Digital Logic Structures

Sequential Logic

K-Maps

Note 0.1

Karnaugh maps are an easy way to make a truth table and convert it into a circuit using the least number of gates.

K-Maps Setup

Karnaugh maps are setup using gray code, which means that only one variable changes between two-adjacent cells. If you examine the values across the top from left to right, or down the side from top to bottom, you'll also see that the activated bits follow a pattern like 00, 01, etc

Definition 0.1

A **gray code** is a binary numerical system that is ordered such that two subsequent values only differ in one bit. It is also known as reflected binary code because the codes are reflected in the first and last n/2 values

In this case, we differ by only 1 bit at time. So for example, 01 -> 10 is NOT a gray code, because 2 bits had to be flipped

K-Maps Grouping Rules

1. We want the biggest groups where the size of the groups are a power of 2
2. We want the least number of groups
3. We can build groups with adjacent cells including wrapping around corners

If something doesn't matter we can just put it to X and we can group with it if wanted.

You make the biggest groups possible, and you analyze the groups for what values don't change and use that to create a logical expression. From there, it is easy to turn the logical expression into circuits.

Level Triggered Logic

There are two types of sequential logic: **level triggered logic and edge triggered logic**. Both rely on the signals of a clock, which is a circuit component that oscillates between a 1 and 0 at a set frequency to help synchronize operations in a circuit.

The difference is when the output changes based on the input signal. In **level triggered logic**, when the clock has a 1 output, the circuit output will match the input, and when the clock has 0 output, the circuit output stays the same. Think of the changes happening when the clock is 1 and level.

> RS Latches, D Latches, and memory are all level triggered

Basic Storage Elements

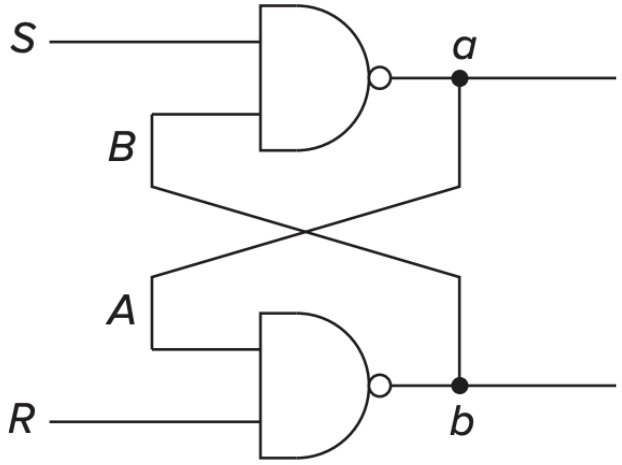
- The other kind of storage element are those that involve the storage of information and those that do not

The RS Latch

Definition 0.2

The **RS Latch** can store one bit of information, a 1 or a 0. Generally, two 2-input NAND gates are connected such that the output of each is connected to one of the inputs of the other. The other inputs are usually held to be zero.

Setting the latch to store a 1 is known as **setting** the latch, while setting the latch to store a 0 is referred to **resetting** the latch



Definition 0.3

The **quiescent** (or quiet) state of a latch is the state when the latch is storing a value, either 0/1, and nothing attempts to change that value.

This happens when S and R are both equal to 1. So as long as the inputs S and R remain as 1, the state of the circuit will not change.

Note 0.2

Setting the latch to a 1 or 0

The latch can be sent to 1 by momentarily setting S to 0, provided that we keep the value of R at 1. Similarly, we can set the patch to 0 by setting R to zero (known as clearing or resetting), provided we keep the value of S at 1.

Logic behind setting to 1: If we set S to 0 for a brief period of time, this causes a and thus A to be equal to 1. Since R is 1 and A is 1, b must be 0, This causes B to be 0, which makes a equal to 1 again. Now when we return S to 1, a remains the same since B is also 0, and 1 0 input to a nand gate is enough to make sure that the NAND gate stays at 1.

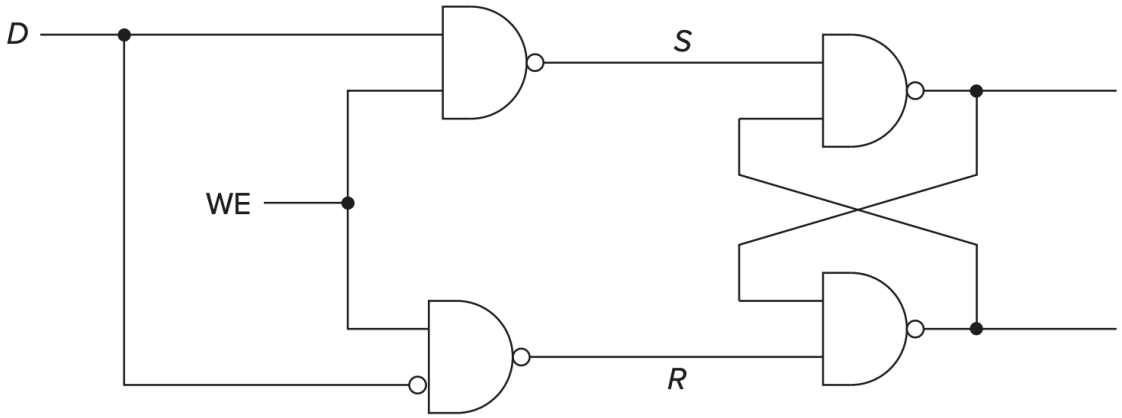
When a digital circuit is powered on, the latch can be in either of its two states, 0 or 1. It does not matter which state since we never use that information until after we have set it to 1 or 0.

The Gated D Latch

Definition 0.4

The D latch helps control when a latch is set and when it is cleared. In the following figure, the latch is set to the value of D whenever WE is asserted. When WE is not asserted, the outputs S and R are both equal to 1.

When WE is momentarily set to 1, exactly one of the outputs S or R is set to 0 depending on the value of D. If D is set to 1, the S is set to 0, else



re 3.19 A gated D latch.

The Concept of Memory

Memory is made up of a (usually large) number of locations, each uniquely identifiable and each having the ability to store a value. We refer to the unique memory location as its *address*. We refer to the number of bits of information stored ine ach location as its *addressability*

Address Space

Definition 0.5

We refer to the total number of uniquely identifiable locations as the **memory's address space**.

For example, 2GB memory has two billion memory locations.

Addressability

Definition 0.6

Addressability: the number of bits stored in each memory location.

2^2 by 3-Bit Memory Example

In this case, the memory has an address space of 4 locations and an addressability of three bits. Since it is 2^2 memory, it takes two bits to specify the address. We specify it using A[1:0]. Since its addressability is 3, that means in each location, it stores 3 bits worth of information/data.

Note 0.3

When specifying a memory location in terms of **A[high:low]**, we are starting from the rightmost spot as index of 0. This means we are looking at the sequence of $h - l + 1$ bits such that high is the leftmost bit number, and low is the rightmost bit number in the sequence.

Access of memory first starts with *decoding the address bits*, using a decoder. We also have WE, which defines whether we are in write-enable mode of not.

The input of A[1:0] defines what the decoder has to select for the correct *word line*. From there, the decoder outputs a line of 1 which is anded across all three D-latches producing the output of that position.

State

Definition 0.7

State: a snapshot of that system in which all relevant items are explicitly expressed.

Ex: for a lock, the state would be open, or 0/1/2 correct operations leading to opening the lock.

Definition 0.8

A **finite state machine** consists of the following elements

1. A finite number of states
2. A finite number of external inputs
3. A finite number of external outputs
4. An explicit specification of all state transitions
5. An explicit specification of what determines each external output value

The state machines we have talked about so far are **asynchronous**, because there is no fixed amount of time in between when these inputs should be fed into the state machine. On the other hand, a **synchronous** state machine (such as most computers) have a fixed amount of time in between inputs.

Note 0.4

The control for the fixed time between state machine changes is controlled by a clock, *whos values alternate between 0 volts and some specified fixed voltage*

Datapath of LC-3

Definition 0.9

Datapath of the LC-3 consists of all the logic structures that combine to process information at the core of the computer.