Chapter 7
Conclusion and Future Lines of Research

7.1 Summary of Contributions

The main focus of this book has been the exploration and optimization of tree-based and mesh-based FPGA architectures. The study's purpose has been to find the ways to improve the overall efficiency of FPGA architecture with or without compromising their principle advantages. In this regard two distinct FPGA architectures have been under consideration: one is the island-style while the other is hierarchical. Mesh-based (island-style) architecture is a known, well explored and thoroughly investigated architecture. Tree-based (hierarchical) architecture, despite its good performance, is relatively less explored FPGA architecture. The two architectures have a lot in common in terms of basic logic and routing resources. However, the global arrangement of logic and routing resources and the detailed interconnect topologies of their switch blocks make them the two distinct architectures.

In this work a major study is carried out on the improvement of logic resource usage in heterogeneous mesh-based and tree-based FPGA architectures. For this purpose separate exploration environments are developed for the two architectures that efficiently place and route certain number of heterogeneous benchmarks on them. Although the primary objective of this work is not to establish the supremacy of one architecture over the other, yet, a detailed comparison between the two architectures is presented to highlight their advantages and disadvantages. Further, to improve the routing resource usage, an exploration of tree-based homogeneous Application Specific Inflexible FPGA (ASIF) is carried out and then its comparison with mesh-based ASIF is performed to evaluate the two ASIFs. Tree-based ASIF is then extended to heterogeneous domain where a detailed exploration of tree-based heterogeneous ASIF and its comparison with mesh-based heterogeneous ASIF is performed.

Some of the major contributions of this book are as follows:

7.1.1 Heterogeneous Tree-Based FPGA Exploration Environment

Chapter 4 presented exploration environments for the exploration of heterogeneous FPGA architectures. The highlight of the work presented in Chap. 4 includes a new environment for tree-based heterogeneous FPGA architecture and an optimized environment for mesh-based heterogeneous FPGA architecture [92]. A significant amount of research work is already done regarding mesh-based heterogeneous architectures, but to the best of our knowledge all the previous work uses predetermined floor-planning technique where hard-blocks are placed in fixed columns. Although this kind of technique can be helpful for an easy and compact layout, it can lead to the wastage of precious logic and routing resources and hence increased area. This work presents an exploration environment for mesh-based FPGA that can optimize automatically the floor-planning of the FPGA architecture for a given set of applications.

A number of techniques are explored for both mesh-based and tree-based architectures using their respective environments. The techniques of the two architectures are then evaluated using the results that are obtained by mapping 21 benchmarks on the two architectures. In order to have a profound analysis of the techniques, these benchmarks are carefully selected to cover different inter-block communication trends. The results obtained after the experimentation suggest that, for a mesh-based architecture, the floor-planning technique based on the movement and rotation of logic and hard-blocks gives the best results and is much better than the one where hard-blocks are fixed in columns (i.e. the technique normally used in mesh-based architectures). For 21 benchmarks, on average, a column based floor-planning takes 19% more area, crosses 8% more switches on critical path, consumes 13% more memories and 20% more buffers than the best non-column floor-planning. Further, the comparison between different techniques of mesh-based and tree-based architecture shows that, on average, the best technique of tree-based architecture is 8.7% more area efficient, crosses 60% less switches on critical path, consumes 11% less memories and almost same number of buffers than the best non-column based technique (i.e. technique based on movement and rotation of logic and hard-blocks) of mesh-based architecture. Also, the best technique of tree-based FPGA is 22% more area efficient and crosses 62% less switches on critical path than the equivalent column-based technique of mesh-based architecture. These results are averaged for 21 benchmarks which cover different aspects of heterogeneous benchmarks.

7.1.2 Tree-Based ASIF Exploration

Chapter 5 presented a new tree-based homogeneous Application Specific Inflexible FPGAs. If a digital product is required to provide multiple functionalities at exclusive times, each distinct functionality represented by an application circuit is efficiently mapped on an FPGA. Later, unused resources of the FPGA are removed to generate an ASIF.

Conventional partitioning/placement and routing algorithms are designed for individual netlists and they do not take into account the inter-netlist optimization. In this work, however, we have modified these algorithms which has led to the exploration of four ASIF generation techniques for tree-based architecture. These techniques are divided on the basis of how each technique uses logic and routing resources of the architecture. ASIF-NPNR, first of the four techniques, uses normal partitioning/placement and routing algorithms that are employed in FPGAs and this technique in general produces the worst results among the four techniques. ASIF-EPNR employs efficient logic sharing technique to optimize the use of logic resources among different netlists but uses normal routing algorithms. This technique produces slightly better results when compared to ASIF-NPNR, but produces poor results compared to other two techniques. ASIF-NPER is the third among the four techniques. It uses no efficient logic sharing but employs efficient routing resource sharing and results in a significant improvement over the first two techniques. ASIF-EPER is the fourth and the most efficient technique as it uses both efficient logic sharing and efficient routing and produces the best over all results.

Four ASIF generation techniques are explored for a set of 16 MCNC benchmarks and comparison of these techniques with tree-based FPGA shows that the most efficient ASIF generation technique (i.e. ASIF-EPER) is 64% more area efficient than an equivalent tree-based FPGA. Further the exploration of LUT and arity size on tree-based ASIF shows that smaller LUTs with larger arity sizes give better area results whereas larger LUTs with larger arity sizes give better performance results. The area-delay product of different LUT-arity combinations reveals that an ASIF with LUT size 4 and arity size 16 produces the most efficient results.

Later the comparison between the best techniques of tree-based and mesh-based ASIFs is performed. For tree-based ASIF, LUT 4 arity 16 combination is used while for mesh-based ASIF LUT 4 is used. The two ASIFs are compared over a range of varying signal bandwidth and the results show that the best tree-based ASIF is 12%, 77% more efficient than the best mesh-based ASIF in terms of routing area and number of wires used.

The quality analysis of ASIF generation method is performed by generating an ASIF for a group of netlists for which an ideal ASIF solution is known. So, an ASIF is generated for same set of netlists. The quality analysis of tree-based ASIF reveals that ASIF-EPER produces much worse results than ideal solution. The major cause of these results is the heuristic based partitioning algorithm. New partitioning techniques need to be explored to further improve area of a tree-based ASIF. Further a quality comparison between mesh-based and tree-based ASIFs reveals that for a group of 10 similar netlists, tree-based ASIF is 33% more area efficient than mesh-based ASIF.

In Chap. 6, a new tree-based heterogeneous ASIF is presented. Similar to tree-based homogeneous ASIF, four ASIF generation techniques are explored for 17 open core benchmarks. Based on the types of hard-blocks used, benchmarks are divided into two sets where first set contains 8 benchmarks each of which contains a mixture of multipliers and logic blocks (SET I) and second set contains 9 benchmarks where each benchmark contains a mixture of multipliers and adders along with logic blocks (SET II). Comparison of different techniques with tree-based FPGA shows that the most efficient ASIF generation technique (i.e. ASIF-EPER) is 64%, 75% more area efficient than an equivalent tree-based FPGA for SET I and SET II respectively. Further the exploration of LUT and arity size on tree-based ASIF shows that an ASIF with LUT size 4 and arity size 16 produces the most efficient area-delay results for SET I and an ASIF with LUT size 3 and arity size 16 produces the most efficient area-delay results for SET II. Comparison of tree-based ASIF with mesh-based ASIF reveals that the best tree-based ASIF is 11.27% better and 1.5% worse than the best mesh-based ASIF in terms of area for SET I and SET II respectively. Further the wire comparison of mesh-based and tree-based ASIFs shows that tree-based ASIF consumes 69%, 70% less wires than mesh-based ASIF for SET I and SET II. Finally the quality analysis of two architectures reveals that for a group of 10 similar netlists, tree-based ASIF is 19%, 20% more area efficient than mesh-based ASIF for SET I and SET II benchmarks respectively.

7.1.3 FPGA and ASIF Hardware Generation for Tree-Based Architecture

Chapters 3 and 5 presented new FPGA and ASIF hardware generation models for tree-based architecture. These two chapters also included the description for hardware generation of mesh-based architecture. The FPGA and ASIF hardware generation models are similar, however, necessary modifications are made in the two models to meet the respective needs of FPGA and ASIF architectures. The hardware generator is integrated with the exploration environment. So the major benefit is that all the architecture level changes in the exploration environment are directly translated in VHDL model. The VHDL model of an FPGA and ASIF is generated for multiple netlists. Layout is performed for FPGAs and ASIFs using 130 nm 6-metal layer CMOS process of ST Microelectronics.

Chapter 6 presented the layout comparison results between tree-based heterogeneous ASIF and sum of ASICs. The results showed that for a group of 9 netlists (SET II), sum of ASICs is 40% smaller than ASIF. However, the area of ASIF is affected by the use of LATCH cells instead of SRAMs and the gap between ASIF and sum of ASICs can be reduced by replacing LATCH cells with SRAMs. Further the area of ASIF can be improved by performing full-custom design of repeatedly used cells in the architecture.

7.2 Suggestions for Future Research

In this work, we explored a number of techniques for heterogeneous tree-based and mesh-based FPGA architectures. Also we improved the density of FPGA architectures by customizing them for a pre-determined set of applications. However, there are a number of aspects that are unexplored yet and a thorough study is required to explore those aspects and in turn further improve the efficiency of FPGA architectures. Some of the suggestions for future research are described below. These suggestions are applicable to both mesh-based and tree-based architectures unless otherwise specified.

7.2.1 Datapath Oriented FPGA Architectures

During past few years FPGAs have seen a rapid growth in their logic capacity which has led to the increasing use of FPGAs for the implementation of arithmetic-intensive applications. Arithmetic-intensive applications often contain large portion of datapath circuits. Datapath circuits usually contain hard-blocks (e.g. multipliers, adders, memories etc.) that are connected together by regularly structured signals called buses. Conventional FPGAs do not use the regularity of datapath circuits. So it is possible to modify the conventional FPGA architectures to exploit the regularity of datapath circuits and achieve significant area savings.

Although some work has been done in this regard for mesh-based homogeneous architecture [127], but to the best of our knowledge no work has been done yet in heterogeneous domain neither for mesh-based nor for tree-based architecture. Both heterogeneous mesh-based and tree-based FPGA architectures can be modified to exploit the regularity of datapath circuits and improve the area efficiency. Generalized proposition of a datapath oriented tree-based architecture is shown in Fig. 7.1. Contrary to the conventional one, the proposed version could be divided into two sub-structures. One sub-structure contains CLBs while other sub-structure contains only HBs. Interconnect between the two sub-structures and inside the sub-structure containing only CLBs is fine-grain while it is coarse-grain (i.e. bus-based) in the sub-structure containing only HBs. Main motivation behind the migration from single structure to two sub-structures is:

1. Most of the arithmetic-intensive applications have two parts: datapath part and the control part. Control part is implemented using CLBs while datapath part is implemented using HBs. So it is better to manage them separately due to their different communication behaviors.
2. Division of single structure into two sub-structures helps optimizing their logic and routing resources independently; thus leading to better logic and routing density of the architecture.

Preliminary experimentation has shown some promising results and for a group of four datapath circuits, the datapath oriented tree-based architecture consumes

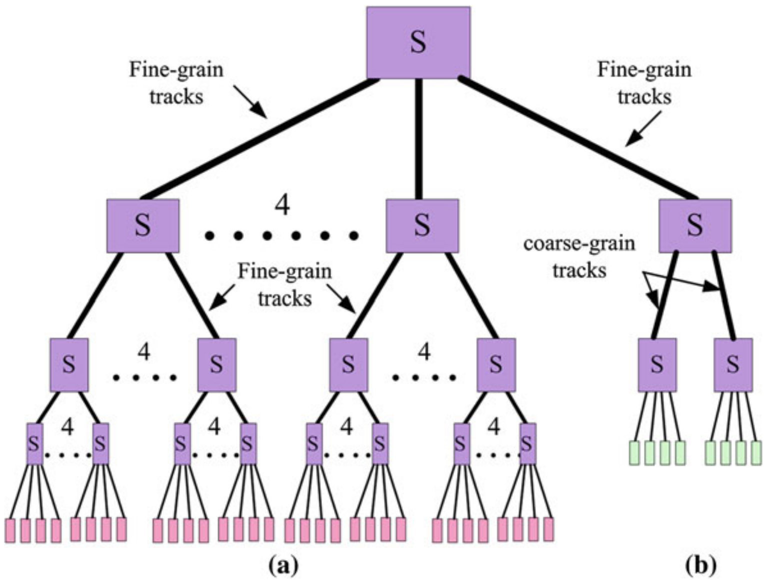

Fig. 7.1 Generalized example of datapath oriented tree-based FPGA architecture. **a** Sub-structure containing only CLBs. **b** Sub-structure containing only HBs

21% less area than the conventional tree-based heterogeneous FPGA architecture. However, these are only preliminary results as the benchmarks are small both in size and in number and improvements regarding resource partitioning are required to further enhance the results.

As far as the mesh-based heterogeneous FPGA architecture is concerned, modification can be made in it in a manner similar to the one suggested by [127]. However, there might be few exceptions as the architecture proposed by [127] uses only homogeneous blocks.

7.2.2 Timing Analysis

One of the major future work is to perform accurate timing analysis of both FPGA and ASIFs. In this work we presented timing results based on the count of critical path switch number. Although this is not accurate as it does not contain the wire delays, it gives an idea about the timing behavior of the architecture. In future we want to perform the complete timing analysis of both mesh-based and tree-based architecture. This timing analysis will include measuring the critical path, optimizing the critical path using timing driven routing, and finding an accurate compromise between area and delay of an architecture.

7.2.3 Integrating ASIF Blocks in an FPGA Architecture

Commercial FPGA vendors use different variety of hard-blocks in their FPGAs. They provide a range of FPGA device variants to fulfill varying domain-specific requirements of their customers. Smaller ASIF blocks can also be integrated in FPGAs to enhance the domain-specific needs of customers. An ASIF block serves as a multi-tasking hard-block that can support a number of different functionalities at mutually exclusive times. ASIF blocks can be designed for some general purpose DSP requirements, or for more specific applications such as video decoders/encoders applications etc.

7.2.4 Further Optimizing the ASIF Generation

In this work a new tree-based ASIF is presented and its comparison with equivalent tree-based FPGA shows that it gives significantly better area results than tree-based FPGA. Tree-based ASIF is basically an intermediate solution between FPGAs and ASICs where a certain amount of flexibility is retained while removing unnecessary resources. Although the removal of unnecessary resources in ASIF gives better area than FPGA, it makes the structure irregular; hence making the layout of ASIF more difficult than FPGA. So from this point on, the possible research directions in ASIF architecture can be as follows (these directions are general in nature and they are applicable to both mesh-based and tree-based ASIFs unless otherwise specified):

- One future direction for ASIF can be to improve their efficiency by further optimizing their logic and routing resources. In Chap. 6, the comparison between tree-based ASIF and Sum of ASICs revealed that tree-based ASIF consumes more area than Sum of ASICs. This is because of the fact that tree-based ASIF uses LUTs and they remain as configurable as in FPGA. So, the possible optimizations in the ASIF generation can be as follows:

  1. Different set of application netlists, mapped on an ASIF, program the SRAM bits of a LUT differently. If all the netlists program a particular SRAM of a LUT in a similar fashion, that SRAM bit can be replaced by a hard-coded 0 or 1 eventually leading to further improvement in the area of ASIF. Similarly, if a 2-input multiplexor in a LUT receives similar hard-coded values, that multiplexor can be removed, and replaced by the hard-coded value. A major element of research is to tailor the LUT configuration of different netlists in such a way that maximum logic block optimization is achieved.
  2. Also, the routing network of ASIFs contains SRAMs that are used to provide limited configurability. However, this configurability is not much of a use as tools do not guaranty the mapping of extra circuits. So, the routing network of ASIF can be further optimized as there is a good probability that certain SRAMs of the routing network have same value over all the circuits that are required

to be implemented on ASIF. In such a case these SRAMs can be replaced with constants; hence resulting in further improvement of area results.

- The generation of ASIF is based on the removal of unused resources which makes it irregular and tile-based layout of ASIF does not remain possible. So, another future direction on ASIF can be to add more flexibility to it than required. The main objective of this additional flexibility would be to have a minimum tile or a set of tiles that can be repeated to perform the layout of ASIF. Although additional flexibility will reduce the area gain of ASIF compared to FPGA, it will make the layout easier; hence resulting in faster time to market. Further, it will increase the probability of implementing additional circuits that are not in the set for whom the ASIF is generated.
- Quality analysis of ASIF in Chaps. 5 and 6 revealed that there is significant room for improvement in the placement/partitioning algorithms. This improvement can be achieved by better exploiting the inter-netlist dependance. However, in this approach, a major element of research would be to optimize the inter-netlist dependance without compromising the intra-netlist optimization.
- In Chap. 6, area comparison between ASIF and sum of ASICs revealed that even for 1 netlist ASIC is 20% smaller than ASIF. This is because of the fact that in case of ASIFs, the netlist synthesis involves a number of open source tools. These tools are not optimal and they add some imperfections while converting the netlist from its hardware description to .net format. On the contrary, ASIC generation involves only one commercial synthesis tool. So, the deficiencies introduced by open source tools play an important role in the area gap between ASIF and sum of ASICs and this gap can be reduced by performing the netlist synthesis using more efficient commercial tools.

The above mentioned optimizations an be combined together to give either an ASIF with better area density; hence reduced gap between ASIF and ASIC or an ASIF with reduced layout efforts; hence faster time to market.

7.2.5 The Unexplored Parameters of Mesh-Based Architecture

In this work we have used single length, unidirectional routing network with CLB size 1 for mesh-based architecture which is at the core of both FPGA architecture and ASIF architecture. However, it would be interesting to explore the behavior of mesh-based ASIF having CLB size greater than 1, or having a mixture of different length routing wires some of whom span a single architecture tile whereas others span multiple tiles or having a mixture of fine-grain and coarse-grain routing wires.