Network Design

**Definition:** A two-level sum-of-products (SOP) realization of a switching function is **minimal** if there exists

(i) no other equivalent expression with fewer products;

(ii) no other equivalent expressions with the same number of products but with fewer literals.

**Problem:**

Given a switching function $f$, find a minimal SOP realization.

So far we have considered optimization based on:

- applying the basic switching algebra theorems;
- Karnaugh maps.
Karnaugh Map Drawbacks:

- essentially a trial and error process of optimization without guarantee of success – irredundant and nonminimal realizations possible;

- based extensively on intuition and not easily automated;

- difficult to use for functions of $n$ variables when $n \geq 6$.

The Quine-McCluskey procedure:

- produces a minimal SOP realization;

- is algorithmic in nature and therefore easily automatable.
First we develop an intuitive notion of the basic concept of a prime implicant of a switching function of $n$ variables.

This is done by considering a constructive procedure based on a K-map, the informally defined notion of cubes of a switching function, their representation as a $n$ bit binary number, and their correspondence to a switching expression.

The function that will be analyzed is

$$F(w, x, y, z) = \sum(0, 2, 3, 6, 7, 8, 9, 10, 13)$$
\[ F( w, x, y, z ) = \sum (0,2,3,6,7,8,9,10,13) \]

1–cubes
1 x in binary digit
2 minterms in
1 X 2 or 2 X 1
rectangle of cells

0,8  x000
9,13  1x01
3,7   0x11
2,6   0x10
8,10  10x0
0,2   00x0
2,3   001x
6,7   011x
8,9   100x
2,10  x010

2–cubes
2 x’s in binary digit
4 minterms in
1 X 4, 4 X 1,
or 2 X 2
rectangle of cells
2,3,6,7  0x1x
covers
3,7  2,6
2,3   6,7
0,2,8,10  x0x0
covers
0,2   0,8
2,10  8,10

There are no 3–cubes
with 3 x’s and 8 minterms in
2 X 4 or 4 X 2 rectangles
$$F(w, x, y, z) = \sum (0, 2, 3, 6, 7, 8, 9, 10, 13)$$

cubes not covered by a larger cube
0, 2, 8, 10 (x0x0)  2, 3, 6, 7 (0x1x)
8, 9 (100x)  9, 13 (1x01)

$$F(w, x, y, z) = x'z' + w'y + wy'z + wx'y'$$
Definition: i-cubes of a switching function $f$ of $n$ variables for $i = 0, \ldots, n$ are defined as follows: A 0-cube is denoted by an $n$ bit binary number and corresponds to the switching function of the minterm with the same binary representation. It also corresponds to the single cell in the K-map associated with the minterm.

A 1-cube is the union of two 0-cubes whose binary representation differ in exactly one bit. The 1-cube is denoted by an $n$ bit binary number with an $x$ in the differing position and the n-1 common bits from the two 0-cube representations. It also corresponds to a 1 by 2 or 2 by 1 rectangle in the K-map containing two minterms. The switching expression associated with the 1-cube is the product of the switching variables associated with the positions of the 1’s and the complement of the switching variables associated with the positions of the 0's.

An i-cube is the union of two (i-1)-cubes whose binary representations are such that the x’s are in identical positions and the remaining bits differ in exactly one position. The i-cube is denoted by a binary digit with the n-1 identical bit and x positions in the (i-1)-cubes’ representations and an additional x in the single differing bit position. It corresponds to a rectangle in the K-map with dimensions that are powers of 2 containing $2^i$ minterms. The associated switching expression is derived in the same manner as described for the 1-cube.
Definition: A switching function $f$ is said to imply another switching function $g$ if whenever $f = 1$ then $g = 1$ as well. $g$ is said to cover or include $f$.

- a 1-minterm of $F$ is an implicant of $F$
- an i-cube of $F$ is an implicant of $F$
- if $F = T_1 + \cdots + T_k$, then any $T_i$ is an implicant of $F$
- if $I_1$ and $I_2$ are both implicants of $F$ then $I_3 = I_1 + I_2$ is also an implicant of $F$
- if $G = F \bullet H$ then $G$ is an implicant of $F$ and $G$ is an implicant of $H$

The geometric objects produced by the K-map procedure above (the i-cubes) can now be used to define a prime implicant.
Implication is written $F \rightarrow G$

\[
\begin{align*}
F = 1 & \rightarrow G = 1 \\
G = 0 & \rightarrow F = 0 \\
G = 1 & \rightarrow F = 0 \text{ or } 1 \\
F = 0 & \rightarrow G = 0 \text{ or } 1
\end{align*}
\]
\begin{itemize}
\item $G = F \cdot H \rightarrow F$
\item $G = F \cdot H \rightarrow H$
\end{itemize}

We are interested when functions are products of literals

\[
F = wx \\
H = yz \\
G = wxyz
\]

If we have:

\[
w = 1 \\
x = 1 \\
y = 1 \\
z = 1
\]

then we have:

\[
wxyz = 1 \\
w = 1 \\
yz = 1
\]

This is easily detectable from binary patterns:

\[
F := 11** \\
H := **11 \\
G := 1111
\]
**Definition:** Given a switching function $f$ of $n$ variables and the associated $i$-cubes for $i = 0, \ldots, n$, a **prime implicant** of $f$ is a switching expression associated with an $i$-cube of $f$ which is **not** covered by the switching expression for some $j$-cube of $f$ where $j > i$.

Note that it is easy to decide if a $j$-cube includes an $i$-cube where $j > i$ by looking at the binary representation of each. Since the binary representation of the $j$-cube contains more $x$’s than the $i$-cube, merely test to see if the binary representation of the $i$-cube is consistent with the bit pattern and $x$’s of the representation of the $j$-cube. For example, the 1-cube 10x0 is included in the 2-cube x0x0.
A Second Example  \[ F(w,x,y,z) = \sum (2,3,5,6,7,9,11,13) \]

- **1-cubes**
  - \(2,3 \), \(2,6 \)
  - \(3,7 \), \(3,11 \)
  - \(5,7 \), \(5,13 \)
  - \(6,7 \)
  - \(9,11 \), \(9,13 \)

- **2-cubes**
  - \(2,3,6,7 \)

- **Prime Implicant's**
  - \(2,3,6,7 \)
  - \(3,11 \)
  - \(5,7 \), \(5,13 \)
  - \(9,11 \), \(9,13 \)

**Complete sum**
\[ w'xz + w'y + xy'z + wy'z + wx'z + x'yz \]
A Second Example  \[ F(w,x,y,z) = \bigvee (2,3,5,6,7,9,11,13) \]

\[ \begin{array}{cccc}
  & 00 & 01 & 11 & 10 \\
 00 & 0 & 4 & 12 & 8 \\
 01 & 1 & 5 & 13 & 9 \\
 11 & 3 & 8 & 15 & 11 \\
 10 & 2 & 6 & 14 & 10 \\
\end{array} \]

Irredundant and minimal
\[ w'y + xy'z + wx'z \]

Irredundant and not minimal
\[ w'y + w'xz + wy'z + x'yz \]
Recall, that we expressed $F$ in terms of the switching expressions for the i-cubes that covered all of the minterms. This motivates the following definition.

**Definition**: The complete sum of a switching function $F$ is the sum of all of the prime implicants of $F$. Clearly, such a sum must realize $F$ due to its construction.

**Fact 1.** A complete sum realization is not necessarily a minimal realization. So the number of products may be reducible.

**Example 1.** Consider $F(w, x, y, z) = \Sigma(2, 3, 5, 6, 7, 9, 11, 13)$ for which we determined the prime implicants via a K-map.

The complete sum of $F$ can be algebraically reduced to a minimal sum. (See the K-maps for the graphical derivation.)

$$f = (w'y + xy'z + w'xz) + wy'z + wx'z + x'yz$$

**Cons. + Assoc.** $= w'y + (xy'z + wy'z + wx'z) + x'yz$

**Dist. + Cons.** $= w'y + xy'z + wx'z + x'yz$

**Comm. + Assoc.** $= (w'y + x'yz + wx'z) + xy'z$

**Cons.** $= w'y + wx'z + xy'z$

Note that all of the product terms in the minimal sum that is produced were also in the complete sum, i.e., we still have a sum of only prime implicants.
Fact 2. Removing a set of redundant prime implicants from the complete sum may not yield a minimal realization, i.e., irredundant not necessarily minimal. So reducing the number of products incorrectly does not achieve the goal.

Example 2. The second example yields an irredundant nonminimal realization — \( F' = w'y + w'xz + wy'z + x'yz. \) (See K-map for the graphical derivation.)

Fact 2a. It can be shown (using a proof similar to that of the Prime Implicant Theorem proof) that any irredundant sum is a subset of the complete sum, i.e., it involves only products that are prime implicants.

(Note that irredundancy is a property of a single SOP; minimality is irredundancy PLUS a condition relative to a set of other sums.)
**Fact 3.** The products are prime in the sense that you cannot remove a literal from any of them without causing the complete sum (and any expression derived from it) to differ from the original switching function. So while the number of products can be reduced the complexity of each cannot.

**Example 3.** The second example yields the following complete sum

\[ w'y + xy'z + w'xz + wy'z + wx'z + x'yz \]

Consider changing \( w'xz \) to \( xz \) by removing the \( w' \) literal. Such a removal implies that more reduction could have been done via the K-map theorem since the complete sum would have a two literal term implying the existence of another 2-cube. As a result, additional minterms are added to the K-map changing the function. In this case, 1111 would evaluate true. This is true of all the prime implicants.
The last fact motivates an alternative definition of a prime implicant.

**Definition:** A prime implicant of \( f(x_1, \ldots, x_n) \) is a product of literals \( y_1 y_2 \ldots y_m \), such that \( m \leq n \), \( f \) includes \( y_1 y_2 \ldots y_m \) and if any \( y_i \) is removed \( f \) does not include the remaining product.

It is clear from Example 3 that the prime implicants defined earlier satisfy this definition.

Even though the complete sum does not solve the problem progress has been made. Based on the previous statements we can state:

**Prime Implicant Theorem:** A minimal SOP realization of a switching function \( f \) must be a sum of prime implicants of \( f \).
Proof of Prime Implicant Theorem: Suppose \( f = P + R \) is a minimal sum realization where \( P \) is a product term that is not a prime implicant.

By definition, \( f \) includes \( P \). Since \( P \) is not a prime implicant it can be written

\[
P = y_i Q
\]

where \( y_i \) is the literal that can be removed from \( P \) while still satisfying

\[
Q = 1 \rightarrow f = 1.
\]

Now since \( P = y_i Q \) it follows that \( Q \) includes \( P \), i.e.,

\[
P = 1 \rightarrow Q = 1
\]
We have

(i) If $Q = 1$ then $f = 1$. ($f$ includes $Q$)

(ii) If $P = 1$ then $Q = 1$. (by construction)

The pair $(P, R)$ can only take on the values $(0,0)$, $(0,1)$, $(1,0)$ and $(1,1)$. So compare $f$ and $g = Q + R$ for these values, i.e., use perfect induction.

Case 1: Assume $P = 0$ or $P = 1$ and $R = 1$. $f = P + 1 = 1$ and $g = Q + 1 = 1$.

Case 2: Assume $P = 1$ and $R = 0$. $f = 1 + 0 = 1$. $g = Q + 0$. But by (ii) $Q = 1$ when $P = 1$, so $g = 1 + 0 = 1$.

Case 3: Assume $P = 0$ and $R = 0$. $f = 0 + 0 = 0$. $g = Q + 0 = Q$. If $Q = 1$ then by (i) $f = 1$. But we know $f = 0$ since $R = 0$, so $Q = 0$ by contradiction. Therefore, $g = Q = 0$.

Now, $f = Q + R$ which has one less literal than $P + R$ so $P + R$ could not have been a minimal realization. Therefore, any minimal realization must contain only prime implicants. □
minterms of $F$

Prime Implicants of $F$

- complete sum
  - covers $F$
  - may be redundant

- irredundant sum
  - covers $F$
  - may be nonminimal

- minimal sum
  - covers $F$
  - minimal cost
So we need two things

(i) an algorithm (as opposed to a K-map manipulation) that produces the prime implicants of $f$;

(ii) an algorithm to identify a set of prime implicants that gives a minimal SOP realization of $f$.

We consider (i) first.
Quine-McCluskey Tabulation of Prime Implicants

We will demonstrate the algorithm on

\[ f(w, x, y, z) = \sum (0, 2, 3, 6, 7, 8, 9, 10, 13) \]

The algorithm works by enumerating the 0-cubes, then the 1-cubes, etc. while monitoring the inclusion of lower order cubes in subsequently produced higher order cubes.

First determine the binary representation of the minterms (0-cubes) and group them based on \( POP(m_i) \) which is defined to be the number of 1’s in the representation or \( m_i \). Sort the groups from lowest population count to highest. Denote \( S(0, k) \) the set with count \( k \).
Set $k = 0$. Build 1-cubes from 0-cubes by comparing every $m_i$ in $S(0, k)$ to every $m_j$ in $S(0, k + 1)$. (Compare even if either has been already marked as covered.) When a pair differs in exactly one bit position, record a new 1-cube and denote it by the common bit position values and an $x$ in the differing position. Also record the minterm indices which define the 1-cube. Mark both 0-cubes as covered. After completing the comparison of $S(0, k)$ to $S(0, k + 1)$, record any 0-cubes that are uncovered in $S(0, k)$ and group the new 1-cubes together into a new $S(1, k)$ since they all have exactly $k$ 1’s in the positions that are not $x$. ($S(0, k)$ is no longer needed.) Increment $k$ and repeat.
<table>
<thead>
<tr>
<th>POP</th>
<th>minterm</th>
<th>binary</th>
<th>covered</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$m_0$</td>
<td>0000</td>
<td>x</td>
</tr>
<tr>
<td>1</td>
<td>$m_2$</td>
<td>0010</td>
<td>x</td>
</tr>
<tr>
<td>1</td>
<td>$m_8$</td>
<td>1000</td>
<td>x</td>
</tr>
<tr>
<td>2</td>
<td>$m_3$</td>
<td>0011</td>
<td>x</td>
</tr>
<tr>
<td>2</td>
<td>$m_6$</td>
<td>0110</td>
<td>x</td>
</tr>
<tr>
<td>2</td>
<td>$m_9$</td>
<td>1001</td>
<td>x</td>
</tr>
<tr>
<td>2</td>
<td>$m_{10}$</td>
<td>1010</td>
<td>x</td>
</tr>
<tr>
<td>3</td>
<td>$m_7$</td>
<td>0111</td>
<td>x</td>
</tr>
<tr>
<td>3</td>
<td>$m_{13}$</td>
<td>1101</td>
<td>x</td>
</tr>
</tbody>
</table>

### 1-cubes

<table>
<thead>
<tr>
<th>Cells</th>
<th>binary</th>
<th>covered</th>
</tr>
</thead>
<tbody>
<tr>
<td>0,2</td>
<td>00x0</td>
<td>x</td>
</tr>
<tr>
<td>0,8</td>
<td>x000</td>
<td>x</td>
</tr>
<tr>
<td>2,3</td>
<td>001x</td>
<td>x</td>
</tr>
<tr>
<td>2,6</td>
<td>0x10</td>
<td>x</td>
</tr>
<tr>
<td>2,10</td>
<td>x010</td>
<td>x</td>
</tr>
<tr>
<td>8,9</td>
<td>100x</td>
<td>x</td>
</tr>
<tr>
<td>8,10</td>
<td>10x0</td>
<td>x</td>
</tr>
<tr>
<td>3,7</td>
<td>0x11</td>
<td>x</td>
</tr>
<tr>
<td>6,7</td>
<td>011x</td>
<td>x</td>
</tr>
<tr>
<td>9,13</td>
<td>1x0x</td>
<td>x</td>
</tr>
</tbody>
</table>
Build 2-cubes from 1-cubes by comparing a series of cubes in \( S(1, k) \) to a series in \( S(1, k + 1) \). (Compare even if either has been already marked as covered.) In this case, we need only compare \( c_i \) in \( S(1, k) \) to \( c_j \) in \( S(1, k + 1) \) if they have an \( x \) in the same position. If the cubes differ in exactly one non-\( x \) position mark the 1-cubes as covered and record a new 2-cube as a binary digit with exactly 2 \( x \)'s (one in the common \( x \) position and one in the differing bit position). Also record the minterm indices involved by taking the union of the minterm indices of the two 1-cubes. NOTE: The 2-cube need not be recorded if it is in fact not new, i.e. if an earlier comparison of \( S(1, k) \) to \( S(1, k + 1) \) already generated it. After completing the comparison of \( S(1, k) \) to \( S(1, k + 1) \) group the new 2-cubes together into a new \( S(2, k) \) since they all have exactly \( k \) 1’s in the positions that are not \( x \). Increment \( k \) and repeat.
For our example, this process is fairly simple. Comparing \( S(1,0) \) to \( S(1,1) \) requires first comparing 0,2 to only 8,10 which forms a new 2-cube 0,2,8,10. 0,8 is then compared only to 2,10 which generates the same 2-cube. Comparing \( S(1,1) \) to \( S(1,2) \) generates only one new 2-cube — 2,3,6,7. Note that 8,9 and 9,13 remain uncovered.

<table>
<thead>
<tr>
<th>2–cubes</th>
<th>Cells</th>
<th>binary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0,2,8,10</td>
<td>x0x0</td>
</tr>
<tr>
<td></td>
<td>2,3,6,7</td>
<td>0x1x</td>
</tr>
</tbody>
</table>
• All cubes that remain uncovered are prime implicants and the expression is easily generated from the binary representation. For the example,

\[ F = x'z' + w'y + wy'z + wx'y' \]

• The algorithm is easily automatable.

• don’t care conditions are easily added – treat \( d \) as a 1 for purposes of Quine-McCluskey except that no prime implicant of all \( d \)'s should be accepted.
Prime Implicant Table

In order to perform the next task – identifying a set of prime implicants that give a minimal realization – we must associate the prime implicants of \( f \) with the minterms of \( f \). This is done via the prime implicant table.

For our example, \( F(w, x, y, z) = \sum(0, 2, 3, 6, 7, 8, 9, 10, 13) \) the initial table is with essential rows (PI’s) marked (rows containing a minterm which has a column with a single mark) and distinguished columns (union of the pattern of essential rows) marked. For this case, the essential PI’s generate a minimal realization.

<table>
<thead>
<tr>
<th>PI</th>
<th>0</th>
<th>2</th>
<th>3</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>* 0,2,8,10 (x0x0)</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 2,3,6,7 (0x1x)</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8,9 (100x)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 9,13 (1x01)</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>
So the PI Table indicates that the 8,9 term can be removed. This is clear from examining the K-map table.

We can demonstrate the algebraic reduction that the Quine-McCluskey has performed automatically. The complete sum can be reduced

\[
F(w, x, y, z) = \sum(0, 2, 3, 6, 7, 8, 9, 10, 13)
\]

\[
F = x'z' + w'y + wy'z + wx'y'
\]

\[
Comm. + Assoc. = w'y + [z(wy') + z'x' + (wy')x']
\]

\[
Cons. = w'y + z(wy') + z'x'
\]

\[
Comm. = x'z' + w'y + wy'z
\]

So the term for 8,9 (100x) is redundant.

Note that this analysis could now be performed automatically.

When the essential PI’s do not generate a minimal realization we must transform the PI table further.