

User's Guide and Reference Manual for

# **LogicAid<sup>TM</sup>**

*Second Edition for Windows<sup>TM</sup>*

CAD Software for Logic Design

**Charles H. Roth, Jr**

*University of Texas at Austin*

April 10, 2003

# Table of Contents

---

<b>Table of Contents .....</b>	<b>ii</b>
<b>Preface.....</b>	<b>iv</b>
<b>LogicAid User's Guide .....</b>	<b>1</b>
1. Introduction.....	1
2. Built-In Help .....	1
3. Entering Logic Equations .....	2
4. Simplifying Logic Equations .....	3
5. Entering Truth Tables .....	4
6. Entering PLA Tables .....	6
7. Entering Minterm and Maxterm Expansions.....	6
8. Entering Karnaugh Maps .....	8
9. Tutorial Mode for Karnaugh Maps .....	11
10. Entering State Tables .....	13
11. State Table Reduction.....	15
12. State Assignment .....	15
13. Derivation of Flip-Flop Input Equations .....	16
14. Checking State Tables .....	17
15. Entering State Graphs .....	18
Converting to State Table .....	20
Using Alphanumeric Inputs .....	20
Multiple Labels on an Arc .....	21
16. Checking Partial State Graphs .....	21
17. Programming a PAL .....	23
18. Entering SM Charts .....	25
19. Design Examples .....	27
Design of a Sequence Detector .....	27
Dice Game Controller .....	29
20. Using LogicAid for Asynchronous Design Problems .....	32
<b>LogicAid Reference Manual .....</b>	<b>35</b>
1. LogicAid Menus & Windows.....	35
2. The File Menu.....	36
2.1. New (Alt+F N or Ctrl+N) .....	36
2.2. Open (Alt+F O or Ctrl+O) .....	37
2.3. Open Text (Alt+F T).....	37
2.4. Close (Alt+F C) .....	37
2.5. Save (Alt+F S or Ctrl+S) .....	38
2.6. Save As (Alt+F A) .....	38
2.7. Print (Alt+F P or Ctrl+P) .....	38
2.8. Print Preview (Alt+F V) .....	38
2.9. Print All (Alt+F L).....	38

2.10. Print Setup (Alt+F R).....	39
2.11. Exit (Alt+F X).....	39
3. The Edit Menu .....	39
3.1. Undo (Ctrl+Z).....	39
3.2. Cut (Ctrl+X).....	39
3.3. Copy (Ctrl+C).....	40
3.4. Paste (Ctrl+V).....	40
4. The Input Menu .....	40
4.1. Terms (Alt+I R or Ctrl+R).....	40
4.2. Equations (Alt+I E or Ctrl+E) .....	46
4.3. Truth Table (Alt+I T or Ctrl+T) .....	48
4.4. PLA Table (Alt+I L or Ctrl+Shift+L).....	51
4.5. State Table (Alt+I S or Ctrl+B) .....	51
4.6. State Graph (Alt+I G or Ctrl+G) .....	55
4.7. SM Chart (Alt+I M or Ctrl+M) .....	61
4.8. Karnaugh Map (Alt+I K or Ctrl+K) .....	64
4.9. Convert to State Table (Alt+I C) .....	68
4.10. Convert to PLA Table (Alt+I V).....	69
4.11. Check Syntax (Alt+I Y or Ctrl+Y) .....	69
4.12. Change Parameters (Alt+I P or Ctrl+Shift+P).....	69
4.13. Change Reset State (Alt+I A) .....	69
5. The Routine Menu .....	70
5.1. Simplification (Alt+R L or Ctrl+L) .....	70
5.2. State Reduction (Alt+R R or Ctrl+R).....	72
5.3. State Assignment (Alt+R A or Ctrl+A) .....	73
5.4. Flip-Flop Equations (Alt+R F or Ctrl+F) .....	75
5.5. State Table Checker (Alt+R K or Ctrl+Shift+K).....	76
5.6. Partial Graph Checker (Alt+R G or Ctrl+Shift+G) .....	77
5.7. Select New Solution File (Alt+R N or Ctrl+Shift+N).....	78
5.8. Make Jedec File (Alt+R J or Ctrl+J).....	78
5.9. Download Jedec File (Alt+R D or Ctrl+D) .....	80
6. Windows Menu.....	81
7. The KMap Menu.....	82
7.1 Derive Min Equation from Map (Alt+K Q or Ctrl+Q).....	82
7.2 Plot Map from Equation (Alt+K M or Ctrl +Shift+M).....	82
7.3 Check Solution (Alt+K S or Ctrl+Shift+S) .....	83
7.4 Enter Tutorial Mode (Alt+K T or Ctrl+Shift+T).....	83
8. The Help Menu .....	86
8.1 LogicAid Help .....	86
8.2 About LogicAid (Alt+H A) .....	86
<b>LogicAid Index .....</b>	<b>87</b>

# Preface

---

As the name implies, LogicAid is a program that serves as an aid to the logic designer. It relieves the designer of many of the tedious computational tasks associated with the logic design process. Its functions include simplification of logic functions, working with state graphs and tables, and derivation of flip-flop input equations. The software provides a quick way to try different design alternatives.

LogicAid introduces students to the use of the computer in the logic design process. It enables them to solve larger, more practical design problems than they could by hand. LogicAid provides tutorial aids for students who are learning to use Karnaugh maps and state graphs. They can also use the software to verify solutions that they have worked out by hand.

LogicAid was originally developed for use in logic design classes at the University of Texas. The software was developed for use with *Fundamentals of Logic Design*, 5th ed., by Charles H. Roth (Brooks/Cole, 2003). However, it can be used with almost any logic design textbook.

The Second Edition of LogicAid has an improved user interface and many other new features. The allowable number of input and output variables has increased and addition of the Espresso algorithm allows faster simplification of PLA tables and logic functions. The SM chart input provides an alternative to using state graphs. JEDEC files for programming 22V10 PALs can be generated and downloaded to a PAL programmer.

The first section of this manual is a LogicAid User's Guide that provides step-by-step instructions for using the various program features. The next section is a LogicAid Reference Manual that explains each command in detail.

## Solution Files

The LogicAid folder on the CD includes files that contain encoded solutions to the problems in Chapters 14, 16, and R2 of *Fundamentals of Logic Design*, 4th Ed. These files are stored in folders labeled Probs-14, Lab-16, and Probs-R2. Although these files cannot be read directly, they can be used to check your solutions to state table and state graph problems as explained in sections 14 and 16 of the LogicAid User's Guide.

## Installation

Be sure to read the release notes, LogicAid\_readme.rtf, before you install the software. LogicAid, 2nd ed. is compatible with Windows 95, 98, NT 4, 2000, ME, and XP. Run Install\_LogicAid.msi to install the program onto your hard drive. To install LogicAid, insert the CD ROM in the drive, and click on Install\_LogicAid.msi. If LogicAid fails to install properly, run setup.exe in the LogicAid folder.

# LogicAid User's Guide

---

## 1. Introduction

LogicAid is a computer-aided design program that performs many functions commonly needed in the logic design process. It is capable of simplifying Boolean functions specified as equations, terms, a truth table, a PLA table, or Karnaugh maps. Given a state table, LogicAid provides routines to do state reduction, to specify state assignments, to generate flip-flop input equations, and to compare with another state table for functional equivalence. In addition, LogicAid accepts state graphs and SM charts as inputs.

LogicAid has two tutorial modes. The Karnaugh map tutorial mode helps you learn to solve Karnaugh maps. The partial graph checker helps you to learn to derive state graphs for sequential networks. LogicAid also allows you to check the answers to state table problems without revealing the correct answer.

LogicAid uses three types of windows. The input window is used for entering data, the output window displays results from the logic-design routines, and the graphics window is used for Karnaugh maps, state graphs, and SM charts. Up to three windows of each type can be open at the same time. The windows are color-coded so that windows derived from a given input window have the same color stripe on the left side. For example, a state table and output equations derived from a state graph would have the same edge color as the state graph.

This manual assumes that you are familiar with basic Windows operations including using the mouse, entering and editing text, and working with menus, dialog boxes, windows, etc. The manual is divided into two main parts. This part is a *User's Guide*, which gives step-by-step instructions for carrying out each of the major functions performed by LogicAid. The second part is a *Reference Manual*; which provides a detailed explanation of every command. Throughout this manual, a dash (–) is used to represent a “don't care” or a missing variable.

The first time you install your copy of LogicAid using the setup program, a dialog box will appear on the screen so that you can personalize your copy. Type your name in the space provided and click OK. Then your name will be printed at the top of any output you print from your copy of LogicAid, and your name will be saved on any files you create.

## 2. Built-In Help

LogicAid provides a limited amount of built-in help using the standard Microsoft help system. To access this help, select **LogicAid Help** from the **Help** menu, and then select an item from the help contents, or you can click on **Index** and search the help index. Help provides a brief summary of the required input formats for equations, truth tables, terms, Karnaugh maps, state tables, state graphs, and SM charts. For more details, refer to this *User's Guide* or to the *Reference Manual*.

Help can also be accessed from any window in LogicAid by pressing the **F1** key. Help for the currently active window is then displayed. For example, if you press **F1** when the state table window is active, help for entering state tables will show up.

### 3. Entering Logic Equations

LogicAid will accept Boolean equations in either sum-of-products or product-of-sums form and then simplify these equations. The steps required to enter equations are as follows:

1. Pull down the **Input** menu and select **Equations** (or type **CTRL+E**).
2. When the Input Equations Format dialog box appears, type in the number of input variables if it is different from the default value of 4. If you want to enter more than one function, hit **Tab** and then enter the number of functions. The default input variable names are X1, X2, X3, ..., and the default function names are Z1, Z2, Z3, .... If you wish to change any of these names, select **Enter Names**. If you want to enter your equation(s) in product-of-sums form instead of sum-of-products form, click on the appropriate button. Then click **OK**.
3. If you have selected **Enter Names**, the Input Variable Names dialog box will appear next. Change any names that you want to be different from the default values and then click **OK**. (You may advance from box to box by using the **Tab** key.)
4. An input window titled "Equations" appears next. Type the equations into the window.

Here are some tips for making your input acceptable to LogicAid:

- You may use the standard Windows editing functions (Cut, Copy, Paste, etc.) as required.
- Each equation must start with a function name followed by "=", and must end with a period.
- The input is case sensitive, so that capital and lower case letters are distinguished. In particular, if default names are used, X and Z must be capitalized in X1, Z1, etc.
- Each variable name should be followed by a space, a complement ('), a plus sign, or another separator.

Examples of equations entered in sum-of-products form are:

$$F1 = A C' + A B'D + A'B'C + A'C D' + B'C'D'.$$

$$F2 = B C'D' + A B C' + A C'D + A B'D + A'B D'.$$

Examples of equations entered in product-of-sums form are:

$$F1 = (A' + C' + D) (A + B' + C) (A + C + D') (B' + C' + D').$$

$$F2 = (B + D) (A + D') (A' + B' + C').$$

In product-of-sums form, a single variable must be enclosed in parentheses:

$$F3 = (A + B) (C) (D') .$$

NOTE: If an equation is too long to fit on a single line, you may type **Enter** after any term to break the line.

- Next you may select **Check Syntax** from the **Input** menu (or type **ALT+I Y** or **CTRL+Y**). If you have made any syntax errors, a dialog box will appear which explains the type of error. Click **OK**, and the cursor in the input window will indicate the approximate location of the error. Edit the input equations to correct the error(s).

If there are no errors in your equation, you receive no message.

- After correcting the errors, you may save the data in the edit window as a LogicAid file by selecting **Save** from the **File** menu. All LogicAid file names must have an extension **.AID**.

## 4. Simplifying Logic Equations

LogicAid provides three algorithms for simplifying logic equations. *PI Chart* uses a procedure similar to the Quine-McCluskey method to find one irredundant solution. It first uses the iterated consensus method to find all of the prime implicants of a function, then uses a prime implicant chart to find an irredundant solution. The solution is usually, but not always, minimum. *Petrick* may be used to find all minimum solutions for functions of six or fewer variables, but it is slower and uses more memory than PI Chart. Petrick finds all prime implicants using iterated consensus, then uses Petrick's method to find the minimum solutions. When finding these solutions, Petrick minimizes the number of gate inputs for a two-level network. *Espresso* is faster and should be used when the number of input variables is greater than ten or multiple-output minimization is required.

After you have entered one or more logic equations, use the following procedure to derive simplified equations:

- Select **Simplification** from the **Routine** menu (or type **ALT+R L** or **CTRL+L**).
- When the Simplification Options dialog box appears, you may change the output format if you wish. Then choose the simplification method. Unless you really need all minimum solutions, it is better to select **Petrick—Maximum of 5 Minimal Solutions** instead of **Petrick—All Minimal Solutions**. Select any other desired options and click **OK**.
- Before simplification begins, the input equations will be syntax checked. If any syntax errors occur, correct them by editing the input window, and then select **Simplification** again.
- The output window will then appear on the screen. Problems with six or fewer variables will usually start generating output in a short time. Problems with more variables may take times of several minutes or longer, depending on the complexity of the problem and the algorithm used. If you wish to terminate the

computation before it is finished, type **Control-Period** (Ctrl+.). The computation will usually be aborted in a short time.

5. You can print both the edit and output windows by selecting **Print All** from the **File** menu. Select **Print** to print only the window on top. The output for the first input example above, with the **Identify Essential Prime Implicants** and **Display Input and Gate Costs** options selected, is as follows:

```

F1  =  A C'* + A'C D'* + B'C'D' + B'C D
      Input Cost = 15          Gate Cost = 5

F1  =  A C'* + A'C D'* + A'B'D' + B'C D
      Input Cost = 15          Gate Cost = 5

F2  =  A B'D* + A'B D'* + A B C'
      Input Cost = 12          Gate Cost = 4

```

Essential prime implicants are identified by an asterisk (\*).

6. When you close the output window, the program will give you an opportunity to save the changes (select **Yes**), discard the changes (select **No**), or **Cancel** the **Close** command.

## 5. Entering Truth Tables

Given a truth table as input, LogicAid will find simplified equations for the functions defined by the truth table. The steps required to enter a truth table are as follows:

1. Select **Truth Table** from the **Input** menu (or type **CTRL+T**).
2. When the Input Truth Table Format dialog box appears, type in the number of input variables if it is different from the default value of 4.
3. If your table has more than one output function, hit **Tab** and then enter the number of functions.
4. The default input variable names are X1, X2, X3, ...; the default function names are Z1, Z2, Z3, .... If you wish to change any of these, select **Enter Names**.
5. The default format for the input combinations in the truth table is Straight Binary Order. If you prefer to use a different input format, select **Decimal**, **Binary** (0, 1, -), **Hexadecimal**, or **Octal**. The default for the output combinations is **Binary** (0, 1, -), but **Decimal**, **Hexadecimal**, or **Octal** may also be selected. After the table has been entered, output values for the remaining rows will default to all 0's unless **1-terms** or **X-terms** is selected.
6. When you have finished selecting formats, click **OK**.
7. If you have selected **Enter Names**, the Input Variable Names dialog box will appear next. Change any names that you want to be different from the default values and then click **OK**. (You may advance from box to box by using the **Tab** key.)



8. An input window titled “Truth\_Table” appears next. The header at the top of the window lists the input variable names and output function names. Enter the truth table below this header. If you have selected the default format for input combinations (straight binary order), the binary input combinations will automatically be generated for you. For example, for three input variables, 000 will appear on the screen.

Type in the corresponding output values and hit **Enter**. Then type in the output values corresponding to row 001, etc. When you have entered the output values for the last row in your table, hit **F12** instead of **Enter**.

NOTE: Once you hit **F12**, the program switches to manual mode and any further input combinations must be typed in manually. The following example shows a truth table with three inputs and three outputs (binary format):

```
Input Variables:  X1 X2 X3
Output Functions: Z1 Z2 Z3
```

000	100
001	010
010	00-
011	001
100	011
101	-1-

9. If you do not use the default format (straight binary order), then for each row of the truth table, type the input combination followed by a space or **Tab**, and then type the output values. The following example shows the input for the same truth table using decimal format for the input combinations:

```
Input Variables:  X1 X2 X3
Output Functions: Z1 Z2 Z3
```

0	100
1	010
2	00-
3	001
4	011
5	-1-

10. Next, you may select **Check Syntax** from the **Input** menu (or type **ALT+I Y**). If you have made any syntax errors, a dialog box will appear which explains the type of error. Click **OK**, and the cursor in the input window will indicate the approximate location of the error. Edit the truth table to correct the error(s). The header cannot be changed in the edit window.

If there are no errors in your truth table, you receive no message.

Select **Change Parameters** from the **Input** menu to change the input or output variable names.

11. Follow the procedure in Section 4, *Simplifying Logic Equations*, to derive simplified expressions for the functions defined by the truth table.

## 6. Entering PLA Tables

Given a PLA table as input, LogicAid will find simplified equations for the functions defined by the PLA table, or it will find a reduced PLA table. The format for entering a PLA table is similar to that for a truth table. The steps required to enter a PLA table are as follows:

1. Select **PLA Table** from the **Input** menu (or type **CTRL+shift+L**).
2. When the Input PLA Table Format dialog box appears, type in the number of input variables and the number of output functions. The default input variable names are X1, X2, X3, ...; the default function names are Z1, Z2, Z3, .... If you wish to change any of these, select **Enter Names**. The format for both inputs and outputs is Binary (0, 1, -). When finished, click **OK**.
3. If you have selected **Enter Names**, the Input Variable Names dialog box will appear next. Change any names that you want to be different from the default values and then click **OK**. (You may advance from box to box by using the **Tab** key.)
4. An input window titled "PLA\_Table" appears next. The header at the top of the window lists the input variable names and output function names. Enter the PLA table below this header. For each row, type the input values followed by a space or tab, followed by the output values. The following example shows a PLA table with four inputs and three outputs:

```
Input Variables:  X1 X2 X3 X4
Output Functions: Z1 Z2 Z3
```

---

```
0-11   110 {first row}
001-   001
01-0   101
10--   --0 {last row}
```

5. Next you may select **Check Syntax** from the **Input** menu (or type **ALT+I Y**). If you have made any syntax errors, a dialog box will appear which explains the type of error. Click **OK**, and the cursor in the input window will indicate the approximate location of the error. Edit the PLA table to correct the error(s). The header cannot be changed in the edit window. Select **Change Parameters** from the **Input** menu to change the input or output variable names.
6. Follow the procedure in Section 4, *Simplifying Logic Equations*, to derive simplified expressions for the functions defined by the PLA table. Use Espresso to derive a reduced PLA table.

## 7. Entering Minterm and Maxterm Expansions

LogicAid also accepts Boolean functions that are expressed as a sum of minterms or a product of maxterms. The steps required to enter functions in minterm or maxterm form are as follows:

1. Select **Terms** from the **Input** menu (or type **CTRL+R**).
2. When the Input Equations Format dialog box appears, type in the number of input variables if it is different from the default value of 4. If you want to enter more than one function, hit **Tab** and then enter the number of functions. The default input variable names are X1, X2, X3, ..., and the default function names are Z1, Z2, Z3, .... If you wish to change any of these, select **Enter Names**. The default input format for minterms or maxterms is decimal. If you wish to use a different format, select **Binary**, **Hexadecimal**, or **Octal**. Then click **OK**.
3. If you have selected **Enter Names**, the Input Variable Names dialog box will appear next. Change any names that you want to be different from the default values and click **OK**. (You may advance from box to box by using the **Tab** key.)
4. An input window titled "Terms" appears next. For each function, type a list of terms into the window followed by a period. The most general input format is:

list of 1-terms, list of 0-terms, list of X-terms

where "1-terms" represent input combinations for which the function value is 1, "0-terms" represent input combinations for which the function value is 0, and "X-terms" represent input combinations for which the function value is a "don't care." Typically, one or two of the three lists may be omitted.

One of the following formats may be used:

- a. Given a minterm expansion with no "don't cares," enter a list of 1-terms.

Example: If  $F1 = \sum m(0,5,9,11,13)$ , enter 0 5 9 11 13.

NOTE: Terms in the list are separated by spaces and the list is terminated by a period.

- b. Given a maxterm expansion with no "don't cares," enter a list of 0-terms preceded by a comma. (The comma which precedes the list indicates that 1-terms have been omitted.)

Example: If  $F2 = \prod M(1,6,7,8,14)$ , enter ,1 6 7 8 14.

- c. Given a minterm expansion with "don't cares," use the following format:

list of 1-terms,, list of X-terms.

(Two commas are used to indicate that 0-terms are omitted.)

Example: If  $F3 = \sum m(1,3, 5, 7) + \sum d(2,4,12)$ , enter 1 3 5 7,, 2 4 12.

- d. Given a maxterm expansion with don't cares, use the following format:

, list of 0-terms, list of X-terms.

Example: If  $F4 = \prod M(0,3,9,13) \cdot \prod D(1,5,7)$ , enter ,0 3 9 13, 1 5 7.

Refer to Section 4.1 of the *Reference Manual* for other possible input formats.

5. Next you may select **Check Syntax** from the **Input** menu (or type **ALT+I Y**). If you have made any syntax errors, a dialog box will appear which explains the type of error. Click **OK**; the cursor in the input window will indicate the approximate location of the error. Edit the lists of terms to correct the error(s).  
If there are no errors, you receive no message.
6. Follow the procedure in Section 4, *Simplifying Logic Equations* to derive simplified expressions for the functions defined by the minterm and maxterm expansions.

## 8. Entering Karnaugh Maps

Given a Karnaugh map as an input, LogicAid will derive the minimum equation(s) from the map, or given an equation, LogicAid will plot the map. The Karnaugh map window has two sections—one for entering a map and one for entering the associated equations. If you want to use LogicAid to input a Karnaugh map, you have several options: (1) type in the equation and ask LogicAid to plot the map for you (2) type in the equation and plot the map; then ask LogicAid to check the solution (3) plot the map and ask LogicAid to generate all the minimum equations (4) enter the tutorial mode and follow the instructions on the screen to find a correct minimum equation for a map. The steps required to enter a Karnaugh map are given below. Since the **Enter** and **F12** keys generally perform different functions, be sure to press the specified key when following these instructions.

1. If you are learning to use Karnaugh maps, you may want to use the tutorial mode to help you. If so, select **Enter Tutorial Mode** from the **KMap** menu (see Section 9, *Tutorial Mode for Karnaugh Maps*). Otherwise, select **Karnaugh Map** from the **Input** menu.
2. When the Input Karnaugh Map Format dialog box appears, enter the number of input variables if it is different from the default value. The default variable names are A, B, C, D, E and the default output function name is F. If you want to change any of these, select **Enter Names**. If the number of inputs is 5, you have to select the output form, which could be **mirror image form** or **diagonal form**.
3. If you have selected **Enter Names**, the Input Variable Names dialog box will appear. For the Karnaugh map input, the max length of the input variable and output function names is 2 characters. Change any of names that you want to be different from the default name and click **OK**.
4. At this point, a graph window will appear. The graph window is divided to 3 parts. The left part is the side menu which shows a menu of 8 items; the right upper part is the map zone used to enter a map, and the right lower part is the equation zone used to enter equations .
5. If you want to plot a map, select **Plot 1's**, **Plot 0's** or **Plot X's** from the side menu. If you select **Plot 1's**, click on any square of the map to enter a **1**. If you drag the mouse, the squares covered by the drag area will all be filled with **1's**. If you select **Plot 0's**, **0's** will be entered, and if you select **Plot X's**, **X's** (don't

cares) will be entered. In each case, selecting a square that has already been plotted will erase the value in that square.

6. Here are some guidelines on how to fill in the map:
  - When you select **Fill 0's** from the side menu, the blanks of the map will all be filled with **0's**; if you select **Fill 1's**, the blanks will all be filled with **1's**.
  - Every map must have at least one **1** or one **0**, but it is not necessary to fill all of the squares.
  - If you only enter **1's**, the blanks will be considered as **0's**; if you only enter **0's**, the blanks will be considered as **1's**.
  - If you enter **1's** and **0's**, the blanks will be **X's**.
  - If you enter **1's**, **0's** and **X's**, there should be no blanks in the map.
7. After you finish entering the data of the map, you can select **Loop** from the side menu to loop the terms. The way to enter loops is similar to the way you enter data. You can click the selected square or drag the mouse to select several squares. If select a square by a mistake, click on the square again to deselect it.
8. After your selection, the terms will be highlighted and you can hit **Enter** to ask LogicAid to check the selected terms. If the selected terms can be combined to an implicant, LogicAid will erase the highlights and loop these terms on the map; otherwise, it will show an error message.
9. If you want to enter an equation, select **Text** from the side menu or click the mouse in the equation zone to switch the side menu to **Text**.

Here are some tips for making your input acceptable to LogicAid:

- You can enter equations in the equation zone only in the Text mode.
- The equations have the same syntax rules as the **Equation Input** (Section 3, *Entering Logic Equations*).
- Each equation must start with a function name followed by “=” . Each variable name should be followed by a space, a complement ('), a plus sign, or another separator. A period at the end is optional. Example of sum-of-products form:

$$F = A C' + A B'D + A'B'C + A'C D' + B'C'D'$$

If product-of-sums form is used, each term or single variable must be enclosed in parentheses:

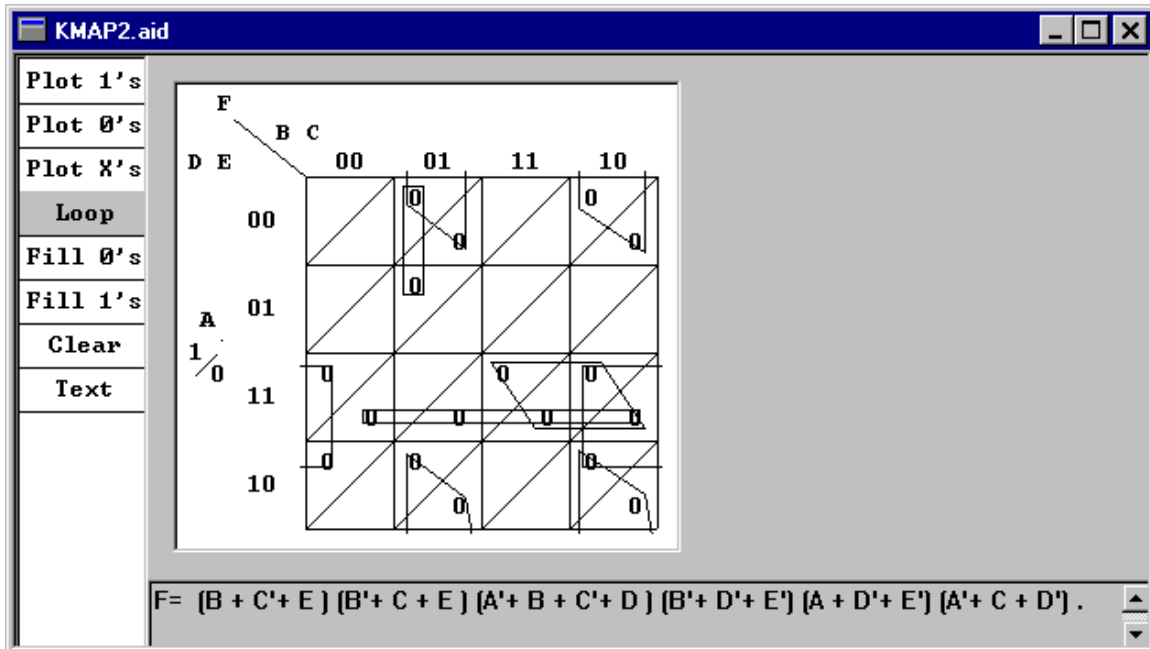
$$F = (A + B') (C) (A' + B + D')$$

10. If you select **Clear** from the side menu, a dialog box will give you four choices: **Clear loops**, **Clear equations**, **Enter new map**, and **Read map from file**. If you select either Enter new map or Read map from file, all the old entries of the map and the output equations will be cleared. Only a blank Karnaugh map will be left.

**NOTE:** If you select **Enter new map**, and the new map window opens partially obscured, select **Window, Tile**, to reposition the window.

11. After you finish entering the output equation or plotting the map, you can select items from the **KMap** menu.
12. If you only enter an equation, you can select **Plot Map from Equation** from the **KMap** menu, and LogicAid will plot the map from the last equation in the equation section of the window.





## 9. Tutorial Mode for Karnaugh Maps

The tutorial mode is designed to help you learn how to derive minimum solutions from Karnaugh maps. In this mode, LogicAid will check your work at each step and give you appropriate error messages if you make a mistake. This tutorial assumes that you already know the basics of using Karnaugh maps. In particular, you should be able to plot terms on a map, to read terms from a map, and to identify prime implicants and essential prime implicants on the map. If you are not familiar with these operations, please study the section on Karnaugh maps in *Fundamentals of Logic Design* or another logic design textbook before proceeding.

1. Select **Enter Tutorial Mode** from the **KMap** menu (or type **ALT+K T**) and set the parameters for the Karnaugh map. Refer to Section 8, *Entering Karnaugh Maps*, steps 2-4.
2. A dialog box for the tutorial options will appear. Select either the **sum of products** or **product of sums** option for the equations, and then select a terms input option. If you select the first option (**type in each term after looping it**), whenever you loop a correct implicant, you have to type in that implicant as a term of the equation. If you select the second option (**type in all essential prime implicants at once, and type in all remaining prime implicants at once**) you must finish looping all the essential prime implicants before typing in the corresponding terms of the equation. Then you must loop all remaining terms for the minimum solution before you type them in.
3. In the tutorial mode, an instruction will always be displayed at the top of the Karnaugh map window. If you follow the instructions, you can enter a Karnaugh map and then derive the minimum equation correctly.

4. The first instruction is **Enter a new map! (Hit F12 when finished)**. When you see this instruction, plot the map (see Section 8, steps 6 and 7). Enter all your 1's, 0's, and X's on the map, then hit **F12** to tell LogicAid that you are finished plotting the map.

NOTE: The following instructions apply to the first terms input option (**Type in each term after looping it**). If you selected the second input option, go to step 12.

5. The second instruction is **Loop an essential prime implicant! (Hit Enter) (Hit F12 when finished)**.

When you see this instruction, look for essential prime implicants. If you find an essential prime implicant, loop it, then press enter (refer to Section 8, *Entering Karnaugh Maps*, step 7 for instructions).

After you loop the terms and hit **Enter**, LogicAid will either tell you the looping is correct or it will display the reason for the error. When the loop is correct, continue with step 6.

If there are no essential prime implicants, go to step 7.

6. In this step, the instruction is **Type in the selected term (Hit Enter)**. Type in the expression for the looped term in the equation zone, then hit **Enter**. Use the same syntax rules you used in Section 3, *Entering Logic Equations*, typing in all the plus signs, parentheses, and spaces where appropriate.

If you type a wrong term, LogicAid will display the error message, and you can edit the term to correct it. When you type in a correct term, LogicAid automatically takes you back to step 5 to loop more essential prime implicants.

7. When you finish looping and typing in all essential prime implicants, hit **F12**. LogicAid will let you know if you have correctly entered them all. If you receive no error message, continue with step 8. Otherwise, you will have to keep looping prime implicants until you find them all (return to step 5).

8. In this step, the instruction is **Loop a remaining term for minimum solution! (Hit Enter) (Hit F12 when finished)**. This step is similar to step 5, except you loop one of the remaining prime implicants that will lead to a minimum solution instead of looping an essential prime implicant.

If there are remaining terms that have not been looped, loop one of them.

9. After looping a term, the instruction is **Type in the selected term (Hit Enter)**. Type in the looped term in the equation area, then hit **Enter**. When you type in a correct term, LogicAid automatically takes you back to step 8 to loop any remaining terms.

When there are no terms left to loop, go to step 10.

10. When you finish looping and typing in the rest of the terms, hit **F12** to tell LogicAid you have finished all the looping. LogicAid gives you a message saying **Correct Solution** if you have correctly entered a minimum equation. Otherwise, you will have to keep looping terms until you find them all (return to step 8).

11. After you receive the **Correct Solution** message, there are no instructions at the top of the screen. You can select **Exit Tutorial Mode** from the **Kmap** menu to



leave the tutorial mode or you can select **Clear** from the side menu. If you select **Clear**, a dialog box will allow you to select **Clear loops**, **Clear equations**, **Enter new map** or **Read map from file**.

NOTE: The following steps apply if you have selected the second terms input option.

12. The second instruction is **Loop an essential prime implicant! (Hit Enter) (Hit F12 when finished)**.

When you see this instruction, look for essential prime implicants. If you find an essential prime implicant, loop it, then press enter (refer to Section 8, *Entering Karnaugh Maps*, step 7 for instructions).

After you loop a term and hit **Enter**, LogicAid will either tell you the looping is correct or it will display the reason for the error.

When the loop is correct, repeat step 12 until all essential prime implicants have been looped; then hit **F12**.

If there are no essential prime implicants, hit **F12** and then go to the next step.

13. In this step, the message is **Type in all selected terms**. Type in all of the selected essential prime implicants, then hit **F12**. Use the same syntax rules you used in Section 3, *Entering Logic Equations*, typing in all the plus signs, parentheses, and spaces where appropriate. If you receive any error messages, correct the errors and then proceed.

14. In this step, the instruction is **Loop a remaining term for minimum solution! (Hit Enter) (Hit F12 when finished)**. This step is similar to step 12, except you loop all of the remaining prime implicants for a minimum solution instead of looping essential prime implicants.

If there are remaining terms that have not been looped, loop them appropriately.

15. When you finish looping and typing in all the remaining terms, hit **F12**. LogicAid will let you know if you have correctly entered them all. If you receive no error message, continue with step 16. Otherwise, you will have to keep looping prime implicants until you find them all (return to step 5).
16. The next message is **Type in all selected terms**. Type in all the remaining terms you have looped.

When there are no terms left to loop, hit **F12** and go to step 10.

## 10. Entering State Tables

You can use LogicAid to reduce the number of states in a state table, make a state assignment, and derive flip-flop input equations to realize the table. The steps required to enter a state table are as follows:

1. Select **State Table** from the **Input** menu (or type **Ctrl+B**).
2. When the Input State Table Format dialog box appears, enter the number of NS (next state) columns, number of input variables, number of output variables and length of state names if they are different from the default values. The default input variable names are X1, X2, X3, ..., and the default output variable names

are Z1, Z2, Z3, .... If you wish to change any of these, select **Enter Names**. If the output(s) depend only on the present state and not on the input values, select **Moore**. The column headings will be in straight binary order unless you select **Enter Heading**.

3. If you have selected **Enter Names**, the Input Variable Names dialog box will appear. Change any names that you want to be different from the default values and click **OK**. (You may advance from box to box by using the **Tab** key.)
4. If you have selected **Enter Heading**, the Enter Table Heading dialog box will appear. Change any column headings that you want to be different from the default values, then click **OK**.
5. At this point, an input window will appear. The header at the top of the window has labels for the different sections of the table: "PS" for Present State, "NS" for Next State, and "OUTPUTs\*". The next part of the header contains the binary column headings followed by the names of the input variables. Note that the input variable values in each column heading are listed vertically instead of horizontally. The output variable names are listed in a footer at the bottom of the table. The header and footer cannot be changed in the edit window.

Here are some tips for making your input acceptable to LogicAid:

- Type in the table row by row.
- State names may be a single letter, a single number, or any combination of two letters and numbers.
- Start each row with a present state, followed by the next states for each column heading, followed by output values for each column heading.
- Separate state names by one or more spaces or a tab. If you use a tab as a separator, the columns will line up with the headings; however, this is not necessary.
- The group of binary output values for each column must not be separated by spaces, but successive groups of output values must be separated by spaces or a tab.
- You may use the standard Windows editing functions (Cut, Copy, Paste, etc.) as required.

The following example shows the input for a Mealy state table with two input variables, two output variables, four NS (next state) columns, and four states:

PS	NS				OUTPUTs*				INPUT-VAR
	0	0	1	1	0	0	1	1	X1
	0	1	1	0	0	1	1	0	X2
S0	S3	S2	S0	S1	00	10	01	11	
S1	S0	S1	S3	S2	10	10	11	11	
S2	S3	S0	S1	S1	00	10	01	11	
S3	S2	S2	S0	S1	00	00	01	01	
* Z1 Z2									

6. When you have finished typing in the rows of the state table, you may select **Check Syntax** from the **Input** menu (or type **ALT+I Y**). If you have made any syntax errors, a dialog box will appear which explains the type of error. Click

**OK**, and the cursor in the input window will indicate the approximate location of the error. Edit the rows of the table to correct the error(s). When the table is correct, it will be reformatted to line up with the column headings.

## 11. State Table Reduction

LogicAid will reduce a state table with no “don’t cares” to a minimum number of rows by eliminating equivalent states. If the table is incompletely specified (that is, it has “don’t care” next states or outputs), LogicAid will attempt to perform some reduction on the table, but in general it will not find a minimum-row table. After you have entered a state table, use the following procedure for state table reduction:

1. Select **State Reduction** from the **Routine** menu (or type **Ctrl+R**).
2. When the State Reduction Option dialog box appears, select **Edit Window** if you want to replace the current state table with the reduced one. You can then make a state assignment and derive flip-flop equations from the reduced table. By default, the reduced table will be displayed in the output window and cannot be used as an input for further operations.

## 12. State Assignment

If you do not specify a state assignment, the assignment will default to straight binary order. For example, for a table with three state variables, the first row is assigned 000, the next row 001, the next row 010, and so on. If you want to use a different state assignment, proceed as follows:

1. Select **State Assignment** from the **Routine** menu (or type **Ctrl+A**).
2. When the State Assignment Options dialog box appears, the minimum number of state variables required to realize the table appears in the Number of State Variables box. If you want to use more than the minimum number, enter a new value in this box. The default names for the state variables are Q1, Q2, Q3, .... If you wish to change these names, select **Enter Names**. If you do not use the default state assignment or if you wish to change assignments, select **Display or Change Current State Assignment**. If you choose this option, also specify the input format for the state assignment. (The default is binary, but you may want to change to another format.) Then click **OK**.
3. If you have selected **Enter Names**, the State Variable Names dialog box will appear next. Change any names that you want to be different from the default values, then click **OK**.
4. If you selected **Display or Change Current State Assignment**, the default (or current) value of the state assignment for each row will appear in the edit window to the left of the state table. Use normal editing procedures to make any changes in these assigned values. Use the **Tab** key to step to the next assigned value. The following example shows the state table of Section 10 with an added state assignment:

ASSIGN	PS	NS				OUTPUTs*				INPUT-VAR
		0	0	1	1	0	0	1	1	X1
		0	1	1	0	0	1	1	0	X2
00	S0	S3	S2	S0	S1	00	10	01	11	
01	S1	S0	S1	S3	S2	10	10	11	11	
11	S2	S3	S0	S1	S1	00	10	01	11	
10	S3	S2	S2	S0	S1	00	00	01	01	
* Z1 Z2										

- When you have finished editing the state assignment, you may select **Check Syntax** from the **Input** menu (or type **ALT+I Y**). If you have made any errors, a dialog box will appear which explains the type of error. Click **OK**; the cursor in the input window will indicate the approximate location of the error. Edit the state assignments to correct the error(s).

If there are no errors in your state table, you receive no message.

## 13. Derivation of Flip-Flop Input Equations

Given a state table with a state assignment, LogicAid will derive and simplify the flip-flop input equations and output equations required to realize the table. JK, SR, D, or T flip-flops may be selected. Use the following procedure to derive flip-flop input equations:

- Select **Flip-Flop Equations** from the **Routine** menu (or type **Ctrl+F**).
- When the Flip-Flop Equations Options dialog box appears, select the desired type of flip-flop. The current state assignment will be used unless you select the **Display Assignment Dialog Box** option. (If you select this option, go back to Section 10, *Entering State Tables*, step 2.)
- When you select a type of flip-flop and click **OK**, a Simplification Options dialog box will appear. This is the same dialog box described in Section 4, *Simplifying Logic Equations*, step 2, of this manual.
- When you have selected the desired options and clicked **OK**, the simplified flip-flop and output equations will be derived and displayed in the output window. If you wish to terminate the computation before it is finished, type **Control-Period (Ctrl+.)**. The computation will be aborted after a short time.
- You can print both the input and output windows by selecting **Print All** from the **File** menu. Select **Print** to print only the window on top. The output for the above state table example, with the **Display Input and Gate Costs** option selected, is as follows:

$$J(Q1) = X1'Q2' + X1 Q2$$

Input Cost = 6                      Gate Cost = 3

$$K(Q1) = X2 Q2 + X1$$

Input Cost = 4                      Gate Cost = 2

$$J(Q2) = X1'X2 + X1 X2' + X1'Q1$$

Input Cost = 9                      Gate Cost = 4

$$K(Q2) = X1'X2' + X1 X2 Q1' + X1'Q1$$

Input Cost = 10                      Gate Cost = 4

```

Z1 = X1'X2 Q1' + X1 X2'Q1' + Q1'Q2 + X1'X2 Q2 + X1 X2'Q2
    Input Cost = 19          Gate Cost = 6

Z2 = X1
    Input Cost = 0          Gate Cost = 0

***Total Input Cost = 48***
***Total Gate Cost = 19***

```

## 14. Checking State Tables

LogicAid allows you to compare two state tables, M1 and M2, to see if they are functionally equivalent. If the tables are completely specified (that is, they have no “don’t cares”), LogicAid will check to see if the reset state of M1 (first row of the table) is equivalent to the reset state of M2. If they are equivalent, and you start both M1 and M2 in their reset states, for any input sequence the output sequences will be the same. In this case, M1 and M2 perform the same function and one can be used in place of the other. Refer to Section 5.5 of the *LogicAid Reference Manual* for discussion of the incompletely specified case. Use the following procedure to check two state tables for functional equivalence:

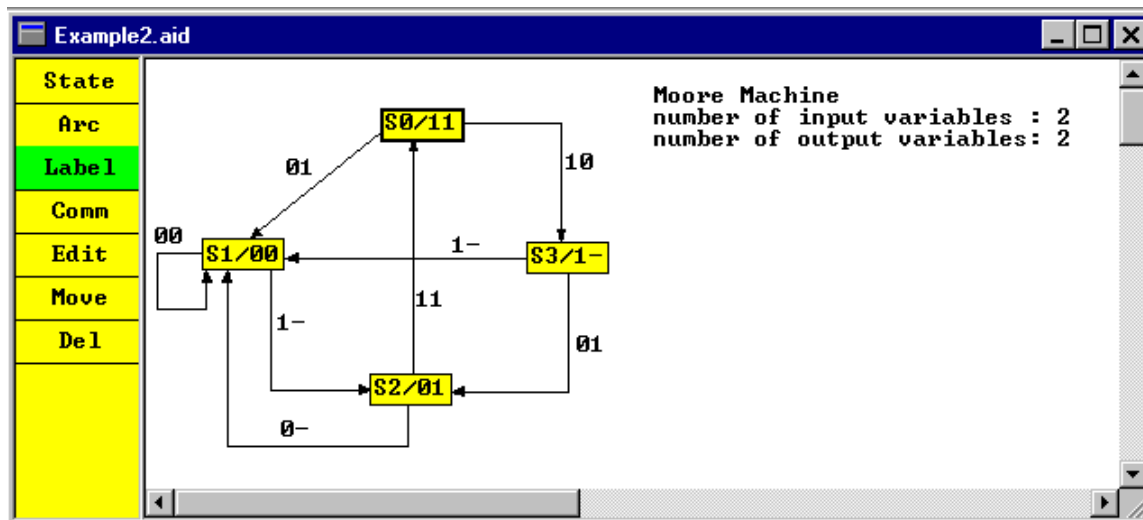
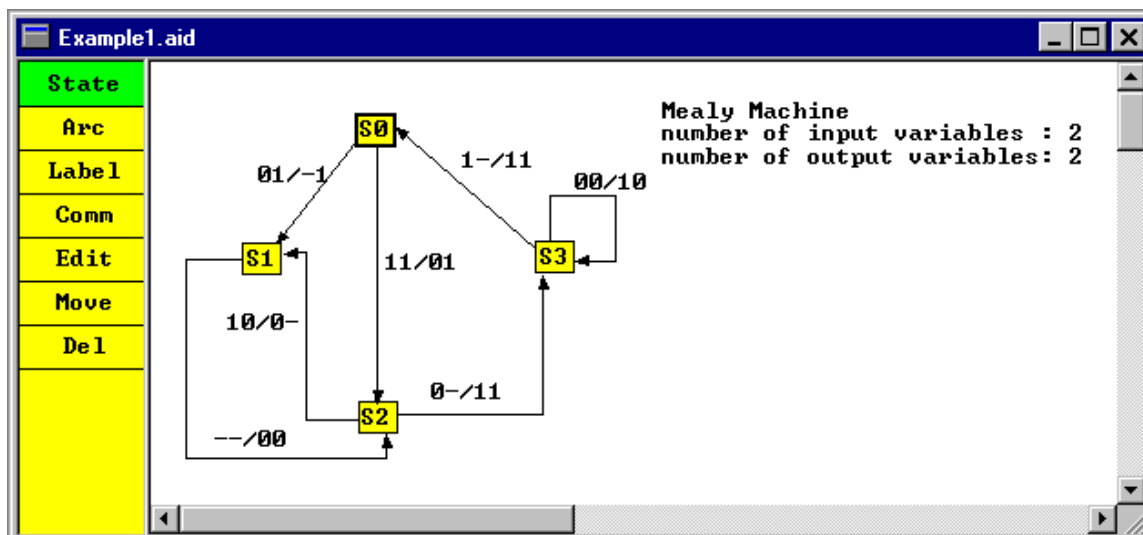
1. It is assumed that state table M2 is already stored as a LogicAid file. Enter the state table for M1 using the procedure given in Section 10, *Entering State Tables*.
2. Select **State Table Checker** from the **Routine** menu (or type **Ctrl+Shift+K**).
3. When the standard file dialog box appears, select the file in which the state table M2 is stored and open it. After the tables have been compared, the output window either will show the message “Correct State Table”, or will show the shortest input sequence that will distinguish the reset states of M1 and M2. If this input sequence is applied starting in the first row of M1 and the first row of M2, the last outputs from M1 and M2 will differ. (For the incompletely specified case, M1 may have a “don’t care” output when M2 has a 0 or 1 output, but not vice versa.)

You may use LogicAid to check your solutions to state table problems without revealing the correct answers. Encoded solution files for the state table problems in *Fundamentals of Logic Design*, 4th ed., are included on your LogicAid disk. You will find it helpful to check your answers using LogicAid before you look at the correct answer. This has two advantages over looking at the answers:

1. Your answer may be correct even if the number of states or the order of the states is different from the answer in the solution book. If you use LogicAid you will know for sure whether your answer is correct or not.
2. If your answer is not correct, LogicAid will give you the shortest sequence of inputs for which your solution fails, without revealing the correct solution. In this way you can try to find and correct your mistake before looking at the correct answer.

## 15. Entering State Graphs

You can use LogicAid to input a state graph and convert it to a state table. After you get a state table, you can reduce the number of states in the state table, make a state assignment, and derive flip-flop input equations to realize the table. The following examples show the input window for Mealy and Moore state graphs with 2 input variables and 2 output variables.



In both examples, notice that each state is represented by a rectangle that contains a state name. The connecting lines between states are called “arcs”. Each arc is terminated by an arrowhead, and each arc must have a label. In example 1, the arc which starts at S0 and ends at S1 has the label “01/-1”.

The steps required to enter a state graph are as follows :

1. Select **State Graph** from the **Input** menu (or type **Ctrl+G**).
2. When the Input State Graph Format dialog box appears, enter the number of input variables and number of output variables if they are different from the default values. The default input variable names are X1, X2, X3, ..., and the default output variable names are Z1, Z2, Z3, .... If you wish to change any of these, select **Enter Names**. If the outputs depend only on the present state and not on the input values, select **Moore**. The default state names are S0, S1, S2, .... If you want to use them, select **Use Default State Names**.
3. If you have selected **Enter Names**, the Input Variable Names dialog box will appear. Change any name that you want to be different from the default name and click **OK**.
4. At this point, a graph window will appear. The graph window has a side menu with 7 items at the left.
5. Always enter the reset (or starting) state first. To enter a new state, select **State** from the side menu and click the input window at the point where you want the state box. If you have selected to use the default state names, they will appear in the box; otherwise, you must enter a state name. For the Mealy type, you cannot enter the output values in the state box, but for the Moore type, you must enter the output values. Use "/" to separate the state name and output values. When finished, press the **Enter** key. If there is any syntax error in the state box, an alert box will appear and give you an error message. Once you have finished entering a state box, you must use **Del** if you want to delete it (see step 11).
6. To connect two state boxes, S1 and S2, select **Arc** from the side menu. Suppose S1 is the present state and S2 is the next state. Move the mouse over the *edge* of the S1 box until it is highlighted, click once, then move the mouse to the *edge* of the S2 box until it is highlighted, then click on the box once. An arc will connect S1 and S2, and an arrowhead, will appear at the edge of the S2 box.  
 Instead of a straight line, an arc may consist of a broken line with two or more segments. To enter this type of arc, click at the starting point, then click at each corner point, and finally click at the end point. (If the end point of an arc is not close enough to a state, that is if the box is not highlighted, clicking on it will not terminate the arc. To terminate an arc at an area outside of a state box, you must double click on the area. To make the box and arc meet, either use the **Move** command to move the arrowhead to the edge of a state box, or delete the arc and try again.)
7. After you enter an arc between two states, you can label it with input values. Select **Label** from the side menu and click near the arc to which you want to attach the label. Place the rectangular box close to the arc you want to label. If you select the wrong arc, press **Esc** and place the box near the arc you would like to label. After you select the correct arc, a cursor will appear at the location you click. For a Moore graph, enter the input values and press the **Enter** key. For a Mealy graph, enter the input values and output values separated by a "/", and then press the **Enter** key. If there is a syntax error, an alert box will appear and give you an error message.

8. If you wish to enter a comment, you can select **Comm** from the side menu and click the input window. When the cursor appears, type anything you want and press **F12**.
9. If you want to edit a state box, a label box or a comment, select **Edit** from the side menu and click at the point you want to edit. After you finish, press **Enter**.
10. If you want to move a state box, a label box, a comment or an arc, select **Move** from the side menu. Select the object you want to move with the mouse and hold down the mouse button as you drag the object to its new location. If you want to move an arc, select the arrowhead at the end of the arc and drag it to its new location. You can also select a corner point and drag it to a new location. After moving an arc, you will probably have to move its label to an appropriate location beside the arc, because the arcs stay attached to the state boxes, but the labels do not follow the moving arcs.
11. If you want to delete a state box, a label box, a comment or an arc, select **Del** from the side menu and click the object which you want to delete. LogicAid will highlight the object in blue and ask you if want to delete it. Then click **OK** to delete the object.
12. To change the reset (or starting) state, select **Change Reset State** from the **Input** menu, and then click on the new reset state.

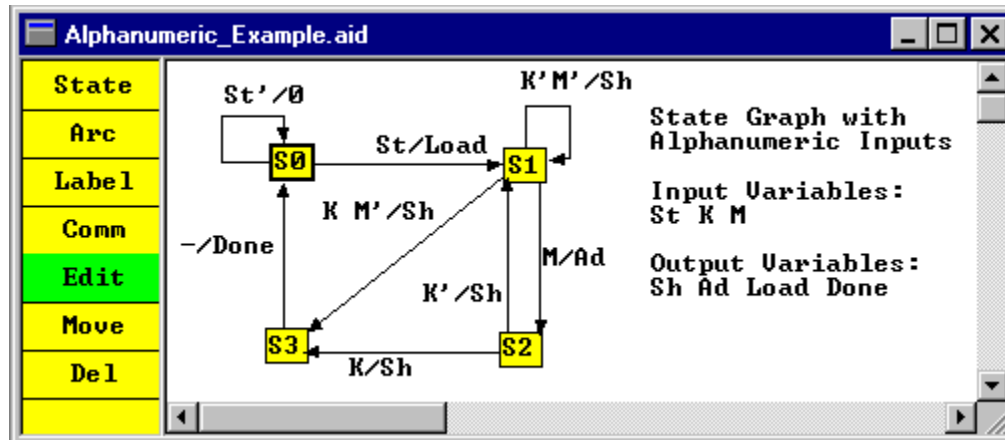
### **Converting to State Table**

13. After you finish the state graph, you can select **Convert to State Table** from the **Input** menu to convert the state graph to a state table. If there is a syntax error, an alert box will appear. Otherwise, a dialog box will allow you to choose the type of column headings for the state table. For a small number of input variables, choose **Use '0' and '1'** and completely specified column headings will be generated. Else, select **Use '0', '1', and '-'** and a column heading will be generated for each unique input combination on one of the arc labels.

### **Using Alphanumeric Inputs**

14. To use alphanumeric inputs for your state graph, choose **State Graph** from the **Input** menu as usual. Then choose **Alphanumeric Format** as your **Input Format** when the Input State Graph Format dialog box appears. The default input variable names are X1, X2, X3, .... and the default output variable names are Z1, Z2, Z3, .... If you wish to change any of these, select **Enter Names**. Then choose your options from the dialog box as you would for a regular state graph.
15. Draw the state graph as usual, except when you label the arcs, use the variable names you selected instead of **'1'** and **'0'**. The following is an example of a state graph with alphanumeric inputs.





### Multiple Labels on an Arc

16. You can place two or more labels on an arc by separating the labels by commas.  
 Binary example: 00/10, 10/11  
 Alphanumeric Example: X1'X2'/Z1, X1 X2'/Z1 Z2

## 16. Checking Partial State Graphs

If you are learning to construct state graphs from a problem statement, you will find it helpful to check your state graph several times during the process of constructing the graph. In this way you can detect and correct errors before your graph gets too large. LogicAid allows you to compare a partially completed state graph with a state table solution saved on the disk to see if they are functionally consistent. The method used to compare them is similar to the method used for Checking State Tables (see Section 14, *Checking State Tables*). However, when a partial state graph is checked, LogicAid ignores the missing data and compares the partial input data against the solution table. Isolated states that are not reachable from the starting state will not be checked.

Use the following procedure to check a partial state graph :

1. Enter part of a state graph using the procedure given in Section 15. You must enter the starting (reset) state first.
2. Select **Partial Graph Checker** from the **Routine** menu (or type **ALT+R G**).
3. When the standard file dialog box appears, select the file in which the state table solution file is stored and open it. After comparison, the output window either will show the message **Partial state graph is consistent with the correct solution**, or will show the shortest input sequence that will distinguish the reset states of the partial graph and solution table. If this input sequence is applied starting in the first state box of the partial graph and the first row of the solution table, the last outputs will be different.
4. If you want to check the partial graph again, you don't need to select the solution file any more. LogicAid will use the old solution file directly and won't show you the standard file dialog box.

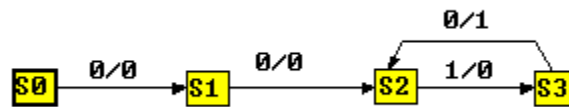
5. If you want to change the solution file, you have to choose **Select New Solution File** from the **Routine** menu and a standard file dialog box will appear. You can then select a new solution file.

The following example illustrates use of the partial graph checker. We will construct a state graph for the following problem:<sup>1</sup>

A clocked Mealy sequential network with one input (X) and one output (Z) is to be designed. The output is to be 0, unless the input is 0 following a sequence of *exactly* two 0 inputs followed by a 1 input.

The correct state table has already been stored on your disk in a file named 14-1 in the PROBS-14 folder. The table is stored in encoded form so you can use it to check your solution even though you cannot read the file directly.

Initially, you will input a graph that gives a 1 output in response to the sequence 0010:



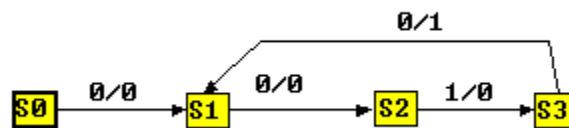
After choosing **Partial Graph Checker** on the **Routine** menu and selecting solution file 14-1, you get the following message in the output window:

\*\*\*\*\* Incorrect State Graph \*\*\*\*\*

The shortest input sequence for failure is:

X = 0 0 1 0 1 0

If you trace this on the above state graph, you get the output sequence  $Z = 0\ 0\ 0\ 1\ 0\ 1$ , which is wrong in the last digit. After correcting the partial state graph, you get

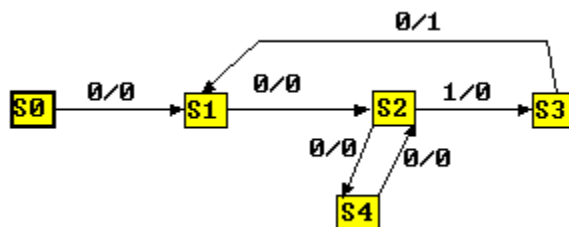


Running the Partial Graph Checker again gives the output message:

\*\*Partial state graph is consistent with correct solution.\*\*

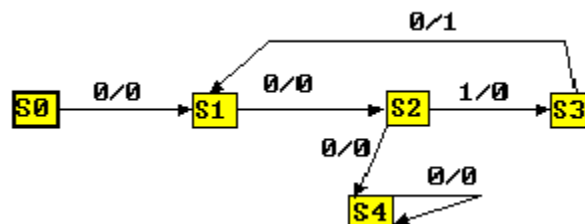
After attempting to add a state that corresponds to **3 or more 0's received**, suppose that you construct the following graph:

<sup>1</sup> Roth, Charles H., *Fundamentals of Logic Design*, 5th ed., Brooks/Cole, 2003, pp. 412-413.



Running the Partial Graph Checker indicates that the shortest input sequence for failure is:

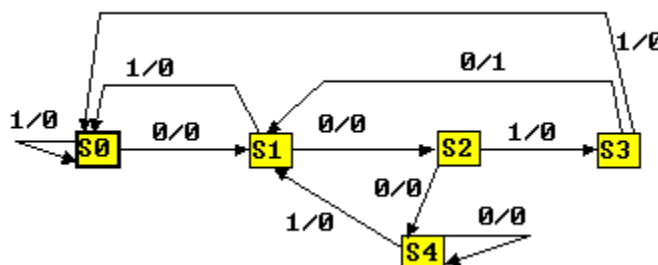
$X = 0\ 0\ 0\ 0\ 1\ 0$ . Tracing the sequence on the above state graph gives  $Z = 0\ 0\ 0\ 0\ 0\ 1$ , which is wrong in the last digit. After correcting the partial graph, you get



The Partial Graph Checker again gives the message:

**\*\*Partial state graph is consistent with correct solution.\*\***

After several more steps of trial and error, you arrive at the following graph:



When the Partial Graph Checker tells us that this graph is consistent with the correct solution, you can choose **Convert Graph to Table** on the **Input** menu and then verify that the table is correct using the **State Table Checker** on the **Routine** menu.

## 17. Programming a PAL

In order to program a PAL, you will first need to generate the required fuse patterns from your logic equations. You can use LogicAid to generate a file that contains the fuse patterns in Jedec format. Then you can load this Jedec file into a PAL programmer to actually program the PAL. The present version of this software only allows you to program a 22V10 PAL.

To generate the Jedec file using LogicAid, proceed as follows:

1. Select **Open** from the **File** menu, or **State Table** from the **Input** menu, and input the desired state table.
2. Select **Flip-flop Equations** from the **Routine** menu. Select **D Flip-Flop** from the dialog box and click **OK**.
3. Use the defaults in the Simplification dialog box, so just click **OK**. The Output window will contain the flip-flop and output equations.

NOTE: If you change the simplification defaults, the equations will be in the wrong format and you will not be able to make a valid Jedec file.

4. Select **Make Jedec File (CTRL+J)** from the **Routine** menu. When the dialog appears, select **MANUAL** pin assignment.
5. When the pin assignment dialog appears, note that CLK is assigned to pin 1, GND to pin 12, Vcc to pin 24, and ARLow (asynchronous reset, active Low) to pin 13. You cannot change these assignments.

Enter the appropriate pin assignments by replacing NC (no connection) with the appropriate values:

X (or X1 if you used X1 for the input variable)

Z (or Z1 if you used Z1 for the output variable)

Q1 (click on the square next to register to indicate a flip flop)

And so on, for other inputs, outputs, and flip-flops

All unused pins must be assigned NC.

6. When finished, click **OK**.
7. Wait for the Jedec file (fuse patterns) to appear. It will appear in a new OUTPUT window.
8. To save the Jedec file on disk, select **Save As** on the **File** menu.

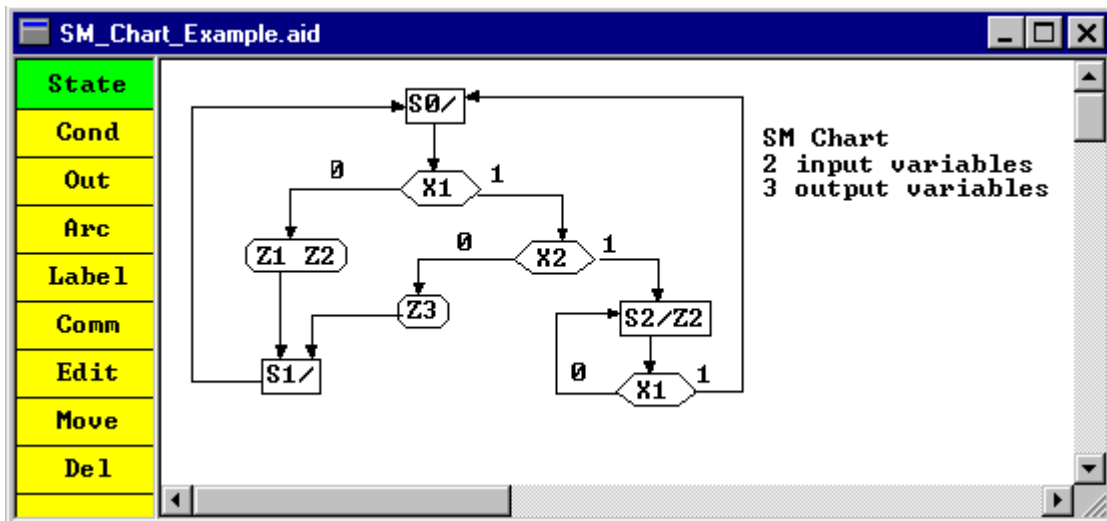
Use a PAL Programmer with a serial port to program your PAL. To download a Jedec file to the programmer, proceed as follows:

1. Make sure that all the IC sockets on the PAL Programmer are empty. Then turn on the programmer.
2. Select the CE 22V10 programmable logic device from the programmer.
3. Set the programmer to receive data.
4. Run LogicAid. To retrieve your saved Jedec file, select **Open Text** on the **File** menu. Then select Jedec and hit **OK**. Select the file in the file dialog box and click Open.
5. Select the **Download Jedec File (CTRL+D)** option from the **Routine** menu. This starts off the communication software between the computer and the programmer. When the dialog appears select the appropriate communication port, baud rate, parity and data bits. Click **OK**.
6. The programmer should receive the data. When the transmission is complete, the Jedec file has been downloaded to the programmer.
7. Then insert the PAL in the programmer socket and complete the programming.

## 18. Entering SM Charts

You can use LogicAid to input an SM chart and convert it to a PLA table. After you get a PLA table, you may choose **Simplification** on the **Routine** menu (see Section 4, *Simplifying Logic Equations*), which will give you the reduced equations for the PLA table.

The following example shows the input window for a SM Chart with 2 input variables and 3 output variables.



Each state is represented by a rectangle that contains a state name. Each hexagon indicates a condition/input and each rounded rectangle indicates an output. The connecting lines between boxes are called “arcs”. Each arc is terminated by an arrowhead, and each arc leaving a condition must have a label of either a “0” or “1”. In the above example, the arc that starts at input X1 and ends at S0 has the label “0”.

The steps required to enter a SM Chart are as follows:

1. Select **SM Chart** from the **Input** menu (or type **Ctrl+M**).
2. When the Input SM Chart Format dialog box appears, enter the number of input variables and number of output variables if they are different from the default values. The default input variable names are X1, X2, X3, ... and the default output variable names are Z1, Z2, Z3, .... To change any of these, select **Enter Names**. The default state names are S0, S1, S2, .... If you want to use them, select **Use Default State Names**.
3. If you have selected **Enter Names**, the Input Variable Names dialog box will appear. Change any name that you want to be different from the default name and click **OK**.
4. At this point, a graph window will appear. The graph window has a side menu with 9 items at the left.

5. To enter a new state, select **State** from the side menu and click the input window at the point where you want the state box. If you have selected to use the default state names, they will appear in the box; otherwise, you must enter a state name followed by “/.” If no output value is associated with the state, leave the part after the “/” empty. When finished, press the **Enter** key. If there is any syntax error in the state box, an alert box will appear and give you an error message. Once you have finished entering a state box, you must use **Del** if you want to delete it (see step 11). Similar to entering conditions and outputs, select **Cond** or **Out** from the side menu and click the input window at the point where you want the condition or output box.
6. To connect two boxes, select **Arc** from the side menu. Click on the *edge* of the first box when it is highlighted, then click on the *edge* of the second box when it is highlighted. To enter an arc that has two or more line segments, click on the starting point, click where you want to make corners, and then click on the ending point. If the end point of an arc is not close enough to a box, that is if the box is not highlighted, clicking on it will not terminate the arc. To terminate an arc at a point outside of a state box, you must double click at the point. To make the box and arc meet, either use the Move command to move the arrowhead to the edge of a state box, or delete the arc and try again.
7. After you enter an arc between a condition box and a state or output box, you can label it with input values. Select **Label** from the side menu and click near the arc to which you want to attach the label. Place the rectangular box close to the arc you want to label. After you select an arc, the arc will be labeled with either a “0” or “1”. The first arc you click on leaving a condition will be automatically labeled with a “0”. When you click on the other arc leaving that same condition box, it will be labeled with a “1”.
8. To enter a comment, select **Comm** from the side menu and click the input window. When the cursor appears, type anything you want and press **F12**.
9. To edit a state box, a condition box, an output box, or a comment, select **Edit** from the side menu and click at the point you want to edit. After you finish, press **Enter**.  
To edit the labels, chose **Edit** from the side menu and click on the label you wish to change. The two arcs leaving from that condition will automatically swap labels. This means the arc originally labeled “0” will now be labeled “1” and vice versa.
10. To move a state box, a condition box, an output box, a label box, a comment or an arc, select **Move** from the side menu. Select the object that you want to move with the mouse and hold down the mouse button as you drag the object to its new location. To move an arc, select the arrowhead at the end of the arc and drag it to its new location. You can also select a corner point and drag it to a new location. After moving an arc, you will probably have to move its label to an appropriate location beside the arc, because the arcs stay attached to the state boxes, but the labels do not follow the moving arcs.
11. To delete a state box, a condition box, an output box, a label box, a comment or an arc, select **Del** from the side menu and click the object that you want to delete.

LogicAid will highlight the object in dark blue and ask you if want to delete it. Then click **OK** to delete the object.

12. After you finish inputting the SM Chart, you can select **Convert to PLA Table** from the **Input** menu to convert the SM chart to a PLA table. If there is a syntax error, an alert box will appear. Otherwise, a new window opens with the generated PLA table.

## 19. Design Examples

The examples that follow illustrate how LogicAid can be used in the design of sequential networks or state machines. Each example shows how a sequence of LogicAid operations can be applied to accomplish the desired result.

### Design of a Sequence Detector

This example illustrates derivation of minimized flip-flop input equations for a sequence detector, which has one input (X) and one output (Z). The detector examines groups of four consecutive X inputs and produces an output  $Z = 1$  if the input sequence 0101 or 1001 occurs. The network resets after every four inputs. The Mealy state table for the network is derived in Section 15.1 of *Fundamentals of Logic Design*. After you type in this state table and choose **Check Syntax** from the **Input** menu (or type **Alt+I Y**), the display in the Edit window is as follows:

PS	NS		OUTPUTs*		INPUT-VARS
	0	1	0	1	X
A	B	C	0	0	
B	D	E	0	0	
C	F	G	0	0	
D	H	I	0	0	
E	J	K	0	0	
F	L	M	0	0	
G	N	P	0	0	
H	A	A	0	0	
I	A	A	0	0	
J	A	A	0	1	
K	A	A	0	0	
L	A	A	0	1	
M	A	A	0	0	
N	A	A	0	0	
P	A	A	0	0	

\* Z

Next, select **State Reduction** on the **Routine** menu (or type **Alt+R R**) and choose the Display Reduced Table in **Edit Window** option. This reduces the table to a minimum number of rows and replaces the original table with the reduced table:

PS	NS		OUTPUTs*		INPUT-VARS
	0	1	0	1	X
A	B	C	0	0	
B	D	E	0	0	
C	E	D	0	0	
D	H	H	0	0	
E	J	H	0	0	
H	A	A	0	0	
J	A	A	0	1	

\* Z

Next you must assign flip-flop states to correspond to each of the states in the table. The state assignment guidelines given in *Fundamentals of Logic Design* indicate that state pairs BC, DE and JH should be given adjacent assignments. Two of the many assignments that satisfy these adjacencies are:

Q2 Q3		Q1	
		0	1
00	A		
01	B	H	
11	C	J	
10	D	E	

Q2 Q3		Q1	
		0	1
00	A		
01	B	C	
11	D	E	
10	H	J	

For the first of these state assignments, A = 000, B = 001, C = 011, D = 010, E = 110, H = 101 and J = 111. To input this assignment, select the **State Assignment** on the **Routine** menu and then choose the **Display or Change Current Assignment** and **Binary** options. When the original straight binary assignment is displayed, edit this assignment to correspond to the first assignment map. The resulting table is

ASSIGN	PS	NS		OUTPUTs*		INPUT-VARS
		0	1	0	1	X
000	A	B	C	0	0	
001	B	D	E	0	0	
011	C	E	D	0	0	
010	D	H	H	0	0	
110	E	J	H	0	0	
101	H	A	A	0	0	
111	J	A	A	0	1	

\* Z

Then you select the **Flip-Flop Equations** on the **Routine** menu and choose the J-K Flip-Flop option. After you choose **Petricks** in the **Simplification Options** dialog box, the following equations are displayed in the output window:



$$\begin{aligned}
 J(Q1) &= X Q2' Q3 + X' Q2 + Q2 Q3' \\
 K(Q1) &= Q3 \\
 J(Q2) &= Q1' Q3 + X Q1' \\
 J(Q2) &= Q1' Q3 + X Q3' \\
 K(Q2) &= Q1' Q3' + Q1 Q3 + X Q3' \\
 K(Q2) &= Q1' Q3' + Q1 Q3 + X Q1 \\
 J(Q3) &= 1 \\
 K(Q3) &= 1 \\
 Z &= X Q1 Q2 Q3
 \end{aligned}$$

If you use common gates where possible, the “cost” of realizing these equations is 11 gates with a total of 27 gate inputs. You can try other state assignments to see if you can find a better solution. Using the second assignment map given on the previous page, again select the **State Assignment** on the **Routine** menu, edit the state assignment, and then select the **Flip-Flop Equation** on the **Routine** menu. The resulting equations are:

$$\begin{aligned}
 J(Q1) &= X Q2' \\
 K(Q1) &= X + Q3' \\
 J(Q2) &= Q3 \\
 K(Q2) &= Q3' \\
 J(Q3) &= Q2' \\
 K(Q3) &= Q2 \\
 Z &= X Q1 Q3'
 \end{aligned}$$

These equations require only 3 gates and a total of 7 gate inputs, which is a substantial improvement over the previous solution.

### Dice Game Controller<sup>2</sup>

The following example shows derivation of the logic equations for a dice game controller from an SM chart. The first step is to enter the chart as shown in Figure 19.1. Since the present version of LogicAid does not derive equations directly from the SM chart, the chart must first be converted to a PLA table using the **Convert to PLA Table** command on the **Input** menu. The conversion routine creates a PLA table assuming that D flip-flops are used to implement the SM chart. Unless otherwise specified, a straight binary state assignment is used. Each link path in the SM chart leads to one row in the PLA table as shown in Figure 19.1. The outputs associated with unused state combinations are treated as don't cares, but these don't cares are not shown explicitly in the table.

After choosing **Simplification** on the **Routine** menu, and either the PI Chart or Petrick option, you obtain the following reduced equations for the Dice Game Controller:

---

<sup>2</sup> Roth, Charles H., *Fundamentals of Logic Design*, 5th ed., Brooks/Cole, 2003, Sections 19.2 and 19.3.

$$D\langle Q1 \rangle = Q1 \text{ Rb} + Q1 \text{ Q3}' + Q1 \text{ D7}' \text{ Eq}' + Q1' \text{ Q2}' \text{ Q3} \text{ Rb}' \text{ D7}' \text{ D11}' \text{ D2312}'$$

$$D\langle Q2 \rangle = Q2' \text{ Q3} \text{ Rb}' \text{ D7} + Q1 \text{ Q3} \text{ Rb}' \text{ Eq} + Q1' \text{ Q2}' \text{ Q3} \text{ Rb}' \text{ D11} + Q2 \text{ Reset}' + Q1' \text{ Q2}' \text{ Q3} \text{ Rb}' \text{ D7}' \text{ D11}' \text{ D2312}$$

$$D\langle Q3 \rangle = Q2 \text{ Q3} \text{ Reset}' + Q2' \text{ Rb} + Q1 \text{ Q3} \text{ D7} \text{ Eq}' + Q1' \text{ Q2}' \text{ Q3} \text{ Rb}' \text{ D7}' \text{ D11}' \text{ D2312}$$

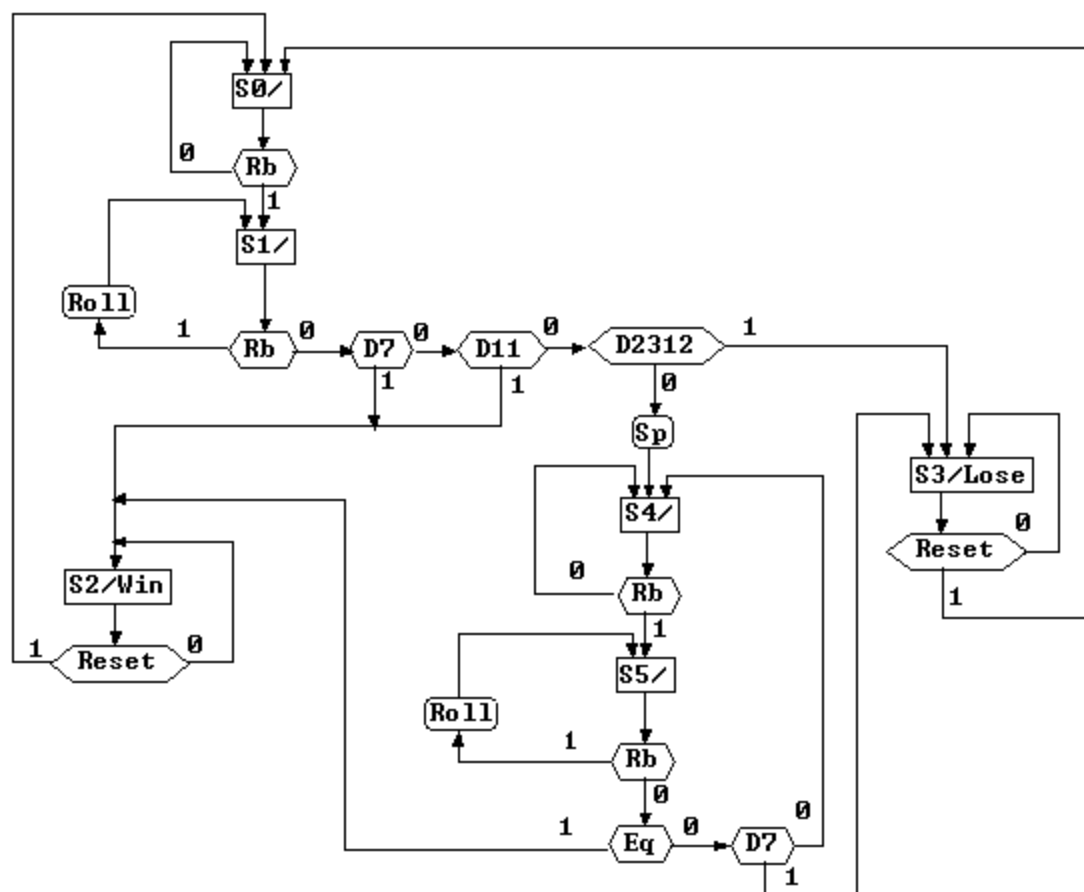
$$\text{Win} = Q2 \text{ Q3}'$$

$$\text{Lose} = Q2 \text{ Q3}$$

$$\text{Roll} = Q2' \text{ Q3} \text{ Rb}$$

$$\text{Sp} = Q1' \text{ Q2}' \text{ Q3} \text{ Rb}' \text{ D7}' \text{ D11}' \text{ D2312}'$$

These equations could easily be implemented using a programmable logic device such as a 22V10.





ASSIGN	PS	NS										
		1	0	0	0	0	0	0	0	0	-	-
		-	-	-	-	-	-	-	-	-	1	0
		-	-	0	0	1	0	-	0	1	-	-
		-	-	0	0	-	1	-	-	-	-	-
		-	-	1	0	-	-	-	-	-	-	-
		-	-	-	-	-	-	1	0	0	-	-
000	S0	S1	S0	-	-	-	-	-	-	-	-	-
001	S1	S1	-	S3	S4	S2	S2	-	-	-	-	-
100	S4	S5	S4	-	-	-	-	-	-	-	-	-
101	S5	S5	-	-	-	-	-	S2	S4	S3	-	-
010	S2	-	-	-	-	-	-	-	-	-	S0	S2
011	S3	-	-	-	-	-	-	-	-	-	S0	S3

\* Win Lose Roll Sp

OUTPUTs*											INPUT-VARS
1	0	0	0	0	0	0	0	0	-	-	Rb
-	-	-	-	-	-	-	-	-	1	0	Reset
-	-	0	0	1	0	-	0	1	-	-	D7
-	-	0	0	-	1	-	-	-	-	-	D11
-	-	1	0	-	-	-	-	-	-	-	D2
-	-	-	-	-	-	1	0	0	-	-	Eq
0000	0000	----	----	----	----	----	----	----	----	----	
0010	----	0000	0001	0000	0000	----	----	----	----	----	
0000	0000	----	----	----	----	----	----	----	----	----	
0010	----	----	----	----	----	0000	0000	0000	----	----	
----	----	----	----	----	----	----	----	----	1000	1000	
----	----	----	----	----	----	----	----	----	0100	0100	

Next, select **Simplification** and derive the flip-flop and output equations using PI Chart or Petrick. The resulting equations are the same as obtained by the PLA table method above.

## 20. Using LogicAid for Asynchronous Design Problems

Several of the features of LogicAid can be used for asynchronous design problems. The state table checker can be used to check solutions for asynchronous tables as well as synchronous tables. In general, asynchronous tables should be reduced before they are entered into LogicAid.

Asynchronous sequential networks without flip-flops are frequently modeled using delays in the feedback path. For this model, the outputs of the delays represent the present state of the network, and the inputs represent the next state. Since the next-state equations are the same as the input equations for D flip-flops, the next-state equations can be derived from the state table using the LogicAid D flip-flop equation routine. If the minimum next-state equations contain hazards, it may be necessary to add extra prime implicant terms to eliminate the hazards.

Use the following procedure to derive hazard-free next-state equations for an asynchronous design problem:

1. Derive the reduced flow table. Make a state assignment in the usual way and modify the table as required to eliminate critical races.
2. Enter the state table derived in step 1 into LogicAid.
3. Enter the state assignment using the **State Assignment** routine.
4. Select the **Flip-Flop Equations** routine and choose D—Delay Flip-Flop. Select the **Display All Prime Implicants** option. The D flip-flop equations are the next-state equations.
5. Add prime implicant terms to the minimized next-state equations as required, to eliminate hazards. Adding all extra prime implicants will eliminate all hazards, but in general all of the prime implicants will not be necessary. For 4- or 5-variable problems, you may use the KMap routines to plot the next-state maps from the minimized equations and determine which prime implicants must be added to the minimum solution. Refer to *Fundamentals of Logic Design* for information about how to see hazards from a Karnaugh map.

When S-R flip-flops are used in the asynchronous design, follow the preceding steps, but select S-R flip-flop in step 4.

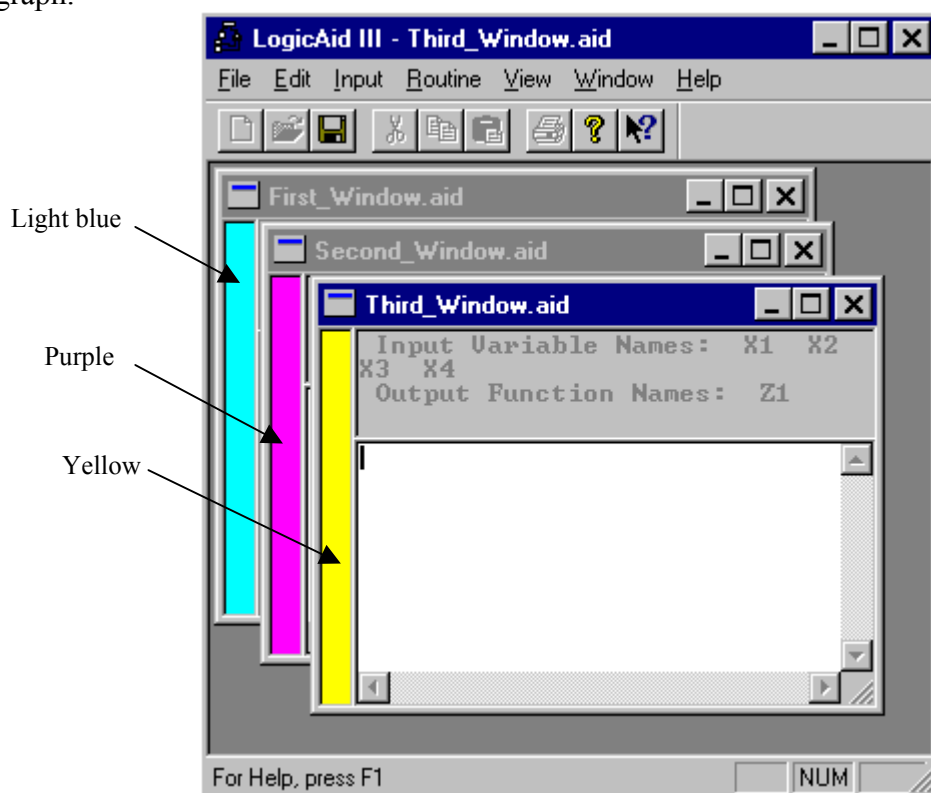


# LogicAid Reference Manual

## 1. LogicAid Menus & Windows

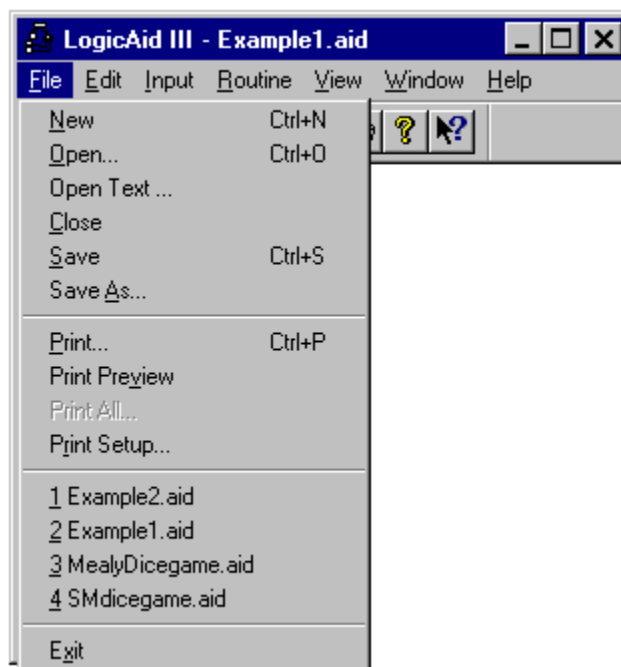
The menus give you with access to the routines that are available in LogicAid. Menu options will change when different input types are selected. For example, when an input window is open, the Edit and Window menu will appear.

The current version of LogicAid allows up to three input windows to be opened simultaneously. This means that you can work on three different logic problems at a time. The input windows are color-coded, with the first window having a light-blue sidebar, second and third windows having a purple and yellow sidebar respectively, as shown in Figure 1.1 below. Windows derived from an input window will have sidebars with a matching color. For example, if a state graph is input, the state table and flip-flop input equations derived from that state graph will have the same color sidebar as the state graph.



**Figure 1.1. Opening 3 input windows simultaneously.**

## 2. The File Menu



The File menu is very similar to the File menus of other Windows applications. Its commands allow you to create, open, close, save, and print documents.

### 2.1. New (Alt+F N or Ctrl+N)

Function: Creates a window for editing a new input document.

This command brings up the New dialog box, shown in Figure 2.1, for selecting one of eight input types: *terms*, *equations*, *truth table*, *PLA table*, *state table*, *state graph*, *SM chart* and *Karnaugh map*. A selection from this dialog box is equivalent to selecting any of the **Terms**, **Equations**, **Truth Table**, **PLA Table**, **State Table**, **State Graph**, **SM Chart** and **Karnaugh Map** commands from the **Input** menu. After you click on **OK** in this dialog box, the program calls the selected routine just as though the equivalent command was selected from the **Input** menu.

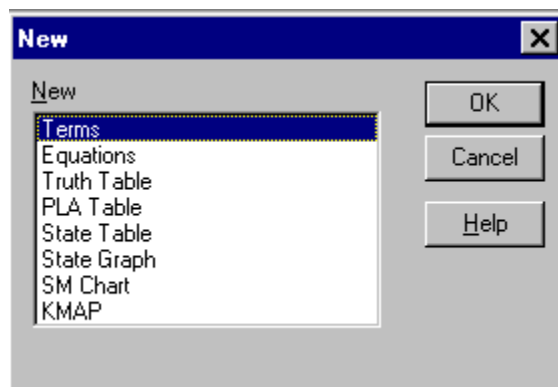


Figure 2.1. New dialog box



Another dialog box is displayed in order to specify the new parameters needed to create a new input document. After **OK** has been clicked in this second dialog box, the program creates a new edit or graph window with the input document name at the top. At this point, you are allowed to enter data into the new window. For more specific information on creating a new input document, refer to Sections 4.1, *Terms*, through 4.7, *SM Chart*.

## 2.2. Open (Alt+F O or Ctrl+O)

Function: Opens a LogicAid file from a disk and displays the contents in an edit window.

This command brings up the Open dialog box, which allows you to select an existing LogicAid file from any accessible drive and open it for editing or other processing.

If a file contains a protected state table, a password dialog box (Figure 2.2) appears, and you must enter the password to open the file. Note that the password is case-sensitive. If you enter an invalid password, an alert box will appear and you can either select cancel or reenter a password. If you want to open a text file created by a text editor, you must select the **Open Text** command to convert it to a LogicAid file (see Section 2.3).

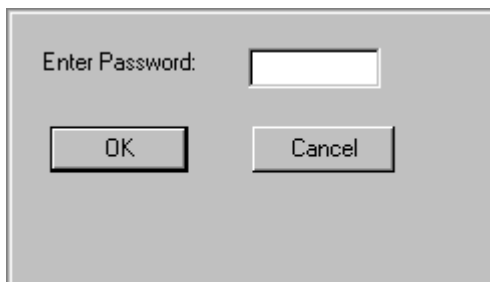


Figure 2.2. Password dialog box

## 2.3. Open Text (Alt+F T)

Function: Converts a text file into a LogicAid file.

This command brings up a dialog box, similar to the one shown in Figure 2.1, for specifying the input type in order to convert the text file into a LogicAid file. After selection of the input type, the **Open** dialog box is displayed. After file selection, the **Input Parameters** box is displayed. The file will then be displayed in LogicAid format and can be saved by selecting **Save** or **Save As**.

## 2.4. Close (Alt+F C)

Function: Closes the edit, graph, or output window.

This command is equivalent to clicking on the Close box located at the upper left-hand corner of a window. If the window has not been modified since the last save, the program will close the window directly.

If you try to close a window in which a document has been modified since the last save, an alert box will give you an opportunity to save the changes (select **Yes**), discard the changes (select **No**), or **Cancel** the **Close** command. For the edit window, the word “modified” means any change made to the text file or any change made to the input-related parameters through the **Change Parameters** command. For the output window, the word “modified” means any new output written since the last save.

## 2.5. Save (Alt+F S or Ctrl+S)

Function: Saves the contents of the active edit, graph or output window.

The active edit or graph window will be saved as a LogicAid file with a **.aid** extension, while an output window will be saved as a text file with a **.txt** extension. For an edit or graph window, all the input-related parameters and selections are saved at the beginning of the file, followed by the text in the edit window or graph window.

## 2.6. Save As (Alt+F A)

Function: Saves the current edit, graph or output window under a new name.

To save a state table as a protected file, hold down the **Ctrl** key and select **Save As**. Then enter a file password when the Password dialog box appears. The state table will then be saved as an encoded file that can only be read if the password is entered.

## 2.7. Print (Alt+F P or Ctrl+P)

Function: Prints the content of current edit, graph or output window.

This command allows you to print the current contents of the active window.

## 2.8. Print Preview (Alt+F V)

Function: Provides a preview of the active document as it would appear when printed.

This command displays a preview of the current document to be printed. When you choose this command, the main window will be replaced with a print preview window in which one or two pages will be displayed in their printed format. The print preview toolbar offers you options to view either one or two pages at a time; move back and forth through the document; zoom in and out of pages; and initiate a print job.

## 2.9. Print All (Alt+F L)

Function: Prints the current edit, graph and output windows' contents.

This command is activated on the menu only if both an edit window and an associated output window are on the screen. The program will print in the following order:

1. Edit window's title
2. Edit window's contents
3. Output window's title

#### 4. Output window's contents

If there is an associated open graph window, its title and contents will also be printed.

### 2.10. Print Setup (Alt+F R)

Function: Provides print options and printer settings.

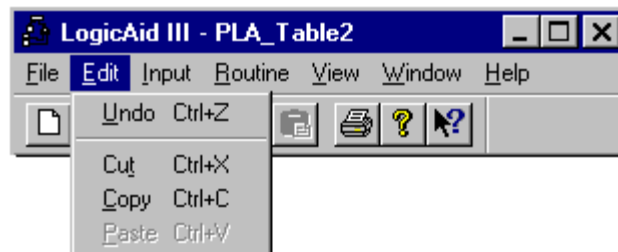
Use this command to select a printer and a printer connection. Clicking on this command will open a Print Setup dialog box, where you specify the printer and its connection, as well as the orientation for the printout and paper size being used.

### 2.11. Exit (Alt+F X)

Function: Quits the application program.

This command first checks whether the edit, graph, or output window has been modified since the last **Save** or **Open** command. If any of the files has been modified, the program will give you an opportunity to save the changes (select **Yes**), discard the changes (select **No**), or **Cancel** the **Exit** command.

## 3. The Edit Menu



The Edit menu contains the standard Windows editing functions. It is only available when an edit window is open.

### 3.1. Undo (Ctrl+Z)

Function: Reverses the effects of the last text editing operation.

This item is activated in a text-editing window. It undoes the last text editing operation at the point of its execution. It will undo typing that has occurred following the last edit operation or immediately following the creation of an editing window.

### 3.2. Cut (Ctrl+X)

Function: Removes the selected text from the edit window and replaces the Clipboard's current contents with the selected text.

This item is not activated unless text has been selected. Material that has been cut can be pasted back into the window, using the **Paste** command, if no other change has been made to the Clipboard.

### 3.3. Copy (Ctrl+C)

Function: Replaces the Clipboard's current contents with a copy of the selected text.

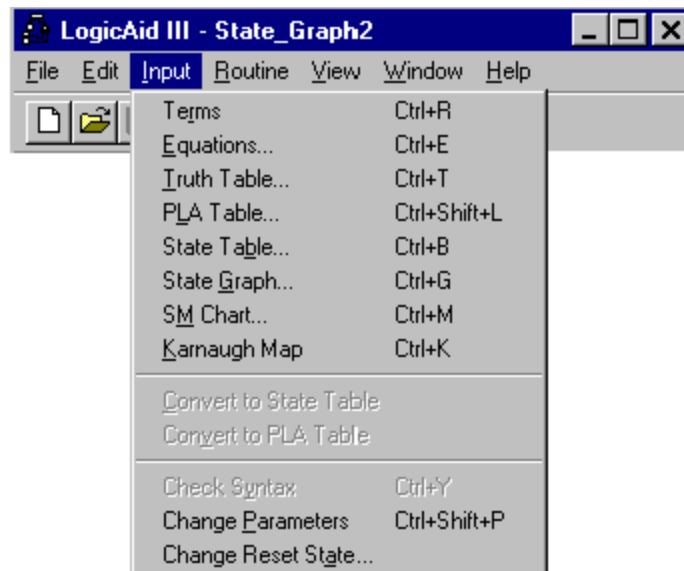
This item is not activated unless text has been selected. Material that has been copied can be pasted elsewhere using the **Paste** command.

### 3.4. Paste (Ctrl+V)

Function: Replaces the current selection with the contents of the Clipboard (if any).

This item is activated whenever the edit window is activated.

## 4. The Input Menu



The Input menu is used for creating new input documents, changing input-related parameters, and checking the current input data syntax. When three edit windows are open, the **Terms**, **Equations**, **Truth Table**, **State Table**, **State Graph**, **SM Chart** and **Karnaugh Map** commands are deactivated. The **Convert to State Table** command is shown only if a state graph window is open. The **Convert to PLA Table** is shown only if a state table window is open. **Change Parameters** is shown only if an input window is open.

#### 4.1. Terms (Alt+I R or Ctrl+R)

Function: Enters Boolean function(s) as minterm or maxterm expansions.

This command brings up a dialog box for specifying the input parameters. A blank edit window will be created for input terms after you click on **OK** this dialog box.

This Input Terms Format dialog box is shown in Figure 4.1, and its contents are described in the following subsections. You may later change the parameters of this dialog box by selecting the **Change Parameters** command, which will display the dialog box for modification.

#### TEXT BOXES:

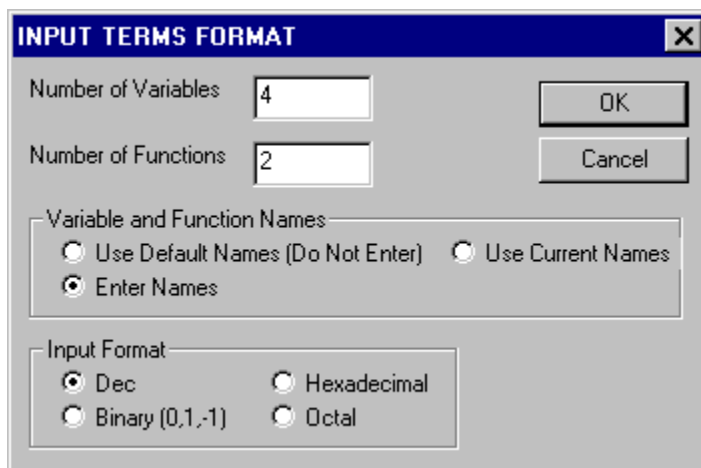
There are two text boxes in this dialog box. The **Tab** key may be used to move from one text box to the other. The input numbers must consist of valid decimal digits.

The *Number of Variables* text box specifies the number of variables in the Boolean functions. Its limitations are listed below:

Default:	4
Minimum:	1
Maximum:	32

The *Number of Functions* text box specifies the number of Boolean functions. Its limitations are listed:

Default:	1
Minimum:	1
Maximum:	32



**Figure 4.1. Input Terms Format dialog box.**

#### VARIABLE AND FUNCTION NAMES:

As shown by Figure 4.1, function and variable names may be specified with one of three options. These are *Use Default Names*, *Enter Names*, and *Use Current Names*.

If you select the **Use Default Names** option, LogicAid initializes the necessary variable and function names. The default variable names are X1, X2, X3, .... Similarly the default function names are Z1, Z2, Z3, ....

If you select the **Enter Names** option and click on **OK** in the dialog box, the program will bring up a second dialog box for entering names, as shown in Figure 4.2. Before this Input Variable Names dialog box is discussed, the **Use Current Names** option will be described.

The **Use Current Names** option may be selected if the same non-default names are to be used, but you do not wish to have the Input Variable Names dialog box displayed.

As shown in Figure 4.2, the Input Variable Names dialog box has a text box available for each variable and function name. In this particular case, there are six text boxes for the four variable names and the two function names.

INPUT VARIABLE NAMES:

apple ball cat dog

OUTPUT FUNCTION NAMES:

f1 f2

Select Range

☒ 1-16 ☐ 17-32

OK Cancel

**Figure 4.2. Input Variable Names dialog box**

When this dialog box is shown for the first time, the text boxes will display the default names. You may therefore edit any or all of the available text boxes. If the Input Variable Names dialog box is displayed again through the **Change Parameters** command, the current names, rather than the default names, are displayed in the text boxes.

The rules for entering names follow:

- No duplicate variable names are allowed.
- No duplicate function names are allowed.
- A variable name may also be a function name.

- Name length must be 1 through 10 characters.
- Input is case sensitive.
- Each character may be any alphanumeric or an underscore character  
(A, B, ..., Z, a, b, ..., z, 0, 1, ..., 9, \_)

After you click on **OK** in the **Input Variable Names** dialog box, the changes to the names do not take effect until all of the names are syntactically correct. If there are duplicate names when the **Input Variable Names** dialog box is displayed, and you click on **Cancel**, it may result in the default names for each name group being used. For example, if the variable names are X3 and X2 and you change the number of variables from two to four, the current names are X3, X2, X3, and X4 when the dialog box is displayed. Note that the first and third names are both X3. If you click on **Cancel**, LogicAid alerts you that it will use the default variable names. You get an opportunity to re-enter the variable names and avoid the defaults.

#### INPUT FORMAT:

The product terms may be specified with one of the following input formats: *Decimal*, *Binary*, *Hexadecimal*, and *Octal*. The legal characters for the input formats are shown in the following listing:

<u>Input Format</u>	<u>Legal Characters</u>
Decimal	0, 1, ..., 9
Binary	0, 1, –
Hexadecimal	0, 1, ..., 9, A, ..., F, a, ..., f
Octal	0, 1, ..., 7

If the input file contains more than one function, all functions must use the same input format. Separate input files must be used if different input formats are desired.

#### SYNTAX

The syntax checker for the **Terms** input routine is rather flexible, but you must observe some rules. The following key symbols and hints are necessary to understand the syntax rules to be discussed later.

<u>Key</u>	<u>Definition</u>
1-terms	product terms for which the function is equal to 1
0-terms	product terms for which the function is equal to 0
X-terms	product terms for which the function is unspecified
termset	1-terms, 0-terms, or X-terms
comma	(,) separates two termsets
period	(.) terminates each function
comments	delimited by a pair of brackets “{” and “}”
separators	combination of one or more <b>Space</b> , <b>Tab</b> , and <b>Enter</b> keys

The 1-terms, 0-terms, and X-terms are defined as a collection of one or more product terms. Each function may consist of one or more of these termsets. Since not every combination of these termsets is allowed as input, the correct combinations of the

termsets in a function are listed below. In addition, since not all the terms of a function must be in the input file, the program will treat the remaining terms as the termset given in the parentheses.

<u>Function Format</u>	<u>Remaining terms are treated</u>
1-terms.	(0-terms)
1-terms,.	(0-terms)
1-terms,,.	(0-terms)
,0-terms.	(1-terms)
,0-terms,.	(1-terms)
1-terms, 0-terms.	(X-terms)
1-terms, 0-terms,.	(X-terms)
1-terms,, X-terms.	(0-terms)
,0-terms, X-terms.	(1-terms)
1-terms, 0-terms, X-terms.	(none)

Given these combinations, the following rules apply:

1. Each function must have at least one 1-term or 0-term.
2. Two functions in an input window do not need to have the same function format.
3. Each termset must be terminated by either a comma or a period, depending upon the input sequence.
4. Each function must be terminated with a period.
5. Each function may have leading and lagging comments and/or separators.
6. Two termsets must be separated by a comma. Each termset may have optional leading and lagging comments and/or separators.
7. Within each termset, two terms must be separated with comments and/or separators. Terms within each termset may be listed in any order.
8. When all three termsets are specified as shown in the last function format listed above, the program will determine whether or not all the terms are listed; therefore, you must make sure the input is complete.



**EXAMPLES:**

		apple ball			
cat dog		00	01	11	10
	00	1	0	0	0
	01	1	1	1	1
	11	0	0	1	0
	10	0	0	1	0
		f1			

		apple ball			
cat dog		00	01	11	10
	00	1	0	0	0
	01	1	1	X	0
	11	0	1	1	1
	10	X	0	X	X
		f2			

**Figure 4.3. Karnaugh maps for the Terms and Equations examples.**

The following example will clarify the syntax rules. For the Karnaugh maps shown in Figure 4.3, the dialog box parameters and selections, shown in Figures 4.1 and 4.2, are listed as follows:

Number of variables:	4
Number of functions:	2
Variable names:	apple ball cat dog
Function names:	f1 f2
Input format:	Decimal

Each function may be written in various ways according to the rules given above and as shown below:

{f1}	0 1 5 9 13 14 15.	{0-terms: 2 3 4 6 7 8 10 11 12}
{f1}	0 1 5 9 13 14 15,.	{same as above}
{f1}	0 1 5 9 13 14 15,,.	{same as above}
{f1}	,2 3 4 6 7 8 10 11 12.	{1-terms: 0 1 5 9 13 14 15}
{f1}	,2 3 4 6 7 8 10 11 12,.	{same as above}
{f2}	0 1 5 7 11 15, 3 4 6 8 9 12.	{X-terms: 2 10 13 14}
{f2}	0 1 5 7 11 15, 3 4 6 8 9 12,.	{X-terms: 2 10 13 14}
{f2}	0 1 5 7 11 15,, 2 10 13 14.	{0-terms: 3 4 6 8 9 12}
{f2}	,3 4 6 8 9 12, 2 10 13 14.	{1-terms: 0 1 5 7 11 15}
{f2}	0 1 5 7 11 15, 3 4 6 8 9 12, 2 10 13 14.	{no other term}

An actual input file may appear as follows:

{Decimal Input Format}

```
{f1} 0 1 5 9 13 14 15.
{f2} 0 1 5 7 11 15, , 2 10 13 14.
```

If the input format is not **Decimal**, the same syntax rules would apply. The following are some examples of the actual input file for f1 and f2 with other input format:

Binary Input Format:

{f1} 000— {combined terms 0 1}  
       —01 {combined terms 1 5 9 13}  
       111— {combined terms 14 15}.  
 {f2} 0000 0001 0101 0111 1011 1111,, 1–10 –010 1101.

Hexadecimal Input Format:

{f1} 0 1 5 9 D E F.  
 {f2} 0 1 5 7 b f, 2 A D E.

Octal Input Format:

{f1} 0 1 5 11 15 16 17.  
 {f2} 0 1 5 7 13 17,, 2 12 15 16.

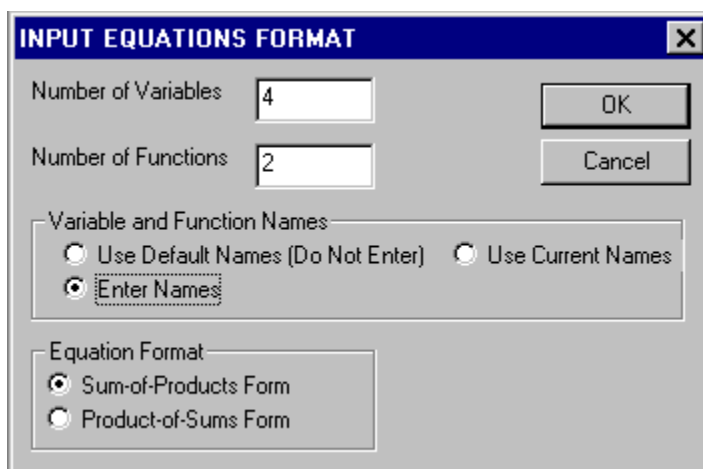
**4.2. Equations (Alt+I E or Ctrl+E)**

Function: Enters Boolean function(s) by equations.

This command brings up a dialog box, shown in Figure 4.4, in which you specify the input-equation parameters and selections. A blank edit window will be created for input equations after you click on **OK** in this dialog box. Except for the Equation Format buttons in this dialog box, the Input Terms Format and Input Equations Format dialog boxes are exactly alike. For information on number of variables, number of functions, and function and variable names, refer to Section 4.1, *Terms*. Equation format and syntax are described in the following subsections.

**EQUATION FORMAT**

The **Equation Format** option provides two options for entering equations. Input equations may be specified with either the *sum-of-products* (SOP) form or the *product-of-sums* (POS) form.



**Figure 4.4. Input Equations Format dialog box.**

**SYNTAX:**

In the following discussion, a separator is a Space, Tab or Enter. A comment is any text delimited by a pair of brackets. Each equation must be terminated by a period.

Each sum-of-products equation consists of a sum of one or more product terms and has the following form:

$$\text{functionname} = \text{term1} + \text{term2} + \dots + \text{termN}.$$

Comments and separators may be placed before and after the functionname, and before and after each term. Each variable in a product term must be followed by a separator or a prime. Separators before and after the prime are optional.

Each product-of-sums equation consists of a product of one or more sum terms and has the following form:

$$\text{functionname} = (\text{sumterm1}) (\text{sumterm2}) \dots (\text{sumtermN}).$$

As indicated, every sum term must be enclosed in parentheses. Since the literals in each sum term are separated by + signs, no additional separators are required. However, extra separators may be placed anywhere in the equation (except, of course, within a variable name). Comments may be placed anywhere except within a term.

#### EXAMPLES:

The following examples will clarify the syntax rules. For the Karnaugh maps shown in Figure 4.3, the Input Equations Format dialog parameters and selections, shown in Figure 4.4, are listed below:

Number of variables: 4  
 Number of functions: 2  
 Function names: f1 f2  
 Variable names: apple ball cat dog  
 Equation format: See below

If the equation format is sum-of-products, the input file may appear as follows:

```
{Equation # 1}
f1      =      apple' ball'   cat '   {term #1}
          +      cat '   dog      {term #2}
          +      apple ball cat   {term #3}
          .      {period to terminate equation}
{Equation # 2}
f2      =      apple'ball'cat'+ball dog+apple cat dog.
```

If the equation format is product-of-sums, the input file may appear as follows:

```
{Equation #1}
f1 =      ( ball + cat' ) (apple + cat ' ) {terms #1 and #2}
          ( apple'+cat+dog)                {term #3}
          (ball'+cat+dog)                   {term #4}
          .                                 {period to terminate equation}
{Equation #2}
f2=(apple'+cat)(ball'+dog)(apple+ball+cat').
```

### 4.3. Truth Table (Alt+I T or Ctrl+T)

Function: Enters Boolean function(s) with a truth table.

This command brings up a dialog box for specifying the truth-table parameters and selections. A blank edit window opens for the input truth table after you click on **OK** in this dialog box. Parts of this dialog box are identical to the Input Terms Format dialog box. For information on the Variable and Function Names buttons, refer to Section 4.1, *Terms*. The other parts of the dialog box are described in Figure 4.5.

**Figure 4.5. Input Truth Table Format dialog box.**

#### TEXT BOXES:

This is similar to section 4.1.

#### FORMAT FOR INPUT COMBINATIONS:

This specifies the input format for the input combinations (input terms). The default selection is the **Straight Binary Order** button. With this selection, the program will display the next input combination when the **Enter** key is pressed; therefore, you will only need to enter the output combinations (output terms). **Straight Binary Order** is not available through the **Change Parameters** or **Open Text File** commands.

While this automatic mode is activated, you are free to modify any part of the truth table; however, the cursor must be at the end of the last line of text in order for the program to display the next input combination. If the insertion point is not at the end of the file, the program will simply display a carriage return.

Once the input combination reaches the maximum value for the number of variables, the program will display an alert dialog box indicating that the automatic mode has been terminated. The other four options are **Decimal**, **Binary**, **Hexadecimal**, and **Octal**. Refer to *Input Format* in Section 4.1, *Terms*, for more information.

**FORMAT FOR OUTPUT COMBINATIONS:**

This specifies the entering format for the output combinations (output terms). The default selection is **Binary**; the other three selections are **Decimal**, **Hexadecimal**, and **Octal**. These selections apply to output values—refer to *Input Format* in Section 4.1 for more information.

**OUTPUT VALUE FOR REMAINING ROWS:**

This specifies how LogicAid should treat the remaining input combinations of the truth table. For example, if the default **0-Terms** is selected, all the unentered input combinations are to be treated as 0-terms for all the output functions.

**SYNTAX**

The syntax for entering a truth table is straightforward. The following key symbols and hints are helpful in understanding the syntax rules.

<u>Key</u>	<u>Definition</u>
input term	input combination typed with the input format
output term	output combination typed with the output format
comments	delimited by a pair of brackets “{” and “}”
separators	combination of one or more <b>Space</b> , <b>Tab</b> , and <b>Enter</b> keys

Each row of the truth table must have an input term and an output term separated by separators and/or comments. A more detailed explanation follows:

- Enter a truth table by rows.
- Each row of the truth table must start with an input term, with optional leading comments and separators.
- Each input term must be followed by either comments and/or separators.
- Each input term must be paired with an output term.
- The truth table input may terminate after one row has been typed. A period is not required to terminate the table.
- The input term of the next row must be separated from the current row's output term with separators and/or comments

**EXAMPLE:**

X1	X2	X3	Z1	Z2	Z3
0	0	0	1	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	-	-	-
1	1	0	-	-	-
1	1	1	-	-	-

**Figure 4.6. Truth table example.**

For the truth table given in Figure 4.6, the dialog box parameters and selections are as shown in Figure 4.5 and are listed as follows:

Number of variables: 3  
 Number of functions: 3  
 Variable & function names: Default  
 Input format: Described below  
 Output format: Described below  
 Output value for remaining rows: X-Terms

The first example uses the default Straight Binary Order input format and the Binary output format. The program first displays a header with the input and output variable names, and then it displays the 000 input term.

Input variables: X1 X2 X3  
 Output functions: Z1 Z2 Z3

000	{Displayed by program}	100	{typed and <b>Enter</b> pressed}
001	{Displayed by program}	010	{typed and <b>Enter</b> pressed}
010	{Displayed by program}	001	{typed and <b>Enter</b> pressed}
011	{Displayed by program}	001	{typed and <b>Enter</b> pressed}
100	{Displayed by program}	011	{typed and <b>F12</b> pressed}

When the **F12** key is pressed:

- a system beep is issued by the program
- the auto-mode is terminated
- the program will treat rows 101, 110, and 111 as X-terms for Z1, Z2, and Z3

you may:

- select **Change Parameters**
- select **Check Syntax** or **Simplification**

You may modify any part of the table during or after the auto-mode.

The second example uses the Binary input format and the Decimal output format. The program displays a blank text window and allows you to enter the truth table without any special program interaction.

Input variables: X1 X2 X3  
 Output functions: Z1 Z2 Z3

000	4	
001	2	
01—	1	{combined rows 010 and 011}
100	3	{end of input—other rows are X-terms}

You may select **Check Syntax**, **Change Parameters**, or **Simplification** at any time during the input.

#### 4.4. PLA Table (Alt+I L or Ctrl+Shift+L)

Function: Enters Boolean function(s) with a PLA table.

This command brings up a dialog box for specifying the truth-table parameters and selections. A blank edit window will be created for the input PLA table after **OK** has been clicked in this dialog box. This dialog box is similar to **Figure 4.5**, except that the only options available are the Variable and Function Names buttons. The limits on the number of variables and functions are the same as for truth tables.

##### FORMAT FOR INPUT AND OUTPUT COMBINATIONS:

The input and output format for the PLA table is similar to the binary (0, 1, –) format in a truth table. Refer to Section 4.3, *Truth Table*, for more information on truth table format. Unlike truth tables, however, the auto-entry mode is not available for PLA tables.

##### SYNTAX:

The syntax for entering a PLA table is the same as for entering truth tables. Refer to Section 4.3, *Truth Table*, for more information.

##### EXAMPLE:

Input Variable Names:	a	b	c	d
Output Function Names:	f1	f2	f3	
01-1	110			
11-1	101			
100-	101			
-01-	100			
--1-	010			
-11-	001			

For the PLA table given above, the dialog box parameters and selections are listed below:

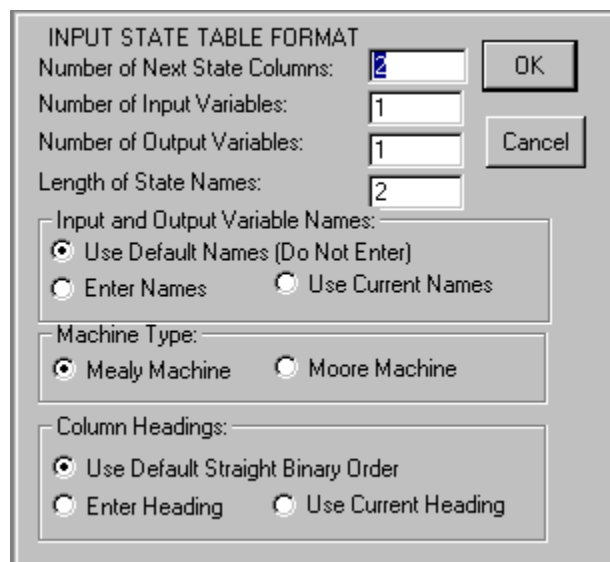
Number of variables:	4
Number of functions:	3
Variable & function names:	Enter Names

You may select **Check Syntax**, **Change Parameters**, or **Simplification** at any time during the input.

#### 4.5. State Table (Alt+I S or Ctrl+B)

Function: Enters a state table for a sequential machine.

This command brings up a dialog box in which you specify the input parameters and selections for a state table. An edit window is created for the new state table after **OK** has been clicked in the dialog box. This dialog box is shown in Figure 4.7; its contents are described in the following subsections.



The dialog box is titled "INPUT STATE TABLE FORMAT". It contains four text input fields: "Number of Next State Columns:" with a value of 2, "Number of Input Variables:" with a value of 1, "Number of Output Variables:" with a value of 1, and "Length of State Names:" with a value of 2. To the right of these fields are "OK" and "Cancel" buttons. Below the input fields are three grouped sections, each with a label and three radio button options:
 

- Input and Output Variable Names:**
  - ☒ Use Default Names (Do Not Enter)
  - ☐ Enter Names
  - ☐ Use Current Names
- Machine Type:**
  - ☒ Mealy Machine
  - ☐ Moore Machine
- Column Headings:**
  - ☒ Use Default Straight Binary Order
  - ☐ Enter Heading
  - ☐ Use Current Heading

**Figure 4.7. Input State Table Format dialog box**

**TEXT BOXES:**

There are four text boxes in this dialog box. The **Tab** key may be used to move from one text box to another. Each input number must consist of valid decimal digits.

The Number of Next State Columns text box specifies the number of columns of next states in the state table. For a Mealy machine, it also specifies the number of output columns in the state table. Its limitations are listed below:

Default: 2  
 Minimum: 1  
 Maximum: the lesser of 32 or  $(256 \div \text{the number of output variables})$

The Number of Input Variables text box specifies the number of input variables for the state table. Its limitations are listed below:

Default: 1  
 Minimum: the smallest integer  $\geq \log_2(\text{Number of Next State Columns})$   
 Maximum: 32

The Number of Output Variables text box specifies the number of output variables for the state table. Its limitations are:

Default: 1  
 Minimum: 0  
 Maximum: the lesser of 32 or  $(256 \div \text{the number of next state columns})$



The Length of State Names text box specifies the maximum number of characters in each state name for the state table. Its limitations are:

Default:	2
Minimum:	1
Maximum:	5

Although there is no text box for entering the number of rows, LogicAid will accept a maximum of 255 rows and a minimum of 1 row. If the **State Table Checker** command is to be selected later, each state table is limited to 8 columns, 16 input variables, 16 output variables, and 32 rows. Refer to Section 5.5, *State Table Checker*, for more information on the checker routine.

#### INPUT/OUTPUT VARIABLE NAMES:

See Section 4.1, *Terms*, for information on this option. In addition to the rules stated in Section 4.1 the input-variable names may not duplicate the state-variable names that are specified with the **State Assignment** command.

#### MACHINE TYPE:

This specifies the state-machine type. The default selection is **Mealy Machine**. In this case, each row of the state table must have the number of output columns equal to the number of next-state columns. On the other hand, selecting **Moore Machine** allows one and only one output column for each row of the state table.

#### COLUMN HEADINGS:

This gives you three options for entering the column headings. The setup of this set of buttons is very similar to the one for variable and function names. The default button is **Use Default Straight Binary Order**. In this case, the default values for a 5-variable and 6-column state table are 00000, 00001, 00010, 00011, 00100, and 00101.

If **Enter Heading** is selected, a second dialog box will be displayed for entering column headings. This Heading dialog box is similar to the one shown in Figure 4.2. The only difference is that the text boxes in the Heading dialog box will only accept characters in the binary (0, 1, –) format. See Section 4.1, *Terms*, for more discussion on the way the Heading dialog box is set up.

#### A WORD ABOUT THE AUTO-FORMATTER:

By default, each state table will be automatically formatted when its syntax is checked. The program will automatically display a header and footer that you cannot edit. You must select the **Change Parameters** command to modify the state-table parameters and options. Each selection of the **Change Parameters** command will cause an automatic update of the header in the window. In addition, the program sets up tab positions to help you enter the state table. Using these tab stops, you may tab from column to column without having to insert separators.

**SYNTAX:**

The state-table input syntax is also straightforward. The following key symbols and hints are helpful in understanding the syntax rules listed below.

<u>Key</u>	<u>Definition</u>
PS	present-state name
NS	next-state name
output	an output value
separators	combination of one or more <b>Space</b> , <b>Tab</b> , and <b>Enter</b> keys
comments	delimited by a pair of brackets "{" and "}"

Each PS and NS name must have a maximum of 1 to 5 alphanumeric characters, as specified in the Length of State Names text box. In the case of an unspecified next state, either "-" or "--" may be used as names. Note that a hyphen is not a valid part of a PS name.

Each output value must be specified in the binary (0, 1, -) format with the length of the binary string equal to the number of output variables. Also, the number of output columns per row depends on the machine type selected.

Following is a list of syntax rules for the state-table input:

- Each row must start with a PS name with optional leading separators.
- Each PS is followed by separators and pairs of NS's and separators.
- The number of NS-and-separator pairs must equal the number of next state columns.
- Output columns follow the NS columns.
- A Moore machine has only one output column.
- A Mealy machine has a number of output columns equal to the number of next state columns.
- There must be spaces or tabs between the output columns.
- Comments are allowed after the last output column of each row.
- Each table must have at least 1 row and at most 255 rows.

Example:

Present State	Next State		Outputs	
	X1	X2=	(Z1 Z2 Z3)	
	00	11	00	11
S0	S2	S1	000	11-
S1	S0	--	001	-00
S2	--	S2	110	111

**Figure 4.8. State table example.**

For the state table in Figure 4.8, the input parameters of the state table are those listed below.

Number of columns: 2  
 Number of input variables: 2  
 Number of output variables: 3  
 Length of state names: 2  
 Variable names: Default  
 Machine type: Mealy  
 Column headings: entered as 00, 11

Input screen: (before any syntax check):  
 (The header and footer are provided by the program)

PS	NS		OUTPUTs*	Input-Var
	0	1	0 1	X1
	0	1	0 1	X2

---

S0	S2	S1	000	11-	{row 1}
S1	S0	—	001	-00	{row 2}
S2	—	S2	110	111	{row 3}

---

\* Z1 Z2 Z3

Input file: (after selecting the **Check Syntax** command):

PS	NS		OUTPUTs*	Input-Var
	0	1	0 1	X1
	0	1	0 1	X2

---

S0	S2	S1	000	11-	{row 1}
S1	S0	—	001	-00	{row 2}
S2	—	S2	110	111	{row 3}

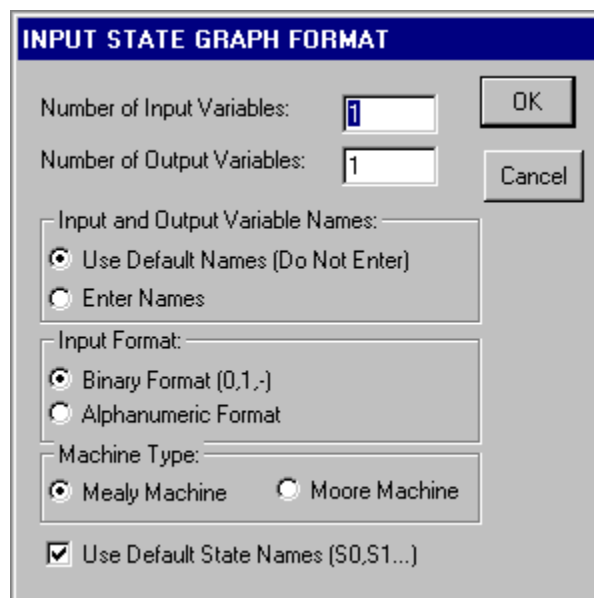
---

\* Z1 Z2 Z3

#### 4.6. State Graph (Alt+I G or Ctrl+G)

**Function:** Enters a state graph for a sequential machine.

This command brings up a dialog box in which you specify the input parameters and selections for a state graph. A graph window is created for the new state graph after you click on **OK** in the dialog box. The dialog box is shown in Figure 4.9; its contents are described in the following subsections.



The dialog box is titled "INPUT STATE GRAPH FORMAT". It contains the following fields and options:

- Number of Input Variables:** A text box containing the value "1".
- Number of Output Variables:** A text box containing the value "1".
- Input and Output Variable Names:** A group box containing two radio buttons:
  - ☒ Use Default Names (Do Not Enter)
  - ☐ Enter Names
- Input Format:** A group box containing two radio buttons:
  - ☒ Binary Format (0,1,-)
  - ☐ Alphanumeric Format
- Machine Type:** A group box containing two radio buttons:
  - ☒ Mealy Machine
  - ☐ Moore Machine
- ☒ Use Default State Names (S0,S1...)

Buttons for "OK" and "Cancel" are located on the right side of the dialog box.

**Figure 4.9 Input State Graph dialog box**

#### TEXT BOXES:

There are two text boxes in this dialog box. The Tab key may be used to move from one text box to another. Each input number must consist of valid decimal digits.

The Number of Input Variables text box specifies the number of input variables for the state graph. Its limitations are listed below:

Default:	1
Minimum:	1
Maximum:	24

The Number of Output Variables text box specifies the number of output variables for the state graph. Its limitations are :

Default:	1
Minimum:	1
Maximum:	24

If you try to convert a state graph to a state table, you cannot complete this command if there will be more than 32 columns in the state table. If you try to check a partial state graph, the maximum number of states is 32, and there cannot be more than 8 columns after you convert the partial graph to a state table.

#### INPUT/OUTPUT VARIABLE NAMES

See Section 4.4, *PLA Table*, for the information on this option.

**INPUT FORMAT**

The default format for the input and output values is Binary. In this case, each input and output variable must be assigned a value of 0, 1 or – (don't care). If you choose Alphanumeric format, then each input combination is specified by a product term, where an unprimed variable represents a 1, a primed variable represents a 0, and a missing variable represents a don't care. Each output combination is specified by a product term, where an unprimed variable represents a 1 output, and either a primed variable or a missing variable represents a 0 output. For example, if the input variables are  $X_1, X_2, X_3$  and the output variables are  $Z_1, Z_2, Z_3$ , then 01– /100 in binary format means the same as  $X_1'X_2 / Z_1$  or  $X_1'X_2 / Z_1 Z_2'Z_3'$  in alphanumeric format.

**MACHINE TYPE:**

This specifies the state-machine type. The default selection is Mealy Machine. In this case, the state box can only show the state name and the arc label must show the input and output values. On the other hand, selecting the Moore Machine allows the state box to show the state name and the output values, but the arc label can only show the input values.

**DEFAULT STATE NAMES OPTIONS:**

If the Default State Names option is checked, LogicAid will attach a default state name ( $S_0, S_1, S_2, \dots$ ) to each the new state box.

**SYNTAX:**

When binary format is used, the syntax rules for the state graph input are:

- '–' and '/' are illegal characters for a state name.
- Only '0', '1', and '–' (don't care) are legal characters for the input and output values.
- There must be a '/' to separate the state name and output values in a state, or to separate input and output values in a label.

**SIDE MENU**

1. 

**State**

 To enter states, first click on **State**. Then move the cursor to the starting location (upper left-hand corner) of a state box and click. When the state box is displayed, type in the state text and hit Enter.
2. 

**Arc**

 To enter the arcs that connect the states, first click on **Arc**. Then click on the *edge* of a state box at the starting point of an arc. Move the cursor to a place where you want to bend the arc and click. Repeat for each angle, and then double click at the end point of an arc on the *edge* of a state box.
3. 

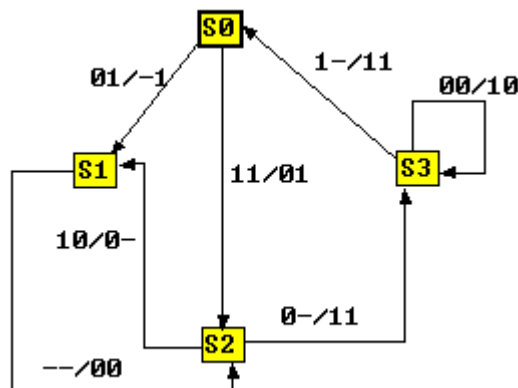
**Label**

 To enter arc labels, first click on **Label**. Then click near one of the arcs at the starting point of a label. LogicAid will then highlight an arc near the click location. If the highlighted arc is the correct arc, type in the label text followed by Enter. If the highlighted arc is the wrong one, type control-Z to

cancel this arc, and LogicAid will highlight another nearby arc. Continue in this manner until the correct arc is highlighted.

4. **Comm** To enter comments, first click on **Comm**. Then click at the starting location of a comment. When the text-cursor appears, type in the comment text followed by **F12**.
5. **Edit** To edit the text in a state box, label or comment, first click on **Edit**. Then click on the text you want to edit. When the text cursor appears, edit the text and then hit Enter (**F12** for comments).
6. **Move** To move a box, label, or comment in the state graph, first click on **Move**. When the cross-cursor is displayed, place it within the object you want to move then drag the object to its new location. You can also move the end-point or a corner point of an arc, but you cannot move the starting point.
7. **Del** To delete an object in the state graph window, click on **Del**. Then click on the object you want to delete. When the alert box appears, click **OK** to confirm deletion or cancel.

EXAMPLES: (1)



**Figure 4.10 Mealy State Graph example.**

For the state graph in Figure 4.10, the parameters are listed below:

Number of Input Variables:	2
Number of Output Variables:	2
Machine Type:	Mealy
Variable names:	Default
Input format:	Binary

After you select **Convert to State Table** from the Input Menu, you get the state table shown in Figure 4.11:

PS	NS				OUTPUTS*				INPUT-VARS
	0	0	1	1	0	0	1	1	X1
	0	1	0	1	0	1	0	1	X2
S0	-	S1	-	S2	--	-1	--	01	
S1	S2	S2	S2	S2	00	00	00	00	
S2	S3	S3	S1	-	11	11	0-	--	
S3	S3	-	S0	S0	10	--	11	11	
* Z1 Z2									

Figure 4.11 State Table for Figure 4.10.

(2)

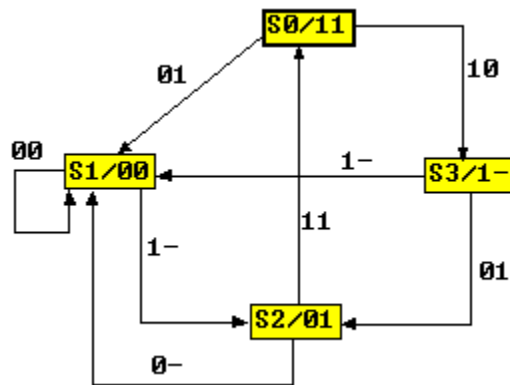


Figure 4.12 Moore State Graph example.

For the state graph in Figure 4.12, the parameters are listed below:

Number of Input Variables:	2
Number of Output Variables:	2
Machine Type:	Moore
Variable names:	Default
Input format:	Binary

After you select **Convert to State Table** from the **Input Menu**, a **Conversion Options** dialog box will open. If '0' and '1' is selected, you get the state table listed on Figure 4.13:

PS	NS				OUTPUTs*	INPUT-VARS
	0	0	1	1		X1
	0	1	0	1		X2
S0	S1	-	S3	-	11	
S1	S1	-	S2	S2	00	
S2	S1	S1	-	S0	01	
S3	-	S2	S1	S1	1-	
* Z1 Z2						

Figure 4.13 State Table for Figure 4.12.

(3)

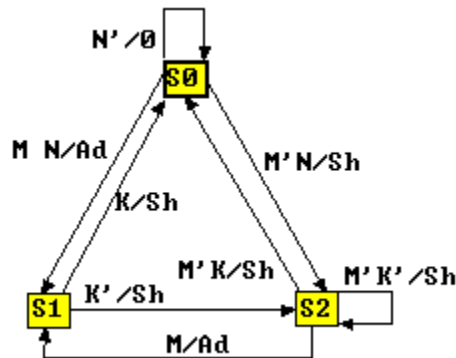


Figure 4.14 State Graph with Alphanumeric Input.

For the state graph in Figure 4.14, the parameters are listed below:

Number of Input Variables: 3  
 Number of Output Variables: 2  
 Machine Type: Mealy  
 Variable names: Entered  
 Input format: Alphanumeric

After you select **Convert to State Table** from the Input Menu and choose the 0, 1, - option, you get the state table listed below:

PS	NS								OUTPUTs*	INPUT-VARS							
	0	1	1	-	-	-	-	-	0	1	1	-	-	-	-	-	N
	-	-	-	1	0	1	-	0	-	-	-	1	0	1	-	0	K
	-	0	1	-	-	0	1	0	-	0	1	-	-	0	1	0	M
S0	S0	S2	S1	-	-	-	-	-	00	10	01	--	--	--	--	--	
S1	-	-	-	S0	S2	-	-	-	--	--	--	10	10	--	--	--	
S2	-	-	-	-	-	S0	S1	S2	--	--	--	--	--	10	01	10	

\* Sh Ad

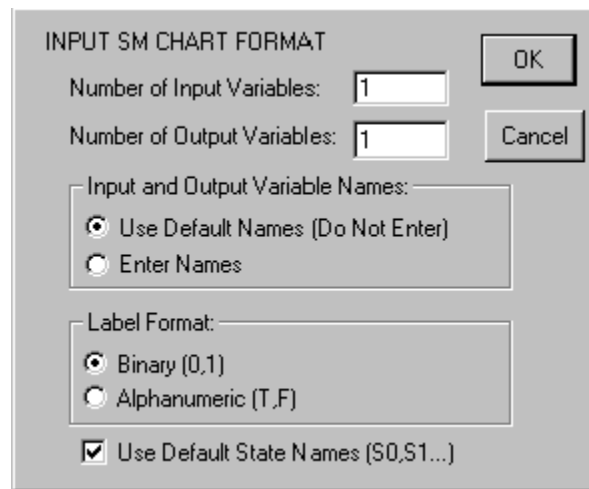
Figure 4.15 State Table for Figure 4.14.



## 4.7 SM Chart (Alt+I M or Ctrl+M)

**Function:** Enters an SM chart for a sequential machine.

This command brings up a dialog box in which you specify the input parameters and selections for an SM chart. A graph window will be created for the new SM chart after **OK** is clicked in the dialog box. The dialog box is shown in Figure 4.16; its contents are described in the following subsections.



**Figure 4.16 Input SM Chart dialog box.**

### TEXT BOXES:

The limitations for the number of input and output variables are the same as for state graphs. Refer to Section 4.6 for more information.

### INPUT AND OUTPUT VARIABLE NAMES

See Section 4.4 for the information on this option.

### LABEL FORMAT

The default format for the input and output values is Binary. In Binary mode, arcs from condition boxes will be labeled 0 or 1. In the Alphanumeric mode, the arcs will be labeled T (true) or F (false).

### DEFAULT STATE NAMES OPTION:

If the Default State Names option is checked, LogicAid will attach a default state name (S0, S1, S2, ...) to each the new state box.

### SYNTAX:

When binary format is used, the syntax rules for the state graph input are:

- ‘–’ and ‘/’ are illegal characters for a state name.
- Only ‘0’, ‘1’, and ‘–’ (don’t care) are legal characters for the input and output values.

- There must be a '/' to separate the state name and output values in a state, or to separate input and output values in a label .

**SIDE MENU**

1. **State** To enter states, first click on **State**. Then move the cursor to the starting location (upper left-hand corner) of a state box and click. When the state box is displayed, type in the state text and hit **Enter**. The state text consists of a state name or, if there is a Moore output, a state name followed by a '/' and a list of output variables that are 1 in the state.
2. **Cond** To enter a condition box, first click on **Cond**. Then click at the starting location, type in a single input variable or its complement, and hit **Enter**.
3. **Out** To enter an output box, first click on **Out**. Then click at the starting location, type in the output variables that are 1, and hit **Enter**. All other output variables are assumed to be 0 for that box. For example, the output expression  $Z1\ Z3$  means that  $Z1 = Z3 = 1$  and  $Z2 = 0$ .
4. **Arc** To enter the arcs which connect states, condition boxes, and output boxes, first click on **Arc**. Then click on the *edge* of a box at the starting point of an arc. Move the cursor to a place where you want to bend the arc and click. Repeat for each angle, and then click at the end point of an arc on the *edge* of a box. Each state or output box can have only one exit arc.
5. **Label** To enter arc labels, first click on **Label**. The two arcs that exit a condition box can only be labeled 0 or 1 (T or F in alphanumeric mode). First click near the arc you want to label with a 0 (or F), and then click near to other arc to label it 1 (or T).
6. **Comm** To enter comments, first click on **Comm**. Then click at the starting location of a comment, and type in the comment text followed by **F12**.
7. **Edit** To edit the text in a box, label or comment, first click on **Edit**. Then click on the text that you want to edit. When the text cursor appears, edit the text and hit Enter (**F12** for comments).
8. **Move** To move a box, label, or comment in the state graph, first click on **Move**. When the cross-cursor is displayed, place it within the object you want to move; then drag the object to its new location. You can also move the end-point or a corner point of an arc, but you cannot move the starting point.

9. **Del** To delete an object in the state graph window, click on **Del**. Then click on the object you want to delete. When the alert box appears, click **OK** to confirm deletion or cancel.

EXAMPLE:

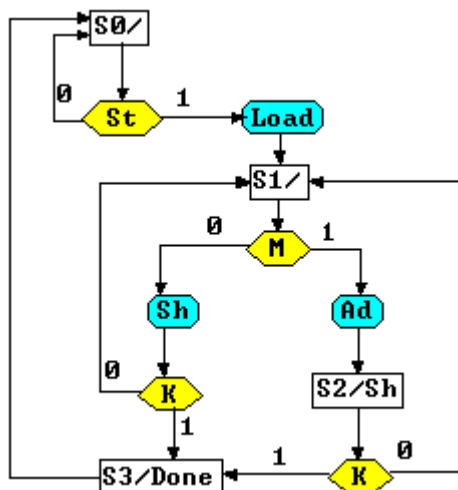


Figure 4.17 SM Chart example.

For the SM chart example in Figure 4.17, the parameters are listed below:

Number of Input Variables: 3  
 Number of Output Variables: 4  
 Variable names: Enter Names  
 Label format: Binary(0,1)

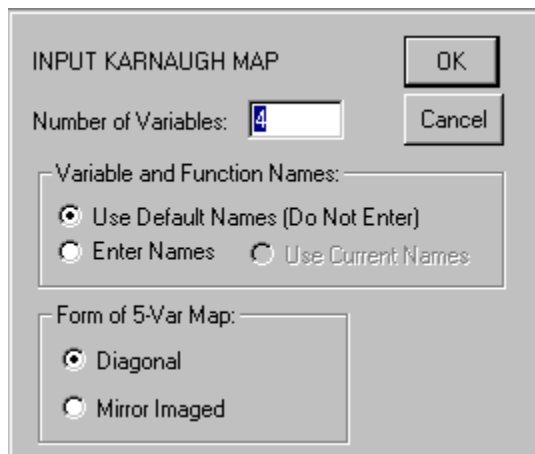
After you select Convert to PLA Table from the Input Menu, you get the following PLA table:

Input Variable Names:	Q1	Q2	St	M	K	
Output Function Names:	D(Q1)	D(Q2)	Load	Ad	Sh	Done
000--	000000					
001--	011000					
01-1-	100100					
01-01	110010					
01-00	010010					
10--0	010010					
10--1	110010					
11---	000001					

#### 4.8 Karnaugh Map (Alt+I K or Ctrl+K)

Function: Input Boolean functions by a Karnaugh map and derive minimum equations from the map, or enter an equation and plot the map from the equation.

This command brings up a dialog box, shown in Figure 4.18, for specifying the input Karnaugh map parameters and selections. A graph window will be created for the new Karnaugh map input after **OK** has been clicked. The contents of the dialog box are described in the following subsections.



**Figure 4.18 Input Karnaugh Map dialog box**

##### NUMBER OF VARIABLES:

The Number of Variables specifies the number of input variables. Its limitations are listed below:

Default:	4
Minimum:	3
Maximum:	5

##### VARIABLE AND FUNCTION NAMES:

The input variable and output function names can be specified with one of three options. These three options are **Use Default Names**, **Enter Names**, and **Use Current Names**.

If the **Use Default Names** option is selected, the program will initialize the necessary variable and function names. The default variables are A, B, C, etc. and the default function name is F.

If the **Enter Names** option is selected and you click on OK, the program will bring up a dialog box as shown in Figure 4.2 for entering names. For Karnaugh map input, the variable and function names are 1 or 2 characters long. Refer to Section 4.1, *Terms*, for other rules for entering the names.

Whenever the **Input Karnaugh Map Format** dialog box comes up with the **Use Default Names** as the current selection, the **Use Current Names** option is deactivated. On the other hand, if the **Enter Names** option is selected, the **Use Current Names**

option is activated. In the latter case, the **Use Current Names** option may be selected if the same non-default names are to be used, but you do not wish to have the **Input Variable Names** dialog box displayed.

#### FORM OF 5-VAR MAP:

For the 3-variable and 4-variable map, there is only one form to display, but for a 5-variable map, LogicAid provides 2 forms, which are **Mirror Imaged** and **Diagonal**. The default selection is **Diagonal**.

#### SIDE MENU

The Karnaugh map window is divided into two sections, the Karnaugh map at the top and a space for equations at the bottom. Use the side menu to plot the map or to enter equations.

1. **Plot 1's** To enter 1's in the map, first click on this button. Then click on each square in which you want a 1, or drag the mouse across a group of several squares in which you want 1's. Selecting a square that already contains a 1 will erase the 1.
2. **Plot 0's** To enter 0's in the map, first click on this button. Then click on each square in which you want a 0, or drag the mouse across a group of several squares in which you want 0's. Selecting a square that already contains a 0 will erase the 0.
3. **Plot X's** To enter X's (don't cares) in the map, first click on this button. Then click on each square in which you want an X, or drag the mouse across a group of several squares in which you want X's. Selecting a square with an X in it will erase the X.
4. **Loop** To enter loops in the map, first click on this button. Then select squares you want to loop by clicking on the squares or by dragging across a group of squares. Each selected square will be highlighted. (To deselect a square, click on it again.) When you have selected a group of 2, 4, 8 or 16 squares which you want to loop, hit **Enter**. If the selected squares form a valid implicant, the loop is drawn; otherwise, an error message is displayed.
5. **Fill 0's** To fill all blank squares with 0's, click on this button.
6. **Fill 1's** To fill all blank squares with 1's, click on this button.
7. **Clear** Click on this button to clear something from the Karnaugh map window. When the dialog box appears, select **Clear Loops**, **Clear Equations**, **Enter new map**, or **Read map from file**. Selecting **Clear Loops** will erase all loops in the map. Selecting **Clear Equations** will clear all the equations in the window.

Selecting **Enter new map** will erase the whole Karnaugh map and display the Karnaugh map input dialog so that a new map can be entered. Selecting **Read map from file**, will erase the whole window and display the file dialog so that a file may be selected.

8. Text To enter equations below the map, first click on this button. A text-cursor will appear in the equation section of the window and you can start typing in the equation text. The syntax rules for the equation input are the same as in Section 4.2, *Equations*. If several equations are input, only the syntax of the last equation will be checked.

#### RULES FOR PLOTTING MAPS

For a map input, you can select “Plot 1’s”, “Plot 0’s”, or “Plot X’s” to fill the map, but you do not have to fill the whole map. The rules are as follows:

- If only 1-terms are entered in the map, all blanks will be taken as 0-terms.
- If only 0-terms are entered in the map, all blanks will be taken as 1-terms.
- If there are 1-terms and 0-terms in the map, all blanks will be taken as X-terms.
- If there are 1-terms, 0-terms and X-terms in the map, no blanks are allowed.
- At least one 1-term or one 0-term must be filled in the map.

#### CORRESPONDENCE BETWEEN EQUATIONS AND LOOPS:

If there are loops entered in the map, the loops must be consistent with the equation; that is, the loops of the map must correspond to the terms of the equation. If you want to enter a map with a **Sum-of-Products** equation, you should enter a map with 1-term loops. On the other hand, if you want to enter a map with a **Product-of-Sums** equation, you should enter a map with 0-term loops. The rules are as follows:

- A 1-term loop is composed of 1-term and X-term entries.
- A 0-term loop is composed of 0-term and X-term entries.
- It is inconsistent to enter a **SOP** equation if the map has 0-term loops.
- It is inconsistent to enter a **POS** equation if the map has 1-term loops.
- If there are no loops on the map, either an **SOP** or **POS** equation may be entered.

#### EXAMPLES:

(1)

Input variables:	4
Equation form:	Sum of Products

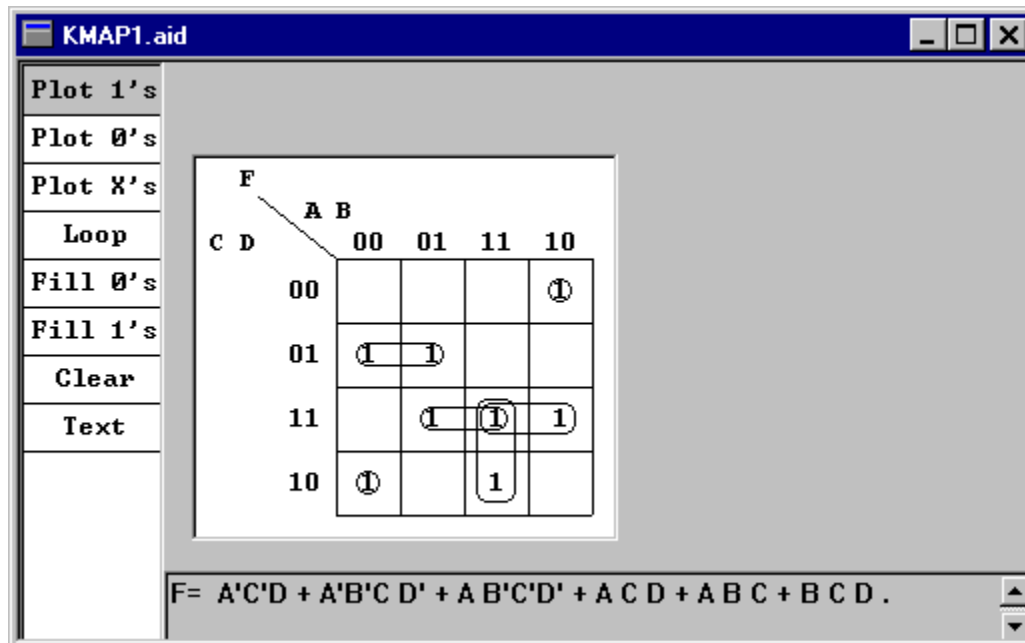


Figure 4.19 Karnaugh Map Example

- (2) Input variables: 5  
Equation form: Sum of Products  
Map form: Diagonal

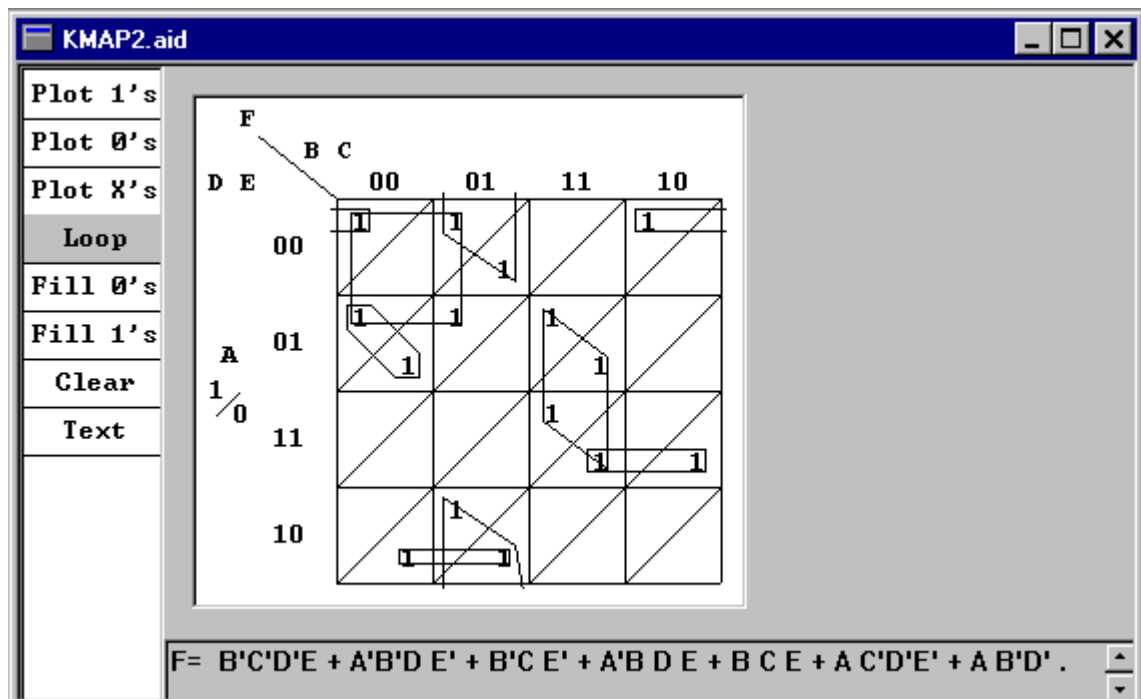


Figure 4.20 Karnaugh Map Example

- (3) Input variables: 5  
Equation form: Product of Sums  
Map form: Mirror-Imaged

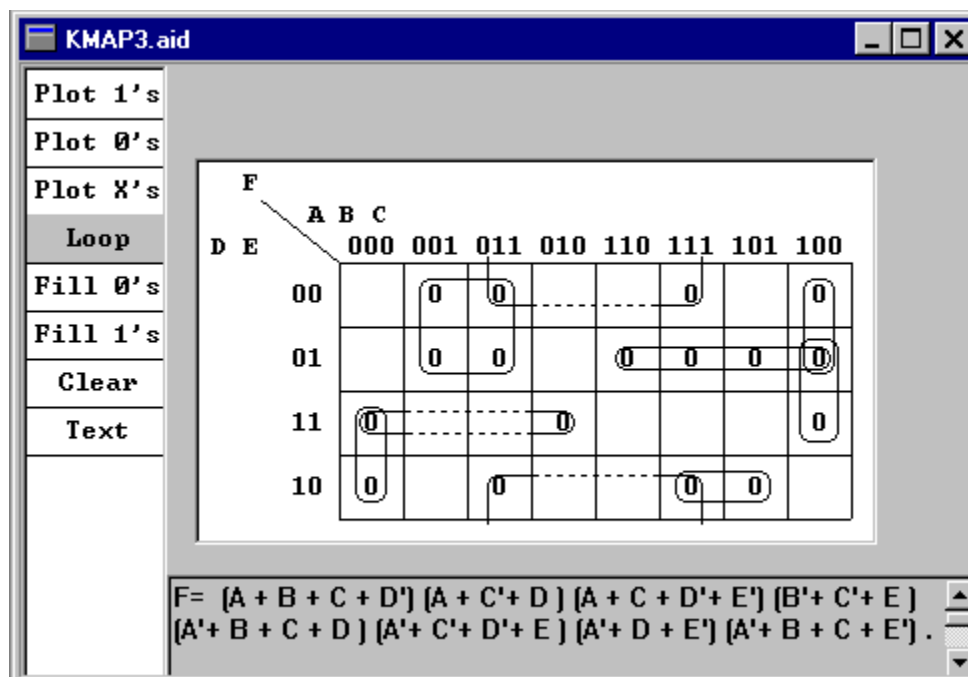


Figure 4.21 Karnaugh Map Example

#### 4.9. Convert to State Table (Alt+I C)

Function: Convert a state graph to a state table.

This command will check syntax errors for the state graph and if there is no error, it will convert the graph to a state table and display it in the Edit window. The syntax rules are given in Section 4.5, *State Table*.

If there are syntax errors, an alert box will appear and give you an error message. If the error comes from an arc, the arc will be highlighted when the alert box appears; but once you click on **OK**, the highlight will disappear. If the error comes from a state box or a label box, a blinking vertical bar will appear in that box, and you should correct the data in that box.

If there are no syntax errors, a dialog box will allow you to choose the type of column headings for the state table. If **Use '0' and '1'** is selected, all of the column headings in the state table will be completely specified, and if there are  $n$  input variables, up to  $2^n$  columns may be generated. If **Use '0', '1', and '-'** is selected, then each unique input combination in an arc label will generate a corresponding column heading.

If you try to modify the parameters or the input data of the state table, an alert box will appear and ask you to reconfirm. After you change the parameters or input data, the state table will not be consistent with the state graph. If you still want to modify it, click **OK** and change anything you want; the alert box will not appear anymore.



#### 4.10. Convert to PLA Table (Alt+I V)

Function: Converts a state graph or SM chart into a PLA Table.

This command will check syntax errors for a state graph or SM chart and if there are no errors, it will convert the graph or chart to a PLA table and display it in a new edit window.

#### 4.11. Check Syntax (Alt+I Y or Ctrl+Y)

Function: Checks the syntax of the current input data.

This command checks the input data for syntax errors. It automatically calls the appropriate syntax-checking routine for the current input document. Each routine adheres strictly to the syntax rules described in Sections 4.1 to 4.4.

After this command has been selected, the cursor will change to an hourglass. If an error is detected, the program will display an alert box giving a detailed description of the first error in the input data. You should click **OK** or press the **Enter** or **F12** key to close the alert box. The blinking bar cursor will be repositioned to indicate the location of the error in the input data. Usually, this cursor will be positioned directly following this first error.

If the input text is syntactically correct, no alert box is displayed. For a state table, the program will reformat the input window only if the table is free of syntax errors. The **Check Syntax** operation is completed when the hourglass cursor becomes an edit or an arrow cursor.

Note that for every routine command, the input data must be syntactically correct before the routine is executed. The routine will automatically call the syntax-checking routine first; therefore, you are not obligated to check syntax before calling any command from the **Routine** menu.

#### 4.12. Change Parameters (Alt+I P or Ctrl+Shift+P)

Function: Allows you to change any parameter of the current input document.

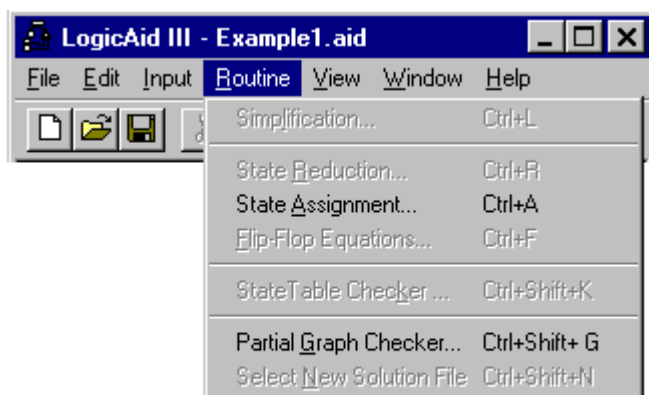
This command redisplay the appropriate input dialog box showing the current parameters and selections. This allows you to update or change any appropriate item in the dialog box. If applicable, the second (Names) or the third (Heading) dialog box will be displayed for modification.

#### 4.13. Change Reset State (Alt+I A)

Function: Change the reset state in a state graph.

This command will allow you to change the reset state in a state graph. When this command is selected, a dialog box will pop up indicating that the first state that is clicked on will become the new reset state. To select a new reset state, simply click on the **OK** button on the dialog box, and then click on the new reset state.

## 5. The Routine Menu



The **Routine** menu is used for selecting the routine appropriate for the input type. The **Simplification** command is the only routine available if the input type is Terms, Equations, or Truth Table. Only the **Partial Graph Checker** command is available for the State Graph input type, while the **State Reduction**, **State Assignment**, **Flip-Flop Equations**, and **State Table Checker** commands are available for the State Table input type.

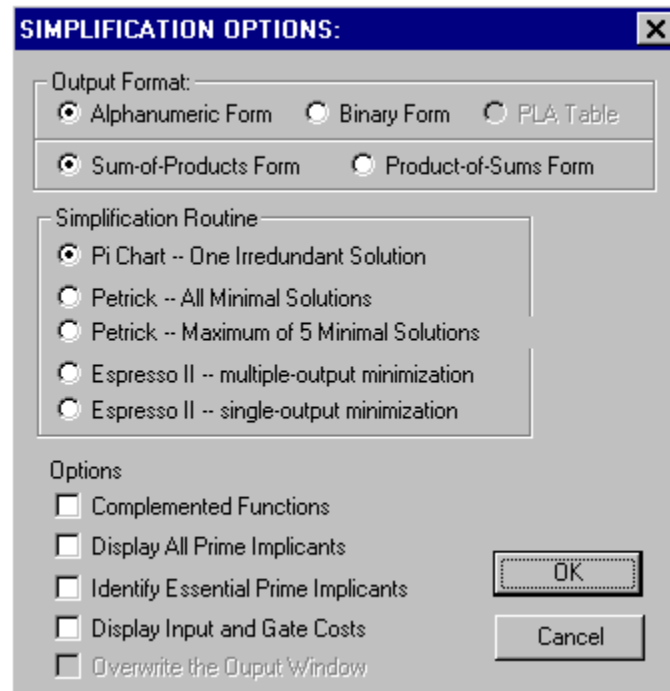
### 5.1. Simplification (Alt+R L or Ctrl+L)

Function: Simplifies the combinational Boolean functions.

This command brings up a dialog box used to specify the simplification options. Before this is done, this routine checks the syntax of the input data. The input file must be syntactically correct before the dialog box is displayed.

The **Simplification** command is activated for Terms, Equations, Truth Table, and PLA table input types. For the State Table input type, this command is called by the **Flip-Flop Equations** command to simplify the flip-flop input equations. The Simplification dialog box is shown in Figure 5.1, and its contents are described in the following subsections.

You may abort the **Simplification** command during execution (after **OK** has been clicked in the dialog box), by simultaneously pressing down the **Control-Period (Ctrl+.)** keys. For a large number of variables, the abort process may take more time. After the command has been aborted, the output window, if one has already been created, will remain on the screen.



**Figure 5.1. Simplification Options dialog box**

#### OUTPUT FORMAT:

There are three parts to the output format. First, the output equations may be displayed in either Alphanumeric or Binary Form. The PLA Table output form is active only if the input is a PLA table. If the Alphanumeric format is selected, the equations will be displayed with the variable names specified by the Input Variable Names dialog box. If the Binary format is selected, the equations will be printed in the 0, 1, and – format. For the third option, the output will be in PLA Table form.

The second half of the output format selects the type of equations to be displayed. As implied by the selections, the equations may be displayed in either the sum-of-products or product-of-sums form.

#### SIMPLIFICATION ROUTINE:

There are currently three simplification routines available in this application program. The default selection is the *PI Chart* routine. PI Chart finds one and only one irredundant two-level solution using the prime implicant chart. This method does not guarantee a minimal solution when there is a cyclic chart.

The other simplification method is the *Petrick* routine. This method finds all the minimal solutions based on minimizing the gate inputs. You have the option of displaying either all the solutions or a maximum of five solutions. This routine first finds the essential prime implicants and then applies the column dominance procedure as in the PI Chart routine. These are done only once to reduce the number of terms before the actual Petrick routine is used. Note that the program will only handle a maximum of 256 prime implicants; this means that the program will not simplify an equation with more than 256 prime implicants after applying the essential prime implicant routine and the column dominance routine. If the equation does exceed 256 prime implicants at that time, the program will display an alert box.

Due to the limitation on the number of prime implicants, the PI Chart is recommended when a large number of terms is to be simplified. Furthermore, Petrick's method is slower than the PI Chart routine. Petrick's method is recommended only when a definite minimal solution is required and when the number of terms and variables is small.

The third simplification method is the Espresso II - Multiple-output minimization and Espresso II - Single-output minimization. In most cases, Espresso finds a minimum-row PLA table or the corresponding logic equations.

#### OPTIONS:

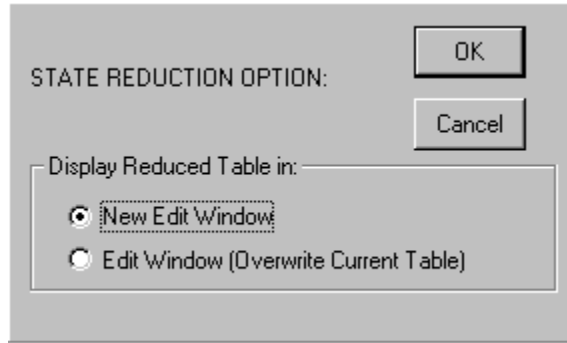
Currently, there are five options in addition to those described above. These five options are available in the form of check boxes; if a box is checked, the corresponding option will be implemented.

1. Complemented Functions:  
Complemented functions are to be simplified and displayed.
2. Display All Prime Implicants:  
All prime implicants of each function are displayed before the simplified equation is displayed. Note that both the PI Chart and the Petrick routines generate prime implicants with the iterated consensus method.
3. Identify Essential Prime Implicants:  
All essential prime implicants are identified with an asterisk in the equation.
4. Display Input and Gate Costs:  
The input and gate costs of each equation are displayed separately right after the equation is displayed. In addition, the total input and gate costs of all flip-flop and output equations are displayed for the **Flip-Flop Equations** command.
5. Overwrite the Output Window:  
The current contents of the output window are overwritten when new outputs are displayed. Note that this option is activated only if the output window is on the screen. In Figure 5.1, this option is deactivated because the output window is not on the screen.

## 5.2. State Reduction (Alt+R R or Ctrl+R)

Function: Reduces a state table.

This command first checks the syntax of the input state table. If the input data contain no syntax errors, the program brings up the dialog box shown in Figure 5.2. You may select the window for displaying the reduced state table. The default selection is the New Edit Window, while the current edit window may be selected to be overwritten. If a state assignment has been made, it will be removed from the reduced state table.



**Figure 5.2. State Reduction Option dialog box**

The state-table reduction routine first applies the row-matching procedure to eliminate redundant states. The state table is then reduced with a next-class table. Note that although the state table allows unspecified next states and output values, the program treats the unspecified next state as a separate state and treats an unspecified output as a separate output value.

You may abort the **State Reduction** command during execution (after you click on **OK** in the dialog box) by simultaneously pressing down the **Control-Period (Ctrl+.)** keys. After the command is aborted, the output window, if one has already been created, will remain on the screen.

### 5.3. State Assignment (Alt+R A or Ctrl+A)

Function: Allows you to specify the state assignment for the current state table, state graph, or SM chart.

The State Assignment Options dialog box is shown in Figure 5.3; its contents are described in the following subsections. LogicAid will check the input syntax before this dialog box is displayed.

#### NUMBER OF STATE VARIABLES:

This text box specifies the number of state variables for the state table. When the dialog box is first displayed, the text box will display either the minimum number or the current number of state variables. The text box limitations are:

Default:	the smallest integer $\geq \log_2(\text{the number of rows})$
Minimum:	the smallest integer $\geq \log_2(\text{the number of rows})$ , or 1 for a 1-row table
Maximum:	(16 minus the number of input variables)

#### STATE VARIABLE NAMES:

This is similar to the **Input and Output Variable Names** option from the Input dialog boxes. Note that if the **Enter Names** option is chosen, the State Variable Names dialog box checks for duplicate names with both variable names and state variable names. Refer to Section 4.1, *Terms*, for more information on entering names.

**Figure 5.3. State Assignment Options dialog box**

#### ASSIGNMENT ROUTINES:

There are three options for the assignment routine. If the **Use Default Straight-Binary Assignment** option is selected, the program will assign a value of 0 to row 1, a value of 1 to row 2, and so on. For this selection, the state assignment will not be displayed.

If the **Display or Change Current Assignment** option is selected, LogicAid will insert the current assignment before the present-state column of each row when the state assignment is not already displayed; if there is no current assignment, the default assignment is displayed. For State Graphs and SM Charts, a new state assignment window will be displayed, showing the state assignment corresponding to each state name, as shown in Figure 5.4. Whenever the state assignment is displayed, the **Tab** key may be used to select the assignment of the next row; therefore, you may easily update the state assignment.

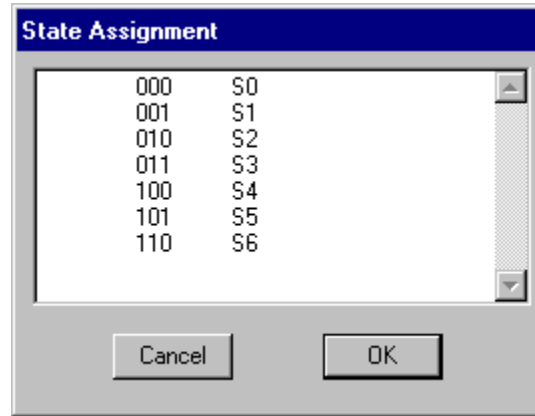
When the **Hide State Assignment** option is selected, the state assignment will be removed from the window. The previously entered assignment will be saved in memory. Note that this option is only activated for state tables, when the state assignment is already displayed in the edit window.

#### FORMAT FOR INPUT ASSIGNMENT:

This specifies the format for entering the state assignment. The buttons are not activated unless the **Display or Change Current Assignment** button is selected. You may choose one of the following options: **Decimal**, **Binary**, **Hexadecimal**, and **Octal**. The only difference between the description of this and any other format options is that the Binary format allows only 0 and 1 as legal characters. Refer to *Input Format* in Section 4.1, *Terms*, for more information on entering input.

In a state table, if the state assignment is currently displayed in the edit window and the button selection is changed, LogicAid will change the displayed state assignment

from one format to another. For example, if **Number of State Variables** is changed from 4 to 3, one of the state assignments is a decimal 8, and the input format is changed from **Decimal** to **Binary**, LogicAid will change this assignment from a decimal 8 to a binary 000. You must be aware that LogicAid does not check syntax while updating the state assignment.



**Figure 5.4** State assignment box for SM Charts and State Graphs.

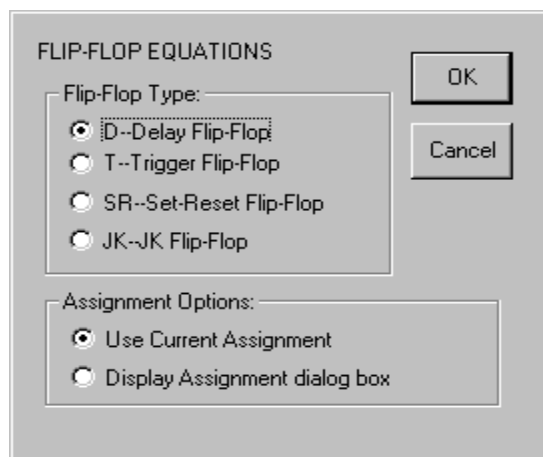
#### 5.4. Flip-Flop Equations (Alt+R F or Ctrl+F)

Function: Generates the specified flip-flop input functions.

This command first checks the syntax of the input state table. In addition, LogicAid checks for duplicate input-variable and state-variable names. If the state table is syntactically correct and there are no duplicate names, the Flip-Flop Equations Options dialog box shown in Figure 5.5 is displayed. Its options are described in the following subsections. After **OK** has been checked in this dialog box, the Simplification dialog box is displayed for specifying the simplification parameters. Refer to Section 5.1 for more information on the Simplification dialog box.

##### FLIP-FLOP TYPE:

There are four types of flip-flops to choose from. These are **Delay**, **Trigger**, **Set-Reset**, and **JK**. For any flip-flop type, the program will generate a number of output equations equal to the number of output variables in the state table. For the **Delay** or **Trigger** flip-flop, the program will generate a number of input equations equal to the number of state variables. For the **Set-Reset** or **JK** flip-flops, the program will generate a number of input equations equal to twice the number of state variables.



**Figure 5.5. Flip-Flop Equations Options dialog box**

#### ASSIGNMENT OPTIONS:

The default selection is **Use Current Assignment**. In this case, the current assignment of the state table is used to generate the flip-flop input equations. If you have not specified an assignment, the default binary assignment is used.

If the **Display Assignment dialog box** button is clicked, all the buttons of the **Flip-Flop Type** option are deactivated. If you click on **OK** in the dialog box with this selection, the State Assignment dialog box is displayed for specifying state assignments. You must select the **Flip-Flop Equation** command again if flip-flop equations are to be generated.

### 5.5. State Table Checker (Alt+R K or Ctrl+Shift+K)

Function: Checks (compares) the current state table against a “solution” state table file.

LogicAid compares the current state table (M1) with a “solution” state table (M2). The table M1 is considered to be “correct” if in an actual application, M1 could be substituted for M2, assuming that both M1 and M2 are started in the reset state which is the first row of the table.

If M1 and M2 are completely specified, the reset states are assumed to be equivalent. For the incompletely specified case, “correct” implies that starting in the first row of both tables and applying the same input sequence, the outputs will be the same whenever the M2 output is specified. In other words, it is acceptable for M1 to have a 0, 1, or don’t care output when M2 has a don’t care output, but it is not acceptable for M1 to have a don’t care output when M2 has a 0 or 1.

This command first checks the syntax of the current input state table (M1). If the state table is syntactically correct, the **Open** dialog box is displayed with the names of both the protected and unprotected LogicAid files. You should choose the name of the solution file.

After the solution-file selection, the program will first make sure that the test file and the solution file have the same number of columns, input variables, and output variables. The two tables also must be of the same machine type and headings. If any of the preceding parameters is different, an appropriate alert box will be displayed. It is not



necessary for the two state tables to have the headings in the same exact order; the program will sort the headings if necessary.

Next, the solution file is checked for syntax errors. Since the solution file is not shown on the screen, the exact error message is not given; instead, a general error message is displayed. If the solution file is syntactically correct, the two state tables are compared. If the test state table is correct, a “Correct” message is displayed. If it is not, an “Incorrect” message is displayed; furthermore, the shortest input sequence that causes an error will be shown on the screen. The sequence will always start in the first state of both tables. Note that there may be multiple shortest-input sequences, but only one sequence is displayed.

As stated above, you must have a current state table in the edit window and must have the solution file on a disk in order to use this routine. The routine assumes that the first state of the current table and the solution table is the starting state of both tables. As discussed in Section 4.4, *PLA Table*, each state table may have a maximum of 8 columns, 16 input variables, 16 output variables, and 32 rows.

You may abort the **State Table Checker** command during execution (after you click on **OK** in the dialog box) by simultaneously pressing down the **Control-Period (Ctrl+.)** keys.

## 5.6. Partial Graph Checker (Alt+R G or Ctrl+Shift+G)

**Function:** Compare the current partial state graph against a “solution” state table.

This command will check for syntax errors for a partial state graph. If there is no error, it will convert the graph to a state table internally and use a method similar to the one described in Section 5.5, *State Graph Checker*, to compare the converted state table with the selected solution file.

The rules for a partial state graph are a little different from those in Section 5.5. In Section 5.5, the unspecified data is supposed to mean “don’t care”; but for a partial state graph, the unspecified data is supposed to mean “unknown” (it is not input yet). The Partial Graph Checker does not check unknown data, it just ignores it.

LogicAid compares the partial graph (M1) with a solution state table (M2). The partial graph M1 is considered to be “consistent” if LogicAid finds it is possible for M1 to be “correct” compared with M2 after the graph for M1 is completely input. If we check the partial data of M1 and find that it is never possible for M1 to be “correct” compared with M2, we will say M1 is inconsistent with M2.

When we say “correct”, it means M1 could be substituted for M2 assuming that both M1 and M2 are started in the reset state, which is the first state for M1 and the first row for M2. If M1 and M2 are completely specified (no ‘-’/don’t care exists), the reset states are assumed to be equivalent. For the incompletely specified case (‘-’ exists), “correct” implies that starting in the first states and applying the same sequence, the output will be the same whenever the M2 output is specified. In other words, it is acceptable for M1 to have a 0, 1, or don’t care output when M2 has a don’t care output, but it is not acceptable for M1 to have a don’t care output when M2 has a 0 or 1. Comparison is only carried out until an undefined next state or arc label is encountered in M1.

This command first checks the syntax of the current input state graph (M1). If it is correct and you have not yet selected any solution file, the **Open** dialog box is displayed with the names of both the protected and unprotected LogicAid files. You could select the name of the solution file. If you selected a solution file before, the Open dialog box will not appear and LogicAid will use the old solution file directly.

After LogicAid gets the solution file name, it will first make sure that the solution file (M2) and M1 have the same machine type, the same number of input variables and output variables, and every column heading of M1 appears as a column heading of M2. If any of these conditions do not exist, an alert box will be displayed, giving you an error message.

Next, the solution file is checked for syntax errors. Since the solution file is not shown on the screen, the exact error message is not given; instead, a general error message will be displayed. If the solution is syntactically correct, M1 and M2 are compared. If we find they are consistent, a “consistent” message is displayed on the Output window. If they are inconsistent, an “inconsistent” message is displayed; furthermore, the shortest input sequence that causes an error will be shown on the screen. The sequence will always start in the first state of M1 and of M2. Note that there may be multiple shortest-input sequences, but only one sequence is displayed.

### 5.7. Select New Solution File (Alt+R N or Ctrl+Shift+N)

Function: Allows you to select a new solution file for the Partial Graph Checker

The first time you select Partial Graph Checker, LogicAid will display a standard file dialog box and you can select a solution file to compare. After that, LogicAid will keep using that solution file and not ask you again. To change the solution file, you must choose Select New Solution File, or close the graph window and start over again.

After you select this command, LogicAid will display the standard file dialog box and you can select a new solution file and click on **OK**. The next time you select Partial Graph Checker, LogicAid will compare the partial graph with the new solution file.

### 5.8. Make Jedec File (Alt+R J or Ctrl+J)

Function: Creates a Jedec file from logic equations (bit file for PROM programmer)

You can use LogicAid to generate a file that contains the fuse patterns in Jedec format from your logic equations. These fuse patterns are used to program a PAL. You can load this Jedec file into a PAL programmer to program the PAL. The present version of this software only allows you to program a 22V10 PAL.

To use this routine, first create a state table or open an existing state table. Then, select the Flip-Flop Equations routine from the Routine menu, and use D Flip-flops. In the simplification dialog box, use the default options. (Changing the simplification options may result in a wrong format for the equations and an incorrect Jedec file.)

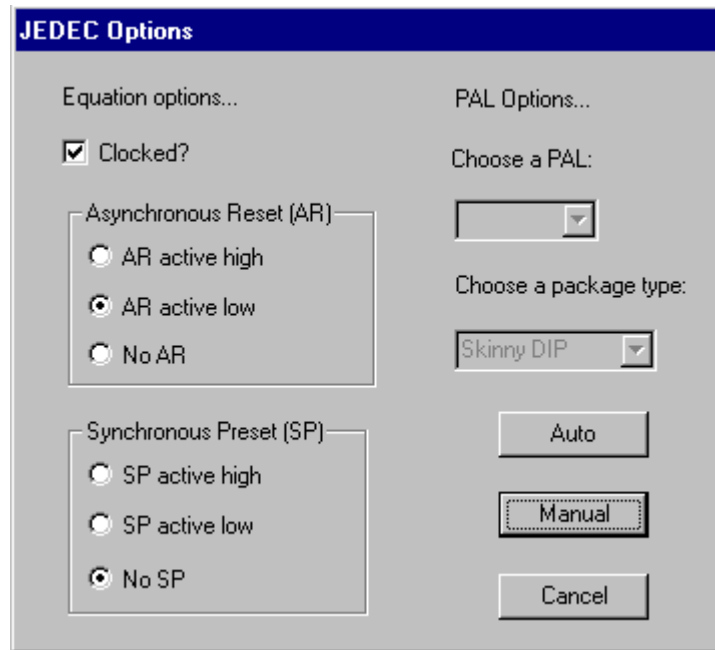
When the output window for the flip-flop equations is displayed, select the Make Jedec File routine. This brings up the JEDEC Options dialog box, as shown in Figure 5.6.

**EQUATIONS OPTIONS:**

The **Clocked?** checkbox is used for sequential networks involving flip-flops. For combinational networks, this checkbox should be unchecked. For the Asynchronous Reset (AR) and the Synchronous Preset (SP) option, you can select either an active low input or an active high input, or you could remove the AR and SP inputs altogether.

**PAL OPTIONS:**

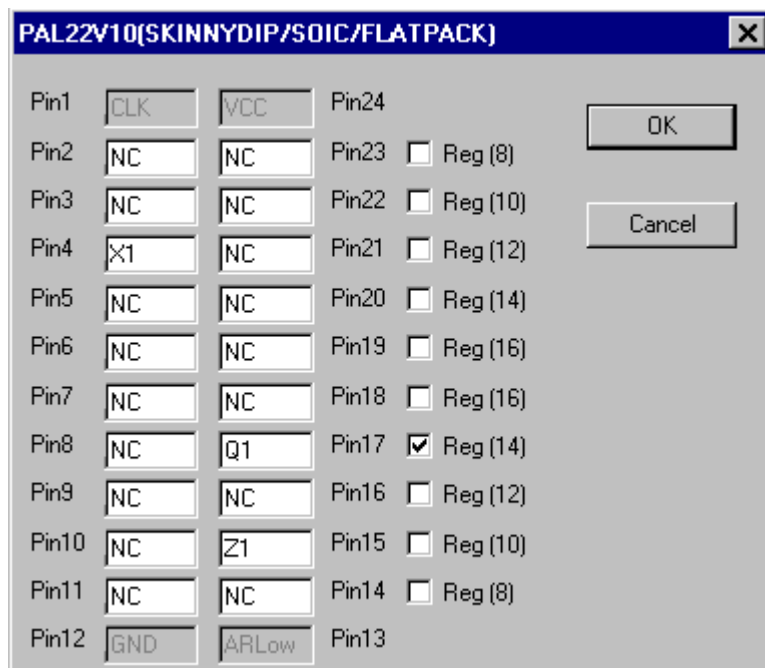
This version of LogicAid defaults to the 22V10 PAL with a Skinny DIP package type. You cannot change the PAL options.



**Figure 5.6. JEDEC Options dialog box**

**PIN ASSIGNMENT OPTIONS:**

You can select either the Manual or the Auto option for pin assignments. If Auto is selected, LogicAid will automatically assign the inputs and outputs to the pins on the PAL. To manually assign the pins for the inputs and outputs, select the Manual option. After the pin assignment option is selected, the pin assignment dialog box will appear, shown in Figure 5.7.



**Figure 5.7 Pin assignment dialog box**

In the pin assignment dialog box, note that CLOCK is assigned to pin 1, GND to pin 12, Vcc to pin 24, and ARLow(Asynchronous Reset, active Low) to pin 13. You cannot change these assignments. To enter the desired pin assignments, replace NC (no connection) with the appropriate values. Outputs and flip-flops can be entered only on the right side of the pin diagram. For flip-flops, you must click on the square next to reg. All unused pins must be assigned NC. When finished, click on **OK**.

The Jedec file (fuse patterns) for the PAL will appear in a new Output window. This file can be saved on disk by selecting Save As on the File menu. A saved Jedec file can be retrieved by selecting Open Text from the File menu and then selecting Jedec in the subsequent dialog box.

### 5.9. Download Jedec File (Alt+R D or Ctrl+D)

Function: Downloads a Jedec file to a PROM programmer via the serial port.

This routine will download the Jedec file onto a PAL programmer connected to the serial port of the computer. If the programmer you are using connects to a parallel port, use the software provided with the programmer to download the Jedec file that you have saved.

To download a Jedec file to a programmer using a serial port, proceed as follows:

1. Turn on the PAL programmer. Make sure that all the IC sockets on the programmer are empty.
2. Select the CE22V10 programmable logic device from the programmer menu.
3. Set the programmer to receive data.

4. Run LogicAid. To retrieve a saved Jedec file, select Open Text on the File menu. Then select Jedec and hit **OK**. Select the file in the file dialog and click Open. The fuse patterns will appear in a new window.
5. Select the Download Jedec File option from the Routine menu. This starts the communication software between the computer and the programmer. When the communication configuration box shown in Figure 5.8 appears, select the appropriate communication port, baud rate, parity, and data bits. Click **OK**.
6. The programmer should receive the data. When the transmission is complete, the Jedec file is downloaded to the programmer.
7. Insert the PAL in the programmer socket and complete the programming.

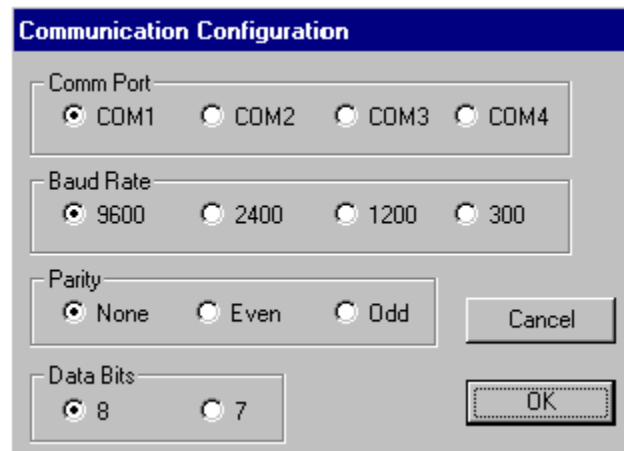
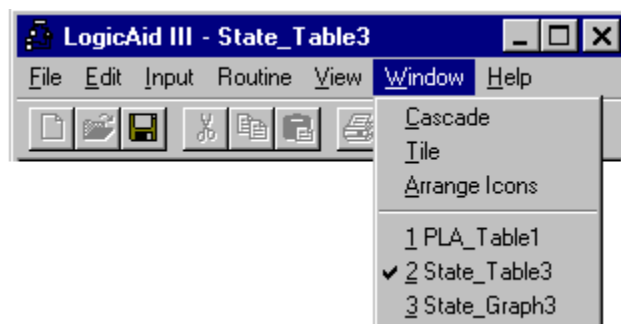


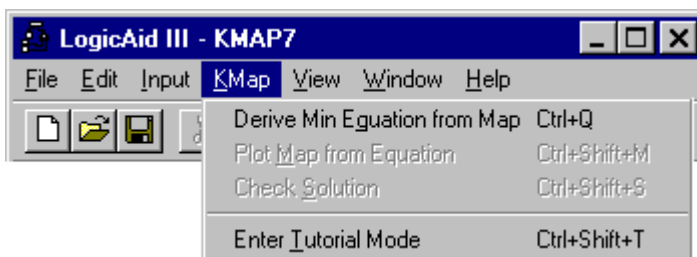
Figure 5.8 Communication Configuration dialog box.

## 6. Windows Menu



This menu lists all the available windows and allows you to select (activate) any available window on the screen by selecting the appropriate menu command. The active window is denoted with a check mark next to its number. It also allows you to display the windows in a cascaded format or tiled format. You can also choose to arrange the icons of the minimized windows.

## 7. The KMap Menu

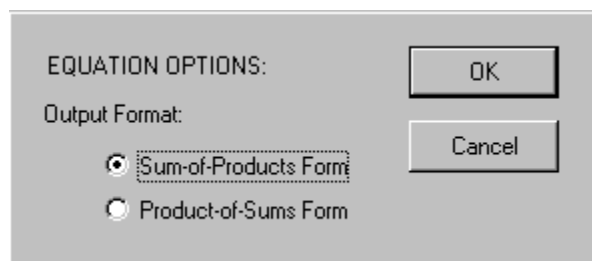


You can **Enter Tutorial Mode** on the **KMap** menu any time you are entering a Karnaugh map. The other items on this menu—**Derive Min Equation from Map**, **Plot Map from Equation**, and **Check Solution**—will be activated only when you have selected Karnaugh Map input.

### 7.1 Derive Min Equation from Map (Alt+K Q or Ctrl+Q)

**Function:** Derive the Boolean function from the Karnaugh Map and simplify it to get all the minimum equations.

This command will bring up a dialog box for specifying the output form of the minimum equations that could be **Sum of Product** or **Product of Sum** (see Figure 7.1).



**Figure 7.1** Equation Options dialog box

LogicAid uses Petrick's Method to get all the minimal solutions. Petrick's method finds all the minimum solutions based on minimizing the gate inputs. The output equations are listed in the equation zone of the Karnaugh map window, which is used to enter the equations of the maps.

### 7.2 Plot Map from Equation (Alt+K M or Ctrl +Shift+M)

**Function:** Derive the Boolean function 1-terms or 0-terms from the last equation in the text area and plot them in the map.

When you select **Plot Map from Equation**, an alert box appears, telling you this option will erase all the old entries of the map; if you click on **OK**, the function continues. LogicAid checks the syntax of the last equation; if there is any error, LogicAid displays the error message and then stops the function. If the last equation is expressed in the **SOP** form, the data filled in the map will all be 1-terms and the loops

will all be 1-term loops corresponding to the terms of the equation; likewise, if the last equation is expressed in the **POS** form, the data filled in the map will be 0-terms and the loops will all be the 0-term loops corresponding to the input terms of the equation.

### 7.3 Check Solution (Alt+K S or Ctrl+Shift+S)

Function: Compares the Karnaugh map with the last equation to see if they represent the same Boolean functions, and determines if the equation is the minimum form or not.

At first, LogicAid will check the syntax of the input map and the last equation. If there are 1-term loops in the map, LogicAid will assume the equation form will be **SOP**; if it is not, LogicAid will display a syntax error message for the equation. Likewise, if there are 0-term loops in the map, the equation form should be **POS**.

Next, LogicAid will compare the Boolean functions of the map and equation. If they do not match, an error message will be displayed.

The third step is to compare the loops of the map with the terms of the equation. If some loops of the map do not correspond to any term of the equation, LogicAid gives the message **The equation doesn't match the map**. But if some terms do not correspond to any loops of the map, no message is given.

The last step is to check if the input equation is minimum or not. If it is, LogicAid will display **Correct Solution**; otherwise, LogicAid will display **The solution is correct but not Minimum**.

### 7.4 Enter Tutorial Mode (Alt+K T or Ctrl+Shift+T)

Function: Allows you to learn how to enter a correct minimum equation for a Karnaugh map under the guidance of LogicAid.

After the KMap input format is selected, the Tutorial Options dialog box (Figure 7.2) is displayed. Its contents are described in the following subsections.

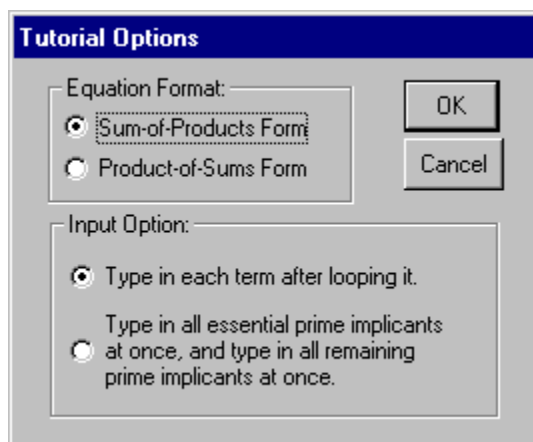


Figure 7.2 Tutorial Options dialog box

**EQUATION FORMAT:**

You select the form of the equation, which can be **Sum of Product** or **Product of Sum**. The default is **Sum of Product**.

**INPUT OPTION:**

There are two input options for you to enter the equations for the map. One is **Type in each term after looping it** and the other is **Type in all essential prime implicants at once, and type in all remaining prime implicants at once**. For the first option, after you loop an implicant in the map, you have to enter the corresponding term in the equation right away. For the second option, you must enter the corresponding terms of the equation after you finish looping all the essential prime implicants in the map. Then you must enter the remaining terms of the equation after you have looped all of them on the map.

In the tutorial mode, before you finish entering the equation for a Karnaugh map, there is always an instruction on the top of the graph window. You must follow the instruction to enter the data. If the instruction is not finished, LogicAid will not display the next instruction. With the different Input Options, the instructions will be different. When a Karnaugh map window is printed in the tutorial mode, KMT will be printed at the top of the listing.

**INSTRUCTIONS :**

For the input option of **Type in each term after looping it**, the instructions are as follows:

- 1. Enter a new map! (Hit F12 when finished)**

LogicAid asks you to plot a Boolean function in the map; and then hit **F12**. In the tutorial mode, you cannot modify the map entries once the map has been completely entered.

- 2. Loop an essential prime implicant! (Hit Enter) (Hit F12 when finished)**

LogicAid asks you to loop an essential prime implicant, and then hit Enter. When you hit Enter, LogicAid will check whether the looped squares form an essential prime implicant; if they do, LogicAid will display the loop and go to the next step; otherwise, it will stay in this step. If you cannot find any other essential prime implicants, you should hit **F12**. When you hit **F12** here, LogicAid will check if there are any essential prime implicants not looped yet; if so, LogicAid will keep asking you to loop essential prime implicants; otherwise it will enter step 4.

- 3. Type in the selected term! (Hit Enter)**

You should enter the corresponding term in the equation and hit Enter. If the terms of the equation do not correspond to the loops of the map, LogicAid will stay here and ask you to modify the equation; otherwise, LogicAid will go back to instruction 2 and wait for another looping.

You must hit **F12** to go to the next step.



4. **Loop a remaining term for minimum solution! (Hit Enter) (Hit F12 when finished)**

This step is very similar to step 2 and the only difference is now you should try to loop one of the remaining prime implicants that can make a minimum equation with the least number of gates. If you loop an incorrect prime implicant, an error message will be displayed; otherwise, LogicAid will go to step 5. If the loops have covered all the 1-terms (or 0-terms), you should hit **F12** and LogicAid will check if this is a minimum solution.

5. **Type in the selected term! (Hit Enter)**

This instruction is the same as step 3. After a term is entered, LogicAid returns to step 4.

For the input option of **Type in all essential prime implicants at once, and type in all remaining prime implicants at once**, the instructions are as follows:

1. **Enter the map! (Hit F12 when finished) (Same as previous step 1.)**
2. **Loop an essential prime implicant! (Hit Enter) (Hit F12 when finished)**  
Same as previous step 2, except LogicAid will not go on to instruction 3 until all essential prime implicants have been looped.
3. **Type in all selected terms! (Hit F12 when finished)**  
You should enter the corresponding terms in the equation now. LogicAid will keep checking the input terms for you. Once LogicAid finds one term not corresponding to any loops, an error message will be displayed. You should erase the term and type in the correct term. Once all of the corresponding terms have been entered, hit **F12** and LogicAid will go to step 4.
4. **Loop a remaining term for minimum solution! (Hit Enter) (Hit F12 when finished)**  
This instruction is very similar to step 2 and the only difference is now you should try to loop the remaining prime implicants that can make a minimum equation with the least number of gates. If you loop an incorrect prime implicant, an error message will be displayed; otherwise, LogicAid will display the loop and you can loop another prime implicant. If the loops have covered all the 1-terms (or 0-terms), you should hit **F12** and LogicAid will check if all the necessary loops are covered; if so, LogicAid will go to step 5, otherwise it will keep asking you to loop remaining terms.

5. **Type in all selected terms! (Hit F12 when finished)**

Same as step 3. After typing in all of the terms, hit **F12** and LogicAid will check to see if the solution is minimum.

After you enter a correct equation for the Karnaugh map, select **Exit Tutorial Mode** from the KMap Menu to leave the tutorial mode or select **Clear** from the left side menu to clear the map or equations and start again.

## 8. The Help Menu



The Help menu provides access to the online help file and version information on the LogicAid software.

### 8.1 LogicAid Help

Function: Opens the Help window.

This command brings up the help index, which contains information on how to enter the different input documents, and also an explanation of the menus available in the LogicAid program.

### 8.2 About LogicAid (Alt+H A)

Function: Provides copyright notice and version information.

This command brings up a dialog box that displays the copyright notice and version information of your copy of LogicAid.

# LogicAid Index

---

- About LogicAid, 86
- asynchronous design, 32, 33
- built-in help, 1
- Change Parameters command, 41, 53, 69
- Change Reset State, 69
- Check Solution command, 10, 83
- Check Syntax command, 3, 55, 69
- Close command, 4, 37
- column dominance procedure, 71
- comments, 47
- consistent partial state graphs, 77
- Convert to PLA Table command, 63, 69
- Convert to State Table, 20, 31, 40, 58, 59, 60
  - command, 68
- Copy command, 40
- Cut command, 39
- Default State Names option, 57
- Derive Min Equation from Map command, 10, 82
- design examples, 27, 28, 29, 31
- diagonal form, 8, 65
- dice game controller example, 29, 31
- Download Jedec File command, 24, 80
- enter names
  - option, 42
  - rules, 42
- Enter Tutorial Mode command, 8, 11, 82, 83
- Equation Format option, 46
- Equation Options dialog box, 82
- equations
  - entering, 2
  - syntax, 46, 47
- Equations command, 2, 46
  - Enter Names option, 2
- Espresso II routine, 72
- Exit command, 39
- Exit Tutorial Mode command, 12, 86
- File Menu, 36
- file names, 3
- Flip-Flop Equations command, 16, 75
  - aborting, 16
  - assignment options, 16
  - Display All Prime Implicants option, 33
  - Display Assignment Dialog Box option, 16
- hazards, 32
- Help Menu, 86
- Input Equations Format dialog box, 2, 46
- Input Karnaugh Map
  - dialog box, 64
  - Format dialog box, 8, 64
- Input Menu, 40
- Input SM Chart dialog box, 61
- Input State Graph dialog box, 31, 56
- Input State Table Format dialog box, 52
- Input Terms Format dialog box, 41
- Input Truth Table Format dialog box, 4, 6, 48
- Input Variable Names dialog box, 2, 4, 6, 7, 8, 14, 19, 42, 43
- Installation, iv
- iterated consensus method, 3, 72
- Jedec file, making, 24
- Karnaugh map
  - command, 8, 64
  - Enter Names option, 8, 64
  - side menu, 8, 9, 65
  - window, 8
- Karnaugh maps
  - clearing, 9, 66
  - Correspondence between equations and loops, 66
  - entering, 8
  - equations for, 9, 66
  - form of 5-variable map, 8, 65
  - Rules for plotting maps, 66
  - tutorial instructions, 85, 86
  - tutorial mode, 11
  - tutorial options, 11, 85
  - Use Current Names option, 65
  - Use Default Names option, 64
- KMap menu, 9, 10, 82
- Length of State Names text box, 53
- LogicAid Help command, 86
- LogicAid Menus & Windows, 35
- Machine type, 53
- Make Jedec File
  - command, 24, 78
  - Equations options, 79
  - PAL options, 79
- Mealy network example, 22
- minterm and maxterm expansions, 6
- mirror image form, 8, 65
- New command, 36
- New dialog box, 37
- Number of Functions text box, 41

- Number of Input Variables text box, 52, 56
- Number of Next State Columns text box, 52
- Number of Output Variables text box, 52, 56
- Number of State Variables, changing, 75
- Number of Variables text box, 41
  
- Open command, 37
- Open Text command, 37
  
- Partial Graph Checker command, 21, 22, 77, 78
- partial state graphs
  - checking, 21
  - rechecking, 21
- password, 37
- Password dialog box, 37
- Paste command, 40
- personalizing LogicAid, 1
- Petrick routine, 3, 71, 82
- PI Chart routine, 3, 71, 72
- Pin assignment dialog box, 80
- Pin assignment options, 79
- PLA table, 29, 31
  - command, 51
  - entering, 6
  - syntax, 51
- Plot Map from Equation command, 9, 82, 83
- prime implicants, 33
- Print All command, 4, 16, 38
- Print command, 4, 38
- Print Preview command, 38
- Print Setup command, 39
- programmable logic device, 30
- programming a PAL, 23, 80
- protected file, 38
  
- Quine-McCluskey method, 3
  
- Routine Menu, 70
  
- Save As command, 38
- Save command, 38
- Select New Solution File command, 78
- separators, 2, 47
- sequence detector design, 27, 28
- simplification
  - routine, 25
  - routines, 3, 71
- Simplification command, 3, 29, 32, 70, 71, 72
  - aborting, 4, 70
- simplification options
  - Complemented Functions, 72
  - Display All Prime Implicants, 72
  - Display Input and Gate Costs, 4, 72
  - Identify Essential Prime Implicants, 4, 72
  - Overwrite the Output Window, 72
  - Petrick, 28
  - Petrick—Maximum of 5 Minimal Solutions, 3
- SM chart, 29
  - arc labels, 26, 62
  - arcs, 26, 62
  - command, 25
  - comments, 26, 62
  - condition box, 26, 62
  - Convert to PLA table command, 27
  - Default State Names option, 61
  - deleting elements, 27
  - editing, 26, 62
  - Enter Names option, 25
  - entering, 25
  - input and output variable names, 61
  - input format, 25
  - label format, 61
  - moving elements, 26, 62
  - new state, 26
  - output box, 26, 62
  - syntax, 61
- SM Chart command, 61
- Solution Files, iv
- State assignment
  - adjacency guidelines, 28
  - command, 15, 31, 53, 73
  - Display or Change Current Assignment, 28, 74
  - Display or Change Current State Assignment, 15
  - Enter Names option, 15
  - options dialog box, 74
  - State Variable Names, 73
  - Use Default Straight-Binary Assignment, 74
- State assignment box for SM Charts and State Graphs, 75
- State Graph command, 19, 55
  - Enter Names option, 19, 20
  - Use Default State Names option, 19
- state graphs, 18, 19
  - alphanumerics inputs, 20
  - arc labels, 19, 21, 57
  - arcs, 19, 57
  - Change Reset State option, 20
  - comments, 20, 58
  - converting to state table, 20
  - converting to state tables, 23
  - Default State Names Options, 57
  - deleting elements, 20, 58, 63
  - editing, 20, 58
  - entering, 18
  - Input Format, 57
  - machine type, 19, 57
  - moving elements, 20, 58
  - side menu, 19, 57
  - SYNTAX, 57
- State Reduction command, 15, 27, 72, 73
  - aborting, 73
  - Edit Window option, 15, 27

- State Reduction Option dialog box, 15, 73
- State Table Checker, 76
- State Table Checker command, 17, 77
  - aborting, 77
  - encoded solution files, 17
- State Table command, 13, 51
  - Enter Heading option, 14
  - Enter Names option, 14
- State Table Reduction, 15
- state tables, 14, 15, 17
  - auto-formatter, 53
  - column heading, 53
  - column headings, 14, 68
  - Column headings, 53
  - entering, 13
  - functional equivalence, 17
  - input format, 13
  - Input/Output Variable Names, 53
  - Machine Type, 53
  - syntax, 14, 54
- State Variable Names dialog box, 73
- straight binary order, 5
  - button, 48

- terms
  - input format, 7, 43
  - syntax, 7, 43, 44, 45
- Terms command, 7, 40
  - Enter Names option, 7
- truth table
  - entering, 4
  - Format for Input Combinations, 48
  - Format For Output Combinations, 49
  - Output Value for Remaining Rows, 49
  - syntax, 49
- Truth Table command, 4, 6, 48
  - Enter Names option, 4, 6
- Tutorial Options dialog box, 83
- Undo command, 39
- Use Current Names option, 42
- Use Default Names option, 41
- Variable and Function Names, 41
- windows, 1
- Windows Menu, 81