EAI Endorsed Transactions
on Cloud Systems
Research Article EAI.EU

Pulse Width Insensitive Design and Verification Methods

Ruchi Shankar[1,*], Shalini Eswaran[1], Sharavathi Bhat[2] and Lakshmanan Balasubramanian[1,†]

[1]Connected MCU, Texas Instruments (India) Pvt. Ltd., Bengaluru, Karnataka, 560 093, India.
[2]Analog Technology Development, Texas Instruments (India) Pvt. Ltd., Bengaluru, Karnataka, 560 093, India.

Abstract

Many embedded controllers have some critical system states that depend on an asynchronous event. Currently handling them in design depends on the availability of always-on slow clocks. In this paper we present a generic asynchronous design scheme that doesn't require a clock and ensure a reliable functionality without associated deadlock scenarios sensitive to exact arrival times of asynchronous events. This is enabled by a novel pulse width insensitive design method, which also requires unconventional verification methodology that ensures thorough and comprehensive pre-silicon design quality. These have been applied on the latest, ultra-low cost embedded micro-controller design targeted for cost sensitive applications.

Keywords: Pulse-width sensitivity, glitch, glitch filter, GLS, SDF, AMS co-simulation, DMS co-simulation, Analog mixed-signal, Digital mixed-signal.

Received on 22 November 2019, accepted on 06 January 2020, published on 10 January 2020

doi: 10.4108/_____

*Corresponding author. Email: r-shankar@ti.com
†Corresponding author. Email: lakshmanan@ieee.org

1. Introduction

Explosion of portable, battery operated, autonomous embedded internet-of-things (IOT) market and related application require low power and low cost as the DNA for all underlying building blocks. This mandates convergence and integration of analog mixed-signal (AMS) contents, power management [1][2][3][4]. This situation demands smart design and verification [5][6] approaches to address the challenges associated with low power system design and AMS integration.

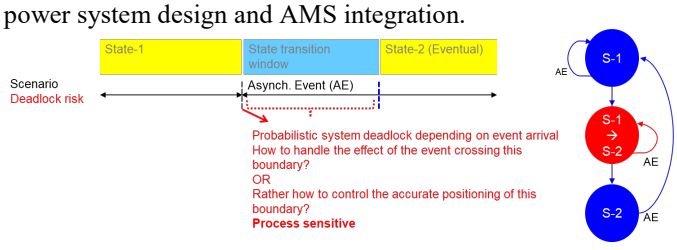


Figure 1. Hazard example

Operations of electrical circuits are sensitive to input glitches i.e. pulse widths of input signals. There are issues with system level function when built with such circuit elements. For sequencing operations, delay chains are used which are critical for asynchronous data or control paths. An example system hazard is illustrated in Figure 1 wherein it is critical to reach the eventual state S-2 if it ever reached state S-1. Though in an ideal system the transition between S-1 and S-2 is considered seamless, in an actual clock-less implementation with delay elements, there is a finite time window during the state transition that is in a pseudo-state. The arrival of asynchronous event during this transition window or pseudo-state can result in the system getting stuck in an irrecoverable forbidden state. This condition will result in a system deadlock, the recovery from which requires reboot or application hard reset. Earlier solutions implement pulse width filtering using standard glitch gobbler that may include sequential elements, delay and other combinatorial elements. The cells used in the delay elements or glitch gobbler circuits are not characterised for pulse width filtering characteristic. This mandates a

costlier SPICE based simulation at system level to verify the robustness of the design. Full handshake based asynchronous design methods may overcome such issues. However, systems or portions of the systems dealing with external asynchronous events that affect the system state are not amenable to such design methods.

Identified existing implementations of such systems are sensitive to pulse width of asynchronous input events. In this paper we propose a novel pulse width insensitive asynchronous design method that doesn't use any flip-flop or edge sensitive circuit elements. This scheme utilizes inherent glitch filtering behaviour of combinatorial gates. However there is no automated gate-level tools/methodology available to design, synthesize or verify such designs. To overcome this challenge we also propose a pulse width sensitive analysis capability for gate level simulation to avoid SPICE based simulations at system level.

The rest of the paper is organized into ten sections. Section 2 describes the proposed design and verification solution. Section 3 details system design aspects. Section 4 details circuit design technique. Section 5 describes the proposed design automation methodology. Section 6 describes the static timing analysis methodology. Section 7 describes the dynamic simulation based verification methodology. An application of the proposed methodology on an example design is illustrated in section 8, while further discussion on wider application and future scope is dealt with in section 9. Section 10 concludes the paper.

2. Proposed design and verification solution

Standard active element based delay elements exhibit glitch filtering behaviour. This fact is not utilised in existing design practices. Such behaviour of any existing circuits cannot be analysed easily with existing standard verification techniques. This requires costly transistor level (SPICE) simulations. Pulse width insensitive design is about ordering of delay elements appropriately to achieve the required glitch filtering behaviour. Simulation based analysis methods exist for similar behaviours in sequential circuits. Combination of such verification components is extended for combinatorial elements in the proposed method.

In this paper, a novel alternate system level design technique is proposed. This is an area and power efficient circuit design technique for pulse width insensitive design which uses delay elements only. Constraints driven design automation methodology is also proposed automate the design process. This involves pulse width sensitivity characterisation of combinatorial elements, including the timing library with this additional information, and use of existing logic and physical design tools for a constraint driven automated design synthesis. A simulation based verification method that can comprehend glitch sensitivity with an extension of functional model for glitch filtering behaviour is proposed for gate level design abstraction. Delay buffers and all combinatorial logic cells exhibit inherent characteristics of pulse width filtering, pulse width modification and propagation delay.

Pulse width filtering is the minimum input pulse width that will pass through (t_{mpw}) a delay cell or result in a valid change in output. Pulse width modification can be elongation or compression of the pulse as illustrated in Figure 2, with compression being a predominant behaviour. Delay models exhibit two types of delay namely transport and inertial delay. Transport delay models the propagation delay of circuit. Inertial delay is a measure of the elapsed time during which a signal must persist at an input of a device in order for a change to appear at an output. It is usually modelled for clock, reset and preset controls of flip flops.

Transport delay → $t_{mpw} = t_D$

Inertial delay → $t_{mpw} \neq t_D$

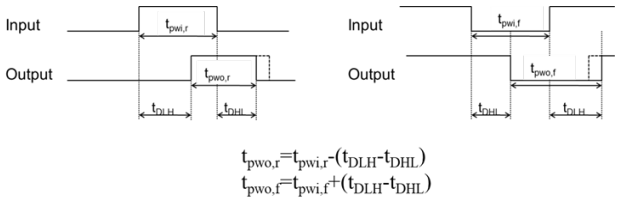


Figure 2. Pulse width filtering characteristic

There are key design concerns in building delay chains. Delay chains cannot be built with any random combination of individual delay cells. For example a delay chain built with multiple instances of the same delay cell is illustrated in Figure 3. As is highlighted, when the delay cell is characterised by pulse width compression behaviour, then for all conditions when the output pulse width of any delay cell is less than the minimum pulse width allowed for a subsequent delay cell, the propagation of the input pulse through the delay chain fails. Implementations involving delay chain as part of the analog modules are easy to analyse in SPICE / transistor level (TL) simulations. But delay chain implementation in semi-custom digital partition is difficult to analyse and verify in robust manner.

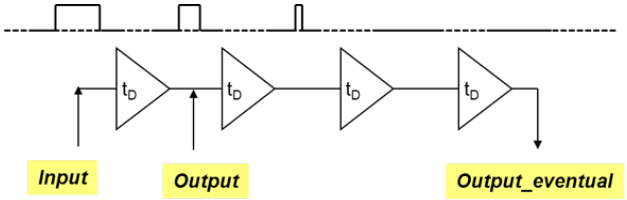


Figure 3. Pulse width filtering in a delay chain

3. System design

This paper proposes asynchronous event look-ahead as shown in Figure 4 for constraint driven design without architectural design cost. Look-ahead information is used to gate the asynchronous event for a short period during state transition window. System robustness is achieved at the cost of probabilistic loss of event in a small time window during state transition.

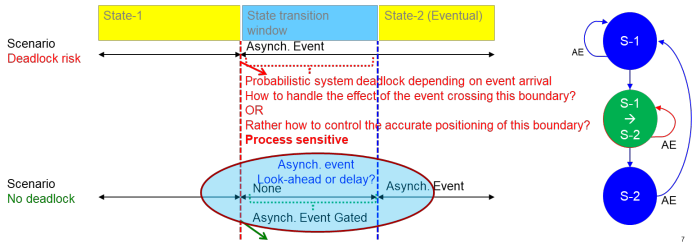


Figure 4. Asynchronous event look ahead

To arrive at a circuit design method we first analysed a delay buffer using transistor level (TL) simulations. The pulse width filtering behaviour of delay buffers are constrained by relations $t_{mpw} < t_D$ and t_{mpw} α t_D. Hence we propose to synthesize the delay chains such that the first delay element provides the safest pulse width filtering ($t_{D,max}=t_n$)>t_{n-1} as illustrated in Fig. 5. Conventional glitch gobbling function is implemented herein but through natural property of the delay cell. First cell uses a delay cell with the largest delay ($t_n=t_{D,max}$) and largest pulse width propagation ($t_{mpw,max}$). Composition of the subsequent portion of chain is immaterial as long as all the cells obey the conditions $t_D < t_{D,max}$ and $t_{mpw} < t_{mpw,max}$. They will not encounter any input with min. pulse width ($t_{mpw,i} > t_{opw,max}$). It should only use cells with delay less than $t_{D,max}$ & $t_{mpw}<t_{mpw,max}$. This scheme is area and power efficient as it does not involve a flip-flop or latch element. Fig. 5 shows an implementation for which any input event that reaches the output of the first delay cell output *(Output)* will eventually result in corresponding event at the end of the delay cell *(Output_eventual)* without fail under any condition.

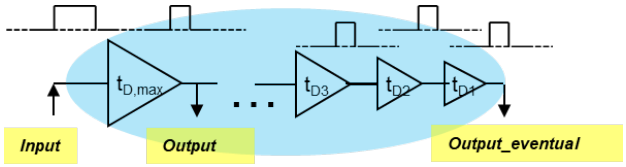


Figure 5. Delay chain composition

5. Pulse width insensitive design automation methodology

Minimum pulse width (*min_pulse_width*) characterization involves a custom automation along with a test bench. A test bench with two instances of the delay cells is used for simulation. One is fed with a high going pulse and other is fed with a low going pulse. The algorithm for evaluating the *min_pulse_width* is illustrated in Figure 6. An alternative method of characterization is as follows:

1. Infinite rise and fall delay is calculated using an input with huge pulse width.
2. A binary search with varied pulse (say, a range of 1:25ns) input is given to cell.
3. The fail criteria is a predefined amount of (say 10%) degraded delay with respect to infinite delay.

1. Apply input stimulus
2. Simulate the cell with SPICE netlist
3. Sweep the input pulse width
4. Check for correctness of propagated output value
    1. If pass Repeat 3 & 4
    2. If fail store the pulse width as t_{mpw}

Figure 6. Characterisatin pseudocode for minimum pulse width

The characterized pulse width is given as a function of input slew in Liberty format timing library file as shown in Figure 7. The Verilog model is updated to have *$width* constraint as shown in Figure 8.

```
pin (A) {
        timing () {
          related_pin : "A";
          timing_type : min_pulse_width;

          rise_constraint (constraint_slew) {
            index_1 ("0.02,0.05,0.1 ");
            values ("1.9697265625,1.9580078125,1.9404296875 "
          }
          fall_constraint (constraint_slew) {
            index_1 ("0.02,0.05,0.1");
            values ("2.3154296875,2.318359375,2.3271484375");
          }
        }
  capacitance : 0.3;
  max_transition : 0.1;
  direction : input;
```

Figure 7. Minimum pulse width characterization in Liberty timing library

```
specify
  (A => Y)  = 1.0 ;
   $width( posedge A, 1.0:1.0:1.0);
   $width( negedge A, 1.0:1.0:1.0);
endspecify
```

Figure 8. Minimum pulse width check in Verilog

Using the design constraints mentioned in section 4, design process can be automated as shown in Figure 9.

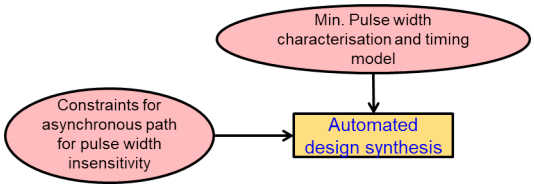


Figure 9. Design automation methodology

Commercially available vendor design synthesis tools don't support such a design process. However, based on design queries and custom automation within the vendor design tool (Ex. Cadence Genus®) framework, the pseudocode illustrated in Figure 10 can be implemented using TCL interface.

- Define timing arc
- Define reference clock
- List of delay cells considered with t_{mpw} & t_D information
- Define the design constraint
  - Max. & min. number of stages (N_{max}, N_{min})
  - Total delay ($t_{D,target}$)
  - Arrive at the delay line structure such that
    - $t_{D,target} = \sum_{j=1}^{N_{min} \le N \le N_{max}} t_{D,i}$
      ➔ Obtain N & Delay cell ordering
    - For j=1, N
      Calculate $t_{mpw;o}$
      $t_{mpw,o} = t_{mpw,i} - E.(t_{DHL}-t_{DLH})$
      E = +1, Rising pulse
      -1, Falling pulse
      such that, $t_{mpw,o,j-1} \geq t_{mpw,i,j}$

Figure 10. Custom synthesis pseudocode

6. STA for pulse width sensitive timing paths

Static timing analysis (STA) for the pulse width sensitive timing paths in design can be performed using commercially available vendor design analysis tools including the synthesis (Ex. Cadence Genus®) and STA tools may be performed using the pseudocode illustrated in Figure 11.

- Define timing arc
- Define reference clock
- List of delay cells considered with t_{mpw} & t_D information
- Find N=Number of cells in the timing arc
  - For j=1, N
    Query $t_{mpw,i}$
    Calculate $t_{mpw;o}$
    (Automatically done by STA tools)
    $t_{mpw,o} = t_{mpw,i} - E.(t_{DHL}-t_{DLH})$
    E = +1, Rising pulse
    -1, Falling pulse
  - Check for violation of design constraint
    (Covered by min. pulse width check in STA & synthesis tools)
    $t_{mpw,o,j-1} \geq t_{mpw,i,j}$
- Check $t_{D,target} = \Delta_{error} + \sum_{j=1}^{N_{min} \le N \le N_{max}} t_{D,i}$

Figure 11. Custom STA pseudocode

Though STA based design closure can be used to ensure all the timing requirements on the pulse width sensitive paths or arcs are met without simulation compute complexity and subjective coverage issues associated with dynamic simulations, it may not ensure the complete functional intent correctness. Hence simulation of gate level (GL) implementation is mandatory to verify complete design intent. This is primarily needed to ensure verification closure when such paths and their impact span beyond digital design partition and into mixed-signal design content.

Once the minimum pulse width characterization is done as described in previous section 5, UDP models are required to be updated to support glitch filtering as showed in Figure 12.

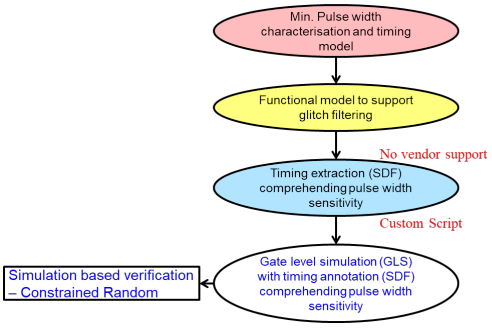


Figure 12. Design verification methodology

Conventionally as described in Figure 13(a), if there is a minimum pulse width violation, UDP table drives an unknown ("X") at the output. Figure 14(a) shows the

simulation result when minimum pulse width violation occurs. Updates in UDP table are proposed in Figure 13(b) that enables pulse width filtering behaviour. Figure 14(b) shows the simulation result with previous value retained when minimum pulse width violation occurs.

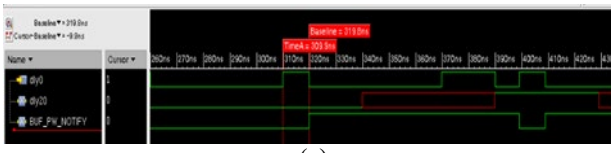
```
table //Conventional
//     A  nt :Y:  Y
       n  ?  :?:  0 ;
       p  ?  :?:  1 ;
       ?  *  :?:  x ;
endtable
```

(a)

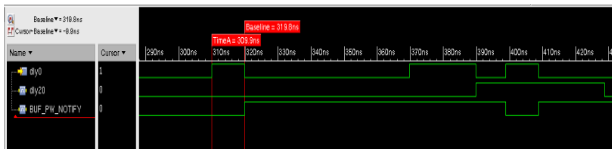
```
table //Proposed
//     A  nt :Y:  Y
       n  ?  :?:  0 ;
       p  ?  :?:  1 ;
       ?  *  :?:  - ;
endtable
```

(b)

Figure 13. Buffer functional modelling



(a)



(b)

Figure 14. Simulation results

Commercially available EDA vendor tools only support SDF generation as mentioned in Figure 15.

```
(CELL
  (CELLTYPE "DLY420_NT")
  (INSTANCE buf20)
  (DELAY
    (ABSOLUTE
    (IOPATH A Y ((tDLH) (tmpwo,f)) ((tDHL) (tmpwo,r)))
    )
//(PATHPULSE A Y ((tmpw))
  )
  (TIMINGCHECK
    (WIDTH (negedge A) (tmpwi,f))
    (WIDTH (posedge A) (tmpwi,r))
  )
)
```

Figure 15. Input SDF from tool implementation flow

$$t_{pwo,f}=t_{pwi,f}+(t_{DLH}-t_{DHL})$$

$$t_{pwo,r}=t_{pwi,r}-(t_{DLH}-t_{DHL})$$

Output pulse width ➔ $T_{mpw,o} = t_{mpw,i} - E.(t_{DHL}-t_{DLH})$

$E = +1$, *Rising pulse*

-1, *Falling pulse*

Figure 16. Calculation of propagated output pulse width

The output pulse width is calculated based on propagation delays and input pulse widths as in the equation shown in Figure 16. An output SDF as described in Figure 17 is generated using custom automation to incorporate the minimum output pulse width *IOPATH* triplet.

```
(CELL
  (CELLTYPE "DLY420_NT")
  (INSTANCE buf20)
  (DELAY
    (ABSOLUTE
    (IOPATH A Y ((tDLH) (tmpwo,f)) ((tDHL) (tmpwo,r)))
    )
//(PATHPULSE A Y ((tmpw))
  )
  (TIMINGCHECK
    (WIDTH (negedge A) (tmpwi,f))
    (WIDTH (posedge A) (tmpwi,r))
  )
)
```

Figure 17. Output SDF generated using SDF flow

8. Case study and results

Figure 18 shows a GLS illustration of a single delay buffer, where the proposed simulation method with appropriate SDF timing annotation is applied to enable the pulse width filtering characteristic accurately. Further the proposed design and verification methods were applied on an ultra-low power & low cost mixed-signal SoC design. Earlier identified system deadlock scenarios with conventional design method were verified to indeed cause irrecoverable state using proposed verification method. Redesign for delay chain was implemented using proposed design method. The scenarios that were earlier failing were verified to be behaving robustly in the newer design.

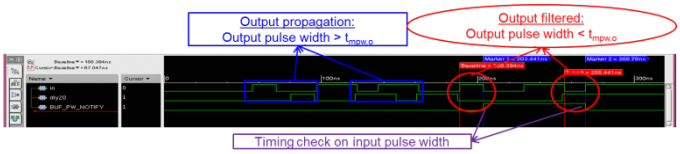


Figure 18. GLS illustration of a buffer with pulse width filtering modelled

9. Discussions and future directions

Fully automated back-end flow that comprehends pulse width sensitivity for design & SDF generation can be enabled with EDA vendor engagement. Additionally formal design and verification [7][8] methods can be enabled for robustness.

10. Conclusions

Pulse width insensitive design and verification method is conceptualised & applied on the design. It involves a novel area and power efficient design method for pulse width insensitive design. It also introduces an automated constraints driven design synthesis methodology that can comprehend the pulse width sensitivity of the design components. Further it involves a simulation based verification method to analyse such behaviours and designs at GL abstraction. It improves the overall verification efficiency by avoiding costly SPICE based simulations to comprehend such scenarios. It allows handling reliable design of asynchronous sections in an otherwise synchronous design. Other known asynchronous design methods are difficult or costlier to apply unconditionally. Thus we enabled cost and power competitive low power design implementation for mixed-signal SoC.

Acknowledgements.

The authors thank Sameer Dabadghav(Ex. TI) of AMD Inc., Somasekar J, Neeraj Saxena, Padmini Sampath(Ex. TI), Ajith Subramonia, & Vivek Singhal of Texas Instruments Inc. for their motivation and support in pursuing the unknown; Niraj Mahapatro of Sankalp Semiconductor and Vinaykumar S Hegde(Ex. TI) of Robby Technologies, Inc. for the library characterisation support; Nikhil Chandrakant Sangani of Texas Instruments Inc. for help in identifying *PATHPULSE* SDF construct; Ramakrishna Reddy(Ex. TI) of AMD Inc. for help towards resolving several simulation and flow issues; Somshubra Paul, Anand Kumar G of Texas Instruments Inc. & Ayon Dey(Ex. TI) of AMD Inc. for several discussions on analog and system level requirements, scenarios, care-abouts and limitations.

References

[1] Chapter: (2011) System Drivers: Mixed-signal Evolution. In *International Technology Roadmap for Semiconductors* (USA: ITRS).

[2] Conference: Balasubramanian, L., Sabbarwal, P., Mittal, R. K., Narayanan, P., Dash, R. K., Kudari, A. D., Manian, S., Polarouthu, S., Parthasarathy, H., Vijayaraghavan, R. C., Turkewadikar, S. (2011) Circuit and DFT techniques for robust and low cost qualification of a mixed signal SoC with integrated power management system. In *Proceedings of Design Automation & Test in Europe Conference & Exhibition (DATE),* Grenoble, France, 14th-18th Mar. 2011 (USA: IEEE), 1-4.

[3] Conference: Mittal, R., Balasubramanian, L., Sontakke, A., Parthasarthy, H., Narayanan, P., Sabbarwal, P., Parekhji, R. A. (2011) DFT for Extremely Low Cost Test of Mixed Signal SOCs with Integrated RF and Power Management. In *Proceedings of IEEE International Test Conference*, Anaheim, CA, USA, 18th-23rd Sep. 2011 (USA: IEEE), 1-10.

[4] Conference: Parekhji, R. A., Nicolaidis, M. (2013) SOC Design Readiness for Automotive Applications. In *Proceedings of Design Automation & Test in Europe Conference & Exhibition (DATE),* Grenoble, France, 18th-22nd Mar. 2013 (USA: IEEE), 1-4.

[5] Balasubramanian, L., Shankar, R., Lele, A. R., Sankaran, V. K., Bhat, S., Hegde, V. S. (2014) Surprise or Shock? Transistor level functional analysis of digital circuits and systems are still needed! In *Proceedings of CDNLive India*, Bangalore, India, 12th-13th Aug. 2014, (USA: Cadence).

[6] Balasubramanian, L., Shankar, R., Bhat, S., Hegde, V. S. (2018) Novel Pulse Width Insensitive Design and Verification Methods. In *Design Special Proceedings of Design Automation Conference (DAC)*, San Francisco, CA, USA, 24th-28th Jun. 2018 (USA: IEEE).

[7] Surendran, S., Ramakrishnan, V. (2018) Timing Modeling Methodology for Formal Verification Tools. In *Designer Special Proceedings of Design Automation Conference (DAC)*, San Francisco, CA, USA, 24th-28th Jun. 2018 (USA: IEEE).

[8] Surendran, S., Schneider, B., Rajmohan, K., Schneider, M., Fritz, S. (2018) Formal & Simulation Method to Detect Unstable States in Asynchronous State Machines. In *Designer Special Proceedings of Design Automation Conference (DAC)*, San Francisco, CA, USA, 24th-28th Jun. 2018 (USA: IEEE).