Designing Power-Efficient & Safe Autonomous Driving Devices: Using Custom Techniques to Quickly Identify/Fix Hotspots at RTL

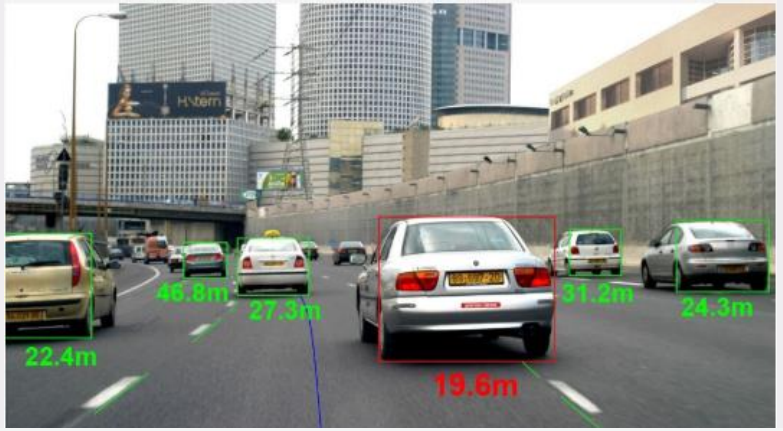
Erez Iluz

Motivation

- Mobileye is currently developing its sixth generation SoC (EyeQ®6), to act as the central computer sensor for Fully Autonomous Driving vehicles that will hit the road in 2021.

- EyeQ® family is required to support complex vision processing, and still maintain **Low Power Consumption** especially while located on the windshield.

- Therefore, several high power consuming accelerators were targeted for RTL changes in order get their power decreased significantly.



Real-time Collision Avoidance

Challenges

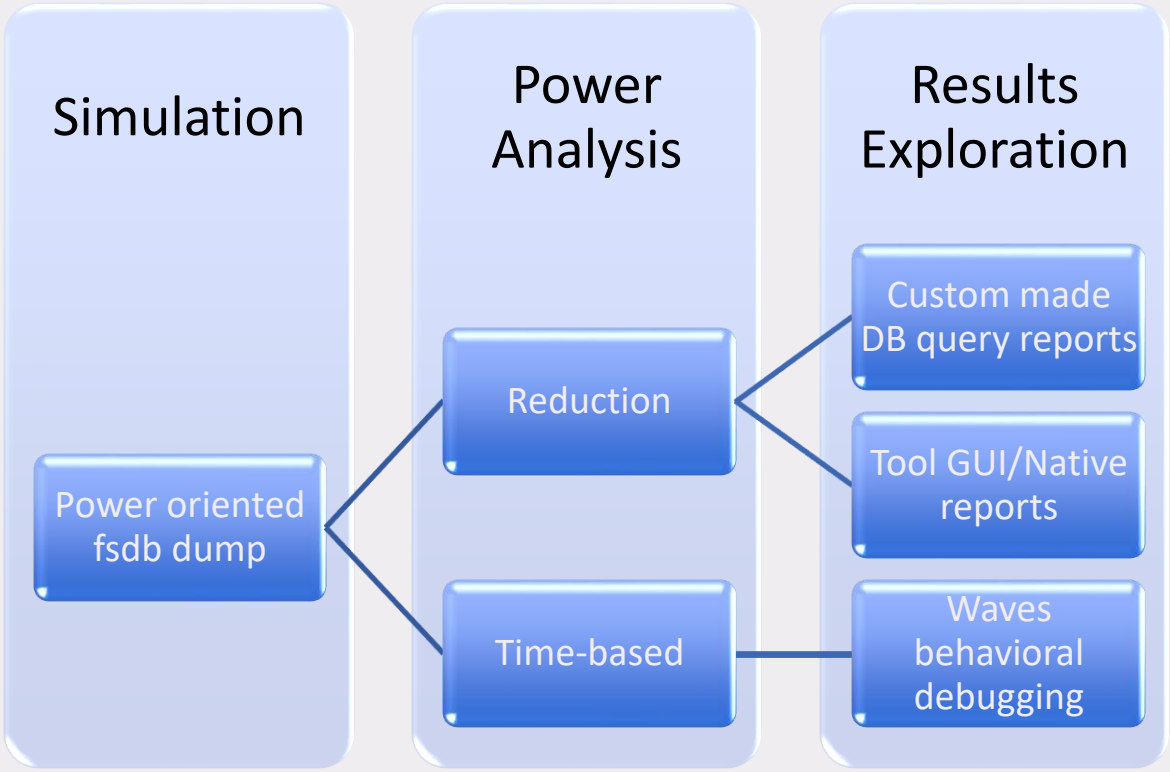
- Automotive industry → **Functional Safety** → Common low power techniques not allowed.
- High frequency (2GHz) → Tight timing constraints.
- Fast RTL change ←→ Power analysis turnaround time → Tight TTM.
- High active scenario → More difficult to find areas for power improvement.

Methodology

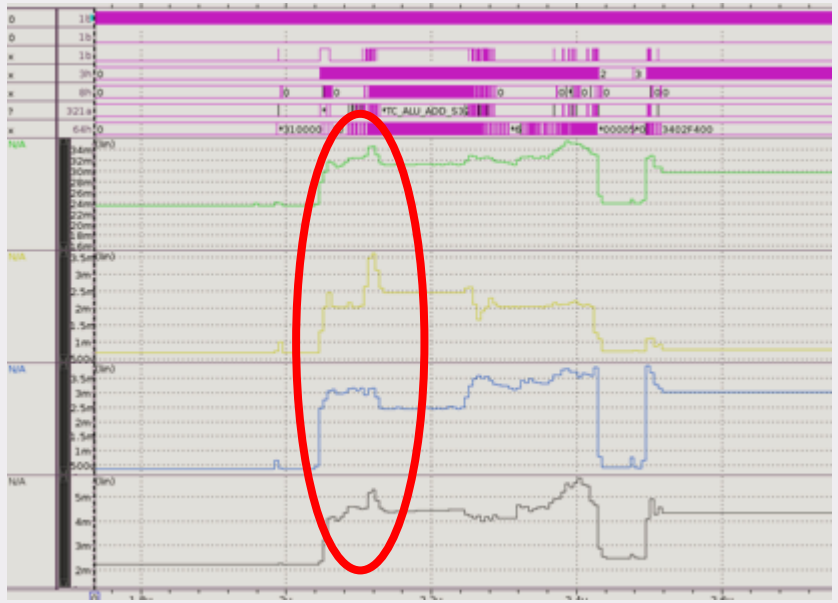
- It was decided to focus on specific cases of:
  - Low active flops
  - Poor CGs enable conditions\locations
  - Non observed logic cones

- **Tool DB query commands** were manipulated to fit our needs.
- Several dedicated reports were created using these commands → Exported to Excel files for more effective data sharing and review.
- Managed to pinpoint exact few desired spots of high, non optimized power consumption, address them properly and reduce power.

Flow



	Instance name	Total Flops	Gated Flop	CGEE %	Wasted Clock Pin
3		271	0	100.00%	-
18		504	0	100.00%	-
296		2436	2359	-	-
412		241	214	23.82%	3.19uW
454				23.61%	161mW
455		12172	12172	30.41%	23.8mW
583		24528	24528	2.41%	13.3mW
584		1533	1533	2.41%	819uW
585		1533	1533	2.41%	819uW
586		818	818	0.00%	173uW
587		149	149	100.00%	-
588		118	118	0.00%	18.7uW
589		40	40	0.00%	90.9uW
590		156	156	100.00%	-
591		41	41	100.00%	-
592		1533	1533	2.41%	832uW



Flow – Power Suite

- Perl based power suite around the tool for power estimation and reduction at RTL stage.
- Significantly ease the usage especially with regards to data preparations and tool configurations.
- User fills a short template and the rest is straight-forward simple commands per stage.
- The suite includes the following self-developed key features:
  - ➢ Links and loads relevant technology libraries and physical calibration data.
  - ➢ Generates design file list and clocks list automatically.
  - ➢ Power oriented FSDB dump utility added to current testing flow.

DB Query Results – Examples

Data activity vs. Clock activity:

Inst	Total Power [uW]	Total Dynamic Power [uW]	Totale Leakage Power [uW]	Register Dynamic Power [uW]	Combinational Dynamic Power [uW]	Memory Dynamic Power [uW]	Clock Dynamic Power [uW]	Clock-Activity	Data-Activity
InstA	302.362	265.469	36.893	4.617	0.216	0	260.636	0.1	0
InstB	70521.862	68286.83	2235.032	32139.48	25234.35	0	10913	0.95	0.145
InstB/InstAA	39376.194	37591.16	1785.034	9015.43	24631.67	0	3944.06	0.87	0.2
InstB/InstAA/InstAAA	18220.342	17535.125	685.217	4568.115	10994.98	0	1972.03	0.91	0.19
InstB/InstAA/InstAAA/InstABC	8724.573	8431.941	292.632	2326.086	5119.84	0	986.015	0.9	0.09

CG efficiency per Gate:

	A	B	C	D	E	F	G	H	I	J
1	Inst Name	Gater Type	Impact	En Eff	Dwnstr Power	Gated Insts	Gated Bits	En Net	File	Line
2	#m78	inferred	0.943	0	0.038		15936		File1.v	217
3	#m870	inferred	0.0234	0	0.000865		332		File2.v	320
4	#m915	inferred	0.023	0	0.00151		320		File3.v	496
5	#m880	inferred	0.0191	0	0.000556		256		File4.v	614

DB Query Results – Examples

Non-gated/Instantiated CG for large Register banks

	A	B	C
1	Flop	Width	CG Type
2	FlopXa[1:0][14:0][255:0]	7360	Instantiated
3	FlopYa[465:0]	466	None
4	FlopZa[1:0][4:0][15:0]	160	Instantiated
5	FlopXb[127:0]	128	Instantiated
6	FlopYb[112:0]	113	None
7	FlopsZb[3:0][15:0]	64	Instantiated

CG Enable efficiency per hierarchy:

1	Instance name	Total Flops	Gated Flops	CGEE %	Wasted Clock Pin power
454	InstA	287727	286513	23.61%	161mW
455	InstA/InstAB	12172	12172	30.41%	23.8mW
583	InstA/InstAC	24528	24528	2.41%	13.3mW
584	InstA/InstAC/InstACD	1533	1533	2.41%	819uW
585	InstA/InstAC/InstACD/InstACDA	1533	1533	2.41%	819uW
586	InstA/InstAC/InstACD/InstACDA/InstX	818	818	0.00%	173uW
587	InstA/InstAC/InstACD/InstACDA/InstX/InstY	149	149	100.00%	-

Results

After several analyses <-> RTL fix iterations, we managed to get outstanding results:

- **Clock gating efficiency improved** **50% → 80%**
- ODC clean → ~15% power saving by reducing unobserved logic.
- Reduced low active flops number 20% → 5%. Power by 10%.

Summary

- Managed to reduce power by 25% in the high-activity mode of operation → Design teams can find significant amount of low-activity logic even in high-activity modes.
- Not jeopardizing functional safety and high frequency timing requirements.
- Tight TTM → Short TA time → Examine only first few items per sheet.
- **New RTL-based power methodology that used custom reports, built on top of tool reduction analysis DB, that targeted fixing specific areas of low activity flops or non observed logic**

Current & Future Work

- Currently, we are building a regression suite that launches a reduction analysis per accelerator once/twice a week.
- Every regression run (based on our Power suite):
  - Imports the latest RTL and elaborates it.
  - Runs selected test -> dumps FSDB
  - Runs reduction analysis
  - Sends generated Excel results file to accelerator owners (+ Me ☺) email for review.
- Collaborate design team leaders and architects in order to create bank of power aware tests for better non optimized power consumption areas pinpoints.

Thank You!

Drive Safe