CHAPTER 12

PROGRAMMABLE LOGIC DEVICES

12.1 INTRODUCTION

The combinational and sequential digital circuits have been discussed in earlier chapters. Various ICs for performing basic digital operations and other functions, such as multiplexers, demultiplexers, adders, comparators, code converters, shift registers and counters, etc. have also been discussed. These ICs are referred to as *fixed-function* ICs, i.e. each one of them performs a specific , fixed function. These devices are designed by their manufacturers and are manufactured in large quantities to meet the needs of a wide variety of applications and are readily available.

To design a circuit, a designer can select from the available ICs most appropriate for the circuit, usually working from a block diagram design concept. The design may have to be modified to meet the special requirements of these devices. The advantages of this method are:

1. Low development cost,
2. Fast turn around of designs, and
3. Relatively easy to test the circuits

Some of the disadvantages of this method are:

1. Large board space requirements,
2. Large power requirements,
3. Lack of security, i.e. the circuits can be copied by others, and
4. Additional cost, space, power requirements, etc. required to modify the design or to introduce more features.

To overcome the disadvantages of designs using fixed-function ICs, *application specific integrated circuits* (ASICs) have been developed. The ASICs are designed by the users to meet the specific requirements of a circuit and are produced by an IC manufacturer (*foundry*) as per the specifications supplied by the user. Usually, the designs are too complex to be implemented using fixed-function ICs.

The advantages of this method are:

1. Reduced space requirement,
2. Reduced power requirement,

3. If produced in large volumes, the cost is considerably reduced,
4. Large reduction in size through the use of high level of integration, and
5. Designs implemented in this form are almost impossible to copy.

The disadvantages of this method are:

1. initial development cost may be enormous, and
2. testing methods may have to be developed which may also increase the cost and effort.

Another approach which has the advantages of both the above methods is the use of *programmable logic devices* (PLDs). A programmable logic device is an IC that is user configurable and is capable of implementing logic functions. It is a VLSI chip that contains a 'regular' structure and allows the designer to customize it for any specific application, i.e. it is programmed by the user to perform a function required for his application.

PLDs have the following advantages of fixed function ICs:

(i) Short design cycle
(ii) Low development cost

The advantages over fixed-function ICs are:

(i) Reduction in board space requirements
(ii) Reduction in power requirements
(iii) Design security
(iv) Compact circuitry
(v) Higher switching speed

PLDs have many of the advantages of ASICs as given below:

(i) Higher densities
(ii) Lower quantity production costs
(iii) Design security
(iv) Reduced power requirement
(v) Reduced space requirement

The PLDs allow designers more flexibilities to experiment with designs because these can be reprogrammed in seconds. The design deficiencies and modifications etc. can be carried out in short time thereby reducing the possibility of huge cost over-runs.

PLDs are also useful for prototyping ASIC designs since foundry produced ASICs may require months of costly development.

Because of various advantages of PLDs mentioned above, a large number of PLDs have been produced by IC manufacturers with variety of flexibilities and options available for a circuit designer and have become very popular. The architecture and various other features of PLDs such as ROMs, programmable logic arrays (PLAs), programmable array logic (PAL), simple programmable logic devices (SPLDs), complex programmable logic devices (CPLDs), and field-programmable gate arrays (FPGA) have been discussed in this chapter. The use of these devices require changes in the traditional design methods, although the basic concepts remain the same.

12.2 ROM AS A PLD

Read-only memories (ROMs) have been discussed in Section 11.5. A read-only memory is basically a combinational circuit and can be used to implement a logic function. A ROM of size $M \times N$ has M number

of locations and N number of bits can be stored at each location. The number of address inputs is P, where $2^P = M$ and the number of data output lines is N. It can also be considered as a logic device with P inputs and N outputs as shown in Fig. 12.1.

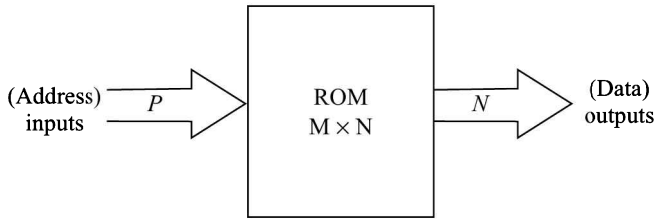


Fig. 12.1 ***ROM as a Combinational Circuit***

The 16-bit ROM array shown in Fig. 11.8 has four inputs and one output, i.e. $M = 16$, $N = 1$ and $P = 4$. The bit pattern stored, as given in Table 11.4, can be considered as truth table with $A_3 A_2 A_1 A_0$ as 4-bit input and Y as the output which is same as the bit stored. The logic function corresponding to this is

$$Y = \Sigma m(0, 6, 9, 12, 13, 15)$$

In general, a P variable, N outputs logic function can be implemented using a ROM of size $2^P \times N$ since all the possible 2^P minterms are effectively generated as is clear from the above discussion.

If a mask programmable ROM is used, the user can specify the bit pattern to be stored according to the requirements of the logic function, whereas the user can himself program it in case of PROM, EPROM and E^2PROM. Since programmable ROMs can be used for logic design, therefore, it is also referred to as a programmable logic device (PLD).

The advantages of using ROM as a programmable logic device are:

1. Ease of design since no simplification or minimization of logic function is required.
2. Designs can be changed, modified rapidly.
3. It is usually faster than discrete SSI/MSI circuit.
4. Cost is reduced.

There are few disadvantages also of ROM-based circuits, such as non-utilization of complete circuit, increased power requirement, and enormous increase in size with increase in number of input variables making it impractical.

12.3 PROGRAMMABLE LOGIC ARRAY

A programmable logic device (PLD) usually consists of programmable array of logic gates and interconnections with array inputs and outputs connected to the device pins through fixed logic elements, such as inverting/non-inverting buffers and FLIP-FLOPs. The logic gates used may be two-level AND-OR, NAND-NAND or NOR-NOR configuration. In some cases, AND-OR-EX-OR configuration is also used. Basically, there are two types of PLDs, *programmable logic array* (PLA) and *programmable array logic* (PAL). These are suitable for implementing logic functions in SOP form.

A PLA consists of two-level AND-OR circuits on a single chip. The number of AND and OR gates and their inputs are fixed for a given PLA chip. The AND gates provide the product terms, and the OR gates logically sum these product terms and thereby generate a SOP expression. It has M inputs, n product terms, and N outputs with $n < 2^M$, and can be used to implement a logic function of M variables with N outputs. Since all

of the possible 2^M minterms are not available, therefore, logic minimization is required to accommodate a given logic function.

The internal architecture of a PLA is shown in block diagram form in Fig. 12.2.

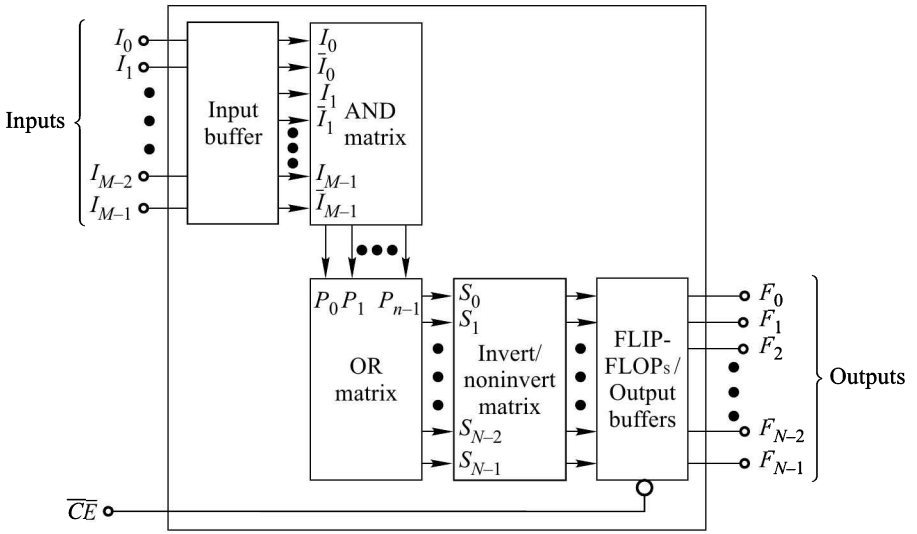


Fig. 12.2 ***Block Diagram of a PLA Device***

12.3.1 Input Buffer

The buffer circuits at the input are required to limit loading of the sources that drive the inputs. It produces inverted as well as non-inverted inputs at the output as shown in Fig. 12.3 for one input. Similar buffers are there for each of the M inputs.

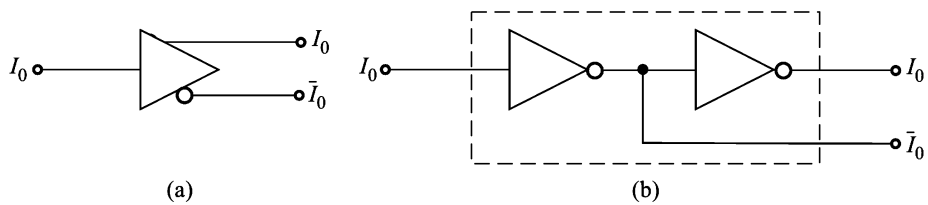


Fig. 12.3 ***An Input Buffer***

12.3.2 AND Matrix

An AND matrix is used to form product terms. A typical AND matrix is shown in Fig. 12.4. It has n AND gates with outputs P_0 through P_{n-1} and $2M$ inputs (I_0 through I_{M-1} and $\bar{I}_0$ through $\bar{I}_{M-1}$) for each AND gate. This shows that each AND gate has all the input variables in complemented and uncomplemented form. There is a nichrome fuse link in series with each diode. All the links are intact in an unprogrammed PLA device. Each AND gate generates one product term which is given by

$$P = I_0 \cdot \bar{I}_0 \cdot I_1 \cdot \bar{I}_1 \ldots I_{M-1} \cdot \bar{I}_{M-1}$$

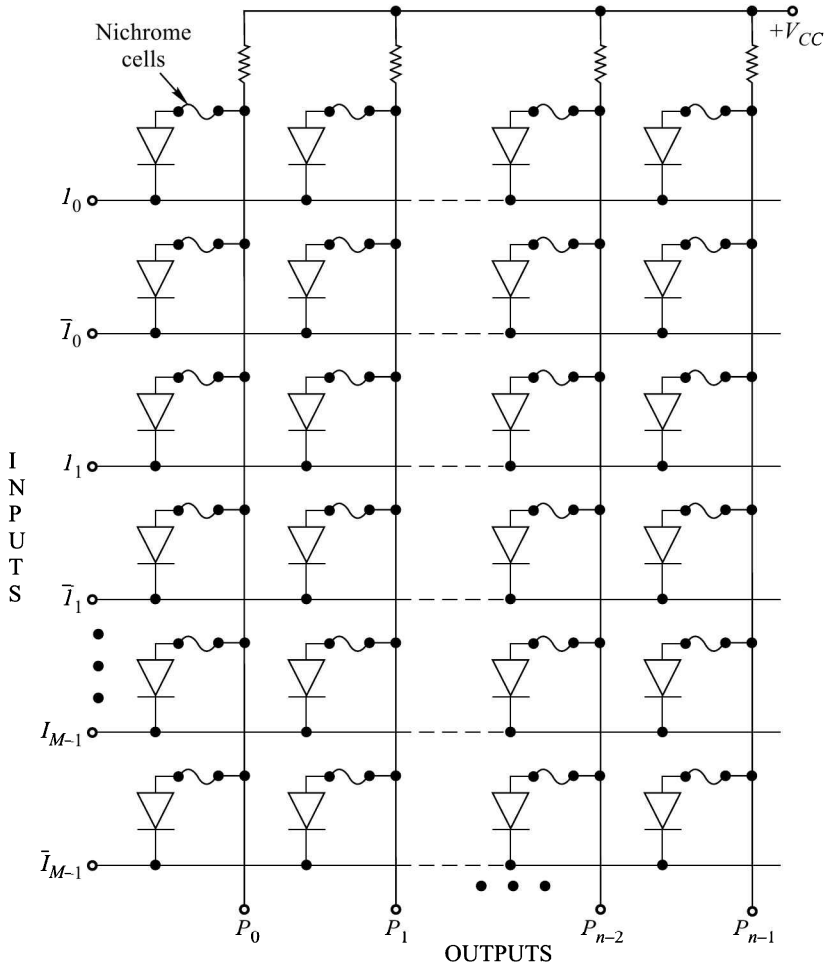


Fig. 12.4 ***An* AND *Matrix***

and it is logic 0 in an unprogrammed device. For generating a required product term the unwanted links are opened through the method of programming by a programmer device in case of a field-programmable logic device (FPLA). A section of the AND matrix for P_0 output is shown in Fig. 12.5*a* and its equivalent logic gate representation is shown in Fig. 12.5*b*. A similar arrangement exists for each of the other outputs.

Example 12.1

In Fig. 12.5, if all the links except the ones present for inputs I_0, $\bar{I}_2$, $\bar{I}_3$, and I_6 are opened, find the product term P_0.

Solution

For an open link, the input to the AND gate is logic 1, whereas for a closed link the corresponding input to the AND gate is same as the voltage applied at that input, therefore,

$$P_0 = I_0 \cdot \bar{I}_2 \cdot \bar{I}_3 \cdot I_6$$

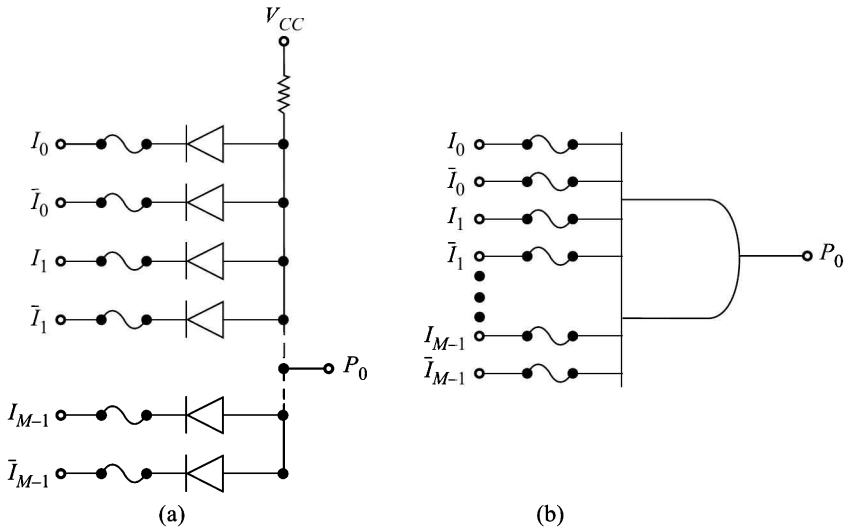


Fig. 12.5 ***A Section of the*** AND ***Matrix***

Figure 12.6 illustrates a convenient method of showing the input buffers and the AND matrix with the interconnections marked as *Xs*. Each AND gate has $2M$ inputs which is indicated by a single line in the figure. When an array is programmed to implement a particular logic function, the desired interconnections are left with *X* marks and the unwanted interconnections without *X* marks.

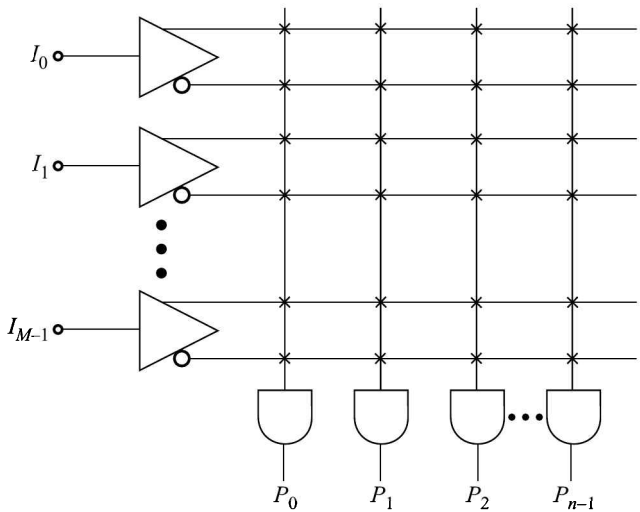


Fig. 12.6 ***Representation of Input Buffers and*** AND ***Matrix***

12.3.3 OR Matrix

The OR matrix is used to produce the logical sum of the product term outputs of the AND matrix. Figure 12.7 shows an OR matrix using RTL circuitry. An OR gate consists of parallel connected transistors with a common

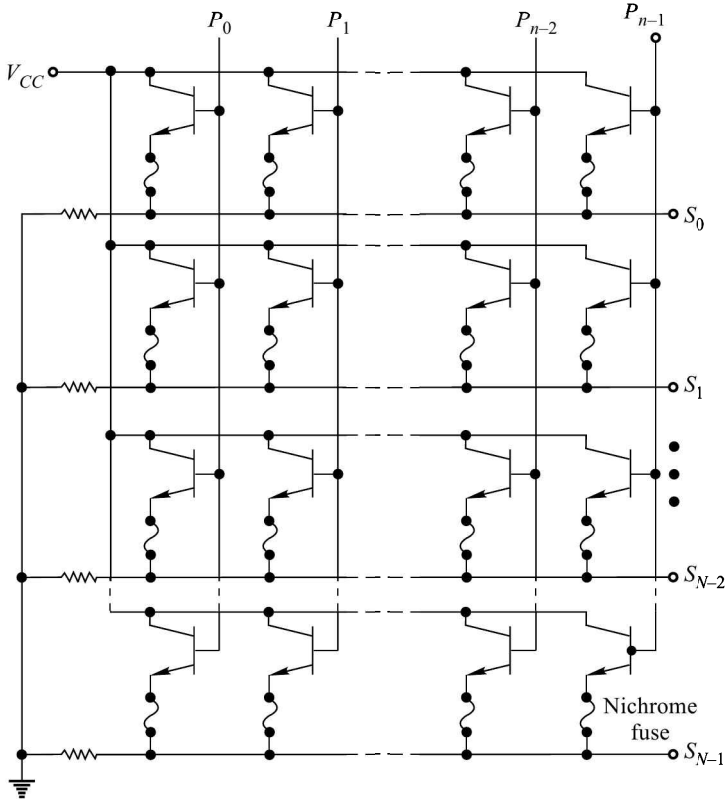


Fig. 12.7 ***An*** OR ***Matrix***

emitter load. The outputs of the OR matrix are obtained at S_0 through S_{N-1}. Consider the output S_0 when all the fuse links are intact, which is given by

$$S_0 = P_0 + P_1 + \ldots + P_{n-1}$$

The required sum terms can be generated by opening the unwanted fuse links. For example, if all the fuse links, except the ones for the product terms P_0 and P_1, are blown off for the output S_0, then

$$S_0 = P_0 + P_1$$

Thus an OR matrix can be programmed by opening the unwanted fuse links, which effectively makes logic level 0 at the corresponding OR gate inputs.

Figure 12.8 gives logic symbol for one of the OR gates, and Fig.12.9 illustrates the relevant portion of the PLA containing OR matrix in a way similar to Fig. 12.6.

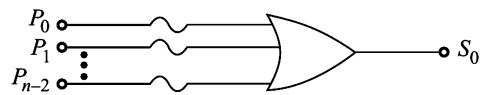


Fig. 12.8 ***Logic Symbol of a Section of*** OR ***Matrix***

12.3.4 INVERT/NON-INVERT Matrix

This is a programmable buffer that can be set for INVERTING or NON-INVERTING operation corresponding to active-low or active-high output, respectively. Typical circuits for this operation are shown in Fig. 12.10.

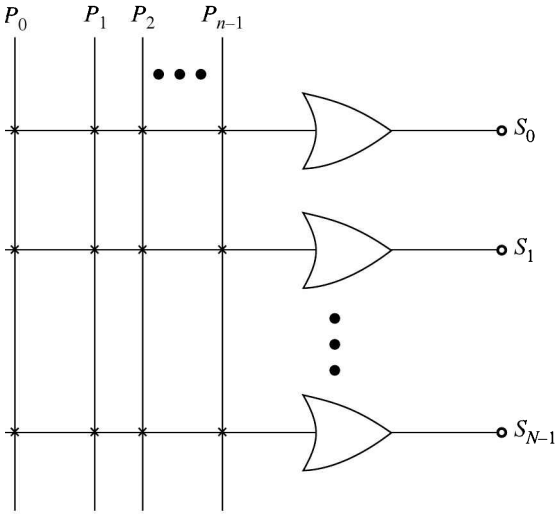
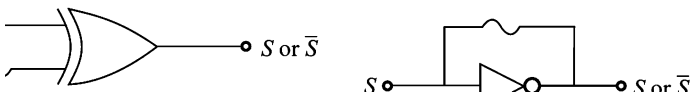


Fig. 12.9 ***Representation of*** OR ***Matrix***

ate, if the fuse is intact, the output is S, whereas the output is $\overline{S}$ if the fuse is ut in case of Fig. 12.10*b* is S or $\overline{S}$ depending upon whether the fuse is intact or



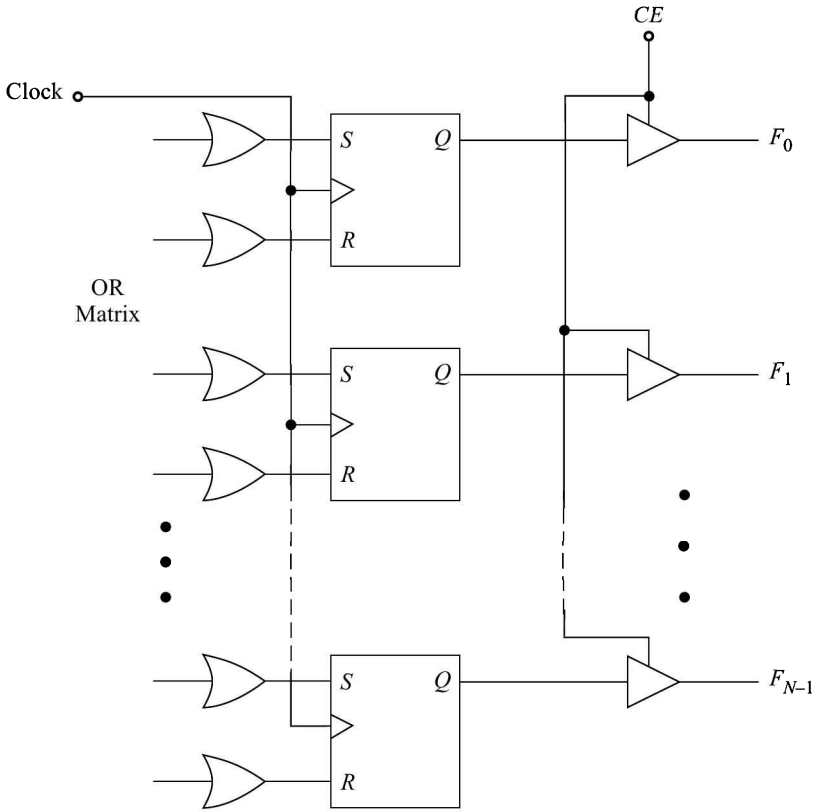


Fig. 12.12 ***A Section of PLA with*** `FLIP-FLOPs` ***in the Output***

12.3.7 Programming the PLA

A PLA device is to be programmed for the desired input–output relationship similar to the programming of ROMs. For a mask-programmable PLA device, the data pattern is to be specified by the customer. The appropriate masks are designed by the manufacturers and the data pattern are built in during the manufacturing process.

An FPLA has all its nichrome fuse links intact at the time of manufacturing. The unwanted links are electrically open circuited during programming. The links to be opened are accessed by applying voltages at the inputs and outputs of the device, The FPLAs are not reprogrammable.

12.3.8 Expanding PLA Capacity

Some applications require a capacity that exceeds the capacity of a single PLA. The required capacity can be achieved by suitably connecting several identical devices together. For increasing the number of outputs, the inputs of two or more devices are to be connected individually in parallel. This connection does not change the number of inputs and the product terms.

For increasing the number of product terms keeping the number of inputs and outputs unchanged, the inputs and outputs of two or more devices are to be individually connected in parallel.

The number of inputs can be increased by making the connections as shown in Fig. 12.13. This circuit has $(M + Q)$ inputs and N outputs. This connection also increases the number of product terms. The number of product terms is $P \times (2^Q - 1)$. The outputs are allowed to be connected together for the devices with passive pull-up only.

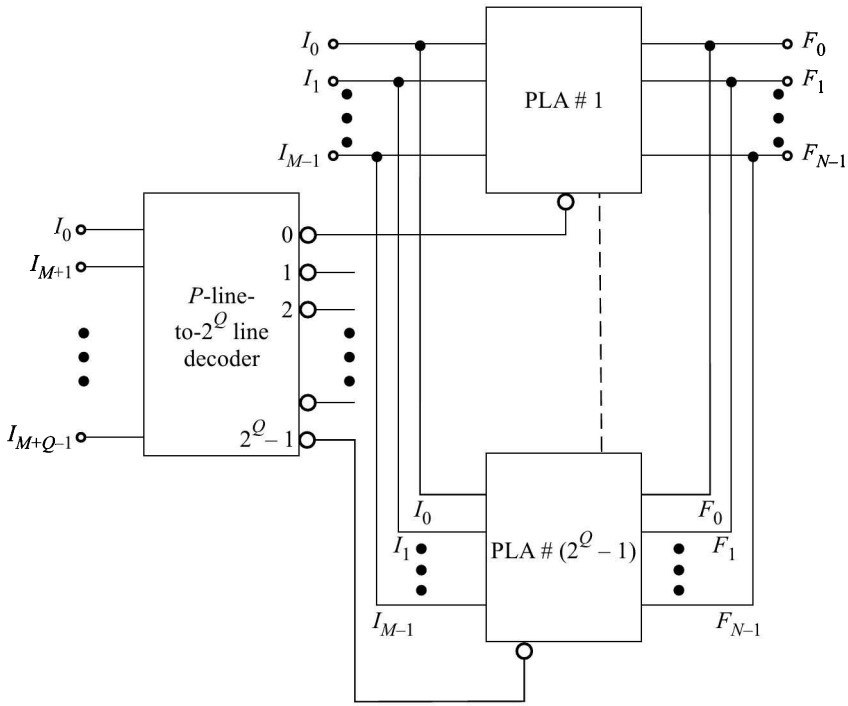


Fig. 12.13 ***Expanding the Input Word Length of PLA***

12.3.9 Applications of PLAs

The PLAs can be used to implement combinational and sequential logic circuits. For the design of combinational circuits, PLAs with output circuit as shown in Fig. 12.10 or Fig. 12.11 are used, whereas the devices having FLIP-FLOPs in the output circuit as shown in Fig. 12.12 are required for the design of sequential circuits.

The following steps can be used for implementing combinational logic functions:

1. Prepare the truth table.
2. Write the Boolean equations in SOP form.
3. Simplify the equations to obtain minimum SOP form. The main criterion is to minimize the number of product terms.
4. Determine the input connections of AND matrix to generate the required product terms.
5. Determine the input connections of OR matrix to generate the required sum terms.
6. Determine the connections required for INVERT/NON-INVERT matrix to set the active logic levels of the outputs.
7. Program the PLA.

Example 12.2

Design a 4-input, 5-output combinational circuit using PLS100 PLA. The input variables are *A, B, C,* and *D*.

$$Y_1 = \Sigma m(0, 3, 5, 6, 9, 10, 12, 15)$$
$$Y_2 = \Sigma m(0, 1, 2, 3, 11, 12, 14, 15)$$
$$Y_3 = \Sigma m(0, 4, 8, 12)$$
$$Y_4 = \Sigma m(0, 2, 3, 5, 7, 8, 12, 13)$$
$$Y_5 = \Sigma m(0, 1, 3, 4, 5, 6, 11, 13, 14, 15)$$

Solution

The PLS 100 PLA device (Fig. 12.14) has 16 inputs, 8 outputs and 48 product terms. The output is through EX-OR gate, for controlling the polarity of the output, and a 3-state buffer enabled through pin 19 of the chip. A low signal

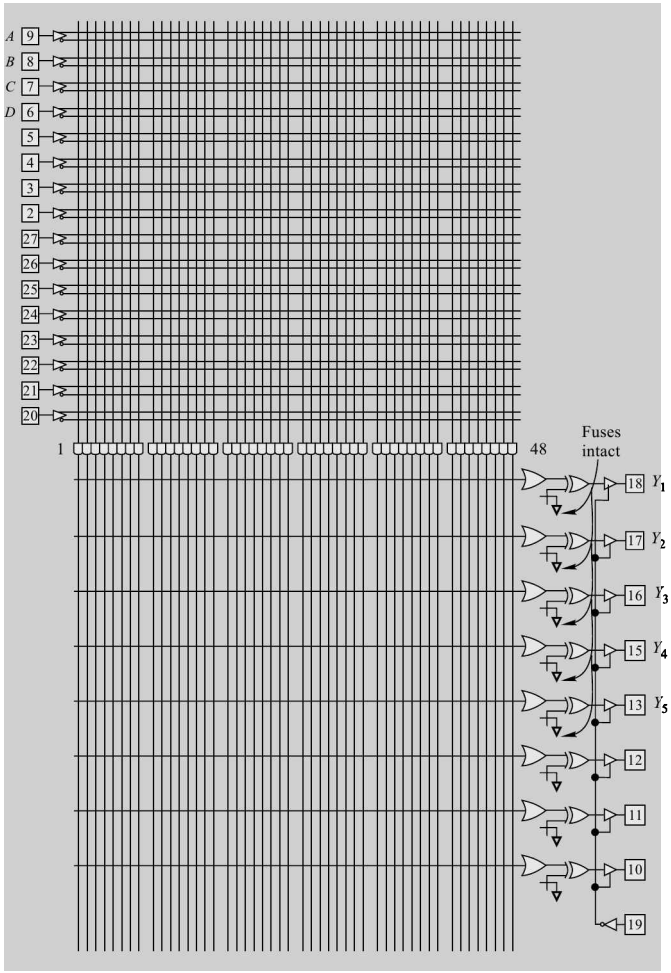


Fig. 12.14 ***PLS Device Programmed for Ex. 12.2***

is required to be applied to this pin for enabling the output buffers. The fuses of EX-OR gates are to be kept intact to maintain these inputs at ground (0) level for active-high outputs.

The functions $Y_1 - Y_5$ are given in minterm form which are 16 in number. This requires only 16 product terms (or 16 AND gates) which are available in the device and, therefore, no simplification or minimisation is required.

The inputs *A, B, C,* and *D* are applied at pins 9, 8, 7 and 6, respectively, and the AND gates are numbered from 1 to 48 starting from the left most gate.

Gates 1 to 16 are used for generating 0 to 15 minterms. The relevant *X* are indicated in Fig. 12.14 in the rows corresponding to the inputs. The output pins used for the outputs Y_1 to Y_5 are 18, 17, 16, 15 and 13, respectively. The relevant *X* are indicated in the figure. One input of each of the output EX-OR gate is connected to ground through a fuse.

The design of sequential circuits follow the same method as discussed in chapter 8 and then the PLA is configured accordingly.

Example 12.3

Design a 4-bit UP/DOWN counter with direction control *M*. Assume *M* = 0 for UP counting and *M* = 1 for DOWN counting.

Solution

The sequential PLS 105 PLA device can be used for this. It has 14 FLIP-FLOPs, 6 of which are buried within the device and are not available for outputs. The outputs of these 6 FFs are fedback to the AND array. The remaining 8 FFs are associated with the device outputs and are not used for feedback to the AND array. Pin 1 is used for clock input and 19 for providing preset input (*P*) to the FLIP-FLOPs and enable input to the output buffers.

Table 12.1 gives the count sequence and the inputs required for the FFs. The K-maps for $S_3, R_3, S_2, R_2, S_1, R_1, S_0$, and R_0 are given in Fig. 12.15, and the expressions for these inputs in minimized form are also given. The device is programmed using these equations and is shown in Fig. 12.16.

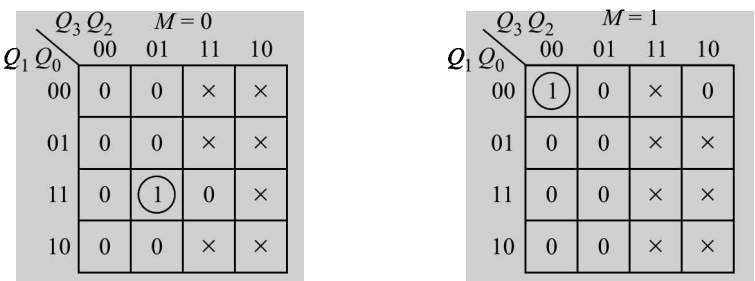
Table 12.1

UP/DOWN	Counter state				S-R FFs inputs							
Control *M*	Q_3	Q_2	Q_1	Q_0	S_3	R_3	S_2	R_2	S_1	R_1	S_0	R_0
0	0	0	0	0	0	×	0	×	0	×	1	0
0	0	0	0	1	0	×	0	×	1	0	0	1
0	0	0	1	0	0	×	0	×	×	0	1	0
0	0	0	1	1	0	×	1	0	0	1	0	1
0	0	1	0	0	0	×	×	0	0	×	1	0
0	0	1	0	1	0	×	×	0	1	0	0	1
0	0	1	1	0	0	×	×	0	×	0	1	0
0	0	1	1	1	1	0	0	1	0	1	0	1
0	1	0	0	0	×	0	0	×	0	×	1	0
0	1	0	0	1	×	0	0	×	1	0	0	1
0	1	0	1	0	×	0	0	×	×	0	1	0
0	1	0	1	1	×	0	1	0	0	1	0	1
0	1	1	0	0	×	0	×	0	0	×	1	0
0	1	1	0	1	×	0	×	0	1	0	0	1
0	1	1	1	0	×	0	×	0	×	0	1	0
0	1	1	1	1	0	1	0	1	0	1	0	1

(Continued)

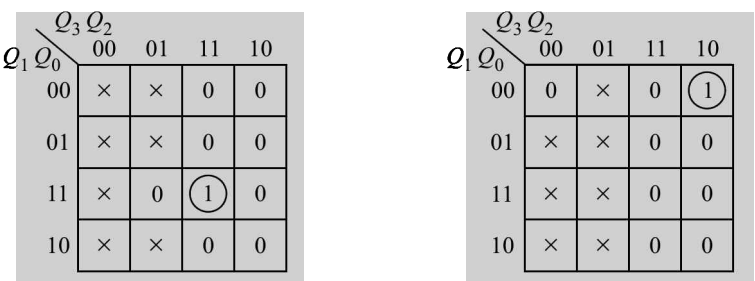
Table 12.1 (*Continued*)

UP/DOWN Control M	Counter state				S-R FFs inputs							
	Q_3	Q_2	Q_1	Q_0	S_3	R_3	S_2	R_2	S_1	R_1	S_0	R_0
1	0	0	0	0	1	0	1	0	1	0	1	0
1	1	1	1	1	×	0	×	0	×	0	0	1
1	1	1	1	0	×	0	×	0	0	1	1	0
1	1	1	0	1	×	0	×	0	0	×	0	1
1	1	1	0	0	×	0	0	1	1	0	1	0
1	1	0	1	1	×	0	0	×	×	0	0	1
1	1	0	1	0	×	0	0	×	0	1	1	0
1	1	0	0	1	×	0	0	×	0	×	0	1
1	1	0	0	0	0	1	1	0	1	0	1	0
1	0	1	1	1	0	×	×	0	×	0	0	1
1	0	1	1	0	0	×	×	0	0	1	1	0
1	0	1	0	1	0	×	×	0	0	×	0	1
1	0	1	0	0	0	×	0	1	1	0	1	0
1	0	0	1	1	0	×	0	×	×	0	0	1
1	0	0	1	0	0	×	0	×	0	1	1	0
1	0	0	0	1	0	×	0	×	0	×	0	1
	0	0	0	0								



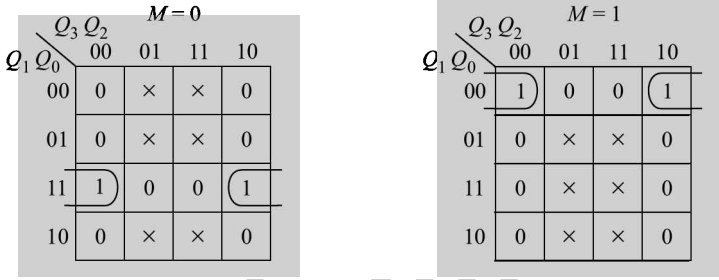
$$S_3 = \overline{Q}_3 \cdot Q_2 \cdot Q_1 \cdot Q_0 \cdot \overline{M} + \overline{Q}_3 \cdot \overline{Q}_2 \cdot \overline{Q}_1 \cdot \overline{Q}_0 \cdot M$$

(a) *K*-map for S_3



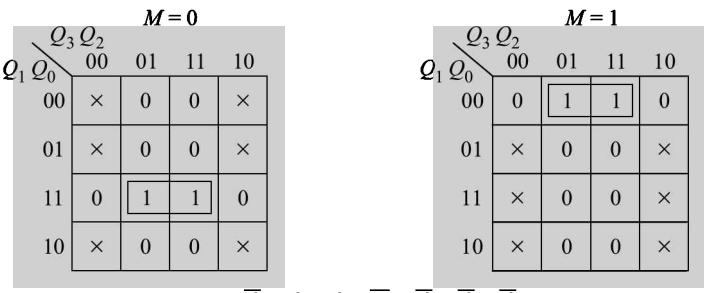
$$R_3 = Q_3 \cdot Q_2 \cdot Q_1 \cdot Q_0 \cdot \overline{M} + \overline{Q}_3 \cdot \overline{Q}_2 \cdot \overline{Q}_1 \cdot \overline{Q}_0 \cdot M$$

(b) *K*-map for R_3



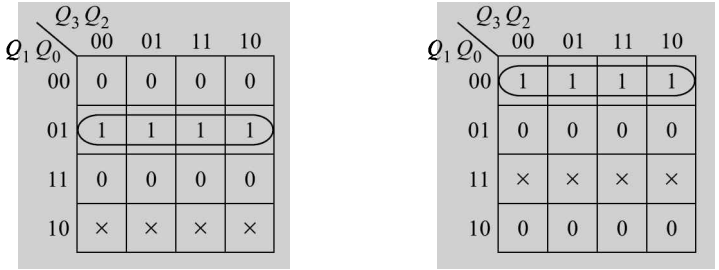
$$S_2 = \overline{Q}_2 \cdot Q_1 \cdot Q_0 \cdot \overline{M} + \overline{Q}_2 \cdot \overline{Q}_1 \cdot \overline{Q}_0 \cdot M$$

(c) *K*-map for S_2



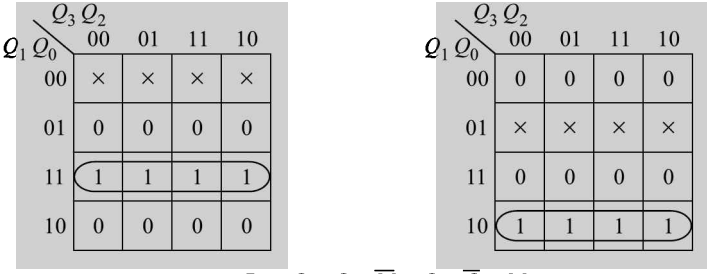
$$R_2 = \overline{Q}_2 \cdot Q_1 \cdot Q_0 \cdot \overline{M} + \overline{Q}_2 \cdot \overline{Q}_1 \cdot \overline{Q}_0 \cdot M$$

(d) *K*-map for R_2



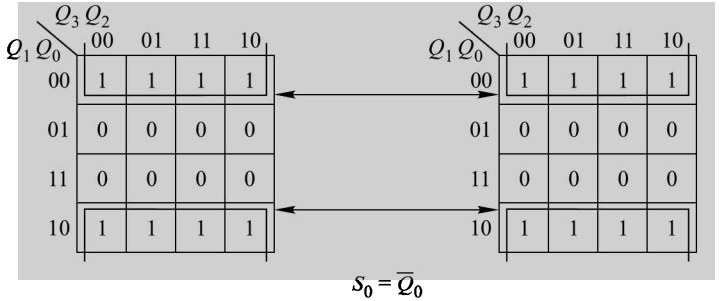
$$S_1 = \overline{Q}_1 \cdot Q_0 \cdot \overline{M} + \overline{Q}_1 \cdot \overline{Q}_0 \cdot M$$

(e) *K*-map for S_1



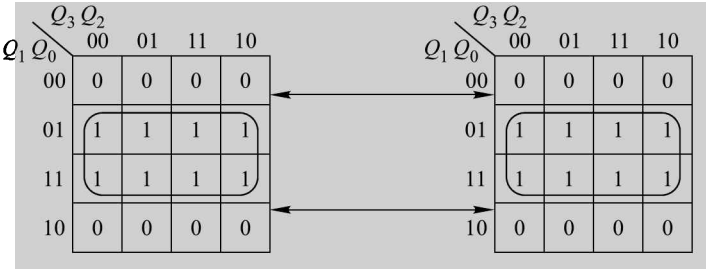
$$R_1 = Q_1 \cdot Q_0 \cdot \overline{M} + Q_1 \cdot \overline{Q}_0 \cdot M$$

(f) *K*-map for R_1



$$S_0 = \overline{Q}_0$$

(g) *K*-map for S_0

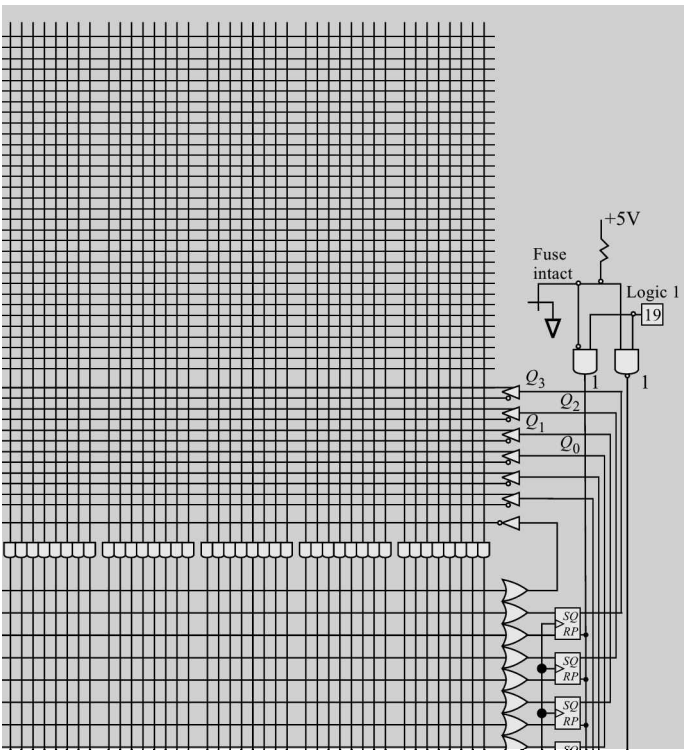


$$R_0 = Q_0$$

(h) *K*-map for R_0

Fig. 12.15 ***K-Maps for Ex. 12.3***

using these equations and is shown in Fig. 12.16.



12.3.10 Available PLAs

Some of the commercially available PLA ICs with their features are given in Table 12.2. The 82S200 and 82S201 are pin-for-pin mask programmable replacements for the 82S100 and 82S101, respectively. The PLS 100 and PLS 105 PLAs are equivalent to 82S100 and 82S105, respectively. The DM 7575 PLA has totem-pole output, whereas DM7576 and IM5200 have passive pull-up. The devices with passive pull-up are useful for expanding functions by wire-ANDing the outputs of similar other devices.

The PLA devices are not used for new designs because of the availability of more powerful and power efficient devices, such as GALs, CPLDs and FPGAs. However, the concept of PLA is used in some CPLDs. The FPLAs are used mostly in state-machine design, where a large number of product terms are needed in each SOP expression.

12.4 PROGRAMMABLE ARRAY LOGIC

A most commonly used type of PLD is *programmable array logic* (PAL). It is programmable array of logic gates on a single chip in AND-OR configuration. In contrast to PLA, it has programmable AND array and a fixed OR array in which each OR gate gets inputs from some of the AND gates, i.e. all the AND gate outputs are not connected to any OR gate. Figure 12.17 illustrates the configuration of AND and OR arrays for a PAL with 5 inputs, 8 programmable AND gates and 4 fixed OR gates. Each AND gate has all the 10 inputs (in complemented and uncomplemented form) with fusible links intact which can be programmed to generate 8 product terms. Each OR gate gets inputs from the outputs of only two AND gates shown by •. The input and output circuits of PALs are similar to those of PLAs. The number of fusible links in a PAL is the product of $2M$ and n, where M is the number of input variables and n is the number of product terms.

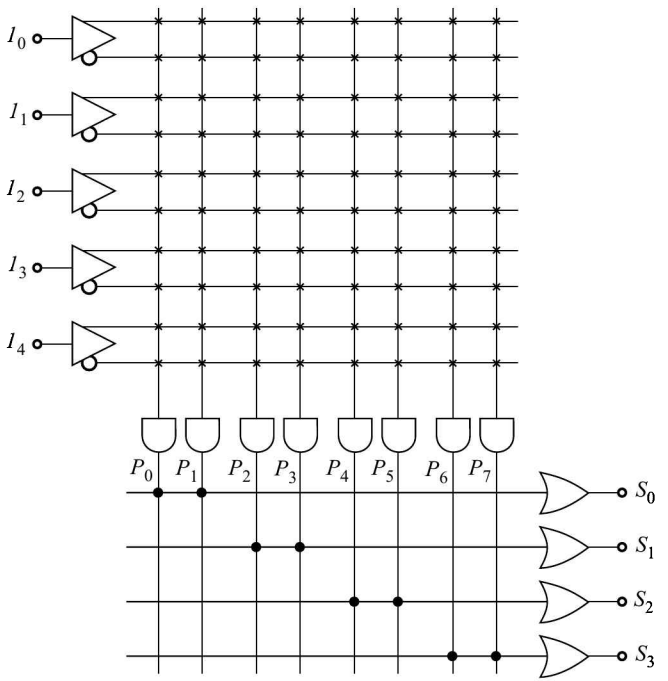


Fig. 12.17 ***Programmable Array Logic***

Table 12.2 ***Some Available PLAs with their Features***

IC No.	PLA/ FPLA	Number of			Output	Access time ns	Supply voltage V	Power dissipation mW	Packaging	Input/ output logic levels
		Inputs	Product-terms	Output						
82S200	PLA	16	48	8	*TS*	80	+5	600	28-pin DIP	TTL compatible
82S201	PLA	16	48	8	*OC*	80	+5	600	28-pin DIP	TTL compatible
82S100	FPLA	16	48	8	*TS*	80	+5	600	28-pin DIP	TTL compatible
82S101	FPLA	16	48	8	*OC*	80	+5	600	28-pin DIP	TTL compatible
DM7575	PLA	14	96	8	*TS*	150	+5	550	24-pin DIP	TTL compatible
DM7576	PLA	14	96	8	*OC*	150	+5	550	24-pin DIP	TTL compatible

12.4.1 Combinational PAL

One of the most commonly used PAL 16L8 is shown in Fig. 12.18. Its programmable AND array consists of 64 gates. It has 16 input variables and 8 outputs, therefore, each AND gate has $2 \times 16 = 32$ inputs. Eight AND gates are associated with each output pin, seven of them provide inputs to a fixed 7-input OR gate and the eighth is connected to the enable input terminal of the output buffer. Thus, any output can perform only logic functions that can be expressed as sums of seven or fewer product terms. Each product term can be a function of up to all 16 inputs but only 7 such product terms are available. Since, in a PAL the inputs to the OR gates are fixed, therefore, no two OR gates can share a product term. Where a product term is needed by two OR gates, it must be generated twice.

There is a three-state inverter between the output of each OR gate and the output pin of the device, therefore, the output may be programmed as always enabled, always disabled, or enabled by a product term involving the inputs (Problem 12.11).

This PAL has 10 pins (I_1 to I_{10}) for 10 inputs, the other six pins are used for inputs as well as for six of the outputs. These are IO_2 to IO_7. O_1 and O_8 are dedicated output pins. The device can be programmed for the following options:

1. If an *IO* pin's output inverter is disabled, then the pin can be used only as an input.
2. If an *IO* pin is not required to be input, then it can be used as an output. The output inverter can be enabled either continuously or for certain input conditions by using a product term. In case it is enabled continuously, its output can also be used as an input.
3. This can be used either for the implementation of logic functions that can not be accommodated in 7 product terms by using two pass AND-OR configuration or for implementing sequential circuits by providing feedback.

Since limited number of product terms are available for each output, logic minimization techniques are required when logic circuits are implemented in PAL devices.

PAL 16H8 is same as 16L8 except that it does not have inverters in the output. Another PAL 16P8 has programmable output polarity using EX-OR gates as shown in Fig. 12.10*a*. The alphabets *L, H,* and *P* used in device numbers stand for active-low, active-high, and programmable outputs, respectively, and these are all *combinational* programmable array logic devices, i.e. these do not have memory elements.

12.4.2 Registered PALs

Sequential digital circuits use FLIP-FLOPs in addition to combinational circuits and, therefore, for the design of sequential circuits, PALs have been developed with FLIP-FLOPs in the outputs and these devices are known as *registered* PALs. Some of the available registered PALs are 16R4, 16R6, 16R8, and their programmable output polarity versions 16RP4, 16RP6, and 16RP8.

Figure 12.19 illustrates the structure of a 16R6 registered PAL. This device has the following features:

- Eight primary (external) inputs, I_1 to I_8.
- Two pins common for input/output similar to the device 16L8, IO_1 and IO_8.
- Six outputs (O_2 to O_7) through positive-edge-triggered D-FFs and three state inverters. $\overline{Q}$ outputs of these FFs are routed back to the AND array, making total number of inputs to the AND array as 16 and 8 outputs.
- The FFs are all controlled by a common clock through a pin and another dedicated input pin is available for output enable control of inverters.

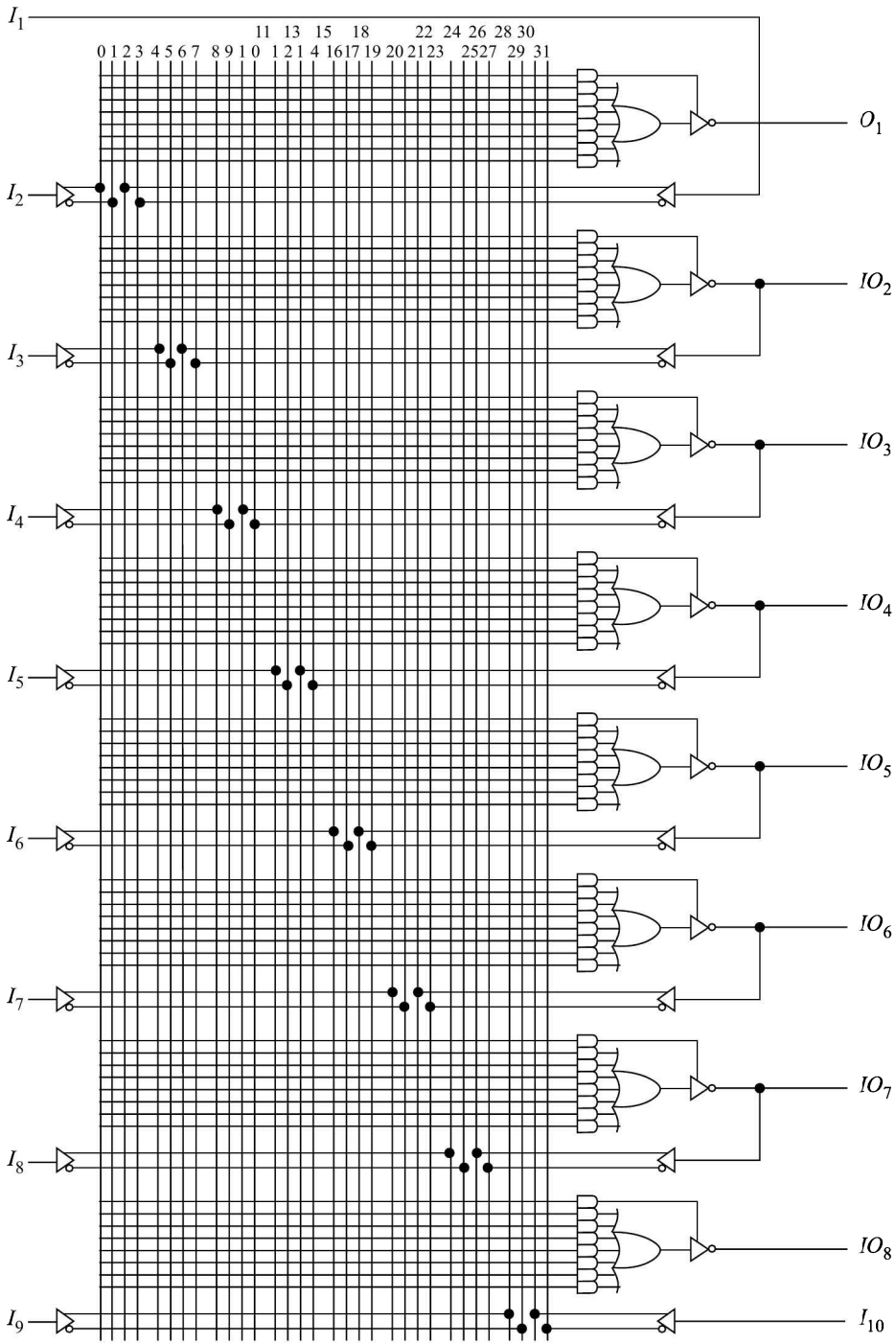


Fig. 12.18 ***Logic Diagram of the PAL 16L8***

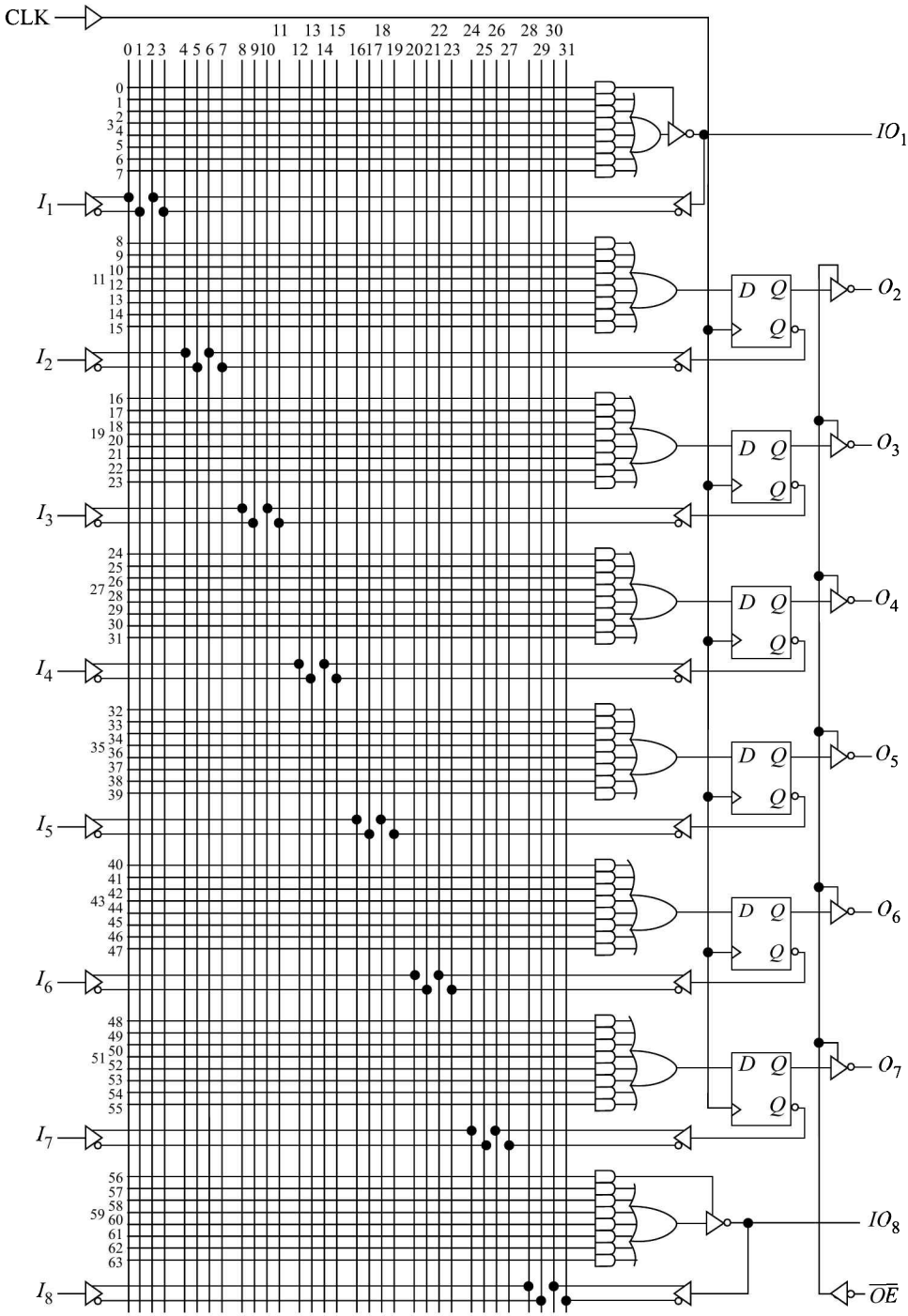


Fig. 12.19 ***Logic Diagram of the Registered PAL 16R6***

12.4.3 Configurable PALs

Developments in the design of programmable array logic devices have led to the introduction of configurable outputs enhancing the output capabilities of such devices. The *configurable* (also known as *generic*) device architecture is provided by equipping the device with enhanced special circuitry, known as *output macrocells*. A macrocell has circuitry with fuses which can be configured for a variety of output options, giving flexibility to the device. A configurable PAL can replace a large number of simpler PAL type devices with a 'one size fits all' device and allows designs to be implemented that are challenging or simply impossible for the simpler PAL devices to handle.

The most popular industry standard PAL devices are 16V8, 20V8, and 22V10.

Figure 12.20 gives logic diagram of 22V10 configurable PAL. It has 22 inputs, 10 outputs, and 120 product terms. Out of 22 inputs, 12 are dedicated inputs (I_1 to I_{12}) and ten may be used for inputs as well as outputs (IO_1–IO_{10}). One of the dedicated inputs I_1 also functions as the common clock input to the positive-edge-triggered D-type FF of each of the ten output macrocells. All the ten outputs are through configurable macrocells and three-state inverters. The output pins can also be used as inputs.

There are two extra product terms also available as seen in Fig. 12.20 which can be used for synchronously presetting (SP) and asynchronously resetting (AR) the D-type FFs in the output macrocells.

Figure 12.21 gives the logic diagram of an output macrocell. All the output macrocells are identical. Each macrocell contains a positive edge-triggered D-type FF and fuse-configurable multiplexers. The two fuses, S_1 and S_0 that control the multiplexers can be configured in four different ways, as shown in Fig. 12.22. From this, it is observed that this configurable PAL can function as registered PAL or combinational PAL, and in each case the output may be in inverted or non-inverted form. It may also be noted from Fig. 12.20 that all the number of product terms available to various OR gates in the device are not same. The number of product terms associated with each OR gate is indicated near the OR gate. Because of this, the logic functions of significantly more complexity can be implemented using 22V10.

12.4.4 Electrically Erasable Programmable Logic Devices (EEPLDs)

The industry standard programmable array logic devices are now available using CMOS electrically erasable flash technology. These EEPLDs are reprogrammable, and are available in a variety of voltage and power-saving options to meet various different requirements. Some of the features of ATF16V8B, ATF20V8B, and ATF22V10B are:

- Programming and erasing are performed using standard PLD programmers.
- A single *security fuse* is provided to prevent unauthorised fuse patterns. It should be programmed last, since once it is programmed, no further data can be programmed.
- There is a provision of *electronic signature*, for which a 64-bit of programmable memory is always available to the user for user-specific data. This feature is available even if the device is secured.
- They can retain data for 20 years and 100 erase/write cycles are possible.
- Their inputs and outputs are CMOS and TTL compatible.

These devices are discussed below.

There is another family of electrically erasable devices that is known as programmable electrically erasable logic (PEEL) device. It uses CMOS electrically erasable technology and it is reprogrammable. The 18CV8 PEEL device will also be discussed below.

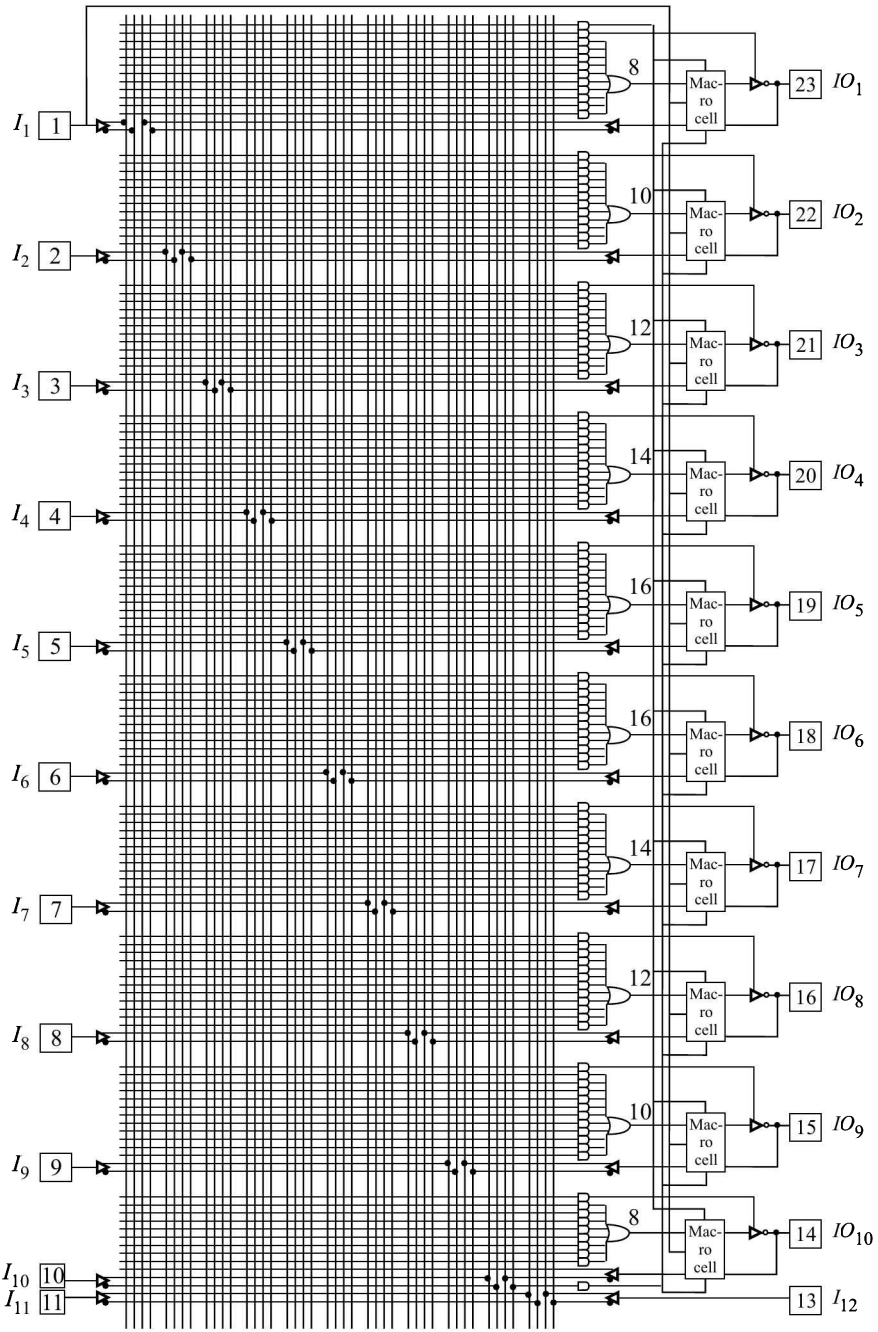


Fig. 12.20 ***22V10 Configurable PAL***

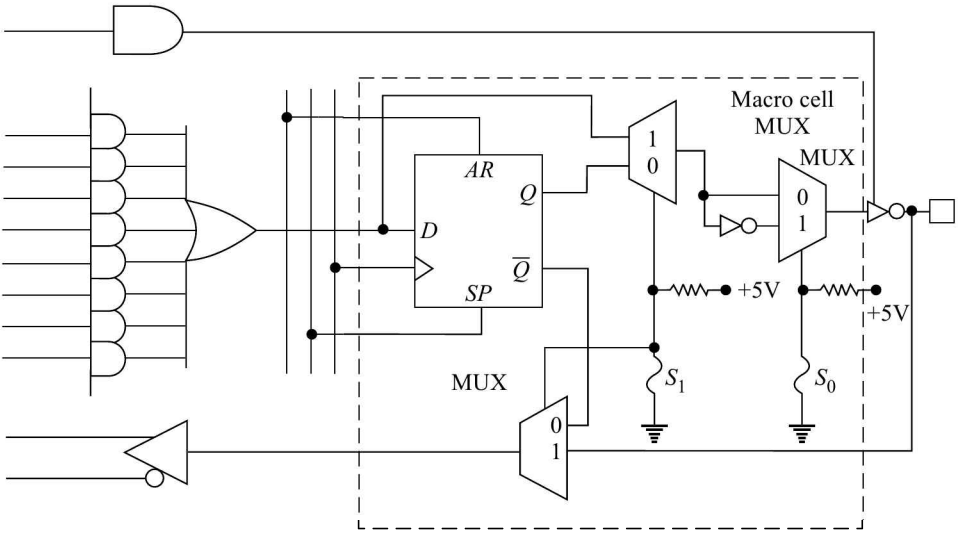


Fig. 12.21 ***Logic Diagram of a 22V10's Output Macrocell***

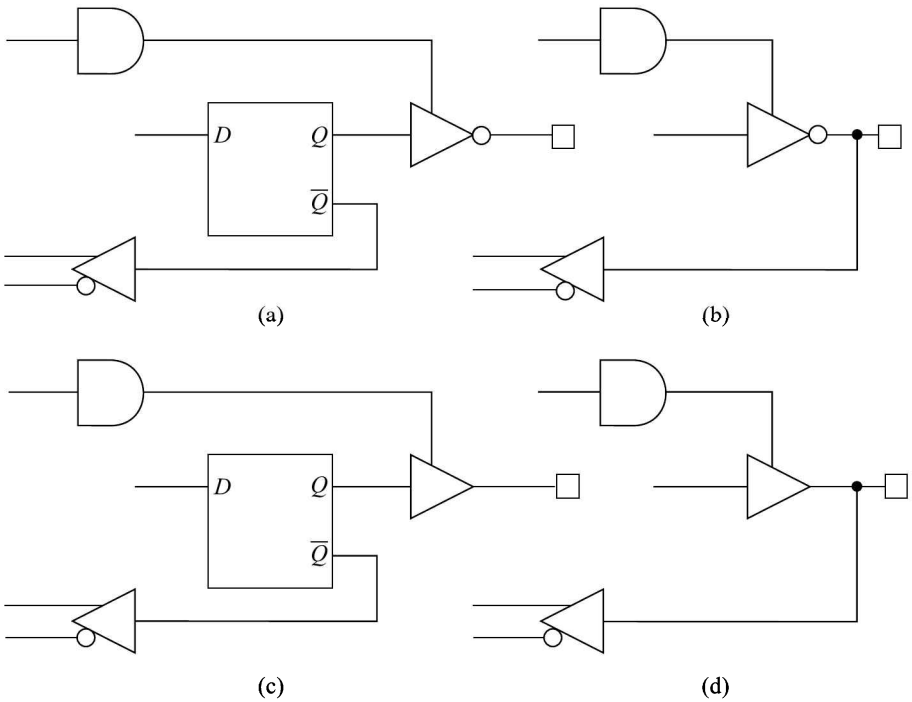


Fig. 12.22 ***Output Configurations for Fuses (a) S_1 and S_0 Intact (b) S_0 Intact and S_1 Open (c) S_0 Open and S_1 Intact (d) S_1 and S_0 Open***

ATF16V8B EEPLD

This device has 16 inputs (8 dedicated inputs and 8 I/O) and is available in 20-pin package. There are eight configurable macrocells which can be configured as

- registered output
- combinatorial I/O
- combinatorial output, or
- dedicated input.

It can be configured in three different modes:

- Registered mode
- Complex mode
- Simple mode

Registered Mode

The registered mode is used if one or more registers are required. In this mode, each macrocell can be configured as registered or combinatorial output, or I/O, or as an input. Figure 12.23 shows the logic diagram and Fig. 12.24 shows its configuration in the registered mode. Here, pin 1 is used for common clock for the registered outputs and pin 11 is used for common output enable ($\overline{OE}$). Figure 12.25 shows the macrocell configuration for combinatorial output, or I/O or as an input.

The following registered PAL devices can be emulated using the registered mode: 16R8, 16R6, 16R4, 16RP8, 16RP6, and 16RP4.

Complex Mode

In the complex mode, combinatorial output and I/O functions are possible. Here, in addition to eight dedicated inputs, the pins 1 and 11 also are used as regular inputs. Pin 13 through 18 have feedback paths back to the AND-array, which makes full I/O capability possible. Pins 12 and 19 are outputs only.

Figure 12.26 shows the logic diagram and Fig. 12.27 shows the macrocell configuration in the complex mode. The PAL 16L8, 16H8, and 16P8 devices can be emulated using complex mode.

Simple Mode

In the simple mode, 8 product terms are allocated to the sum term. Pins 15 and 16 are permanently configured as combinatorial outputs. The remaining six macrocells can be configured as either inputs or combinational outputs with pin feedback to the AND-array. Pins 1 and 11 are regular inputs. Figure 12.28 shows the logic diagram and Fig. 12.29 shows the macrocell configuration in the simple mode.

The following PALs can be emulated using the simple mode:

10L8	10H8	10P8
12L6	12H6	12P6
14L4	14H4	14P4
16L2	16H2	16P2

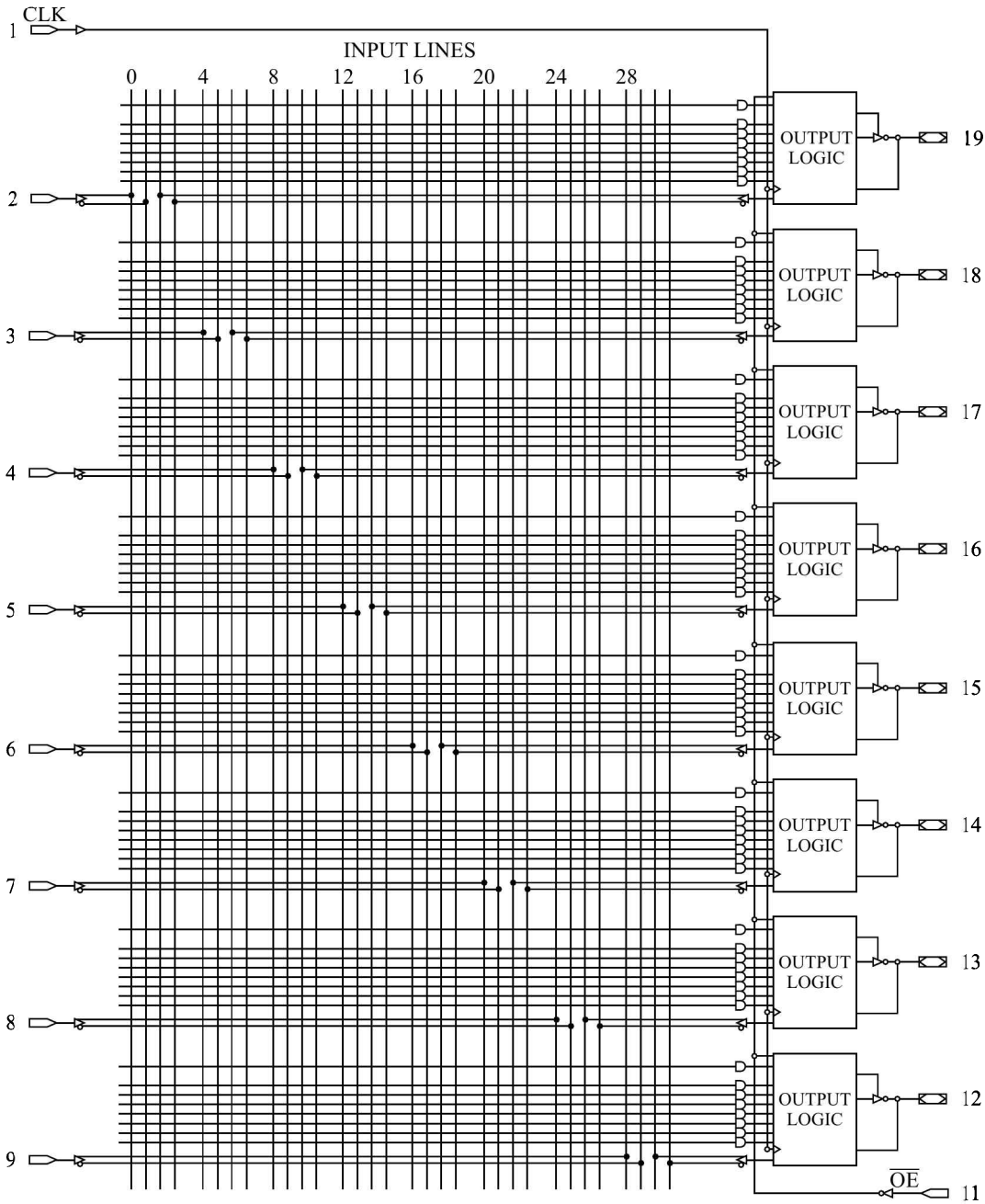


Fig. 12.23 ***Logic Diagram of 16V8B in Registered Mode***

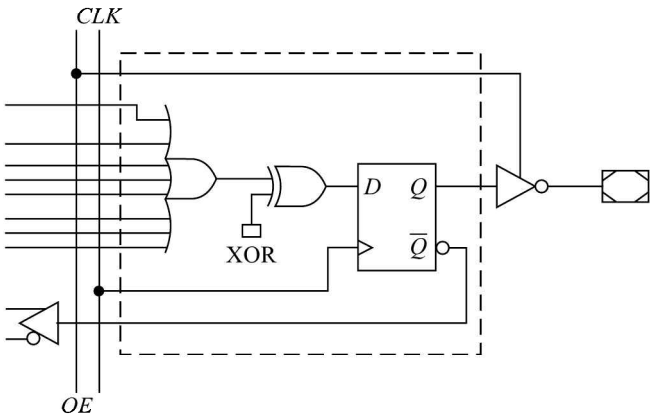


Fig. 12.24 ***Registered Mode Output Cell Configuration of 16V8B***

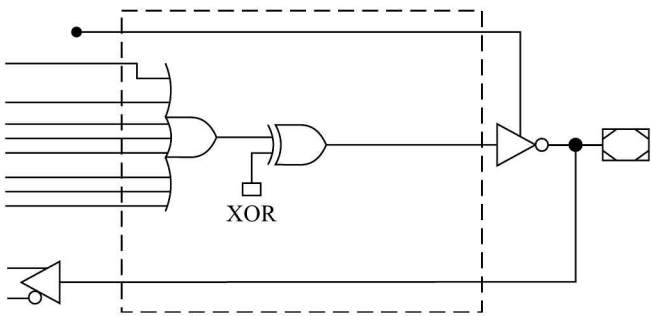


Fig. 12.25 ***Combinatorial Configuration in Registered Mode Configuration of 16V8B***

ATF20V8B EEPLD

The 20V8 device has 20 inputs (8 I/O and 12 dedicated inputs) and is available in 24 and 28 pin packages. There are eight configurable macrocells which can be configured in registered, complex, and simple modes in a way similar to the 16V8 device. It incorporates a superset of the generic architectures, which allows direct replacement of the 20R8 family and most of the 24-pin combinatorial PLDs.

18CV8 PEEL

The 18CV8 PEEL device (Fig. 12.30) has ten dedicated inputs and 8 I/O providing upto 18 inputs and 8 outputs. It can implement upto eight SOP logic expressions. Its output macrocell is shown in Fig. 12.31 and can be configured in twelve different configurations (Problem 12.10).

12.4.5 Generic Array Logic (GAL) Devices

The electrically erasable PLDs of Lattice Semiconductor Corporation have GAL as their registered trademark. The SPLD devices are GAL 16V8, GAL 20V8, and isp GAL 22V10 corresponding to 16V8, 20V8, and

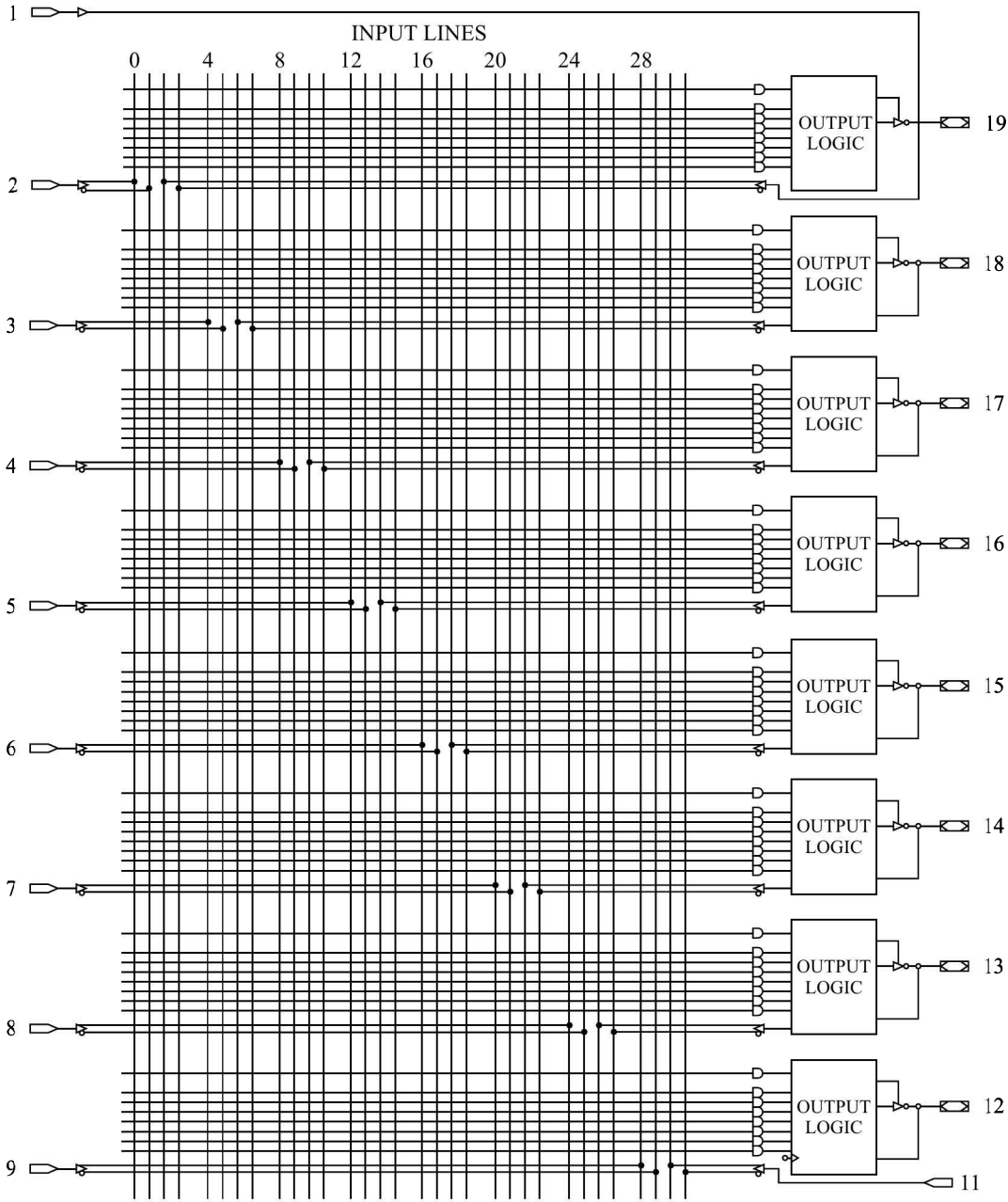


Fig. 12.26 ***Complex Mode Logic Diagram of 16V8B***

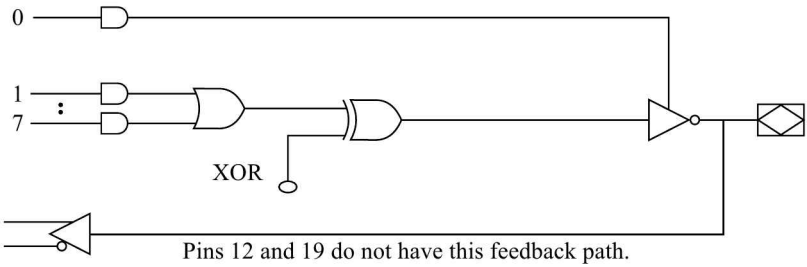


Fig. 12.27 ***Macrocell Configuration for Complex Mode***

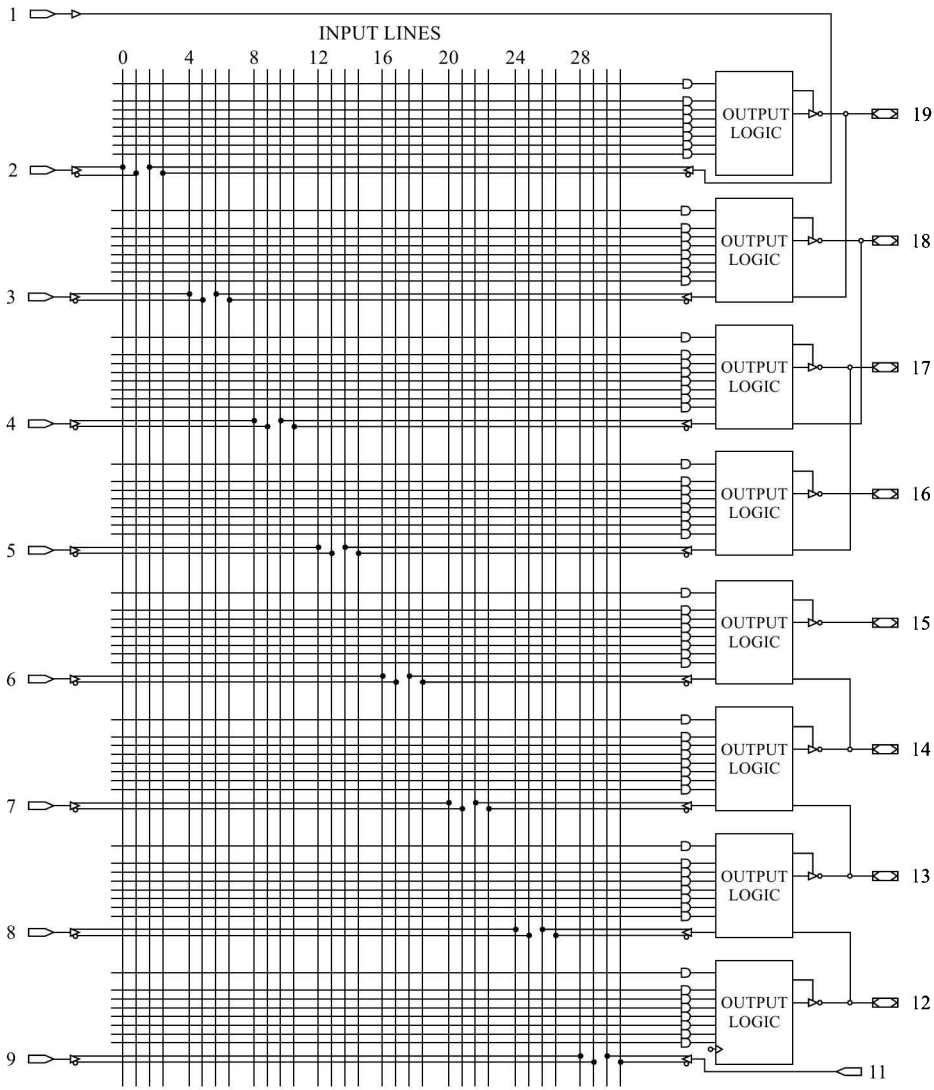


Fig. 12.28 ***Simple Mode Logic Diagram of 16V8B***

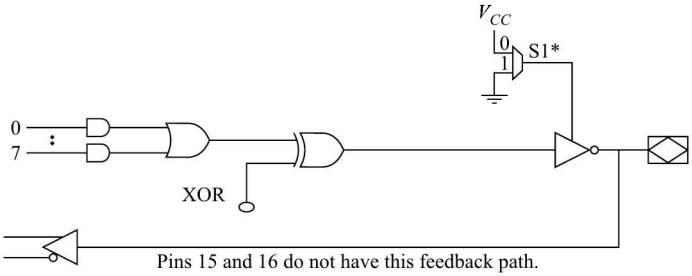


Fig. 12.29 ***Macrocell Configuration for Simple Mode***

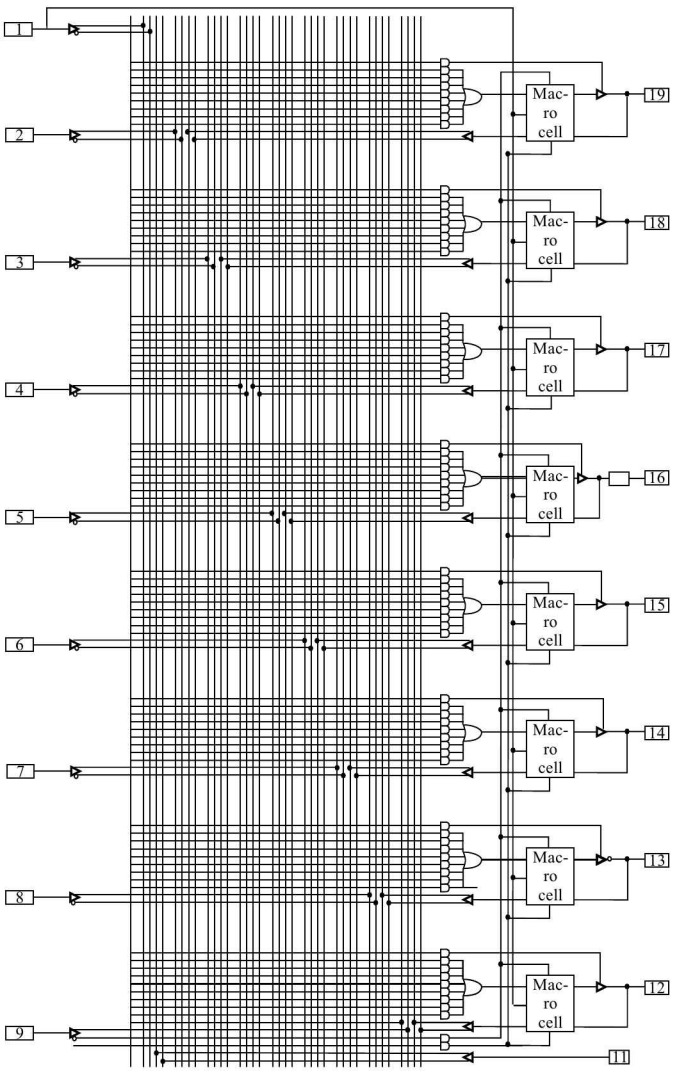


Fig. 12.30 ***18CV8 PEEL Device***

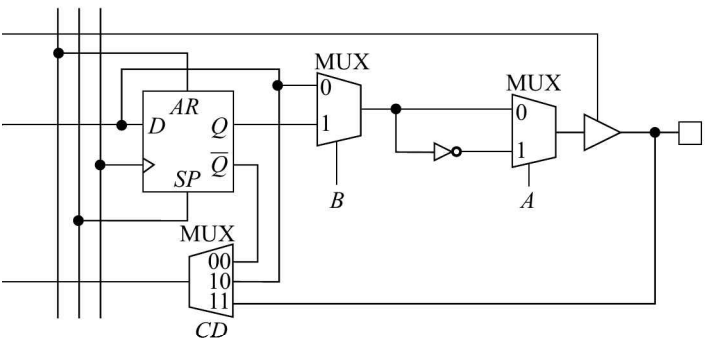
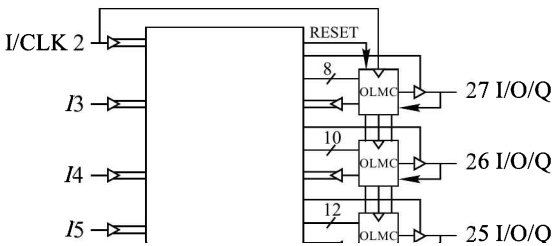


Fig. 12.31 ***Macrocell Details of 18CV8***

eatures and modes of GAL 16V8 and GAL 20V8 are similar to the 16V8 and cussed earlier. The isp GAL 22V10 has some additional features in comparison manufacturers. These are discussed below.

ically erasable
ully function,
mpatible with
2V10 devices.
100 ms) which
figure quickly



Once the function is programmed, the non-volatile E^2CMOS cells will not loose the pattern even when the power is turned off.

The GAL devices and other EEPLDs can be used for DMA control, state machine control, and high-speed graphics processing etc.

12.4.6 EX-OR PALs

The use of EX-OR gates in PALs to implement fuse configurable output polarity has already been discussed. EX-OR gates are also used in AND-OR-EX-OR configuration as shown in Fig. 12.33 in the EX-OR PAL device. Here, the EX-OR outputs are fed to the inputs of the FLIP-FLOPs. Each of these EX-OR gates is fed in turn by two sum-of-products arrays of two product terms each. This configuration is used to reduce the amount of logic required for many applications, particularly counters which allow complex designs to be implemented using very few product terms.

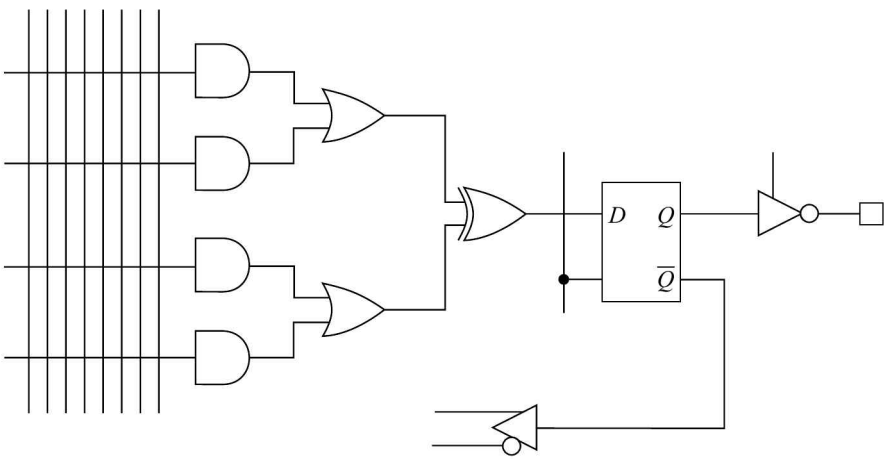


Fig. 12.33 EX-OR *PAL*

There are a number of EX-OR PAL devices available. Figure 12.34 shows the logic diagram of EX-OR PAL 20 × 10A. It is an EX-OR registered 24-pin TTL programmable logic device. It has EX-OR gates preceding each FLIP-FLOP. The EX-OR gate combines two sum terms, each composed of two product terms as shown in Fig. 12.33. Similar to EEPLDs and GALs, this EX-OR PAL also has security fuse. After programming and verification, the design can be secured by programming the security fuse. When the security fuse is programmed, the array will be read as if every fuse is intact. Therefore, once programmed, the internal programmed pattern can not be read by a device programmer securing proprietary designs from competitors.

The 20 × 8A and 20 × 4A EX-OR PALs have 2 and 6 combinational outputs respectively in addition to their EX-OR registered outputs.

12.4.7 Available SPLDs

Some of the commercially available SPLDs with their features are given in Table 12.3.

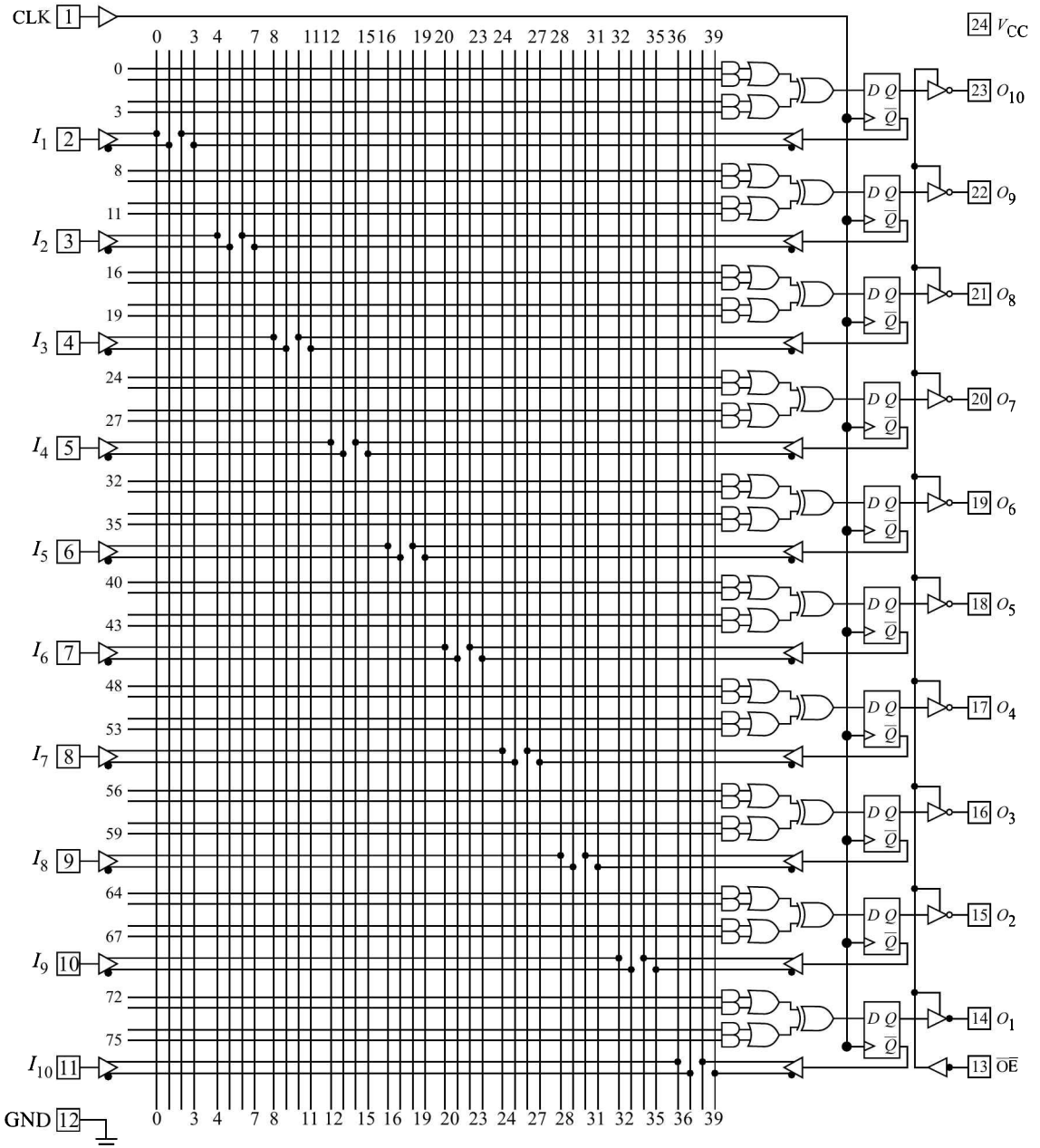


Fig. 12.34 ***Logic Diagram of*** EX-OR ***PAL 20 × 10A***

Table 12.3 ***Some available SPLDs***

Device		Inputs			Outputs				Package pins
		External	Feedback	Total	Bidirectional combinational	Registered	Combinational	Total	
PAL	10L8	10	0	10	0	0	8	8	20
PAL	16L8	10	6	16	6	0	2	8	20
PAL	20L8	14	6	20	6	0	2	8	24
PAL	20L10	12	8	20	8	0	2	10	24
PAL	16R4	8	8	16	4	4	0	8	20
PAL	16R6	8	8	16	2	6	0	8	20
PAL	16R8	8	8	16	0	8	0	8	20
PAL	20R4	12	8	20	4	4	0	8	24
PAL	20R6	12	8	20	2	6	0	8	24
PAL	20R8	12	8	20	0	8	0	8	24
PAL	20X4	10	10	20	6	4	0	10	24
PAL	20X8	10	10	20	2	8	0	10	24
PAL	20X10	10	10	20	0	10	0	10	24
PAL/ GAL/ EEPLD	16V8	8	8	16	8	8	8	8	20
	20V8	12	8	20	8	8	8	8	24
	22V10	12	10	22	10	10	0	10	24
PEEL	18CV8	10	8	18	8	8	8	8	24

12.4.8 Manufacturers of SPLDs

Some of the major manufacturers of SPLDs, alongwith their some of the SPLD products and their WWW locators are given in Table 12.4

Table 12.4 ***SPLD Manufacturers***

Manufacturer	SPLD Products	WWW Locator
Altera	Classic	http://www.altera.com
Atmel	PAL	http://www.atmel.com
Cypress	PAL	http://www.cypress.com
Lattice	GAL	http://www.latticesemi.com
Philips	PLA, PAL	http://www.philips.com
Vantis	PAL	http://www.vantis.com

12.5 COMPLEX PROGRAMMABLE LOGIC DEVICES (CPLDS)

The simple programmable logic devices (SPLDs), such as PALs, EEPLDs, and GALs etc. have limited number of inputs, product terms, and outputs. These devices, therefore, can support up to about 32 total number of inputs and outputs only (see Table 12.3).

For implementation of circuits that require more inputs and outputs than that are available in a single SPLD chip, either multiple SPLD chips can be employed as discussed in Sec. 12.3.8 or more sophisticated type of chip, referred to as *complex programmable logic device* (CPLD) can be used.

The expansion of PLD using multiple SPLD chips have the following disadvantages:

- PC board area requirement increases with the number of chips.
- Connecting wires will result in adverse capacitive effects.
- Power requirement increases with the number of chips.
- The system cost increases.

Another method of increasing the I/O and product terms can be designing of PLDs using the architecture of SPLDs. This approach was discarded by the designers because of the following problems associated with this approach

- increase in capacitive effects
- increase in leakage currents
- decrease in speed
- cost effectiveness

In view of the above difficulties, complex programmable logic devices (CPLDs) were evolved. A CPLD is just a collection of individual PLDs on a single chip and programmable interconnection structure. By using programming methods, the resources available in various PLDs can be shared in different ways to design complex logic functions.

The complexity of any digital IC chip can be specified in terms of number of equivalent 2-input NAND gates. A typical PAL has 8 macrocells, if each macrocell represents about 20 equivalent gates, then the PAL can accommodate a circuit that needs up to about 160 gates. For circuits requiring a very large number of gates, CPLDs having large number of macrocells (say 512 macrocells) can implement circuits of up to about 10,000 equivalent gates. There are a number of manufacturers of CPLDs manufacturing a wide range of products with different features. Some of the major manufacturers of CPLDs are given in Table 12.5.

Table 12.5 ***CPLD Manufacturers***

Manufacturer	CPLD Products	WWW Locator
Altera	MAX 3000, 7000 and 9000	http://www.altera.com
Atmel	ATF, ATV	http://www.atmel.com
Cypress	FLASH 370, Ultra 37000, Delta 39 K, Quantum 38 K	http://www.cypress.com
Lattice	isp LSI 1000 to 8000	http://www.lattice.com
Philips	XPLA	http://www.philips.com
Vantis	MACH 1 to 5	http://www.vantis.com
Xilinx	XC9500, CoolRunner-II	http://www.xilinx.com

12.5.1 Block Diagram

Figure 12.35 gives block diagram of a complex programmable logic device (CPLD). It consists of a number of *PAL-like blocks, I/O blocks*, and a *set of interconnection* wires. The PAL-like blocks are connected to a set of interconnection wires and each block is also connected to an I/O block to which a number of chip's input and output pins are attached.

A PAL-like block usually consists of about 16 macrocells. Each macrocell consists of an AND-OR configuration, an EX-OR gate, a FLIP-FLOP, a multiplexer, and a tri-state buffer. A typical macrocell is shown in Fig. 12.36. Each AND-OR configuration usually consists of 5-20 AND gates and an OR gate

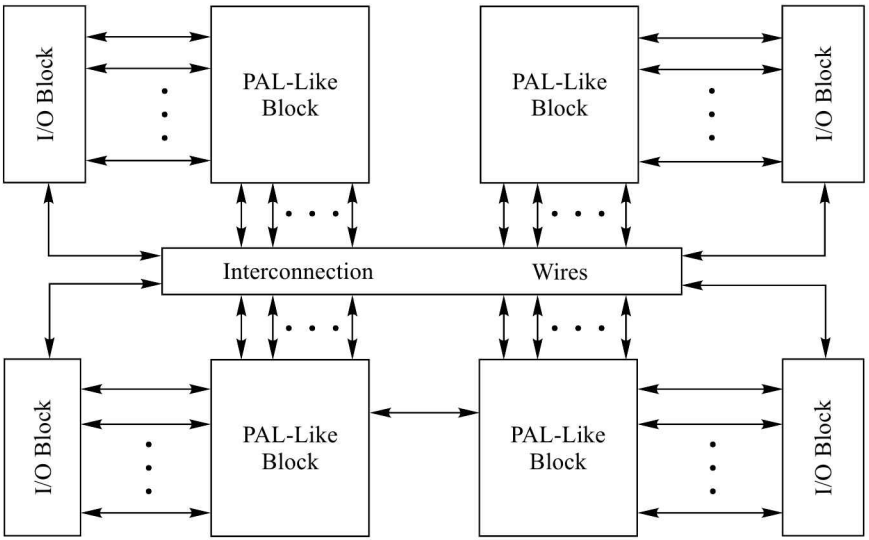


Fig. 12.35 ***Block Diagram of a CPLD***

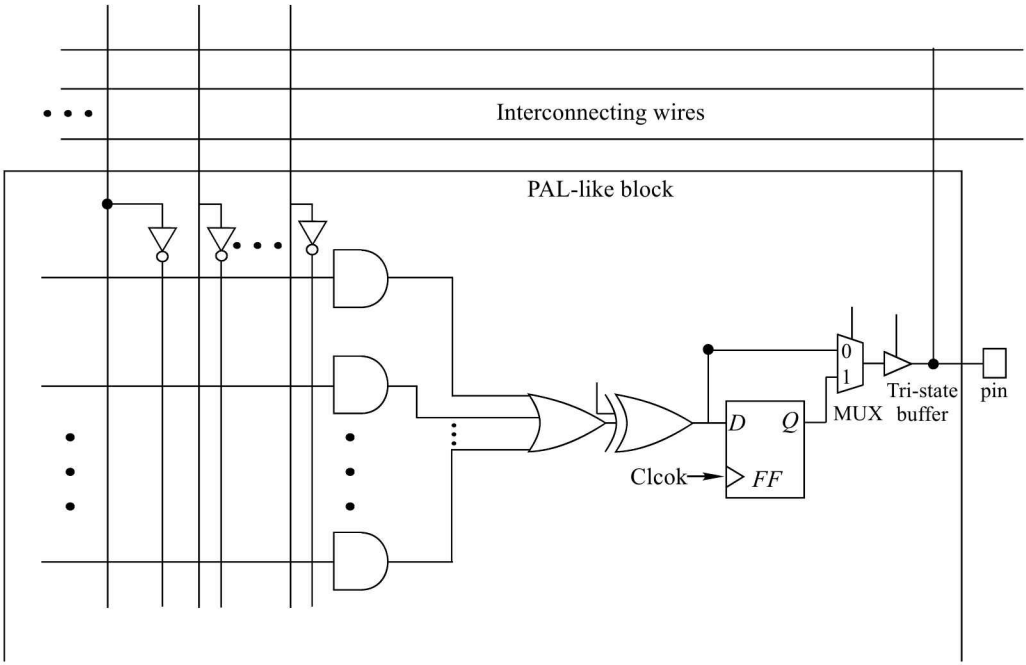


Fig. 12.36 ***A Typical Macrocell of a CPLD***

with 5-20 inputs. An EX-OR gate is used to obtain the output of OR gate in inverted or non-inverted form depending upon its other input being 1 or 0 respectively. A D-FF stores the output of the EX-OR gate, a multiplexer selects either the output of the D-FF or the output of the EX-OR gate depending upon its select

input (1 or 0). The tri-state buffer acts as a switch which enables the chip's pin to be used either as an output (tri-state enabled) or as an input (tri-state disabled). In case the chip's pin is used as an input pin, an external source can drive a signal on to the pin which can be connected to other macrocells using the interconnection wiring. When used as an input pin, the macrocell becomes redundant and it is wasted.

12.5.2 Programming

Programmable logic devices, SPLDs and CPLDs, are implemented using *electrically erasable programmable read-only memory (EEPROM)* technology. These are programmed in the same way as EEPROMs. The SPLD chips have a small number of pins and can therefore be taken out of the circuit board, without much of inconvenience, and put in a programming unit. In the case of CPLDs, instead of relying on a programming unit to configure a chip, it would be very convenient and advantageous if it is possible to perform the programming with the chip remaining attached to the circuit board itself. This method of programming is known as *in-system programming* (ISP). There are two main reasons for employing ISP.

- CPLDs have large number of pins (may even exceed 200) on the chip package, and these pins are fragile and easily bent.
- A socket is required to hold the chip in a programming unit. For large CPLDs the packages used are very expensive, sometimes more expensive than the CPLD device itself.

For the reasons mentioned above, CPLD devices usually support the ISP technique. For programming SPLDs and CPLDs a large number of programmable switches are required to be configured, hence it is not practically feasible for a user of these chips to specify manually the desired state of each switch. For this purpose computer-aided design (CAD) systems are employed. Once the user has completed the design of a circuit using CAD tools, a *programming file* or *fuse map* is generated, that specifies the state of each switch in the target PLD required to realize the designed circuit. A computer system that runs the CAD tools is connected by a cable to the programming unit. In the case of the ISP technique a small connector is included on the printed circuit board (PCB) that houses the CPLD and the computer system is connected by a cable to this connector. The programming involves transferring the programming file generated by the CAD system from the computer into the CPLD through this cable. The circuitry on the CPLD that allows in-system programming has been standardized by the IEEE and is usually called a *JTAG port*. The abbreviation JTAG stands for Joint Test Action Group. It uses four wires to transfer information between the computer and the device being programmed. Figure 12.37 illustrates the use of a JTAG port for programming two CPLDs on a circuit board.

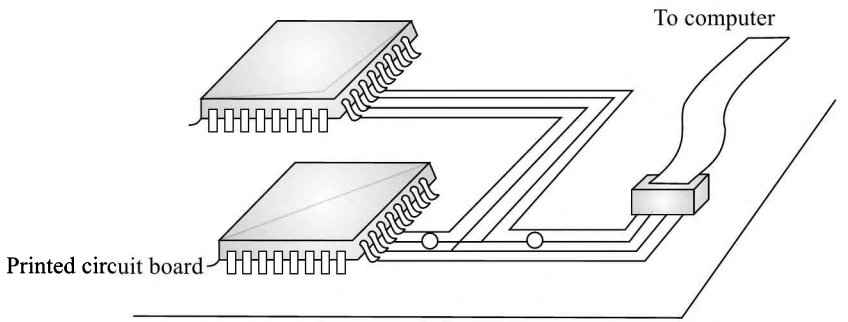


Fig. 12.37 ***JTAG ISP Programming***

12.5.3 Packaging

CPLDs have a large number of pins, making it impractical to use dual-in-line packaging (DIP). Some of the commonly used packages for CPLDs are:

Plastic-Leaded Chip Carrier (PLCC) A PLCC package has pins on all the four sides that 'wrap around' the edges of the chip, rather than extending straight down as in the case of a DIP. The IC socket of PLCC is soldered to the PCB, and the chip is held in the socket by friction. Figure 12.38 illustrates a PLCC package with socket.

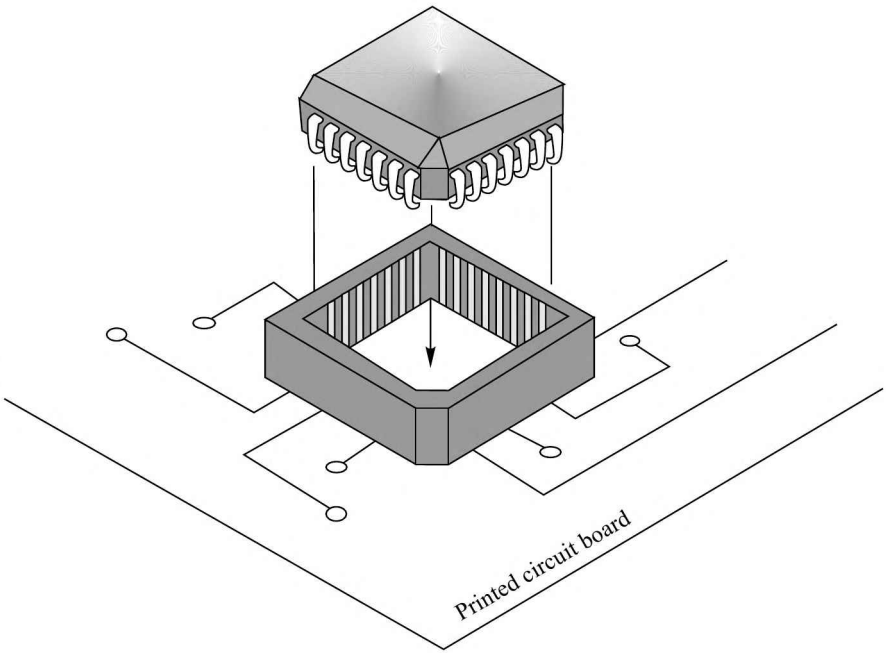


Fig. 12.38 ***A PLCC Package with Socket***

Quad flat pack (QFP) A QFP package also has pins on all four sides like a PLCC package, but with pins extending outward from the package with a downward-curving shape as shown in Fig. 12.39. The QFP's pins are much thinner than those on a PLCC, making it suitable for supporting a larger number of pins. QFPs are available with more than 200 pins, whereas PLCCs are limited to fewer than 100 pins. Some of the varieties of QFPs available are: Plastic quad flat pack (PQFP), power quad flat pack (RQFP), and 1.0 mm thin quad flat pack (TQFP).

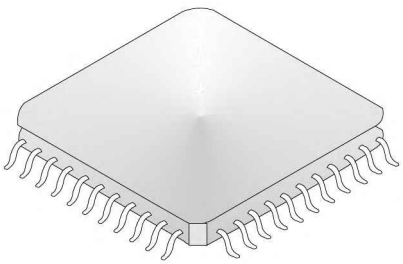


Fig. 12.39 ***A QFP Package***

Ceramic pin grid array (PGA) It has pins extending straight outwards from the bottom of the package in a grid pattern. It can accommodate a few hundred pins in total. Figure 12.40 illustrates bottom view of a PGA package.

The ball grid array (BGA)
grid array (PGA) packaging
round balls, instead of posts.
are very small, hence more
ackage.

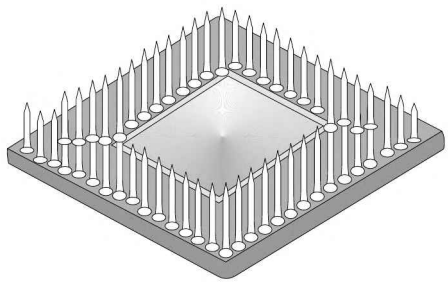


Fig. 12.40 ***Bottom View of a PGA Package***

Ds

lable from various manufac-
ated using CMOS EEPROM
ent features of ALTERA's
) A, MAX 7000, and MAX
The CoolRunner-II CPLD family of Xilinx is discussed below. Data sheets for
es manufactured by various manufacturers can be consulted for details.

'ERA CPLDs

	MAX 3000 A Device Family	MAX 7000 Device Family	MAX 9000 Device Family
	600-10,000	600-5,000	6,000-12, 000
	32-512	32-256	320-560
	2-32	2-16	
	34-208	36-164	168-216
	4.5-7.5	6-12	10-15

Table 12.7 ***CoolRunner-II CPLD Family Packages and I/O Count***

	XC2C32A	XC2C64A	XC2C128	XC2C256	XC2C384	XC2C512
QFG32*	21	–	–	–	–	–
VQ44	33	33	–	–	–	–
VQG44*	33	33	–	–	–	–
QFG48*	–	37	–	–	–	–
CP56	33	45	–	–	–	–
CPG56*	33	45	–	–	–	–
VQ100	–	64	80	80	–	–
VQG100*	–	64	80	80	–	–
CP132	–	–	100	106	–	–
CPG132*	–	–	100	106	–	–
TQ144	–	–	100	118	118	–
TQG144*	–	–	100	118	118	–
PQ208	–	–	–	173	173	173
PQG208*	–	–	–	173	173	173
FT256	–	–	–	184	212	212
FTG256*	–	–	–	184	212	212
FG324	–	–	–	–	240	270
FGG324*	–	–	–	–	240	270

*The letter "G" as the third character indicates a Pb-free package.

macrocells is 32 and the largest number is 512. The maximum I/O count varies from 33–270. These devices are fabricated using 0.18 micron CMOS technology and are industry's very fast low power consumption CMOS CPLDs optimised for 1.8 V systems.

Architecture

Figure 12.41 shows architecture of the CoolRunner-II CPLD family. There are a number of *function blocks* (FBs) each containing 16 macrocells (MCs). The FBs are interconnected with *advanced interconnect matrix* (AIM). The FBs use programmable logic array (PLA) configuration which allows all product terms to be routed and shared among any of the macrocells of the FB. The BSC path is the *JTAG boundary scan control path*. The BSC and ISP block has the JTAG controller and *in-system programming* (ISP) circuits.

Function Block

Figure 12.42 shows the block diagram of a *function block* (FB) of the CoolRunner-II CPLD family. All the FBs are identical and have 40 entry sites for signals to arrive for logic creation and connection. The internal logic engine is a 56 product term (p-term) PLA. The CoolRunner-II CPLD family uses PLA which has many advantages over other CPLDs which normally use PAL structure. Most of these PAL based CPLDs rely on capturing unused p-terms from neighbouring macrocells to expand their p-terms tally, when needed. Therefore, this type of architecture gives variable timings for capturing different p-terms and also may not be using the available unused logic within the FB. The PLA based FB has the following options available:

- Any p-term can be attached to any OR gate inside the FB macrocells.
- Any logic function can have as many p-terms as needed within the FB, subject to an upper limit of 56.

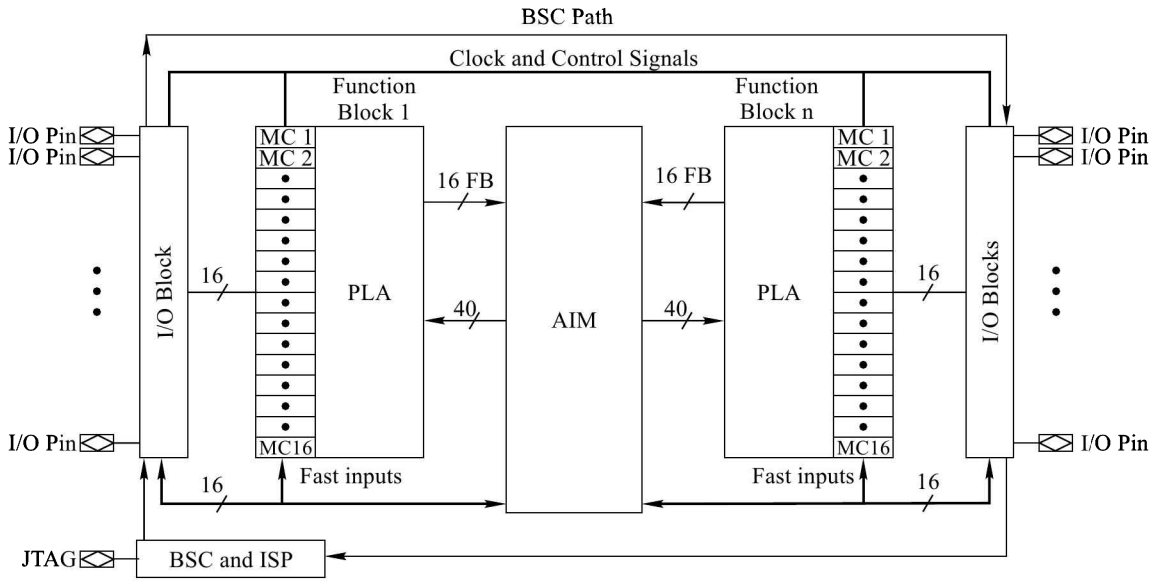


Fig. 12.41 ***CoolRunner-II Architecture***

- p-terms can be re-used at multiple macrocells OR functions so that within a FB a particular logic product need only be created once, but can be used upto 16 times (number of MCs in a FB) within the FB.

Therefore, the advantages of using PLA based CPLDs are:

- Product terms can be shared by macrocells in a FB and as many functions as possible can be placed into FBs by the software.
- The functions need not share a common clock, common set/reset, or common output enable which allows full use of the PLA.
- Every p-term arrives with the same time delay incurred. Therefore, there are no cascade time adders for putting more p-terms in the FB.
- When all the p-terms have been used in the FB, then additional logic can be created by routing signals to another FB. This puts a small interconnecting timing penalty. The Xilinx design software handles all this automatically.

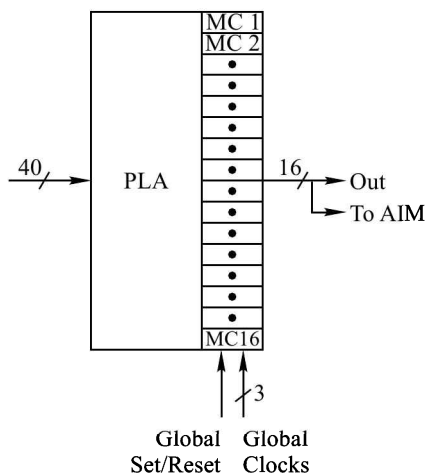
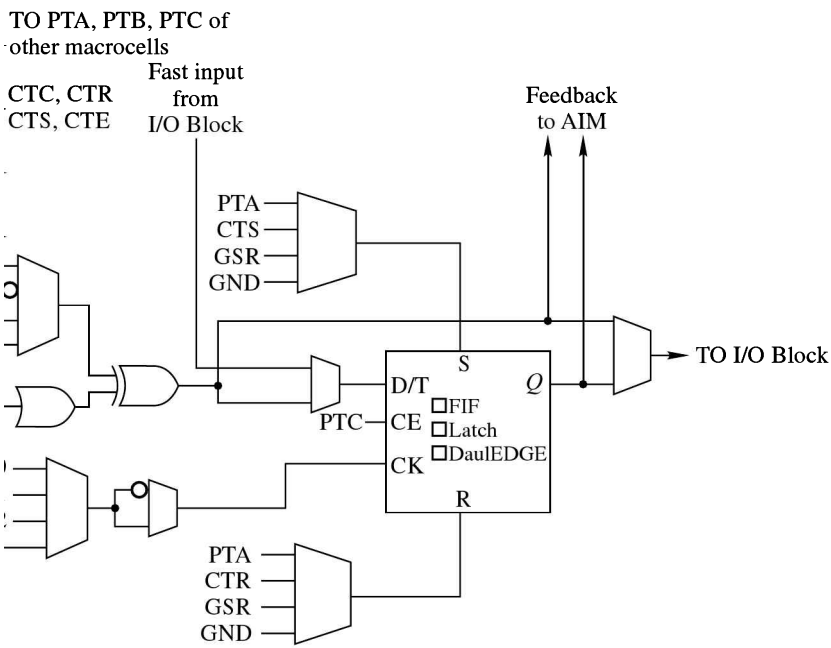


Fig. 12.42 ***CoolRunner-II Function Block***

Macrocell

The CoolRunner-II CPLDs macrocell is shown in Fig. 12.43. It consists of PLA, a FLIP-FLOP, number of multiplexers, and an EX-OR gate. The FLIP-FLOP has a clock enable and can be clocked on either edge


TO PTA, PTB, PTC of
other macrocells
Fast input
from
I/O Block
CTC, CTR
CTS, CTE
Feedback
to AIM
PTA
CTS
GSR
GND
S
D/T
Q
TO I/O Block
FIF
PTC
CE
Latch
DaulEDGE
CK
R
PTA
CTR
GSR
GND

ıed are marked X's.
:erms) are fed to the
the outputs of the
ıe macrocell output
:A of Fig. 12.44 is
2.9 combined in one
\, the outputs of the

Output
$I_1 \cdot I_2 + I_{40}$
$I_1 \cdot I_2 + I_{40}$

Not used
ıts corresponding to

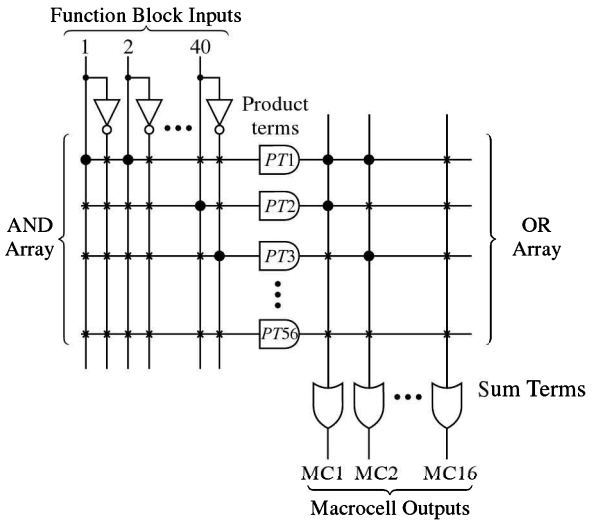


Fig. 12.44 ***PLA Architecture of CoolRunner-II Family***

ıt signals and their
wn in consolidated
:ly, the p-terms have
:ed form. The top AND corresponds to 49 p-terms (PT1 through PT49), the next
s (PT50 through PT53), and the remaining 3 p-terms (PT54, PT55, PT56) are
:C respectively. The four p-terms PT50 through PT53 correspond to four CT
ım, the intersections are marked as small dots (•) that are programmable.

From the conditions (iii) and (iv), we observe that the OR gate is avoided and therefore, it becomes a high speed path. In this way, p-term passes through the EX-OR gate rather than sum term which saves typically 0.3 ns delay time of the OR gate. This path is especially useful for microprocessor address decoding for fast operation. This approach can also be used to build faster shift registers, counters, and some simple state machines.

Design Security

In CoolRunner-II CPLD designs can be secured during programming using four independent levels of security provided on-chip. This eliminates any electrical or visual detection of configuration patterns, which prevents either any accidental overwriting or pattern theft via readback. The security bits programmed can be reset only by erasing the entire chip.

In-system programming

All CoolRunner-II CPLD parts are 1.8 V in-system programmable. They derive their programming voltage and currents from the 1.8 V V_{CC} on the part.

12.6 FIELD-PROGRAMMABLE GATE ARRAY (FPGA)

The programmable logic devices(SPLDs and CPLDs) are based on similar basic architecture—the programmable array logic (PAL) or the programmable logic array (PLA). Over the years, programmable arrays have increased in size and complexity, and highly configurable output macrocells have been added to enhance their flexibility and expandability. To increase the effective size and to add more functionality in a single programmable device, alternative architectures have been developed which are known as *field-programmable gate arrays* (FPGAs). The logic densities of FPGAs are much higher than those of CPLDs. They range in size from a few thousands to hundreds of thousands equivalent gates. From modern standards digital circuits with hundreds of thousands of gates is not too large. FPGA devices support implementation of relatively large complex logic circuits.

The FPGAs do not contain AND, OR planes, instead they provide logic blocks for implementation of the required digital functions.

An FPGA is composed of a number of relatively independent configurable logic blocks (CLBs), configurable I/O blocks, and programmable interconnection paths (known as routing channels). All the resources of the device are uncommitted and that these must be selected, configured and interconnected by a user to form a logic circuit for his application. The basic architecture of an FPGA is shown in Fig. 12.45.

There are a number of manufacturers of FPGA devices. The various families of FPGAs manufactured by different manufacturers differ primarily in the number of logic modules (from few hundreds to hundreds of thousands), supply voltage range, power consumption, speed, architecture, process technology, number of pins, and type of packages, etc. Some of the major manufacturers of FPGAs manufacturing a wide range of products to suit various types of requirements are given in Table 12.8.

The basic FPGA architecture consists of an array of configurable logic blocks (CLBs). The logic blocks are surrounded by configurable input/output blocks. There are rows and columns of programmable interconnection paths. The I/O blocks can be individually configured as input, output, or bidirectional.

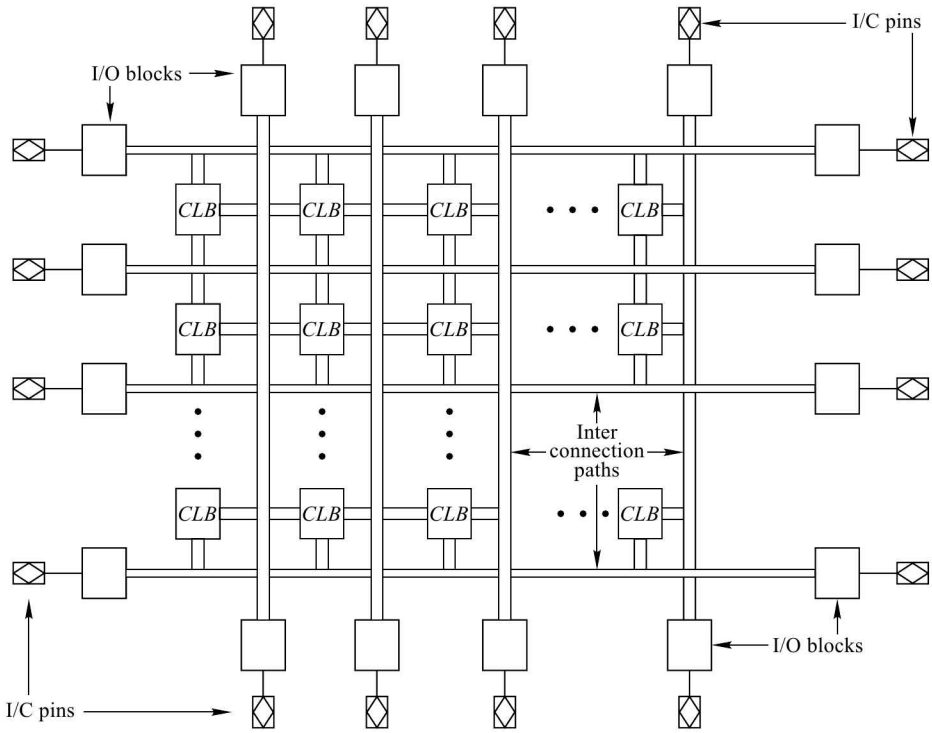


Fig. 12.45 ***Basic Architecture of FPGA***

Table 12.8 ***FPGA Manufacturers***

Manufacturer	FPGA products	www locator
Actel	Act 1, 2, 3, IGLOO	http://www.actel.com
Altera	Flex 6000, 8000, 10K, APEX 20K, Stratix II, III, IV	http://www.altera.com
Atmel	AT6000, AT40K	http://www.atmel.com
Lucent	Lattice SC, ECP2, XP2	http://www.lucent.com
Quick Logic	PASIC 1, 2, 3, Eclipse	http://www.quicklogic.com
Vantis	VF1	http://www.vantis.com
Xilinx	XC4000, XC5200, Virtex, Spartan	http://www.xilinx.com

Configurable Logic Blocks

There are a number of configurable logic blocks (CLBs) in an FPGA organized as an array of rows and columns. The logic blocks are connected to the I/O blocks through common row/column programmable interconnects. The common row/column interconnects are known as global interconnects. A logic block consists of a number of logic modules (LMs). The logic modules are the basic logic elements in an FPGA. The logic modules within a CLB are connected through local programmable interconnects. Figure 12.46 shows a CLB.

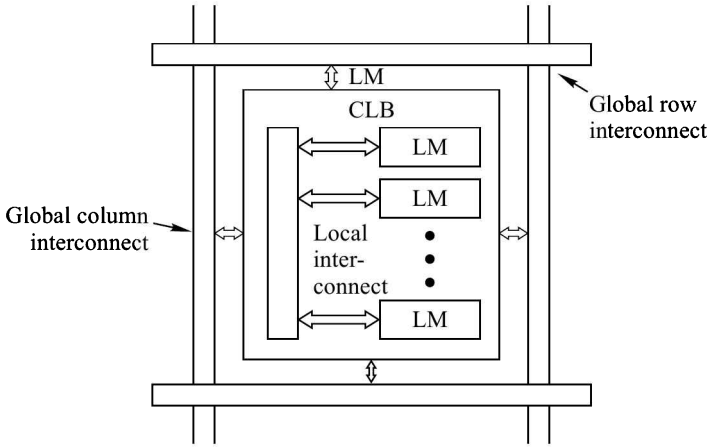


Fig. 12.46 ***Basic Configurable Logic Block***

Logic Module

A logic module consists of an LUT (look-up table), a D-type `FLIP-FLOP` and a mutliplexer (MUX). Most of the FPGAs are based on 4-input LUT. Figure 12.47 shows a block diagram of a logic module with 4-input LUT. Output of the LUT becomes the output of the logic module either directly or through D-type `FLIP-FLOP`. Thus, the output can be configured for combinational or registered (i.e., through `FLIP-FLOP`).

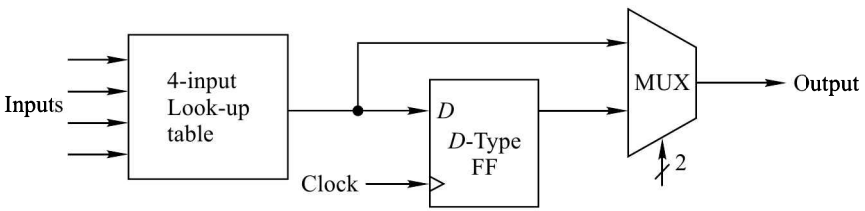


Fig. 12.47 ***Block Diagram of a Logic Module***

Look-Up Table (LUT)

An LUT (look-up table) consists of a programmable memory and it can be used to generate logic function in SOP form. For example, from Table 11.4, we can see that this table is similar to a 4-input and one output logic circuit's truth table. Therefore, a memory can generate canonical product terms. Figure 12.48 shows a block diagram of an LUT. It consists of a memory and a multiplexer (MUX). Let us assume the memory contents as given in the figure. Since, it is an 8-bit memory, therefore an 8 : 1 multiplexer is required. If the 3-bit logical input is $A_2 A_1 A_0$, then

$$Y = \overline{A}_2 \overline{A}_1 A_0 + \overline{A}_2 A_1 \overline{A}_0 + A_2 \overline{A}_1 \overline{A}_0 + A_2 A_1 A_0$$

The look-up table in most of the commercially available FPGAs is 4-input circuit. Larger LUTs would allow for more complex logic to be performed per logic block, thus reducing the wiring delay between blocks as fewer blocks would be needed. This will require larger multiplexer and an increased chance of waste if all of the functionality of the larger LUTs were not to be used. On the other hand, smaller look-up tables

may require a design to consume a large number of logic blocks, thus increasing wiring delay between blocks while reducing per logic block delay. Therefore, 4-input LUT structure makes the best trade-off between area and delay for a wide range of circuits. However, some of the latest FPGAs have been designed with 6-input LUT structure. The Xilinx Virtex-5 family of FPGAs have used 6-input LUT technology.

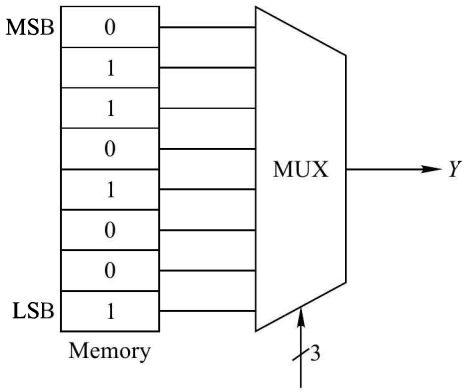


Fig. 12.48 ***Block Diagram of an LUT***

FPGA Cores

A commercially available FPGA may have all the CLBs available for a user to program them according to his requirements. However, some FPGAs are available in which a portion of CLBs is used by the manufacturer to provide a specific built-in function that can not be changed by a user. This is referred to as *hard-core* logic. The hard-core logic approach has the following advantages:

- The hard-core logic may normally be implemented using lesser number of CLBs than the same logic being programmed by a user. This saves the available chip resources, i.e., programmable area to the user.
- There is saving in the development time of user in developing a digital system.
- The built-in hard-core function can be thoroughly tested by the manufacturer, thereby increasing its reliability.

Some of the commonly used functions, such as microprocessors, standard I/O interfaces, and digital signal processors (DSPs) are available in hard-core FPGAs. Since the hard-core designs are developed by the manufacturers', therefore, these are the manufacturers' *intellectual property* (IP).

In case, the manufacturer's programmed function has some programmable features also, it is known as a *soft-core* function. Some intellectual properties may be combination of both hard-core and soft-core functions. The FPGAs containing either or both hard-core and soft-core embedded processors and other functions are known as the *platform FPGA* because they can be used to implement an entire system without the need for any external devices.

FPGA Process Technology

There are different process technologies used by various FPGA manufacturers. These are:

- SRAM technology—It is based on static memory technology. The CMOS devices are fabricated by this technology and these are in-system programmable (ISP) and are reprogrammable.
- Antifuse technology—The antifuse technology developed by Actel Corporation of America is used for processing CMOS FPGAs which are one-time programmable (OTP).
- EPROM technology—It may be one-time programmable (OTP) or ultraviolet erasable type of CMOS device.
- EEPROM technology—It is electrically erasable which may be in-system programmable type or off-system programmable type CMOS technology.
- Flash technology—It is flash-erase CMOS technology which may be in-system programmable type.
- Fuse technology—It is a bipolar one-time programmable FPGA.

Configuration Memory

Most of the modern FPGAs use SRAM. The SRAM bits are used to hold the user defined configuration values.

12.6.1 Xilinx Virtex FPGAs

The field-programmable gate array was invented by Xilinx in 1984 and they are the leading manufacturers of FPGA devices. There are two major lines of Xilinx FPGAs, Spartan and Virtex. The Extended Spartan-3A and the Virtex-5 families are the latest available FPGAs from Xilinx.

Configurable Logic Blocks

There are a number of CLBs organised as an array. Each CLB contains multiple basic logic units called logic cells (LCs). The logic cells are same as the logic modules (LMs) discussed earlier. The logic cells are LUT based. Each logic cell consists of an LUT, a `FLIP-FLOP`, and a multiplexer. In the Virtex-4 family, XC4VLX200 FPGA has CLB array of 192X116 containing 200,448 logic cells. The number of LUTs and the internal registers, i.e., `FLIP-FLOPs` is 178,176 each. In the Virtex series of FPGAs there is a concept of *slice*. A CLB of Virtex-4 family of FPGAs consists of 89,088 slices. There are two LUTs and two FFs in each slice. A CLB is made up of four slices.

The Virtex-5 family of FPGAs has a maximum of 240 × 108 array of CLBs, 51,840 slices, each slice contain four LUTs and four FFs. A CLB of Virtex-5 family FPGAs is made up of two slices. The function generators are configurable as 6-input LUTs or dual-output 5-input LUTs.

In addition to function generators and storage elements (FFs), each slice in both of the above FPGA families, contain arithmetic logic gates, large multiplexers, and fast carry look-ahead chain.

Configuration

The devices of Virtex family are configured by loading the bitstream into internal configuration memory in various modes.

IP Cores

In these devices, there are IP cores for commonly used complex functions including DSP, bus interfaces, processors, and processor peripherals.

Process Technology

The Virtex-4 family of FPGAs are produced using 90-nm copper CMOS process technology, whereas the Virtex-5 family of FPGAs are produced using 65-nm copper CMOS process technology.

12.6.2 Altera Stratix FPGAs

Altera Corporation of America is producing wide range of FPGAs to meet different requirements. The Stratix series of FPGAs started in 2002 with the introduction of Stratix family. Subsequently, Stratix GX(2003), Stratix II (2004), Stratix II GX (2005), Stratix III (2006) and Stratix IV (2008) were introduced in the years mentioned in parentheses along the family. The basics of Stratix II family of FPGAs have been chosen for discussion here.

Stratix FPGAs contain a two dimensional row- and column-based architecture to implement custom logic. A series of column and row interconnects of varying length and speed provides signal interconnects between logic array blocks (LABs) and various other blocks, such as memory block structures and digital signal processing (DSP).

Logic Array Block (LAB)

The configurable logic block (CLB) is called as logic array block in Altera FPGAs. Each LAB consists of eight adaptive logic modules (ALMs). An ALM is the basic building block of logic for efficient implementation of user logic functions. LABs are grouped into rows and columns across the device. In addition to eight ALMs, each LAB contains carry chains, shared arithmetic chains, LAB control signals, local interconnect, and register chain connection lines. The LAB structure is shown in Fig. 12.49. The local interconnect transfers signals between ALMS in the same LAB. The local interconnect is driven by column and row interconnects, ALM outputs in the same LAB, and neighbouring LABs from the left and right through the direct link connection. The direct link connection feature helps in minimising the use of row and column interconnects which increases the performance and flexibility. Multiple LABs are linked together via the global row and column interconnects.

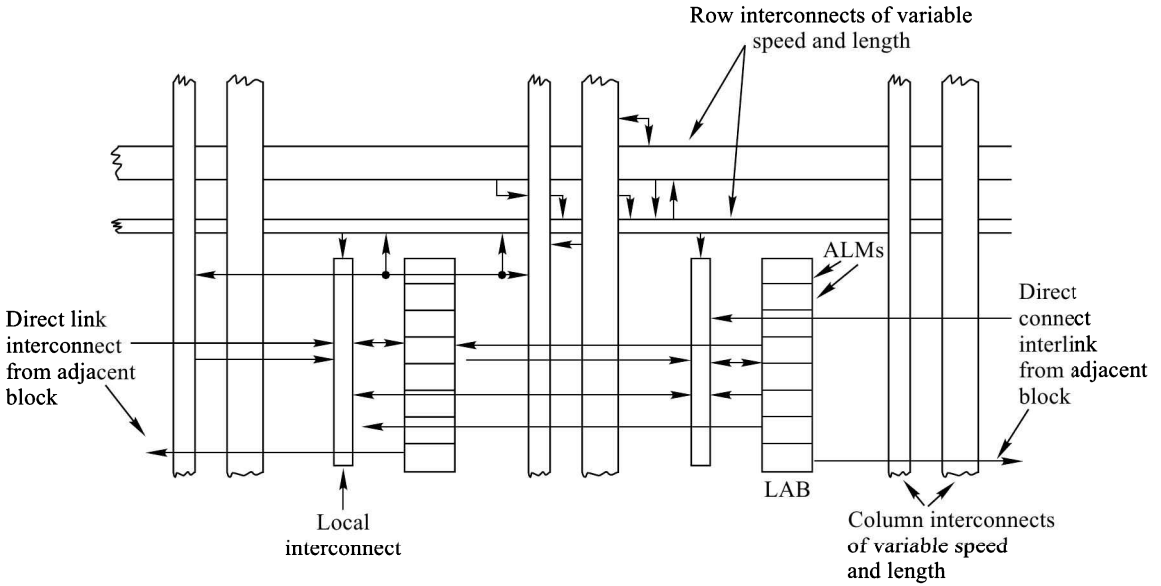


Fig. 12.49 ***Stratix II LAB Structure***

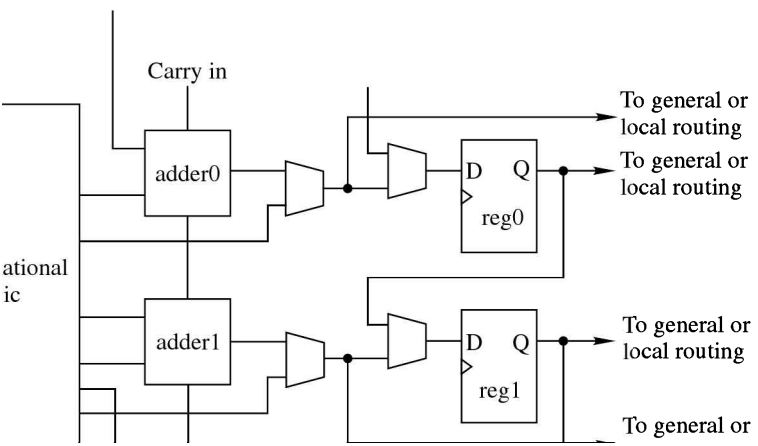
Adaptive Logic Modules (ALMs)

The basic building block of logic in these FPGAs is the adaptive logic module (ALM). Each ALM contains a variety of look-up table (LUT)-based resources that can be divided between two adaptive LUTs (ALUTs). There are eight inputs in an ALM and one ALM can be used to implement various combinations of two functions including any function of upto six inputs and certain seven input functions.

In addition to the two ALUTs, each ALM contains two programmable registers (D-type FFs), two dedicated full adders, a carry chain, a shared arithmetic chain, and a register chain. Using these resources, the ALM can

functions and shift registers. Each ALM drives all types of interconnects: local, / chain, shared arithmetic chain, register chain, and direct connect interlinks. agram of the ALM. The eight data inputs are: data a, data b, data c, data d, data f1. The first four data inputs, data a, data b, data c, and data d can be shared ta e0 and data f0 are dedicated to adder 0 and reg 0, and data e1 and data f1 eg 1.

outputs (combinational and registered) that drives the global and local routing output can be either LUTs output or adder output. For combinational output, e registered output is obtained via the register. The two sets of outputs are



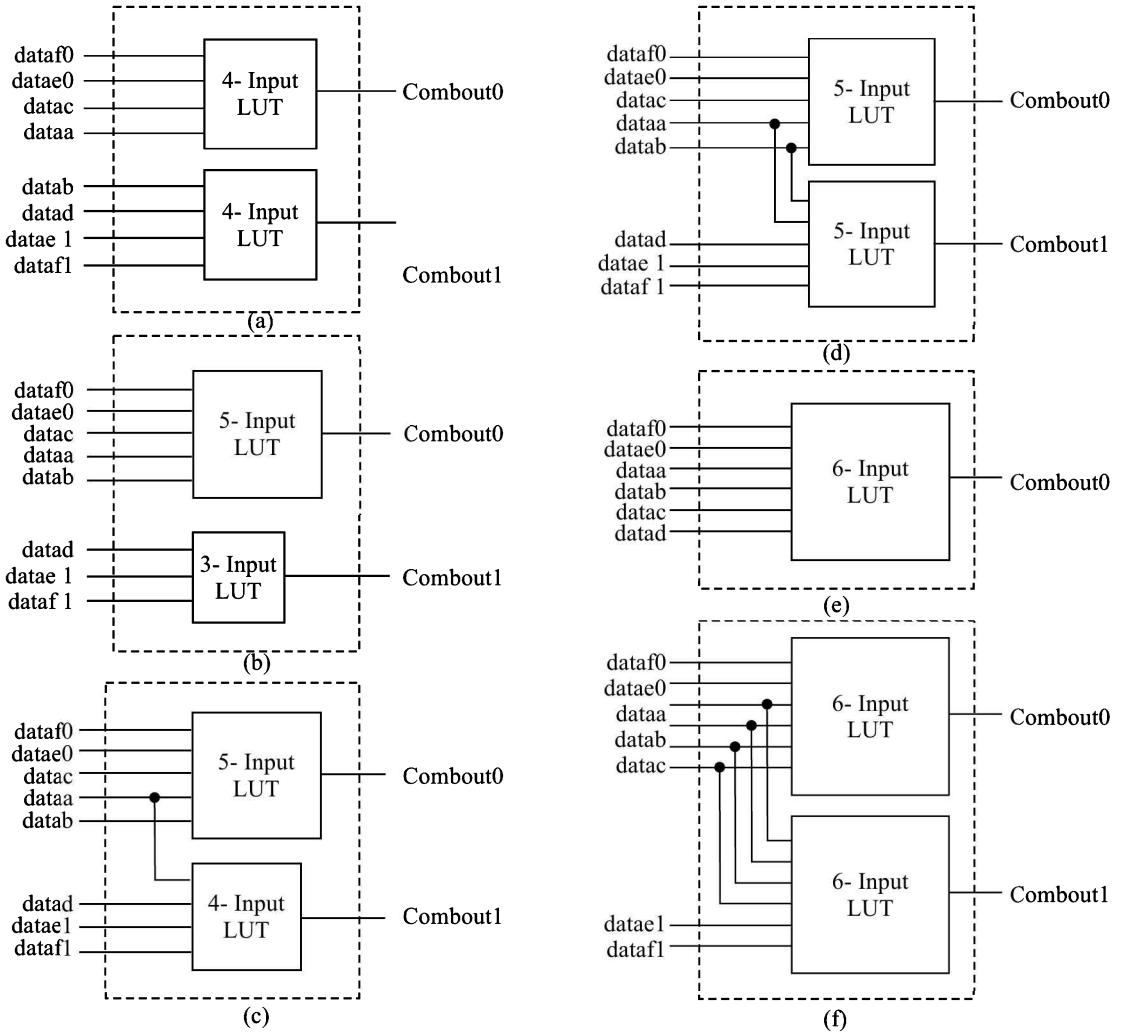


Fig. 12.51 ***Various Combinations of Logic Functions Generated using an ALM (a) Two 4-Input Functions (b) 5 and 3 Input Functions (c) 5 and 4 Input Functions (d) Two 5-Input Functions (e) 6-Input Function (f) 6 and 2 Input Functions***

Extended LUT Mode Some specific seven-input functions can be implemented by making two five input functions, sharing four inputs, and applying these two five-input functions to a 2 : 1 multiplexer. Figure 12.52 shows its implementation.

Arithmetic Mode The arithmetic mode is used for implementing adders, counters, accumulators, comparators, and parity functions. In arithmetic mode, an ALM uses two sets of two four-input LUTs alongwith two dedicated full adders.

Shared Arithmetic Mode In shared arithmetic mode, an ALM can implement a three-input add. In this mode, the ALM is configured with four 4-input LUTs.

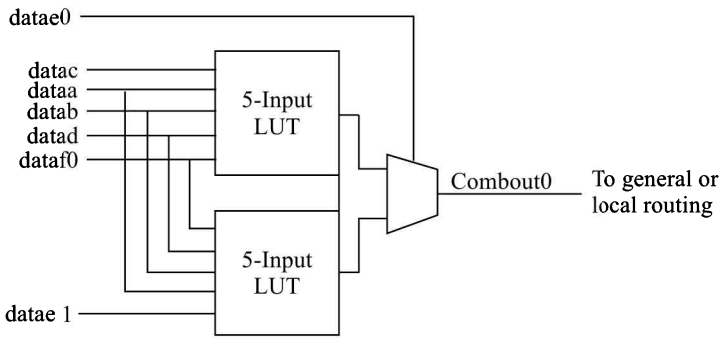


Fig. 12.52 ***Extended LUT Mode of an ALM***

SUMMARY

The basic concepts of programmable logic devices and programmable gate arrays have been introduced. With the development of these devices, it has become possible to design complex digital systems. However, high-level design techniques and computer-aided tools are required to produce efficient PLD and FPGA implementations. Testing of PLD and FPGA implementations also require computer-assisted test tools. The design and test tools for these programmable devices are beyond the scope of this book.

The emergence of these devices has revolutionized the design of digital systems similar to the emergence of microprocessor. The programmable logic concept has emerged as a technology that has given the power to design one's own custom ICs which cannot be copied by others.

The design of embedded systems in small size has become possible by using FPGAs with intellectual properties, such as microprocessors, and DSP, etc. The embedded systems without using external devices and battery operated digital systems have proved very useful because of the availability of low-power FPGAs with intellectual properties.

GLOSSARY

Antifuse A programmable element invented by Actel Corporation named as PLICE (programmable low-impedance circuit element).

ASIC (Application specific integrated circuit) An IC configured by the manufacturer as per the specifications supplied by the user for a specific application.

BGA (Ball grid array) An IC package used for ICs requiring large number of pins. The pins are of small round ball shapes.

CLB (Configurable logic block) A logic block in an FPGA consisting of logic modules and local programmable interconnect that is used to connect logic modules within the CLB to generate required logic functions.

Configuration memory A memory in an FPGA meant to store bit pattern for configuring the FPGA.

CPLD (Complex programmable logic device) A programmable logic device containing a large number of equivalent gates.

ant for general purpose specific functions.

gate array) A programmable logic device containing very large number of

logic array) A PLA programmable by the user.

A type of configurable PAL.

ɔgic provided by the manufacturer in an FPGA for some commonly required ɔurer's intellectual property (IP).

lard-core and soft-core provided by a manufacturer in an FPGA.

ɛ arrangement for providing connections between various logic elements, logic

) A technique for programming large programmable logic devices such as he device is programmed on the circuit board itself rather than in a programming

p) An IEEE standard for ISP.

ame as configurable logic block (CLB).

'ule) A basic unit of logic in an FPGA that usually contains an LUT (look-up xer.

of memory that is programmable for creating the desired logic function in SOP

unction provided in large programmable logic devices which are used to design

S

›f programmable _____ arrays.
tal circuit of 32 variables _____ PLAs with 16 inputs and 8 outputs are

›f programmable _____ gates.
d by the _____.
le security of a digital circuit _____ design is preferred.
circuits of the complexity of a few hundreds of gates _____ is preferred.
ins, the number of inputs to each of the AND gates is _____.
\s _____ programming technique is the most suitable.
ımming uses _____ IEEE standard.
for ICs with more than 200 pins is _____.
_.
›gramming _____ FPGAs.
be used for digital circuits requiring more than 200,000 equivalent gates.
re programmed using _____ tools.
gital circuit ASIC is _____ expensive than designing using FPGA.
and PAL devices have _____ outputs.
orated between the OR gate and output buffer in a registered PAL.
gic device, an _____ is provided for programming the polarity of output.
d using _____ technology.
_____ -system programmable.

PROBLEMS

12.1 Design a BCD-to-Excess-3 code converter using a (a) PROM, (b) PLA, (c) PAL.
12.2 Design an Excess-3-to-BCD code converter using a (a) PROM, (b) PLA, (c) PAL.
12.3 Design a BCD-to-seven segment decoder using a (a) PROM, (b) PLA, (c) PAL.
12.4 How will you obtain 16-bit output word using 82 S100 FPLAs?
12.5 Explain the function of the circuit of Fig. 12.13.
12.6 What is meant by the term 'architecture of a PLD'? Give some suitable examples.
12.7 For 16L8 PAL device shown in Fig. 12.18, find out the input lines (columns) corresponding to the inputs I_1 through I_{10} and IO_2 through IO_7.
12.8 For 16R6 Registered PAL shown in Fig. 12.19, find out the input lines (columns) corresponding to the inputs I_1 through I_8, IO_1, IO_8, and feedback connections corresponding to D-FFs in O_2 through O_7 lines.
12.9 For the output macrocell of 18CV8 PEEL device shown in Fig. 12.31, determine the twelve output configurations for the select inputs $A = 0, 1$; $B = 0, 1$ and $CD = 00, 10, 11$ of the multiplexers.
12.10 In the PAL 16L8 device, program the output as
(a) Always enabled
(b) Always disabled
(c) Enabled by a product term $ABCD\overline{E}\,\overline{F}GH$ of eight variables.
12.11 A 2-input look-up table (LUT) of an FPGA's logic block is shown in Fig. 12.53. Determine its truth table.

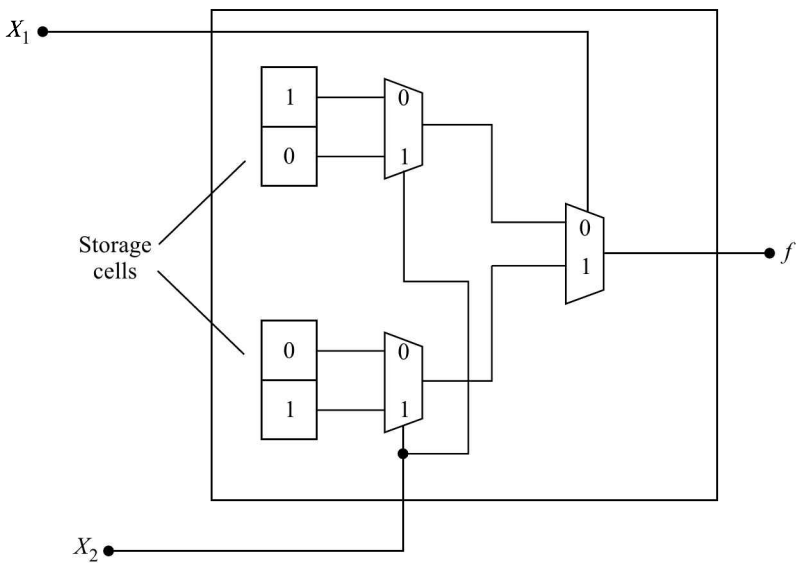


Fig. 12.53

12.12 A 3-input LUT is shown in Fig. 12.54. Determine the bits to be stored in the storage cells to realize the logic function.

$$f = \bar{x}_1\bar{x}_2x_3 + \bar{x}_1x_2\bar{x}_3 + x_1\bar{x}_2\bar{x}_3 + x_1x_2x_3$$

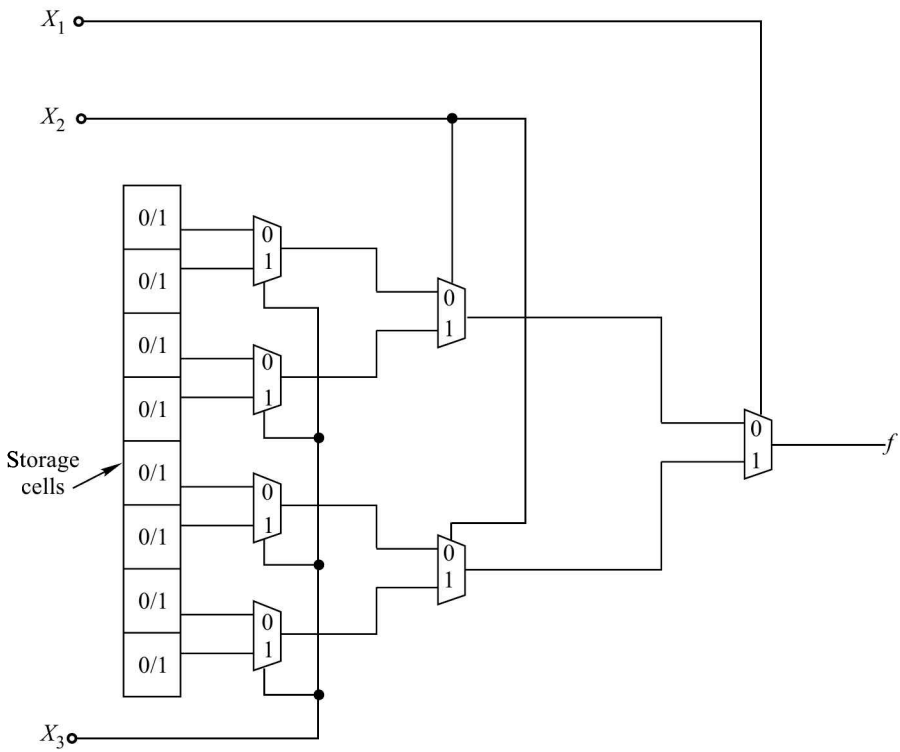


Fig. 12.54