Chapter 5 Conclusions and further research

In this dissertation, we have examined the concept of programmable cellular logic arrays at several different levels. A number of examples have been given to demonstrate the feasibility of PL specifically. Several of these examples have been shown to be superior in important ways to more conventional versions. More importantly, however, we have also discussed PL in more abstract terms in an attempt to contribute some understanding and coherence which will be more useful than specific examples to following researchers in this field.

In order to attach a more global meaning to research of this kind, the work must be viewed in the larger context of systems design. From this vantage point, it may be said that the principal tenet underlying this work is that of maximizing design congruence. The property of congruence is used here to refer to the match, fit, and consistency of a design with its environment. This dissertation has attempted to improve several congruence relationships inherent in the design of digital computer hardware through the use of programmable cellular logic.

One such relationship is that which exists between a design and the technology available for physically realizing that design. We have discussed large-scale integrated circuit technology and have shown several ways in which programmable logic arrays can potentially provide better utilization of LSI technology than other cellular forms. In particular, PL offers a new solution to the customization problem, facilitates testing, and improves failure tolerance and repairability.

As a contribution to a better understanding of cellular arrays in general, we have presented several characteristics of arrays which may be viewed as the dimensions of an n-space of all cellular arrays. It is felt that previous research in this area has too often concentrated only on specific array designs (isolated points in the space) without relating to other arrays or attempting a real understanding of the design. Therefore, we have discussed a number of ways in which arrays may be compared, contrasted, and related.

Another congruence relationship of interest is that which exists between a design and the set of tasks it is intended to perform. It is asserted that generality (i.e. the power, scope, and range of functions performed; opposite to specificity) is a very important characteristic in the systems design process. The degree of generality required to synthesize a given set of functions affects the design of a machine to perform those functions in fundamental and important ways. In addition, we may use the dimension of generality to order and relate various functions and machines. In this work, we have ordered the discussion of PL arrays into two broad areas -- low generality and high generality. Via several examples, we have been able to demonstrate the feasibility of PL in both areas.

We began by presenting two slightly different versions of an array intended to perform the simple arithmetic and parity functions. Despite the high similarity between the two cells, one array was seen to be more efficient at performing the arithmetic functions, while the other was superior on the parity functions. This simple example demonstrates how small changes in an array can be used to adjust the array's efficiency over the target set of functions. This result is made stronger by noting that, because of the specificity of their functions, low-q arrays are inherently less "tunable" than other arrays.

A significantly different low-g example is the shift register array S1. Each cell in this array contains an n-bit shift register and connections to the four nearest-neighbor cells. An expression was derived which gives the optimal value of n as a function of the expected manufacturing yield over the array. We have shown how this array design may be used to achieve significantly better utilization of LSI technology than the discretionary wiring or 100-percent-yield processes in the task of creating shift registers in the presence of numerous manufacturing defects. This example demonstrates congruence both to its task and to its physical implementation.

We began the discussion of the high-generality area with the presentation of a cell schema for high-g arrays. The principal new idea here (in addition to the fact that cell forms are nearly non-existent in the literature) is that the cell function takes its inputs from the outputs of the cell connection function, rather than from the inputs to the cell. A number of properties fundamental to the efficient application of PL arrays to high-g problems were also discussed and related to this cell schema.

Several new techniques have been developed which can improve the congruence between an array and the function it realizes. One such technique involves a scheme for adjusting the angle at which signal buses traverse a rectangular array of cells in order to provide a suitable range of fan-in and fan-out on these signals. This new idea has applicability over a wide range of cellular array types.

Some results have been derived concerning the usefulness of various 3-variable functions in synthesizing arbitrary combinatorial functions. A group of transformations consistent with PL array synthesis was introduced. It includes input permutation, negation, and duplication and output inversion. Under these transformations, it was shown that only 10 functions are necessary to generate all 256 3-variable functions. Several of these 10 functions are each capable of generating nearly 1/3 of all 3-variable functions. It is suggested that these same functions are more efficient than other 3-variable functions when used to synthesize arbitrary combinational functions under the same transformations. Several functions often proposed as primitives for this task (e.g., majority gate and XOR) were shown to be the weakest of all 3-variable functions in this comparison.

As an application of high-g PL arrays, we have discussed the problem of control in digital computers. A PL control array was presented as an alternative to the conventional microprogramming approach. Potential advantages of the PL control array over microprogrammed control include greater flexibility and parallelism, improved failure tolerance and diagnosability, and elimination of irregular (non-iterative) logic. It was also shown that, for all machines above a certain (small) size, the total number of components required with PL control is less than that with current microprogramming techniques. Control for a small computer was implemented in detail using a PL array.

A number of interesting research problems have been suggested by this work. The most obvious one is the development of algorithms for automatic programming of PL arrays. This may be approached in several ways. One type of algorithm necessary is that which assigns states and paths, etc., to build up a given structure in an unprogrammed array. Another type of algorithm is that which concentrates on correcting spontaneous defects in the array by making very local perturbations in the array.

Although the arrays proposed in this work are largely within the scope of present LSI technology, some research in this area would be appropriate. PL often requires more complex connections between circuit layers and/or more layers than conventional LSI. In order to actually build these devices, a more thorough knowledge of the added costs of connections, layers, slanted buses, etc., should be obtained. Also, an important possibility to be investigated is that of using slow-write fast-read control elements in the parameter part of the PL cell.

Our discussions of the usefulness of 3-variable functions under certain transformations logically deserves extension. Given a set of transformations of cell inputs and outputs, what primitive or set of primitives are most effective in synthesizing all n-variable functions for a given n? How does one find these primitives? Another possibility is to weight each of the object classes of functions according to its desirability. Then the question becomes one of optimizing the "fit" of the class generated by the primitives to the object class.

Another theoretical consideration is that of testing PL arrays. How does the flexibility of the interconnection function affect the effectiveness of various testing procedures? What algorithms are most efficient in testing an array with a given connection topology and expected yield?

The most logical specific extension of this work has to do with the combinatorial functions discussed in Section 4.4.1. The effectiveness of imbedding several of these 3-variable functions (perhaps in combination) into a complete high-g cell should be investigated. Comparisons should be made between an array of these cells and conventional methods (as well as other cellular forms). This would also serve to further substantiate the applicability of the high-q cell schema which was presented.

Another specific problem concerns the shift register array S1. How does the fault-avoidance capability of this array vary with increasingly complex connection structures? What is the optimal connection function for a given range of defect densities? Solutions to these problems would have implications for fault-avoidance in other more general arrays as well.

It remains to be seen whether the computer science community and/or the commercial manufacturers will take up the banner of PL. Hopefully, this dissertation has provided significant justification for further interest in this technique.

References

1. D. B. Armstrong
On finding a nearly minimal set of fault detection tests for combinatorial logic nets.
IEEE Trans. EC-15, Feb. 1966, pp. 66-73.

2. T. F. Arnold, C. J. Tan, and M. M. Newborn
Iteratively realized sequential circuits.
IEEE Trans. C-19, No. 1, Jan. 1970, pp.54-66.

3. G. H. Barnes, et al.
The ILLIAC IV computer.
Dept. of Computer Science, Univ. of Ill.

4. K. E. Batcher
Sorting networks and their applications.
Proc. SJCC 1968, pp. 307-314.

5. H. R. Beelitz, S. Y. Levy, R. J. Linhardt, and H. S. Miller
System architecture for large-scale integration.
Proc. FJCC, 1967, pp. 185-200.

6. M. Canning, R. Dunn, and G. Jeansonne
Active memory cells for discretion.
Electronics, Vol. 40, No. 4, Feb. 20, 1967, pp. 143-154.

7. H. A. Curtis

Systematic procedures for realizing synchronous sequential machines using flip-flop memory.

Part I - IEEE Trans. C-18, No. 12, Dec. 1969, pp. 1121-1127.

Part II - IEEE Trans. C-19, No. 1, Jan. 1970, pp. 66-73.

8. G. Estrin, B. Russell, R. Turn, J. Bibb

Parallel processing in a restructurable computer system.

IEEE Trans. EC-12, Dec. 1963, pp. 747-755.

9. D. E. Farina

Large-scale integration: a status report.

Datamation, Feb. 1968, pp. 22-29.

10. M. Golay

Hexagonal parallel pattern transformations.

IEEE Trans. C-18, No. 8, Aug. 1969, pp. 733-740.

11. R. F. Graham

Semiconductor memories: evolution or revolution?

Datamation, June 1969, pp. 99-104.

12. J. Hartmanis

On the state assignment problem for sequential machines. I.

IRE Trans. EC-10, June 1961, pp. 157-165.

13. L. Hellerman

A catalog of three-variable OR-invert and AND-invert logical circuits.

IEEE Trans. EC-12, June 1963, pp. 198-223.

14. E. R. Jones and C. H. Mays

Automatic test generation methods for large-scale integrated logic.

IEEE J. of Solid-State Circuits, Vol. SC-2, No. 4,

Dec. 1967, pp. 221-226.

15. W. H. Kautz

Cellular interconnection arrays.

IEEE Trans. C-17, May 1968, pp. 443-451.

16. W. H. Kautz

Cellular logic-in-memory arrays.

IEEE Trans. C-18, Aug. 1969, pp. 719-727.

17. W. H. Kautz

Testing for faults in combinational cellular logic arrays.

Proc. 8th Annual Symp. on Switching Theory and Automata Theory,

Oct. 1967, pp. 161-174.

18. K. K. Maitra

Cascaded switching networks of two-input flexible cells.

IRE Trans. EC-11, April 1962, pp. 136-143.

19. K. Maling and E. L. Allen

A computer organization and reprogramming system for

automated maintenance.

IEEE Trans. EC-12, Dec. 1963, pp. 887-895.

20. D. E. Marshall

Integrated circuit fabrication: Where can costs be sliced?

Computer Design, May 1968, pp. 32-34.

21. R. C. Minnick

Cobweb cellular arrays.

Proc. FJCC 1965, pp. 327-340.

22. R. C. Minnick

Cutpoint cellular logic.

IEEE Trans. EC-13, Dec. 1964, pp. 685-698.

23. R. C. Minnick

Survey of microcellular research.

Stanford Research Institute, July 1966,

also J. ACM, Vol. 14, No. 2, April 1967, pp. 203-241.

24. R. C. Minnick and R. A. Short

Cellular linear-input logic.

Stanford Research Institute, 1964, also AD 433 802.

25. R. C. Minnick et al.

Cellular arrays for logic and storage.

Stanford Research Institute, April 1966, also AD 643 178.

26. F. Miyata

Realization of arbitrary logical functions using majority elements.

IEEE Trans. EC-12, June 1963, pp. 183-191.

27. M. Newborn

A synthesis technique for binary input - binary output synchronous sequential Moore machines.

IEEE Trans. C-17, No. 7, July 1968, pp. 697-699.

28. A. J. Nichols
Minimal shift-register realizations of sequential machines.
IEEE Trans. EC-14, No. 5, Oct. 1965, pp. 688-700.

29. W. A. Notz, E. Schisha, J. L. Smith and M. G. Smith
Large-scale integration.
Electronics, Vol. 40, No. 4, Feb. 20, 1967, pp. 130-141.

30. T. T. Pariser
Connection considerations with a view toward batch fabrication.
Proc. Nat. Symp on the Impact of Batch Fabrication on Future Computers, April 1965, pp. 213.

31. Y. N. Patt
A complex logic module for the synthesis of combinatorial switching circuits.
Proc. SJCC, 1967, pp. 699-705.

32. R. L. Petritz
Current status of large-scale integration technology.
Proc. FJCC, 1967, pp. 65-85.

33. R. L. Petritz
Large-scale integrated electronics.
Proc. FJCC, 1966, pp. 71-87.

34. C. V. Ramamoorthy and S. C. Economides
Fast multiplication cellular arrays for LSI implementation.
Proc. FJCC, 1969.

35. D. L. Rohrbacher

Advanced computer organization study.

Goodyear Aerospace Corporation, April 1966, also AD 631 870-1.

36. R. F. Rosin

Contemporary concepts of microprogramming and emulation.

Computing Surveys, Vol. 1, No. 4, Dec. 1969, pp. 197-212.

37. B. R. Seeds

Yield, economic, and logistic models for complex digital arrays.

IEEE Convention Record 1967, pp. 60-61.

38. R. E. Stearns and J. Hartmanis

On the state assignment problem for sequential machines. II.

IRE Trans. EC-10, Dec. 1961, pp. 593-603.

39. H. S. Stone

A logic-in-memory computer.

IEEE Trans. C-19, No. 1, Jan. 1970, pp. 73-78.

40. E. Tammaru

Efficient testing of combinational logic cells in large-scale arrays.

Stanford Electronics Labs, Stanford U., May 1968.

41. E. Tammaru and J. B. Angell

Redundancy for LSI yield enhancement.

IEEE J. of Solid-State Circuits, Vol. SC-2, Dec. 1967, pp. 172-181.

42. C. G. Thornton

What has happened to LSI - a supplier's view.

Proc. FJCC, 1969, pp. 369-378.

43. L. L. Vadasz et al.

Silicon gate technology.

IEEE Spectrum, Oct. 1969, pp. 28-35.

44. S. E. Wahlstrom

Programmable logic arrays.

Electronics, Vol. 40, No. 25, Dec. 11, 1967, pp. 90-95.

45. R. M. Warner, Jr.

A comparison of MOS and bipolar integrated circuits.

IEEE Spectrum, June 1967, pp. 50-58.

46. L. S. Watkins

Inspection of integrated circuit photomasks with intensity spatial filters.

Proc. of the IEEE, Vol. 57, No. 9, Sept. 1969, pp. 1634-1639.

47. S. Weber et al.

Large-scale integration.

Electronics, Vol. 40, No. 4, Feb. 20, 1967, pp. 124-182.

48. S. S. Yau and C. K. Tang

Universal logic circuits and their modular realizations.

Proc. SJCC, 1967, pp. 297-305.

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Carnegie-Mellon University Department of Computer Science Pittsburgh, Pennsylvania 15213	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP

3. REPORT TITLE

PROGRAMMABLE CELLULAR LOGIC ARRAYS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Scientific Interim

5. AUTHOR(S) (First name, middle initial, last name)

Richard G. Shoup

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
March, 1970	190	48

8a. CONTRACT OR GRANT NO. F44620-67-C-0058 b. PROJECT NO. 9718 c. 6154501R d. 681304	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

1. This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
TECH, OTHER	Air Force Office of Scientific Research 1400 Wilson Boulevard (SRMA) Arlington, Virginia 22209

13. ABSTRACT

In recent years, technological advances have provided the designer of computing hardware with the ability to batch-fabricate large numbers of logic components on a single semiconductor slice. Numerous researchers have investigated the synthesis of various kinds of digital logic by using arrays of identical cells. This dissertation considers cellular arrays whose individual cell functions are determined by parameter flip-flops and logic gates in the cell, rather than by a physical customizing operation during manufacture. Potential advantages of this technique include functional variability after manufacture, more efficient testing, and enhanced failure tolerance.

Arrays may be classified according to their generality, i.e., the number and range of the tasks which they are designed to perform. Two significantly different examples of low-generality arrays are presented and analyzed. One, a shift register array, is shown to be more effective than some conventional techniques for creating shift registers in the presence of numerous manufacturing defects.

A new cell schema is introduced which exhibits properties important in the synthesis of high-generality functions. Techniques are presented for improving the match between an array design and the target class of tasks. As an example, the problem of central processor control logic is approached in terms of a programmable logic array. A small computer is implemented in detail using these techniques. This method of synthesizing control is compared to a more conventional microprogramming approach.

DD FORM 1 NOV 65 1473

Security Classification

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT

Security Classification