Placement Rent Exponent Calculation Methods, Temporal Behaviour and FPGA Architecture Evaluation

Joachim Pistorius
Altera Corp.
101 Innovation Drive
San Jose, CA
408-544-7604
jpistori@altera.com

Mike Hutton
Altera Corp.
101 Innovation Drive
San Jose, CA
408-544-0253
mhutton@altera.com

ABSTRACT

In the design of FPGA architectures, it is important to understand wiring requirements of placed circuits. Rent's Rule is an empirical metric of connectivity and congestion in a circuit that has applications in the prediction of interconnect usage.

Traditional methods of calculating Rent exponents are based on recursive partitioning, with the exception of some recent work [21], [22] that defines an alternative Rent exponent of a circuit based on a placement-induced partitioning tree.

In this paper we take a different look at the calculation of Rent exponents in placement, contrasting several different methods empirically and outlining the relevant biases in each. We will compare the Rent exponent observed for timing-driven vs. purely congestion-driven placement algorithms, and for different types of benchmark circuits. We also observe the temporal behaviour of Rent exponents through a simulated annealing placement and its correlation to the placement cost function and wirelength. Finally we apply the empirical results to the analysis of the Cyclone FPGA architecture and comment on the routability of the device.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**] Design Aids -- *layout, placement and routing.*

General Terms

Algorithms, Measurement, Experimentation, Design, Theory.

Keywords

SLIP, interconnect prediction, Rent, FPGA architecture.

1. INTRODUCTION

Rent's Rule [15] is a well-known power-law equation found to describe the empirical relationship between the size of a sub-circuit and the number of connectors that sub-circuit has to the outside world. It is usually phrased in the form $T = k B^r$ where B is the size of a typical block or sub-circuit and T is the number of connectors (terminals or pins) on it. The parameter *r* is the Rent exponent, which is commonly viewed as an intrinsic property or characteristic of the individual design, and *t* is often defined as the average number of pins incident to an atomic block.

Early formulations of Rent's Rule do not specify an algorithm for computing the Rent relationship, though the typical assumption is to use the regions defined by a recursive partitioning tree. However even then the result is subject to the algorithm which generates the partitioning tree.

This motivated Hagen et. al. [13] to define the *intrinsic* Rent parameter as that based on the "best" partition tree possible for a given circuit, and they showed that the results do change empirically based on the choice and quality of algorithms.

Since the primary motivation of Rent's Rule is the understanding of placement properties and interconnect usage there have been a number of phrasings with reference to placement. Donath [11] related the Rent parameter of a placed circuit to the Rent calculation induced by applying a recursive partitioning tree to the placement itself. Feuer [12] argued that for a "good" placement an arbitrary region of the placement grid with a center and constant radius could be used to describe regions for the calculation of Rent's Rule. More recently, Verplaetse *et. al.* [21] and Yang *et. al.* [22] defined and computed the *placement* Rent exponent of a circuit by imposing a partitioning hierarchy on the placed circuit with recursive bisection as suggested by Donath's theory. These studies showed that the Rent exponents computed from a placement-induced partitioning tree are larger than those arising from a natural partitioning hierarchy.

Though interesting in general, the primarily use of Rent's Rule arises from extensions that attempt to predict interconnect requirements for a circuit based on its Rent characteristic. (See [7], [17] as representative of a body of literature on this topic). There are two applications of this concept. First, and more common, is to estimate interconnect as a preprocessor to placement, providing useful information to the placement tool. However, a second application is in the design of FPGA architectures ([4], [14], [10], [16]). Some key differences in the second application is that the FPGA architecture problem requires accuracy over speed of calculation, and is more sensitive to worst-case interconnect than to average interconnect – both in contrast to interconnect estimation for CAD tools.

A thesis of this paper is that partitioning-based methods of calculating the Rent exponent are not natural reflections of interconnect because of hidden biases, we prefer other methods more reminiscent of Feuer's theory.

SLIP'03, April 5–6, 2003, Monterey, California, USA.

The structure of the paper is as follows: Section 2 outlines our experimental platform. In Section 3 we propose and contrast alternative methods for calculating the placement Rent exponent. As a natural extension of calculating Rent exponents during placement we show results in Section 4 which illustrate the behaviour of the Rent exponent of individual circuits through the progress of a simulated-annealing based placement tool, and show correlations between the placement cost-function, Rent exponent and bounding-box wirelength. Section 5 briefly discusses wirelength measurement for the Cyclone FPGA architecture. Section 6 applies the empirical results from the previous sections to analyze the routability of Cyclone. We conclude in Section 7.

2. EXPERIMENTAL PLATFORM

The empirical results in this paper are based on a number of industrial VHDL/Verilog circuits targeting the Cyclone FPGA architecture [1], and using Altera's Quartus II V3.0 software for synthesis, placement and routing.

Because the circuits and target architecture contain embedded RAM and arithmetic circuitry (dedicated ripple carry), both of which generate placement constraints, we have some loss of abstraction. However, what we lose in loss of abstraction is compensated for by the volume of data we can generate on a large number of "real" circuits.

Briefly, the Cyclone FPGA consists of logic elements, which pair a 4-input lookup tables (LUTs) with a DFF. These are grouped into clusters called LABs, each of which contains 10 logic elements. For the purposes of this paper, the connectivity in a LAB is a full crossbar. LABs are arranged in an n by m grid that can be thought of as similar to a gate-array with large grid-elements. Prefabricated interconnect exists between the LABs in the form of horizontal (H) and vertical (V) wires, all of length 4, which can be driven at their end-point but tapped at any point. The width of each channel in the grid is 80 tracks (wires), independent of orientation. Logic elements in a given LAB are able to drive 20 distinct H lines and 40 distinct V, and are able to receive signals from 80 H and 160 V lines. Simultaneously, however, a maximum of 26 H and V lines can drive into the same LAB. A device contains roughly $n \cdot m \cdot 10$ logic elements, $n \cdot 80 \cdot m/4$ H lines and $m \cdot 80 \cdot n/4$ V lines; roughly because some number of LABs are replaced by dedicated RAM blocks.

The programming of an FPGA sets SRAM bits controlling muxes (see Figure 1) in the switching network of the device so that a given input to a given logic element receives a specific signal from among the set of potential signals just described. To the right of Figure 1 an input to a LAB is shown, similarly each H, V driver contains programmable muxing to determine which signal drives out on it.

Our CAD flow consists of a greedy and annealing based clustering of logic elements into LABs, followed by a full simulated annealing placement. The placement engine uses LABs. The cost function for placement is a combination of wirelength, congestion (maximum H and V channel width), timing and some other factors. Move generation is a function of a number of factors involving wirelength, congestion and timing.

The number of iterations of simulated annealing is a function of the device and design size which, for this density, can be thought of as "about 75". The number of moves per iteration is slightly superlinear in the design size.

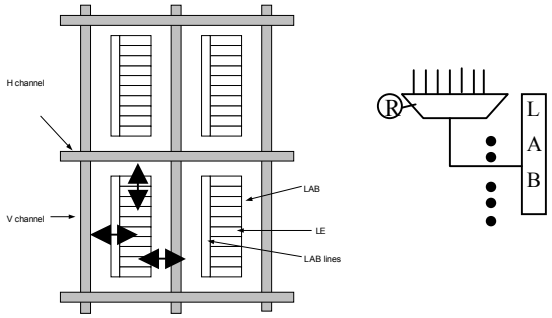


Figure 1. Cyclone architecture and connectivity.

We will use 4 different devices in this study. The C3, C6, C12 and C20 devices have approximately 3000, 6000, 12000 and 20000 4-LUTs respectively. A 4-LUT can be thought of as about 3-4 gates of combinational logic, so the general range is from 10000 to 100000 2-input combinational gates not including flip-flops, memories or arithmetic (carry chains). Approximately 80 industrial circuits will be used, ranging from 15 to 25 circuits per individual device.

Note that the number of tracks per channel is constant, and is not a function of device size. When feasible, such consistency can significantly improve the efficiency of full-custom layout efforts over the multiple members of a device family.

3. RENT CALCULATION AND ANALYSIS

As outlined earlier, our primary goal is to investigate the expected placement Rent exponents for a large set of industrial circuits, using different calculation methods and through time. One of the motivations for this arises from some biases inherent in the partitioning-based calculation.

Define the first calculation method for placement Rent exponents, PART, as based on [21] and [22]: given a fixed placement impose a recursive partitioning tree and sample resulting points.

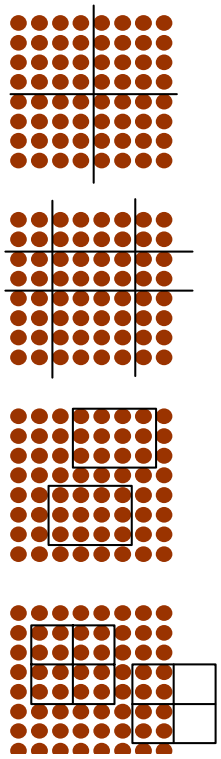
We immediately note one inherent tradeoff in this process. Either one has exponentially more points of smaller size (there are $O(2^i)$ regions of size i) or, as done in both studies, all regions of a given size are averaged. The latter results in very few points for the linear regression, which can affect its statistical reliability.

A second criticism of this method is that it is not "natural" to draw these boundaries at exactly the bisection boundaries. Unless artificially constructed, circuits are neither a natural power of 2 in size, nor square. But also, most partitioners do not attempt to cut circuits exactly in half, so we could be sampling regions that are slightly off of what the partitioner saw as being a cut if the placer is partitioning based also.

This motivates exploring alternative methods to generate measurement regions, inspired by Feuer's theory [12] of placement regions for Rent analysis.

3.1 Defining Rent regions

We define three alternative methods for generating points or regions for measuring Rent exponents, all three of which are based on random sampling of placement regions. See Figure 2.



PART: Regions arise from recursive partitioning.

RND_xy_xy: Randomly choose two (x,y) points to define the region boundary.

RND_xy_len: Randomly choose x and y lengths, then one (x,y) anchor point.

RND_xy_rad: Randomly choose (x,y) coordinate of center, and radius (with truncation).

Figure 2. Alternate methods for region calculation.

RND_xy_xy: Choose two random points in the grid, and define the sampling region as the set of logic elements bounded by the two points.

RND_xy_len: Choose a random region size (x and y lengths). Then, from the set of feasible (x,y) points for the lower-left corner of the region of these dimensions, choose its location.

RND_xy_rad: Choose a random (x,y) location of a region "center", and a random radius. In this case, we allow regions to overlap area outside the chip by truncation.

These definitions are somewhat arbitrary; the point is to represent alternative reasonable approaches to generating regions, and examine the behaviour of each. For the first, size is a by-product of location, the second the size of the box is paramount, and the third the location of the box is paramount with at somewhat variable size.

Note that all of these approaches, and the definition of our architecture mean that we are using rectangular regions in general. Though the mathematical modeling of this is discussed in Dambre *et. al.* [8], we haven't made any attempts to adjust the empirical calculation accordingly. One observation by Dambre was that rectangular boxes with less-square aspect ratios generated regions with more than typical pin to logic ratios. This was also observed by Betz [2] in the study of non-square FPGA architectures. To avoid the Region II and Region III effects on Rent behaviour [19] we resampled any regions found to contain more than 1/3 of the chip, and to make reasonable aspect ratio boxes we resampled any regions with aspect ratio greater than 4. The Region III issue was largely solved by using a LAB (10 logic elements) as the smallest measurable region for Rent calculation.

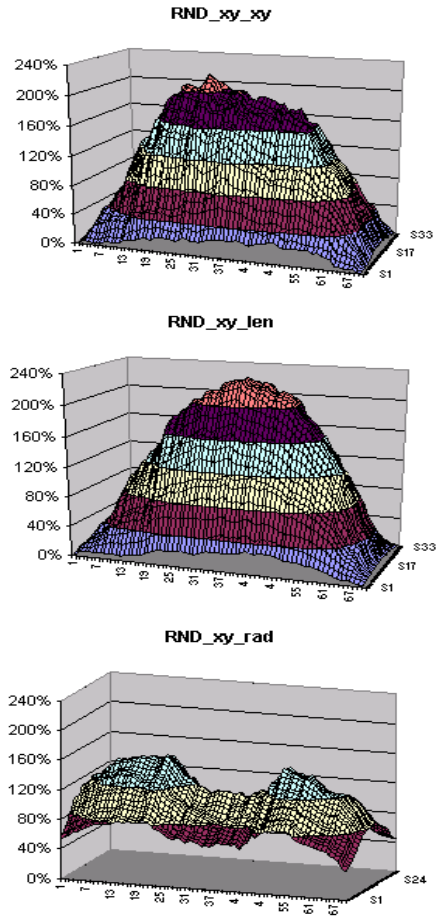


Figure 3. Sampling coverage per selection method.

Figure 3 illustrates sampling bias. The PART method of choosing regions is "fair" – every cell contributes equally to the Rent calculation. This is not true for the other methods: RND_xy_xy and RND_xy_len both have a significant bias against the contribution of edge logic elements to the Rent calculation. RND_xy_rad is much more fair, but not completely fair. In these figures the x and y axes represent LAB locations on the chip, and the z axis the sampling incidence relative to the average (100%).

RND_xy_rad, though obviously the best for fairness, has an interesting distribution, which arises from a combination of the part having more columns than rows, and that this method allows truncation combined with the filtering, for all methods, of regions more than 1/3 the size of the chip. If the latter is removed, the distribution unimodal is as the previous two, but much flatter.

Figure 4 illustrates size bias on a linear scale. PART is inherently biased towards very small regions (exponentially so). RND_xy_xy is less, but obviously, biased. RND_xy_len is inherently unbiased by construction (and could be made moreso by forcing square regions). RND_xy_rad is more complicated to consider due to truncation, but is reasonable.

This is better viewed in Figure 5, which shows log-log Rent plots for one circuit (chosen to be a typical representative). All methods use the same number of Rent regions for the calculation. PART and RND_xy_rad are fair samplings of points (not visible

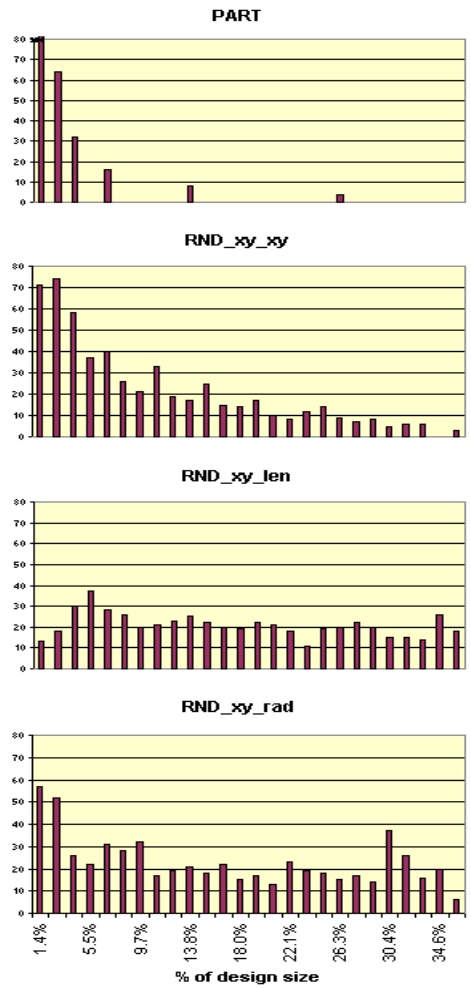


Figure 4. Region sizes per selection method.

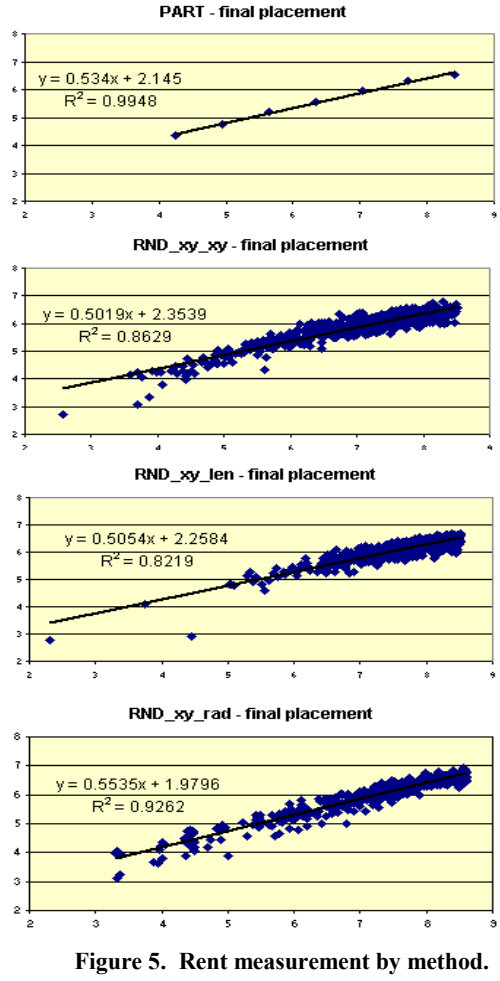


Figure 5. Rent measurement by method.

in these graphs). On both the linear and logarithmic scales, it is clear that RND_xy_rad yields the best sampling. Hidden in the PART calculation is the averaging of many points at the smaller end, which would otherwise show its bias to the smaller regions. Note that a visually even distribution on the log-plot would represent an over-sampling of smaller regions, not a truly even distribution.

Figure 6 shows the aggregate results for each method, sorted by RND_xy_rad. There is surprisingly little correlation between the different methods, which we would have expected from most of the general theory to be similar. Figure 7 shows a scatter-plot between RND_xy_rad and the PART method to differentiate the two. Recalling that Yang *et. al.* [22] found that the placement Rent exponent was typically larger than that arising from a pure partitioning tree, we note that the RND_xy_rad method, which we believe more natural and statistically accurate for placement, would appear to further raise the Rent exponent.

Looking at absolute Rent exponents we observe a range from 0.45 to 0.70. Given the large sampling size and relative reliability coming from many large and "real" circuits, it would thus appear that 0.5 to 0.65 would be a more reasonable range for generating Rent exponents when using random circuit generators rather than values up to 0.75 as sometimes seen. Chiba [5] found a range of 0.5 to 0.6 in his survey. Many other early papers quoted regions such as "0.6 to 0.75".

We also see no significant correlation between the device size and the extracted Rent parameter. For designs compiled on the C3, C6, C12 and C20 the average Rent exponent was 0.57, 0.61, 0.61, 0.58 respectively with a relatively common distribution.

3.2 Circuit Characterization

Many early studies made hypotheses that Rent parameters were related to the functionality or amount of parallelism present in circuits. Thus it is interesting to compare Rent parameters against the type of circuit.

We manually divided the circuits in the benchmark set into four categories. This division is only approximate, because such classifications are clearly not black-and-white in nature. Networking designs, arising from switching applications, are generally expected to have multiplexors, packet processing and other switching components. Image processing (which includes audio processing) involves transformation and modification of a bitstream. Digital signal processing typically contains many multipliers and adders, and is heavily datapath oriented. The control logic category includes designs such as processors, controllers and arbitrators.

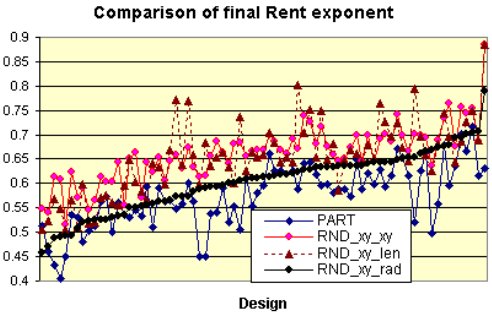


Figure 6. Final Rent exponents, by method.

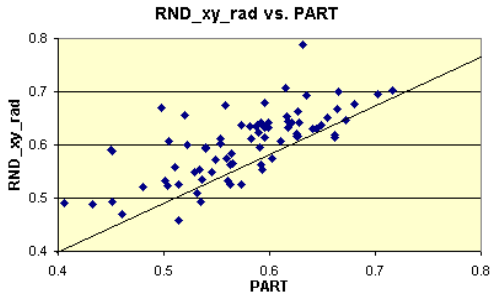


Figure 7. RND_xy_rad gives larger Rent than PART.

Figure 8 shows the results. It would appear that there is no real correlation for this design-set by classification. The averages for each category of design lie between 0.58 and 0.60. The most likely reason for this is that designs of this size are really composed of many smaller functional blocks. Any trends that exist for very specific circuits would thus be averaged out over the course of a system-level design.

One might expect utilization (fullness) of the device to be relevant as well. However, there is no correlation here either (Figure 9). Note that this is partly by construction – the clustering algorithm generally tries to pack LABs fully for timing reasons and, given that, there is no particular incentive for an 80% full design to behave differently than a sub-placement of a 100% full design.

3.3 Timing-Driven Compilation

The default flow for our software is to perform timing-driven compilation (TDC) to minimize the worst-case combinational delay between registers. One would expect that this would adversely affect wirelength, and this is clear empirically – Figure 10 shows the average change in wirelength, per circuit, when timing-driven compilation is on vs. off. Wirelength here is measured as post-routing H and V usage of all nets.

There is an expected increase in Rent exponent (Figure 11). On average, the Rent exponent reported in previous data would decrease by 4.3% were we to use the non-TDC compilations.

It is not true, however, that the two are directly correlated. Figure 12 shows the two charts superimposed and sorted by wirelength. In general, though TDC increases both Rent exponents and wirelength on average, this varies by circuit. (In Figures 6,8,10-12 each bar represents the result for one test circuit).

4. TEMPORAL RENT BEHAVIOUR

There is a generally held belief that the Rent parameter can also be viewed as a metric of the quality of different algorithms

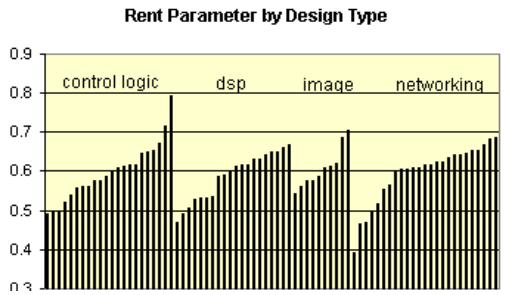


Figure 8. Rent parameters by design type.

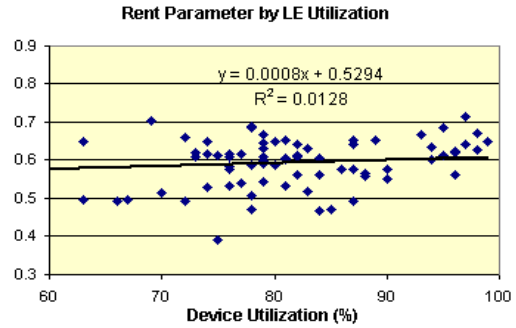


Figure 9. Rent parameter by utilization (% full).

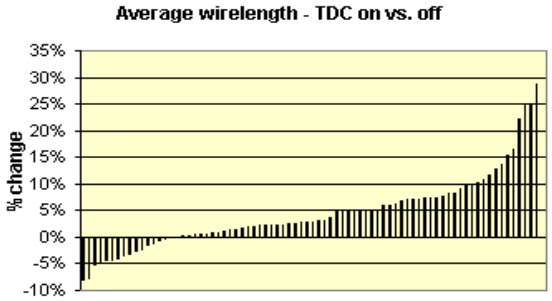


Figure 10. Average routed wirelength.

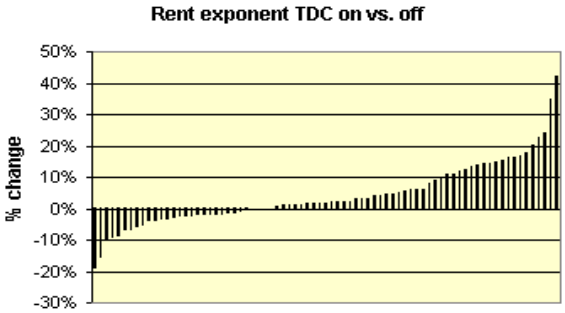


Figure 11. Rent exponent change, TDC on vs. off.

applied to the same circuit. Verpletse et. al. [21] looked at the change in the Rent exponent over the course of a refined placement with simulated annealing, and found a decrease in Rent exponent from 0.967 to 0.635 during the placement of one circuit

Figures 13 and 14 show the behaviour of the extracted Rent exponent and bounding-box wirelength, which is a primary component of the placement cost function, for two circuits. We note a surprisingly strong linear correlation between the two

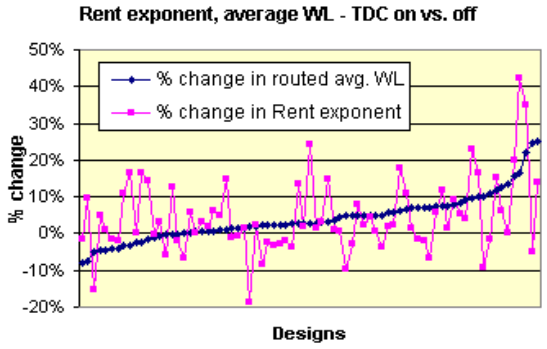


Figure 12. Rent exponent and wirelength.

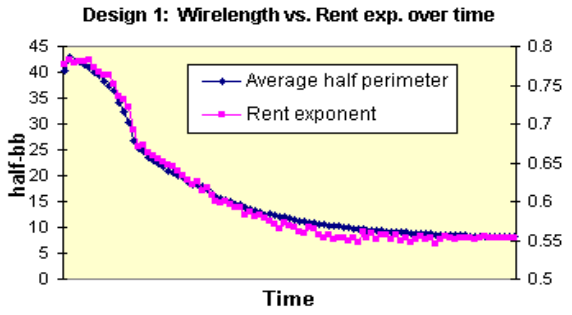


Figure 13. Wirelength and placement Rent exponent.

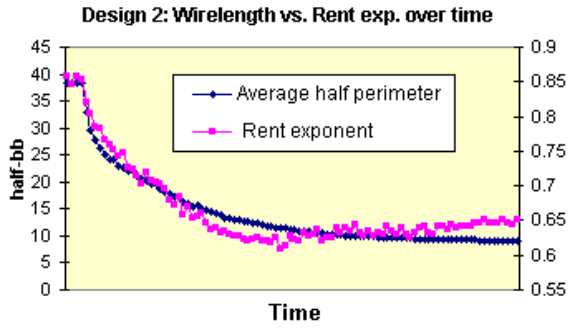


Figure 14. Wirelength and placement Rent exponent.

curves. One would not expect Rent exponents and wirelength to correlate linearly, but it does appear that from the point of view of placement quality measurement on the same circuit (thus with all other netlist properties held constant) the relationship is strong.

Note the increased Rent exponent towards the last half of the anneal. This represents a behaviour that can be seen as unique to FPGAs: once the maximal wirelength requirements are met for an individual design, wires are "free" and the placer is free to concentrate more heavily on timing if such moves are found and seen as beneficial.

5. WIRELENGTH ESTIMATION

Valid wirelength measurement requires some additional thought for our experimental setup. Because of the architecture of Cyclone we have largely ignored all the nets that lie inside a LAB for Rent analysis. This is reasonable for Rent parameter extraction alone, but to estimate wirelength we are ignoring essentially half the nets in the circuit by measuring only the global wires outside of a LAB. This is clear from a naïve plotting of wirelength as estimated by the Feuer model [12] (not shown) in which there is essentially no correlation between estimated and actual wirelength. The Feuer model computes expected average wirelength as purely a function of the number of cells N in the netlist and the Rent parameter.

We adjust Feuer as follows: for the measured wirelength (total H + V lines after routing), we know empirically that about ½ of nets are local to a LAB, while ½ of all nets are global. Approximating a 10 LE LAB to consume a roughly 3x3 grid of physical space, we then re-state the measured wirelength as 3*measured + 3*N/2 – the first component simply multiplies the global wirelength by the grid-size, and the second adds in the unmeasured local nets with an average length of 3. The latter constant is simply a guess. We could use a half-bounding box of 6, but in reality the LAB is more dominated by the constant area and delay in muxing the connections through LAB lines than in the distance traveled, and there is no geometric placement in the LAB, so 3 "feels better".

For the estimated wirelength, we have the following from Feuer :

$$\alpha = 2 - 2*r$$

$$\overline{R} = \sqrt{2} * \frac{(2-\alpha)(5-\alpha)}{(3-\alpha)(4-\alpha)} * L^{(r-0.5)}$$

Note that the wirelength calculation is based on LABs rather than LEs, so the computation uses the number of LABs in the circuit.

To compute the expected total wirelength, we then multiply $\overline{R}$ by N/2 (for number of global-driving nets) and do the same transformation made to the measured wirelength to normalize to the grid-size and add the local nets.

The results are shown in Figures 15 and 16, for the Rent exponents calculated by PART and RND_xy_rad respectively.

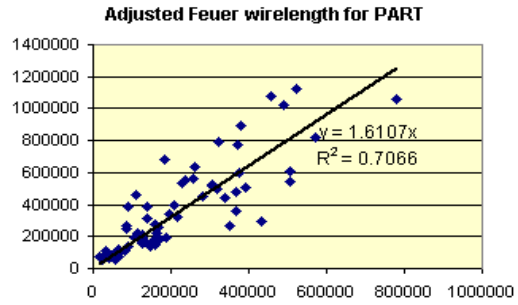


Figure 15. Adjusted Feuer wirelength prediction for PART method of computing Rent.

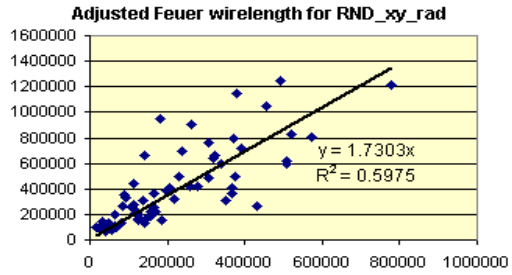


Figure 16. Adjusted Feuer wirelength prediction for RND_xy_rad method of computing Rent.

We note that, as generally expected, the wirelength estimates are close to 2X the actual.

These wirelength predictions should not be taken too seriously. A useful follow-up to this study would be to formally define and measure the true wirelength more accurately, and to modify the mathematical justifications of the Feuer model to match the architecture. A more aggressive goal would be to modify the more involved Davis [9] theory to this placement architecture. However, it is nonetheless interesting that we can get a reasonably good measurement (off by the expected constant) with this naïve modification, despite the loss of abstraction resulting from the FPGA architecture, timing-driven compilation effects, carry chains, placement restrictions and memories. This is also a verification of Feuer's thesis that an arbitrary region continues to satisfy the properties of Rent's Rule normally associated with a partitioning tree.

It is disappointing that the results are more consistent, as measured by R^2 (goodness of fit coefficient) for the regressions, for the PART method. We continue to believe that more natural models of measuring placement Rent exponents should generate better results, but the bottom-line data does not support that belief. Not shown are the RND_xy_xy and RND_xy_len methods, both of which performed worse than RND_xy_rad as measured by the regression coefficient R^2.

6. FPGA ARCHITECTURE APPLICATION

To use the empirical analysis in this study, we will apply basic Rent analysis to measure the routability of the Cyclone architecture itself.

Even though the architectural model is complicated by the heterogeneous and hierarchical nature of the interconnect imposed by LABs, it is not hard to estimate a Rent exponent for the device.

As previously mentioned, the architecture consists of row and column interconnect with a channel width of 80 in both directions, and all wires of length 4. Given the set of regions with their x*y size, we compute the possible number of device terminals for the region as the number of wires entering or leaving the region, unless the size of the region is small enough such that the LAB connectivity (26 inputs to a LAB + one output per LE) is smaller. The latter is relatively important, because even though 80 H and 80 V wires pass over a given LAB of size 10 LEs, only these 26+10 terminals can possibly enter and leave the LAB.

Overall, this gives an empirical Rent exponent for the architecture of 0.7256. It is worth noting that this is actually the average of two constant Rent exponents, as can be seen by the plot in Figure 17, and these can be computed exactly. The lower portion of the Rent curve corresponds to the LAB-limited regions, which have a Rent exponent of exactly 1.0. For this area of the curve a doubling of the block-size also doubles the number of terminals. The upper portion corresponds to the interconnect-limited regions with a Rent parameter of exactly 0.5. This Rent exponent arises from the fact that the channel width is a constant, so additions to block-size correspond to additions in terminals directly proportional to the perimeter of the block.

Shown are the Rent plots for three individual circuits against the Rent plot for the FPGA itself. The three designs are chosen as typical, stressed and very stressed in terms of routing resources. One can see here one goal of FPGA architecture design, which is

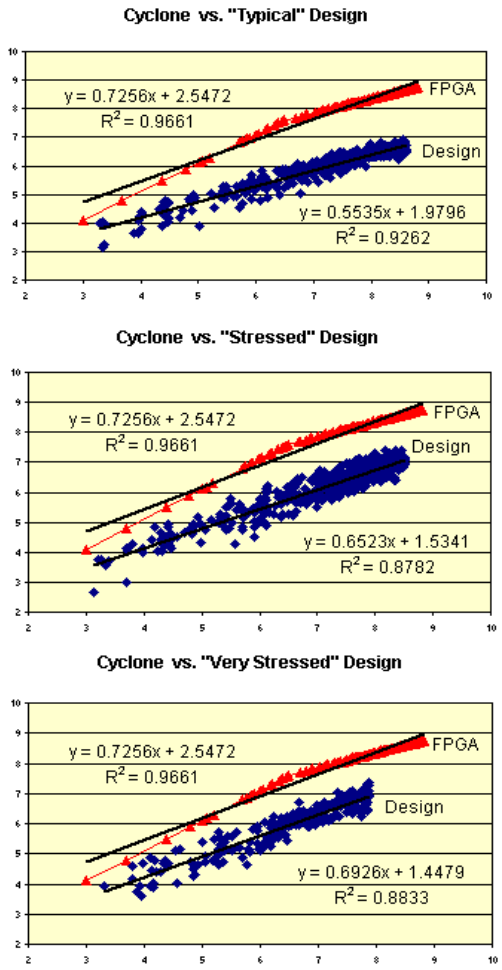


Figure 17. Placement Rent plots vs. maximum wiring availability of the Cyclone architecture.

that the worst case designs are able to place&route comfortably in the device, even though the average design might have more than sufficient resources.

Since the average Rent exponent of all designs is 0.60 and the standard deviation 0.063, we note that the architecture of Cyclone has a Rent exponent of approximately average plus two standard deviations, which can be seen as a proxy for fitting "about 95% or more of all designs" by construction.

Though nearly perfect fitting is generally the goal of all commercial FPGA production, it is important to note that it is not necessarily the most efficient possible goal. DeHon, among others, has argued [10] that one would be better to design architectures which fail to fit some proportion of designs in order to use fewer programmable wires overall.

In a previous study [14], similar calculations for the Apex FPGA architecture showed average circuit Rent exponents between 0.5 and 0.8, and a Rent exponent for Apex itself of about 0.78. We believe that the ability to decrease the Rent exponent of the device in this time reflects an improvement both in CAD tools to give more efficient placements and also ever-improving technology in the design of the architectures itself, which allow for the existing wires to be more accessible to the programmable routing fabric.

The latter point is actually a key issue not touched upon in this paper, which is that the appropriate design of the programmable switching architecture is as important to routability as the raw wire-counts. It is well-known in the industry that minor mistakes in the detailed switching architecture within the prototype software can dramatically affect the routability of the device independently of the available interconnect.

This aspect of FPGA architecture is not modeled at all by the Rent theory, though in some ways the measurement of Rent exponents of designs vs. the expected can be seen as a metric of the switching network efficiency as it is similarly seen as a metric of the CAD tool efficiency.

7. CONCLUSIONS

In this paper we have made a number of contributions. We gave empirical measurements for a commercial FPGA architecture and tools on a large number of industrial FPGA circuits against the Rent-based theory that underlies a body of research in wirelength estimation. We argued that the traditional methods of estimating Rent exponents via partitioning, though satisfying the goal of quick estimation for placement preprocessing, are biased and less natural than sampling regions of the final placement as suggested by Feuer. In analyzing alternative methods, we also showed empirical justification of Feuer's theory of contiguous regions of placement following Rent's Rule.

We showed that, though different methods for generating Rent exponents perform similarly on average they don't correlate well on individual designs. We found that for a fixed netlist in placement there is a surprisingly strong linear correlation between the Rent exponent extracted from the placement and the bounding-box estimated wirelength, and thus the Rent exponent linearly measures placement quality on a given design. We presented a minor modification to Feuer's wirelength estimation in order to apply the theory to the Cyclone architecture.

Finally, we measured the Rent exponent of the Cyclone FPGA architecture, and contrasted the placement Rent behaviour of place&route on several designs against the designed-in Rent parameter of the device.

Further work arising from this analysis would be to work on a mathematically sound method for wirelength calculation on a heterogeneous FPGA architecture, for example to express the site occupation probabilities of the Davis model, which shows accurate wirelength estimation for general grids.

REFERENCES

[1] Altera Corp. "Cyclone Family Data-Sheet". Available at www.altera.com.

[2] V. Betz, "Directional Bias and Non-Uniformity in FPGA Global Routing Architectures", *IEEE TVLSI*, pp. 445-456, Sept. 1998.

[3] V. Betz, J Rose and A. Marquardt. *Architecture and CAD for Deep-Submicron FPGAs*. Kluwer, 1999.

[4] P. Chan, M. Schlag and J. Zien, "On Routability Prediction for Field-Programmable gate Arrays." In. *Proc. ACM/IEEE Design Automation Conference (DAC)*. pp. 326-330, 1993.

[5] T. Chiba, "Impact of the LSI on high-speed computer packaging," *IEEE Trans. Comput*., Vol C-27, pp. 319-325, 1978.

[6] P. Christie, "Rent Exponent Prediction Methods." *IEEE Trans. VLSI* Vol 8(6), pp. 639-688, 2000.

[7] P. Christie and D. Stroobandt, "The Interpretation and Application of Rent's Rule". *IEEE Trans. VLSI.* Vol 8(6), pp. 639-648, 2000.

[8] J. Dambre, P. Verplaetse, D. Stroobandt and J. Van Campenhout, "Getting more out of Donath's hierarchical model for interconnect prediction." In *Proc. ACM/SIGDA Int'l Workshop on System-Level Interconnect Prediction (SLIP)*. pp 9-16, 2002.

[9] J. Davis, V. De and J. Meindl, "A stochastic wire-length distribution for gigascale integration (GSI): Part II: Application to clock frequency, power dissipation and chip-size estimation," *IEEE Trans. Electron Devices*, Vol. 45. pp 590-597. Mar, 1998.

[10] A. DeHon. "Balancing Interconnect and Computation in a Reconfigurable Computing Array (or, why you don't really want 100% LUT utilization)", In *Proc. ACM/IEEE Symposium on FPGAs (FPGA)*. pp. 69-77, 1999.

[11] W.E. Donath, "Placement and Average Interconnection Lengths of Computer Logic." *IEEE Trans. Circuits and Systems*, Vol 26(4), pp. 272-277, 1979.

[12] M. Feuer, "Connectivity of Random Logic", *IEEE Trans. On Computers*, Vol C-31, pp. 29-33, 1982.

[13] L. Hagen, A.B. Kahng, F.J. Kurdahi and C. Ramachandran, "On the Intrinsic Rent Parameter and Spectra-Based Partitioning Methodologies." *IEEE Trans. CAD.* Vol 13:1, pp. 27-37, 1994.

[14] M. Hutton. "Interconnect Prediction for Programmable Logic Devices." In *Proc. ACM/SIGDA Int'l Workshop on System-Level Interconnect Prediction (SLIP)*. pp. 125-134, 2001.

[15] B Landman and R. Russo, "On a Pin vs. Block Relationship for Partitions of Logic Graphs." *IEEE Trans. On Computers*, Vol. C-20. pp 1469-1479, 1971.

[16] A. Singh and M. Marek-Sadowska. "FPGA Interconnect Planning". In *Proc. ACM/SIGDA Int'l Workshop on System-Level Interconnect Prediction (SLIP)*. pp. 23-30, 2002.

[17] D. Stroobandt, *A Priori Wire Length Estimates for Digital Design*, Kluwer, 2001.

[18] D. Stroobandt, "A Priori System-Level Interconnect Prediction: Rent's Rule and Wire Length Distribution Models", In *Proc. ACM/SIGDA ACM/SIGDA Int'l Workshop on System-Level Interconnect Prediction(SLIP)*. pp. 3-21, 2001.

[19] D. Stroobandt, "On an Efficient Method for Estimating the Interconnection Complexity of Designs and on the Existence of Region III in Rent's Rule", In *Proc. Great Lakes Symposium on VLIS (GVLSI)*. pp. 136-141, 1999.

[20] P. Verplaetse, J. Van Campenhout and D. Stroobandt. "On Synthetic Benchmark Generation Methods." In *Proc. Intl. Symp. On Circuits and Systems (ISCAS)* pp. IV 213-216, May 2000.

[21] P. Verplaetse, J. Dambre, D. Stroobandt and J.Van Campenhout, "On Partitioning vs. Placement Rent Properties." In *Proc. ACM/SIGDA Int'l Workshop on System-Level Interconnect Prediction (SLIP)*. pp 33-40, 2001.

[22] X. Yang, E. Bozorogzadeh and M. Sarrafzadeh, "Wirelength Estimation Based on Rent Exponents of Partitioning and Placement." In *Proc. ACM/SIGDA Int'l Workshop on System-Level Interconnect Prediction (SLIP)*. pp 25-32, 2001.

[23] P. Zarkesh-Ha, J. Davis and J. Meindl, "Prediction of Net-Length Distribution for Global Interconnects in a Heterogeneous System-on-a-Chip." *IEEE Trans. on VLSI*, Vol. 8, No. 6. pp 649-659, Dec 2002.