Below is a **step‐by‐step walkthrough** for creating a Karnaugh map (K‐map) from a truth table for a specific output (here, ) when you have three inputs :

**1. Label the K‐map Axes**

1. **Determine the size of the K‐map**:
   * With 3 inputs, you’ll have cells.
2. **Assign one input to the rows** (often ):
   * The row for is on top; the row for is on the bottom.
3. **Assign the other two inputs to the columns** (often ):
   * Use **Gray code** order to ensure adjacent columns differ by only one bit. A common sequence is:

The resulting K‐map grid should look roughly like this:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | (X1X0) 00 | 01 | 11 | 10 |
| X2 = 0 | ? | ? | ? | ? |
| X2 = 1 | ? | ? | ? | ? |

**2. Scan the Truth Table**

1. **Go down each row** in the truth table for .
2. **Check the value of**  in that row.
   * If , you will put a 1 in the corresponding K‐map cell.
   * If , you will put a 0.

**Mapping each row** means:

* Identify which cell of the K‐map corresponds to that combination.
* Fill in 1 or 0 depending on the table’s value.

**3. Fill All Eight Cells**

1. **Repeat for all 8 rows** of the truth table (for 3 variables).
2. When you’re done, you’ll have an **8‐cell map** with a pattern of 1s and 0s exactly matching the truth table’s column—but laid out so that cells which differ by only one input bit are adjacent.

**4. Group the 1‐Cells (Optional Next Step)**

* **If you’re simplifying** the Boolean expression for , look for **groups of adjacent 1‐cells** (in powers of 2: 1, 2, 4, or 8). Each grouping will let you eliminate variables that change within that group.
* **Write down** the simplified expression from these groupings.

**That’s it!** By following this procedure—labeling axes, going row by row in the truth table, and filling in the corresponding map location—you end up with a clear visual representation of when is 1 or 0. From there, you can apply K‐map techniques to simplify the logic.