BALLEW, WALTER DAVID

B(2):P(m) DUAL RADIX SYSTEMS - THEORY, DESIGN, AND I(SQUARE)L IMPLEMENTATION

The University of Oklahoma PH.D. 1981

PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy. Problems encountered with this document have been identified here with a check mark __✓__.

1. Glossy photographs or pages ✓
2. Colored illustrations, paper or print ✓
3. Photographs with dark background _____
4. Illustrations are poor copy _____
5. Pages with black marks, not original copy _____
6. Print shows through as there is text on both sides of page _____
7. Indistinct, broken or small print on several pages ✓
8. Print exceeds margin requirements _____
9. Tightly bound copy with print lost in spine _____
10. Computer printout pages with indistinct print _____
11. Page(s) _____ lacking when material received, and not available from school or author.
12. Page(s) _____ seem to be missing in numbering only as text follows.
13. Two pages numbered _____. Text follows.
14. Curling and wrinkled pages _____
15. Other _____

THE UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

B(2):P(m) DUAL RADIX SYSTEMS - THEORY, DESIGN, AND I^2L IMPLEMENTATION

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

degree of

DOCTOR OF PHILOSOPHY

BY

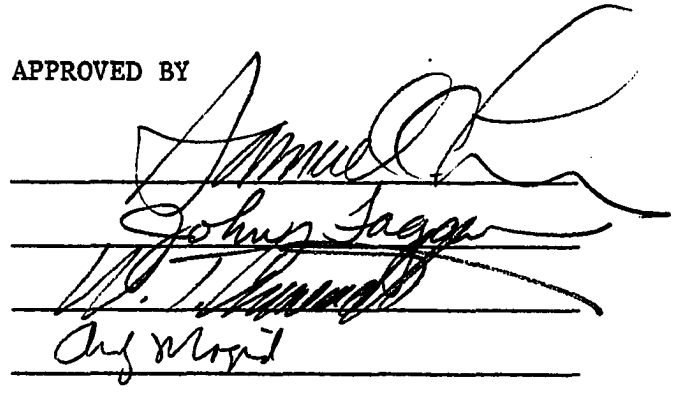
WALTER DAVID BALLEW

1981

B(2):P(m) DUAL RADIX SYSTEMS - THEORY, DESIGN,

AND I^2L IMPLEMENTATION

APPROVED BY



DISSERTATION COMMITTEE

ACKNOWLEDGEMENTS

The author is grateful for the opportunity to have studied and worked with Dr. S. C. Lee and is especially appreciative of his guidance throughout this effort. Drs. W. T. Cronenwett, J. E. Fagan, and A. R. Magid each offered their counsel willingly and helped to make this writing a pleasant task. A special thanks to Ms. La Vonne Whitney for her patience and the meticulous concern which she demonstrated in the preparation of the manuscript.

To my wife Sally and my children Mitchell, Meredith, and Sarah Beth, thank you for your total support and understanding. Finally, I am pleased to have had the Western Electric Company as an employer and a sponsor, without which, this work could not have been done.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

The possibility of a binary computer being compatible with computers operating with a radix or base greater than two has been suggested by several authors [4-6]. Also, much of the motivation to study algebras and their realizations comes from the influence and predominance of switching theory and hardware. Dao, McCluskey, and Russell chose to work with four-valued logic because it was a power of two. They believed that it was important to be able to easily convert between binary systems and multivalued systems [5]. The algebraic relationships between Boolean and Post functions have been developed [7]. Comparisons between higher-order Boolean algebras, $B(2^N)$ for N greater than one, and Post algebras were developed by Metze and Wojcik [4]. Su and Sarris have studied relationships between multivalued switching algebra and Boolean algebra under different definitions of the complement [8], and it has been shown that any multivalued function can be represented by a vector Boolean function [9]. It is well known that the two-valued Boolean algebra and two-valued Post algebra are identical, $B(2) \simeq P(2)$.

Rather than establishing equivalence relationships for B(2) and P(m) algebras (m > 2), the motivation is to choose an algebra and a realization that would be functionally complete for two radices. The suggestion of a dual radix machine has been made [6]. However, to date, no serious investigation has been made to show if a dual radix system offers any advantages or is even in fact achievable. That is, would a dual radix implementation of a MAX/OR gate, for example, be more complicated and difficult to build than the disjoint equivalent of the circuit operating as both a MAX and OR gate? If such a circuit can be designed, then what special features would it offer over the same disjoint equivalent?

Background

The interest in multiple-valued logic (MVL) systems continues to grow as understanding of both the algebra and the implementation increases. Recent articles are offering hardware realization [18-21], where earlier work emphasized the algebra only [22]. However, after nearly four decades of hardware and software activity that is firmly established in binary switching theory, the transition to higher radix systems is slow at best. Some degree of upward compatibility from B(2) machines to multivalued machines would be attractive to those who find themselves involved with a computer world which is, for the most part, running on binary hardware.

An alternative to hardware upward compatibility would be to design new compilers which would allow source code from older base two machines to be compiled into object codes which would execute on higher

radix hardware. This will certainly be done when the new hardware's advantages outweigh its disadvantages. All of the burden of upward compatibility would then be placed upon the integrity of the new software. For that segment of the computer community which prefers to have the hardware and the architecture invisible to the user, this alternative makes sense. The problems associated with total hardware upward compatibility involve complexity, cost, reliability, total burden, etc. Aside from the purely theoretical motivation, research activity which might answer these questions could be valuable to both the hardware designer as well as the end user.

To date, several algebras have been formulated and their implementation and application investigated [16]. For purposes of comparison and analysis in this paper, one of two algebras will be used to describe the F(m) multivalued systems. The algebra of Vranesic, et al, is described in [1] and the algebra of Allen and Givone in [10]. Both of these algebras share the following definitions:

Let T be a switching algebra with the following characteristics:

Definition 1.1. T contains a set of variables (x,y...z) which can assume m logic values from the set (0,1,...,m-1)
$0 < 1 < \ldots < m-1$.

Definition 1.2. There are two operations, (+) and (·), in T such that

$x + y = MAX\ (x,y)$ where $x,y \in S = \{0,1,\ldots m-1\}$

$x \cdot y = MIN\ (x,y)$

Definition 1.3. Given $x,y,z \in S$

Idempotent:	$x + x = x$	, $x \cdot x = x$;
Commutative:	$x + y = y + x$	, $x \cdot y = y \cdot x$;
Associative:	$(x + y) + z = x + (y + z)$,	
	$(x \cdot y) \cdot z = x \cdot (y \cdot z)$;	
Distributive:	$x + y \cdot z = (x + y) \cdot (x + z)$,	
	$x \cdot (y + z) = (x \cdot y) + (x \cdot z)$;	
Absorption:	$x + x \cdot y = x$	, $x \cdot (x + y) = x$
Null Element:	$x + o = x$	, $x \cdot o = o$
Universal Element:	$x + p = p$	, $x \cdot p = x$, $p = m-1$;

The two algebras differ in the choice of unary operators. Post [7, 17] has shown that the cycling operation and the product operation are a functionally complete set. Vranesic has chosen the following unary operation to complete his definition.

Definition 1.4. Given $x,k \in S$, x a variable in S, k a constant in S

a) The M unary "inverter" operation is

$x^k = k$ if $x = o$

$x^k = o$ otherwise

b) The M unary "clockwise cycling" operation is

$x^{\overrightarrow{r}} = (x \oplus r) \bmod M$, $r \in S$ and constant, $\oplus$ = arithmetic addition.

The unary operation proposed by Allen-Givone to complete their definition is:

Definition 1.5. Given $x,a,b \in S$, x a variable in S , a,b constants in S , $a \leq b$

The unary operator (a,b) on the variable x, called a literal and denoted by ${}_a^b x$ is:

$$ {}_a^b x = \begin{cases} m-1 & a \le \text{logical value of } x \le b \\ 0 & \text{otherwise} \end{cases} $$

Realizations of any of the multivalued algebras has been slow to develop [22]. The success of any higher radix logic family will depend upon its ability to offer concrete advantages in the areas of cost, speed, ease of implementation, etc., over conventional binary systems [23]. Recently, [19], the unary operation given by Definition 1.5 has been constructed using GaAs metal semiconductor field effect transistors (MESFET). In order for the MVL gates to work, it is necessary for the MESFETs to exhibit selectable pinchoff voltages. Work on selective epitaxial layer etching for modification of MESFET pinchoff voltage is currently under way at the State University of New York at Buffalo [19]. More predominate in the area of MVL gate realization is the integrated injection logic (I^2L) technology [5, 12, 14, 18, 20].

Integrated injection logic was first developed in 1972 by separate groups at IBM and Phillips Research Laboratory [14]. I^2L offers the speed of the bipolar junction transistor (BJT) and the high packaging density and low power dissipation of the metal-oxide semiconductor transistor (MOS). The delay power product is typically 10 pico-joules with an injector current of 10 microamps. Newer I^2L technology offers a significant delay-power product improvement by the use of a Schottky-base I^2L (SBI^2L) structure [24]. For the most

part, the I^2L family has offered the multiple-valued designer the best choice for implementing MVL algebras. I^2L gates necessary to realize a four-valued logic system were developed in 1977 [5]. In 1979 McCluskey [18] introduced strong and weak literals which could be used to develop three connectives ($\overline{\text{MAX}}$, $\overline{\text{PLUS}}$, and $\overline{\text{INHIBIT}}$). Each of these functions were combined into a single universal gate. This arrangement stresses a closer relationship between the algebra and the actual integrated circuit.

Research Objectives

This research will begin with an examination of the relationship between binary Boolean algebra and Post algebras. The work will show that there exist in every Post algebra, P(m), $m = 2^N$, N an integer greater than one, 2^{N-1} homomorphic mappings of B(2). This is true only if the pseudo-complement or strong negation operation is defined in P(m). In almost all cases throughout the research, the value of m will be four, the smallest Post algebra for which the mappings are defined. It would be impossible to consider all mappings, not to mention that the circuit design and realizations would become particularly troublesome and recondite without really offering any additional insight to the research. Also, the quaternary logic system, as has been cited, is practically achievable and this provides an even greater motivation for the choice.

The research will proceed by examining the burden in P(4) which is imposed by one or more of the homomorphic images of B(2). It would

be unsafe to assume one mapping to be more favorable than the other. Therefore, consideration will be given to both. At this stage of the work the concept of dual radix combinational design and realizations will be emphasized. Although it is not required that all P(4) circuitry be able to execute as a B(2) circuit, the analysis will be made for both bases. While it is unnecessary for functional completeness in both radices, it may be necessary for purposes of signal routing or architecture to force a B(2) mapping through a particular P(4) circuit. Circuitry which will perform in more than one radix without modification will be shown to have no natural radix.

A formal approach to dual radix design will be established. The algorithm will allow the designer to choose the homomorphic image of B(2) in P(m) which will give the lowest cost realization. The method will be demonstrated on a dual radix full adder VLSI design using I^2L.

A proposed goal of this research is to consider a tightly coupled B(2):P(4) processor. Tightly coupled implies that all primary signal paths throughout the machine be used for both binary and quaternary functions, and that dual radix hardware be used in the combinational and sequential portions of the processor. Therefore, a memory element which can operate in two radices is obviously necessary. It is possible that memory elements developed in previous work might accommodate both bases. Sequential design considerations will be examined and modified if required to meet the B(2):P(4) requirements. Finally, the hardware for dual radix buses will be developed and B(2):P(4) machine architecture will be discussed.

In an effort to emphasize the practical implications of this work, all B(2):P(4) analysis will be done at the gate level using the integrated injection logic family. Standard B(2):P(4) building blocks will be designed and constructed using I^2L. Actual electrical parameters will be measured for each circuit designed in Chapter II. Operating points will be chosen for injection current amplitude which will provide the greatest noise immunity and minimize logic level deterioration. Quantizers are necessary for practical MVL realizations. Hopefully, their usage can be limited to output stage buffering. The gate level implementation of dual radix circuitry can be transferred to master slice VLSI devices since the I^2L inverter arrays and PNP/NPN transistor arrays used for circuit construction in the appendix are electrically compatible with the master slice. Suggestions for improvements in present I^2L technologies which will enhance MVL dual radix implementation will be offered.

CHAPTER II

B(2):P(4) MAPPINGS AND REALIZATION

The switching algebra defined in Chapter I by Definitions 1.1-1.5 provides a basis of handling multivalued or nonbinary switching functions. In fact, two different algebras were defined [1,10]; however, both of these algebras are structure isomorphic [16]. Moreover, the four commonly used multivalued algebras have been shown to be isomorphic. This includes the Post algebra developed by Epstein, the monotonic system, the free system, and the Herrmann systems. Each of these algebras are equivalent in that any algebra may be obtained from the other by a transformation and that all are Postian algebras [16].

In Post's work [7], a system of m-valued logic for finite many-valued propositional logic was first introduced. By means of two primitive functions, the function of negation and the function of disjunction, the variable elementary propositions are combined to form propositional functions of the system. In Rosenbloom's work [17], these concepts were called the cycle gate, the join (MAX), and the meet (MIN). These operations were defined in Definition 1.4(b) (for $r = 1$) and Definition 1.2, respectively.

When working with Boolean algebras and Post algebras, it has been convenient to define additional operations in the Post algebra which are closed in P(m) [4,8]. In this work two definitions for a complement operation in P(m) will be used so that the homomorphic images of B(2) in P(m) can be considered. The Post algebra, P(m) is defined by Definitions 2.1 and 2.2.

Definition 2.1. P(m) is a Post algebra with m a fixed integer greater than or equal to two. This distributive lattice contains a zero element (0) and unit element (u) and elements e_i, $e_i = i$, $i = 0,1,2,\ldots m-1$ such that $0 = e_0 < e_1 < e_2 <\ldots< e_{m-1} = m$. Also, the lattice operations given by Definition 2.2 hold.

Definition 2.2. For $x\ y \in P(m)$ the following operations exist

a) $x + y = MAX\ (x,y)$

b) $x \cdot y = MIN\ (x,y)$

c) $x^{\rightarrow} = (x \oplus 1)\ MOD\ m$

Since a Boolean algebra is a distributed, complemented lattice with a zero element and a unit element [11], it is clear that not every pair of elements in P(4) will satisfy the requireements of B(2). Wojcik and Metze have studied the relationships between higher-ordered Boolean algebras and Post algebras [4]. In their work they have defined the complement operation in P(m) as follows:

Definition 2.3. For x and $x' \in P$, x' is the complement of x if and only if $x + x' = u$ and $x \cdot x' = 0$.

Using Definition 2.3 every Post algebra contains a unique two-element Boolean algebra.

Example 2.1. Let $x = 1$ and $y = 2$, $x, y \in P(4)$.

$$x + y = 1 + 2 = 2 \neq 3 = u = \text{LUB}$$

$$x \cdot y = 1 \cdot 2 = 1 \neq 0 = \text{GLB}$$

$$x \neq y'$$

Obviously from Example 2.1 x and y are not complements. In Example 2.2 x and y are chosen such that Definition 2.3 does hold.

Example 2.2. Let $x = 0$ and $y = 3$, $x, y \in P(4)$

$$x + y = 0 + 3 = 3 = \text{LUB}$$

$$x \cdot y = 0 \cdot 3 = 0 = \text{GLB}$$

If the 0 and 1 elements of B(2) are mapped to the 0 and 3 elements in P(4), which are complements, then P(4) contains a unique two element Boolean algebra.

Definition 2.4. Given B(2), a two element Boolean algebra with 0 = GLB and 1 = LUB, and P(m) a Post algebra of m elements

$B(2) \rightarrow P(m)_{a,b}$ is a mapping from B(2) to P(m) such that

$\text{GLB} = 0 \rightarrow a$ $\quad a < b;\ a, b \in S,$

$\text{LUB} = 1 \rightarrow b$ $\quad S = \{0, 1, \ldots, m-1\}$

Using definition 2.4, the six possible mappings of B(2) into P(4) are:

$B(2) \rightarrow P(4)_{0,1}$ $\quad B(2) \rightarrow P(4)_{1,2}$

$B(2) \rightarrow P(4)_{0,2}$ $\quad B(2) \rightarrow P(4)_{1,3}$

$B(2) \rightarrow P(4)_{0,3}$ $\quad B(2) \rightarrow P(4)_{2,3}$

By Definition 2.2 and 2.3 only $B(2) \rightarrow P(4)_{0,3}$ is homomorphic. For each of the other mappings the complement operation in P(4) does not hold; therefore, the resulting B(2) lattice is not complemented.

Theorem 2.1. $B(2) \rightarrow P(4)_{0,3}$ is the only homomorphic mapping using Definition 2.3.

Proof: Assume that other homomorphic mappings exist. There are twelve possible ways to map B(2) into P(4). However, P(4) is a Post algebra and thus partial ordering exists. Therefore, by Definition 2.4, only six possible mappings exist (since $a < b$). Clearly, $B(2) \rightarrow P(4)_{0,3}$ is homomorphic. Given the operations of "AND," "OR," and complement in B(2), and $\phi = B(2) \rightarrow P(4)_{0,3}$, then ϕ is a homomorphism and $P(4)_{0,3}$ is the homomorphic image of B(2) if the "AND" function in B(2) is mapped to the MIN function in P(4), and "OR" function in B(2) is mapped to the MAX function in P(4), and the complement function in B(2) is mapped to the complement in P(4).

"AND" $\rightarrow$ MIN

$$(0 \cdot 1)\phi = (0)\phi \cdot (1)\phi$$

$$(0)\phi = 0 \cdot 3$$

$$0 = 0 = \text{GLB}$$

"OR" $\rightarrow$ MAX

$$(0+1)\phi = (0)\phi + (1)\phi$$

$$(1)\phi = 0 + 3$$

$$3 = 3 = \text{LUB}$$

Complement B(2) → Complement P(4)

$$(0')\phi = (0)\phi'$$

$$(1)\phi = 0'$$

$$3 = 3$$

$$(1')\phi = (1)\phi'$$

$$(0)\phi = 3'$$

$$0 = 0$$

Likewise, $\phi = B(2) \rightarrow P(4)_{0,1}$ can be shown to be a homomorphism by showing

$(xy)\phi = (x)\phi\ (y)\phi$ for x, y ∈ B(2) with the "AND," "OR" and complement operations.

"AND" → MIN

$$(0 \cdot 1)\phi = (0)\phi \cdot (1)\phi$$

$$(0)\phi = 0 \cdot 1$$

$$0 = 0 = \text{GLB}$$

"OR" → MAX

$$(0+1)\phi = (0)\phi + (1)\phi$$

$$(1)\phi = 0 + 1$$

$$1 = 1 = \text{LUB}$$

Complement B(2) → Complement P(4)

$$(0')\phi = (0)\phi'$$

$$(1)\phi = 0'$$

$$1 \neq 3$$

Therefore, $\phi = B(2) \rightarrow P(4)_{0,1}$ is not a homomorphism since 0, 1 ϵ P(4) are not complements. Similarly, the four other possible mappings are not homomorphic since $0' \neq 2$, $1' \neq 2$, $1' \neq 3$, and $2' \neq 3$. Thus, $B(2) \rightarrow P(4)_{0,3}$ is the only homomorphic mapping using Definition 2.3.

The six possible mappings from B(2) to P(4) using Definition 2.2 for the MAX and MIN function are shown in Figure 2.1. An important observation can be made from an examination of these mappings. For all $B(2) \rightarrow P(m)_{a,b}$ the "AND" operation in B(2) will map to the MIN operation in P(m), and the "OR" operation in B(2) will map to the MAX operation in P(m). The restriction placed upon elements in P(m) by Definition 2.3 has forced only the $B(2) \rightarrow P(4)_{0,3}$ mapping to be homomorphic. Because the "AND" and "OR" operations in B(2) are closed in P(4) for all mappings, non-homomorphic mappings are attractive even though Definition 2.3 does not hold.

If a different definition of the complement in P(4) is used, then it would be interesting to examine the six mappings shown in Figure 2.1, for any new homomorphic images. The notion of a new definition of the complement in P(4) is given in Definition 2.5 [8].

Definition 2.5. Given P(m) a Post algebra with m elements and x a variable in P(m),

$\bar{x} = (m-1) - x$ = complement of x (also known as the pseudo-complement or strong negation).

This operation in P(4) is considered to be the closest equivalent to the true complement in B(2). The truth table of the complement operation given in Definition 2.5 for P(4) is shown in Table 2.1.

x \ +y	0	1	2	3
0	0	1	-	-
1	1	1	-	-
2	-	-	-	-
3	-	-	-	-

MAX (0,1)

x \ ·y	0	1	2	3
0	0	0	-	-
1	0	1	-	-
2	-	-	-	-
3	-	-	-	-

MIN (0,1)

$B(2) \rightarrow P(4)_{0,1}$

x \ +y	0	1	2	3
0	0	-	1	-
1	-	-	-	-
2	1	-	1	-
3	-	-	-	-

MAX (0,2)

x \ ·y	0	1	2	3
0	0	-	0	-
1	-	-	-	-
2	0	-	1	-
3	-	-	-	-

MIN (0,2)

$B(2) \rightarrow P(4)_{0,2}$

x \ +y	0	1	2	3
0	0	-	-	1
1	-	-	-	-
2	-	-	-	-
3	1	-	-	1

MAX (0,3)

x \ ·y	0	1	2	3
0	0	-	-	0
1	-	-	-	-
2	-	-	-	-
3	0	-	-	1

MIN (0,3)

$B(2) \rightarrow P(4)_{0,3}$

x \ +y	0	1	2	3
0	-	-	-	-
1	-	0	1	-
2	-	1	1	-
3	-	-	-	-

MAX (1,2)

x \ ·y	0	1	2	3
0	-	-	-	-
1	-	0	0	-
2	-	0	1	-
3	-	-	-	-

MIN (1,2)

$B(2) \rightarrow P(4)_{1,2}$

x \ +y	0	1	2	3
0	-	-	-	-
1	-	0	-	1
2	-	-	-	-
3	-	1	-	1

MAX (1,3)

x \ ·y	0	1	2	3
0	-	-	-	-
1	-	0	-	0
2	-	-	-	-
3	-	0	-	1

MIN (1,3)

$B(2) \rightarrow P(4)_{1,3}$

x \ +y	0	1	2	3
0	-	-	-	-
1	-	-	-	-
2	-	-	0	1
3	-	-	1	1

MAX (2,3)

x \ ·y	0	1	2	3
0	-	-	-	-
1	-	-	-	-
2	-	-	0	0
3	-	-	0	1

MIN (2,3)

$B(2) \rightarrow P(4)_{2,3}$

FIGURE 2.1. Six Possible Mappings from B(2) to P(4)

Table 2.1. Strong Negation in P(4)

x	$\bar{x}$
0	3
1	2
2	1
3	0

Table 2.2. Complement in P(4) for
a) $B(2) \rightarrow P(4)_{0,3}$
b) $B(2) \rightarrow P(4)_{1,2}$

a)

B(2)	P(4)	
x	x	$\bar{x}$
0	0	3
-	1	2
-	2	1
1	3	0

b)

B(2)	P(4)	
x	x	$\bar{x}$
-	0	3
0	1	2
1	2	1
-	3	0

Using Definition 2.5., the $B(2) \rightarrow P(4)_{1,2}$ mapping now becomes homomorphic. Let $\phi = B(2) \rightarrow P(4)_{1,2}$,

"AND" → MIN

$$(0 \cdot 1)\phi = (0)\phi \cdot (1)\phi$$

$$(0)\phi = 1 \cdot 2$$

$$1 = 1$$

"OR" → MAX

$$(0+1)\phi = (0)\phi + (1)\phi$$

$$(1)\phi = 1 + 2$$

$$2 = 2$$

Complement B(2) → Complement P(4)

$$(0')\phi = (0)\phi'$$

$$(1)\phi = 1'$$

$$2 = 2$$

$$(1')\phi = (1)\phi'$$

$$(0)\phi = 2'$$

$$1 = 1$$

The $B(2) \rightarrow P(4)_{0,3}$ and $B(2) \rightarrow P(4)_{1,2}$ mappings are obvious first choices in any consideration for upward compatibility of B(2) algebras into P(4) algebras. However, in attempting to minimize the realization burden in both algebras all mappings may prove to be interesting, especially for special functions. Only the burden of an independent realization of the B(2) complement is involved in choosing a non-homomorphic mapping.

Before leaving this topic to discuss hardware realizations, it might be of interest to examine the more general case of upward compatibility. Obviously, a two-valued Boolean algebra has a homomorphic image in P(8), P(16), etc.

Theorem 2.2. Given P(m), a Post algebra, $m = 2^N$, N an integer greater than one, with MAX and MIN operations given in Definition 2.2, and the complement operation given in Definition 2.5. Also, B(2), a two-valued Boolean algebra, there exists 2^{N-1} homomorphic mappings of B(2) in P(m), as given by Definition 2.4.

Proof: The number of elements in $P(m) = 2^{N-1}$; Given

$0 = e_0 \leq e_1 \leq e_{m-1} = u$; Definition 2.1

$\overline{x} = (m-1) - x$; Definition 2.5

$\overline{x} = (2^N-1) - x$; $m = 2^N$

Choose x_i and x_j such that $x_i < x_j$ (i.e., begin with $x_i = 0$), $x_i = e_i$, $x_j = e_j$, $e_i, e_j \in P(m)$ $i \neq j$ such that $x_i = \overline{x}_j$.

Since m is a power of 2, then m is an even integer and every element must have a complement according to Definition 2.5.

Choose a, b $\in$ P(m) such that $a = e_i$ and $b = e_j$. (This is not a contradiction since $a < b$ by Definition 2.4, and $e_i < e_j$ by the choice of x_i and x_j.)

Let ϕ_1 represent a possible homomorphism

$\phi_1 = B(2) \rightarrow P(m)$ a, b ; Definition 2.4

$\phi_1 = B(2) \rightarrow P(m)$ e_i, e_j

"AND" $\rightarrow$ MIN

$(0 \cdot 1)\phi_1 = (0)\phi_1 \cdot (1)\phi_1$, Definition 2.4

$(0 \cdot 1)\phi_1 = e_i \cdot e_j$, Definition 2.4

$(0 \cdot 1)\phi_1 = e_i$, Definition 2.2

$(0 \cdot 1)\phi_1 = x_i$, Choice of x_i

$(0)\phi_1 = x_i$, Definition "AND" operation in B(2)

$a = x_i$, Definition 2.4

"OR" $\rightarrow$ MAX

$(0+1)\phi_1 = (0)\phi_1 + (1)\ \phi_1$, Definition 2.4

$(0+1)\phi_1 = e_i + e_j$, Definition 2.4

$(0+1)\phi_1 = e_j$, Definition 2.2

$(0+1)\phi_1 = x_j$, Choice of x_j

$(1)\phi_1 = x_j$, Definition "OR" operation in B(2)

$b = x_j$, Definition 2.4

Complement B(2) → Complement P(m)

$(0')\phi_1 = (0)\phi_1'$, Definition 2.4

$(0')\phi_1 = e_i'$, Definition 2.4

$(0')\phi_1 = x_j$, Choice of x_i and x_j

$(1)\phi_1 = x_j$, Definition of complement in B(2)

$b = x_j$, Definition 2.4

$(1')\phi_1 = (1)\phi_1'$, Definition 2.4

$(1')\phi_1 = e_j'$, Definition 2.4

$(1')\phi_1 = x_i$, Choice of x_i and x_j

$(0)\phi_1 = x_i$, Definition of complement in B(2)

$a = x_i$, Definition 2.4

ϕ_1 is a homomorphism.

If for ϕ_1, i = 0 then

$\overline{x}_j = (2^N-1) - x_i$; Definition 2.5

$= (2^N-1) - e_i$

$= (2^N-1) - 0$

$\overline{x}_j = 2^N - 1 = e_j$

AND

$e_i < e_j$ since $0 < 2^N - 1$

$e_i = \overline{e}_j$; Definition 2.5

If ϕ_2 is the next choice for a possible homomorphism, then let i = 1

$$\overline{x}_j = (2^N-1) - x_i \qquad \text{; Definition 2.5}$$
$$= (2^N-1) - e_i$$
$$= (2^N-1) - 1$$
$$\overline{x}_j = 2^N-2 = e_j$$
$$e_i < e_j \text{ since } 1 < 2^N-2 \qquad \text{; } N > 1$$
$$e_i = \overline{e_j} \qquad \text{; Definition 2.5}$$

AND ϕ_2 can be shown to be a homomorphism in the same manner as ϕ_1.

Now the total number of elements in P(m) = 2^N, and the total number of pairs of complemented elements in P(m) is $\frac{2^N}{2} = 2^N-1$. If the ϕ_3 mapping makes a similar choice, a and b, such

that $a = e_2$, $b = e_{2N-3}$; $x_2 = e_2$ and $x_{2N-3} = e_{2N-3}$

then $e_2 < e_{2N-3}$ since $2 < 2^N-3$; $N > 2$

$$e_2 = \overline{e}_{2N-3}$$

and $x_2 = \overline{x}_{2N-3}$

So ϕ_3 is a homomorphism in the same manner as ϕ_1 and ϕ_2, and there are clearly 2^N-1 choices for ϕ.

A summary of the results of Theorem 2.2 is shown in Table 2.3.

Table 2.3. The Possible Homomorphic Mappings of B(2) and P(m)

N	m	Mappings	2^{N-1}	
2	4	$B(2) : P(4)_{0,3}$	2	
		$B(2) : P(4)_{1,2}$		
3	8	$B(2) : P(8)_{0,7}$	4	
		$B(2) : P(8)_{1,6}$		
		$B(2) : P(8)_{2,5}$		
		$B(2) : P(8)_{3,4}$		
4	16	$B(2) : P(16)_{0,15}$	8	
		$B(2) : P(16)_{1,14}$		
		$B(2) : P(16)_{2,13}$		
		$B(2) : P(16)_{3,12}$		
		$B(2) : P(16)_{4,11}$		
		$B(2) : P(16)_{5,10}$		
		$B(2) : P(16)_{6,9}$		
		$B(2) : P(16)_{7,8}$		
'	'	'	'	
'	'	'	'	
'	'	'	'	
'	'	'	'	
N	2^N	$B(2) : P(2^N)_{i,j}$	2^{N-1}	,i = o
		$B(2) : P(2^N)_{i+1,j-1}$		j = m-1
		' '		
		' '		
		' '		
		$B(2) : P(2^N)_{i+2^{N-1}-1,j-2^{N-1}-1}$		

Hardware Realizations Using I^2L

Integrated injection logic, commonly referred to as I^2L was developed in 1972 by separate groups at IBM and Phillips Research Laboratories [14]. I^2L offers the speed of the bipolar junction transistor (BJT) and the high packaging density and low power dissipation of the metal oxide semiconductor transistor (MOS). Since isolation borders and diffused resistors are not required, packaging densities of over 300 gates/mm^2 are possible [12]. The basic building block is a current mirror which is essentially direct-coupled transistor logic operating in the current mode. Figure 2.2 is the schematic of the direct-coupled transistor logic (DCTL) which is used in MOS large scale integration. Since the bases of Q_3 and Q_4 are connected together they can be

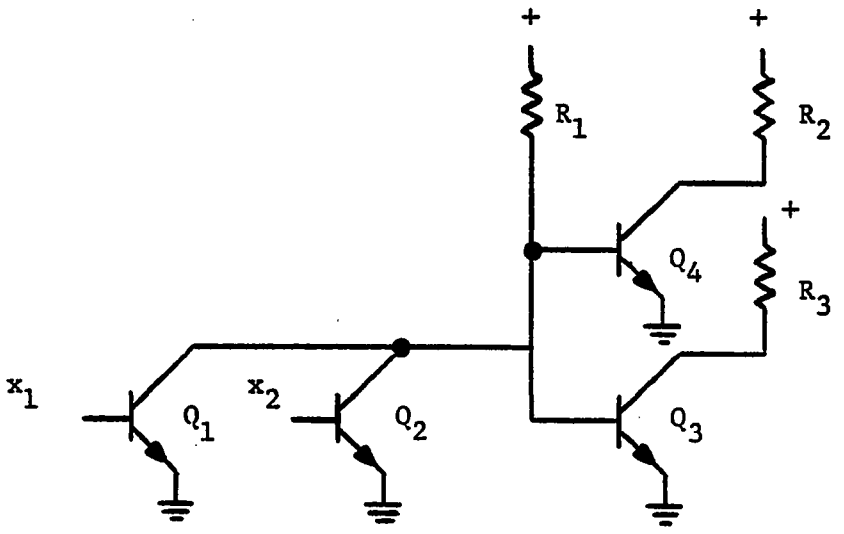


Figure 2.2. The Direct-Coupled Transistor Logic Gate

merged into a single positive region. Also, the emitters of Q_3 and Q_4 can be merged into a single negative region. Figure 2.3 shows a cross section of the output stage after the merging of Q_3 and Q_4. An additional P isolation diffusion area is required or all the collectors of Q_1, Q_2, Q_3, and Q_4 would be in a single N epitaxial region [12]. However, since all collectors are not at the same potential this is impossible. All emitters are at ground potential.

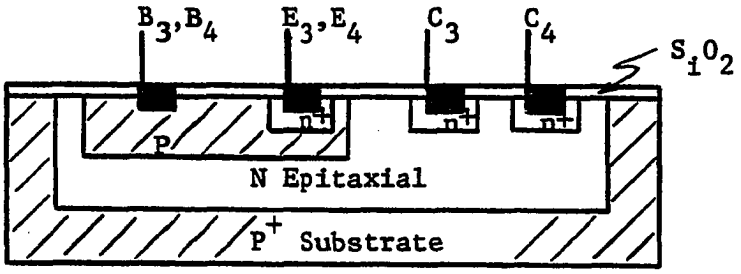


Figure 2.3. A Cross Section of the Output Stage

The P isolation diffusion area may be omitted if the collectors and emitters are interchanged. Thus, a multiple emitter transistor can be made and it can be operated in the inverse mode. One of the disadvantages of DCTL has now been eliminated. Since Q_3 and Q_4 now share the same base-emitter junction, their input characteristics are identical. Figure 2.4 is the new cross section of the output stage.

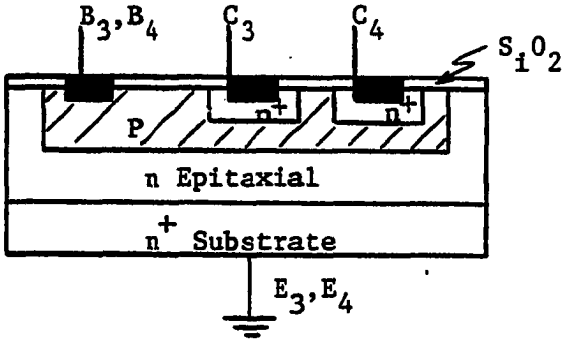


Figure 2.4. The Output Stage with a Multiple Emitter Device Operating in the Inverse Mode

Ideally, R_1 should be eliminated as the collector load for Q_1 and Q_2 since it requires a lot of the LSI silicon real estate. The larger the value of R_1 the better the gain of the input states. However, an infinite value will not do since this resistor must also provide the base current for the output transistors. R_1 may be replaced by a current source. A PNP transistor operating in the common base mode will provide the required source. Figure 2.5 shows the schematic of the current source. R_1 may now be off of the chip as opposed to a diffused resistor. Thus, the current injected into the output stages is completely controlled by the value of R_1 and Vcc.

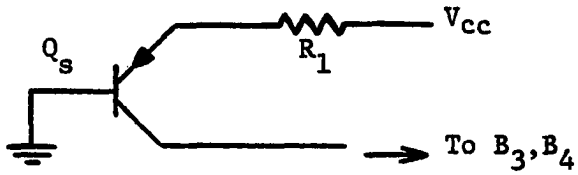


Figure 2.5. A PNP Current Injector

If the collector of the current injector is merged with the bases of Q_3 and Q_4 in Figure 2.4 and the base of the injector is merged with the grounded emitters, the I^2L gate is complete. The final cross section of both the output and input stage is shown in Figure 2.6.

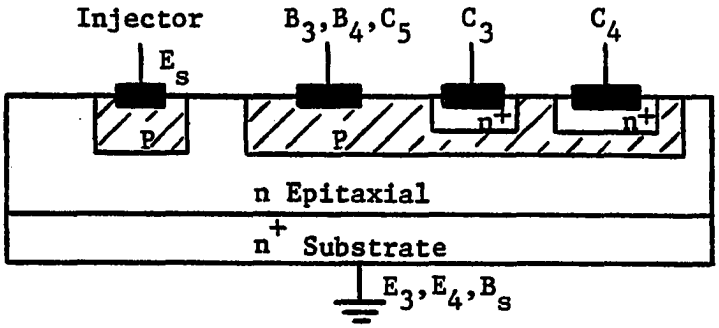


Figure 2.6. A Cross Section of the I^2L Gate

The input to the I^2L gate can connect to the junction of B_3, B_4 and C_5. The schematic of the cross section in Figure 2.6 is shown in Figure 2.7. All of the original labels have been carried forward from Figure 2.2 so

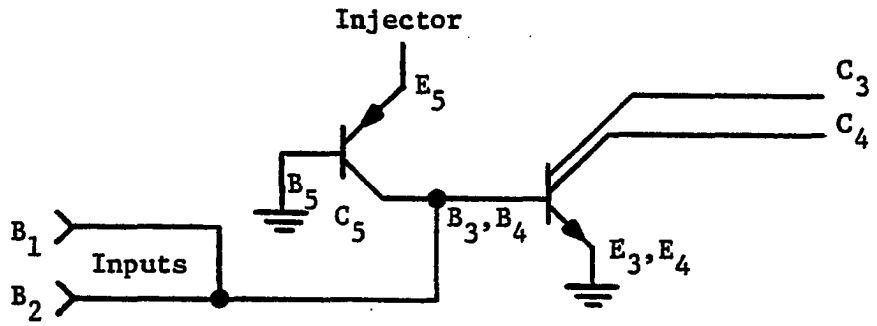


Figure 2.7. The I^2L Gate

that the development from the basic DCTL gate can easily be seen. In actual construction a single external resistor can be connected to E_5 to provide the injector current for all gates. The P region for the injector emitter has a lateral configuration. The P-type diffusion line can connect to all the output transistors which it is adjacent to. These are called injector rails. The vertical real estate is allocated to the output NPN transistors. In this way the actual gate layout may grow in two dimensions to allow an organized LSI device. Figure 2.8 shows a layout of four I^2L gates. Each square represents an input and each circle represents and output. In actual construction, a gate array may contain a larger number of multiple-output I^2L inverters grouped together. A typical gate array consisting of eight multiple output inverters sharing a set of four injectors is currently manufactured by companies offering I^2L master chips [15].

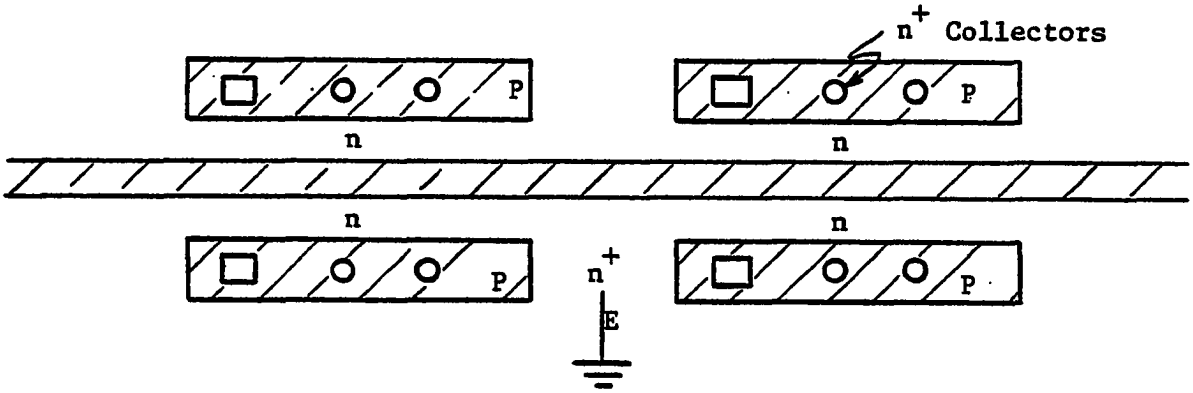


Figure 2.8. Injection Logic Layout for Four I^2L Gates

Complement Gate

Hardware realizations for the two homomorphic mappings given in Table 2.2 are shown in Figure 2.9. The inputs and outputs from the gate will be characterized using the following notation:

Given logical inputs $(x_1, x_2, \ldots x_n)$ and logical outputs $(y_1, y_2, \ldots y_n)$, then for inputs

$0 < x_1, x_2 \ldots x_N < i_{m-1}$ there exists a corresponding output

$0 < y_1, y_2, \ldots y_N < i_{m-1}$ such that y_1 is the response to x_1, y_2 is the response to x_2, etc.

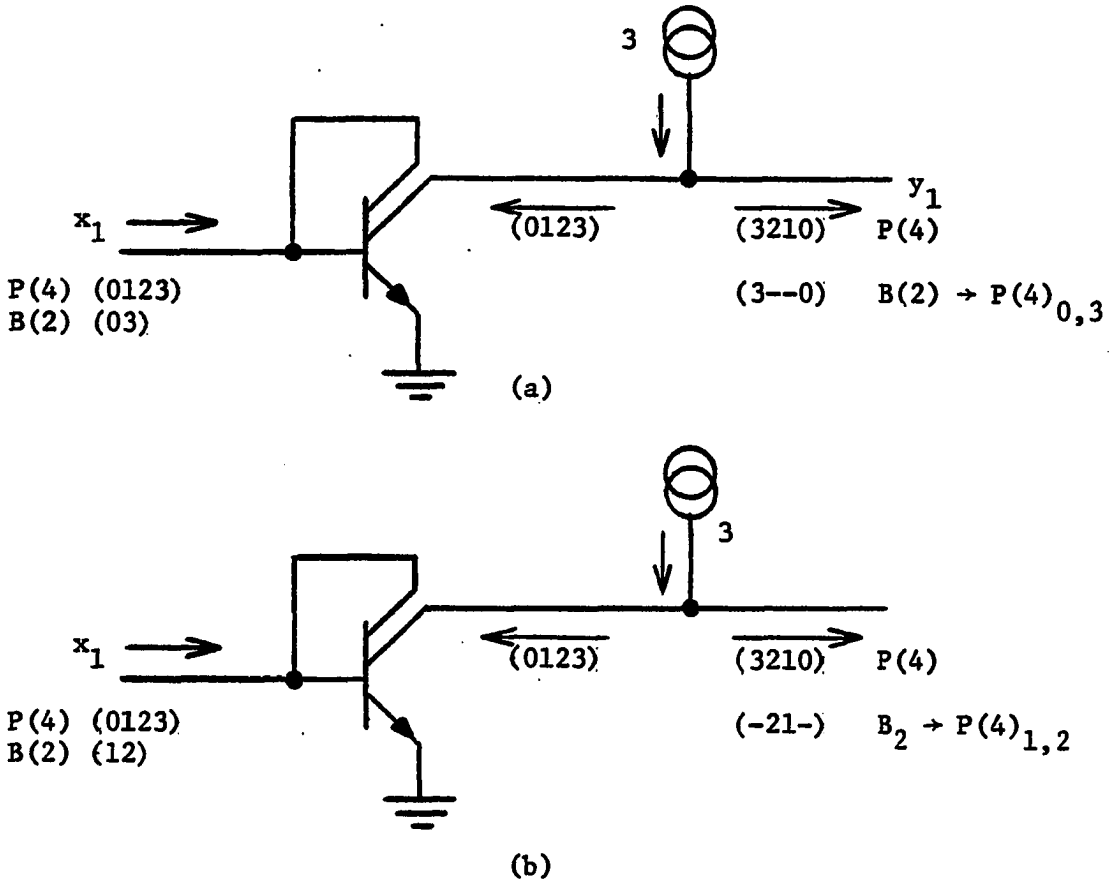


Figure 2.9. Complement Gates for Two Mappings
a) $B_2 \rightarrow P(4)_{0,3}$, b) $B_2 \rightarrow P(4)_{1,2}$

Note that for either mapping, the single current mirror which was developed and shown in Figure 2.7 is capable of realizing the complement function in both B(2) and P(4) with no additional gate burden. In Figure 2.9(a) it is shown that for an input current of $x_1 = 2$, for example, that the second collector; that is the one not connected for feedback, will mirror the input. With two units of current flowing to the left out of the three unit current source, only one unit of current is free to flow to the right; therefore, $y_1 = 1$ for $x_1 = 2$. In a

similar manner, the gate in Figure 2.9(b) can accomplish the complement in P(4) or the $B(2) \rightarrow P(4)_{1,2}$ mapping. Notice that the gates in (a) and (b) are identical.

The MAX/OR Gate

The I^2L realization for a single radix (r = 4) $\overline{MAX}$ gate was first proposed in 1977 [5]. Figure 2.10 is the $\overline{MAX}$ gate followed by the inverter shown in Figure 2.9. The MAX function was closed in P(4) for all six mappings shown in Figure 2.1. Therefore, the single radix gate can accomplish the "OR" operation in B(2) for any mapping and thus become a dual radix gate with no additional gate burden. Shown in Table 2.4 is the truth table of the actual circuit currents which characterize the behavior in P(4) as well as both of the homomorphic mappings.

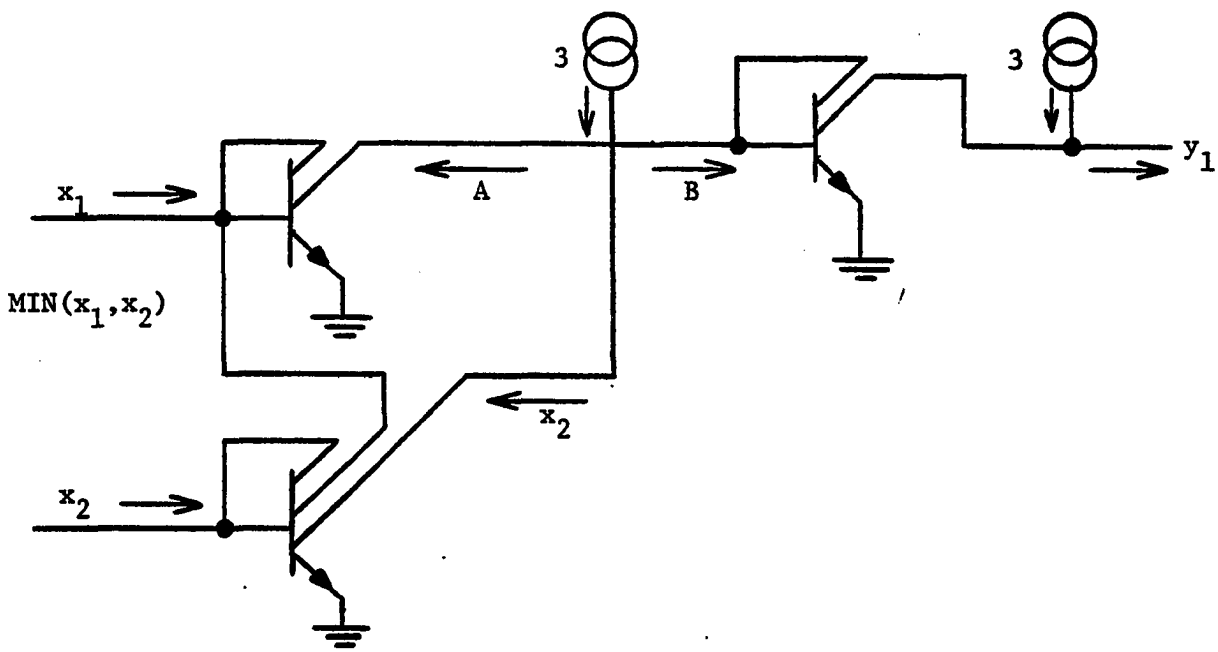


Figure 2.10. MAX Operation in P(4), "OR" Operation for All B(2) Mappings

Table 2.4. Current Characterization of the Dual Radix MAX/OR Gate

x_1x_2	A	B	y_1	$P(4)_{0,3}$	$P(4)_{1,2}$
00	0	3	0	0	X
01	0	2	1	X	X
02	0	1	2	X	X
03	0	0	3	3	X
10	1	2	1	X	X
11	0	2	1	X	1
12	0	1	2	X	2
13	0	0	3	X	X
20	2	1	2	X	X
21	1	1	2	X	2
22	0	1	2	X	2
23	0	0	3	X	X
30	3	0	3	3	X
31	2	0	3	X	X
32	1	0	3	X	X
33	0	0	3	3	X

The MIN/AND Gate

If the x_1 and x_2 inputs to the circuit in Figure 2.10 are complemented and the output is not complemented, then the circuit will perform the MIN function in P(4). The MIN function was closed in P(4) for all six mappings shown in Figure 2.1. Therefore, the single radix gate can accomplish the "AND" operation in B(2) for any mapping and thus become a dual radix gate with no additional gate burden. The circuit shown in Figure 2.11 will accomplish the MIN operation in P(4) and the "AND" operation in B(2) for all mappings. Table 2.5 is the truth table of the actual currents which characterize the MIN function in P(4) as well as the $B(2) \rightarrow P(4)_{1,2}$ and $B(2) \rightarrow P(4)_{0,3}$ homomorphic mappings.

A functionally complete set of gates now exists for either of the binary mappings in P(4) with the complement, MAX/OR gate, and MIN/AND gate. The $B(2) \rightarrow P(4)_{0,3}$ and $B(2) \rightarrow P(4)_{1,2}$ mappings can both be

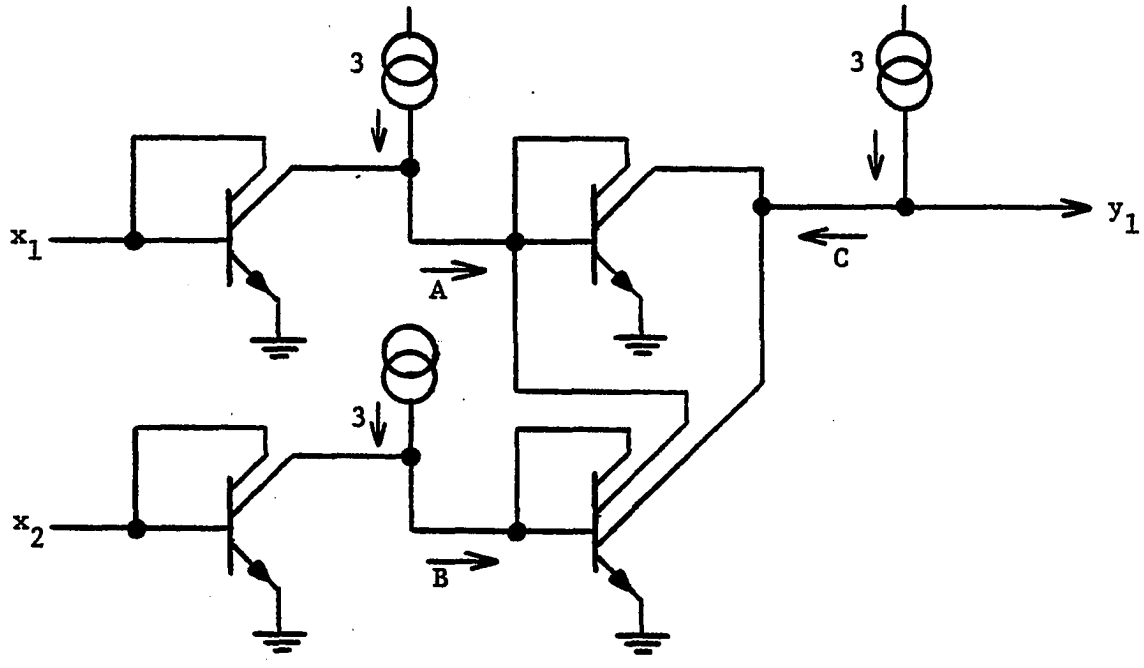


Figure 2.11. MIN Operation in P(4), "AND" Operation for all B(2) Mappings

realized with no additional hardware burden. To establish functional completeness in P(4), it is necessary to construct circuitry which can perform the unary operations given in Definitions 1.4 and 1.5. Either of the unary operations are sufficient for P(4) completeness. Circuit design and dual radix considerations will be examined for each definition.

The Unary "Inverter"

To achieve a completeness in P(4) Vranesic chose the unary operators given in Definition 1.4. For convenience, that definition will be repeated here.

Definition 1.4. Given $x,k \epsilon S$, x a variable in S, k a constant in S

a) The M unary "inverter" operation is

$x^k = k$ if $x = o$

$x^k = o$ otherwise

The truth table for the unary inverter is given in Table 2.6. Examining the truth table reveals that the $B(2) \rightarrow P(4)_{0,3}$ mapping holds for the unary "inverter" while the $B(2) \rightarrow P(4)_{1,2}$ mapping does not. In considering the consequence of the $B(2) \rightarrow P(4)_{0,3}$ mapping, it is important to remember that the unary "inverter" definition is necessary for the multivalued case only. For the binary case it is obviously not required. However, in an effort to insure common signal paths and enhance the tightly coupled feature of any dual radix circuitry, a complete analysis of the $B(2) \rightarrow P(4)_{0,3}$ mapping should be done. The truth table for the binary case is given in Table 2.7. The conventional realization of this truth table with binary gates is shown in Figure 2.12. The function can be

Table 2.5. Current Characterization of the Dual Radix MIN/AND Gate

x_1x_2	A	B	C	y_1	$P(4)_{0,3}$	$P(4)_{1,2}$
00	3	3	3	0	0	x
01	3	2	3	0	x	x
02	3	1	3	0	x	x
03	3	0	3	0	0	x
10	2	3	3	0	x	x
11	2	2	2	1	x	1
12	2	1	2	1	x	1
13	2	0	2	1	x	x
20	1	3	3	0	x	x
21	1	2	2	1	x	1
22	1	1	1	2	x	2
23	1	0	1	2	x	x
30	0	3	3	0	0	x
31	0	2	2	1	x	x
32	0	1	1	2	x	x
33	0	0	0	3	3	x

Table 2.6. Unary "Inverter" in P(4)

x^k	0	1	2	3
0	0	1	2	3
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0

generated using either the "AND" gate or the "OR" gate. This interpretation of the unary "inverter" function is awkward for two reasons. The first being the fact that the "AND" and "OR" gates are mapped to MIN and MAX gates in P(4). The second reason is that the function can be realized with fewer gates by using a different interpretation of the truth table. For the case of k = 0, the function is trivial. For k = 1, the output is the complement of the input. For the binary case, then, the unary "inverter" may be thought of as a gated inverter. With k enabled the circuit performs the complement. With k = 0 the circuit is disabled. Figure 2.13 shows the gated inverter interpretation of the unary "inverter" for the binary situation and the $B(2) \rightarrow P(4)_{0,3}$ mapping.

Table 2.7. Unary "Inverter" in B(2)

x	k	x^k
0	0	0
0	1	1
1	1	0
1	0	0

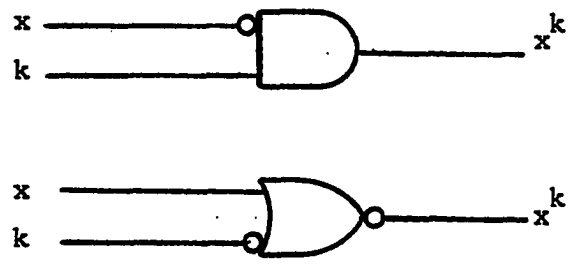


Figure 2.12. B(2) Unary "Inverter" Realization

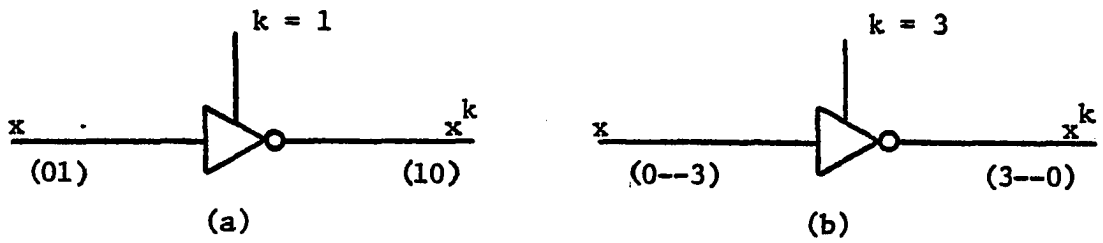


Figure 2.13. Gated Inverter a) B(2), b) B(2) → $P(4)_{0,3}$

Using this idea of a gated inverter, the unary "inverter" realization in P(4) should perform the gated inverter function for B(2) → $P(4)_{0,3}$. The I^2L circuit shown in Figure 2.14 is capable of performing the unary "inverter" function in P(4). It is also possible for it to perform as a gated inverter for the B(2) → $P(4)_{0,3}$ case. A comment concerning Definition 1.4(a) is in order. The idea of the definition is that k is a constant in the algebra. For the I^2L circuit in Figure 2.14, the k input should be determined by a fixed injector current to be compatible with the definition. This would present a problem for dual radix circuitry. If the value of k was fixed to a logical one or two, then the unary "inverter" in Figure 2.14 could not operate as a gated inverter for the B(2) → $P(4)_{0,3}$ case. However, the circuit of Figure 2.14 has a greater flexibility. The value of k can

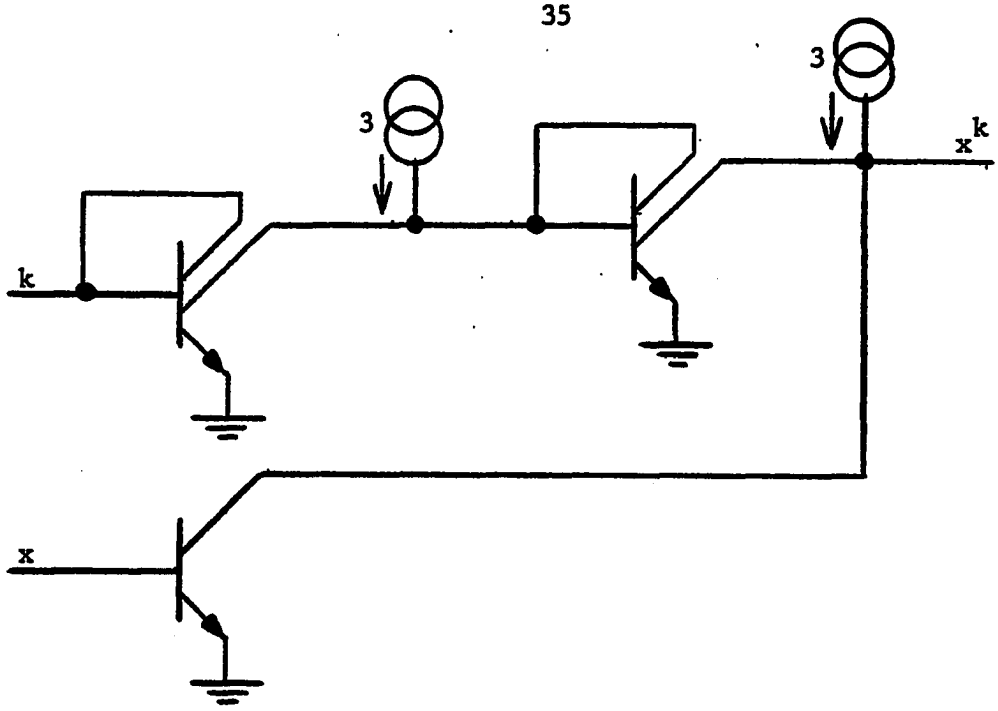


Figure 2.14. Unary "Inverter"

default to a logical value of three for the binary case. This requires the burden of a mode control line to accomplish the dual radix concept. The suggestion that dual radix circuitry requires a mode line, and thus places additional burden on the circuitry, will force this type of function to be considered more carefully in Chapter III. The idea of common signal paths and tightly coupled implementations is still a bit naive at this point and may prove to be of little value if the shared function is an awkward pairing for the dual radix concept.

The M Unary "Clockwise Cycling" Operation

The second half of Definition 1.4 describes the "clockwise cycling" operation.

Definition 1.4 b) The M unary "clockwise cycling" operation is

$$x^{\overset{r}{\rightarrow}} = (x \oplus r) \bmod M \quad x \text{ a variable in } S, \quad r \text{ a constant in } S.$$

The truth table for the "clockwise cycling" operation is given in Table 2.8. Examining the table shows that neither of the homomorphic mappings of B(2) to P(4) hold. However, there is a column of the table which is closed. For the case of r = o and the $B(2) \rightarrow P(4)_{0,3}$ mapping, the function becomes a non-inverting buffer. If the r input to a P(4) "clockwise cycling" operation would default to a logical value of zero for the binary case, then common signal paths and the tightly coupled concept would be preserved. The idea is similar to the gated inverter using the k line as the mode control. In this case, the r input would be the mode control line. Once again the notion of r being a constant in the algebra is lost here. This does not change the function in P(4) to allow r to become a logical zero for the binary case. On the contrary, the dual radix pairing of functions is very convenient. That is, the "clockwise cycling" operation is simply a non-inverting buffer in either radix for r = o. The circuit shown in Figure 2.15 will function as a non-inverting buffer. The i_1 and i_2 current sources are shown to be a function of r.

Table 2.8. "Clockwise Cycling" Operation in P(4)

x \ r	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

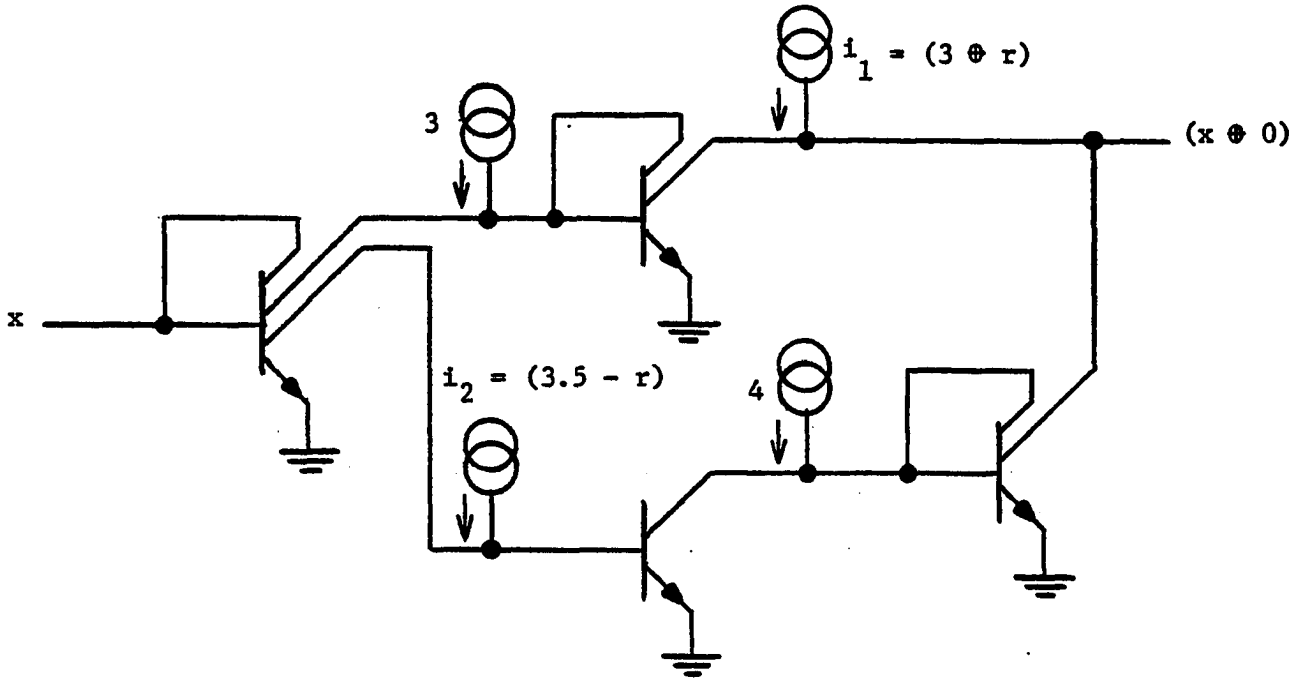


Figure 2.15. Non-Inverting Buffer (r = 0) for B(2) and P(4)

The non-inverting buffer is a special case of Definition 1.4(b) to accommodate the dual radix concept for the $B(2) \rightarrow P(4)_{0,3}$ case. Another special case of "clockwise cycling" has been defined [5] as a successor function. For the successor function r has a value of one. Rather than constructing several different circuits with different current sources for i_1 and i_2 to perform the special case functions as

well as the "clockwise cycling" function, the circuit in Figure 2.16 should be considered. This circuit will perform the M unary "clockwise cycling" operation as defined. The value of r could obviously be held at a constant logical value to satisfy the intent of the definition. For the dual radix case it is especially attractive to at least allow r to default to a logical zero for the binary case.

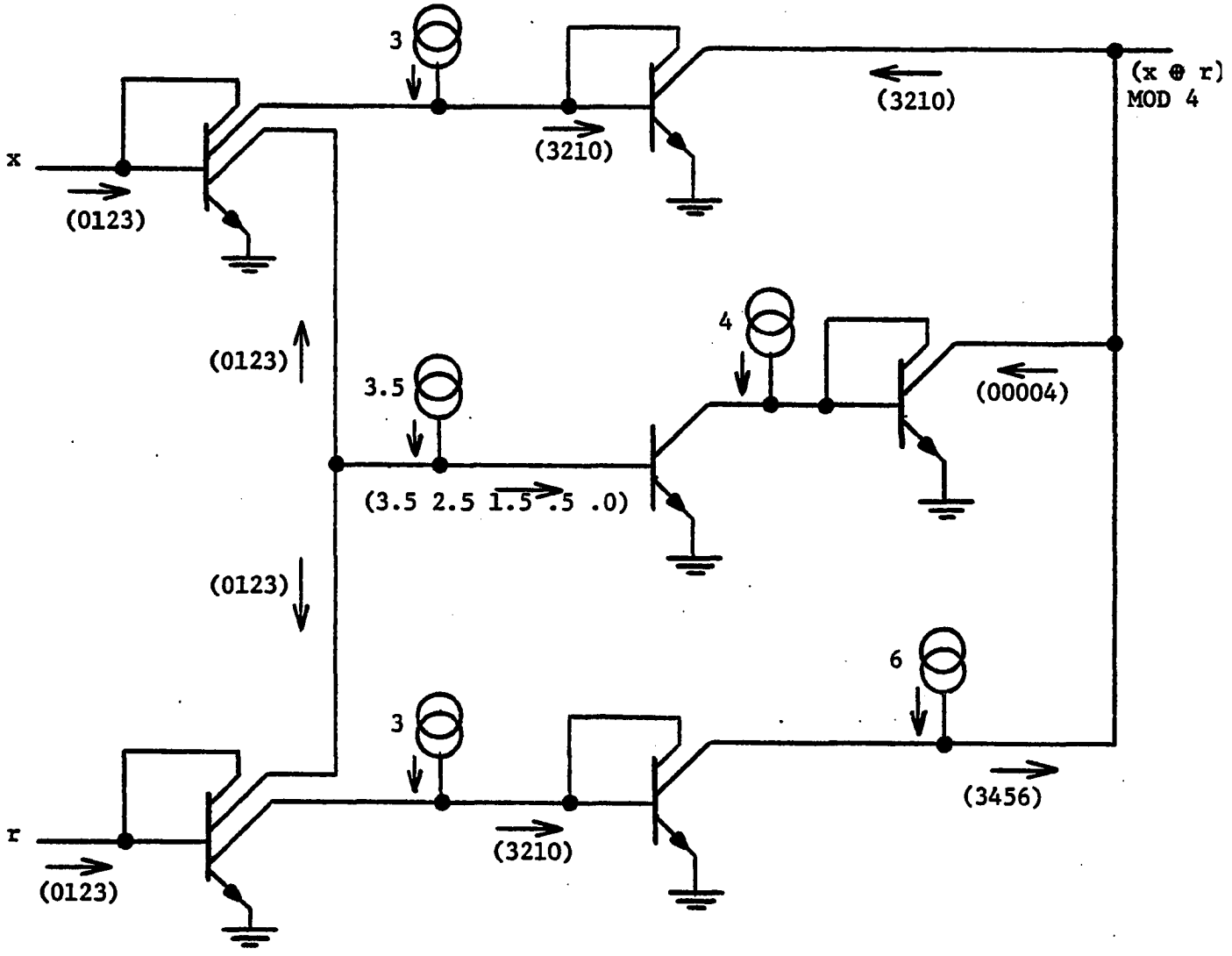


Figure 2.16. Universal M Unary "Clockwise Cycling" Operation

The Literal

To achieve a completeness in P(m) Allen and Givone chose the unary operator given in Definition 1.5. For convenience that definition will be repeated here.

Definition 1.5. Given $x,a,b \in S$, x a variable in S , a,b constants in S , $a \leq b$

The unary operator (a,b) on the variable x, called a literal and denoted by ${}_a x^b$ is:

$$ {}_a x^b = \begin{cases} m-1 & a \leq \text{logical value of } x \leq b \\ o & \text{otherwise} \end{cases} $$

The truth table for the literal operator is given in Table 2.9. Since the output is always a logical zero or logical three, the $B(2) \rightarrow P(4)_{0,3}$ mapping holds. Notice for this mapping that the a and b inputs may take on values of {0,0}, {0,3}, and {3,3} since $a \leq b$ by Definition 1.5. Once again, there is a binary operation for each of these three choices. For the {0,0} case, the literal operation in P(4) becomes an inverter in B(2). For the {3,3} case, the function becomes a non-inverting buffer in B(2). The {0,3} is really a little awkward since it disables the operation with a logical one output for the B(2) case. The difficulty with the literal operator is with the physical realization of the truth Table 2.9. Traditionally, only a single column of the table is realized in a given circuit. If common signal paths are to be achieved, then the B(2) signal must be able to propagate through any P(4) literal realization. To accomplish this, a universal literal realization must be designed which

Table 2.9. The Literal Operator in P(4)

$a_x b$

x \ ab	00	01	02	03	11	12	13	22	23	33
0	3	3	3	3	0	0	0	0	0	0
1	0	3	3	3	3	3	3	0	0	0
2	0	0	3	3	0	3	3	3	3	0
3	0	0	0	3	0	0	3	0	3	3

can perform the functions of all ten of the columns given in Table 2.9. If such a circuit can be designed, then the a and b inputs could default to {0,0} or {3,3} depending upon the desired circuit response in B(2). The circuit shown in Figure 2.17 is capable of performing the literal operation in P(4) for any choice of a and b, as well as performing the inverting or non-inverting function in B(2) for the $B(2) \rightarrow P(4)_{0,3}$ mapping for a and b equal to {0,0} and {3,3} respectively. For a and b equal to {0,3} the function is obviously preserved for either radix; however, its usefulness in B(2) is doubtful.

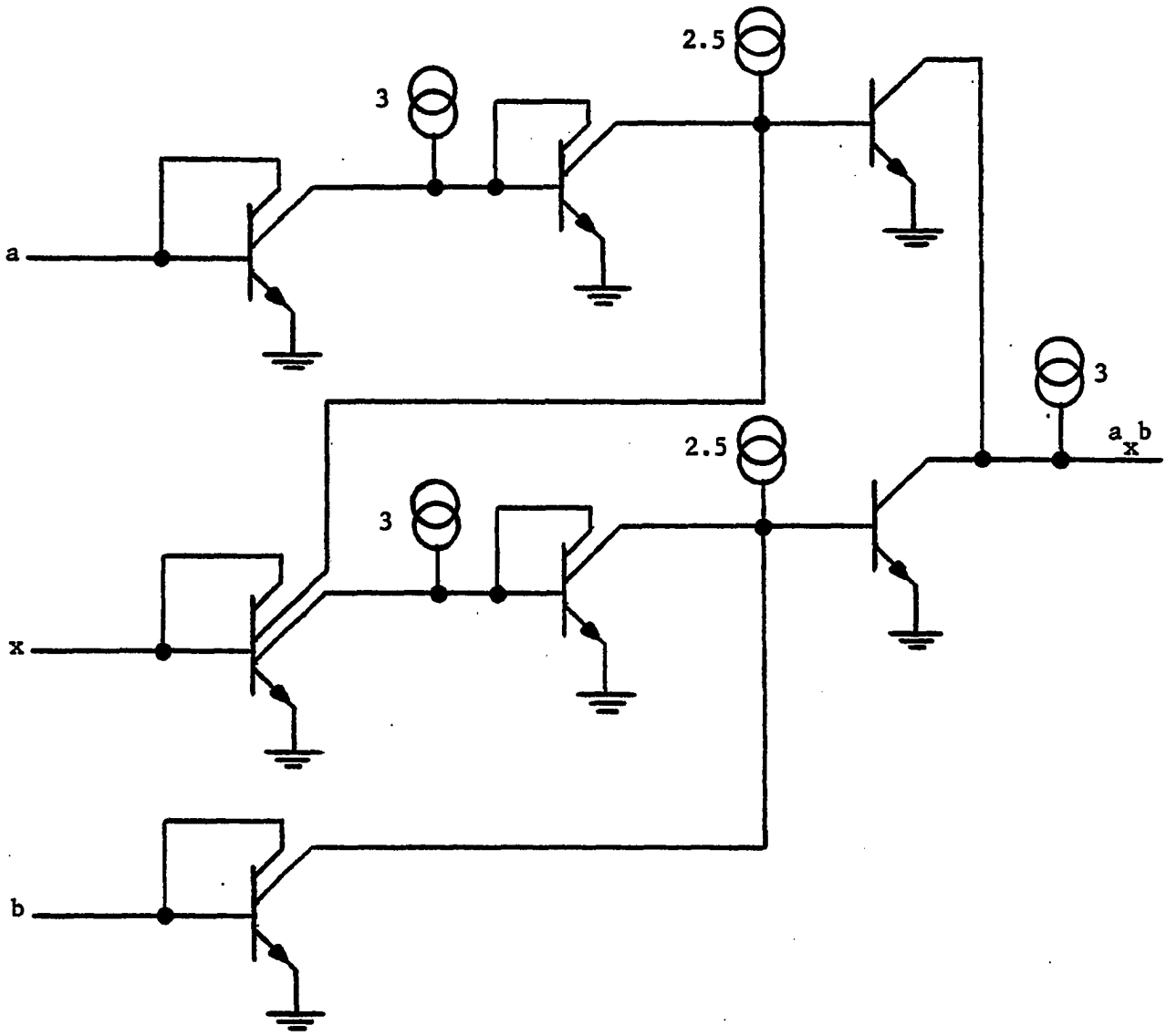


Figure 2.17. The Literal Operator in P(4),
Inverter in B(2) → $P(4)_{0,3}$ for a,b = {0,0}
Non-Inverter in B(2) → $P(4)_{0,3}$ for a,b = {3,3}

Summary

Out of the six possible mappings from B(2) to P(4), it was shown that using Definition 2.3 only one mapping is homomorphic, the $B(2) \rightarrow P(4)_{0,3}$. If Definition 2.5 is used to define the pseudo-complement or strong negation, then two homomorphic relationships exist; the $B(2) \rightarrow P(4)_{1,2}$ as well as the $B(2) \rightarrow P(4)_{0,3}$. The general extension of this argument given by Theorem 2.2 states that there exist 2^{N-1} homomorphic mappings of B(2) in P(m), $m = 2^N$, N an integer greater than one. Table 2.3 is a summary of the mappings defined by the theorem.

The advantages of integrated injection logic were discussed and the basic I^2L gate was developed. The extension of the basic gate architecture to an actual gate array was considered.

The complement, MAX, and MIN functions in P(4) were shown to realize the complement, "OR," and the "AND" functions in B(2). The MAX and MIN functions hold for all six mappings shown in Figure 2.1. The complement holds for the two homomorphic mappings. The circuitry to operate in either radix is identical. No mode control line is required to change from one base to the other. Also, there is no additional hardware burden. In a sense, then, the circuitry really has no natural radix because it will work in either.

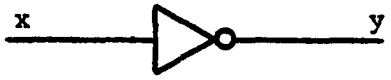
The unary operations defined by Vranesic, et al, and by Allen and Givone were examined under the dual radix concept. It is obviously not required that these circuits perform at all in B(2); however, in an effort to insure common signal paths and enhance the tightly coupled feature of any dual radix circuitry, a complete analysis was done. A functional pairing exists for the $B(2) \rightarrow P(4)_{0,3}$ mapping for each

unary operation defined for P(m). Table 2.10 is a summary of the functional pairing for dual radix circuitry.

The I^2L realizations for all of the operations summarized in Table 2.10 were developed in this chapter. A detailed analysis of the development and all of the electrical design considerations is given in Appendix A, Dual Radix I^2L Circuit Considerations. Also, some of the I^2L circuits are too awkward to be redrawn many times. Therefore, the schematic symbols shown in Figure 2.18 will be used throughout the remainder of the text to represent the actual circuitry.

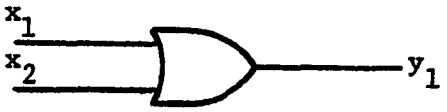
Table 2.10. Summary of Functional Pairing for Dual Radix Circuitry

Algebra	Function in P(4)	Function in B(2)	Possible Mapping	Mode Control Required	Figure
Vranesic	"Unary Inverter"	Inverter	$P(4)_{0,3}$	k = 3	2.14
Et Al	"Clockwise Cycling"	Non-Inverting Buffer	$P(4)_{0,3}$	r = 0	2.16
Allen and	Literal	Inverter	$P(4)_{0,3}$	a,b = {0,0}	2.17
Givone	Literal	Non-Inverting Buffer	$P(4)_{0,3}$	a,b = {3,3}	2.17
	MAX	OR	ALL	NONE	2.10
Both	MIN	AND	ALL	NONE	2.11
	Complement (Def.2.5)	Complement	$P(4)_{1,2}$; $P(4)_{0,3}$	NONE	2.9



(Figure 2.9)

a) Complement

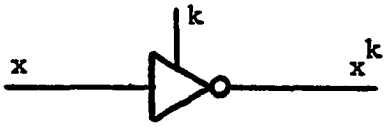


(Figure 2.10)

b) MAX/OR Gate

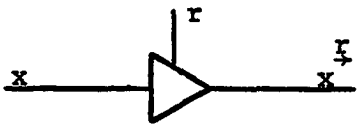
(Figure 2.11)

c) MIN/AND



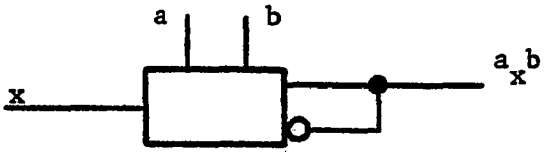
(Figure 2.14)

d) "Unary Inverter"/Inverter



(Figure 2.16)

e) "Clockwise Cycling"/
Non-Inverting Buffer



(Figure 2.17)

f) Literal/Inverter (a,b = {0,0})
/Non-Inverting Buffer (a,b = {3,3})

Figure 2.18. Schematic Symbols for Dual Radix Circuitry
a) Complement, b) MAX/OR Gate, c) MIN/AND Gate, d) "Unary Inverter"/
Inverter, e) "Clockwise Cycling"/Non-Inverting Buffer
f) Literal/Inverter; Literal/Non-Inverting Buffer

CHAPTER III

B(2):P(4) COMBINATIONAL DESIGN

The motivation in Chapter II was to choose an algebra or algebras and realizations that would be functionally complete in two radices. The concept is a stronger marriage than equivalence. The B(2) hardware is available in the P(4) realization. Hopefully, the advantages of a B(2):P(4) realization outweigh the disadvantages. Although no combinational design has been done, as yet, it might be nice to examine the pros and cons of this dual radix concept before continuing.

Examination of the burden in P(4) which is imposed by one or more of the homomorphic images of B(2) is in order. Figure 2.1 clearly shows no additional hardware burden is imposed on MAX/OR and MIN/AND B(2):P(4) realizations for any of the six mappings. Definition 2.5 reduces the mappings from six to two unless some independent realization of the complement in one of the radices is added, which is undesirable. Thus, the complement requires no hardware burden in B(2) for B(2) $\rightarrow$ $P(4)_{0,3}$ and B(2) $\rightarrow$ $P(4)_{1,2}$. At this point completeness can be achieved in B(2) but not in P(4).

Vranesic chose the M unary "inverter" and the M unary "clockwise cycling" operations to allow completeness in P(4). The unary "inverter" shown in Figure 2.13 clearly requires k = 3 for the circuit to perform as an inverter in B(2)for $B(2) \rightarrow P(4)_{0,3}$. However, the schematic in Figure 2.14 shows that no extra hardware is required for the B(2) case. The unary "clockwise cycling" operation will perform as a non-inverting buffer with r = 0 for the $B(2) \rightarrow P(4)_{0,3}$ mapping. The circuit shown in Figure 2.15 requires three current mirrors and a switch compared to two current mirrors and two switches for the circuit developed by Dao, McCluskey and Russell [5]. Counting collectors, or fan out, Figure 2.15 requires eight compared to seven in the circuitry by Dao, et al. Forcing the hardware to work in B(2):P(4) configurations demands that r default to zero for the B(2) case. This complicates the circuitry of Figure 2.15, but also provides the motivation for a more universal "clockwise cycling" gate which is shown in Figure 2.16. This circuit will operate for any r , r = {0, m-1}. Figure 2.16 requires five current mirrors and one switch, for a total of thirteen collectors. A direct comparison with this circuit and a successor circuit developed in previous work has little relevance.

Combinational Considerations

What must be addressed now is whether or not the mode control lines should be considered additional burden for B(2):P(4) designs. Although the collector count does not go up, there is the added requirement that the k and r lines default to three and zero respectively for the B(2) case. To better understand the burden of mode control, or defaulting to a constant logic value for B(2), two specific combinational

designs will be considered. The first design will be for the algebra of Vranesic, et al, and the second for Allen and Givone.

Example 3.1. Given the completely specified function of x_1 and x_2 in Table 3.1, find the P(4) realization from the canonical expression. Show the B(2):P(4) circuit and discuss the k and r mode line considerations.

Table 3.1

x_1 \ x_2	0	1	2	3
0	2	2	2	3
1	1	2	3	0
2	1	0	0	0
3	2	1	1	0

Any n-variable m-valued switching function $f(x_1,\ldots x_n)$ has a "sum of products of sums" canonical form which is described in [26]. For the function given in Table 3.1 the canonical form is:

$$f(x_1,x_2) = [(\overset{3}{\rightarrow}{x_1} + x_2)(\overset{2}{\rightarrow}{x_1} + x_2)(\overset{\rightarrow}{x_1} + \overset{3}{\rightarrow}{x_2})(\overset{\rightarrow}{x_1} + \overset{2}{\rightarrow}{x_2})]^1$$
$$+ [(x_1 + x_2)(x_1 + \overset{3}{\rightarrow}{x_2})(x_1 + \overset{2}{\rightarrow}{x_2})(\overset{3}{\rightarrow}{x_1} + \overset{3}{\rightarrow}{x_2})(\overset{\rightarrow}{x_1} + x_2)]^2$$
$$+ [(x_1 + \overset{\rightarrow}{x_2})(\overset{3}{\rightarrow}{x_1} + \overset{2}{\rightarrow}{x_2})]^3$$

Using the theorems in [1] the function can be simplified to:

$$f(x_1,x_2) = \overset{3}{\rightarrow}{x_1}\, x_2^2 + \overset{2}{\rightarrow}{x_1}\, x_2 + x_1^3\, x_2$$

The P(4) realization of the above function is shown in Figure 3.1.

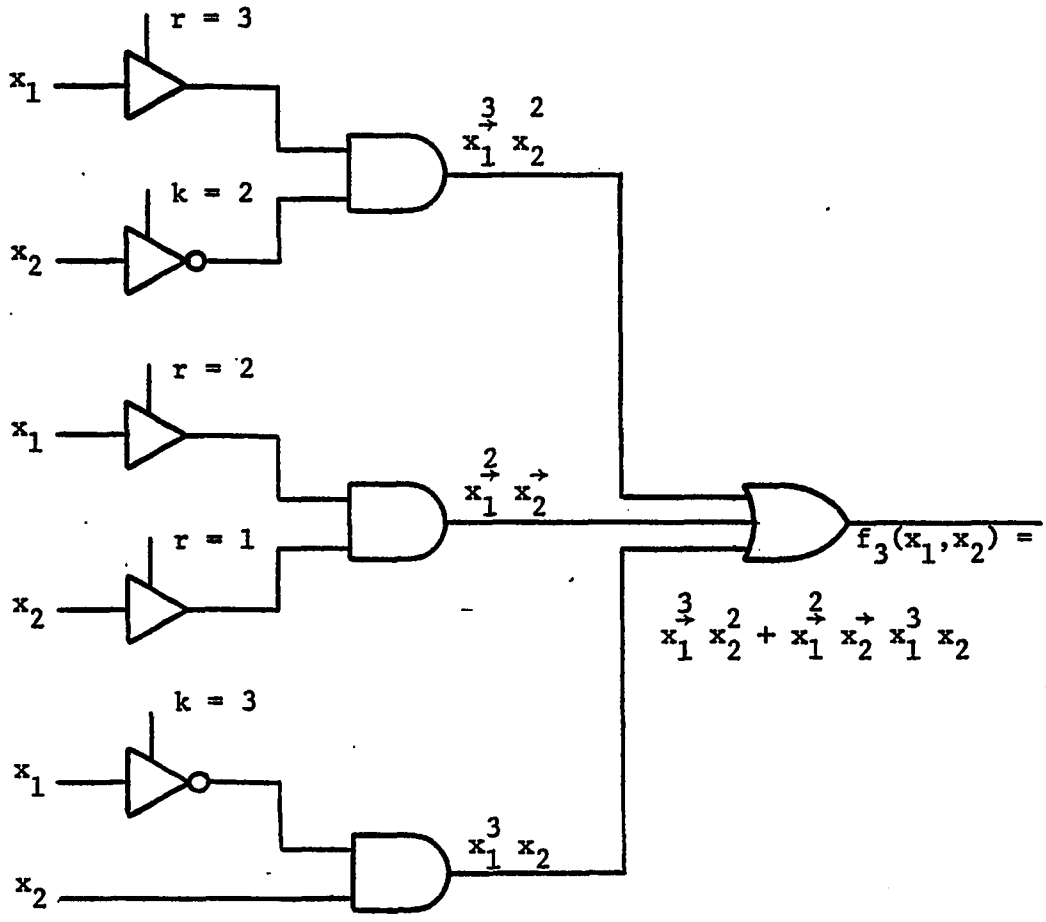


Figure 3.1. P(4) Realization of

$$f(x_1,x_2) = \overset{3}{\rightarrow}\ x_1^2\ x_2 + \overset{2}{\rightarrow}\ \overset{\rightarrow}{x_1}\ x_2 + x_1^3\ x_2$$

It has been shown that if r = 0 and k = 3, then a $B(2) \rightarrow P(4)_{0,3}$ mapping can be realized with the circuitry in Figure 3.1. Figure 3.2 shows the P(4) realization of Figure 3.1 with the mode control lines configured for the B(2) case.

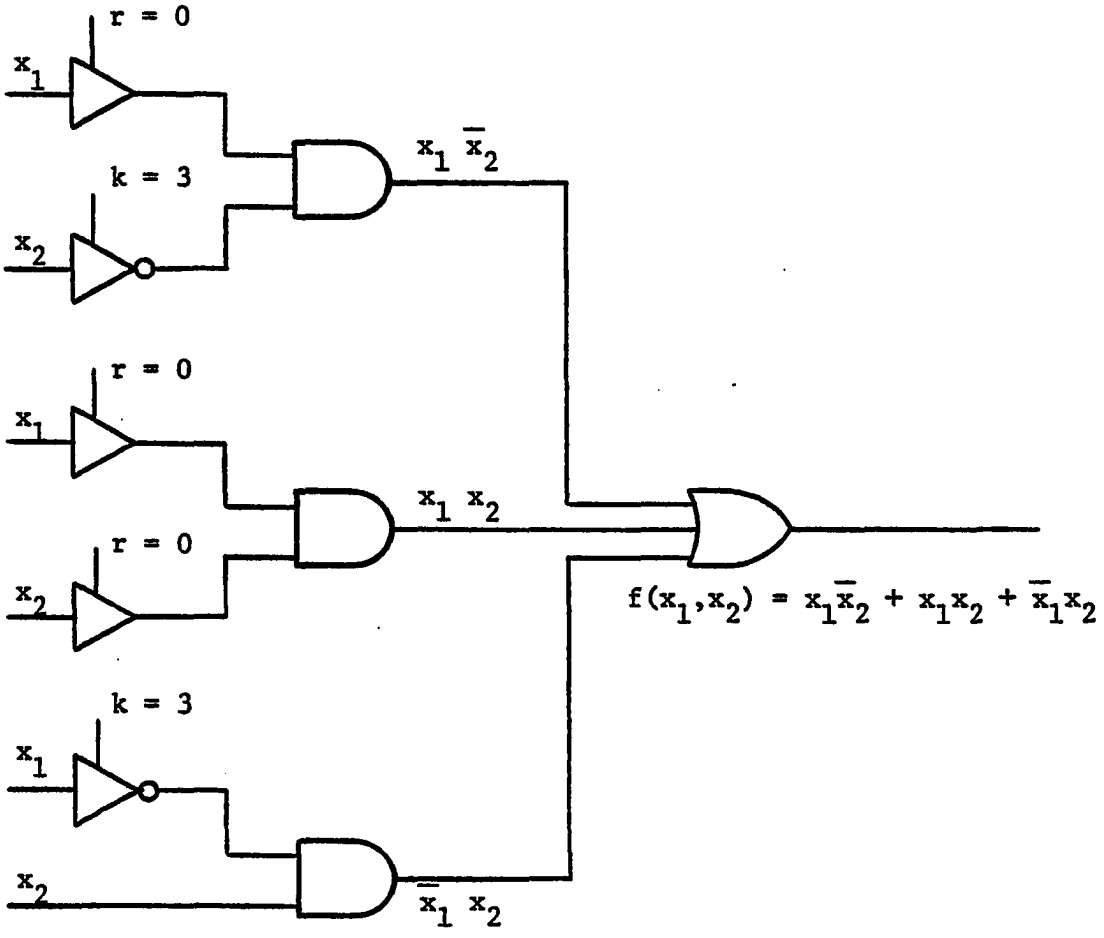


Figure 3.2. B(2):P(4) Realization of

$f(x_1,x_2) = x_1 \bar{x}_2 + x_1 x_2 + \bar{x}_1 x_2$

The function shown in Figure 3.2 can be simplified to:

$$f(x_1,x_2) = x_1 + x_2$$

Obviously, no simplification is possible in B(2) if the P(4) function of x_1 and x_2 is to be preserved. Therefore, a minimized P(4) realization does not guarantee the B(2) $\rightarrow$ P(4)$_{0,3}$ function will be minimized.

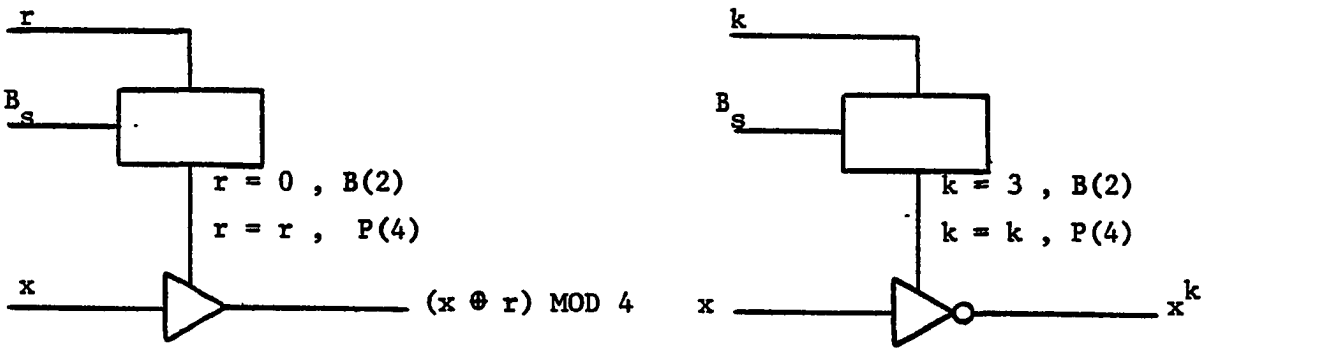
Upon closer examination of the B(2) function shown in Figure 3.2 it can be seen that the k and r lines are awkward to handle in going from P(4) to B(2). Rather than the k and r lines changing, a better method would be to define a control signal which changes the hardware from P(4) to B(2). In processor architectures, this control signal could operate in much the same way that the interrupt is turned on or off in a microprocessor [13]. That is, it could be enabled and disabled under the control of a software instruction so that the logic could change from P(4) to B(2) "on the fly." The logic shown in Figure 3.3 could be used to allow the control signal to change the k and r lines from variables in P(4) to k = 3 and r = 0 in B(2). The binary select line now becomes the single line which can be used where necessary to allow the B(2):P(4) circuitry to operate in both radices. Each of the circuits shown in Figure 3.3(c) contains five collectors. Also, the B_s line is an obvious burden that is necessary to support the B(2):P(4) concept. The conclusion is that the mode control line, or B_s line, places an additional burden of five collectors and one control lead on the hardware.

Besides additional hardware, two other concepts were introduced in Chapter II which seriously affect the appeal of any B(2):P(4) circuitry

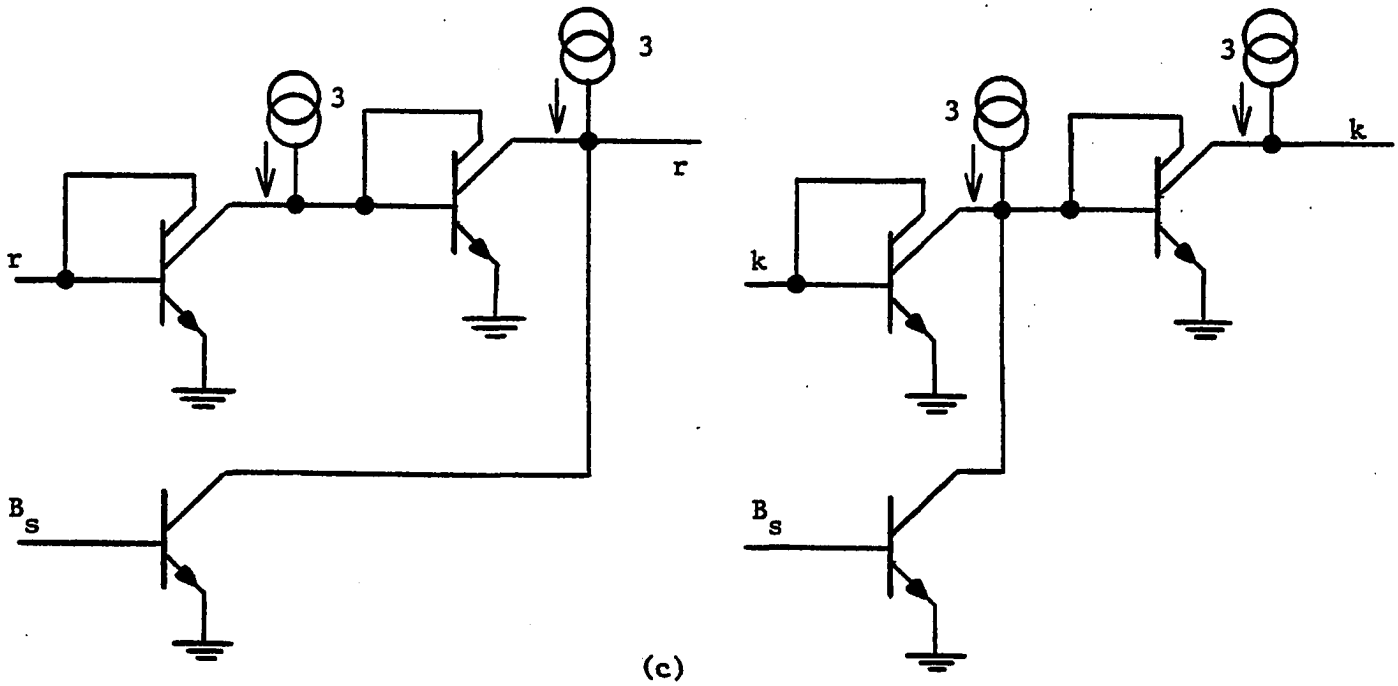
B_s	r		B_s	k
0	r	P(4)	0	k
3	0	B(2)	3	3

B_s = Binary Select

(a)



(b)



(c)

Figure 3.3. Binary Select Function a) Truth Table, b) Block Diagram, c) Schematic

to the designer. First of all, there is the idea of common signal paths to insure a tightly coupled machine. Second, the idea of functional pairing was introduced in Table 2.10. An awkward pairing would be considered an obvious burden. To be unable to bench mark these two B(2):P(4) concepts, there is a need for standard building blocks. However, the motivation for standard circuitry is really greater for another reason. With LSI and VLSI technology where it is today [24, 25, 26] the emphasis is on standardization. Traditionally minimization leads to standardization. Today it is probably more important to standardize even if a design is not minimal than to design a larger number of devices. Since so many silicon devices can be placed on one mm^2 of device area, the number of gates required to implement a given function is of only secondary importance. Of primary importance is reducing the proliferation of expensive art masters and choosing standard functions which will insure greater device usage and in turn lower cost.

A function which meets the above tests and would be a good candidate to bench mark as a B(2):P(4) building block is shown in Figure 3.4. For any two-variable four-valued switching function the "product of sums" realization can be made using the circuitry shown. Outputs from this circuit can be fed into a single MAX/OR gate to achieve the canonical "sum of product of sums" form described in Example 3.1. The binary select function described in Figure 3.3 has been added to enhance the mode control feature of the unary operators. Also, x_3 is an expansion input for the "unary inverter." The $f_2(x_1,x_2)$ output can be connected to the x_3 input of the following stage if more than two implicants exist in the switching function specification in the "product

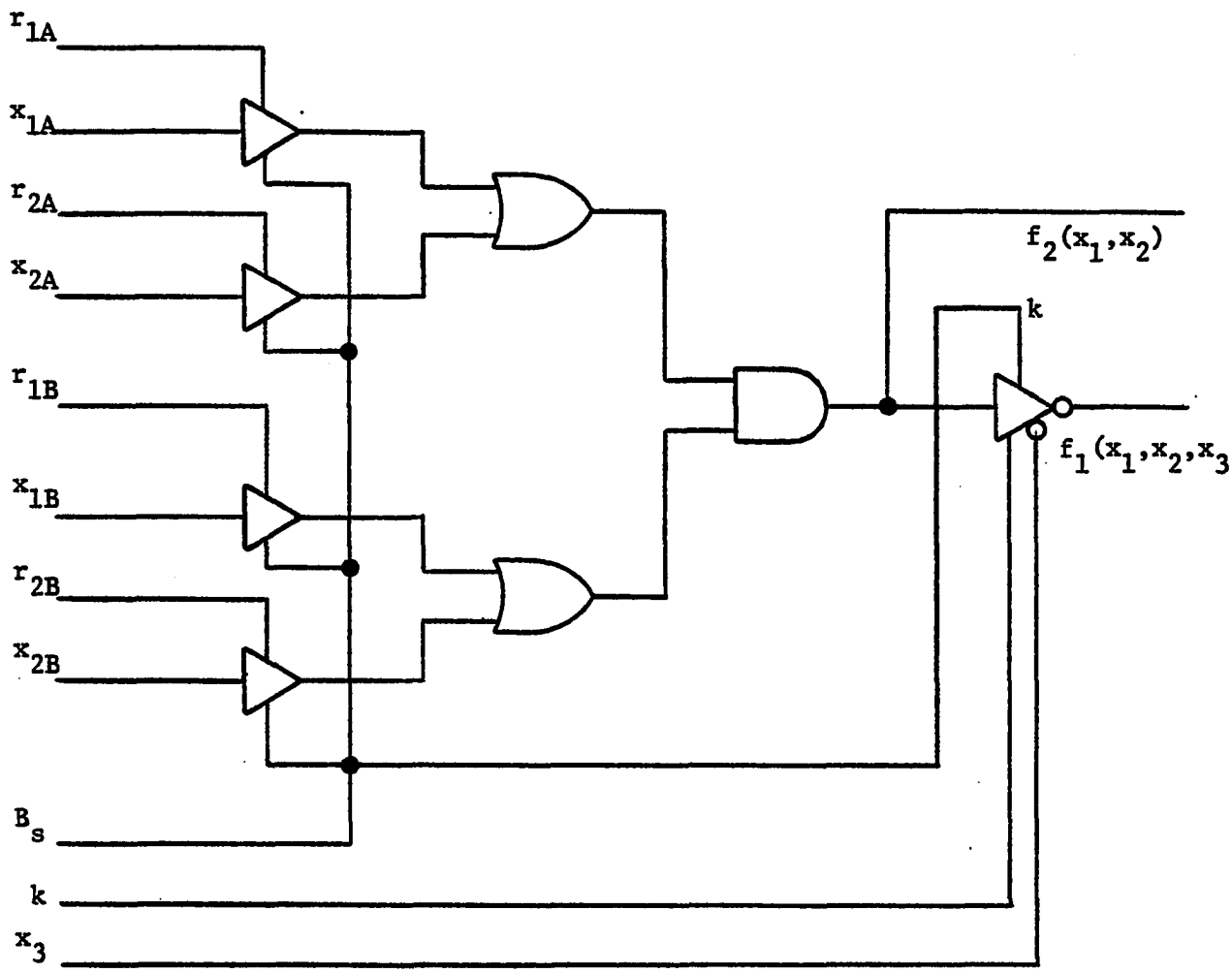


Figure 3.4. A B(2):P(4) "Product of Sums" Gate

of sums" term. If not, the output would be $f_1(x_1,x_2,x_3)$. The two outputs are thus defined as follows:

$$f_1(x_1,x_2,x_3)_4 = [(x_{1A}^{r_{1A}} + x_{2A}^{r_{2A}})(x_{1B}^{r_{1B}} + x_{2B}^{r_{2B}})(x_3)]^k$$

$$f_2(x_1,x_2)_4 = [(x_{1A}^{r_{1A}} + x_{2A}^{r_{2A}})(x_{1B}^{r_{1B}} + x_{2B}^{r_{2B}})]$$

The subscript four will be used to indicate that the expression is for P(4) algebra. Similarly, a subscript of two will be used for the B(2) case. The two expressions for the binary case are:

$$f_1(x_1,x_2,x_3)_2 = \overline{(x_{1A} + x_{2A})(x_{1B} + x_{2B})(x_3)}$$

$$f_2(x_1,x_2)_2 = (x_{1A} + x_{2A})(x_{1B} + x_{2B})$$

Examination of Figure 3.4, and in turn the individual schematics in Figures 2.10, 2.11, 2.14, 2.16, and 3.3, will show that for all functions involved the primary signal path is the same for the B(2) and the P(4) case. Clearly, the only difference in the circuitry is the logic level of the B_s line. In Figure 3.3 $B_s = 0$ for P(4) and $B_s = 3$ for B(2). This choice was made since the circuitry which is currently being used is for the $B(2) \rightarrow P(4)_{0,3}$ mapping. To some extent the logic levels are arbitrary. One of the levels in Figure 3.3 must be a logic zero, but this could be changed. The important question is whether control leads need to be four valued, or can they be binary? Most control signals are considered either enabled or disabled. However, there may be some motivation for four-valued control. This will be considered in Chapter IV.

The functional pairing in Figure 3.4 seems to be well adjusted to B(2):P(4) designs. In P(4) the circuitry is a basic gate for the "sum of products of sums" canonical form of Vranesic's algebra. In B(2) the function is similar to an SN74LS52 [27]. There are many possible configurations for the gates shown in Figure 3.4. The one shown is not necessarily the best pairing for B(2):P(4) combinational design. However, it can be used as a building block for dual radix logic and does meet the bench mark tests defined. As a final comment, the circuit shown in Figure 3.4 contains nine switches and 49 mirrors for a total of 110 collectors. Using the smallest master slice, an XR-200, three of the B(2):P(4) "product of sums" gates could be placed on a single 98 x 119 mil chip.

The second design example will be used to demonstrate a B(2):P(4) combinational design for the algebra of Allen and Givone.

Example 3.2. Given the completely specified function of x_1 and x_2 in Table 3.1, find the P(4) realization from the minimized canonical expression. Show the B(2):P(4) circuit and discuss the a and b mode line considerations.

Any n-variable m-valued switching function $f(x_1,x_2...x_n)$ has a "sum of products" canonical form which is described in [10]. For the function given in Table 3.1 the canonical form is:

$$f(x_1,x_2) = 2 \cdot {}^{0}x_1^{0} \cdot {}^{0}x_2^{0} + 2 \cdot {}^{0}x_1^{0} \cdot {}^{1}x_2^{1} + 2 \cdot {}^{0}x_1^{0} \cdot {}^{2}x_2^{2}$$

$$+3 \cdot {}^{0}x_1^{0} \cdot {}^{3}x_2^{3} + 1 \cdot {}^{1}x_1^{1} \cdot {}^{0}x_2^{0} + 2 \cdot {}^{1}x_1^{1} \cdot {}^{1}x_2^{1}$$

$$+3 \cdot {}^{1}x_1^{1} \cdot {}^{2}x_2^{2} + 1 \cdot {}^{2}x_1^{2} \cdot {}^{0}x_2^{0} + 2 \cdot {}^{3}x_1^{3} \cdot {}^{0}x_2^{0}$$

$$+1 \cdot {}^{3}x_1^{3} \cdot {}^{1}x_2^{1} + 1 \cdot {}^{3}x_1^{3} \cdot {}^{2}x_2^{2}$$

$$f(x_1,x_2) = 1 \cdot ({}^{1}x_1^{1} \cdot {}^{0}x_2^{0} + {}^{2}x_1^{2} \cdot {}^{0}x_2^{0} + {}^{3}x_1^{3} \cdot {}^{1}x_2^{1})$$

$$+2 \cdot ({}^{0}x_1^{0} \cdot {}^{0}x_2^{0} + {}^{0}x_1^{0} \cdot {}^{1}x_2^{1} + {}^{0}x_1^{0} \cdot {}^{2}x_2^{2} + {}^{1}x_1^{1} \cdot {}^{1}x_2^{1}$$

$$+ {}^{3}x_1^{3} \cdot {}^{0}x_2^{0})$$

$$+3 \cdot ({}^{0}x_1^{0} \cdot {}^{3}x_2^{3} + {}^{1}x_1^{1} \cdot {}^{2}x_2^{2})$$

Following the algorithmic minimization process defined in [2], a product term

$$S_1 \Pi_1 = S_1 \cdot {}^{a_1}x_1^{b_1} \cdot {}^{a_2}x_2^{b_2}$$

subsumes a second product term

$$S_2 \Pi_2 = S_2 \cdot {}^{c_1}x_1^{d_1} \cdot {}^{c_2}x_2^{d_2}$$

if and only if both

$$S_1 \leq S_2 \text{ and } c_i \leq a_i \leq b_i \leq d_i \text{ for all } x_i,\ i = 1, 2, \ldots, n$$

A product term is said to be an implicant of f if and only if $S_1\ \Pi(x) > 0$ for some $x \in S^N$ and $S_1\ \Pi(x) \leq f(x)$ for all $x \in S^N$. Also, an implicant is said to be prime if it subsumes no other implicant of f. Thus, the above function simplifies to:

$$f(x_1,x_2) = 1 \cdot ({}^{0}_{x_1}{}^{3} \cdot {}^{0}_{x_2}{}^{0} + {}^{3}_{x_1}{}^{3} \cdot {}^{1}_{x_2}{}^{2})$$

$$+2 \cdot ({}^{0}_{x_1}{}^{0} \cdot {}^{0}_{x_2}{}^{3} + {}^{0}_{x_1}{}^{1} \cdot {}^{1}_{x_2}{}^{2} + {}^{3}_{x_1}{}^{3} \cdot {}^{0}_{x_2}{}^{0})$$

$$+3 \cdot ({}^{0}_{x_1}{}^{0} \cdot {}^{3}_{x_2}{}^{3} + {}^{1}_{x_1}{}^{1} \cdot {}^{2}_{x_2}{}^{2})$$

Since ${}^{0}_{x}{}^{m-1} = m-1$, the function can be further simplified to:

$$f(x_1,x_2) = 1 \cdot ({}^{0}_{x_2}{}^{0} + {}^{3}_{x_1}{}^{3} \cdot {}^{1}_{x_2}{}^{2})$$

$$+2 \cdot ({}^{0}_{x_1}{}^{0} + {}^{0}_{x_1}{}^{1} \cdot {}^{1}_{x_2}{}^{2} + {}^{3}_{x_1}{}^{3} \cdot {}^{0}_{x_2}{}^{0})$$

$$+3 \cdot ({}^{0}_{x_1}{}^{0} \cdot {}^{3}_{x_2}{}^{3} + {}^{1}_{x_1}{}^{1} \cdot {}^{2}_{x_2}{}^{2})$$

The P(4) realization of the above function is shown in Figure 3.5.

There are two possible ways to utilize the literal function in B(2) for the $B(2) \rightarrow P(4)_{0,3}$ mapping. If a,b = {0,0}, the function becomes an inverter. If a,b = {3,3}, the function becomes a non-inverting

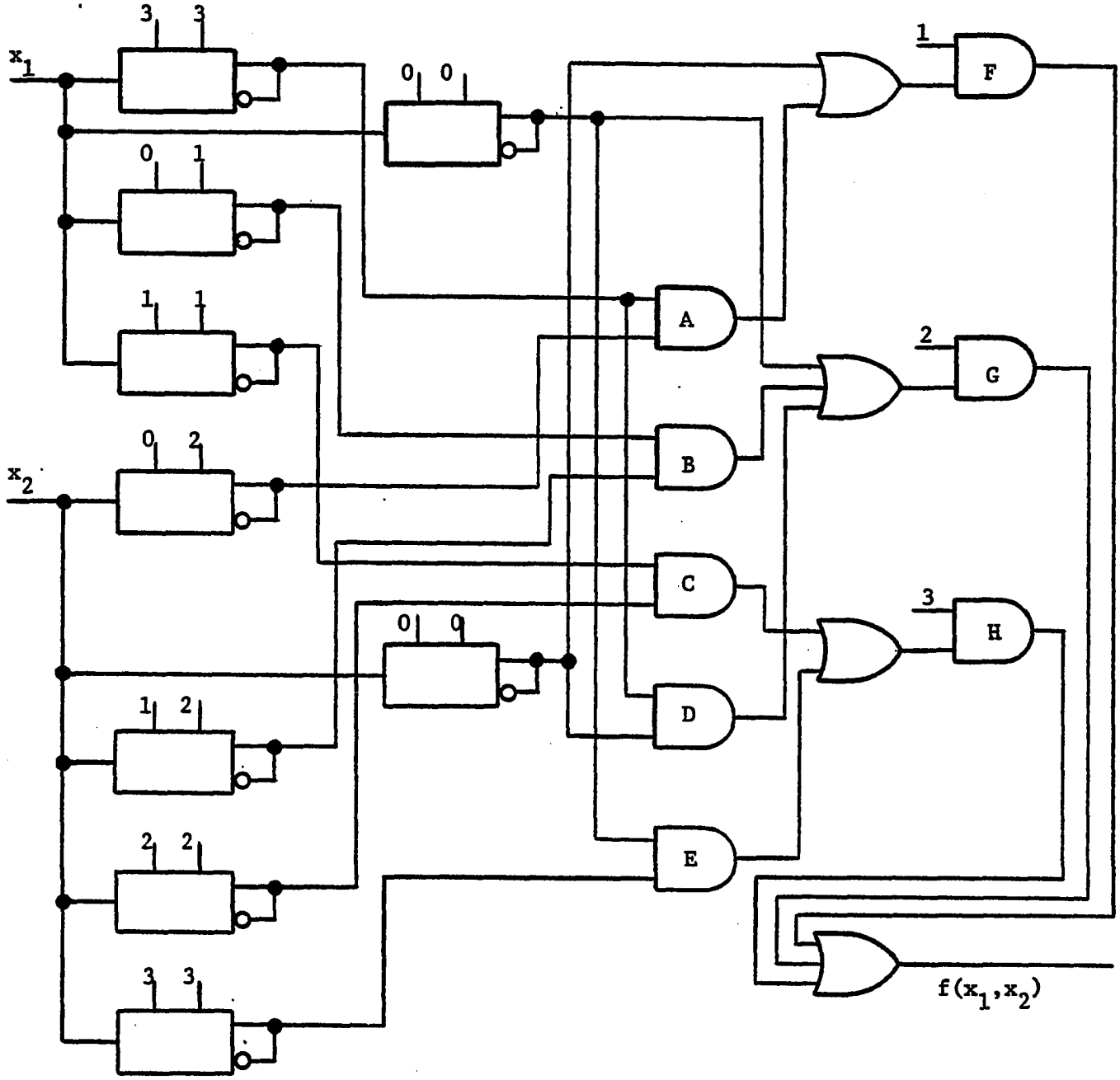
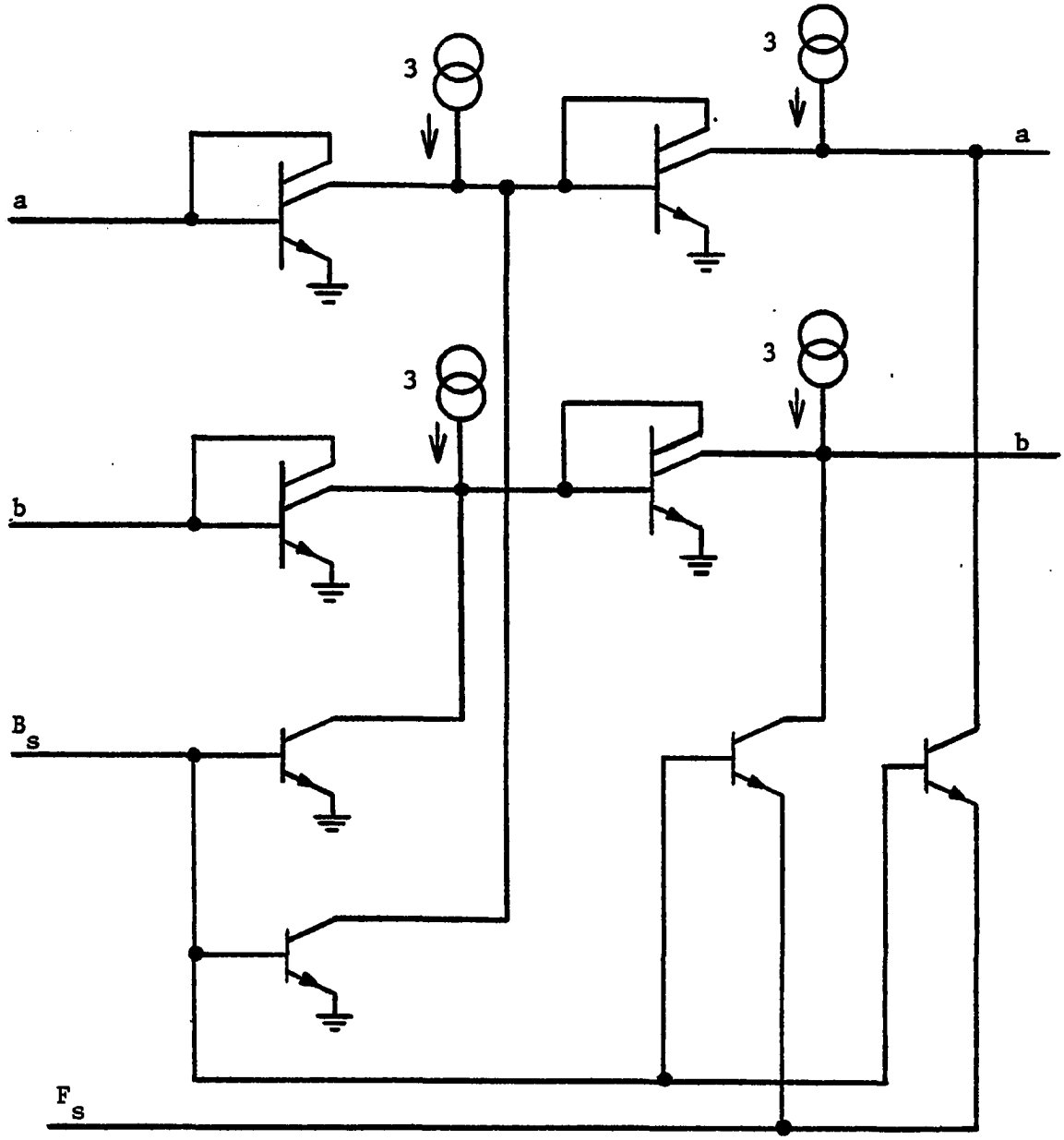


Figure 3.5. P(4) Realization of Example 3.2.

buffer. Where the mode control lines were allowed to default to a constant value in the algebra of Vranesic, et al, the mode control lines could have the flexibility of two default values, and thus two binary functions, for this case. Changing the function from an inverter to a buffer would probably be a manual strapping option as opposed to the B_s line for which software control has been proposed. The circuit shown in Figure 3.6(a) would allow the mode lines to be controlled as suggested. The function select line, F_s, controls the default values for a and b. The truth table for the literal B_s and F_s control lines is shown in Figure 3.6(b).

Before considering a B(2):P(4) realization for the literal function, some observations can be made concerning Figure 3.5(a). Each of the MIN gates, A-E, are a redundant realization of $x_1 \cdot x_2$ or $\overline{x}_1 \cdot \overline{x}_2$ depending on the logical value of the F_s line. Also, the MIN gates F, G, and H have constant logical inputs which are awkward to handle for the $B(2) \rightarrow P(4)_{0,3}$ mapping. With this in mind, it would be appropriate for comparison purposes to construct a standard configuration for the Allen and Givone algebra which would be roughly equivalent to B(2):P(4) "product of sums" gate shown in Figure 3.4 for the Vranesic algebra.

Figure 3.7 is the B(2):P(4) "sum of products" gate for the Allen and Givone algebra. The MIN gates F, G, and H in Figure 3.5 were not included in this configuration due to the awkward nature of the constant logical values. For any two-variable four-valued switching function the "sum of products" realization can be made. Outputs from this circuit must be fed into a MIN/AND and MAX/OR gate to achieve the actual canonical form. The B_s and F_s control functions shown in



(a)

B_s	F_s	a	b
0	0	a	b
0	3	a	b
3	0	0	0
3	3	3	3

(b)

Figure 3.6. B_s and F_s for the Literal Function

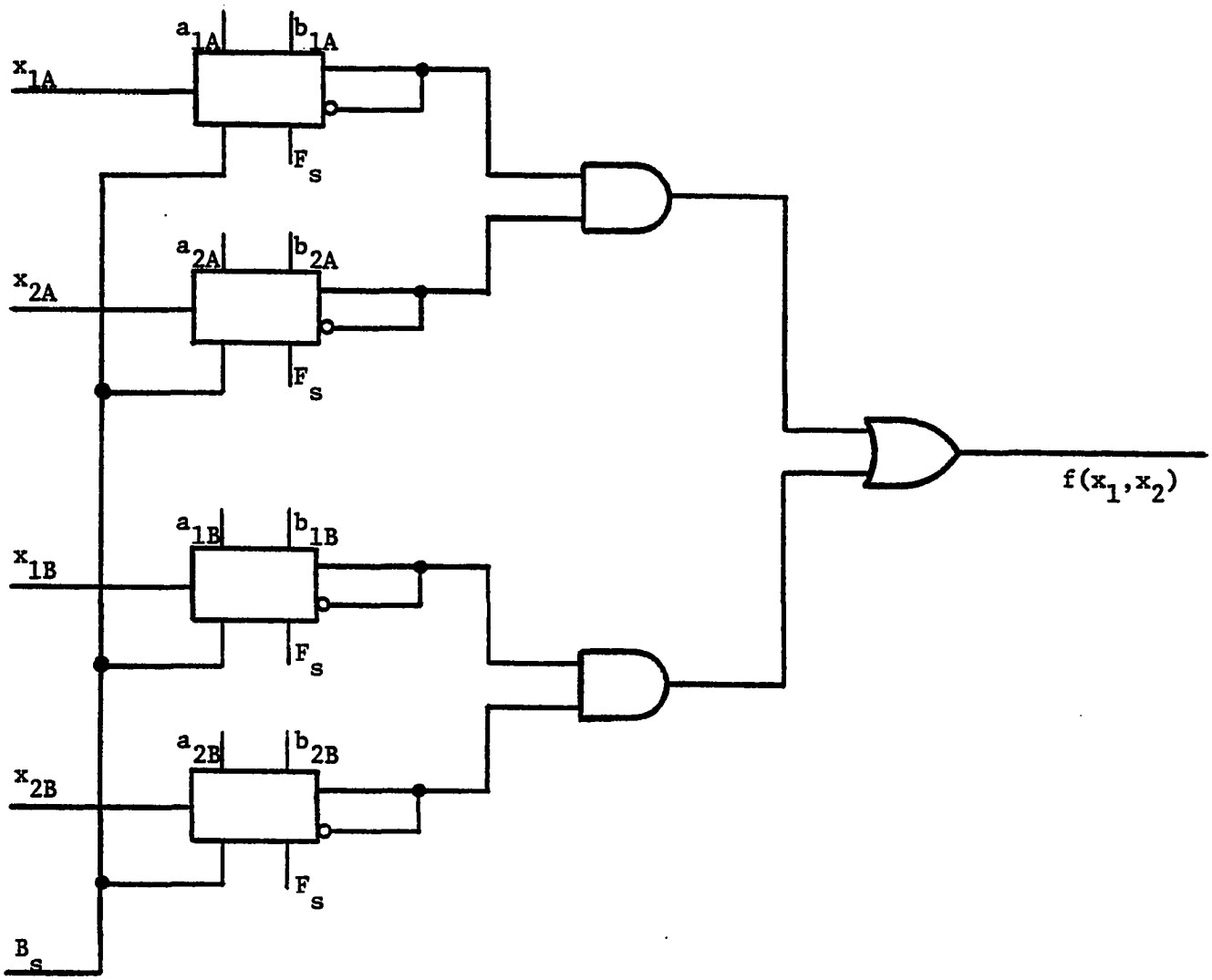


Figure 3.7. A B(2):P(4) "Sum of Products" Gate

Figure 3.6 must be added to allow proper B(2):P(4) operation. The three outputs of the circuit are:

$$f(x_1,x_2)_4 = \overset{a_{1A}\ b_{1A}}{x_{1A}} \cdot \overset{a_{2A}\ b_{2A}}{x_{2A}} + \overset{a_{1B}\ b_{1B}}{x_{1B}} \cdot \overset{a_{2B}\ b_{2B}}{x_{2B}}$$

$$f(x_1,x_2)_2 = x_{1A} \cdot x_{2A} + x_{2A} \cdot x_{2B} \text{ for } F_s = 3$$

$$f(x_1,x_2)_2 = \overline{x}_{1A} \cdot \overline{x}_{2A} + \overline{x}_{2A} \cdot \overline{x}_{2B} \text{ for } F_s = 0$$

Examination of Figure 3.7 and, in turn, the individual schematics in Figures 2.10, 2.11, and 2.17 will show that for all functions involved the primary signal path is the same for both the B(2) and P(4) case. The additional hardware burden for the binary select and function select lines is shown in Figure 3.6(a). Four mirrors, four switches, and twelve collectors are required. This is compared to five mirrors, two switches, and thirteen collectors for the literal operator shown in Figure 2.17. The hardware overhead for the B(2):P(4) realization is nearly 100%.

The functional pairing in Figure 3.7 is well adjusted to B(2):P(4) designs. In P(4) the circuitry is a basic gate for the "sum of products" canonical form for the Allen and Givone algebra. In B(2) the function is similar to a SN74LS54 [27]. It should be noted that MIN gates F, G, and H in Figure 3.5 were omitted from the standard building block of Figure 3.7. If the constant logic values are not handled in a manner similar to the r, k, a, and b lines, then B(2) signals cannot be routed through these gates. The constant lines can be treated as variables in B(2) with the addition of two mirrors and one switch. This would allow the goal of common signal paths. Since this is not required for the

circuitry in Figure 3.7, the schematic is not shown here. The circuit shown in Figure 3.7 requires 24 switches and 46 mirrors for a total of 123 collectors. Using the smallest master slice, an XR-200, two of the B(2):P(4) "sum of products" gates could be placed on one chip.

Before going to the next topic it would seem to be in order to make a comparison of the kinds of ideas and developments that have been made thus far in Chapter III. The notion is not to discard anything at this time, but rather to get a better idea of just where we might be. Table 3.2 is a summary of the bench marks which were defined from the outset. It is not an exhaustive comparison; however, it does concisely take a snapshot of just where the work has progressed to for the B(2):P(4) realizations in both algebras.

Other Operators

The emphasis thus far has been on two algebras and their simultaneous realization for both binary and quaternary logic. The motivation has been to move from the algebras to the realizations. Work has been done by beginning with the practical implementation and moving to the algebra [18]. The circuits developed in Chapters II and III used current mirrors and switches. Both of these functions are easy to construct with I^2L. However, two additional circuit operations are readily available when using I^2L. They are linear summation and thresholding. If design of combinational circuits proceeds with the standard circuits of Figure 3.4 and Figure 3.7, many realizations would become extremely awkward. While it is true that no single standard set of basic gates will be best for every system, it is also true that no one design technique will show which functions to use.

Table 3.2. Summary of B(2):P(4) Bench Marks for Both Algebras

Bench Mark	Vranesic Et Al	Allen and Givone
1. B(2):P(4) Hardware	110 Collectors 49 Mirrors 9 Switches	123 Collectors 46 Mirrors 24 Switches
2. P(4) Hardware Only	85 Collectors 39 Mirrors 4 Switches	75 Collectors 30 Mirrors 8 Switches
3. B(2) Burden	25 Collectors - 23% 10 Mirrors - 20% 5 Switches - 56%	48 Collectors - 39% 16 Mirrors - 35% 16 Switches - 67%
4. Common Signal Paths	100%	100%
5. Functional Pairing P(4) B(2)	f_4 = "Product of Sums" f_{12}^4 = $\overline{\text{OR-AND}}$ f_{22} = OR-AND	f_4 = "Sum of Products" f_{12}^4 = AND-OR , $F_s = 3$ f_{22} = NOT-AND-OR , $F_s = 0$
6. Standard Circuit	Figure 3.4	Figure 3.7
7. Mode Control	B_s = B(2) Select $B_s = 0$; P(4) $B_s = 3$; B(2)	B_s = B(2) Select $B_s = 0$; P(4) $B_s = 3$; B(2) F_s = Function Select $F_s = 0$; a = b = 0 $F_s = 3$; a = b = 3

Linear summation can be accomplished using the current mirror. If the desired signal is nx_1, where n = 0,1...m-1, then n collectors can be tied together. The realization of $x_1 \oplus 2x_2 \oplus 3x_3$ is shown in Figure 3.8. The current sinked by the output collectors of the mirrors can easily be tied together.

Thresholding in its simplest form is a switch, or transistor which is either saturated or unsaturated. Many unary operations can be implemented with threshold circuitry. A Post algebra monotone unary operator can be described as [20]:

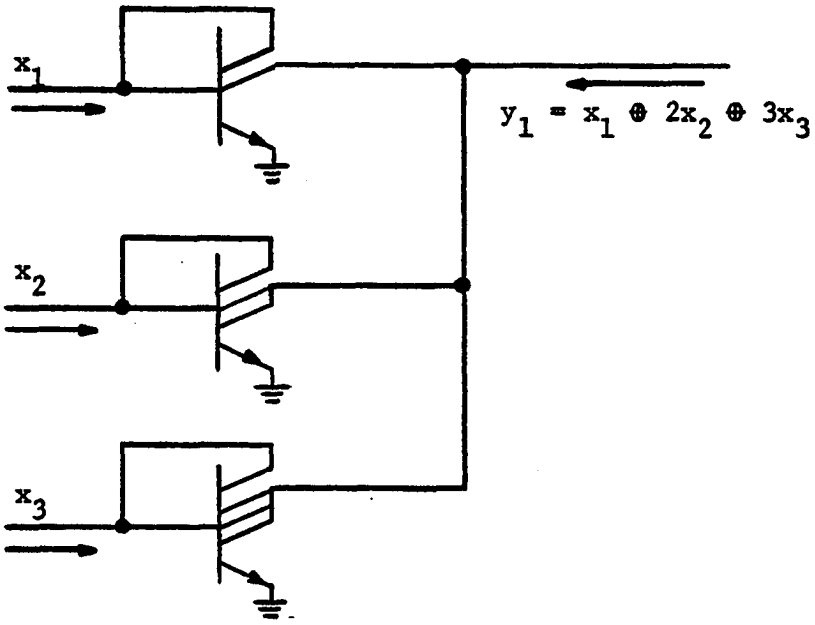


Figure 3.8. I^2L Realization of $x_1 \oplus 2x_2 \oplus 3x_3$

$$D_i(x) \ , \ i = 1,2,\ldots m-1$$

$$D_i(x) = \begin{cases} m-1 & \text{if } x \geq i \\ 0 & \text{if } x < i \end{cases}$$

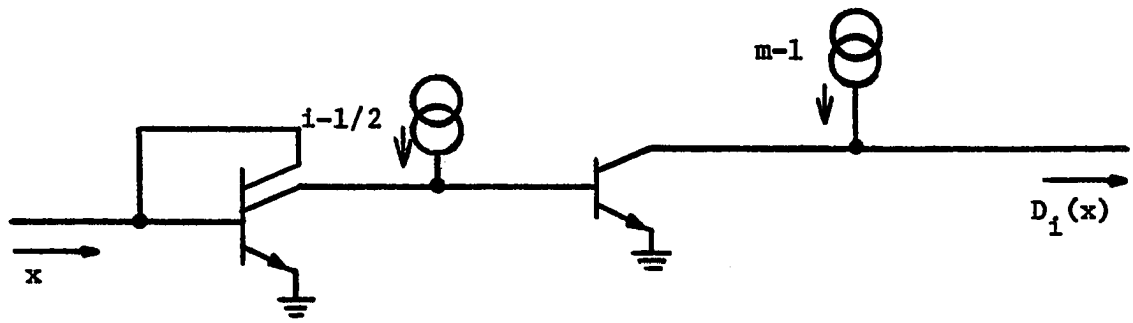
$$\overline{D}_i(x) = \begin{cases} 0 & \text{if } x \geq i \\ m-1 & \text{if } x < i \end{cases}$$

Similarly a Post algebra disjoint unary operator can be described as [20]:

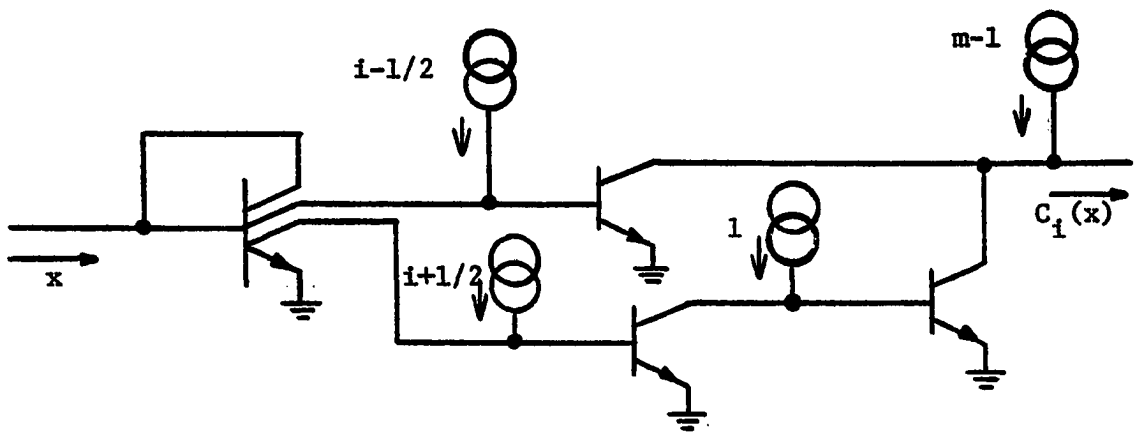
$$C_i(x) \quad , \; i = 0,1,\ldots m-1$$

$$C_i(x) = \begin{cases} m-1 & \text{if } x = i \\ 0 & \text{if } x \neq i \end{cases}$$

Figure 3.9 shows an implementation of the $D_i(x)$ and $C_i(x)$ threshold operators.



(a)



(b)

Figure 3.9. a) I^2L Realization of $D_i(x)$
b) I^2L Realization of $C_i(x)$

These circuits from Pugsley and Silio [20], are similar to the strong threshold literals, $U_t(x)$ and $D_t(x)$, and the strong delta literals, x^I, described by Mc Cluskey [18]. The truth tables in P(4) for $D_i(x)$ and $C_i(x)$ show that the $B(2) \rightarrow P(4)_{0,3}$ mapping exists for both of these threshold operators.

Table 3.3. $D_i(x)$ and $C_i(x)$ in P(4)

x \ i	0	1	2	3
0	3	0	0	0
1	3	3	0	0
2	3	3	3	0
3	3	3	3	3

$D_i(x)$

a)

x \ i	0	1	2	3
0	3	0	0	0
1	0	3	0	0
2	0	0	3	0
3	0	0	0	3

$C_i(x)$

b)

For $D_i(x)$ with i = 3, the B(2) function is a non-inverting buffer. For $C_i(x)$ with i = 0, the B(2) function is an inverting buffer and with i = 3 the function is a non-inverting buffer. The design of B(2):P(4) circuitry for the monotone and disjoint unary operators would proceed in the same manner as the "clockwise cycling" operator. The current source, i, must have a default value for the B(2) case. The circuitry for $D_i(x)$ and $C_i(x)$ in B(2):P(4) is shown in Figure 3.10. The binary select line equals three for the B(2) case just as in the other circuits. The hardware burden is a single switch for $D_i(x)$ and two switches for $C_i(x)$.

Some well known equivalence relationships exist between various unary operators. The Post algebra identity is:

$$C_i(x) = D_i(x) \cdot \overline{D}_{i \oplus 1}(x)$$

Also, relationships between Definition 1.5 and the Post monotone unary operators are:

$$a_x{}^b = D_a(x) \cdot \overline{D}_{b \oplus 1}(x)$$

$$a_x{}^a = C_a(x)$$

$$a_x{}^{m-1} = D_a(x)$$

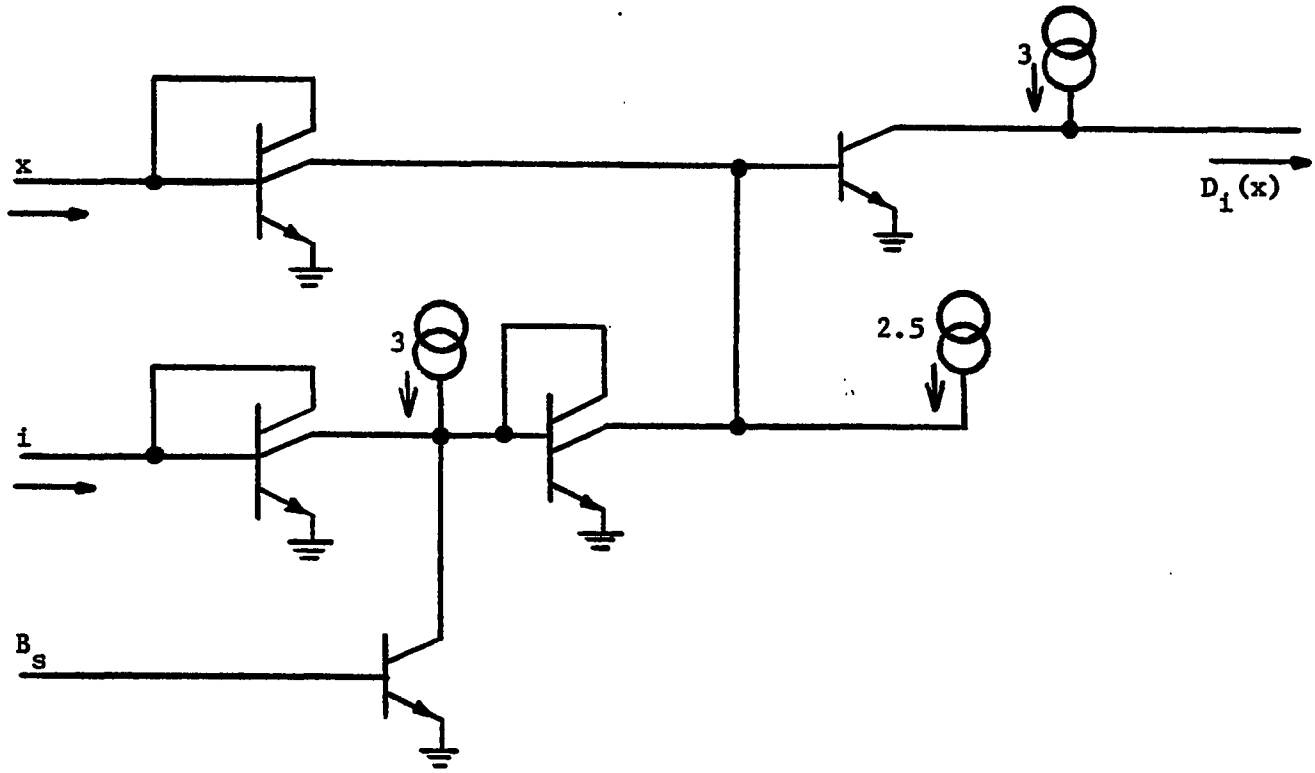
$$0_x{}^{a-1} = \overline{D}_a(x)$$

Relationships between Definition 1.4 and 1.5 are as follows:

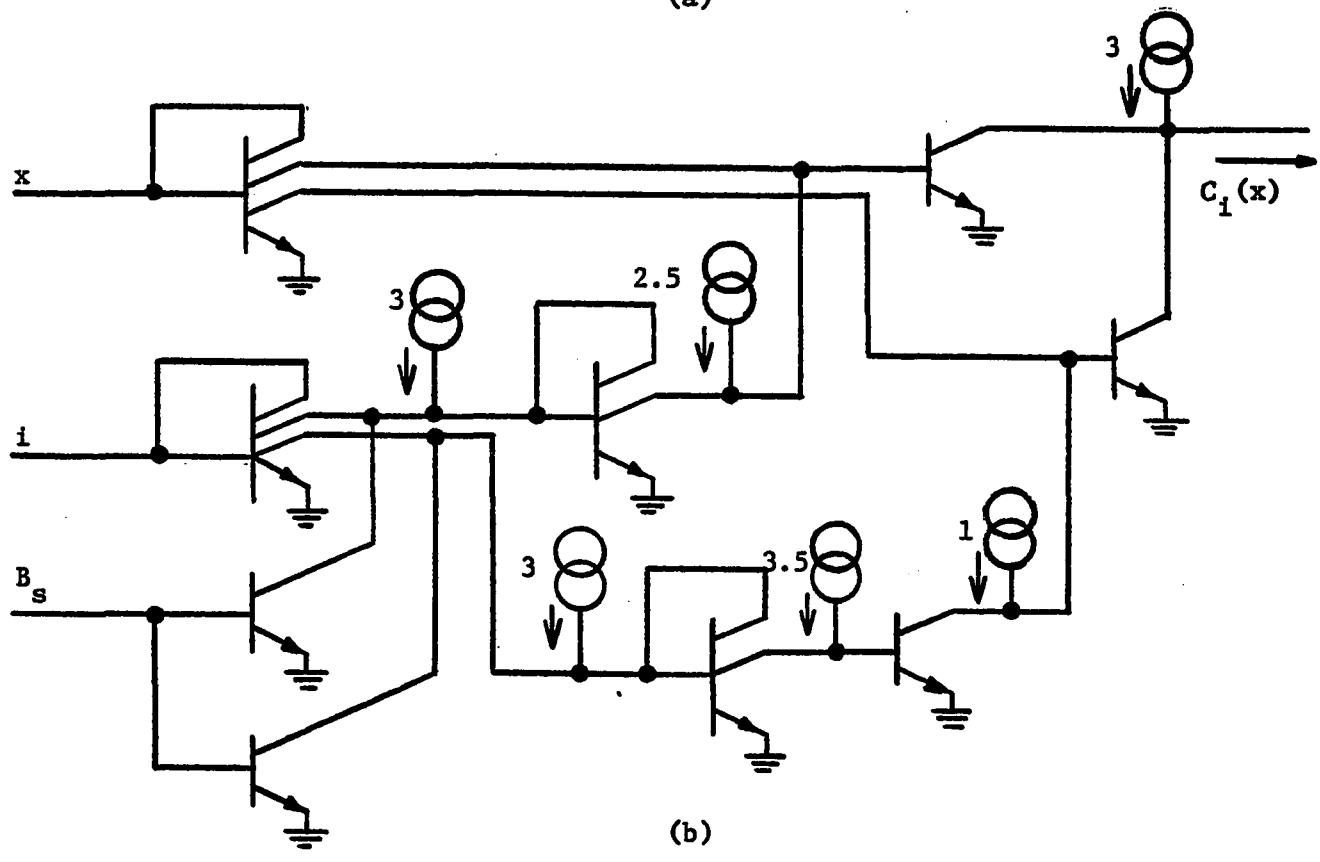
$$(x^{\overrightarrow{R}})^k = k \cdot x^{m-R}{}^{m-R}$$

$$m-R_x{}^{m-R} = (x^{\overrightarrow{R}})^{m-1}$$

It has already been mentioned that designs of multiple-valued circuits will take a particular direction depending upon the algebra chosen. Also, no uniform design technique exists that will guarantee that a set of gates will provide the best realization. All of this is further complicated by the B(2):P(4) requirement. Further, while the $B(2) \rightarrow P(4)_{0,3}$ mapping offers the best functional pairing for circuits examined thus far, the $B(2) \rightarrow P(4)_{1,2}$ mapping may provide a better system for some realizations. The following B(2):P(4) design example for a full adder will demonstrate many of the dual radix concepts developed in Chapters II and III.



(a)



(b)

Figure 3.10. (a) $D_i(x)$ and (b) $C_i(x)$ for B(2):P(4)

Example 3.3. Design a B(2):P(4) full adder, giving consideration to both the $B(2) \rightarrow P(4)_{0,3}$ and $B(2) \rightarrow P(4)_{1,2}$ mappings. Determine which unary operator gives the best realization.

The truth tables for the sum and carry out for a P(4) full adder are shown in Table 3.4. Carry in and carry out lines need to take on two values only.

Table 3.4. Sum and Carry Out in P(4)

x_1 \ x_2	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

$C_{in} = 0$

x_1 \ x_2	0	1	2	3
0	1	2	3	0
1	2	3	0	1
2	3	0	1	2
3	0	1	2	3

$C_{in} = 1$

a) Sum

x_1 \ x_2	0	1	2	3
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	1	1	1

$C_{in} = 0$

x_1 \ x_2	0	1	2	3
0	0	0	0	1
1	0	0	1	1
2	0	1	1	1
3	1	1	1	1

$C_{in} = 1$

b) Carry Out

The truth tables for the sum and carry out for a B(2) full adder for $B(2) \rightarrow P(4)_{0,3}$ and $B(2) \rightarrow P(4)_{1,2}$ are shown in Table 3.5. First, an attempt should be made to find the mapping from B(2) to P(4) which offers

Table 3.5. Sum and Carry Out in B(2)

x_1 \ x_2	0	3
0	0	3
3	3	0

$C_{in} = 0$

x_1 \ x_2	0	3
0	3	0
3	0	3

$C_{in} = 3$

$B(2) \rightarrow P(4)_{0,3}$

x_1 \ x_2	1	2
1	1	2
2	2	1

$C_{in} = 1$

x_1 \ x_2	1	2
1	2	1
2	1	2

$C_{in} = 2$

$B(2) \rightarrow P(4)_{1,2}$

a) Sum

x_1 \ x_2	0	3
0	0	0
3	0	3

$C_{in} = 0$

x_1 \ x_2	0	3
0	0	3
3	3	3

$C_{in} = 3$

$B(2) \rightarrow P(4)_{0,3}$

x_1 \ x_2	1	2
1	1	1
2	1	2

$C_{in} = 1$

x_1 \ x_2	1	2
1	1	2
2	2	2

$C_{in} = 2$

$B(2) \rightarrow P(4)_{1,2}$

b) Carry Out

the greatest compatibility. For the sum, the greatest agreement occurs for the $B(2) \rightarrow P(4)_{0,3}$ mapping. For the carry out, neither mapping is compatible. If the carry out in P(4) is assigned the logical value of three instead of one, then the carry in P(4) is compatible with B(2) for $B(2) \rightarrow P(4)_{0,3}$. For these choices the new truth tables in Table 3.6 show the degree of compatibility for a B(2):P(4) realization. The two values enclosed in a square are not compatible in both radices. The two

in P(4) would be a zero in B(2). The one in P(4) would be a three in B(2). In any realization it will be necessary for the B_s line to force the value changes in B(2). For $C_{in} = 0$ the truth table for the sum in P(4) is identical to Table 2.8. Thus, strong consideration should be given to the "clockwise cycling" unary operator. In Figure 2.16, the

Table 3.6. Sum and Carry Out in B(2):P(4) for $B(2) \rightarrow P(4)_{0,3}$, $C_{out} = 0,3$

x_1 \ x_2	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

$C_{in} = 0$

x_1 \ x_2	0	1	2	3
0	1	2	3	0
1	2	3	0	1
2	3	0	1	2
3	0	1	2	3

$C_{in} = 3$

a) Sum

x_1 \ x_2	0	1	2	3
0	0	0	0	0
1	0	0	0	3
2	0	0	3	3
3	0	3	3	3

$C_{in} = 0$

x_1 \ x_2	0	1	2	3
0	0	0	0	3
1	0	0	3	3
2	0	3	3	3
3	3	3	3	3

$C_{in} = 3$

b) Carry Out

design was done in such a manner that r could be thought of as a variable. Therefore, Figure 2.16 can be thought of as a full adder for two inputs x and r. If provisions were made for a carry in and carry out, then a

P(4) realization for Table 3.6 would exist. Figure 3.11 is the realization of a full adder in P(4) which was designed by beginning with Figure 2.16. This circuit has 22 collectors compared to 18 for a quarternary adder in [5].

In order for the circuit in Figure 3.11 to operate as a B(2) full adder for B(2) → $P(4)_{0,3}$ it is necessary for the two previously mentioned changes in Table 3.6 to be implemented when $B_s = 3$. Table 3.7 gives the conditions precisely. The circuitry in Figure 3.12 will implement Table 3.7. If the sum outputs of Figures 3.11 and 3.12 are connected the combined circuit is a dual radix B(2):P(4) full adder. Can another unary operator be selected to achieve a better realization? What would the circuit look like if it had not been possible "to see" the similarity between Tables 3.6 and 2.8.

Table 3.7. Additional Requirements for B(2)

x_1	x_2	C_{in}	B_s	Sum
0	0	3	0	1
3	3	0	0	2
0	0	3	3	3
3	3	0	3	0

(Otherwise Table 3.6.)

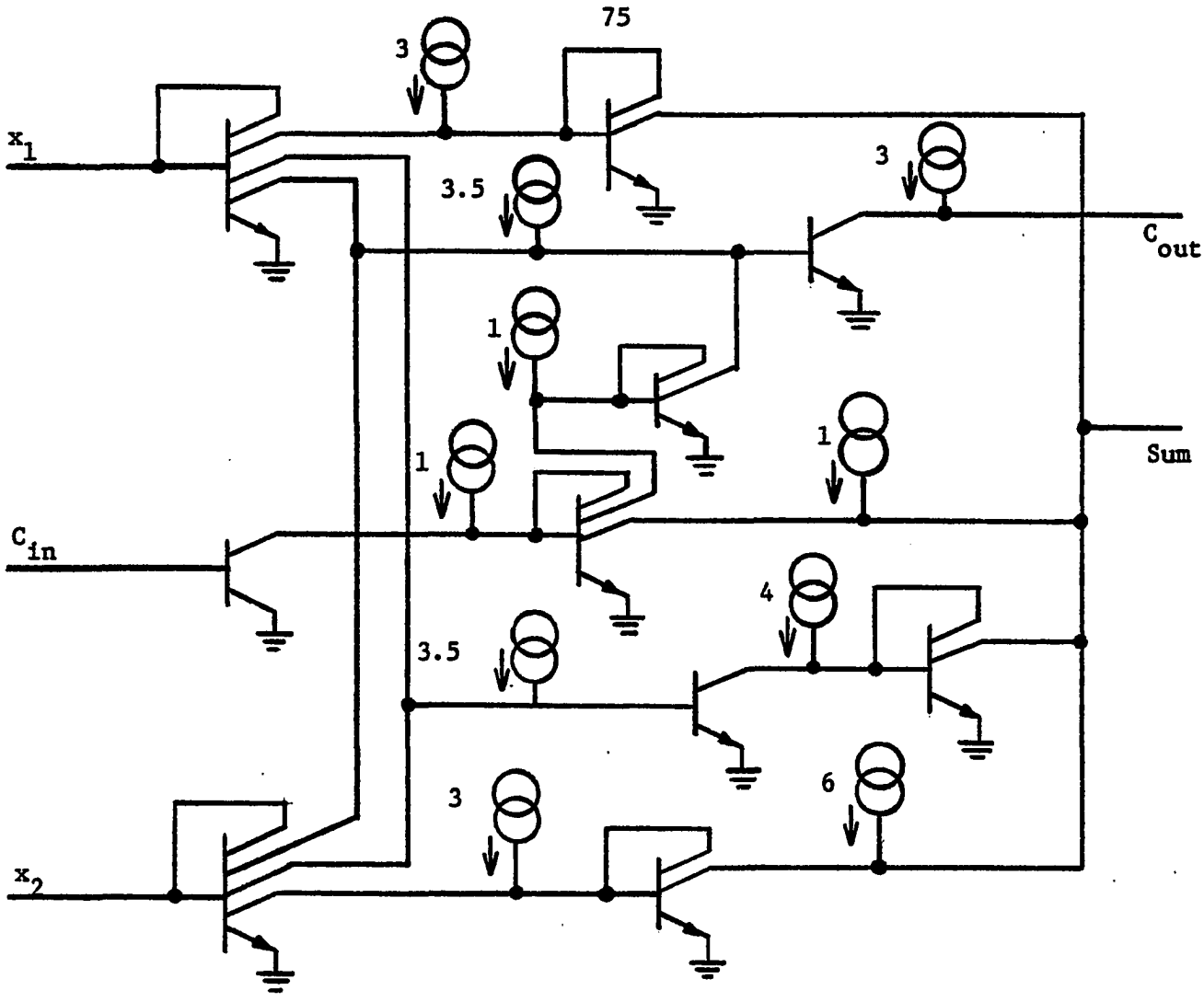


Figure 3.11. P(4) Full Adder , C_{out} = 0,3

While the circuits of Figures 3.4 and 3.7 offer standard B(2): P(4) circuitry for two algebras, a systematic approach is needed to determine what basic set of gates offers the best realization. No such approach exists now for P(m) algebras [20]. However, many minimization schemes have been reviewed by Smith [28, 29, 30] which provide a minimal or near minimal solution for P(m) algebras. Any method for finding the best B(2):P(4) realization must begin by finding the maximum compatible mapping from B(2) to P(4) and identifying any incompatibilities. The two classes of problems which will be considered are completely and

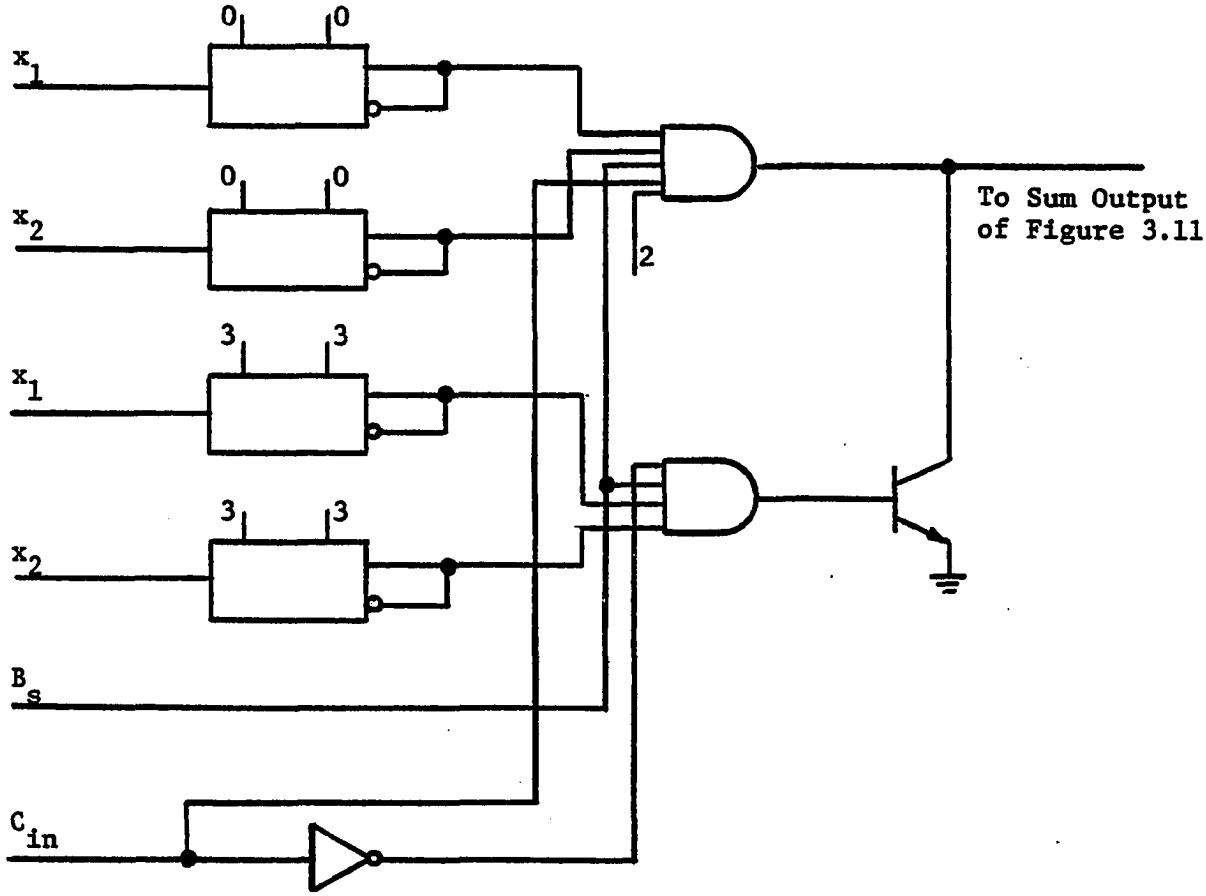


Figure 3.12. Requirements for B(2):P(4)

incompletely specified functions. A procedure for finding the maximum compatible mapping and all incompatibilities for completely specified functions in P(4) and B(2) can proceed as follows:

Given: An n variable four-valued function in P(4) and an n variable two-valued function in B(2) such that

$$x_{14}, x_{24}, \ldots x_{n4} \in P(4)$$

$$x_{12}, x_{22}, \ldots x_{n2} \in B(2)$$

$$f_4 = f(x_{14}, x_{24}, \ldots x_{n4})$$

$$f_2 = f(x_{12}, x_{22}, \ldots x_{n2})$$

From Table 2.3, there are two possible mappings for m = 4,

$$B(2) \rightarrow P(4)_{i,j} \qquad i = \bar{j} \text{ and } i,j = \begin{cases} 0,3 \\ 1,2 \end{cases}$$

Find all mapping compatibilities:

Step 1. Set $i = 0$, $j = 3$

Step 2. $f_2(i,j) = f_4(i,j)$ set $\delta_{i,j} = f_2(i,j) = f_4(i,j)$

or $\delta_{i,j}$ = B(2):P(4) implicant

$f_2(i,j) \neq f_4(i,j)$ set $\alpha_{i,j} = f_2(i,j)$

or $\alpha_{i,j}$ = B(2) term

$\beta_{i,j} = f_4(i,j)$

$\beta_{i,j}$ = P(4) term

Step 3. Find the remaining implicants for

ii, jj, and ji

Step 4. If $\Sigma\delta_{ij} = 4$, then $B(2) \rightarrow P(4)_{0,3}$ exists for f_4

If $\Sigma\delta_{ij} < 4$, then save all terms and repeat process for $i = 1$, $j = 2$.

Notice that this procedure assumes the number of variables in each radix is the same. This is not restrictive since every effort has been made thus far to insure common signal paths. That is, every signal path in P(4) is a signal path in B(2). Also, this procedure can easily be generalized for P(m).

Given: An n variable m-valued function in P(m) and an n variable two-valued function in B(2) such that

$$x_{1m}, x_{2m}, \ldots x_{nm} \in P(m)$$

$$x_{12}, x_{22}, \ldots x_{n2} \in B(2)$$

For $m = 2^N$ N an integer greater than one

$$B(2) \rightarrow P(m)_{i,j} \qquad i = \bar{j} \text{ and } i,j = \begin{cases} i = 0 \ , \ j = m-1 \\ i \oplus 1 \ , \ j - 1 \\ \\ \\ i \oplus 2^{N-1}-1 \ , \ j - 2^{N-1}-1 \end{cases}$$

Find all mapping compatibilities:

Step 1. Set $i = 0$, $j = m-1$

Step 2. $f_2(i,j) = f_m(i,j)$ set $\delta_{i,j} = f_2(i,j) = f_m(i,j)$
or $\delta_{i,j} = B(2):P(m)$ implicant

$f_2(i,j) \neq f_m(i,j)$ set $\alpha_{i,j} = f_2(i,j)$
or $\alpha_{i,j} = B(2)$ term
set $\beta_{i,j} = f_m(i,j)$
or $\beta_{i,j} = P(m)$ term

Step 3. Find the remaining implicants:
Repeat Steps 1 and 2 2^n-1 times for each mapping.

Step 4. If $\Sigma\delta_{i,j} = 2^n$, then $B(2) \rightarrow P(m)_{i,j}$ exists
If $\Sigma\delta_{i,j} < 2^n$, then save all terms and repeat the process $2^{N-1}-1$ times to check all mappings. ($i = 1$, $j = m-2$, etc.)

This procedure can be programmed easily to provide a systematic and automated method for finding mappings. For $n > 2$, it becomes difficult to identify compatibilities even in P(4). For $n > 2$ in P(m) algebras with $N > 2$, the task becomes extremely difficult. From the above procedure the following observations can be made:

If $\delta_{i,j} = 2^n$, then $B(2) \rightarrow P(m)_{i,j}$ exists for f_m

$\delta_{i,j} < 2^n$, then $B(2) \rightarrow P(4)_{i,j}$ exists if and only if there is a function $F_m^*(i,j)$ such that

$$f_2(i,j) = f_m(i,j) \dot{+} F_m^*(i,j)$$

for every

$$f_2(i,j) \neq f_m(i,j)$$

The form which $F_m^*(i,j)$ takes depends upon which algebra or basic set of gates has been chosen to represent f_2 and f_4. All of the information necessary to determine $F_m^*(i,j)$ is contained in $\alpha_{i,j}$ and $\beta_{i,j}$. For $\Sigma\delta = 3$, a single set of α and β terms exist and

$$F_m^*(i,j) = S_{ij} \cdot {}^{i}x_{1m}^{i}\ {}^{j}x_{2m}^{j} \quad \text{where } \alpha_{i,j} = S_{ij}$$

For $\Sigma\delta = 0$, 2^n sets of α and β terms exist and for $n = 2$,

$$F_m^*(i,j) = S_{ij} \cdot {}^{i}x_{1m}^{i}\ {}^{j}x_{2m}^{j} + S_{ii} \cdot {}^{i}x_{1m}^{i}\ {}^{i}x_{2m}^{i} + S_{ji}\ {}^{j}x_{1m}^{j}\ {}^{i}x_{2m}^{i} + S_{jj} \cdot {}^{j}x_{1m}^{j}\ {}^{j}x_{2m}^{j}$$

Clearly, $F_m^*(i,j)$ can be expanded for $n > 2$. Also, there exists a unique $F_m^*(i,j)$ such that

$$f_2(i,j) = f_m(i,j) \dot{+} F_m^*(i,j)$$

for every

$$f_2(i,j) \neq f_m(i,j)$$

The expansion for $F_m^*(i,j)$ is in the canonical form for the algebra of Allen and Givone. Using the equivalence relationships established earlier

in this chapter, $F_m^*(i,j)$ can be expressed in the canonical form of the algebra of Vranesic, et al. For $\Sigma\delta = 3$,

$$F_m^*(i,j) = \vec{x}_{1m}^{\,m-i} \cdot \vec{x}_{2m}^{\,m-j}\; k_{ij} \qquad \text{where } \alpha_{i,j} = k_{i,j}$$

For $\Sigma\delta = 0$, 2^n sets of α and β terms exist and for $n = 2$,

$$F_m^*(i,j) = \vec{x}_{1m}^{\,m-i} \cdot \vec{x}_{2m}^{\,m-j}\; k_{ij} + \vec{x}_{1m}^{\,m-i} \cdot \vec{x}_{2m}^{\,m-i}\; k_{ii}$$

$$+ \vec{x}_{1m}^{\,m-j} \cdot \vec{x}_{2m}^{\,m-i}\; k_{ji} + \vec{x}_{1m}^{\,m-j} \cdot \vec{x}_{2m}^{\,m-j}\; k_{jj}$$

The full adder described in Table 3.6 of Example 3.3 will be used to demonstrate the procedure just defined.

Given: A two-variable four-valued function in P(4) and a two-variable two-valued function in B(2) such that

For $C_{in} = 0$

$$f_4\ \text{sum} = 1 \cdot [{}^0x_1^0\ {}^1x_2^3 + {}^1x_1^1\ {}^0x_2^2 + {}^2x_1^3\ {}^3x_2^3 + {}^3x_1^3\ {}^2x_2^3]$$

$$+ 2 \cdot [{}^0x_1^0\ {}^2x_2^3 + {}^1x_1^1\ {}^1x_2^2 + {}^2x_1^2\ {}^0x_2^1 + {}^3x_1^3\ {}^3x_2^3]$$

$$+ 3 \cdot [{}^0x_1^0\ {}^3x_2^3 + {}^1x_1^1\ {}^2x_2^2 + {}^2x_1^2\ {}^1x_2^1 + {}^3x_1^3\ {}^0x_2^0]$$

$$f_4\ C_{out} = 3 \cdot [{}^2x_1^3\ {}^2x_2^3 + {}^1x_1^3\ {}^3x_2^3 + {}^3x_1^3\ {}^1x_2^3]$$

$$f_2\ \text{sum} = x_1\bar{x}_2 + \bar{x}_1x_2$$

$$f_2\ C_{out} = x_1x_2$$

For $C_{in} = 3$

$$f_4 \text{ sum} = 1 \cdot [{}^{0}x_1^{0}\,{}^{0}x_2^{2} + {}^{1}x_1^{3}\,{}^{3}x_2^{3} + {}^{3}x_1^{3}\,{}^{1}x_2^{3} + {}^{2}x_1^{3}\,{}^{2}x_2^{3}]$$

$$+ 2 \cdot [{}^{0}x_1^{0}\,{}^{1}x_2^{2} + {}^{1}x_1^{1}\,{}^{0}x_2^{1} + {}^{2}x_1^{3}\,{}^{3}x_2^{3} + {}^{3}x_1^{3}\,{}^{2}x_2^{3}]$$

$$+ 3 \cdot [{}^{0}x_1^{0}\,{}^{2}x_2^{2} + {}^{1}x_1^{1}\,{}^{1}x_2^{1} + {}^{2}x_1^{2}\,{}^{0}x_2^{0} + {}^{3}x_1^{3}\,{}^{3}x_2^{3}]$$

$$f_4 \; C_{out} = 3 \cdot [{}^{3}x_1^{3} + {}^{3}x_2^{3} + {}^{2}x_1^{3}\,{}^{1}x_2^{3} + {}^{1}x_1^{3}\,{}^{2}x_2^{3}]$$

$$f_2 \text{ sum} = \overline{x}_1\overline{x}_2 + x_1x_2$$

$$f_2 \; C_{out} = x_1 + x_2$$

For $m = 4$ $\quad i,j = \begin{cases} 0,3 \\ 1,2 \end{cases}$

Test for f_2 sum, f_4 sum with $C_{in} = 0$

Step 1. Set $i = 0$, $j = 3$

Step 2. f_2 sum $(0,3) = f_4$ sum $(0,3)$, $\delta_{0,3} = 3$

Step 3. f_2 sum $(3,0) = f_4$ sum $(3,0)$, $\delta_{3,0} = 3$

f_2 sum $(0,0) = f_4$ sum $(0,0)$, $\delta_{0,0} = 0$

f_2 sum $(3,3) \neq f_4$ sum $(3,3)$, $\alpha_{3,3} = 0$, $\beta_{3,3} = 2$

Step 4. $\Sigma\delta = 3 = \delta_{0,3};\delta_{3,0};\delta_{0,0}$

$\Sigma\delta < 4$

$\therefore$ Repeat Steps 1-4 for $i = 1$, $j = 2$

Step 1. Set $i = 1$, $j = 2$

Step 2. f_2 sum $(1,2) \neq f_4$ sum $(1,2)$, $\alpha_{1,2} = 2$, $\beta_{1,2} = 3$

Step 3. f_2 sum (2,1) $\neq$ f_4 sum (2,1) , $\alpha_{2,1} = 2$, $\beta_{2,1} = 3$

f_2 sum (1,1) $\neq$ f_4 sum (1,1) , $\alpha_{1,1} = 1$ $\beta_{1,1} = 2$

f_2 sum (2,2) $\neq$ f_4 sum (2,2) , $\alpha_{2,2} = 1$ $\beta_{2,2} = 0$

Step 4. $\Sigma\delta = 0$

Test for f_2 sum, f_4 sum with $C_{in} = 3$

Step 1. Set i = 0 , j = 3

Step 2. f_2 sum (0,3) = f_4 sum (0,3) , $\delta_{0,3} = 0$

Step 3. f_2 sum (3,0) = f_4 sum (3,0) , $\delta_{3,0} = 0$

f_2 sum (0,0) $\neq$ f_4 sum (0,0) , $\delta_{0,0} = 3$, $\beta_{0,0} = 1$

f_2 sum (3,3) = f_4 sum (3,3) , $\delta_{3,3} = 3$

Step 4. $\Sigma\delta = 3 = \delta_{0,3};\delta_{3,0};\delta_{3,3}$

$\Sigma\delta < 4$

$\therefore$ Repeat steps 1-4 for i = 1 , j = 2

Step 1. Set i = 1 , j = 2

Step 2. f_2 sum (1,2) $\neq$ f_4 sum (1,2) , $\alpha_{1,2} = 1$, $\beta_{1,2} = 0$

Step 3. f_2 sum (2,1) $\neq$ f_4 sum (2,1) , $\alpha_{2,1} = 1$, $\beta_{2,1} = 0$

f_2 sum (1,1) $\neq$ f_4 sum (1,1) , $\alpha_{1,1} = 2$, $\beta_{1,1} = 3$

f_2 sum (2,2) $\neq$ f_4 sum (2,2) , $\alpha_{2,2} = 2$, $\beta_{2,2} = 1$

Step 4. $\Sigma\delta = 0$

Test for f_2 C_{out} , f_4 C_{out} with $C_{in} = 0$. Following the same procedure the results are:

For i = 0 , j = 3

$\Sigma\delta = 4 = \delta_{0,3};\delta_{3,0};\delta_{3,3};\delta_{0,0}$

$\delta_{0,3} = 0$

$\delta_{3,0} = 0$

$\delta_{3,3} = 3$

$\delta_{0,0} = 0$

Similarly, the test for $f_2\ C_{out}$, $f_4\ C_{out}$ with $C_{in} = 3$ yields:

For $i = 0$, $j = 3$

$\Sigma\delta = 4 = \delta_{0,3};\delta_{3,0};\delta_{3,3};\delta_{0,0}$

$\delta_{0,3} = 3$

$\delta_{3,0} = 3$

$\delta_{3,3} = 3$

$\delta_{0,0} = 0$

Summarizing the tests above for C_{out},

For $B(2) \rightarrow P(4)_{0,3}$

$\Sigma\delta = 4$

$f_2\ C_{out} = f_4\ C_{out}$ for $C_{in} = 0$

$f_2\ C_{out} = f_4\ C_{out}$ for $C_{in} = 3$

Summarizing the tests for sum,

For $B(2) \rightarrow P(4)_{0,3}$	For $B(2) \rightarrow P(4)_{1,2}$
For $C_{in} = 0$	For $C_{in} = 0$
$\Sigma\delta = 3$	$\Sigma\delta = 0$
$\alpha_{3,3} = 0$, $\beta_{3,3} = 2$	$\alpha_{1,2} = 2$, $\beta_{1,2} = 3$
	$\alpha_{2,1} = 2$, $\beta_{2,1} = 3$
	$\alpha_{1,1} = 1$, $\beta_{1,1} = 2$
	$\alpha_{2,2} = 1$, $\beta_{2,2} = 0$

For $C_{in} = 3$

$\Sigma\delta = 3$

$\alpha_{0,0} = 3$, $\beta_{0,0} = 1$

For $C_{in} = 3$

$\Sigma\delta = 0$

$\alpha_{1,2} = 1$, $\beta_{1,2} = 0$

$\alpha_{2,1} = 1$, $\beta_{2,1} = 0$

$\alpha_{1,1} = 2$, $\beta_{1,1} = 3$

$\alpha_{2,2} = 2$, $\beta_{2,2} = 1$

The apparent choice for the sum is the $B(2) \rightarrow P(4)_{0,3}$ mapping.

For $C_{in} = 0$

f_2 sum $(3,3) \neq f_4$ sum $(3,3)$

f_2 sum $(3,3) = f_4$ sum $(3,3) \dot{+} F_4^*(3,3)$

For $C_{in} = 3$

f_2 sum $(0,0) \neq f_4$ sum $(0,0)$

f_2 sum $(0,0) = f_4$ sum $(0,0) \dot{+} F_4^*(0,0)$

For the full adder the two $F_m^*(i,j)$ terms can be computed:

$$F_4^*(3,3) = 0 \cdot {}^{3}x_{14}^{3}\,{}^{3}x_{24}^{3} \text{ , for the sum, } C_{in} = 0$$

$$F_4^*(0,0) = 3 \cdot {}^{0}x_{14}^{0}\,{}^{0}x_{24}^{0} \text{ , for the sum, } C_{in} = 3$$

Some interpretation of these terms and how they can be used to achieve a total B(2):P(4) solution is necessary. First of all there is no need in a single radix solution to realize a zero term. For this example the $F_4^*(0,0)$ term is necessary to force a value of zero in B(2) which is otherwise two in P(4). There are two ways to accommodate this term in a straightforward manner. The first method takes advantage of the definition of the binary select line.

$$F_4^*(3,3) = \overline{B}_s \cdot {}^3x_{14}^3\ {}^3x_{24}^3$$

Since the B_s line has a value of either zero or m-1, it can be included in the MIN term without modifying its behavior in P(4). Likewise, when $B_s = 3$, for the B(2) case, it will insure that the term goes to zero. This method is fine for the case where it is required for the term to become zero in B(2). However, a general solution which will accommodate either logic value for $B(2) \rightarrow P(4)_{0,3}$ is available. The standard B(2): P(4) "sum of products" gate in Figure 3.7 will allow any MIN term in the canonical form to be modified by $F_m^*(i,j)$ with no additional hardware. Shown in Figure 3.13 is the realization of $F_4^*(0,0)$ and $F_4^*(3,3)$ which allows the full adder to become B(2):P(4) compatible.

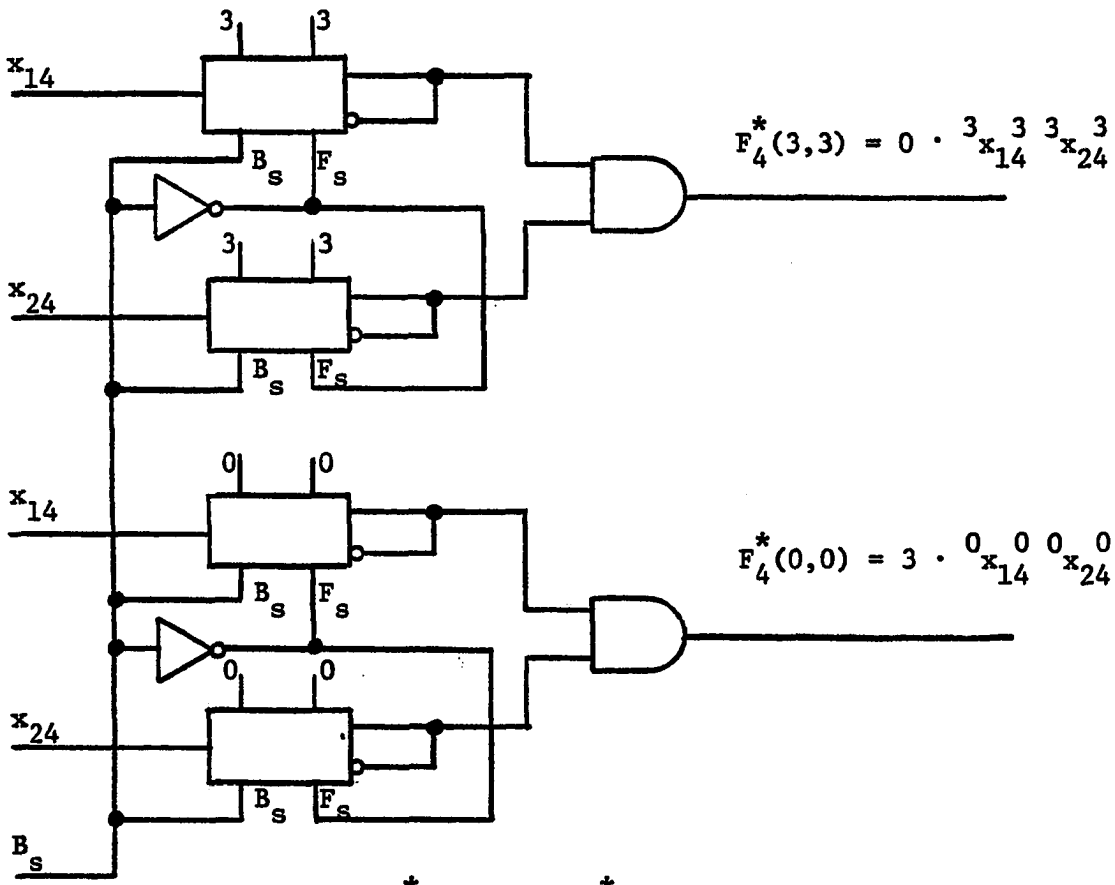


Figure 3.13. $F_4^*(0,0)$ and $F_4^*(3,3)$ for Full Adder Solution in B(2):P(4)

The hardware shown in Figure 3.13 realized the following terms in P(4):

$$^{3}x_{1}^{3}\ ^{3}x_{2}^{3} \quad \text{for } C_{in} = 0$$

and

$$^{0}x_{1}^{0}\ ^{0}x_{2}^{0} \quad \text{for } C_{in} = 3$$

The ability to modify any term in the canonical form is accomplished through the B_s and F_s lines. Remembering that the logical values of zero and three are the only valid levels for $B(2) \rightarrow P(4)_{0,3}$, logic values of one and two can be treated as "don't cares" for the mapping. Therefore, for any single variable it is necessary to map the valid B(2) inputs to a zero or three while allowing the P(4) algebra to function normally. The truth table in Table 3.8 demonstrates how the B(2):P(4) literal in Figure 3.7 will perform as described by properly choosing the B_s and F_s values. For $B_s = 0$, the circuit of Figure 3.7 will operate as a P(4) literal according to Table 2.9. This is indicated in Table 3.8 by showing that $^{i}x_{14}^{i}$ will take on its normal P(4) value while being a "don't care" in B(2). For $B_s = 3$, the circuit will become an inverter or non-inverter in B(2) depending on the value of the F_s line. Thus, any valid input in B(2) can be forced to either a logic zero or three by properly selecting the B_s and F_s values. This is precisely the function of $F_4^*(i,j)$. Note that even though a logic value of one or two is illegal in $B(2) \rightarrow P(4)_{0,3}$, the circuit will insure the output is a zero. While this has no immediate consequence for the example, it may be of value in minimizing hazards in B(2):P(4) circuits.

In summary, for the full adder, two terms are necessary for the B(2):P(4) realization using the $B(2) \rightarrow P(4)_{0,3}$ mapping with $C_{out} = 3$. These two terms are:

$$F_4^*(3,3) = 0 \cdot {}^{3}x_{14}^{3}\,{}^{3}x_{24}^{3} \quad \text{for } f_4 \text{ sum} \quad , C_{in} = 0$$

$$F_4^*(0,3) = 3 \cdot {}^{0}x_{14}^{0}\,{}^{0}x_{24}^{0} \quad \text{for } f_4 \text{ sum} \quad , C_{in} = 3$$

While the standard circuit of Figure 3.7 was shown to be capable of modifying any term in the canonical form with no hardware burden or modification, solutions should be chosen with the smallest value of $\delta_{i,j}$. Actually, two inverters were added to the standard circuit (Figure 3.13) to show the convenient realization of B_s and F_s for the full adder solution.

Table 3.8. B(2):P(4) Literal of Figure 3.7 for $\alpha_{i,j}$, i = 0,3 (x = don't cares)

x_{14}	i	B_s	F_s	${}^{i}x_{14}^{i}$	${}^{i}x_{12}^{i}$
0	P(4)	0	x	P(4)	x
0	0	3	0	x	3
0	3	3	3	x	0
1	P(4)	0	x	P(4)	x
1	0	3	0	x	0
1	3	3	3	x	0
2	P(4)	0	x	P(4)	x
2	0	3	0	x	0
2	3	3	3	x	0
3	P(4)	0	x	P(4)	x
3	0	3	0	x	0
3	3	3	3	x	3

The first solution offered in Figures 3.11 and 3.12 requires 112 collectors, 37 mirrors, and 9 switches to build. The motivation for this solution comes from the fact that the "clockwise cycling" operation provides a truth table very similar to Table 3.6(a). Figure 3.11 takes complete advantage of linear summation and threshold detection. On the other hand, the second approach to a solution began by expressing Table 3.6(a) and (b) in a near minimal form using the literal unary operation. If this approach were used, approximately 15 circuits similar to the standard "sum of products" circuit of Figure 3.7 would be required. This would require over 1500 collectors, 600 mirrors, and 300 switches. The literal unary operator has more than an order of magnitude greater hardware burden than the "clockwise cycling" operator. This example clearly demonstrates that no uniform design technique exists that will guarantee that a set of gates will provide the best realization.

The basic set of gates or unary operators which are chosen to realize a given P(4) truth table are totally invisible to the method proposed for finding maximum compatibility in P(4) for a particular B(2) function. Even so, the functions proposed by the algorithm (Figure 3.13) are very similar to those developed for the first solution using the intuitive approach (Figure 3.12).

Clearly this procedure for finding the maximum compatible mapping and all incompatibilities for completely specified functions makes no attempt to minimize either the P(4) or B(2) realization. It does minimize the B(2):P(4) circuitry by choosing the best mapping and defining the functions required for total compatibility, if any. If incompletely specified functions in either P(4) or B(2) are to be analyzed, the procedure can be altered to accommodate "don't cares."

Step 2. $f_2(i,j) = f_m(i,j)$ set $\delta_{i,j} = f_2 i,j) = f_m(i,j)$

or $\delta_{i,j}$ = B(2):P(m) implicant

$f_2(i,j) \neq f_m(i,j)$

If $f_m(i,j)$ = "Don't Care" set $\alpha_{i,j} = f_2(i,j)$

or $\alpha_{i,j}$ = B(2) term

set $\beta_{i,j} = \alpha_{i,j}$

and $\delta_{i,j} = \alpha_{i,j}$

Or $f_2(i,j)$ = "Don't Care" set $\beta_{i,j} = f_m(i,j)$

or $\beta_{i,j}$ = P(m) term

if $\beta_{i,j}$ = i or j

set $\alpha_{i,j} = \beta_{i,j}$

and $\delta_{i,j} = \beta_{i,j}$

otherwise

set $\alpha_{i,j} = f_2(i,j)$

or $\alpha_{i,j}$ = B(2) Don't Care

Otherwise $f_2(i,j) \neq f_m(i,j)$, no don't cares are involved,

set $\alpha_{i,j} = f_2(i,j)$

or $\alpha_{i,j}$ = B(2) term

set $\beta_{i,j} = f_m(i,j)$

or $\beta_{i,j}$ = P(m) term

Summary

In this chapter B(2):P(4) combinational design was done for the two algebras of Vranesic, et al, and Allen and Givone. Two designs were considered to help expand the dual radix concepts. An example examined a completely specified P(4) function for Vranesic's algebra. The idea of the k and r lines being made to default to a logic value

necessary for B(2) using a binary select line was introduced. The burden of the binary select on the B(2):P(4) hardware was considered . Also, a standard B(2):P(4) "product of sums" gate for this algebra was designed and is shown in Figure 3.4. The standard circuit was bench marked against the concepts of functional pairing and common signal paths, as well as hardware burden.

The second example is a B(2):P(4) combinational design for Allen and Givone's algebra. For the literal operator the a and b lines can default to two different values for $B(2) \rightarrow P(4)_{0,3}$. This requires the introduction of a function select line to determine the a and b values for each case. Figure 3.6 shows the additional hardware and the truth table for the binary select and function select lines for the literal operator. A standard B(2):P(4) "sum of products" gate for this algebra was designed and is shown in Figure 3.7. Table 3.2 is a summary of the B(2):P(4) bench marks for both algebras.

Other operators were considered since no single standard set of gates will be best for every situation and no one design technique will show which functions to use. Linear summation and thresholding are operations which can be easily realized in I^2L. Both Post algebra monotone and disjoint operators were introduced. The circuitry for $D_i(x)$ and $C_i(x)$ in B(2):P(4) is shown in Figure 3.10. Equivalence relationships for each of the unary operators were given.

A third example in this chapter was a B(2):P(4) full adder. A P(4) adder was designed taking advantage of the "clockwise cycling" operation. The schematic for this P(4) circuit is shown in Figure 3.11. This circuit can be modified by the literals shown in Figure 3.12 to

accomplish a B(2):P(4) full adder for $B(2) \rightarrow P(4)_{0,3}$ with $C_{out} = 3$. Also, the realization in Figures 3.11 and 3.12 is an order of magnitude less costly than one using the literal only.

Finally, an algorithm for finding the maximum compatible mapping from B(2) to P(m) for completely and incompletely specified functions was developed. It also identifies the incompatibilities in such a way as to define a function $F_m^*(i,j)$. This algorithm for both classes of problems is given here for purposes of clarification and summarization.

Given: An n variable m-valued function in P(m) and an n variable two-valued function in B(2) such that

$$x_{1m}, x_{2m}, \ldots x_{nm} \in P(m);\ m = 2^N,\ N > 1$$

$$x_{12}, x_{22}, \ldots x_{n2} \in B(2)$$

$$\text{For } B(2) \rightarrow P(m)_{i,j},\ i,j = \begin{cases} i = 0,\ j = m-1 \\ i \oplus 1,\ j - 1 \\ \\ i \oplus 2^{N-1}-1,\ j-2^{N-1}-1 \end{cases}$$

Step 1. Set $i = 0$, $j = m-1$

Step 2. $f_2(i,j) = f_m(i,j)$ set $\delta_{i,j} = f_2(i,j) = f_m(i,j)$

or $\delta_{i,j}$ = B(2):P(m) implicant

$f_2(i,j) \neq f_m(i,j)$

If $f_m(i,j)$ = "Don't care" set $\alpha_{i,j} = f_2(i,j)$

or $\alpha_{i,j}$ = B(2) term

set $\beta_{i,j} = \alpha_{i,j}$

and $\delta_{i,j} = \alpha_{i,j}$

Or	$f_2(i,j)$ = "Don't Care"	set $\beta_{i,j} = f_m(i,j)$
		or $\beta_{i,j}$ = P(m) term
		if $\beta_{i,j}$ = i or j
		set $\alpha_{i,j} = \beta_{i,j}$
		and $\delta_{i,j} = \beta_{i,j}$
		otherwise
		set $\alpha_{i,j} = f_2(i,j)$
		or $\alpha_{i,j}$ = B(2) Don't Care
Otherwise	$f_2(i,j) \neq f_m(i,j)$	set $\alpha_{i,j} = f_2(i,j)$
		or $\alpha_{i,j}$ = B(2) term
		set $\beta_{i,j} = f_m(i,j)$
		or $\beta_{i,j}$ = P(m) term

Step 3. Find the remaining implicants. Repeat Steps 1 and 2 2^n-1 times for each mapping.

Step 4. If $\Sigma\delta_{i,j} = 2^n$, then $B(2) \rightarrow P(m)_{i,j}$ exists
If $\Sigma\delta_{i,j} < 2^n$, then save all terms and repeat steps 1 to 4 $2^{N-1}-1$ times to check all mappings (i = 1 , j = m-2, etc.).

Step 5. If $\Sigma\delta_{i,j} < 2^n$ for all mappings then $B(2) \rightarrow P(m)_{i,j}$ exists if and only if there is a function $F_m^*(i,j)$ such that

$$f_2(i,j) = f_m(i,j) \dot{+} F_m^*(i,j)$$

for every

$$f_2(i,j) \neq f_m(i,j)$$

For the worse case $\Sigma\delta_{i,j} = 0$, 2^n sets of α and β terms exist and for n = 2 (for Allen and Givone's algebra)

$$F_m^*(i,j) = S_{ij} \cdot {}^{i}x_{1m}^{i}\,{}^{j}x_{2m}^{j} + S_{ii} \cdot {}^{i}x_{1m}^{i}\,{}^{i}x_{2m}^{i} + S_{ji} \cdot {}^{j}x_{1m}^{j}\,{}^{i}x_{2m}^{i} + S_{jj} \cdot {}^{j}x_{1m}^{j}\,{}^{j}x_{2m}^{j}$$

where $S_{ij} = \alpha_{ij}$

Or for Vranesic, et al

$$F_m^*(i,j) = \vec{x}_{1m}^{m-i} \cdot \vec{x}_{2m}^{m-j}\,k_{ij} + \vec{x}_{1m}^{m-i} \cdot \vec{x}_{2m}^{m-i}\,k_{ii} + \vec{x}_{1m}^{m-j} \cdot \vec{x}_{2m}^{m-i}\,k_{ji} + \vec{x}_{1m}^{m-j} \cdot \vec{x}_{2m}^{m-j}\,k_{jj}$$

Step 6. Choosing the largest $\delta_{i,j}$ in Step 5 will minimize the number of $F_m^*(i,j)$ terms necessary to guarantee $B(2) \rightarrow P(m)_{i,j}$ exists.

CHAPTER IV

DUAL RADIX PROCESSOR CONSIDERATIONS

B(2):P(4)

The motivation for much of the work done thus far has been to propose and design hardware and methods for dual radix logic. While the examples demonstrate the feasibility for the B(2):P(4) case, it has been shown that P(m) designs can also be considered. Beyond the combinational work of Chapters II and III, additional development needs to be done in the sequential area to allow dual radix processor considerations to become a reality. Since many of the combinational circuits proposed have been constructed and reported on in Appendix A, hopefully the attempt to keep the work practically oriented is obvious. However, from the outset nearly every author who has considered multi-valued memory elements has dealt with the inability to compete with their B(2) counterparts. This work will be no different. It has been projected that B(2) memory elements will continue to decrease in price by two orders of magnitude by 1990 [31]. This fact may not diminish the academic appetite for MVL memory design, but it does call for an awareness that higher radix processors may have a hybrid architecture initially. That is, from a

practical sense, parts of a P(4) processor may be constructed from a B(2) component. At this point the direction of the work on memory devices is potentially schizophrenic. The idea is to examine the difficulty of "bringing along" a B(2) machine into a higher radix architecture. The antithesis of this thinking is to force portions of a higher radix machine back into B(2) constructions. Nevertheless, both ideas will be examined.

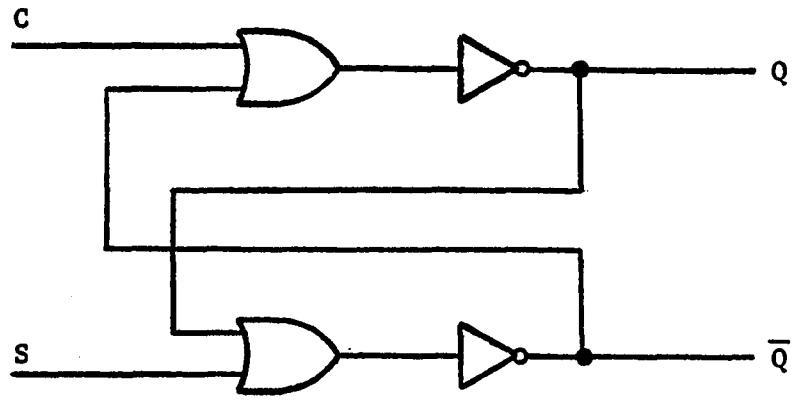
MVL Memory Elements

Irving and Nagle [32] have proposed a family of memory devices based on several multi-valued logic operators. They have described five properties that an MVL memory element must exhibit. They are:

1. The device must be defined for any N_s. (N_s an integer greater than two.)
2. The device must have N_s stable states.
3. The device must have at least one output which presents a different logic value for each of the N_s stable states.
4. The device must remain in each stable state indefinitely in the absence of external excitation.
5. The device must be able to obtain any stable state A from any other stable state B in a single transition with proper excitation.

The four-valued memory device suggested in [32] is shown in Figure 4.1(a) and the table of next states for each combination of inputs and present state is shown in Figure 4.1(b). The next-state equation for this circuit is:

$$Q(t \oplus 1) = S + \overline{C} \cdot Q(t)$$



(a)

	S	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
	C	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
	0	0	0	0	0	1	1	1	x	2	2	x	x	3	x	x	x
Q(t)	1	1	1	1	0	1	1	1	x	2	2	x	x	3	x	x	x
	2	2	2	1	0	2	2	1	x	2	2	x	x	3	x	x	x
	3	3	2	1	0	3	2	1	x	2	2	x	x	3	x	x	x

Q(t ⊕ 1)

(b)

Figure 4.1. (a) Four-Valued Memory, (b) Next-State Truth Table

An x entry for the next state in Figure 4.1(b) indicates that the output is not deterministic for the given inputs. For a memory circuit using the MAX/OR gate, the non-deterministic input conditions occur in P(4) when the arithmetic sum of the uncomplemented inputs is greater than or equal to four [32].

Much analysis has been done on the cross-coupled $\overline{\text{MAX}}$ MVL memory element. Wills has developed a behavioral model for MVL memory elements

[36]. Sheafor has studied the problem of state assignment for multiple-valued circuits [34]. Tull has reported on a circuit instability for the S transition from 0 to 2 and back to 0 [6]. While the circuit has been thoroughly examined, it is necessary to see how easily it can be adapted to the B(2):P(4) case. First of all, the next-state equation needs to be tested for both mappings. The next-state table of Figure 4.1(b) is shown in Table 4.1 for $B(2) \rightarrow P(4)_{1,2}$ and $B(2) \rightarrow P(4)_{0,3}$. From Table 4.1

Table 4.1. Next-State Table for $B(2) \rightarrow P(4)_{1,2}$ and $B(2) \rightarrow P(4)_{0,3}$

	S	1	1	2	2
	C	1	2	1	2
Q(t)	0	1	1	2	x
	1	1	1	2	x
	2	2	1	2	x
	3	2	1	2	x

$Q(t \oplus 1)$

$B(2) \rightarrow P(4)_{1,2}$

	S	0	0	3	3
	C	0	3	0	3
Q(t)	0	0	0	3	x
	1	1	0	3	x
	2	2	0	3	x
	3	3	0	3	x

$Q(t \oplus 1)$

$B(2) \rightarrow P(4)_{0,3}$

it can be seen that any possible sequence in either mapping exists. That is, for any legal input sequence, the output sequence is legal for the same mapping. An additional property for $B(2) \rightarrow P(m)_{i,j}$ compatible MVL memory is required.

6. An MVL memory device is $B(2) \rightarrow P(m)_{i,j}$ compatible if for any input sequence such as

$$\{x_1x_2,\ x_1\bar{x}_2,\ \bar{x}_1\bar{x}_2,\ \bar{x}_1x_2,\ \bar{x}_1\bar{x}_2,\ x_1\bar{x}_2,\ x_1x_2\}$$

$$x_1,\ x_2 \in P(m)_{i,j}$$

there exists an output sequence $f(x_1,x_2)$ such that

$$f(x_1,x_2) \in P(m)_{i,j}$$

and there is at least one $f(x_1,x_2) = i$ and $f(x_1,x_2) = j$. If a single input can send the MVL memory device to a stable state contained in the same mapping as the input for all present states, not necessarily in the same mapping, the device is said to have total mapping recovery.

7. If for every $Q(t) \neq \in P(m)_{i,j}$ there exists at least one input function $\{x_1,x_2\}$, $x_1,x_2 \in P(m)_{i,j}$ such that $Q(t \oplus 1) \in P(m)_{i,j}$ then the MVL memory device is said to have total mapping recovery, TMR, for $B(2) \rightarrow P(m)_{i,j}$.

The cross-coupled $\overline{MAX}$ MVL memory device in Table 4.1 is $B(2) \rightarrow P(4)_{0,3}$ and $B(2) \rightarrow P(4)_{1,2}$ compatible according to Property 6. Also, this memory device has TMR for $B(2) \rightarrow P(4)_{0,3}$ and $B(2) \rightarrow P(4)_{1,2}$. The I^2L realization of Figure 4.1(a) is shown in Figure 4.2. A single EXAR XR-500 I^2L master chip, 122 x 135 mils, can accommodate 128 cross-coupled $\overline{MAX}$ MVL memory devices. This makes no allowances for address and data bus multiplexing and decoding. However, even with this overhead, the approximate cost per bit is ten cents. The cost of binary storage was in this range about 1972 to 1975 [31].

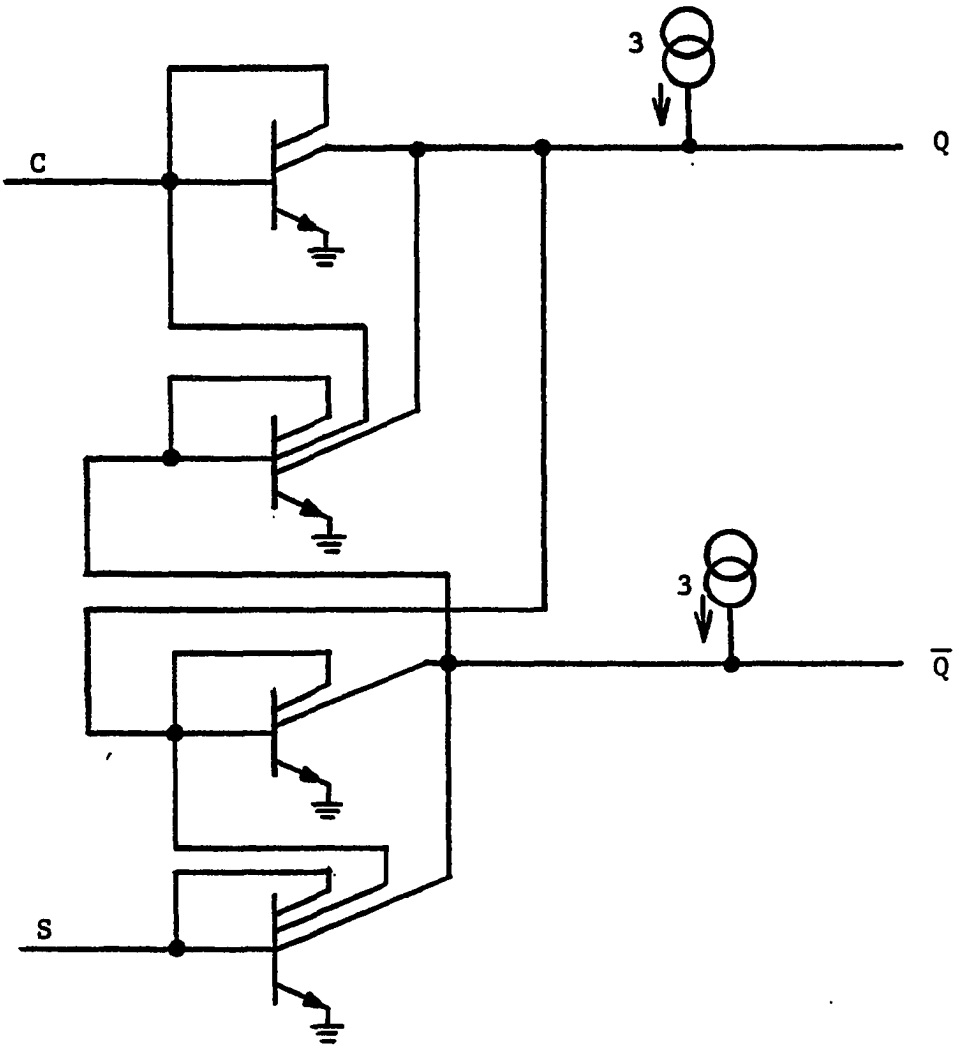


Figure 4.2. I^2L Four-Valued $\overline{MAX}$ Cross-Coupled Memory Device

The cross-coupled $\overline{MAX}$ MVL memory element is well suited for dual radix work for either mapping for B(2):P(4) work. The only feature which compromises its usefulness is the presence of non-deterministic next-states. The RS flip-flop has non-deterministic next-states in B(2). The same method for eliminating these states in B(2) will work in P(m).

The solution is the J-K flip-flop. The $\overline{Q}$ output is "ANDED" with the J input for S, and the Q output is "ANDED" with the K input for C. Wojcik [33] has reported on an MVL analog of the binary J-K flip-flop. The next-state equation is given as:

$$Q(t \oplus 1) = J \cdot \overline{Q(t)} + \overline{K} \cdot Q(t) + J \cdot \overline{K}$$

Based upon his model for asynchronous combinational circuits [35], Wojcik has developed a method of state assignment for defining asynchronous sequential behavior [33]. The next-state table for $B(2) \rightarrow P(4)_{0,3}$ and $B(2) \rightarrow P(4)_{1,2}$ for the J-K MVL memory device is given in Table 4.2. Since

Table 4.2. Next-State for $B(2) \rightarrow P(4)_{1,2}$ and $B(2) \rightarrow P(4)_{0,3}$ for J-K MVL Memory

	J	1	1	2	2
	K	1	2	1	2
	0	1	1	2	2
Q(t)	1	1	1	2	2
	2	2	1	2	1
	3	2	1	2	1

$Q(t \oplus 1)$

$B(2) \rightarrow P(4)_{1,2}$

	J	0	0	3	3
	K	0	3	0	3
	0	0	0	3	3
Q(t)	1	1	0	3	2
	2	2	0	3	1
	3	3	0	3	0

$Q(t \oplus 1)$

$B(2) \rightarrow P(4)_{0,3}$

a memory device could have been in any previous state, all values of Q(t) are valid. However, once an input sequence in a given mapping occurs the present state and next-state must reside in the same domain. Thus, while there are no non-deterministic next-states for the J-K MVL memory device, there are input values which will never occur. For any B(2):P(4) circuitry these inputs are {01, 02, 10, 13, 20, 23, 31, 32}. If it is

assumed that the MVL memory device operates in the fundamental mode, then these inputs present no problems for B(2):P(4) circuitry.

The J-K MVL memory device is well-suited for dual radix work. It satisfies the seven properties for $B(2) \rightarrow P(4)_{1,2}$ and $B(2) \rightarrow P(4)_{0,3}$. It has no non-deterministic next-states. If two MIN gates (Figure 2.11) are included in the schematic of Figure 4.2, a J-K MVL memory device can be constructed at a cost of 28 collectors and 12 mirrors.

Bus Design

In a P(m) machine ($m = 2^N$, $N > 1$) there are 2^{N-1} B(2) machines available for consideration. Since there can be m different logic values on a given signal path, it would be convenient to allow all B(2) circuitry access to any mapping. This is easily accomplished with an m-valued bus. Figure 4.3 is a block diagram picturing an m-valued bus communicating to several $B(2) \rightarrow P(m)_{i,j}$ circuits.

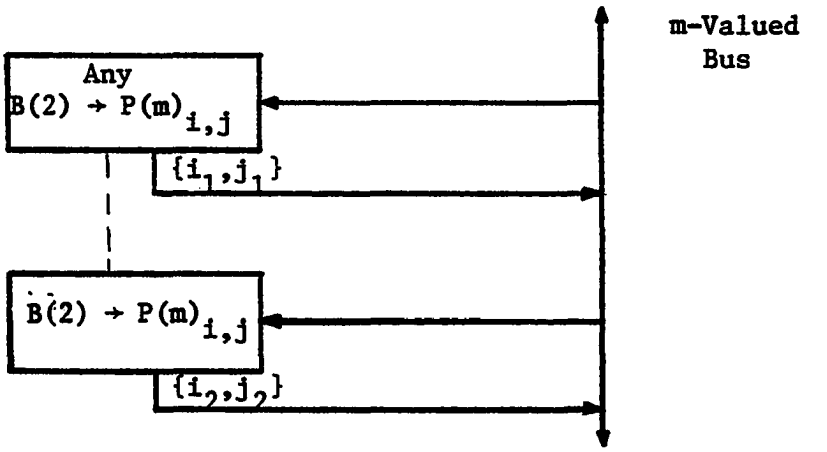


Figure 4.3. M-Valued Bus Connections

If all B(2) machines operating within a single host P(m) machines use the same mapping ($i_1 = i_2$, $j_1 = j_2$ in Figure 4.3), then the bus

structure is completely defined. Greater flexibility in B(2):P(4) circuitry would demand that all B(2) circuits operating within a single host P(m) machine be allowed to communicate regardless of the mapping. For P(4) the situation of two different mappings communicating with one another is shown in Figure 4.4. The circuits which will permit two different mappings to communicate are shown in Figure 4.5. If a P(4)

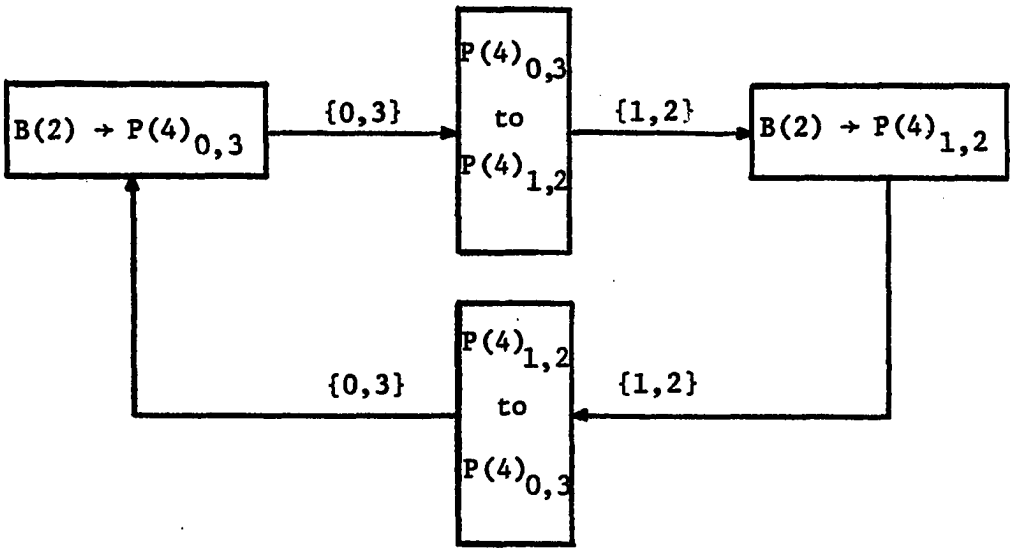
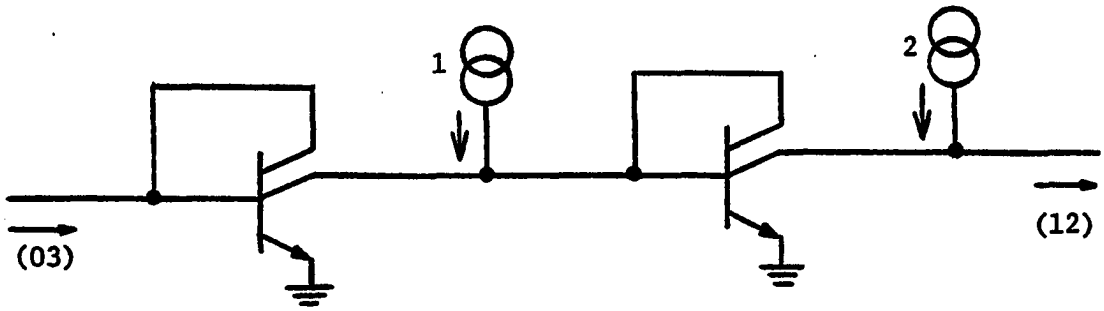


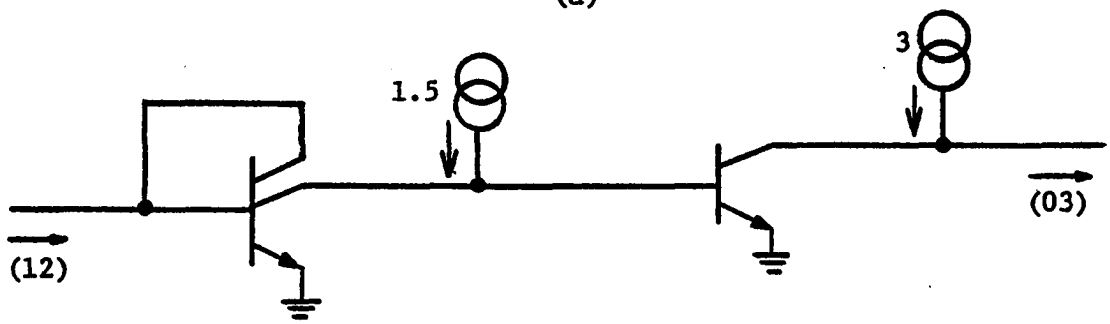
Figure 4.4. Bus Connection for Two Mappings

bus is driving the $P(4)_{0,3} \rightarrow P(4)_{1,2}$ circuit, the only possible output is {12}. Likewise, when a P(4) bus drives the $P(4)_{1,2} \rightarrow P(4)_{0,3}$ circuit the only possible output is {03}. In actual operation it should be encumbent upon the driving circuitry to output only logical values valid for the mapping under which it is functioning. Having B(2) buses which allow disjoint machines to interface is desirable. However, proliferation of the P(m) bus structure into binary buses should be done for additional flexibility and should enhance the overall architecture.



$P(4)_{0,3} \rightarrow P(4)_{1,2}$

(a)



$P(4)_{1,2} \rightarrow P(4)_{0,3}$

(b)

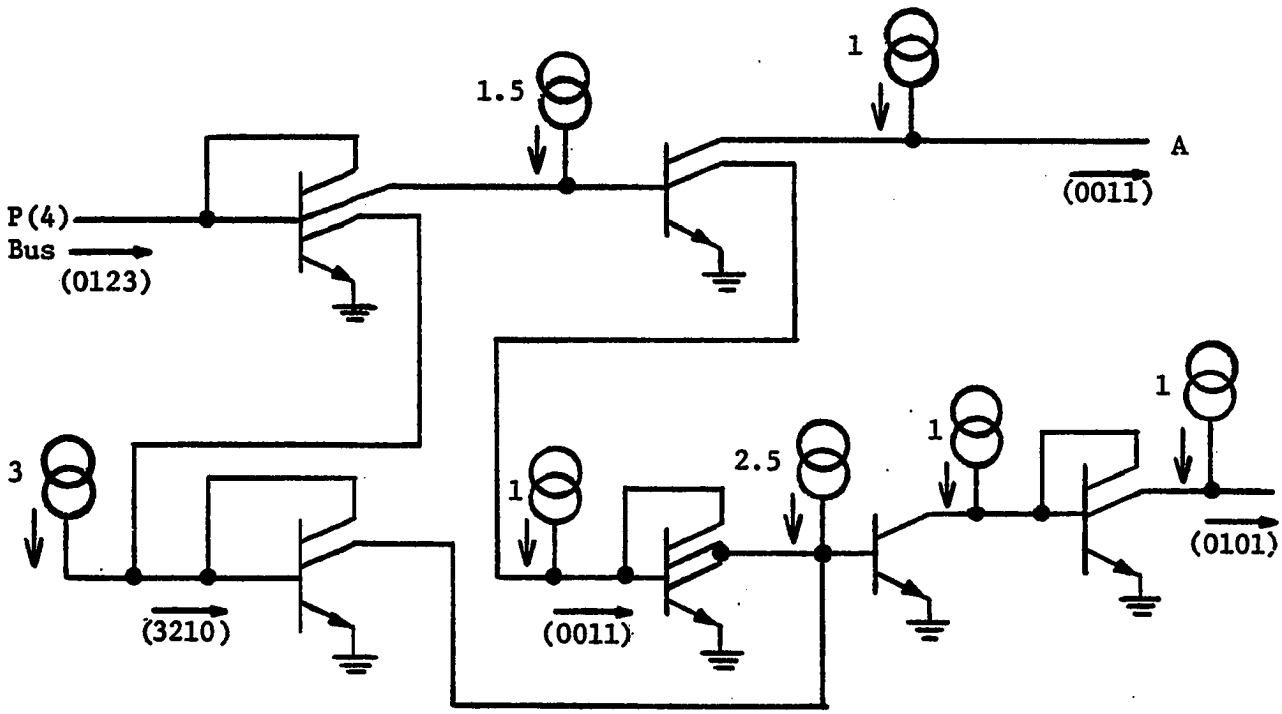
Figure 4.5. (a) $P(4)_{0,3} \rightarrow P(4)_{1,2}$, (b) $P(4)_{1,2} \rightarrow P(4)_{0,3}$

A single P(m) bus line could be allowed to have all m signals represented by N binary lines ($m = 2^N$). For P(4) the decoding/encoding

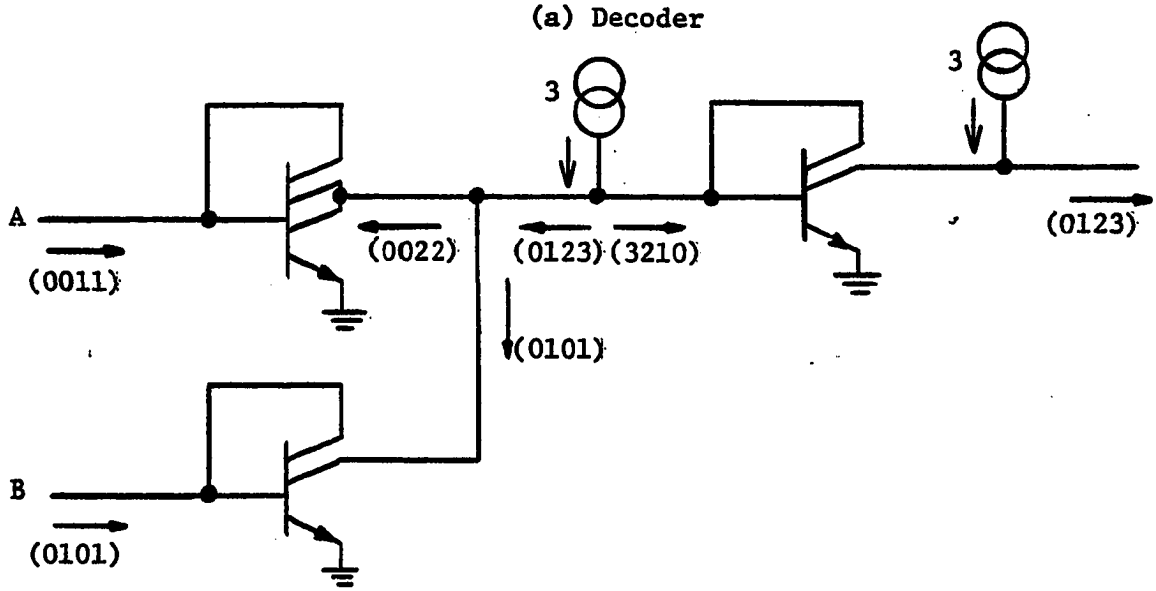
Table 4.3. P(4) to B(2) Decoding/Encoding

P(4)		B(2) AB		P(4)
0	→	00	→	0
1	→	01	→	1
2	→	10	→	2
3	→	11	→	3

method shown in Table 4.3 is logical. Circuits which will accomplish the decoding/encoding scheme were proposed by Edwards [37]. The schematics are shown in Figure 4.6. The circuits in Figure 4.6 expect the



(a) Decoder



(b) Encoder

Figure 4.6. (a) P(4) → B(2) Decoder, (b) B(2) → P(4) Encoder

bus to "source" current. Also, the A and B outputs in Figure 4.6(a) "source" the values of current shown. Thus, the circuitry in Figure 4.6 (a) and (b) could be cascaded and the original P(4) signal would be obtained. In Edwards' work [37], a sink to source conversion needs to be added to the $\overline{A}$ output to allow cascading of his design. Also, Edwards chose to treat inputs as "sources" and outputs as "sinks." Throughout this work, inputs have been treated as "sinks" and outputs as "sources."

As mentioned in the introduction of this chapter, parts of a P(4) machine might be constructed from B(2) components. The obvious B(2) component is the memory element. Using the circuits of Figure 4.6(a) and (b) the necessary interfacing can be accomplished. Shown in Figure 4.7 is a 256 x 8 B(2) memory interfaced to P(4) address and data buses.

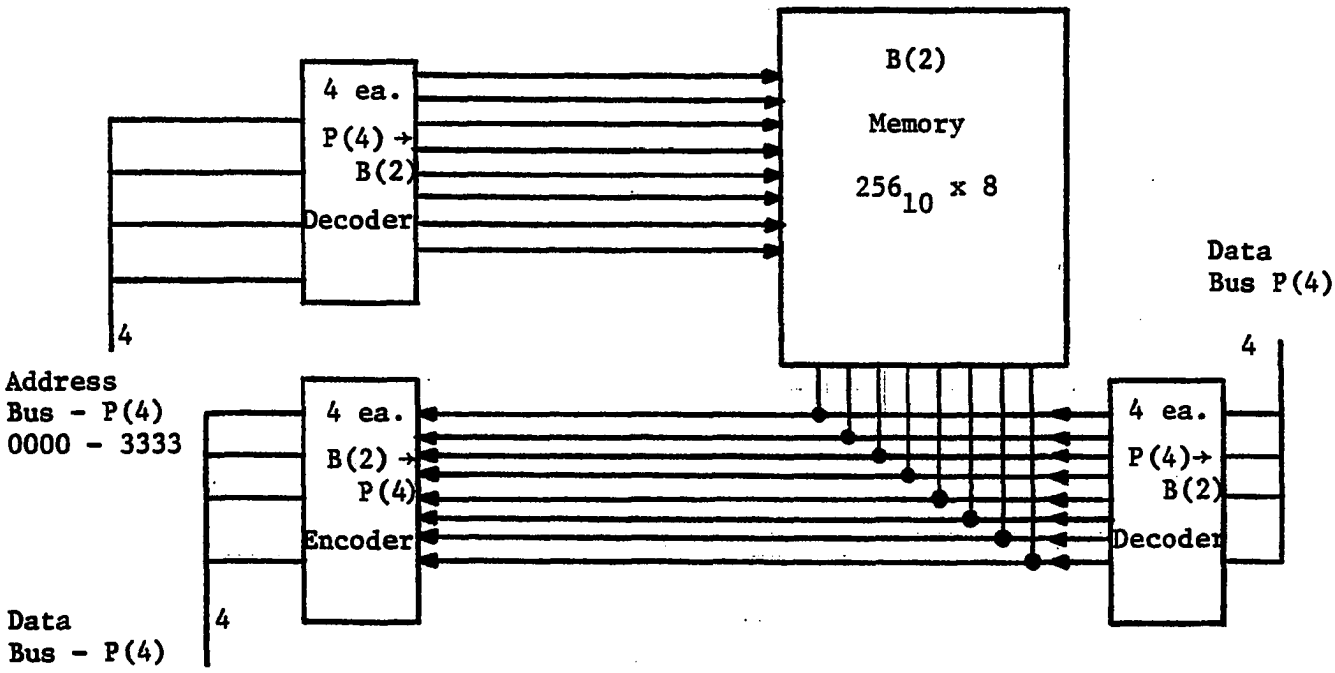


Figure 4.7. 256 x 8 B(2) Memory Interfacing to P(4)

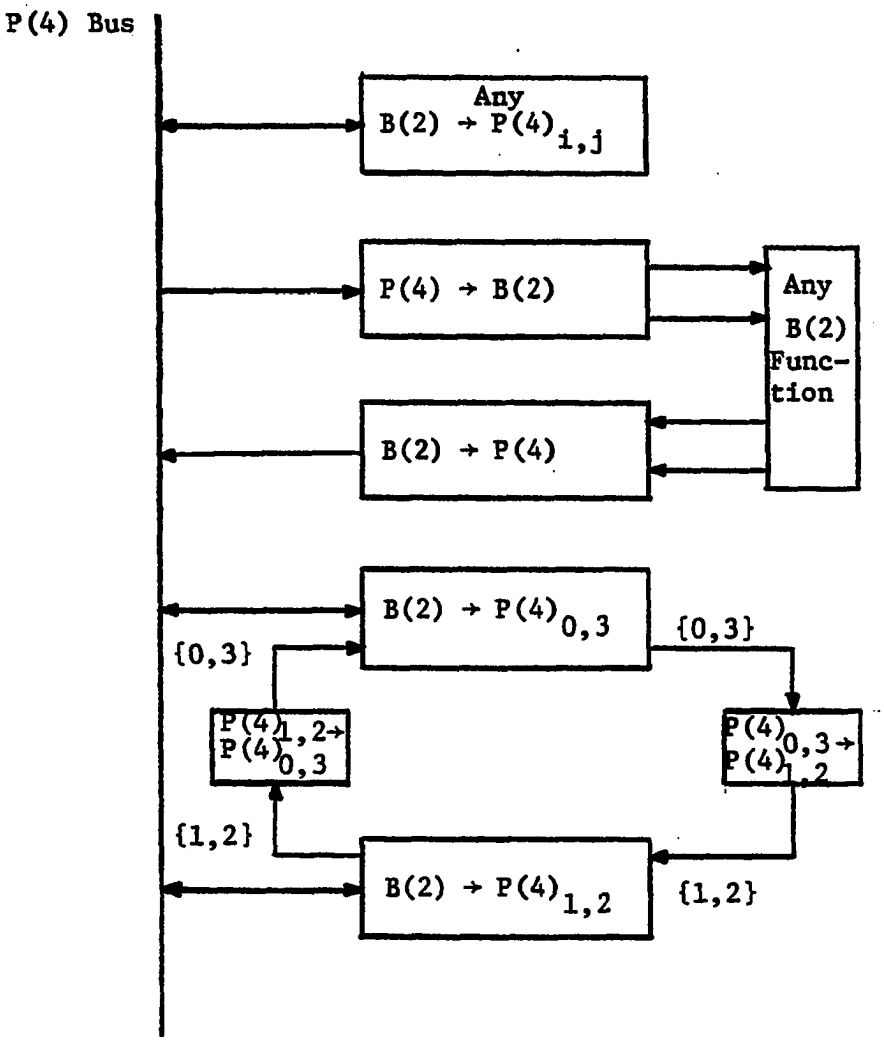
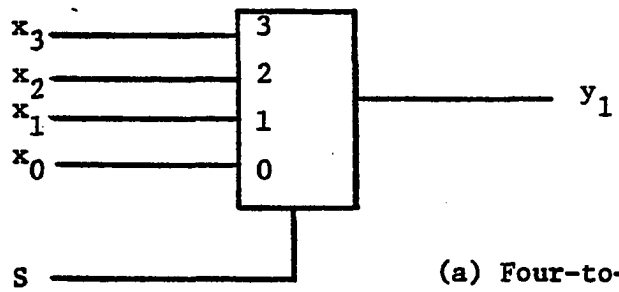


Figure 4.8. Summary of Bus Structure

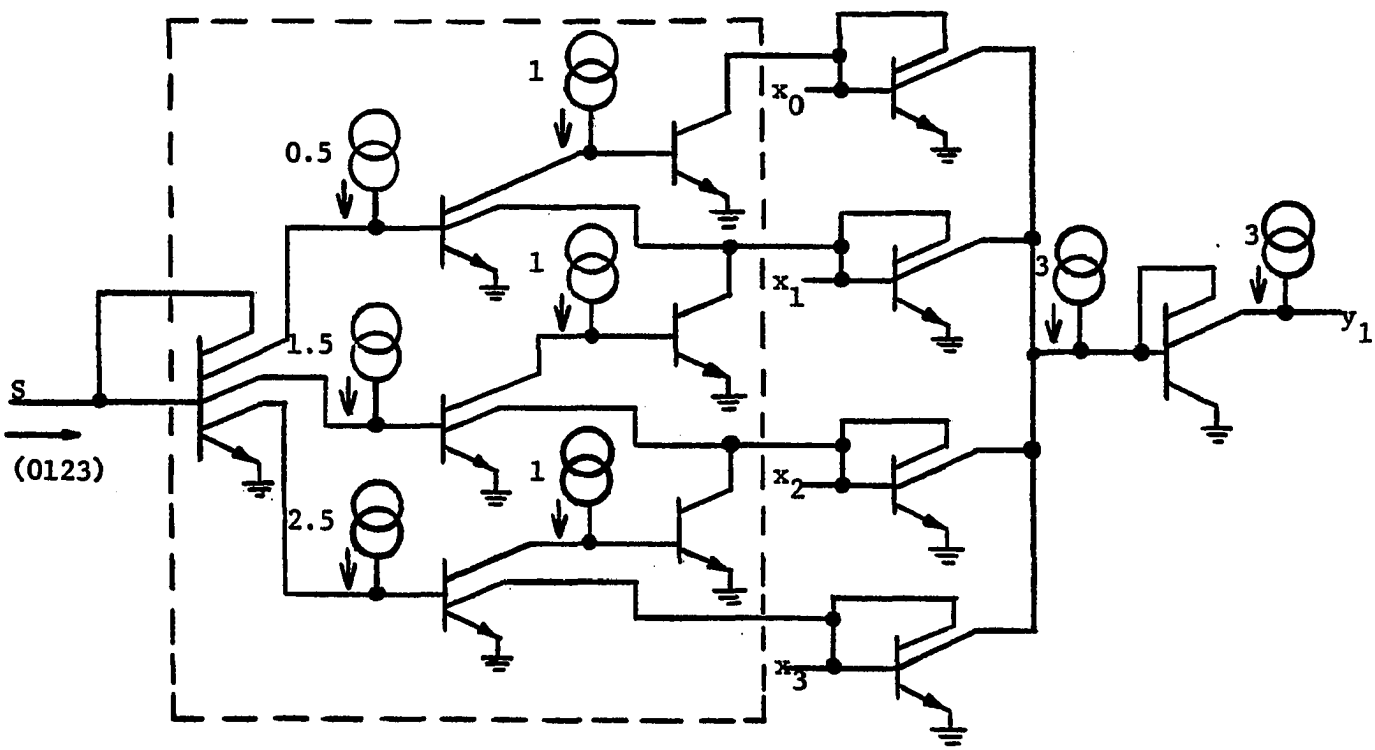
Figure 4.8 is a summary of the interfacing and bus circuitry developed in this section.

A common problem involving bus structures which must be handled concerns directing signals onto and off of the buses. Gating or enabling signals from a control unit tell multiplexing devices how to direct the traffic. The problems of control signals and multiplexing will both be

dealt with separately. A four-valued multiplexer or T-gate has been described in [5]. The T-gate can be used to realize any multivalued function [38]. Shown in Figure 4.9 is the I^2L realization for the T-gate, this circuit can be used to allow four different signals to be gated onto



(a) Four-to-One Multiplexer



(b) I^2L Four-Valued Multiplexer

Figure 4.9. Four-to-One Multiplexer (a) Diagram, (b) Schematic.

a single bus, or it can be used to allow several different registers to talk to a special register such as the accumulator. An enable line can be added to the circuit of Figure 4.9(b), permitting more than one multiplexer to be tied to a bus and still have common S inputs. The circuit shown in Figure 4.10 will cause the T-gate to be disabled for E = 0 and selected for E ≥ 1. This could easily become complemented for design convenience. Reversing the concept, a one-to-four demultiplexer would

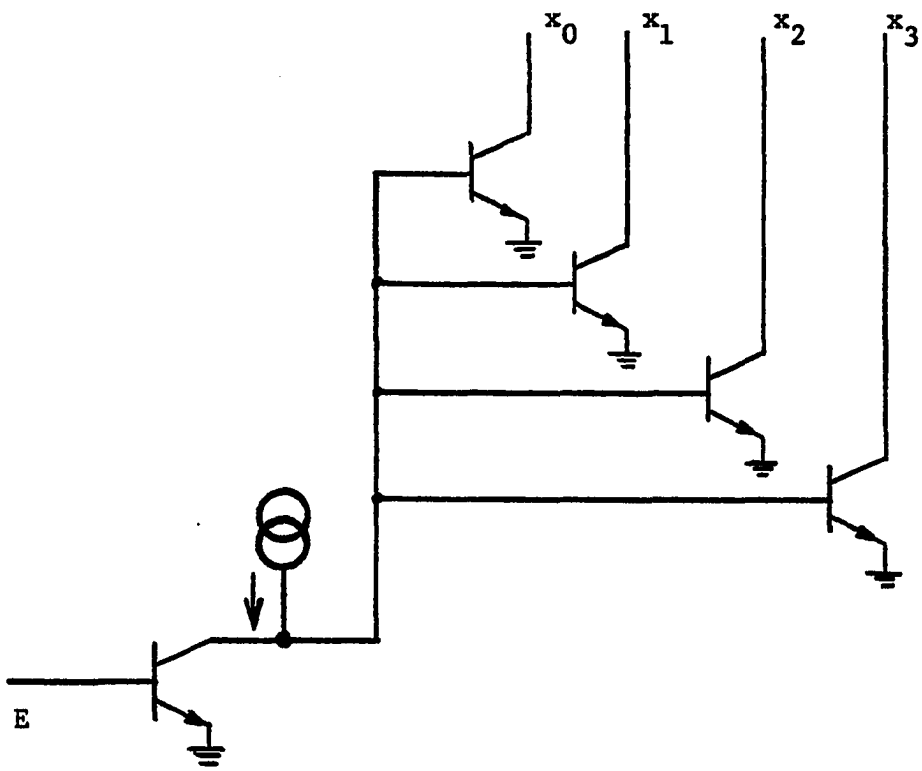
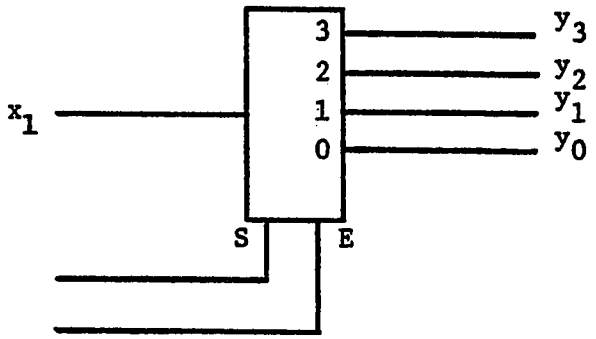


Figure 4.10. T-Gate Select Circuit

allow the bus to fan out to four destinations. In Figure 4.11 is a one-to-four demultiplexer with an enable lead. The threshold detecting circuitry contained within the dashed lines of Figure 4.9(b) is used in this demultiplexer; however, for simplicity only a block diagram is shown in Figure 4.11.

Control Unit

The control unit of any processor often times is a "catchall" in the machine architecture. It is usually responsible for such things as instruction decoding, machine cycle encoding, and various timing and control lines that help enable and disable the major components in the machine. The traditional approach toward this section of a processor is that control functions are inherently binary in nature. That is, a control line is either active or inactive and the thing which is being controlled requires no more. Vranesic has reported on the use of multivalued signalling in daisy chain bus control [39]. His conclusion is that, "It may be wise to look for possibilities of using multivalued circuits in such ways that they improve the performance of binary schemes, but can also revert to the basic binary mode of operation as a default situation." Upon closer examination of the control signals used thus far, both B(2) and P(4) lines have been employed. The B_S and F_S lines defined in Chapter III took on values of zero and three. While this assignment was somewhat arbitrary, the only other choice was logical one and two. The first choice of {0,m-1} seems more natural. Besides the $B(2) \rightarrow P(4)_{0,3}$ mapping offers some real advantages in some of the B(2):P(4) hardware designed. The full adder is a good example. On the other hand, for multiplexing and demultiplexing, the S line is a four-valued control line. The idea of routing more output and input primary signal paths with a single control line is desirable. Clock signals for memory devices might be best handled with either B(2) or P(4) control lines, depending upon the memory type. An edge-triggered flip-flop would use a B(2) control line. A master-slave flip-flop could



(a) One-to-Four Demultiplexer

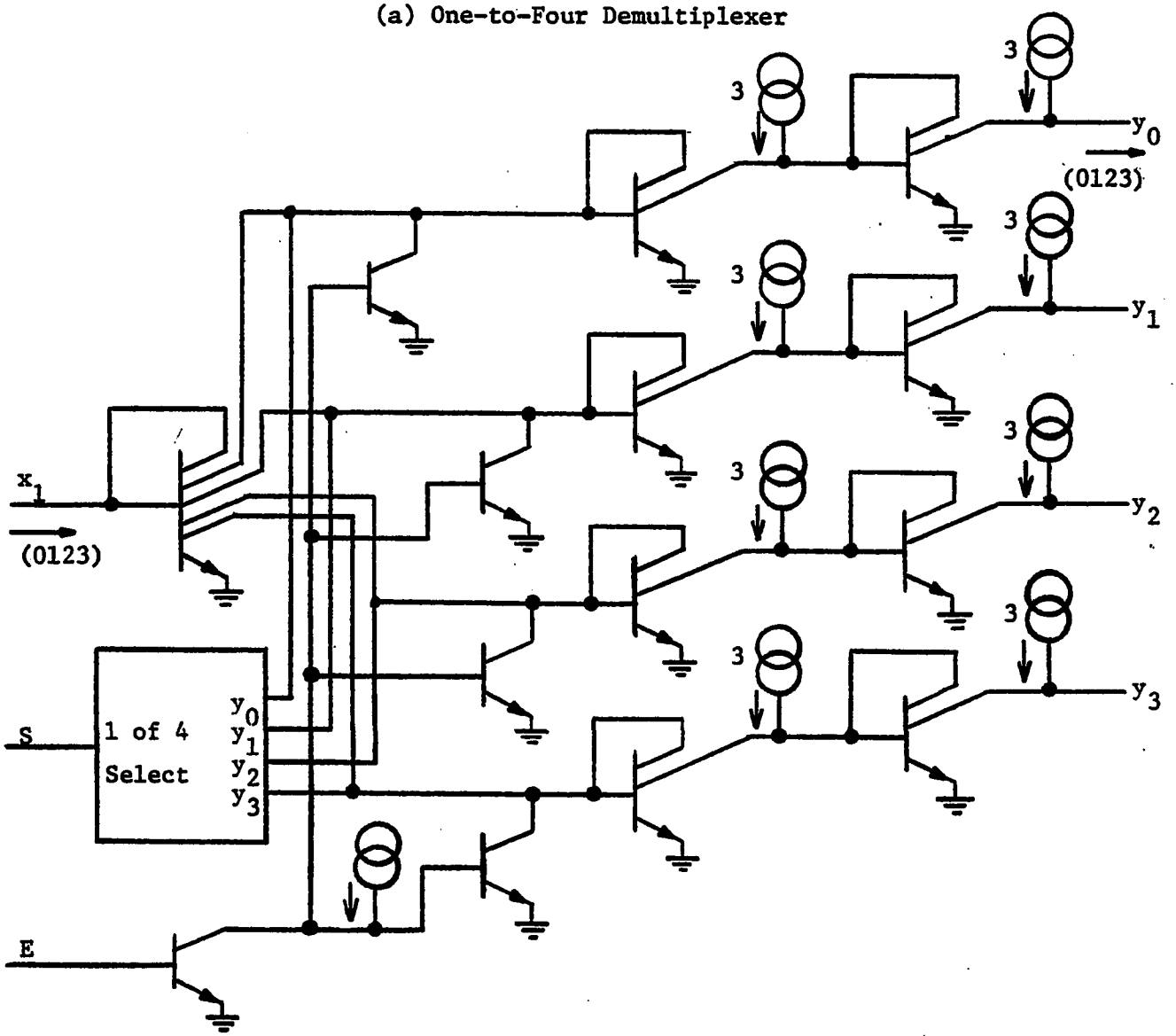


Figure 4.11. One-to-Four Demultiplexer (a) Diagram, (b) Schematic.

take advantage of the P(4) control line to define the four clocking steps which are necessary to provide total input and output isolation. Taking advantage of both the edge and the level in P(4) gives the logic designer six possible triggering mechanisms in a single transition from zero to three as compared to only two for the B(2) case. Many other specific examples can be cited; however the fact is that there are situations in a B(2):P(m) machine where control lines other than binary can make a significant contribution to the architectural improvement.

ALU

The arithmetic logic unit is the last section of the B(2):P(4) processor to be considered here. The dual radix full adder designed in Chapter III is a fundamental part of any ALU. Other functions include the logical operations, shifts both right and left, and circuitry to detect certain flag conditions. Sufficient combinational hardware and design techniques were presented in Chapter III to allow a total B(2):P(4) ALU to be designed and constructed. Kabat and Wojcik have reported on the design of a four-valued ALU [40]. With the penalty of a significant amount of extra logic, binary operations could be simulated with P(4) logic. Their conclusion was that learning base four is not that difficult and it wouldn't be necessary to carry the extra burden if this were done. The B(2):P(4) hardware allows the operator to function at either base. No simulation is involved.

The transition to higher radices is not a trivial problem. A well thought out ALU suitable for B(2):P(4) adaptation is a recondite topic. How much influence should traditional B(2) ALU organization have on higher radix architecture? It is the author's contention that

intuition is not a good guide here. A single set of gates and design methodology might not render an elegant solution. Where the literal operator was an order of magnitude more costly than the "clockwise cycling" operator for the full adder design, it might prove less costly in another area of the ALU. This section of any processor is complicated enough that a good realization could easily be a hybrid one using the best of several gate sets. All of the work in [40] was done with T-gates. Those P(4) circuits could be compared with B(2):P(4) circuits for the same functions to establish a bench mark for future work. First of all, the full adder of Kabat and Wojcik should be modified for B(2):P(4). Then a comparison of the B(2):P(4) full adder, a significant part of any ALU, could be made. Modifying the full adder of Kabat and Wojcik [40] to implement Table 3.6 yields the schematic in Figure 4.12. The entire circuit is constructed using the multiplexer of Figure 4.9. Notice that the S input has become one of the input variables. The B_s line is used to force the two necessary default values for the B(2) case. Each multiplexer requires 23 collectors, 9 mirrors, and 3 switches. Since a total of 13 multiplexers are required for the B(2):P(4) full adder, the circuit complexity is more complicated than the "clockwise cycling" realization and less complicated than the literal realization. Table 4.4 is a hardware comparison at the gate level for each of the B(2):P(4) adders considered. Since all designs have been done at the gate level and many of the circuits proposed have been constructed (Appendix A), to design the entire ALU in this work seems impractical. The statement that a gate level design is non-trivial should have been adequately substantiated based upon the full adder results. However,

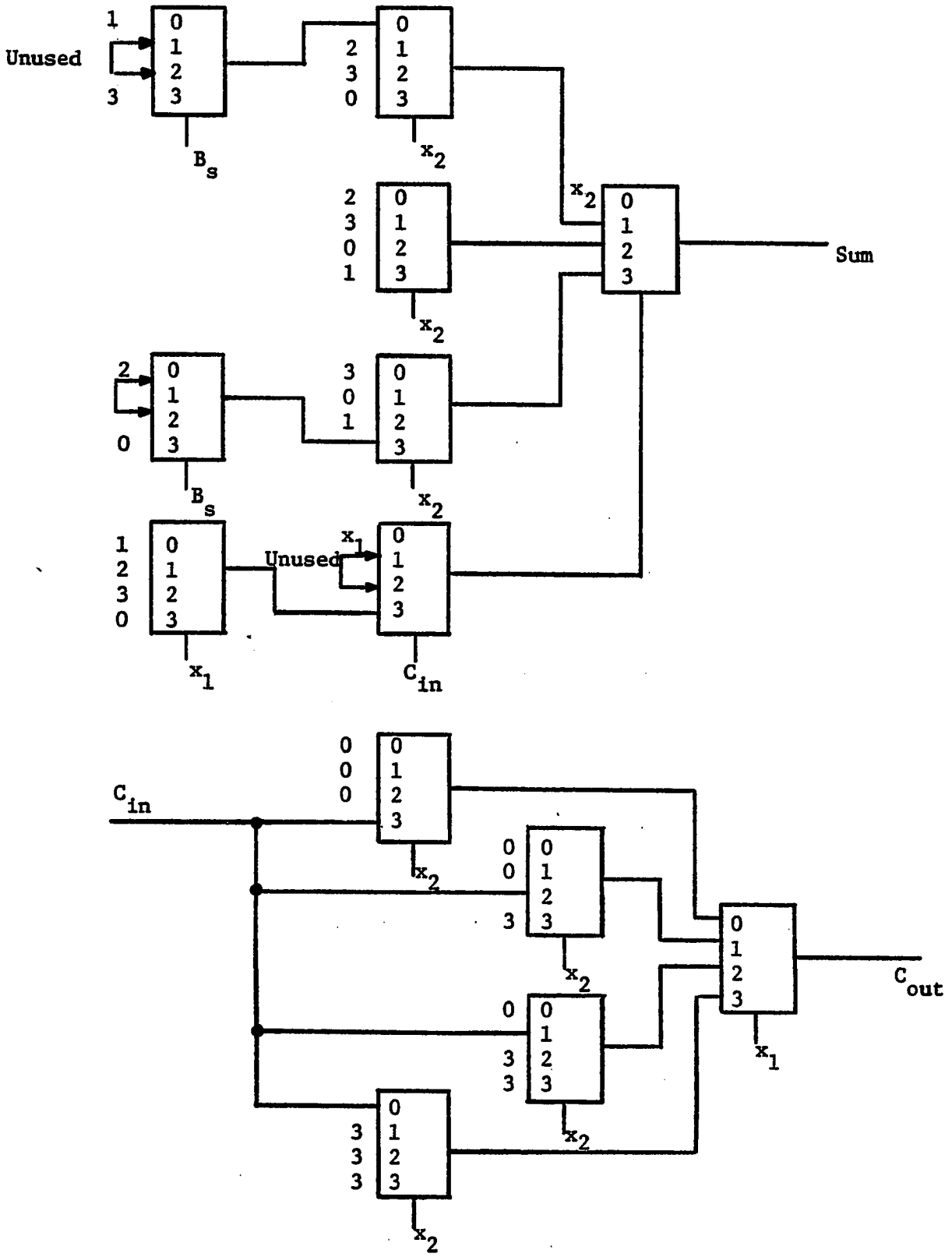


Figure 4.12. B(2):P(4) Full Adder, T-Gate Realization

the fact that an entire gate level realization is not presented here, neither diminishes the author's enthusiasm toward the effort, nor does it lessen the practicality or reality of this section of a processor. The tools have been provided.

Table 4.4. Hardware Comparison for Full Adders

Operator	B(2):P(4)	P(4)	B(2) Burden
"Clockwise Cycling"	166 Collectors	21 Collectors	145 Collectors
	57 Mirrors	7 Mirrors	50 Mirrors
	28 Switches	3 Switches	25 Switches
Literal	>1622 Collectors	>1500 Collectors	122 Collectors
	> 646 Mirrors	> 600 Mirrors	46 Mirrors
	> 324 Switches	> 300 Switches	24 Switches
Multiplexer	299 Collectors	253 Collectors	46 Collectors
	117 Mirrors	99 Mirrors	18 Mirrors
	39 Switches	33 Switches	6 Switches

Total Architecture

Having examined memory elements, bus design, control signals, and a portion of the ALU, it is now possible to briefly comment on total processor architecture. As each of the different sections of a B(2):P(m) machine becomes a hardware reality, the designer is given a greater motivation to consider applications. Some ideas are presented here to encourage the use of MVL and demonstrate potential advantages which MVL architectures offer above conventional B(2) machines. There are the obvious improvements in line count and through put. In the area of

fault diagnosis it would appear that greater resolution could be achieved in designing test vectors and test sets if each possible B(2) machine residing in a P(m) host were allowed to do a separate logical audit. Since the logical values for every B(2) sub machine are disjoint, each diagnosis would contain a different set of test vectors. Another idea suggests the possibility of simultaneous activities by using logic levels not required by the "active" B(2) machine to handle peripheral or secondary requirements. For example, a $B(2) \rightarrow P(4)_{0,3}$ mapping could take advantage of the fact that the logic one and two values occur twice during a single clock cycle. Even though these values have no responsibilities to the B(2) machine, they could be used to report on its welfare by gating self-diagnostic information to the "outside world." Input/output requirements often tax a small machine beyond its real time capabilities. The flexibility of having additional logic levels existing in a fixed relationship with the required levels gives an added dimension to the processor's capability. Software polling loops that are inherently controlled by machine cycle time could execute more quickly by taking advantage of the adjacent pair of logic levels existing in a P(4) machine.

These ideas point out the fact that a B(2) machine operating under a P(m) host is a more powerful architecture than a stand-alone B(2) processor. Upward capability of B(2) machines also presents the designer with greater flexibility and adaptability. Hopefully, the B(2): P(m) concept will encourage a gradual migration of emphasis to MVL circuits, processors, and applications.

Summary

Two types of memory devices previously designed for usage in multiple-value work were examined. Both the set-clear and J-K flip-flop were found to be compatible for both mappings in P(4). Two additional properties for MVL memory devices were defined to insure their ability to operate in any B(2) → $P(m)_{i,j}$. These properties check the memory for mapping compatibility and recovery. The J-K flip-flop is more desirable since there are no non-deterministic next-states.

Next m-valued bus designs were considered. The circuits shown in Figure 4.5 allow each of the mappings in P(4) to communicate. Table 4.3 defines the encoding/decoding scheme required to do a total mapping from P(4) to B(2) and back again. The circuits to do this are shown in Figure 4.6. This allows B(2) components to be interfaced with P(4) circuits. Multiplexers and demultiplexers capable of handling fan-in and fan-out for the bus were shown in Figures 4.9 and 4.11.

The control unit and arithmetic logic unit were both mentioned briefly. There are situations in a B(2):P(m) machine where control lines other than binary can make a significant architectural improvement. An additional P(4) full adder designed by Kabat and Wojcik, using T-gates was modified for B(2):P(4), and is shown in Figure 4.12. The gate level design of an entire B(2):P(4) ALU was not attempted; however, a comparison of three B(2):P(4) full adders was made in Table 4.4.

Finally, a few comments were made with regard to total B(2):P(m) processor architecture. Several potential applications were mentioned. With the greater flexibility offered by the upward compatibility of B(2) machines operating under a P(m) host, hopefully the dual radix concept

provides a synergistic approach to circuit and systems design that is impossible to achieve with stand-alone architecture.

CHAPTER V

CONCLUSIONS

Summary of Results

This dissertation examines the upward compatibility of binary Boolean algebras, B(2), with Post algebras, P(m), $m = 2^N$, N an integer greater than one. Theorem 2.2 states that there are 2^{N-1} homomorphic mappings of B(2) in P(m). Table 2.3 shows the possible mappings. Although much of the work considers the general case for P(m), all of the examples are for P(4). Using Definition 2.5 the relationships for P(4) are $B(2) \rightarrow P(4)_{1,2}$ and $B(2) \rightarrow P(4)_{0,3}$. These are the only two mappings for which the complement holds for B(2). The MAX and MIN functions in P(4) were shown to realize the "OR" and the "AND" functions in B(2) for all six mappings (see Figure 2.1). For these functions the circuitry will operate in either radix, thus the B(2):P(4) hardware places no additional burden on the P(4) realization. The unary operators defined by Vranesic, et al, and by Allen and Givone to obtain functional completeness in P(m) were examined under the dual radix concept. Even though it is not required for these operators to have a function in B(2), the B(2):P(4) circuits were submitted to the bench

marks of 1) all primary signal paths be used for both B(2) and P(4) signals, 2) convenient functional pairing, and 3) minimal B(2) hardware burden. A summary of the functional pairing for all B(2):P(4) circuitry developed in Chapter II is summarized in Table 2.10. All of the circuit designs in this work were done at the gate level using integrated injection logic, I^2L. The actual construction, a detailed analysis of the development, and all of the electrical design considerations for all of the circuits developed in Chapter II are given in Appendix A.

In Chapter III combinational designs for the two algebras were done for B(2):P(4) circuits. Actual design examples helped expand the dual radix concept and the binary select, B_s, and function select, F_s, lines were introduced. Two standard B(2):P(4) circuits were proposed. A "Product of Sums" gate for Vranesic's algebra and a "Sum of Products" gate for Allen and Givone were designed and Table 3.2 is a summary and comparison of the I^2L realizations against the bench marks.

Since no single set of gates will always provide the best circuit realization, and no one design technique will show which operators to use, other unary operators were considered. The Post algebra monotone and disjoint operators were defined. Also, equivalence relationships were given for each of the unary operations.

A B(2):P(4) full adder was designed taking advantage of the "clockwise cycling" operation. The P(4) adder of Figure 3.11 can be modified by the literals shown in Figure 3.12 to accomplish a dual radix design. The same example was solved again using an algorithm for finding the maximum compatible mapping from B(2) to P(m). It is designed to work for both completely and incompletely specified functions. The

algorithm will find all $F_m^*(i,j)$ terms necessary to guarantee that $B(2) \rightarrow P(m)_{i,j}$ exists. Also, it will determine the mapping which requires the minimum number of terms.

Chapter IV examines the necessary components to construct a dual radix machine. The memory elements, bus structure, control unit, and arithmetic logic unit are each dealt with. Two types of memory devices previously designed for usage in MVL work were examined. The set-clear and J-K flip-flop were found to be compatible for both mappings in P(4). To insure that any MVL memory device will operate for all $B(2) \rightarrow P(m)_{i,j}$, two properties were defined. These properties check the memory for mapping compatibility and recovery. Next, m-valued bus designs were done which allow all $P(m)_{i,j}$ machines to communicate. Table 4.3 defines the encoding/decoding scheme required to do a total mapping from P(4) to B(2) and back. The circuits to do this are shown in Figure 4.6. Multiplexers and demultiplexers capable of handling fan-in and fan-out for the bus were shown in Figures 4.9 and 4.11. A comparison of three B(2):P(4) full adders was made in Table 4.4. A few comments were made with regard to total B(2):P(m) architecture and several potential applications were discussed. Synergism is available for the B(2) machine operating under a P(m) host as well as for the dual radix machine.

Suggestion for Further Studies

Clearly, the idea of a dual radix machine and the hardware to support it has been conceived and brought to a practical level of existence. Many of the sophisticated features of a single radix machine need to be examined under the dual radix design concept. Features such

as direct memory access, memory bank switching and memory mapped I/O are interesting topics. Much effort can be expended in the area of selecting the best gate set in the P(m) algebra for realizing a given function. Obviously, any improvements here are similarly improvements for B(2):P(m). More work can be done to explore other unary operators and their binary mappings in an effort to build additional "standard" B(2):P(4) building blocks. If minimization techniques become less important due to the influence of VLSI technology, then algorithms to test a function against several standard building blocks would be in order.

Beyond the hardware, the influence of dual radix processing on the overall processor and the programmer is overwhelming. It's almost as if they can begin again to investigate the computing possibilities presented by the rich set of hardware and design methods made available to them by allowing a B(2) machine to operate under a P(m) host.

LIST OF REFERENCES

1. Vranesic, Z. G., Lee, E. S., and Smith K. C., "A Many-Valued Algebra for Switching Systems," IEEE Trans. on Computers, Vol. C-19, pp. 964-971, 1970.

2. Rine, D. C., Computer Science and Multiple-Valued Logic, North-Holland, 1977.

3. Birkhoff, G., MacLane, S., A Survey of Modern Algebra, MacMillan, 1965.

4. Metze, G., Wojcik, A. S., "An Analysis of Some Relationships Between Post and Boolean Algebras," J. of ACM, Vol. 21, pp. 680-696, Oct. 1974.

5. Dao, T. T., McCluskey, E. J., and Russell, L. K., "Multivalued Integrated Injection Logic," IEEE Trans. on Computers, Vol. C-26, pp. 1233-1241, 1977.

6. Tull, M. P., "A New Method for Realizing Parallel Processing Machines Using Multiple-Valued Logic," Ph.D. Thesis, University of Oklahoma, 1980.

7. Post, E. L., "Introduction to a General Theory of Elementary Propositions," American J. of Math, Vol. 43, pp. 163-185, 1921.

8. Su, S. Y. H., Sarris, A. A., "The Relationship between Multivalued Switching Algebra and Boolean Algebra Under Different Definitions of the Complement," IEEE Trans. on Computers, pp. 479-485, May 1972.

9. Lee, S. C., "Vector Boolean Algebra and Calculus," IEEE Trans. on Computers, Vol. C-25, Sept 1976.

10. Allen, C. M., Givone, D. D., "A Minimization Technique for Multiple-Valued Logic Systems," IEEE Trans. on Computers, Vol. C-17, pp. 182-184, Feb. 1968.

11. Lee, S. C., Modern Switching Theory and Digital Design, Prentice-Hall, New Jersey, p. 19, 1978.

12. Millman, J., Micro-Electronics, McGraw-Hill New York, 1979.

13. MCS 85 User's Manual, Intel Corporation, 1977.

14. Berger, H. H. and Wiedmann, S. K., "Merged-Transistor Logic (MTL), a Low-Cost Bipolar Logic Concept," IEEE J. Solid State Circuits, pp. 340-346, Oct. 1972.

15. Linear and Digital Semi-Custom IC Design Programs, EXAR Integrated Systems, Inc., Sunnyvale, CA., 1980.

16. Nutter, R. S., Rine, D. C. and Swartwout, R. E., "Equivalence and Transformations for Post Multivalued Algebras," IEEE Trans. on Computers, Vol. C-23, pp. 294-299, March 1974.

17. Rosenbloom, P. C., "Post Algebras I Postulates and General Theory," Amer. J. Math., Vol. 64, pp. 167-188, 1942.

18. McCluskey, E. J., "Logic Design of Multivalued I^2L Logic Circuits," IEEE Trans. on Computers, Vol. C-28, pp. 546-559, August 1979.

19. Givone, D. D. and Tront, J. G., "A Design for Multiple-Valued Logic Gates Based on MESFET'S," IEEE Transactions on Computers, Vol. C-28, pp. 854-862, Nov. 1979.

20. Pugsley, J. H. and Silio, C. B., "Some I^2L Circuits for Multiple-Valued Logic," Proc. Eighth International Symposium on Multiple-Valued Logic, Chicago, Ill., 1978, pp. 23-31.

21. Etiemble, D., "TTL Circuits for a 4-Valued Bus," Proc. Eighth International Symposium on Multiple-Valued Logic, Chicago, Ill., 1978, pp. 7-13.

22. Smith, K. C., "Circuits for Multivalued Logic - A Retrospective and Prospective," Presentation ISMVL-80, Northwestern University, June 3-5, 1980.

23. Vranesic, Z. G., "Practical Implementation Prospects for MVL in 1980's," Presentation ISMVL-80 Northwestern University, June 3-5, 1980.

24. Bahraman, A., Chang, S. Y., Romeo, D. E. and Schuegraf, K. K., "Schottky-Base I^2L: A High-Performance LSI Technology," IEEE J. of Solid State Circuits, Vol. SC-14, pp. 578-584, 1979.

25. Dao, T. T., "Threshold I^2L and its Applications to Binary Symmetric Functions and Multivalued Logic," IEEE Journal of Solid State Circuits, Vol. SC-12, No. 5., pp. 463-472, October 1977.

26. Vranesic, Z. G., "A Multi-Valued Switching Theory," Ph.D. Dissertation, University of Toronto, Ontario, Canada, April 1968.

27. The TTL Data Book, Texas Instruments, 1976.

28. Smith, W. R. III, "Minimization of Multi-Valued Functions," Proc. 1974 International Symposium on Multiple-Valued Logic, Morgantown, W. Va., 1974.

29. Smith, W. R. III, "Some Algebraic Methods and Minimization Techniques for Multi-Valued Lattice Logics," Proc. 1972 Symp. on Theory and Applications of Multiple-Valued Logic Design, Suny, Buffalo, N.Y., 1972, pp. 163-174.

30. Rine, D. C., Computer Science and Multiple-Valued Logic, North-Holland, 1977, pp. 221-261.

31. List, Bernard, "Impact of Semiconductors on the Mini/Microcomputer Business," Keynote Address at The 1979 International Micro and Minicomputer Conference, Houston, Texas, November 14-16, 1979.

32. Nagle, H. T. and Irving, T. A., "An Approach to Multi-Valued Sequential Logic," Proc. 1973 International Symposium on Multiple-Valued Logic, Univ. of Toronto, pp. 89-105.

33. Wojcik, A. S., "Multi-Valued Asynchronous Sequential Circuits," Proc. 1974 International Symposium on Multiple-Valued Logic, West Virginia Univ., pp. 401-411.

34. Sheafor, S. J., The Design of Multiple-Valued Asynchronous Sequential Circuits, Ph.D. Thesis, Univ. of Illinois, Urbana, 1974.

35. Wojcik, A. S., "Multi-Valued Asynchronous Circuits," Proc. 1973 International Symposium on Multiple-Valued Logic, Univ. of Toronto, pp. 217-227.

36. Wills, M. S., "A Behavior Model and Triggering Modes for MVL R-Flops," Proc. 1978 International Symposium on Multiple-Valued Logic, Chicago, Ill., pp. 226-234.

37. Edwards, C. R., "I^2L Threshold Circuits for Binary-Quaternary Encoding and Decoding," Research Note, University of Bath, Nov. 17, 1977.

38. Higuchi, T. and Kameyama, M., "Ternary Logic Systems Based on T-Gate," Proc. 1975 International Symposium on Multiple-Valued Logic, Indiana Univ., pp. 290-304.

39. Vranesic, Z. G., "Multivalued Signalling in Daisy Chain Bus Control," Proc. 1979 International Symposium on Multiple-Valued Logic, Bath, England, pp. 14-18.

40. Kabat, W. C. and Wojcik, A. S., "On the Design of 4-Valued Digital Systems," Proc. 1980 International Symposium on Multiple-Valued Logic, Evanston, Ill., pp. 153-170.

APPENDIX A

DUAL RADIX I^2L CIRCUIT CONSIDERATIONS

Introduction

Actual construction of each of the circuits designed in previous chapters will be done in this section. Due to the nature of integrated injection logic, it is not possible to realize the basic I^2L gate shown in Figure 2.7 with discrete devices. Matching the device characteristics of the output section is not likely. Therefore, the design of multi-valued gates must proceed with available I^2L circuitry.

The design will be done with integrated circuits offered by EXAR integrated systems in their I^2L custom IC design kit [15]. Several conventional monolithic I^2L circuits are offered as basic building blocks. Included in these are I^2L inverter arrays and PNP/NPN transistor arrays which have electrical characteristics compatible with the I^2L circuits. Any circuit implementaion which is done using these integrated circuits can be transferred to a VLSI master slice. The master slice has a high packaging density of I^2L gate structures which can be customized to realize the design done at the building block level using three masking steps. The first mask is used to define the collectors of the I^2L gates.

The second mask opens the contact windows on the chip. The third mask defines the desired metal interconnection pattern.

Electrical Characteristics

The typical electrical characteristics of I^2L gates in the custom design kit are shown in Table A.1 [15]. One of the first choices in designing with I^2L circuits, whether for binary circuits or higher radices, is the injector current amplitude. In looking at Table A.1, it can be seen that as the injector current increases the propagation delay decreases. All of the other parameters remain fairly constant as injector current increases, compared to the rapid decrease in propagation delay. Shown in Figure A.1, is the basic I^2L gate which was originally discussed in Figure 2.7. The fan-out capability of the I^2L gate is a measure of its ability to sink output current, I_o. The loop gain must be specified for each of the gate outputs. Obviously a minimum loop gain of unity is required for a fan out of one. Figure A.2 is a typical curve of loop gain versus injector current for a five output I^2L gate.

The loop gain of the basic I^2L gate is a problem for the designer when using the device in multiple value logic designs. As shown in Figure A.2, the product of α and β is low. The common base current gain of the PNP transistor is low because this stage is operated near saturation. Also, the common emitter gain of the output state is small since the device is operating in the inverse mode. The base current, I_i, is shared among the multiple outputs; therefore, the loop gain varies inversely with the number of outputs. If it is assumed that the output current of each collector is the same, then the following equations hold:

I_T = The total output sink current of a five output I^2L gate

I_{c1} = The output sink current of collector one and $I_{c1} = I_{c2} = I_{c3} = I_{c4} = I_{c5} = I_o$

$I_T = I_{c1} \oplus I_{c2} \oplus I_{c3} \oplus I_{c4} \oplus I_{c5}$

$I_T = 5I_o$

$I_T = NI_o$, where N = the fan out of the output stage

Therefore since

$$I_o = \alpha\beta I_j \quad \text{and} \quad I_o = I_T/N$$

Then $I_T = N\alpha\beta I_j$ or

$$\frac{I_T}{I_j} = N\alpha\beta$$

For an I^2L gate with five outputs, the loop gain must be greater than five or the current gain will not be adequate to prevent signal deterioration. In looking at Figure A.2, one can see that for the output closest to the injector one should operate between 1μA and 100μA of injector current. At low levels of injector current the loop gain falls off due to carrier recombination and injection efficiency. At high currents, debiasing of the output stage emitter occurs due to base resistance [15].

A second problem which affects the ability of I^2L gates to be used in MVL applications is the decrease in loop gain as the distance of the output collector from the injector increases. As can be seen in Figure A.2, this loop gain drops off rapidly as the injector current

Table A.1. Typical Electrical Characteristics of Five-Output I^2L Gates.
(Note: Output Characteristics refer to each output.)

Parameter	Typical Characteristics at Various Injector Currents			
	I_j = 100nA	I_j = 1μA	I_j = 10μA	I_j = 100μA
Output Sink Current, I_O	300nA	4μA	40μA	350μA
Output Sat. Voltage, V_{OL}	3mV	3mV	4mV	10mV
Input Threshold	0.48mV	0.54mV	0.60mV	0.66mV
Pwr.-Delay Product (V^+=1 V)	0.6pJ	0.6pJ	1.0pJ	3pJ
Average Prop. Delay	6μsec	0.6μsec	100nsec	50nsec
Max. Toggle Freq. (D F/F)	6kHz	60kHz	400kHz	2MHz
Input OFF Current (V_{IN}=0)	150nA	1.5μA	15μA	130μA
Output Breakdown Voltage	3V	3V	3V	3V

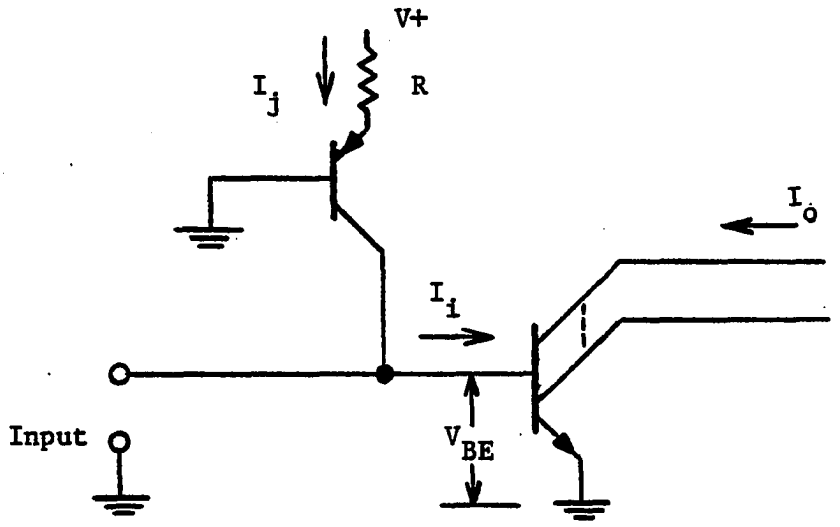


Figure A.1. Basic I^2L Gate

Given:

α = Common base current gain of the PNP transistor

β = Common emitter current gain of the NPN transistor

I_j = Injector current

I_o = Maximum output sink current of a single collector

I_i = Input base current of the output stage

R = External resistor for injector biasing

V_{BE} = Base to emitter voltage of the output stage

$$I_j = \frac{V^+ - V_{BE}}{R}$$

$$I_i = \alpha I_j$$

$$I_o = \beta\alpha I_j$$

$$\frac{I_o}{I_j} = \beta\alpha \text{ ; the loop gain of the } I^2L \text{ gate}$$

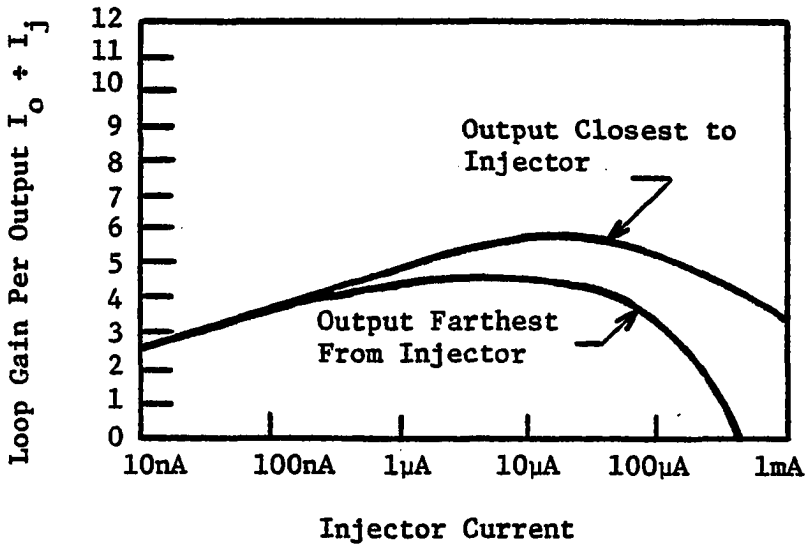


Figure A.2. Loop Gain Versus Injector Current for a Five Output I^2L Gate

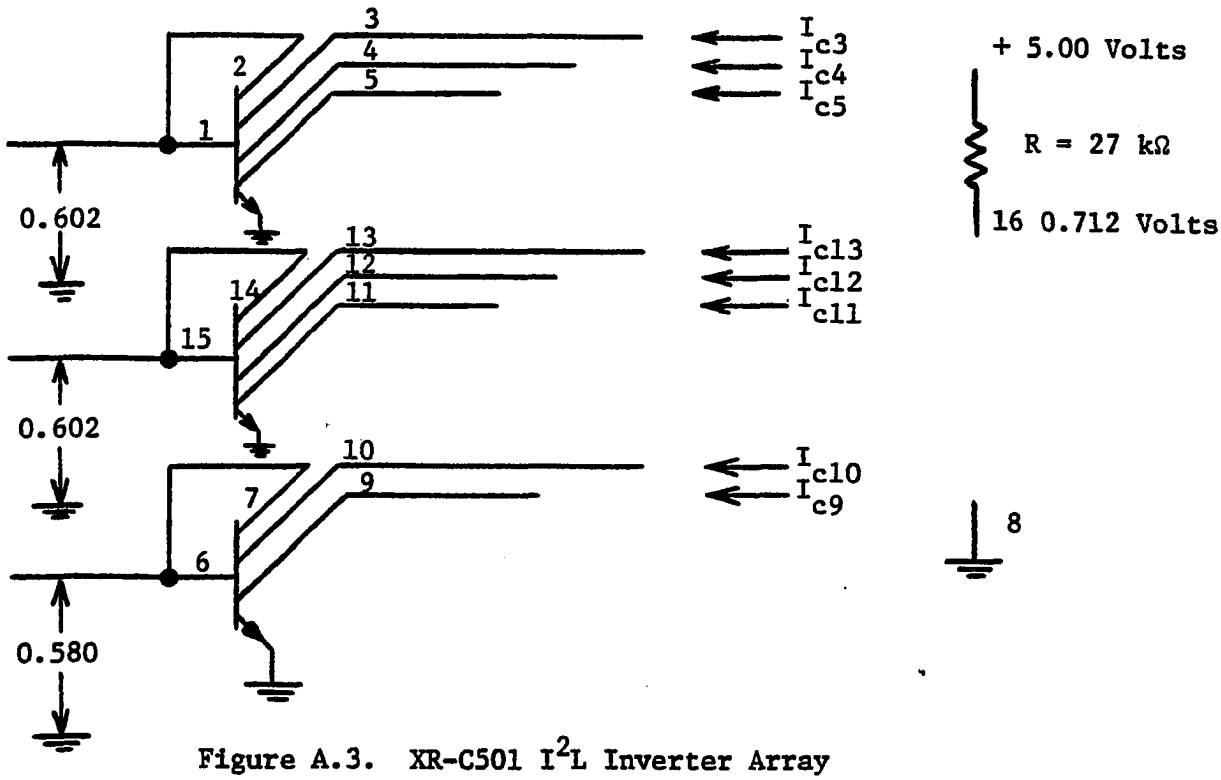


Figure A.3. XR-C501 I^2L Inverter Array

goes beyond 100μA. When working with the individual integrated circuits, the loop gain as a function of separation can be easily measured. Table A.2 shows the circuit response of the XR-C501 I^2L inverter array. The array was connected as shown in Figure A.3.

Table A.2. XR-C501 Circuit Response

Output Collector	Sink Current
I_{c3}	14.35μA
I_{c4}	14.33μA
I_{c5}	14.32μA
I_{c13}	14.39μA
I_{c12}	14.39μA
I_{c11}	14.35μA
I_{c10}	16.40μA
I_{c9}	15.03μA

Pin 16 of the XR-C501 is connected to +5.00 volts through a 27.6 K Ω resistor. This external resistor is used to establish the injector current amplitude for all three of the inverter arrays on the integrated circuit. Since the base emitter voltage of the output stage is directly proportional to I_i, it can be seen that the base current decreases as the fan out decreases. For the XR-C501, the average injector current is equal to 1/4 of the total injector current applied to Pin 16 [15]. Thus, $I_i = \frac{156}{4} = 39\mu A$ for the inverter array in Figure A.3. Obviously, the empirical data of Table A.2 does not agree with this rough approximation. Since the arrays are connected as current mirrors, the output current

should equal I_i. Thus, to accurately adjust I_i on the XR-C501, the stage should be connected as a current mirror and the adjustment can be made measuring the output current. The actual selection of the value of R when designing with gate arrays at the master slice level will be discussed in much greater detail in the specific design offered later in this appendix. In looking at the sink current amplitudes in Table A.2 it can be seen that the lowest individual collector current is 13% smaller than the highest. This points up the fact that the loop gain is smaller for the higher fan out array. Also, it can be seen that the outputs on Pin 3, 13, and 10 are physically closest to the injector.

The information in Table A.2 is important in selecting the operating point for MVL. In binary applications the sensitivity of loop gain to fan out and physical separation is not as critical. Also, there are ways of correcting the inaccuracy of the mirror due to the low gain. One way proposed by Dao [25] is to undersize the feedback collector with respect to the others. This is not achievable with master slice layouts. In prototyping MVL circuitry using the XR-C501 inverter array, close attention will be given to the loop gain problem and how it affects the final master slice layout.

In selecting the proper values of injector currents, pull-up resistors, and operating voltage for the dual radix circuits developed in Chapter II, one must accurately specify the electrical characteristics of the current mirror. Since the basic I^2L gate has design limitations as were just discussed, any practical design must begin by quantifying these limitations. It was shown in Table A.2 that the current mirror with the smaller fan out offered the better gain. All of the dual radix

circuitry developed in Chapter II can be constructed using this array on the XR-C501. In transferring this construction to the master slice technology no problem exists even though the gate array is comprised of devices with a fan out of five. If a higher loop gain is required, thus demanding a lower fan out which is alright, then the emitters on the output stage may be opened with little difficulty.

The first choice to be made is the operating voltages. It would be nice if any of the I^2L gates could interface with TTL logic since it has the greatest usage as a logic family. Therefore, a +5.0 volt supply for the injectors would be appropriate. However, the maximum breakdown voltage from collector to emitter is 2.5 volts. A second supply is required for external collector pull-ups if voltage interfacing is desirable. A voltage of +2.1 volts will be used for the second supply. If 100 kΩ resistors are used for collector pull-ups, the $I_c max = 21\mu A$. In B(2):P(4) circuitry this would be equivalent to a logical three. Table A.3 shows the ideal values of currents and voltages for the voltages and resistance chosen for B(2):P(4) circuits. Obviously, these values

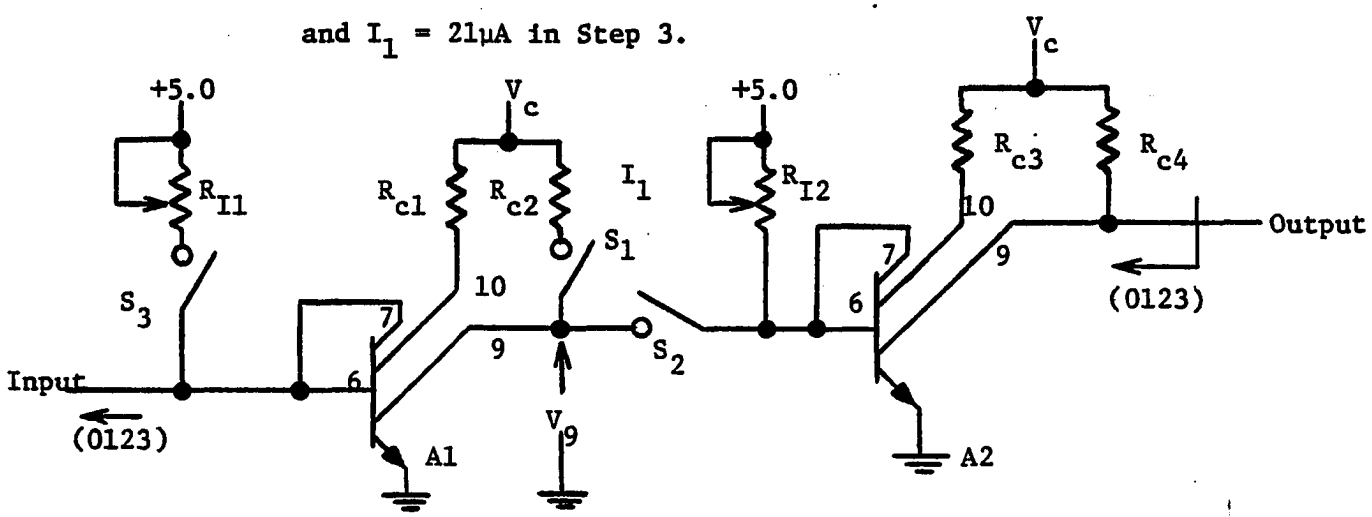
Table A.3. Ideal Voltages and Currents for B(2):P(4) Circuitry

V_i = +5.0 volts , V_c = +2.1 volts , R_c = 100 kΩ

Logic Value	Current	Voltage
0	0μA	0 Volts
1	7μA	0.700 Volts
2	14μA	1.400 Volts
3	21μA	2.100 Volts

cannot be obtained due to current mirror inaccuracies. The circuit in Figure A.4 is capable of measuring the ability of the current mirror to accurately propagate logical values. Gate arrays A1 and A2 are wired as inverters. The following procedure can be used to measure any deterioration between the input and output.

1. With $S_1 = S_3$ = Closed and S_2 = Open
2. Adjust R_{I2} so I out = 21μA (V out ≤ 0.500 volts)
3. Adjust R_{I1} so $I_1 = 0$ (V_9 = + 2.1 volts)
4. With S_1 = Open and $S_2 = S_3$ = Closed
5. Measure I out and V out
6. Repeat steps 1, 3, 4, and 5 with I_1 = 7μA, I_1 = 14μA, and I_1 = 21μA in Step 3.



$R_{c1} = R_{c2} = R_{c3} = R_{c4}$ = 100 kΩ
$R_{Ij} = R_{I2}$ = 500 kΩ
V_c = + 2.1 volts

$R_{c1} = R_{c2} = R_{c3} = R_{c4}$ = 267 kΩ
$R_{I1} = R_{I2}$ = 1mΩ
V_c = + 2.4 volts

Figure A.4. Current Mirror, Test Circuit

The above procedure can be modified slightly to allow a better match between the A1 and A2 array.

1. With $S_1 = S_3$ = Closed and S_2 = Open
2. Adjust R_{I1} so I_1 = 14μA (V_9 = 0.700 volts)
3. With S_1 = Open and $S_2 = S_3$ = Closed
4. Adjust R_{I2} so I out = 7μA (V out = 1.400 volts)
5. With $S_1 = S_3$ = Closed and S_2 = Open
6. Adjust R_{I1} so I_1 = 0μA (V_9 = + 2.1 volts)
7. With S_1 = Open and $S_2 = S_3$ = Closed
8. Measure I out and V out
9. Repeat Steps 5 to 8 with I_1 = 7μA and I_1 = 21μA in Step 6.

The data collected using the modified procedure is given in Table A.4. In looking at the voltages it can be seen that the actual values differ from the ideal values considerably. The greatest differences are due to

Table A.4. Current Mirror Accuracy Measurements for V_c = +2.1 volts, R_c = 100 kΩ

INPUT ARRAY A1		OUTPUT ARRAY A2	
V_9	V_{10}	V_9	V_{10}
0.534	0.200	2.080	2.080
0.910	0.700	1.584	1.400
1.465	1.400	0.858	0.530
2.090	2.090	0.415	0.098

Value Set Point for R_{I2} ☐

the gain variations of the two output collectors. Obviously C_{10} is physically closer to the injector. The output voltage associated with a

logical three is not a problem depending on how the tolerances of each of the values are defined. If the criteria for acceptable logic values is as defined in Table A.5, then there is a minimum of 200mV or 2μA of noise immunity between logic bands. The bands established in Table A.5 will allow the measured values of Table A.3 to become acceptable logic levels for B(2):P(4) circuitry. The gain variations are more pronounced at higher levels of collector current. It was shown in Figure A.2 that the gain of the collector furthest from the injector falls off more rapidly than the closer ones. The collector current may be reduced by

Table A.5. B(2):P(4) Logic Values for V_c = +2.1 Volts and R_c = 100 kΩ

BAND	TYPICAL	TOLERANCE
$0 \leq I_0 \leq 3\mu A$	2μA	+1μA −2μA
$5\mu A \leq I_1 \leq 9\mu A$	7μA	±2μA
$12\mu A \leq I_2 \leq 16\mu A$	14μA	±2μA
$18\mu A \leq I_3 \leq 21\mu A$	19μA	+2μA −1μA

increasing R_c. This would eliminate the possibility of greater gain variations at higher currents. At higher values of R_c, the leakage current which is approximately 0.2μA, cannot be ignored. Also, the propagation delay will necessarily go up.

If a second set of values for V_c and R_c are chosen, there is a possibility of improving the width of the logic bands and noise immunity by decreasing the sensitivity to gain variations. In Figure A.4 set the

value of V_c to +2.4 volts and R_c to 267 kΩ. Table A.6 shows the ideal values of currents and voltages for these new parameters.

Table A.6. Ideal Voltages and Currents for B(2):P(4) Circuitry

V_i = +5.0 volts , V_c = +2.4 volts , R_c = 267 kΩ

Logic Value	Current	Voltage
0	0μA	0 Volts
1	3μA	0.8 Volts
2	6μA	1.6 Volts
3	9μA	2.4 Volts

Table A.7 shows the current mirror inaccuracies after following the same modified procedure for gathering the data in Table A.4. If the bands shown in Table A.8 are established for each logic value, the measured

Table A.7. Current Mirror Accuracy Measurements

for V_c = +2.4 Volts , R_c = 267 kΩ

INPUT ARRAY A1		OUTPUT ARRAY A2	
V_9	V_{10}	V_9	V_{10}
0.592	0.400	2.210	2.200
0.931	0.800	1.616	1.504
1.638	1.600	0.876	0.700
2.330	2.340	0.455	0.264

values in Table A.7 are acceptable for B(2):P(4) circuitry. Each of the circuits developed in Chapter II will be built using the XR-C501 inverter array and the XR-C506 I^2L compatible NPN transistor array.

Table A.8. B(2):P(4) Logic Values for V_c = +2.4 Volts and R_c = 267 kΩ

Band	Typical	Tolerance
$0 \leq I_0 \leq 2.2\mu A$	1.5μA	+0.7μA -1.5μA
$2.7\mu A \leq I_1 \leq 3.7\mu A$	3.2μA	±0.5μA
$5.7\mu A \leq I_2 \leq 6.3\mu A$	6.0μA	±0.3μA
$7.5\mu A \leq I_3 \leq 8.9\mu A$	8.2μA	±0.7μA

Dual Radix Circuitry

Each of the circuits will be checked by generating logic levels from zero to three using the circuit shown in Figure A.5. The typical values in Table A.8 will become the $(x_1,\ldots x_n)$ input currents for the circuit under test. The output values $(y_1,\ldots y_n)$ will then be given in the truth table for that function. The circuit in Figure A.5 will source each logic current value depending upon the position of switch S_1.

Testing the complement gate of Figure 2.9 yields the values given in Table A.9. The measured currents meet the logic bands and

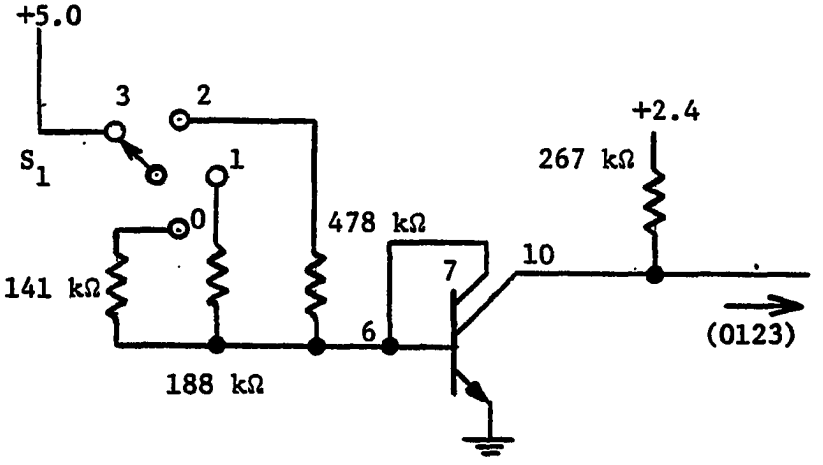


Figure A.5. Logic Generator for Values of 0 to m-1

Table A.9. B(2):P(4) Logic Values for the Complement Gate

x_1	y_1
0.2μA	8.0μA
3.0μA	6.3μA
6.0μA	3.2μA
8.0μA	0.2μA

typical values established in Table A.8. The schematic in Figure A.6 shows the complement gate, with necessary resistor values, constructed from an XR-C501 inverter array. The 2.39 mΩ resistor helps to bias the current mirror to allow for inverter inefficiency on the integrated circuit.

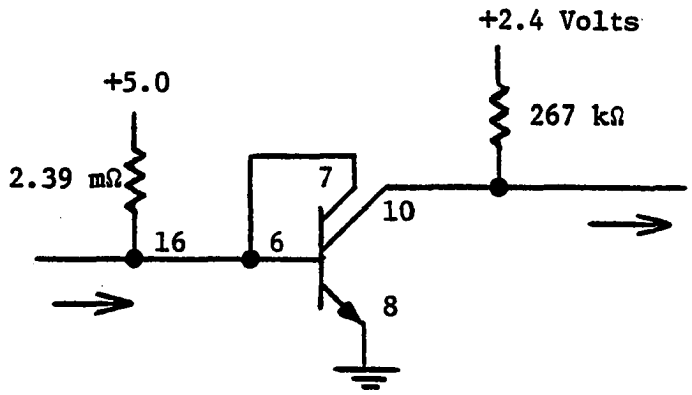


Figure A.6. Complement Gate Construction

The MAX/OR gate shown in Figure 2.10 is redrawn in Figure A.7 showing actual values. Table A.10 gives the measured output currents obtained when exciting the circuit with two of the logic generators of Figure A.5. All values are within the logic bands of Table A.8 except the two noted. A quantizer on the output signal would allow the circuit to be constructed as shown. The two currents which are out of band

could be brought into limits by adjusting the values of the 1.6 mΩ resistors. These resistors set the quiescent operating point of the current mirror making allowances for injector inefficiencies as was done

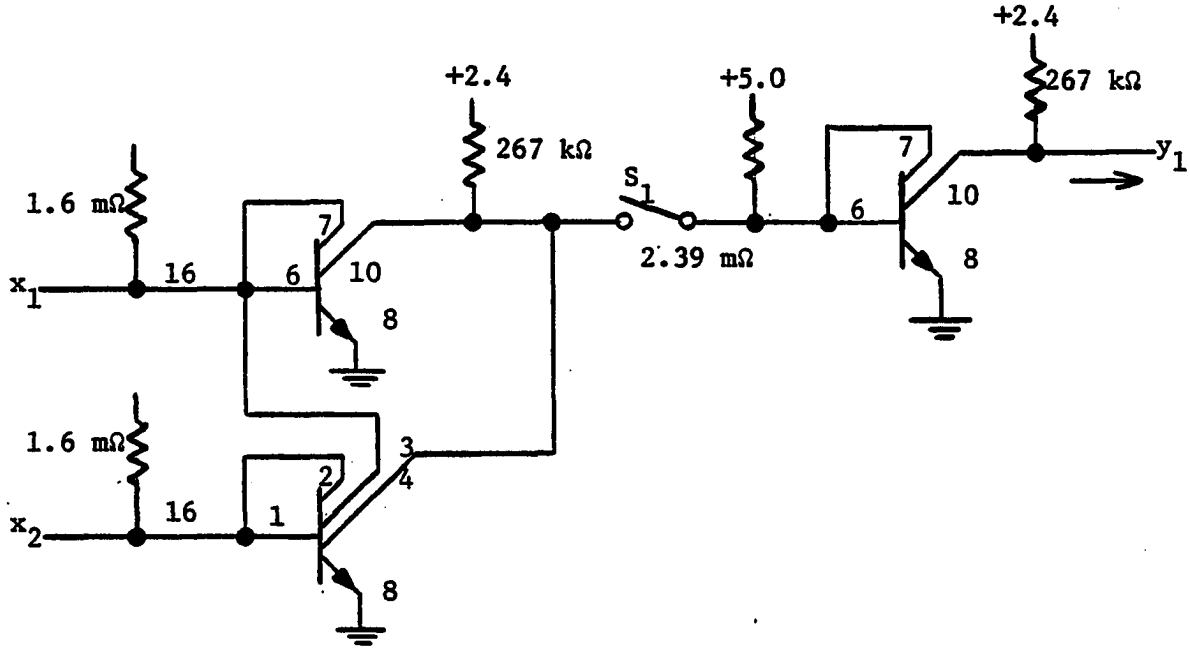


Figure A.7. MAX/OR Gate Construction

Table A.10. B(2):P(4) Logic Values for MAX/OR Gate

x_1 \ x_2	0	1	2	3
0	2.4*	3.3	6.0	7.9
1	3.2	3.5	6.0	7.9
2	5.8	6.0	6.7*	7.9
3	7.5	7.7	8.3	8.6

Output, y_1 in μAmps

*2.4 μAmps is 9.0% out of the band given in Table A.8.

6.7 μAmps is 6.3% out of the band given in Table A.8.

with the 2.39 mΩ resistor on the complement gate. The resistors used in constructing the MAX/OR gate had a 5% tolerance. The resistors could have been 1% tolerance; however, in a way, the resistor value foreshadows the real problem. Variation in current mirrors from device to device (different XR-C501 inverter arrays) requires different resistor values for setting injector levels. This is a phenomena which will not exist when working with the master slice. Also, the injector resistor can be either on or off of the actual master slice layout.

If switch S_1 in Figure A.7 is opened and the output is taken at this point, then by inverting the x_1 and x_2 inputs the data for the MIN/AND gate may be taken. Table A.11 gives the measured current values for the MIN/AND gate. Once again, two of the measured currents are slightly out of the logic bands of Table A.8. For the same reasons previously stated this does not present a serious problem.

The unary "inverter" of Figure 2.14 is shown as it was actually constructed in Figure A.8. Comparing the values of injector resistors to those in Figure A.7 will demonstrate the wide range in electrical characteristics from device to device. Table A.12 is the actual measured output currents for all possible input combinations. All of the currents are within the logic bands of Table A.8. The low current values and zero readings occur due to the switching action of the XR-C506 NPN compatible transistor.

Table A.11. B(2):P(4) Logic Values for the MIN/AND Gate

x_1 \ x_2	0	1	2	3
0	1.3	1.6	2.0	2.2
1	1.9	2.8	3.3	3.6
2	1.9	3.3	6.3	6.9*
3	1.9	3.3	6.6*	8.6

Output , y_1 in μAmps

*6.9 μAmps in 9.5% out of the band given in Table 8.

6.6 μAmps is 4.7% out of the band given in Table 8.

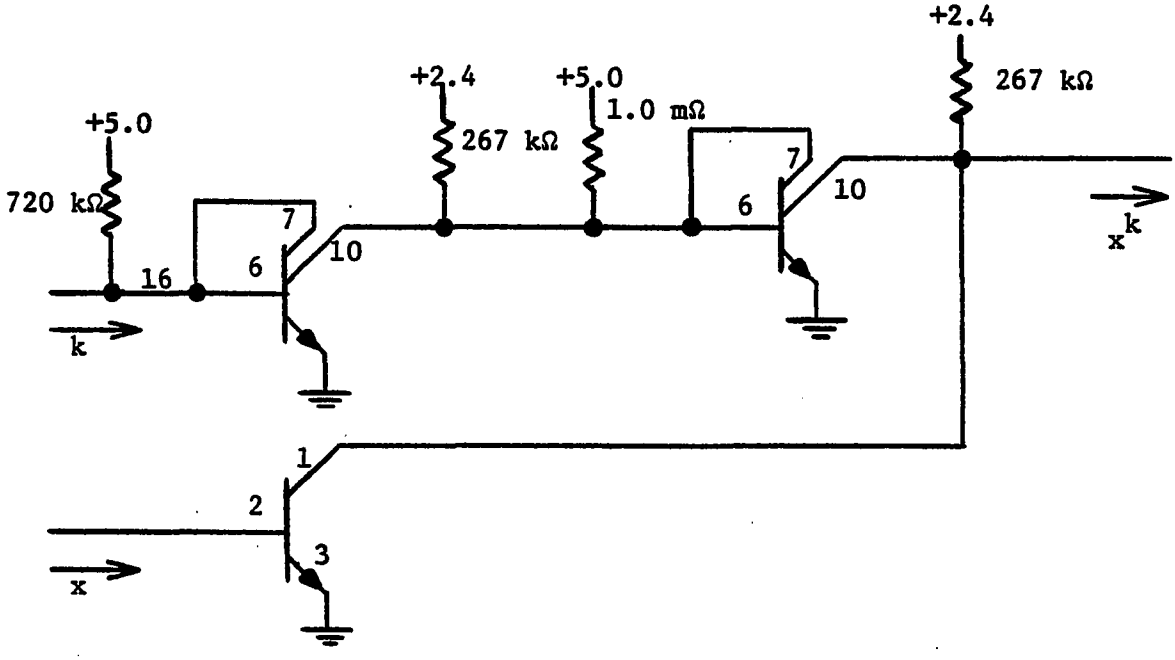


Figure A.8. Unary "Inverter" Construction

Table A.12. B(2):P(4) Logic Values for the Unary "Inverter" Gate

x^k	0	1	2	3
0	2.1	3.2	5.8	7.5
1	0.2	0.2	0.3	0.4
2	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0

Output, x^k in μAmps

The universal M unary "clockwise cycling" operation originally shown in Figure 2.16 is shown as it was actually constructed in Figure A.9. Five different XR-C501 integrated circuits were used. The resistors, R_{I1} to R_{I5}, are selected to match the electrical characteristics of the individual circuits.

The literal operation shown in Figure 2.17 was modified in Chapter III to accommodate the B_s and F_s lines (Figure 3.6). The B(2):P(4) full adder designed in Chapter III used a modified "clockwise cycling" operation and the literal operation. The complete schematic is shown in Figures 3.11 and 3.12. This circuit can be duplicated on an XR-200 master slice. The layout for a one bit B(2):P(4) full adder is shown in Figure A.10. Special consideration must be given to the injector rail configuration to minimize the series resistance. (Electrical measurements of the master slice performance are not available at the time of writing.) Also, on a prototype layout the injector bias resistor should be external. In fact, several different injector resistors should be used to help "fine tune" any new design. The design rules for master slice construction are intended to aid in binary applications. For multiple-valued

logic the layout and variation in gate array electrical parameters are more critical.

I^2L is the most promising logic family for MVL design today. However, current mirror inaccuracies and problems associated with injector layout need to be overcome. It would be nice if some amount of flexibility could be achieved in the basic gate array and still maintain the master slice concept. For example, rather than all devices having a fan out of five, in situations where higher loop gain is necessary it would be nice to open various emitters on the output stage as needed. Also, if the feedback collector could be identified and undersized to correct for mirror inaccuracies, then MVL designs would be greatly improved. Neither of these suggestions really contradict the master slice concept. Several arrays with varying numbers of fan out collectors and previously determined feedback elements could be provided in a standard configuration.

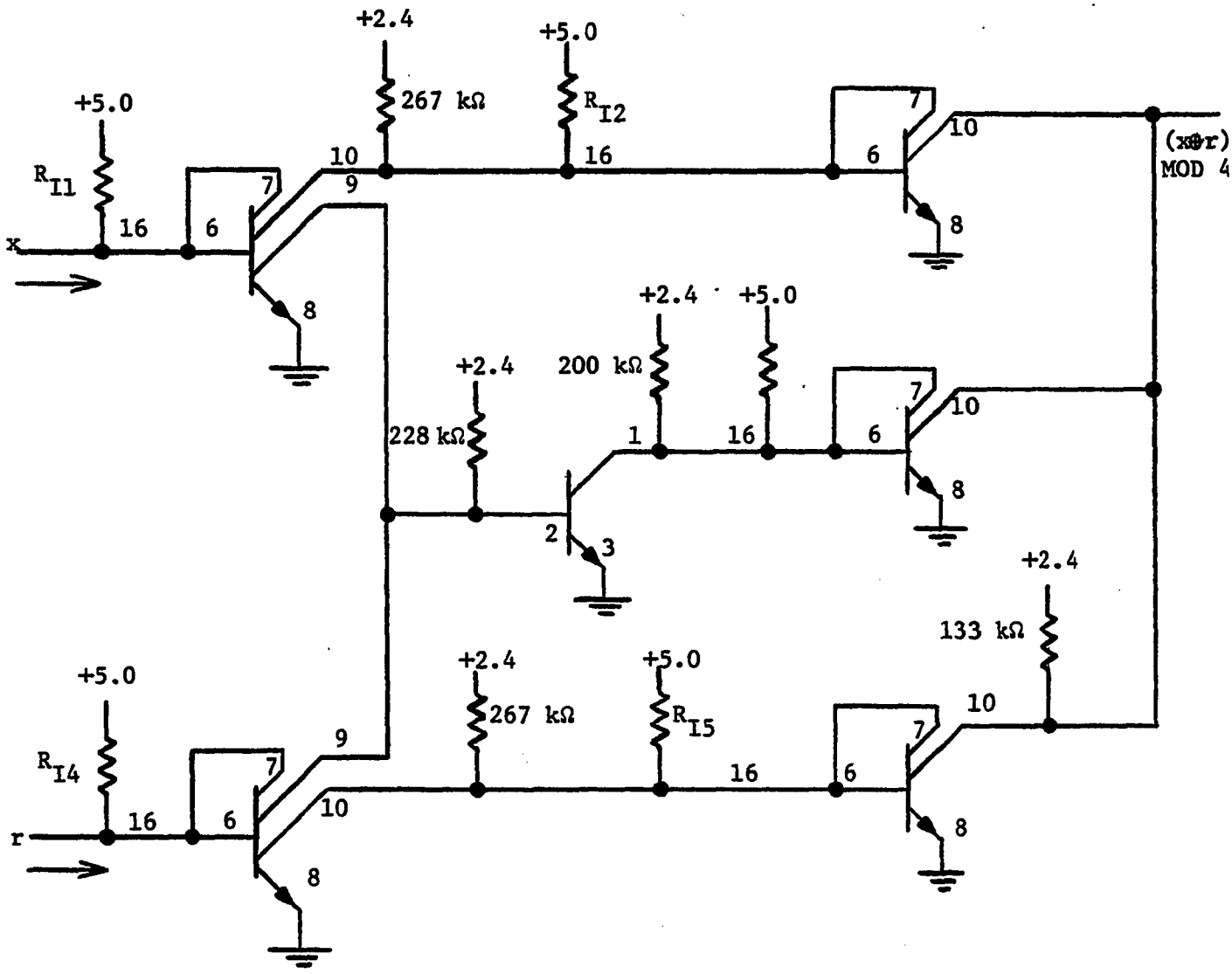


Figure A.9. Universal M Unary "Clockwise Cycling" Operation Construction

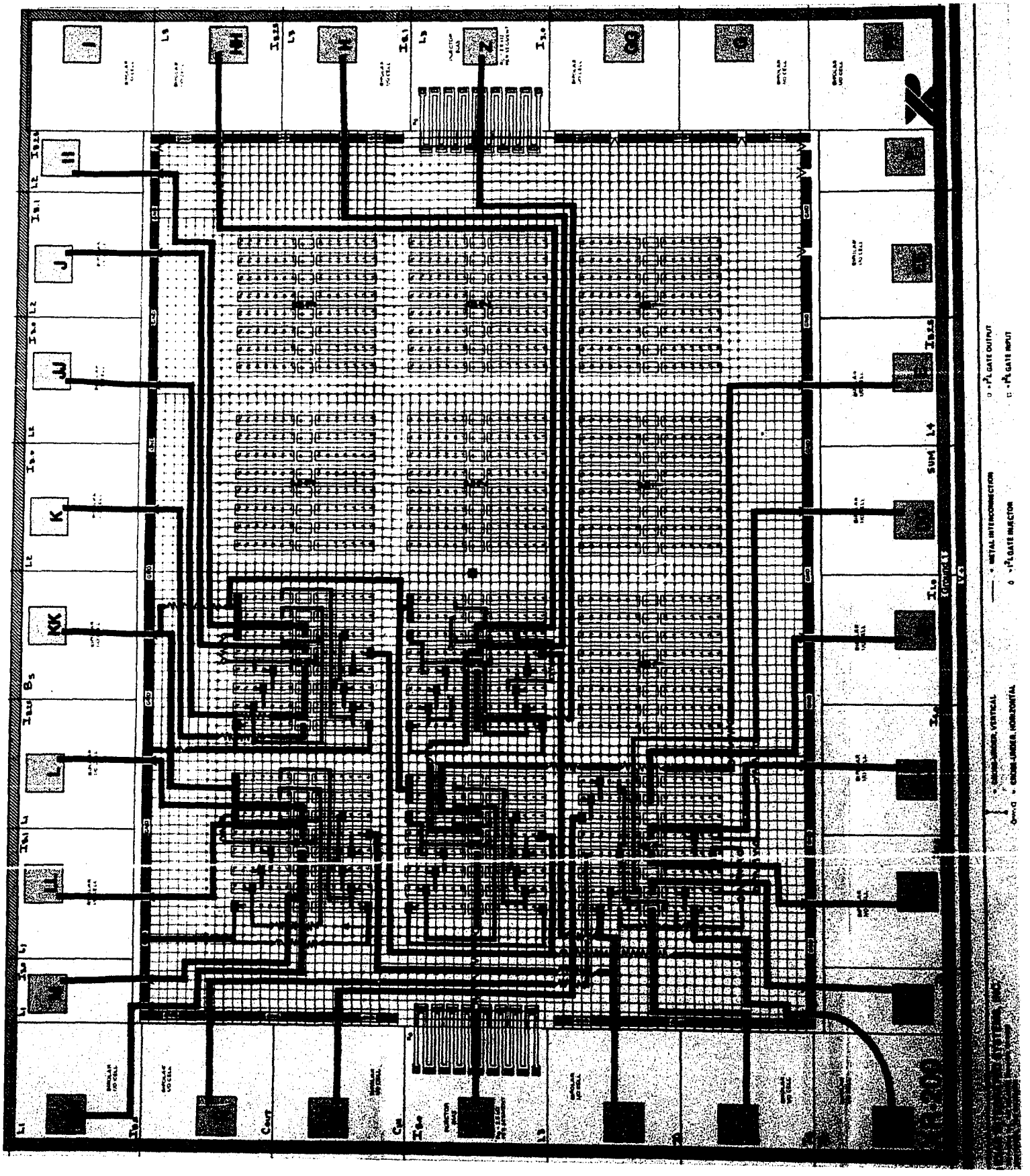


Figure A.10. I^2L Master Slice Layout
for a B(2):P(4) Full Adder