Timing Modeling for Antifuse Based FPGAs

N S Nagaraj†, Paul D Krivacek‡, Mark G Harward‡

†Texas Instruments(India)  ‡Texas Instruments Inc.
Bangalore, Dallas, TX
India                 USA

Abstract

An accurate timing model plays a key role in performance driven synthesis, placement and routing besides timing verification. Timing modeling for antifuse based FPGAs is different from standard cells or gate arrays due to technology and architecture differences. In this paper, the methodology followed to develop a timing model for an antifuse based FPGA architecture is presented. The methodology includes extraction of interconnect parasitics in the form of RC trees from the place and route results. A reduced order approximation obtained for the driving point admittance of RC trees is coupled with SAS equations to compute load dependent delay and output slew. Rapid Interconnect Circuit Evaluator(RICE) is used for computing the wire delay. Experimental results of using the timing model to find correlation between delay and fanout is presented.

1. Introduction

FPGAs couple the flexibility of mask programmable gate arrays with convenience of field programmability. The programming element in different FPGA architectures include SRAM(XILINX, AT&T), E/EPROM(Altera, Lattice, AMD), dielectric antifuse(Actel, TI) and amorphous silicon antifuse(QuickLogic, CrossPoint). This paper focuses on antifuse based FPGAs[8][9]. An antifuse based channelled FPGA consists of rows of Logic Modules(LMs) interspersed with routing channels. Horizontal routing resources are available in routing channels in the form of pre-defined horizontal segments of varying spans. These may be connected to the input pins of the Logic Modules and joined together by antifuses to form longer segments, when required. The output drivers of the Logic Modules run vertically across a fixed number of channels and form the primary vertical routing resources. Additional vertical resources help in making long connections. An accurate timing model is needed to account for differences in architecture and technology for these FPGAs, to enable timing driven synthesis, timing driven physical design and timing verification. This paper discusses the methodology used to develop an accurate timing model for antifuse based FPGAs at Texas Instruments Inc.. The rest of the paper is organized as follows. Section 2 includes definition of terminology and notations used in this paper and gives an overview of the proposed timing model and methodology followed to obtain the model. Section 3 describes the RC extraction methodology for generating RC trees from place and route results. Section 4 reviews the driving point admittance calculation[2] to obtain a reduced order approximation of the RC trees generated for every net in the design. Results of SPICE[7] and SAS[4] regression analyses are presented in Section 5. The wire delay analysis using RICE[5] is described in Section 6. Experimental results on using the timing model to determine the correlation between delay and fanout for family of FPGAs under development at Texas Instruments Inc. is discussed in section 7. Section 8 concludes with remarks on future work.
2. Introduction to FPGA Timing Model

2.1 Definitions

2.1.1 Driver Node

It is the node driving the interconnect RC network. It is denoted by \( O \) in Fig. 2.1. It is the Source node for a net.

2.1.2 Input Node

It is the node driven by the driver. The input nodes are denoted by \( A, B, \ldots, N \) in Fig. 2.2. It is the Sink node of a net.

2.1.3 Intrinsic Delay

It is the delay between an input pin and point \( x \) shown in Fig. 2.1. However, in the proposed timing model intrinsic delay specified is the maximum of all path delays between between input pins and point \( x \). It is denoted by \( d_i \) in Fig. 2.1.

2.1.4 Driving Point Admittance(DPA)

It is the reduced order approximation for the admittance seen at the driver node \( O \) due to interconnect RC tree. It is calculated using the algorithm described in [2]. The whole RC tree is reduced to a \( \Pi \)-network shown in Fig. 2.3.

2.1.5 Load Dependent Delay

It is the delay through the output driver when driving the RC interconnect. It is denoted by \( d_f \) in Fig. 2.1. This computation of this delay as a function of DPA is described in section 5.

2.1.6 Input Slew

It is the rise time of the input signal. It is denoted by \( s_i \) in Fig. 2.1. Fall time is assumed to be equal to rise time. All slews is measured from 20\% to 80\% of signal strength(5V) unless otherwise specified.
2.1.7 Pre-drive Slew
It is the rise time of the signal at the input of the output driver. It is denoted by $s_x$ in Fig. 2.1. Fall time is assumed to be equal to rise time.

2.1.8 Output Slew
It is the rise time of the output signal. It is denoted by $s_o$ in Fig. 2.1. Fall time is assumed to be equal to rise time.

2.2 Notations
Following notations are used in the timing model:

- $d_l$: Load dependent delay
- $d_i$: Intrinsic delay
- $s_l$: Input slew
- $s_o$: Output slew
- $s_x$: Slew at intermediate node $x$
- $s_A$: Slew at input node $A$
- $s_B$: Slew at input node $B$
- $s_N$: Slew at input node $N$
- $d_{OA}$: Interconnect wire(pin2pin) delay between driver node $O$ at input node $A$
- $d_{OB}$: Interconnect wire(pin2pin) delay between driver node $O$ at input node $B$
- $d_{ON}$: Interconnect wire(pin2pin) delay between driver node $O$ at input node $N$
- $R$: Resistance of DPA II-network
- $C_1$: First capacitance of DPA II-network (Fig 2.3)
- $C_2$: Second capacitance of DPA II-network (Fig 2.3)

$R$, $C_1$ and $C_2$ are referred to as DPA parameters in later sections.

2.3 Methodology
The methodology followed for creating timing model is shown in Fig. 2.4. A brief explanation of each of the steps followed is given below:

1. A procedure for extracting interconnect RC trees from the place and route results was first defined. This procedure was decided from SPICE analysis performed to verify the error introduced by approximating/simplifying RC trees.

2. In order to establish a relationship on load dependent delay and slew, a driving point admittance(DPA) calculator was created using the approach described in [2]. The DPA calculator helps to obtain reduced order approximation for the driving point admittance of a RC tree. SPICE analysis was used to verify the DPA calculator accuracy.

3. The dependence of input slew on intrinsic delay, load dependent delay and output slew was verified to decide on comprehending input slew in the timing model. This analysis helped to simplify the timing model.

4. RC trees were extracted from place and route runs made using TIAEF[10](TIA Architecture Evaluation Flow). SPICE runs were made to obtain load dependent delay and output slew using DPA of the RC trees. SAS regression analysis was performed to obtain a relationship between load dependent delay in terms of DPA parameters $R$, $C_1$ and $C_2$. A similar analysis was performed to obtain a relationship between output slew in terms of DPA parameters $R$, $C_1$ and $C_2$. 
5. RICE was used to obtain interconnect delays. We have also developed a delay approximation technique [11] which can be used in place of RICE.

6. Using the relationships for load dependent delay and output slew in terms of DPA parameters $R$, $C_1$ and $C_2$, net delays were calculated for designs placed and routed in TIAEF and analyzed for correlation between fanout and delay.

Fig. 2.4 Timing Modeling methodology

2.4 Proposed Timing Model

The proposed timing model is as follows:

1. $d_i = f_1(s_i)$  \hspace{1cm} \ldots \hspace{1cm} 2.4.1
   Intrinsic delay is a function of input slew

2. $d_i = f_2(R, C_1, C_2)$  \hspace{1cm} \ldots \hspace{1cm} 2.4.2
   Load dependent delay is a function of DPA parameters $R$, $C_1$ and $C_2$.

3. $s_o = f_3(R, C_1, C_2)$  \hspace{1cm} \ldots \hspace{1cm} 2.4.3
   Output slew is a function of DPA parameters $R$, $C_1$ and $C_2$.

4. Slew at logic module input pins, $s_A$, $s_B$ etc. (nodes A, B etc. shown in Fig. 2.2) are calculated from RICE or by [11].

5. Interconnect delays $d_{OA}$, $d_{OB}$ etc. (see Fig. 2.2) are calculated from RICE or any other approximate analysis such as one proposed in [11].

3. RC Tree Extraction from Place and Route Results

In this section, RC extraction is briefly described in this section. From SPICE analysis on the error introduced by simplifying the RC modeling, it was decided to use a \Pi-model(C-R-C) to model the interconnect metal resistance and capacitance instead of using a lumped model or a more complicated distributed model. FPGA specific parasitics such as antifuse resistance and capacitance, junction
capacitance and gate(oxide) capacitance are also components of the RC trees as shown in Fig. 3.2. The values used for RC model depends on the technology and programming methods[1].

An example of a net using multi-segment horizontal track is shown in Fig. 3.1 and the corresponding RC tree is shown in Fig. 3.2. In the RC trees generated, all the vertical and horizontal wire segments(except input segment) are modeled with the II−model(C−R−C). More details can be found in [12].

![Diagram](image)

**Fig. 3.1 Example net showing usage of multiple-segment horizontal track**

![Diagram](image)

**Fig. 3.2 RC tree for example net shown in Fig. 3.1**

### 4. Driving Point Admittance calculation

Ignoring resistances of an RC tree driven by a gate for calculating load dependent delays can result in an increasingly significant error as resistance values increase. Published results for programmed antifuse resistance lie in the range of 100−500Ω. Clearly, ignoring these resistances for calculating load dependent delays would cause significant errors. Hence a reduced order approximation for the admittance seen by the driver is needed to improve accuracy of timing modeling of antifuse based FPGAs. Driving Point Admittance calculation described in [2] has been used for this purpose.

A comparison of the load dependent delay calculated using SPICE for full RC tree and II−network obtained using DPA calculator is shown in Fig. 4.1 and Fig. 4.2. In Fig. 4.1 and Fig. 4.2, x−axis represents index associated with every DPA parameter set. For example, Index 1 refers to DPA parameter set of R = XΩ, C1 = YpF and C2 = ZpF. Similarly indices 2,3 etc. correspond to different DPA parameter sets. Please note that the indices associated with DPA parameter sets could be different in different sections in this paper depending on number of DPA parameter sets used for comparison. In order to show the difference between DPA and SPICE results, only a portion of the graph (from index 141 to 195) is shown in Fig. 4.1 whereas error graph is shown for all 250 DPA parameter sets used for comparison. It can be seen that the results obtained by using DPA were found to agree with SPICE results within 1%.
5. SPICE analysis and SAS Equations for Load Dependent Delay and Output Slew

The characteristics of the output driver could be comprehended by an RC model and using the DPA calculated for the interconnect RC tree, the load dependent delay and output slew could be computed. Alternatively, regression analysis on SPICE simulation results could be used to comprehend the logic module driver characteristics. We have followed this method in creating the timing model, since it would be easy to comprehend process variations at a later stage in timing model development.

5.1 Effect of Input Slew on Delay and Output Slew

Intrinsic delay was found to vary with input slew as shown in Fig. 5.1.1. A quadratic model has been found to be good enough to predict the variation between input slew and intrinsic delay as shown in Fig. 5.1.1 and Fig. 5.1.2. It was found that input slew was not significantly influencing load dependent delay and pre-drive slew ($s_p$) as the input slew propagates through several logic gate stages in FPGA logic module. This analysis helped to simplify the timing model in that load dependent delay and pre-drive slew (and hence output slew) is independent of input slew.
5.2 Effect of DPA on Load Dependent Delay

Load dependent delays were calculated using SPICE and CODAC[3] for various DPA values obtained by executing DPA calculator on RC trees generated from place and routed netlists in TIAEF. SAS[4] regression analysis was performed to obtain an equation for load dependent delay in terms of DPA parameters \( R \), \( C_1 \) and \( C_2 \). The equation fitted for load dependent delay as a function DPA parameters was found to agree within 5% of SPICE values from Fig. 5.2.1 and Fig. 5.2.2. The regression analysis was done on about 2600 sets of DPA parameters. The x-axis in Fig. 5.2.1 and Fig. 5.2.2 refers to index associated with every parameter set.

5.3 Effect of DPA on Output Slew

Output Slew values were calculated using SPICE and CODAC[3] for various DPA values obtained by executing DPA calculator on RC trees generated from place and routed netlists in TIAEF. SAS[4] regression analysis was performed to obtain an equation for output slew in terms of DPA parameters \( R \), \( C_1 \) and \( C_2 \). The equation fitted for output slew as a function DPA parameters was found to agree within 5% of SPICE values from Fig. 5.3.1 and Fig. 5.3.2. The regression analysis was done on about 2600 sets of DPA parameters. The x-axis in Fig. 5.3.1 and Fig. 5.3.2 refers to index associated with every parameter set.
6. Interconnect Delay Analysis

Delay and slew characteristics of the propagating signal are affected due to the parasitics of the interconnect. RICE[5], a rapid interconnect circuit evaluator useful for finding wire delays and slews, has been used to perform wire delay and slew analysis. Experimental verification of RICE results with SPICE results showed less than 2% error on RC trees for representative nets of designs placed and routed in TIAEF. We have also implemented a new approximation technique[11] which is used in conjunction with DPA calculation described in [2].

7. Delay v/s Fanout correlation

Seven TGC100[12] netlists were synthesized using Synopsys[6] and placed and routed in TIAEF. RC trees were extracted and analysis on delay was performed using the timing model described in earlier sections. In this section, experimental results obtained on delay v/s fanout is presented.

Fig. 7.1 shows a distribution of pin2pin(wire) delay for different fanouts. Fig. 7.2 shows a distribution of total net delay for various fanouts, as seen at the macro level(logic view). This means that fanout of a net is 2 if it connects to 2 pins of one or more macros in the logic view. However, in antifuse based FPGA implementation, it may be connecting to more than two physical pins. A distribution of total delay for different fanouts in physical view is shown in Fig. 7.3.

A graphical representation of the statistical analysis of the delay data with respect to fanout are shown in Fig. 7.4, Fig. 7.5 and Fig. 7.6. Fig. 7.5 shows the variation of total net delay as a function of fanout(logical view). It can be seen that mean(average) net delay varies almost linearly with fanout. Some of irregularity in the Max curve can probably be eliminated with more data points in higher fanout range. Fig. 7.5 shows the wire delay variation as a function of fanout. Here also, mean wire delay varies linearly with fanout. Min curve is not shown in Fig. 7.5, since minimum almost coincides with x-axis(this happens when driver gets connected to a pin of adjacent module). The irregularity in the Max curve can probably be eliminated with more data points in higher fanout range. Fig. 7.6 shows the variation of the sum of load dependent and wire delay for various fanouts. Here also, mean delay varies almost linearly with fanout. The irregularity in the Max curve can probably be eliminated with more data points in higher fanout range.

The statistical data obtained from the statistical analysis is helpful in development of synthesis libraries for timing driven synthesis. The statistical data is also helpful for pre-layout delay estimation. The data can also be used for performance driven placement and routing. We are currently exploring different applications of the timing model analysis results.
Fig. 7.1 Distribution of Pin-Pin interconnect delay for different fanouts

Fig. 7.2 Distribution of total net delay for various fanouts (logic view)

Fig. 7.3 Distribution of total net delay for various fanouts (physical view)

Fig. 7.4 Variation Total delay for different fanouts (logic view)

Fig. 7.5 Variation pin2pin delay for different fanouts (logic view)

Fig. 7.6 Variation Load+wire delay for different fanouts (logic view)
8. Conclusion

A methodology for timing modeling of antifuse FPGAs is presented. Interconnect RC trees comprehend parasitics due to metal and programming components in FPGAs. It has been found that for logic modules with at least two levels of logic, load dependent delay and output slew can be treated as independent of input slew and only intrinsic delay is a function of input slew. Driving point admittance calculation described in [2] is a good approximation to represent the entire RC tree seen by the output driver. SPICE simulations have been coupled with SAS regression analysis to comprehend the driving gate characteristics of the logic module, within an error margin of 5%. Wire delay analysis performed by RICE is within 2% of SPICE results and that proposed in [11] is found to be within 5%. From the analysis done using the proposed timing model, it has been found that mean delay varies almost linearly with fanout. The results obtained by using the proposed timing model on a large sample of designs can be used for performance driven placement and routing as well as for synthesis library development. Future work includes comprehension of variation of voltage, process and temperature in the timing model. The distribution of delays for different fanouts will be further refined on large samples of designs.

Acknowledgment

The authors wish to thank the FPGA team at Texas Instruments for their support during the course of this work. We would like to thank Dr. Larry T. Pillage of Univ. of Texas at Austin for providing RICE which was invaluable to accomplish this work.

References


[5] Lawrence T. Pillage et al., "RICE 4.0 – Rapid Interconnect Circuit Evaluator User’s Guide," Univ. of Texas at Austin, USA


