DOI 10.31114/2078-7707-2018-1-63-68

SDF Report Generation Methodology for Digital Delay Lines without Simulations

V.Sh. Melikyan, Z.M. Avetisyan, A.A, Hovsepyan, A.Kh. Mkhitaryan, A.K. Hayrapetyan, A.A. Petrosyan, A.E. Mkrtchyan

Synopsys Armenia CJSC, Yerevan. Vazgen.Melikyan@synopsys.com, Zaven.Avetisyan@sysnopsys.com, Aristakes.Hovsepyan@sysnopsys.com

***Abstract* — In this paper a methodology of report generation for Digital Delay Lines (DDL) is presented. The mentioned approach is based on DDL report generation using Perl scripting without any Spice and Standard Delay Format (SDF) simulations. The need for this kind of approach is since generation of the DDL spreadsheets using Spice simulations brings to a great loss of time, because parallel to the increase of the number of elements, the extract netlists are also being increased. The presented approach saves more time compared to SDF simulations as well, because for large designs SDF files have very big sizes, which causes increase of machine resources and time for parsing and back-annotation of that SDF files. The methodology suggested in this paper uses the data of DDL SDF files to generate reports neglecting the parsing and back-annotation of SDF files.**

***Keywords* — DDR SDRAM; SDF parsing; SDF annotation; DDL**

I. INTRODUCTION

Because of its speed, pipeline features and burst access synchronous dynamic random access memory (SDRAM) is widely used in embedded system memory design. The double data rate (DDR) is an improvement of the traditional SDRAM. The benefit of DDR is transferring the data using both edges of each clock which doubles the data transfer rate of the RAM without a corresponding increase in clock frequency. One of the advantages of keeping the slower clock frequency is that the signal integrity requirements on the circuit board connecting the memory to the controller are being reduced.

As the sphere of DDR SDRAM grows very rapidly, the requirements for the SDRAM controller are increasing. The design and verification of the controller become more complicated.

Connections between the SDRAM and the user interface are being performed by SDRAM controller (Fig. 1) [1]. It is created for providing appropriate commands for initialization, read, write and memory refresh. The controller makes it easy for the user to work with SDRAM commands by converting the system interface to the user interface of microprocessor.

For read operation, the data and strobe signals are coming from SDRAM aligned by edges. As a result, data and strobe signals should come to the controller at the same time. To avoid setup/hold violations caused by the fact of the same time arriving of the mentioned signals an internal delay is performed in controller, which shifts the received strobe to the center of the received data eye [2]-[4]. Ideally the mentioned delay should be close to one fourth of the data clock period, which will move the strobe close to the center of data eye.

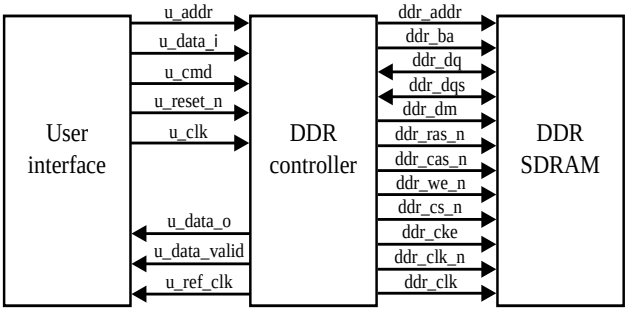

Fig. 1. System level block diagram for SDRAM

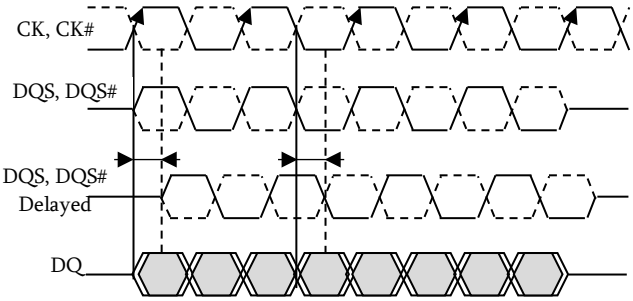

Fig. 2. Data eye centering for read/write operations

For write operation, before transferring to SDRAM, the controller must align the strobe signals to the center of data, i. e. the strobe and data transitions are 90 degrees phase shifted (Fig. 2) [2]-[4].

Digital delay line (DDL) circuit is proposed to generate these desired delays for data strobe signals.

II. DIGITAL DELAY LINE ARCHITECTURE

Nowadays it is very common to use all the digital delay lines (DDL) [5]-[8], which should cover rather large range of frequencies (from 400MHz to 3200MHz and more). All-

digital delay lines are more tolerant to power-supply noise, process-voltage-temperature conditions, are highly portable across multiple processes. For these reasons, digital delay lines are becoming more common in all-digital clock generation, synchronization and distribution circuits.

DDLs are used in several DDR standards [9] to provide appropriate phase shift (90 degrees) between data and strobe signals during both read and write operations. In other words, strobe signal is being centered into the data eye. In some DDR PHYs DDLs are coming to replace the delay locked loops (DLL). In DDL delay time is determined by delay setting inputs. DDL delays the input signal by a delay amount depending on the setting of tap select input (DDL Select<8:0>). The DDL used in this work has overall 511 steps. When the tap select input code is being increased by 1, the DDL delay should increase by one step delay (average 5ps for the fast case in this work). The delay values must always be increasing as delay setting value increases (Fig. 3). It is important for the delay line to be linear (i.e. that the step size be constant for all valid delay select values). The important thing is that the delay values are monotonic. From Fig. 3 it can be seen, that for curve 2 the monotonicity is violated, because on the delay to tap select dependency curve there are regions, where the tap select increases, but the delay is decreased.

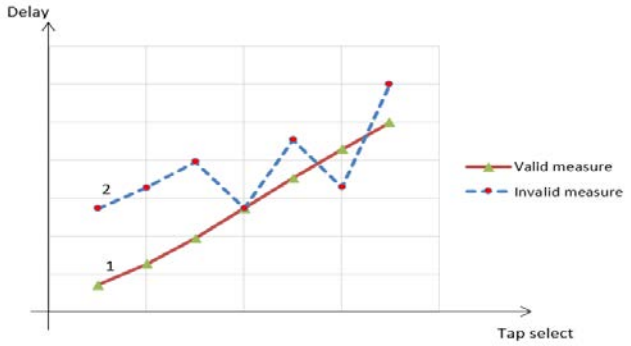

Fig. 3. Monotonicity of DDL

Considering the fact, that the average step in the fast corner is about 5ps and the DDL has 511 steps, the delay range of the DDL should be 2555ps (5ps x 511 steps).

As it can be seen from the block-diagram of DDL architecture, used in this work (Fig. 4), fine-coarse delay generation technics has been used in the design, which is rather common in design of DDLs [5]-[8]. DDLs, which are based on the fine-coarse technics, are widely used in low-power applications and microprocessors, in high speed SDRAM timing generation [4]. These kind of delay lines have several advantages like wide operating ranges and low power consumption.

The first stage of the DDL is the Fine delay block, which is used to get rather small delay steps. Fine delay block is being controlled by DDL Select<3:0> and is providing the 0-15 fine delay steps, each of which should be 5ps for the fast case. After coming out from the fine block, the signal is passing the stages of the Coarse block. A coarse delay element (CDE) is being used to shift the input signal by amount of time equal to 16 fine delays (80ps). So, it has no need to have such a small and correct delay steps, as the Fine block. Coarse blocks are being controlled by DDL Select<8:4>. Each step-size is corresponding to one-sixteenth of the coarse delay step size.

Constructions and architectures of several Fine/Coarse delay elements can be found in [5]-[8].

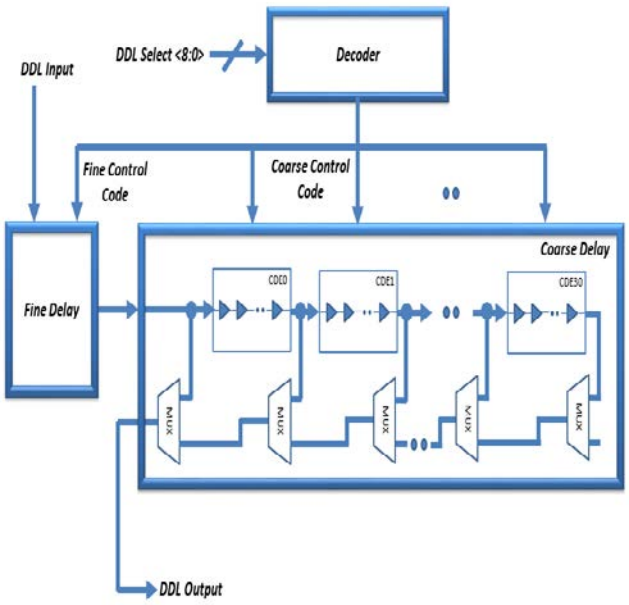

Fig. 4. Digital delay line architecture

III. COMMON REPORTING APPROACH USING SDF

Parallel to the increasing of the number of controller elements, the extract netlists are also being increased. As a result, generation of the DDL spreadsheets using Spice simulations brings to a great loss of time. Therefore, new methods have been prepared to deal with similar problems over the years. It is more efficient to use the Standard Delay Format (SDF) reporting method, which significantly decreases the simulation time, but reduces the accuracy of the results as well.

It is very helpful for the final state of design, where the reports must be done from top level of design after placement and routing, because spice simulations for that step are very slow and require long machine time.

A. Standard Delay Format

The delay and timing information of electronic circuits are being represented in SDF [10], which is a textual file format. There are EDA tools, which generate the timing data in SDF file. This information is important for almost any stage of design process. SDF was designed to for transferring timing information and constraints between different EDA tools. Many SDF constructs are similar to the ones in Verilog, because SDF was initially designed for the use by tools, using Verilog language.

Timing constraints, delays, timing checks, timing environment information, some scaling, environmental and technology parameters can be included in SDF file, but the cell and interconnect delays are the most typical parts of the SDF file.

As an example, input to output delay paths for a multiplexer (Fig. 5) and the corresponding part of SDF file (Fig. 6) can be considered.

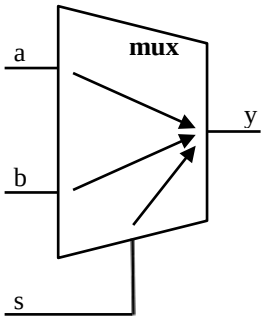

Fig. 5. Input/output path delay of a multiplexer

```
(INSTANCE mux)
(DELAY
   (ABSOLUTE
     (IOPATH a y (0.0225::0.0228) (0.0211::0.0213))
     (IOPATH b y (0.0221::0.0223) (0.0211::0.0213))
     (IOPATH (posedge s) y (0.0253::0.0261)
(0.0294::0.0301))
     (IOPATH (negedge s) y (0.0253::0.0260)
(0.0294::0.0300))
   )
)
```

Fig. 6. A representative SDF file

B. Problems

Correspondence of the information, presented in SDF file, to the timing models and design description is being checked by an annotator. For each part of the design, represented in SDF file, the timing model should be found. Data in SDF file should be applied to the appropriate parameters of the timing model. To apply the information in SDF file to the corresponding part of the design hierarchy the annotator is being used. Annotator is searching for the parts specified in the SDF file starting at this hierarchy point. The correspondence of the SDF file data with the design will not be found out, if the SDF file is not prepared in the mentioned way of usage [10].

It is very difficult to perform the dynamic simulation with back-annotation on a large design. SDF files are rather big and may run into gigabytes. Some difficulties may pop up when parsing these kinds of SDF file with significantly large sizes. There are different algorithms and methodologies presented in literature which reduce these kinds of issues connected with parsing and simulation times [11]-[12].

C. Algorithms

It is known [11], that common EDA algorithms are based on the idea of parallelism using multi-core processors. This fact will cause increase of the time taken by the parser. This is rather big present in all flow, which is causing a need to make the parser performance better. A methodology for efficient multi-trading of SDF parser is presented in [11].

The main challenge for the mentioned approach is to find out the functions, which can be made thread-safe. Based on this, the parsing flow is being divided into two phases. The main processing phase can run in parallel, but the post-processing phase cannot be parallelized [11].

In [12] a methodology is presented to improve performance of gate level timing simulation. The static timing analysis (STA) is being used at the block level. For block-level timing the critical path delay, identified by STA, is being used during simulation instead of actual cell delays. For large designs the SDF back-annotation will take rather long time and will have negative influence on the performance of gate-level timing simulation. To improve the performance of gate-level timing simulation, a hybrid approach is suggested to be used [12].

IV. PROPOSED APPROACH

Considering all the mentioned problems, at the final stage of design verification it is suggested to use the methodology proposed within the scope of this work. It helps to save machine time and resources.

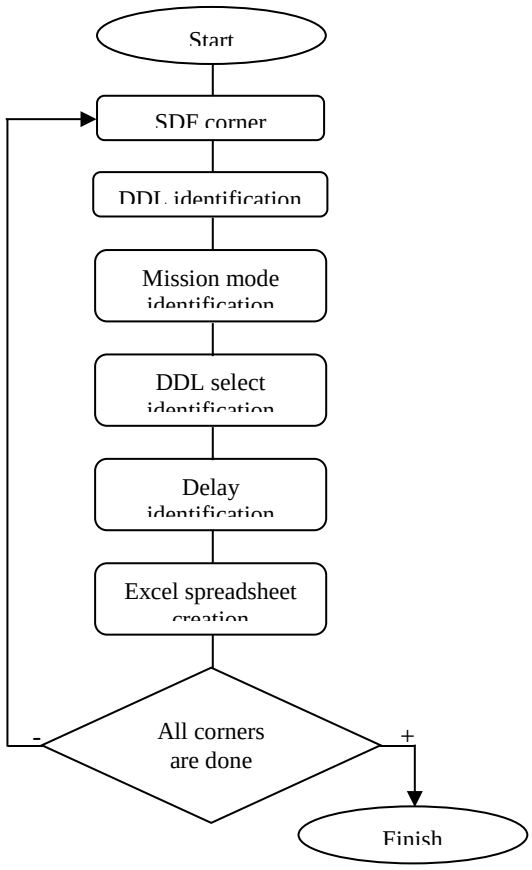

Fig. 7. Block-diagram of the basic program

The main purpose of this paper is to present a reporting methodology for DDL's spreadsheet used in a large design without any simulations. This approach doesn't suppose any

SDF annotations or parsing. This methodology is based on pure script running.

The main program is performed in Perl scripting language [13]-[14]. The list of all process-voltage-temperature (PVT) corners in SDF format is being given as an input for the program. After running the program gives the output with an Excel spreadsheet, which contains DDL's tap select values with corresponding delays for each PVT corner in separate tabs of an Excel spreadsheet.

A. Description of the basic algorithm

The main program uses the following steps to create an Excel spreadsheet with the DDL delay distribution information (Fig. 7):

1) Start: the algorithm is starting to work.

2) SDF corner selection: for any kind of design reporting should be done for several set of corners. In this step a single corner is being chosen from the corner list.

3) DDL identification: a subprogram identifies the DDL from SDF file by its instance name and separates it for future calculations.

4) Mission mode identification: for future calculations, a subprogram determines the mission mode of DDL from the values of input pins of the separated DDL instance.

5) DDL select identification: a subprogram determines tap selects from the values of input pins and applies corresponding decimal selects.

6) Delay identification: for any tap select a subprogram identifies the delay value. The null delay value for DDL is being identified as well.

7) Excel spreadsheet creation: the subprogram creates different tabs in Excel spreadsheet (Fig. 8).

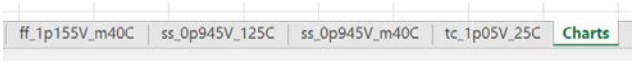

Fig. 8. Tabs of Excel spreadsheet

For each SDF corner a subprogram creates a new tab in Excel spreadsheet and fills it with appropriate data (Fig. 9).

- select: this column indicates input tap select values for DDL.
- delay: indicates the input to output delay of DDL.
- null_delay: indicates the insertion null delay of DDL.
- step_size: indicates the step size of DDL between two sequential tap selects.
- delay: indicates the delay between the primary and null delay outputs of DDL for each select.
- Average: indicates the average step size of all delay selects.

Besides the tables, the subprogram creates a tab in Excel spreadsheet for charts. In this tab charts are being created, which display different dependencies between parameters of DDL. DDL's delay dependency on the tap select (Fig. 10) and the dependencies of step sizes on the tap select (Fig. 11) for the considered corners are being generated in the report Excel spreadsheet. From chart of delay dependences (Fig. 10) the monotonicity of DDL can be seen.

8) Finish: if the program has calculated delay values for all corners, it enters the finish state.

	A	B	C	D	E	F
1	select	delay(ps)	null_delay	step_size	delay	Average
2	0	193.1	192	--	1.1	8.361328
3	1	198.8	192	5.7	6.8	
4	2	205.9	192	7.1	13.9	
5	3	213.9	192	8	21.9	
6	4	222.2	192	8.3	30.2	
7	5	229.9	192	7.7	37.9	
8	6	237	192	7.1	45	
9	7	243.1	192	6.1	51.1	
10	8	250.3	192	7.2	58.3	

Fig. 9. Excel spreadsheet view

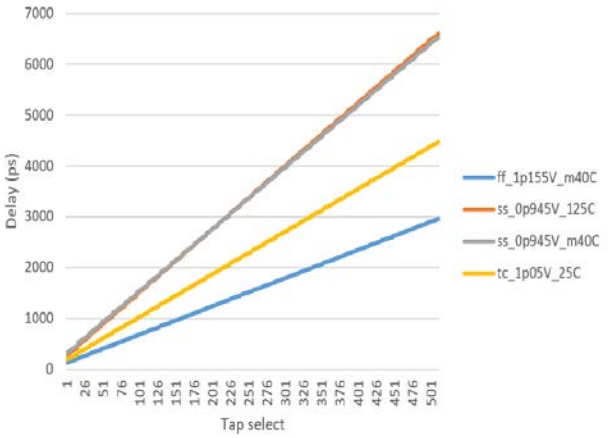

Fig. 10. Delay dependence on tap select of DDL

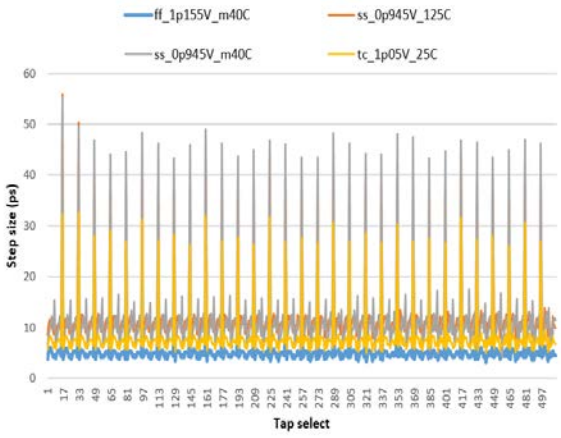

Fig. 11. DDL step sizes dependence on tap select

B. Benefits of the proposed method

As it is described in previous sections reporting with SDF simulations has parsing and back-annotation steps, each of which brings to several problems connected with the long machine time requirements. Proposed reporting method doesn't have mentioned problems and the reporting

is being done without simulations. Because of that this method reduces machine time requirements.

The second benefit relates to simulation environment creation. Usually there is a need for reporter to perform huge work to create a right working environment for SDF simulations. If for some projects there is only need to report the DDLs' delays and there is no created environment for SDF simulations yet, by using this reporting method reporter can avoid the environment creation.

V. Results and Conclusions

Some comparisons have been done between the DDL results obtained with Spice simulations, common SDF reporting approach and the proposed methodology (Fig. 12 and Table 1). The Spice simulation results have been chosen as reference and the results got with the other methodologies have been checked via these results.

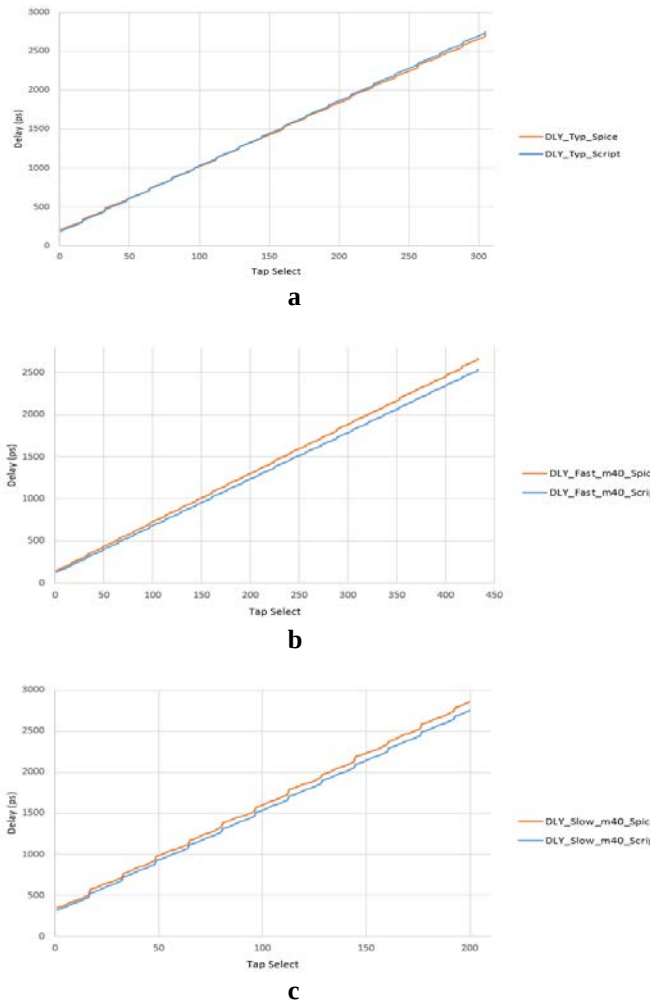

Fig. 12. Comparison of DDL delay dependence on tap select for 3 PVT corners

Comparison (Fig. 12) of DDL delay dependence on tap select in 3 PVT (a – typical, b – FF, c – SS) corners for Spice and the proposed method is done.

All the comparison results got with Spice simulation, common SDF reporting method and the proposed scripting approach have been gathered in the Table 1. As it can be seen from the mentioned table, results of the proposed scripting method have minor differences from the SDF simulation results. Using the proposed method helps to reduce the machine resources and time compared with SDF simulation approach (1320 times faster for slow case and 420 times faster for fast case).

Table 1

Comparative table of common and proposed methodologies

	Spice simulation	SDF simulation	Proposed methodology
Average step size (ff/tt/ss)	5.8/8.2/12.6	5.6/8.4/12.2	5.6/8.4/12.2
Minimum step size (ff/tt/ss)	3.5/4.7/5.3	3/5/6	3.3/5.2/5.6
Maximum step size (ff/tt/ss)	26.2/37.2/63.9	22/33/55	21.2/32.7/55.2

From the performed work it can be concluded, that compared with the common SDF reporting methodology, approach proposed in this paper helps to have significant machine time and resources savings with minor changes in results.

References

[1] URL:http://www.latticesemi.com/view_document?document_id=3467 (access date: 12.04.2018)

[2] Chung CC., Chen PL., Lee CY. An All-Digital Delay-Locked Loop for DDR SDRAM Controller Applications // IEEE International Symposium on VLSI Design, Automation and Test. 2006. P. 1–4.

[3] Garside J.D., Furber S.B., Temple S., Clark D.M., Plana L.A. An Asynchronous Fully Digital Delay Locked Loop for DDR SDRAM Data Recovery // IEEE 18th International Symposium on Asynchronous Circuits and Systems. 2012. P. 49–56.

[4] URL:https://www.micron.com/~/media/documents/products/technical-note/dram/tn4605.pdf (access date: 12.04.2018)

[5] Giordano R., Ameli F., Bifulco P., Bocci V., Cadeddu S., Izzo V., Lai A., Mastroianni S., Aloisio A. High-Resolution Synthesizable Digitally-Controlled Delay Lines // IEEE Journal of Solid-State Circuits. 2015. V. 62. № 6. P. 3163–3171.

[6] Abdulrazzaq B.I., Halin I.A., Sidek R.M., Shafie S., Yunus N. A., Kawahito S. Sub-Picosecond Jitter Resolution Wide Range Digital Delay Line for SoC Integration // IEEE International Circuits and Systems Symposium (ICSyS). 2015. P. 44-48.

[7] Raha P., Randall S., Jennings R., Helmick B., Amerasekera A., Haroun B. A Robust Digital Delay Line Architecture in a 0.13µm CMOS Technology Node for Reduced Design and Process Sensitivities // Proceedings of the International Symposium on Quality Electronic Design (ISQED). 2002. P. 148-154.

[8] Sourikopoulos I., Frappe A., Cathelin A., Clavier L., Kaiser A. A Digital Delay Line with Coarse/Fine tuning

throughGate/Body biasing in 28nm FDSOI // IEEE 42nd European Solid-State Circuits Conference (ESSCIRC). 2016. P. 145-148.
[9] URL: https://www.jedec.org/ (access date: 12.04.2018)
[10] The Institute of Electrical and Electronics Engineers, Inc. IEEE Standard for Standard Delay Format (SDF) for the Electronic Design Process. 2001.
[11] Shanbhag P., Gopalakrishnan C., Ghosh S. A Case Study in Developing an Efficient Multi-threaded EDA Parser: Synopsys SDF Parser // IEEE Computer Society Annual Symposium on VLSI. 2012. P. 297-301.
[12] Ahmad T.B., Ciesielski M.J. Fast STA Prediction-based Gate-level Timing Simulation // Proceedings of the IEEE conference on Design, Automation & Test in Europe (DATE). 2014. P. 1486-1492.
[13] Schwartz R.L., Foy B.D., Phoenix T. Learning Perl, Sixth Edition. O'Reilly Media, Inc., Sebastopol, CA. 2011. 390 p.
[14] URL: https://www.tutorialspoint.com/perl/index.htm(access date: 12.04.2018)

Методология создания SDF отчета цифровых линий задержки без симуляций

В.Ш. Меликян, З.М. Аветисян, А.А. Овсепян, А.Х. Мхитарян, А.К. Айрапетян, А.А. Петросян, А.Э.Мкртчян

ЗАО Синопсис Армения, г. Ереван. Vazgen.Melikyan@synopsys.com, Zaven.Avetisyan@sysnopsys.com, Aristakes.Hovsepyan@sysnopsys.com

***Аннотация* — В данной работе представлена методология создания отчета цифровых линий задержки (ЦЛЗ). Указанный подход основан на создании отчетов ЦЛЗ с использованием скриптового языка Perl без Spice и SDF симуляций. Необходимость в подходе данного типа заключается в том, что создание отчетов ЦЛЗ с использованием моделирования на схемном уровне (Spice) приводит к большим временным затратам по той причине, что параллельно росту количества элементов в управляющем блоке их файлы описания схемы также расширяются. Представленная методология также обеспечивает значительную экономию времени по сравнению с SDF симуляциями, так как для больших проектов SDF файлы могут иметь большие размеры, от чего возникает необходимость в больших машинных ресурсах и времени для разбора и аннотации таких SDF файлов. Методология, представленная в данной работе, использует SDF файлы ЦЛЗ для создания отчета, пренебрегая разбором и аннотацией SDF файлов. Отчет, созданный по представленной методологии, включает в себя информацию о различных параметрах ЦЛЗ, в особенности зависимости задержки и шага ЦЛЗ от входного цифрового кода управления.**

***Ключевые слова* — СДППД; SDF Разбор; SDF Аннотация; ЦЛЗ**

ЛИТЕРАТУРА

[1] URL:http://www.latticesemi.com/view_document?document_id=3467 (access date: 12.04.2018)
[2] Chung CC., Chen PL., Lee CY. An All-Digital Delay-Locked Loop for DDR SDRAM Controller Applications // IEEE International Symposium on VLSI Design, Automation and Test. 2006. P. 1–4.
[3] Garside J.D., Furber S.B., Temple S., Clark D.M., Plana L.A. An Asynchronous Fully Digital Delay Locked Loop for DDR SDRAM Data Recovery // IEEE 18th International Symposium on Asynchronous Circuits and Systems. 2012. P. 49–56.
[4] URL:https://www.micron.com/~/media/documents/products/technical-note/dram/tn4605.pdf (access date: 12.04.2018)
[5] Giordano R., Ameli F., Bifulco P., Bocci V., Cadeddu S., Izzo V., Lai A., Mastroianni S., Aloisio A. High-Resolution Synthesizable Digitally-Controlled Delay Lines // IEEE Journal of Solid-State Circuits. 2015. V. 62. № 6. P. 3163–3171.
[6] Abdulrazzaq B.I., Halin I.A., Sidek R.M., Shafie S., Yunus N. A., Kawahito S. Sub-Picosecond Jitter Resolution Wide Range Digital Delay Line for SoC Integration // IEEE International Circuits and Systems Symposium (ICSyS). 2015. P. 44-48.
[7] Raha P., Randall S., Jennings R., Helmick B., Amerasekera A., Haroun B. A Robust Digital Delay Line Architecture in a 0.13μm CMOS Technology Node for Reduced Design and Process Sensitivities // Proceedings of the International Symposium on Quality Electronic Design (ISQED). 2002. P. 148-154.
[8] Sourikopoulos I., Frappe A., Cathelin A., Clavier L., Kaiser A. A Digital Delay Line with Coarse/Fine tuning throughGate/Body biasing in 28nm FDSOI // IEEE 42nd European Solid-State Circuits Conference (ESSCIRC). 2016. P. 145-148.
[9] URL: https://www.jedec.org/ (access date: 12.04.2018)
[10] The Institute of Electrical and Electronics Engineers, Inc. IEEE Standard for Standard Delay Format (SDF) for the Electronic Design Process. 2001.
[11] Shanbhag P., Gopalakrishnan C., Ghosh S. A Case Study in Developing an Efficient Multi-threaded EDA Parser: Synopsys SDF Parser // IEEE Computer Society Annual Symposium on VLSI. 2012. P. 297-301.
[12] Ahmad T.B., Ciesielski M.J. Fast STA Prediction-based Gate-level Timing Simulation // Proceedings of the IEEE conference on Design, Automation & Test in Europe (DATE). 2014. P. 1486-1492.
[13] Schwartz R.L., Foy B.D., Phoenix T. Learning Perl, Sixth Edition. O'Reilly Media, Inc., Sebastopol, CA. 2011. 390 p.
[14] URL: https://www.tutorialspoint.com/perl/index.htm(access date: 12.04.2018).