11.2 GALS AND PALS

Among the most basic types of PLDs are Generic Array Logic™ (GAL) devices.* GALs are enhanced variants of the older Programmable Array Logic™ (PAL) architecture that is now essentially obsolete. The term PAL is still widely used, but people are usually referring to GAL devices or other PLD variants when they use the term. PALs became obsolete, because GALs provide a superset of their functionality and can therefore perform all of the functions that PALs did. GALs are relatively small, inexpensive, easily available, and manufactured by a variety of vendors (e.g., Cypress, Lattice, and Texas Instruments).

It can be shown through Boolean algebra that any logical expression can be represented as an arbitrarily complex sum of products. Therefore, by providing a programmable array of AND/OR gates, logic can be customized to fit a particular application. GAL devices provide an extensive programmable array of wide AND gates, as shown in Fig. 11.2, into which all the device's input terms are fed. Both true and inverted versions of each input are made available to each AND gate. The outputs of groups of AND gates (products) feed into separate OR gates (sums) to generate user-defined Boolean expressions.

Each intersection of a horizontal AND gate line and a vertical input term is a programmable connection. In the early days of PLDs, these connections were made by fuses that would literally have to be blown with a high voltage to configure the device. Fuse-based devices were not reprogrammable;

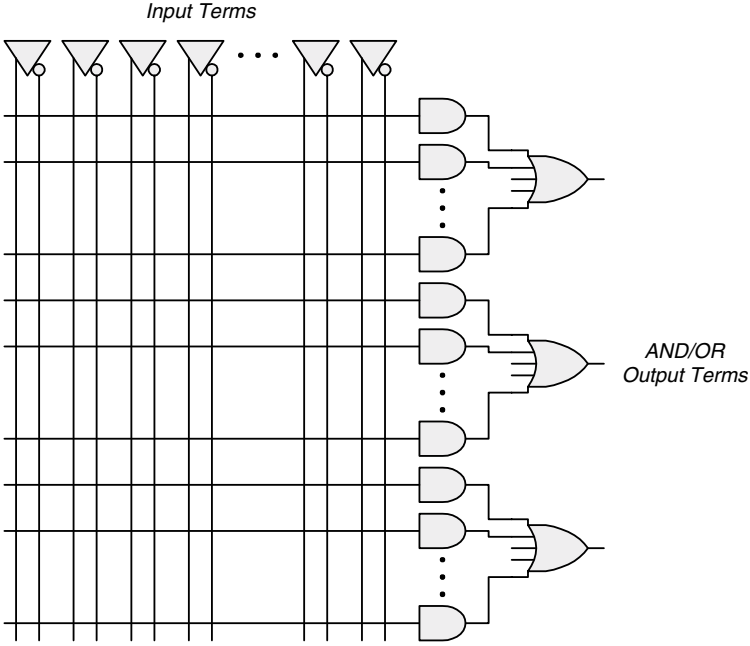

FIGURE 11.2 GAL/PAL AND/OR structure.

* GAL, Generic Array Logic, PAL, and Programmable Array Logic are trademarks of Lattice Semiconductor Corporation.

once a microscopic fuse is blown, it cannot be restored. Today's devices typically rely on EEPROM technology and CMOS switches to enable nonvolatile reprogrammability. However, fuse-based terminology remains in use for historical reasons. The default configuration of a connection emulates an intact fuse, thereby connecting the input term to the AND gate input. When the connection is blown, or programmed, the AND input is disconnected from the input term and pulled high to effectively remove that input from the Boolean expression. Customization of a GAL's programmable AND gate is conceptually illustrated in Fig. 11.3.

With full programmability of the AND array, the OR connections can be hard wired. Each GAL device feeds a differing number of AND terms into the OR gates. If one or more of these AND terms are not needed by a particular Boolean expression, those unneeded AND gates can be effectively disabled by forcing their outputs to 0. This is done by leaving an unneeded AND gate's inputs unprogrammed. Remember that inputs to the AND array are provided in both true and complement versions. When all AND connections are left intact, multiple expressions of the form $A\&\overline{A} = 0$ result, thereby forcing that gate's output to 0 and rendering it nonparticipatory in the OR function.

The majority of a GAL's logic customization is performed by programming the AND array. However, selecting flip-flops, OR/NOR polarities, and input/output configurations is performed by programming a configurable I/O and feedback structure called a *macrocell*. The basic concept behind a macrocell is to ultimately determine how the AND/OR Boolean expression is handled and how the macrocell's associated I/O pin operates. A schematic view of a GAL macrocell is shown in Fig. 11.4, although some GALs may contain macrocells with slightly different capabilities. Multiplexers determine the polarity of the final OR/NOR term, regardless of whether the term is registered and whether the feedback signal is taken directly at the flop's output or at the pin. Configuring the macrocell's output enable determines how the pin behaves.

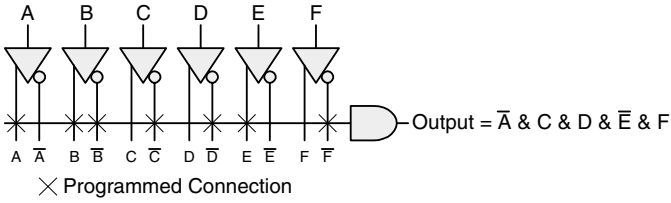

FIGURE 11.3 Programming AND input terms.

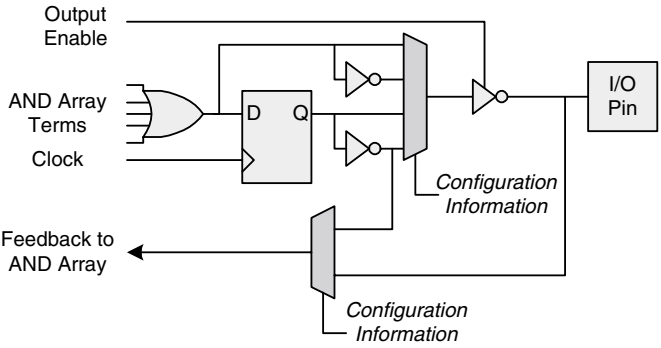

FIGURE 11.4 GAL macrocell.