



Calhoun: The NPS Institutional Archive
DSpace Repository

Theses and Dissertations

1. Thesis and Dissertation Collection, all items

1995-09

A CMOS current-mode full-adder cell for multi-valued logic VLSI

Barton, Robert James

Monterey, California. Naval Postgraduate School

<http://hdl.handle.net/10945/7557>

Downloaded from NPS Archive: Calhoun



Calhoun is a project of the Dudley Knox Library at NPS, furthering the precepts and goals of open government and government transparency. All information contained herein has been approved for release by the NPS Public Affairs Officer.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

<http://www.nps.edu/library>

NAVAL POSTGRADUATE SCHOOL Monterey, California



THESIS

A CMOS CURRENT-MODE FULL-ADDER CELL
for
MULTI-VALUED LOGIC VLSI

by

Robert J. Barton III

September, 1995

Thesis Advisor:
Co-Advisor:

Douglas J. Fouts
Amr Zaky

Approved for public release; distribution is unlimited.

Thesis
B242344

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CA 93943-5101

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave Blank)		2. REPORT DATE September 1995	3. REPORT TYPE AND DATES COVERED Master's Thesis	
4. TITLE AND SUBTITLE A CMOS CURRENT-MODE FULL-ADDER CELL FOR MULTI-VALUED LOGIC VLSI (U)			5. FUNDING NUMBERS	
6. AUTHOR(S) Barton III, Robert James				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/ MONITORING AGENCY NAME(S) AND ADDRESS(ES)			10. SPONSORING/ MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) This thesis describes the design and implementation of a carry save adder cell for multi-valued logic VLSI. A four-valued system was chosen and the logic was analyzed and minimized using the HAMLET CAD tool [1]. SPICE was used to design and simulate the required behavior of the current-mode CMOS circuits. A VLSI test and evaluation integrated circuit was implemented with MAGIC and fabricated through the MOSIS service. The completed IC was tested and evaluated using a specially designed binary-to-multi-valued logic converter and decoder. Engineering modifications to the original current-mode inverter cells used by HAMLET were made leading to significant power savings in a complete design. The fabricated device performed as predicted by SPICE simulation. Exhaustive functional testing produced correct steady-state output signals for all cases of input loadings. Finally, we show HAMLET minimization heuristics are not efficient in the design of adder cells by comparison with an alternative modulo 4 carry save adder cell in current-mode CMOS.				
14. SUBJECT TERMS VLSI, Multi-Valued Logic, Current-Mode CMOS, High-Radix Circuitry			15. NUMBER OF PAGES 98	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT Unlimited	

Approved for public release; distribution unlimited.

A CMOS CURRENT-MODE FULL-ADDER CELL
FOR MULTI-VALUED LOGIC VLSI

Robert James Barton III
Lieutenant, United States Navy
B.S., United States Naval Academy, 1985

Submitted in partial fulfillment of the
requirements for the degrees of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

and

MASTER OF SCIENCE IN COMPUTER SCIENCE


from the

NAVAL POSTGRADUATE SCHOOL
September 1995


Author:

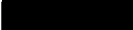

Robert James Barton III

Approved By:


Douglas J. Fouts, Thesis Advisor


Amr Zaky, Co-Advisor


Michael A. Morgan, Chairman,
Department of Electrical and Computer Engineering


Ted Lewis, Chairman
Department of Computer Science

T. No. 1
B. 24/2346
C. 2

ABSTRACT

The rapidly expanding fields of digital control and data signal processing require high performance compact arithmetic circuitry. In particular, the multiplication of large data words currently requires a network of carry-save adders to perform the addition of sets of partial products. This method suffers with scalability, particularly in the required die area and power required for carry-save adder trees. One possible solution is to take advantage of a compact design incorporating a high-radix signed number system.

This thesis describes the design and implementation of a carry save adder cell for multi-valued logic VLSI. A four-valued system was chosen and the logic was analyzed and minimized using the HAMLET CAD tool [1]. SPICE was used to design and simulate the required behavior of the current-mode CMOS circuits. A VLSI test and evaluation integrated circuit was implemented with MAGIC and fabricated through the MOSIS service. The completed IC was tested and evaluated using a specially designed binary-to-multi-valued logic converter and decoder. Engineering modifications to the original current-mode inverter cells used by HAMLET were made leading to significant power savings in a complete design. The fabricated device performed as predicted by SPICE simulation. Exhaustive functional testing produced correct steady-state output signals for all cases of input loadings. Finally, we show HAMLET minimization heuristics are not efficient in the design of adder cells by comparison with an alternative modulo 4 carry save adder cell in current-mode CMOS.

TABLE OF CONTENTS

I.	INTRODUCTION	1
A.	HISTORICAL PERSPECTIVE	1
B.	CURRENT WORK	3
C.	MULTIPLE VALUED LOGIC ARITHMETIC	4
II.	DESIGN OF MODULO 4 ADDER CELL	7
A.	DESCRIPTION OF THRESHOLD DETECTORS AND CURRENT GENERATORS	7
B.	HAMLET - A CAD TOOL FOR MVL DESIGN	11
C.	MINIMIZATION OF LITERALS USING HAMLET HEURISTICS	15
III.	IMPLEMENTATION	17
A.	CURRENT-MODE CMOS LOGIC	17
B.	LIMITATIONS OF THE PRESENT CAD TOOL - HAMLET	18
1.	HAMLET's PLA Generator module	18
2.	Power Consumption	18
3.	Scalability	19
C.	ENGINEERING MODIFICATIONS TO STEP-UP, STEP-DOWN, AND COLUMN GENERATOR CELLS	19
1.	Transistor Sizing	19
2.	Improved Logic Values and Switching Thresholds	20
3.	Simulation of New Cells	21
4.	Simulation of a Combined Minterm Using the New Cells	22
D.	CIRCUIT LAYOUT	23
1.	Basic Floor Plan	23
2.	Input Current Mirror Design and Layout	24
3.	VDD and Ground Rail Considerations	24
4.	Simulation of Completed Chip Design	25
IV.	TESTING AND EVALUATION OF MULTI-VALUED LOGIC (MOD4) CARRY SAVE ADDER	27
A.	DESIGN OF MODULO FOUR TESTBENCH	27
1.	Binary-to-Ideal Current Source Conversion	27
2.	Current Decoding at Outputs	28
B.	SUMMARY OF TEST RESULTS	29
1.	Static Power Tests	29
2.	Functional Testing	31
3.	Transient Analysis	32
4.	Results Discussion	34
V.	CONCLUSIONS	35
A.	HAMLET RECOMMENDATIONS	35
1.	Port to an X-Windows Application	35
2.	Design of Current-Mode Logic Cells	35
B.	AN ALTERNATIVE DESIGN FOR A RADIX-4 ADDER CELL IN	

CURRENT-MODE CMOS	36
1. A Different Approach	36
2. Simulation of Alternative Design	39
3. Conclusions of Alternative Design Method	40
C. RECOMMENDATIONS FOR FURTHER STUDY	40
1. Charged-Coupled Device (CCD) Logics	40
2. Resonant-Tunneling Diodes (RTD) Logics	40
3. Hierarchical MVL Design for Symmetric Functions.	41
4. Modulo 16 (hexadecimal) Current Mode CMOS Full Adder Cell	41
REFERENCES	43
APPENDIX	45
A. HAMLET REPORT - ORIGINAL SOP EXPRESSION	45
B. HAMLET REPORT - DEUCK & MILER MINIMIZATION	48
C. HAMLET REPORT - SIMULATED ANNEALING MINIMIZATION	50
D. SPICE FILE - STEP UP GENERATOR	52
E. SPICE FILE - STEP DOWN GENERATOR	54
F. SPICE FILE - COLUMN OUTPUT GENERATOR	56
G. SPICE FILE - TERM $3^*X(2,2)Y(3,3)C(2,2)$	58
H. SPICE FILE - CURRENT MIRROR	62
I. SPICE INPUT FILE - MODULO 4 ADDER DESIGN	65
J. MAGIC LAYOUTS	68
K. SCOPE PHOTOGRAPHS - TRANSIENT ANALYSIS	77
L. TRANSIENT ANALYSIS SCOPE PLOT - MODULO 4 ADDER CELL	81
M. FABRICATION PHOTOGRAPH	82
INITIAL DISTRIBUTION LIST	83

LIST OF FIGURES

2.1 CMOS Inverter With Current Input	7
2.2 Transfer Characteristics for Current Input Single Stage and Two-Stage CMOS Inverters	9
2.3 Step Down Function Generator	9
2.4 Step Up Function Generator	10
2.5 Transfer Curves for Function Generators	10
2.6 Example of a 2 Variable 4-Valued Function $f(X)$	11
2.7 Mapping for Carry Save Adder Sum Function in Radix 4	12
2.8 Mapping for Carry Save Adder Carry Out Function in Radix 4	14
3.1 Current Logic Levels	17
3.2 Final Layout of Design	23
3.3 Current Mirror Response	24
3.4 Current Measurements at VDD and Ground Rails	25
3.5 Complete Functional Test	26
4.1 Testbench Input Circuitry	27
4.2 Testbench Output Circuitry	28
4.3 Measured and SPICE Simulation of No Load Static Power Consumption	30
4.4 SPICE Model Propagation Delay for Sum and Carry Out	33
5.1 Current Input Complementary Pass Gate	37
5.2 Block Diagram of New Modulo 4 Adder	38
5.3 Chip Layout of Alternative Design	38
5.4 Transient Analysis of Alternate Design	39
A.1 Step Up Generator Cell	68
A.2 Step Down Generator Cell	69
A.3 Column Output Generator Cell	70
A.4 Term $3*X(2,2)Y(3,3)C(2,2)$	71
A.5 Current Mirror	72
A.6 HAMLET Design Modulo 4 Adder	73
A.7 Current-Input Complementary Transmission Gate	74
A.8 Alternate Design Modulo 4 Adder	75
A.9 Fabrication Layout Alternate Design Adder	76
A.10 T_r Sum 3->0	77
A.11 T_r Sum 0->3	77
A.12 T_{pdf} Sum 3->0	78
A.13 T_{pdf} Sum 0->3	78
A.14 T_r Carry Out 1->0	79
A.15 T_r Carry Out 0->1	79
A.16 T_{pdf} Carry Out 1->0	80
A.17 T_{pdf} Carry Out 0->1	80
A.18 Sum 0->3 Power Transients	81
A.19 Die Photo Modulo 4 Adder Cell	82

LIST OF TABLES

2.1 SOP Expression for Sum Function	13
2.2 SOP for Carry Function	15
2.3 Reduced SOP for Sum Function	16
2.4 Reduced SOP for Carry Function	16
3.1 New Logic and Threshold Current Values	20
3.2 New Step Up and Step Down Generator Output Current Design	21
3.3 Timing and Power Simulation of Cells	21
3.4 Term 3 X(2,2) Y(3,3) C(2,2)	22
4.1 Component Static Power	29
4.2 Full Power Measurements	31
4.3 Functional Test for Device 1	32
4.4 Propagation Delay	33

Acknowledgement

For Pam and Baby Kay.



I. INTRODUCTION

A. HISTORICAL PERSPECTIVE

Circuits which implement binary logic can be characterized by the effect of the circuit itself on the signal or signals being processed. A **restoring circuit** is capable of interpreting logic signals and then generating an output which is very narrowly constrained to the given signal space. In logic function implementations in higher radix(r) systems, the signal space does not significantly increase, whereas the number of required logic levels represented within the signal space increases in r . Narrower signal logic bands within the circuit present greater difficulty in maintaining functional integrity across cascaded non-restoring partitions within the circuit.[2]

In the latter half of the 1970s, a great deal of interest in ternary logic devices was prevalent. The design approach was, and in many instances remains, implementation of ternary (or higher) radix logic using binary components. For example, in a ternary system, the physical medium must have three equiprobable states and the distances between the states must be equal, or very nearly so. The use of a **cyclic** physical medium would allow optimal design of such a system. Alas, quantifiable cyclic phenomenon is extremely rare in the field of electronic devices. The approach then is to use the cyclic properties of binary circuits in such a way as to represent higher radices. In general, these circuits can be **voltage mode** or **current mode**. During this period various circuits were proposed using TTL(Transistor-Transistor Logic), ECL(Emitter-Coupled Logic), and CMOS.[3]

By the end of the decade Current Mode Logic (CML) was regarded as a simple, compact solution for higher radix arithmetic circuits. Several test example devices of full adder circuits were designed and implemented in current-mode CMOS, mainly in modulo 4. Greater speed performance was achievable using high-speed ECL, but the primary goal was to reduce circuit size and thus reduce cost and power requirements for arithmetic devices operating on large data words.[4]

In order to achieve greater speeds, GaAs MESFET logic was becoming an important area of research for multi-valued circuits. Circuits which are general, i.e., expandable to higher radices, and able to perform at low power with good noise margin performance are highly desirable. An approach presented by Tront [5] in 1979 included using MESFETS in which the individual transistors on the device had varying pinchoff voltages, at least one at each logic level desired. The difficulty with this approach is that V_p is a property which is only controlled at fabrication, either through channel thickness or doping concentrations. Even now, fabrication facilities have difficulty keeping threshold voltage within 50% of design tolerances.

During the late 1970s and into the 1980s, a great deal of work was done on multiple valued logic theory and modeling, as well as minimization techniques, but very few physical test devices were implemented. It is the explosion of progress in fabrication technology that allows much tighter control of device intrinsic properties. In the early 1980s Charge-Coupled Device logic became of interest. CCD implementation is in the category of charge-mode circuits. These circuits have many similarities with current-mode logic design, most significantly, the basic gate required for implementation is a common storage well. The storage well acts as a kind of "logical accumulator", much in the same way current mode CMOS makes extensive use of the "wired sum" to represent various signal levels.

By the end of the 1980s, current-mode CMOS was gaining in popularity for multi-valued function implementations. First, integrated circuit process technology is the same for binary CMOS as well as current-mode CMOS. This allows for the integration of multi-valued current-mode modules and binary logic circuits on the same VLSI device. Utilizing bi-directional current-mode CMOS formed the basis for the first presented CAD tool for implementing multi-valued functions in VLSI. This CAD tool was developed largely at Naval Post Graduate School, and is called HAMLET. HAMLET is a heuristics analyzer and CAD tool for multi-valued programmable logic arrays. In its current form, current mode CMOS is used exclusively for function implementation. The advantages are

simplicity of design and very inexpensive fabrication costs. In the future, it is envisioned that HAMLET will be capable of implementations in CCD logic, and possibly an implementation based upon resonant tunneling diode devices (RTDs).

B. CURRENT WORK

In recent years, continued improvements in VLSI fabrication processes have led to a renewed interest in current-mode CMOS high-radix arithmetic circuits. Of particular importance is the development of high speed compact multiplier circuits for the rapidly expanding fields of digital signal processing and digital control systems. In most modern high-speed arithmetic units, multiplication of long data words is performed by simultaneously generating sets of partial products and then summing them together with a network of carry save adders (CSAs) in an operation that is referred to as "row reduction." Although the network of CSAs lends itself very well to pipelining in high-speed processors, binary multipliers using the Wallace Tree [6] approach suffer from scalability problems. Scalability difficulties can be overcome by utilizing a high-radix signed number system to significantly reduce the number of transistors and the die area required for large data-word arithmetic.

Presently, current-mode CMOS logic is not a suitable solution for the generation of partial products in a large multiplier circuit. One alternative is to use binary CMOS circuits to implement a modification of Booth's algorithm[7][8]. However, the design of high-radix adders lends itself well to current-mode CMOS, primarily because of the wired sum[9] function. One of the key elements of the adder circuit is the threshold detector[10]. This particular circuit has, in the past, proved to be difficult to scale down to minimum VLSI implementation device sizes. With the vast and continuing improvements in CMOS fabrication processes, this design problem can be minimized. Of recent interest is the development of alternative low-power high-speed threshold detector circuits such as those found in CML current-mode full adders[11].

In this thesis we demonstrate the design and implementation of a radix-4, carry-save adder cell for multi-valued VLSI. The adder receives current inputs X, Y, and Carry_{IN}, generating the Sum and Carry_{OUT} outputs. The Carry_{IN} input of the carry save adder accepts all possible radix-4 inputs (0:3) so that it may be used as a three-to-two row reduction unit in the CSA adder network previously described.

The first test IC was designed and implemented using the HAMLET cad tool for multi-valued logic expressions. A second test IC of the same radix and function was designed and implemented without the assistance of the CAD tool, using a different technique which takes advantage of the symmetrical property of the addition logic. The alternative design actually uses two radices, modulo-4 for input and output, and modulo-12 logic within the device.

C. MULTIPLE VALUED LOGIC ARITHMETIC

As previously mentioned, high-radix multiplication does not lend itself to current-mode CMOS, and is therefore not discussed in this thesis. Before an adder can be designed in a high-radix system (greater than binary), we need a convenient way to express the logic combinations of the inputs. The expression which describes the sum function in any radix is:

$$\text{SUM}\{f(C_i, X_i, Y_i)\} = C_i(\text{XOR})X_i(\text{XOR})Y_i \quad (1.1)$$

This expression generates a set of product terms of the form:

$$\text{SUM}_k = 1^{*3}C^{31}X^{31}Y^1 \quad (1.2)$$

This notation translates as for an input at C of 3, an input at X of 1,2, or 3, and an input at Y of 1, generate an output of 1. Thus, the sum function is the summation of this set of product terms:

$$\text{SUM}\{f(C_i, X_i, Y_i)\} = \sum_{k=1}^n \text{SUM}_k \quad (1.3)$$

In current mode CMOS, the distinction between levels of logic at the inputs is accomplished by a threshold detector. The output value for each term in the sum is realized

by a current generator. To accomplish the required summation of the product terms, the outputs of the current generators for each product term are physically “wired together”, thus the notion of a “wired sum” in current mode CMOS multiple-valued VLSI. In the following chapter we discuss the use of the CAD tool HAMLET and develop the equations and device modules for the threshold detectors and current generators.

II. DESIGN OF MODULO 4 ADDER CELL

Described herein is the design of a radix-4 carry save adder cell utilizing the CAD tool HAMLET. The ideal threshold detectors and current generators are also derived. We show the result of HAMLET minimization heuristics on a set of product terms that fully describes the required sum and carry-out functions.

A. DESCRIPTION OF THRESHOLD DETECTORS AND CURRENT GENERATORS

At the core of the current-mode circuits required to implement an MVL expression is a CMOS inverter with a *current* input.

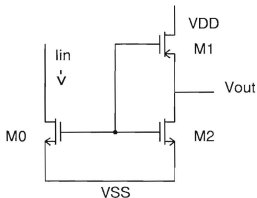


Figure 2.1: CMOS Inverter With Current Input

In Figure 2.1, The nFET device M0 operates in saturation because $V_{DS} = V_{GS}$ and $V_{DS} > V_T$. Ignoring the Early voltage and bulk effects which are minimal, the saturation value of the drain current is given by:

$$I_D = K(V_{GS} - V_T)^2 \quad (2.1)$$

where the process-dependent constant K can be determined by:

$$K = 1/2(\mu_n C_{ox}(W/L)) \quad (2.2)$$

Since the drain current I_d is the input current, then the operating point at which the nMOS device M0 becomes saturated is found by:

$$I_{SW} = C(V_{SW} - V_t)^2(W/L) \quad (2.3)$$

I_{SW} and V_{SW} refer to the “switching” current and switching voltage respectively.

When M0 is saturated, the input voltage to the CMOS inverter stage is high, both the pMOS (M1) and the nMOS(M2) devices are in the saturation region such that,

$$I_{Dsp} = -I_{Dsn} \quad (2.4)$$

$$I_{Dsp} = K_p(V_{SW} - V_{Dsp} - V_{tp})^2 \quad (2.5)$$

$$I_{Dsn} = K_n(V_{SW} - V_{tp})^2 \quad (2.6)$$

Since $K_n = K_p$ on the same chip, solving equations 2.5 - 2.7 for switching voltage yields:

$$V_{SW} = (V_{Dsp} + V_{tp} + V_{tn})/2 \quad (2.7)$$

As seen by equation 2.7 the switching voltage is a function of the threshold voltages of the transistors M1 and M2, which is constant through the chip. Thus, the switching voltage of these inverters are constant on the same chip. With a constant switching voltage V_{SW} , the switching current I_{SW} of equation 1.4 is strictly a function of transistor geometry (W/L). Therefore the inverter will generate a V_{OUT} of 0 when the designed I_{SW} is experienced at the drain of M0. In practice, we will use two inverter stages for a “sharper” transition at V_{out} . [12][10]

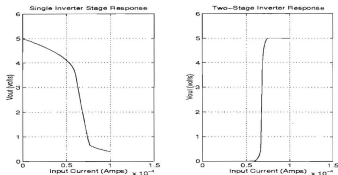


Figure 2.2: Transfer Characteristics for Current Input Single Stage and Two-Stage CMOS Inverters

The basic cells used to implement the current mode logic consist of the current-input CMOS inverter with the output connected to a single nMOS or pMOS transistor (figure 2.11). Since the inverter output will produce $V_{OUT} \gg$ threshold voltage of M3, this circuit acts as a constant current source when the threshold current I_{SW} is detected. The step-down generator produces a constant I_{out} when the input current is **less than** the switching current, else 0. The step-up generator has an output current $I_{out} = 0$ unless the input current is **greater than** I_{SW} .

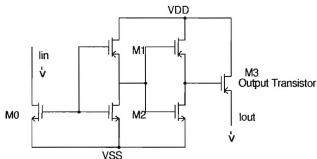


Figure 2.3: Step Down Function Generator

B. HAMLET - A CAD TOOL FOR MVL DESIGN

To realize a logic function in multi-valued log (MVL), a design method is required to develop the abstraction into a format on which CAD tools can perform heuristics. Unlike binary logic design, MVL of radix greater than 2 quickly becomes difficult to conceptualize. For example, while a two-input NAND gate is readily described in binary logic, there is no “symbol” to functionally describe a two-input NAND gate in a logic system of 6 variables.

The MVL CAD tool HAMLET uses a sum-of-products (SOP) expression as formatted input[]. Since HAMLET will minimize the SOP expression, any valid SOP expression which completely describes the functionality of the design is sufficient. The SOP is derived from a mapping technique which resembles the familiar Karnaugh Map method.

Let $X = \{x_1, x_2, \dots, x_n\}$ be a set of n variables in a logic system of radix r , where x_i takes on values from $R = \{0, 1, 2, \dots, r-1\}$. The function $f(X)$ can be fully described in the following example:

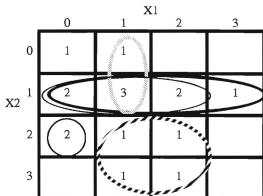


Figure 2.6: Example of a 2 Variable 4-Valued Function $f(X)$

The groups in Figure 2.5 represent implicants of the function $f(X)$. The SOP representation for this particular choice of implicants in the format for HAMLET would be the following:

$$f(x_1x_2) = 1^0x_1^0x_2^0 + 1^1x_1^1x_2^0 + 1^0x_1^0x_2^1 + 1^1x_1^1x_2^1 + 1^0x_1^0x_2^2 + 1^1x_1^1x_2^2 + 2^0x_1^0x_2^3 + 2^0x_1^0x_2^2 \quad (2.8)$$

The single-bit adder has 3 inputs, namely, a carry in C_i , 1st addend X_i , 2nd addend Y_i , and 2 outputs, the sum S_i , and the carry out C_o . Recalling for the full adder:

$$\text{SUM}\{f(C_i, X_i, Y_i)\} = C_i(\text{XOR})X_i(\text{XOR})Y_i \quad (2.9)$$

By utilizing the mapping technique described earlier, the SUM function for the carry save adder in radix 4 can be completely described by:

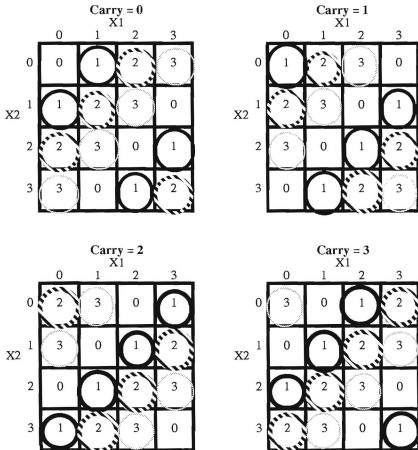


Figure 2.7: Mapping for Carry Save Adder Sum Function in Radix 4

In order for the expression to be accepted into the HAMLET tool, the mappings described above must first be translated into the following SOP expression:

TABLE 2.1: SOP Expression for Sum Function

$1 * x1(1,1) * x2(0,0) * x3(0,0)$	$+ 2 * x1(0,0) * x2(0,0) * x3(2,2)$
$+ 2 * x1(2,2) * x2(0,0) * x3(0,0)$	$+ 3 * x1(1,1) * x2(0,0) * x3(2,2)$
$+ 3 * x1(3,3) * x2(0,0) * x3(0,0)$	$+ 1 * x1(3,3) * x2(0,0) * x3(2,2)$
$+ 1 * x1(0,0) * x2(1,1) * x3(0,0)$	$+ 3 * x1(0,0) * x2(1,1) * x3(2,2)$
$+ 2 * x1(1,1) * x2(1,1) * x3(0,0)$	$+ 1 * x1(2,2) * x2(1,1) * x3(2,2)$
$+ 3 * x1(2,2) * x2(1,1) * x3(0,0)$	$+ 2 * x1(3,3) * x2(1,1) * x3(2,2)$
$+ 2 * x1(0,0) * x2(2,2) * x3(0,0)$	$+ 1 * x1(1,1) * x2(2,2) * x3(2,2)$
$+ 3 * x1(1,1) * x2(2,2) * x3(0,0)$	$+ 2 * x1(2,2) * x2(2,2) * x3(2,2)$
$+ 1 * x1(3,3) * x2(2,2) * x3(0,0)$	$+ 3 * x1(3,3) * x2(2,2) * x3(2,2)$
$+ 3 * x1(0,0) * x2(3,3) * x3(0,0)$	$+ 1 * x1(0,0) * x2(3,3) * x3(2,2)$
$+ 1 * x1(2,2) * x2(3,3) * x3(0,0)$	$+ 2 * x1(1,1) * x2(3,3) * x3(2,2)$
$+ 2 * x1(3,3) * x2(3,3) * x3(0,0)$	$+ 3 * x1(2,2) * x2(3,3) * x3(2,2)$
$+ 1 * x1(0,0) * x2(0,0) * x3(1,1)$	$+ 3 * x1(0,0) * x2(0,0) * x3(3,3)$
$+ 2 * x1(1,1) * x2(0,0) * x3(1,1)$	$+ 1 * x1(2,2) * x2(0,0) * x3(3,3)$
$+ 3 * x1(2,2) * x2(0,0) * x3(1,1)$	$+ 2 * x1(3,3) * x2(0,0) * x3(3,3)$
$+ 2 * x1(0,0) * x2(1,1) * x3(1,1)$	$+ 1 * x1(1,1) * x2(1,1) * x3(3,3)$
$+ 3 * x1(1,1) * x2(1,1) * x3(1,1)$	$+ 2 * x1(2,2) * x2(1,1) * x3(3,3)$
$+ 1 * x1(3,3) * x2(1,1) * x3(1,1)$	$+ 3 * x1(3,3) * x2(1,1) * x3(3,3)$
$+ 3 * x1(0,0) * x2(2,2) * x3(1,1)$	$+ 1 * x1(0,0) * x2(2,2) * x3(3,3)$
$+ 1 * x1(2,2) * x2(2,2) * x3(1,1)$	$+ 2 * x1(1,1) * x2(2,2) * x3(3,3)$
$+ 2 * x1(3,3) * x2(2,2) * x3(1,1)$	$+ 3 * x1(2,2) * x2(2,2) * x3(3,3)$
$+ 1 * x1(1,1) * x2(3,3) * x3(1,1)$	$+ 2 * x1(0,0) * x2(3,3) * x3(3,3)$
$+ 2 * x1(2,2) * x2(3,3) * x3(1,1)$	$+ 3 * x1(1,1) * x2(3,3) * x3(3,3)$
$+ 3 * x1(3,3) * x2(3,3) * x3(1,1)$	$+ 1 * x1(3,3) * x2(3,3) * x3(3,3);$

In a similar fashion, the following mapping fully describes the carry out (C_o) function:

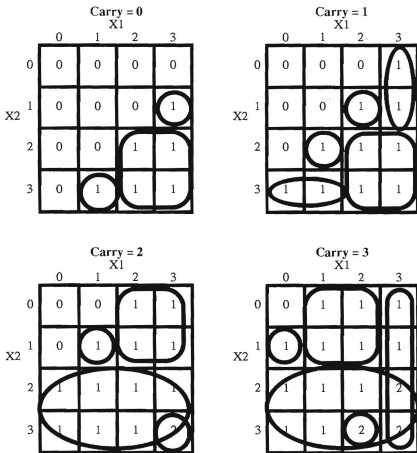


Figure 2.8: Mapping for Carry Save Adder Carry Out Function in Radix 4

The translated SOP expression resulting from this mapping is:

TABLE 2.2: SOP for Carry Function

$1 * x1(3,3) * x2(1,1) * x3(0,0)$	$+ 1 * x1(1,1) * x2(1,1) * x3(2,2)$
$+ 1 * x1(1,1) * x2(3,3) * x3(0,0)$	$+ 1 * x1(0,3) * x2(2,3) * x3(2,2)$
$+ 1 * x1(2,3) * x2(2,3) * x3(0,0)$	$+ 1 * x1(3,3) * x2(3,3) * x3(2,2)$
$+ 1 * x1(3,3) * x2(0,1) * x3(1,1)$	$+ 1 * x1(1,2) * x2(0,1) * x3(3,3)$
$+ 1 * x1(2,2) * x2(1,1) * x3(1,1)$	$+ 1 * x1(0,0) * x2(1,1) * x3(3,3)$
$+ 1 * x1(1,1) * x2(2,2) * x3(1,1)$	$+ 1 * x1(3,3) * x2(0,3) * x3(3,3)$
$+ 1 * x1(0,1) * x2(3,3) * x3(1,1)$	$+ 1 * x1(0,3) * x2(2,3) * x3(3,3)$
$+ 1 * x1(2,3) * x2(2,3) * x3(1,1)$	$+ 1 * x1(2,2) * x2(3,3) * x3(3,3)$
$+ 1 * x1(2,3) * x2(0,1) * x3(2,2)$	

C. MINIMIZATION OF LITERALS USING HAMLET HEURISTICS

Once the file containing the required SOP terms was input into the HAMLET CAD tool, a report was generated which returned the original expression with matrices representing the mappings given above for the sum and carry functions. Two heuristic minimization techniques were chosen to minimize the terms required for this design. The first was the Deuck&Miller and Proper&Armstrong heuristics[1]. This technique resulted in a reduction from 48 to 32 terms required for the sum function, and from 17 to 15 terms required to realize the carry function. Simulated Annealing minimization was also used, and HAMLET reported the same performance. In each case, the tool verifies each minimization by producing the mappings associated with each result. Copies of all HAMLET generated reports and a copy of the input data file are included in the appendix. Table 2.3 and Table 2.4 give the reduced SOP expressions returned by HAMLET which were utilized to implement the modulo-four adder. (Appendices A-C)

The reduced SOP expression generated by HAMLET for the sum function:

TABLE 2.3: Reduced SOP for Sum Function

$1 * x1(3,3) * x2(0,1) * x3(3,3)$	$+ 1 * x1(3,3) * x2(2,3) * x3(0,0)$
$+ 1 * x1(3,3) * x2(0,2) * x3(2,2)$	$+ 3 * x1(2,2) * x2(1,1) * x3(0,0)$
$+ 2 * x1(0,1) * x2(0,0) * x3(2,2)$	$+ 1 * x1(0,1) * x2(3,3) * x3(2,3)$
$+ 1 * x1(2,3) * x2(3,3) * x3(0,0)$	$+ 2 * x1(0,0) * x2(3,3) * x3(0,0)$
$+ 1 * x1(1,1) * x2(0,0) * x3(0,2)$	$+ 1 * x1(3,3) * x2(3,3) * x3(3,3)$
$+ 2 * x1(0,0) * x2(2,2) * x3(1,1)$	$+ 1 * x1(3,3) * x2(1,3) * x3(1,1)$
$+ 1 * x1(2,3) * x2(1,2) * x3(2,2)$	$+ 3 * x1(1,1) * x2(2,2) * x3(0,0)$
$+ 1 * x1(1,3) * x2(3,3) * x3(1,1)$	$+ 1 * x1(1,1) * x2(2,3) * x3(2,3)$
$+ 1 * x1(2,3) * x2(0,1) * x3(3,3)$	$+ 2 * x1(2,2) * x2(0,0) * x3(0,0)$
$+ 3 * x1(0,0) * x2(1,1) * x3(2,2)$	$+ 2 * x1(1,1) * x2(1,1) * x3(0,1)$
$+ 3 * x1(2,2) * x2(2,2) * x3(3,3)$	$+ 3 * x1(0,0) * x2(0,0) * x3(3,3)$
$+ 1 * x1(0,1) * x2(2,3) * x3(3,3)$	$+ 3 * x1(2,2) * x2(3,3) * x3(2,2)$
$+ 3 * x1(3,3) * x2(0,0) * x3(0,0)$	$+ 1 * x1(1,3) * x2(1,1) * x3(3,3)$
$+ 3 * x1(2,2) * x2(0,0) * x3(1,1)$	$+ 1 * x1(0,1) * x2(0,1) * x3(1,1)$
$+ 1 * x1(0,0) * x2(2,3) * x3(0,0)$	$+ 1 * x1(0,0) * x2(1,2) * x3(0,1)$
$+ 1 * x1(2,3) * x2(2,3) * x3(1,1)$	$+ 1 * x1(2,3) * x2(2,2) * x3(2,2);$

The reduced SOP expression for the carry function is:

TABLE 2.4: Reduced SOP for Carry Function

$1 * x1(2,2) * x2(0,1) * x3(2,3)$	$+ 1 * x1(2,2) * x2(1,1) * x3(1,1)$
$+ 1 * x1(1,1) * x2(2,2) * x3(1,1)$	$+ 1 * x1(0,2) * x2(3,3) * x3(1,2)$
$+ 1 * x1(0,2) * x2(2,2) * x3(2,2)$	$+ 1 * x1(3,3) * x2(3,3) * x3(2,2)$
$+ 1 * x1(3,3) * x2(1,3) * x3(0,0)$	$+ 1 * x1(1,2) * x2(3,3) * x3(0,0)$
$+ 1 * x1(0,0) * x2(1,3) * x3(3,3)$	$+ 1 * x1(1,1) * x2(1,1) * x3(2,2)$
$+ 1 * x1(3,3) * x2(0,3) * x3(1,3)$	$+ 1 * x1(1,3) * x2(2,3) * x3(3,3)$
$+ 1 * x1(2,2) * x2(2,2) * x3(0,1)$	$+ 1 * x1(2,2) * x2(3,3) * x3(3,3);$
$+ 1 * x1(1,1) * x2(0,1) * x3(3,3)$	

III. IMPLEMENTATION

A. CURRENT-MODE CMOS LOGIC

To implement the MVL expression, Current-Mode CMOS was utilized. In essence, different values of current correspond to the four different logic levels. A serious drawback to this implementation is that it requires current to be constantly flowing in the circuit. The logic levels and switching points were designed as shown in figure 3.1. Currents are shown above are in μA .

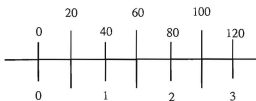


Figure 3.1: Current Logic Levels

To achieve these characteristics, three components are required. The step-up and step-down generators supply the appropriate currents to the column output generator[10]. Together, these components form one minterm of the logic equation.

The column output generator outputs a current of predetermined value if there is no current flow at its input. This input is the wired-OR of the step-up and step-down generators. The predetermined value will be one of the designed logic levels. This value is set by the gate dimensions of the output transistor. Note that the column output generator is insensitive to the value of the input current when it is present, i.e. it does not produce an output if the input is not zero. There is one column output generator per minterm.

It is the job of the step-up and step-down generators to produce zero current when an output current is desired for that term. The step-down generator cuts off the output when the input rises above the desired threshold values. These threshold values correspond to the

switching points for the desired logic level and are set by the gate dimensions of the input transistor. Typically, there is a step-up and step-down generator for each input.

Figure 2.3 illustrates the use of two inverter stages between the sensing transistor of the threshold detector and the output transistor for each component. Unfortunately, both stages are required in order to produce accurate and sharp transitions at the appropriate logic current levels of input. The consequence of this design is the added propagation delay incurred across two inverters, as well as an increase in the number of transistors and the power of the device as a whole.

B. LIMITATIONS OF THE PRESENT CAD TOOL - HAMLET

During the implementation phase of the radix-4 adder, certain limiting features of HAMLET were discovered and re-engineered using the current MOSIS 2.0 micron design rules.

1. HAMLET's PLA Generator Module

A program created by Ko[10] in support of the HAMLET project generates a PLA in current mode CMOS when given an MVL SOP expression. The resulting PLA conforms to MOSIS design rules. When originally designed, this module could be run on ISIS graphics workstations or a VAX. Currently, there is no operating version of this tool available for use on any modern graphical workstation available at NPS. However, the individual cells can still be assembled by hand, and a custom layout vice a generated PLA was created for this device.

2. Power Consumption

In the present tool, the step-up function generator modules are designed to produce output currents in the range of 150uA to 180uA. Likewise, the step-down function generators produce output currents of approximately 240uA. However, these outputs only function as input to column generators, which have a switching threshold of approximately

20uA. Thus, internally, the switching currents produced by a device implemented using these cells tend to consume more power than necessary.

3. Scalability

In order for a large number of terms to function correctly, a high degree of accuracy is required in the individual cells. Column generators must produce output currents that are very nearly the ideal logic values, or small errors will tend to compound quickly as terms are connected for the wired sum function. Also, the switching current levels of the threshold detectors in the step-up and step down generators should ideally “split” the ideal logic values for maximum effectiveness against introduced errors. As can be readily seen by figures 4.2-4.4 of [10], the original pla generator tool suffered from both inaccuracies in the output generator as well as at the inputs to the threshold detectors. For example, if the outputs from two column generators are wired together, and each is designed to produce a logic 1, the wired sum would be approximately 110uA, a logic 2. However, the step-up function generator reports detection of a logic 3 input beginning at 110uA. Thus, if this wired sum was to be used as an input to another term which included the step-up 3 function, an error would occur. Similar examples can be contrived for the step down function cells.

C. ENGINEERING MODIFICATIONS TO STEP-UP, STEP-DOWN, AND COLUMN GENERATOR CELLS

In order to improve the implementation of the MVL expression design, several engineering changes were made to the basic cells used by HAMLET and implemented in a custom layout of the final device.

1. Transistor Sizing

Using new MOSIS design rules, minimum wire width is reduced from $4\mu\text{m}$ to $2\mu\text{m}$ with a λ of $1.0\mu\text{m}$. This allows more precise control of threshold detector values and column output generator current levels.

2. Improved Logic Values and Switching Thresholds

In order to reduce power requirements for this design and improve noise margin performance, the logic value thresholds and the ideal current values produced by the step-up and step-down generators were redesigned as follows:

TABLE 3.1: New Logic and Threshold Current Values

Logic Values	Original HAMLET Design Values:			
	Step Down Generator	Ideal Current	Current Generator	Step Up Generator
0	0 μ A	0 μ A	*	10 μ A
1	20 μ A	50 μ A	58 μ A	60 μ A
2	80 μ A	100 μ A	100 μ A	110 μ A
3	130 μ A	150 μ A	136 μ A	not defined
New Design Values:				
0	0 μ A	0 μ A	*	20 μ A
1	20 μ A	40 μ A	40 μ A	60 μ A
2	60 μ A	80 μ A	80 μ A	100 μ A
3	100 μ A	120 μ A	120 μ A	150 μ A

A substantial power savings was also realized by reducing the current output from the step up and step down cells. The column output generators require much less current for switching purposes than in the original HAMLET design. Table 3.2 summarizes the redesign of the step up/down generator output current levels.

TABLE 3.2: New Step Up and Step Down Generator Output Current Design

Output Currents From Step Up/Down Generators:		
	Step Up Generator	Step Down Generator
Original Cells	180 μ A	240 μ A
New Cell Design	60 μ A	70 μ A

3. Simulation of New Cells

The individual circuits were implemented using MAGIC and then extracted to SPICE for simulation and analysis. Because of the nature of multi-valued logic, the normal definition of noise margin does not apply. For this circuit, the noise margin can be defined as the difference between the output logic level and the input switching thresholds of the next gate. The optimum noise margin can be achieved only by centering the output logic value between its associated switching thresholds. These nominal current values have been achieved within 2 μ A.

The timing delays and power consumption of the various components are detailed below.

TABLE 3.3: Timing and Power Simulation of Cells

	Step Up Generator	Step Down Generator	Column Output Generator
T_r(ns)	4.19	1.43	1.23
T_f(ns)	2.04	3.12	1.51
T_{PLH}(ns)	11.48	4.29	2.16
T_{PHL}(ns)	2.74	9.61	5.69
P_{STATIC}(mW)	1.78(H)/ 0.0L	1.07	0.42(H)/ 0.81(L)

TABLE 3.3: Timing and Power Simulation of Cells

	Step Up Generator	Step Down Generator	Column Output Generator
$P_{LH(peak)}(mW)$	1.78	1.34	0.87
$P_{HL(peak)}(mW)$	1.78	2.25	1.60

The power dissipation for this circuit appears to be fairly high. This is expected because the proper operation of this type of multi-valued logic circuit requires continuous current flow at different levels to achieve the desired results. Unlike conventional CMOS logic, the static power dissipation is much greater than the transistor leakage current.

4. Simulation of a Combined Minterm Using the New Cells

The components were combined to yield a fairly typical minterm for analysis. The minterm chosen was:

$$3 X(2,2) Y(3,3) C(2,2) \quad (3.1)$$

This term was implemented in MAGIC and extracted to SPICE for simulation and analysis. The results are compiled in the following table:

TABLE 3.4: Term 3 X(2,2) Y(3,3) C(2,2)

$T_r(ns)$	4.19
$T_f(ns)$	2.04
$T_{PLH}(ns)$	11.48
$T_{PHL}(ns)$	2.74
$P_{STATIC}(mW)$	1.78(H)/0.0L
$P_{LH(peak)}(mW)$	1.78
$P_{HL(peak)}(mW)$	1.78

As expected, the rise and fall times are fairly close and reflect the contribution of the column output generator. The propagation delays are a result of the series connection of the step generators and the column output generator. In this configuration, there are a total of

4 CMOS inverter stages that a signal must propagate through prior to the generator output transistor. The power results are expected and reflect the affect of choosing a current-mode CMOS implementation. The static power dissipation is fairly significant, while the dynamic power is of much lower value. Unfortunately, these will add to the static figure during logic transitions and result in a substantial power transient. (Appendices D-G)

D. CIRCUIT LAYOUT

The circuit was laid out using MAGIC for a MOSTIS 2.0 micron process, using n-well process data from Orbit fabrication facility.

1. Basic floor plan

The basic floor plan of the chip consists of 2 "towers" wired together at the output to realize the truncated sum function, a single tower for the carry out function, and a set of current mirrors to replicate the inputs that are required for each term. The cell has 3 input and 2 output pads, as well as separate power and ground inputs for the sum function towers (2), carry function, and current mirror cell, respectively. The sum output is composed of 47 minterms, while the carry is composed of 17 minterms. The final layout for this device is shown in figure 3.2.

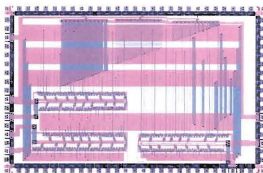


Figure 3.2: Final Layout of Design

2. Input Current Mirror Design and Layout

Each of these terms, in general, requires all three input current signals. A current mirror was designed to provide for these input requirements. We centered the current replicator design at 80uA because of the nonlinear nature of the current mirror circuit with a varying current reference. This results in less than a 10uA deviation from the desired current at the other logic levels.

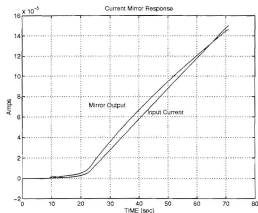


Figure 3.3: Current Mirror Response

3. VDD and Ground Rail Considerations

Power consumption was also a major design consideration, as discussed earlier. Care must be exercised to ensure that the circuit will not fail due to electromigration problems associated with excess current within the interconnect network. In addition, to separate power and ground rails for each tower and the current mirror unit, metal 2 was used which has a nominal rating of 1 mA/ μm of width. As can be observed in the power plot of Figure 3.4, at no point does the current exceed 30mA on any rail. These measurements were recorded during the functional test (Figure 3.5) These values are relatively high, but do not pose a threat to the circuit.

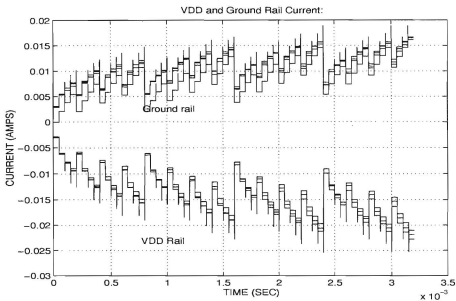


Figure 3.4: Current Measurements at VDD and Ground Rails

Finally, there are transients in the output due to the nature of the circuit. The various minterms turn “on” and “off” at different times depending on the current levels at their inputs. This results in transients while the circuit is stabilizing between input changes. Provided the transients do not exceed power ratings (as previously shown) and the output is sampled once it has stabilized, these hazards will not affect the circuit operation. The time delay for transients to subside can be included in the delay of the adder.

4. Simulation of Completed Chip Design

A series of square current waveforms were applied at each input in order to simulate all combinations possible. The SPICE model produced output showing the correct counting sequence at the output nodes for all input combinations.

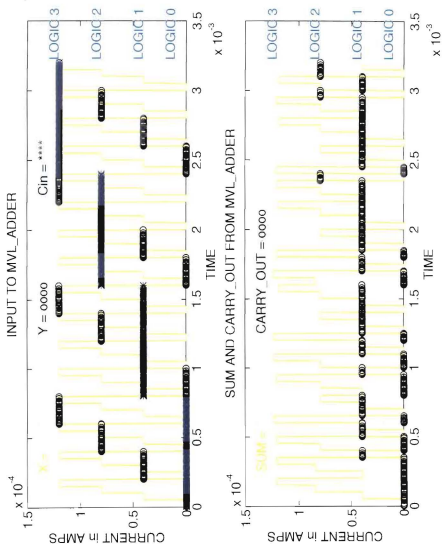


Figure 3.5: Complete Functional Test

Power and transient models obtained from SPICE are included in the results chapter for comparison with measured and observed values.

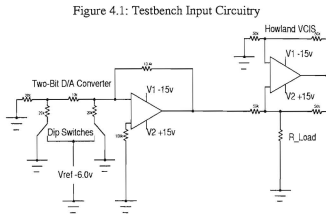
IV. TESTING AND EVALUATION OF MULTI-VALUED LOGIC (MOD4) CARRY SAVE ADDER

This section summarizes the performance characteristics and functionality of the multiple-valued logic (modulo four) carry save adder designed and implemented in current mode CMOS. A total of 15 devices were returned from fabrication, 12 of which were mounted in appropriate packages. Four devices from the lot were completely tested (dev1 - dev4). In order to readily test these devices, a custom testbench was designed and constructed.

A. DESIGN OF MODULO FOUR TESTBENCH

1. Binary-to-Ideal Current Source Conversion

It is desirable to have a method for converting binary digits to current levels as required by the inputs to the full adder. Conversely, a convenient method for measuring the sum and carry out is also necessary. Each of the three inputs, X, Y, and Carry in are driven by the following circuit:



The D/A converter stage is controlled by two dip switches which select between V_{ref} and open-circuit. The use of dip switches is dependent on the ability of the op-amp to

produce a virtual ground at the inverting terminal, and the input resistance of the op-amp being significantly greater than R ($50k\Omega$). The output of the D/A converter is connected to the V_{in} node of a Howland voltage-controlled current source.[23] The Howland circuit was chosen because the output current is nearly ideal over a wide range of loading. This is required because the current-sensing transistors in the threshold detectors will have a variable resistance as they switch in and out of saturation.

2. Current Decoding at Outputs

The output circuitry has three stages, a current-controlled voltage source (ICVS), a series of voltage comparators, and a digital logic LED display circuit. The output circuit is specifically designed to detect the proper switching thresholds designed into the modulo four adder. The output circuitry is shown in figure 4.2.

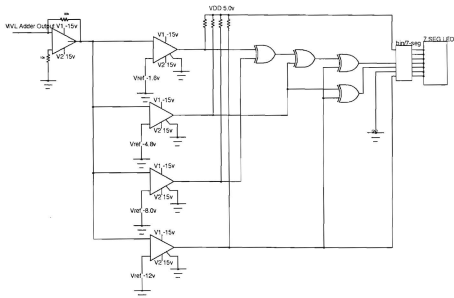


Figure 4.2: Testbench Output Circuitry

It should be noted that this testbench is useful for functional testing and transient analysis. For transient testing, the dip switch input is changed to a clocked voltage waveform from a function generator.

B. SUMMARY OF TEST RESULTS

1. Static Power Tests

Static power test results for no-load and full-load conditions were found to be slightly higher than those obtained from SPICE simulation of the layout. As previously mentioned, VDD and Ground inputs were provided for each of the major components of the device. This allowed static power testing of each component individually. The following table shows the power consumption of the major components of the third device in the test lot:

TABLE 4.1: Component Static Power

MVL_adder Device 3		
	Design No Load	Measured No Load
Sum (right tower)	2.93mA	3.12mA
Sum (left tower)	2.88mA	3.02mA
Carry out tower	2.95mA	3.23mA
Current mirror	0	0.0mA
Pad ring	0	0.0mA

The four test devices were measured under no load conditions (all inputs set to 0uA). As VDD was ramped from 0 to 5.0v, the power supply current was recorded. The static power consumption was calculated by finding an average current for the four devices and then multiplying by a constant VDD of 5.0v.

$$I_{ave} = (I_{dev1} + I_{dev2} + I_{dev3} + I_{dev4}) / 4 \quad (4.1)$$

$$P_{ave} = I_{ave} * VDD \quad (4.2)$$

The resulting average static power under no load conditions is plotted as a function of VDD in Figure 4.5.

The design power consumption model was obtained using SPICE for no-load conditions and VDD was once again ramped from 0 to 5.0v, and the current through the power supply was obtained. The average static power under no-load conditions was calculated as previously described:

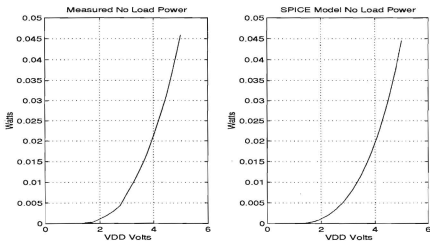


Figure 4.3: Measured and SPICE Simulation of No Load Static Power Consumption

For full load testing, all inputs were held high ($120\mu\text{A}$ each) and VDD was set to 5.0v. Once again, the power supply current was recorded for each test device. The measured values are shown compared with the full power simulation obtained from the SPICE model in Table 4.2

TABLE 4.2: Full Power Measurements

	MVL_adder Fab ID: N46EFF1		Spice Model (VDD set 5.0v)	
	Measured Current	Calculated Power	VDD Current	VDD Power
Device 1	70.5mA	352.5mW	66mA	330mW
Device 2	69.3mA	346.5mW		
Device 3	70.2mA	351.0mW		
Device 4	69.3mA	346.5mW		

The highest current level at the outputs occurs when the sum output transitions from a logic 2 to a logic 3, and vice versa. The measured value was determined by using a ramped current input from 0 to 150 μ A and observing the transitions at the sum terminal. A transient current spike of 195 μ A was recorded. This effect was seen for transitions in both directions. This transient result correlates with the SPICE model which showed similar transients of 195 μ A for these transitions(Appendix L). This result translates into a peak power rating for the device of 353.63mW, assuming all three inputs experience a simultaneous transition from logic 2 to logic 3.

2. Functional Testing

Steady-state functional values for output currents were within 1.5% of design, and most were found to be significantly less than 1.0%. Exhaustive functional logic testing was conducted (64 input combinations), and no deviation from predicted output values was observed in steady-state. The following table is a sample set from the input test on dev1. Shown are the design output values from the SPICE model vs. actual measured currents observed on the testbench.

TABLE 4.3: Functional Test for Device 1

[Xin, Yin, Cin]	Sum Output		Carry Output	
	design (mA)	measured	design (mA)	measured
[1,1,0]	80	82.2	0	0
[2,1,0]	120	119.3	0	0
[1,3,0]	0	0	40	41.3
[2,0,1]	120	119.7	0	0
[2,1,2]	40	41.2	40	40.3
[3,0,1]	0	0	40	40.9
[3,2,1]	80	80.9	40	40.0
[0,3,2]	40	40.3	40	40.3
[3,3,2]	0	0	80	79.2
[0,2,2]	0	0	40	40.2

3. Transient Analysis

A transient analysis was conducted. However, due to the very small currents being measured, accurate rise/fall times and propagation delays were difficult to obtain. The device was found to be able to maintain correct functionality at clocking rates of up to 600Khz, at a power supply voltage of 5.0v. In order to measure the output response of the adder, the sum and carry out nodes were connected through a 1Kohm resistor to ground. This leads to errors due to the introduction of additional resistive loading on the overall RC delay in the circuit. The longest delay times were observed in the sum circuit. When switching between logic level 0 and 3, (0 μ A and 120 μ A respectively) the sum circuit was found capable of clock rates not greater than approximately 500Khz. The measured values were determined from the scope displays as seen in appendix K.

Design delays were predicted using the extracted SPICE model of the adder. Customized pulsed current waveforms were provided to the input nodes. The output waveforms for the sum and carry out were obtained with a 1Kohm resistive load. This was done in order to gain a quantitative result which could be compared to the measured values. Measured and simulated delays are given in Table 4.4.

TABLE 4.4: Propagation Delay

	Sum 0->3->0		Carry 0->1->0	
	measured	SPICE	measured	SPICE
t_r	280nS	60nS	223nS	10nS
t_f	254nS	110nS	215nS	10nS
t_{dr}	135nS	90nS	170nS	120nS
t_{df}	120nS	55nS	73nS	40nS

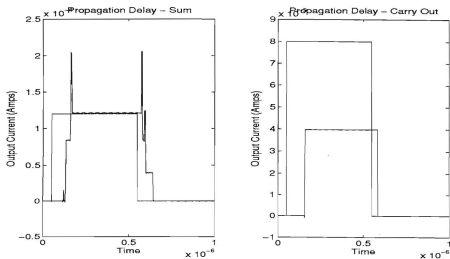


Figure 4.4: SPICE Model Propagation Delay for Sum and Carry Out

4. Results Discussion

Steady-state functional operation conformed very closely to design and simulation. Output currents were on average within 1 percent of ideal operation for V_{dd} set to 5.0v. This is important due to the fact that these devices are designed to operate in both parallel (carry save) and serial (ripple) adder configurations. Static power consumption for no load, full load, and peak power were very close to design values.

The timing measurements were difficult to obtain. The output currents were converted to voltage signals across a 1Kohm resistor, inherently increasing propagation delays, especially rise/fall times. Measured propagation delays fell between a low of 2 and high of 20 times larger than the simulated values. In this case, the measured values are open to a certain degree of speculation for accuracy. Attempts to use smaller resistances failed to produce a voltage signal strong enough to be distinguishable from background noise. With such small measurable signals, the inherent capacitance in the testing boards and connections proved to be significant.

V. CONCLUSIONS

HAMLET was successfully utilized in the design phase of a radix-4 adder cell. Minimization heuristics correctly produced a set of sum-of-product expressions for the sum and the carry out function, which, when implemented and tested, correctly computed the desired functions.

A. HAMLET RECOMMENDATIONS

The following recommended actions are applicable to the HAMLET module `mvll` which generates a pla representation of a multiple-valued logic expression.

1. Port to an X-Windows Application

The original versions of `mvll` were platform specific, i.e., written specifically to operate only on ISIS and VAX workstations. A working version of this program is required in order to effect further upgrades. Since MAGIC is an X-Windows application, the pla generator needs to incorporate the appropriate coordinate-free libraries (.cfl files) that contain the drawing commands used in the versions of MAGIC which are currently being utilized.

2. Design of Current-Mode Logic Cells

As the MOSIS design rules change, and as more precise fabrication processes are made available for these devices to be implemented, the basic current-mode cells used by `mvll` (step up/down and column generators) need to be redesigned for better noise margin and propagation delay performance. Smaller resolution implementation of logic values results in less power consumption, or, the capability to design and implement higher radix devices.

B. AN ALTERNATIVE DESIGN FOR A RADIX-4 ADDER CELL IN CURRENT-MODE CMOS

As a reference for comparison purposes, a current mode CMOS radix 4 carry save adder was designed and implemented without use of the cad tool HAMLET. The goal of this custom design was to produce and implement a device which could then be compared to that which was created using HAMLET.

1. A Different Approach

In attempting to design the same radix-4 adder on a smaller device, the fact that an adder is a **symmetric function** can be taken advantage of. Recall that the value of a binary symmetric function depends only on the **total** number of inputs which are 1. This same principle applies to multiple-valued functions as well. In the current-mode adder, for example, both the sum and the carry functions can be determined solely on the **wired sum** of all inputs.

Determining the output function values based upon the wired sum of the inputs has several advantages. Foremost, the complexity of the logic implementation is greatly reduced. In the case of the adder, for example, instead of requiring the replication of all three inputs, there is now only a single input to the column generator terms. Furthermore, the terms themselves contain at most one step-up and one step-down generator, vice up to 6 required generators when using all 3 inputs in one term.

To accomplish this, the method is to use a **hierarchical** radix model for the realization of the function. A lower radix is experienced at the input and output nodes of the device. However, inside the device, the circuit uses a system in which the radix is determined by the wired sum of the inputs, as given by equation 5.1.

$$RADIX_{HIGH} = \prod_{k=1}^n (RADIX_{LOW} - 1) \quad (5.1)$$

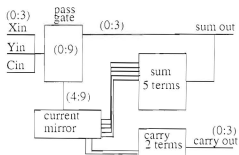


Figure 5.2: Block Diagram of New Modulo 4 Adder

The final design contains far fewer transistors than the previous device. The die size of the adder designed with HAMLET measured 5.5 X 3.5 mm, compared with the die layout of the alternative design which is MOSIS standard size **Tiny** measuring 2.22 X 2.25mm. As can be seen in figure 5.3, the tiny size die contains four complete radix 4 adders, and easily has enough room to accommodate up to 6.

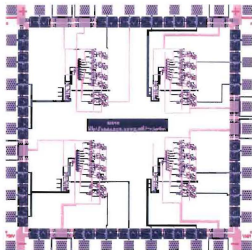


Figure 5.3: Chip Layout of Alternative Design

2. Simulation of Alternative Design

The final layout of the new device was extracted from MAGIC into a SPICE model and simulated. The same functional test inputs were applied to the input nodes as those used for the HAMLET design. All input combinations demonstrated correct function results for the adder.

The static full power analysis illustrated the benefit of both fewer terms and lower ideal current logic values. A single adder cell on this chip consumes only 12.2mW vice the 330mW of the HAMLET design. Similar results were encountered for a no load test. This power savings, however, is realized at the expense of further degradation in transient performance. A transient analysis was conducted on the device layout using the SPICE model. Once again, the input functions used were the same as used for the previous design.

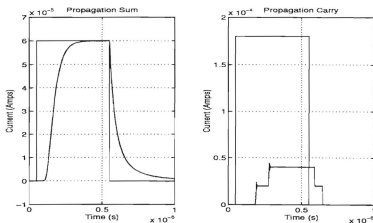


Figure 5.4: Transient Analysis of Alternate Design

Figure 5.4 shows rise/fall and propagation delay on the same order as the previous design. This is due to two factors. By halving the current values used to realize this function, this layout is most susceptible to RC delay of the interconnect network. The other contributing factor is that conservative metal 1 and metal 2 widths were used in order to

ensure safety during testing and operation. An inexpensive venture to begin to alleviate the propagation delay would be to simply return to the layout and re-draw the interconnect more aggressively, with emphasis on minimum width wire and routing distances.

3. Conclusions of Alternative Design Method

The new design accomplished the same function with a total of 7 column generator terms as opposed to the 47 required by the SOP expressions generated by HAMLET. In addition, each of the 7 terms in the new design has at most 2 threshold detector cells vice the 6 needed in some column generator terms of the previous device. With fewer transistors, the device itself requires much less space to layout, on the average about a single order of magnitude smaller than the first chip. The smaller design, as expected, also used about an order of magnitude less power under full load. The alternate design did not help the slow transient performance of the modulo four adder design in current-mode CMOS.

C. RECOMMENDATIONS FOR FURTHER STUDY

1. Charged-Coupled Device (CCD) Logics

A programmable logic array implementation using CCDs is an appropriate evolutionary step for the HAMLET project. CCDs have been found to be useful in the design of memory units. Hitachi has implemented a 16 valued memory. Although multiple valued logic CCD is slower than CMOS, it is much more dense. The use of MVL CCDs can increase storage capacity significantly, perhaps replacing the disk[20].

2. Resonant-Tunneling Diodes (RTD) Logics

Quantum resonant tunneling devices offer the highest speed performance for multi valued logic implementation to date. At present, RTDs exist primarily as discrete devices, hence, RTD VLSI implementation and modeling is in the future. RTDs will produce extremely simple and high speed A/D and D/A converters which are also a significant part of the MVL VLSI implementation problem.

3. Hierarchical MVL Design for Symmetric Functions

A generalized formal discussion of the use of multiple radices (at least two) in the implementation of a totally symmetric multiple valued logic function is desirable. One possible approach is to consider the shortest path problem which arises in graph theory. In a dense graph of many nodes, a guaranteed shortest path solution algorithm, whether breadth first or depth first, rapidly becomes exceedingly expensive as the graph scales. However, if a single large graph (flat topology) is carved into a network of subgraphs (hierarchical topology) the shortest path computation is greatly reduced.

This is similar to the two-radix approach of the second design. The input radix (mod 4) is immediately converted to mod 10 (upper level hierarchy). The function value is computed in base 10, and then the result is converted to mod 4 by the column output generators. An interesting result would be to compute the power and transistor budget savings using this method.

4. Modulo 16 (hexadecimal) Current Mode CMOS Full Adder Cell

Using the same basic components as in the latter mod 4 adder design described above, it is possible to construct a radix 16 full adder cell with approximately the same die area and power consumption. Such a cell is currently being investigated using as little as $10\mu\text{A}$ for each logic level. In the near future the MOSIS design rules currently in effect will allow sub-micron design and layout. This will greatly enhance the ability to implement the required threshold detectors for such a small proposed signal space. As in the previous modulo 4 adder cell, a current-sensing complementary transmission gate passes the first 15 levels of logic signal directly to the sum terminal, without further circuit interaction. For inputs totaling 16 or more, the wired sum signal is passed to the sum and carry logic for processing.

REFERENCES

- [1] J. M. Yurchak and J. T. Butler, "HAMLET - An Expression Compiler/Optimizer for the Implementation of Heuristics to Minimize Multiple Valued Programmable Logic Arrays," *Proc. 20th ISMVL*, pp. 144-152.
- [2] K. C. Smith, Z. G. Vranesic, L. Janczewski, "Circuit Implementations of Multivalued Logic," *Proc. 1971 ISMVL*, 1971, pg. 133.
- [3] D. Etiemble, M. Israel, "Implementation of Ternary Circuits With Binary Integrated Circuits," *Proc. 7th ISMVL*, 1977, pp. 125-130.
- [4] K. W. Current, D. A. Mow, "Four-Valued Threshold Logic Full Adder Circuit Implementations," *Proc. 8th ISMVL*, 1978 pp95-100.
- [5] J. G. Tront, D. Givone, "An Implementation of Multiple-Valued Logic Gates Using MESFETs," *Proc. 9th ISMVL*, 1979, pp.175-177.
- [6] C. S. Wallace, "A Suggestion for a Fast Multiplier," *IEEE Trans. Elect Comp.*, Vol EC-13, No. 1, Feb 1964, pp.14-17.
- [7] S. Kawahito, M. Karneyama, T. Higuchi, H. Yamada, "A High-Speed Compact Multiplier Based on Multiple-Valued Bi-directional Current-Mode Circuits," *Proc. 17th ISMVL*, May 1987, pp.172-174.
- [8] A. D. Booth, "A Signed Binary Multiplication Technique," *Quart. J. Mech. Appl. Math.*, Vol 4, Part 2, 1951, pages 236-240.
- [9] S. Kawahito, M. Karneyama, T. Higuchi, H. Yamada, "VLSI-Oriented Bi-directional Current-Mode Arithmetic Circuits Based on the Radix-4 Signed Number System," *Proc. 16th ISMVL*, 1986, pp. 70-77.
- [10] Yong Ha Ko, "Design of Multi-Valued Programmable Logic Arrays," Master's Thesis, Naval Postgraduate School, December 1988, pp 3-7.
- [11] A Kazeminejad, K. Navi, and D. Etiemble, "CML Current Mode Full Adders for 2.5 Volt Power Supply," *Proc. 24th ISMVL*, May 1994, pp-10-14.
- [12] Adel S. Sedra and Kenneth C. Smith, *Microelectronic Circuits*, third edition, Saunders, Philadelphia, PA 1991, ch 3-4.
- [13] Y. H. Chang and J. T. Butler, "The Design of Current Mode CMOS Multiple-Valued Circuits," *Proc. 21st ISMVL*, May 1991, pp. 130-138.
- [14] K. W. Current, F. A. Edwards, and D. A. Freitas, "A CMOS Multiple Valued Logic Test Chip," *Proc. 17 ISMVL*, May 1987, pp. 16-19.

- [15] K. Wayne Current, "Multiple Valued Logic: Current-Mode CMOS Circuits," *Proc. 23rd ISMVL*, May 1993, pp.176-181.
- [16] K. W. Current, F. A. Edwards, and D. A. Freitas, "CMOS Quaternary Threshold Logic Full Adder Circuits," *Proc. 15th ISMVL*, May 1985, pp. 318-322.
- [17] A Kazeminejad, K. Navi, and D. Etiemble, "Performance of CMOS Current Mode Full Adders," *Proc. 24th ISMVL*, May 1994, pp.27-34.
- [18] J. T. Butler, "Multiple-valued Logic," *IEEE Potentials vol. 14*, April/May 1995, pp. 11.
- [19] K. Lei and Z.G. Vranesic, "On the Synthesis of 4-Valued Current Mode CMOS Circuits," *Proc. 21st ISMVL*, May 1991, pp. 147-155.
- [20] Konreid Lei and Zvonko G. Vranesic, "Towards the Realization of 4-Valued CMOS Circuits," *Proc. 22nd ISMVL*, May 1991, pp. 147-155.
- [21] S. P. Onneweer and H. G. Kerkhoff, "Current-Mode CMOS High-Radix Circuits," *Proc. 16th ISMVL*, May 1986, pp. 60-69.
- [22] W. S. Scott, R. N. Mayo, and G. Hamachi, "1986 VLSI Tools," Report No. UCN/CSD 86/272, University of California, Berkeley, December 1985.
- [23] William D. Stanley, *Operational Amplifiers with Linear Integrated Circuits*, third edition, Macmillan, New York, 1994, pp. 82-84, ch 10.
- [24] Neil H. E. Weste and Kamran Eshraghian, *Principles of VLSI Design*, second edition, Addison Wesley, Reading, MA, 1992, ch 4-6.

APPENDIX

A. HAMLET REPORT - ORIGINAL SOP EXPRESSION

Verification of original expression

19 february 1994

mvlc -M mvl_full_adder

4: 3:

```
+ 1 * x1(1,1) * x2(0,0) * x3(0,0)
+ 2 * x1(2,2) * x2(0,0) * x3(0,0)
+ 3 * x1(3,3) * x2(0,0) * x3(0,0)
+ 1 * x1(0,0) * x2(1,1) * x3(0,0)
+ 2 * x1(1,1) * x2(1,1) * x3(0,0)
+ 3 * x1(2,2) * x2(1,1) * x3(0,0)
+ 2 * x1(0,0) * x2(2,2) * x3(0,0)
+ 3 * x1(1,1) * x2(2,2) * x3(0,0)
+ 1 * x1(3,3) * x2(2,2) * x3(0,0)
+ 3 * x1(0,0) * x2(3,3) * x3(0,0)
+ 1 * x1(2,2) * x2(3,3) * x3(0,0)
+ 2 * x1(3,3) * x2(3,3) * x3(0,0)
+ 1 * x1(0,0) * x2(0,0) * x3(1,1)
+ 2 * x1(1,1) * x2(0,0) * x3(1,1)
+ 3 * x1(2,2) * x2(0,0) * x3(1,1)
+ 2 * x1(0,0) * x2(1,1) * x3(1,1)
+ 3 * x1(1,1) * x2(1,1) * x3(1,1)
+ 1 * x1(3,3) * x2(1,1) * x3(1,1)
+ 3 * x1(0,0) * x2(2,2) * x3(1,1)
+ 1 * x1(2,2) * x2(2,2) * x3(1,1)
+ 2 * x1(3,3) * x2(2,2) * x3(1,1)
+ 1 * x1(1,1) * x2(3,3) * x3(1,1)
+ 2 * x1(2,2) * x2(3,3) * x3(1,1)
+ 3 * x1(3,3) * x2(3,3) * x3(1,1)
+ 2 * x1(0,0) * x2(0,0) * x3(2,2)
+ 3 * x1(1,1) * x2(0,0) * x3(2,2)
+ 1 * x1(3,3) * x2(0,0) * x3(2,2)
+ 3 * x1(0,0) * x2(1,1) * x3(2,2)
+ 1 * x1(2,2) * x2(1,1) * x3(2,2)
+ 2 * x1(3,3) * x2(1,1) * x3(2,2)
+ 1 * x1(1,1) * x2(2,2) * x3(2,2)
+ 2 * x1(2,2) * x2(2,2) * x3(2,2)
+ 3 * x1(3,3) * x2(2,2) * x3(2,2)
+ 1 * x1(0,0) * x2(3,3) * x3(2,2)
+ 2 * x1(1,1) * x2(3,3) * x3(2,2)
+ 3 * x1(2,2) * x2(3,3) * x3(2,2)
+ 3 * x1(0,0) * x2(0,0) * x3(3,3)
+ 1 * x1(2,2) * x2(0,0) * x3(3,3)
+ 2 * x1(3,3) * x2(0,0) * x3(3,3)
+ 1 * x1(1,1) * x2(1,1) * x3(3,3)
+ 2 * x1(2,2) * x2(1,1) * x3(3,3)
+ 3 * x1(3,3) * x2(1,1) * x3(3,3)
```

$+ 1 * x1(0,0) * x2(2,2) * x3(3,3)$
 $+ 2 * x1(1,1) * x2(2,2) * x3(3,3)$
 $+ 3 * x1(2,2) * x2(2,2) * x3(3,3)$
 $+ 2 * x1(0,0) * x2(3,3) * x3(3,3)$
 $+ 3 * x1(1,1) * x2(3,3) * x3(3,3)$
 $+ 1 * x1(3,3) * x2(3,3) * x3(3,3);$

0 1 2 3.
 1 2 3. 0
 2 3. 0 1
 3. 0 1 2

1 2 3. 0
 2 3. 0 1
 3. 0 1 2
 0 1 2 3.

2 3. 0 1
 3. 0 1 2
 0 1 2 3.
 1 2 3. 0

3. 0 1 2
 0 1 2 3.
 1 2 3. 0
 2 3. 0 1

4: 3:

$+ 1 * x1(3,3) * x2(1,1) * x3(0,0)$
 $+ 1 * x1(1,1) * x2(3,3) * x3(0,0)$
 $+ 1 * x1(2,3) * x2(2,3) * x3(0,0)$
 $+ 1 * x1(3,3) * x2(0,1) * x3(1,1)$
 $+ 1 * x1(2,2) * x2(1,1) * x3(1,1)$
 $+ 1 * x1(1,1) * x2(2,2) * x3(1,1)$
 $+ 1 * x1(0,1) * x2(3,3) * x3(1,1)$
 $+ 1 * x1(2,3) * x2(2,3) * x3(1,1)$
 $+ 1 * x1(2,3) * x2(0,1) * x3(2,2)$
 $+ 1 * x1(1,1) * x2(1,1) * x3(2,2)$
 $+ 1 * x1(0,3) * x2(2,3) * x3(2,2)$
 $+ 1 * x1(3,3) * x2(3,3) * x3(2,2)$
 $+ 1 * x1(1,2) * x2(0,1) * x3(3,3)$
 $+ 1 * x1(0,0) * x2(1,1) * x3(3,3)$
 $+ 1 * x1(3,3) * x2(0,3) * x3(3,3)$
 $+ 1 * x1(0,3) * x2(2,3) * x3(3,3)$
 $+ 1 * x1(2,2) * x2(3,3) * x3(3,3);$

0 0 0 0
 0 0 0 1
 0 0 1 1
 0 1 1 1

0 0 0 1
0 0 1 1
0 1 1 1
1 1 1 1

0 0 1 1
0 1 1 1
1 1 1 1
1 1 1 2

0 1 1 1
1 1 1 1
1 1 1 2
1 1 2 2

B. HAMLET REPORT - DEUCK & MILLER MINIMIZATION

minimization of SOP expression using Deuck&Miller
and Proper&Armstrong heuristics.

19 february 1994

mvlc HG E 0.01 mvl_full_adder

The resulting expression is:

4:3:

```
+ 2 * x1(3,3) * x2(2,2) * x3(1,2)
+ 3 * x1(0,0) * x2(2,2) * x3(1,1)
+ 1 * x1(3,3) * x2(0,2) * x3(2,2)
+ 3 * x1(0,0) * x2(3,3) * x3(0,0)
+ 1 * x1(2,3) * x2(1,1) * x3(2,3)
+ 1 * x1(2,2) * x2(0,2) * x3(3,3)
+ 3 * x1(3,3) * x2(0,0) * x3(0,0)
+ 1 * x1(3,3) * x2(3,3) * x3(3,3)
+ 3 * x1(1,1) * x2(1,1) * x3(1,1)
+ 1 * x1(0,0) * x2(0,0) * x3(1,3)
+ 3 * x1(0,0) * x2(1,1) * x3(2,2)
+ 2 * x1(3,3) * x2(3,3) * x3(0,1)
+ 1 * x1(3,3) * x2(1,1) * x3(1,1)
+ 1 * x1(0,1) * x2(3,3) * x3(2,3)
+ 3 * x1(0,0) * x2(0,0) * x3(3,3)
+ 2 * x1(2,2) * x2(0,0) * x3(0,1)
+ 1 * x1(0,0) * x2(0,0) * x3(2,3)
+ 1 * x1(1,3) * x2(3,3) * x3(1,1)
+ 1 * x1(0,1) * x2(2,3) * x3(3,3)
+ 2 * x1(0,0) * x2(1,1) * x3(1,2)
+ 1 * x1(1,1) * x2(1,3) * x3(3,3)
+ 2 * x1(2,2) * x2(2,2) * x3(2,3)
+ 2 * x1(1,2) * x2(0,0) * x3(1,1)
+ 1 * x1(2,2) * x2(3,3) * x3(0,2)
+ 2 * x1(0,0) * x2(2,2) * x3(0,1)
+ 1 * x1(0,2) * x2(1,1) * x3(0,0)
+ 2 * x1(3,3) * x2(0,1) * x3(3,3)
+ 1 * x1(3,3) * x2(2,2) * x3(0,0)
+ 1 * x1(1,1) * x2(2,3) * x3(2,2)
+ 1 * x1(1,1) * x2(0,2) * x3(0,0)
+ 3 * x1(2,2) * x2(1,1) * x3(0,0)
+ 3 * x1(2,2) * x2(3,3) * x3(2,2)
+ 3 * x1(1,1) * x2(2,2) * x3(0,0)
+ 3 * x1(1,1) * x2(0,0) * x3(2,2)
+ 1 * x1(2,2) * x2(2,2) * x3(1,1);
```

Case: 1 User: 48

Heur: Gold(D&M) Perf: 32

The resulting expression is:

4:3:

+ 1 * x1(0,3) * x2(3,3) * x3(3,3)
+ 1 * x1(2,3) * x2(1,3) * x3(1,3)
+ 1 * x1(0,1) * x2(3,3) * x3(1,2)
+ 1 * x1(3,3) * x2(2,2) * x3(3,3)
+ 1 * x1(2,3) * x2(0,0) * x3(2,3)
+ 1 * x1(3,3) * x2(1,3) * x3(0,0)
+ 1 * x1(3,3) * x2(0,0) * x3(1,1)
+ 1 * x1(0,1) * x2(2,2) * x3(2,3)
+ 1 * x1(1,1) * x2(0,1) * x3(3,3)
+ 1 * x1(0,0) * x2(1,1) * x3(3,3)
+ 1 * x1(2,2) * x2(2,3) * x3(0,0)
+ 1 * x1(1,1) * x2(1,1) * x3(2,2)
+ 1 * x1(3,3) * x2(3,3) * x3(2,2)
+ 1 * x1(1,1) * x2(3,3) * x3(0,0)
+ 1 * x1(1,1) * x2(2,2) * x3(1,1);

C. HAMLET REPORT - SIMULATED ANNEALING MINIMIZATION

Simulated annealing optimization 19 february 1994

mvlc -HSA -E -0.01 mvl_full_adder

Case: 1 User: 48

Heur: SA Perf: 32

resulting expressions:

4: 3:

```
+ 1 * x1(3,3) * x2(0,1) * x3(3,3)
+ 1 * x1(3,3) * x2(0,2) * x3(2,2)
+ 2 * x1(0,1) * x2(0,0) * x3(2,2)
+ 1 * x1(2,3) * x2(3,3) * x3(0,0)
+ 1 * x1(1,1) * x2(0,0) * x3(0,2)
+ 2 * x1(0,0) * x2(2,2) * x3(1,1)
+ 1 * x1(2,3) * x2(1,2) * x3(2,2)
+ 1 * x1(1,3) * x2(3,3) * x3(1,1)
+ 1 * x1(2,3) * x2(0,1) * x3(3,3)
+ 3 * x1(0,0) * x2(1,1) * x3(2,2)
+ 3 * x1(2,2) * x2(2,2) * x3(3,3)
+ 1 * x1(0,1) * x2(2,3) * x3(3,3)
+ 3 * x1(3,3) * x2(0,0) * x3(0,0)
+ 3 * x1(2,2) * x2(0,0) * x3(1,1)
+ 1 * x1(0,0) * x2(2,3) * x3(0,0)
+ 1 * x1(2,3) * x2(2,3) * x3(1,1)
+ 1 * x1(3,3) * x2(2,3) * x3(0,0)
+ 3 * x1(2,2) * x2(1,1) * x3(0,0)
+ 1 * x1(0,1) * x2(3,3) * x3(2,3)
+ 2 * x1(0,0) * x2(3,3) * x3(0,0)
+ 1 * x1(3,3) * x2(3,3) * x3(3,3)
+ 1 * x1(3,3) * x2(1,3) * x3(1,1)
+ 3 * x1(1,1) * x2(2,2) * x3(0,0)
+ 1 * x1(1,1) * x2(2,3) * x3(2,3)
+ 2 * x1(2,2) * x2(0,0) * x3(0,0)
+ 2 * x1(1,1) * x2(1,1) * x3(0,1)
+ 3 * x1(0,0) * x2(0,0) * x3(3,3)
+ 3 * x1(2,2) * x2(3,3) * x3(2,2)
+ 1 * x1(1,3) * x2(1,1) * x3(3,3)
+ 1 * x1(0,1) * x2(0,1) * x3(1,1)
+ 1 * x1(0,0) * x2(1,2) * x3(0,1)
+ 1 * x1(2,3) * x2(2,2) * x3(2,2);
```

Case: 2 User: 17

Heur: SA Perf: 15

4: 3:

```
+ 1 * x1(2,2) * x2(0,1) * x3(2,3)
+ 1 * x1(1,1) * x2(2,2) * x3(1,1)
+ 1 * x1(0,2) * x2(2,2) * x3(2,2)
```

+ 1 * x1(3,3) * x2(1,3) * x3(0,0)
+ 1 * x1(0,0) * x2(1,3) * x3(3,3)
+ 1 * x1(3,3) * x2(0,3) * x3(1,3)
+ 1 * x1(2,2) * x2(2,2) * x3(0,1)
+ 1 * x1(1,1) * x2(0,1) * x3(3,3)
+ 1 * x1(2,2) * x2(1,1) * x3(1,1)
+ 1 * x1(0,2) * x2(3,3) * x3(1,2)
+ 1 * x1(3,3) * x2(3,3) * x3(2,2)
+ 1 * x1(1,2) * x2(3,3) * x3(0,0)
+ 1 * x1(1,1) * x2(1,1) * x3(2,2)
+ 1 * x1(1,3) * x2(2,3) * x3(3,3)
+ 1 * x1(2,2) * x2(3,3) * x3(3,3);

D. SPICE FILE - STEP UP GENERATOR

```
** SPICE file created for circuit step_up2.gen
** Technology: scmos
.MODEL nfet NMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=1
+ VTO=0.8673 DELTA=4.9450E+00 LD=3.5223E-07 KP=4.6728E-05
+ UO=569.7 UEXP=1.7090E-01 UCRT=5.9350E+04 RSH=1.9090E+01
+ GAMMA=0.4655 NSUB=4.3910E+15 NFS=1.980E+11 VMAX=5.7510E+04
+ LAMBDA=3.9720E-02 CGDO=4.3332E-10 CGSO=4.3332E-10
+ CGBO=3.5977E-10 CJ=1.0096E-04 MJ=0.8119 CJSW=4.6983E-10
+ MJSW=0.323107 PB=0.800000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is -9.0180E-08
.MODEL pfet PMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=-1
+ VTO=-0.9506 DELTA=4.5950E+00 LD=3.7200E-07 KP=1.6454E-05
+ UO=200.6 UEXP=2.6690E-01 UCRT=7.9260E+04 RSH=4.9920E+01
+ GAMMA=0.6561 NSUB=8.7250E+15 NFS=3.27E+11 VMAX=9.9990E+05
+ LAMBDA=4.5950E-02 CGDO=4.5769E-10 CGSO=4.5769E-10
+ CGBO=3.8123E-10 CJ=3.1469E-04 MJ=0.5687 CJSW=3.1456E-10
+ MJSW=0.275802 PB=0.800000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is -2.2400E-07

.TRAN 1us 100us

Vdd 1 0 5
VD1 100 0 5
Vgnd 105 0 0

*Input:
Vi 42 104
Iin 0 42 PWL(0us 0uA 100us 100uA)

Vout 107 0 0

M0 100 101 101 1 pfet L=2.0U W=7.0U
M1 100 101 102 1 pfet L=2.0U W=7.0U
M2 100 102 103 1 pfet L=2.0U W=3.0U
M3 104 104 105 0 nfet L=2.0U W=6.0U
M4 101 104 105 0 nfet L=2.0U W=6.0U
M5 100 103 106 1 pfet L=2.0U W=3.0U
M6 102 102 105 0 nfet L=3.0U W=6.0U
M7 103 102 105 0 nfet L=2.0U W=3.0U
M8 106 106 107 0 nfet L=18.0U W=3.0U
M9 106 103 105 0 nfet L=2.0U W=3.0U
C0 107 0 13F
** NODE: 107 = STEP_UP_OUT
C1 106 0 47F
** NODE: 106 = 8_30_45#
```

```
C2 105 0.67F
** NODE: 105 = GND
C3 103 0.33F
** NODE: 103 = 8_2_21#
C4 104 0.21F
** NODE: 104 = STEPIn
C5 102 0.43F
** NODE: 102 = 8_41_25#
C6 101 0.41F
** NODE: 101 = 8_89_48#
C7 100 0.54F
** NODE: 100 = Vdd
** NODE: 0 = GND!
** NODE: 1 = Vdd!
```

E. SPICE FILE - STEP DOWN GENERATOR

```
** SPICE file created for circuit step_down1.gen
** Technology: scmos
.MODEL nfet NMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=1
+ VTO=0.8673 DELTA=4.9450E+00 LD=3.5223E-07 KP=4.6728E-05
+ UO=569.7 UEXP=1.7090E-01 UCRIT=5.9350E+04 RSH=1.9090E+01
+ GAMMA=0.4655 NSUB=4.3910E+15 NFS=1.980E+11 VMAX=5.7510E+04
+ LAMBDA=3.9720E-02 CGDO=4.3332E-10 CGSO=4.3332E-10
+ CGBO=3.5977E-10 CJ=1.0096E-04 MJ=0.8119 CJSW=4.6983E-10
+ MJSW=0.323107 PB=0.800000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is -9.0180E-08
.MODEL pfet PMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=-1
+ VTO=-0.9506 DELTA=4.5950E+00 LD=3.7200E-07 KP=1.6454E-05
+ UO=200.6 UEXP=2.6690E-01 UCRIT=7.9260E+04 RSH=4.9920E+01
+ GAMMA=0.6561 NSUB=8.7250E+15 NFS=3.27E+11 VMAX=9.9990E+05
+ LAMBDA=4.5950E-02 CGDO=4.5769E-10 CGSO=4.5769E-10
+ CGBO=3.8123E-10 CJ=3.1469E-04 MJ=0.5687 CJSW=3.1456E-10
+ MJSW=0.275802 PB=0.800000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is -2.2400E-07

.TRAN 1us 100us

Vdd 1 0 5
VD1 100 0 5
Vgnd 105 0 0

*Input:
Vi 42 104
Iin 0 42 PWL(0us 0uA 100us 100uA)

Vout 107 0 0

** NODE: 2 = Error
M0 100 101 101 1 pfet L=2.0U W=7.0U
M1 100 101 102 1 pfet L=2.0U W=7.0U
M2 100 102 103 1 pfet L=2.0U W=3.0U
M3 104 104 105 0 nfet L=2.0U W=6.0U
M4 101 104 105 0 nfet L=2.0U W=6.0U
M5 100 103 106 1 pfet L=2.0U W=3.0U
M6 102 102 105 0 nfet L=8.0U W=7.0U
M7 103 102 105 0 nfet L=2.0U W=3.0U
M8 100 106 107 1 pfet L=9.0U W=3.0U
M9 106 103 105 0 nfet L=2.0U W=3.0U
C0 107 0 11F
** NODE: 107 = STEP_DOWNout
C1 106 0 30F
```

```
** NODE: 106 = 8_30_45#  
C2 105 0 68F  
** NODE: 105 = GND  
C3 103 0 33F  
** NODE: 103 = 8_2_21#  
C4 104 0 21F  
** NODE: 104 = STEPin  
C5 102 0 44F  
** NODE: 102 = 8_41_31#  
C6 101 0 41F  
** NODE: 101 = 8_89_48#  
C7 100 0 63F  
** NODE: 100 = Vdd  
** NODE: 0 = GND!  
** NODE: 1 = Vdd!
```


F. SPICE FILE - COLUMN OUTPUT GENERATOR

```
** SPICE file created for circuit column_gen1
** Technology: scmos
.MODEL nfet NMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=1
+ VTO=0.8673 DELTA=4.9450E+00 LD=3.5223E-07 KP=4.6728E-05
+ UO=569.7 UEXP=1.7090E-01 UCRIT=5.9350E+04 RSH=1.9090E+01
+ GAMMA=0.4655 NSUB=4.3910E+15 NFS=1.980E+11 VMAX=5.7510E+04
+ LAMBDA=3.9720E-02 CGDO=4.3332E-10 CGSO=4.3332E-10
+ CGBO=3.5977E-10 CJ=1.0096E-04 MJ=0.8119 CJSW=4.6983E-10
+ MJSW=0.323107 PB=0.800000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is -9.0180E-08
.MODEL pfet PMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=-1
+ VTO=-0.9506 DELTA=4.5950E+00 LD=3.7200E-07 KP=1.6454E-05
+ UO=200.6 UEXP=2.6690E-01 UCRIT=7.9260E+04 RSH=4.9920E+01
+ GAMMA=0.6561 NSUB=8.7250E+15 NFS=3.27E+11 VMAX=9.9990E+05
+ LAMBDA=4.5950E-02 CGDO=4.5769E-10 CGSO=4.5769E-10
+ CGBO=3.8123E-10 CJ=3.1469E-04 MJ=0.5687 CJSW=3.1456E-10
+ MJSW=0.275802 PB=0.800000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is -2.2400E-07

.TRAN 1us 100us

Vdd 1 0 5
VD1 100 0 5
Vgnd 105 0 0

*Inputs:
Vi 42 104
Iin 0 42 PWL(0us 0uA 100us 100uA)

Vout 107 0 0

M0 100 101 101 1 pfet L=2.0U W=7.0U
M1 100 101 102 1 pfet L=2.0U W=7.0U
M2 100 102 103 1 pfet L=2.0U W=3.0U
M3 104 104 105 0 nfet L=2.0U W=6.0U
M4 101 104 105 0 nfet L=2.0U W=6.0U
M5 100 103 106 1 pfet L=2.0U W=3.0U
M6 102 102 105 0 nfet L=6.0U W=3.0U
M7 103 102 105 0 nfet L=2.0U W=3.0U
M8 100 106 107 1 pfet L=7.0U W=3.0U
M9 106 103 105 0 nfet L=2.0U W=3.0U
M10 107 107 105 1 pfet L=2.0U W=3.0U
C0 107 0 15F
** NODE: 107 = COLUMN_GENout
C1 106 0 30F
```

```
** NODE: 106 = 8_30_45#  
C2 105 0 75F  
** NODE: 105 = GND  
C3 103 0 33F  
** NODE: 103 = 8_2_21#  
C4 104 0 21F  
** NODE: 104 = COL_GENin  
C5 102 0 43F  
** NODE: 102 = 8_45_23#  
C6 101 0 41F  
** NODE: 101 = 8_89_48#  
C7 100 0 67F  
** NODE: 100 = Vdd  
** NODE: 1 = Vdd!  
** NODE: 0 = GND!
```

G. SPICE FILE - TERM 3*X(2,2)Y(3,3)C(2,2)

```
** SPICE file created for circuit term_3X2_2Y3_3C2_2
** Technology: scmos
.MODEL nfet NMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=1
+ VTO=0.8673 DELTA=4.9450E+00 LD=3.5223E-07 KP=4.6728E-05
+ UO=569.7 UEXP=1.7090E-01 UCRIT=5.9350E+04 RSH=1.9090E+01
+ GAMMA=0.4655 NSUB=4.3910E+15 NFS=1.980E+11 VMAX=5.7510E+04
+ LAMBDA=3.9720E-02 CGDO=4.3332E-10 CGSO=4.3332E-10
+ CGBO=3.5977E-10 CJ=1.0096E-04 MJ=0.8119 CJSW=4.6983E-10
+ MJSW=0.323107 PB=0.800000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is -9.0180E-08
.MODEL pfet PMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=-1
+ VTO=-0.9506 DELTA=4.5950E+00 LD=3.7200E-07 KP=1.6454E-05
+ UO=200.6 UEXP=2.6690E-01 UCRIT=7.9260E+04 RSH=4.9920E+01
+ GAMMA=0.6561 NSUB=8.7250E+15 NFS=3.27E+11 VMAX=9.9990E+05
+ LAMBDA=4.5950E-02 CGDO=4.5769E-10 CGSO=4.5769E-10
+ CGBO=3.8123E-10 CJ=3.1469E-04 MJ=0.5687 CJSW=3.1456E-10
+ MJSW=0.275802 PB=0.800000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is -2.2400E-07

.TRAN 1us 100us

Vdd 1 0 5
VD1 100 0 5
Vgnd 105 0 0

* Inputs:
Vix 40 109
Iinx 0 40 PWL(0us 0uA 100us 100uA)
Viy 41 114
Iiny 0 41 PWL(0us 0uA 100us 100uA)
Vic 42 122
Iinc 0 42 PWL(0us 0uA 100us 100uA)

* Column Generator Output Node for this term:
Vout 128 0 0

M0 100 101 102 1 pfet L=2.0U W=7.0U
M1 100 102 103 1 pfet L=2.0U W=3.0U
M2 100 103 104 1 pfet L=2.0U W=3.0U
M3 102 102 105 0 nfet L=3.0U W=6.0U
M4 103 102 105 0 nfet L=2.0U W=3.0U
M5 100 104 106 1 pfet L=9.0U W=3.0U
M6 104 103 105 0 nfet L=2.0U W=3.0U
M7 100 101 101 1 pfet L=2.0U W=7.0U
M8 100 101 107 1 pfet L=2.0U W=7.0U
```

M9 100 107 108 1 pfet L=2.0U W=3.0U
M10 109 109 105 0 nfet L=2.0U W=6.0U
M11 101 109 105 0 nfet L=2.0U W=6.0U
M12 100 108 110 1 pfet L=2.0U W=3.0U
M13 107 107 105 0 nfet L=3.0U W=9.0U
M14 108 107 105 0 nfet L=2.0U W=3.0U
M15 110 110 106 0 nfet L=18.0U W=3.0U
M16 110 108 105 0 nfet L=2.0U W=3.0U
M17 100 111 111 1 pfet L=2.0U W=7.0U
M18 100 111 112 1 pfet L=2.0U W=7.0U
M19 100 112 113 1 pfet L=2.0U W=3.0U
M20 114 114 105 0 nfet L=2.0U W=6.0U
M21 111 114 105 0 nfet L=2.0U W=6.0U
M22 100 113 115 1 pfet L=2.0U W=3.0U
M23 112 112 105 0 nfet L=3.0U W=9.0U
M24 113 112 105 0 nfet L=2.0U W=3.0U
M25 100 115 106 1 pfet L=9.0U W=3.0U
M26 115 113 105 0 nfet L=2.0U W=3.0U
M27 100 116 117 1 pfet L=2.0U W=7.0U
M28 100 117 118 1 pfet L=2.0U W=3.0U
M29 100 118 119 1 pfet L=2.0U W=3.0U
M30 117 117 105 0 nfet L=3.0U W=6.0U
M31 118 117 105 0 nfet L=2.0U W=3.0U
M32 100 119 106 1 pfet L=9.0U W=3.0U
M33 119 118 105 0 nfet L=2.0U W=3.0U
M34 100 116 116 1 pfet L=2.0U W=7.0U
M35 100 116 120 1 pfet L=2.0U W=7.0U
M36 100 120 121 1 pfet L=2.0U W=3.0U
M37 122 122 105 0 nfet L=2.0U W=6.0U
M38 116 122 105 0 nfet L=2.0U W=6.0U
M39 100 121 123 1 pfet L=2.0U W=3.0U
M40 120 120 105 0 nfet L=3.0U W=9.0U
M41 121 120 105 0 nfet L=2.0U W=3.0U
M42 123 123 106 0 nfet L=18.0U W=3.0U
M43 100 124 124 1 pfet L=2.0U W=7.0U
M44 100 124 125 1 pfet L=2.0U W=7.0U
M45 123 121 105 0 nfet L=2.0U W=3.0U
M46 100 125 126 1 pfet L=2.0U W=3.0U
M47 106 106 105 0 nfet L=2.0U W=6.0U
M48 124 106 105 0 nfet L=2.0U W=6.0U
M49 100 126 127 1 pfet L=2.0U W=3.0U
M50 125 125 105 0 nfet L=6.0U W=3.0U
M51 126 125 105 0 nfet L=2.0U W=3.0U
M52 100 127 128 1 pfet L=6.0U W=7.0U
M53 127 126 105 0 nfet L=2.0U W=3.0U
M54 128 128 105 1 pfet L=2.0U W=3.0U
C0 128 0 23F

** NODE: 128 = COLUMN_GENout
C1 127 0 29F

** NODE: 127 = 8_338_509#
C2 126 0 33F
** NODE: 126 = 8_310_485#
C3 125 0 43F
** NODE: 125 = 8_264_487#
C4 124 0 41F
** NODE: 124 = 8_220_417#
C5 123 0 47F
** NODE: 123 = 8_134_449#
C6 121 0 33F
** NODE: 121 = 8_106_425#
C7 122 0 21F
** NODE: 122 = Cin
C8 120 0 48F
** NODE: 120 = 8_58_431#
C9 119 0 30F
** NODE: 119 = 8_134_331#
C10 118 0 33F
** NODE: 118 = 8_106_307#
C11 117 0 43F
** NODE: 117 = 8_66_313#
C12 116 0 50F
** NODE: 116 = 8_16_357#
C13 115 0 30F
** NODE: 115 = 8_134_215#
C14 113 0 33F
** NODE: 113 = 8_106_191#
C15 114 0 21F
** NODE: 114 = Yin
C16 112 0 48F
** NODE: 112 = 8_60_195#
C17 111 0 41F
** NODE: 111 = 8_16_123#
C18 110 0 47F
** NODE: 110 = 8_134_97#
C19 108 0 33F
** NODE: 108 = 8_106_73#
C20 109 0 21F
** NODE: 109 = Xin
C21 107 0 48F
** NODE: 107 = 8_58_79#
C22 106 0 107F
** NODE: 106 = COL_GENin
C23 105 0 392F
** NODE: 105 = GND
C24 104 0 30F
** NODE: 104 = 8_134_22#
C25 103 0 33F
** NODE: 103 = 8_106_46#
C26 102 0 43F
** NODE: 102 = 8_66_40#

```
C27 101 0 50F
** NODE: 101 = 8_16_5#
C28 100 0 359F
** NODE: 100 = Vdd
** NODE: 1 = Vdd!
** NODE: 0 = GND!
```

H. SPICE FILE - CURRENT MIRROR

** SPICE file created for circuit mirror

** Technology: scmos

**

```
.MODEL nfet NMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=1
+ VTO=0.8673 DELTA=4.9450E+00 LD=3.5223E-07 KP=4.6728E-05
+ UO=569.7 UEXP=1.7090E-01 UCRIT=5.9350E+04 RSH=1.9090E+01
+ GAMMA=0.4655 NSUB=4.3910E+15 NFS=1.980E+11 VMAX=5.7510E+04
+ LAMBDA=3.9720E-02 CGDO=4.3332E-10 CGSO=4.3332E-10
+ CGBO=3.5977E-10 CJ=1.0096E-04 MJ=0.8119 CJSW=4.6983E-10
+ MJSW=0.323107 PB=0.800000
```

* Weff = Wdrawn - Delta_W

* The suggested Delta_W is -9.0180E-08

```
.MODEL pfet PMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=-1
+ VTO=-0.9506 DELTA=4.5950E+00 LD=3.7200E-07 KP=1.6454E-05
+ UO=200.6 UEXP=2.6690E-01 UCRIT=7.9260E+04 RSH=4.9920E+01
+ GAMMA=0.6561 NSUB=8.7250E+15 NFS=3.27E+11 VMAX=9.9990E+05
+ LAMBDA=4.5950E-02 CGDO=4.5769E-10 CGSO=4.5769E-10
+ CGBO=3.8123E-10 CJ=3.1469E-04 MJ=0.5687 CJSW=3.1456E-10
+ MJSW=0.275802 PB=0.800000
```

* Weff = Wdrawn - Delta_W

* The suggested Delta_W is -2.2400E-07

```
.TRAN lus 150us
```

```
Vdd 1 0 5
```

```
VD1 100 0 5
```

```
*dumb:
```

```
Vi 42 116 0
```

```
Iin 0 42 PWL(0uA 0us 150uA 150uS)
```

** NODE: 0 = GND

** NODE: 1 = Vdd

** NODE: 2 = Error

```
M0 100 101 102 1 pfet L=2.0U W=7.0U
```

```
M1 100 101 103 1 pfet L=2.0U W=7.0U
```

```
M2 100 104 104 1 pfet L=2.0U W=7.0U
```

```
M3 100 105 105 1 pfet L=2.0U W=7.0U
```

```
M4 100 106 106 1 pfet L=2.0U W=7.0U
```

```
M5 107 107 0 0 nfet L=2.0U W=6.0U
```

```
M6 104 107 0 0 nfet L=2.0U W=6.0U
```

```
M7 100 101 109 1 pfet L=2.0U W=7.0U
```

```
M8 110 110 0 0 nfet L=2.0U W=6.0U
```

```
M9 105 110 0 0 nfet L=2.0U W=6.0U
```

```
M10 111 111 0 0 nfet L=2.0U W=6.0U
```

M11 106 111 0 0 nfet L=2.0U W=6.0U
M12 100 112 112 1 pfet L=2.0U W=7.0U
M13 100 101 101 1 pfet L=2.0U W=7.0U
M14 100 101 113 1 pfet L=2.0U W=7.0U
M15 100 114 114 1 pfet L=2.0U W=7.0U
M16 115 115 0 0 nfet L=2.0U W=6.0U
M17 112 115 0 0 nfet L=2.0U W=6.0U
M18 116 116 0 0 nfet L=2.0U W=6.0U
M19 101 116 0 0 nfet L=2.0U W=6.0U
M20 117 117 0 0 nfet L=2.0U W=6.0U
M21 114 117 0 0 nfet L=2.0U W=6.0U
M22 100 101 118 1 pfet L=3.0U W=12.0U
M23 100 101 119 1 pfet L=2.0U W=7.0U
M24 100 120 120 1 pfet L=2.0U W=7.0U
M25 100 121 121 1 pfet L=2.0U W=7.0U
M26 100 122 122 1 pfet L=2.0U W=7.0U
M27 123 123 0 0 nfet L=2.0U W=6.0U
M28 120 123 0 0 nfet L=2.0U W=6.0U
M29 124 124 0 0 nfet L=2.0U W=6.0U
M30 121 124 0 0 nfet L=2.0U W=6.0U
M31 100 101 125 1 pfet L=2.0U W=7.0U
M32 126 126 0 0 nfet L=2.0U W=6.0U
M33 122 126 0 0 nfet L=2.0U W=6.0U
M34 100 101 127 1 pfet L=2.0U W=7.0U

v8 127 123 0
v7 125 107 0
v6 119 124 0
v5 118 126 0
v4 113 115 0
v3 109 117 0
v2 103 111 0
v1 102 110 0

C0 127 0 16F
** NODE: 127 = out8
C1 100 0 256F
** NODE: 100 = Vdd
C2 0 0 259F
** NODE: 0 = GND
C3 125 0 16F
** NODE: 125 = out7
C4 126 0 21F
** NODE: 126 = in5
C5 124 0 21F
** NODE: 124 = in6
C6 122 0 39F
** NODE: 122 = 8_142_169#
C7 123 0 21F
** NODE: 123 = in8


```
C8 121 0 39F
** NODE: 121 = 8_304_143#
C9 120 0 39F
** NODE: 120 = 8_231_133#
C10 119 0 16F
** NODE: 119 = out6
C11 118 0 16F
** NODE: 118 = out5
C12 117 0 21F
** NODE: 117 = in3
C13 115 0 21F
** NODE: 115 = in4
C14 114 0 39F
** NODE: 114 = 8_152_23#
C15 116 0 21F
** NODE: 116 = STEPin
C16 113 0 16F
** NODE: 113 = out4
C17 112 0 39F
** NODE: 112 = 8_302_16#
C18 109 0 16F
** NODE: 109 = out3
C19 111 0 21F
** NODE: 111 = in2
C20 110 0 21F
** NODE: 110 = in1
C21 107 0 21F
** NODE: 107 = in7
C22 106 0 39F
** NODE: 106 = 8_306_138#
C23 105 0 39F
** NODE: 105 = 8_166_138#
C24 104 0 39F
** NODE: 104 = 8_207_144#
C25 103 0 16F
** NODE: 103 = out2
C26 102 0 16F
** NODE: 102 = out1
C27 101 0 113F
** NODE: 101 = 8_89_48#
** NODE: 1 = Vdd!
** NODE: 0 = GND!
```

I. SPICE INPUT FILE - MODULO 4 ADDER DESIGN

```
** SPICE file created for circuit Modulo Four Adder Cell
** Technology: scmos
**

**
*****
**
**INPUT HEADER FILE FOR SPICE MODEL
**
**ALL REFERENCED TESTS ON THE MOD 4 ADDER DESIGN
**UTILIZED THIS SPICE HEADER FILE TO GENERATE THE
**DESIRED INPUT WAVEFORMS AND OTHER VARIOUS SIGNALS
**
*****

*****
**
**MODEL PARAMETERS PROVIDED BY ORBIT FOR
**A TYPICAL N-WELL PROCESS
**
*****
.MODEL nfet NMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=-1
+ VTO=0.8673 DELTA=4.9450E+00 LD=3.5223E-07 KP=4.6728E-05
+ UO=569.7 UEXP=1.7090E-01 UCRT=5.9350E+04 RSH=1.9090E+01
+ GAMMA=0.4655 NSUB=4.3910E+15 NFS=1.980E+11 VMAX=5.7510E+04
+ LAMBDA=3.9720E-02 CGDO=4.3332E-10 CGSO=4.3332E-10
+ CGBO=3.5977E-10 CJI=1.0096E-04 MJ=0.8119 CJSW=4.6983E-10
+ MJSW=0.323107 PB=0.800000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is -9.0180E-08
.MODEL pfet PMOS LEVEL=2 PHI=0.600000 TOX=4.2100E-08 XJ=0.200000U TPG=-1
+ VTO=-0.9506 DELTA=4.5950E+00 LD=3.7200E-07 KP=1.6454E-05
+ UO=200.6 UEXP=2.6690E-01 UCRT=7.9260E+04 RSH=4.9920E+01
+ GAMMA=0.6561 NSUB=8.7250E+15 NFS=3.27E+11 VMAX=9.9990E+05
+ LAMBDA=4.5950E-02 CGDO=4.5769E-10 CGSO=4.5769E-10
+ CGBO=3.8123E-10 CJI=3.1469E-04 MJ=0.5687 CJSW=3.1456E-10
+ MJSW=0.275802 PB=0.800000
* Weff = Wdrawn - Delta_W
* The suggested Delta_W is -2.2400E-07

.TRAN .1ns 1000ns
*.TRAN 1us 200us

Vdd 1 0 5
```

```

*****
*Power and ground to the chip:
*
*connect VDD to PLA:
Vvdd_pla1 1 112 0
Vvdd_pla2 1 100 0
Vvdd_pla3 1 864 0
Vvdd_pla4 1 296 0

*connect GND to PLA:
Vgnd_pla1 0 119 0
Vgnd_pla2 0 105 0
Vgnd_pla3 0 857 0

**Note: PLA_GND4 is connected to PLA_GND1

*connect VDD to PAD ring:
Vvdd_pads 1 1389 0
*connect GND to PAD ring:
Vgnd_pads 0 1393 0
*****

*These dummy sources are required in
*order to generate and measure input
*currents and power:
Vxin 42 519 0
Vyin 43 872 0
Vcin 44 1102 0

*outputs
Vsum 155 3000 0
Rload_sum 3000 0 1K

Vcy 858 4000 0
Rload_cy 4000 0 1K

*****
*
*FUNCTIONAL TESTING
*
*These 3200us pulse trains generate all
*possible combinations of inputs:
*
*EX1 0 42 PULSE(0uA 40uA 50us 1us 1us 50us 200us)
*EX2 0 42 PULSE(0uA 80uA 100us 1us 1us 50us 200us)
*EX3 0 42 PULSE(0uA 120uA 150us 1us 1us 50us 200us)

*TY1 0 43 PULSE(0uA 40uA 200us 1us 1us 200us 800us)
*TY2 0 43 PULSE(0uA 80uA 400us 1us 1us 200us 800us)
*TY3 0 43 PULSE(0uA 120uA 600us 1us 1us 200us 800us)

```

```

*IC1 0 44 PULSE(0uA 40uA 800us 1us 1us 800us 3200us)
*IC2 0 44 PULSE(0uA 80uA 1600us 1us 1us 800us 3200us)
*IC3 0 44 PULSE(0uA 120uA 2400us 1us 1us 800us 3200us)
*****
*
*STATIC POWER TESTING
*
*no load - all inputs at logic 0:
*Vdd 1 0 PWL(0 0uA 200us 5)

*Ixin 0 42 0uA
*Iyin 0 43 0uA
*Icin 0 44 0uA

*full load - all inputs at logic 3:
*Vdd 1 0 5
*Ixin 0 42 120uA
*Iyin 0 43 120uA
*Icin 0 44 120uA

*****
*
*TRANSIENT ANALYSIS
*
*sum output from 0 to logic 3:
*Ixin 0 42 PULSE(0uA 120uA 50ns .1ns .1ns 500ns 1000ns)
*Iyin 0 43 0uA
*Icin 0 44 0uA
*
*carry out from 0 to logic 1:
Ixin 0 42 PULSE(0uA 80uA 50ns .1ns .1ns 500ns 1000ns)
Iyin 0 43 PULSE(0uA 80uA 50ns .1ns .1ns 500ns 1000ns)
Icin 0 44 0uA

.END

```


J. MAGIC LAYOUTS

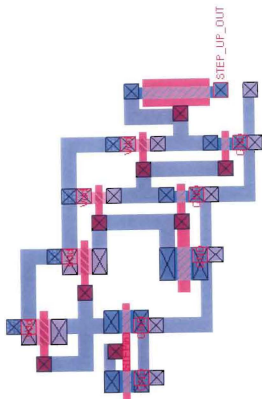


Figure A.1: Step Up Generator Cell

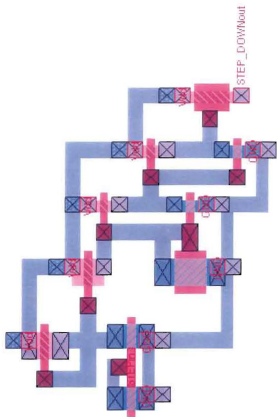


Figure A.2: Step Down Generator Cell

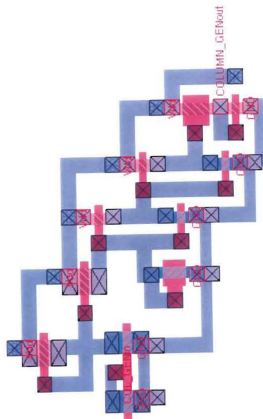


Figure A.3: Column Output Generator Cell

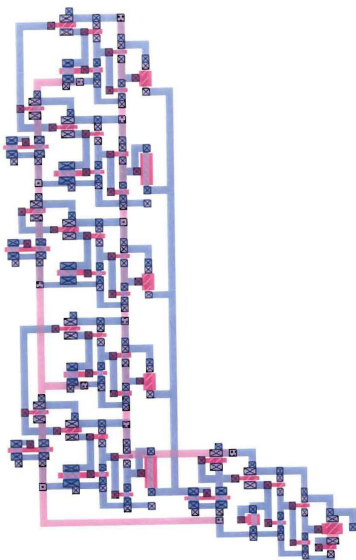


Figure A.4: Term $3^6X(2,2)Y(3,3)C(2,2)$

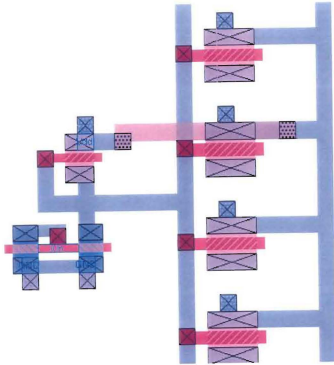


Figure A.5: Current Mirror



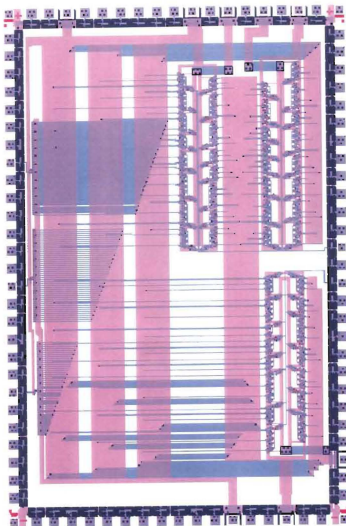


Figure A.6: HAMLET Design Modulo 4 Adder

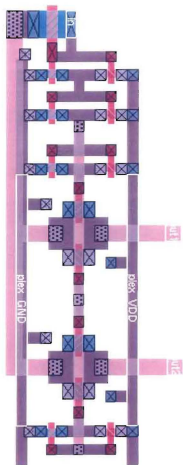


Figure A.7: Current-Input Complementary Transmission Gate



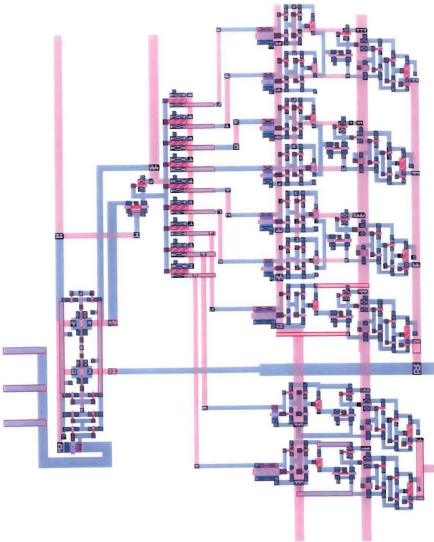


Figure A.8: Alternate Design Modulo 4 Adder

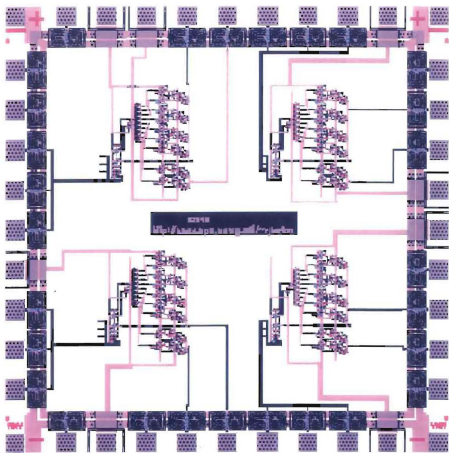


Figure A.9: Fabrication Layout Alternate Design Adder

K. SCOPE PHOTOGRAPHS - TRANSIENT ANALYSIS

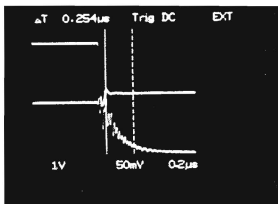


Figure A.10: T_f Sum 3 \rightarrow 0

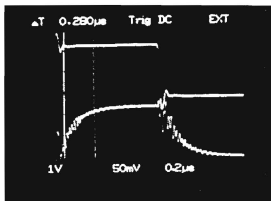


Figure A.11: T_f Sum 0 \rightarrow 3

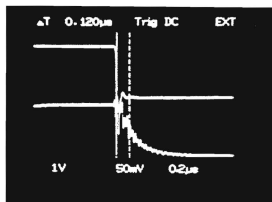


Figure A.12: T_{pdf} Sum 3->0

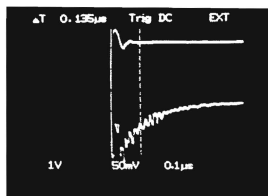


Figure A.13: T_{pdf} Sum 0->3

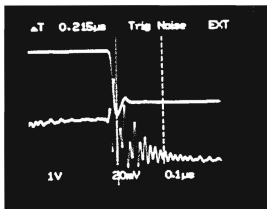


Figure A.14: T_T Carry Out 1->0

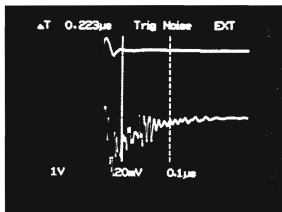


Figure A.15: T_T Carry Out 0->1

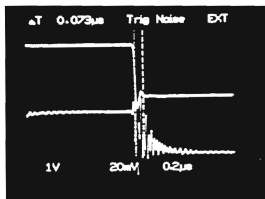


Figure A.16: T_{pdf} Carry Out 1->0



Figure A.17: T_{pdr} Carry Out 0->1

L. TRANSIENT ANALYSIS SCOPE PLOT - MODULO 4 ADDER CELL.

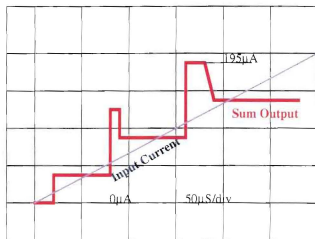


Figure A.18: Sum 0->3 Power Transients

M. FABRICATION PHOTOGRAPH

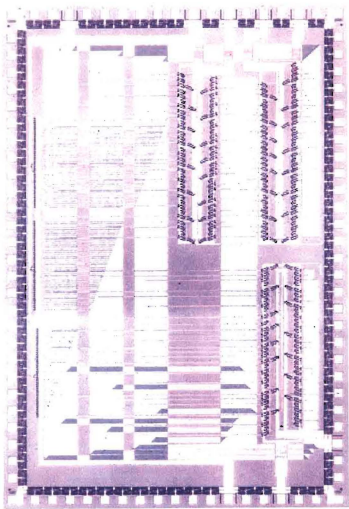
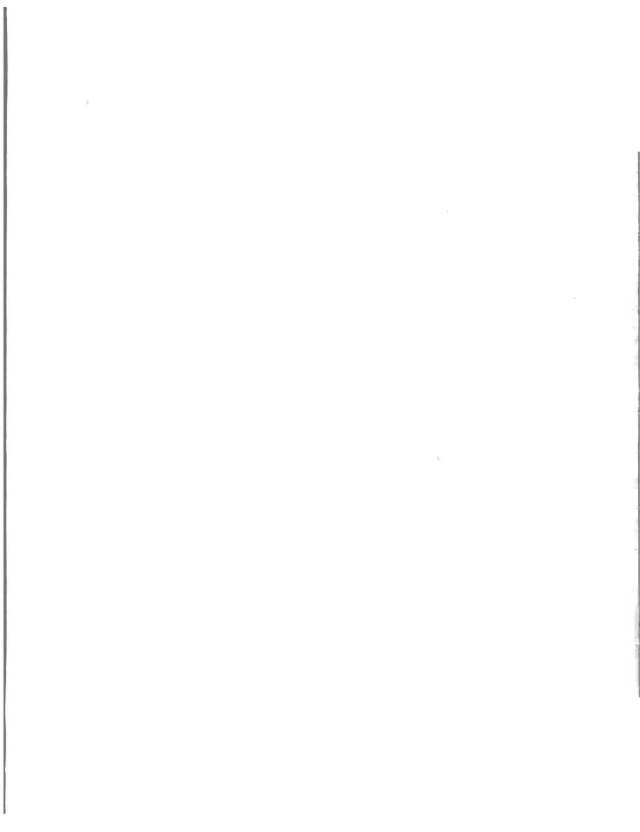


Figure A.19: Die Photo Modulo 4 Adder Cell



INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
Cameron Station
Alexandria, VA 22304-6145
2. Dudley Knox Library, Code O13 2
Naval Postgraduate School
Monterey, CA 93943-5101
3. Chairman, Code EC 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, CA 93943-5121
4. Prof. D. Fouts, , Code EC/Fs 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, CA 93943-5121
5. Prof. J. T. Butler, Code EC/Bu 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, CA 93943-5121
6. Cdr. Robert J. Voigt, Code EC 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, CA 93943-5121
7. Lt. Col. Khaled A. Shehata 1
SGC #1837
Naval Postgraduate School
Monterey, CA 93943-5121
8. Lt. Robert J. Barton III, Code EC 2
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, CA 93943-5121
9. Cdr. John M. Yurchak 1
VAW-116
FPO, AP 96601



DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CA 93943-5101

DUDLEY KNOX LIBRARY



3 2768 00319154 5