CPSC 121 - Models of Computation
Module 09. Sequential Circuits

Prof. Karina Mochetti

2020.W1
What is new!

http://www.cs.ubc.ca/~mochetti/CPSC121.html
Goals

- Trace the operation of a DFA (deterministic finite-state automaton) represented as a diagram on an input, and indicate whether the DFA accepts or rejects the input.
- Translate a DFA into a sequential circuit that implements the DFA.
- Explain how and why each part of the resulting circuit works.
- **Combinational Circuit**: Its output is completely determined by its input/output table.
- **Sequential Circuit**: Its output is a function not only of the input to the circuit but also of the state the circuit is in when the input is received.

**Finite-State Automaton**
An idealized machine that embodies the essential idea of a sequential circuit.

**NFA vs DFA**
DFA refers to *Deterministic finite automaton*, a finite-state automaton in which for an input symbol, there is a single resultant state. NFA refers to *Nondeterministic Finite Automaton*, in which there is more than one possible transition from one state on the same input symbol.
Finite-State Automaton

An idealized machine that embodies the essential idea of a sequential circuit.

- The **unlabeled arrow** indicates the initial state.
- The **double circle** indicates an accepting state.
- The **arrows** that link the states indicate what happens when a particular input is made.
Finite-State Automata Objects
1. A finite set $I$, called the input alphabet, of input symbols
2. A finite set $S$ of states the automaton can be in
3. A designated state $s_0$ called the initial state
4. A designated set of states called the set of accepting states
5. A next-state function $N : S \times I \rightarrow S$ that associates a next-state to each ordered pair consisting of a current state and a current input.

Language Accepted by an Automaton
Let $A$ be a finite-state automaton with set of input symbols $I$. Let $I^*$ be the set of all strings over $I$, and let $w$ be a string in $I$. Then $w$ is accepted by $A$ if, and only if, $A$ goes to an accepting state when all the symbols of $w$ are input to $A$. The language accepted by $A$, denoted $L(A)$, is the set of all strings that are accepted by $A$. 
Yay! Very well done overall!!! \o/

Question 4: Which of these correctly describes the language accepted by this DFA? In all cases, we use the phrase "any string" to refer to strings drawn from the alphabet of this DFA: a, b, and c.
A circuit that implements a finite state machine of either type needs to remember the current state, it needs memory.

- A latch
- A flip-flop
- A register (multiple side by side flip-flops with a common clock)
Design a circuit which changes the state of a light bulb each time a (momentary) button is pressed.

What signal does the button generate?
Combinational systems generate output based only on the current inputs:

- **input set** $I$: all possible binary sequences that can be delivered as input to the digital system.

- **output set** $O$: all valid output binary sequences that can be generated by the system.

- **output function** $f : I \rightarrow O$: defines what output sequence will be generated for each input binary sequence.

**Feedback Loops**

There is no connection from the output of a component to the input of another component. **Combinational systems do not have feedback loops!**
Latch is an electronic logic circuit with two stable states that has a feedback path to retain the information.

- When $c$ is low, the MUX retains its current value.
- When $c$ is high, it changes its value to $d$ instead.

<table>
<thead>
<tr>
<th>$d$</th>
<th>$c$</th>
<th>$q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>q</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>q</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Complete the circuit based on the required behaviour:

- Button is not pressed: circuit keeps same output
- Button is pressed: circuit changes / inverts the output
Complete the circuit based on the required behaviour:

- Button is not pressed: circuit keeps same output
- Button is pressed: circuit changes / inverts the output

The AND gate is necessary to allow us to initialize the state of the flip-flop.
What is wrong with our solution?

a. We should have used XOR instead of NOT.
b. The light will be in a random, unpredictable state.
c. The delay introduced by the NOT gate is too long.
d. There is some other problem with the circuit.
e. Nothing is wrong.
What is wrong with our solution?

a. We should have used XOR instead of NOT.
b. The light will be in a random, unpredictable state.
c. The delay introduced by the NOT gate is too long.
d. There is some other problem with the circuit.
e. Nothing is wrong.
It stores the incoming value on $d$, only at the instant the clock $c$ input changes from low to high.

Connecting multiple flip-flops to a common clock signal allows all of the flip-flops to update their outputs at the same time!
Latches vs Flip Flop

- Latch changes the value to $d$ while the clock is high, only maintaining the value on the low.

- Flip Flop changes the value to $d$ when the clock goes from low to high, ignoring any change on $d$ until this change happens again.
Questions?

Ask CPSC 121

http://www.cs.ubc.ca/~mochetti/askCPSC121.html
Observe that the two select input are never the same.
Assume that everything starts with 0.
The clock goes high.
We set $d$ to 1.
The clock goes low.
The clock goes high.
The clock goes low.
We set $d$ to 0.
The clock goes high.
The clock goes low.
We set $d$ back to 1.
The clock goes high.
And we get the following improved circuit for our button and light problem.
Question 1
Analyzing Sequential Circuits

For time = 1.00 => Clock = __________
Questions?

Ask CPSC 121

http://www.cs.ubc.ca/~mochetti/askCPSC121.html
Design a DFA for a vending machine that sells one of three items (lemon juice, whiteboard markers, and corn flour) for 35¢ each. It should accept 5¢, 10¢ and 25¢ coins, and does not need to return change.
Real numbers:
We can write numbers in decimal using the format
(-)? \(d+ (.d+)?\)
where the \(( )?\) mean that the part in parentheses is optional,
and \(d+\) stands for “1 or more digits”.

1. Design a DFA that will accept input strings that are valid real
numbers using this format. You can use else as a label on an
edge instead of listing every character that does not appear on
another edge leaving from a state.

2. Design a circuit that turns a LED on if the input is a valid real
number, and off otherwise.