CHAPTER 1

INTRODUCTION

A digital circuit consists of circuit components and their connections. To realize it, a sequence of processes is performed starting from user specifications and their refinement, circuit design, simulation, and testing. Each of these processes is separately performed and optimized individually. The optimization is necessary as it leads to improvement in circuit performance both in terms of speed of operation and the area required to implement it. Once a circuit is implemented physically, it needs to be tested for its functional correctness as the manufacturing process can lead to fault introduction in terms of shorts and voids of connecting wires. This chapter provides a detailed introduction about design steps ranging from fault simulation, circuit partitioning, floorplanning and channel routing. To address these optimization problems, various *Evolutionary Algorithms (EA)* and their variants are described. The algorithms provide superior solutions in terms of various optimization constraints like a number of test patterns for fault simulation, the number of interconnections in circuit partitioning, overall bounding rectangle size in floorplanning and number of rows in channel routing.

1.1 CIRCUIT DESIGN IN VLSI

The Circuit’s made of electronic components are playing a very important role in today’s world. A VLSI IC contains a very large number, i.e. greater than $10^7$ transistors. This
A tremendous increase in circuit complexity is due to vast and well-adapted use of Electronic Design Automation (EDA) tools such as Xilinx’s ISE Web pack, Synopsys’s Virtuoso, and Tanner’s LEdit and many more. These tools are widely used to streamline and simplify the complex design problems by performing the complex design tasks themselves and hiding low-level circuit details from the designer.

*For every VLSI design problem, the main targets to be met are maximizing speed of operation while minimizing their power requirement and wire length required to connect all the logic modules and area to implement that design.*

In manufacturing a VLSI circuit, the circuit designer completes the following steps, as shown in Figure 1.1.

- **Specification:** It produces functional details of the design to be implemented.
- **Logic design:** It converts the functional details into a logic expression using Boolean equations.
- **Circuit design:** It converts the logic description into circuit components by using a library of circuit components.
- **Physical design:** It translates the circuit design into a physical package represented using a set of connected components which when connected leads to a working circuit.
- **Fabrication:** It uses the available manufacturing technology to fabricate an IC.
- **Testing:** It verifies whether during the fabrication process some defects are formed which may lead to the faulty behavior of the circuit.
The circuit design has nowadays become complex and designing and testing them is a trivial task for the designer. So, to aid that optimization algorithms are used to simplify the task and maintain the required yield. [Gerez Sibih II, 2006 & Cohoon James, et al., 2003]. Optimization of all the physical design steps is very well explained in the literature and by numerous researchers such as [Shanavas I, et al., 2014; SR Jay, et al., 2014; Singh Shivam, et al., 2013 & Gaur Priti, 2013]. Still, it is a topic of research due to the new set of optimization techniques and their promising results, when compared with the existing techniques [Shiang Chyi, et al., 2013]. These techniques, when get incorporated into the EDA tools result in the better optimization results for the circuits under consideration. To address the exponential growth of VLSI design in system functionality, an equal advancement in EDA tools is required.

The physical design phase of the circuit design includes partitioning, placement, and routing. The various processes involved in VLSI design can be solved and improved using optimization techniques. The physical design methodologies are facing several challenges such as power dissipation and speed of operation brought forth by the aggressive scaling of semiconductor technologies. The system performance and timing optimizations performed in early design stages are typically based on the estimation of interconnect delay. In addition to this, placement algorithms may generate un-routable placements due to routing congestions in some regions of the chip area. Thus, algorithms are needed to handle this congestion due to increased data flow to cope with the scaling of technologies. The optimization algorithms aid the EDA tools by providing the best possible solution in each step of the physical design problem.

1.2 EVOLUTIONARY ALGORITHMS (EA) OVERVIEW

Various Artificially Intelligent (AI) methods are available and are applied in order to raise the intelligence level of intellectual CAD systems to expand their features and increase effectiveness. One of the most obvious and determining approaches is by using Evolutionary Algorithms (EA).

Evolutionary algorithms are search and optimization algorithms which operate on a set of solutions constituting a population and evolve the best solution out of them by using heuristics or rule-based transformations. The final solution obtained using a set of constraints is the optimized result for the targeted problem.
The general operation of EA is given in Figure 1.2.

![Figure 1.2: Operational Flow chart of Evolutionary Algorithms](Mazumdar P, et al., 2000)

Evolutionary Algorithms (EA) are mainly classified in three categories: Genetic Algorithms (GA) which have been proposed by Holland in 1975, Evolutionary Programming (EP) which have been proposed by Fogel in 1966 and Evolutionary Strategies (ES) which have been proposed by Rechenberg in 1973. All the techniques have one thing in common, for each probable solution, there is a fitness value assigned which characterize its possibility of moving in future generations. [Drechsler Rolf, 1998].

In Genetic Algorithm (GA), the solution is encoded and is represented as a chromosome. The encoding scheme used must lead to transformations in the form of other population members. GA’s are optimizing techniques and are not always providing the global optimum for a given problem, but they generally provide a good and acceptable solution. GA’s are heuristic based algorithms whose output depends on evolution strategies of the optimization function used to find the fittest of all the population elements. They mimic the evolution by natural selection. Before a genetic algorithm can be run, an Encoding (or representation) for the problem must be devised. A fitness function, which assigns a figure of merit to each encoded solution, is also required. Populations of Binary Strings (chromosomes) are created by the random generation of each bit (gene) and the chromosome length is equal to the number of inputs (n). The chromosome represents a solution of the original problem, and performance of GA strictly depends on its various parameters like population size, crossover, mutation, and inversion. The evolution of a new generation of solutions takes place by the use of Crossover and Mutation. The crossover takes two randomly chosen
individuals from the mating pool and recombines them to generate two new individuals, which carry a mixture of their parental genetic information. The mutation brings newness in the population [Mazumdar P, et al., 2000].

The abstract form of the genetic algorithm is represented below.

1. 
   [Start] Generate an initial random population of n chromosomes

2. 
   [Fitness] Evaluate the fitness \( f(x) \) of each chromosome \( x \)

3. 
   [New population] Create a new population by repeating the following steps
   Until the new population is complete
   3.1 [Selection] Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)
   3.2 [Crossover] Cross over the parents to form a child with probability \( P_c \)
   3.3 [Mutation] Alter some attributes of new offspring at a locus (position in the chromosome) with certain probability \( P_m \).
   3.4 [Accepting] Introduce the new offspring in a population

4. 
   [Replace] Repeat the algorithm with the new mating pool

5. 
   [Test] Check if stopping criteria is reached, stop then, and return the best solution

6. 
   [Loop] Go to step 2

One most common transformation process for both Genetic Algorithm (GA) and Evolutionary Strategies (ES) techniques is the crossover, where the two chosen solutions are allowed to exchange their strings to generate two new solutions termed as Offspring. Another important transformation is called Mutation. Out of a set of population elements, any element is selected at random and is modified. The mutation is used in all the Genetic Algorithm (GA), Evolutionary Strategy (ES), and Evolutionary Programming (EP) algorithms.

- In a Genetic Algorithm (GA), the mutation operator is applied to the solution in itself.
- In Evolutionary Programming (EP), a copy of the parent solution is made and the mutation is applied over it.
In **Evolutionary Strategy (ES)** techniques, where the crossover operator is used, the mutation is applied to the offspring in an in-place manner.

Evolutionary algorithms in their basic form start by forming a set of solutions initially. The population members initially generated are random solutions to represent the various portions of the solution space. It leads to a large collection of solution characteristics which are a combination of both good and bad population elements. This set of probable solutions is called a **Population**. The population numbers are subjected to a sort of competition with each other in order to get selected as parent members. It is called **Parent Selection Process**. In the case of **Genetic Algorithm (GA)**, the selected members are termed as **parents**, and this selection is based on the probability where more fit members are chosen first. In the case of **Evolutionary Strategy (ES)** model, all parent solutions have the same probability of being chosen. These selected members are used to generate new members by performing the crossover operation between two parents. Followed to this is the mutation operator which operates on a random population element to alter its properties. Then a tournament selection technique decides which elements of the previous iteration and the child members are carried forward to the next generation.

In **Genetic Algorithm (GA)**, the solutions are favored based on their fitness. In the case of **Evolutionary Strategy (ES)**, out of the total population consisting of offspring and parents the fittest is selected. This tendency of favoring the more fit solutions implies that algorithm iterates toward solutions with better performance characteristics. In order to introduce new variants in the existing population, mutation operation is performed on randomly selected member where their few properties have been altered. The minimal changes due to mutation allow the escape from local optimum and directs it toward the global optimum solution. The **stopping criteria** are whether the solution quality has reached the appropriate level or a predefined number of iterations have been performed. When more than one objective needs to be addressed Multi-objective optimization is used to optimize over two or more independent criteria. Apart from **Genetic Algorithm (GA)**, another popular class of EAs is **a Memetic Algorithm (MA)** which uses a local search technique to further improve the fitness of the generated individuals by using techniques such as hill climbing. MA is also called as **Hybrid Evolutionary Algorithm (HEA)**. The mixing approach termed as **Hybrid** is used to pace up the search for good solutions which otherwise may have taken too long time to discover by using local or primary search process alone. This benefit of an increase
in fitness of individuals using local search alone comes at a price of increased complexity in the design and development of the algorithm.

1.3 Fault Simulation and Physical Design Using Evolutionary Algorithms

After the device is manufactured and due to some defects in the manufacturing process few or more faults may occur in the device, which results in a faulty response. Fault analysis is normally the last process to ensure the effective functioning of the resulting design. After the fabrication of the design, the functional verification is judged using the test generation method. The set of test vectors is generated and are used to target all possible defects that may result in a circuit design process. It has a direct implication on the yield of the manufacturing process.

The main steps in Physical Design and Automation are:

1.3.1 Circuit Partitioning

The primary task of partitioning is to divide the system into subsystems which can be separately realized effectively. It is done to reduce the complexity of circuit designing of bigger circuits or when the silicon area available to realize the circuit is insufficient for the whole design [Bui T. N, et al., 1994]. The obtained sub-circuits are realized individually. The sub-blocks are depending on each other due to the interconnection between them and it gives rise to interconnect complexities. The main focus in partitioning is on various interconnects present between different sub-blocks and the goal is to minimize the number of interconnects between them [Sherwani N. A, et al., 2002]. Various Heuristic Algorithms such as Group Migration, Network Flow, and Clustering Algorithms are presented in the literature. The group migration algorithms are first proposed in this category by Kernighan and Lin (KL) in 1970 [Kernighan, et al., 1970]. Since then many variations have been reported to improve the efficiency and effectiveness of the method. The group migration method is faster and easier to implement [Cherng J. S, et al., 1996; Hoffman A. G, et al., 1994]. The metric allocation method is costlier in computing time than group migration method, and hardest to implement since it requires numerical programming. Apart from KL or its improvements like Fiducca Mattheyes (FM), various Evolutionary techniques like
**GA, Particle Swarm Optimization (PSO) & Ant Colony Optimization (ACO)** [Arora Manoj, et al., 2013; Arora S, et al., 2011; Sun T. Y, et al., 2006 & Priya K, 2014] are also used and good partitioning results have been reported by numerous authors [Sipakoulis G. C, et al., 1999; Gill S. S, et al., 2009; Kolar D, et al., 2004 & Singh R, et al., 2012]. To combat the drawbacks of the partitioning algorithm combinations of them are reported which are called **Hybrid Algorithms (HA)** [Chantngarm P, et al., 2004; Arora S, et al., 2011].

1.3.2 Circuit Floorplanning

The task of finding the module shapes and positions with the objective of composite block area minimization and aspect ratio requirement is referred to as **Floorplanning**. The various circuit blocks are assigned to their concerned geometrical locations on each sub-circuit or composite circuit as a whole [Shiang Chyi, et al., 2013; Sivaranjani P, et al., 2013 & RRajine Swetha, et al., 2014]. The empty spaces in between the blocks in the form of rows are called **Channels**, and are used for interconnecting the various circuit components. For all possible design styles, the main goal is to minimize the total floorplanning area of the chip and to minimize the length of the longest interconnect [Haghani Fahimeh, et al., 2013; KA Sumithra, et al., 2014]. Depending upon the applications, the placement algorithms can be divided into different categories for standard cell design, macro cell design, and FPGA [Bernardeschi C, et al., 2013], as per the size and location of these cells as given in Figure 1.3.

![Figure 1.3: Placement styles](Cohoon James, et al., 2003)

The block placement is to be done in such a way so that the routability can be ensured by reducing the congestion in between the cells as much as possible. The locations are assigned
by looking upon constraints like no overlap between cells and timing constraints. The interconnection cost is minimized by optimizing the length of routing by calculating overall routing length or by calculating individual interconnection costs between different modules [Gaur Priti, 2013]. The placement algorithms are earlier represented by generating an initial set of solutions, usually generated randomly [Shahookar, et al., 1990]. Their results are comparable to simulation annealing based placement techniques [Mohan S, et al., 1993]. Further, the runtime is greatly reduced by making a parallel implementation of a GA that runs on a distributed network of computers [Ahmed A.M, et al., 2014]. Different population members are evaluated on different processors and a migration scheme is used to exchange genetic material between them. It leads to speed up in result formulation. A linear speedup can be achieved with this technique. A parallel GA for placement problem is presented [Schnecke, et al., 1996] particularly for macro cells. Pure Genetic Algorithm (GA) is modified, i.e. Modified Genetic Algorithm (MGA) by using numerous heuristic operators to improve its performance. GA is also proved to be applicable for placement of gate matrix design [Shahookar K, et al., 1994]. Basic GA is used and is then modified by using a unique algorithm for generating instructions that use only a small subset of solution space.

1.3.3 Channel Routing

In the physical design process, Routing follows the placement/floorplanning activity and is done to find out the prospective path of the interconnecting lines between the circuit blocks. It is concerned with the connection of all the pins belonging to the same net and is performed in two steps i.e. Global Routing and Detailed Routing. A Global Router generates a loose interconnection pattern between the nets, i.e. set of terminals connected by a single wiring element. Then, the Detailed Routing is done to assign specific resources and connection pathways available in a channel to realize the interconnection. The Global Routing techniques are usually graph based search techniques where the nodes represent the different routing regions such as tiles or channels of the circuit. The focus is on one net routing at a time. For some design like gate matrix and FPGA design styles, the routing is the final step in the physical design. However, usually for block level based designs, the final process is the Compaction. It is the process of making connections between circuit pins subject to routing constraints and is done both for circuits inside a single chip and also for Multi-Chip-Modules (MCM). Compaction is divided into Global Routing to lay down the routing between nets into certain regions and detailed routing to assign nets to exact positions inside
the routing channels as shown in Figure 1.4. In Figure 1.4, IC stands for Integrated Circuit, MCM stands for Multi-Chip Modules and PCB stands for Printed Circuit Board.

In Detailed Routing, the regions are routed one by one in a sequential manner. The detailed routing can be separated into the Channel Routing and Switch Box Routing. In channel routing, the pins are placed on two parallel sides of the routing area and in the switch box routing the pins are placed on all four sides of the routing area as shown in Figure 1.5.

The GA, in terms of its solution encoding, [Lienig J, et al., 1994] depends on the problem to be addressed. The floorplanning is represented as the 3D structure and the chromosomes
acting as cells represent various coordinate points of the solution [Lienig J, 1997]. The cell value provides the net information rooted in a particular coordinate point and the negative cell value represents the fixed assignment as shown in Figure 1.6.

![Sample routing (left) and its Genetic Encoding (right)](image)

Figure 1.6: Sample routing (left) and its Genetic Encoding (right) [Thulasiraman K, et al., 1994]

For Switchbox Routing process, the GA is proposed [Lienig J, et al., 1996] and the solution is in the form of a lattice and it corresponds to the coordinate points of the floorplanning.

*The final step in physical design is done to provide a final squeeze to the resultant circuit without affecting the performance characteristics using compaction technique.*

### 1.3.4 Circuit Compaction

Compaction of the circuit can be achieved either by better placement of the given block in a given area or by minimizing the interconnect length. The size of the resultant circuit after placement and routing is reduced, with a goal of minimizing the size of resulting circuit representation [Goodman E, et al., 1994].

The steps are implemented by considering the design rules. The compaction algorithms are divided into three categories as:

- **One dimensional** where the reduction is performed in one direction at a time.
- **Two-dimensional algorithms** where the compaction is performed in both $x$ and $y$ direction simultaneously.
- **Topological algorithms** where the cells are moved according to the routing constraints.
1.4 MOTIVATION OF RESEARCH

The motivation of this research work stems from the fact that the present day circuits are so complex in nature, both in terms of a number of logic components and their interconnections that their design, as well as testing, is beyond the range of human hands. So Computer Aided Design (CAD) comes to the rescue of the circuit designer. It involves the Electronic Design Automation (EDA) tools which help to automate the various circuit design tasks which otherwise becomes impossible to handle. Due to the rapid increase in the demand for incorporating more function, at each logic implementation step, the process is needed to be optimized otherwise the process may not function or yield properly. Thus, by using optimization through a class of EA, the best solution or results in terms of optimization parameters can be obtained. These solutions are obtained by keeping a set of constraints to be followed and thus they provide the best input for subsequent processes. The Evolutionary Algorithms (EA) techniques can play a major role by giving an optimized solution for the design and test steps such as circuit partitioning, circuit placement followed by routing. Once these circuits are designed then EA helps by minimizing the set of test vectors required for fault simulation [Roy Saradindu, et al., 2012; Singh Shivam, et al., 2013]. The optimal results obtained can have a direct impact on area consumed, number of interconnections and yield of the manufacturing process.

1.5 LITERATURE REVIEW

In this thesis, the work is to design different types of Evolutionary Algorithms (EA) for various categories of VLSI design problems. To address the problem effectively, a thorough study of related work is required to understand the problem clearly and also to compare the results obtained with the existing techniques.

1.5.1 Literature related to Fault Simulation

A two-phase fault simulation technique for sequential circuits is proposed by Wu, Lee, & Chen. The true value simulation with several initial patterns is performed first and then performing the fault simulation with the rest of the patterns [Wu W. C, et al., 1993]. A simple GA is used by Rudnick, Holm, Saab, & Patel to generate populations of candidate test vectors and to select the best vector in each time frame. They have found the execution times to be lower than the deterministic test vector [Rudnick E. M, et al., 1994]. Prinetto,
Rebaudengo, & Reorda have presented an approach of an automated test pattern generator for large synchronous sequential circuits. The advantage is the tradeoff between fault coverage and CPU time to suit the circuit requirements [Prinetto P, et al., 1994]. For the generation of test patterns, O’ Dare & T Arslan have used a technique which utilizes a genetic algorithm that detects single stuck-at faults in combinational VLSI circuits. The GA is used to search the solution space for patterns that detect the highest number of remaining faults in the fault list [O’Dare M. J, et al., 1994].

The candidate test vectors for sequential circuit test generation have been evolved in a GA framework by O’ Dare & T Arslan. The technique has been verified with ISCAS 89 circuits and execution times are also found to be lower than the deterministic test vector [Rudnick E. M, et al., 1994]. A Dynamic global reference table for fault simulation using GA is proposed by O’Dare & Arslan. The test pattern generation is guided by dynamically evolving a global reference table and the circuits are explored in a hierarchical manner for improved fault simulation [O’Dare M. J, et al., 1995]. Corno, Prinetto, & Reorda have used the cellular automata to generate input patterns to detect stuck-at faults inside FSM. The GA has been used to identify cellular automata to achieve fault coverage [Corno F, et al., 1996]. A new technique for generation of test vector pairs that can detect both delay and stuck at fault conditions in digital logic circuits is presented by ’Dare & T Arslan. The GA has been used to generate tests from a complex search space [O’Dare M. J, et al., 1996].

The stuck-at fault and delay faults using GA have been targeted for combinational circuits by T Arslan & O’Dare [O’Dare M. J, et al., 1994 & O’Dare, et al., 1997]. The effectiveness of GA is studied by Rudnick, Patel, Greenstein & Niermann for sequential circuits by various GA parameters like alphabet size, fitness function, generation gap, population size and mutation rate as well as selection and crossover schemes [Rudnick E. M, et al., 1997]. Hartanto, Boppana, Patel, & Fuchs have presented a method for the diagnostic test generation in sequential circuits by modifying a conventional test generator using a circuit netlist modification [Hartanto I, et al., 1997]. A parallel genetic algorithm for simulation based sequential circuit test generation has been used by Krishnaswamy, Hsiao, Saxena, Rudnick, & Patel. The algorithm is portable and scalable over a wide range of distributed and shared memory machines [Krishnaswamy D, et al., 1997]. A new objective function by Rajesh & Jain for generation of test patterns for the sequential circuit using genetic algorithm have been proposed. The simulation based approach is effective even for large sequential circuits [Rajesh V, et al., 1998]. A diagnostic fault simulator is
proposed by Wu & Rudnick that diagnoses both feedback and non-feedback bridge faults in combinational circuits using information from fault simulation of single stuck-at faults [Wu, J, et al., 2000]. Kim, Agrawal, & Saluja have proposed an algorithm to model multiple stuck-at faults as a single stuck-at fault. The technique allows simulation and test generation for any arbitrary multiple faults in combinational and sequential circuits [Kim Y. C, et al., 2002]. A technique to remove faults from a set of suspected faults have been proposed by Takahashi H, Boateng, Saluja, & Takamatsu. It depends on the result of multiple-fault simulation at a primary input agreeing or disagreeing with the observed value [Takahashi H, et al., 2002]. The test vector simulation method for path delay faults using stuck at faults has been used by Ohtake, ohtani, & Fujiwara. A two pattern test for fault detection is obtained by transforming an original circuit into leaf dog using path leaf transformation, and using it to generate test patterns for sticking at faults [Ohtake S, et al., 2003].

Zhongliang has used the circuit realizations which uses XOR gates, tree construction and generalized Reed-Muller expressions of logic functions. With this technique, all AND and OR bridging faults can be detected [Zhongliang P, 2003]. A PSO-based approach for test generation have been used by Hou, Zhao, & Liao. The synchronous sequential circuits are considered by targeting three aspects: initialization, test sequence generation, and test set compaction [Hou Y, et al., 2006]. Ibrahim, Elchouemi, & Amer have used a two-phase genetic algorithm for test vector selection and condition coverage by using an intelligent test vector selection [Ibrahim W, et al., 2006].

After going through the related work of fault simulation, it has been felt that an optimization algorithm is required, which can address more than one fault at a time. The algorithm has to be efficient enough to minimize the total number of test patterns which can uncover all the faults. It is also felt that a technique has to be devised which can perform fault simulation in both combinational and sequential circuits.

1.5.2 Literature related to Circuit Partitioning

A comparison of GA and SA approaches for solving the circuit partitioning problem have been presented by Cain & Manikas. When compared to SA, the GA has been found to produce similar or better results [Cain J. T, et al., 1996]. Johannes F has used a two-way partitioning of a circuit using simulated annealing. It leads to a minimum delay in the resulting circuit. The various control parameters are estimated and their influence on the
delay between the partitions is evaluated [Johannes F, et al., 1996]. A top-down quadrisect ion based global placer for standard cell floorplanning is used by Huang & Kahng. They have used a gain update scheme along with multilevel petitioner as the basis for a new place [Huang J. H, et al., 1997].

Alpert, Huangq, & Kahng have presented a multilevel partitioning algorithm which recursively clusters the instance until its size is smaller than a given threshold, then un-clusters the instance while applying a partitioning refinement algorithm [Alpert C, et al., 1998]. A hypergraph partitioning algorithm based on the multilevel paradigm have been used by Karypis, Agarwal, Kumar, & Shekhar. On the basis of smallest hypergraph bisection, the bisection of an original hypergraph is computed by successively projecting and refining the bisection to the next level finer hypergraph [Karypis G, et al., 1998]. The dual purpose of balancing the size of two partitions and even distribution of connections among partitions is targeted by Zoltan, Octavian, & Kalman using GA in circuit partitioning problem. The performance is compared with SA and found that the execution time of GA is lower [Zoltan B, et al., 1999].

A hybrid algorithm is used by Apiradee Yodtean to improve the performance in circuit partitioning by using fewer resources. The algorithm combines the advantages of SA and GA and thus produces a superior result [Apiradee P. C, et al., 2004]. Guo-Fang Nan, Min-Qiang Li & Ji-Song Kou have explained the two styles of genetic algorithms based on different encoding strategies for circuit partitioning. The first adopts the form of 0-1 encoding, and the second uses integer encoding based on the module number [Guo Fang Nan, et al., 2004]. The min-cut placement technique is integrated with fixed-outline floorplanning by Adya, Chaturvedi, Roy & Markov to solve the more general placement problem, including cell placement, floorplanning and achieving routability [Adya S. N, et al., 2004]. Dalibor, Branica, & Julijana have proposed a two-way partitioning of a circuit represented as a graph using the simulated annealing procedure. By using a proper choice of the initial temperature and cooling rate, a relatively good solution can be obtained in less time [Dalibor K, et al., 2004]. The heuristics for the optimization of VLSI netlist bi-partitioning have been proposed by Sadiq M Sait. These are based on GA, TS, and SA. The multi-objective cost function is handled by incorporating fuzzy rules [Sait Sadiq M, et al., 2006 & Jayanthy S, et al., 2015]. A GA-based approach for improving the speed up over conventional software implementation is used by Coe, Arebi, & Moussa. They have used a reconfigurable computing based hybrid memetic algorithm for circuit partitioning [Coe S,
Subbaraj, Sivasundari, & Kumar have used a memetic algorithm for optimization of VLSI partitioning. GA has been used for global search and simple local search for handling the multi-objective [Subbaraj P, et al., 2007].

An evolutionary time series model and a statistical glitch prediction using a neural network with a selection of global feature by making use of the clustering method model, for partitioning a circuit have been proposed by Sumitra Devi, Banashree, & Abraham. The performance of all approaches has been compared using MCNC standard benchmark netlist [Sumitra Devi, et al., 2007]. The swarm intelligence based approach have been proposed by X.Y to optimally partition combinational CMOS circuits. It is based on the circuit’s maximum primary input cone size and minimum fan-out values to decide the location and number of partitions [X. Y, et al., 2007]. A GA-based technique for circuit partitioning has been proposed by A Deep, which produces better results using an improved crossover operator. Space and time complexity of the algorithm is much better than simple GA [Deep A, et al., 2009]. The performance of ACO and GA for VLSI circuit bi-partitioning have been compared by Deep, Singh, Singh, & Singh. The results obtained show that GA outperforms ACO technique when tested on a VLSI circuit bi-partitioning problem [Gill S. S, et al., 2009]. Wankhede & Deshkukh have presented a technique for structural design of circuit by using GA. They have utilized a library of devices for synthesis procedure. It leads to success in searching of highly complex space and structures by circuit criteria [Wankhede M, et al., 2009]. A GA-based technique by S.S. Gill for multiway circuit partitioning have been proposed. The results show that because of inherent parallelism and robustness of GA, it can be used effectively for NP-hard problems [Gill S. S, et al., 2010].

To generate random realistic benchmark circuits for analysis, a method have been presented by Prasad & Reddy. A prediction model has been used that predicts the length, width, area and power of the benchmark circuit. The wire length is estimated to calculate the width, area and power of the circuit [Prasad V. V, et al., 2010]. A GA-based circuit partitioning algorithm has been used by S. S. Gill which typically partitions logic gates or instances of the standard cell. The encoding strategy used is an integer encoding based on module number. A swarm intelligence based approach for circuit partitioning is described which is tested on 1671 circuit partitioning instances given on the MARCO GSRC VLSI CAD bookshelf website [Gill S. S, et al., 2010]. S.S. Gill has used the simulated annealing for the circuit netlist bipartitioning. Various parameters of SA like initial temperature, cooling rate and a number of iterations have been changed to analyze the effect of their variation on the result [Gill S. S, et al., 2011]. An algorithm for circuit partitioning has been presented
by S Supreet Singh. It starts from a good initial solution and the end result is better in terms of a minimum number of interconnects and can be obtained in less time. The proposed algorithm is tested on NETD instances available on GSRC bookshelf, USC San Diego site [Singh S. Supreet, et al., 2011].

Gnanamurthy has presented a technique of wire length reduction using memetic algorithms. Its performance is compared with genetic algorithm and it have been found to be 45% faster, leading to a reduction in delay and area in partitioning and floorplanning [Gnanamurthy H. S, et al., 2011]. A hybrid GA is used by VÍLCU for solving the partitioning problem of combinational circuits by defining a mathematical model and an objective function for partitioning problem. By using GA operators, a way of local hybridizing is defined by an efficient heuristic algorithm acting as hyper mutation operator [VÍLCU A, et al., 2012]. Gill, Chandel, & Chandel have presented a netlist partitioning using particle swarm optimization technique. They have used PSO evolutionary technique for circuit partitioning for solving non-polynomial hard problems [Gill S. S, et al., 2012]. A multiway circuit partitioning approach using genetic algorithm has been presented by S. S. Gill, Rajeevan Chandel & Ashwani Chandel. They have found that by merely increasing the crossover points the fittest does not necessarily increases [Gill S. S, et al., 2013].

The circuit partitioning problem is an essential step as it attempts to minimize the number of interconnect crossovers between sub-circuits. An Evolutionary Algorithm (EA) based technique is required to address this problem as the delay and power dissipation of the circuit can be optimized by proper tuning of parameters. A suitable modification in the basic form of GA can target the problem more effectively by minimizing the number of interconnects and reducing the wire length. Another advantage which an EA can offer is that local improvement in an existing population is possible by using a combination of two existing optimization technique like Simulated Annealing and Genetic Algorithm.

1.5.3 Literature related to Floorplanning

The simulated annealing based floor planning algorithms have been proposed by Wong & Liu. The representation scheme used is the normalized polish expression and both the area minimization and total interconnect length in the final solution have been minimized [Wong D. F, et al., 1986]. Wang & Ting have used an optimal algorithm for the floor plan area optimization problem. The branch and bound algorithm is extended to handle large floorplan
problems and the run time comparison is found to be comparatively smaller than existing techniques [Wang D. F, et al., 1990]. To solve the floorplan design problem, Cohoon, Hedge, Martin, & Richards have presented a method using distributed genetic algorithm. The proposed technique performed better than the simulated annealing approach in terms of average cost and obtained solution [Cohoon J. P, et al., 1991].

H Esbensen has proposed a GA-based algorithm for the macro cell placement problem. The encoding of the problem has been done in a generic way and the corresponding genetic operators have been described [Esbensen H, et al., 1992]. The combination of both a heuristic graph bipartitioning procedure and a slicing tree representation have been used by Srinivasan & Nallasamy. The slicing tree representation is used to provide an efficient tree traversal operation using the recursion process for obtaining area efficient floor plans [Mani Nallasamy, et al., 1995]. Vornberger & Volker have presented an adaptive parallel GA-based approach for VLSI floorplanning optimization. The combined optimization of placement and routing is presented and the main focus has been on self-adaptation of the search process [Vornberger O, et al., 1996]. A novel hybrid genetic algorithm is used by Schnecke & Vornberger, for problem specific genotype encoding. It solves the macro cell placement problem and due to a tree-structured genotype representation, the proposed approach is showing good performance [Schnecke, et al., 1997]. Yingxin, Chung, & Takeshi have presented a deterministic algorithm based on O-tree representation. The main improvement as claimed is the reduction in the floorplanning implementation [Yingxin P, et al., 2000]. The normalized postfix encoding scheme is used in GA by Valenzuela & Pearl to solve the floorplanning problem. The algorithm uses a slicing tree construction procedure and has run time scaling comparable with prevailing approaches [Valenzuela C. L, et al., 2000 & Valenzuela C, et al., 2002]. Y.C. Change has used a B* tree-based approach for handling non-slicing floorplan. The flexibility of B*trees to handle rotated, pre-placed, soft, and rectilinear modules is presented [Chang Y. C, et al., 2000]. An optimization algorithm has been presented by P, Chen using linear programming methods for floorplanning. The approach can handle any topological constraints as well as soft/hard/preplaced blocks, and timing constraints [Chen P, et al., 2000]. F Y Younge has proposed a technique to handle soft modules, in general, non-slicing floorplan problem using a Lagrangian relaxation technique. The resulting subproblem is so simple that the optimal size of each module can be computed in linear time [Young F. Y, et al., 2001]. A GA-based placement algorithm for mixed and macro cell have been presented by W Theodore. The relationship information between macro and standard cells is used to determine a more efficient placement solution.
Evolutionary Algorithms for VLSI Applications

[Theodore W, et al., 2002]. The multilevel placement package is presented using an enhanced approach by Chan, Cong, Kong, Shinnerl, & Sze. The main improvements targeted are an unconstrained quadratic relaxation on small sub-problems at every level of the hierarchy, improved interpolation and iterated V-cycles with additional geometric information [Chan T. F, et al., 2003].

Lee, Chang, Hsu, & Yang have used a multilevel floorplanning framework based on B*tree representation to handle the floorplanning and packing for large scale building modules. It uses a clustering technique and preserves the geometric relations among modules which make it an ideal data structure for the multilevel floorplanning framework [Lee H.C, et al., 2003]. The floor plan design problem using GA is targeted by Kimura & Ida to improve the efficiency of the calculation, the maintenance of the solution’s population diversity, and reduction of the number of parameters [Kimura Y, et al., 2004]. Sebastian & Tang have used an O-tree based GA for floorplanning problem due to its smaller search space among other floorplan representations. The experimental results show that the GA provides better results than the deterministic algorithm [Sebastian, et al., 2005]. Maolin Tang has used a memetic algorithm for a non-slicing and hard-module VLSI floorplanning problem. It uses an effective genetic search method to explore the search space and an efficient local search method to exploit information in the search region [Maolin Tang, et al., 2007]. An area utilization improvement technique based on B*-tree is presented by Fubing Mao. The simulated annealing is embedded inside the Tabu search for the floorplan and results into the improvement in area utilization in a short time [Fubing Mao, et al., 2008]. J. Chen, G. Chen, & Wenzhong have proposed a discrete PSO algorithm for Multi-Objective Problem (MOP) for the floorplanning problem. The technique is tested over MCNC benchmark circuits and provide multiple floorplanning schemes for users [Chen J, et al., 2009]. Shanavas & Gnanamurthy have presented a combination of global and local search strategy leading to a hybrid optimization approach. The Memetic Algorithm (MA) involves local search to improve the fitness of individuals in the population to produce optimal or nearly optimal solutions for all the popular benchmark problems [Shanavas H, et al., 2009]. For standard cell placement, a GA-based algorithm has been proposed by Mahajan, Saxena, & Khehra for minimization of chip area. The transformations have been applied to the chromosomal representation of the physical floorplanning instead of transformations on the physical floorplanning [Mahajan R, et al., 2010]. Amrutha & Sundaararajan have presented a faster method for developing wire-optimized floorplan. It has used a trapezoidal algorithm for floorplanning. It achieves lesser wire estimate than SA in orders of magnitude.
An MA-based EA technique for wire length minimization in partitioning and floorplanning have been presented by H Shanavas. Reducing the minimum delay in partitioning and the area in floorplanning helps to minimize the wire length. The technique claims to be 45% faster than simple GA by reducing delay and area and enables to get the near optimal solution [Shanavas H, et al., 2011].

S Subbaraj has used a simulated annealing algorithm based heuristic to target the combinatorial optimization in the VLSI floorplanning problem. The technique is tested on MCNC and GSRC benchmark circuits and proves to be efficient in producing floor plans with a very minimal dead space [Subbaraj S. A, et al., 2012]. The thermal-aware floorplanning framework using GA has been presented by Rani, Rajaram, Nivethitha, & Sudarsan. It minimizes the total area required to accommodate all blocks and reduce the high temp and distribute it evenly across a chip in a framework. The B*tree representations with GA is used to calculate floorplanning based on power dissipation [Rani G. N, et al., 2012]. Hoo, Jeevan, Ganapathy, & Ramiah have used a Variable-Order Ant System, which is combined with a floor plan model, namely Corner List (CL) to optimize the area and wire length [Hoo C. S, et al., 2013].

**Floorplanning problem is about the placement of the modules in their respective places such that the resultant bounding rectangle requires small area and less interconnect length. The EA based optimization techniques are well suited for this class of problems as they allow to optimize a set of modules by forming a compact combination of blocks which can be collectively considered as a hard block. The best property of GA based technique is that it allows carrying forward good characteristics of parents in their child helps in evolving the population faster. This way the final floorplan solution will be a compact solution.**

1.5.4 Literature related to Channel Routing

Rivest & Fiduccia have used a GA based approach for channel routing. A problem specific representation scheme for problem specific genetic operators have been proposed [Rivest R. L, et al., 1982]. Two algorithms for channel routing problem have been proposed by Yoshimura & Kuh which merge nets instead of assigning horizontal tracks to individual nets [Yoshimura T, et al., 1982]. Ho, Vijayan, & Wong have proposed an algorithm for standard cell global routing by considering information about all the nets throughout the global routing process. An approach of constructing a rectilinear Steiner tree of a given set of points in the plane, starting from a minimum spanning tree is presented. The resulting
RST has 8-33% lower cost than MST [Ho J. M, et al., 1989 & Ho J. M, et al., 1990]. A global router has been used by Chiang, Sarrafzadeh, & Wong, where each step consists of finding a tree, called **Steiner min-max tree**, which is a Steiner tree with maximum-weight edge minimized [Chiang C, et al., 1990].

Sechen has used a new algorithm for timing drive placement and routing of rectilinear shaped macro cells. They have used SA and negative feedback scheme which optimizes the relative weighting between primary objective and penalty function [Sechen W. S, et al., 1990]. A new channel routing algorithm, i.e. Montreal based on modified GA have been proposed by Benedetto, Orlando, Sorbello, & Vassallo. The approach presents a high level of parallelism and avoids the local minimum trapping for the algorithm [Benedetto B, et al., 1991]. Brown, Rose, & Vranesic have presented a memetic algorithm based approach to solve the global routing problem, particularly the wire length minimization and reduction of channel capacitances are put under consideration [Brown S, et al., 1992]. A channel routing scheme based on greedy approach has been presented by Cong & Preas. They have tried to maximize the utility of the wiring produced using rule-based techniques [Cong J, et al., 1992]. Kahng & Robins have proposed a technique to iteratively find optimal Steiner points which can be added to the floorplanning. They have compared and found a 2% to 3% wire length reduction over the best previous heuristics [Kahng A. B, et al., 1992].

A routing approach for FPGAs have been presented by Thulasiraman & Lienig have demonstrated and the technique of coarse graph expansion for detailed routing in an FPGA has been described [Leinig J, et al., 1994]. Borah, Michael, & Irwin have presented a new approach for finding a rectilinear Steiner tree of a set of nodes. The comparison of the approach with conventional approaches is done and run time comparison is found to be orders of magnitude better [Borah M, et al., 1994].

A generalized detailed routing problem in segmented channels for row based FPGA have been presented by Bhatia & Shankar. They have used a greedy-based approach to route channels and performs generalized segmented channel routing [Bhatia D, et al., 1996]. Goni, Arslan, & Turton have used a new GA-based router which concurrently optimizes both areas over the cell and the main channel. The GA advances through the solution space by rules and identifies multi-terminal net circuit segments for over the cell routing [Goni B. M, et al., 2000].

A new algorithm called **GRAPE** by Krashinsky have been used. It uses a GA to simultaneously optimize the placement and routing of a circuit is described [Krashinsky
Unveren & Acan have presented an evolutionary assignment-ordering approach to combinatorial optimization problems in VLSI floorplanning design, namely, channel routing [Unveren A, et al., 2003]. An implementation of an efficient minimal tree has been represented by Zhou algorithm. It targets the worst case running time and similar performance as an iterated Steiner algorithm [Zhou H, 2004]. A novel approach to placement and routing in standard cell arrays inspired by geometric constraint usage in traditional CAD systems have been presented by Fudos, Kavousianos, Markouzis, & Tsiato. The routing is performed by a variety of the maze algorithm enhanced by a set of heuristics that have been tuned to maximize performance [Fudos I, et al., 2008].

Channel routing is about proper utilization of available channel space in routing a set of terminals. It can be considered as an optimization problem as the channel space is limited and as routing progresses the available space further reduces. GA based technique helps in effective search of channel space using some heuristics such that resultant solution is an optimal routing.

In this thesis, a set of algorithms has been proposed to address different types of VLSI design problems. This way, it has been targeted to provide a complete package or solution right from fault simulation to testing. By using this set of algorithms, the user or designer can produce a better design. It also helps in achieving good results in less design time.

1.6 OBJECTIVES OF THE RESEARCH WORK

The objectives of the proposed research work are multifold and are divided into multiple phases i.e.:

(i) **Literature survey:** This research work demands an extensive review of literature about the work earlier reported. As the proposed work targets various optimization problems in sequence, so the literature review is an initial step as well as an ongoing process which continues throughout the work.

(ii) **Study and analysis of various Evolutionary Algorithms (EA):** The Evolutionary algorithms are inspired by the natural process of evolution. The EA is a population-based iterative class of algorithms which are chosen because of the wide initial base and the chances of obtaining a relatively better solution are always good.
(iii) **Implementation of EAs for various VLSI design problems using programming language**: Several of the tasks involved in the VLSI design process involve optimization problems which are NP-complete and, therefore, optimization techniques are to be used to obtain solutions. To address the problem, the optimization algorithm has to be coded in a programming language.

(iv) **Comparison and analysis of various evolutionary algorithms**: It involves the comparison of EA techniques on the basis of execution time, final results and optimization constraints with other optimization strategies such as simulated annealing, hybrid algorithms etc.

*The main goal is to understand the class of problems for which different classes of the EA are most suited, and, in particular, the class of problems on which they outperform other search algorithms.*

### 1.7 THESIS ORGANIZATION

The complete thesis is organized in a set of six chapters which are arranged in their order of occurrence in the physical design cycle and testing of manufactured digital circuits.

Chapter 1 is the Introduction, where a discussion about the different techniques has been covered. All algorithms and the design steps have been explained. Also, an exhaustive literature review has been included in it.

Chapter 2 includes the Fault Simulation techniques. Various faults such as *Stuck-at faults, Delay faults, and Bridge faults* are addressed in *combinational and sequential circuits*. As the fault simulation work starts earlier, so it is included as a chapter prior to another chapter of the physical design cycle.

Chapter 3 explains the circuit partitioning problem in detail. It is the first step in optimization after the functional simulation is done. The system is considered as a single entity here. The partitioning problem includes dividing the circuit into bipartition or multi-partition such that the resulting circuit can be grouped in a smaller area with a minimum number of crossover jumps between individual partitions.
Chapter 4 addresses the work carried out in floorplanning. Here, the circuit has been partitioned into a set of modules, i.e. rectangular blocks and the task to position and place them on the canvas of silicon area such that the resulting bounding rectangle has minimum area.

Chapter 5 explains the channel routing problem which follows the floorplanning problem. It explains the different types of routing possibilities such that the minimum interconnect length gets consumed and the number of vias required are also minimum such that the resulting circuit is possible to be routed in given channel space.

Chapter 6 includes the conclusion and future scope. This chapter explains and discusses the approaches and techniques which are used in this work. It also presents the final outcome of the thesis from a circuit point of view and presents the benefits and drawbacks of the approaches. It also gives the future scope of the work and its benefits.

Annexures A and B are included at the end of the thesis; Annexure A includes the different formats of the net lists used in the techniques for optimization algorithms. Annexure B consists of different test circuits used by evolutionary algorithms in the optimization of design parameters.