentence**.)

- Like the 7489 chip you used above, this chip has separate data input pins and data output pins, instead of combined data input/output pins.

- We're wiring half of the data input pins to one data switch, and half to another data switch. Depending on the settings of these two switches, we have four possible values at the chip's data inputs:
  - 00000000 (which is equal to hex $00)
  - 00001111 (= $0F)
  - 11110000 (= $F0)
  - 11111111 (= $FF)

Next, simulate the circuit, and use the switches to load the following data into the first 16 memory locations. **Don't stop the simulation until I check your work below.** Stopping simulation will cause the data in the memory to be lost, so you'll have to re-enter it.

Data Table #4

Hex Address	Hex Data
$0000	**$00**
$0001	**$0F**
$0002	**$F0**
$0003	**$FF**
$0004	**$00**
$0005	**$0F**
$0006	**$F0**
$0007	**$FF**
$0008	**$00**
$0009	**$0F**
$000A	**$F0**
$000B	**$FF**
$000C	**$00**
$000D	**$0F**
$000E	**$F0**
$000F	**$FF**

After you write the data, read each location to be sure that the data was accepted. Then call me over to check your work.

_____