ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

Development of EDA Tool with Easy Plugin for New VLSI Algorithms

Ashwini Desai, Ashwini Baligatti, Harsha Pal, Savita Y. Barker, Uday Wali

Abstract*— An EDA tool has been developed with an emphasis on teaching-learning of various algorithms related to graph theory and VLSI design. The tool provides basic graphics operations, coordinate control and a command processor. Extensions to the code can be done easily. Command processor allows commands to be issued from an external source or within an application. The tool provides a platform for learning and testing various algorithms with a visual feedback. The tool is primarily designed for students to practice algorithms in Graph Theory (GT), physical VLSI, GIS and other such graphic oriented technologies. We have presented design overview of the tool and sample applications in GT and VLSI. The tool has been developed using C# language and runs on Microsoft Windows platform as well as Linux Mono platform.

Index Terms— **Electronic Design Automation, Floor planning, Simulated Annealing, Channel Routing.**

I. INTRODUCTION

With a boom in the electronic design, there is a need to train high quality programmers exposed to VLSI technology and algorithms related to VLSI design. Physical design of VLSI is often a very time consuming process and hence is automated to the maximum extent possible. However, many of the problems faced in physical VLSI design are NP Hard. So, statistical, and heuristic methods have been used. Floor planning may also be done using genetic algorithms. Most of these methods give a near optimal solution rather than an exact and minimal solution. By their very nature, the algorithms keep improving over period of time and hence there is a need to educate large number of students in this area. To facilitate implementation of new algorithms and to test it, we need a common platform where one can easily compare performance of algorithms. Physical design cycle of VLSI includes floorplanning, routing and compaction stages, among many others. Instead of developing individual solution, we have developed an EDA tool on common platform for implementation of various VLSI automation algorithms. The implementation includes algorithms such as simulated annealing, channel routing, compaction and also contains applications of graph theory such as MST routing and constraint graphs.

A tool such as the one we have developed should be capable of certain minimum features and be extendable. Good vector graphics will be required to represent the underlying structures in physical and abstract structures. Basic transformations like scale, rotate, pan, zoom, view ports, etc will be required. Accuracy of representation is a prime concern. Both integer and floating point representations have accuracy problems specific to graphics applications. Hence, most of the graphics engines use a fixed point representation. Graphics engine should support internal scaling that converts world coordinates to a fixed point or scaled integer representation. Some additional features include transparency, shading, symbol library and fonts will also be required.

A command processor will channelize the commands from various sources like mouse, keyboard, file or network in to a single stream so that integration becomes easy. Command processor should support basic commands like add a point, tentative points, event triggers on user input, event triggers on command completion, command error processing, etc. The command processor acts like an interface between the graphics engine and the user. It receives user inputs, serializes the input command, transport it to the graphics engine or persistence module that will create storage as necessary. It will also be able to take stored procedures or objects and generate necessary graphics and or objects as necessary. The command processor should support adding new commands to a basic set of commands. Users can add their own commands using this command processor. For example, Geometry class provides point as a data structure. However, a GT application may call a point as Node, extending basic features of Point data structure. The GT application will always handle point information as node and not a point. So, a GT application should be capable of adding a new command named ‘Add Node’ instead of generic ‘Add Point.’

In this paper details of our implementation of such a tool are given. We have built the tool and used it to write many algorithms related to physical design of VLSI. Graph theory is extensively used in many of these algorithms.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

Hence, a basic graph theory module is also presented.

II. DESIGN OVERVIEW OF THE TOOL

Basic geometry structure contains a descriptor of geometry type and an array of points required to construct it graphically. For example, if we want to define a line, the geometry type is set to LINE and two points are stored. A rectangle is also described by two points but the type is set as RECTANGLE, triangle by three points which is set as TRIANGLE, etc. In case of a polygon, closing edge is implied to be between the first and the last points. Other geometries supported are point, line, Rectline (rectilinear), circle, arc, and spline. Filled and wire frame geometries can be drawn. The base *Geometry* class can be used as a basic structure for building more complex geometric shapes [3, 5]. A document contains a set of geometries. Geometry class is serializable. Derived classes have to serialize properties defined by them. Geometry class also provides a rudimentary rotate functionality: each point is rotated with respect to a common point passed as an argument to the rotate method. This is good enough for most of the cases but in some cases, overriding rotate method may be required.

The command processor generates an event several times during definition of geometry. Following is a list of operations that trigger an event:
a) Begin of create geometry
b) Addition of a point to geometry
c) End of creation of geometry

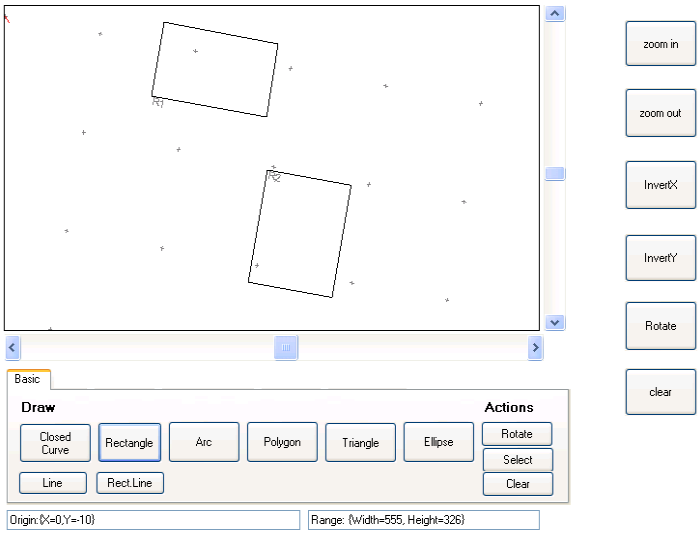

Fig.1: GUI of a View port [5]

Calling application can install a callback on each of these events. This gives application developers flexibility in their design.

Viewports can be created and scaled independently. This gives users freedom to see multiple views of the same underlying document. Each viewport is an instance of the main form used to display the geometry. So, each viewport is independent of the other. View operations on each of these view ports are applicable to the specific viewport only. There are some commands on the view port (e.g. clear document) which affect the document. Viewports provide basic operations like scale, translate and rotate to implement zoom and pan.

Part of GUI of a view port is shown in Figure.1. The basic geometry object provides methods to estimate the *range* of object. This method needs to be overridden for most of the application classes, when using non-rectangular objects. A *draw* method is provided that supports all basic geometries like rectangle, curves and polygons [3, 5]. All application class objects must override this method to produce relevant images.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

III. EXAMPLE IMPLEMENTATIONS

It is possible to develop applications that implement specific applications on top of this basic tool discussed above. We will discuss some applications developed by us, in the following paragraphs.

A. Simulating problems in graph theory

A set of classes have been defined to facilitate testing graph theoretic problems. Figure.2a shows creation of a graph with arbitrary number of nodes and weighted edges. Figure.2b and 2c show minimum spanning tree based on length and cost. The main classes added to the tool are NetGraph, Nodes and Edges. Nodes and Edges can store properties associated with them. They are derived from the geometry class and hence override the *draw* method. The NetGraph class implements several frequently used algorithms like Kruskal's algorithm for minimum spanning tree [5]. It is implemented only as a capability demonstrator. Additional methods and algorithms can be easily added. Minimum spanning tree algorithm has been used to implement MST routing in physical VLSI modules, discussed later.

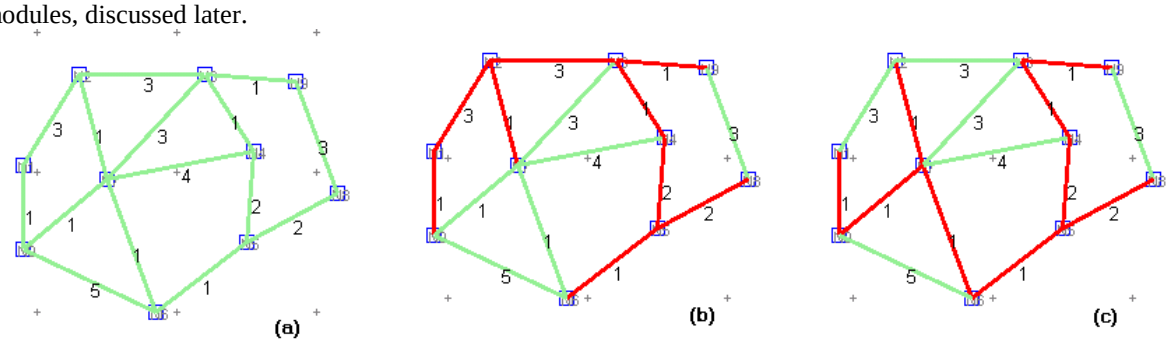

Fig.2: Implementation of a Graph Algorithm, using the EDA Tool [5]

B. Simulated Annealing

Floorplanning is considered as hard problem as it has to meet several constraints for all the modules. Simulated annealing is one of the accepted methods to achieve near optimal floor plans [1]. This method is similar to thermodynamic process of annealing metals. In annealing process metal is heated to high temperature and allowed to cool, on a predefined cooling curve. In the application, free space around a module is estimated and the module is moved within free area such that area of floor plan is minimized [3].

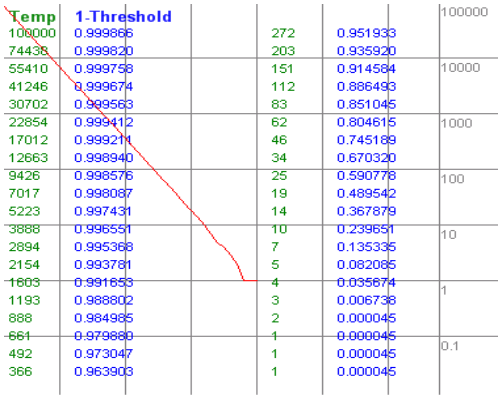

Fig.3: Logarithmic cooling schedule [3]

Simulated annealing process starts at a higher temperature and is decreased gradually. The process takes some random initial floorplan and performs set of move and rotate operations which change the configuration and area of floorplan. One advantage of simulated annealing algorithm is it accepts both the uphill moves and downhill moves i.e. the moves that result in increased and decreased floorplan area, but the probability of accepting the moves that result in increased floorplan area depends on temperature. At higher temperature both uphill and downhill move are accepted as the temperature achieves a lower value, only downhill moves are accepted [1].

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

In the application estimation of free area around each block is calculated and the blocks are being moved in the free area towards the top-left corner of bounding rectangle until no vacant space is left.

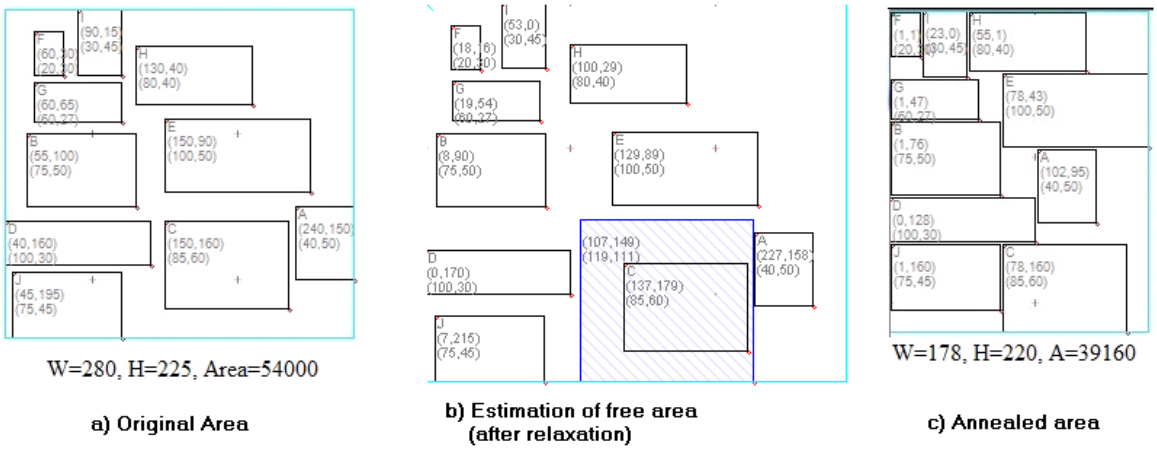

Fig.4: Results of Simulated Annealing on a Small Area [3]

Floor plan optimization often uses horizontal and vertical constraint graphs (HCG & VCG respectively). A module has been added to the tool to create horizontal and vertical constraint graphs [1]. This is also used in a compaction algorithm discussed below. Figure.5 below shows a set of rectangles and the two constraint graphs for the rectangles. We have used different colors to identify the two graphs. Rectangles have been drawn arbitrarily. CGNode class implements these methods. Compaction is relatively an easy operation, once the constraint graph is determined.

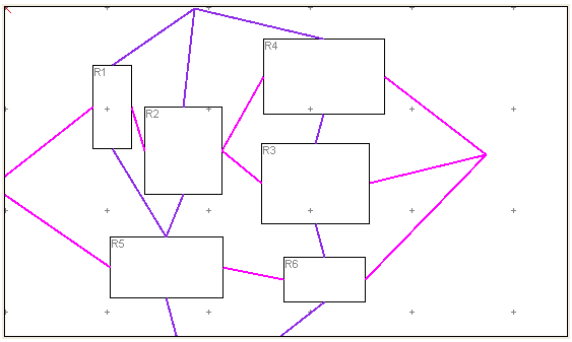

Fig.5: Horizontal and Vertical Constraint Graphs [4]

The operation essentially involves moving the rectangles along the x and y axes, as given by the left and top edges of a rectangle. The constraint graphs clearly show the vacancy position in all directions of any rectangle [4]. Figure.6 below shows how HCG can be used to compact the layout horizontally.

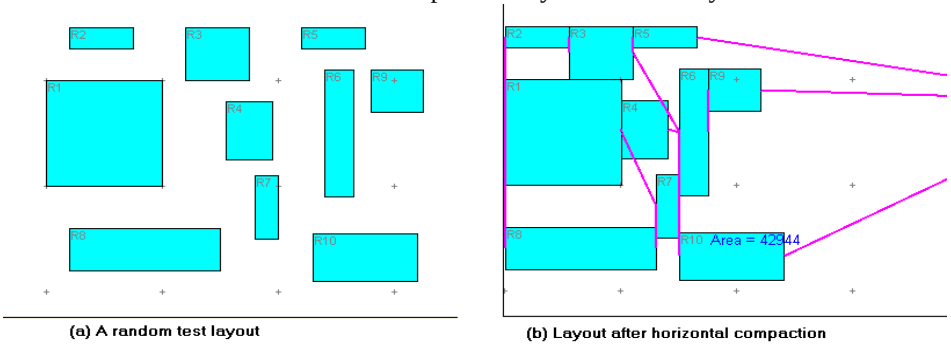

Fig.6: Using HCG to Compact a Layout Horizontally [4]

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

C. Routing

Channel Routing is a special case of routing problem in which wires are connected within a channel. The main objective of channel routing is to minimize the channel height by using minimum number of tracks. As a sample implementation, we have implemented a variant of Left edge routing as it provides complexity sufficient to prototype a routing tool. Channel Routing is implemented using CRChannel class.

This class uses several classes like CRTrunk, CRTrunkSegment, CRTrack, CRTrackSegment, CRTerminal, etc. Several enums have also been defined to identify layers, type of routing required, etc. CRTrack class represents one track which may contain a list of segments. On the other hand, CRTrunk, representing a vertical track may contain several segments, which are maintained as an array. Figure.7 shows a small channel routing implementation on this tool.

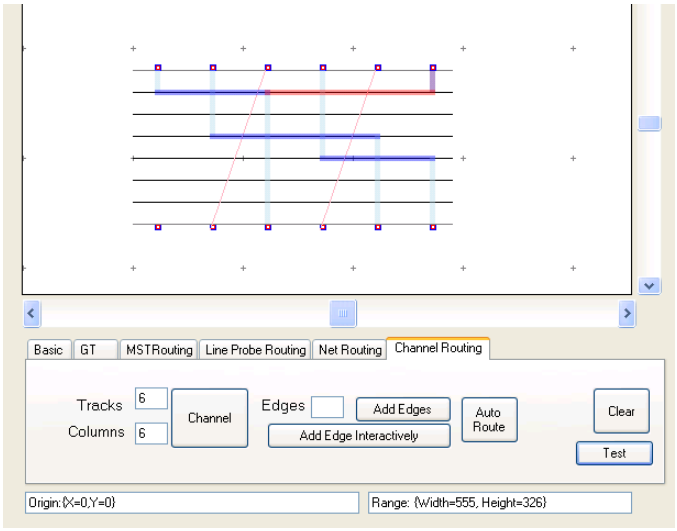

Fig.7: Channel Routing Implementation on our EDA Learning Tool [5]

IV. CONCLUSION

The application which is developed consists of Graph Theory and VLSI design algorithms. The developed EDA tool helps us to analyze various algorithms for student's research. These applications have been implemented on .NET platform using C# language. The developed tool can be extended to implement other design processes such as Partitioning, Line probe routing etc.

ACKNOWLEDGMENT

The authors would like to thank C-Quad for their continuous involvement in the project and support. The authors would also like to thank the college for providing necessary infrastructure.

REFERENCES

[1] Naveed Sherwani, - "Algorithms for VLSI Physical Design Automation", Kluwer Academic Publisher, 3rd Edition.

[2] E Balaguruswamy, -"Programming in C# ", A Primer, Second edition.

[3] Ashwini Baligatti, "Simulated Annealing in Floorplanning", M.Tech Project Thesis submitted to VTU June 2014.

[4] Harsha Pal, "Area Compaction in Floorplanning for EDA in C#", M.Tech Project Thesis submitted to VTU June 2014.

[5] Savita Barker, "Implementation of Multi Layer Channel Routing", M.Tech Project Thesis submitted to VTU June 2014.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

AUTHOR BIOGRAPHY

Prof. Ashwini Desai completed her BE (E & CE) from GIT, Belgaum affiliated to Karnataka University, Dharwad and was awarded M.Tech. by VTU, Belgaum in the year 2005. She is currently pursuing her PhD, while working as Asst. Professor at KLEDRMSSCET Belgaum. She has presented her work in National and International conferences.

Ms. Ashwini Baligatti completed her Bachelor of Engineering with major in Telecommunication Engineering at KLECET, Belgaum. Presently she is Pursuing Master of Technology in VLSI Design and Embedded systems at KLEDRMSSCET Belgaum.

Ms Harsha Pal completed her Bachelor of Engineering with major in Electrical and Electronics Engineering at GIT, Belgaum. Presently she is Pursuing Master of Technology in VLSI Design and Embedded systems at KLEDRMSSCET Belgaum.

Ms. Savita Y Barker completed her Bachelor of Engineering with major in Electronics and Communication Engineering at SDMCET, Dharwad. Presently she is Pursuing Master of Technology in VLSI Design and Embedded systems at KLEDRMSSCET Belgaum.

Dr. U. V. Wali is a Professor at KLEDRMSS CET Belgaum and Director at C-Quad computers, Belgaum. He was awarded Ph D from IIT Kharagpur in the year 1986 for his work. He has published papers in 8 International conferences and has also published 6 Journals. His research interests include VLSI physical design, testing and verification, software design and architecture, among others. He is a fellow of Institution of Engineers, India.