Expert System Perimeter Block Placement Floorplanning

Richard Auletta
Cadence Design Systems

Abstract

With the dramatic increase in the size and block count of systems on a chip (SOC) over their application specific integrated circuit (ASIC) counterparts, engineers now need assistance beyond the clerical optimization tasks of placement and routing, they need assistance in applying their own expert abilities to design planning. This paper presents an investigation in applying expert systems to the automated floorplanning of systems on a chip. The investigation presents some background on expert systems, and then the implementation and results of an expert system based edge placer for perimeter placement of floorplan hard blocks.

1. Introduction

The capture of designer knowledge into rule based expert systems in the field of electronic design automation, and ASIC and SOC design, has seen little application or success. We believe the reason is that in the past tasks like floorplanning a design could be easily handled by an individual in a reasonable time. This is no longer true with large SOC designs consisting of 10's of partitions and 100's of blocks. Such systems require expert assistance to allow their design exploration and implementation in a reasonable amount of time by a reasonable number of engineers.

2. The CAD Dilemma

The fundamental CAD dilemma is that electronic design automation (EDA) algorithms are primarily focused on optimization, and cannot express general design constraints, or apply the expert knowledge of SOC and ASIC engineers. Automated floorplanning [10] typically globally optimizes an abstract objective function, such as minimizing total wire-length, to arrive at a proposed solution. Unfortunately these solutions are commonly infeasible as they do not take into account other objectives, such as maintaining a contiguous placement region or optimal block associations.

A macro placement that creates as regular a core region as possible is a meta-objective, and can result in a floorplan with better characteristics for placement, routing, power distribution, clocks insertion, and timing closure. While an algorithm might discover such a solution by chance, typical objective functions do not and cannot express such meta-objectives either directly or indirectly.

2.1 Expert Systems

Expert systems (ES) and artificial intelligence (AI) broadly split machine reasoning into two categories; rule based and reasoning based. The latter implies a system that can reason using formal logic about a domain of knowledge. The former, rule based expert systems [1,2], are based on information derived from a human expert, and the rules consist of a pattern match and an associated action. It is this latter type of expert system, that uses an inference engine, that we have applied to the block floorplanning problem.

Other approaches to applying artificial intelligence or biological reasoning to floorplanning include genetic algorithms, neural networks [6], fuzzy logic [7], and of course knowledge based approaches [4, 5]. Our results differ from the above in several significant ways. Our approach is a classic expert system with a user (the design that needs floorplanning) that interacts with an inference engine with a tightly coupled knowledge base. Our approach is deterministic, correct by construction, repeatable, explainable, and focused on a specific task and style of design, resulting in the ability to analyze the results of our expert system and hence intelligently improve its inference engine. The expert placer never resorts to blind optimization or search, such as genetic algorithms or neural networks, and we avoid the pitfalls of backtracking, self-directed learning, or using optimization as a starting point for expert refinement.

2.2 The Expert System Dilemma

Expert Systems have never enjoyed much popularity in electronic CAD, irrespective of what was claimed as early as 1990 [5], probably because it has alway been believed there were too many permutations and combinations to allow any reasonable set of rules to even begin to capture the expert's knowledge. We know of no commercial EDA tools that claim an expert system capability nor any research tool in common use.

And while the complexity of engineering tasks is in general true, for specific tasks such as floorplanning hard blocks such as memories, PLLs, and processor cores, we believe that the rule set is bounded for any particular design style, and that any initial solution space is a set of reasonable solutions, not the search for a single globally optimal solution.

1530-1591/04 $20.00 (c) 2004 IEEE

Recently, Wolfram [8] has again popularized the notion that cellular automata with simple rules can describe complex systems and behaviors. We note a correspondence with expert driven EDA tools, that a relatively simple set of rules when focused to solve a specific task, can give rise to a solution what appears to be driven by human expertise, creativity, and intuition.

The discussion of expert systems in regard to EDA always generates strong opinions, but in fact, many EDA tools attempt to automate and improve upon what experts do. A good example is "the theory of logical effort" [9] that is used as the basis for commercial EDA tools from Magma Design Automation. In this case, a method was developed by experts to allow non-experts to determine the electrical and logical effort associated with a circuit path. In addition, EDA tools for RTL and design quality analysis are in fact expert systems whose actions from an inference match are typically limited to reporting design flaws as opposed to their correction.

2.3 Expert System as Assistant

Nothing in this manuscript should be taken to imply that expert systems will replace or supplant current EDA algorithms. Current EDA algorithms are and will remain essential for optimization and handling complex clerical problems. While it might be possible to conceive of an AI system that could optimize a logical path for area or timing by abstract reasoning, we see little to recommend it as a practical solution.

Even with their existing weakness in understanding a design, the speed and accuracy of expert systems make them more efficient at both collecting the information a human expert uses and sorting through that information looking for the same relationships the human expert seeks. The fundamental problems in building an expert system is understanding what rules the human uses, the nuances they consider in making their design decisions, and most difficult of all, replicating human perception.

3. Our Expert System Floorplanners

We have so far only examined expert systems that are purely rule based and do not make use of classical optimization algorithms. We do this to better understand to what extent expert systems can perform in comparison to existing classical optimization algorithms. Systems that integrate both expert and algorithmic optimization would surely produce the best possible results.

We have investigated two floorplanners. The first is an edge placer that packs hard blocks around the perimeter of the design. Placing hard blocks around the edge of a die or a hierarchical partition is a popular and common floorplanning technique for both flat ASICs and hierarchical SOCs, and is intended to create a regular region of cell placement free of signal and power routing obstructions. The popularity of the edge placement style of floorplanning with current in-design large high-performance chipsets was our inspiration. We have also examined the rules for a grid placer that places macros across the area the design.

3.1 Implementation

The edge placer expert system to place macros around the perimeter of the placement region has been implemented in under 3000 lines of TCL and is illustrated in Figure 1. The implementation environment is Cadence Design Systems' First Encounter (FE) using the First Encounter TCL API. The FE TCL API is primarily used to obtain initial physical information (pin placement, block size), logical information (the hierarchical paths of the blocks) and for design analysis. The FE TCL API and LEF data do have shortcomings, especially in terms of obtaining information about block pins (clock, reset, scan, control) and block functionality (memory, PLL, analog.) The implementation in First Encounter allows us to work with current high-performance industrial designs, in a complete SOC design environment, and take the designs through an entire design flow including full detailed placement, routing, power, and design analysis.

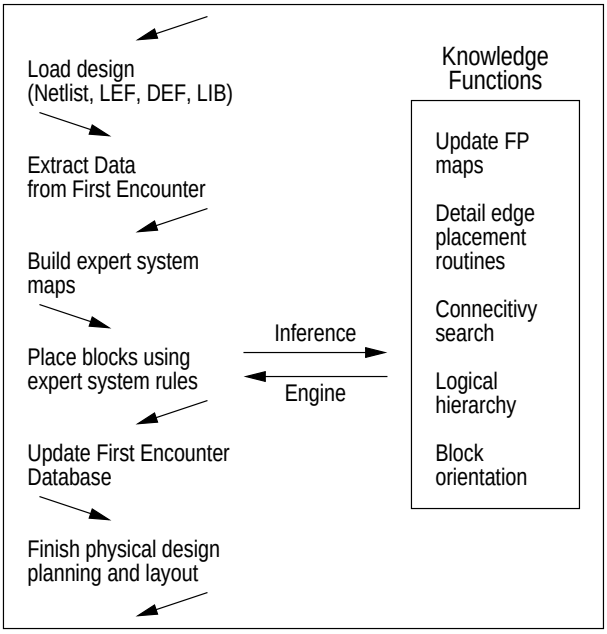

Figure 1: Edge Place Flow

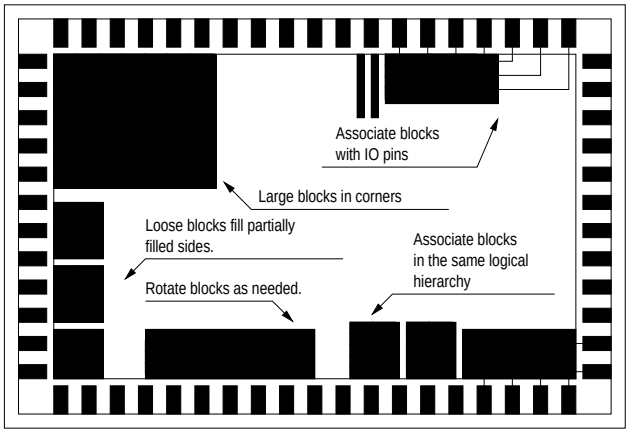

Figure 2: Example Expert Rules

3.2 Edge Placer Expert Rules

The Edge Placer inference engine has three major rules:

(1) Place blocks by their associated IO pins.

(2) Place blocks by previously placed blocks in the same logical hierarchy.

(3) Fill perimeter with unplaced blocks.

Figure 2 illustrates these three rules and some additional secondary rules.

While the expert system only has three main rules, it has numerous sub-rules and sub-inference engines to actually place the blocks. For example, there is an inference engine that is called when an overlap is detected, and pre and post inference engines that preorder blocks by their dimension within logical hierarchy bins, and post inference that improves post placement by pushing blocks into a corners.

The placement inference engine uses procedures that find placements for blocks that cannot be placed at their optimal location due to some existing blockage, or to apply expert rules to blocks that don't have a well defined optimal location (locating a block adjacent to a previously placed block or placing loose blocks without associations.)

It also has rules for orientation (rotations to minimize congestion), to make associations, and some special rules for special blocks (e.g. relatively large blocks placed in corners). Many small rules can mimic the small optimizations an experienced designer performs such as fitting blocks along a single edge, pushing a block into a corner, and stacking blocks.

3.3 Results

Figure 3 compares floorplan results produced by both the Amoeba Placer and Block Placer from Cadence Design Systems and our expert Edge Placer. Run times for our expert system are very short, never more than minutes, because it is not an optimizer and runs in linear time with respect to the number of blocks. Its implementation using a slow TCL API nullifies the validity of any runtime comparisons.

Figure 3 shows compares edgePlace against the FE's amoebaPlace and blockPlace. Example 1 and example 2 in figure 3 are the results of a small testcase. Note the US/UA block on the left edge of the die in the amoebaPlace run of example 1. AmoebaPlace places its three associated blocks are in the upper right corner, while edgePlace stacks them in a row next to US/UA. BlockPlace, as shown in example 2 of figure 3, does a better job in terms of detecting IO connectivity, but noticeably leaves some blocks overlapping others. The IO pin placement for the example 1 and example 2 in figure 3 were the same for all tool runs.

Example 3 in figure 3 is an industrial design with 217,437 placeable instances, and an area of 3.5e+07 umˆ2, and has no blocks with connectivity to terminals, and shows little wirelength improvement over the amoebaPlace design, but has better clock skew, congestion, and routability results. The edgePlace runtime was 55 seconds on a 2 GHz Pentium class laptop, with most of that time spent on locating blocks in the design (searching instances) and and running the connectivity inference engine (that has not been optimized for speed).

3.4 Practical Methodology

For most designs, IO pin placement is driven by either external constraints or practical matters, be these board level constraints or system level constraints. In such cases, running edgePlace and fixing the blocks is sufficient to converge on a solution ready for routing and placement, otherwise a typical iteration is required to converge the placement of pins, cells, and macros.

4. Future Work

Unlike other approaches to floorplanning, we find clues in every floorplan we examine to improve the expert system, from new inference rules, to new floorplan styles. Unlike other approaches, expert systems can be improved in a methodical manner, using insights from a community of designers to intelligently improve the quality of results over time, with respect to both changes in technology and ASIC and SOC design requirements.

5. Conclusion

We have found that even simple rule based expert systems can demonstrate an uncanny ability to assist with the floorplanning of ASICs and SOCs, especially when applied to a particular design problem and style, in this case perimeter edge placement of hard blocks. Our perimeter Edge Placer uses rules that come from a limited set of experts, uses a very simple expert system inference engine, knowledge base, and rule set, but yet with very short run times produces results that are comparable to skilled ASIC and SOC engineers.

6. References

(1) Joseph C. Giarratano and Gary D. Riley, *Expert Systems: Principles and Programming*, PWS Publishing, 1998.

(2) Michael Will, *An Introduction to Expert Systems*, Picodoc Corporation, 2001.

(3) Rainer Bruck, Karl-Heinz Temme, and Heike Wronn, "FLAIR A Knowledge-Based Approach to Integrated Circuit Floorplanning", in the *International Workshop on Artificial Intelligence for Industrial Applications*, 1988.

(4) Marwan A. Jabri, "BREL - A Prolog Knowledge-Based System Shell for VLSI CAD", in the *27th ACM/IEEE Design Automation Conference*, 1990.

(5) Chen-Xiong Zhang, Andreas Vogt, Dieter A. Mlynksi, "Floorplan Design Using a Hierarchical Neural Learning Algorithm", in the *IEEE International Symposium on Circuits and Systems*, 1991.

(6) Kazuhiko Eguchi, Naok Tsuju, et al., "Application of Fuzzy Inference and Genetic Algorithms to VLSI Floorplanning Design", in the *26th IEEE IECON Conference*, 2000.

(7) Stephan Wolfram, *A New Kind of Science*, Wolfram Media, Inc., 2002.

(8) Ivan Sutherland, Bob Sproull, and David Harris, *Logical Effort Designing Fast CMOS Circuits*, Morgan Kaufman Publishers, 1999.

(9) Hiroyuki Watanabe and Bryan Ackland, "Flute - A Floorplanning Agent for Full Custom VLSI Design," in the *23rd Design Automation Conference*, 1986.

(10) Ralph Otten and Lukas Ginneken, "Floorplan Design Using Annealing" in the *2nd ACM/IEEE ICCAD, 1984.*

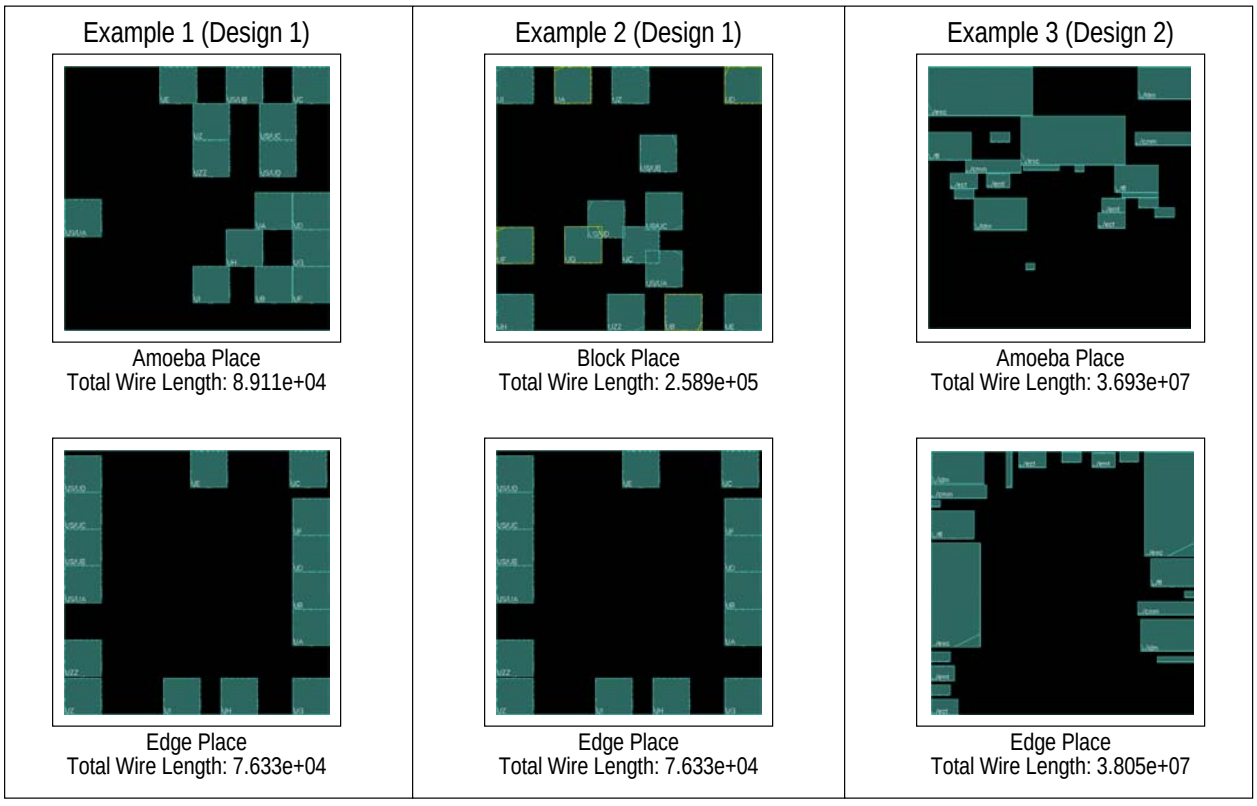

Figure 3: Edge Place versus Amoeba Place