Unveiling the Core Truth: Advanced Glitch Power Analysis and Optimization Using Statistical Methodology

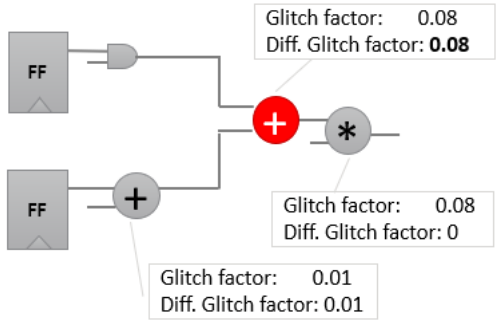
Chenyang Zhang Sanechips Co.
Chen Lin Sanechips Co.
Yuchao Liu Sanechips Co.
Jinbo Zhu Sanechips Co.
Chao Gu Sanechips Co.
Zhongming Hou Ansys Inc.

62 THE CHIPS TO SYSTEMS CONFERENCE

SPONSORED BY CEDA IEEE Council on Electronic Design Automation sigda acm

Background and Motivation

Glitch power has significant impact on the chip design and needs to be addressed in early design stage



Glitch power refers to the additional power consumption caused by glitches in the circuit, which are usually caused by logic contention, metastability or other timing issues. It could cost up to 40% of total power. It is too late and hard to do algorithms or architecture change in P&R stage. So, it is important to identify such risk and optimize the design in RTL stage.

Glitch power in different circuit

Circuit	% glitch
alu4	25.7
apex2	29.2
apex4	30.3
bigkey	29.6
clma	24.2
des	45.4
diffeq	5.8
dsip	29.9
elliptic	12.2
ex1010	35.0

The difficulty to do Glitch analysis in RTL stage

The glitches are caused by the imbalance of combinational signal timing paths. The combinational logic and wire delay are two main factors. Industry tends to use P&R engine to simulate the backend flow and estimate the delay. But it is very time-consuming and complicated process.

We have developed a novel flow based on uniform delay aware glitch engine and statistical methodology to estimate the P&R wire delay. We achieve very good accuracy, and the flow is much faster and easy to use.

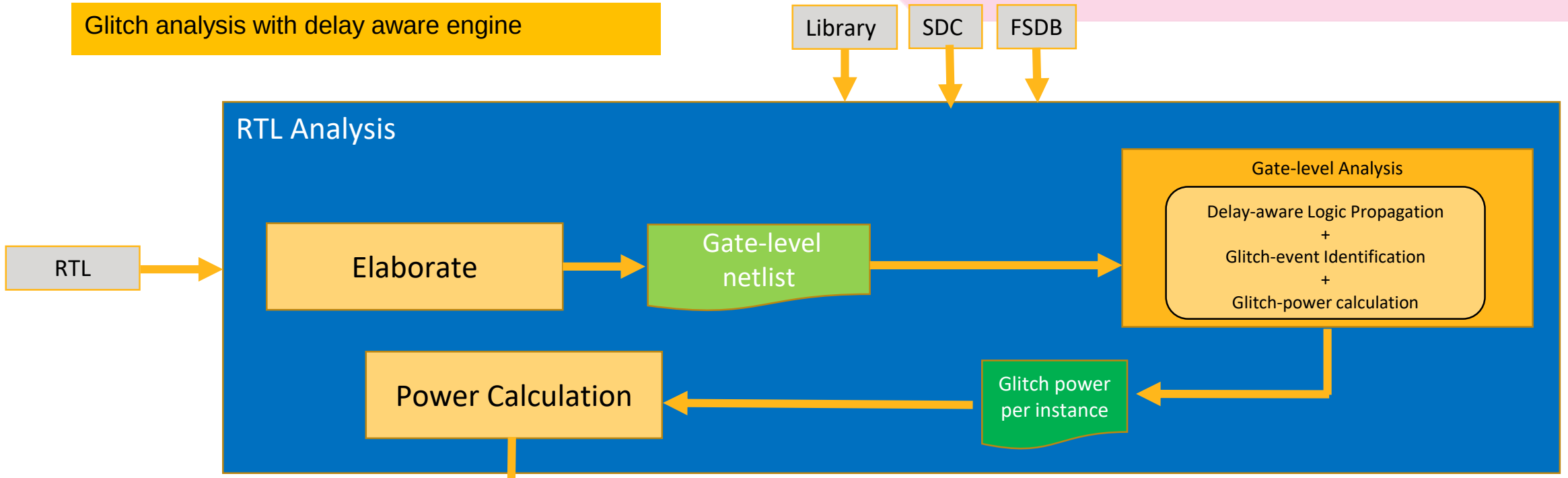
Source: Synopsys

Main Idea: using statistical scale factor to estimate RTL glitch propagation

The P&R engine are driven by many constraints such as timing, performance, area and congestion. The final gate netlist is the trade-off of these constraints. Most backend teams use the same flow and scripts for their design in the same technology node.

We experimented with making power scaling between P&R delay glitch and uniform delay glitch with the statistical methodology. But the question is: could we use the same scale factor for all designs or it is just a design-dependent scale factor. In this paper, we have tested 8 blocks in a large chip. They have different logic functions but have the same technology node and same main frequency. The result is quite good when they are using the SAME scale factor. The glitch power distribution inside the block is also matched.

Methodology: RTL glitch analysis flow with statistical methodology

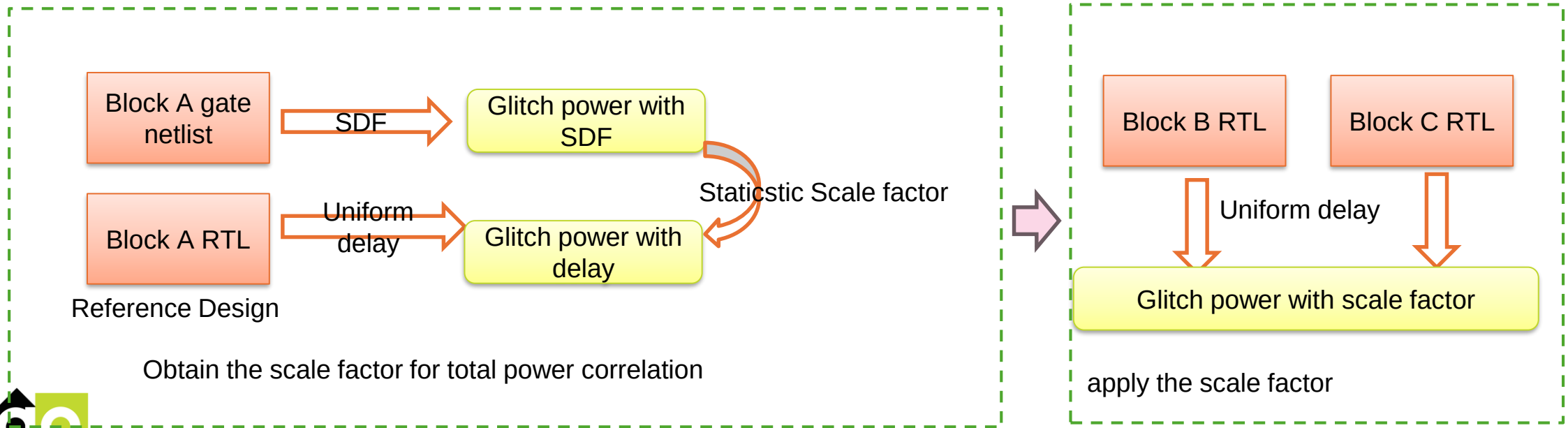


The block diagram explains a cohesive flow for computing glitch power consumption at RTL stage using state-of-the-art EDA technology for glitch pulse identification & power computation.

This approach keeps the inputs & outputs simple for RTL designers while computing delays and glitch pulse identification within EDA solution.

Methodology: RTL glitch analysis flow with statistical methodology

The following diagram explains how the flow work. First we analyze the gate netlist power with SDF. This will be our golden reference. Then we use uniform delay to do delay aware RTL glitch power analysis for the same design. A scale factor is selected in the glitch propagation engine to control the glitch pulse possibility. By adjusting the scale value, we can make the total power and the golden reference power matched. Then we apply this scale factor to other designs.

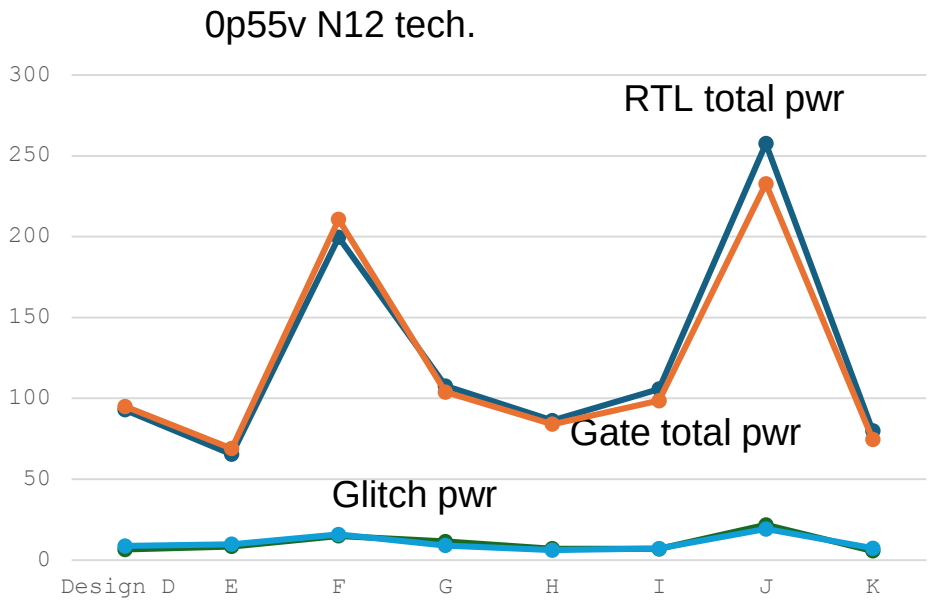


Methodology: RTL glitch analysis flow with statistical methodology

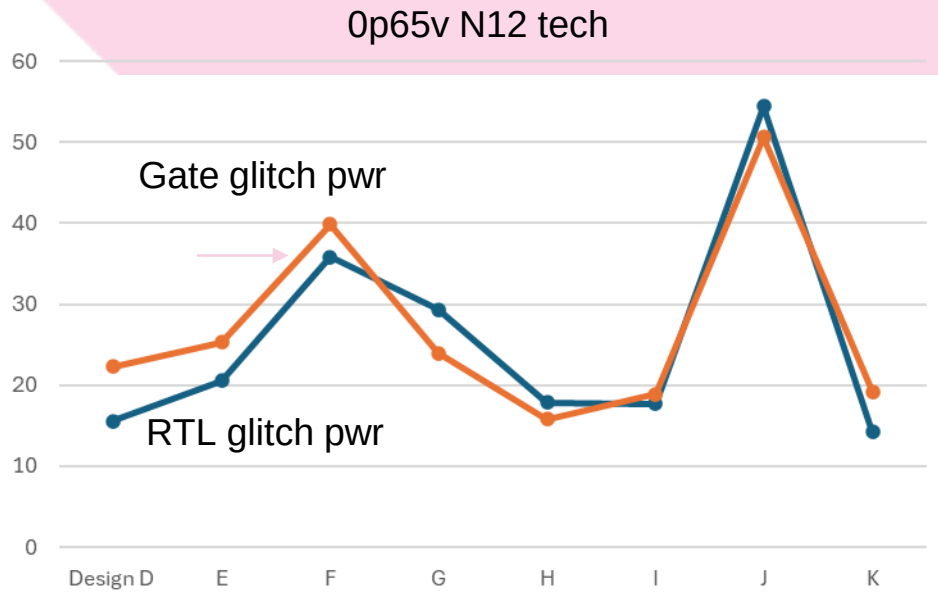
More discussion on glitch scale factor

- The reason we can use statistical scaling to estimate RTL glitch power is that the impact of routing delay on glitch power has a statistically significant average effect.
- We have tested N12 and N7. The scale factor need to be recalibrated with gate netlist power when the technology node changed. Fortunately the new scale factor can still be applied to all blocks in the new node.
- We also expect that even in the same technology node, if the change of frequencies lead to completely different P&R strategies, the scale factor will also need recalibration.

Evidence: Power analysis Result and Correlation



		PACE with scaled glitch		gate netlist			
corner	design	Glitch pwr	Total Pwr(mW)	glitch pwr	Total Pwr	glitch diff.	total diff.
0p65v	Design D	15.548	188.6	22.264	198.168	-30%	-4.83%
	E	20.516	173.88	25.3	195.04	-19%	-10.85%
	F	35.788	389.16	39.836	427.432	-10%	-8.95%
	G	29.348	232.76	23.92	231.472	23%	0.56%
	H	17.848	179.4	15.824	176.64	13%	1.56%
	I	17.664	225.4	18.86	212.704	-6%	5.97%
	J	54.464	550.16	50.6	496.984	8%	10.70%
	K	14.168	165.6	19.228	170.016	-26%	-2.60%
0p55	Design D	6.486	92.92	8.7032	94.9808	-25%	-2.17%
	E	8.418	65.412	9.752	68.9724	-14%	-5.16%
	F	14.904	199.64	15.824	210.68	-6%	-5.24%
	G	11.5	107.64	8.9516	103.868	28%	3.63%
	H	7.0288	86.296	6.1456	83.9868	14%	2.75%
	I	6.9552	105.8	7.1392	98.532	-3%	7.38%
	J	21.804	257.6	19.136	232.76	14%	10.67%
	K	5.612	79.856	7.268	74.6488	-23%	6.98%



- The same scale factors are applied to obtain better correlation on Total power .
- The RTL glitch power and gate glitch power have the same trend and distribution in different blocks.
- Combination of RTL stage glitch analysis & scaling methodology enables fast, early & accurate glitch analysis that can support optimization.

Evidence: performance impact

Design Size 3.6M gates

Technology Node N7

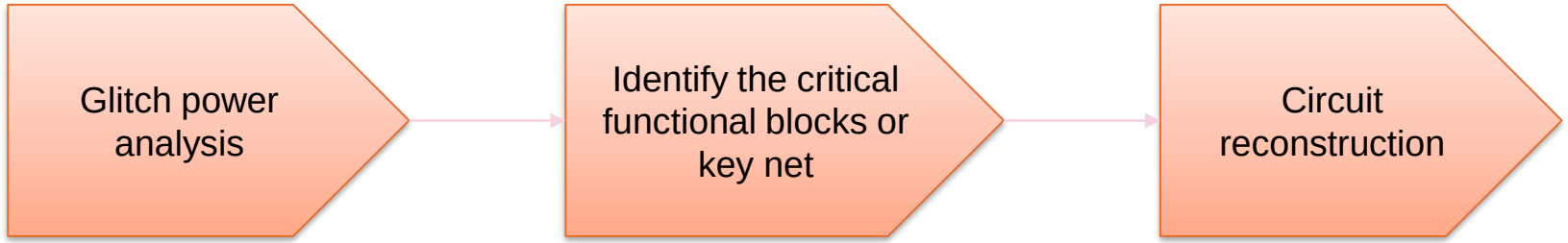
Average power(8us window)

runtime(no glitch)	run time(with delay aware glitch)	run time(Glitch flow with P&R engine)
15:22 min	15:35 min	hours

Conclusion: the delay aware propagation glitch analysis has minor impact on performance but it saves much time on P&R flow.

Value: Make glitch power more predictable and opportunity to optimize at RTL stage

- In order to reduce glitch power, we can do the following
  - PPA tradeoff and circuit reconstruction
  - backend means such as delay insertion, placement and wire length control
- With the statistical methodology, the P&R step is missing. On the other side, the RTL designers are more familiar with algorithms and architectures. It is more feasible for them to do PPA tradeoff to control the risk of glitch power and avoid big surprise in the P&R stage.



Value: Make glitch power more predictable and opportunity to optimize at RTL stage

- The delay aware glitch power with statistical methodology can also be used to analysis glitch power in hierarchy level. According to the principles of statistics, the glitch power variation will be increased when the total nets quantity is reduced. For very small logic, the glitch power is more depends on its final physical implementation. But the relative power accuracy is still enough for us to identify the glitch critical functional blocks since they have large combinational logic. Then we can choose different implementation and compare their PPA. i.e. Parallel multiplier and pipelined multiplier.

```
1. Summary
======================
1.1 Number of glitch source instances in design:        899
1.2 Number of glitch source instances with glitch power > 5 % of total glitch power:    0
1.3 Number of glitch source instances with self-glitch power > 5 % of total glitch power:       0


2. Detailed Report
======================

2.1 Top 10 Glitch sources (Highest impact source being 1st)
Rank    Self-Glitch_Power       Downstream_Glitch_Power Self+Downstream_Glitch_Power    Glitch_Source_Instance_Name     File_Name       Line_Number
-------------------------------------------------------------------------------
1       3.72e-08        1.37e-06        1.41e-06        /top/cpu/arc_top/d_cpu_isle/id_quarc/id_xy_mem/U_d_xy_acu/#c1602        /nfs/data/design/rtl/d_xy_acu.v 3405
2       1.48e-08        6.11e-07        6.26e-07        /top/cpu/hw_fft_top/hw_fft/#c1828       /nfs/data/design/rtl/hw_fft.v   415
3       5.59e-07        9.66e-09        5.69e-07        /top/cpu/arc_top/d_cpu_isle/id_arc_ram/id_xy_rwrp/U_d_b0_xdata_mem/#m1  /nfs/data/design/rtl/d_xy_ram32.v       154
4       3.83e-08        4.45e-07        4.83e-07        /top/cpu/arc_top/d_cpu_isle/id_quarc/id_xy_mem/U_d_xy_acu/#c3104        /nfs/data/design/rtl/d_xy_acu.v 2293
5       1.06e-07        2.44e-07        3.50e-07        /top/cpu/arc_top/d_cpu_isle/id_quarc/id_ueconversions/PFtoFloat/#b10    /nfs/data/design/rtl/PFtoFloat.v        33
6       8.48e-09        2.83e-07        2.91e-07        /top/cpu/arc_top/d_cpu_isle/id_quarc/id_alu/id_xalu/#c2979      /nfs/data/design/rtl/d_xalu.v   2687
7       5.23e-08        2.34e-07        2.86e-07        /top/cpu/arc_top/d_cpu_isle/id_quarc/id_xy_mem/U_d_xy_acu/#c3233        /nfs/data/design/rtl/d_xy_acu.v 2268
8       5.59e-08        2.12e-07        2.68e-07        /top/cpu/arc_top/d_cpu_isle/id_quarc/id_ue/TransEngine/arg_by_int/norm_d/#c36   /nfs/data/design/rtl/fops.v     510
9       2.05e-07        3.44e-08        2.39e-07        /top/cpu/arc_top/d_cpu_isle/id_quarc/id_ue/TransEngine/arg_by_int/#c269 /nfs/data/design/rtl/TransEngine.v      921
10      2.30e-08        2.16e-07        2.39e-07        /top/cpu/arc_top/d_cpu_isle/id_quarc/id_ueconversions/IntToFloat/#c361  /nfs/data/design/rtl/IntToFloat.v      16
```

Summary: Conclusion

- Glitch power is a big challenge in modern low power design. But it is difficult to analyze it in RTL stage. The flow using P&R engine is complex and time-consuming.
- This solution provides fast & accurate glitch power analysis in RTL stage.
- By using statistical methods to evaluate the impact of P&R on glitch propagation, it helps designer to quickly evaluate the risk of glitch power and identify the high glitch power logic blocks. The designer can try to balance the logic or optimize the algorithms based on the analysis result.
- There is no need to do P&R simulation. So it is much faster and easy to use.
- Identification of glitchy logic modules at RTL stage enables significant glitch optimization opportunity that's otherwise lost if we wait till P&R stage.