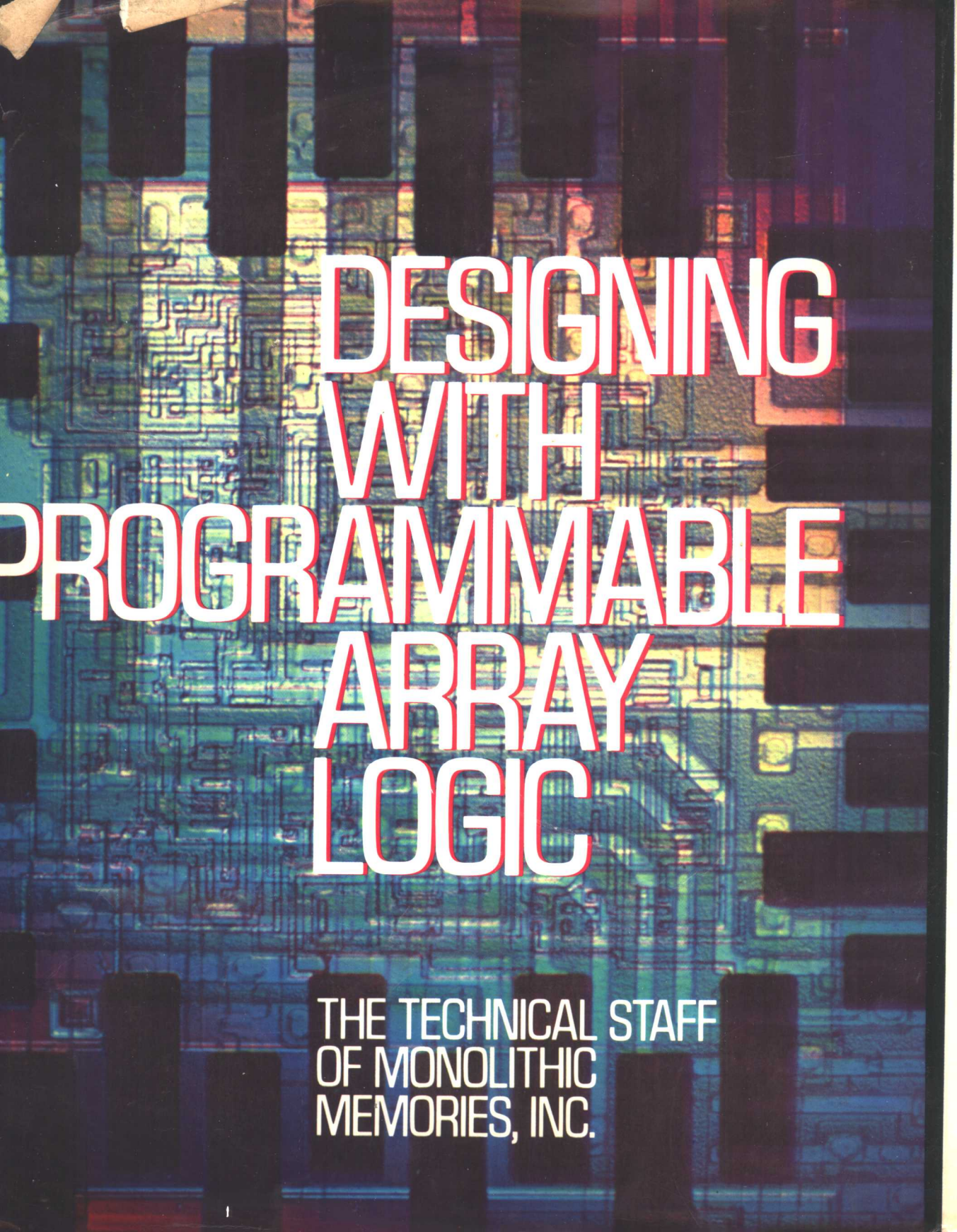
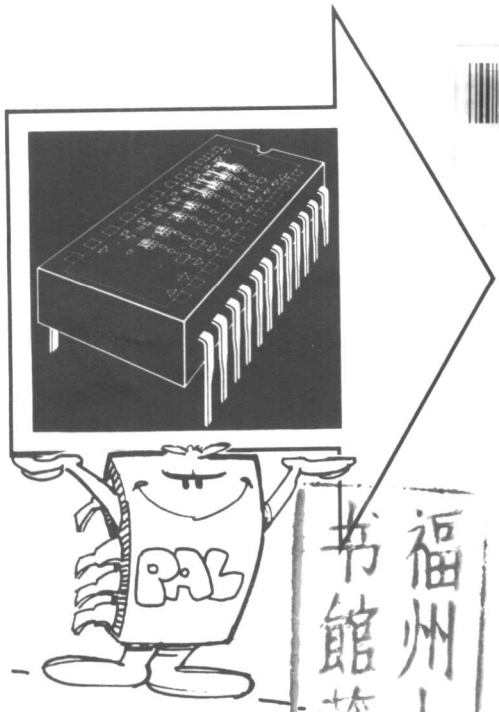

DESIGNING
WITH
PROGRAMMABLE
ARRAY
LOGIC
THE TECHNICAL STAFF
OF MONOLITHIC
MEMORIES, INC.

DESIGNING WITH PROGRAMMABLE ARRAY LOGIC

by the Technical Staff of
Monolithic Memories, Inc.

SECOND EDITION



McGraw-Hill Book Company

New York St. Louis San Francisco Auckland
Bogotá Hamburg Johannesburg London
Madrid Mexico Montreal New Delhi
Panama Paris São Paulo Singapore
Sydney Tokyo Toronto

Monolithic Memories MMI

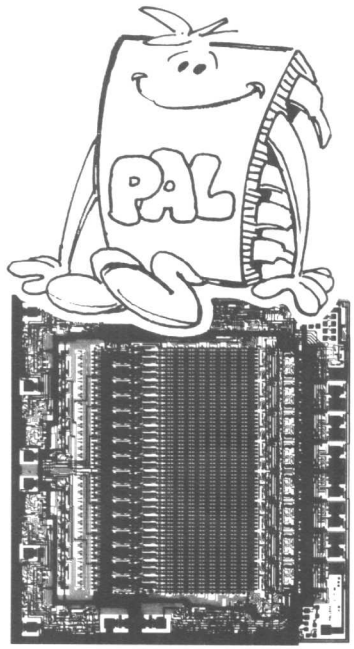
1234567890 DOC DOC 89876543

ISBN 0-07-042723-2

The editors for this book were Harry H. Helms and Ruth L. Weine.

Printed and bound by R. R. Donnelley & Sons Company.

Table of Contents



The PAL™ Concept

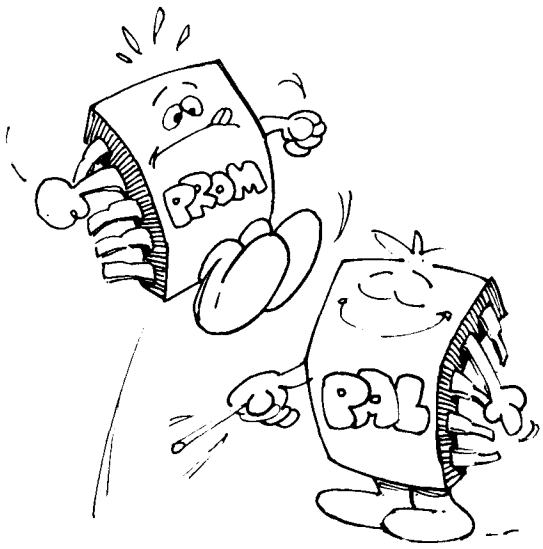
Monolithic Memories' family of PAL devices gives designers a powerful tool with unique capabilities for use in new and existing logic designs. The PAL saves time and money by solving many of the system partitioning and interface problems brought about by increases in semiconductor device technology.

Rapid advances in large scale integration technology have led to larger and larger standard logic functions; single I.C.s now perform functions that formerly required complete circuit cards. While LSI offers many advantages, advances have been made at the expense of device flexibility. Most LSI devices still require large numbers of SSI/MSI devices for interfacing with user systems. Designers are still forced to turn to random logic for many applications.

The designer is confronted with another problem when a low to medium complexity product is designed. Often the function is well defined and could derive significant benefits from fabrication as an integrated circuit. However, the design cycle for a custom circuit is long and the costs can be very high. This makes the risk significant enough to deter most users. The technology to support maximum flexibility combined with fast turn around on custom logic has simply not been available. Monolithic Memories offers the programmable solution.

The PAL family offers a fresh approach to using fuse programmable logic. PALs are a conceptually unified group of devices which combine programmable flexibility with high speed and an extensive selection of interface options. PALs can lower inventory, cut design cycles and provide high complexity with maximum flexibility. These features, combined with lower package count and high reliability, truly make the PAL a circuit designer's best friend.

The PAL—Teaching Old PROMs New Tricks



MMI developed the modern PROM and introduced many of the architectures and techniques now regarded as industry standards. As the world's largest PROM manufacturer, MMI has the proven technology and high volume production capability required to manufacture and support the PAL.

The PAL is an extension of the fusible link technology pioneered by Monolithic Memories for use in bi-polar PROMs. The fusible link PROM first gave the digital systems designer the power to "write on silicon." In a few seconds he was able to transform a blank PROM from a general purpose device into one containing a custom algorithm, microprogram, or Boolean transfer function. This opened up new horizons for the use of PROMs in computer control stores, character generators, data storage tables and many other applications. The wide acceptance of this technology is clearly demonstrated by today's multi-million dollar PROM market.

The key to the PROM's success is that it allows the designer to quickly and easily customize the chip to fit his unique requirements. The PAL extends this programmable flexibility by utilizing proven fusible link technology to implement logic functions. Using PALs the designer can quickly and effectively implement custom logic varying in complexity from random gates to complex arithmetic functions.

ANDs and ORs

The PAL implements the familiar sum of products logic by using a programmable AND array whose output terms feed a fixed OR array. Since the sum of products form can express any Boolean transfer function, the PAL's uses are only limited by the number of terms available in the AND - OR arrays. PALs come in different sizes to allow for effective logic optimization.

Figure 1 shows the basic PAL structure for a two input, one output logic segment. The general logic equation for this segment is

$$\text{Output} = (I_1+\overline{f_1})(\overline{I_1}+\overline{f_2})(I_2+\overline{f_3})(\overline{I_2}+\overline{f_4}) + (I_1+\overline{f_5})(\overline{I_1}+\overline{f_6})(I_2+\overline{f_7})(\overline{I_2}+\overline{f_8})$$

where the "f" terms represent the state of the fusible links in the PAL's AND array. An unblown link represents a logic 1. Thus,

fuse blown, f = 0

fuse intact, f = 1

An unprogrammed PAL has all fuses intact.

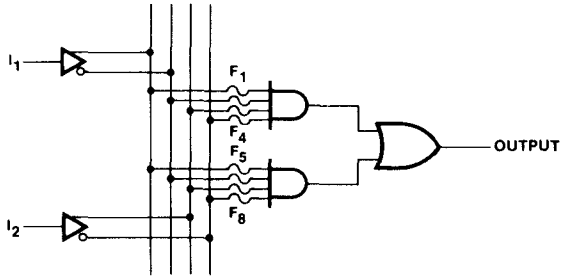


Figure 1

PAL Notation

Logic equations, while convenient for small functions, rapidly become cumbersome in large systems. To reduce possible confusion, complex logic networks are generally defined by logic diagrams and truth tables. Figure 2 shows the logic convention adopted to keep PAL logic easy to understand and use. In the figure, an "x" represents an intact fuse used to perform the logic AND function. (Note: the input terms on the common line with the x's are not connected together.) The logic symbology shown in Figure 2 has been informally adopted by integrated circuit manufacturers because it clearly establishes a one-to-one correspondence between the chip layout and the logic diagram. It also allows the logic diagram and truth table to be combined into a compact and easy to read form, thereby serving as a convenient shorthand for PALs. The two input - one output example from Figure 1 redrawn using the new logic convention is shown in Figure 3.

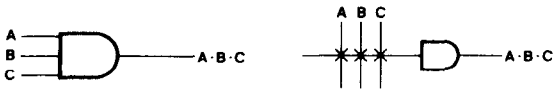


Figure 2

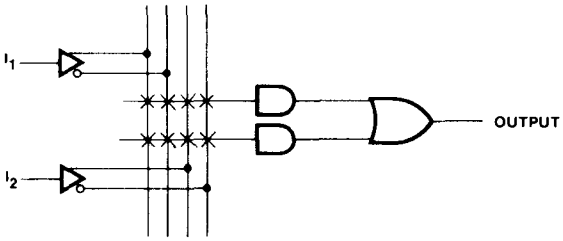


Figure 3

As a simple PAL example, consider the implementation of the transfer function:

$$\text{Output} = I_1\overline{I_2} + \overline{I_1}I_2$$

The normal combinatorial logic diagram for this function is shown in figure 4, with the PAL logic equivalent shown in figure 5.

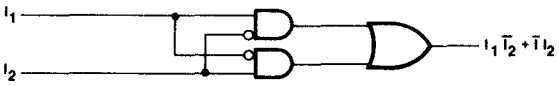


Figure 4

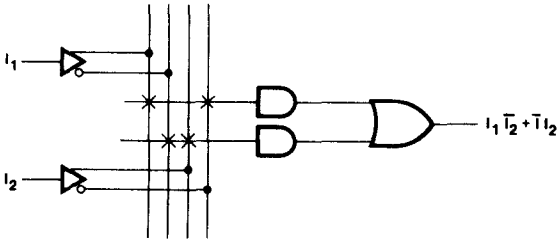


Figure 5

Using this logic convention it is now possible to compare the PAL structure to the structure of the more familiar PROM and PLA. The basic logic structure of a PROM consists of a fixed AND array whose outputs feed a programmable OR array (figure 6). PROMs are low-cost, easy to program, and available in a variety of sizes and organizations. They are most commonly used to store computer programs and data. In these applications the fixed input is a computer memory address; the output is the contents of that memory location.

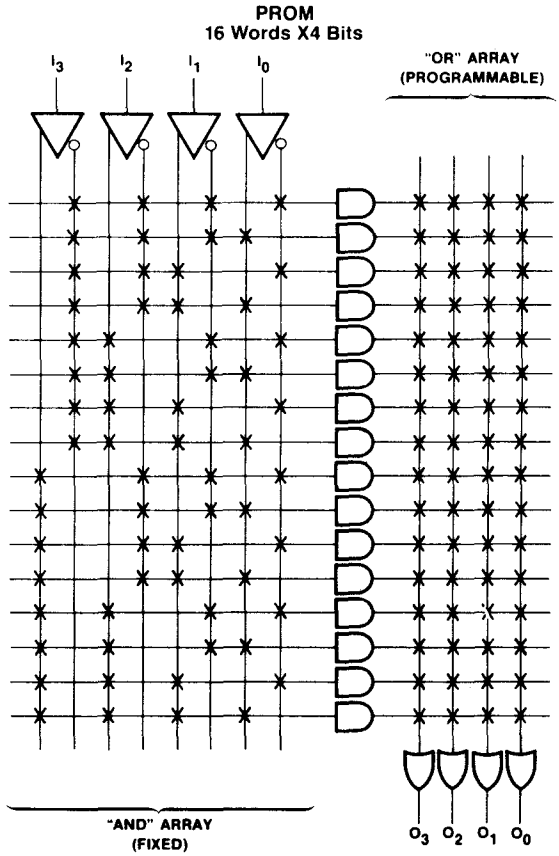


Figure 6

The basic logic structure of the PLA consists of a programmable AND array whose outputs feed a programmable OR array (Figure 7). Since the designer has complete control over all inputs and outputs, the PLA provides the ultimate flexibility for implementing logic functions. They are used in a wide variety of applications. However, this generality makes PLAs expensive, quite formidable to understand, and costly to program (they require special programmers).

The basic logic structure of the PAL, as mentioned earlier, consists of a programmable AND array whose outputs feed a fixed OR array (Figure 8). The PAL combines much of the flexibility of the PLA with the low cost and easy programmability of the PROM. Table 1 summarizes the characteristics of the PROM, PLA, and PAL logic families.

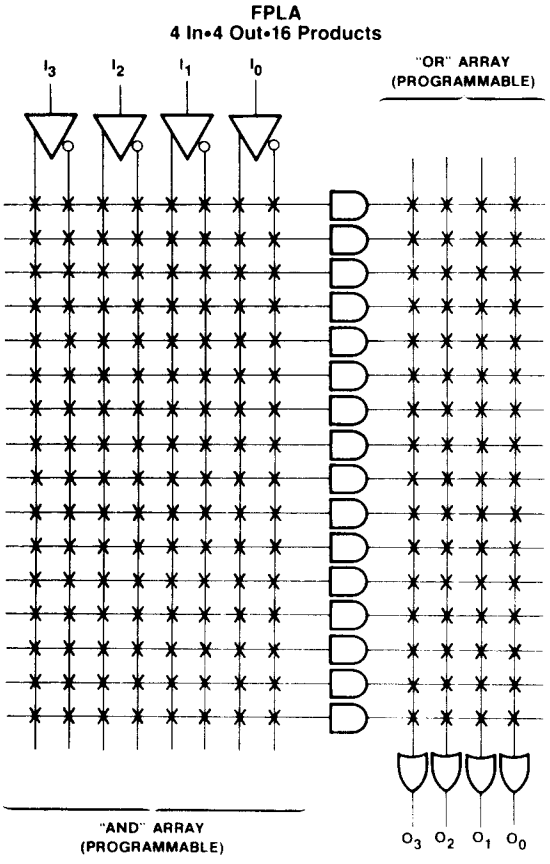


Figure 7

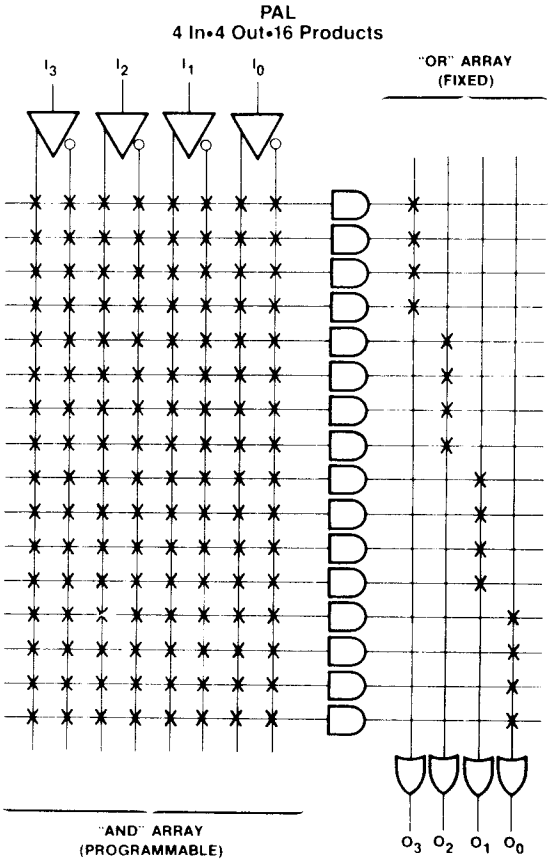


Figure 8

	AND	OR	OUTPUT OPTIONS
PROM	Fixed	Prog	TS, OC
FPLA	Prog	Prog	TS, OC, Fusible Polarity
FPGA	Prog	None	TS, OC, Fusible Polarity
FPLS	Prog	Prog	TS, Registered Feedback, I/O
PAL	Prog	Fixed	TS, Registered Feedback, I/O

Table 1

PAL Input/Output/Function/Performance Chart

PART NUMBER	INPUT	OUTPUT	PROGRAMMABLE I/O'S	FEEDBACK REGISTER	OUTPUT POLARITY	FUNCTIONS	PERFORMANCE			
							STD	A	-2	-4
PAL10H8	10	8			AND-OR	AND-OR Gate Array	X		X	
PAL12H6	12	6			AND-OR	AND-OR Gate Array	X		X	
PAL14H4	14	4			AND-OR	AND-OR Gate Array	X		X	
PAL16H2	16	2			AND-OR	AND-OR Gate Array	X		X	
PAL16C1	16	2			BOTH [1]	AND-OR Gate Array	X		X	
PAL20C1	20	2			BOTH [1]	AND-OR Gate Array	X			
PAL10L8	10	8			AND-NOR	AND-OR Invert Gate Array	X		X	
PAL12L6	12	6			AND-NOR	AND-OR Invert Gate Array	X		X	
PAL14L4	14	4			AND-NOR	AND-OR Invert Gate Array	X		X	
PAL16L2	16	2			AND-NOR	AND-OR Invert Gate Array	X		X	
PAL12L10	12	10			AND-NOR	AND-OR Invert Gate Array	X			
PAL14L8	14	8			AND-NOR	AND-OR Invert Gate Array	X			
PAL16L6	16	6			AND-NOR	AND-OR Invert Gate Array	X			
PAL18L4	18	4			AND-NOR	AND-OR Invert Gate Array	X			
PAL20L2	20	2			AND-NOR	AND-OR Invert Gate Array	X			
PAL16L8	10	2	6		AND-NOR	AND-OR Invert Gate Array	X	X	X	X
PAL20L10	12	2	8		AND-NOR	AND-OR Invert Gate Array	X			
PAL16R8	8	8		8	AND-NOR	AND-OR Invert Array w/Reg's	X	X	X	X
PAL16R6	8	6	2	6	AND-NOR	AND-OR Invert Array w/Reg's	X	X	X	X
PAL16R4	8	4	4	4	AND-NOR	AND-OR Invert Array w/Reg's	X	X	X	X
PAL20X10	10	10		10	AND-NOR	AND-OR-XOR Invert w/Reg's	X			
PAL20X8	10	8	2	8	AND-NOR	AND-OR-XOR Invert w/Reg's	X			
PAL20X4	10	4	6	4	AND-NOR	AND-OR-XOR Invert w/Reg's	X			
PAL16X4	8	4	4	4	AND-NOR	AND-OR-XOR Invert w/Reg's	X			
PAL16A4	8	4	4	4	AND-NOR	AND-CARRY-OR-XOR Invert w/Reg's	X			

[1] Simultaneous AND-OR and AND-NOR outputs

Table 2

PALs For Every Task

The members of the PAL family and their characteristics are summarized in Table 2. They are designed to cover the spectrum of logic functions at reduced cost and lower package count. This allows the designer to select the PAL that best fits his application. PALs come in the following basic configurations:

Gate Arrays

PAL gate arrays are available in sizes from 12x10 (12 input terms, 10 output terms) to 20x2, with both active high and active low output configurations available (figure 9). This wide variety of input/output formats allows the PAL to replace many different sized blocks of combinatorial logic with single packages.

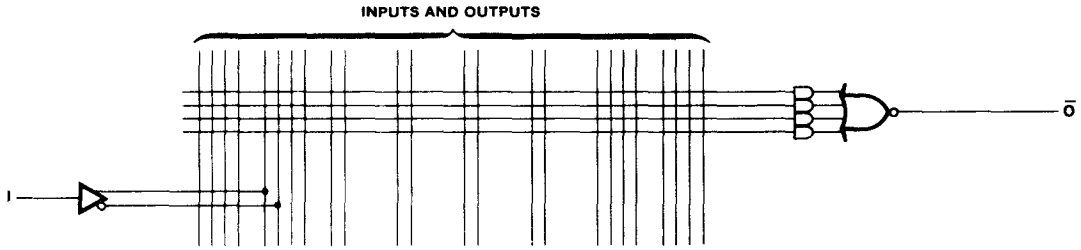


Figure 9

Programmable I/O

A feature of the high-end members of the PAL family is programmable input/output. This allows the product terms to directly control the outputs of the PAL (Figure 10). One product term is used to enable the three-state buffer, which in turn gates the summation term to the output pin. The output is also fed back into the PAL array as an input. Thus the PAL drives the I/O pin when the three-state gate is enabled; the I/O pin is an input to the PAL array when the three-state gate is disabled. This feature can be used to allocate available pins for I/O functions or to provide bi-directional output pins for operations such as shifting and rotating serial data.

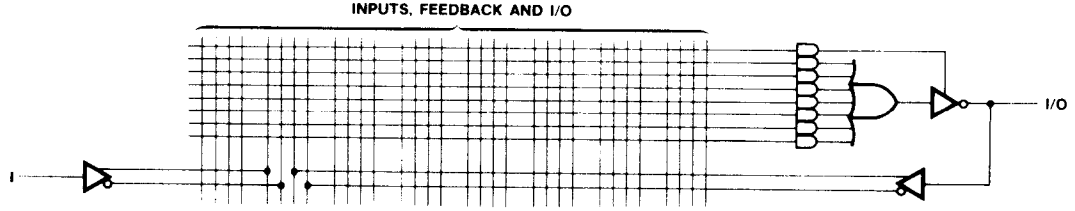


Figure 10

Registered Outputs with Feedback

Another feature of the high end members of the PAL family is registered data outputs with registered feedback. Each product term is stored into a D-type output flip-flop on the rising edge of the system clock (Figure 11). The Q output of the flip-flop can then be gated to the output pin by enabling the active low three-state buffer.

In addition to being available for transmission, the Q output is fed back into the PAL array as an input term. This feedback allows the PAL to "remember" the previous state, and it can alter its function based upon that state. This allows the designer to configure the PAL as a state sequencer which can be programmed to execute such elementary functions as count up, count down, skip, shift, and branch. These functions can be executed by the registered PAL at rates of up to 20 MHz

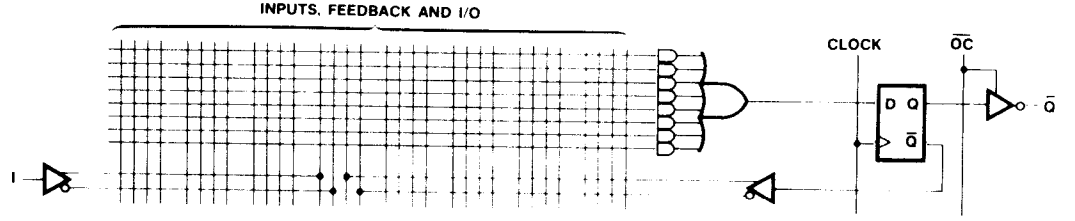


Figure 11

XOR PALs

These PALs feature an exclusive OR function. The sum of products is segmented into two sums which are then exclusive ORed (XOR) at the input of the D-type flip-flop (Figure 12). All of the features of the Registered PALs are included in the XOR PALs. The XOR function provides an easy implementation of the HOLD operation used in counters and other state sequencers.

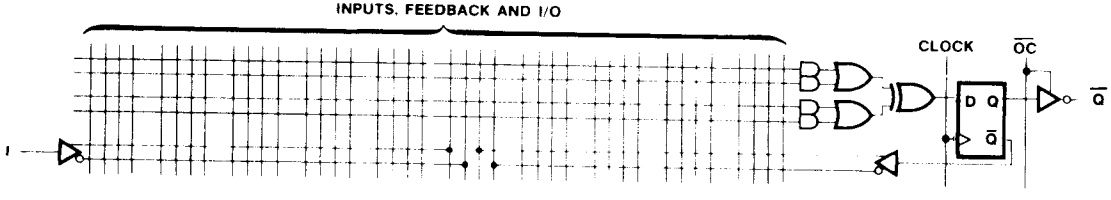


Figure 12

Arithmetic Gated Feedback

The arithmetic functions (add, subtract, greater than, and less than) are implemented by addition of gated feedback to the features of the XOR PALs. The XOR at the input of the D-type flip-flop allows carrys from previous operations to be XORed with two variable sums generated by the PAL array. The flip-flop Q output is fed back to be gated with input terms A (Figure 13). This gated feedback provides any one of the 16 possible Boolean combinations which are mapped in the Karnaugh map (Figure 15). Figure 14 shows how the PAL array can be programmed to perform these 16 operations. These features provide for versatile operations on two variables and facilitate the parallel generation of carrys necessary for fast arithmetic operations.

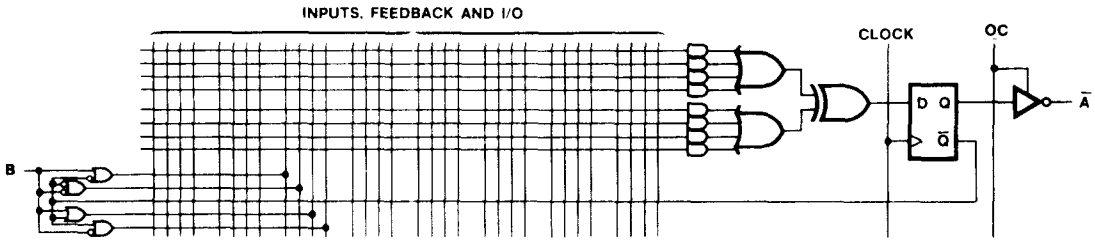


Figure 13

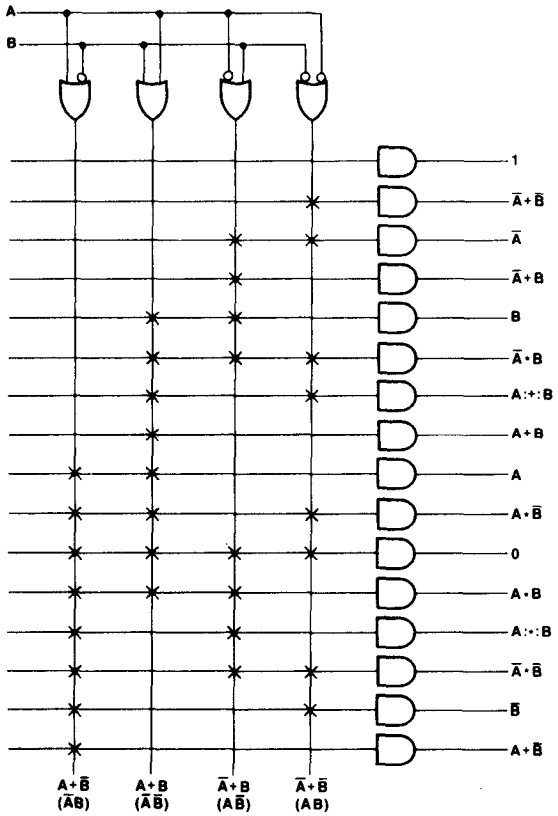


Figure 14

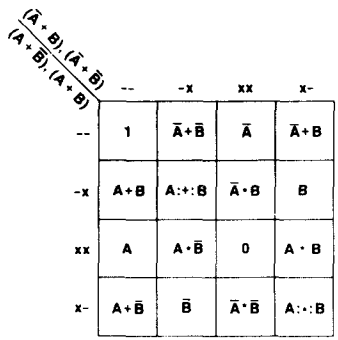


Figure 15

It should now be clear that the PAL family can replace most Small-Scale Integrated Logic (SSI) logic in use today, thereby lowering product cost and giving the designer even greater flexibility in implementing logic functions.

PAL Programming

PALs can be programmed in most standard PROM programmers with the addition of a PAL personality card. The PAL appears to the programmer as a PROM. During programming half of the PAL outputs are selected for programming while the other outputs and the inputs are used for addressing. The outputs are then switched to program the other locations. Verification uses the same procedure with the programming lines held in a low state.

PALASM (PAL Assembler)

PALASM is the software used to define, simulate, build, and test PALs. PALASM accepts the PAL Design Specification as an input file. It verifies the design against an optional function table and generates the fuse plot which is used to program the PALs. PALASM is available upon request in many source code media and is documented in the PAL Design Concepts section.

HALs (Hard Array Logic)

The HAL family is the mask programmed version of a PAL. The HAL is to a PAL just as a ROM is to a PROM. A standard wafer is fabricated to the 6th mask. Then a custom metal mask is used to fabricate Aluminum links for a HAL instead of the programmable Ti-W fuse array used in a PAL.

The HAL is a cost-effective solution for large quantities and is unique in that it is a gate array with a programmable prototype.

HMSI (HAL Medium Scale Integration)

The HMSI family is derived from the PAL using HAL technology. These devices perform predetermined functions which are not available in the existing TTL family. Because they are produced in volume, the user receives the benefit of volume pricing. HMSI PAL designs are given in the Applications section with their 74LS number in line 2 of the PAL design Specification.

PAL Technology

PALs are manufactured using the proven TTL Schottky bipolar Ti-W fuse process to make fusible-link PROMs. An NPN emitter follower array forms the programmable AND array. PNP inputs provide high-impedance inputs (0.25 mA max) to the array. All outputs are standard TTL drivers with internal active pull-up transistors. Typical PAL propagation delay time is 25 ns, and all PALs are packaged in space saving 20-pin and 24-pin SKINNYDIP™.

PAL Data Security

The circuitry used for programming and logic verification can be used at any time to determine the logic pattern stored in the PAL array. For security, the PAL has a "last fuse" which can be blown to disable the verification logic. This provides a significant deterrent to potential copiers, and it can be used to effectively protect proprietary designs.

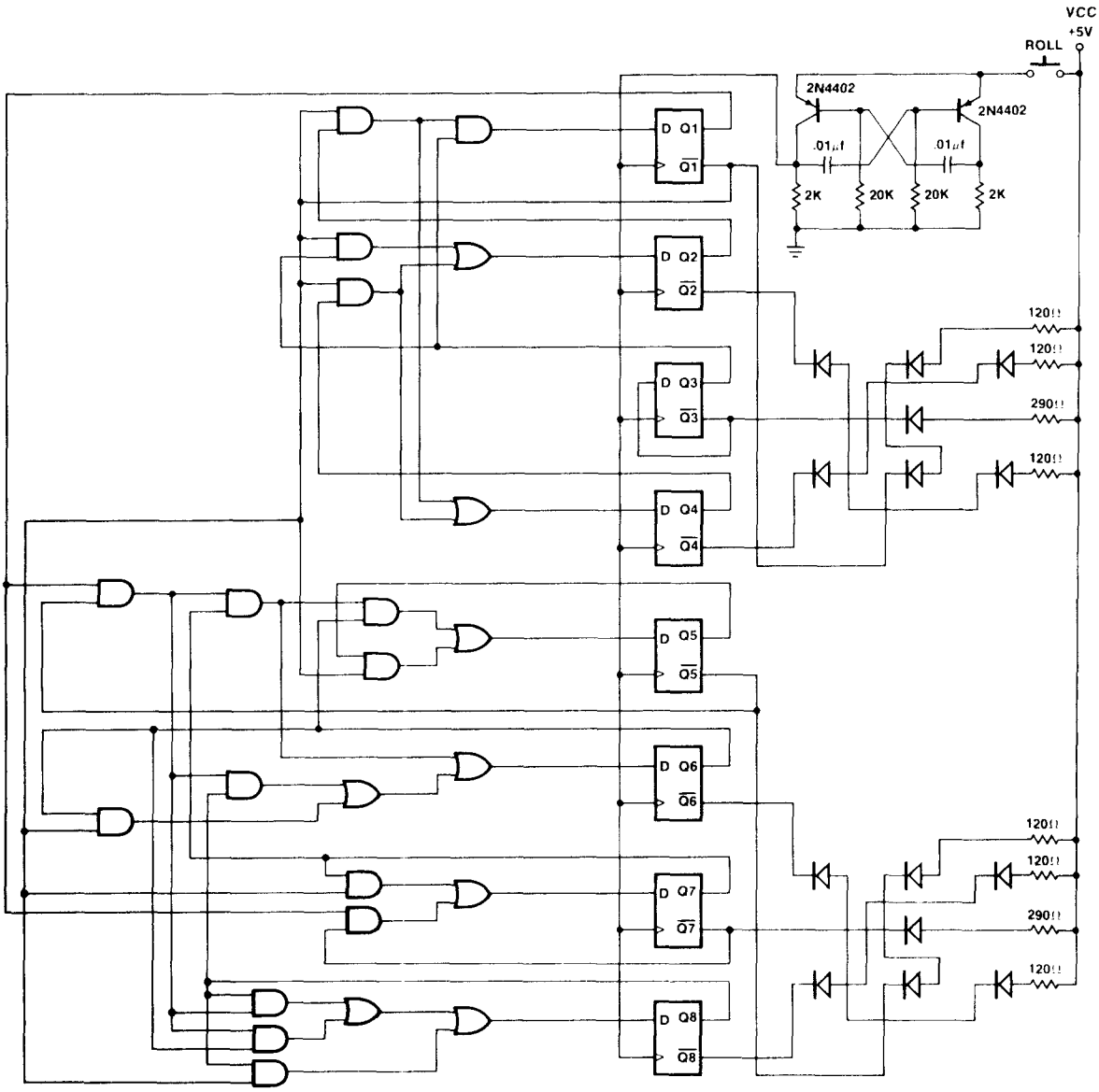

VCC
+5V
ROLL
2N4402
2N4402
.01μf
.01μf
2K
20K
20K
2K
D Q1
Q1
D Q2
Q2
D Q3
Q3
D Q4
Q4
D Q5
Q5
D Q6
Q6
D Q7
Q7
D Q8
Q8
120Ω
120Ω
290Ω
120Ω
120Ω
120Ω
290Ω
120Ω

Figure 16

PAL Part Numbers

The PAL part number is unique in that the part number code also defines the part's logic operation. The PAL parts code system is shown in Figure 17. For example, a PAL14L4CN would be a 14 input term, 4 output term, active-low PAL with a commercial temperature range packaged in a 20-pin plastic dip.

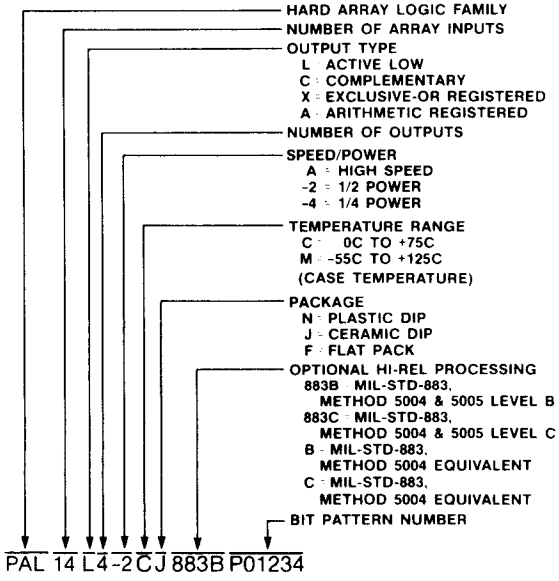


Figure 17

PAL Logic Symbols

The logic symbols for each of the individual PAL devices gives a concise functional description of that PAL's logic function. This symbol makes a convenient reference when selecting the PAL that best fits a specific application. Figure 18 shows the logic symbol for a PAL10H8 gate array.

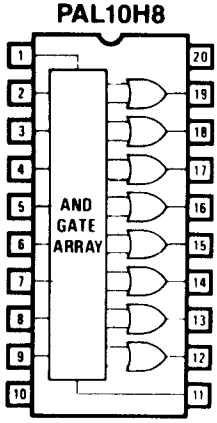


Figure 18

A PAL Example

As an example of how the PAL enables the designer to reduce costs and simplify logic design, consider the design of a simple, high volume consumer product: an electronic dice game. This type of product will be produced in extremely high volume, so it is essential that every possible production cost be minimized.

The electronic dice game is simply constructed using a free running oscillator whose output is used to drive two asynchronous modulo six counters. When the user "rolls" the dice (presses a button), the current state of the counters is decoded and latched into a display resembling the pattern seen on an ordinary pair of dice.

A conventional logic diagram for the dice game is shown in Figure 16. (A detailed logic derivation is shown in the PAL applications section of this handbook). It is implemented using standard TTL, SSI and MSI parts, with a total I.C. count of eight: six quad gate packages and two quad D-latches. Looks like a nice, clean logic design, right? Wrong!!

PAL Goes to the Casino

A brief examination of Figure 16 reveals two basic facts: first, the circuit contains mostly simple, combinatorial logic, and second, it uses a clocked state transition sequence. Remembering that the PAL family contains ample provision for these features, the PAL catalog is consulted. The PAL16R8 has all the required functions, and the entire logic content of the circuit can be programmed into a single PAL shown in Figure 19.

In this example, the PAL effected an eight to one package count reduction and a significant cost savings. This is typical of the power and cost effective performance that the PAL family brings to logic design.

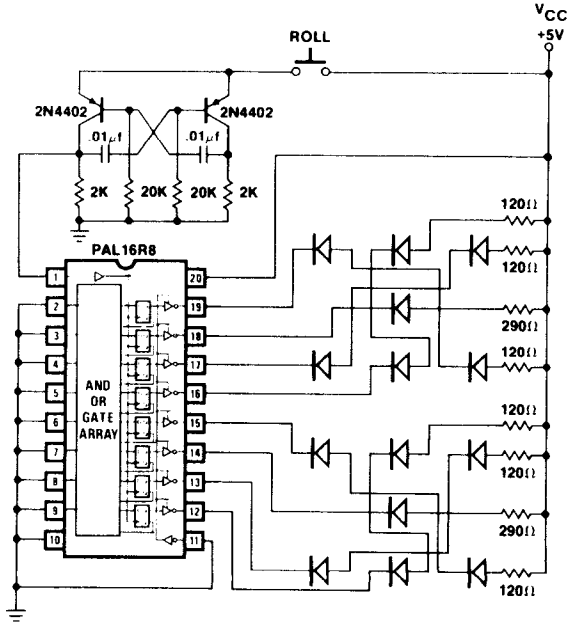
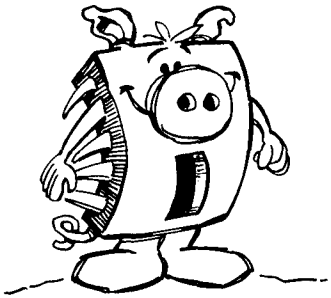


Figure 19

Advantages of Using PALs



The PAL has a unique place in the world of logic design. Not only does it offer many advantages over conventional logic, it also provides many features not found anywhere else. The PAL family:

- Programmable replacement for conventional TTL logic.
- Reduces IC inventories substantially and simplifies their control.
- Reduces chip count by at least 4 to 1.
- Expedites and simplifies prototyping and board layout.
- Saves space with 20-pin and 24-pin Skinny DIP packages.
- High speed: 25ns typical propagation delay.
- Programmed on standard PROM programmers.
- Programmable three-state outputs.
- Special feature eliminates possibility of copying by competitors.

All of these features combine together to lower product development costs and increase product cost effectiveness. The bottom line is that PALs save money.

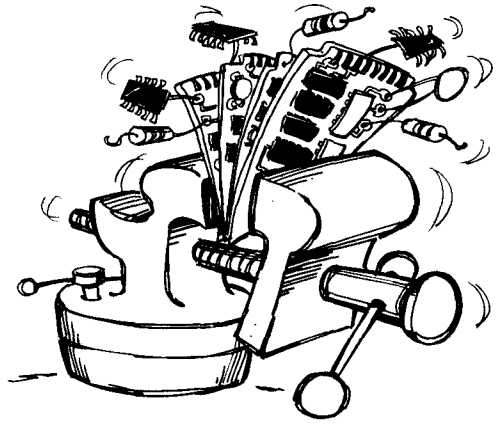
Direct Logic Replacement

In both new and existing designs the PAL can be used to replace various logic functions. This allows the designer to optimize a circuit in many ways never before possible. The PAL is particularly effective when used to provide interfaces required by many LSI functions. PAL flexibility combined with LSI function density makes a powerful team.

Design Flexibility

The PAL offers the systems logic designer a whole new world of options. Until now, the decision on logic system implementation was usually between SSI/MSI logic functions on one hand and microprocessors on the other. In many cases the function required is too awkward to implement the first way and too simple to justify the second. Now the PAL offers the designer high functional density, high speed, and low cost. Even better, PALs come in a variety of sizes and functions, thereby further increasing the designer's options.

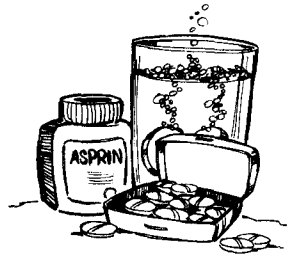
Space Efficiency



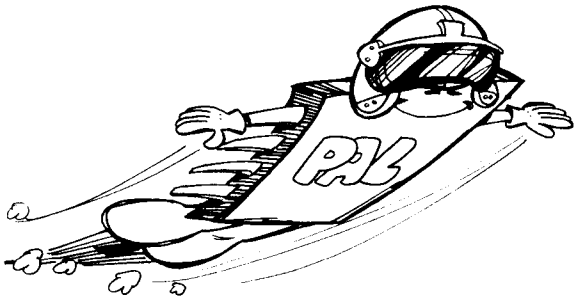
By allowing designers to replace many simple logic functions with single packages, the PAL allows more compact P.C. board layouts. The PAL's space saving 20-pin "skinny dip" helps to further reduce board area while simplifying board layout and fabrication. This means that many multi-card systems can now be reduced to one or two cards, and that can make the difference between a profitable success or an expensive disaster.

Smaller Inventory

The PAL family can be used to replace up to 90% of the conventional TTL family with just 25 parts. This considerably lowers both shelving and inventory cataloging requirements. Even better, small custom modifications to the standard functions are easy for PAL users, not so easy for standard TTL users.



High Speed

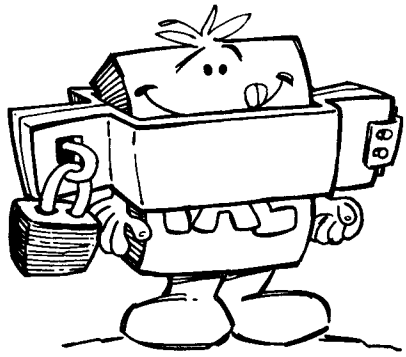


The PAL family runs faster or equal to the best of bipolar logic circuits. This makes the PAL the ideal choice for most logical operations or control sequence which requires a medium complexity and high speed. Also, in many microcomputer systems, the PAL can be used to handle high speed data interfaces that are not feasible for the microprocessor alone. This can be used to significantly extend the capabilities of the low-cost, low-speed NMOS microprocessors into areas formerly requiring high-cost bipolar microprocessors.

Easy Programming

The members of the PAL family can be quickly and easily programmed using standard PROM programmers. This allows designers to use PALs with a minimum investment in special equipment. Many types of programmable logic, such as the FPLA, require an expensive, dedicated programmer.

Secure Data



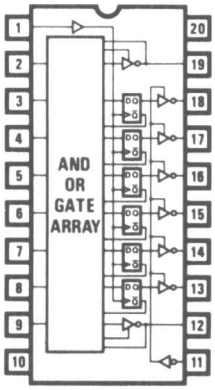
The PAL verification logic can be completely disabled by blowing out a special "last link." This prevents the unauthorized copying of valuable data, and makes the PAL perfect for use in any application where data integrity must be carefully guarded.

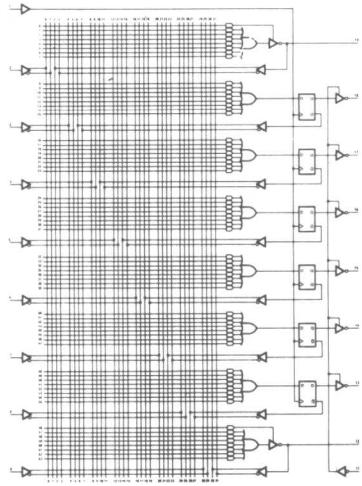
Summary

The 25 member PAL family of logic devices offers logic designers new options in the implementation of sequential and combinatorial logic designs. The family is fast, compact, flexible, and easy to use in both new and existing designs. It promises to reduce costs in most areas of design and production with a corresponding increase in product profitability.

A Great Performer!

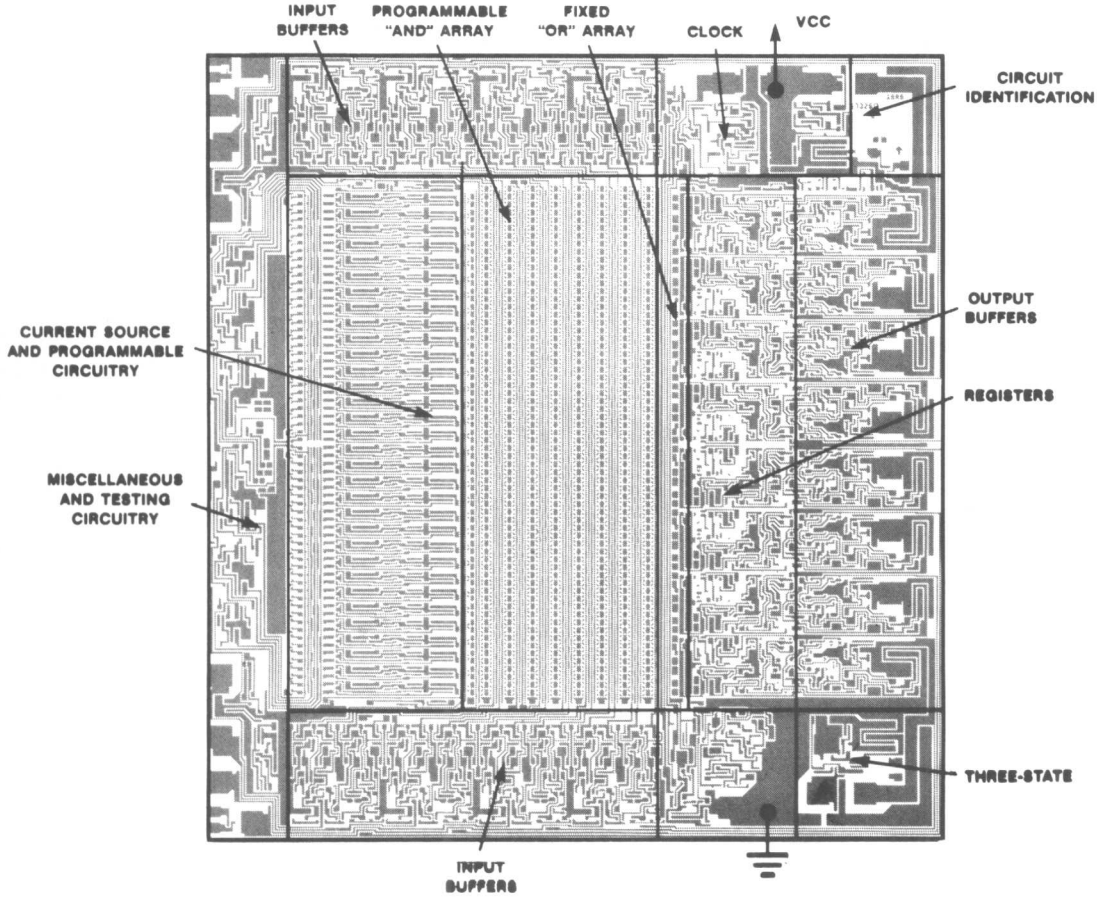
PAL16R6





PAL16R6 Logic Symbols

PAL16R6 Logic Diagram



PAL16R6 Metalization