ANNOUNCEMENTS

- **Lab project:**
  - No lab sections will be held next Tuesday (Veterans Day). A special section will be held next Monday at 6 PM for the Tuesday-section students to pick up their Tutebot kits.
  - For extra credit (full points on HW portion of course grade):
    - Each team of 2 must demo. 1 additional behavior
    - Each team of 3 must demo. 2 additional behaviors
    - Extra credit for top Tutebot in class: 5 pts on course grade

OUTLINE

- Sequential logic circuits

Reading: Schwarz & Oldham pp. 411-420

---

Further Comments on Karnaugh Maps

- The algebraic manipulations needed to simplify a given expression are not always obvious. Karnaugh maps make it easier to minimize the number of terms in a logic expression.

- **Terminology:**
  - “2-cube: 2 squares that have a common edge (-> product of 3 variables)
  - “4-cube: 4 squares with common edges (-> product of 2 variables)

- In locating cubes on a Karnaugh map, the map should be considered to fold around from top to bottom, and from left to right.
  - Squares on the right-hand side are considered to be adjacent to those on the left-hand side.
  - Squares on the top of the map are considered to be adjacent to those on the bottom.
  - **Example:**
    - The four squares in the map corners form a 4-cube
Flip-Flops

- One of the basic building blocks for sequential circuits is the flip-flop:
  - 2 stable operating states → stores 1 bit of info.
  - A simple flip-flop can be constructed using two inverters:

![Flip-Flop Diagram](image)

The S-R ("Set"-"Reset") Flip-Flop

- **Rule 1:**
  - If $S = 0$ and $R = 0$, $Q$ does not change.

- **Rule 2:**
  - If $S = 0$ and $R = 1$, then $Q = 0$

- **Rule 3:**
  - If $S = 1$ and $R = 0$, then $Q = 1$

- **Rule 4:**
  - $S = 1$ and $R = 1$ should never occur.
Realization of the S-R Flip-Flop

### Table

<table>
<thead>
<tr>
<th>R</th>
<th>S</th>
<th>Q&lt;sub&gt;n&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Q&lt;sub&gt;n-1&lt;/sub&gt;</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>(not allowed)</td>
</tr>
</tbody>
</table>

Clock Signals

- Often, the operation of a sequential circuit is synchronized by a **clock signal**:

- The clock signal regulates when the circuits respond to new inputs, so that operations occur in proper sequence.

- Sequential circuits that are regulated by a clock signal are said to be **synchronous**.
**Clocked S-R Flip-Flop**

- When \( CK = 0 \), the value of \( Q \) does not change
- When \( CK = 1 \), the circuit acts like an ordinary S-R flip-flop

**The D ("Delay") Flip-Flop**

- The output terminals \( Q \) and \( \overline{Q} \) behave just as in the S-R flip-flop.
- \( Q \) changes only when the clock signal \( CK \) makes a positive transition.

<table>
<thead>
<tr>
<th>( CK )</th>
<th>( D )</th>
<th>( Q_n )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( \times )</td>
<td>( Q_{n-1} )</td>
</tr>
<tr>
<td>1</td>
<td>( \times )</td>
<td>( Q_{n-1} )</td>
</tr>
<tr>
<td>↑ 0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>↑ 1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
D Flip-Flop Example (Timing Diagram)

Realization of the D Flip-Flop
Registers

- A **register** is an array of flip-flops that is used to store or manipulate the bits of a digital word.

**Example:** Serial-In, Parallel-Out Shift Register

---

Conclusion (Logic Circuits)

- Complex combinational logic functions can be achieved simply by interconnecting NAND gates (or NOR gates).
- Logic gates can be interconnected to form flip-flops.
- Interconnections of flip-flops form registers.
- A complex digital system such as a computer consists of many gates, flip-flops, and registers. Thus, logic gates are the basic building blocks for complex digital systems.