Area Estimation of VLSI Integrated Circuits

Technical Report CRI-85-05

Fadi J. Kurdahi[1] and Alice C. Parker

24 July 1985

U. S. ARMY RESEARCH OFFICE

CONTRACT DAAG29-83-K-0147

Department of Electrical Engineering-Systems
University of Southern California
Los Angeles, CA 90089-0781

---

[1]Also supported by the National Council for Scientific Research, Beirut, Lebanon.

Abstract

In this report, we present the problem of estimating the area of digital integrated circuits. This problem is important for reasons of chip yield, floor planning and design turnaround time. Area estimation is to be done at different levels of hierarchy. A model is presented for estimating the dimensions of the random logic blocks of a chip, given the description of these blocks as their constituent cells and their interconnections. We will pursue this research towards building a complete area estimation system which can handle different layout methodologies, can perform estimation at higher levels of design description (namely the Register-Transfer level), and be a useful aid to floorplanning and total layout as part of the ADAM system.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 The long term goal of this research

In this report we examine the problem of estimating the area of digital integrated circuits layouts. The problem we are addressing here can be stated as follows : given a circuit design description, how can we estimate the dimensions of the physical layout of that circuit on a chip, without actually performing the layout tasks (such as placement, routing or compaction).

Stated as such, the problem seems to be too general, and two questions arise:

- What type of layout methodology is used?
- What level of circuit description is given?

The first question arises because there are different layout methodologies that are currently being used for chip layouts, and it is not known how estimation techniques might vary for different methodologies. This issue will be addressed later in the report.

The second question arises because of the tremendous increase in the design and circuit complexity. With the advent of VLSI, it has become almost impossible to manage the design information at one level of description, hence the need of representing the design in different *views*. For our purposes, the design can be described at three views.

- **The Behavioral Description** is the most abstract view of the design. It is usually the initial input specification to the design process. At this level of description, the behavior of the system is specified in terms of abstract entities such as operations and values, without really specifying the crude internal details and structure of the end product.

- **The Structural Description** also known as *The Register-Transfer Level* (RTL) description, is when the functional structures of the system are identified. These functional structures are sometimes called register-transfer elements. From a functional point of view. Structures fall into two broad categories, operators and storage elements. Even though these structures perform very well defined functions and operations, they are still abstractions of the actual silicon layout.

- **Physical (layout) Description** is the view at which the various structures in the design have actually been mapped onto silicon. So, we need to describe the topology of the circuit as it would look on the actual chip.

These three views contain different (although not totally disjoint) information. In a physical layout, for example, it is extremely hard to find any information regarding the behavior of the design. On the other hand, the information regarding the sizes and shapes of the various blocks of the design is readily available. The behavior of the design is clearly specified in the behavioral view, but unfortunately, there is little or no information there regarding the dimensions of the final circuit layout. Even though the structural view describes the functions of the various blocks in the system, it has little information on their dimensions.

The design process generally starts with a behavioral description of the circuit, referred to as the design *specification*. A structural design description is then *synthesized* from the specification. The final stage in the design process is the *layout* of the design on the chip.

1.2 The importance of area estimation

Predicting the area of a digital design layout in the early stages of the design process, prior to performing the low level (physical) layout, is a very important task to pursue. We can state the following reasons for this:

1. **Yield considerations:** Integrated circuit chips are built on *dies* of silicon. Dies are cut from *wafers*. Not all the dies on a wafer are guaranteed to be functional due to the presence of *defects* in the silicon at some regions of the wafer. The *yield* of the wafers, defined as the ratio of functional (defect free) dies to the total number of dies, has been found to be an exponentially decreasing function of the die size [Muroga 82]. The larger the die is, the higher the probability is of finding defects in it. Since cost is directly affected by yield, estimating the size of the chip before performing the time-consuming and costly physical design will enable the designer to assess the cost and feasibility of the design at hand in the early stages of the process.

2. **Floor planning considerations:** During the design process, and prior to starting the layout procedure, the designer usually tries to *floor plan* the design. Informally, floor planning means roughly allocating space on the chip for each of the major parts of the design (such as the data path, memory, microcode and control parts in the case of a microprocessor) such that the total chip area is as small as possible, the chip aspect ratio is within a certain range, and the global communication length between the major design parts is as small as possible. Area and shape estimation of the major design parts at

the floor planning stage is a useful tool because good estimates mean floor plans that do not have to be greatly modified during layout.

3. **Synthesis considerations:** Area estimation could also be a useful tool during logic synthesis. In logic synthesis, there are usually some constraints on the final design. In many cases, such constraints involve the area of the design. Having an area estimate of the design early in its design process is useful in predicting if the design would satisfy the area constraints and hence avoiding a lot of time that would be spent on further carrying out the synthesis of a design which would not satisfy such constraints.

1.3 The problem formulation

In this section, we define the research problem. Before doing so, we first define some terms and state some assumptions to be followed throughout this report, unless otherwise stated.

1.3.1 Chips, blocks and cells

Given the tremendous increase in the IC design complexity and the amount of information to be handled, it has become necessary to use hierarchy to describe digital designs. In Section 1, we described the different *views* of a design. Here we are mostly interested in the structural view of the design. In describing the structure of design and the correspondence between structure and layout (or physical view), we recognize three levels of hierarchy , depicted in Fig. 1. They are

1. The **Chip Level**: this is the highest level of the hierarchy at which the global functions to be performed by the design are defined, as well as the input/output interaction with the outside world.

2. The **Block Level**: A block is defined as *a structural-level construct performing a well-defined function or set of functions. The layout of a block has a rectangular shape and is done using one layout methodology.* A chip is composed of a set of blocks and their interconnections. A microprocessor chip, for example, may contain the following blocks : ALU, register file, control, I/O interface. The process of allocating spaces to blocks and topological routes to their interconnections is known as *floor planning.*

3. The **Cell Level**: A cell is defined as a structure performing a well-defined function of SSI/MSI complexity, whose layout (or information on its layout) is readily available (generally in a *cell library*), along with the definition and location of its I/O pins. A block is composed as a set of cells and their

interconnections, known as the *cell* and *net lists*. The layout of a block consists of the process of placement of the cells that compose it and routing of the interconnections. Depending on the placement and routing of cells, a block layout can have different configurations, areas and shapes (aspect ratios).

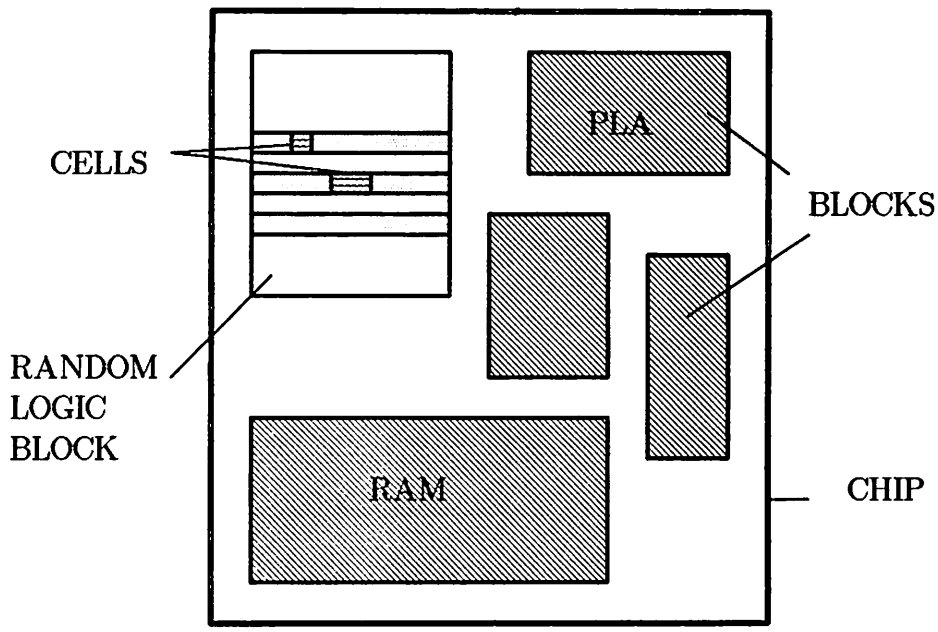


Figure 1: Hierarchy in structure

1.3.2 The problem statement

Ideally the designer would like to be able to estimate the dimensions of his chip, once the initial behavioral specifications are given. This, however, seems to be a hard task for the following reasons :

1. There is no formal model that describes the mapping between behavior and structure, least of all between behavior and layout[1]. In other words, the general design process of going from behavioral specification to structure to silicon offers little predictability for the outcome.

2. For a given behavior, the design space (set of possible design implementations) is so large that any simple area estimate (or even any other estimate of performance) does not convey any meaningful information.

On the other hand, once a designer has synthesized a structure from his input

[1]We do not consider silicon compilers in this context.

behavior, an area estimator becomes a very useful and plausible tool because of the following :

1. In order to floor plan the design, one needs good estimates of the sizes of the various blocks that constitute the chip.

2. The design space is now drastically reduced in size. The information now available on the design blocks is more concrete and more indicative of the dimensions of the blocks.

Based on these discussions, we can state the long term goal of this work as follows: given a structural register-transfer level description of a circuit design to be implemented on a chip:

1. for each block, estimate the areas and shapes of the possible configurations of its layout, and

2. estimate the area of the whole chip layout.

Estimating block area and shape

Estimating the areas and shapes of a block layout requires knowledge of the layout methodology (or design style) used to layout that block. There are two broad classes of methodologies:

- Random logic methodologies, including *Gate Array, Standard Cell*, and *Custom* design styles, and

- structured logic methodologies, including RAM/ROM, PLA and Array structure design styles.

In this section, we will not discuss structured logic methodologies, as they are well known, and will be overviewed in Section 5.

Gate arrays

Gate arrays consist of a fixed array of predesigned, preplaced, primitive cells with routing channels between them. The masks for a gate array are all prefabricated, except for the interconnection masks (hence the name *Master Slice* sometimes used for gate

arrays) which are customized for the circuit to be embedded on the array. The use of gate arrays reduces the layout problem to that of assigning gates in the circuit to cells on the array and routing them. These properties make gate arrays ideal for low volume (in house) applications requiring a short turnaround time. Gate arrays are, however, quite wasteful in chip area, which directly affects the chip yield and/or chip count.

Standard cells

With the standard cell layout methodology, the designer chooses his hardware components from a library of predesigned cells. These cells are usually of the same height but of different widths and can abut vertically. The designer arranges these cells in rows and routes wires between the rows. Standard cells are further described in Section 4.

Custom cells

In this methodology, cells are not constrained in height, width or location. The only constraint is that all cells must be rectangular in shape. Smaller cells are grouped together and laid out to generate larger cells (sometimes referred to as *macrocells*) thereby achieving levels of hierarchy in the layout. The exploitation of hierarchy in layouts reduces the problem complexity by constraining the designer to look at small subproblems, one at a time. Custom cell layouts are, however, much more complex than either gate array or standard cell layouts because the blocks are of arbitrary size and shape. It has the advantage of consuming less area than either one of previous methodologies.

We chose to restrict the layout methodology for random logic to standard cells only. Hence the layout methodologies that can be used to lay out blocks are

- standard cells,
- RAM/ROM,
- PLA,
- Array structures.

This choice was made for the following reasons:

- almost all types of RT-level structures can be efficiently implemented on silicon in one or more of these design styles,

- all these methodologies are automated and have been used in layout systems to build many "real life" chips,

- all these styles are compatible, in the sense that blocks laid out in different styles can be fabricated together on the same monolithic chip, and

- the proposed layout methodology for the ADAM silicon compiler is very similar to the standard cell design style and hence an estimation model for the latter can be easily modified to handle the silicon compiler area estimation.

Estimating chip area

The second part of the problem statement deals with estimating the total area of the chip, once the block area estimation has been done. This involves the following tasks :

- (a) decide on which of the possible configurations for each block is to be used, and hence on its area and shape,

- (b) perform floor planning on the chip and allocate space for the blocks such that the overall chip area and shape are satisfactory, and

- (c) given the allocation in (b), estimate the amount of space needed for wiring the connections between the blocks and from that deduce a better estimate of the total chip area.

Note that in an actual design process, the system may iterate through these steps, selecting a configuration for each of the blocks, and attempting to floor plan the chip. If the overall area and/or shape are not satisfactory, then the system may go back to (a) selecting a new configuration for one or more blocks.

Each one of these problems must be treated and solved before an accurate area estimate of the chip become plausible. Although it is an important problem in chip design, the floor planning problem will not be rigorously treated in this work; rather we will assume that it is done interactively and will attempt to develop various aids for the designer to successfully perform the floor planning task.

The research results of this report are to be applied and embedded into the ADAM system currently under development at USC [Granacki 84] [Knapp 83a] [Knapp 83b]. Next we present a brief description of the ADAM system.

1.4 The ADAM system

The ADAM (Advanced Design AutoMation) system, currently being built at USC, is an integrated computer aided design system whose purpose is to aid the designer throughout the digital design process. A view of the system is shown in Fig. 2. The main features of the systems are the design database, a knowledge based planner and an expert system which aids in the design of testable circuits [Abadir 84] [Breuer 84]. The system components can be divided into three classes. They are:

1. The actors, the procedures and subsystems that act on the database information. Some of these are
    - the planner, a knowledge based subsystem which acts as a "monitor" during the design process, calling the design tools in the proper order.
    - the synthesis tools, include the RT-level allocator, the logic synthesizer, the pipeline synthesizer [Park 85], the clocking scheme generator [Park 84], and the silicon compiler, and
    - the analysis tools, including the area estimator, the critical path finder and the collision detector [Knapp 84a].
2. The objects, including all the pieces of information related to the design process. The main objects are
    - the design data structure (DDS) [Knapp 83b], a multilevel, multiview design representation,
    - the knowledge base, which contains information about the design process, and
    - the design plans. These are "annotated tours through the knowledge base" [Granacki 84], i.e. a sequence of activities which would, hopefully, achieve the preset goal.
3. The system interfaces, including
    - A natural language interface, which provides an English-like interactive media for specification,

- the 3DIS interface, which is an efficient method for browsing through the database [Afsarmanesh 84], and

- the Agis interface, an interactive graphical tool for adding or retrieving information from the database [Knapp 84b].

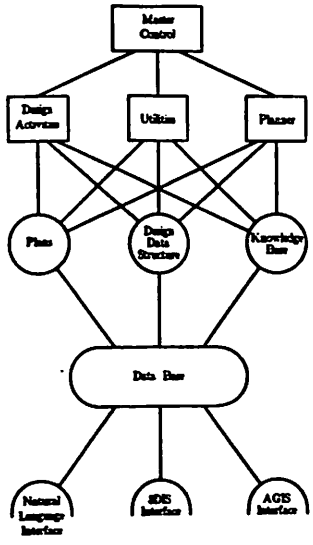


Figure 2: The ADAM system

2 Previous related work

The area estimation problem has received attention lately. There are mainly two approaches in the research of the problem: theoretical, which concentrates on wireability analysis and wiring space estimation, and experimental, which aims at developing area estimators for some specific layout systems based on previous experience with these systems.

2.1 Wireability analysis and wiring space estimation

Wireability analysis mainly deals with the problem of modelling the behavior of wires on a chip layout with the aim of predicting the amount of space to be allocated for wiring the interconnections. Almost all the wiring models proposed so far are *stochastic* in nature. One of the first well- known works in this field is by Sutherland and Oestreicher [Sutherland 73] in which they develop "a theory for choosing printed circuit board dimensions in order to avoid crowding of the printed wiring". Components are assumed to be randomly placed on the board. By assuming also that a "good" router is

used for wiring, the expected number of wires crossing a cutline at the center of the board is found to be $\frac{1}{4}$ the number of active pins on the circuit. It was also predicted that small boards are less efficient in area than large ones and that square boards are minimal in area. This work exhibits a good balance between theory and practice in the sense that each theoretical assumption and result is explained from a physical and intuitive point of view. The main drawback is that the assumption that components can be randomly placed can not be extended to integrated circuits, since it doesn't take into consideration the connectivity information of the various subcircuits.

2.1.1 Wiring space estimation for gate arrays

One of the early attempts to model the placement of components was with the observation by E.F. Rent of IBM of an empirical relation existing in well placed and partitioned gate array designs between the number of components in a partitioned subcircuit and the number of its external connections to other subcircuits. The relation (known since as Rent's rule) is further discussed in Section 4.7. Also at IBM, Heller et al [Heller 77], developed a stochastic model for the prediction of the wiring space on a 1-D placement of cells. Given the average wire length, the number of wires emerging from a cell is assumed to be a Poisson distributed random variable with a given parameter λ, and given a number of tracks allocated for wiring, the model predicts the probability of successfully routing the placement within the allocated space (in other words, the routability of the placement). The authors propose a heuristic extension to a 2-D placement by collapsing some of the rows and columns into a two row placement. This proposed extension has some weak points, especially in the calculation of the effective new model parameters after collapsing.

Feuer [Feuer 82] looked into the problem of predicting the wire length distribution and estimating the average wire length of IC's. The main result of that work is that if the partitioning of a logic graph exhibits Rent's rule, then the wire length distribution is expected to be of the form $q(r) \sim r^{-2(2-p)}$, where p is Rent's exponent and r is the wire length. One criticism of this model is that it assumes an "infinite" chip. Due to that, it may not correctly model the behavior of a real "finite" chip near its edges.

In Heller's wiring model, a linear placement is assumed to be "sliced" from an infinite size row. In [YehC 82], the model was modified to handle finite size placements by setting a limit on the maximum length of a wire, say L_{max} ports. Hence a port can only be "affected" by the L_{max} ports to its left (assuming wires travel from left to right). The same model is used to handle 2-D cell arrays by mapping each of the main diagonals in the cell array into one "effective" cell in the linear placement. The 1-D model is then used to estimate the routability of the array. This approach, however, gives the probability of successfully routing the array given the *sum* of the horizontal and vertical tracks at each cell, without looking at the distribution of these tracks in each direction. Hence, it might lead to results which may be overly optimistic. For example, the routing may result in insufficient tracks in, say, the vertical routing channel, but the sum of the tracks in the vertical *and* horizontal channels could still be predicted as "sufficient" by the model if tracks were overallocated to the horizontal channel and underallocated to the vertical channel.

In his thesis, Sastry [Sastry 85] addressed three problems related to wireability analysis in gate arrays. They are

- wiring space estimation,
- wire length distribution and average wire length, and
- routability.

For the first problem, a gate array is modelled as a grid of *channel intersections* and the problem of wiring space estimation is reduced to estimating the dimensions of these intersections. This is done by classifying the wires at an intersection into six different types, each modelled as a Poisson distributed random variable. From this and the knowledge of the wire length distribution, the expected widths and heights of each intersection are obtained. Asymptotic results were also obtained regarding the widths of the intersections for "large" chip, which were found to grow as the second moment of the wire length distribution. In treating the second problem, an equivalence relation was established between Rent's rule and the wire length distribution in the sense that Rent's rule (or any similar relation) does indeed provide all the necessary information about the

wire length distribution (or vice versa). In the particular case of Rent's rule, it was found to correspond to wire length distributions of the Weibull family. This was experimentally validated with data from real layouts. The third problem of routability was treated as two overlapping problems, vertical and horizontal routabilities. vertical (horizontal) routability is defined as the probability of successfully routing a chip for a given number of vertical (horizontal) channels and *sufficient* horizontal (vertical) channels. The approach was to find the factor by which to multiply the previous estimates of channel intersection dimensions in order to achieve a given routability. This model, however, separately treats the routability problem for vertical and horizontal channels and does not look at the coupling of the effects of wiring in the two dimensions. Furthermore, a chip is modelled as a "slice" of a doubly infinite array and hence, for real "finite" chips, the estimation of channel intersection dimensions near the edges may not be as accurate as those for the intersections well inside the chip.

2.1.2 Wiring space estimation of custom layouts

The above works assumed that designs are laid out in the gate array design style. The problem of wiring space estimation of custom logic was addressed by Syed [Syed 81]. He assumes that a placement of arbitrarily-sized rectangular blocks is given, along with their interconnections. He constructs a *channel graph* for the placement, where edges are the routing channels between the blocks and vertices are the intersections of these channels. The widths of the channels (edges of the graph) are then estimated using a stochastic model for wiring in which pins are assumed to be generated along a channel with a Poisson distribution, and wires lengths are exponentially distributed. Once this is done, the initial placement of the blocks is then modified so as to accommodate the channel width estimates with minimal total displacement of blocks. Routing is performed in two phases. First topological routes are assigned to wires. In the second phase, tracks are assigned to wire segments. During this phase, the placement may be further modified if more space than predicted is needed in some channels. This process of track assignment and placement modification is repeated iteratively until complete routing is achieved. Syed's model concentrates on estimating the area needed for *global* routing between the major blocks of a chip and does not deal with estimating the local wiring area within blocks and hence, the area of the blocks themselves.

Another approach to custom layout area estimation is by Ngai [Ngai 83]. He assumes given a channel, a number of tracks allocated to it and a set of nets of three types: right and left nets entering the channel from right and left, respectively, and center nets which are born and die inside the channel. The problem is to estimate the routability of the channel over all possible pin permutations. The channel wiring is modelled as a Markovian stochastic process with state changes occurring at net terminals (pins). A state at a pin is defined as a quadruple of random variables representing the number of nets of different types up to that pin, the density at that pin being a simple function of these variables. The conditional transition probabilities are found and a recurrence relation is used to find the state occupancy probabilities from which the distribution of the density function is determined and the routability estimated. Since this model predicts routability over all possible pin permutations and not the subset corresponding to "good" placements, the predicted routability figures may deviate from the actual ones.

2.2 General area estimation

In the field of general area estimation, we note two works. The first is by Liu [Liu 83] in which two models were developed : the point model for gate arrays and the rectangle model for general cells. In the point model, the well known separator theorem from VLSI complexity theory is used to prove some orders of growth of wiring area as a function of the number of cells. Besides being well known corollaries of the separator theorem, the results have little practical applicability to physical design since they are only expressed as orders and not as concrete numbers. For the rectangle model, the author develops an algorithm for testing the routability of a set of blocks given their relative placements and another algorithm to route the channels. Finally, the blocks in a chip are assumed to be implemented as PLA's and an algorithm is presented for partitioning a large PLA into several smaller PLA's.

The work by Ueda et al. [Ueda 85] is a more experimental approach to the area estimation problem. The authors describe a layout system, ALPHA, which uses the standard cell design style and in which an area estimator, CHAMP, is implemented. During the floor planning process, CHAMP estimates the areas of the different standard

cell blocks by using empirical formulas obtained by running numerous layout experiments on several designs. The area estimation figures presented for some chips are within 10% of the actual area. The estimation formulas are, however, empirical in nature and no theoretical backing is provided (in Section 4.7 we will attempt to theoretically justify one particular relation). Hence it is doubtful whether such formulas are applicable in another system.

3 Problem approach

The chip area estimation problem will be approached as follows. We first identify the problem and the research issues. This was done in section 1 and the problem was pratitioned into three levels of hierarchy, namely

- cell level,
- block level, and
- chip level.

The cell level area estimation problem can be approached in one of two ways:

- either by assuming that a library of *pre-constructed* cells is available, or
- establishing a simple *measure of complexity* of a cell and its relation to the cell area.

The block level area estimation is approached by first developing a 1-D model for estimating the area of a random logic block in the standard cell design style, given a cell level description of the block. Next, the model must be verified with respect to the assumptions made, through simulation, and finally we must validate the model assumptions with real layout data. The next step is to look at other layout methodologies for strucutred logic and developing different estimation models for different types of block design styles, namely

- RAM/ROM,
- PLA, and
- array structures.

Future research topics include extending the model to estimate area at the functional (register-transfer) structural level and integrating it within the ADAM system.

4 Research results

The emphasis of this work is on developing a probabilistic model for estimating the area of random logic blocks, given a cell level description in which cells are assumed to be laid out in the standard cell layout methodology. First we briefly look at the area estimation problem for individual cells. Next, the standard cell layout methodology is described in some detail. A model for standard cell placement is then presented in the following sections and finally some simulation verification and real data validations are presented.

4.1 Cell area estimation

Many layout methodologies have available a set of pre-constructed layouts of the most commonly used primitive functions. These layouts are known as cells and they implement functions such as simple and complex gates, flip-flops, register cells, adders etc.. The collection of cells is known as a cell library. Once a function definition is given as a logic graph, the next step is to map the nodes of the logic graph elements into a corresponding set of cells. This set is sometimes called the cell list, the set of nets representing the interconnections is known as the net list.

In many cases, the mapping between the nodes and the cells is straightforward, this is the case, when, for example, the function of a node in the logic graph has one and only one corresponding cell in the cell library. In other cases, however, the mapping is not quite as simple. This is due to several reasons :

1. There may exist more than one cell which implements a graph node function, each cell having slightly different added features. An example of this is a D flip flop node for which there are three corresponding cells, one with an "extra" preset input, another with a clear input and a third with both.

2. Fan-out and fan-in requirements enforce the choice of cells with the appropriate drive capabilities. Driver circuits usually take up relatively large amounts of layout area.

3. Many cell libraries have complex gate cells, which implement boolean functions more complex than simple boolean frimitive functions. In many cases, these cells may have embedded in them the functions of more than one node. Recognizing the appropriate node "pattern" for such complex cells is not a trivial task.

When a cell library is not at hand, we are faced with the problem of estimating the area of the cells that would implement the logic graph. In order to do that, we must establish a certain measure of the "complexity" for a given function. This measure would be used to estimate the area of the corresponding cell and hence it must embody the factors that have a first order effect on area. One such measure of complexity for simple functions which is generally used is the number of transistors that are needed to implement the function. In general, the area of simple functions layouts is mostly taken by active elements (transistors) and hence the number of transistors gives a good estimate of cell area. Also in many technologies, the number of transistors is proportional to the number of inputs to the function (in CMOS simple gates, for example, the number of transistors is twice the number of inputs). So, the number of inputs to a simple function could also be used as an estimate for its complexity and its area. Once the areas of the cells in a block have been estimated, the next step is to estimate the area of the block layout. The next sections deal with that issue by first discussing the standard cell design style in some detail and then proposing a model for placement and area estimation of standard cell blocks.

4.2 The standard cell design style

As explained in Section 1.3, the standard cell layout scheme uses a library of pre-designed cells (usually of the SSI/MSI level complexity). These cells usually have the same height, but different widths depending on the complexity of the function they are to implement. When placed together, the cells will abut vertically so that their power, ground (and sometimes global clock) lines will be connected automatically. Other signals are available on pins situated on top and bottom of the cells. In many cases cells are designed so that any cell signal is available on equivalent pins on top and bottom of the cell thereby giving the designer the freedom of connecting on the top or bottom. These cells are sometimes called *double entry* cells. The cells are arranged in rows of near

equal sizes. The spaces between adjacent rows are called channels. Channels are used to route the signal wires between the cells. Routing between cells in non-adjacent rows can be achieved by using *feedthroughs*, which consist of cells whose input and output pins are electrically equivalent. Feedthroughs are inserted in a row to permit multirow signals to be routed across the row instead of going around it.

4.2.1 Complexity of standard cell layouts

Several observations concerning the complexity of standard cell layouts have been noted. In one reference [Hick 83], the following is observed :

- As row length increases, the routing area grows quadratically with N (the number of cells) if the cells are *randomly* placed.
- This rate of growth becomes linear for *optimal* placement.
- Since it is very hard to obtain an optimal placement, the best that one can hope to achieve is an order of growth around $N^{1.5}$.

4.2.2 Placement

The placement procedure for standard cell layouts consists of assigning cells to rows (i.e. partitioning the design),and of assigning locations to cells within a row (i.e. finding a "good" permutation of the cells on the row). There are two major phases in the placement procedure, the *initial placement* and the *iterative improvement* phases. There are several techniques for placement. Among them are row assignment, pair linking, clustering, force directed, bipartitioning and row folding [Richard 84]. Of particular interest to us is the last technique. In our model, we chose to make the assumption that placement of standard cells is done by going through the following steps (Fig. 5) :

1. Place all the cells on a single row such that the average wire length (or some similar objective function) is minimized. This procedure is known as one-dimensional (1-D), or linear placement.
2. Fold the row into a number of smaller rows so that the resulting block layout satisfies some *a priori* set constraints on area and shape.

We chose to make this assumption for the following reasons

- 1-D placement and folding is a placement technique which has been successfully used for a long time in many layout systems (e.g. [Kang 83] [Ueda 85] [Schuler 72] [Supowit 83], [Feller 78]) and has been known to produce good placements.

- 1-D placement is much easier to model than 2-D placement (which is the case for most other placement techniques), hence it enables us to better analyze the placement problem and yield more confident area estimates.

4.2.3 Routing

After placement is finished, the routing of nets is to be done. This is usually done in two major phases, *feedthrough assignment* and *channel routing.*

Feedthrough assignment

Before carrying out the actual routing, one has to decide on the number and location of feedthroughs and nets which will be assigned to each feedthrough. Some cell libraries have two types of feedthroughs: *cell feedthroughs* which are cells inserted in rows during this phase and would permit the connection between two consecutive channels, and *terminal feedthroughs* which are parts of some cells. These are electrically equivalent terminals that run vertically through a cell which would otherwise be performing its ordinary function.

Several criteria are used to select feedthroughs such as: the number of cells to be inserted in a row, the maximum number of rows a net can feedthrough, the horizontal length a wire must travel before reaching the next feedthrough and whether to pass a wire through a feedthrough or to route around the row.

Channel Routing

Once feedthroughs are assigned and the global route each net must take is known, the track assignment is to be done. A track is a "clearance" which one wire can run through. The size of the track depends on the technology used and its associated design rules which specify the minimal spacing between wires. The track assignment is known as the classical channel routing problem for which several algorithms and variations exist. Among the algorithms for channel routing we have: The Lee, Hightower, Mattison, Hashimoto and Stevens, Rivest and Fiduccia algorithms [Lee 61], [Hightower 69], [Mattison 72], [Hashimoto 71], [Rivest 81]. In most technologies, vertical routes run

on one layer, horizontal routes on another. Doglegging is used to break routing conflicts [Deutsch 76]. Most good routers usually need one more track in addition to the channel density for routing the channels. The *local density* at a certain ordinate is defined as the number of nets crossing a cutline at that ordinate. The maximum local density over the channel length is the *channel density*. The channel density represents a lower bound on the channel width. Local density is usually computed at grid points and is a discontinuous function.

4.3 A probabilistic model for standard cell area estimation

The main research problem addressed in this section is that of estimating the amount of space needed for wiring in a standard cell block. Wiring area is measured as the number of horizontal tracks needed for routing between the rows. It is assumed that the *cell level* description of the block is given as a list of cells and their interconnections. We present a simple probabilistic model for cell placement and interconnections.

4.3.1 Assumptions

We make a number of assumptions about standard cell layouts. They are

- Cells are arranged in rows of roughly equal sizes.
- Cells have pins to which wires can be connected.
- Cells are of the **double entry** type, i.e., equivalent pins are present on both top and bottom parts of each cell.
- All cells have constant **pin pitch** which is defined as *the distance between two consecutive pins or pin slots* and cell width is an integer multiple of the pin pitch. As a result, the cell widths will be expressed in pin pitches. For example, a cell may be five pitches wide and have two pins, one at pin position (slot) 2, the second at pin slot 4.
- All nets are two point nets (i.e. connecting exactly two pins)
- All wires follow **minimal rectilinear paths**; no backward moves are allowed.
- As mentioned in Section 4.2.2, placement is done by first placing the cells on *one* row so that the total routing area in minimized (or nearly minimized), then *folding* the row into the desired number of rows so that a desired aspect ratio is achieved.

While this last assumption seems rigid, it allows us to analyze the problem in a much simpler fashion. The main idea can be expressed as follows ; *Given the extreme complexity of modelling the problem as a two-dimensional one, assume that the placement is done by folding. Starting with the one row linear placement, which is relatively easier to model, predict the wiring space requirements after folding, given that pin positions are transformed in a predictable manner as explained in Section 4.5.* If the resulting estimates are grossly inaccurate, we may need to relax this assumption later in the research.

4.3.2 Description

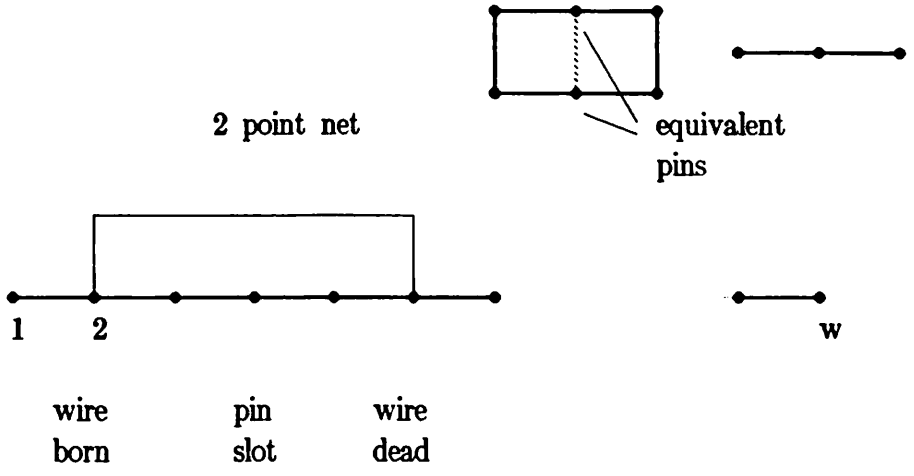


Figure 3: The row model

An example of the row model is shown in Fig. 3. Some of the characteristics of that model are the following :

- Since cells are of the *double entry* type, the top and bottom equivalent pins in a cell are actually "collapsed" into one "equivalent" pin in the model.
- A cell row is modelled as a row of *pin slots* where the distance between two adjacent slots is equal to the pin pitch.
- A pin slot is occupied if a wire emerges from it.
- The total number of pins slots in the block (i.e. the sum of the widths of all the cells, in pin pitches) is w.

- The total number of nets in the block is **N**.

4.4 Single row case

We now look at the case of estimating the routing requirements of a single row configuration, that is, when the cells are all placed on one single row. This analysis is necessary for the more general multiple row case, which will be presented in the Section 4.5. We make the following additional assumptions :

- Pin slots are labelled 1,2, . . . ,w from left to right.
- Wires travel from left to right. A wire starting at a pin slot is said to be *born* at that slot. The birth of a wire at a pin slot i is a random event with probability $p_B(i)$.
- A wire born at a pin slot i will *terminate* (or *die*) at a certain pin j to the right of i. This event is assumed to be dependent only on the distance between i and j, i-j, (which is also the length of the wire) and not on the individual values of i and j. In other words, the length of a wire does not depend on where it is born.
- The length of a wire is assumed to be a random variable with a probability density function $p_L(l) = Pr\{L = l\}$.

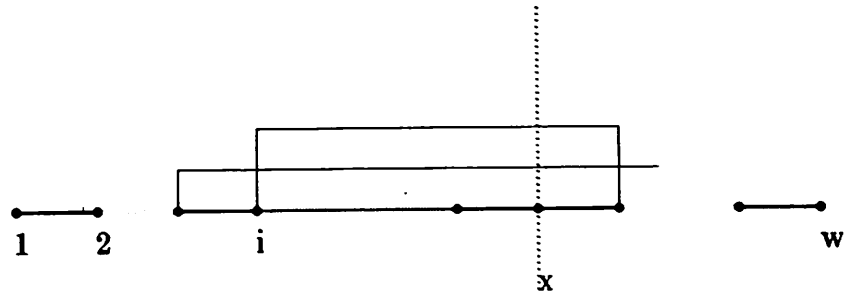


Figure 4: Density at a point x

We will define the **density** a point x, $d(x)$, as *the number of wires crossing a vertical cutline at* x as shown in Fig. 4. The objective is to estimate the average value of $d(x)$. To do that, we have to look at the following events:

- A(i) = A wire is born at i, and
- B(i,x) = A wire *crosses* x given that is is born at i.

The probability of the first event is given by $p_B(i)$. Now a wire born at i will cross x iff its length L is such that

$$x - i \leq L \leq w - i$$

So,

$$Pr\{B(i,x)\} = Pr\{x - i \leq L \leq w - i\} = \sum_{m=x-i}^{w-i} p_L(m)$$

Now, the probability of a wire being born at $i \leq x$ *and* crossing x is Pr{A(i)}.Pr{B(i,x)}. So the **average** number of wires crossing x is given by

$$E\{d(x)\} = \sum_{i=1}^{x} p_B(i) \sum_{m=x-i}^{w-i} p_L(m) \qquad (1)$$

The number of tracks needed is equal to the maximum density throughout the row. This means that, in order to estimate the number of tracks, one must estimate the average value of the variable $d' = \max_x d(x)$. This, however, may be a very difficult task, especially for arbitrary forms of $p_L(l)$ and $p_B(i)$. One way of approximating the average value of d' is to use $\max_x E\{d(x)\}$. It was actually shown [Sastry 85] that

$$\max_x E\{d(x)\} \leq E\{\max_x d(x)\}$$

This means that the approximation is actually a lower bound on $E\{d'\}$ In section 4.8 we will show some simulation results and compare the two variables.

The next task is now to make realistic assumptions about $p_B(i)$ and $p_L(l)$. The most general assumption for $p_B(i)$ is to assume that pins are **uniformly** distributed among the w slots. Hence for N wires, the probability of a wire being born at i is

$$p_B(i) = p_B = \frac{N}{w}$$

.

The choice of $p_L(l)$ has been discussed in the literature on wiring space estimation [Sastry 85] [Heller 77] and many distributions were suggested, such as the Poisson, exponential and geometric distributions. In this model we assume that $p_L(l)$ is *geometric.*

So, $p_L(l) = p\,q^{l-1}$,where $\frac{1}{p} =$ average wire length and $q = 1 - p$

Given these two forms of $p_B(i)$ and $p_L(l)$, it is possible to obtain a closed form for $E\{d(x)\}$

$$E\{d(x)\} = \frac{N}{wpq}(1-q^x)(1-q^{w-x+1}) \tag{2}$$

Now, in order to find the maximum of $E\{d(x)\}$, we set $\frac{d\{E\{d(x)\}\}}{dx}$ to zero and solve for the root(s), so

$$\frac{N}{wpq}\{-q^x log\ q(1-q^{w-x+1}) + q^{w-x+1} log\ q(1-q^x)\} = 0$$

or,

$$-q^x + q^{w-x+1} = 0$$

So the maximum local density occurs at $x^{max} = \lceil\frac{w+1}{2}\rceil$[2], and the estimate for the track requirements is $E\{d(x^{max})\}$.

4.5 The Multiple Row Model

Before we extend the model in the previous section to the multiple row case, we should remind ourselves of the assumption that the placement of standard cells is done by first placing the cells in one single row so the average wire length is made as small as possible. The linear placement is then folded into n rows. The characteristics of the single row are discussed in the previous section. In this section, we focus on modelling the folding process, its effects on wiring, and how to estimate the wiring area after folding.

Once a placement is done on a single row, the folding is performed as shown in Fig. 5. The initial row is "snaked" around itself into n rows. Start with a single row linear placement and consider a pin slot p on that row. Now fold the linear placement into n rows, each of width $r = \frac{w}{n}$. Let $y(p), 1 \leq y(p) \leq n$ denote the row in which the pin slot p will be located, and let $x(p), 1 \leq x(p) \leq w$ be its new x position within that row. The following lemma indicates the relation between p,$x(p)$, and $y(p)$

Lemma 1: Consider a pin slot p in a single row block of width w. After

---

[2]Even though the function differentiated is a *continuous* function of x, it can be easily shown that the function defined over the discrete subset of its domain is indeed maximum at x^{max}.

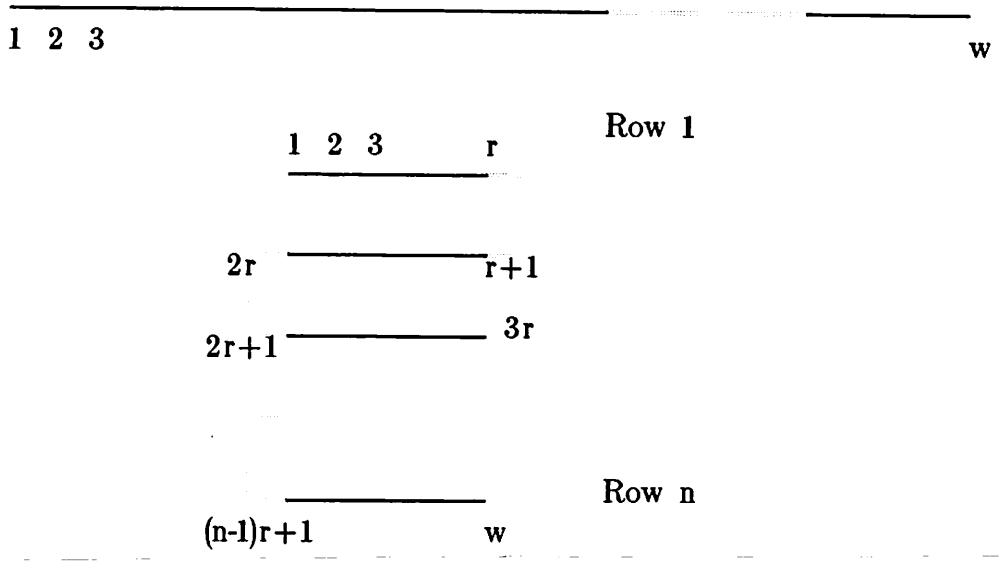


Figure 5: Folding of a row

folding the row into n rows, each of width $r = \frac{w}{n}$, the new position of p will be given by $x(p)$,$y(p)$, such that

$$x(p) = \begin{cases} p-2kr & \text{if } (p \; div \; r)=2k \quad \text{(odd rows)} \\ 2kr-p+1 & \text{if } (p \; div \; r)=2k+1 \quad \text{(even rows)} \end{cases}$$
$$y(p) = (p \; div \; r) + 1$$

where (p div r) denotes the integer division of p by r.

Proof: The proof can be directly inferred from Fig. 5.

As an example of Lemma 1, take $p = r+5$, and assume $r > 5$. The new position of the pin slot will be in the second row ($(r+5)$ div $r+1=2$) and its position in that row is $2r-(r+5)+1 = r-4$.

Since the cells are of the double entry type, wires can be routed from either the top or the bottom equivalent pins for any cell. When folding is performed, the wires must be re-routed and, in general, do not "snake around" with the cell rows. Hence folding refers only to the placement of the cells and the new folded configuration of cells requires a new pass by the router.

4.5.1 Estimating channel density

Now that the effect of folding on pin locations has been modelled, the next step is to estimate the wiring space after the folding of the one row placement modelled in Section 4.4. We do so by attempting to predict the channel densities of the placement. The **local channel density** at x in an n row block, $d_n(x)$, is defined as the total number of wires crossing a vertical cutline that passes through x, $1 \leq x \leq r$, and runs through all n rows of the block. Effectively, $d_n(x)$ is the sum of the local densities at x of all the n routing channels. Fig. 6 illustrates a cutline at x. Clearly, a wire will cross that cutline iff it starts to the left (right) of x and terminates to the right (left) of x. This is so because wires are assumed to follow minimal rectilinear paths with no backward moves.

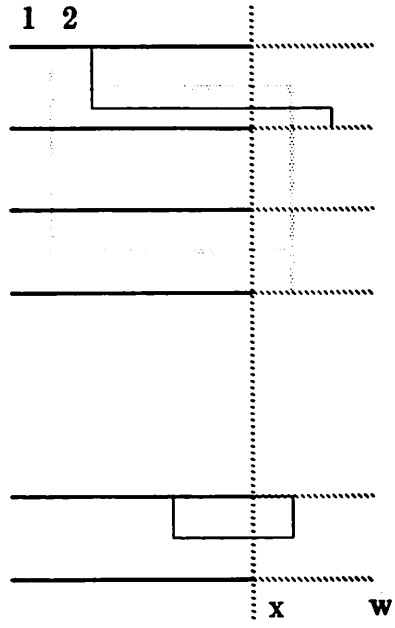


Figure 6: Cutline at x

Note that $d_n(x)$ does not change with the route a wire takes from source to destination. In Fig. 6 for example, a wire from row 1 to row 4 can either go vertically down to row 4 then horizontally, or alternatively horizontally through row 1 then down to row 4, or even "zigzag" down through rows 1,2,3 and 4. In every case, that wire will contribute one track to the local density at x. In other words, *the local density is only dependent*

on the placement of the components and is independent of the routing, assuming minimal rectilinear paths of wires.

Since linear placement has been fixed prior to folding, the aim is to predict the effects of folding on the linear placement and from that, the channel density. In Fig. 7, we describe the relation between the folded placement and the corresponding linear placement as follows :

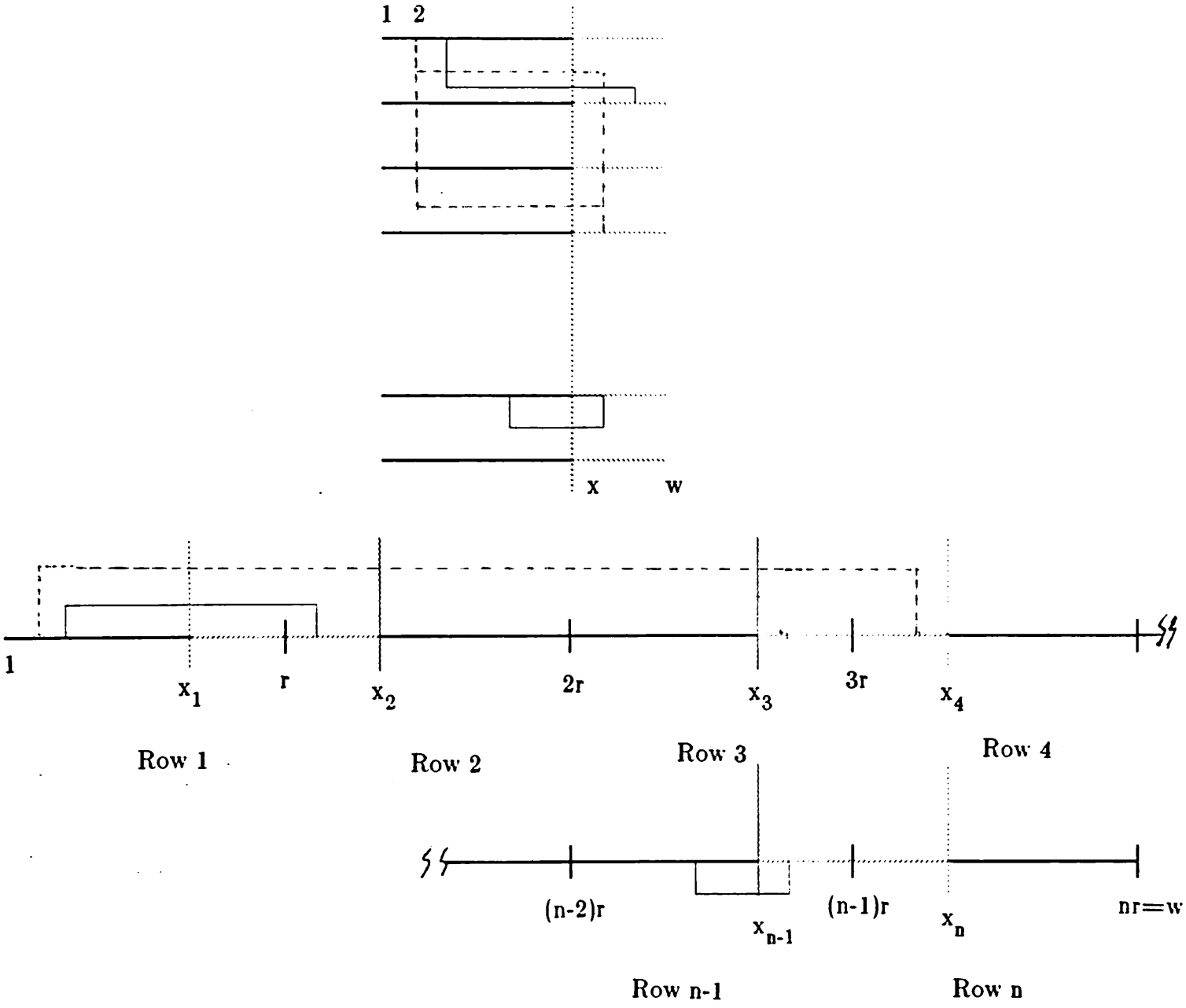


Figure 7: The unfolding of a row

- Part (a) of Fig. 7 shows the folded block and a cutline at x. Regions to the right of x are cross-hatched, those to the left are solid.

- Part (b) of Fig. 7 shows the unfolded row. Note that the left and right regions in (a) are now alternating in (b). Also note that, since the cutline at x crosses all the rows in (a), it will translate into n cutlines whose combined densities represent the local density at x, $d_n(x)$. The location of the n resulting cutlines, $x_1, x_2, \ldots, x_n$ is given by the following lemma.

  Lemma 2: A cutline at x in an n row block will correspond to n cutlines in the unfolded row, located at $x_1, x_2, \ldots, x_n$, given by

$$x_i = \begin{cases} 2kr - x + 1 & i = 2k \\ 2kr + x & i = 2k+1 \end{cases}$$

  Proof: This can be proved directly from Lemma 1.

Having described the effects of folding on a linear placement, we look next at estimating the local density at x. Let $W_n(x)$ be the random variable denoting the number of wires crossing a cutline at x in the folded placement. In order to specify which wires will cross the cutline (and hence contributing to $W_n(x)$), let's define the intervals $I_1(x), I_2(x), \cdots, I_n(x), I_w(x)$ on the unfolded row, given by

$$I_i(x) = [x_{i-1}, x_i], \quad i = 2, \ldots, n$$

$$I_1(x) = [1, x],$$

$$I_w(x) = [x_n, w]$$

The following theorem characterizes the wires crossing the cutline at x.

Theorem 3: A wire born in $I_i(x)$ will contribute one to the cut density at x iff it terminates in $S_i(x)$, where $S_i(x) = \{I_{i+1}(x), I_{i+3}(x), \ldots, I_{i+2k+1}, \ldots\}$.

Proof: The theorem can be proven by noting that the intervals in $S_i(x)$ correspond, after folding, to the regions on the other side of cutline x. In other words, if $I_i(x)$ corresponds to a region on the left (right) of x, then the intervals in $S_i(x)$ correspond to the regions on the right (left) of x.

By Theorem 3, the total number of wires crossing the cutline at x is the sum of all the wires starting in $I_1(x), I_2(x), \cdots, I_n(x)$, and terminating in $S_1(x), S_2(x), \cdots, S_n(x)$,

respectively. Let $WI_i(x)$ be the random variable denoting the number of wires starting in $I_i(x)$ and terminating in $S_i(x)$. $W_n(x)$ can now be written as

$$W_n(x) = \sum_{i=1}^{n} WI_i(x)$$

$WI_i(x)$ represents the *contribution* of the wires starting in interval $I_i(x)$ to the local density at x. Clearly, $WI_i(x)$ is the sum of the wires starting in $I_i(x)$ and terminating in each of the intervals $I_j(x)$ such that $I_j(x) \in S_i(x)$. Now let $WI_{i,j}(x)$ be the random variable denoting the number of wires born in an interval $I_i(x)$ and terminating in $I_j(x)$, where $i < j$, then $WI_i(x)$ can be written as

$$WI_i(x) = \sum_{j | I_j \in S_i(x)} WI_{i,j}(x)$$

We must next find an estimate for $WI_{i,j}(x)$. To do so, let's take a wire born at point t inside $I_i(x)$, having a length L [Fig. 8]. This wire will terminate in $I_j(x)$ if its length is such that $x_{j-1} - t \leq L \leq x_j - t$. This corresponds to the event $C(t,j,x)$

$$C(t,j,x) = \{x_{j-1} - t \leq L \leq x_j - t \text{ given that a wire is born at t}\}$$

which has a probability

$$\Pr\{C(t,j,x)\} = \sum_{m=x_{j-1}-t}^{x_j-t} p_L(m)$$

The average value of $WI_{i,j}(x)$ is then given by

$$E\{WI_{i,j}(x)\} = \sum_{t=x_{i-1}}^{x_i} p_B(t) \Pr\{C(t,j,x)\}$$

or,

$$E\{WI_{i,j}(x)\} = \sum_{t=x_{i-1}}^{x_i} p_B(t) \sum_{m=x_{j-1}-t}^{x_j-t} p_L(m) \tag{3}$$

Let's use $E\{WI_{i,j}(x)\}$ as an estimate of $WI_{i,j}(x)$. We can then write the average contribution of $I_i(x)$ to $W_n(x)$ as the expected value of $WI_i(x)$, given by

$$E\{WI_i(x)\} = \sum_{j | I_j \in S_i(x)} \sum_{t=x_{i-1}}^{x_i} p_B(t) \sum_{m=x_{j-1}-t}^{x_j-t} p_L(m) \tag{4}$$

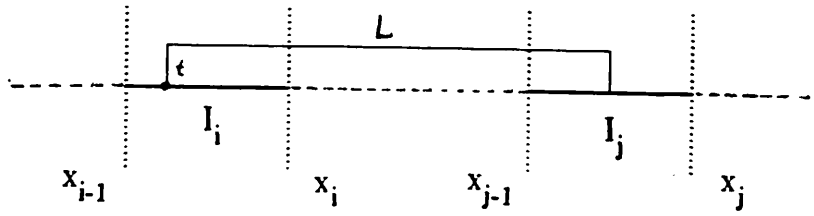


Figure 8: Estimating $WI_{i,j}(x)$

We classify the WI_i's into two groups, $WI_{2k}(x)$ and $WI_{2k+1}(x)$, depending on i being even or odd (The contributions of the intervals at the edges, $I_1(x)$ and $I_w(x)$ are treated separately). This is necessary because $I_{2k}(x)$ and $I_{2k+1}(x)$ (and also their corresponding S_i's) have different expressions for the interval bounds (and hence different summation limits). So for an *even* interval $I_{2k}(x) = [x_{2k-1},x_{2k}] = [2(k-1)r+x,2kr-x+1]$, and for an *odd* interval $I_{2k+1}(x) = [x_{2k},x_{2k+1}] = [2kr-x+1,2kr+x]$. So now we can write the average value of the contributions of $WI_{2k}(x)$ and $WI_{2k+1}(x)$ as follows :

$$E\{WI_{2k}(x)\} = \begin{cases} \sum_{l=k}^{n/2-1} E\{WI_{2k,2l+1}(x)\} & n \text{ even} \\ \sum_{l=k}^{\frac{n-1}{2}} E\{WI_{2k,2l+1}(x)\} & n \text{ odd} \end{cases} \tag{5}$$

and,

$$E\{WI_{2k+1}(x)\} = \begin{cases} \sum_{l=k+1}^{\frac{n}{2}} E\{WI_{2k+1,2l}(x)\} & n \text{ even} \\ \sum_{l=k+1}^{\frac{n-1}{2}} E\{WI_{2k+1,2l}(x)\} & n \text{ odd} \end{cases} \tag{6}$$

From these equations we can get the average total contribution of the odd and even intervals as follows :

$$E\{WI_{even}(x)\} = \begin{cases} \sum_{k=1}^{n/2-1} E\{WI_{2k}(x)\} & n \text{ even} \\ \sum_{k=1}^{\frac{n-1}{2}} E\{WI_{2k}(x)\} & n \text{ odd} \end{cases} \tag{7}$$

and that of the odd intervals

$$E\{WI_{odd}(x)\} = \begin{cases} \sum_{k=1}^{n/2-1} E\{WI_{2k+1}(x)\} & n \text{ even} \\ \sum_{k=1}^{\frac{n-3}{2}} E\{WI_{2k+1}(x)\} & n \text{ odd} \end{cases} \quad (8)$$

We must also consider the effects of the first and last intervals, namely,

$I_1(x) = [1, x_1 = x]$

and

$I_w(x) = [x_n, w]$

For $I_1(x)$,

$$E\{WI_{1,2k}(x)\} = \sum_{t=1}^{x} p_B(t) \sum_{m=x_{2k-1}-t}^{x_{2k}-t} p_L(m)$$

and the total contribution of $I_1(x)$ is

$$E\{WI_1(x)\} = \begin{cases} \sum_{k=1}^{\frac{n}{2}} E\{WI_{1,2k}(x)\} & n \text{ even} \\ \sum_{k=1}^{\frac{n-1}{2}} E\{WI_{1,2k}(x)\} & n \text{ odd} \end{cases} \quad (9)$$

Finally, for the last interval, $I_w(x)$, two cases appear

- n even, so

$$E\{WI_{2k,w}(x)\} = \sum_{t=x_{2k-1}}^{x_{2k}} p_B(t) \sum_{m=x_n-t}^{w-t} p_L(m)$$

and the contribution of $I_w(x)$ is

$$E\{WI_w(x)\} = \sum_{k=1}^{\frac{n}{2}} E\{WI_{2k,w}(x)\} \quad (10)$$

- n odd, so

$$E\{WI_{2k+1,w}(x)\} = \sum_{t=x_{2k}}^{x_{2k+1}} p_B(t) \sum_{m=x_n-t}^{w-t} p_L(m)$$

and the contribution of $I_w(x)$ is

$$E\{WI_w(x)\} = \sum_{k=1}^{\frac{n-1}{2}} E\{WI_{2k+1,w}(x)\} + E\{WI_{1,w}(x)\} \tag{11}$$

$$E\{WI_w(x)\} = \sum_{k=1}^{\frac{n-1}{2}} E\{WI_{2k+1,w}(x)\} + \sum_{t=1}^{x} p_B(t) \sum_{m=x_n-t}^{w-t} p_L(m)$$

Finally the average value of $W_n(x)$ can be written as

$$E\{W_n(x)\} = E\{W_{even}(x)\} + E\{W_{odd}(x)\} + E\{W_w(x)\} + E\{W_1(x)\} \tag{12}$$

Equation (12) gives an average value of $W_n(x)$ as a function of $p_B(t)$ and $p_L(m)$. In order for the estimate to be of practical value, it is important to know these two distributions. As in the previous section, we will assume that

(a) Pins are uniformly distributed along cell rows, so $p_B(t) = \frac{N}{w}$

(b) Wire lengths are geometrically distributed, so $p_L(m) = pq^{m-1}$, where $\frac{1}{p}$ is the average wire length and $q = 1-p$.

Under these assumptions, it is possible to find closed form expressions for $E\{W_{even}(x)\}$, $E\{W_{odd}(x)\}$, $E\{W_1(x)\}$ and $E\{W_w(x)\}$, and hence $E\{W_n(x)\}$ can be expressed as

$$E\{W_n(x)\} = \frac{N}{w} \times \frac{1}{pq(1-q^{2r})} \left\{ 2(1-q^w)(1-q^{2r-2x+2})(1-q^x) \right. \tag{13}$$

$$\left. + (1-q^{2x})(1-q^{2r-2x+2}) \left[n-2-2q^{2r} \frac{1-q^{w-2r}}{1-q^{2r}} \right] \right\}$$

For n even, and

$$E\{W_n(x)\} = \frac{N}{w} \times \frac{1}{pq(1-q^{2r})} \left\{ 2(1-q^{w-r})(1-q^{r-x+1})(1-q^x)(2+q^x+q^{r-x+1}) \right. \tag{14}$$

$$+ (1-q^{2x})(1-q^{2r-2x+2}) \left[n-2-2q^{2r} \frac{2-q^{w-r}-q^{w-3r}}{1-q^{2r}} \right] \Bigg\}$$

For n odd, $n \geq 3$.

Equations (13) and (14) give an estimate for $d_n(x)$, the local density of a cutline at x. One must, however, allocate at least a number of tracks equal to the *maximum* density across any cutline parallel to x, or CH_n. So, as in the single row case, the track requirement is *approximated* by

$$CH_n = \max_x \{E\{W_n(x)\}\} \tag{15}$$

Note that, as in the single row model, this represents a lower bound on the actual track requirements.

Another approximation is also introduced by the fact that we are estimating, at each x, the *sum of the local densities* of the individual routing channels instead of looking at each channel separately from the others, as shown in Fig. 9.

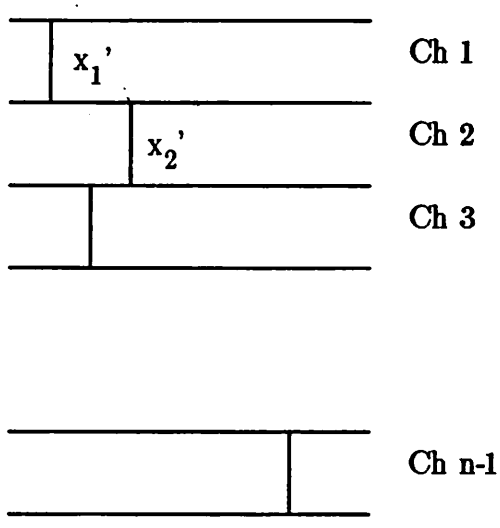


Figure 9: Maximum density calculation

Suppose that $C_1(x), C_2(x), \ldots, C_n(x)$, are the local densities at x of channels 1,2, . . . ,n, respectively, then

$$d_n(x) = \sum_{i=1}^{n} C_i(x)$$

and

(16)

$$CH_n = max_x \{d_n(x)\} = max_x \{\sum_{i=1}^{n} C_i(x)\}$$

This is not, in fact, the actual track requirement estimate since the functions $C_1(x), C_2(x), \ldots, C_n(x)$ may be maximum at different values of x, say, $x_1', x_2', \ldots, x_n'$, respectively, and the actual track requirement will be

$$CH_n' = \sum_{i=1}^{n} C_i(x_i') = \sum_{i=1}^{n} max_x\{C_i(x)\} \tag{17}$$

The following theorem establishes a relation between CH_n and CH_n'.

Theorem 4: $CH_n \leq CH_n'$ with equality iff $\forall i, x_i' = x_0'$, where x_0' is the point for which $d_n(x_0') = CH_n$.

Proof: The expression for CH_n in (16) can be re-written, assuming that maximum channel density occurs at x_0', as

$$CH_n = \sum_{i=1}^{n} C_i(x_0') \tag{18}$$

note that each C_i is maximum at x_i', hence

$$\forall i, C_i(x_0') \leq C_i(x_i')$$

and the inequality between Equations (18) and (17) follows.

This approximation is, however, not a severe one due to two reasons :

(a) The channel densities are, in general maximum near the centers of the channels. Hence the points $x_1', x_2', \ldots, x_n'$ are, in practice, close to each other.

(b) Local densities do not, in general, vary a lot around maximum density points.

Under these considerations, it is safe to assume, in general, that $x_1', x_2', \ldots, x_n'$ occur in the neighborhood of x_0' and that $CH_n \approx CH_n'$.

As in the single row case, estimating x_0' is done by setting the derivative of $E\{W_n(x)\}$ to zero and solving for x. It may not be possible, however, to obtain an analytical solution for x_0', in which case numerical methods must be used to find an approximate solution. In the particular case of Equations (13) and (14), the derivative of $E\{W_n(x)\}$ is a fourth degree polynomial in q^x whose root(s) can be found numerically using the Newton-Raphson method.

4.6 Estimating the feedthroughs

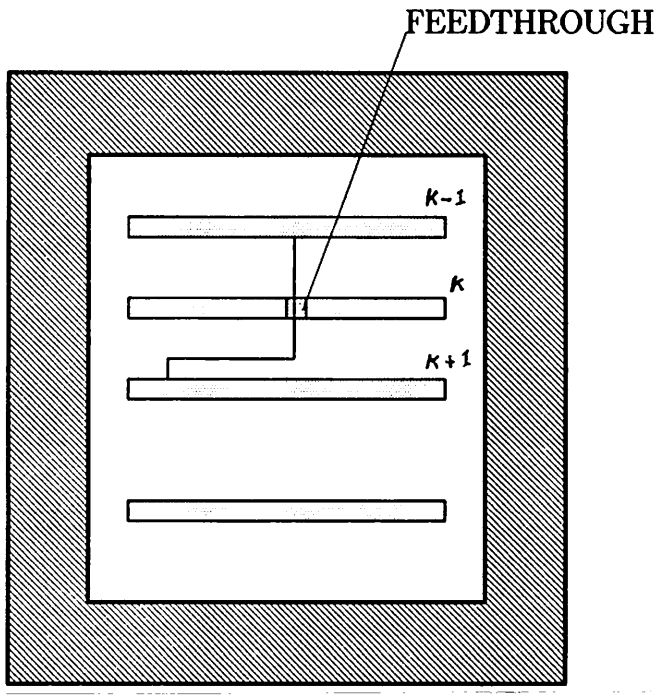


Figure 10: Feedthroughs in a chip

Now that we have estimated the wiring space requirements in the vertical direction (i.e. the channel densities), the next step is to estimate the amount of space needed in the horizontal dimension. We make the assumption that vertical wires which run more than one row are routed using feedthroughs [Fig. 10]. Estimating the space required by the vertical wires requires estimating the number of feedthroughs in the cell rows. Feedthroughs are used to route wires that connect cells which are placed more than one row apart. They have the advantage of avoiding the routing of wires "around" the cell rows, thereby avoiding the need for possible extra tracks. Feedthrough cells must,

however, be inserted in cell rows[3], hence the necessity of adjusting the cell placement during the routing phase. Feedthroughs also increase the chip width since, by inserting them in cell rows, they increase the sizes of these rows. So it is necessary to have an estimate for the increase in row sizes due to feedthroughs for an area estimate of the chip to be accurate.

The model described in the previous section can be used to estimate the number of feedthroughs in cell rows. Let FT_i be the random variable denoting the number of feedthroughs in row i . A wire will need a feedthrough if it connects cells which are more than one cell row apart. For example, in Fig. 10, a wire born in row k-1 and dead in row k+1 will require a feedthrough in row k. Conversely, if we look at row k , all wires born in row 1 or 2 or . . . or k-1, *and* dead in rows k+1 or k+2 or . . . or n will require one feedthrough each in row k . Hence, the task of estimating the number of feedthroughs in row k is equivalent to estimating the number of such wires.

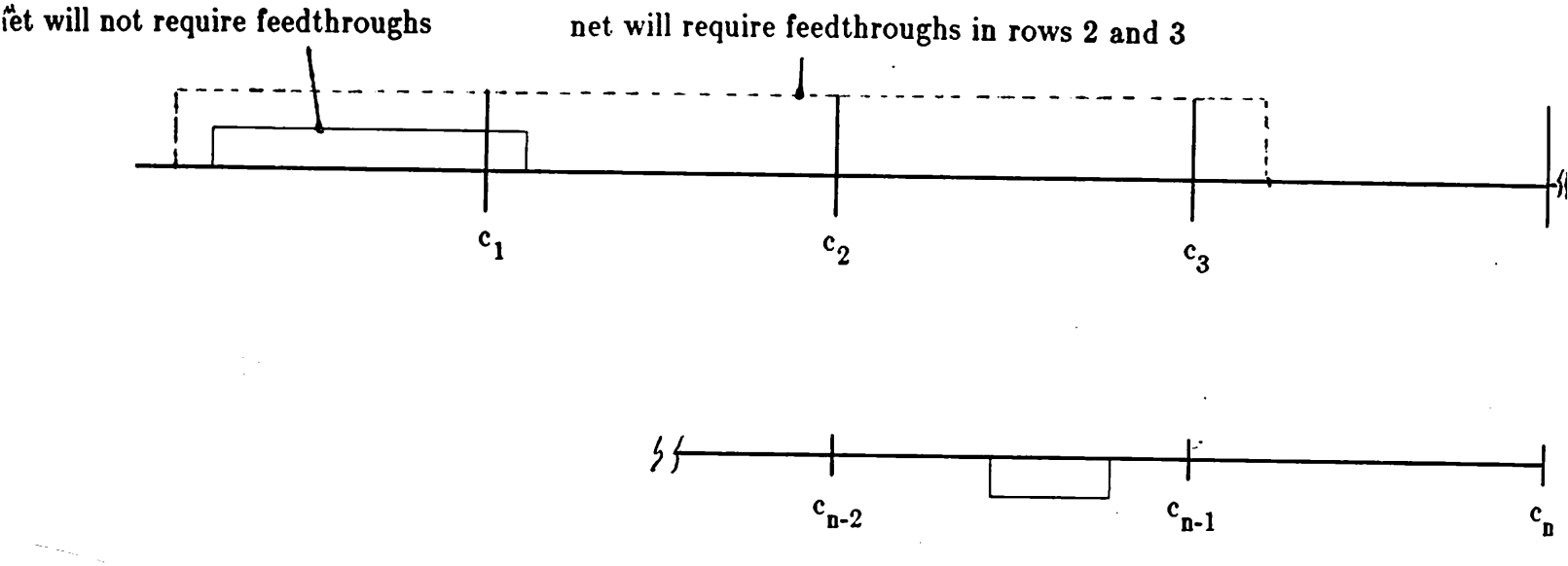


Figure 11: Unfolded row

Let's look again at the unfolded single row placement shown in Fig. 11. Let

[3]This is the only type of feedthroughs allowed in the model

$c_1, c_2, \ldots, c_{n-1}$ be the pin slots at which the linear row is folded. Since the resulting cell rows are assumed to be of equal sizes, then $c_1 = r, c_2 = 2r, \ldots, c_{n-1} = (n-1)r$ and a row k can be looked at as an interval $[c_{k-1}+1, c_k]$.

Now let $WR_{i,j}$ be the random variable denoting the number of wires born in row i and dead in row j. As in the previous section, we can express $E\{WR_{i,j}\}$ as

$$E\{WR_{i,j}\} = \sum_{t=c_{i-1}+1}^{c_i} p_B(t) \sum_{m=c_{j-1}+1-t}^{c_j-t} p_L(m) \tag{19}$$

$$= \sum_{t=(i-1)r+1}^{ir} p_B(t) \sum_{m=(j-1)r+1-t}^{jr-t} p_L(m)$$

and the estimate for the number of feedthroughs in row k is equal to the average number of wires born in rows 1 or 2 . . . or *k-1* and dead in *k+1* or *k+2* . . . or n, given by

$$E\{FT_k\} = \sum_{i=1}^{k-1} \sum_{j=k+1}^{n} E\{WR_{i,j}\} \qquad k=2, \ldots, n-1 \tag{20}$$

The chip width (without I/O) must be at least as wide as the widest cell row, since the rows are assumed to be of equal sizes initially. The widest cell row is the one that has the maximum number of feedthroughs, so we can write the width of the widest row as $r + max_k\{E\{FT_k\}\}$.

Now if we assume, as in the previous sections, that the pins are *uniformly* distributed along slots, and wire lengths are *geometrically* distributed, it is possible to obtain a closed form expression for $E\{FT_k\}$:

$$E\{FT_k\} = \frac{Nq^r}{wp}(1 - q^{(n-k)r})(1 - q^{(k-1)r}) \tag{21}$$

Setting $\frac{dE\{FT_k\}}{dk} = 0$, and solving for k, the maximum number of feedthroughs occurs in row k^{max}, where

$$k^{max} = \lceil \frac{n+1}{2} \rceil$$

4.7 The effect of the number of rows on track requirements

We now take a different and rather intuitive approach to the problem of predicting the order of growth of routing track requirements as the number of rows is increased. This question is of some importance because of the following :

- The number of rows in a chip determines, in general, the shape and aspect ratio of the chip (or block).
- The larger the chip is, the more rows are needed for placement in order to avoid making the chip oddly shaped.

As seen in the previous sections, the densities of the routing channels depend mainly on the placement of the components on the chip and thus, the better the placement is, the less routing space is needed. So, in order to determine a relation between routing space and chip configuration, it is important to be able to characterize "good" placements. Put in different terms, the question is, what behavior (if any) do good placements exhibit, and how?

One observed characteristic of some "good" placements is the empirical relationship known as **Rent's rule**. Rent's rule is an empirical relation which has been observed in well partitioned and placed circuits and chips between the number of components in a subcircuit and the number of external terminals of this subcircuit. Qualitatively, it says that the number of external terminals of a subcircuit tends to grow slower than the number of components in that subcircuit. The intuitive explanation of this behavior is as follows. As the number of components in a partition increases, a proportional increase in the number of available pins would take place. Not all these pins, however, would be expected to be external pins. This is because components in the same partition in a well partitioned circuit will have, on the average, higher connectivity to components inside the partition than to ones outside the partition. Hence many of the pins will actually be "consumed" inside the partition and a relatively small part of the pins will actually connect to components outside the partition. Quantitatively, this rule exhibits itself as a power law relation of the form

$$T=AC^{p} \tag{22}$$

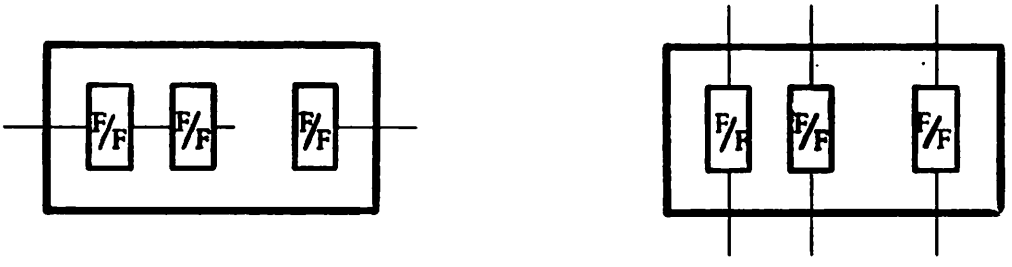
where

- T = number of external terminals (pins) in a subcircuit
- A = average number of terminals (pins) per component
- C = number of components in the subcircuit
- p = Rent's exponent, $0 \leq p \leq 1$

This relation was first observed by E. F. Rent of IBM in the late 1960s, and independently, by several others. Donath [Donath 69] of IBM derived the same relation from a stochastic model which assumes a hierarchical design process. In 1971 Landman and Russo published an extensive study of the relation [Landman 71] where they carried out numerous experiments on partitioning large "real-life" circuits. Their main conclusions were

1. Rent's rule is actually a two region relationship, where region I is the power law relation of Equation 1 and region II is governed by a more complex relation.

2. Rent's exponent, p, lies, in practice, between 0.47 and 0.75 and depends upon the structure of the circuit and the partitioning algorithm. In a subsequent paper, Russo concluded that, for the same partitioning algorithm, p tends to be high for high performance circuits, and low for low performance circuits. This can be visualized by the simple example in Fig. 12.



n bit shift register
T = 2 , p = 0

n bit parallel latch
T = K*C, p = 1

Figure 12: Extreme cases for Rent's rule

The extensive amount of data that has been investigated successfully for its adherence to Rent's rule suggests that it tends to apply to well placed circuits. So, in the following,

Rent's rule will be assumed to hold in standard cell layouts and, based on that, we will try to to find its relation to routing track requirements.

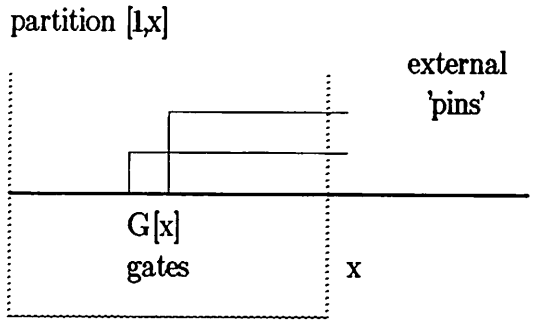


Figure 13: partition [1,x]

Consider again the model described in the previous sections. Take a row of width w, denoted by the interval [1,w] in Fig. 13. Next take a cutline at a point x. The region [1,x] to the left of x can be looked at as a partition of the row [1,w] having a number of gates which, in general, depends on the value of x. It would actually be realistic to assume that the number of gates is proportional to x, since the cells are of equal height and their internal layout is mainly composed of gates, hence a gate usually has a fixed "pitch" in the x direction. For generality, assume that the number of gates in [1,x] is given by $G(x)$. Now, if we further assume that Rent's rule does actually characterize good placements, as should be the case in the row [1,w], then the average number of terminals crossing the partition [1,x] is given by Rent's rule:

$$T_1(x) = A[G(x)]^p$$

$T_1(x)$ actually represents the average number of wires that cross the partition boundary, or the local density at x.

Now let's keep the cutline at x and assume that the chip has n rows instead of only one, as shown in Fig. 14

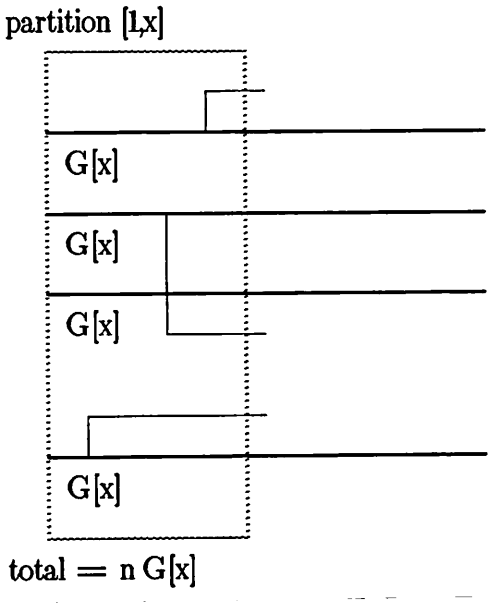


Figure 14: Case for n rows

For each row, the average number of gates in [1,x] is still equal to $G(x)$, but now, if we look at the partition [1,x] over the whole chip, the total number of gates inside that partition (i.e. to the left of cutline x) is the sum of the number of gates in the n [1,x] partitions over the n rows, which is $nG(x)$.

Again assuming that Rent's rule is still applicable, we get the average number of wires crossing the cutline at x, $T_n(x)$, given by

$$T_n(x) = A[nG(x)]^p \tag{23}$$
$$T_n(x) = An^p[G(x)]^p$$
$$T_n(x) = n^p T_1(x)$$

Put in words, (23) states that, as the number of rows in a chip is increased, the local density at an arbitrary point x grows as the number of rows raised to Rent's exponent. Since p is, as described before, less than 1, then the relation implies that the density at x actually grows slower than the number of rows on the chip (or block). What remains now is to find some validation for that relation.

The main source of validation for the above claim comes from [Ueda 85] in which a

layout system, ALPHA, is described for standard cells layout. The system essentially subdivides the chip into functional blocks most of which are laid out using standard cells. Floor planning is then done to achieve good packing on the chip. The system embodies an area estimation routine which uses empirical relations for estimating the routing space prior to placement and routing. These relations were empirically derived from running numerous experiments on several chips and different row configurations. Of importance here is an empirical relation which gives an estimate of the number of tracks,T, needed for routing on a standard cell block, which can be written as

$$T = 0.25b - 9 + b.R^{0.65} \tag{24}$$

Where R is the number of rows in the block and b is a parameter which depends on the number of nets and the total cell width (w in the model of the previous sections). The authors obtained equation (24) by fitting to the curves in Fig. 15 reproduced from [Ueda 85]. The total number of tracks grows as $R^{0.65}$ for a fixed cell and net lists. Values of Rent's exponent p around 0.65 are quite frequently observed in data where the relation has been investigated.

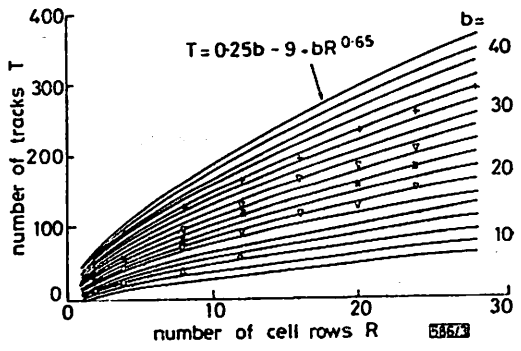


Figure 15: Experimental curves from [Ueda 85]

4.8 Simulation results

Now that we have obtained formulas for the chip area parameters (tracks and feeds), it is important to verify the correctness (and accuracy) of these formulas. A first step in this validation is simulation.

The inputs to the simulation program are the total cell width,w, the number of nets, N, the number of rows, n, and the average wire length. The program then simulates a single row of width w. N simulated wires are born from N randomly selected pins, and have lengths drawn from a geometric distribution with average $p = \frac{1}{average\ wire\ length}$. The single row is then folded into n rows and the local densities are computed, along with the new average wire length. This process is repeated for s samples and the average densities are computed. Two quantities are also computed, *AvgMax*, which is the average value of the maximum density in each sample over all samples, and *MaxAvg*, which is the maximum value of the average densities. More formally, if $d_i(j)$ is the local density at pin slot j in the i^{th} sample, then

$$AvgMax = \frac{1}{s}\sum_{i=1}^{s} max_j\, d_i(j)$$

$$MaxAvg = max_j\left[\frac{1}{s}\sum_{i=1}^{s} d_i(j)\right]$$

The track estimates obtained using Eqs. (13) and (14) are in the last column. Some of the criteria for the selection of the data points are the following :

- The average wire lengths were chosen as "reasonable" estimates, based on some values from the real data described in the next section.
- In each case, the number of samples (100) is thought to be a fair size of the sample space for simulation to be valid.
- In each case, the number of rows was chosen so as to make the chip, according to the estimates, as square as possible (which is the criterion for selecting the number of rows in the real world).

Table shows some results of these simulations.

Samples	w	n	N	1/p	AvgMax	MaxAvg	Est
100	500	8	150	30	60.6	56.7	55.7
100	600	8	200	32	77.0	72.6	69.8
100	800	8	300	35	101.4	95.6	99.0
100	1000	9	400	38	134.3	127.8	122.5
100	1500	10	600	50	190.3	182.6	179.3
100	2000	10	800	65	248.1	238.2	233.0

Table 1: Simulation results

4.8.1 Observations

The following was observed :

- In all cases, *AvgMAx* is within 6% of *MaxAvg*, so, based on this, the approximation of $E\{\max_x d(x)\}$ with $\max_x E\{d(x)\}$ seems to be a valid one.

- The estimations from (13) and (14) is, in most cases, more optimistic than both *AvgMax* and *MaxAvg*. This may be due to the fact that we neglected, in our estimates, the wires that "spill over" the edge at w. The difference between the estimate and the actual value (*MaxAvg*) is, however, within 4%.

4.9 Real data validation

The next logical step in proposing a model for a problem is to validate the model by applying it to "real world" cases and comparing the model predictions to the actual values in these cases.

The only data that was currently available for analysis consisted of a set of six layouts for six different small register-transfer (RT) level implementations of the same data-flow specification. The RT level designs were automatically synthesized using mixed integer-linear programming (MILP) [Hafer 83]. By varying the performance requirements (reflected by varying the cost function), it was possible to obtain six different RT level implementations, reflecting the cost-delay tradeoffs.

In a previous experiment [Granacki 82], Granacki and Parker constructed actual layouts of the six design implementations for two purposes. The first reason was to

compute the areas and the delays of these designs in order to study the effect of the RT-level design tradeoffs. The second purpose was to compare these values with Hafer's predictions. The layouts were done automatically using standard cell design methodology and the **MP2D** layout system [Feller 78].

The overall characteristics of the designs are summarized in Table 2. The number of cells in a design ranged from 108 to 261 cells, having total width from 533 to 949 pin pithes, and placed in 7 to 10 rows. The cell sizes ranged from an inverter cell to a master-slave D flip-flop. The cells are of the "double entry" type, meaning that all the cells have their pins available on both top and bottom.

Chip#	Width w	Rows n	2-point nets N	Avg wire Len 1/p
1	533	7	161	31.0
2	648	7	242	31.7
3	670	8	248	24.0
4	788	9	307	24.2
5	783	9	307	24.9
6	949	10	382	21.0

Table 2: Chips characteristics

The standard structure of an MP2D chip is chown in Fig. 16. The chip technology parameters are shown in Table 3.

4.9.1 Estimating the chip area

In order to evaluate the predictions of the proposed model and compare them to the actual values, we are currently in the process of coding the PLEST standard cell area estimator (which would eventually be a part of the ARREST area estimator). PLEST is currently "tuned" for MP2D chip area estimation. It prompts the user for design parameters, such as total cell width, number of nets, and average wire length. For a given row configuration, the program then uses Equations (13) and (14) to generate estimates of the number of tracks needed for routing. It first sets the derivative of (13) (or (14), depending on the number of rows) to zero and solves for the point of maximum average density using the Newton-Raphson method. The average density at that point is

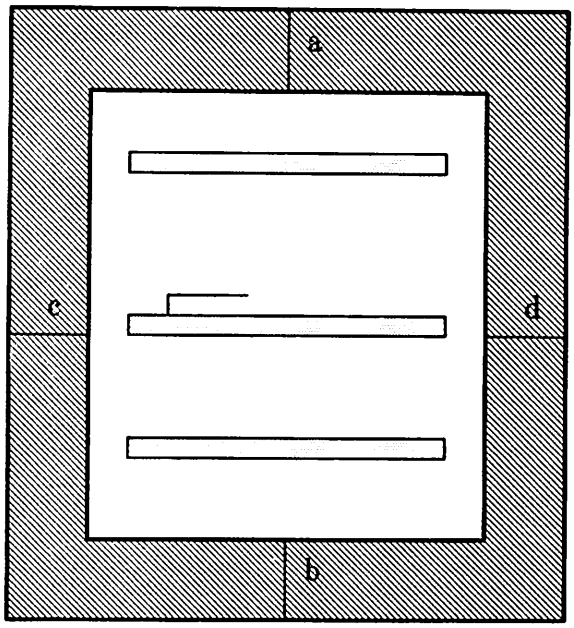


Figure 16: The frame of an MP2D chip

Parameter	Value(mils)
Track height	0.6
Cell height	5.6
Frame height	31.0
Frame width	37.0
Pin pitch	0.9
Feeds width	1.8

Table 3: Technology parameters for MP2D

taken as the number of tracks needed for routing. It then uses Equation (21) to estimate the maximum number of feedthroughs in the cell rows, which occurs at the center rows. The width of the center rows plus the feedthroughs is taken as the width of the standard cell block (i.e. chip width minus frame width). This process is repeated for a range of possible row configurations specified by the user, the idea being for the designer to choose the configuration that best fits his/her purposes. The program also produces plots of different estimates for chip parameters such as number of rows, height, width, aspect ratio, and area.

PLEST was applied to the set of six chip layouts. Table 4 shows the estimation results, along with the real values of the parameters as extracted from the layouts.

# Tracks (Actual)	# Tracks (Est)	Area (Actual)	Area (Est)	%err (Area)
63	54	10593	10924	3.1
65	73	12584	13871	10.2
64	68	12535	13240	5.6
92	82	14375	15431	7.3
89	85	14625	15403	5.3
96	93	16864	17653	4.7

Table 4: Estimation results

We have also studied the variations in area with different row configurations. Fig. 17 shows a typical curve of the area vs. the number of rows for the layouts. The area estimate is, in most cases, minimal at or near the row configurations which were selected by MP2D. Fig. 18 shows some typical area estimate vs aspect ratio curves for a chip. The area estimate seems to be minimal for aspect ratios around 1 (i.e. square chips). Also note that the area estimate does not vary a lot in that region. Furthermore, it is a well known "rule of thumb" in chip design that chips that are as close to square as possible tend to have minimal areas. The same phenomenon was observed in our model.

5 Future research

5.1 A global perspective

So far, we have been concerned with estimating the area of random logic (in the standard cell design style). In practice, a chip may also contain blocks of logic implemented in different design styles. This is because some parts of a digital design cannot be implemented efficiently in standard cells. Random Access Memories (RAMs), for example, would be very wasteful in area and design effort if implemented with standard cells. Also, the designer may choose to implement some parts of a design in a different design style for simplicity and modularity of the design. Such may be the case, for example, in the design of the control part of a circuit using Programmable Logic Arrays (PLAs).

A global area estimator must be able to handle blocks of different design styles. Such

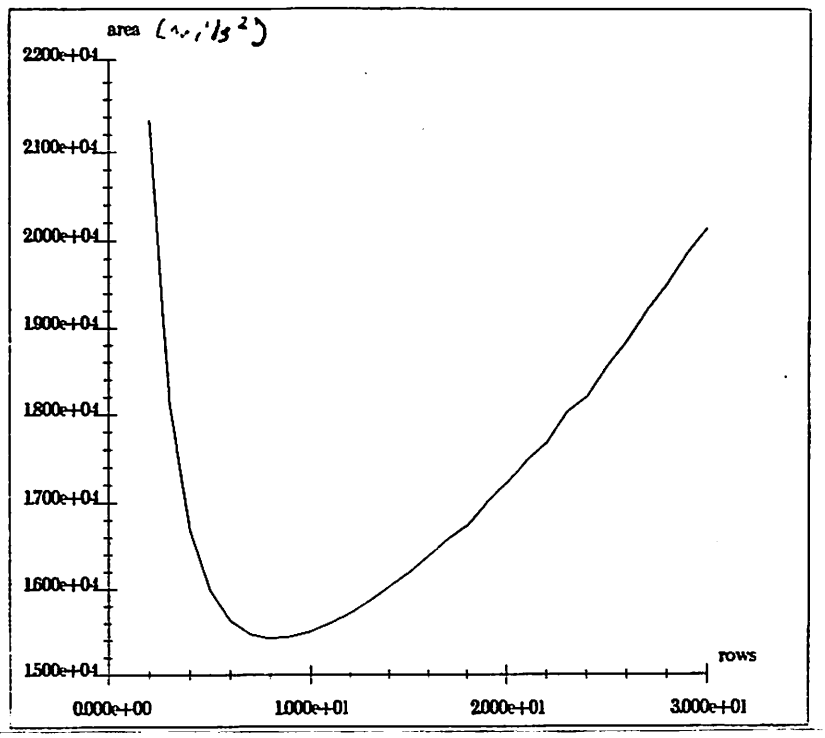


Figure 17: Area vs number of rows

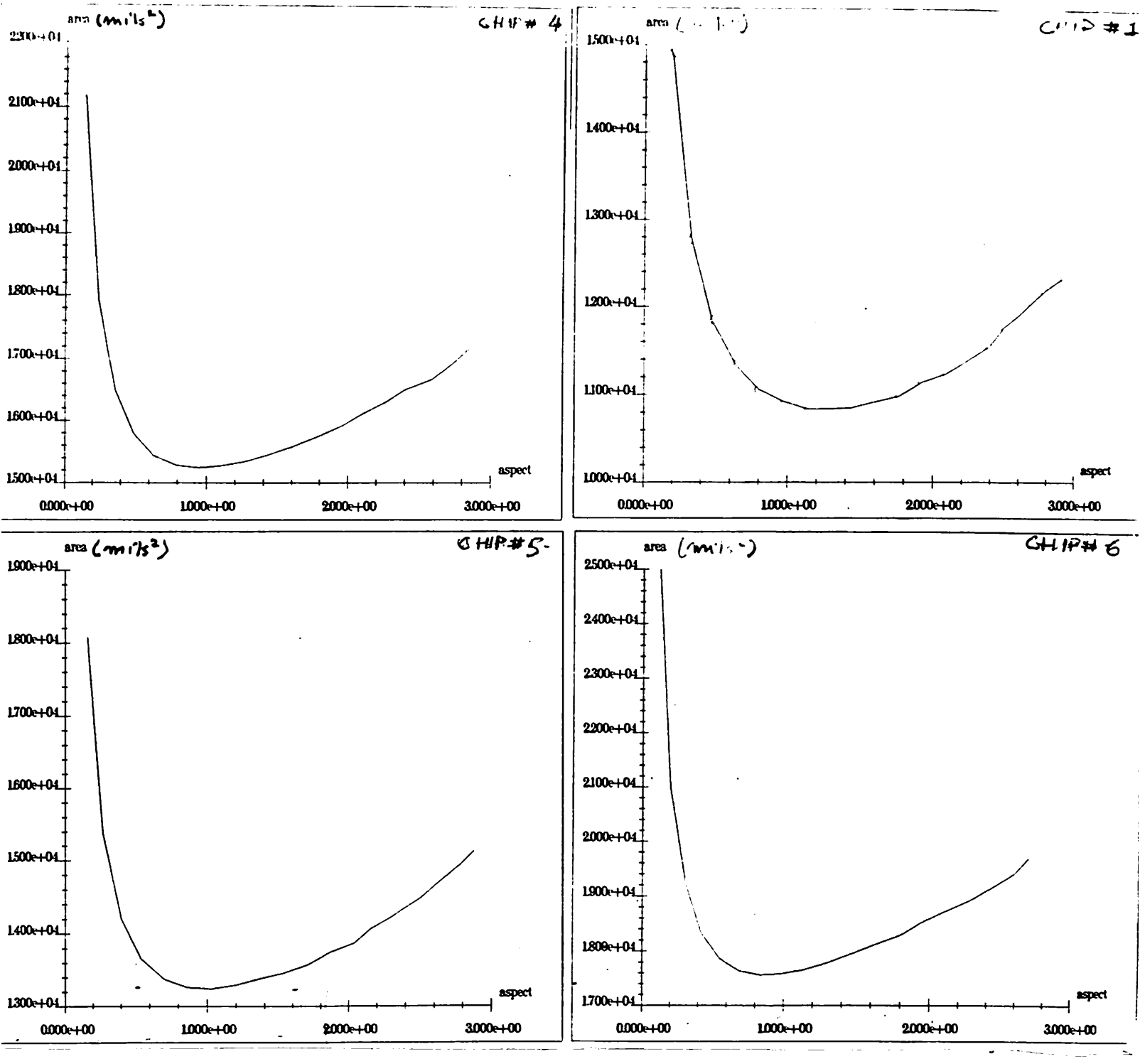


Figure 18: Area vs aspect ratio

an estimator can provide the designer with a set of possible, "reasonable" configurations for each of the blocks that compose a chip, along with estimates of their physical dimensions. The designer can then select an appropriate configuration for each of the blocks so as to satisfy some preset constraints on the individual blocks or on the whole chip.

We classify the design styles for blocks into the following categories :

- Standard cells (for random logic),
- RAM/ROM,
- PLA, and
- regular array structures [Breuer 83]

5.2 Higher level estimation

Up until now, all estimation procedures have dealt with gate (or more appropriately *cell*) level design descriptions. this means that the user must supply as input to the estimator(s) a list of cells and their interconnections as a design description. This assumes that the designer has already synthesized the design up to the gate level, which may actually be an overconstraint for some target usages of the estimator.

Of particular interest is the usage of the estimator as part of a logic synthesis procedure under some constraints on area in particular. During synthesis, the procedure generates a *candidate* design, usually at the intermediate Register-Transfer (RT) structural level. It now needs to check if the design satisfies the area constraints which were imposed on the design during the specification phase. At this point, it is very useful to have an estimator which takes such a design as input and estimates its area. Such a procedure would provide the logic synthesizer with area estimates at the early stages of the design process, thereby avoiding a waste of resources and time in further pursuing the synthesis of a design which may or may not satisfy the area constraints.

6 Summary

In this report, we looked at the problem of predicting the amount of silicon space a certain circuit design will occupy on a chip. We approach the area estimation problem by defining three levels of structure with respect to the physical layout tasks: *cells*, which are primitives implementing simple Boolean functions; *blocks*, composed of sets of cells and their interconnections; and *chips*, composed of sets of blocks and their interconnections.

For block area estimation, we developed a simple probabilistic model for estimating the area of *standard cell* blocks. We start by modelling the linear placement of cells on one single row. The row is then folded into a number of smaller rows so as to satisfy some aspect ratio constraints. Given various design parameters, the average number of horizontal tracks needed for routing the folded placement is then estimated. The same model is also used to estimate the space needed by the vertical wires. The model was applied to a set of six small test chips. The estimated chip area is, for all six chips, within 10% of the measured area.

We will carry out further research to refine and complete the previous model and to develop models for dealing with different layout methodologies, namely RAM/ROM, PLA and array structures. We will also work on higher level area estimation at the intermediate register-transfer level design description.

References

[Feller 78] Feller, A. and Noto, R.
A Speed Oriented, Fully-Automatic Layout Program for Random Logic VLSI Devices.
In *AFIPS Conference Proceedings Vol.47, 1978 National Computer Conference*, pages 303-311. AFIPS, June, 1978.

[Abadir 84] Abadir, M. and Breuer, M.
A Knowledge Based System for Designing Testable VLSI Chips.
Submitted to IEEE Design and Test , 1984.

[Afsarmanesh 84] Afsarmanesh, H. and McLeod, D.
A Framework for Semantic Database Models.
In *Proceedings of NTU Symposium on New Directions for Database Systems*. May, 1984.

[Breuer 83] Melvin A. Breuer and David W. Knapp.
Row-Column Synthesis of VLSI Macrocells.
In *Proceedings of the IEEE International Symposium on Circuits and Systems*. IEEE, May, 1983.

[Breuer 84] Breuer, M. and Zhu, X.
A Knowledge Based System for Selecting a Test Methodology for a PLA.
In *Submitted to 22nd Design Automation Conference*. ACM and IEEE, 1984.

[Deutsch 76] Deutsch, D.N.
A 'dogleg' channel router.
In *Proceedings of the 13th DA workshop*. IEEE, 1976.

[Donath 69] Donath, W.
Hierarchical Structure of Computers.
Technical Report RC 2392, IBM T. J. Watson Res. Cen., Yorktown Heights,N.Y., March, 1969.

[Feuer 82] Feuer, M.
Connectivity of random logic.
IEEE Transactions on computers C-31(1):29-33, January, 1982.

[Granacki 82] Granacki, J.J. and Parker, A.C.
The Effect of Register-Transfer Design Tradeoffs on Chip Area and Performance.
In *Proceedings of the 20th Design Automation Conference*, pages . December, 1982.

[Granacki 84] Granacki, J., Knapp, D. and Parker, A.
The ADAM Advanced Design Automation System: Overview, Planner, and Natural Language Interface.
DISC Report 84-4, EE-Systems Dept. USC, 1984.

[Hafer 83] Hafer, L., and Parker, A.
A Formal Method for the Specification Analysis, and Design of Register-Transfer Level Digital Logic.
IEEE Transactions on Computer-Aided Design CAD-2(1), January, 1983.

[Hashimoto 71] Hashimoto, A. and Stevens, J.
Wire routing by optimizing channel assignment within large apertures.
In *Proceedings 8th DA workshop*, pages 155-169. IEEE, 1971.

[Heller 77] Heller, W.R. at al.
Prediction of Wiring Space Requirements for LSI.
In *Proceedings of the 14th DA Conference*, pages 20-22. IEEE/ACM, June, 1977.

[Hick 83] Hick, P.J. ed.
Semi-custom IC design and VLSI.
IEE, 1983.

[Hightower 69] Hightower, D.W.
A solution to line-routing problems on the continuous plane.
In *Proceedings of the 6th DA workshop*, pages 1-24. IEEE, 1969.

[Kang 83] Kang, S.
Linear ordering and application to placement.
In *proceedings of the 20th DAC*, pages 457-464. IEEE, ACM, 1983.

[Knapp 83a] Knapp, D. ,Granacki, J. and Parker, A. C.
An Expert Synthesis System.
In *Proceedings of the International Conference on Computer Aided Design(ICCAD)*, pages 419-424. ACM-IEEE, September, 1983.

[Knapp 83b] Knapp, D. and Parker, A.
A Data Structure for VLSI Synthesis and Verification.
Technical Report, Digital Integrated Systems Center, Dept. of EE-Systems, University of Southern California, October, 1983.

[Knapp 84a] Knapp, D.
The V Collision Detector.
USC DA Group, available on-line.
1984

[Knapp 84b] Knapp, D.
The Agis Data Structure Editor.
USC DA Group, available on-line.
1984

[Landman 71] Landman, B. and Russo, R.
On a Pin Versus Block Relationship for Partition of Logic Graphs.
IEEE Transactions on Computers C-20:1469, 1971.

[Lee 61] Lee, C.Y.
An algorithm for path connections and its applications.
IRE transactions on electronic computers :346-365, September, 1961.

[Liu 83] Liu, Wen-Tai.
Techniques for Estimation of the Area of Integrated Digital Circuits.
PhD thesis, Dept. of Electrical Engineering, University of Michigan, Ann Arbor, MI, 1983.

[Mattison 72] Mattison, R.L.
A high quality, low cost router for MOS/LSI.
In *Proceedings 9th DA workshop*, pages 94-103. IEEE, 1972.

[Muroga 82] Muroga, S.
VLSI systems design.
Wiley-Interscience, 1982.

[Ngai 83] Ngai, J. Y.
Computational stochastic models for channel routing track demand.
Technical memo, Dept. of Computer Science, California Institute of Technology, 1983.
Tecnical memo : 5094.

[Park 84] Park, N. and Parker, A.
Synthesis of Optimal Clocking Schemes for Digital Systems.
Technical Report DISC/84-1, Dept. of EE-Systems, University of Southern California, May, 1984.

[Park 85] Park, N. and Parker, A.
Synthesis of Optimal Pipeline Clocking Schemes.
Technical Report DISC/85-1, Dept. of EE-Systems, University of Southern California, January, 1985.

[Richard 84] Richard, B. D.
A standard cell initial placement strategy.
In *Proceedings of the 21st DAC*, pages 392-398. IEEE, ACM, June, 1984.

[Rivest 81] Rivest, R.L. et al.
Provably good channel routing algorithms.
In *Proceedings of the CMU conference on systems and computations*, pages 153-159. October, 1981.

[Sastry 85] Sastry, S.
Wireability Analysis of Integrated Circuits.
PhD thesis, USC, January, 1985.

[Schuler 72] Schuler, D.M. and Ulrich, E.G.
Clustering and linear placement.
In *Proceedings of the 9th design automation workshop*, pages 50-56. 1972.

[Supowit 83] Supowit, K.J.
Reducing channel density in standard cell layout.
In *Proceedings of the 20th DAC*, pages 263-269. IEEE, ACM, 1983.

[Sutherland 73] Sutherland, I. E. and Oestreicher, D.
How big should a printed circuit board be?
IEEE Transactions on computers :537-542, May, 1973.

[Syed 81] Zahir A. Syed.
On Routing for Custom Integrated Circuits.
PhD thesis, Dept. of Electrical Engineering, University of Southern California, july, 1981.

[Ueda 85] Ueda, K., Kitazawa, H. and Harada, I.
CHAMP: chip floor plan for hierarchical VLSI layout design.
IEEE Transactions on Computer-Aided Design CAD-4(1):12-22, January, 1985.

[YehC 82] Yeh, C.
The prediction of the requirement of wiring space for VLSI.
PhD thesis, EE dept., The University of Pennsylvania, 1982.