2014 DESIGN AND VERIFICATION DVCON CONFERENCE AND EXHIBITION INDIA
Sept 25-26, 2014
Hotel Park Plaza
Bangalore, India

MDLL & Slave Delay Line performance analysis using novel delay modeling

Abhijith Kashyap, Avinash S and Kalpesh Shah
Backplane IP division, Texas Instruments, Bangalore, India
E-mail : abhijith.r.kashyap@ti.com
avinash.s@ti.com
kalpu@ti.com

***Abstract* — In this paper, we present a novel approach to comprehensively verify timing sensitive blocks, specifically Delay Lines and Delay Locked Loops by modelling transient behavior into the RTL. This paper focuses on reduction of simulation time and increase in design turnaround time. Information gathered from a small set of spice simulations is modelled into a Verilog based environment, which gives significant flexibility in terms of test cases, data collection and analysis.**

Keywords—Delay line; DLL; RTL modelling; DLL verification; Noise and jitter analysis

I. INTRODUCTION

Delay lines are digital-to-time converters which are used extensively in interface IPs like DDRPHY, MLB PHY and EMMC. Their function is to provide a constant delay across process, temperature and voltage (PTV) variations. This is achieved using a Master Delay-locked Loop (MDLL) which is essentially a linear control system.

Due to the analog-like behavior, traditional digital verification flows fail to capture transient behavior of the circuit as PTV variations are not modelled. Even though gate level simulation with SDFs account for delays in the design, these are static and effects of instantaneous delay variations on the design cannot be verified.

Thus there is a need to model On Chip Variation (OCV) and supply noise impact on delay elements in functional verification (RTL) of MDLL and slave Delay Line (SDL). This will help identify corner cases where the MDLL may get into an infinite loop condition, without further tracking. This will also give data on key performance parameters like jitter, loop response time and bring out shortcomings in the loop design.

II. TYPICAL ARCHITECTURE

The MDLL continuously tracks changes in VT and provides a PTV compensated code. It takes a reference clock and measures its period in terms of delay taps. This involves phase detectors and a control loop to track the phase alignment [1] [2]. This code is passed on to the slave. The slave instantiates an identical delay line and delays the input by a ratio of the clock period.

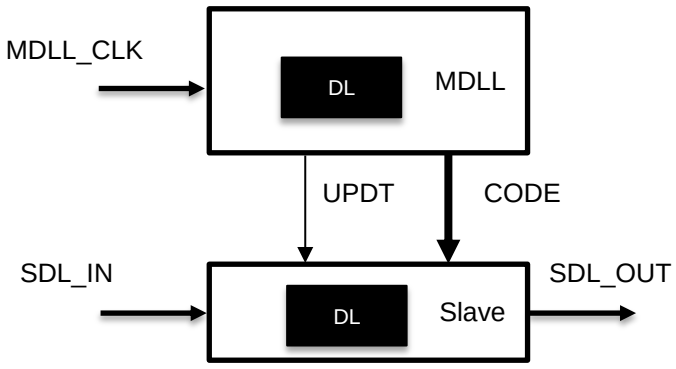

Figure 1: Block Diagram of typical MDLL and Slave delay line

III. Overview of the Proposed Flow

At the early design stage, we would have a defined requirement which will decide the structure of the delay line in terms of technology, delay elements to be chosen and length of the delay line. With this data, we create an initial RTL database and if possible a concurrent layout database. The following steps outline the procedure to enhance the flow.

A. RTL and verification stage

- Setup initial database with first cut architecture for the delay line and the MDLL
- Run spice characterization on the chosen delay element and determine effect of OCV and supply noise
- Derive a linear delay-variation model for the delay elements to mimic the spice data. The delay line being a chain of delay elements, would exhibit the summation of delay-variation from all these individual delay elements
- Run verification with test cases which exercise various noise waveforms as an input. The data is used for RTL refinement and possible architecture changes in case the requirement is not met. Rerun the verification on every new database to ensure thorough verification.
- Signoff checks and physical design execution to arrive at silicon ready database

B. Optional Steps for post silicon analysis

We may observe deviation in behavior during silicon checkout due to shortcomings in delay modelling, noise assumptions, input assumptions etc. To pinpoint the root cause, we can again revisit the verification with refined models and inputs

- Refine the noise models. Mimic input conditions seen on silicon.
- Run verification suite again to recreate scenarios seen on silicon. We can localize the problem easily since it is in Verilog format.
- Propose design changes or restrictions on input to avert the issue.

IV. Characterization and Modelling of the Delay Line

A. RTL hooks for the delay line

The delay given by an element can be split as the constant component which is the average delay, and the variable component. There are two aspects to delay variability of delay elements. Thus each Delay element modeled with a fixed delay component t_f and a variable delay component t_v with the total delay being $t_{d.}$

$$t_d = t_f + t_v$$

1. The static component is a result of OCV and other pseudo static phenomenon like temperature effect. This does not change with simulation time, and hence can be clubbed with the average delay to get the t_f component
2. The variable component is largely due to supply variation. This is assumed to affect all the delay elements in the same manner as the entire elements share the common voltage supply fabric. Although, care has to be taken care during layout to ensure minimal IR drop variation across elements.

Both these parameters 't_f' & 't_v' modeled as **Verilog real** data type, and they can be altered by the test bench at run time.

The full delay line is nothing but the chaining of all these delay elements back to back. Thus by modelling the delay elements, we have modelled the complete delay line.

B. *Spice Characterization of delay elements*

1) Characterization for t_f

a) Setup for a particular process condition

b) Run transistor - mismatch (Monte-Carlo) simulations

c) Delay difference between input and output of the delay line is tabulated

d) Average, Min and Max values obtained from this simulation is used to set 't_f'

$$t_f = avg \ \pm rand(avg - Min : Max \ - avg)$$

OCV being nearly a uniform random distribution, we can easily make use of the rand() function in Verilog to model t_f.

2) Characterization for t_v

a) Setup for a particular corner condition

b) Model the supply variation in spice

c) Use single sinusoid on supply, with different noise amplitude. Repeat for different frequency.

d) Measure delay variation with respect to applied noise

e) Derive a linear relationship for amplitude(A) of delay change with respect to noise amplitude as a function of the frequency. Typically this is frequency independent in the range of interest viz. < 1GHz.

f) We can apply superimposition principle and model the delay variation as

$$t_v = \sum_i A_i . \sin(2\pi F_i T)$$

where F is noise frequency, T is simulation time

This represents the transient behavior of delay elements

V. CASE STUDY

A. *Design specifics*

The proposed methodology was implemented on an MDLL and SDL which were hardened IPs in 28nm technology. The delay line is expected to provide delays over a large range from 1.7ns to 4.5ns. Jitter performance on the output of the SDL was critical as this was clocking a high speed media interface. The SDL always produces a quarter cycle shift w.r.t its input clock period. The relation between MDLL_CLK and SDL_CLK is always 2:1.

B. *Spice data for the delay element*

For the chosen delay element, many spice simulations were carried out in different process corners. We will concentrate on the data from the strong process for brevity.

1) Characterization for t_f

Average delay	20.23 ps
Min delay	18.63 ps
Max delay	21.82 ps
Spread	3.19 ps

$$t_f = 20.23 \ \pm \ 1.6$$

2) Characterization for t_v

a) No effect on frequency on amplitude of delay variation upto 2.5GHz ⇔ A_i is independent of F_i

b) 4% supply variation ⇔ 8% delay variation w.r.t average delay. 1x variation on supply leads to 2x delay variation. On chip as well, the specification was 4% supply variation. Thus A = 8% of 20.23ps = 1.61ps.

c) Frequency of input noise = Frequency of delay variation

$$t_v = 1.61 \sum_i \sin(2\pi F_i T)$$

3) Other parameters in the complete simulation environment

- Verification environment contains both MDLL and SDL hooked up as in Fig.1
- Delay lines in MDLL as well as SDL are modelled with dynamic delay variation.
- The supply noise is modeled as a sinusoid with the components 1 MHz, 64 MHz and 200 MHz. This was chosen based on clock frequencies on the supply line.
- 1 MHz Noise Amplitude > 64 MHz Noise > 200 MHz Noise. Expectation based on decoupling capacitors on the supply lines as well as decap-cells placed in the design.

VI. OBSERVATIONS

The test cases were designed to enable and disable each noise component independently. The cycle - cycle delay is measured between rising edges of SDL_IN and SDL_OUT. The following plots show the delay variation with simulation time advancing on the X-axis. The noise models were turned on only after capturing other verification parameters in the initial phase, and hence we see no variation initially.

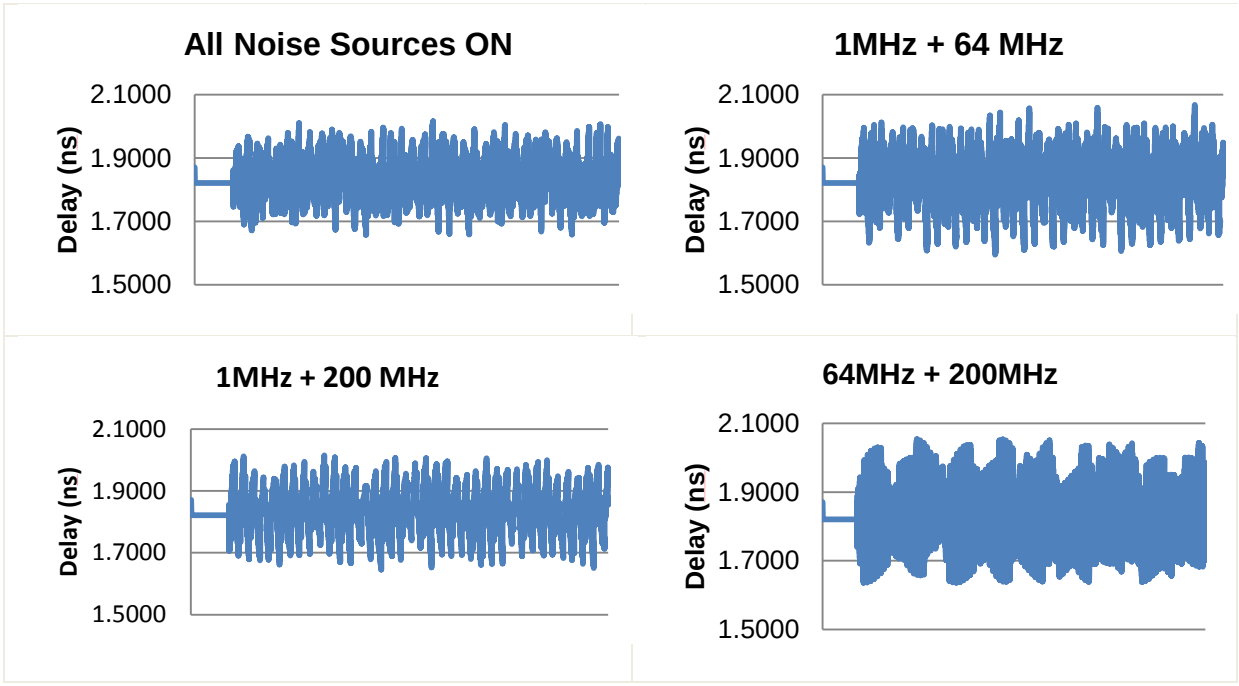

Figure 2 : Delay variation vs Time for different noise patterns

Table 1 Delay variation with different noise sources

Noise Model	SDL Delay in ps		
	Min	Max	Peak-to-Peak
All noise source ON (1MHz+64MHz+200MHz)	1657	2016	359
1MHz+64MHz	1594	2067	473
1MHz+200MHz	1645	2014	369
64MHz+200MHz	1636	2054	418

VII. INFERENCES AND DESIGN CHANGES

1) Main inference is that MDLL code changes do not respond appropriately in noisy conditions. The loop response has to be dampened to reduce jitter. This led to the introduction of an averaging logic on the code output in the MDLL. The averaging logic averages MDLL code over multiple samples before updating Slave Delay Line. Verification was re-run on the new database.

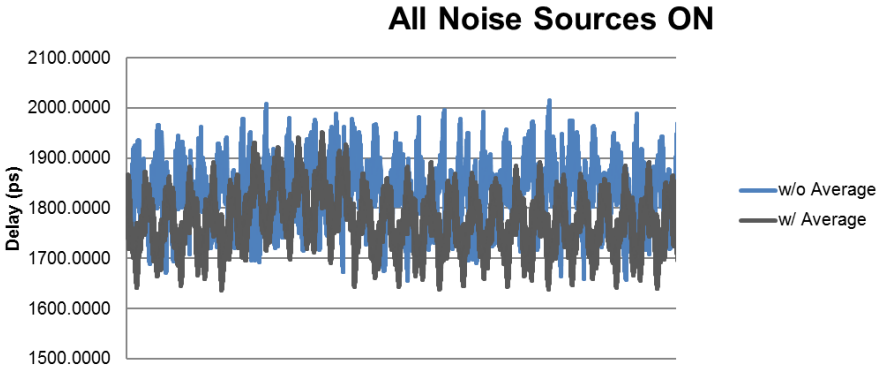

Figure 3 Comparison of design changes

The average logic on MDLL's code results in less delay variation on the Slave Delay Line. This can be seen in Figure 3 as the whole jitter envelope is contained well within that of previous database.

2) A bug was found in the update logic which would result in the MDLL being stuck. This was observed when noise input had spikes introduced.

VIII. SUMMARY

- Traditionally delay lines used are modeled using a fixed delay step which does not comprehend the reality that delay elements show variation due to PTV variations. Using the proposed model helps in verifying of MDLL + SDL with OCV and Supply noise on delay elements.
- The impact of various noise models can be analyzed much faster. As a comparison metric, we were able to run 15 different noise model simulations with simulation time of 1s in approximately 20 minutes. A single Spice simulation would take 3 days given the size of the delay lines.
- This ultimately leads to a more robust verification of the MDLL + SDL and gives an early feel of Jitter performance & helps to identify solution for improved performance. In this particular case, adding average logic helped in improving the jitter performance

Scope for further development

- Similar methodology for digital circuits having sensitivity to transient changes in PTV.
- A method to have variable delay (over that annotated using SDF) implemented at GLS can be explored. Currently there is no way to vary delay dynamically after SDF annotation

REFERENCES

[1] S. Sidiropoulos, Mark A. Horowiz, "A Semidigital Dual Delay Locked Loop," in IEEE J. Solid Sate Circuits, Vol.32, No.11, Nov 1997

[2] Jinyeong Moon, Hyeyoung Lee, "A DualLoop Delay Locked Loop with Multi Digital Delay Lines for GHz DRAMs," in Circuits and Systems (ISCAS) IEEE International Symposium, 2011

[3] Verilog Language Reference Manual - IEEE Std 1364™-2005