sciendo INTERNATIONAL JOURNAL ON SMART SENSING AND INTELLIGENT SYSTEMS

Research Article | DOI: 10.2478/ijssis-2025-0042 Issue 1 | Vol. 18 (2025)

High-level area estimation model for FPGA-based designs using LLVM

Rachna Singh[1] and Gunjan Gupta[2]

[1]Department of Electronics & Communication Engineering, JIIT, Noida, Uttar Pradesh, India

[2]Cape Peninsula University of Technology, Bellville, South Africa

*E-mail: rachna.singh@mail.jiit.ac.in; guptag@cput.ac.za

Received for publication October 11, 2024.

Abstract

The functional segregation of a system into interacting hardware and software components needs estimation of the hardware area at an early design space exploration. However, the early estimation of design parameters from high-level programs is a time-consuming process, so a model is required for faster estimation of these parameters. This study presents a mathematical model for fast and accurate estimation of hardware area for implementations using the FPGAs family. In this study, a mathematical model is presented, which estimates the maximum number of LUTs and flip-flops consumed by different FPGAs. The input to this mathematical model is a high-level description in C language. The hardware synthesis of different FPGAs is done by using a low-level virtual machine (LLVM). The FPGAs used for the above work are Spartan 3E, Virtex-2pro, and Virtex-5, for which accuracy and run time for each model were determined. The results show that the estimation error for LUT is in the range of 1.11%–2.5% for Spartan 3E, 0.94%–2.4% for Virtex-2pro, and 1.32%–2.75% for Virtex-5. Similarly, the estimation error for flip-flops is in the range of 2.9%–4.9% for Spartan 3E, 3.2%–5.0% for Virtex-2pro, and 3.5%–5.2% for Virtex-5.

Keywords

FPGA, High-level synthesis, Design Space Exploration, Low-Level Virtual Machine

I. Introduction

FPGAs are becoming popular due to the faster growth of transistor density, which offers a very lucrative, inexpensive, and customized VLSI implementation platform. The most important requirement for an FPGA-based implementation is an estimation of design metrics like area, power, and latency. System-level estimation of these design metrics at an early stage of design is difficult; therefore, FPGA-based automatic hardware/software partitioning tool is required to estimate these parameters. It is difficult to predict these FPGA resources in the early phase of designing. The hardware area for FPGAs can be defined in terms of LUTs, flip-flops, CLBs, etc.

The hardware resources for a design are available after compilation and synthesis of the design, which might take a few minutes or even hours depending on the complexity of circuit to be implemented.

The post-technology mapping report clearly lists these hardware resources for a design, but it is not until the compilation, synthesis, and mapping phases that the data become clearly available. Depending on the size of the system, this entire process could take minutes or even hours. Therefore, a model pertaining to HW/SW partitioning is necessary in order to obtain a pre-synthesis estimate of hardware area.

To obtain a pre-synthesis estimate of the area of the FPGA resources, the area estimation model described in this work was created. A comparison

between several approaches and the suggested model for various benchmark circuits is provided. The advantages of the suggested work are listed as follows:

i) While the majority of state-of-the-art models are for specific designs, the area estimation model is generic in nature, meaning it may be utilized for the area estimation of any digital design.
ii) When compared to current approaches for estimating hardware area, the suggested method appears to be better in terms of estimation time and accuracy.
iii) This method is used in an open-source HLS tool to quickly explore the design space and collect crucial hardware design metrics.
iv) The model's simplicity allows for a quick evaluation, which in turn allows for a thorough search of the available design parameters.
v) The main concept is to derive the new expressions from a straightforward data analysis.

The remaining sections of this study are arranged as follows. Section II reviews some of the previous research on the topic of hardware area estimate. A summary of the suggested high-level area estimating methodology is provided in Section III. Section IV discusses the formulation for area estimation based on FPGA. Section V presents the tests and findings along with a comparison of the various approaches. Section VI wraps up the work and provides a quick overview of the potential for future advancements.

II. Related Work

A number of studies have addressed the importance of hardware area estimation in recent years. While some researchers have concentrated on estimating power usage, others have found hardware area, delay, and latency. It is crucial to estimate these attributes quickly and accurately in order to inform the decision-making process.

A thorough literature review has attempted to investigate the maximum state-of-the-art models associated with the proposed task. I/O performance and package count have improved as a result of an approach that incorporates functional partitioning into a synthesis methodology, as presented by Vahid et al. [21].

In a study [1], a superb analysis of the methods for estimating hardware attributes is provided.

The method suggested by Niu et al. [3] uses mapping process models to predict area and timing. The area is analyzed by predicting LUTs and CLBSs mapping and placement, where the input will be register transfer level (RTL) description. Analyzing CLB, routing, and input-to-output delays is necessary for the ensuing time estimation, which is also important. Since the placement tool must take into account the placement data on the area estimation, this method is heavily integrated into the design flow.

The many input description languages utilized for area estimate, such as C [2, 4, 5], SA-C [6], SystemC [7], MATLAB [8], Simulink [9], and VHDL [10], have been extensively studied. In most published work, the input RTL description is transformed into an Intermediate Representation (IR), such as Trimaran IR [4], Control Data Flow Graph (CDFG) [9], and VHDL AST [10]. The estimation procedure can then be applied to the intermediate format to estimate the area. The majority of methods identify opportunities for resource sharing through scheduling [10–11], design complexity [12], or the use of a derived resource usage formula [6].

Current methods leverage either FPGA- [4–10, 14–20] or ASIC-based designs [5] as their target technology. A physical model for the FPGA-based design has been created by Shi et al. [14], which uses an actual mapping to estimate the area. Other methods include creating a large database for every potential resource configuration [18–19] or using modeling equations of the FPGA functional resources [4–10, 15–17]. While most of the aforementioned research focuses on data path area estimation and ignores control logic, some approaches combine control logic and data path estimation into a single tool flow [4, 19], whereas others solely focus on estimating control logic area usage [20]. Numerous previous attempts on the subject of area estimate appear to have focused on incorporating the algorithms into CAD tools in order to create more extensive frameworks such as the one created by Shi et al. [14]. The study by Papakonstantinou et al. [23] is an exploration of multilevel granularity parallelism with HLS order of efficiency. Together with the estimating models and design layout data, it uses an effective space search heuristic to determine a configuration that performs close to optimal. Wang et al. [24] presented a generalized memory-partitioning paradigm that effectively resolves the bank access conflict issue while enabling high data throughput of on-chip storage. The performance bottlenecks of the OpenCL model on FPGA architecture are revealed by Wang et al. [25],

which presents an analytical performance model to predict the performance of OpenCL workloads on FPGAs [26].

The suggested modeling technique is designed for designs that can be broken down into small granularity modules and is quite general. Mathematical area estimating models can be used to determine the area for each of these modules in terms of LUTs and flip-flops. For benchmark circuits, the area in terms of LUTs and flip-flops is appropriately estimated by the suggested model. In each instance, MATLAB's curve fitting tool was used to determine the model coefficients.

III. Resource Estimation Methodology

This section outlines the process for precisely estimating the hardware area for an FPGA-based solution, as illustrated in Figure 1. An application's high-level specifications are used to extract the parameters for the application in terms of operators such as multiplexers, shift registers, and adders. Although the methodology in the presented work can be applied to any FPGA, a target platform is taken into consideration for application implementation. The target platforms are Virtex-2pro, Virtex-5, and Spartan 3E. For each of these platforms, a library of mathematical equations is developed, and the parameters and library are used to estimate the area. Each application generates a synthesis report, and the outcome is validated by comparing the estimated area with the area derived from the synthesis report. The high-level application written in C is fed into the mathematical model that is being presented, which uses low-level virtual machine (LLVM) to estimate the maximum number of LUTs and flip-flops that the hardware produced for various FPGAs might use.

The tool does not include scheduling, resource allocation, or binding algorithms because the suggested area estimation model for any given code to be implemented on FPGA will estimate the area by extracting the parameters in terms of operators, such as adders and multipliers, which do not contain any low-level structural or timing information. Pre-calculating a set of general resource consumption equations for every operator is the approach. The resource consumption of the entire C code is then estimated using these formulas. A bottom-up modeling technique serves as the foundation for the estimating process. Small granularity modules are taken into consideration throughout the modeling process, and by giving each module the appropriate characteristics, the area estimates of all these modules are then added up. A collection of modules with tiny to medium granularity makes up the model's lowest level. These modules are a predetermined collection of fundamental building elements known as operational units, which include adders, multipliers, registers, counters, and more. These operational units can be used to model a wide variety of designs because they are parameterized and reusable.

a. Library creation

It has been observed that registers, full adders, and multiplexers take up the bulk of space in any system

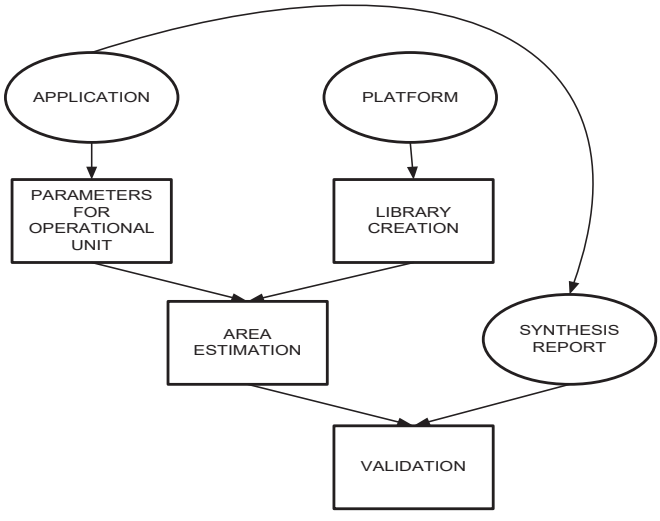

Figure 1: Research methodology flowchart.

design, such as DSP and communication applications. As a result, the majority of the design can be built using just these few fundamental operational units.

For designing the area estimation model, the steps used are as follows:

1) A pre-trained library is made for every operational unit. The target platform is necessary for the library to function. Spartan 3E, Virtex-2pro, and Virtex-5 are the target platforms for the suggested model. The Xilinx ISE (Xilinx (now part of AMD) and was succeeded by the Vivado Design Suite) synthesizer tool will be used to gather data for each of the specified target devices. Xilinx is used to synthesize the VHDL code for each of these operational units, and the number of LUTs, flip-flops, and DSP blocks/18 × 18 multipliers used to implement the code will be noted.
2) MATLAB's (MathWorks) curve fitting tool is used to derive the mathematical formula for each of these operational units. Regression analysis has been performed using MATLAB, with data preprocessing done with cftool. The stool's input is data that were obtained using the Xilinx ISE synthesizer to synthesize a VHDL code with variable numbers of inputs, ranging from 2 bits to 64 bits. The *x*- and *y*-axes in this study stand for the number of inputs and LUTs, respectively. The data supplied into the cftool are used for the parametric data fitting. Fitted coefficients and goodness-of-fit statistics are shown in the result area following data fitting. For every operational unit, the aforementioned procedure is repeated and the coefficients are noted. The operational units' regression analysis is displayed in Figure 2.
3) Finally, the equations have been derived by the analysis of the coefficients acquired in step 2. All operational units, such as shifters, multipliers, and comparators, are subjected to this study, and mathematical formulas have been developed for each one.

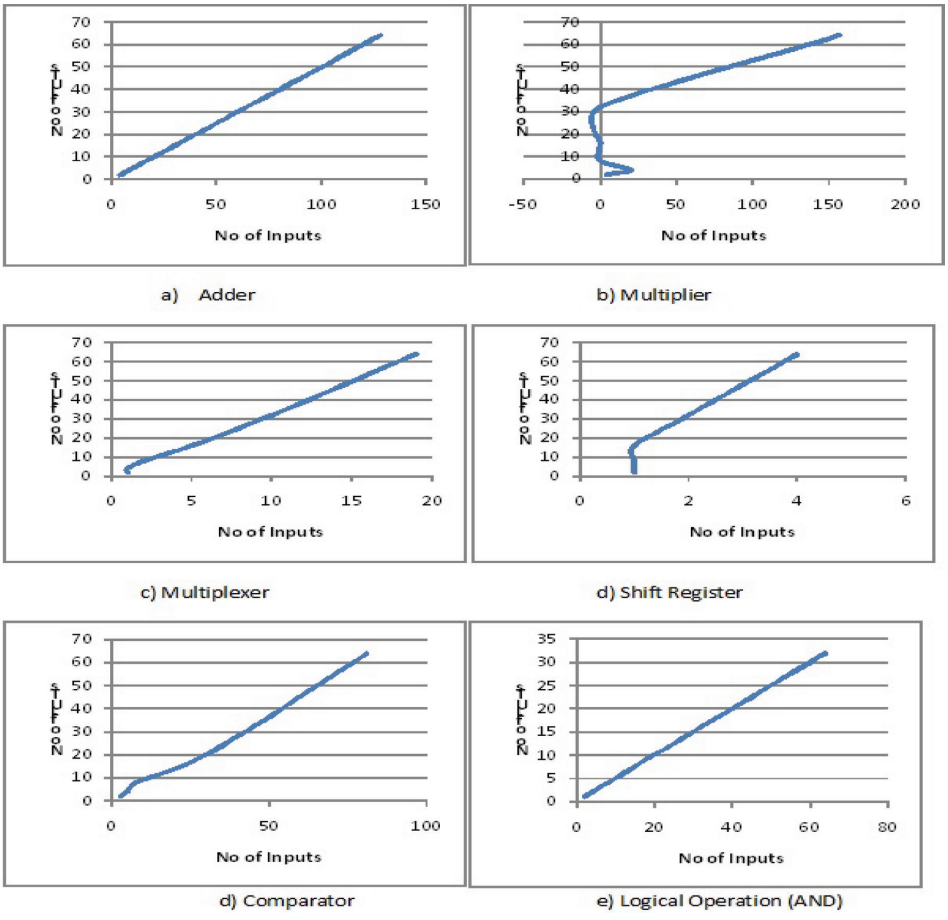

Figure 2: Regression Analyses of Operational Units.

IV. Area Estimation Model

The target FPGA devices' area estimation model is developed. Virtex-2pro, Virtex-5, and Spartan 3E are the target devices. Four 4-I/p LUTs, 16-bit shift registers, two storage elements that can be configured as latches or D flip-flops, carry logic, and arithmetic logic gates make up a slice of Spartan 3E FPGA and Virtex-2pro. Additionally, it has 18 × 18 specialized multipliers.

A slice of Virtex-5 is made up of a 32-bit shift register (16 bits × 2 shift registers) and four 6-input LUTs or dual-output-5-input LUTs. Additionally, it has 48 DSP48E slices, each of which has a 48-bit adder, a subtractor, and an accumulator as well as a 25 × 18 two-complement multiplier. Its dual-port RAM block size is 36 kb.

a. Number of LUT estimation

For Add/Sub, shifter, etc., the number of LUTs and the parameters has a linear relationship. The number of LUTs for adders, shifters, and other components can be estimated using the above mathematical expression. The result derived from these formulas is rounded to provide the actual number of LUTs, which is an integer. Let the estimated number of LUTs in a module be represented by $f_k(x)$.

The bit width for each input parameter is "x," and the operational units, such as the comparator, adder/subtractor, and bitwise logical operation, have two inputs. The number of bits is indicated by "x" for the shift register and multiplier. "x" indicates the number of inputs for the multiplexer. The estimated LUT use, $f_k(x)$, is typically a function of the number of inputs (x). Eq. (1) provides the generalized formula for the operational units. Table 1 lists the number allocated to each operating unit.

$$f_{k(x)=}\sum_{i=0}^{N^k} P_i^k x^i \tag{1}$$

One for k = 1–4 and two for k = 5 and 6 will be the value of "N^k." P_i^k is the coefficient that results from using MATLAB's curve fitting tool to fit the data. The value of P_i^k for the operational units is given in Table 2 for Spartan 3E, Table 3 for Virtex-2pro, and Table 4 for Virtex-5.

b. Number of flip-flop estimation

Here is the mathematical formula for calculating the flip-flop count in a shift register. The result produced by these formulas is rounded to provide an integer that represents the actual number of flip-flops.

Flip-flop usage is estimated as $f_f(x)$, which is often a function of the number of inputs (x). Eq. (2) provides the generalized formula for the operational units to estimate the number of flip-flops.

$$f_{f(x)=}\sum_{i=0}^{N} P_i^f x^i \tag{2}$$

For the Spartan 3E and Virtex-2pro, the value of "N" will be 1, and for the Virtex-5, it will be 4. Table 5 provides the value of P_i^f for the Spartan 3E, Virtex-2pro, and Virtex-5.

c. Number of DSP blocks/18 × 18 multipliers

The area utilization in this instance is unusual because it will include an additional area metric in addition to the LUTs and flip-flops, which are dedicated multipliers, since some FPGA devices include a dedicated hardwired multiplier (within their embedded DSP blocks).

If the number of bits multiplied in Virtex-5 falls between 8 and 32, the multiplier's area consumption will only be measured in terms of the DSP slice. The area used will only be in terms of LUTs if the number of bits multiplied is between 2 and 4, but it will be in terms of both LUTs and DSP slices if the number of bits is ≥64.

Comparably, for Virtex-2pro and Spartan3E, the area used for multiplying binary bits will be expressed in terms of the 18 × 18 multipliers for numbers between 2 and 16, and in terms of both LUTs and the 18 × 18 multipliers for numbers ≥32.

Therefore, it is evident from the explanation above that the dedicated multiplier might be designed to do multiplication up to 18 × 18 bits in the target FPGA

Table 1: Operational Unit Assignment

Number(k)	Operational Unit
1	Bitwise logical operation
2	Adder/Subtractor
3	Shift register
4	Multiplexer
5	Comparator
6	Multiplier

Table 2: Coefficients for LUT estimation models for Spartan 3E(XC3S500E)

Operational Unit	P_0^K	P_1^K	P_2^K	P_3^K	P_4^K	P_5^K
Bitwise logical operation	0	0.5	–	–	–	–
Adder/Subtractor	0.1292	0.9694	9.309×10^{-4}	–	–	–
Shift register	0	1	–	–	–	–
Multiplexer	–0.3062	0.5759	-1.329×10^{-4}	–	–	–
Comparator	–5.886	4.484	–0.1481	0.004	-3.345×10^{-5}	–
Multiplier	5.676	–3.161	0.704	–0.02215	4.935×10^{-4}	-3.76×10^{-6}

Table 3: Coefficients for LUT estimation models for Virtex-2pro(XC2VP2)

Operational Unit	P_0^K	P_1^K	P_2^K	P_3^K	P_4^K	P_5^K
Bitwise logical operation	0	1	–	–	–	–
Adder/Subtractor	0.1292	0.9694	9.309×10^{-4}	–	–	–
Shift register	0	1	–	–	–	–
Multiplexer	–0.3062	0.5759	-1.329×10^{-4}	–	–	–
Comparator	–5.886	4.484	–0.1481	0.004	-3.345×10^{-5}	–
Multiplier	0.8698	1.133	–0.2899	0.01588	-1.454×10^{-4}	–

Table 4: Coefficients for LUT estimation models for Virtex-5(XC5VLX50)

Operational Unit	P_0^K	P_1^K	P_2^K	P_3^K	P_4^K	P_5^K
Bitwise logical operation	–0.3062	0.5759	-1.329×10^{-4}	–	–	–
Adder/Subtractor	0	2	–	–	–	–
Shift register	0.9587	0.02605	–0.004192	2.039×10^{-4}	-2.081×10^{-6}	–
Multiplexer	1.122	0.1609	0.04224	-1.231×10^{-3}	1.061×10^{-5}	–
Comparator	4.33	–0.848	0.2229	-6.81×10^{-3}	5.978×10^{-5}	–
Multiplier	0.8698	1.133	–0.2899	0.01588	-1.454×10^{-4}	–

Table 5: Coefficients for Flip-flop estimation models for Target Device

Target Device(f)	P_1^f	P_2^f	P_3^f	P_4^f	P_5^f
Spartan 3E(1)	0.7667	0.8726	0	0	0
Virtex-2pro(2)	–0.3918	0.6792	0	0	0
Virtex-5(3)	0.8767	0.02006	-8.2×10^{-4}	1.965×10^{-5}	-1.32×10^{-7}

devices. Eq. (3) shows how DSP blocks are used in FPGA devices to carry out multiplication, where x is the maximum bit-width of inputs.

$$\text{DSP count} = \frac{\log_2 x}{8} \quad (3)$$

V. Experiment and Results

A wide range of benchmark circuits representing various resource quantities and applications have been used to test the resource estimating model. Spartan

3E, Virtex-2pro, and Virtex-5 were the three FPGA families that were targeted. Run-time and estimation accuracy were the two key factors that were utilized to evaluate the quality of the outcomes.

The tool flow receives a high-level description of the code that will be implemented on an FPGA. The statistics report, which provides information about the operation type, input bit-width, any constant inputs, loop limits, etc., is obtained using the LLVM tool flow. The area is estimated using these data as input to a mathematical model.

For a representative collection of benchmark circuits, the areas estimated using the model that was provided and those that were achieved by the actual synthesis followed by PAR have been compared. The precise number of flip-flops and LUTs utilized in the comparison is based on information supplied by Xilinx at the time of synthesis. The modeling inaccuracy is expressed as a relative proportion of the actual number of flip-flops and LUTs. The source of the error is as follows:

$$\text{Error}\% = \frac{E_a - s_a}{s_a} \times 100 \quad (4)$$

where *Ea* is the estimated area obtained from the proposed model.

S_a is the synthesis area obtained from ISE synthesis report.

The machine with an Intel Core i3 (Intel) CPU running at 3.30 GHz was used to compile the findings. The suggested tool's area estimation time ranged from 50 s to 80 s, whereas the Xilinx synthesis tool (XST) required 3–10 min to estimate the area for the benchmark circuits mentioned above.

A design's total area is made up of several parts, such as flip-flops, BRAMs, LUTs, multipliers or DSP blocks, and more. The number of blocks used in each of these components can be added to determine the overall number of BRAMs and multipliers or DSP blocks.

The Xilinx synthesizer automatically optimizes the design during synthesis, placement, and routing, and the user is not aware of this optimization process, so it is impossible to estimate the total number of LUTs and flip-flops in a complete design by adding up the number of LUTs and flip-flops in each component.

The results obtained are contrasted with those of other area estimating approaches in the section that follows. The outcomes are contrasted for reference circuits such as FIR, DCT, and IIR. Figure 3 shows the error percentage on the *x*-axis and the several benchmark circuits utilized for analysis on the *y*-axis. Spartan 3E was the target device for Kunz et al. [2], followed by Virtex-5 for Niu et al. [3] and then Virtex-2pro for Abdelhalim and Habib [9] and Deng et al. [22].

Evaluation of different estimation techniques is as follows:

1) The error percentage has been compared using an 8-point FIR filter. Figure 3 illustrates that the percentage inaccuracy ranges from 1.65% to 3.3%. The provided model has the lowest error of 1.65%, and Den et al.'s [22] model has an error of 2%. The model coefficients were also obtained using a curve fitting tool by Papakonstantinou et al. [23].

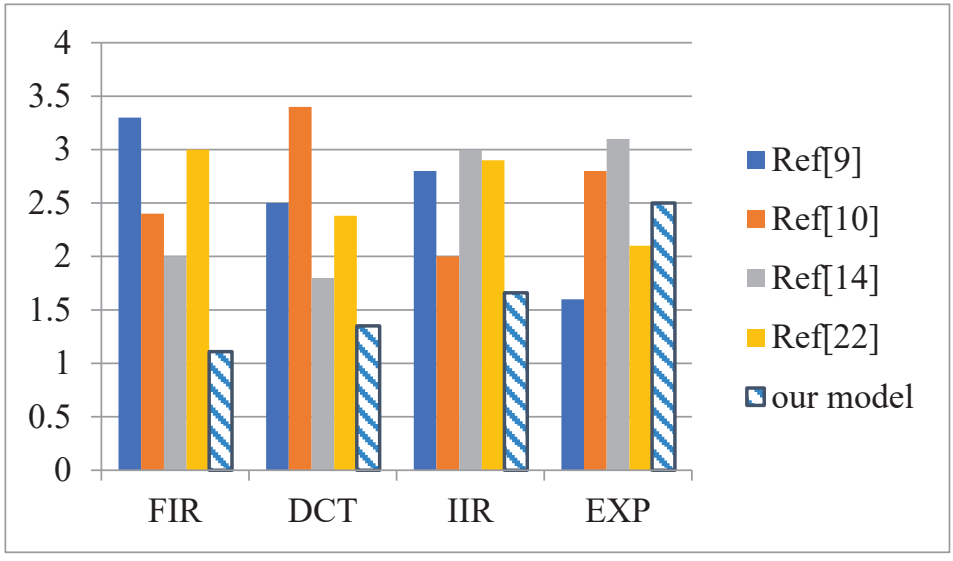

Figure 3: Area estimation error comparison chart.

2) Similarly, the proposed model has the lowest error for DCT and IIR, and Wang et al.'s [25] model has a lower error 1.75%.
3) However, the study shows that for the exponential function, the proposed model has an error of 2.5%, while Kunz et al.'s [2] model has the lowest error of 1.6%.

VI. Conclusion and Future Scope

This study presents an area modeling strategy for FPGA designs that uses both analytical and empirical techniques. At a very early stage of the design process, the suggested area estimating methodology can be used. The mathematical model is developed using a small number of straightforward, well-defined components known as operational units. By contrasting the estimated results with the synthesis findings, the estimation model's accuracy is confirmed. For Spartan 3E, Virtex-2pro, and Virtex-5, the estimation error for LUT falls between 1.11% and 2.5%, 0.94% and 2.4%, and 1.32% and 2.75%, respectively. Likewise, the flip-flop estimation error is between 2.9% and 4.9% for Spartan 3E, 3.2% and 5.0% for Virtex-2pro, and 3.5% and 5.2% for Virtex-5. The designer receives instant input from the run-times, which are typically substantially faster than synthesis run-times. The proposed model, which was compiled on a machine with an Intel core i3 processor with a clock frequency of 3.30 GHz, ranged in area estimation time from 50 s to 120 s, while the results of XST for the same machine took 3–10 min to estimate the area for the benchmark circuits. Although the model is tailored for Xilinx Spartan 3, Virtex-2pro, and Virtex-5, the approach is universal and easily translatable to other FPGAs.

In future, the proposed model can be used with other FPGAs such as Virtex UltraScale. Also, we can focus on the development of a complete mathematical model that can estimate latency and power consumption for a VLSI circuit apart from estimating area.

References

[1] Pablo González de Aledo Marugán, Javier González-Bayón and Pablo Sánchez Espeso, "Hardware performance estimation by Dynamic Scheduling", in Proc. Of Forum on specification and Design Languages, pp. 1-6, 2011.

[2] 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 Rang Imaging System", In 22nd International conference on Field Programmable Logic and Applications (FPL), Oslo, Norway, pp. 591-594, Aug 29-31, 2012.

[3] XiaoxiaNiu,YanxiaWu,Bowei Zhang, "Rapid FPGA-based Delay Estimation for the Hardware/Software Partitioning", Journal of networks,vol.8,pp. 1183-1190,5 May 2013.

[4] Xiaoxia Niu, Yanxia Wu, Bowei Zhang, Guochang Gu, Guoyin ZHANG "Auto Estimation Model of FPGA based Delay for the Hardware/Software Partitioning", Journals of Computational Information system, vol.9, pp. 6767-6774,1 September 2013.

[5] Ville Eerola, JariNurmi "High-level parameterizable area estimation modeling for ASIC designs", INTEGRATION, the VLSI journal, vol.47, pp. 461-475, 23 January 2014.

[6] Muzzaffar Rao, Thomas Newe, Ian Grout "AES implementation on Xilinx FPGAs suitable for FPGAbasedWBSNs", In 9th International conference on Sensing Technology (ICST), pp. 773-778, Dec 8-10, 2015.

[7] Bianca Silveira, Cláudio Diniz, Mateus Beck Fonseca, Eduardo Costa "SATD Hardware Architecture Based on 8x8 Hadamard Transform for HEVC Encoder", In IEEE International conference on Electronics, Circuits and Systems(ICECS), pp. 576-579, Dec 6-9, 2015.

[8] Paul Schumacher and PradipJha, "Fast and Accurate Resource Estimation of RTL-based designs targeting FPGA's", International conference on field programmable logic & applications,SanJose,CA, pp. 59-63,2008.

[9] M. B. Abdelhalim and S. E.-D. Habib, "Fast FPGA-based Area and Latency Estimation for a Novel Hardware/Software Partitioning Scheme", the 2nd intl. IEEE Design and Test Workshop,IDT07,Cairo,Egypt,pp. 775-779,2008.

[10] RoelMeeuws, "A Quantitative Model for Hardware/Software Partitioning", MSc thesis, *Delft University of Technology*, Delft, Netherland, Tech.Rep. RCOSY DES.6392,pp. 735-739,2007.

[11] Peter A. Milder, Mohammad Ahmad, James C. Hoe, and Markus Püschel, "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.

[12] 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.

[13] 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.

[14] 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 and Applications (FPL)*, LHCS 3203,pp.423-433,2004.

[15] A. Nayak, M. Haldar, A. Choudhary, and P. Banerjee,"Accurate Area and Delay Estimators for FPGAs", *DATE'02*, Paris,France, pp. 862-869, 2002.

[16] F. Vahid and 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] Lanping Deng, Kanwaldeep Sobti, *Chaitali Chakrabarti*, "Accurate Models for Estimating Area and Power of FPGA Implementations", In IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP - Las Vegas, NV, United States, pp. 1417-1420, 2008.

[23] Alexandros Papakonstantinou, Yun Liang, John A. Stratton, Karthik Gururaj, Deming Chen, Wen-Mei W. Hwu, Jason Cong, "Multilevel Granularity Parallelism Synthesis on FPGAs", In 19th IEEE International Symposium on Field-Programmable Custom Computing Machines (FCCM), Salt Lake City, UT, USA, pp. 178-185, 2011.

[24] Yuxin Wang, Peng Li, Jason Cong, "Theory and Algorithm for Generalized Memory Partitioning in High-Level Synthesis", Proceedings ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA), Monterey, California, USA, pp. 199-208, 2014.

[25] Shuo Wang, Yun Liang, Wei Zhang, "FlexCL: An Analytical Performance Model for OpenCLWorkloads on Flexible FPGAs", Proceedings of the 54th Annual Design Automation Conference, Austin, TX, USA, Article no. 27, June 18-22, 2017.

[26] D. Pradhan, B. K. Meher and P. K. Meher, "Digit-Size Selection for FPGA Implementation of Generic Digit-Serial Multiplication Over GF(2m)," *2023 1st International Conference on Circuits, Power and Intelligent Systems (CCPIS)*, Bhubaneswar, India, 2023, pp. 1-6, doi: 10.1109/CCPIS59145.2023.10291975.