# Why?

So far, you've used registers to build stateful circuits, like accumulators. However, having studied assembly programming, you also know that programs depend on a powerful abstraction of state: an interface that lets the program use data (i.e., an address) to refer to other data (such as an integer, or an object). The *addressable memory*.

In this activity, you'll learn to distinguish addressable memories by their parameters and to predict the behavior of an addressable memory.

# Model 1: Readable memory

Consider the digital logic component consisting of two 8-bit registers and a multiplexor (mux).

|  |  |
| --- | --- |
| **Inside of component** | **Outside of component** |
|  | Select  Output |

1. What is the data width of the mux?

8

1. What is the number of bits in the select input of the mux?

1

1. Describe the output (Output) in terms of the input (Select) and the values held in RegisterA and RegisterB.

If Select is 0, Output is RegisterA. If Select is 1, Output is RegisterB.

# Read This!

We can think of this circuit as a two-element ***addressable memory*** (like the array of bytes abstraction). Each element is a byte stored in a register. We've relabeled the Select and Output signals below.

|  |  |
| --- | --- |
| **Inside of component** | **Outside of component** |
|  | Addressable memory  ReadData  ReadAddress |

1. Why is ReadData an appropriate name for that signal?

Because we want to know what the data is at an address.

1. Why is ReadAddress an appropriate name for that signal?

Because you are looking for what is being stored at that address.

1. What would be the ReadAddress if I wanted to read the first byte? The second byte?

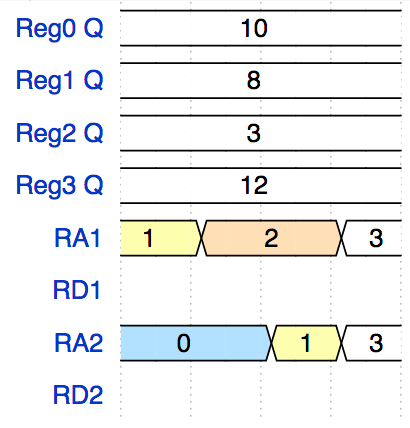
# Read This!

Together, the ports ReadAddress and ReadData are known as a ***read port*** because they are both needed to read a single element of the memory. A memory with 2 read ports would be able to read two different elements at the same time. Let's say that the ***width*** (W) is the number of bits per element and the ***length*** (L) is the number of elements.

1. Characterize the addressable memory shown in Model 1 by giving values for

* L=2
* W=8
* number of read ports=1

1. Suppose there is a W=8, L=4, 2 read ports memory. Fill in RD1 (ReadData1) and RD2 (ReadData2) for the following waveform. RA1 and RA2 are the ReadAddress of port 1 and 2.





1. In the #8 example, how many bits are there for each input/output?

* RA1 - 2
* RD1 - 8
* RA2 - 2
* RD2 - 8

1. What is the relationship between L, W, bitwidth of RA, and bitwidth of RD?
2. Based on what you observed in #8, why might you need a memory with more than 1 read port?

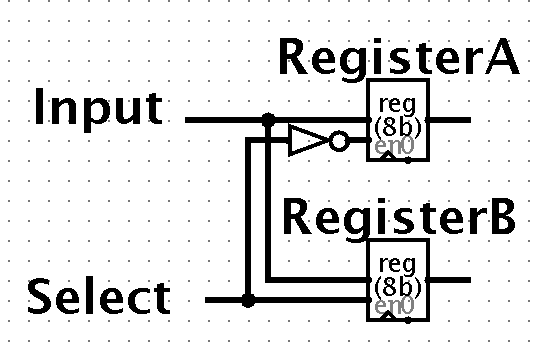
# Exercises

1. Suppose our computer has a readable memory containing a total of 256KiB (1 KiB = 210 bytes). Give two possible configurations (L, W) of such a memory and the bitwidths of their input RA and output RD.

# Model 2: Writeable memory

Consider this digital logic component consisting of two 8-bit registers and inputs Input and Select. The clock signal is omitted for clarity, but you can assume a clock is attached to both registers.

|  |  |
| --- | --- |
| **Inside of component** | **Outside of component** |
|  |  |

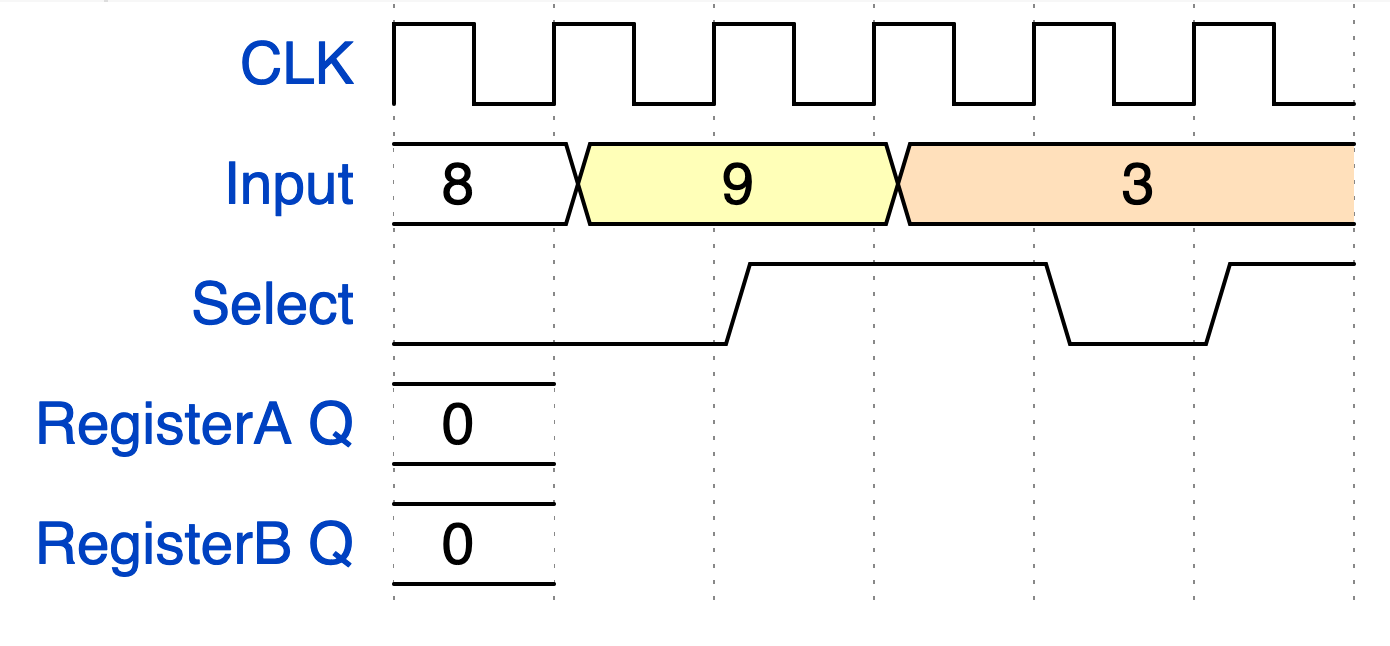


Select

Input

Legend: When the Enable input of the register (on the left side labeled as "en") is 1, the register acts normally, capturing the value at D on the rising edge of the clock. When Enable=0, the register just keeps its current value.

1. Draw the value of the Q outputs of the registers given the following Input signal.



1. **Summarize:** what does the Select input do?

# Read This!

A ***write port*** consists of two inputs, the ***WriteData*** and the ***WriteAddress***.

Here is Model 2 with labels on the inputs.

|  |  |
| --- | --- |
| **Inside of component** | **Outside of component** |
|  | Writeable  Addressable Memory  WriteAddress  WriteData |

1. Based on your answers above, why is **WriteAddress** and good name for that input?
2. Characterize the addressable memory shown in Model 2 by giving values for

* L
* W
* number of write ports

1. Suppose we built a writeable memory with W=8, L=4, 1 write port. How many bits are needed for

a) the WriteAddress input?

b) the WriteData input?

1. Suppose we built a writeable memory with W=16, L=4, 1 write port. How many bits are needed for

a) the WriteAddress input?

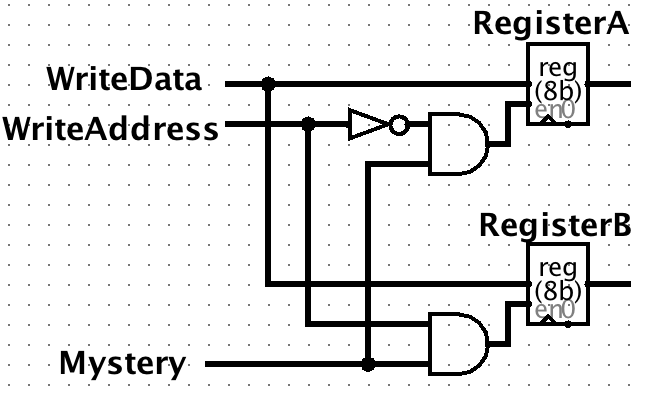
b) the WriteData input?

# Exercises

1. Combine what you learned about reading memory in Model 1 with writing memory in Model 2. Suppose we built a readable/writeable memory with W=20, L=64, 3 read ports, and 1 write port. List all the inputs and outputs this memory would have, along with the number of bits for each.

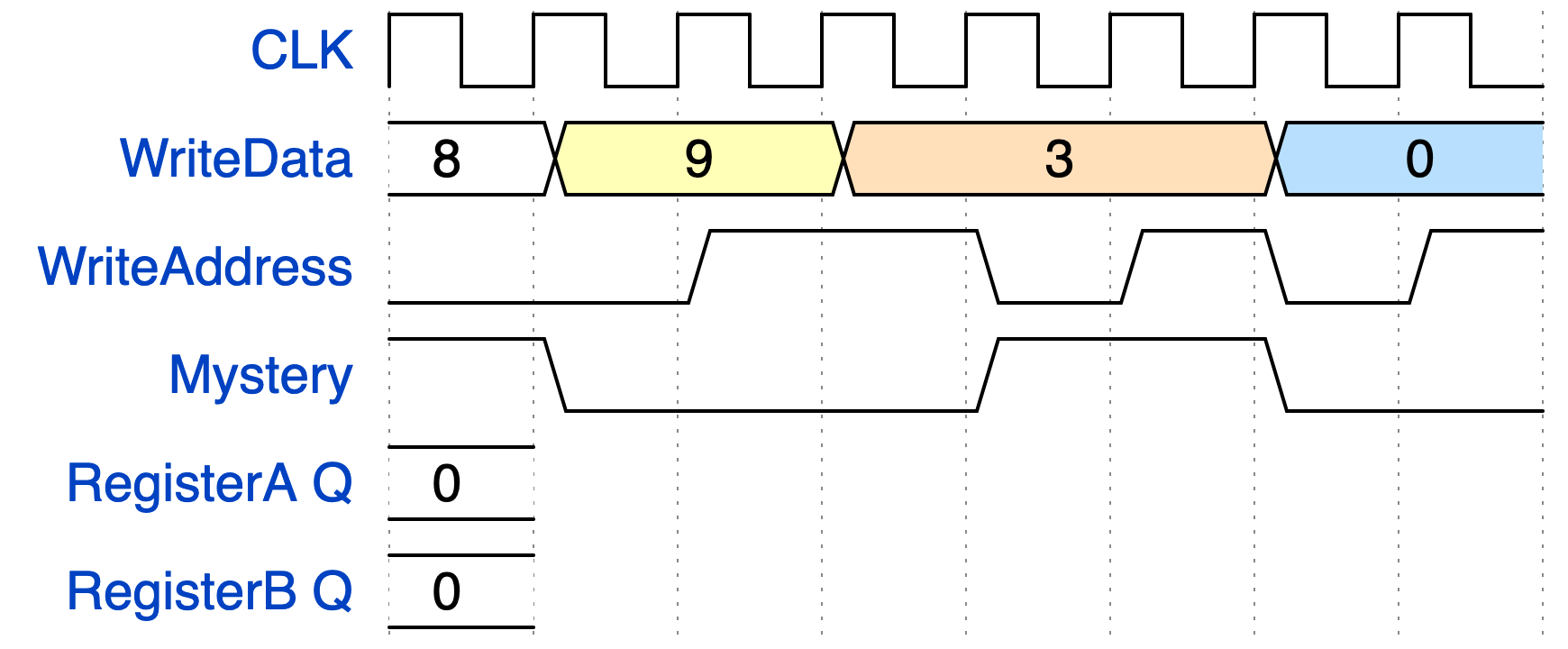
# Model 3: conditionally-writeable memory

Consider this adaptation of Model 2’s writeable memory, consisting of two 8-bit registers and inputs WriteData, WriteAddress, and Mystery. The clock signal is omitted for clarity, but you can assume a clock is attached to both registers.



Legend: When the Enable input of the register (on the left side labeled as "en") is 1, the register acts normally, capturing the value at D on the rising edge of the clock. When Enable=0, the register just keeps its current value.

1. Draw the value of the Q outputs of the registers given the following input signals.



1. **Summarize:** what does the Mystery input do?

# Read This!

A ***write enable*** tells the memory whether or not to write to the memory.

1. Why might a *write enable* signal be useful for a writeable memory?

# Appendix:

Wavedrom starters; go to <https://wavedrom.com/editor.html> and modify

**model1**

{signal: [

{name: 'Reg0 Q', wave: '2...', data: '10'},

{name: 'Reg1 Q', wave: '2...', data: '8'},

{name: 'Reg2 Q', wave: '2...', data: '3'},

{name: 'Reg3 Q', wave: '2...', data: '12'},

{name: 'RA1', wave: '34.2', data: '1 2 3'},

{name: 'RD1', wave: ''},

{name: 'RA2', data: '0 1 3', wave: '5.32'},

{name: 'RD2'},

],

}

**model 2:**

{signal: [

{name: 'CLK', wave: 'p.....'},

{name: 'Input', wave: '23.4..', data: '8 9 3 0'},

{name: 'Select', wave: '0.1.01', data: '3'},

{name: 'RegisterA Q', wave: '2', data: '0'},

{name: 'RegisterB Q', wave: '2', data: '0'},

],

}

**model 3:**

{signal: [

{name: 'CLK', wave: 'p.......'},

{name: 'WriteData', wave: '23.4..5.', data: '8 9 3 0'},

{name: 'WriteAddress', wave: '0.1.0101', data: '3'},

{name: 'Mystery', wave: '10..1.0.'},

{name: 'RegisterA Q', wave: '2', data: '0'},

{name: 'RegisterB Q', wave: '2', data: '0'},

],

}