IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 13, Issue 4 (Jul. - Aug. 2013), PP 01-06
www.iosrjournals.org

A Review of FPGA-based design methodologies for efficient hardware Area estimation

Ms.Rachna Singh[1],Dr.Arvind Rajawat[2]
(Department of Electronics & communication Engg SISTEC-E,Bhopal,India)
(Department of Electronics & communication Engg.MANIT,Bhopal,India)

Abstract: *In recent years, FPGA's have become increasingly important and have found their way into system design. So, the desire emerges for a means that allows early area and performance estimation Understanding how a design maps to them and consumes various FPGA resources can be difficult to predict, so typically designers are forced to run full synthesis on each iteration of the design. For complex designs that involve many iterations and optimizations, the run-time of synthesis can be quite prohibitive .However, to achieve high performance; FPGA must be supported by efficient design methodology and optimization techniques. The motivation behind this work is to review different FPGA based design methodology and optimization techniques that can be employed to efficiently estimate hardware area utilized in terms of look up table (LUT'S) or configurable logic blocks (CLB'S).*

Keyword:*HW/SW Partitioning, Area Estimation, Latency Estimation*

I. Introduction

In recent years, FPGA's have become increasingly important and have found their way into system design. So, the desire emerges for a means that allows early area and performance estimation. The benefit of such an aid is two fold: On one hand it allows to roughly quantify an FPGA design and therefore enables early comparison to traditional approaches. On the other hand, it supports especially less experienced designers and thus, can help to make FPGA's more popular and regarded. Several approaches for estimating area and performance perimeter of FPGA design's have been discussed in the given literature review. The estimation results are intended to give an idea of the implementation characteristics and to enable early design space exploration and trade-off consideration. As a benefit the tedious run through the standard FPGA design flow (synthesis, mapping, placement and routing) are minimized.

FPGA's have become quite diverse in the types of applications that utilize them. These applications can include everything from real time video processing and other complex digital signal processing (DSP) functions: to various forms of soft instruction set processors for control based applications; to communication and networking designs. In order to target these diverse applications, the resources available on these FPGA platforms have also become quite diverse in their own right. Modern FPGA's contain not just basic , programming building blocks including look-up tables(LUT's),flip-flop's(FF's) and additional carry logic , but more complex on-chip blocks as well, such as block memories(eg.block RAM'S) DSP elements(eg: multipliers) and high speed transceivers[1].

It is important for designer that uses these diverse, heterogeneous FPGA's to understand the consequences of decision made during the process of capturing the design in a synthesizable manner. As pointed out in [1], FPGA resource usage is an important measure of hardware cost (besides path delay and power consumption). The sooner the designer is aware of the hardware impact of coding decision, the sooner he can make any necessary improvements and correction's before they become hidden in a large design implementation. Performing full synthesis at each design iteration to obtain hardware resource estimation can become quite time consuming, especially for large complex designs. A fast method to obtain detailed hardware resource estimation is essential in modern FPGA design. An excellent survey of the hardware characteristic estimation techniques is presented in [3].

For the area estimation, some techniques are tailored for certain partitioning schemes [4, 5]. Area estimation for different input description languages is widely studied(C[3,6,7],SA-C[8],SYSTEM C[9] ,MATLAB[10] SIMULINK[11],VHDL[2]……etc).Most of the published work performs a transformation step to express the input description into an intermediate representation (IR) such as Trimaran IR[6],control data flow graph(CDFG)[7] and VHDL-AST[2], and then apply the estimation process on the intermediate format.

This paper presents a review of FPGA-based design methodology and optimization techniques used for efficient hardware area estimation .The remainder of this paper is organized as follows section II describes the steps used in FPGA design methodology .In section III various techniques that are used to achieve efficient FPGA-based hardware realization are further reviewed and discussed. Finally section IV presents the concluding remarks.

II. Fpga Design Methodology

FPGA design methodology as shown in fig. 1 is used as a guideline for the hardware realization of algorithms. The first step in FPGA design methodology is to capture the algorithm to be implemented on FPGA using hardware description languages (HDLs) or schematic depending on the complexity of the design.

After specifying the design using HDLs or Schematic, the designer needs to validate the logical correctness of the design. This is performed using functional or behavioral simulation. Designers usually go through this step right after they finish the coding and logic synthesis. Logic synthesis converts HDL or schematic-based design into a netlist of actual gates/blocks specified in FPGA devices. After logic synthesis, technology mapping is done.

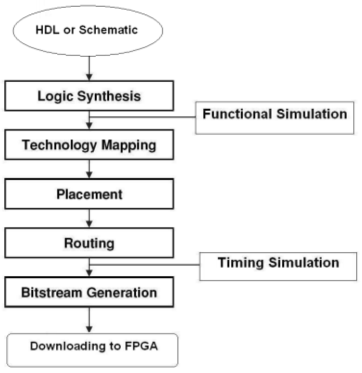

Figure 1. FPGA Design Methodology

In this step, the tool transforms a netlist of technology independent logic gates into one consisting of logic cells and input/output blocks (IOBs) in the target FPGA architectures [12,13].

Placement which follows technology mapping selects the optimal position for each block in a circuit. The basic goal of an FPGA placement is to locate functional blocks such that then interconnect required to route the signals between them is minimized. A good placement is extremely important forFPGA designs. It directly affects the routability and the performance of a design on FPGA [14]. A poor placement will lead to lower maximum operating speed and increased power consumption. FPGA placement algorithms can be broadly classified as routability-driven and timing-driven [15].

The main objective of routability-driven algorithms is to create a placement that minimizes the total interconnect
required. In addition to optimizing for routability, timing algorithms use timing analysis to identify critical paths and/or connections and optimize the delay of those connections.

Routing is the last step in the design methodology prior to generating the bitstream to program the FPGA [16-18]. FPGA routing is a tedious process because it has to use only the prefabricated routing resources such as wire segments, programmable switches and multiplexers [17]. Hence it is a challenging task to achieve 100% routability in FPGAs.

After placement and routing, timing simulation is performed to validate the logical correctness of the design taking into account the delays of the FPGA device. Power consumed by a design is further estimated by doing the power analysis such as XPower and PowerPlay tools used in XilinxISE and Altera Quartus II tools respectively.

The final step in the FPGA design methodology is bitstream generation. It takes the mapped, placed and routed design as input and generates the necessary bitstream to program the logic and interconnects to implement the intended logic design and layout on the target device.

III. Area Estimation Techniques

Area estimation technique employed at the design as well as the implementation phase play a significant role in realizing efficient FPGA resources. Fast and accurate resource estimation technique for an FPGA-based design is essential for the efficient utilization of the hardware resources in any design. In FPGA based design the hardware area utilized is provided in terms of look-up table (LUT's) or configurable logic blocks (CLB's) slices. However for comparison of design based on similar FPGA devices, all the resources must be considered. Some of the most commonly used FPGA resources are:-

- No of 4-input LUT's
- No of Slices
- No of slice Flip-flop
- No of IOB's

A design utilizing dedicated resources of modern FPGA such as embedded multiplier or DSP blocks will consume less logical resources(LUT's and CLB slices) as compared to design that implements the functionality without using dedicated resources.
The following sub-sections present different design techniques for estimating area, latency etc. for the FPGA based design. These techniques can be grouped together according to certain parameters like: Resource sharing, optimizing speed etc.

3.1) Resource Sharing:

Resource sharing is an optimization technique that uses a single functional block to implement several operates in the HDL code. Since resources on an FPGA are limited, the designer must spend more effort on resource sharing.

Resource sharing adds additional logic levels to multiplex the Inputs to implement more than one function. Therefore, it is not recommended to use it for authentic functions that are part of the designs time critical path [19].

Peter A.Miider et al [22] present an equation based resource utilization model for automatically generated DFT soft core IPS. The parameterized DFT IP generator allows a user to make customizedtrade off between cost and performance and between utilization of different resource classes.

***3.2) Proper Reset Strategy*:**

For FPGA architectures, the use of a reset and the type of reset can have serious
implications on the design performance. An improper resetstrategy can create an unnecessarily large design and stallcertain area optimizations. Sub-optimal reset strategies can:
• prevent the use of a device library component, such as shift register look-up table (SRL)
• prevent the use of synchronous elements of dedicated hardware blocks
• prevent optimizations of the logic inside the fabric
• Severely constrain placement and routing becausereset signals often have high fan-out
For Xilinx FPGAs, avoid using resets on shift registersbecause it prevents inference of area and performanceoptimized SRL library cells. If we use reset, the function will be implemented with generic logic resources and will occupy more area. Similarly, it is recommended to avoid using asynchronous reset because it prevents packing of additionalregisters into dedicated resources [20].

In Xilinx FPGAs, Block RAM (BRAM) elements havesynchronous resets only. Therefore, if we use synchronous reset, the synthesis tool will be able to implement the codewith a single BRAM element. However, if we implement thesame RAM with an asynchronous reset, the synthesis tool willbe forced to use smaller distributed RAM blocks, additionaldecode logic to create the appropriate size RAM, andadditional logic to implement the asynchronous reset [20]. For area optimal design, it is recommended to avoid set and resetwhenever possible.

3.3) Optimizing speed:

For complex designs that involve many iterations and optimizations the run-time of synthesis can be quite prohibitive. So we need a fast method of estimating the FPGA resources of any design.

Paul et al[22] presents a fast and accurate method of estimating the FPGA resources of any RTL-based design. In this work the design is not actually mapped to the FPGA hardware rather only modeling the steps that synthesis is expected to take. The tool provides estimation within 30 seconds for typical designs.

M.B.Abdelhalim et al [23] presents a fast and accurate area and latency estimation tool for FGPA-based design. It is developed in the context of a hardware /software partitioning tool. Rather than modeling the hardware implementation as a single alterative, it models the hardware as two extreme alternatives that bound latency range for different hardware implementations. Area estimations are within 7.5% of the actual no of logic elements consumed with an average ever of 3.2% for stratix FGPA's.

Frank vahid et al [5] introduces a technique for obtaining the estimation of hardware size in two orders of magnitude less time without scarifying substantial accuracy, by incrementally updating a design model for a changed partition rather than re-estimating entirely.

D.Kulkarni et al[8] presents a fast compile-time area estimation technique to guide the compiler optimizations. The estimation time is in the order of milliseconds as compared to several minutes for a synthesis tool. The tool not allow application programmers to directly implement their algorithm into hardware, they also provide a tremendous opportunity for the compiler to perform extensive optimizations.

Per Bjures et al[9] presents a simulation –based technique to estimate area and latency of an FPGA implementation of a Matlab specification. During simulation of the Matlab model,a trace is generated that can be used for multiple estimations. The run time of the estimator is approximately only 1/10 of the simulation time, which is typically fast enough to generate dozens of estimates within a few hours and to build cost-performance trade-off curves for a particular algorithm and input data. The estimator also reports on the scheduling and resource binding used for estimation. From this trace, an acyclic data flow graph is derived. The operations of the DFG are scheduled and bound to FPGA resources by way of a greedy scheduling and binding algorithm.

3.4) Target technology:

Regarding the target technology, current approaches target either ASIC-based designs [31,32] or FPGA based designs [6-11,24-30] FPGA-based area estimates either incorporate a physical model for the FPGA and estimate the area by performing actual Mapping [33], by using modeling equations of the FPGA function al resources [6-11] or building a large database for all possible resources configurations [34]

M.B.Abdelhalim et al [23] selectedFPGA’s as target platform. Even though tool is specific for the Altera 4-inputs LUT-based FPGA’s, the approach could be easily adapted for use with a variety of other FPGA’s. In [35] the provided estimations are limited to Altera stratix family specific resources, i.e. pre-fabricated hardwired multiplies and thus ignored the estimation of configurable logic –based ovary and constant multipliers.

Leipo yen et al [36] presents an estimation model for the coarse grained reconfigurable architectures implemented on FGPA platform. Ohm et al [37] developed an estimation technique for predicting the area required to implement a behavioral description for a given performance goal. The architecture they considered is standard cell based ASIC.

kulkrarian et al [38] proposed on estimation technique to estimate the FPGA area consumption of the data flow graphs(PFG’s)from applications. The technique in the DFG’s into different categories and developed a formula for each category to estimate the area.

V. Conclusion

Several factors that play a significant role in FPGA-based design include:-proper selection of FPGA architecture, design methodology & optimization techniques etc. In this paper a review of FPGA-based design methodologies used for efficient hardware area estimation have been presented. For the area estimation some techniques are tailored for certain partitioning schemes &some are for different input description languages like C,SA-C,System C, Simulink, VHDL etc. Most of the techniques detect the resource sharing opportunities through scheduling. FPGA based area estimators either incorporate physical model for the FPGA and estimate the area by performing actual mapping, or by using modeling equations of the FPGA functional resources.

Table 1.Comparison chart of different FPGA –design based research methodologies

S.no	Author/ Reference paper	Area Estimation Time	Area Estimation Accuracy	Approach	latency	Resource Sharing	Scheduling	Optimization	Remark
1	M.B.Abdelhalim et al [23]	One sec	+-7.5% of the actual no of Logic element	VHDL description	√	√	ASAP scheduling	√	*curve fitting approach
2	Paul et al[22]	30 sec(60 times faster than synthesis run time	22% of the actual mapped slices	RTL based	-	√	-	√	*Macro level estimation *curve fitting eq.
3	Peter et al[21]	Sub micro sec	6.1%	DFT soft core IP	√	-	-	√	*Equation based model for estimating slice usage
4	Frank Vahid et al[5]	1.5 hrs	comparable	VHDL behavioral description	-	-	-	-	*control unit/Data path model
5	Per Bjureus et al[9]	1/10 of simulation time	+-10%	MATLAB specification	√	√	Greedy algorithm	-	*simulation based technique *Trace used for multiple estimation
6	Chi et al[1]	Sec to min	10% of post mapping	SIMULINK model	-	-	-	-	*Pre-netlisting tool in system generator *MATLAB function’s used
7	D .Kulkarni et al[8]	millisec	*2.5% for small image processing operator *5% for larger benchmarks	High –level SA-C code	-	√	-	√	*Complie-time estimation technique
8	Pablo de Marugan et al[39]	second	-	High level specification(C or C++)	-	√	Dynamic Scheduling	-	*“IIVM” framework used
9	Michael Kunz et al[40]		28.61x10^{3}					word length optimization	*LUT estimation

References

[1] Shi, C., Hwang, J., McMillan, S., Root, A., and Singh, V.,"A System Level Resource Estimation Tool for FPGAs",*International Conference on Field Programmable Logic andApplications (FPL)*, 2004.

[2] C. Brandolese, W. Fornaciari, and F. Salice. "An AreaEstimation Methodology for FPGA Based Designs at SystemCLevel,"*DAC'04*, San Diego, California, USA, pp. 129 - 132, 2004.

[3] RoelMeeuws, "A Quantitative model for Hardware/Software partitioning," MSc thesis, *Delft University of Technology*, 2007.

[4] V. Srinivasan, S. Govindarajan, and R. Vemuri, "Fine-grained and coarse-grained behavioral partitioning with effective utilization of memory and design space exploration for multi-FPGA architectures," *IEEE Transactions on Very Large Scale Integration(VLSI) Systems*, Vol. 9, no. 1, pp. 140–158, 2001.

[5] F. Vahid and D. D. Gajski, "Incremental hardware estimation during hardware/software functional partitioning," in *Readings in hardware/software co-design*, G. De Micheli, R. Ernest, and W.Wolf (eds.), Morgan Kaufmann, pp. 516–521, 2002.

[6] L. Yan, T. Srikanthan, and N. Gang, "Area and Delay Estimation for FPGA Implementation of Coarse-Grained Reconfigurable Architectures," *LCTES*, Ottawa, Ontario, Canada,pp.182–188, 2006.

[7] R. Enzler, T. Jeger, D. Cottet, and G. Troster, "High level area and performance estimation of hardware building blocks on FPGAs," *FPL 2000*, Villach, Austria, pp. 525–534, 2000.

[8] D. Kulkarni, Walid A. Najjar, R. Rinker and F. J. Kurdahi,"Compile-Time Area Estimation for LUT-Based FPGAs," *ACMTODAES*, Vol. 11, No. 1, pp. 104–122, 2006.

[9] P Bjureus, M. Millberg, and A. Jantsch, "FPGA resource and timing estimation from MATLAB execution traces," *CODES 2002,*Estes Park, Colorado, USA, pp. 31–36, 2002.

[10] L. M. Reyneri, F. Cucinotta, A. Serra, and L. Lavagno, "A hardware/software co-design flow and IP library based on Simulink," *DAC '01*, Las Vegas, Nevada, USA, pp. 593 - 598,2001.

[11] A. Nayak, M. Haldar, A. Choudhary, and P. Banerjee,"Accurate area and delay estimators for FPGAs,"*DATE'02*, Paris,France, pp. 862-869, 2002.

[12] V. Manohararajah, S. D. Brown, and Z. G. Vranesic, "Heuristics for area minimization in LUT–based FPGA technology mapping," *IEEETrans.on Computer Aided Design of Integrated Circuits and Systems*,Vol. 25, No. 11, pp. 2331–2340, Nov. 2006.

[13] A. Mishchenko, S. Chatterjee, R. K. Brayton, "Improvements to Technology Mapping for LUT–Based FPGAs," *IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems* , Vol. 26,No. 2, pp. 240–253, Feb. 2007.

[14] W. K. Mak and L. Hao, "Placement for modern FPGAs," *Proc. of Emerging Information Technology Conf.*, Aug. 2005, pp.1–4.

[15] A. Marquardt, V. Betz, and J. Rose, "Timing–driven placement for FPGAs," *Proc. of the ACM/SIGDA Intl. Symp.on Field ProgrammableGate Arrays*, Feb. 2000, pp. 203–213.

[16] M. J. Alexander and G. Robins, "New–performance driven FPGA routing algorithms," *IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems*, Vol. 15, No. 12, pp. 1505–1517, Dec.1996.

[17] G. J. Nam, F. Aloul, K. A. Sakallah, and R. A. Rutenbar, "AComparative study of two boolean formulations of FPGA detailed routing constraints," *IEEE Trans. on Computers*, Vol. 53, No. 6, pp.688–696, June 2004

[18] http://www.tutorial-reports.com/computer-science/fpga/routing.php

[19] Xilinx Inc., *Synthesis and Simulation Design Guide*. June 2008.

[20]. P. Garrault and B. Philofsky, "HDL Coding Practices to Accelerate Design Performance," *Xilinx White Paper*, wp231, pp. 1–22, Jan 2006.

[21] Peter A. Milder, Mohammad Ahmad, James C. Hoe, and Markus P¨uschel, "Fast and Accurate Resource Estimation of Automatically Generated Custom DFT IP Cores"Proc. ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA), Monterey, California, USA.pp. 211-220, 2006.

[22] Paul Schumacher and PradipJha, "Fast and Accurate Resource Estimation of RTL-based designs targeting FPGA's",IEEE,2008.

[23] M .B. Abdelhalim and S. E. -D. Habib, "Fast FPGA-based area and latency estimation for a novel hardware/software partitioning scheme" IEEE,2008

[24] M. Xu and F. J. Kurdahi, "Accurate prediction of quality metrics for logic level design targeted toward lookup-table-based FPGAs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 7, No. 4, pp. 411–418, 1999.

[25] T. Jiang, X. Tang, P. Banerjee, "Macro-models for High Level Area and Power Estimation on FPGAs," *GLSVLSI'04*, Boston, Massachusetts, USA, pp 162-165, 2004.

[26] D. Chen, J. Cong, Y. Fan, and Z. Zhang, "High-Level Power Estimation and Low-Power Design Space Exploration for FPGAs,"*ASPDAC'0,* Yokohama, Japan, pp. 529-534, 2007.

[27] Yang Qu and J.-P. Soininen, "Estimating the utilization of embedded FPGA coprocessor," *DSD '03*, Oulu, Finland, pp. 214-221, 2003.

[28] M. Vootukuru, R. Vemuri and N. Kumar, "Partitioning of Register-Level Designs for Multi-FPGA Synthesis," *VIUF'96,*Santa Clara, California, USA, pp 99-106, 1996.

[29] F. Vahid, T. Dm Le, and Yu-Chin Hsu, "Functional Partitioning Improvements over Structural Partitioning for Packaging Constraints and Synthesis: Tool Performance," *ACM TODAES,* Vol. 3, No. 2, pp. 181–208, 1998.

[30] C. Menn, O. Bringmann, and W. Rosenstiel, "Controller Estimation for FPGA Target Architectures during High-Level Synthesis," *ISSS'02*, Kyoto, Japan, pp. 56-61, 2002.

[31] R. L. Ernst and J. Henkel, "High-level estimation techniques for usage in hardware/software co-design," *ASPDAC '98,* Yokohama,Japan, pp. 353–360, 1998

[32] M. Nemani and F. N. Najm, "High-level area and power estimation for VLSI circuits," *ICCAD '97*, San Jose, California,USA, pp. 114–119, 1997

[33] M. Xu and F. J. Kurdahi, "Accurate prediction of quality metrics for logic level design targeted toward lookup-table-based FPGAs," *IEEE Transactions on Very Large Scale Integration (VLSI)Systems*, Vol. 7, No. 4, pp. 411–418, 1999

[34] M. Vootukuru, R. Vemuri and N. Kumar, "Partitioning of Register-Level Designs for Multi-FPGA Synthesis," *VIUF'96,*Santa Clara, California, USA, pp 99-106, 1996

[35] D. Chen, J. Cong, Y. Fan, and Z. Zhang, "High-Level Power Estimation and Low-Power Design Space Exploration for FPGAs," *ASPDAC'0,* Yokohama, Japan, pp. 529-534, 2007.

[36] Leipo Yan, ThambipillaiSrikanthan, Niu Gang,"Area and Delay Estimation for FPGA Implementation ofCoarse-Grained Reconfigurable Architectures",LCTES'06, Ottawa, Ontario, Canada,June 14–16, 2006.

[37] S. Y. Ohm, F. J. Kurdahi, N. Dutt, and M. Xu.A comprehensive estimation technique for high-level synthesis. In *Proceedings of the 8th international symposium on System Synthesis*, pages 122–127,1995.

[38] D. Kulkarni, W. A. Najjar, R. Rinker, and F. J. Kurdahi. " Fast area estimation to support compiler optimizations in fpga-based reconfigurable systems". In *Proceedings of the 10th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 239–247, 2002.

[39] Pablo González de Aledo Marugán, Javier González-Bayón and Pablo Sánchez Espeso, "Hardware performance estimation by Dynamic Scheduling" ;in Proc. FDL, 2011, pp.1-6.

[40] Michael Kunz, Martin Kumm, Martin Heide, Peter Zipf, "Area Estimation of Look-Up Table Based Fixed-Point Computations on the example of a Real-Time High Dynamic Range Imaging System"In 22nd International conference on Field Programmable Logic and Applications(FPL),pages 591-594,Aug 29-31 2012