Digital Electronics

Ivin Lee

October 21, 2020

1 Combinatorial Logic

1.1 Logic Gates

- NOT
- AND
- OR
- XOR
- NAND
- NOR

1.2 Boolean Algebra

- Commutation
  - $a + b = b + a$
  - $a.b = b.a$
- Association
  - $(a + b) + c = a + (b + c)$
  - $(a.b).c = a.(b.c)$
- Distribution
  - $a.(b + c + ...) = (a.b) + (a.c) + ...$
  - $a + (b.c....) = (a + b).(a + c)....$

- Absorption
  - $a + (a.c) = a$
  - $a.(a + c) = a$
- DeMorgan's Theorems
  - $\overline{a + b + c} = \overline{a}.\overline{b}.\overline{c}$
  - $\overline{a.b.c} = \overline{a} + \overline{b} + \overline{c}$

The DeMorgan's Theorems can be used to make circuits that only use NAND or NOR gates.

- $f = a.b.\overline{c} + \overline{a}.b.c = \overline{\overline{a.b.\overline{c}}.\overline{\overline{a}.b.c}}$

1.3 Disjunctive Normal Form

$$f = \overline{x}.\overline{y}.\overline{z} + \overline{x}.\overline{y}.z + \overline{x}.y.\overline{z} + \overline{x}.y.z + x.y.z \tag{1}$$

Each term is called a **minterm**, which must contain all variables

A Boolean function expressed as the disjunction (ORing) of its minterms is said to be in the Disjunctive Normal Form (DNF)

A Boolean function expressed as the ORing of ANDed variables (not necessarily minterms) is often said to be in Sum of Products (SOP) form

1.4 Conjunctive Normal Form

maxterm: $(x + y + z)$

Write

$$\overline{f} = x.\overline{y}.\overline{z} + x.\overline{y}.z + x.y.\overline{z}$$

Applying DeMorgan twice gives

$$f = (\overline{x} + y + z).(\overline{x} + y + \overline{z}).(\overline{x} + \overline{y} + z)$$

This is the conjunctive normal form.

A Boolean function expressed as the ANDing of ORed variables (not necessarily maxterms) is often said to be in Product of Sums (POS) form

1.5 Karnaugh Mapping

Visual way of identifying minterms

- Group minterms with value 1 for sum of products
- group minterms with value 0 for product of sums

Definitions

- Cover: a term covers a minterm if that minterm is part of the term
- Prime Implicant: a term that cannot be further combined
- Essential Prime Implicant: a prime implicant that covers a minterm that no other prime implicant covers
- Covering Set: minimum set of prime implicants which includes all essential terms plus any other prime implicants required to cover all the minterms

1.6 Quine-McCluskey Method

Steps

- List all the minterms, ordered by bitcount
- Find adjacent terms that differ by 1 bit - adjacent minterms, and the ones that aren't a subset of others are prime implicants

1.7 Binary Adders

Making a binary adder just using full adders results in a long delay before the result can be obtained, because it takes time to spit out the carry bit and feed into the next full adder

We can calculate each carry bit independently from the binary inputs and the first carry input

In practice, we have 4-bit adder blocks where within the block, ripple carry adders are used for the carry bit, but the last carry bit is calculated separately

The time taken for the last carry bit to be calculated is roughly equal to the time taken for the other outputs

Basically two parallel tasks instead of one task

1.8 Combinatorial Logic Design

1.8.1 Multilevel Logic

- We can always use '2-level' logic implementations from sum-of-products/product-of-sums
- However it may be unwise to use this implementation
  - Commercially available logic gates usually only have 2-3 inputs
  - Reduce system complexity
  - Reduces the number of literals (total number of times that variables appear in expression)

1.8.2 Gate Propagation Delay

- Due to the finite switching speed of gates
- Hazards: unwanted brief logic level changes
  - Static hazards: momentary change in value of output when it is supposed to be unchanged
  - Dynamic hazard: output changes more than once when it is supposed to change just once

You can demonstrate hazard formation by using `NOT` gates to delay a signal

To remove hazards, draw the K-map of the output, and add another term that overlaps the essential terms

2 Sequential Logic

Use of memory elements to store previous state

2.1 RS Latch

Set, reset and hold conditions

We can create a state transition table based on the Q, R and S values, get the equations for each state transition, and then draw a state diagram.

An RS latch is asynchronous, but most sequential circuits use synchronous

operation. This means the output either changes when the clock signal is 1, or when the clock signal turns 1.

2.2 Transparent D Latch

- Use of D and $\overline{D}$ to prevent reaching the invalid state
- Use of enable signal to turn either inputs off or on

2.3 Master Slave Flip Flops

- The transparent D latch is 'level' triggered, but it is more simple to design circuits when the outputs change on the rising edge of the clock signal.
- Combine 2 transparent D latches into a Master-Slave configuration
-
  - When the clock signal is off, the value to change to is stored by the master latch
  - When the clock signal is turned on, the master latch remembers the signal, so even though the slave latch can change its output, it reads the same input so the value is fixed.
  - This cycle repeats

The master-slave configuration has been superceded by new flip flop circuits which are easier to implement and hvae better performance

2.4 JK Flip Flops

Illegal state is replaced by a toggle state

2.5 T Flip Flops

Only two states: hold and toggle

All these circuits have additional override inputs to reset/set output values, and these are mainly used to force circuits into a known state, say at start-up.

2.6 Timings

- Setup time: minimum duration that the data must be stable at the input before the clock edge
- Hold time: the minimum duration that the data must remain stable on the input after the clock edge – takes some time for the circuit to realise that the clock has turned off and to stop reading input

2.7 Flip Flop Applications

- Counters
- Memory
  - parallel to serial conversion
  - serial to parallel conversion
  - together, serial data communication system

2.8 Counters

2.8.1 Ripple Counters

The output of one flip flop is input as the clock signal for the next flip flop.

Flip flops are triggered on the leading edge of the clock signal so every flip flop causes the signal's frequency to decrease by 2

However due to timing delays, it is not possible to know when the output for the ripple counter is valid, leading to miscounting.

To count to a specific value, we can link the end value to a reset input for the flip flops, i.e. when the flip flop counter has reached the required value, we reset it back to 0.

2.9 Synchronous Counters

All the flip flop clock inputs are directly connected to the clock signal and change their output at the same time.

However more complex combinatorial logic is required to generate the output

Design process:

1. Characteristic table: given the current output and current inputs, what is the next output
2. Excitation table: given the current output and the desired next output, what should the current inputs be
3. Modified state transition table: given the current state and next state, what are all the inputs
4. Derive boolean formula for the inputs from the current state using boolean algebra, k-maps etc
5. If there are some current states that aren't used, can add them as don't care states

2.10 Shift Register

Implemented using a chain of flip flop