Path-Programmable Logic

Tony M. Carter and Kent F. Smith

UUCS-89-016

Department of Computer Science
University of Utah
Salt Lake City, UT 84112 USA

August 22, 1990

Abstract

Path-Programmable Logic (PPL) is a structured IC design methodology under development at the University of Utah. PPL employs a *sea-of-wires* approach to design. In PPL, design is done entirely using cells for both functionality and interconnect. PPL cells may have modifiers that change either their connections or functionality. Wires in the PPL *design plane* are segmentable at any cell boundary. PPL is implemented as a set of cell libraries (NMOS, CMOS, and GaAs) and a suite of tools that permit the designer to create, modify, simulate and check PPL circuit designs and to generate mask data for them. PPL exhibits little or no area penalty with respect to full custom densities while permitting system design to be done more rapidly than with gate arrays or standard cells. PPL may be implemented as a sea-of-gates gate array to provide fast turnaround. [1]

[1]This research was supported through DARPA contract number DAAK11-84-K-0017. All opinions, findings, conclusions or recommendations expressed in this document are those of the author and do not necessarily reflect the views of the sponsoring agencies.

VLSI Systems Research Group, Technical Report UUCS-89-016
University of Utah, Department of Computer Science

Path-Programmable Logic

TONY M. CARTER AND KENT F. SMITH* *(carter@cs.utah.edu)*
University of Utah
Dept. of Computer Science
3190 Merrill Engineering Building
Salt Lake City, Utah 84112

Keywords: IC Design, PPL, Structured Logic

Abstract. Path-Programmable Logic (PPL) is a structured IC design methodology under development at the University of Utah. PPL employs a *sea-of-wires* approach to design. In PPL, design is done entirely using cells for both functionality and interconnect. PPL cells may have modifiers that change either their connections or functionality. Wires in the PPL *design plane* are segmentable at any cell boundary. PPL is implemented as a set of cell libraries (NMOS, CMOS and GaAs) and a suite of tools that permit the designer to create, modify, simulate and check PPL circuit designs and to generate mask data for them. PPL exhibits little or no area penalty with respect to full-custom densities while permitting system design to be done more rapidly than with gate arrays or standard cells. PPL may be implemented as a sea-of-gates gate array to provide fast turnaround.

1 Introduction

Path-Programmable Logic (PPL) [7, 5, 8, 9] is an integrated circuit design methodology in which the entire design process is well structured and under control of the designer. Other structured integrated circuit design methodologies such as gate arrays and standard cells provide a structured method of design, but do not provide much flexibility for the designer. Furthermore, the entire design process is not under the control of the designer since automatic place and route software is used to produce the actual layout once the designer has selected the desired collection of gates/cells and has specified their interconnections. A substantial area penalty is paid in both gate arrays and standard cell approaches; a penalty not paid in PPL. PPL gives the integrated circuit designer the ability to perform not only gate level or cell level design, but also transistor level design if the need arises.

*This research was supported through DARPA contract number DAAK11-84-K-0017.

2 The PPL Design Methodology

In most circuits, wiring density is at least as important as transistor density when considering overall circuit area. This observation leads to the following conclusion: *a methodology that gives wiring a chance to be as dense as is possible is preferable to methodologies that consider only functional or transistor density.* If wiring takes up more than 50% of the chip area, then reducing the size of a given circuit module by that last micron (at the expense of wiring) can easily increase the overall size and cost of an integrated circuit.

The sea-of-gates approach has a dual — the sea-of-wires. Instead of considering an integrated circuit to be initially covered with a sea of transistors or gates (spread out at predefined intervals) that can be interconnected with wires on two (or more) layers, we think of it being covered with two sets of orthogonal wires under which we will place transistors or gates to add function. As seen in figure 1 the two sets of wires run horizontally (the row wires) and vertically (the column wires). These wires are spaced so that a reasonable number of contacts between wires and transistors can be made in an area of a given size. Subsets of the wires in each direction are collected into an area known as a unit cell. The size of the unit cell is dictated by two considerations, the number of wires that must pass through in each direction and constraints dictated by how many transistors must be present in the simplest functional unit cells. Once the sea-of-wires has been partitioned into an array of unit cell locations, it is called the *design plane.*

With the design plane initially containing only blank cells, the circuit is then designed by conceptually replacing blank cells with cells that add function (e.g., collections of interconnected transistors) or interconnect. In figure 2, a cell containing two transistors replaces a single blank unit cell in the design plane. Cells form the heart of PPL (and other Cell Matrix methodologies [1, 2]). Every circuit structure, including interconnect, is embodied as a cell. A cell has ports corresponding to the points where signal wires (those not carrying power and ground) cross cell borders. A cell may contain any collection of connected transistors or wires as long as power and ground are uninterrupted and as long as all wires that must communicate with other (possibly adjacent) cells use the port locations defined by the wires in the design plane.

Each PPL cell need not be a complete and independent gate. Indeed, AND and OR gates are constructed in a distributed fashion using simple PPL cells that correspond roughly to the cells of a static PLA.

PPL cells may have modifiers that either change the way a cell hooks up to its ports or change its functionality or both. When a modifier is present, it is said to be asserted. In the simplest case, modifiers will affect only a cell's ports. In figure 3, we see a cell with three modifiers. Modifier 1 hooks the input of the cell to the next-to-rightmost column wire and modifier 2 hooks the output of the cell to this same wire. This same cell has a third modifier that radically alters its functionality

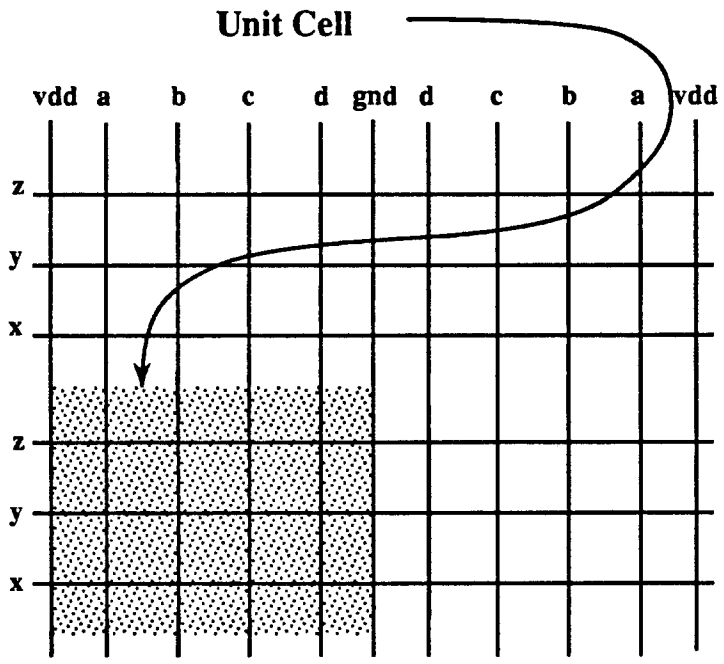

Figure 1: **The Cell Matrix "Sea-of-Wires"**

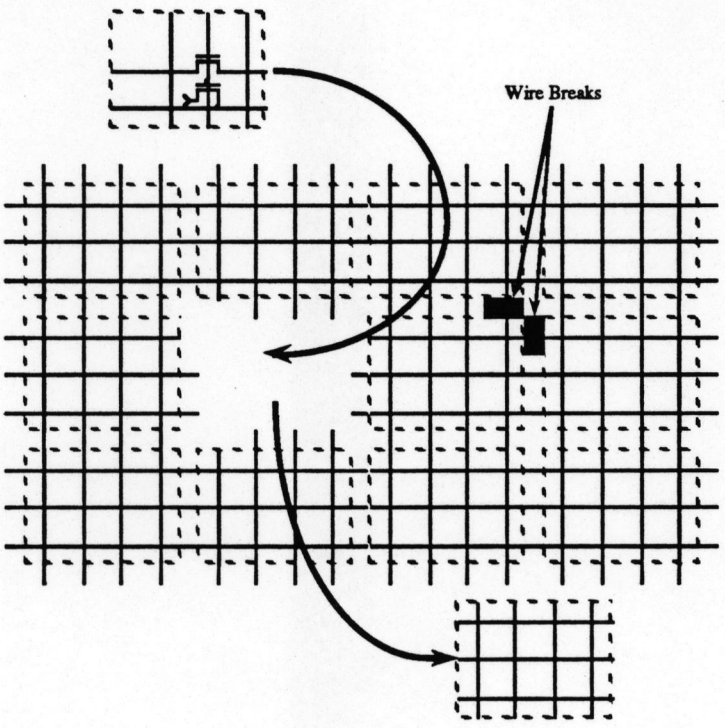

Figure 2: **Design Construction in the Plane**

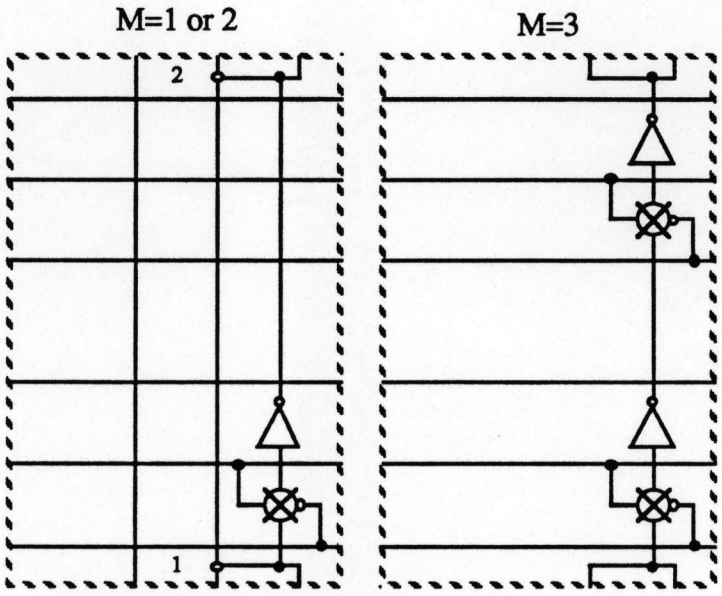

Figure 3: **PPL Cell Modifiers**

while maintaining its size and aspect ratio.

In PPL, signal wires leave (or enter) a cell at predefined port locations. Initially, the column wires are as long as the circuit is tall and the row wires are as long as the circuit is wide. In other words, wires span the entire width or height of the circuit being designed.

When a cell is placed in the design plane, it is automatically connected to its neighbors on all sides at all port locations. If a connection to a port of a neighboring cell is not desired, then a small piece of wire connecting the two adjacent ports must be removed. The designer performs this task symbolically by indicating that a wire is to be broken. Physically, the signal wires (both row and column) of a cell do not extend all the way to the edge of the cell. When two cells are placed adjacent to each other, there are in reality no connections between them. CAD tools add connections, or small rectangles of metal (or some other material), at the port location centered on the coincident edges of both cells.

Wires may therefore be broken between any two adjacent cells as in figure 2. In some cells, a specific port may not be used and the wire intersecting the cell at the location of the port is permanently broken, indicating that it can never be connected to the outside world. The concept of breaking wires at cell boundaries permits the

design plane to be arbitrarily segmented and enables modules to be easily embedded in the plane.

One of the primary advantages of PPL is that there are multiple useful symbologies that can represent each cell in a circuit. Since the design plane is entirely tessellated with cells, there are no areas in which the layout of the circuit cannot be represented abstractly through the use of symbols. Even interconnect is represented symbolically since it is achieved by using cells or cells with modifiers. Figure 4 illustrates four possible symbolic representations for one physical PPL cell. Clockwise from the upper left hand corner there is the transistor schematic, a logic schematic with the row wires implementing a NAND gate, a logic schematic with the row wires implementing a NOR gate with true low inputs and finally, the symbol "1" indicating that this cell senses the fact that the column contains a logic 1. Each symbol has its useful context.

3 The PPL Tools and Cellsets

A comprehensive suite of CAD tools has been developed for the design of NMOS, CMOS and GaAS PPL circuits. These tools are written in C and run on a wide variety of machines ranging from professional workstations (e.g., Sun) to personal computers (e.g., IBM PC and compatibles running MS-DOS) and mainframes (e.g., VAXes). The basic suite of tools uses only a textual representation of the PPL cells and does not require graphics. This makes it possible to design PPL circuits using virtually any terminal or personal computer. The basic PPL tool suite includes:

- `tiler`, a text-based design editor
- `simpplex`, a circuit extractor and design checker
- `simppl`, a functional simulator
- `timppl`, a worst-case path timing analyzer
- `pplpr`, a printout generator (conventional or laser)
- `ppl2cif`, a mask data generator (CIF, etc.)
- `ppl2spice`, a spice deck generator
- `ppl2hilo`, hilo simulator input generator

The basic PPL tool suite uses a set of cellset databases for a variety of processes:

- MOSIS Scalable CMOS
- MOSIS NMOS
- Vitesse GaAs
- Triquint GaAs

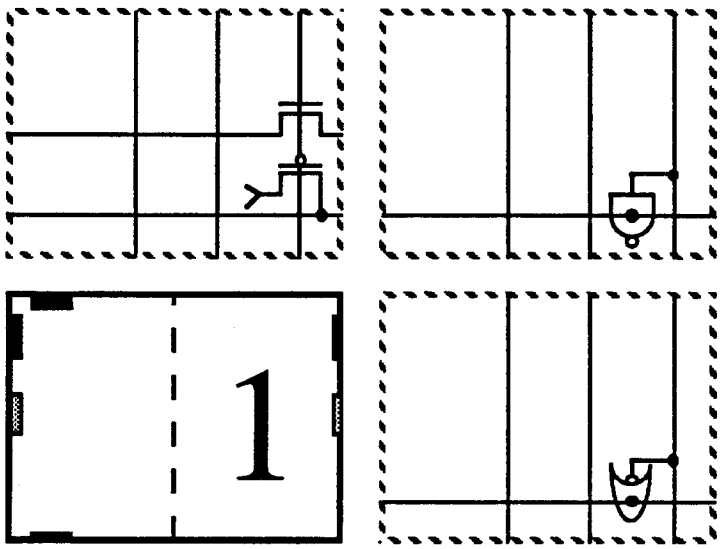

Figure 4: **PPL Cell Symbols**

There is also a 40 pad PPL design frame available as a 2 micron MOSIS Scalable CMOS "tiny chip" which gives users the ability to prototype small 2.2 by 2.2 mm chips for $400.

Other more advanced tools are in various stages of development and run only on professional workstations. Some of these PPL tools are:

- `masher`, a circuit layout optimizer
- `assassin`, a state machine generator
- `insted`, a graphics-based design editor
- `rippl`, a sea-of-gates gate array cell editor
- `testppl`, an automatic test vector generator

`insted` [6] includes the capability to automatically build significant parameterized structures such as register files and counters at a given location. `rippl` [10] is a tool devised to aid in designing PPL cells for use in PPL sea-of-gates gate arrays. PPL is being implemented in full-custom and as conventional mask programmable and laser customizable sea-of-gates gate arrays.

4 The Benefits of PPL

There are many substantial benefits of PPL in the design of integrated circuits. First and foremost, the designer simultaneously designs both logic and layout. Since each cell has a one-to-one correspondence with a cell layout and since the design plane is completely tessellated by cells, the symbolic level design of a circuit also shows the size and aspect ratio of the circuit or module being designed. The designer therefore has more control over how well modules fit together than with other semi-custom methodologies. It allows a single structure, the design plane, to contain many independent modules that communicate by any number of signals.

Second, since all design is done with cells and since all cells can be completely characterized with respect to resistance, capacitance and function, speed can be easily estimated at design time. In fact, it is possible to develop a design editor that indicates to the designer when some predetermined timing constraint has just been violated.

Third, using a character based symbology for cells, design can be done using inexpensive personal computers or even alphanumeric terminals attached to a mainframe or workstation. Many of the most complex tasks normally performed by high-performance mainframe computers, such as design rules checking or gate placement and routing, are altogether eliminated in PPL circuit design once the cell library is in place. Some tasks, such as circuit extraction, that are traditionally very time consuming are significantly simplified because of the additional and complete structure employed in PPL.

Fourth, Cell Matrix design gives the designer more flexibility than most other methodologies while limiting the degrees of design freedom to a manageable level.

Fifth, design times are drastically reduced over other known methodologies while transistor densities are close to those achieved with full-custom design. The primary basis for this observation is a pair of studies that compared the design times and densities of circuits designed using PPL with those achievable with other methodologies [4, 1]. Overall, PPL has excellent functional density ranging from 5% better to 30% worse than full custom and between 2 and 4 times better than gate arrays and standard cell circuits. Design times are also excellent, being 30-40 times faster than full-custom and several times faster than gate array design. For example, a Hogenauer filter [3] containing nearly 20,000 transistors was designed in one day. It has a density of 952 μ^2 per transistor (including pads) as compared to an equivalent standard cell circuit having 3,248 μ^2 per transistor which literally took many man months to design.

The abstraction mechanisms of PPL permit the designer to concentrate more fully on problems of system architecture rather than on layout details while giving direct feedback about the impact of the architectural design on area, aspect ratio and speed.

The benefits of PPL for integrated circuit design are significant, particularly de-

sign time reduction and high circuit density. PPL is an exciting alternative to the design techniques now generally in use with respect to design time, circuit area efficiency and cost of design hardware and tools. In general, PPL cell sets are simple enough that they can be reimplemented in a new technology in less than three months, making it possible to move circuit designs between fabrication processes with relative ease and speed. We believe the PPL design technique represents a quantum leap in integrated circuit design technology.

References

1. T. M. Carter, K. F. Smith, S. R. Jacobs and R. M. Neff, "Cell Matrix Methodologies for Integrated Circuit Design", Tech. Report UUCS-89-004, Univ. of Utah, Dept. of Computer Science, Jan. 1989.

2. T. M. Carter, *Structured Arithmetic Tiling of Integrated Circuits*, Ph.D. Dissertation, Univ. of Utah, Dept. of Computer Science, Dec. 1983.

3. E. B. Hogenauer, "A Class of Digital Filters for Decimation and Interpolation", *Proc. IEEE Int'l Conf on Acoustics, Speech and Signal Processing*, Apr. 1980, pp. 271-274.

4. P.D. Israelsen and K. F. Smith, "Comparison of the Path Programmable Logic Design Methodology with Other Custom and Semicustom Approaches", *Proceedings of ICCD 1985*, Oct. 1985, pp. 73-76.

5. B. E. Nelson, D. Morrell, C. Read and K. F. Smith, "The PPL Integrated Circuit Design Methodology", *Computer Aided Design*, Nov. 1986.

6. R. M. Neff, *INSTED — A Hierarchical Structured Tiling Editor*, M.S. Thesis, Univ. of Utah, Dept. of Computer Science, June 1986.

7. K. F. Smith, B. E. Nelson, T. M. Carter and A. B. Hayes, "Computer-Aided Design of Integrated Circuits Using Path-Programmable Logic", *IEEE Electro '83 Professional Program Session Record*, Apr. 1983.

8. K. F. Smith, *Design of Integrated Circuits with Structured Logic Using the Storage/Logic Array (SLA): Definition and Implementation*, Ph.D. Dissertation, Univ. of Utah, Dept. of Electrical Engineering, Mar. 1982.

9. K. F. Smith, T. M. Carter and C. E. Hunt, "Structured Logic Design of Integrated Circuits Using the Stored/Logic Array",, *IEEE Trans. on Electron Devices*, ED-29:4, Apr. 1982, pp. 765-776.

10. K. N. Smith, *Restructurable Interconnect of Path-Programmable Logic*, M. S. Thesis, Univ. of Utah, Dept. of Computer Science, June 1989.