Arnav Goyal

Microprocessors & Microcomputers

Winter 2024 / ECE 3375

Contents

Preface

The purpose of this document is to act as a comprehensive note for my understanding on the subject matter. I may also use references aside from the lecture material to further organize my understanding, and these references will be listed here.

References

- Provided Course Material & Lecture Notes
- Digital Design with an Introduction to the Verilog HDL - 5e - M. Mano, D. Ciletti

Memory Mapping

This chapter will discuss the useful concept of **memory mapping**, which is a technique used to assign the memory address space to different physical devices.

Memory

A **memory device** is a construct capable of storing a large quantity of binary information. Memory devices are made up of smaller cells.

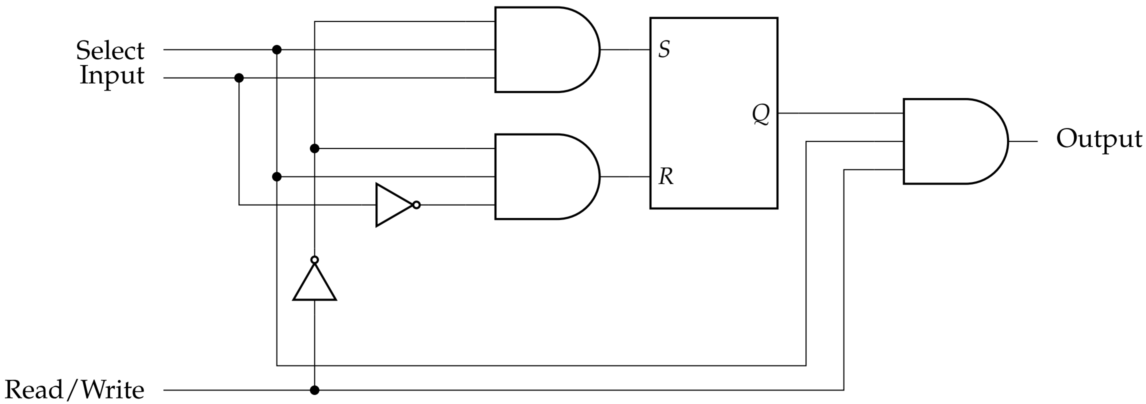
- There are two types of memory: **Random Access Memory (RAM)** & **Read Only Memory (ROM)**
- RAM supports the fundamental **read** and **write** operations; ROM just supports **read**.

The Memory Cell

Memory cells are the fundamental storage component present in a memory device. A 1-bit memory cell is called a **binary cell (BC)** and can be modeled by a D-latch and some additional gates. The binary cell has 4 ports: Input, Output, Read/Write, and Select.

- The select signal acts as the **enable** signal of the cell
- The Read/Write signal meaning corresponds to **read = 1** and **write = 0**
- The input accepts data to be stored within the cell
- The output provides the data currently stored within the cell

The equivalent logic for a binary cell looks like the following:



This BC can now be used to create RAM by grouping many of these together.

- BCs can be **grouped** by sharing select and read/write signals. This allows them to all be enabled at once, and read/write together.

- If we group n BCs together, we can write n bits of information and read n bits of information by supplying an appropriate read/write and select signal.
- If we make multiple groups, and find a way to select only one group at a time (through a decoder) we have essentially created RAM.

Note: This design is **not physically implementable!** When a BC is not selected it has an output of 0 (GND) onto the shared bus, we actually need **tri-state logic** to implement this with a shared bus. This example mainly serves the purpose of explaining the **equivalent logic** behind RAM, not actual implementation.

Random Access Memory (RAM)

Now that we have learned to create RAM, lets learn about it as a block.

For starters, the time it takes to transfer information to and from any **random desired location** is always the same.

Consider a RAM with k-address lines, which correspond to a maximum of 2^k memory addresses/words (# of groupings), and n-bits per word (# of BCs per grouping). This RAM has 4 ports with the following specifications. We denote this as a $2^k \times n$ RAM.

- n-bit input line
- n-bit output line

- k-bit address line
- 1-bit read/write line

Because we have n bits per group, and 2^k maximum groupings, our **memory capacity** is $2^k \times n$ bits.

There are a couple **optimizations** we can do to make this RAM better, such as using 2 dimensional decoding instead of 1, and combining the input/output data lines into a shared bus using tri-state logic. However, this is not a digital design class, thus it is covered within the textbook.

All RAM is also not the same, there are two types: **Static RAM (SRAM)** and **Dynamic RAM (DRAM)**.

- SRAM consists of internal latches that store the information, information is retained as long as power is provided (**volatile memory**).
- DRAM stores the information as electric charge on capacitors within the chip through MOSFETS. This charge slowly leaves over time, and needs to be **periodically refreshed**.

We created SRAM earlier, and in general it is easier to use and has shorter read-/write cycles. DRAM offers reduced power consumption and larger storage capacity within a single chip which is what makes it commonplace in industry.

Read Only Memory (ROM)

read-only-memory is memory device in which **permanent** binary information is stored.

- Once stored, it stays within the unit even after the power is turned off. (**non-volatile memory**)
- The k-inputs provide the address for the memory and the n-outputs provide the stored data-bits. Denoted as a $2^k \times n$ ROM.
- it is organized the same as the RAM we created, thus it has a maximum capacity of $2^k \times n$ bits.

Contrary to its naming convention, we actually can store data within the ROM. We do this by providing the **truth table** containing all possible memory addresses and stored data for each address, so that we can program it into the ROM.