Timing Diagrams

- **Graphical representation of circuit behavior over time**
  - illustrates the logic behavior of signals in a digital circuit as a function of time

- **May be used as a device specification**
  - illustrates device performance

- **May be used as a module or system specification**
  - identifies a requirement for system performance

- **May be used as a tool in system analysis**

Timing Diagram Notation

![Timing Diagram Notation](image-url)

- **H**
  - Changing values
  - Stable Value, high or low
  - Changing values

- **L**
  - Clean transitions
  - Tri-state
  - Stable, driven
  - High impedance
**Timing Terminology**

**Caution:** terminology may vary slightly between vendors.

**Set-up time,** $t_{su}$: the minimum length of time that a signal must be valid at a circuit input before a second triggering signal arrives at a second input.

*Usually a clock*

**Delay time,** $t_{dc}$: the length of time that a circuit requires for its output(s) to begin to change in response to a triggering signal arriving at an input. (also called propagation delay)

**Hold time,** $t_{ho}$: the minimum length of time that a signal must be kept valid at a circuit input after a triggering signal has been received at a second input.

**Timing skew,** $t_{skew}$: the maximum range of times over which a particular signal transition can occur.

-- Due to variations in driver output impedance

-- Problems in clock distribution

---

**Nomenclature for Timing Diagrams**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The input must be valid</td>
<td>The output will be valid</td>
</tr>
<tr>
<td></td>
<td>If the input were to fall</td>
<td>Then the output will fall</td>
</tr>
<tr>
<td></td>
<td>If the input were to rise</td>
<td>Then the output will rise</td>
</tr>
<tr>
<td>XXXXXX</td>
<td>Don't care, it will work regardless</td>
<td>Don't know, the output value is indeterminate</td>
</tr>
<tr>
<td></td>
<td>Nonsense</td>
<td>High impedance, tristate, HiZ. Not driven, floating</td>
</tr>
</tbody>
</table>
**Example Timing Diagram**

- **Functional Timing Diagram (idealized)**
  - Assumes zero delays
  - Simply demonstrates logic relations

- Arrows show cause and effect, which input transactions cause which output transactions, especially in complex timing diagrams.

**Example Timing Diagram**

- **Timing Diagram (more realistic)**
  - Shows delays using typical or maximum values

- \( t_{\text{pHL}} = \text{HIGH to LOW propagation delay} \)
- \( t_{\text{pLH}} = \text{LOW to HIGH propagation delay} \)
**74LS244 Buffer 1**

- Propagation delay = 12nS (enabled) - typical value
  - Analysis (design) should use **MAXIMUM** values for worst case.

**74LS244 Buffer 2**

- tri-state buffer (see later slide)
- Propagation delay = 12nS (enabled)
Karnaugh Maps (K-Maps) are a graphical method of visualizing the 0's and 1's of a boolean function. K-Maps are very useful for performing Boolean minimization.

- Will work on 2, 3, and 4 variable K-Maps in this class. Variable-Entered-Maps will be used for systems with more than 4 variables.

- Karnaugh maps are much easier to use than boolean equations for minimization.

Karnaugh Map Method

- K-map is an alternative method of representing the truth table that helps visualize adjacent terms in up to 6 dimensions.

Numbering Scheme: 00, 01, 11, 10

Gray Code -- only a single bit changes one code word to the next.
2-Variable K-Map

Row | A  | B | F(A,B) |
--- | --- | --- | --- |
0   | 0  | 0 | 0     |
1   | 0  | 1 | 1     |
2   | 1  | 0 | 1     |
3   | 1  | 1 | 0     |

F(A,B) = \overline{A}B + AB

Row | A  | B | F(A,B) |
--- | --- | --- | --- |
0   | 0  | 0 | 0     |
1   | 0  | 1 | 0     |
2   | 1  | 0 | 1     |
3   | 1  | 1 | 1     |

F(A,B) = AB

Boolean Adjacency

- Note on the three variable map:

  - Each square on the 3-variable map is Boolean Adjacent. Adjacent squares
    - only differ by ONE BOOLEAN VARIABLE
  - Although drawn as a 2-D diagram the edges wrap round left-right and top-bottom.
3-Variable K-Map

F(A, B, C) = \sum m(0, 2, 6)

Plotting 4-Variable Functions

F(A, B, C, D) = \sum m(? ? ? ?)
**Plotting 4-Variable Functions**

<table>
<thead>
<tr>
<th>Row</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>F(A,B,C,D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Minimization via K-Maps**

<table>
<thead>
<tr>
<th>Row</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>F(A,B,C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ F(A,B,C) = \sum m(2,6) \]

\[ = \overline{A}BC + AB\overline{C} \]

\[ = BC(\overline{A} + A) \]

\[ = BC \]

**Boolean adjacency can be used to minimize functions!**
**Simplification using K-Maps**

- **Grouping blocks of ‘1’**
  - A group must consist of 16, 8, 4, 2 or 1 cells
  - Each cell must be horizontally and/or vertically adjacent to cells in the other group
  - Always include the largest number of ‘1’s in a group
  - Each ‘1’ in the map should be included in a group
  - Groups can overlap
  - Map edge cells connect in a loop to cells at the opposite edge

- **Naming groups**
  - The product description for a group will include ALL variables that are CONSTANT over the group
  - For example, for a 4-variable map
    - An 8-cell group is described by a 1-variable product term
    - An 4-cell group is described by a 2-variable product term
    - An 2-cell group is described by a 3-variable product term
    - An 1-cell group is described by a 4-variable product term

---

**Groupings on K-Maps**

- Grouping can be read DIRECTLY as “BC” by looking at what is COMMON within the circled group.

\[
\begin{array}{ccc}
A & 0 & 1 \\
B & C & 0 & 1 \\
00 & 0 & 0 & - \\
01 & 0 & 0 & - \\
11 & 0 & 0 & - \\
10 & 1 & 1 & - \\
\end{array}
\]

\[F(A, B, C) = \overline{BC}\]
Example Groupings on 3-Variable K-Maps

\[
\begin{array}{ccc}
A & B & C \\
00 & 1 & 0 \\
01 & 1 & 0 \\
11 & 0 & 0 \\
10 & 0 & 0 \\
\end{array}
\]

\[
F(A, B, C) =
\]

\[
\begin{array}{ccc}
A & B & C \\
00 & 1 & 1 \\
01 & 0 & 0 \\
11 & 0 & 0 \\
10 & 1 & 1 \\
\end{array}
\]

\[
F(A, B, C) =
\]

\[
\begin{array}{ccc}
A & B & C \\
00 & 1 & 1 \\
01 & 1 & 1 \\
11 & 0 & 0 \\
10 & 0 & 0 \\
\end{array}
\]

\[
F(A, B, C) =
\]

Multiple Groupings

\[
\begin{array}{ccc}
A & B & C \\
00 & 1 & 0 \\
01 & 1 & 1 \\
11 & 0 & 0 \\
10 & 0 & 0 \\
\end{array}
\]

\[
F(A, B, C) =
\]

Try to cover all ‘1’s with largest possible groupings.

\[
\begin{array}{ccc}
A & B & C \\
00 & 0 & 1 \\
01 & 0 & 0 \\
11 & 1 & 0 \\
10 & 1 & 0 \\
\end{array}
\]

\[
F(A, B, C) =
\]

Groupings of only a single ‘1’ are ok if larger groupings cannot be found.
**Illegal Groupings**

<table>
<thead>
<tr>
<th>BC</th>
<th>00</th>
<th>01</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Illegal Grouping! Minterms are not boolean adjacent!

\( \overline{ABC}, ABC \) will NOT reduce to a single product term

\[ \overline{ABC} + ABC = \overline{B}(\overline{AC}+AC) \]

- Valid groupings will always be a power of 2.
  - will cover 1, 2, 4, 8, etc minterms.

---

**Groupings on four Variable Maps**

<table>
<thead>
<tr>
<th>AB</th>
<th>CD</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ F(A,B,C,D) = \]
### Other Groupings

<table>
<thead>
<tr>
<th></th>
<th>AB 00</th>
<th>AB 01</th>
<th>AB 11</th>
<th>AB 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD 00</td>
<td>1 0 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD 01</td>
<td>1 0 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD 11</td>
<td>1 0 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD 10</td>
<td>1 0 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
F(\text{A,B,C,D}) =
\]

### More than one way to group.....

More than one way to group.....

Want LARGEST groupings that can cover ‘1’.

\[
F(\text{A,B,C,D}) = \text{BD} + \text{CD} + \text{CD}
\]

\[
F(\text{A,B,C,D}) = \text{B} + \text{D}
\]
**Four Corner Grouping on 4-Variable Map**

<table>
<thead>
<tr>
<th>AB</th>
<th>CD</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>00 01 11 10</td>
</tr>
<tr>
<td>00</td>
<td>1 0 0 1</td>
</tr>
<tr>
<td>01</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>11</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>10</td>
<td>1 0 0 1</td>
</tr>
</tbody>
</table>

4 Corner grouping is valid on four variable map

\[ F(A,B,C,D) = \]

**K-map Definitions**

- **Implicant**
  - Any single 1 or any group of 1’s is called an implicant of \( F \).
  - Any possible grouping of ‘1’\’s is an implicant.

- **Prime Implicant**
  - A implicant that cannot be combined with some other implicant to eliminate a variable
**Minimum Sum-Of-Products (SOP)**

- The minimum SOP expression consists of some (but not necessarily all) of the prime implicants of a function.

- If a SOP expression contains a term which is NOT a prime implicant, then it CANNOT be minimum.

---

**Prime Implicants**

Minimum SOP will have some or all of these prime implicants. The included prime implicants must cover all of the ONEs.

\[
\begin{array}{cccc}
  & A & B & C & D \\
 00 & 0 & 1 & 1 & 0 \\
 01 & 1 & 1 & 1 & 0 \\
 11 & 1 & 0 & 0 & 0 \\
 10 & 0 & 0 & 0 & 0 \\
\end{array}
\]

Each of these coverings is a **prime implicant** (i.e. cannot be reduced).

- \(BC\)
- \(ACD\)
- \(ABD\)

\[
F(A,B,C,D) = BC + ABD \\
= BC + ABD + ACD \\
\nequiv ABD + ACD
\]

(minimum set of PIs)

(valid set of PIs, but not minimum)

(both PI's, but all '1's not included!)
**Non-Essential vs. Essential Prime Implicants**

<table>
<thead>
<tr>
<th>AB</th>
<th>CD</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>1 1 1 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>1 1 1 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1 0 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

EACH of the coverings is a **PRIME IMPLICANT**.

- **NON-ESSENTIAL prime implicant**
  - Prime Implicant $\overline{A}CD$ is non-essential because its ‘1’s are covered by other PIs. A PI is **ESSENTIAL** if it covers a MINTERM that cannot be covered by any other PI.

F($A,B,C,D$) = $BC + \overline{A}BD$  
(minimum # of PIs)

**An example with more than one solution**

<table>
<thead>
<tr>
<th>AB</th>
<th>CD</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>1 1 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>1 1 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1 0 0 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0 0 1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

EACH of the coverings is a **PRIME IMPLICANT**.

- $AC$  
- $ACD$  
- $\overline{ABD}$  
- $BCD$

Recall that a covering is a Prime Implicant if it cannot be combined with another covering to eliminate a variable.
Two Solutions

Each solution is equally valid.

\[ F(A,B,C,D) = \overline{A}C + ACD + \overline{A}BD \]

Essential PIs

\[ F(A,B,C,D) = \overline{A}C + ACD + BCD \]

Non-Essential PIs

Minimal Solution

- A minimal SOP will consist of prime implicants.

- A minimal SOP equation will have all of the essential prime implicants on the map. By definition, these cover a minterm that may not be covered by some other prime implicant.

- The minimal SOP equation may or may not include non-essential prime implicants. It will include non-essential prime implicants if there are ‘1’s remaining that have not been covered by an essential prime implicant.