RC SYNTHESIS OF VLSI MACROCELLS*

DAVID KNAPP

and

MELVIN A. BREUER

DIGITAL INTEGRATED SYSTEMS CENTER REPORT

DISC/83-3

DEPARTMENT OF ELECTRICAL ENGINEERING-SYSTEMS
UNIVERSITY OF SOUTHERN CALIFORNIA
LOS ANGELES, CALIFORNIA 90089-0781

MAY 1983

*This work was supported in part by the National Science Foundation under grant ECS-8005957 and by the Department of the Army, Ft. Monmouth, New Jersey, under contract no. DAAK20-80-C-0278.

RC Synthesis of VLSI Macrocells

1 ABSTRACT

A technique for implementing rectangular macrocell functional blocks with variable height/width ratios is presented. The technique is applicable to cellular "vector" functions that have limited intercellular connections. An analysis of the dimensions of such arrays is presented. When the macro is large in comparison to the cells it is composed of, its area is a function of its shape, almost independently of the aspect ratio of the cells.

2 INTRODUCTION

The macrocell approach to the design of integrated circuits is sometimes complicated by incompatibilities between the placement of the macrocells and their shape, as shown in Figure 1.

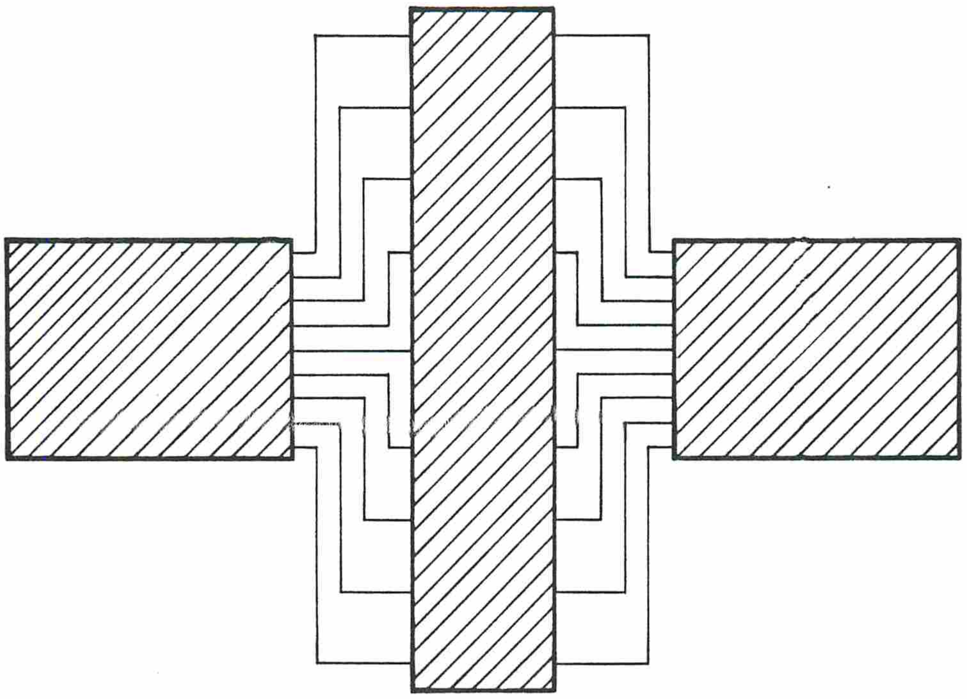


Figure 1: Global Inefficiency Caused by Minimum-Size Macro

In situations like the one of Figure 1, the cost (in area) of using the macro in the middle can be expressed as three nearly independent costs: first, there is the cost of the macro itself, i.e. its area; second, there is the cost of the wiring between the macro and its neighbors, which in this case is quite large; and third, there are the costs associated with the environment of the three macros shown in the figure. The third set of costs, the environment costs, can be quite large if the macro is in a space-critical path, because the entire chip may have to be stretched in order to accomodate the one macro. Such a situation is shown in Figure 2. In circumstances like these, there are two alternatives: one can change the placement, or one can change the macro.

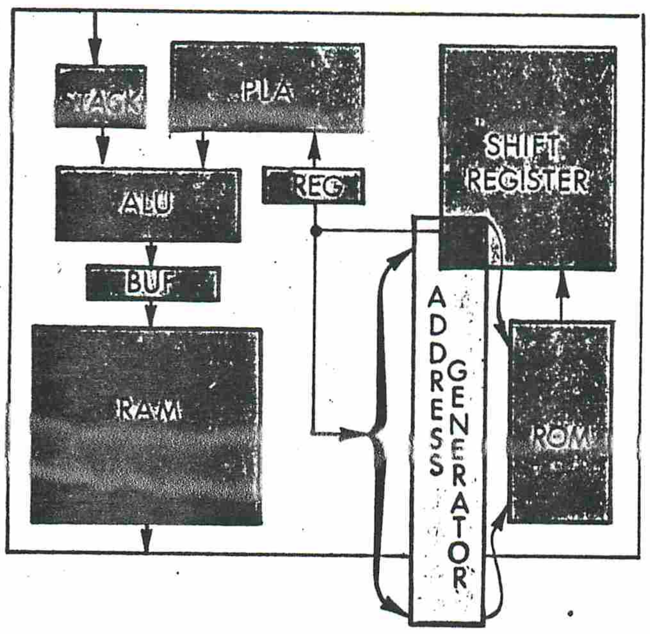


Figure 2: Overall Nonoptimality Caused by Inappropriate Macro Shape

Changing the placement is not always an easy matter; it is often a function of the intermacro wiring on the chip, and there is no guarantee that the situation will be improved by rearranging the macros. It would therefore be convenient to have a program that would construct macros in accordance with the shapes and sizes demanded by particular situations; it would also be convenient to be able to predict the dimensions of the macro accurately. For macros that implement certain classes of functions, such estimation and construction programs are not difficult to implement.

The Row-Column (RC) methodology described in this paper is one approach to this general problem, applicable to macros that meet the following criteria:

- The function implemented by the macro must be a vector function, i.e. operating on a vector or vectors of bits;
- The function must be decomposable into an array of identical or near-identical cells;
- The ith cell may only be connected to the (i+1)st and (i-1)st cells if there are any intercell interconnections at all. Common connections, e.g. power, control, and clock signals, are exempt from this restriction.

Examples of functions that meet these criteria are:

- parallel adders that have a carry chain and at most limited lookahead logic
- shift and storage register arrays
- bus multiplexers and decoders

The RC approach consists of replicating a representative, or **kernel** cell, in rows and columns to form a rectangular array. Because the array is regular, it is easy to generate all the necessary intramacro wiring automatically, thereby providing the designer with a macro the internal details of which need not be considered (see Figure 3). By varying the numbers of rows and columns in the array, its aspect ratio (defined as height divided by width) can be varied on a large scale; by varying the shape of the original kernel cell, finer variations can be achieved. There is a penalty for this flexibility: for a given cell, the overall area of RC macros is usually[1] greater than that of the original cells. In large macros, the size of the power-supply wiring is an important factor in this area increase; in smaller macros the area of the macro is approximately linearly proportional to the number of columns. In laying out such an array, there are several issues that must be addressed. Each may be amenable to more than one kind of solution; the following discussion addresses some of the relevant issues in general terms.

[1]In the particular case of a single column the area of the macro can be made equal to the area of the cells.

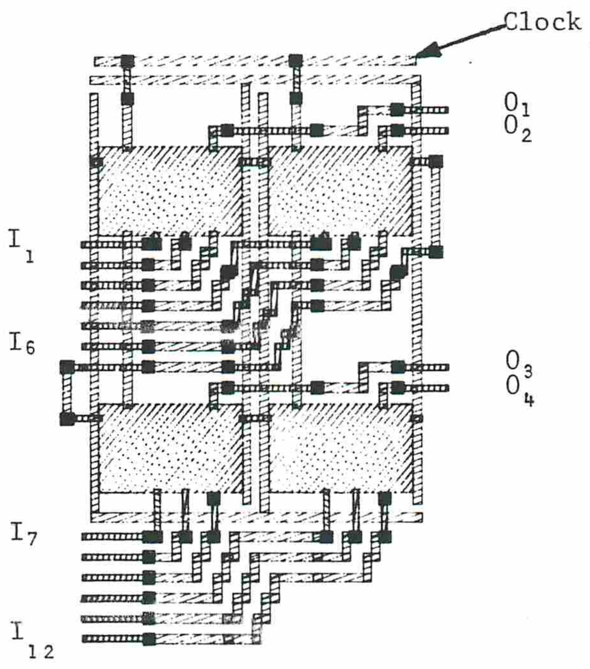


Figure 3: Example of an RC Layout

3 BASIC TOPOLOGY

3.1 Connection Points

A fundamental issue in the layout of any macro is the positioning of its connection points. In general, the connections should be at those points that will lead to the best placement and routing solutions for the entire chip; however, computational difficulties make it difficult to take the entire chip into account except in highly structured methodologies [2]. A more or less acceptable compromise for the layout of arbitrary macros is to offer the designer some choice in the connection points of the macro under consideration; i.e. some number of options or connection styles from which the design best suited to the problem at hand can be chosen. These options can be divided into those that apply to the I/O wiring, the power supply connections, and the clock and control connections.

Three basic styles of I/O wiring are shown in Figure 4. These styles are called Straight, Corner and Folded.

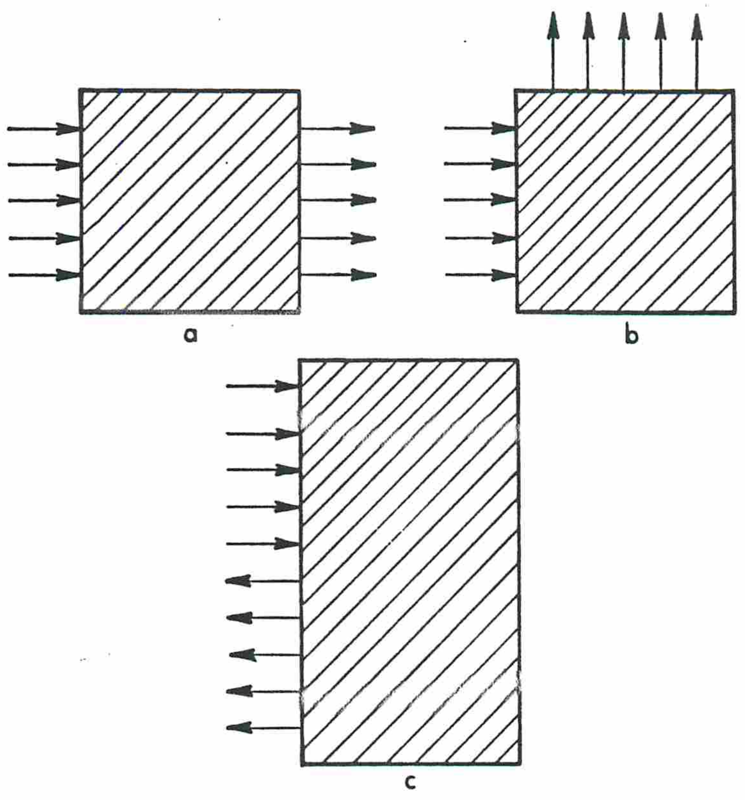


Figure 4: Three Connection Styles
(a) Straight Layout
(b) Corner Layout
(c) Folded Layout

An example of a straight layout, with all of its internal wiring shown, is shown in Figure 3. The detailed structure of the cells themselves has been suppressed for clarity, and the size and number of the cells is unusually small. The shape of the I/O structures results from the use of a river routing algorithm [1].

3.2 I/O wiring

The river routing algorithm produces an overall structure that is trapezoidal in its gross form, and of nearly minimal area. An alternative would avoid the staircase shape characteristic of river routing by using inclined wires. Because the minor base of the trapezoid is made shorter by the use of inclined wires, and because the minor base length is of great importance in determining the amount of track sharing that can be done in between two rows, (see Figure 5) such a strategy would lead to a smaller overall layout.

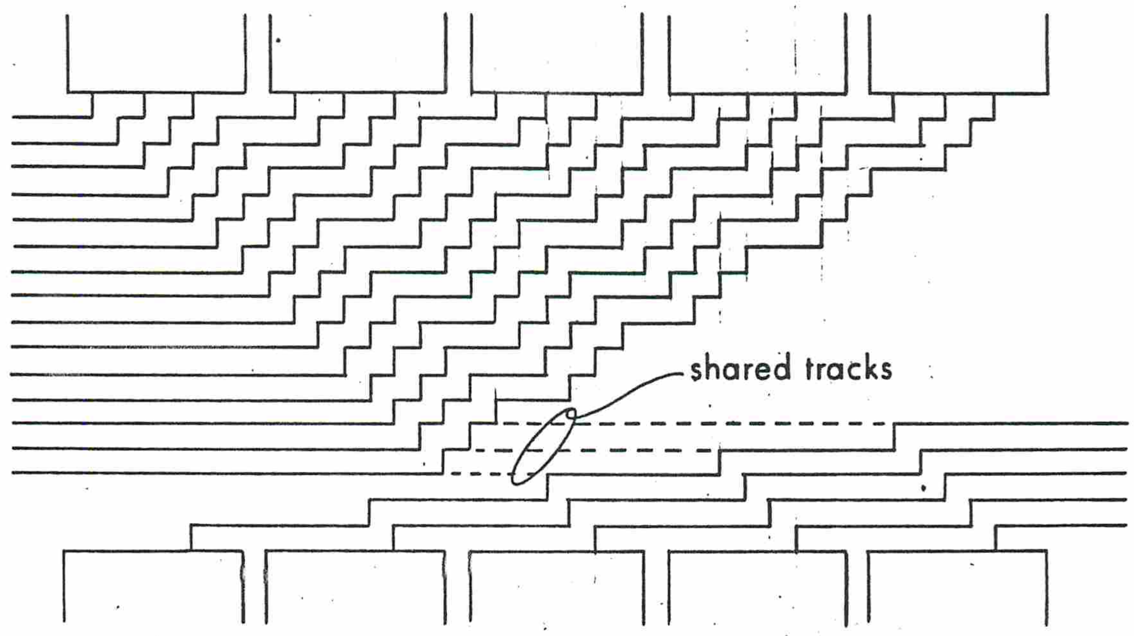


Figure 5: Track Sharing Between Input and Output Structures

3.3 I/O Layering Strategy

Multiple routing layers are of use in a number of basic ways. First, in some layout scenarios the power-supply and clock/control wires must have the signal I/O wiring routed under them; this jumpering must be done on a secondary layer. Second, if enough routing layers are available, the I/O structures can be constructed with overlapping wires. This strategy reduces macro area but does not change the characteristics of the area/aspect ratio tradeoff. If the designer would rather have a small macro than a fast one, for example, then the polysilicon (and in extreme cases the diffusion) layers could be used with some area saving. Furthermore, the designer of the kernel itself may decide to provide connection points on any one of the available layers; for these reasons a good RC program should support many layering options. Furthermore, the RC program should be able to take both the I/O wiring layers and the power supply configuration into account; in cases where the I/O wiring is on the metal layer, special treatment may be necessary to get the I/O connections past the power supply connections. This problem is most severe in the case of horizontal power-supply wires, for in these cases the power-supply wires are either adjacent to or quite close to the edge of the cell, and it is necessary to jumper into another layer right at the edge of the cell; the power-supply wire must therefore be offset to allow room for the necessary contact (Figure 6).

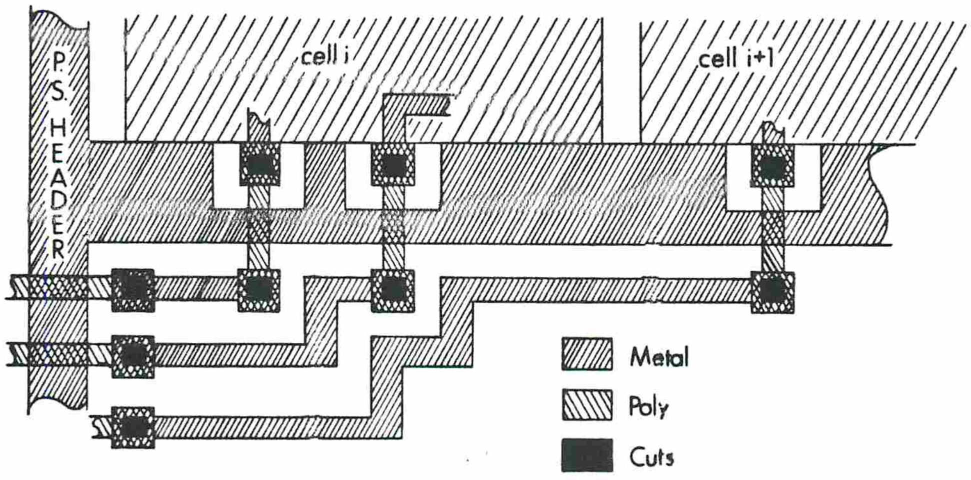


Figure 6: Horizontal Power and Metal I/O

3.4 Neighbor Connections

The connections that go from a cell to its neighbors need special treatment. These signals, which are not the same as the clocks and controls in that they are not distributed throughout the macro, must tie the cells together in a (logically) linear array; a good example of a neighbor connection is a ripple carry line in an adder. Because the neighbor connections create a linear array, notice must be taken of the ordering of the cells along each row and column, so that the I/O wires are properly ordered.

There are two decisions that must be made in implementing these connections. First, the direction of the connections must be specified. This direction may be either horizontal or vertical, i.e. a cell may be connected to either its neighbors to left and right, or to its neighbors above and below. An important consequence of the horizontal scheme is that the I/O connections can be ordered in a natural way;

the vertical neighbor connection does not result in a "nice" ordering of the macro's I/O connections (Figure 7). The vertical configuration will usually result in a smaller macro than the horizontal configuration.

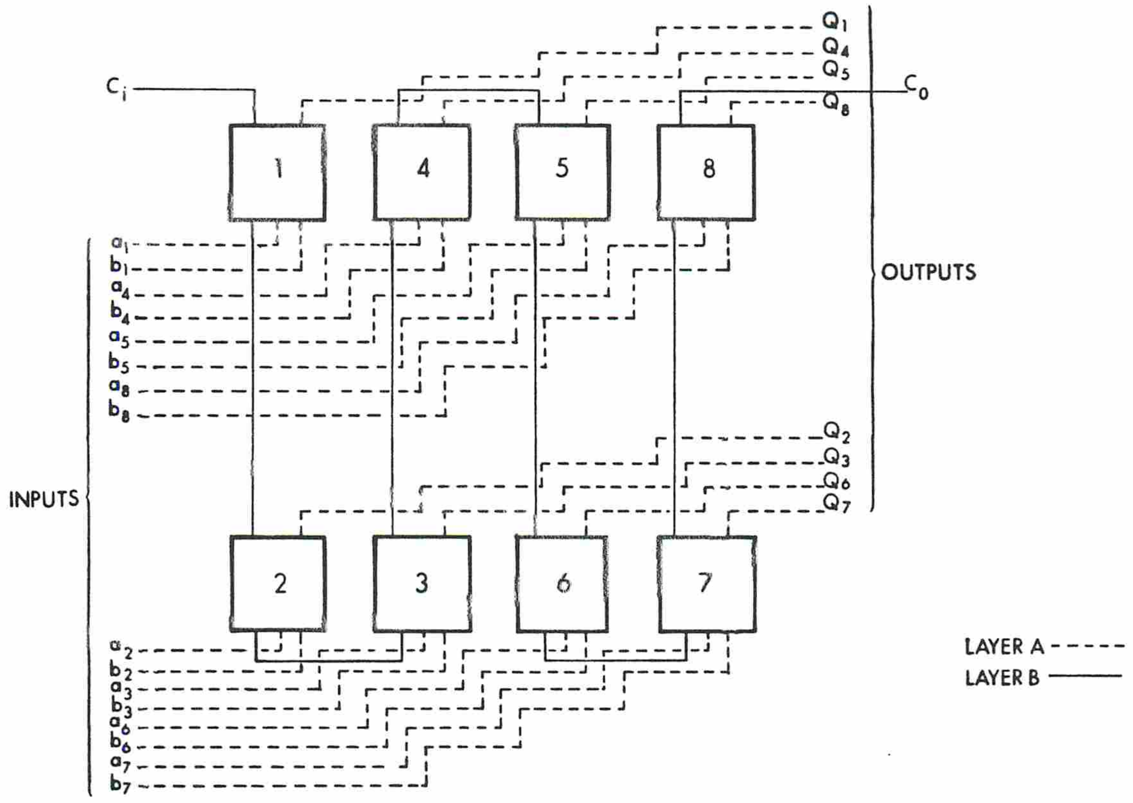


Figure 7: Permuted I/O Busses Caused By Vertical Neighbor Connections

The second decision that must be made that concerns the neighbor connections concerns the treatment of the ends of rows (or columns). There are two basic ways the neighbor connection can be carried from one row (column) to the next (Figure 8). First, the connections can always go from right to left (or vice versa) with an endaround from the end of each row (column) to the beginning of the next; or second, the connections can go in opposite directions in alternate rows (columns). It is important to note that the second connection style results in a permuted ordering of the I/O wires, and that it also assumes that either (a) all neighbor connections are bidirectional, or (b) the cells of every other row are mirrored about the y-axis; otherwise outputs will be tied to outputs and inputs will be tied to inputs[2]. The

[2]The problem of permutation comes up here as well: if the cells are mirrored then the inputs to a cell will be permuted. Permutation (as well as interleaving) also occurs when the Folded I/O connection style is used.

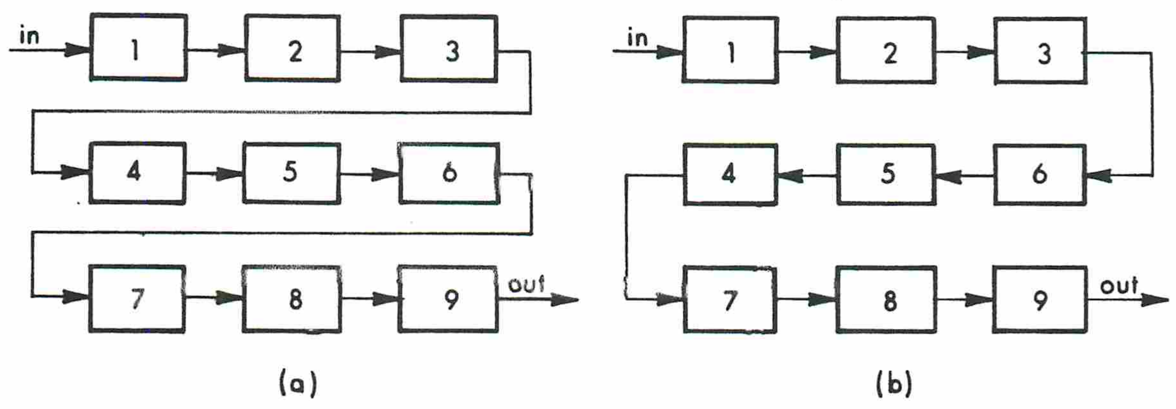


Figure 8: Endaround and Endover Neighbor Connections

endaround connection is thus expected to be preferred most of the time. In many cases, the length of the interrow connection will warrant the use of a metal layer wherever there are no obstructions; this may have a critical effect on the speed of the macro, as such neighbor connections can (as in the case of an adder) determine the macro's overall speed.

The current RC generator program, called "rowcol", supports horizontal neighbor connections in either endaround or endover configurations; it supports any originating layer, but does not permit combinations of power, clock, control and neighbor connections where the combination in question would lead to a layering conflict in the single-metal-layer technology it assumes.

3.5 Clocks and Controls

Clock and control wires are distinguished from neighbor connections in that they are presumed to represent single signal nets. As in the case of neighbor connections, there are two basic distribution strategies, horizontal and vertical; unlike the neighbor connections, common headers can be used to distribute the signal between columns (in the vertical case) or rows (in the horizontal case). Layering is an important issue in the layout of clocks and controls, because of the relatively low driving impedance that is required; therefore, clock and control lines should be routed on a metal layer. Clock and control wires can be routed through the body of a cell, but this should only be done when the cell is specifically designed with such use in mind. The program "rowcol", in fact, permits only this kind of clock and control connection (Figure 9).

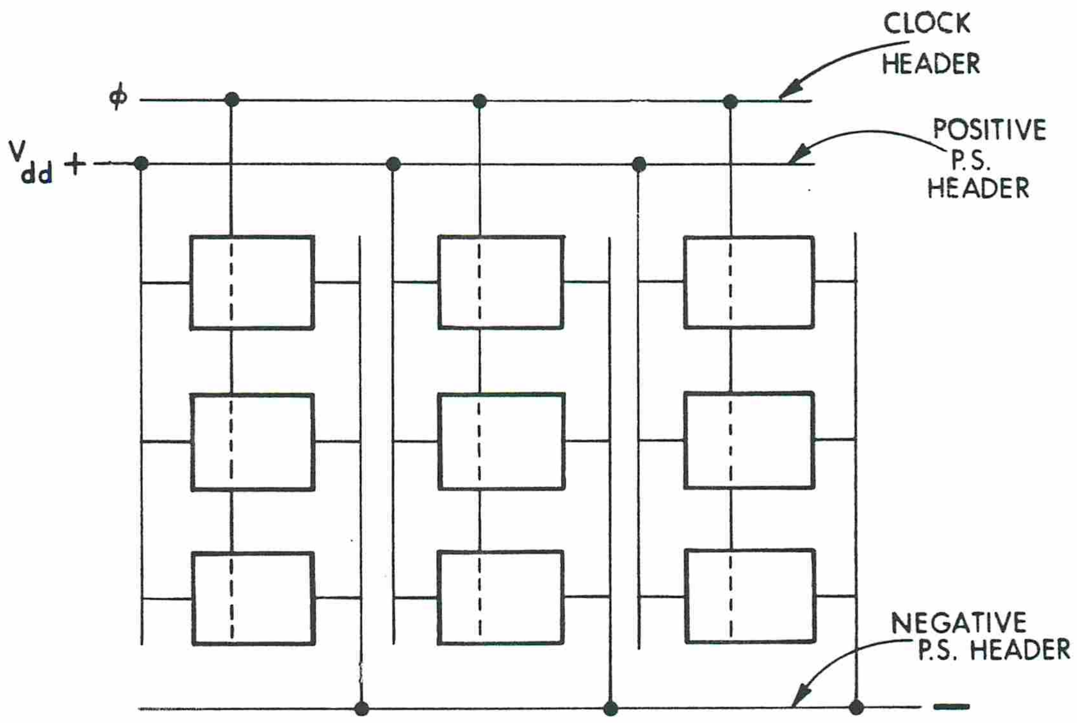


Figure 9: Clock and Control Connections

3.6 Power Supply Routing

The power supply for an RC macro can be laid out in the form of a pair of interdigitated combs. The headers and fingers of the combs may be tapered; this will result in some saving, at the cost of increased program and layout complexity. For large arrays, the size of the power-supply wires becomes a significant part of the overall area of the array, because their area grows as the square of the linear dimensions of the macro. The combs may have either horizontal or vertical fingers, but some combinations of power supply, neighbor, and clock/control wiring can cause difficulties.

Sometimes it is possible to save some power-supply routing area by mirroring alternate rows (columns) about the x (y) axis; this applies to situations where the resulting perturbations of (a) neighbor ordering, (b) clock and control connections, and (c) power-supply comb direction are tolerable. In small arrays, this saving is governed by the design rules for width and spacing of metal wires, and will be roughly two-thirds of the power-supply wiring area; as the array gets larger this savings will decrease to a very small percentage, because resistance and current-density constraints will become more important than design rule constraints. The program "rowcol" uses untapered wires sized to meet the overall power requirements of the macro, and mirrors columns whenever mirroring is both possible and desired by the user.

3.7 Cell Constraints

There are very few intrinsic constraints to the shape of the cells that can be used in an RC array; but there are so many possibilities that a program capable of handling all of them would be quite large and therefore prone to errors. The current RC generator, "rowcol", accepts cells of the following description:

- the cell must be rectangular,
- inputs and outputs must be on opposite sides of the cell,
- power and ground must be on opposite sides of the cell,

- neighbor inputs and outputs must be on opposite sides of the cell and those sides may not be the same sides as the inputs and outputs are on,

- clock and control connections must be on the same sides of the cell that the inputs and outputs use, and furthermore they must exit the cell at the same x (or y) coordinate that they enter on,

- all connections are flush to the edges of the cell, i.e. it is not necessary to run wires into the body of the cell in order to make the connections,

- power and clock wires must be on the metal layer,

- if neighbor connections are on the metal layer then they cannot be on the same sides of the cell as the power supply connections,

- if there is more than one power supply connection for either Vdd or ground along a side, then a metal wire can be laid down adjacent to the boundary of the cell all along its length, unless there is a metal I/O connection on that edge, in which case the power connections must respect the design rules with respect to that connection.

These constraints are by no means universal; there are ways to get around all of the difficulties that they obviate. However, a program that would deal with a truly general cell (or family of cells) would be a difficult and expensive program to write, and its relative value would be questionable.

4 ESTIMATION OF AREA

The area of an RC array can be expressed as a sum of two areas: the area of its constituent cells and the area taken by the aggregated extracell wiring. The area of the cells is, for a given function size, a constant; the area taken up by the wiring can be further decomposed into two components. These two wiring components are: first, the area of the power supply wiring; and second, the area of those parts of the I/O wiring that are not underneath the power supply wiring. The area of the power supply wiring is roughly proportional to the linear dimensions of the array in cases where design rule constraints determine the power supply wire sizes; where sheet resistance (or current density) is the governing factor, the power supply wiring area is quadratic in the linear dimensions of the array. That part of the I/O wiring that does not lie under the power supply wiring has an area that is proportional to the number of columns in the array.

In this context, let the area of a cell be its height multiplied by the period of horizontal replication. The area of the macro is then given by the expression

$$A=RCXy + CX(H+H'+(R-1)(Y-y)) \qquad 1$$

where the terms R and C represent the number of rows and columns in the array respectively, and all other dimensions are as shown in Figure 10.

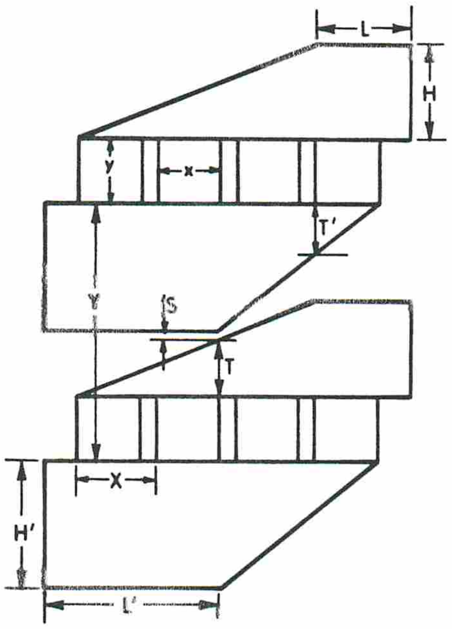


Figure 10: Definition of Dimensions

The first term in Eq. 1 represents the area taken by the cells; X is the horizontal period, y is the cell height, and RC is the total number of cells. X, which is the horizontal period, is in the horizontal power-supply case simply the width of a cell added to the minimum distance that must be allowed between two cells in order to assure conformity to design rules. In the case where a vertical power supply comb is used, X is given by the expression

$$X=x+2\max(W,GIR((R-1)(Y-y)+H'+H+y)/V+S \qquad (2)$$

where x is the width of the kernel, I is its power-supply current, G is the sheet resistance of the metal layer, and V is the permissible voltage drop along the length of a power-supply finger. The terms W and S are the minimum width and minimum separation of metal features respectively. Eq. 2 applies only to the case where both positive and negative power supply conductors must be routed along each intercolumn space; if only one or the other need be routed along the intercolumn space, the S term does not appear.

The second term of Eq. 1 represents aggregate wiring and "wasted" area; its size is therefore a measure of what the designer will pay (in area) for changing the shape of the array. Because (Y-y), H', and H are proportional to C, and RC is a constant, the second term of Eq. 1 is roughly proportional to C.

H' and H can be calculated as the sum of the widths and the spacings of all the input and output wires respectively, as given by

$$H'=(Wj+Sj) \qquad (3a)$$

$$H=(Wj+Sj) \qquad (3b)$$

where Wj is the width of the jth wire, as determined by the relevant design rule, and Sj is the spacing required between the jth wire and its predecessor. When a horizontal power supply comb is used, a further term h must be added to H' and/or H if any of the I/O wires come out of the cell on the metal layer (Figure 6); this term represents the added width of the power supply fingers due to the necessity of jumpering the I/O wires under the fingers, and is given by

$$h=3Wrb/2 + Wps + 2S \qquad (4)$$

where Wrb is the width of a poly-to-metal contact cell, and Wps is the minimum width of the power-supply finger.

In the endaround case any neighbor connections are counted as being the last wires of the input structure; they are treated exactly as if they were input wires, except for the purpose of correcting for the presence of metal I/O lines in the horizontal-grid case.

The total interrow spacing is given by

$$(Y-y)=\max(H+T,\ H'+T') + S \qquad (5)$$

Eq. 5 expresses the interrow spacing (Y-y) as the greater of two terms, plus a spacing that represents the minimum distance between the last wire of the input structure and the first of the output structure. Note that Eq. 5 is only valid in the case where the outputs of the array are on the opposite side from the inputs, i.e. the "Straight" I/O topology; in the "Folded" topology, a simpler expression holds.

$$(Y-y)=H'+H+S \tag{6}$$

The reason for the comparative complexity of Eq. 5 is that some space can be saved by track sharing in the I/O structures. An estimate of this track sharing can be made by means of the approximations

$$H' = L'+D' \tag{7a}$$

$$H = L+D \tag{7b}$$

where D and D' are the distances from the leftmost output to the right edge of the kernel, and from the rightmost input to the left edge of the kernel respectively. When these approximations are made the similarity of triangles can be applied to the problem of determining T and T':

$$T=H'(CX-H)/(CX-H') \tag{8a}$$

$$T'=H(CX-H')/(CX-H \tag{8b}$$

These values can now be substituted into Eq. 5 and the total area calculated by applying Eq. 1.

For a macro consisting of a given number of cells, solutions tend to take the form shown in Figure 11, which shows the area plotted against the aspect ratio (defined as height divided by width) for a family of 36-cell macros. Only 'good' solutions, i.e. those where RC=36, are shown. All cells had the same area, but their aspect ratios varied from 4.4 to 0.1. It is interesting to note that the area of a macro of given aspect ratio is almost independent of the aspect ratio of the kernel cell whenever there are more than a few cells in the macro. Note that in Figure 11 the results of using kernels of the same area but widely differing aspect

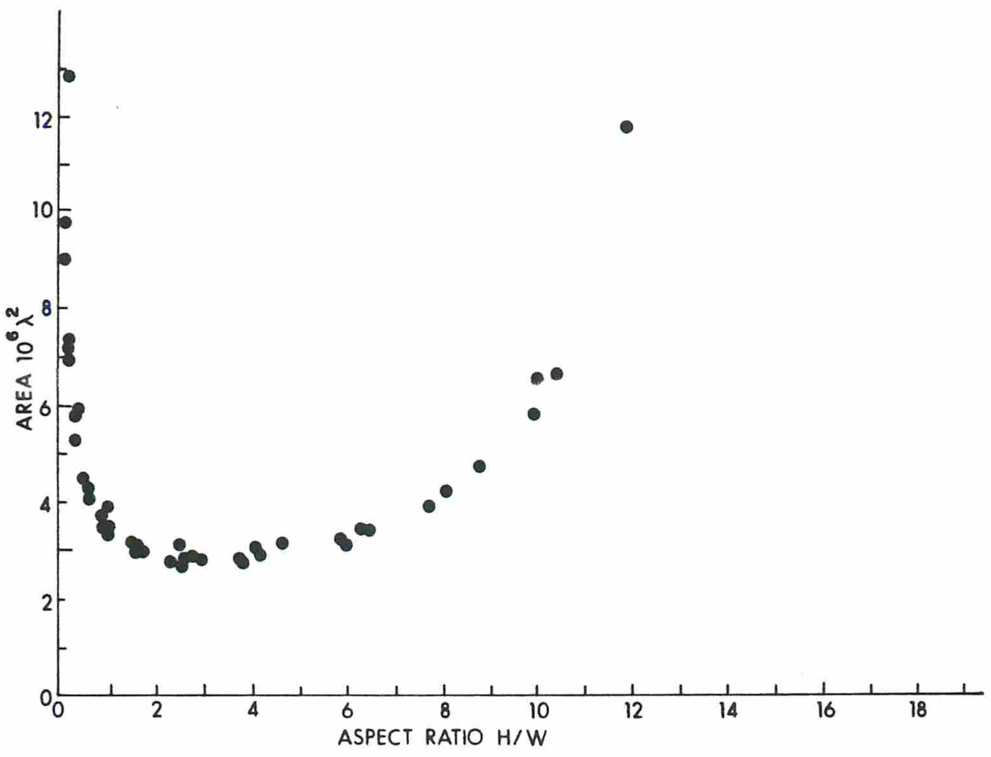


Figure 11: Area and Aspect Ratio of a Family of 36-cell Macros

ratio lie close to the same curve. The explanation for this effect is most clearly seen when the area of the macro is expressed as the sum of its signal I/O wiring, its power supply wiring, and the area of its constituent cells.

The area of the I/O wiring is affected by a fundamental property of functions such as are suitable for RC arrangement. These functions have fixed numbers of inputs and outputs to each cell in the array; therefore, for a given set of layers the height of the aggregated I/O wiring (i.e. the input and output buses) of a macro is fixed, and almost independent of the arrangement of its cells. Hence the area of the I/O wiring is roughly proportional to the width of the macro, except in the special case of a single column, where rotation of the kernel can result in substantial savings. This strategy is an increasingly common one in cases where entire data paths are constructed as a single macro, as in the OM2 chip [2]. Even then, it can be argued that the I/O wiring area has merely been disguised as part of the active cell area; the wires must pass through the cells. It can also be said that the amount of track sharing that can be accomplished with a given number of inputs and outputs is nearly independent of the kernel's aspect ratio and the numbers of rows and columns in the macro. This comes about because the trapezoids used to route the I/O wiring are similar trapeziods no matter how many columns the trapezoids must serve: if one adds a column, then one must also add i/o wires for that column, thereby multiplying the minor base, the major base, and the height all by the same constant factor of 1 + 1/C. Because the I/O trapezoids are similar in all the layouts for a given kernel, for all the layouts the amount of track sharing is roughly constant. Deviations from a constant amount of track sharing come from the detailed structure of the river routed structures, which have "steps"; the way these steps fit into one another will determine the exact number of tracks that can be shared.

It can further be said that the amount of track sharing is a function of the relative numbers of input and output wires. Clearly, the most "economical" use of the interrow area is that which has the most track sharing; this is the case when the kernel has equal numbers of inputs and outputs, because the interrow spacing is determined by the maximum of the heights of the two trapeziods. Figure 12 shows this effect: it represents a family of sets of macros, the families being constructed from kernels having different numbers of inputs and outputs. The salient point of Figure 12 is that it costs nearly the same amount of area to implement a macro having four inputs and one output per cell as it does to implement a macro having four inputs and four outputs; and of course the same kind of argument applies to cells having two and three inputs as well.

The power-supply wiring area has a tendency to make macros of extreme shape uneconomical, but has little effect over a middle range of solutions. The explanation for this lies in the assumption that power connections may not be interleaved [3].This is equivalent to assuming that power is connected at two points on the periphery of each macro: one for power and one for ground. Internal wires must meet some constraint on aspect ratio; therefore long wires are penalized as the square of their length. This in turn means that macros of extreme aspect ratio will

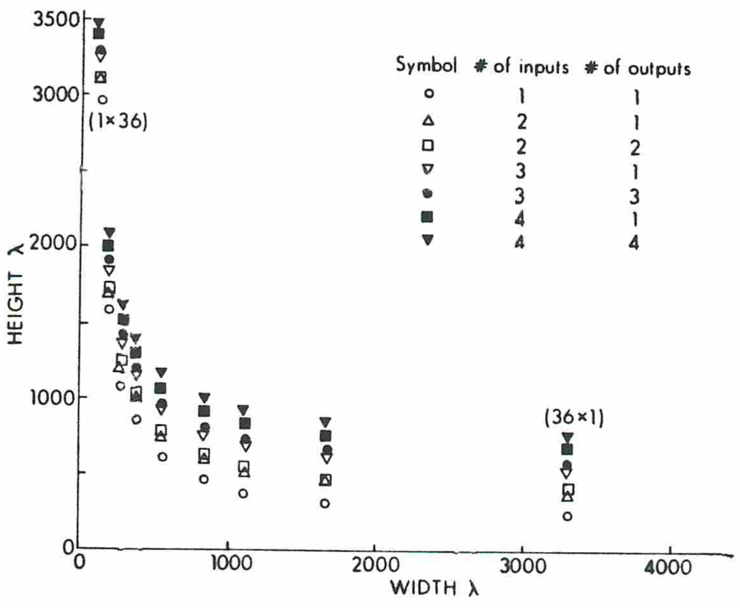


Figure 12: Height and Width of Macros with Differing Numbers of Inputs and Outputs

incur a large power-supply wiring penalty regardless of the actual positions of the power connections. The properties of this penalty are shown in Figure 13; in it, the power supply current of a kernel was varied over a wide range. The edge of the shaded area that lies closest to the axes represents macros that have only minimal power supply current, and hence minimal power supply routing area; as the power supply current increases, the area increases. In the particular case graphed in Figure 13, the combs were vertical; hence the layouts that are taller and narrower tend to pay the greatest penalty, which is largely a width penalty. Very wide layouts also pay a penalty, albeit a lesser one, because while the vertical fingers are many, they are also short; but in that case the horizontal headers are long and they contribute substantially to the overall macro height. In the middle of the graph, where the macro's height is close to its width, we find those layouts that pay the least penalty for high current consumption: these are skewed slightly toward the wider layouts because of the greater numbers of vertical wires. Naturally, this skewing would be reversed in the case of a macro with horizontal power-supply combs.

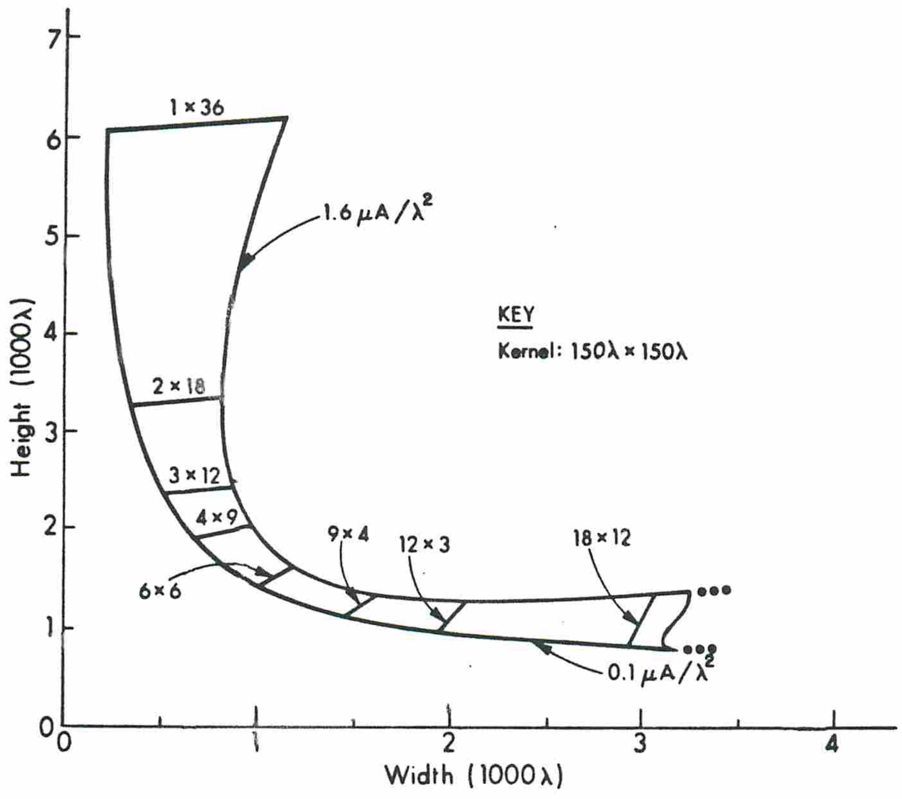


Figure 13: Height and Width of Macros with Varying Current Requirements

If it is assumed that the area of the cells is constant regardless of their aspect ratio, then the area of the macro can be expressed as the sum of the constant cell area, an I/O wiring area proportional to the macro's width, and a quadratic penalty for deviating from a square shape. This is the explanation for the shape of the curve of Figure 11. The relative importances of the three components is determined by the particular case in question; in the case of Figure 11, none of them is negligible.

5 CONCLUSIONS

A technique for generating rectangular macros of variable aspect ratio and general expressions for the dimensions of such macros have been presented. Macros generated by this technique tend to take up area as a function of their aspect ratio, nearly independently of the aspect ratio of their component cells. A particular example of a set of macros was presented in which all of the macros fell at different points close to the same curve.

REFERENCES

1. C. E. Leiserson and R. Y. Pinter. Optimal Placement for River Routing. Conference on VLSI Computations and Systems, CMU, 1981.

2. Carver Mead and Lynn Conway. Introduction to VLSI Design. Addison-Wesley, 1980.

3. Zahir A. Syed. On Routing for Custom Integrated Circuits. Ph.D. Th., Dept. of Electrical Engineering, University of Southern California, july 1981.

APPENDIX

I. Example RC Layouts

In this appendix a few example RC layouts are given. All were generated by the program "rowcol". The details of the kernels themselves are suppressed for clarity, and the numbers of rows and columns have been reduced; the size of the kernels have also been set rather small in an effort to improve the clarity of the figures.

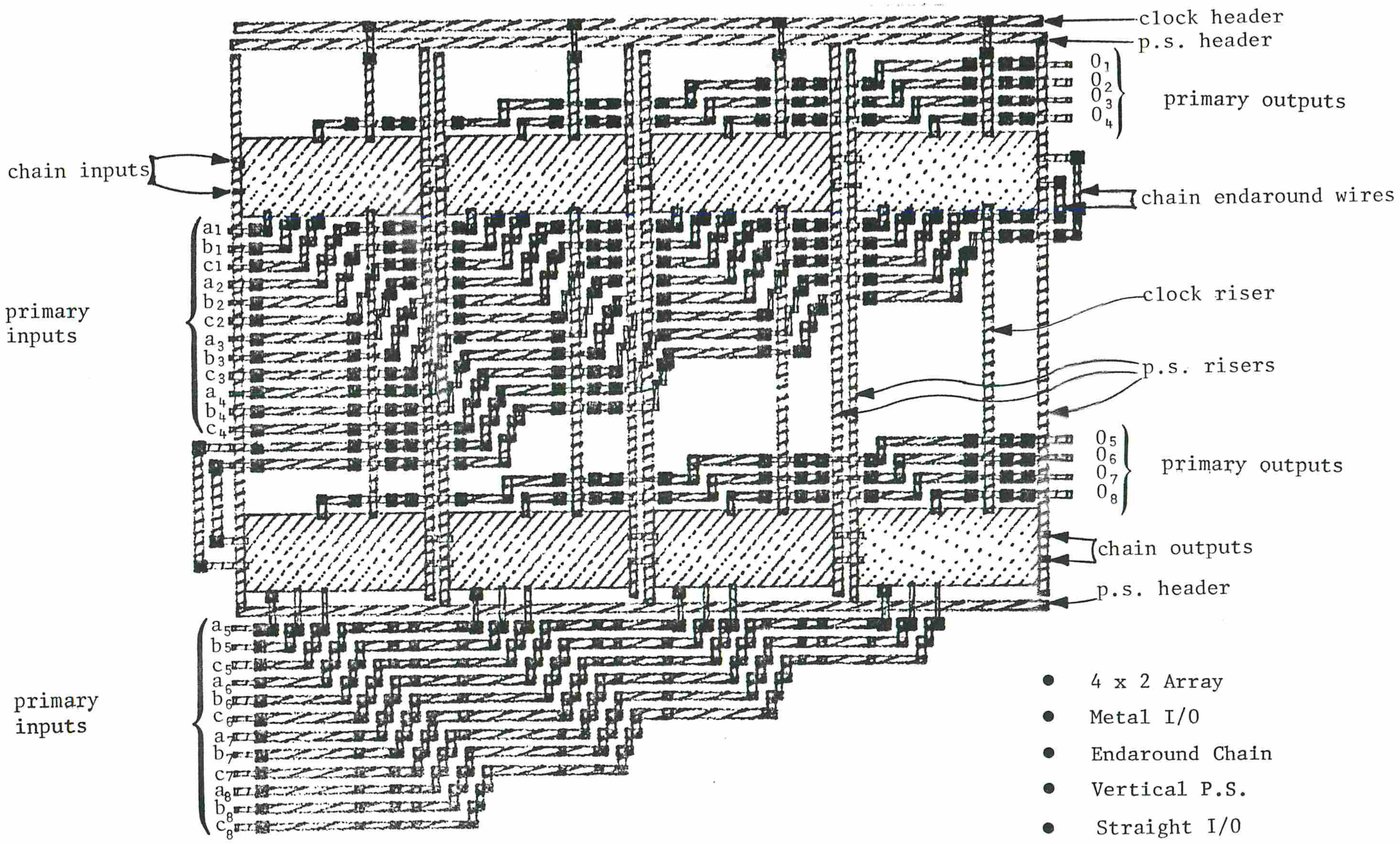
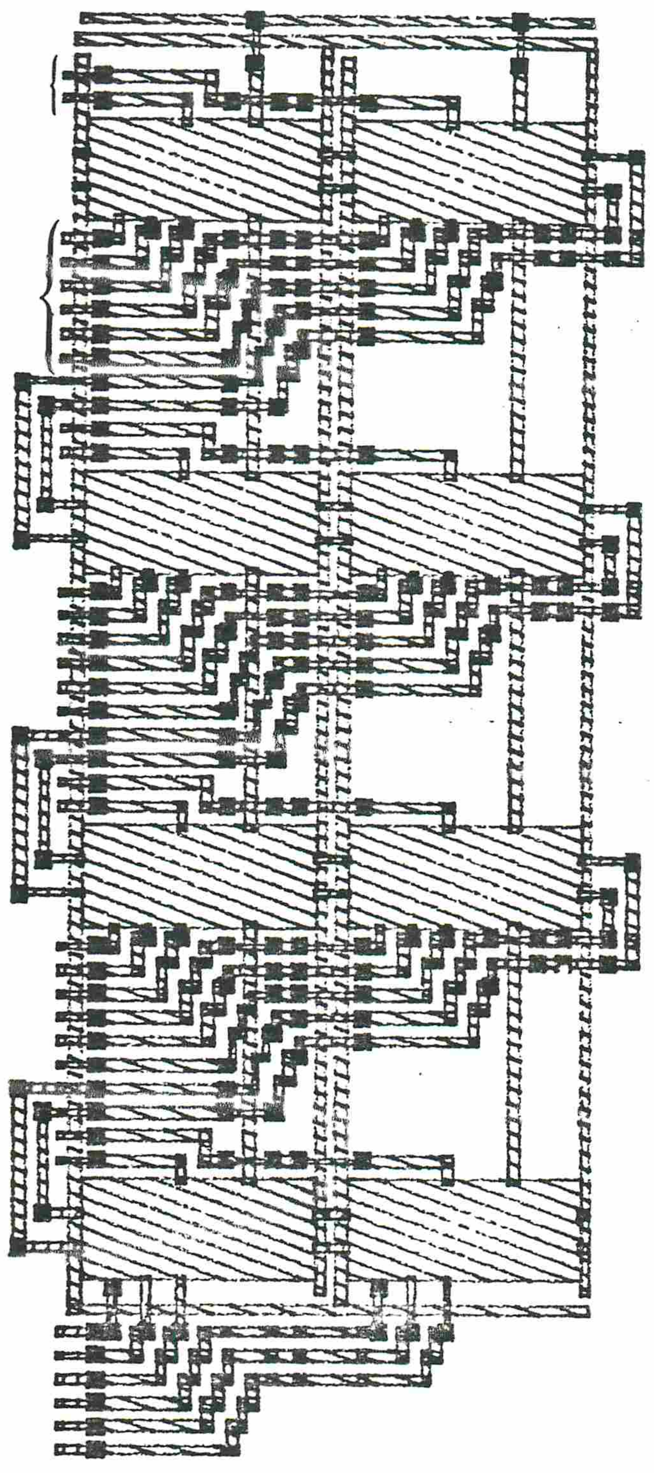
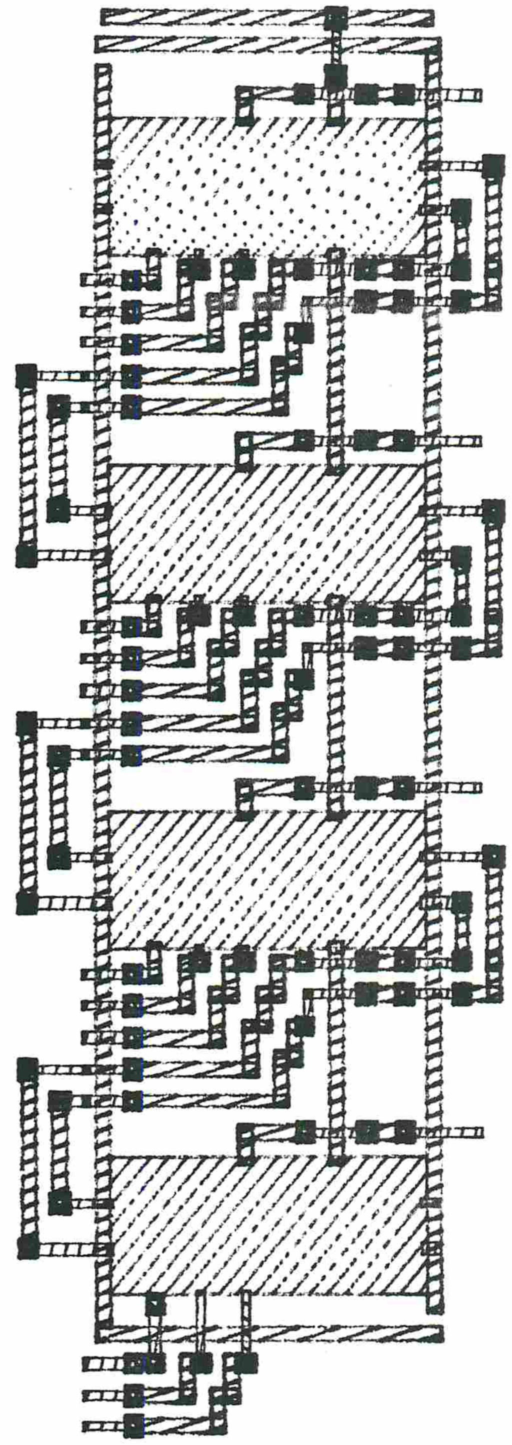

clock header
p.s. header
O_1
O_2
O_3
O_4
primary outputs
chain inputs
chain endaround wires
a_1
b_1
c_1
a_2
b_2
c_2
a_3
b_3
c_3
a_4
b_4
c_4
primary inputs
clock riser
p.s. risers
O_5
O_6
O_7
O_8
primary outputs
chain outputs
p.s. header
a_5
b_5
c_5
a_6
b_6
c_6
a_7
b_7
c_7
a_8
b_8
c_8
primary inputs
● 4 x 2 Array
● Metal I/O
● Endaround Chain
● Vertical P.S.
● Straight I/O

Figure A.1



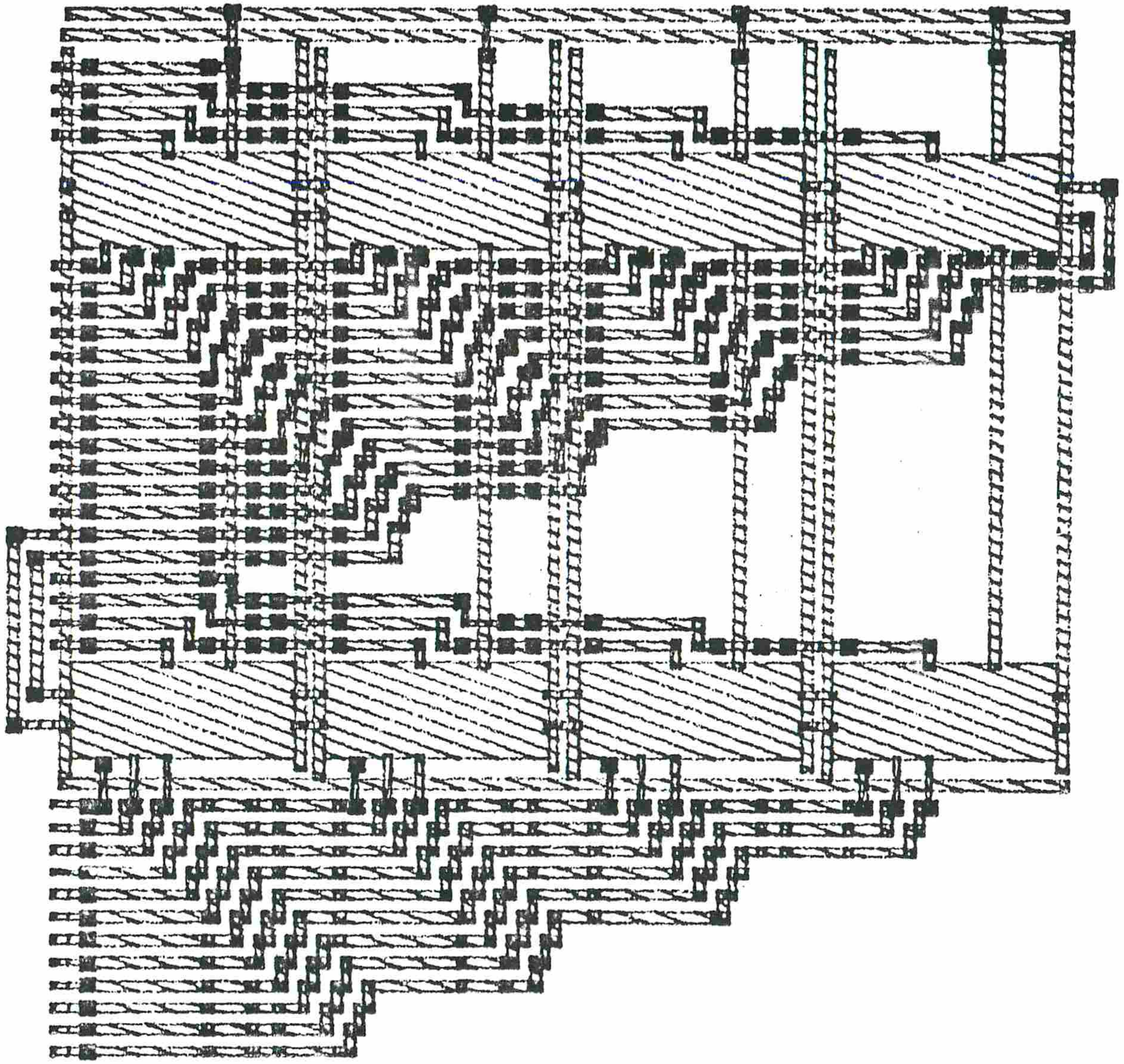
- 2 x 4 Array
- Metal I/O
- Folded I/O
- Endaround Chain

Figure A.2



- ● 1 x 4 Array
- ● Metal I/O
- ● Endaround Chain
- ● Straight I/O

Figure A.3



- 4 x 2 Array
- Folded I/O
- Metal I/O
- Endaround Chain

Figure A.4

- 4 x 1 Array
- Straight I/O
- Metal I/O

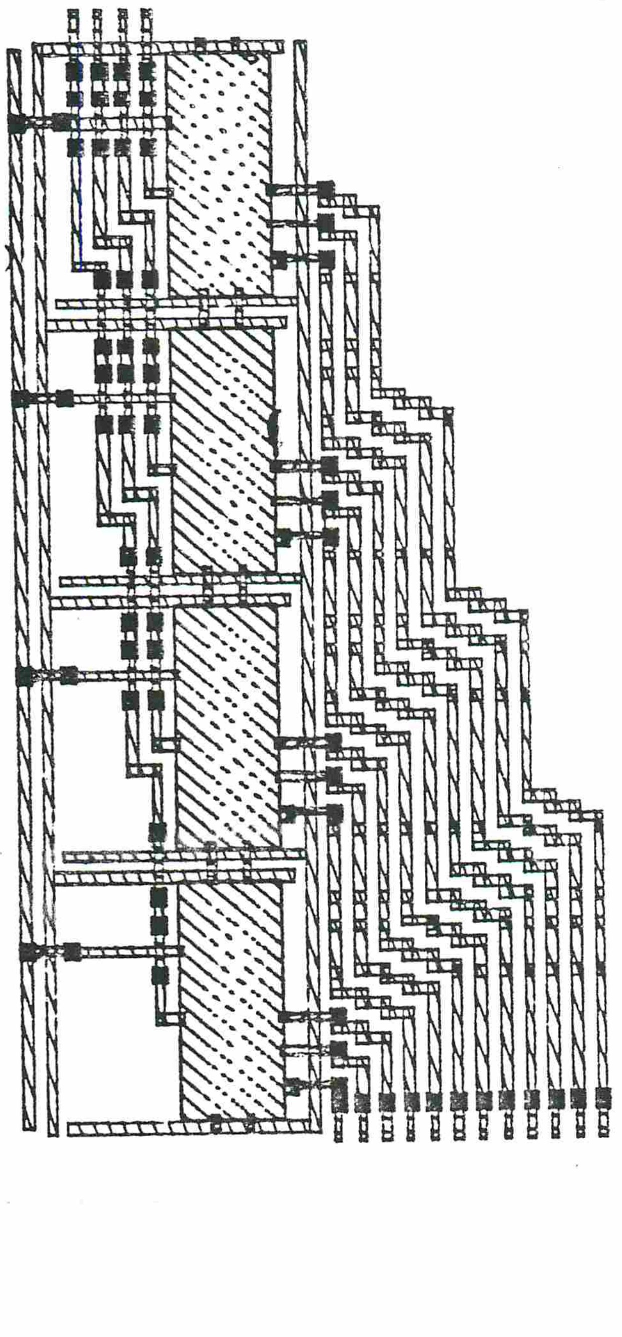
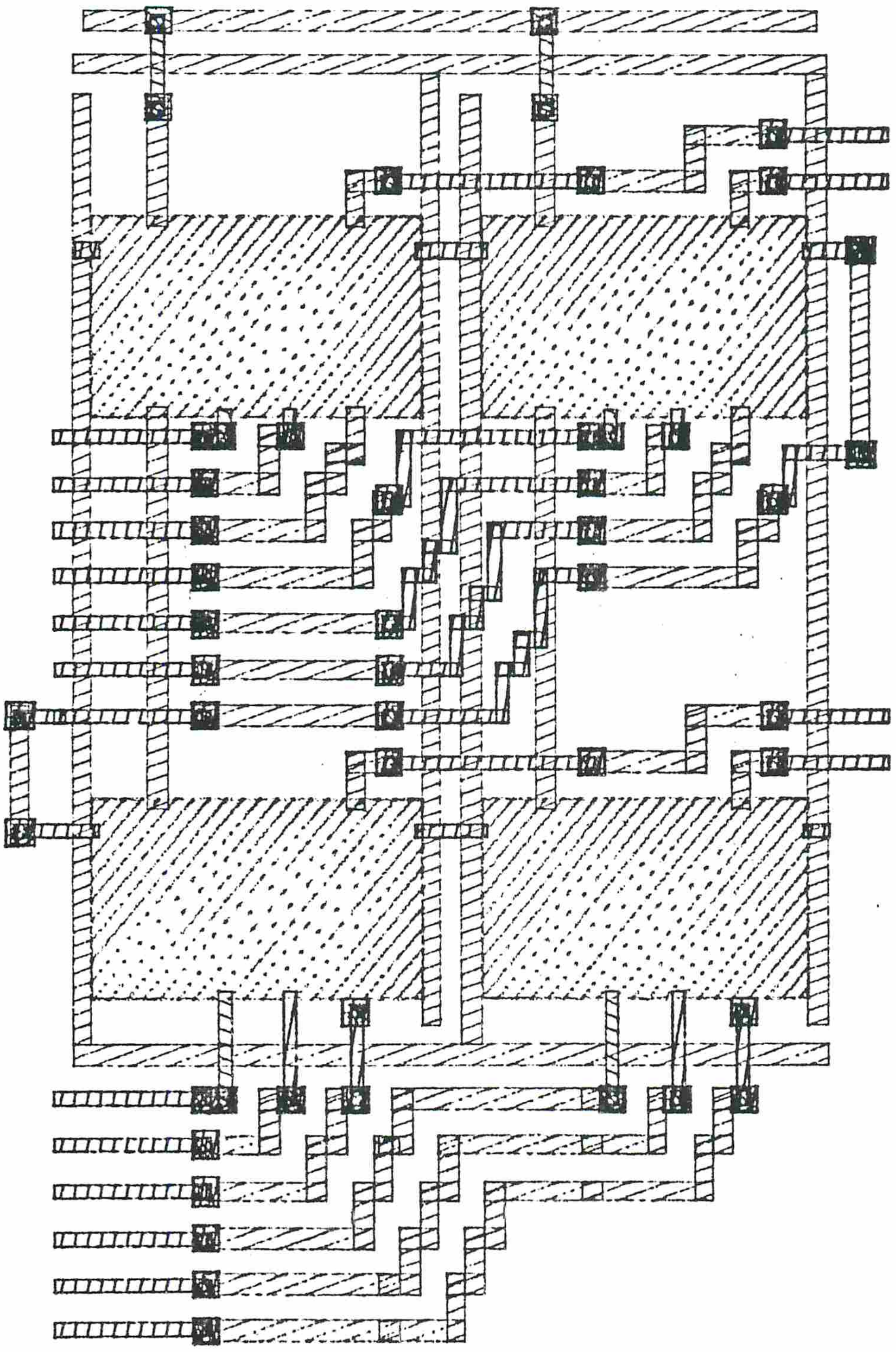


Figure A.5



- 2 x 2 Array
- Metal I/O
- Straight I/O
- Endaround Chain

Figure A.6