By the start of class, you should be able to
- 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.
- Deduce the language accepted by a simple DFA after working through multiple example inputs.

Quiz 9 feedback:
Well done.
Many fine answers to the push-button light question.
We will revisit this problem soon.

CPSC 121: the BIG questions:
1. How can we build a computer that is able to execute a user-defined program?
   a) Computers execute instructions one at a time.
   b) They need to remember values, unlike the circuits you designed in labs 1, 2, 3 and 4.
   c) That is, a computer is a very large and very complicated sequential circuit.
Module 9: Sequential Circuits

- By the end of this module, you should be able to:
  - Translate a DFA into a sequential circuit that implements the DFA (lab 8).
  - Explain how and why each part of the resulting circuit works.

Module Summary

- Latches, toggles and flip-flops.
- Analyzing sequential circuits.
- Other problems and exercises.

Module 9: Sequential Circuits

- Announcements:
  - Pre-class quiz #10 is due Monday November 16th at 19:00.
  - Textbook sections:
    - Epp, 4th or 5th edition: 5.1 to 5.4
    - Epp, 3rd edition: 4.1 to 4.4
    - Rosen, 6th edition: 4.1, 4.2
    - Rosen, 7th edition: 5.1, 5.2
  - Assignment #4 is also due November 16th at 19:00.

- Announcements (continued):
  - Pre-class quiz #11 is tentatively due Monday November 30th at 19:00.
  - Textbook sections:
    - Epp, 4th or 5th edition: remainder of 6.1, 7.1
    - Epp, 3rd edition: remainder of 5.1, 6.1
    - Rosen, 6th edition: remainder of 2.1, 2.3 up to the top of page 136.
    - Rosen, 7th edition: remainder of 2.1, 2.3 down to the bottom of page 141.
Module 9.1: Latches, toggles and flip-flops

- 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)

Recall the latch from lab #5:
- When \(en\) is low, the MUX retains its current value.
- When \(en\) is high, it changes its value to \(d\) instead.

Problem:
Design a circuit that changes state every time a button is pushed.

What signal does the button generate?

- high
- low
Module 9.1: Latches, toggles and flip-flops

- Complete the circuit...

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.

This toll booth has a similar problem.

Instead use this:

P.S. Call this a “bar”, not a “gate”, or we’ll tie ourselves in (k)nits.
Module 9.1: Latches, toggles and flip-flops

- The circuit version of this improved tollbooth is called a flip-flop:

![Flip-flop circuit diagram]

Assume the value stored in the flip-flop is 1 and \( d = 0 \). As long as the clock remains low:

![Flip-flop state diagram]

Observe that the two select inputs are never the same.

![Select input states diagram]

Now the clock goes high:

![Clock high state diagram]
Module 9.1: Latches, toggles and flip-flops

- Now the clock goes low again:

- Finally we set $d = 1$:

- And we get the following improved circuit for our button and light problem:

- Module Summary
  - Latches, toggles and flip-flops.
  - Analyzing sequential circuits.
  - Other problems and exercises.
Module 9.2: Analyzing sequential circuits

- How does a sequential circuit work?
  - When the clock is low, the “bar” on every flip-flop and register is down – they do not take on a new value.
  - Values propagate everywhere else in the circuit.
  - When the clock goes from low to high, the “bars” go up, and the values sitting at the D inputs of flip-flops and registers go into them.
  - The “bars” then come back down, and the process is repeated.

Example: consider the following circuit:

- Time \( t = 0 \)

- Time \( t = 0.5 \)

- Time \( t = 1.0 \)
Module 9.2: Analyzing sequential circuits

- What will be the state of the circuit at time $t = 5.0$?

Module 9.2: Analyzing sequential circuits

- How do you design a sequential circuit?
  - Figure out which state it needs to perform its task.
  - Add one or more flip-flops or registers to store it.
  - Determine how each part of the state changes from one clock cycle to the next.
  - Add gates to compute the new value of each part of the state from the old state information.

Module 9: Sequential Circuits

- Module Summary
  - Latches, toggles and flip-flops.
  - Analyzing sequential circuits.
  - Other problems and exercises.

Module 9.3: Other problems and exercises

- 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”.
  - 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.
Module 9.3: Other problems and exercises

- Real numbers (continued)
  - Then design a circuit that turns a LED on if the input is a valid real number, and off otherwise.
    - Hint: Logisim has a keyboard component you can use.
    - Hint: my DFA for this problem has 6 states.

- 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.