Choosing the best Standard Cell Library without falling into the traps of traditional benchmarking methods

Introduction

Assessing the comparative performances of several Standard Cell Libraries in a reliable way is a tricky project as it deals with statistical issues.

The methodology traditionally used in the industry to benchmark Standard Cell Libraries is the so-called "cell-by-cell" approach. It consists in taking one or two basic cells, such as a NAND2 and/or a FLIP-FLOP, and comparing their area, dynamic power consumption, leakage and speed. This method has three major drawbacks:

- the "cell-by-cell" approach only assesses a few cells, which are not necessarily representative of the user's SoC.
- this "cell-by-cell" approach is even less relevant for benchmarking libraries with different structures such as a traditional Complex Cell Stem Library (CCSL) and a Reduced Cell Stem Library (RCSL)
- this approach does not take into account the implementation issues linked to the logic flow. Through a cell-by-cell approach, the Standard Cell Libraries are not compared in terms of ease-of-use and time-for-convergence during the four implementation steps of logic synthesis, placement, clock tree synthesis and routing.

The objective of this paper is dual. The first objective is to demonstrate that the « cell-by-cell » approach to compare libraries is inconsistent with actual performances results obtained after P&R of libraries on a logic circuit. The second objective is to present benchmarks and methods to compare efficiently and reliably different libraries with different architectures (e.g. CCSL versus RCSL).

The suggested benchmarks and methods are:

- The "SOFIA" benchmark which is based on 6 representative cells, along with a statistical distribution of these cells,
- "Thalie", a SOFIA based predictor of performances for the targeted logic block using the targeted library,
- The "Red" benchmark based on Motu Uta, a standard logic block: the comparison is made through the full logic flow,

- And finally the Try&Compare approach, which is based on a trial on the targeted logic block using appropriate scripts from synthesis to routing (the "4 seasons" scripts).

Each of these methods enables to compare area, leakage, dynamic power and speed of several Standard Cell Libraries with different accuracies. But the last two approaches also provide a comparison of the ease-of-use and time-for-convergence of the library.

For reasons of protection of confidentiality, all the values given in this article are close to but not the exact values of a specific library.

To visit our web page Standard Cell Benchmark:
http://www.dolphin.fr/flip/sesame/sesame_benchmark.php

From the « cell-by-cell » approach to SOFIA benchmark

Comparing two standard cell libraries (e.g. a high density library with a general purpose library) in 0.18 µm with the NAND2 cell indicates that the total gain expected using the high density library is 12 % for the area, with a dynamic power consumption 12 % better compared to the general purpose library:

NAND	Area (µm²)	Dynamic power (µW/MHz)
	Value	Value
high density library @ 1.8 V	9,22	0,0131
general purpose @ 1.8 V	10,48	0,0149

On actual cases (which means on logic blocks after P&R) using both libraries, the results show a larger gain in terms of area (around 35 – 45 %) with a gain in terms of dynamic power consumption of around 5 %.

In a different illustration, if we compare a Reduced Cell Stem Library (RCSL) with a Complex Cell Stem Library (CCSL) using one FLIP-FLOP cell, what we obtain is a gain in terms of area of 45 % with a power consumption divided by 2!

FLIP-FLOP	Area (µm²)	Dynamic power (µW/MHz)
	Value	Value
RCSL @ 1.8 V	27,66	0,0457
CCSL high density library @ 1.8 V	48,40	0,0959

If we compare the same two libraries using the NAND2 cell, what we obtain is a gain in terms of area of 15 % with a loss in term of power consumption of 30 %!

NAND2	Area (µm²)	Dynamic power (µW/MHz)
	Value	Value
RCSL @ 1.8 V	7,90	0,0172
CCSL high density library @ 1.8 V	9,22	0,0131

On actual cases (which means on logic blocks after P&R) using both libraries, the results show a smaller gain in terms of area (around 20 %) with an improvement in terms of dynamic power consumption of around 50 %.

These three examples demonstrate that the conclusions made from a simple cell-by-cell comparison give us an indication which can be wrong!

For a better accuracy, the SOFIA benchmark uses 6 cells representative of the typical paths in a majority of logic circuits. Each cell is weighted depending on the percentage that it represents in the path, obtained from a large sample of circuits. These weights vary depending on the nature of library (the traditional CCSL approach, or the RCSL approach like SESAME from the Dolphin Integration offering).

Area

n µm²	FlipFlop (dfc3)		Simple boolean (nd21)		Complex boolean (anr2)		Multiplexer (mx22)		Adder (add2)		Inverter and buffer (in01)		FoM area	FoM area normalized
	Value	Weight	Value	Weight	Value	Weight	Value	Weight	Value	Weight	Value	Weight		
RCSL @ 1.8 V	27,66	14%	7,90	35%	13,83	29%	15,80	9%	43,46	3%	3,95	21%	70,13	1,74
CCSL HIGH DENSITY LIBRARY @ 1.8 V	48,40	14%	9,22	29%	13,83	40%	18,44	2%	57,62	1%	6,91	14%	59,19	1,47
CCSL GENERAL PURPOSE LIBRARY @ 1.8 V	80,33	14%	10,48	29%	20,96	40%	24,44	2%	73,34	1%	6,98	14%	40,21	1,00

Dynamic power consumption

n µW/MHz	FlipFlop (dfc3)		Simple boolean (nd21)		Complex boolean (anr2)		Multiplexer (mx22)		Adder (add2)		Inverter and buffer (in01)		FoM dynamic	FoM dynamic normalized
	Value	Weight	Value	Weight	Value	Weight	Value	Weight	Value	Weight	Value	Weight		
RSCL @ 1.8 V	0,0457	70%	0,0172	35%	0,0288	29%	0,0155	9%	0,0700	3%	0,0095	21%	19,2721	1,61
CCSL HIGH DENSITY LIBRARY @ 1.8 V	0,0959	70%	0,0131	29%	0,0237	40%	0,0200	2%	0,0138	1%	0,0094	14%	12,1584	1,02
CCSL GENERAL PURPOSE LIBRARY @ 1.8 V	0,0919	70%	0,0149	29%	0,0290	40%	0,0206	2%	0,1595	1%	0,0096	14%	11,9669	1,00

Comparing the three libraries, the results obtained with SOFIA are in line with the experience on real circuit after P&R. In fact:

- the gain in terms of area between the high density library and the general purpose library is around 47 %,
- the gain in terms of power consumption between the high density library and the general purpose library is of some %,
- the gain in terms of area between the RCSL library and the CCSL high density library is around 20 %,
- the gain in terms of power consumption between the RCSL library and the CCSL high density library is over 60 %.

The SOFIA benchmark provides an objective comparison at the pre-synthesis level of the performances of libraries (area, dynamic consumption, leakage, speed) in just 30 minutes. The results we show, and the experience we have on different logic blocks, underline that SOFIA provides an accurate comparison among libraries, which is not the case with the “cell-by-cell” approach.

The "Thalie" formula to compare libraries on a target SoC

In order to obtain a measurement of the performances of a given library on the User's SoC, the Thalie formula is proposed. This formula enables the User to compute the area of a logic bloc starting from its complexity in terms of gates and the SOFIA benchmark.

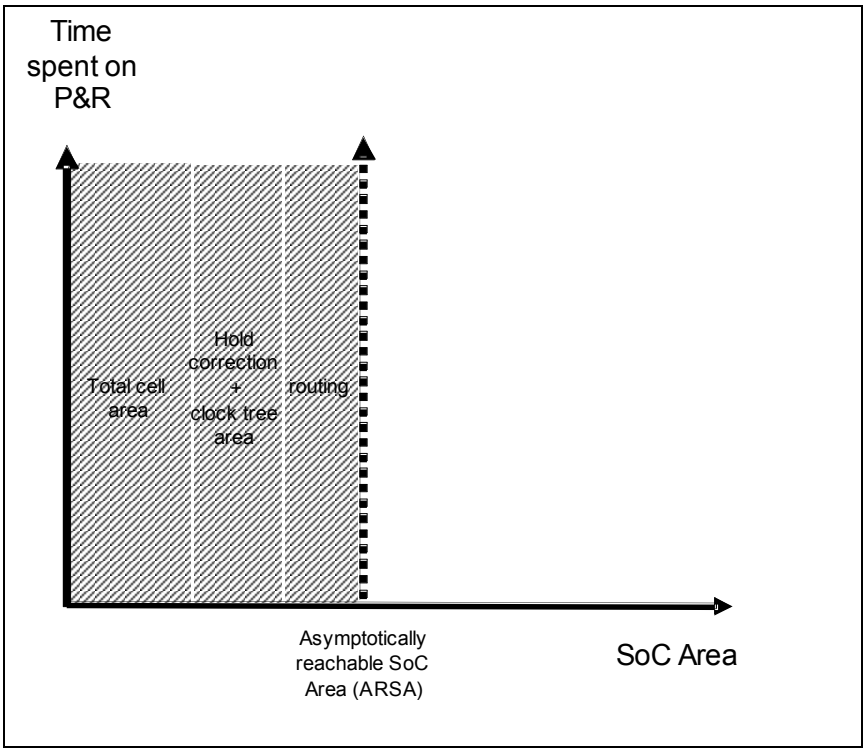
How to predict the performances of a logic block in terms of area

The smallest silicon area achievable for a given design remains a question mark for the majority of designers.

Let us name this smallest achievable area the "Asymptotically Reachable SoC Area" or "ARSA".

The actual reachable SoC Area will depend on the ARSA, but also on additional constraints (e.g. form factor) and the time budget allocated to the Place and Route. The Thalie formula is dedicated to the ARSA evaluation of a logic block. Thalie can estimate ARSA starting from various parameters describing the logic block (result of a logic synthesis, estimation of number of flip flops…). The accuracy of the estimation will depend on the accuracy of the input parameters

Area Performance after P&R predicted starting from the SOFIA Benchmark



The goal of this approach is to select the minimum asymptotically achievable SoC area achievable in P&R.

The input parameters of Thalie are:

1. Complexity of the logic block in number of gates
2. Average fanout and size of the average buffer for the Clock tree
3. Hold constraints for the FlipFlop with scan for the input D and Scan In
4. Vertical track of the library
5. Horizontal track of the library
6. Number of metal layers

Based on input 1, the Thalie formula estimates the "Total cell area" after synthesis of the targeted logic block. This is done by using the distribution of the cells provided by the weight of SOFIA.

Based on inputs 2 and 3, the Thalie formula estimates the area of the Clock tree. In fact, starting from the complexity of the logic block and the weight of the FlipFlop in a design, it is possible to estimate the number of FlipFlops in the design. With the area of the average buffer for the clock tree and the average fanout, it is possible to estimate the number of buffers to be used for the clock tree.

In the same way, starting from the number of FlipFlops and the hold constraints, it is possible to estimate the number of cells to be added in order to correct all the hold violations during P&R.

Based on inputs 4, 5 and 6, the Thalie formula estimates the number of nets which can be routed (available routable net) within the cells. In order to check if the routing can be completed successfully within the cells, the "available routable net" is compared to the actual number of nets to be routed for the target design and the final area of the logic block is finally computed.

The table below shows an example of the Thalie implementation on the Motu Uta standard (see following chapter for the definition of Motu Uta):

Digital block (Motu Uta)	160000	number of gates
Clock rate	100	MHz
Switching activity	30	%
Power supply	1,8	V
Process	TT	
Temperature	25	°C

Starting from the SOFIA, we computed the number of instances per cell type.

Distribution for the 6 cells of SOFIA

		Weight in SOFIA
=> number of FlipFlop (7,5 nand2 equivalent)	9314	12%
=> number of simple boolean (nand2)	23950	32%
=> number of complex boolean (1,8 nand2 equivalent)	19958	26%
=> number of mux (2 nand2 equivalent)	5988	8%
=> number of adder (5,5 nand2 equivalent)	1996	3%
=> number of inverter/buffer (0,5 nand2 equivalent)	14636	19%

This provides a Total area of the cells of 946297 μm^2 and a dynamic power consumption of 86.8 mW at 100 MHz.

With the number of instances per cell, we are able to compute the number of nets of the circuit after synthesis, which is equal to 82770 nets.

With the number of FlipFlop, we anticipate the size of the clock tree and the size due to the hold violation corrections.

In order to compute the available routable net, we need the information on the structure of the library and the metal Top of the SoC:

Vertical track	0,56
Horizontal track	0,56

Number of metal layers for routing, including metal TOP	6

Finally, we compare the 82770 nets to be routed with the available routable net and we estimate the final ARSA of the circuit: in this case the ARSA is equal to 1.15 mm².

This means that with a medium effort during P&R, we can achieve ARSA + 10 % in terms of area.

The results we obtain with the Motu Uta after P&R is 1.26 mm², which corresponds to the 1.15 mm² + 10 %.

The second conclusion is that, in only a few minutes, the THALIE formula provides the User with a estimation of the performances of a Standard Cell Library on his targeted circuit with an accuracy of 10 % in terms of area and 20 % in terms of power consumption.

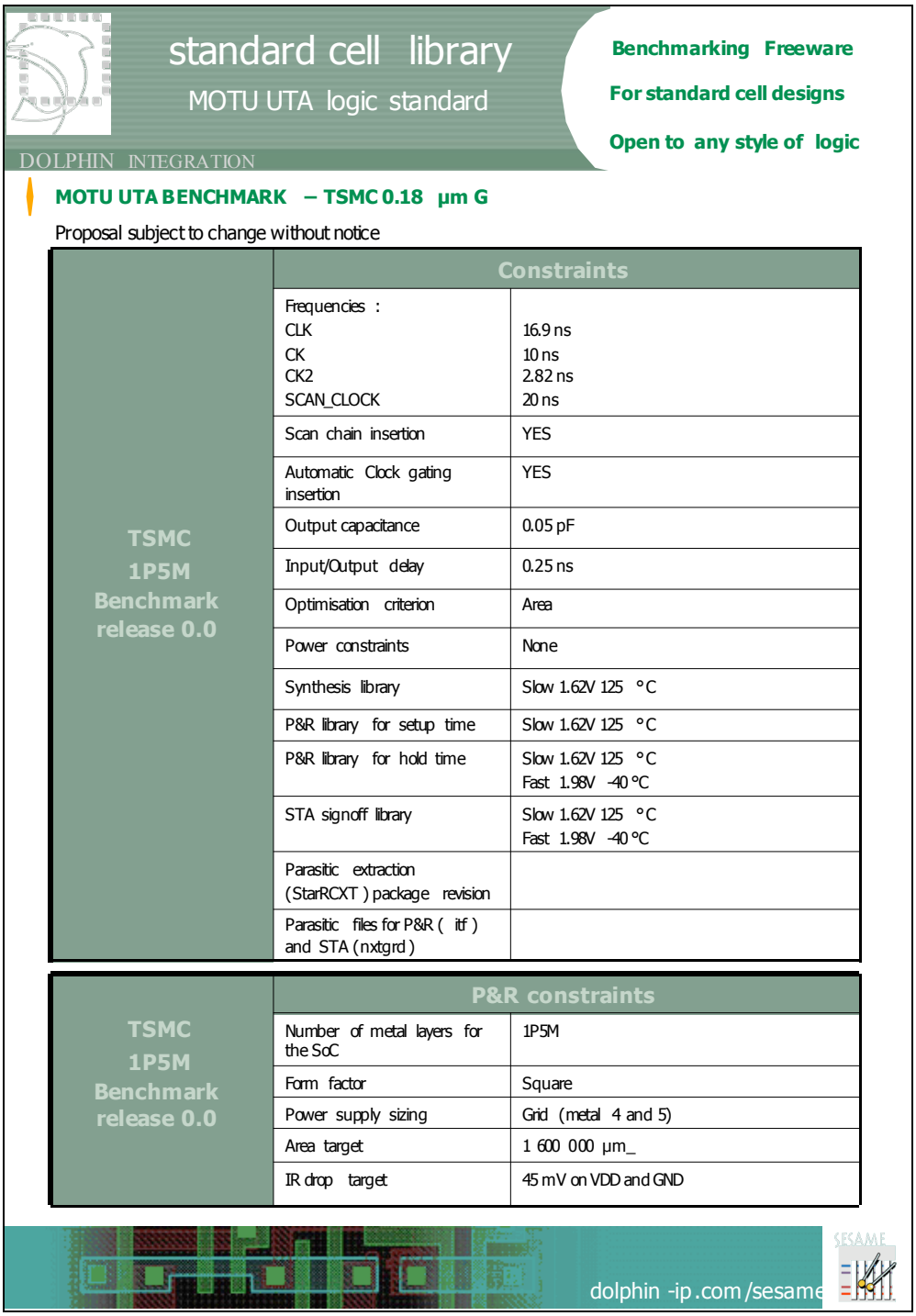
The Red benchmark applied to the Motu Uta logic standard

With SOFIA and Thalie, it is possible to perform a fair comparison of the performances of two different libraries and assess the performances of a targeted SoC.

The missing dimension of a comparison based on SOFIA and Thalie only is that the libraries are not compared in terms of ease-of-use and time-for-convergence during the four implementation steps of the logic flow: logic synthesis, placement, clock tree synthesis and routing.

Motu Uta is a public logic standard (logic block in RTL), which can be downloaded for free from the Dolphin Integration website. The purpose is to enable benchmarking of performances of any Standard Cell Library by performing synthesis, placement, clock tree synthesis and routing based on the Red Benchmark. Thanks to its structure, Motu Uta is representative of typical logic blocks in all dimensions: area, power consumption and speed (for more information, see http://www.dolphin-ip.com/flip/sesame/benchmark/sesame_motuuta.php).

The Red benchmark is a list of constraints providing all the needed information to set the constraints for Motu Uta through the 4 steps of logic flow:



The third conclusion is that, through Motu Uta, the comparison between two libraries is not only made on electrical or physical performances (timings, power consumption or area) but also on the performances in terms of implementation (time to silicon, etc…).

Benchmark on the targeted SoC through the Try & Compare

With Motu Uta, the comparison between two different libraries of standard cells is made for all performances. Nonetheless, there are two cases in which the SoC integrator may wish to perform further verifications.

The first case is for applications with performances which challenge a given library in terms of speed. It is then important to check that each library effectively meets the speed constraint of the targeted logic block.

The second case is for very specific designs, with unusual distributions of standard cells, such as RTL code based exclusively on latches or asynchronous logic blocks.

The “Try & Compare” is a structured methodology enabling to compare truly and efficiently the performances of standard cell libraries. The performances of any logic block depend on: the library, the benchmark and the SoC Integrator’s capability for floorplanning and optimizing the implementation of logic blocks using the P&R EDA solutions. The optimization rests on the implementation during the following four steps: synthesis, placement, clock tree synthesis and routing.

For this purpose, the Try & Compare evaluation kit includes all the necessary library views to proceed to a performance assessment on any logic circuitry including the public logic standard Motu Uta (see above) together with scripts enabling a full optimization of the library usage at each implementation step:

- The **Chun Ji** script is dedicated to the optimization of the Data Path Synthesis,
- The **Xia Ji** script is dedicated to the optimization during placement,
- The **Qiu Ji** script is dedicated to the optimization of the Clock Tree,
- The **Dong Ji** script is dedicated to the optimization at Routing level.

Such scripts are optimized for a given library.

Conclusion

Approach	Compare 1 cell (ex. NAND2)	SOFIA	MOTU UTA	THALIE	Try & Compare
In average or SoC specific	In average	In average	SoC in average	SoC specific	SoC specific
Assessment	Subjective	Objective	Objective	Objective	Objective
Thoroughness	Pre-synthesis	Pre-synthesis	Post-synthesis and Post-P&R	Post-P&R	Post-P&R
Scope	Area/Speed/ Power Consumption	Area/Speed/ Power Consumption	Area/Speed/ Power Consumption /Scan	Area	Area/Speed/ Power Consumption /Scan/Congestion/IR Drop…

To visit our web page Standard Cell Benchmark:
http://www.dolphin.fr/flip/sesame/sesame_benchmark.php

Andrea BONZO,
CAE Libraries

About the Author

Andrea Bonzo serves as the Central Application Engineer for memories and standard cells for Dolphin Integration.
He is in charge of the technical interface with prospects (before sales) and with customers (after sales).
Prior to this, Mr. Bonzo was in charge of the development of analog IPs for 4 years before starting the activity in the field of the memory generators and later on the development of library of standard cells based on a Reduced set of cells.