A LAYOUT AND DESIGN METHODOLOGY FOR DEEP SUB-MICRON APPLICATIONS USING NETWORKS OF PLAs

by

Sunil. P. Khatri, Robert K. Brayton
and Alberto Sangiovanni-Vincentelli

Memorandum No. UCB/ERL M98/68

19 May 1998

A LAYOUT AND DESIGN METHODOLOGY FOR DEEP SUB-MICRON APPLICATIONS USING NETWORKS OF PLAs

by

Sunil P. Khatri, R. K. Brayton and
Alberto Sangiovanni-Vincentelli

Memorandum No. UCB/ERL M98/68

19 May 1998

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

A Layout and Design Methodology for Deep Sub-Micron Applications using Networks of PLAs

Sunil P. Khatri (linus@ic.eecs.berkeley.edu) *

Robert K. Brayton (brayton@ic.eecs.berkeley.edu) *

Alberto Sangiovanni-Vincentelli (alberto@ic.eecs.berkeley.edu) *

Abstract

We propose a VLSI layout and design methodology which addresses the major problems faced in Deep Sub-Micron (DSM) integrated circuit design. The advantages of our flow are high speed and logic density, with a minimal overall area penalty compared to a design flow using standard cells. Our scheme virtually eliminates noise between signal wires, a problem which is becoming aggravated in DSM designs. Also, in our scheme, inductive and capacitive parasitics are highly predictable due to the regular arrangement of conductors. Additionally, we automatically get a power and ground distribution network with a very low resistance at any point on the die.

We utilize two separate layout "fabrics" in our methodology, one for areas where logic is laid out, and another for routing regions between logic islands. The first fabric results from our using a specialized PLA style of design. It allows us to design logic at very high densities, and with very high speeds. We show that the use of this design style results in a 2.5× reduction in area, and at least 2× improvement in speed, over a traditional standard-cell style design. The second fabric eliminates the conventional notion of power and ground routing on the die. Power and ground are essentially "pre-routed" all over the die. By a clever arrangement of power/ground and signal pins, we almost completely eliminate the capacitive effects (and therefore crosstalk noise) between signal wires as well.

Our synthesis flow involves decomposing the design into a network of PLAs, which have a bounded width and height. In our synthesis flow, the number of inputs and outputs of each PLA are flexible, as long as the resulting width of the PLA is bounded. We also fold the resulting PLAs so as to achieve yet better logic densities.

We have implemented our scheme using public domain synthesis, layout, placement and routing software. The crosstalk immunity, high speed, low area overhead, quick design turnaround time, and high predictability of our methodology suggest that it holds much promise as the layout and design flow of choice for DSM integrated circuit design.

*Department of Electrical Engineering and Computer Sciences, University of California, 550-B Cory Hall, Berkeley, CA 94720.
This work was supported by the SRC under grant number 324-040.

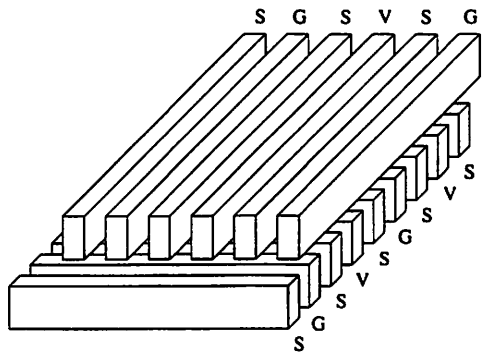

Figure 1: Arrangement of conductors as in [1].

1 Introduction

With the rapid development of VLSI fabrication technologies, we have reached an era where the minimum feature sizes of the leading processes is well below 1 μm. Such processes are called Deep Sub-Micron (DSM) processes [1]. With shrinking feature sizes the fraction of the total delay of a circuit which occurs due to its wires increases. The capacitance of a wire to its neighboring wires increases as well. This affects signal integrity, and also causes problems like unpredictability of circuit delay.

We introduce a layout and synthesis flow which is designed to alleviate these problems. Our scheme is motivated by the fabric introduced in [1]. In this work, we introduced a regular layout fabric in which signal wires alternate with power and ground wires in the layout, on *all* metal layers. If a signal wire is denoted by S, a power wire is denoted by V and a ground wire is denoted by G, then on every metal layer, any sequence of wires will be labeled $\cdots VSGSVSGSV \cdots$. Also, metal wires on any layer run perpendicular to those in the layers above and below it. As a result, the entire chip is maximally gridded with wires in all directions. This layout arrangement is shown in Figure 1.

The advantages of this scheme are several. Firstly, delay variations and signal integrity violations due to crosstalk are almost completely eliminated. Secondly, high quality power and ground routing for the entire chip is achieved automatically. Also, inductances of all wires on the die are highly predictable and low, since they each have a current

[1] Appendix 1 shows a table describing the characteristics of several DSM processes. We base the results and simulations of this work on this table.

return path a minimum distance away.

The disadvantage of the scheme of [1] was an increased area utilization compared to the standard cell methodology.

This paper describes a new layout and synthesis flow which utilizes the best features of the scheme of [1], with a much lower area utilization. In addition, the new scheme has several other advantages as well.

The essential idea of this paper is to implement the logic circuit in the form of a network of multi-output PLAs. These PLAs are highly immune to crosstalk problems, as we will describe. Also, these PLAs are extremely fast (at least 2× faster than the corresponding standard cell implementation) and very area-efficient (about 2.5× smaller than the corresponding standard cell implementation). These PLAs are placed on a regular grid, with the routing region between them metallized as in [1]. Metal layers that are not utilized in the layout of the PLA are gridded maximally throughout the die. This gives rise to a highly efficient power and ground distribution network throughout the die. Since the PLAs are placed on a regular grid, each PLA must have a bounded width and height.

The synthesis algorithm that generates the network of PLAs essentially maps the nodes of the circuit into PLAs such that the total width of each PLA is bounded. Also, it ensures that the number of cubes of each PLA is bounded as well.

The output of the synthesis program is placed and routed using VPR [2]. VPR places the PLAs on a regular grid, using a simulated annealing based algorithm. The final routing is done by this tool as well. The routing tool is tailored towards FPGA routing, which is a much more constrained routing problem than the general area routing problem in ICs. As a result, we expect our area results to improve when we use a router tailored towards integrated circuit routing.

The resulting flow that we propose has the benefits of high crosstalk immunity, high speed, low area overhead, very quick design turnaround times, and highly predictable parasitics. Buffer insertion is easily achieved in our scheme, since power and ground are routed all over the routing area in a regular gridded fashion. Also, circuit delays can be pre-calculated, since they are a function of the maximum number of levels of PLAs between the primary inputs and

primary outputs, and the delays of individual PLAs. Since both these are known after synthesis is completed, we can have accurate timing estimates immediately following a synthesis iteration.

The remainder of this paper is organized as follows. Section 2 discusses some prior work in this area. Section 3 describes our layout and synthesis flow. Layout, synthesis and placement/routing in our flow are treated separately, with results of each immediately following it. Finally, in Section 4, we make concluding comments and discuss further work that needs to be done in this area.

2 Previous Work

The problems faced in designing and manufacturing chips with DSM processes are not entirely new. Over the last few years, these problems have slowly become significant enough that academia and industry have begun to take notice.

In the past, some techniques to ensure characterizability and reliability of designs have been proposed and implemented. For example, in the DEC Alpha chip [3], metal layers 3 and 6 (in a 6 layer metal process) were exclusively dedicated to power and ground routing. In this scheme, a signal on any of the remaining layers has a constant voltage plane either above or below it, thus ensuring that a majority of its total capacitance is to a node of constant voltage. In the absence of such a scheme, signal wires on higher metal layers would have very small capacitances to a node of constant voltage, on account of their large distance from the substrate. Also, these wires would have relatively large capacitances to neighboring signal conductors. In such a situation, if the neighboring conductor would undergo a signal transition, it could couple a large noise voltage into the signal of interest, resulting in a loss of signal integrity. It could also result in altered delays when the neighboring signal conductor undergoes an signal transition while the wire of interest is undergoing a transition.

The above solution worked well for the DEC Alpha microprocessor. However, with decreasing feature sizes, we find that the capacitance of a wire to its neighboring wire is an increasing fraction of the total capacitance of the wire. This was shown analytically and empirically in [1]. As a consequence, a solution such as the DEC Alpha's solution

is not likely work for smaller feature sizes.

Another solution was proposed by [4]. In this work, the concept of *digital sensitivity*, which accounts for signals that have opposing transitions that occur nearly simultaneously. They use this concept to analyze crosstalk effects in a circuit. Additionally, they introduce layout synthesis tools for crosstalk avoidance. This work attempts to modify the existing design process to tackle the problems of crosstalk and signal integrity. Our work has a more global scope, resulting in further benefits like easy characterizability, regular power and ground routing, and low on-chip signal inductances.

Other regular layout structures have been used in the past, for reasons of ease of programmability, and shorter times to market. However these structures have not been used to address DSM circuit problems.

The scheme of this paper is motivated by the layout fabric introduced in [1]. Since our scheme retains the advantages of [1], we list the salient features of this work below.

- First of all, the capacitance of any signal wire is entirely predictable, since the immediate neighbors of any signal wire are always one power and one ground wire. The capacitance of a signal wire to the nearest *signal* wires is negligible, and was shown to be at least an order of magnitude smaller than the capacitance to the wire's immediate neighbor.

  3-Dimensional parasitic capacitances were extracted for the configuration of Figure 1. Using these parasitics, a SPICE model was constructed of three wires routed at minimum space on Metal2, for a length of 50 μm, in the 0.1 μm process. Each wire was driven by a 10$\times$ minimum inverter. In this scenario, the delay of the central wire was found to vary between 14.11ps to 28.03ps, depending on whether its neighbors underwent a like or unlike transition. It is apparent that this delay range will increase with decreasing feature size since the capacitance of a wire to its neighboring wires becomes increasingly dominant with decreasing feature size.

  However, with the scheme of [1], the delay effect on any signal, of its neighboring signal wires undergoing transitions was found to be +/- 2% of its nominal delay. This is because of the significant reduction of the

capacitance of a signal wire to its nearest signal wire.

- Secondly, the routing of power and ground for the entire chip is simultaneously achieved in this scheme. At every point where a power (or ground) wire on metal layer i overlaps with a power (or ground) wire on metal layer $i-i$, a via is introduced. Given the large number of such intersections, the power and ground resistance at every point is held very low, and almost constant. For the 0.1 μm process, the power and ground resistance was about 20$\times$ smaller than for a traditional standard-cell style of layout. The variation of power and ground resistance at different parts of the die was found to be very low (38%). Also, it was observed that this value of resistance was insensitive to local breaks in the gridding pattern on the lowest two metal layers, since the major contribution towards the low power and ground resistance was made by the upper metal layers.

- Thirdly, each signal has a current return path which is adjacent to it, hence its inductance is very low and extremely predictable. In the existing layout paradigms of today, the inductance of signals can vary greatly, since different wires have current return paths at a different distance apart, depending on the exact layout of the circuit in the neighborhood of the signal.

- Our scheme results in a constant density of wires in any region of the chip, and on every metal layer. This has the added advantage that it results in a much tighter control of t_{ins}, which in turn translates into a much tighter control of capacitances.

3 Our Approach

In order to maximally obtain the benefits of the scheme of [1], while reducing the associated area overhead, we propose to implement the circuit as a network of PLAs. Each PLA is a multi-output structure, which is naturally immune to crosstalk. Each PLA can implement its logic with very high density and high speeds, as we will show in this section. We place these PLAs in a regular gridded fashion, and route them together. The routing region between PLAs is organized as in [1], giving rise to highly predictable, crosstalk-immune routes. Metal layers that are not

utilized in the layout of the PLA are gridded maximally throughout the die, using the technique of [1]. This gives rise to a highly efficient power and ground distribution network throughout the die. The local breaks in the power and ground gridding structure which occurs where the PLAs are placed was simulated, and determined to cause a negligible change in the power and ground resistance. This is because the contribution of upper metal layers to a low power and ground resistance outweighs that of the lowest metal layers.

The remainder of this section is organized as follows. Section 3.1 describes our design flow using a network of PLAs, including a detailed characterization of the PLAs we use. Our synthesis algorithm is described in Section 3.2. Details of our PLA folding and node clubbing algorithms are discussed in this section. Finally, Section 3.3 describes the placement and routing flow we used, and reports the overall area results that we obtained.

3.1 PLAs in DSM VLSI Design

In this section, we make the case for PLAs as a layout methodology for use in DSM circuit designs. We show that these structures are naturally crosstalk-immune, and hence a perfect candidate for a DSM design methodology. In Section 3.1.1 we describe the basic structure of our proposed PLAs. In Section 3.1.2 we discuss the system view of the network of PLAs and the constraints it imposes on the synthesis tools. Section 3.1.3 describes experiments to characterize the electrical and area characteristics of these PLAs, as compared to a standard-cell based design methodology. Section 3.1.4 gives some arguments why the PLA implementation is ideally suited for DSM applications.

3.1.1 PLAs in DSM VLSI Design - the Layout View

First, we define some terminology that will be used in the sequel. Consider a PLA consisting of n input variables $x_1, x_2, \cdots, x_n$, and m output variables $y_1, y_2, \cdots, y_m$. Also, let k be the number of rows in the PLA. A *literal* l_i is defined as an input variable or its complement.

Suppose we want to implement a function f represented as a sum of cubes $f = c_1 + c_2 + \cdots + c_k$, where each cube $c_i = l_{i_1} \cdot l_{i_2} \cdots l_{i_r}$. We consider PLAs which are of the *NOR-NOR* form. This means that we actually implement f as

$$\overline{f} = \overline{c_1 + c_2 + \cdots + c_k}$$

$$\overline{f} = \overline{\overline{c_1} + \overline{c_2} + \cdots + \overline{c_k}}$$

$$\overline{f} = \overline{(\overline{l_{1_1}} + \overline{l_{1_2}} + \cdots + \overline{l_{1_r}}) + (\overline{l_{2_1}} + \overline{l_{2_2}} + \cdots + \overline{l_{2_r}}) + \cdots + (\overline{l_{k_1}} + \overline{l_{k_2}} \cdots \overline{l_{k_r}})}$$

Notice that this is in a NOR-NOR form, where the PLA output $\overline{f}$ is a NOR of a series of expressions, each corresponding to the NOR of the complement of the literals present in the cubes of f. In the PLA, each such expression is implemented by *word lines*, in what is often termed as the *AND plane*. Without loss of generality, let us assume that these word lines run horizontally. Literals of the PLA are implemented by vertical-running *bit lines*. For each input variable, there are two bit lines, one for each of its literals. The output of the PLA is implemented by *output lines*, which also run vertically. This portion of the PLA is often referred to as the *OR plane*.

As we saw in Section 2, crosstalk between signal wires is a big problem in DSM designs and is expected to get worse with shrinking process feature size. We showed that if no special layout techniques are employed, there can be a 1.94:1 variation in the delay of a 50 μm wire in a 0.1 μm process, depending on whether its neighbor switches in the opposite direction, or in the same direction. We now proceed to show how a careful layout of PLAs can result in crosstalk-immunity of signals, at no additional area cost.

The schematic view of our PLA core is shown in Figure 2. We pre-charge our PLA word-lines, and pre-discharge the output lines, as shown in Figure 2. We observe that if we utilize a NOR-NOR PLA to implement a logic function, each word line of the PLA switches from high to low at the end of any computation, if it switches at all. As a result, there is no delay deterioration effect due to crosstalk with neighboring word lines, and hence there is no necessity to place VDD or GND shield wires between the word-lines. In the vertical direction, complementary versions of any input are shielded from each other by a GND wire, which is required by devices in the AND plane in any case.

As shown in Figure 3, we implement word lines in METAL2. In the vertical direction, we separate the literals of each input variable by a GND wire, which is used by the word line pull-down devices that are connected to either literal as required. This automatically shields the literals of any variable from each other, at no additional area cost. Each group of 3 wires consisting of two literals and a GND wire utilize METAL1. They are separated from their

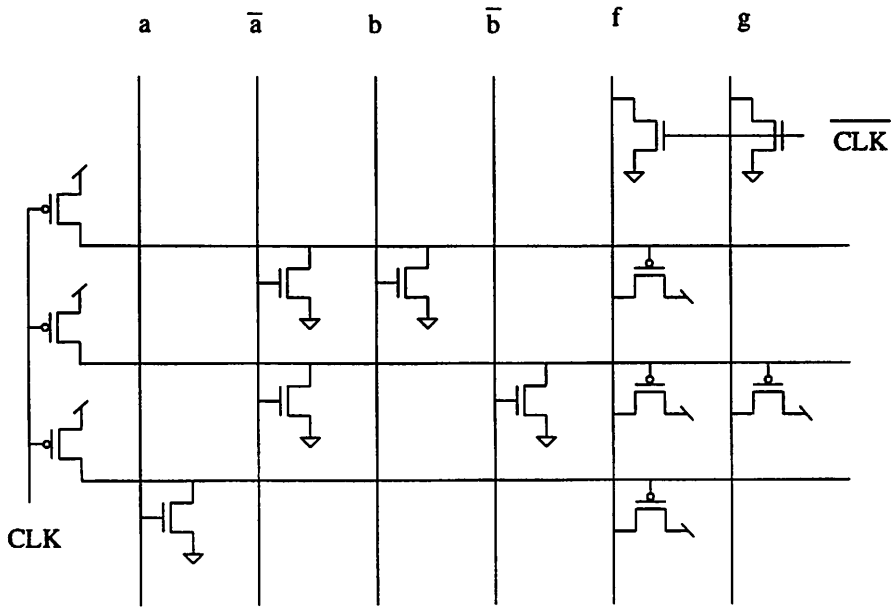

Figure 2: Organization of the PLA core

neighboring group by a blank track, which is used to contact the word line pull-down devices to the corresponding word line.

As a result, by using a NOR-NOR PLA as the layout building block of our methodology, we incur no extra area penalty, either in the horizontal or vertical direction. At the same time, the PLA structure is crosstalk immune, which makes it an ideal choice for our methodology.

Observe that for a PLA with n inputs and m outputs, the width of the core is $4 \cdot n + 2 \cdot m$, since the each input requires 4 vertical tracks, and each output requires 2.

3.1.2 PLAs in DSM VLSI Design - the System View

When a circuit is implemented as a network of PLAs of the kind described in Section 3.1.1, certain additional factors need to be taken into account.

First of all, in order that the network of PLAs correctly computes the output, we need to impose some constraints on the individual PLAs in the network. It should not be the case that some input of a PLA p_i depends upon an output of a PLA p_j, while some input of p_j depends on some output of p_i. This cyclic dependency between PLAs implies

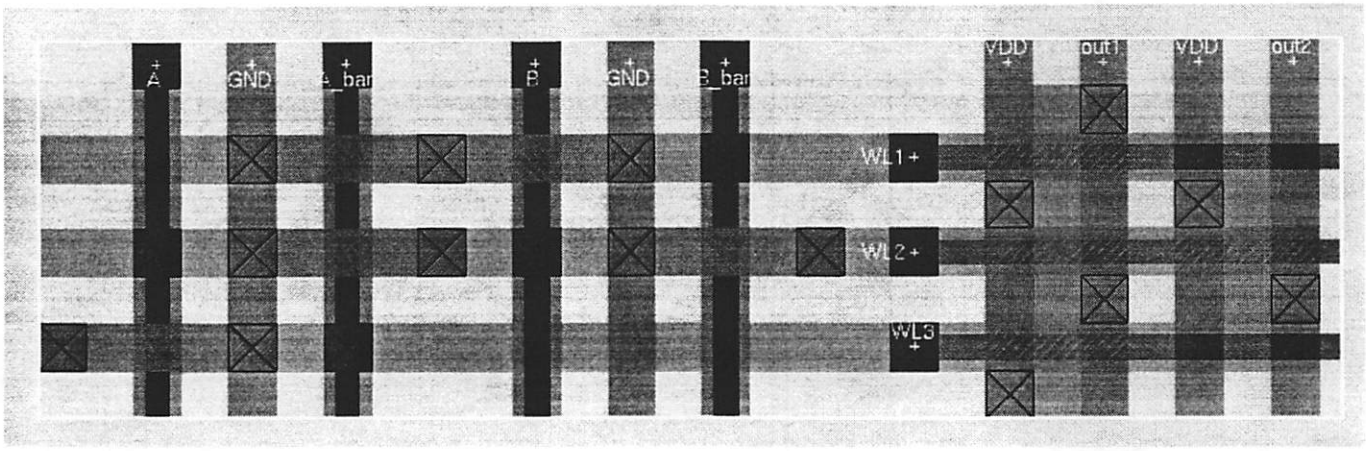

Figure 3: Layout of the PLA core

that p_i cannot compute its function until the outputs of p_j settle, and vice versa.

Definition 1 *The* PLA dependency graph *G(V, E) of a network of PLAs is a directed graph such that* $V = \{v_1, v_2, \cdots, v_r\}$, *where each vertex* v_i *corresponds to a PLA in the network.*

$(v_i, v_j) \in E$ iff *an output of PLA* p_i *is an input to PLA* p_j.

Definition 2 *A network of PLAs is* valid *iff its corresponding PLA dependency graph G(V, E) is acyclic.*

Definition 3 *For a valid network of PLAs, we define a* level *of each PLA p as follows.*

If p depends purely on primary inputs of the circuit, level(p) = 0

Otherwise, level(p) = max_fanin_level + 1, *where* max_fanin_level *is the maximum of the levels of all the PLAs whose outputs drive the inputs of p.*

If the PLA dependency graph is acyclic, the computation of any PLA p is begun only after the computation of all the PLAs whose outputs feed p is completed. This ensures valid operation of the network.

Secondly, since the word lines are precharged, it must be seen to that their charge is not lost by glitching activity on inputs that feed the PLA at a given level i. This is avoided by gating the inputs of each PLA p with a signal that indicates that the slowest PLA whose output feeds an input of p has completed its computation.

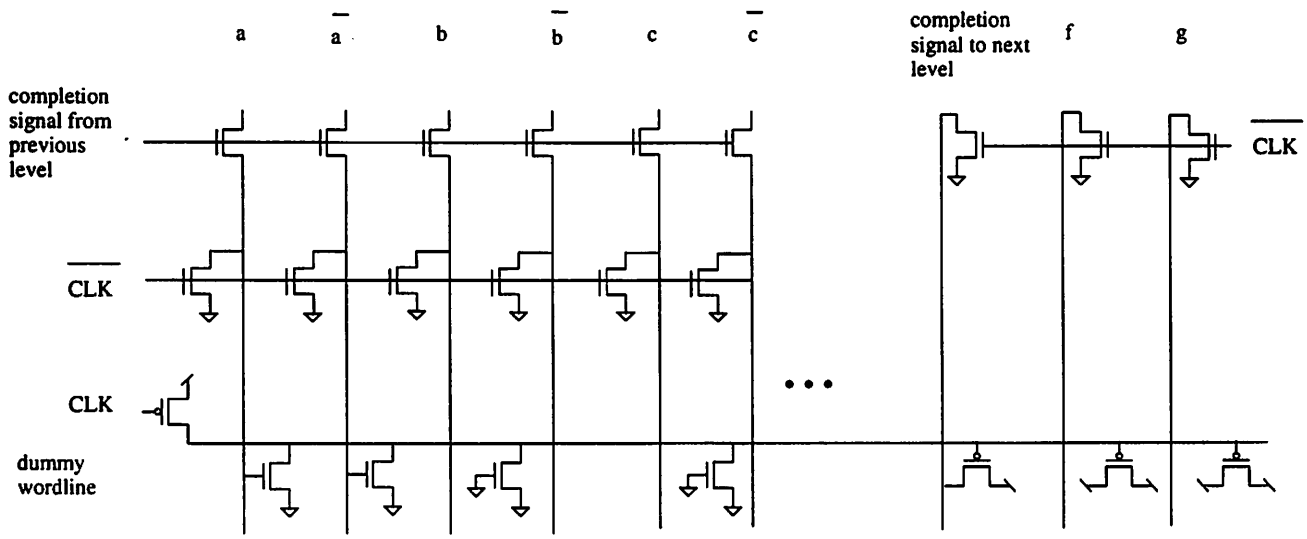

Figure 4: Generation of completion signal for a PLA

Each PLA p with level j generates a completion signal. Given the regularity of the PLA structure, the worst case delay of each PLA is easily known. It corresponds to the delay of a single word-line pulldown device discharging a (maximally loaded) word line and a single output device charging the output line. This completion signal is generated with an overhead of one additional word line and one additional output line in each PLA. This is schematically shown in Figure 4.

Figure 5 shows the relative orientation of precharge devices, muxes and drivers in our layout of each PLA. We implemented the input signals buffers and inverters, and the output drivers, outside the footprint of the PLA (ie in the routing channel). This gives rise to a much lower control overhead for our PLA cells. The only effect it has on the routing channel is the introduction of one via per driver. We were able to complete the layout of all control signals with an additional cost of 4 horizontal tracks.

3.1.3 Characterization of PLAs

Figure 6 shows the pattern of wires occuring within the core of our PLAs. Capacitive parasitics for the wires within our PLA core were extracted using a using a 3-dimensional parasitic extractor called *Space3D* [5]. The input to *Space3D* is a 3-dimensional circuit layout (dimensions are shown in Table 5 in Appendix 1), and the output is the

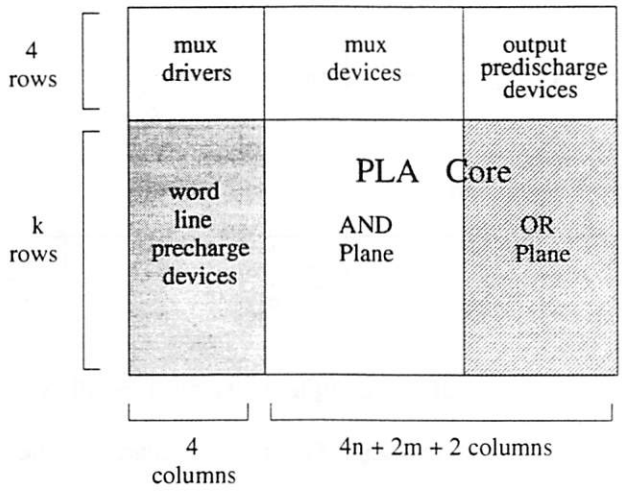

Figure 5: Layout Floorplan of PLA

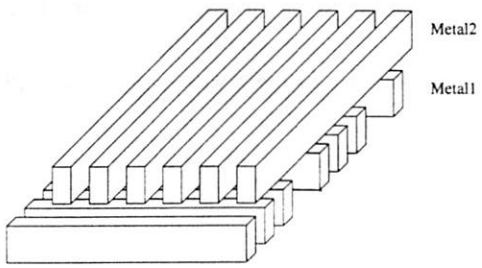

Figure 6: Arrangement of conductors in the PLA core

Layer	$C^1_{i,i}$	$C^2_{i,i}$	$C_{i,0}$	$C_{i,i+1}$
1	47.17	14.57	13.72	15.78
2	48.37	-	0.77	5.96

Table 1: 3-Dimensional Parasitics for Figure 6 (10^{-18}F per μ)

value of the parasitics between different features of that layout. *Space3D* uses a boundary element method to compute interconnect capacitances.

The results of these extractions for a 0.1 μm process are shown in Table 1. In this table $C^1_{i,i}$ refers to the capacitance between two metal conductors on the same level i, which are separated by minimum spacing. $C^2_{i,i}$ refers to the capacitance between two such conductors separated by twice the minimum spacing. $C_{i,0}$ is the capacitance of a conductor to the substrate, and $C_{i,i+1}$ is the capacitance of a conductor on level i to other conductors on level $i+1$.

In order to accurately characterize our PLAs with varying m, n, and k, we performed several SPICE [6] runs. For these runs, we used the PLA device sizes from our layout experiments. The layout gave us strict upper bounds on the size of each device in the SPICE run. Results from some of these runs are provided in Table 2. Runs that correspond to a constant PLA width are grouped together horizontally. In Table 2, the "Area" column lists the PLA area, in square grids. This area includes the overhead associated with generating control signals for the PLAs, as described in Section 3.1.2. For a PLA with n inputs, m outputs, and k word-lines, the PLA core size is $(4 \cdot n + 2 \cdot m) \cdot k$. The area of the PLA, including control signals, is $(4 \cdot (n+1) + 2 \cdot (m+1)) \cdot (k+4)$. This is because we need 4 vertical tracks for precharging the word-lines and two vertical tracks to generate the completion signal. Also, we need 4 horizontal tracks for the bit-line muxes and output pre-charge devices. The ratio of the PLA area to the PLA core area is listed under the "Overhead" column. The "Delay" column lists the worst case delay of the PLA. For this condition, we simulate a maximally loaded word-line, which is discharged by a single pull-down device. The output line is pulled up by a single output pull-up device. The loads on the line are n diffusion loads in the AND plane, and m gate loads in the OR plane. Parasitics from Table 1 were utilized in the simulation.

We notice that with constant PLA width, there is an increase in delay and power as m increases. This is because

adding a output line increases the load on a maximally loaded word-line significantly, since the load added is a gate load, and also the size of the device is larger than minimum.

Also, the delay and power are higher for larger k values, since longer bit-lines have larger capacitances.

To compare the PLA style of layout with the Standard-cell style, we took four examples and implemented them in both styles. After performing some technology-independent optimizations in SIS [7], we mapped the circuit using a library of 14 standard cells. This was then placed and routed using the OCT [8] toolset. We use the *wolfe* tool within *OCT* to do the placement and routing. *Wolfe* in turn calls *TimberWolfSC-4.2* [9] to do the placement, and YACR [10] to do the routing.

The results of this comparison are listed in Table 3. For each layout style, D refers to the delay in picoseconds, A refers to the layout area of the resulting implementation in square pitches, and P refers to the power consumption. The A^P column denotes the area overhead incurred per PLA, in implementing its control circuitry. Note that for the Standard cell layout style, D and P values are the maximum values obtained after simulating about 20 input vectors. In general the worst case delay and power could be much worse. However, in the case of the PLA layout style, the D and P are worst-case values.

In spite of this, the PLA layout style shows impressive improvements over the Standard cell layout style. The PLA layout utilizes between 0.27 and 0.75 times the area of the Standard cell layout. The average area utilization is 0.42 times that of the Standard cell layout style, which is an impressive reduction. The delay value for the PLA is on average 0.5 times that of the Standard-cell implementation. This is despite the fact that we dont account for wire capacitances in the Standard cell implementation, which would only increase its delay. For the PLA implementation, we modelled parasitic capacitances using Table 1.

The power consumption of the PLA is usually larger than that of the Standard cell implementation, mainly because of the fact that capacitances are charged and discharged on every cycle. PLAs with large values of k exhibit a much higher power consumption. This suggests that when performing a decomposition of a circuit into a network of PLAs, we should use a value of k within 25, a conclusion supported by Table 2.

n	m	k	Area	Overhead	Delay	Power
9	2	20	1104	1.380000	131.68830	0.000333269
8	4	20	1104	1.380000	151.80617	0.000397285
7	6	20	1104	1.380000	169.69595	0.000470769
11	3	20	1344	1.344000	147.97938	0.000366811
10	5	20	1344	1.344000	165.35169	0.000453139
9	7	20	1344	1.344000	182.00918	0.000452887
8	9	20	1344	1.344000	198.35342	0.000542754
13	3	20	1536	1.324138	151.66726	0.000379504
12	5	20	1536	1.324138	169.70882	0.000446495
11	7	20	1536	1.324138	185.55882	0.000485877
10	9	20	1536	1.324138	201.68339	0.000557410
9	2	30	1564	1.303333	142.27610	0.000528303
8	4	30	1564	1.303333	162.40827	0.000618604
7	6	30	1564	1.303333	181.27023	0.000675271
11	3	30	1904	1.269333	157.92366	0.000573887
10	5	30	1904	1.269333	177.47087	0.000666201
9	7	30	1904	1.269333	195.12406	0.000735019
8	9	30	1904	1.269333	211.38385	0.000811727
13	3	30	2176	1.250575	162.17471	0.000588542
12	5	30	2176	1.250575	181.17960	0.000678957
11	7	30	2176	1.250575	198.87404	0.000770631
10	9	30	2176	1.250575	215.15343	0.000841833
9	2	40	2024	1.265000	152.01090	0.000741870
8	4	40	2024	1.265000	173.48348	0.000863662
7	6	40	2024	1.265000	192.58371	0.000944179
11	3	40	2464	1.232000	168.62387	0.000797166
10	5	40	2464	1.232000	188.21763	0.000912675
9	7	40	2464	1.232000	206.41055	0.001016865
8	9	40	2464	1.232000	223.97746	0.001111249
13	3	40	2816	1.213793	173.24674	0.000818470
12	5	40	2816	1.213793	192.48303	0.000916195
11	7	40	2816	1.213793	210.30422	0.001023576
10	9	40	2816	1.213793	227.50960	0.001135595

Table 2: Variation of Delay, Power and Area of PLA with m, n, k

	PLA implementation							Standard Cell			Ratios		
Example	m	n	k	D	A	A^P	P	D	A	P	D	A	P
cmb	16	4	15	160.6	53.3k	37.2	3.4	304.1	194k	6.15	0.528	0.2757	0.553
cu	14	11	19	223.6	69.5k	30.36	5.5	421.9	218k	4.24	0.530	0.3188	1.297
x2	10	7	17	179.8	45.3k	37.2	3.9	291.2	141k	1.82	0.617	0.3212	2.143
z4ml	7	4	59	189.0	95.2k	24.57	13	576.3	128k	3.17	0.328	0.7437	4.100

Table 3: Comparison of Standard-Cell and PLA implementation styles

To estimate the effect of crosstalk between literals of neighboring variables in the PLA, we simulated a PLA with 40 cubes. Let l_i be a literal of variable x_i, and l_{i+1} be a literal of variable x_{i+1}. Assume l_i and l_{i+1} are separated by a blank track. In this situation, there is a 1:1.0156 delay variation for l_i, depending on whether l_{i+1} switches in the opposite or similar direction. This delay variation is small enough to be disregarded.

3.1.4 Why PLA-based Layout is Superior to Standard Cell Layout

The reason why PLAs result in very favorable area and delay characteristics compared to a Standard cell layout are the following:

- First, the PLAs implement the functions in a 2-level form, which results in superior delay characteristics as long as k is bounded. On the other hand, in a Standard cell implementation, considerable delay is incurred in traversing the different levels (ie gates) of the design.
- In DSM processes, it is often stated that an increasing fraction of a signal's delay is attributable to wiring. In the PLA implementation scheme, however, local wiring is collapsed into a compact 2-level core, which is naturally crosstalk-immune. Hence wiring delays are reduced.
- Most of the devices in the PLA core are minimum-sized, giving rise to extremely compact layouts. Such is not the case for Standard cell layouts.
- In the PLA core we implemented, only NMOS devices are used in the AND plane, and only PMOS devices are used in the OR plane. As a result, devices can be placed extremely close together in either plane. However, in

a Standard cell layout, both PMOS and NMOS devices are usually present in each cell. As a result, the PMOS and NMOS diffusion spacing requirement results in a loss of layout density.

3.2 Synthesis of a Network of PLAs

Problem Definition: Given an arbitrary logic circuit C, find a decomposition of C into a network N of interconnected PLAs, subject to :

- the PLA dependency graph of the resulting network N of PLAs is acyclic.
- each PLA should have a height no larger than a specified maximum.
- each PLA should have a width no larger than a specified maximum.

Our algorithm decomposes C into a network of PLAs which is acyclic by construction, and is shown in Algorithm 1. We begin by performing some technology independent optimizations on C . Next, we decompose C into a network C^* of nodes with at most p inputs. Now C^* is *levelized*, meaning that primary inputs are assigned a level 0, and other nodes are assigned a level equal to the maximum level of all their inputs plus one. The resulting levelized list of nodes is sorted and placed into an array L.

Now we successively club together nodes from L, such that the resulting PLA implementation of the clubbed nodes N^* does not violate the constraints of PLA width and height. This check is performed in the *check_PLA* routine, which calls *espresso* on the PLA, to minimize the number of cubes. Additionally, *check_PLA* also calls a *PLA folding* routine, which attempts to fold PLA inputs so as to implement more complex PLAs in the same area. It also checks that the final PLA, after folding and simplification using *espresso*, satisfies the maximum height and width constraints. If so, we attempt to club another node from L, otherwise we append the last PLA which satisfied the height and width constraints to the output.

We performed extensive benchmarking of the PLA network generation code, and found that a good choice of parameters was $p = 5$, and *max_width* = 15 to 25. Increasing *max_width* beyond 30 did not usually result in a reduction

Algorithm 1 PLA network generation algorithm

```
C = simplify_network(C)
C* = decompose_network(C, p)
L = levelize_and_sort_nodes(C*)
N* = 0
RESULT = 0
while first_element(L) != NIL do
  N* = N* ∪ first_element(L)
  P = make_PLA(N*)
  if check_PLA(P) then
    continue
  else
    Q = remove_last_element(N*)
    RESULT = RESULT ∪ N*
    N* = Q
  end if
end while
```

in the total number of PLAs generated. Folding the PLAs resulted in a decrease of between 20% and 50% in the total number of PLAs required for a network.

Our *PLA Folding* algorithm folds only PLA inputs. It constructs a list of candidates to fold, and then assigns a figure of merit to each candidate. This figure of merit awards folds that allow subsequent folds to proceed without hinderance. We implemented several heuristics based on this idea, and found that no heuristic consistently outperformed the others, for the small size of PLAs that our algorithm encounters

Detailed results of our synthesis algorithm are not provided for brevity.

3.3 Placing and Routing a Network of PLAs

The synthesized network of PLAs was placed and routed using a FPGA routing tool called *VPR* [2]. Although this tool was written with an FPGA architecture in mind, the placement problem it addresses is exactly that which we face in our flow.

VPR performs placement by simulated annealing. The placed result is routed using an FPGA routing methodology. Therefore the results of routing are pessimistic; these results would improve when we use a router applicable for custom integrated circuit design. In the long term we plan to use VPR only for placement.

	Std Cell	Our scheme						
Example	*Area*	*Width*	*Height*	*Folds*	*Area*	W_{max}	H_{max}	*#PLAs*
C432	34.7K	45.67/50/34	20.00/25/13	6.42/9/2	38.6K	50	25	12
C499	72.7K	47.25/58/32	20.62/25/12	5.94/9/4	70.0K	60	25	16
C880	63.8K	53.30/60/44	16.20/25/8	8.60/11/4	90.3K	60	25	20
alu2	49.2K	51.85/60/36	17.69/2012	6.92/12/2	51.1K	60	20	13
apex6	172.2K	52.88/60/22	17.09/24/4	7.44/12/0	180.8K	60	25	32
count	16.6K	35.43/40/18	14.57/23/4	4.86/7/1	14.7K	40	30	7
decod	5.3K	52/52/52	16/16/16	0/0/0	1.1K	52	16	1
pcle	6.4K	50/58/42	18/20/16	4.50/6/3	3.61K	60	20	2

Table 4: Area results for our scheme

Results showing the area penalty of our overall layout and synthesis flow are shown in Table 4. This table describes the results for a series of benchmarks, and compares the area of a Standard cell implementation (column 2), and our approach (column 6). All areas are in square pitches. Columns 3, 4, and 5 list the average/minimum/maximum width, height and number of folds of the resulting PLA network. Columns 7 and 8 list the maximum width and height specified for the PLAs, and column 9 lists the number of PLAs in the final decomposition.

We note that over all these examples, the area overhead of our method was low. Excluding the last two examples, the average area overhead was 7.7%. The last two examples could be decomposed into trivial networks of PLAs (with 1 and 2 PLAs respectively) and hence showed very impressive area ratios. We expect better results using a routing tool specifically targetted towards IC routing, since the FPGA-style routing places additional constraints on wires that are not present for traditional integrated circuit routing.

4 Conclusions and Future Work

We have presented a layout and design flow for use in DSM circuits. In this methodology, we decompose the original circuit into a network of PLAs of bounded width and height. We implement these PLAs in a noise-immune fabric, achieving impressive area and speed characteristics. We place these PLAs in a regular grid, with the routing between PLAs done using a regular layout fabric as in [1]. The advantages of our method are as follows:

- High speed. Each PLA is shown to be at least 2× faster than its corresponding standard-cell based circuit implementation.
- Low area of PLAs. Each PLA is shown to be 2.5× smaller than its corresponding standard-cell based circuit implementation.
- Elimination of cross-talk and signal integrity problems that are common in DSM designs.
- Power and ground routing is done implicitly, and not in a separate step in the design methodology.
- Power and ground resistances are very low and vary much less compared to the power and ground distribution used in the standard cell methodology.
- Variations in delay of a signal wire due to switching activity on its neighboring signal wires is less than 2%, compared to a 1.94:1 variation using conventional layout techniques.
- Smaller and uniform inductances for all wires on the chip, compared to larger and unpredictable values using the existing layout styles.
- Rapid design turn-around time due to highly regular structures and regular parasitics.

We believe that this technique will significantly simplify the design of chips with minimum feature sizes in the DSM range.

In the future, we plan to try out other regular structures. We also plan to try out better place and route tools that better exploit the regularity of a design which uses our technique. We are exploring the idea of relaxing the gridding on certain metal layers, so as to allow for better logic densities. The regularity of geometry, parasitics and delays that is the core of our scheme opens up many new CAD and synthesis problems, which we plan to motivate and tackle. We are exploring several other decomposition and folding algorithms other than those described, including unate network decompositions. A more aggressive utilization of free tracks in the PLAs of our design can result in further area reductions, which we plan to explore. Alternative PLA styles will also be explored.

5 Appendix 1: Interconnect Parameters for DSM Processes

To obtain estimates of interconnect geometries for future processes, we used the VLSI interconnect trends from the NTRS [11], as well as those from Sematech [12]. From these, we came up with our "strawman" interconnect geometry parameters. Next we circulated these within some leading semiconductor manufacturing companies, to obtain further critiques on our estimates. The outcome of this exercise is listed in Table 5. This table lists various process parameters for three processing generations that we consider. Here, V_{DD} refers to the power supply voltage, L_{eff} is the effective channel length of a transistor and t_{ox} is the gate oxide thickness of a transistor. For each conductor, H is the height, W its width, "space" refers to the minimum allowable spacing and t_{ins} is the thickness of the dielectric between metal layers. It is assumed that all wires are made of copper.

Of the three processes we consider, the first process is used in aggressive circuit designs today, while the remaining processes are still a few years from being used.

References

[1] S. Khatri, A. Mehrotra, R. Brayton, R. Otten, and A. Sangiovanni-Vincentelli, "A noise-immune VLSI layout methodology with highly predictable parasitics," Tech. Rep. UCB/ERL M98/24, Electronics Research Laboratory, University of California, Berkeley., April 1998.

[2] V. Betz and J. Rose, "VPR: A new packing, placement and routing tool for FPGA research," in *Proceedings of the International Workshop on Field Programmable Logic and Applications*, 1997.

[3] B. A. Gieseke *et al.*, "A 600MHz Superscalar RISC Microprocessor with Out-of-Order Execution," in *Digest of Technical Papers, International Solid State Circuits Conference*, 1997.

[4] D. A. Kirkpatrick, *The Implications of Deep Sub-micron Technology on the Design of High Performance Digital VLSI Systems*. PhD thesis, University of California at Berkeley, 1997.

[5] "Physical Design Modelling and Verification Project (SPACE Project)." `http://cas.et.tudelft.nl/research/space/html`.

[6] L. Nagel, "Spice: A computer program to simulate computer circuits," in *University of California, Berkeley UCB/ERL Memo M520*, May 1995.

[7] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "SIS: A System for Sequential Circuit Synthesis," Tech. Rep. UCB/ERL M92/41, Electronics Research Laboratory, Univ. of California, Berkeley, CA 94720, May 1992.