Unit 21 Programmable Logic Devices (PLD)

- A PLD is a universal Sum of Products circuit in which input Product and output Sum combinations are selected by either fusing links or by activating memory transistor links within the integrated circuit.
- This customization gives an application specific integrated circuit (ASIC) at low cost.
- In bipolar technology devices the fuses are metal links which can be melted by application of a current.
- In CMOS devices the links are transistors which are open circuit when supplied by the manufacturer and can be closed during the programming.
- The CMOS devices are Electrically Erasable and can be reprogrammed.
- **JEDEC** stands for Joint Electronic Devices Engineering Council which sets standards in the industry.
- The **PLD**—Programmable Logic Device family includes:
  - **PROM**—Programmable Read Only Memory, AND inputs fixed, OR inputs programmable .
  - **PAL**—Programmable Array Logic, AND inputs programmable, OR inputs fixed.
  - **PLA**—Programmable Logic Arrays, both AND and OR gates programmable.
  - **GAL**—Gate Array Logic.
  - **FPLA**—Field Programmable Logic Array, the fuses can be blown by the user instead of being blown at time of manufacture.

In Units 9 to 20 we have examined how Boolean logic systems are specified, how the specification can be converted into arrays of fundamental logic AND, OR, XOR gates and how the complexity of the logic arrays can either

be minimized so as to reduce the gate count or can be structured to permit feasible testing of the logic structure. This is the theory of combinatorial logic systems. Now we come to some of the practicalities. How can such logic systems be implemented at a reasonable cost so as to customize application specific integrated circuits (ASICs) given that it is no longer a feasible proposition to wire up large quantities of individual AND, OR, XOR logic gates on printed circuit boards?

The approach that has been taken by the industry is to develop generalized integrated circuit chips which carry many logic gates and to customize these chips for specific applications by superimposing an interconnecting layer onto the generalized array of logic gates. A number of different structures have been used by different chip manufacturers for the underlying logic gate array which represent different design philosophies. We will now examine the main families of PLDs (Programmable Logic Devices) used by various manufacturers.

Consider a truth table which has n inputs. There will be 2^n rows in the truth table. Such a truth table could be implemented by using a PROM (Programmable Read Only Memory) having n address lines. Many PROMs are fabricated to give a 8 bit output byte, so use of a PROM to directly implement a truth table will typically give the option of 8 distinct outputs per ROM chip used. This approach is therefore the simplest and most direct method of implementing a truth table and is frequently used in small system.

However, there are a number of snags associated with this approach. The number of address lines on typical PROMs is of the order of 16, which restricts us to not more than 16 input variables which is equivalent to a $2^{16} = 65536$ row truth table recorded in a 1600 page notebook. This small number of inputs restricts the number of applications which can be accommodated and the large mass of documentation associated with the full enumeration of even a 16 input truth table is a further restriction. Another difficulty is that all of the rows of the truth table have to be programmed into the PROM so that the initial hand typing of 65536 rows into the PROM Programmer can be tedious even if many of the rows are at the default logic 1 output and therefore do not require typing.

Significant simplifications can be obtained by using PLDs (Programmable Logic Devices) which are arrays of AND gates each having $2n$ inputs with the outputs of the AND gates connected to give p product terms. The number of AND gates is less than 2^n but in any truth table most of the combinations will give a logic 0 output so a significant saving can be made by only considering the logic 1 output rows, that is the combinations specified by the minterm list. This means that the number of inputs to the PLD can be increased to much more than the 16 or so inputs associated with PROMs. Since the

number of terms in the minterm list is usually much less than than 2^n, we then have $p \ll 2^n$.

An elementary cell of such a system is shown in Figure 21.1. This is used to implement one term from the canonical minterm list or one row of the truth table for which the output is a logic 1. The particular row which is implemented is determined by blowing some of the fuses in the input lines to the AND gate after the integrated circuit has been manufactured.

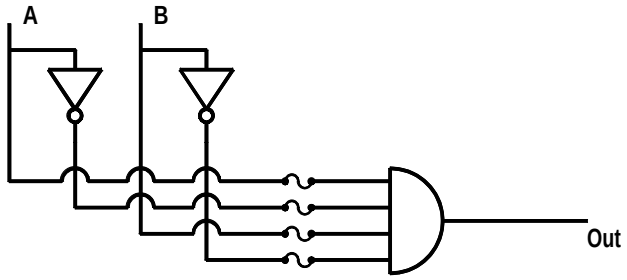

Figure 21.1: A 2 input system using a $2 \times 2 = 4$ input AND gate with all possible input signals connected through fuses.

For example, the circuit shown in Figure 21.2 has had two of the links blown and the $\overline{A}$ and B inputs are now connected to the AND gate so that the circuit has an output $Q = \overline{A}.B$. It should be noted that the unconnected (blown fuse) inputs to the AND gate must default to a logic 1.

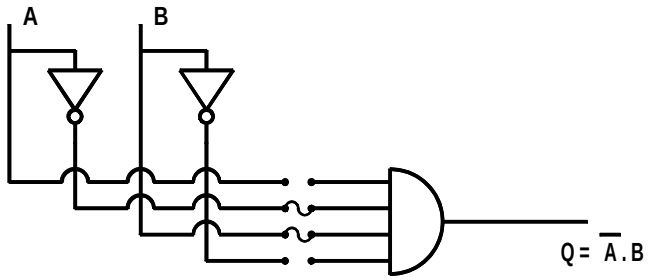

Figure 21.2: The same AND gate with two of the fuses burned out giving a logical output of $Q = \overline{A}.B$. The unconnected inputs to the AND gate default to logic 1.

This circuit can only implement one line of the truth table or one Boolean Product term. A more general circuit is shown in Figure 21.3 in which all four possible combinations of the two input truth table are available.

In this case, we select the Product terms by blowing the fuses at the inputs to the AND gates and we select the Sum terms by blowing the fuses at the inputs to the OR gates. Note that the blown or unconnected inputs to the OR gates must default to logic 0.

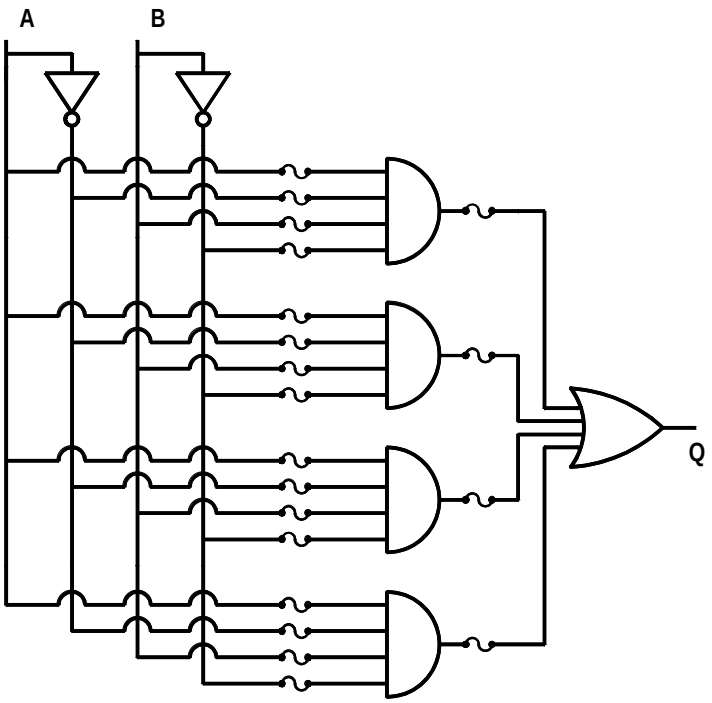

Figure 21.3: The structure which allows the implementation of any or all of the four canonical terms of the SOP formed from two inputs, A and B.

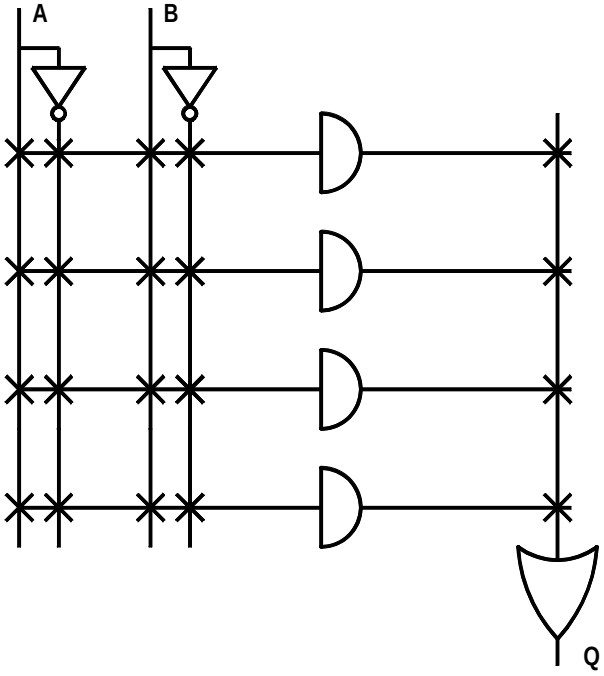

Figure 21.4: The same structure as that in Figure 21.3 but with the fusible links shown as $\times$.

The structure in Figure 21.3 in which we show all $2n$ lines to each of the AND gates is too complex for any large system, so we simplify the circuit diagrams as shown in Figure 21.4 in which the $2n$ lines are replaced by a bus and the fused connections are in each of the bus lines. We also show the fusible interconnections as $\times$ on the diagram. The circuits of Figures 21.3 and 21.4 are therefore equivalent.

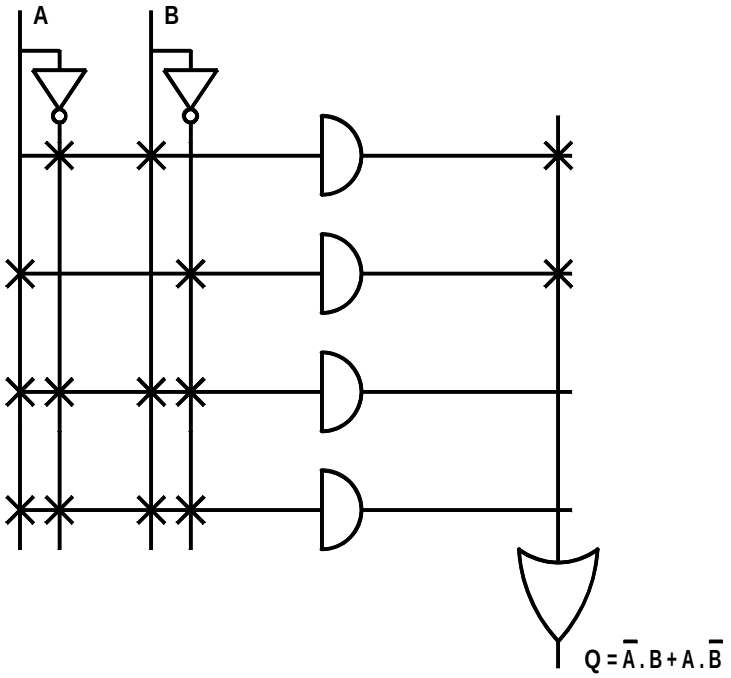

Figure 21.5: The reduced schematic for the XOR function $Q = \overline{A}.B + A.\overline{B}$. In this bipolar technology circuit, the intact fuses are shown with an $\times$.

The circuit which implements the logic expression $Q = \overline{A}.B + A.\overline{B}$ for the XOR function is shown in Figure 21.5. Note that the lower two AND gates are not connected to the OR gate input and therefore it was not necessary to blow the fuses on these inputs.

The 2 input system we have just described uses four 4 input AND gates. If we had a sixteen input system we would need 65 thousand 32 input AND gates. Therefore some compromises have to be made in the construction of larger systems. In any realistic combinatorial logic system it is usually possible to effect some significant simplification of the canonical SOP expression by use of Karnaugh map or other techniques so that the Boolean expression can be simplified. Therefore it is not necessary to be able to implement all of the canonical terms in the circuit. This then leads to a split in the approach to implementation of PLDs.

In the Programmable Array Logic (PAL) type devices the inputs to the AND gates are selectable by fuses but the OR gates are connected to fixed combinations of AND gate outputs. This configuration is shown in Figure 21.6. This PAL configuration is the most common type of PLD in use. Note that in this diagram we have used a different configuration for the input symbol which gives the A and $\overline{A}$ lines and also we use the CMOS type device where the links to the AND gate inputs are initially open and are formed in the programming process. This follows the convention of Advanced Micro Devices (AMD) and other manufacturers.

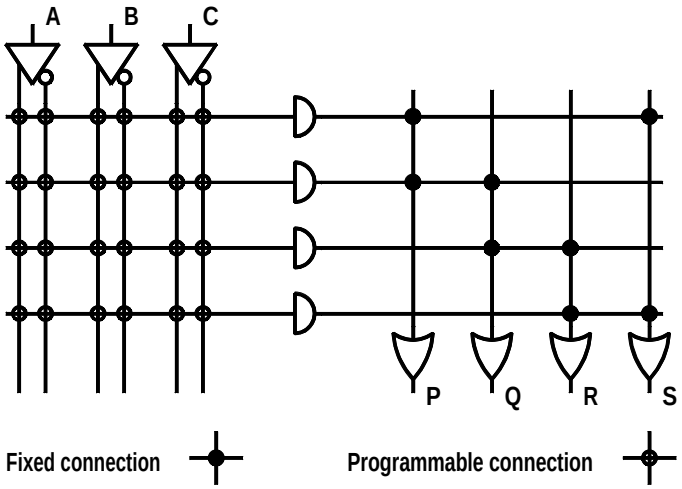

Figure 21.6: A Programmable Array Logic (PAL) configuration. These devices utilize CMOS technology and the links at the grid crossings are only formed when the device is programmed. The programmable links are shown with a circle. The manufactured-in fixed links are shown with a filled circle.

Another form of PLD, the Programmable Logic Array (PLA), is shown in Figure 21.7. In this configuration the inputs to both the p AND gates and the m OR gates are programmable but there are not enough AND gates for all of the canonical terms so again some simplification of the canonical form of the SOP expression will be necessary. This PLA type of device tends to be more expensive and therefore the usage is not as great as that of the PAL.

The restriction on the programmability in the PAL and PLA gives a large reduction of the number of logic gates on the IC, of the programmable links required and of the necessary drive capacity of the output of the AND gates. The resulting saving in the number of gates and in cost fully compensates for the restrictions on the programmability. However it should not be forgotten that PALs and PLAs are essentially the same—a hardware implementation of the Boolean Sum of Products expression.

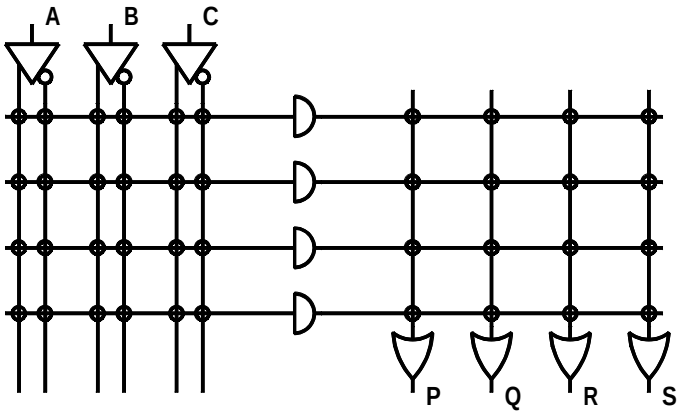

Figure 21.7: The Programmable Logic Array (PLA) configuration.

While the PAL and PLA devices consist of AND-OR combinations, a more recent introduction is the Programmable Gate Array (PGA) which consist of an array of logic blocks containing a significant degree of functionality hardwired into the blocks which can then be interconnected with Field Programmable fusible links.

Some PLDs are manufactured using bipolar technology and these devices utilize fusible links which are blown by passing a fusing current through the link for a short time. The fusing of these type of devices is permanent. In the case of PLDs manufactured using CMOS technology, the links are setup using memory cells and the devices can therefore, be reprogrammed by wiping the memory and starting the programming again.

In general, before any PLD device can be used, it is necessary to have access to design software and to a programmer. The design software is used to convert the specification in the form of a Boolean logic expression or truth table into a JEDEC file which is then passed to the device programmer. combination of fused links. The programmer is the device which applies the necessary currents to the pins of the PLD to burn out the fuses or program the memory links. Some of the design software also has facilities for full simulation of the design which allows debugging before actually programming a device. The design software may also give a high level design file which serves as design documentation. Web sites where software can be obtained are given at the end of this unit. Two pieces of software of note are PALASM2 and ABEL.

These devices are examples of ASICs (Application Specific Integrated Circuit) and Figure 21.9 shows the approximate relation between the number of inputs and the relative internal gate capacity for the different types of ASIC.

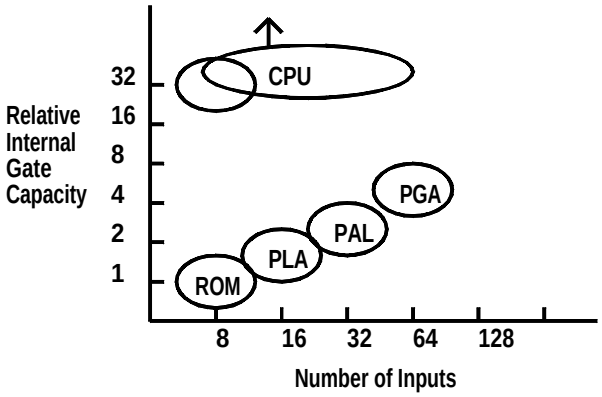

Figure 21.8: ASIC device complexity.

21.1 Web sites for further information

`www.viewlogic.com` Viewlogic Corporation *Vendor of design software packages*

`www.vantis.com/literature/` Advanced Micro Devices *Many excellent pages of explanation and product data from a leading manufacturer*

`www.vantis.com/software/swtools/universaltoolsc.html` Advanced Micro Devices *A source site for the ABEL (Advanced Boolean Expression Language) compiler for use with all types of PLDs.*

`www.actel.com` Actel

`www.` Company *A source site for the PALASM2 PAL assembler for generating a fuse map from a JEDEC format file.*

21.2 References

Herbst, L.J. (1996)
Integrated Circuit Engineering Establishing a Foundation
Oxford University Press

21.3 Problems

21.1 Mark in, on the generic PAL device shown below, the connections to the AND gate inputs which have to be programmed in so as to implement the following functions:

(a) $P = \overline{A}.B.\overline{C} + A.\overline{B}.C$

(b) $Q = A.\overline{C} + \overline{A}.c$

(c) $R = \overline{B}$

(d) $S = \overline{A}.B + C$

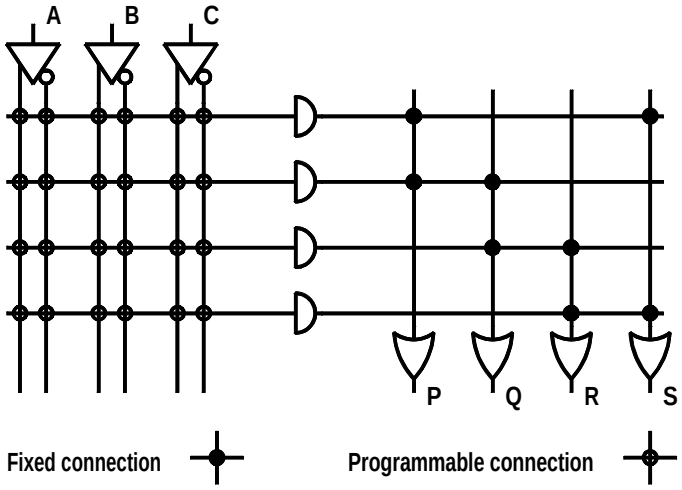

Figure 21.9: Problem 20.1. The generic PAL configuration.