CS1101: Lecture 24
The Digital Logic Level: Clocks & Memory

Dr. Barry O'Sullivan
b.osullivan@cs.ucc.ie

Course Homepage
http://www.cs.ucc.ie/˜osullb/cs1101

Lecture Outline

- Clocks
- Subcycles
- Sequencing
- Memory
- The S-R Latch
- **Reading**: Tanenbaum, Chapter 3, Sections 2 & 3

Clocks

- In many digital circuits the order in which events happen is critical - sometimes one event must precede another, sometimes two events must occur simultaneously.
- To allow designers to achieve the required timing relations, many digital circuits use clocks to provide synchronization.
- A **clock** in this context is a ciruit that emits a series of pulses with a precise pulse width and precise interval consecutive pulses.
- The time interval between the corresponding edges consecutive pulses is known as the **clock cycle time**.
- Pulse frequencies are commonly between I and 500 MHz, corresponding to clock cycles of 1000 nsec to 2 nsec.

Clocks - Subcycles

- In a computer, many events may happen during a single clock cycle.
- If these events must occur in a specific order, the clock cycle must be divided into subcycles.
- A common way of providing finer resolution than the basic clock is to tap the primary clock line and insert a circuit with a known delay in it, thus generating a secondary clock signal that is phase-shifted from the primary.
- We will now see a timing diagram which provides four time references for discrete events:
  - Rising edge of C1
  - Falling edge of C1.
  - Rising edge of C2.
  - Falling edge of C2.

Clocks

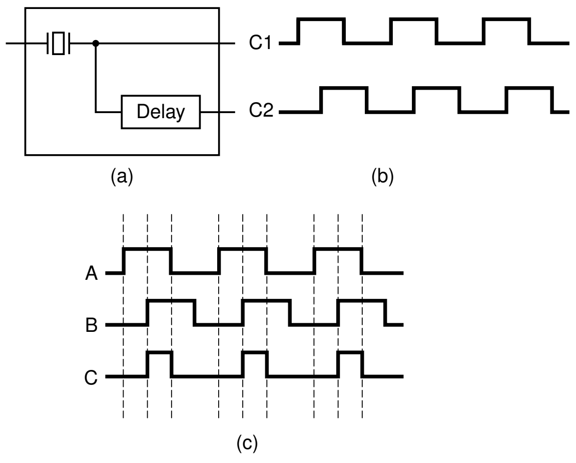


Figure 3-21. (a) A clock. (b) The timing diagram for the clock. (c) Generation of an asymmetric clock.

Clocks - Sequencing

- By tying different events to the various edges, the required sequencing can be achieved.
- If more than four time references are needed within a clock cycle, more secondary lines can be tapped from the primary, with different delays.
- In some circuits one is interested in time intervals rather than discrete instants of time.
- For example, some event may be allowed to happen any time Cl is high, rather than precisely at the rising edge – another event may only happen when C2 is high.

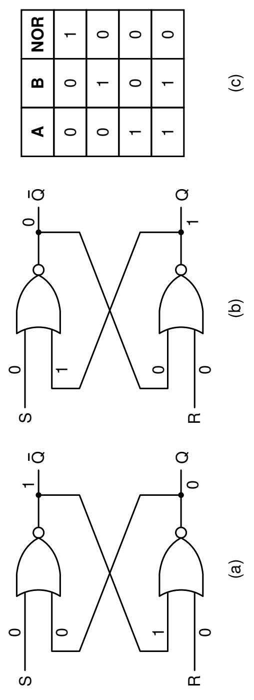
Clocks - Sequencing

- If more than two intervals are needed, more clock lines can be provided or the high states of the two clocks can be made to overlap partially in time.
- In the latter case four distinct intervals can be distinguished:
  - $\overline{C1}$ AND $\overline{C2}$
  - $\overline{C1}$ AND $C2$
  - $C1$ AND $\overline{C2}$
  - $C1$ AND $C2$

Memory

- An essential component of every computer is its memory.
- Memory is used for storing both instructions to be executed and data.
- To create a 1-bit memory, we need a circuit that somehow ”remembers” previous input values.
- Such a circuit can be constructed from two NOR gates.
- Analogous circuits can be built from NAND gates.
- Both are conceptually identical.
- As an example we will consider an **S-R Latch**

An S-R Latch



A	B	NOR
0	0	1
0	1	0
1	0	0
1	1	0

Figure 3-22. (a) NOR latch in state 0. (b) NOR latch in state 1. (c) Truth ta for NOR.

An S-R Latch

- The SR latch. has two inputs, S, for Setting the latch, and R, for Resetting (i.e., clearing) it.
- It also has two outputs, Q and $\overline{Q}$, which are complementary.
- Unlike a combinational circuit, the outputs of the latch are not uniquely determined by the current inputs.

SR Latch - State 0

- Assume that both S and R are 0 – which they are most of the time.
- Also assume that $Q = 0$.
- Thus, the output of the upper NOR gate is 1.
- Thus $\overline{Q}$ is 1
- The 1 is fed back into the lower gate, which then has inputs 1 and 0, yielding $Q = 0$.
- This is consistent

SR Latch - State 1

- Assume $Q = 1$, with R and S still 0.
- The upper gate has input of 0 and 1, and an output $\overline{Q} = 0$, which is fed back to the lower gate.
- This state is also consistent.

SR Latch - Consistent States

- A state with both outputs equal to 0 is inconsistent
- Similarly, it is impossible to have both outputs equal to 1.
- Our conclusion is simple: for $R = S = 0$, the latch has two stable states, which we will refer to as 0 and 1, depending on Q.

SR Latch - Effect of Inputs

- Suppose that S becomes 1 while Q = 0.
- This forces the $\overline{Q}$ output to 0.
- This change makes both inputs to the lower gate 0, forcing the output to 1.
- Thus setting S (i.e., making it 1) switches the state from 0 to 1.
- Setting R to 1 when the latch is in state 0 has no effect because the output of the lower NOR gate is 0 for inputs of 10 and inputs of 11.
- Using similar reasoning, it is easy to see that setting S to 1 when in state Q = 1 has no effect but that setting R drives the latch to state Q = 0.

SR Latch - Summary

- When S is set to 1 momentarily, the latch ends up in state Q = 1, regardless of what state it was previously in.
- Likewise, setting R to 1 momentarily forces the latch to state Q = 0.
- The circuit ”remembers” whether S or R was last on.
- Using this property we can build computer memories.
- Two latch circuits which incorporate clocks are:
  - Clocked SR Latches
  - Clocked D Latches

  The **Clocked D Latch** is a true 1-bit memory.