FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Comparison of full-adder cell layouts for 90 nm CMOS technology

Ana Ferreira Araújo

Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: João Canas Ferreira

March 25, 2024

Abstract

Full-adders are an important component in many large scale systems because they perform a fundamental operation: addition. This arithmetic operation is used in numerous applications from small microprocessors to laptop computers, and its performance can affect the overall performance of the system.

The importance of addition has lead to a large number of alternative implementations, creating a wide range of architectures for the same logic function. However, different architectures may be subject to different relative performance variations when implemented in more recent technologies. Therefore, the rapid advancement of technology necessitates a performance comparison analysis to ensure that a particular topology best meets the system requirements. To ensure accurate results the corresponding layouts have also to be taken into consideration.

This study focuses on the performance comparison of four static full-adder architectures in CMOS 90 nm technology: Conventional, Mirror CMOS (CMC), Transmission gate (TG) and Complementary pass-transistor logic (CPL) full-adders. The layouts were designed and simulated with industrial-grade tools: Cadence Virtuoso and Spectre, respectively. Characterisation was performed using Liberate, including layout-derived parasitics. The implementations were compared in terms of power, delay and power-delay product for different process corners, temperatures and supply voltages.

It was found that the CPL adder presented the worst performance results of all the analysed architectures and that the Conventional adder proved to be the most robust under temperature and supply voltage variations. The TG adder shows very similar results to the Conventional adder, but with a smaller area and a lower transistor count. Finally, the CMC adder exhibits the lowest power dissipation at the expense of speed.

Keywords: adder, full-adder, delay, power, power-delay product, CMOS Digital Integrated Circuits, performance analysis.

Resumo

O somador completo é um componente importante em vários circuitos de grande escala uma vez que desempenha uma operação fundamental: a adição. Esta operação aritmética está presente em múltiplas aplicações desde pequenos microprocessadores até computadores e a sua performance pode afetar o desempenho de todo o sistema.

A importância da adição levou a uma grande variedade de implementações alternativas criando uma vasta escolha de arquiteturas para a mesma função lógica. Contudo, arquiteturas diferentes podem estar sujeitas a variações relativas de desempenho quando implementadas em tecnologias mais recentes. Deste modo, o rápido avanço da tecnologia necessita de análises comparativas de forma a assegurar que uma determinada topologia é a mais adequada para os requisitos do sistema. Com vista a assegurar resultados precisos, os layouts respetivos foram tidos em conta para este projeto.

Este estudo foca-se na comparação do desempenho de quatro somadores completos usando tecnologia CMOS de 90 nm: *Conventional*, *Mirror CMOS* (CMC), *Transmission gate* (TG) e *Complementary pass-transistor logic* (CPL). Os layouts foram produzidos e simulados em ferramentas de ambiente industrial: Cadence Virtuoso e Spectre, respetivamente. A caracterização foi efetuada usando o programa Liberate, incluido as capacidades parasitas derivadas do layout. As implementações foram comparadas em termos de potência, atraso de propagação e do produto dos mesmos para diferentes processos, temperaturas e tensões de alimentação.

Foi observado que o CPL apresenta o pior desempenho das quatro arquiteturas analisadas e que o somador *Conventional* demonstrou ser o mais robusto em relação a variações de temperatura e de tensão de alimentação. O TG apresenta resultados muito semelhantes aos do *Conventional* mas com uma menor área e um menor número de transistores. Por último, o somador CMC exibe a menor dissipação de potência comprometendo a velocidade.

Acknowledgements

The months of working on my dissertation were one of the most challenging academic experiences I have faced and I would like to thank some of the people that made it possible.

First I would like to thank my supervisor Professor João Canas Ferreira for the opportunity to further my knowledge in this area, and for his support since the start of this dissertation.

I would like to give a special thanks to my family and friends for the motivation, help and patience that they given me throughout these past years.

Ana Ferreira Araújo

"Efforts may lie, but will never be in vain."

Yuzuru Hanyu

Contents

List of Figures

List of Tables

Abbreviations

CPL	Complementary Pass-transistor
DRC	Design Rule Check
DRD	Dual-Rail Domino
FA	Full-Adder
HTML	HyperText Markup Language
IC	Integrated Circuit
LP	Low-Power
LVS	Layout Versus Schematic
PDP	Power-Delay Product
PVT	Process, Voltage and Temperature
TFA	Transmission Function Adder
TG	Transmission Gate
TGDC	Transmission Gate with Driving Capability
VLSI	Very Large Scale Integration

Chapter 1

Introduction

This chapter presents a contextual overview of designing various architecture full-adder layouts to analyse and compare their performances. Section 1.1 introduces the full-adder, some of its applications and the motivations for the design and performance evaluation. The objectives of this study are listed in section 1.2 and the structure of this document is outlined in the last section of this chapter, section 1.3.

1.1 Context

Full-adders are the basis of most arithmetic operations besides the simple addition, such as subtraction, multiplication, and division [13], integrated in larger common use devices such as the laptop computers. The improvement of this fundamental part of many larger circuits includes minimizing the number of transistors used, minimizing the power consumption and increasing the speed of operation by minimizing delay [1][15]. Considering that the full-adder is in the critical path of numerous circuits [2], improving the performance of this cell can improve significantly the performance of the larger circuit in which it is inserted.

Since each architecture has strengths and limitations, when choosing the full-adder architecture for a specific application, the designer should consider previous performance analysis of each adder and the application system requirements [7][14]. This highlights the need of having well documented performance and tradeoff analysis covering different testing and operating conditions, such as temperature and supply voltage.

Nanometer technology, such as CMOS 90 nm, causes circuits to be susceptible to environmental and physical factor variation; process, voltage and temperature (PVT) parameters cause variations that affect the circuit's overall performance and response [3], so it is important to consider the effects of these parameter variations to obtain a more detailed and complete analysis. Besides the usual individual comparisons of area, delay and power consumption of each layout, the power-delay product (PDP) is another quantitative measure that represents the "efficiency" of the tradeoff between circuit latency and power dissipation [2][6].

1.2 Objectives

Numerous authors have already presented streamlined, optimized layouts tailored to specific architectures across a range of nanotechnology scales, facilitating comparative analysis. This dissertation proposes the design of full custom cell layouts and characterisation for known architectures in CMOS 90 nm technology.

Four known static logic architectures have been chosen and respective custom layouts have been produced and characterised in terms of power, delay and power-delay product. Their advantages and disadvantages are emphasized and also compared with the results of previous authors, namely [16][10][9]. The four topologies chosen for design and analysis are the Conventional, Mirror CMOS, Transmission gate and Complementary Pass-transistor logic full-adders (chapter 2 presents the reasons for this choice).

The main particular goals of this dissertation are:

1. Design the layouts for each architecture.
2. Characterisation of each layout design in terms of power and delay on how PVT variations affect the overall performance.
3. Tradeoff analysis and comparison of the different architectures and with previous works.

By comparing the four different architectures with relation to different temperatures, supply voltages and processes, one can differentiate with more ease which is the best choice for a specific application regarding the requisites for that particular application and the compromises it allows.

1.3 Document structure

The remaining part of the document is structured as follows:

- Chapter 2 begins with a background overview of the technology and explores four full-adder designs as well as a comparison of previous proposed solutions and implementations.
- Chapter 3 presents a detailed description of the topologies under analysis, an outline of the simulation strategies and design methodology for the final layout of each cell.
- Chapter 4 focuses on the characterisation of the circuits for power and delay performances and the results are analysed and compared.
- Chapter 5 presents a summary of the document as well as a reflection on the future work.

Chapter 2

Literature review

2.1 Introduction

This chapter details the previously introduced problem and compares four different approaches of full-adder architectures. The first section is dedicated to the full-adder logic and operation that serves as basis for a better understanding of the different architectures and their properties. Moreover, it presents some basic knowledge of what it is, how it works and main applications. In section 2.3 four different architectures are presented along with some information regarding their properties and electric schematic, as well as performance assessments based on previous experiments of other authors. It is worth noting that this exposition is concise, considering that each architecture will undergo a more in-depth analysis in chapter 3. Lastly, section 2.4 is dedicated to a comparison analysis of the different architectures based on the results of other authors.

2.2 Full-adder

Addition is one of the most important and fundamental operations in arithmetic, due to the fact it can be used as part of other basic operations such as subtraction, multiplication and division. Embedded systems and Very Large Scale Integration (VLSI) systems, such as Digital Signal Processing (DSP) and microprocessors are some examples of integration designs in which the addition operation is a fundamental operation [17]. Other applications include low power Integrated Circuits (ICs) such as laptop computers, cellular communications, and audio and video processing [2][18].

Since addition is usually in the critical path that affects the delay and overall performance of the system, it is important to design it to be as efficient as possible [13]. The main performance requisites of any digital circuit are: low power dissipation, high speed, and low area [11]. Seeing that a single adder can not excel in every mentioned characteristic, there is always a compromise and different architectures can offer better results in some areas while "lacking" in others.

The schematic symbol of the full-adder can be observed in figure 2.1. The 1-bit full-adder has 3 inputs: A, B and Cin. The first two are the bits to be added and Cin is the *carry* bit that should be

taken into account when building multi-bit full-adders, using the cells in cascade. As outputs, it has solely two: *Sum* and *Cout*, the sum of bits *A* and *B* and the *carry* bit, respectively. Regarding single cell testing and simulation, the input *carry* bit is treated as a typical input and when it is zero, the output *carry* is merely indicative of the necessity of using an extra bit to represent the result of the sum. Given that the full-adder cells are often used in chain structured applications [2] the critical path is frequently from the *Cin* to the *Cout* [6]. In these cases, it is of significant importance for the generation of *Cout* to be as fast as possible.

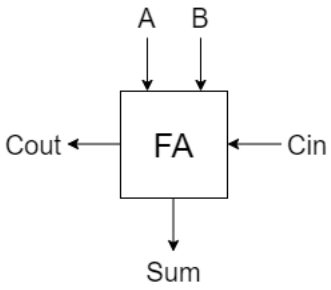


Figure 2.1: Schematic symbol of a full-adder.

The full-adder behaviour is determined by equations 2.1 and 2.2, and table 2.1; the direct implementation of the equations using XOR, AND and OR logic gates is depicted in the block diagram as shown in figure 2.2.

$$Sum = A \oplus B \oplus Cin \tag{2.1}$$

$$Cout = AB + Cin(A \oplus B) \tag{2.2}$$

Table 2.1: Truth table of a full-adder cell.

A	B	*Cin*	*S*	*Cout*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

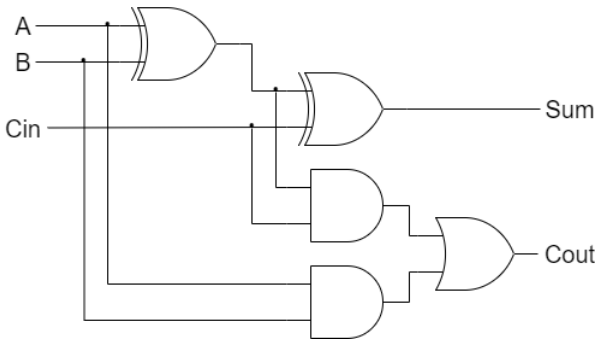


Figure 2.2: Block diagram of a full-adder.

There are multiple designs and architectures for the full-adder implementation, each with strengths and limitations compared to others. The full-adders chosen for design, analysis and comparison are representative of different static logic techniques such as conventional, pass transistors and transmission gates. Static logic is known for being robust against noise thus providing more reliable results at the cost of larger area layouts, which may not be very efficient for implementation within more complex circuits [10]. The list of full-adders styles to be explored is presented bellow along with the initialism to which they will be referred to throughout the document.

- Conventional full-adder (Conv.).
- Mirror CMOS full-adder (CMC).
- Transmission Gate full-adder (TG).
- Complementary Pass-transistor Logic full-adder (CPL).

2.3 Full-adder architectures

2.3.1 Conventional full-adder

The Conventional full-adder is the simplest implementation style of a full-adder logic. It is generally designed with 28 transistors and performs the desired operation with good driving capability and full swing in output voltage. However, the high number of transistors leads to an higher power consumption and delay, increasing input capacitances [16] [18].

An electric schematic of the full-adder is depicted in figure 2.3 according to the authors of [16]. The relative size of the transistors is not included in this schematic due to the reason that it may vary and it will be further discussed on the following chapter.

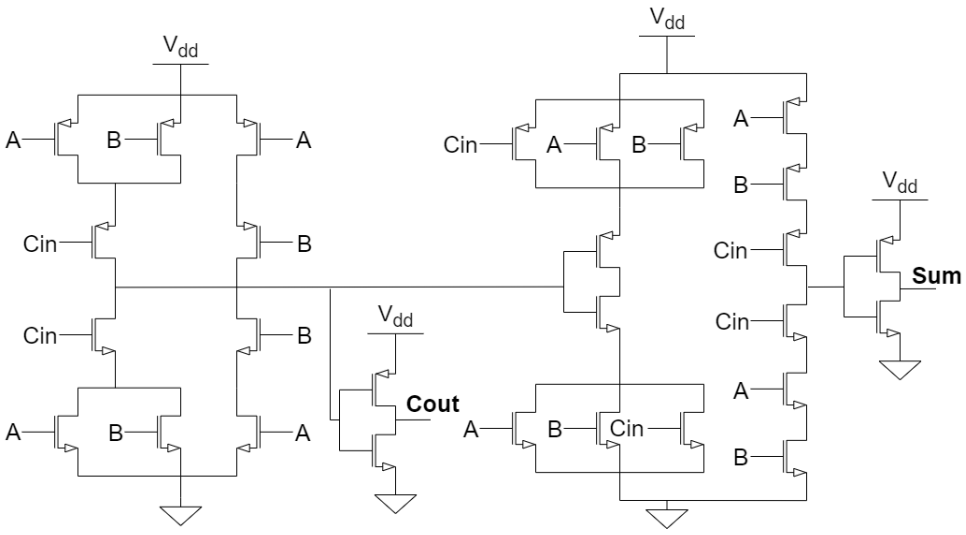


Figure 2.3: Electrical schematic of a Conventional full-adder.

This full-adder architecture was chosen because it is the simplest implementation of the full-adder operation and its performance results are undoubtedly relevant for architecture comparison.

2.3.2 Mirror CMOS full-adder

Similar to the case of the Conventional full-adder, the CMC uses static technology. However, the latter reduces the number of transistors used (from 28 to 24 transistors), providing a better speed performance at the same power [7] and making the layout more uniform. The circuit is called Mirror adder because the circuit is mirrored on itself horizontally: the NMOS and PMOS chains are completely symmetrical, as depicted in figure 2.4.

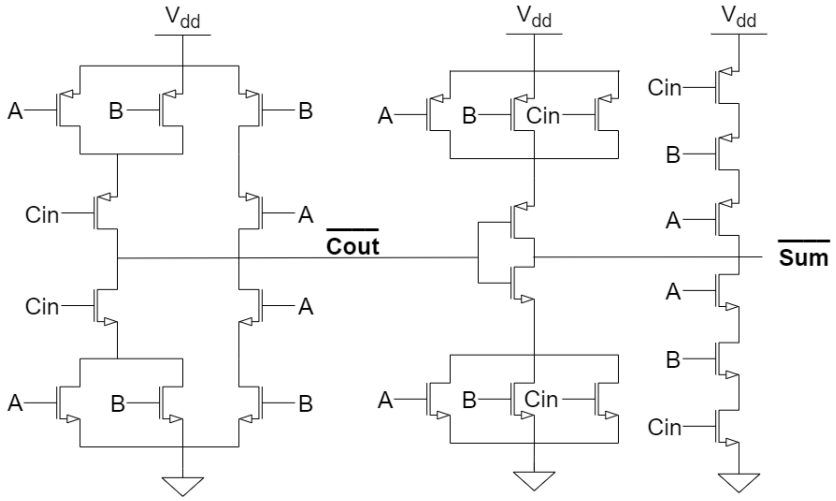


Figure 2.4: Electrical schematic of a CMC full-adder.

This circuit does not generate the previously seen *Sum* and *Cout* signals, instead it generates their complementary signals: $\overline{Sum}$ and $\overline{Cout}$. Note that the $\overline{Cout}$ is used to generate the $\overline{Sum}$ which

can be used directly in a multi-bit adder without signal inversion in the carry path. This topology is suitable for general purpose full-adders [7].

2.3.3 Transmission gate full-adder

The Transmission gate full-adder uses static logic and requires only 20 transistors as it can be observed in figure 2.5. The schematic used was based on the work of [16].

To properly understand both the transmission gate and pass transistor used for the TG and CPL full-adders, one needs to understand the concept known as *strength* of a signal. The *strength* of a signal is evaluated by how close it approximates the ideal voltage source, be it VDD or GND. A NMOS transistor passes a *strong* 0 and a *weak* 1, since the high voltage level is slightly less than VDD due to its DC characteristics [20]. Contrarily, a PMOS transistor exhibits the opposite behaviour, passing a *weak* 0 but a *strong* 1. When either an NMOS or PMOS transistor are used independently as a switch, they are called pass transistors.

The transmission gate is formed by connecting in parallel a PMOS and an NMOS and the complementary control signals are what drives them; as the NMOS only passes 0s and the PMOS only passes 1s, the output signal is always a *strong* signal. The transmission gate selectively blocks or passes a signal level from input to output and is broadly used in CMOS technology style to implement digital functions since it significantly simplifies circuit design [20].

The XOR gates of the FA are implemented using the transmission gates which use much less transistors than the Conventional full-adder. Although this circuit is less complicated than the Conventional full-adder, the power consumption is greater [16].

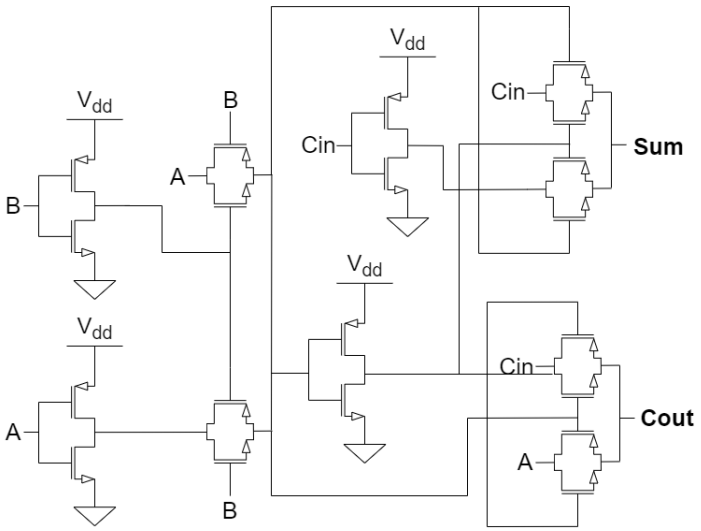


Figure 2.5: Electrical schematic of a Transmission gate full-adder.

2.3.4 Complementary pass-transistor logic full-adder

The CPL full-adder is a fully differential circuit. For this architecture, the *Sum* and *Cout* signals are generated separately in two circuits, top and bottom circuits, respectively, depicted in figure 2.6.

From all the architectures presented in this document, the CPL full-adder is the only one that has the usual full-adder signals as well as their complements, both inputs and outputs.

Due to its high transistor count, the CPL is not suited for low power despite showing a better overall performance that the Conventional FA, according to [13].

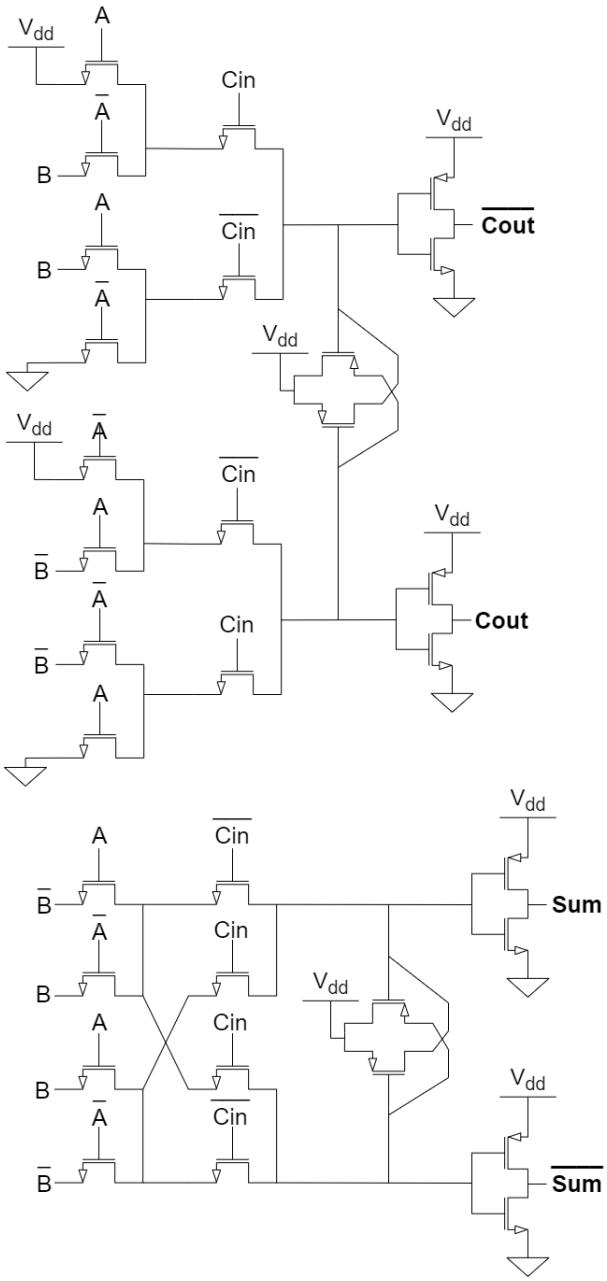


Figure 2.6: Electrical schematic of a CPL full-adder. On top is the circuit for the generation of the *Cout* signal and its complement. At the bottom is the circuit for the generation of the *Sum* signal and its complement.

2.4 Summary

As exposed before, different architectures have different characteristics and behaviours which lead to different advantages and disadvantages that should be taken into concern depending on the desired application.

For instance, the CPL full-adder has a very high transistor count and therefore high switching activity and larger area, which makes this architecture not suited for small and/or portable devices such as smartphones. On the other hand, it exhibits a lower delay than the Conventional full-adder, making it appropriate for circuits in which speed is the main constraint [9]. The Conventional full-adder architecture demonstrates to be very robust against transistor sizing and voltage scaling [3] and it is very widely used due to its simplicity, known technology and general good performance. The Mirror adder offers somewhat the same characteristics of the Conventional FA with the advantage of a lower transistor count and symmetric electric schematic and layout. Lastly, the Transmission gate full-adder presents a very low number of transistors in comparison with the other architectures. This leads to a much smaller area making it more suitable for applications where size is one of the constricting requirements. However, it presents a poor driving capability [2].

Table 2.2 summarises some of the results of other authors that used the same 90 nm technology to produce and evaluate some of the architectures chosen in terms of delay, power and power-delay product (PDP). These results will be further used for architecture comparison at the end of chapter 4.

Table 2.2: Performance results of previous works.

FA	Source	T. count	Supply Voltage (V)	Delay (ns)	Power (nW)	PDP (fJ)
Conv.	[16]	28	1.2	0.4779325	99.218	0.0461
	[10]	28	1.0	0.5919	7080	-
CMC	[9]	24	1.2	0.135	1720	0.352
TG	[16]	20	1.2	0.350115	341.49	0.00143
	[9]	20	1.2	0.141	1715	0.242
CPL	[9]	32	1.2	0.093	2183	0.203

Having considered the information gathered by other authors experiments and basic 90 nm MOS technology properties and characteristics, the next chapter focuses on the methodology used for the implementation and design of these four full-adder cells.

Chapter 3

Methodological approach

3.1 Introduction

This chapter presents the tools used for the design and simulation of the schematics and layouts for each proposed architecture. The process flow is explained with detail and the various differences among the architectures are emphasized by the electric schematics, simulation waves (both input and output waves) and layout topologies.

3.2 Technologies and tools

The schematics and layouts were designed in a Cadence Virtuoso environment. The technology used for the transistors was *gpdk090* and the simulations were conducted with the *ADE L* simulator for both the schematic and post-layout simulations. The layouts were produced on *Layout XL* of the same program and the Design Rule Check (DRC) and Layout Versus Schematic (LVS) verifications were done with *Assura* and the parasitic extraction with *Quantus*.

3.3 Architectures

The following subsections present the specifics of each architecture: electric schematic, including the transistor sizing of each circuit, test-benches, simulation definitions and waveforms, stick diagrams for the layout planning, and the final layout produced for characterisation and analysis.

3.3.1 Conventional full-adder

The first step for the schematic set up would be sizing the transistors. The size of the NMOS transistors was defined as minimum (length of 100 nm and width of 120 nm) and the PMOS transistors were designed with double area, same length but width of 240 nm. This was based on the minimum-sized symmetrical inverter due to the fact that the circuit is relatively balanced: pull up network and pull down network are almost completely symmetrical.

In figure 3.1 is depicted the electric schematic for the Conventional full-adder, based on [16].

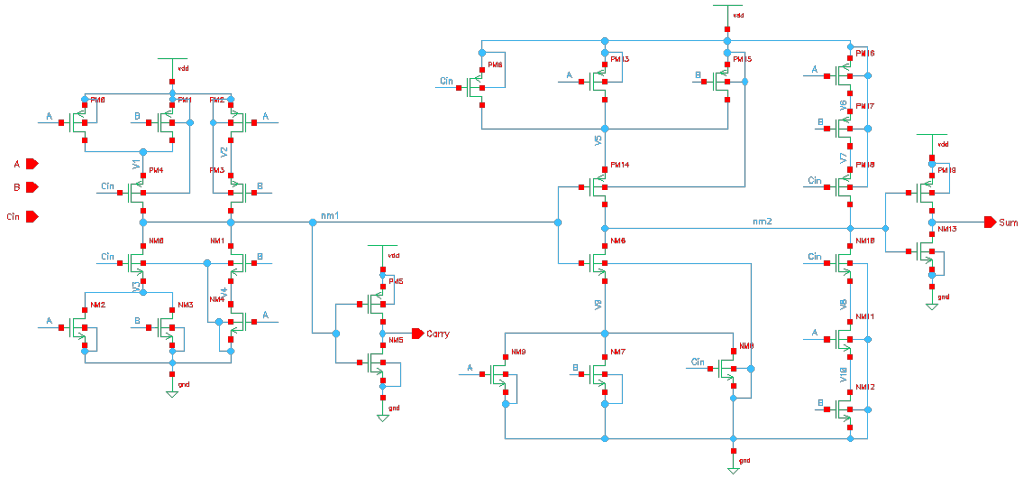


Figure 3.1: Electrical schematic of the Conventional full-adder on Virtuoso.

After the schematic has been drawn, it should be tested and its results validated. For such, figure 3.2 presents the proposed simulation setup. The inputs were defined as square waves using the *vpulse* cell from the *analogLib* library with a rise time of 2 ns. The full-adder cell's inputs pass through buffers, two cascaded symmetrical minimum-size inverters (PMOS as 240/100 nm and NMOS as 120/100 nm) to produce realistic input signal waveforms, proposed in [6] [2]. The outputs are also loaded with buffers, 4 minimum-size parallel symmetrical inverters, to give proper loading conditions [13] [7]. The supply voltage (VDD) using the *vdc* cell from *analogLib* as well and *gnd* from *basic* library, all available on the *gpdk090* technology. VDD was defined as the nominal supply voltage for the technology, 1.2 V, according to the 90 nm technology reference manual [4].

The simulation was set as transient with stop time of 1 µs and its accuracy defaults set as conservative. Table 3.1 presents the parameters for each input such as frequency (*fA, fB* and *fCin* will be used to notate the frequency of *A*, *B* and *Cin*, respectively), period, delay time, rise time, fall time, and pulse width. These parameters are maintained the same for all analysed technologies.

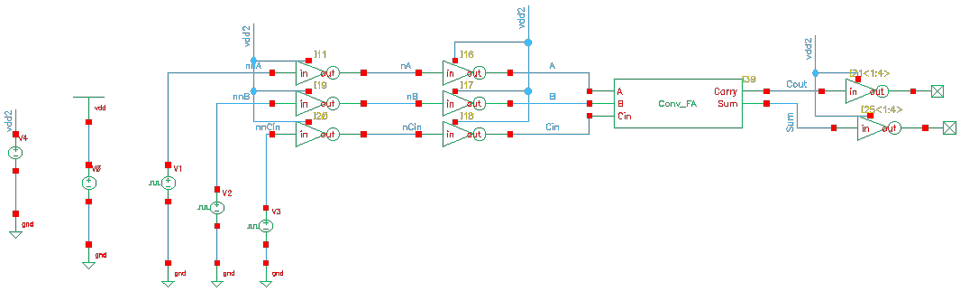


Figure 3.2: Electric schematic of the test-bench to simulate the Conventional full-adder schematic on Virtuoso.

Table 3.1: Input values for the circuit simulation on *ADE L*.

Inputs	A	B	Cin
frequency (MHz)	5	5	11
period (s)	1/fA	1/fB	1/Cin
delay time (s)	0	0.25/fB	0
rise time (s)	0.01/fA	0.01/fB	0.01/fCin
fall time (s)	0.01/fA	0.01/fB	0.01/fCin
pulse width (s)	0.5/fA	0.5/fB	0.5/fCin

As for the outputs, the simulation waves are shown in figure 3.3 with the inputs for functional verification of the full-adder. Several simulations were run for the verification considering all possible input transitions.

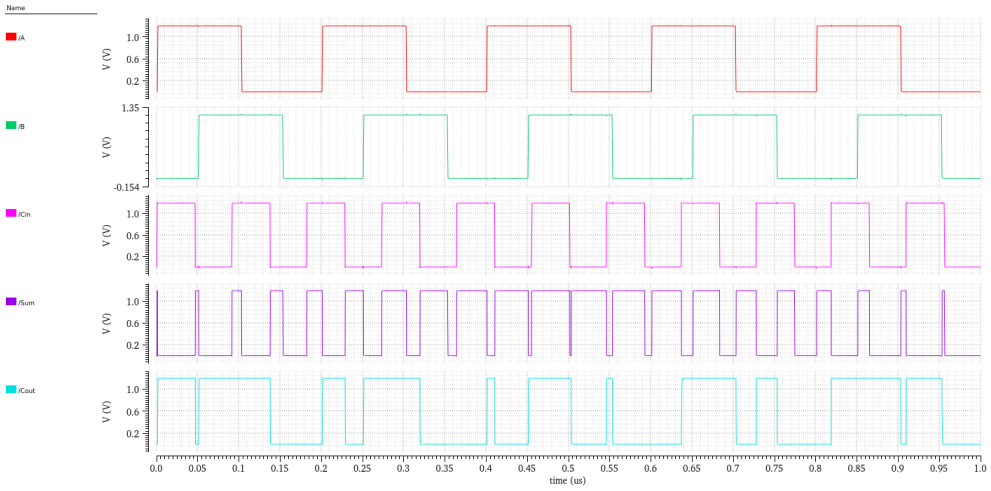


Figure 3.3: Input and output waves of the Conventional full-adder schematic simulation.

Subsequently, after validating the schematic, a stick diagram was drafted taking into consideration the names of the transistors defined in the schematic as well as the nodes for easier implementation on the layout. Both the stick diagram and the layout were designed using 2 metals only and with the goal of minimizing the area by pairing the transistors, as shown in figure 3.4. The two metals used, *metal1* and *metal2*, are illustrated in blue and red, respectively, identical to the layout colours for easier perception. The *poly* path is in green for the same purpose. The several inputs and outputs (here the *Cout* output is labeled as *Carry*), as well as intermediate nodes are properly identified along the connections. The circuit was divided in two parts (part one being the left side of the circuit until transistors PM5 and NM5, and part two the right side of the circuit, figure 3.1), separated by 0.24 nm, the minimum distance allowed by the DRC rules of the technology. The final version of the layout is depicted in figure 3.5 and it has an area of 28.147 μm^2.

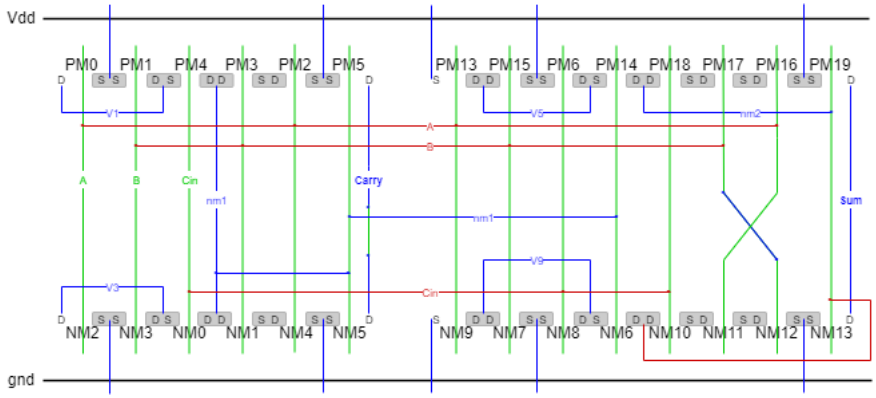


Figure 3.4: Stick diagram for the Conventional full-adder layout.

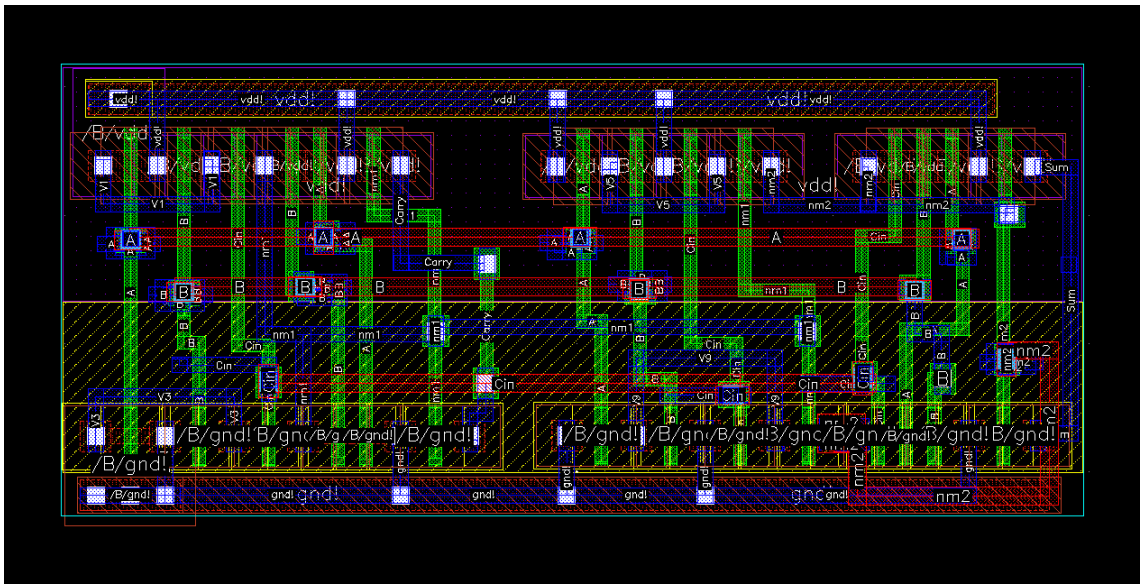


Figure 3.5: Final layout for the Conventional full-adder.

After the layout had been produced and before running the characterisation on liberate (described thoroughly in the next chapter), the simulation was again run with the post-layout circuit and parasitics extracted from the layout. To ensure that the layout was generating the same results as the previously tested schematic, a test-bench was built for confirmation in accordance with figure 3.6; the resulting waveforms of this simulation are portrayed in figure 3.7. This simulation was conducted for manual verification of the logic function and not for observation of the differences in the output waves before and after the layout and parasitic extraction.

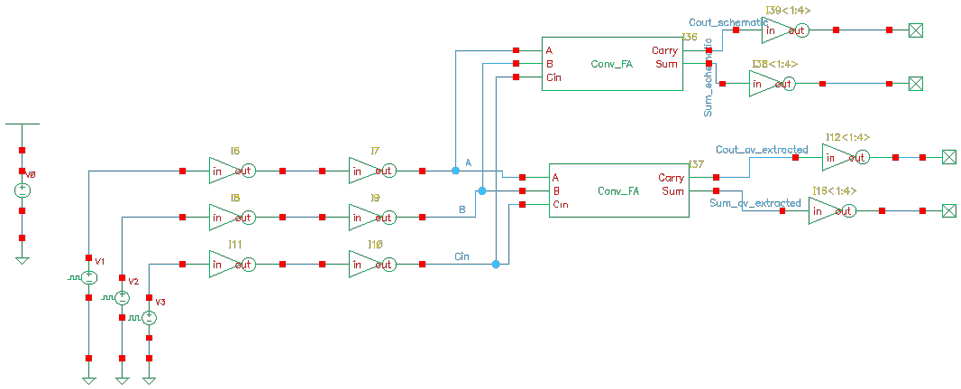


Figure 3.6: Simulation waveforms of the Conventional full-adder post-layout.

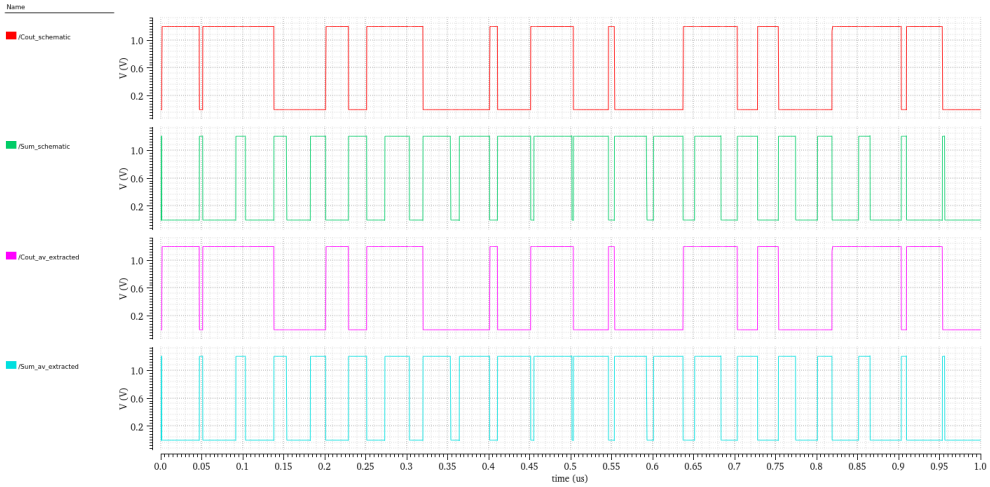


Figure 3.7: Outputs of the simulation of the Conventional full-adder schematics and post-layout for comparison.

3.3.2 Mirror CMOS full-adder

Considering the Mirror CMOS adder architecture, the transistors were sized as suggested in [20], "with transistor sizes optimized to favor the critical path". All transistors have a minimum length of 100 nm allowed by the technology and only the widths differ, setting the transistors with different areas. Observing figure 3.8, in which is depicted the electric schematic of the CMC full-adder, the three PMOS transistors on the left (PM0, PM1 and PM2) have a width of 960 nm (8 times the minimum width - 120 nm) and the NMOS transistors below (NM0, NM1 and NM2) have a width of 480 nm (4 times the minimum width). This part of the circuit follows the minimum-size symmetrical inverter area ratio of PMOS/NMOS $= 2$. All other transistors have minimum-sized length and width.

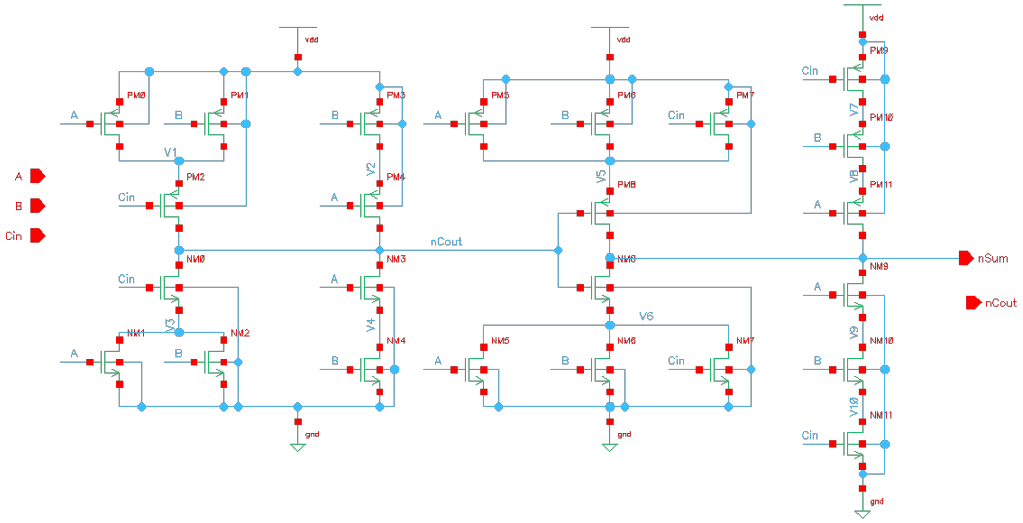


Figure 3.8: Electric schematic for the Mirror full-adder on Virtuoso.

As for the simulation of the full-adder cell, the setup used is similar to the one of Conventional FA: the input parameters are the ones on table 3.1. However, the output waveforms are inverted since the outputs produced are $\overline{S}$ and $\overline{Cout}$, here labeled as *nSum* and *nCout*, respectively. These waveforms are depicted in figure 3.9.

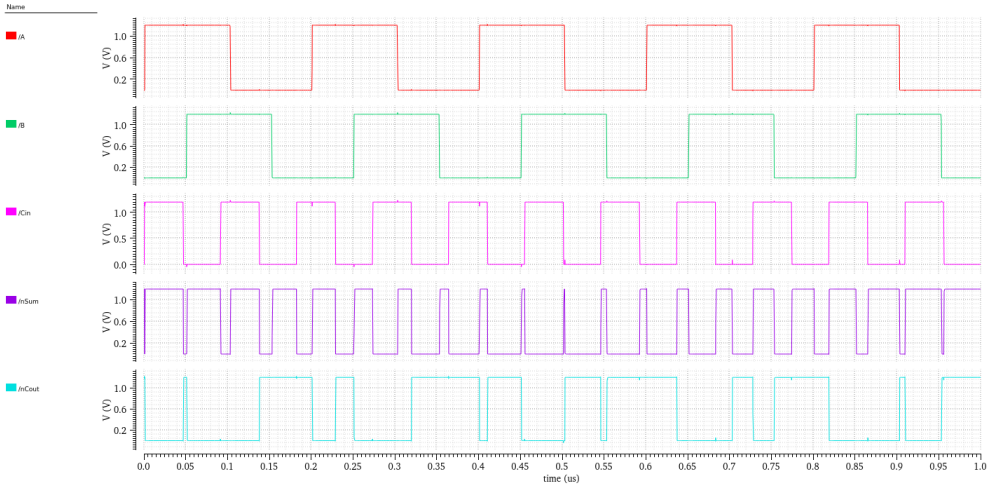


Figure 3.9: Input and output waveforms of the Mirror full-adder schematic simulation.

For the functional verification of the full-adder cell, a different test-bench was built to compare the output waveforms of the CMC and the previously validated of the Conventional FA. As the output signal of the CMC full-adder are inverted, two additional inverters were added at the outputs of the Conv. FA to generate the complement for easier comparison. This test-bench and waveforms are depicted in figures 3.10 and 3.11, respectively.

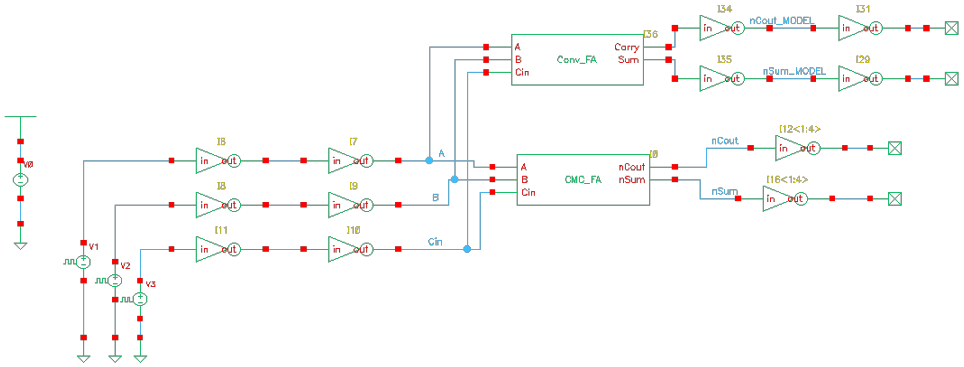


Figure 3.10: Electric schematic of test-bench for comparison of the output waves of the Mirror full-adder and the Conventional full-adder.

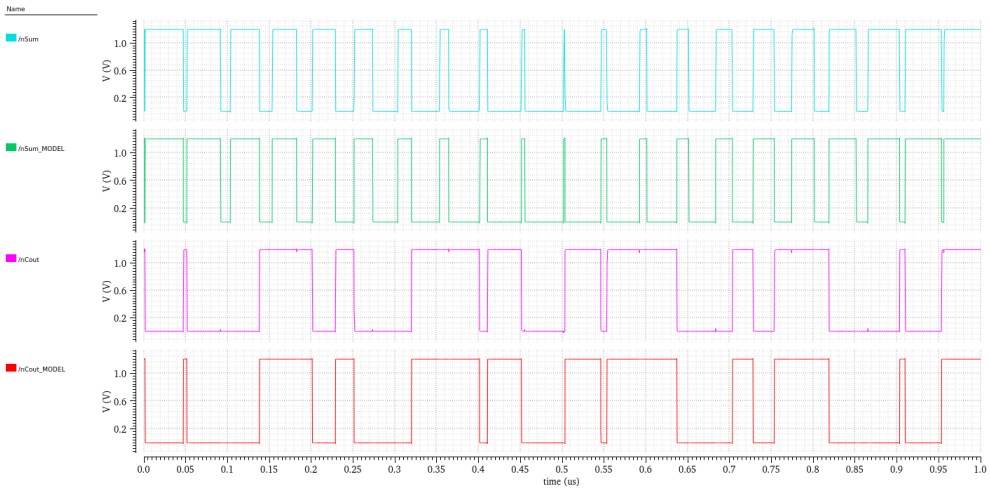


Figure 3.11: Output waves of the Mirror and the Conventional full-adder for verification.

With the electric schematic validated, and following the same procedure of the Conventional full-adder, a stick diagram was drafted. Similarly to the Conventional FA, the circuit was divided in two parts (left until transistors PM3 and NM4, and right), separated by 0.24 nm, i.e., the minimum distance allowed by the DRC rules of the technology. The stick diagram in figure 3.12 served as guide for the final layout of this architecture and it was adjusted along with the design of the layout, depicted in figure 3.13. This layout was completed with an area of 29.238 μm^2. This area was larger that the previous architecture but understandably so as a consequence of the higher transistor size. The differences in those areas will be further discussed on the next chapter.

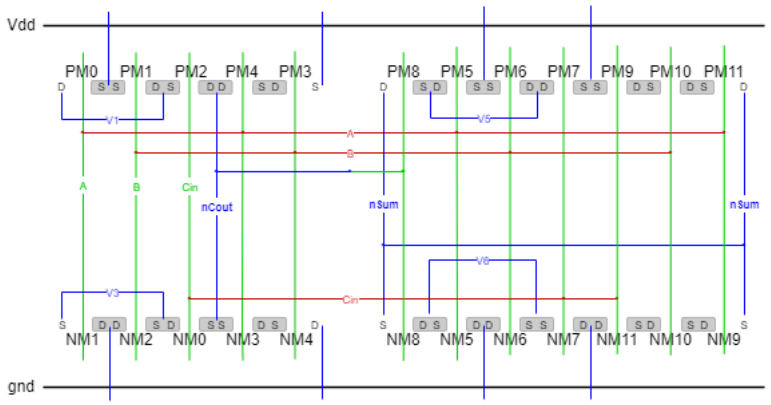


Figure 3.12: Stick diagram for the Mirror full-adder layout.

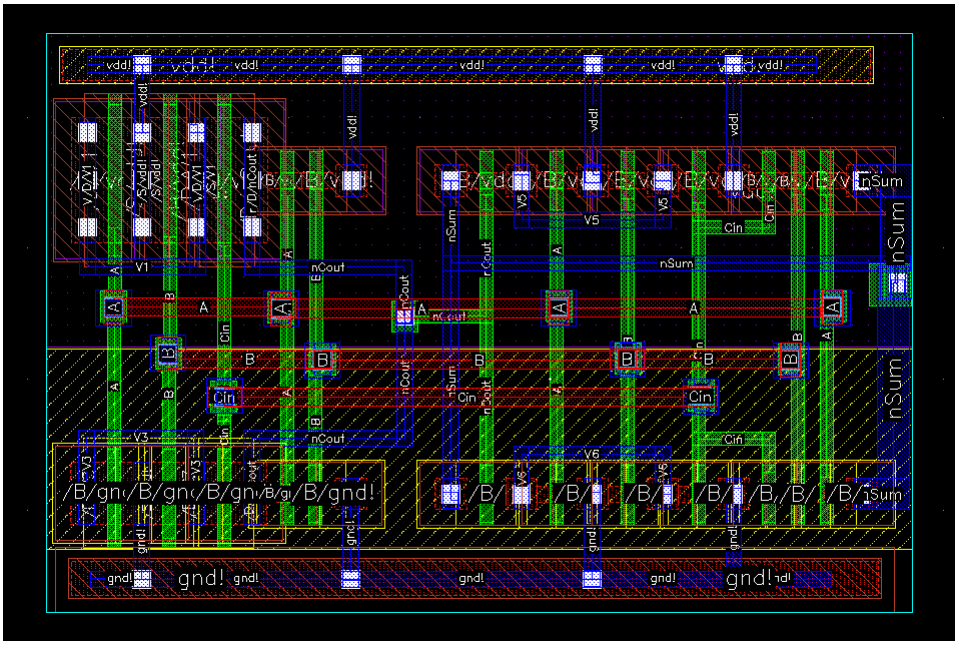


Figure 3.13: Final layout for the Mirror full-adder.

3.3.3 Transmission gate full-adder

Following the same principle used in the transistor sizing of the Conventional full-adder, the transistors of this cell have the same relative size: the NMOS transistors are minimum sized and the PMOS transistors have the minimum length and twice the minimum width. This balance is even more significant as a consequence of the pass-transistor logic implemented (as previously described in chapter 2.3.3).

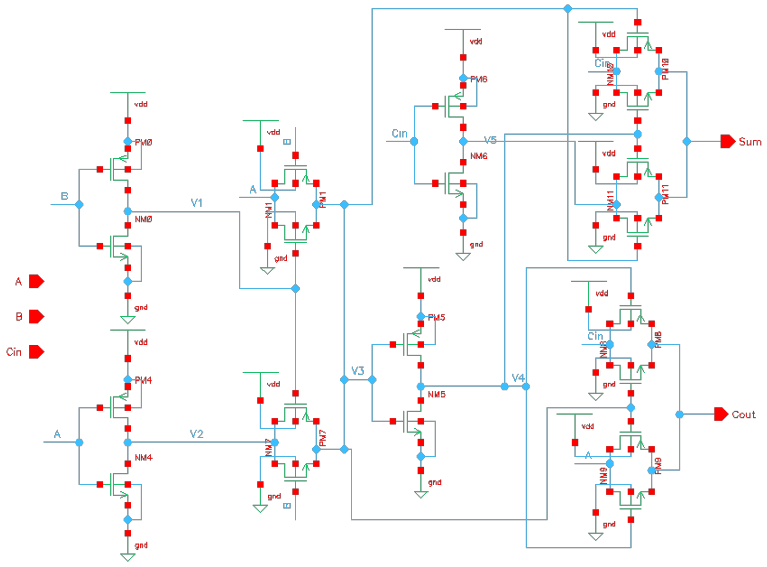


Figure 3.14: Electric schematic for the Transmission gate full-adder on Virtuoso.

The electric schematic of the Transmission gate FA is depicted in figure 3.14 and the simulation setup and test-bench are identical to the previous architectures. The input waveforms suffer some interference and reveal some "spikes" due to the transmission gates, in which the transistors do not switch simultaneously. This causes the current to move both ways, which affects the output signals as well is this is depicted in figure 3.15.

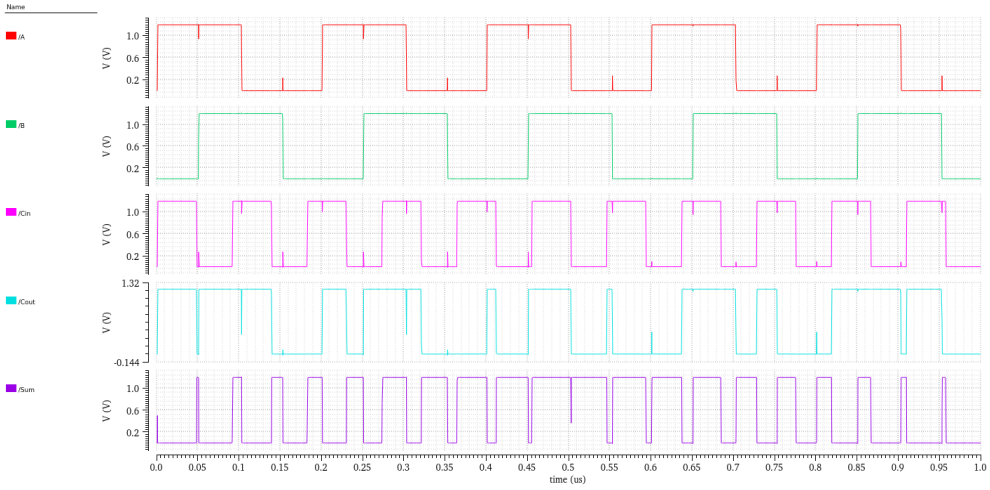


Figure 3.15: Input and output waves of the Transmission gate full-adder schematic simulation.

The functional validation of the circuit was achieved by comparison of the outputs of both the Transmission gate FA and Conventional FA using the test-bench illustrated in 3.16 and the results are displayed in figure 3.17. As one can observe, the "spikes" present on the inputs do not interfere significantly with the outputs and the logical function is nevertheless correctly performed.

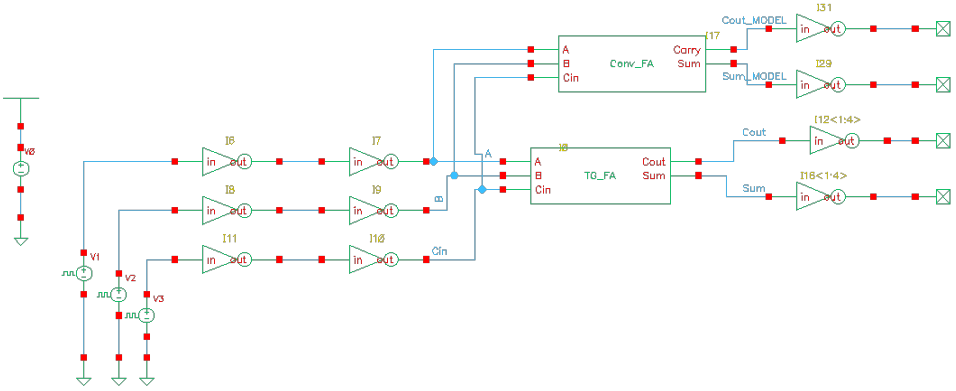


Figure 3.16: Electric schematic of test-bench for comparison of the output waves of the Transmission gate full-adder and the Conventional full-adder.

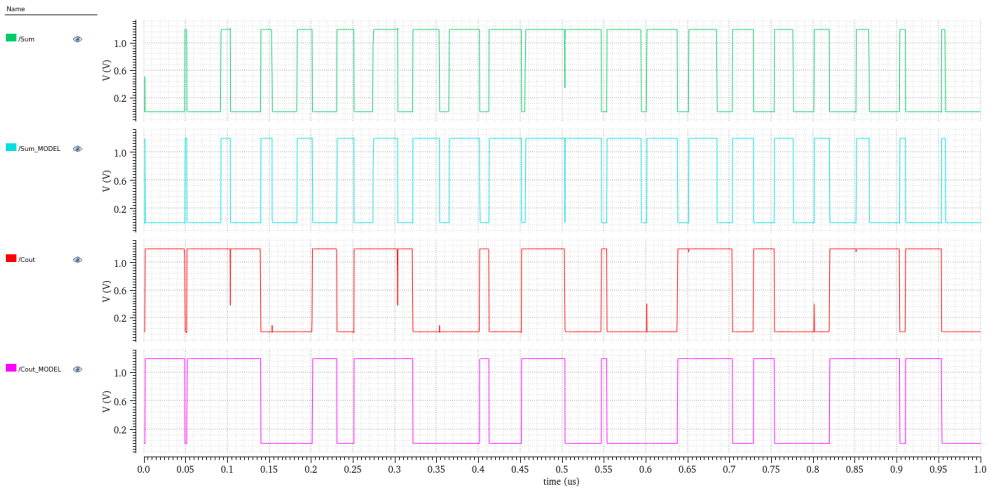


Figure 3.17: Output waves of the Transmission and the Conventional full-adder for verification.

When drafting the stick diagram for this particular cell, there were not many drawbacks. The circuit is very small since it has the fewer number of transistors of the analysed adders. It was not necessary to divide the cell and the complete set of NMOS transistors were paired thus reducing the layout area. The stick diagram and the final layout for this architecture are depicted in figures 3.18 and 3.19, respectively. This layout has an area of 25.059 μm^2 and it is the smallest of the four characterised architectures.

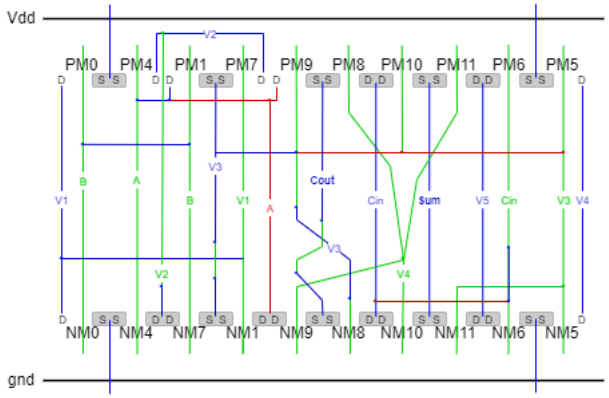


Figure 3.18: Stick diagram for the Transmission gate full-adder layout.

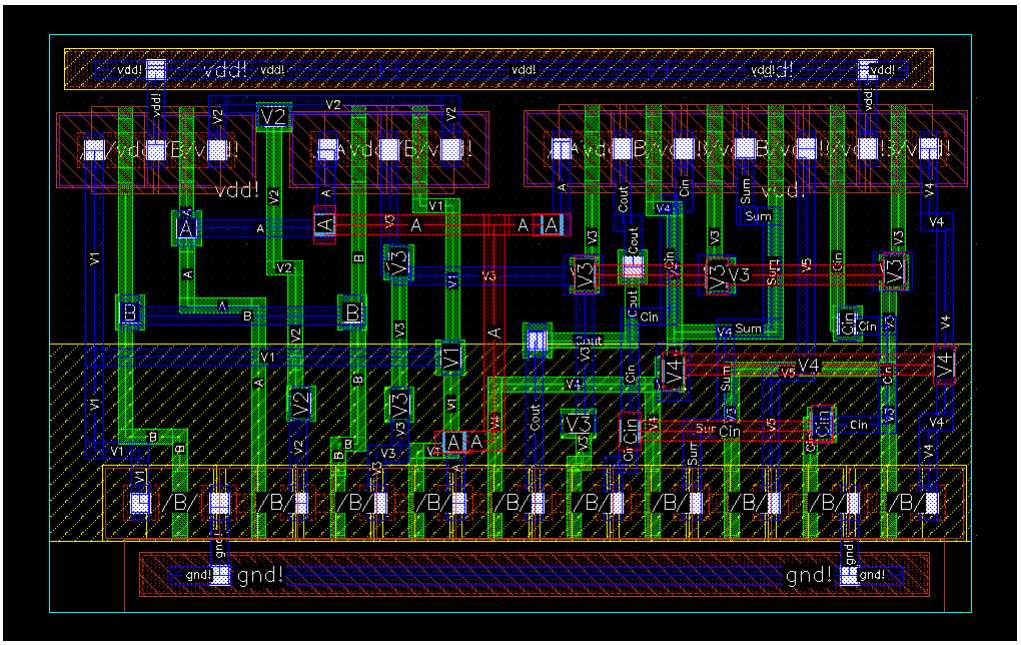


Figure 3.19: Final layout for the Transmission gate full-adder.

3.3.4 Complementary pass-transistor logic full-adder

Lastly, the CPL full-adder electric schematic was produced. This schematic is somewhat different from the previous architectures, considering that it is divided into two subcircuits, one for *Cout* and its complement, and another for *Sum* and its complement. It is the only one out of the four architectures analysed that has all inputs (*A*, *B*, *Cin*) and outputs (*Sum*, *Cout*) and their respective complements (*nA*, *nB*, *nCin*, *nSum*, *nCout*), as depicted in figure 3.20. All transistors have minimum-size parameters except the inverters PMOS transistors (PM0, PM1, PM3, PM4), to balance the inverters, as it was carried out previously.

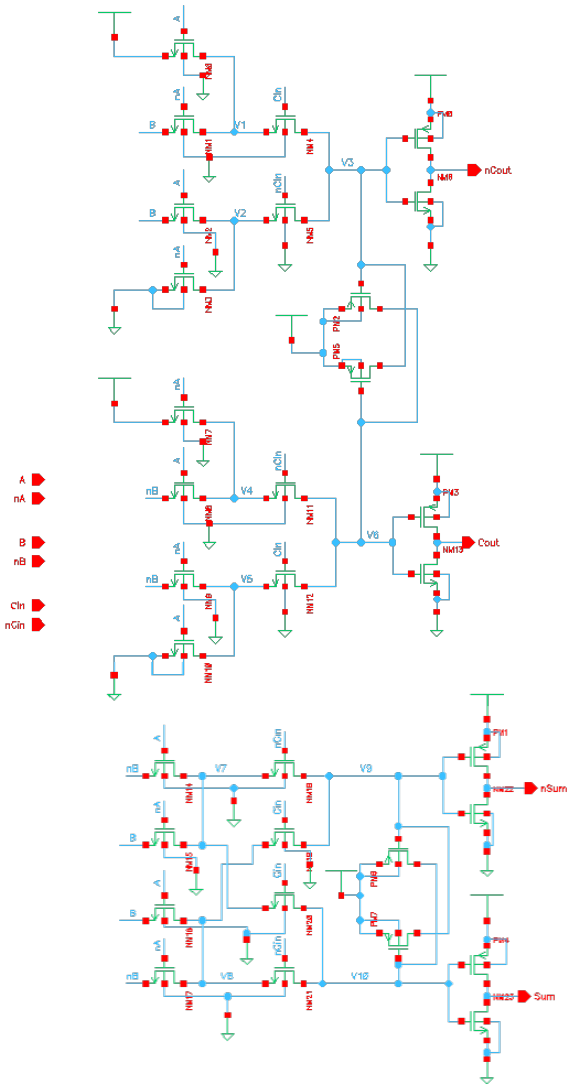


Figure 3.20: Electric schematic for the Complementary pass-transistor full-adder on Virtuoso.

The test-bench to validate the full-adder was designed similarly to the previous ones and two additional inverters were added before the inputs to produce the complementary inputs, *nA*, *nB* and *nCin*, as shown in figure 3.21.

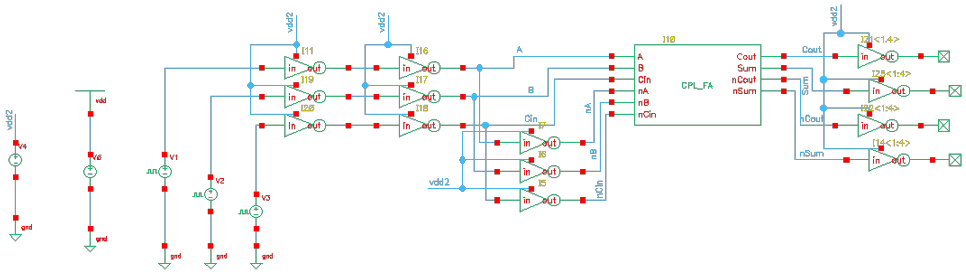


Figure 3.21: Electric schematic of test-bench for the Complementary pass-transistor full-adder simulation.

As it was previously asserted regarding the Transmission gate FA, the pass-transistor logic on the CPL full-adder causes the "spikes" most visible on the signal *B* waveform and its complementary, depicted in figure 3.22. To simplify the analysis of the input and output waveforms, the input signals and their complements were grouped in figure 3.23.

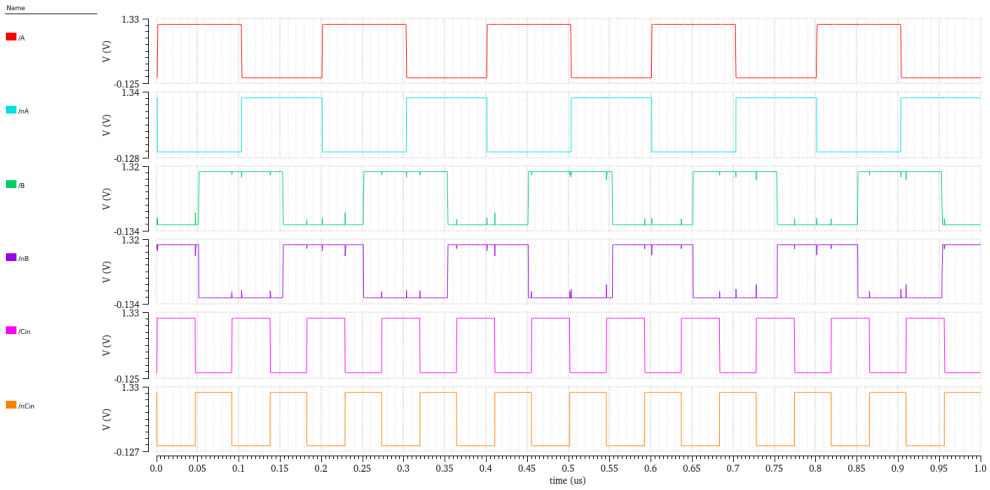


Figure 3.22: Input waves of the Complementary pass-transistor full-adder schematic simulation.

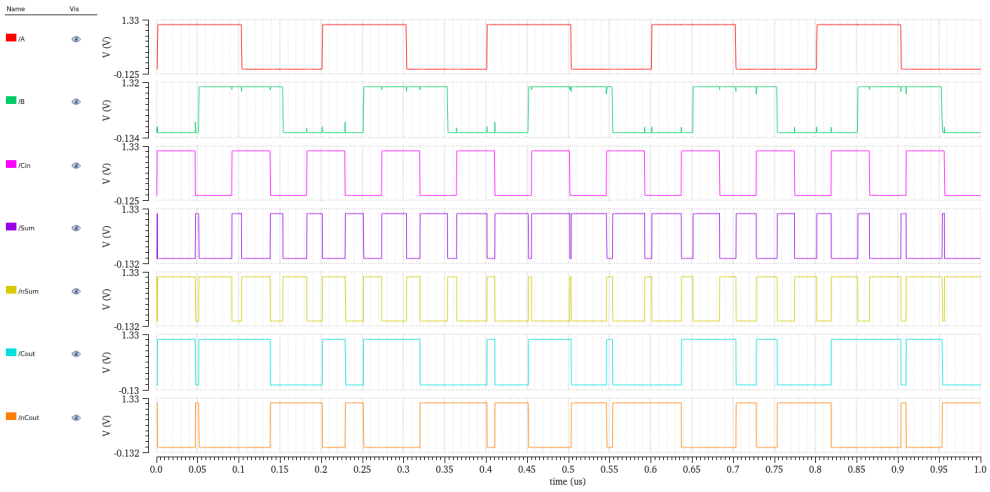


Figure 3.23: Input and output waves of the Complementary pass-transistor full-adder schematic simulation.

Once more, the functional verification was conducted through comparison with the Conventional FA cell. The test-bench and output waveforms are illustrated in figures 3.24 and 3.25, respectively.

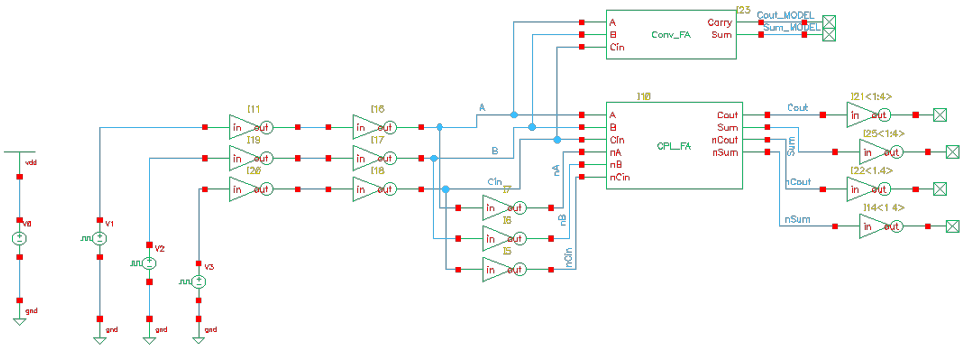


Figure 3.24: Electric schematic of test-bench for comparison of the output waves of the Complementary pass-transistor full-adder and the Conventional full-adder.

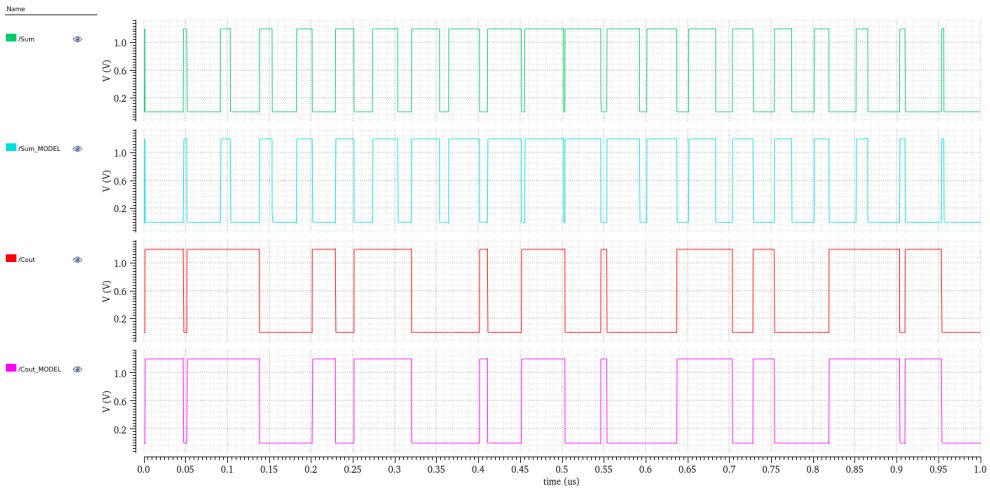


Figure 3.25: Output waves of the Complementary pass-transistor and the Conventional full-adder for verification.

The most challenging layout was, without a doubt, the CPL full-adder layout due to the asymmetry of the circuit as well as large number of NMOS transistors (24) in comparison with the 8 PMOS. On the contrary of the previously presented stick diagrams, this one, depicted in figure 3.26, displays several rows of NMOS transistors in order to reduce the layout area. The final layout is depicted in figure 3.27 and it has an area of 62.064 μm^2, the largest of all four layouts.

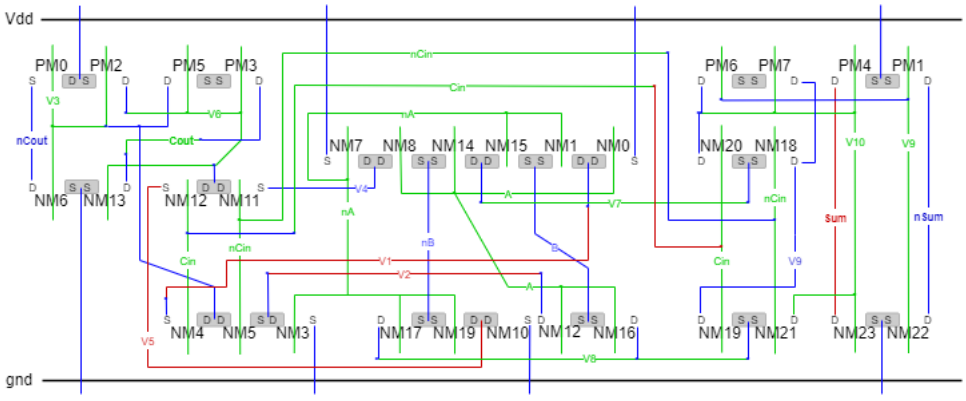


Figure 3.26: Stick diagram for the Complementary pass-transistor full-adder layout.

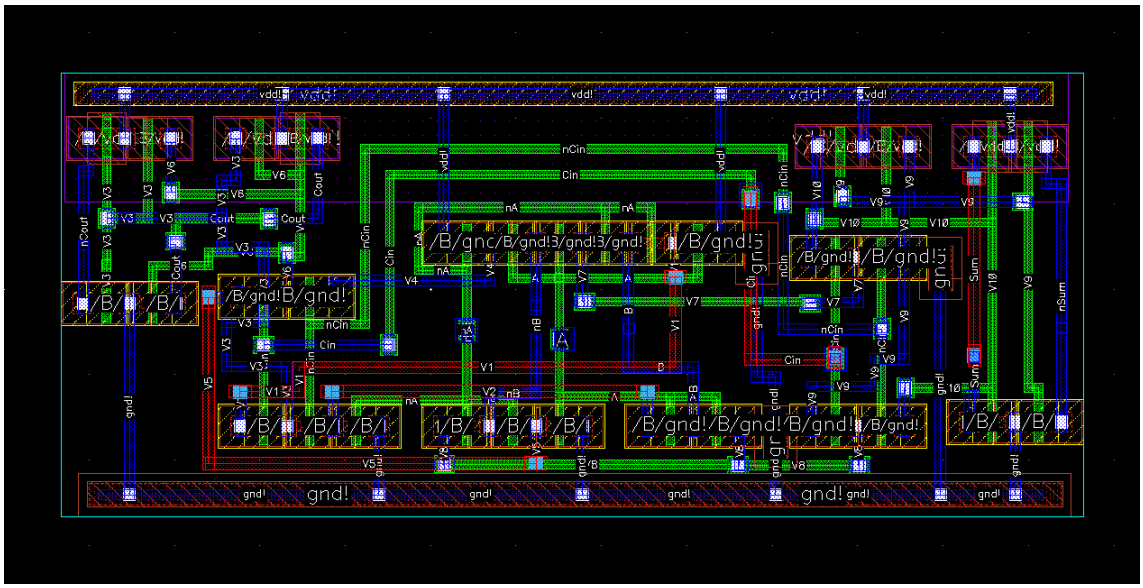


Figure 3.27: Final layout for the Complementary pass-transistor full-adder.

3.4 Characterisation

Once the layout is completed for each architecture, the next step is the characterisation of each cell. As mentioned before, it was decided on earlier stages of the work that the most interesting comparison parameters for the different cell layouts would be area, delay and power.

The area is simply measured on the layout environment, *Layout XL*. As for the timing and power measurements, they were obtained using the *Liberate* program [5]. For the program to run and produce the desired results it needed the post-layout netlists of each architecture layout produced, the models of the transistors used, and three Liberate specific files: *char.tcl*, *template.tcl* and *settings.tcl*. These files define the commands and settings for the program to run and, as outputs, it produces libraries with the results and html datasheets for each architecture with the defined settings (see appendix A for an example of a datasheet). Both the files, steps and results of the characterisation are further analysed over the next chapter.

3.5 Summary

Since all architectures use the same static logic, the methodological approach is very similar for each of the them. The tools used are the same and, with the exception of the CPL, all layouts were designed with the same PMOS/NMOS balancing strategy. This facilitated the process because, for instance, for the logic function testing and circuit validation, it was possible to utilize the same test-bench templates and input signals.

Chapter 4

Results and discussion

This chapter is subdivided into three main sections: the characterisation setup, including an insight on the files as well as the chosen intervals for the PVT analysis, the final results, and it ends with the discussion of those results as well as some conclusions and final remarks on the full-adders' performance. The result exposition is analysed by process initially, comparing the leakage power, propagation delay, and power-delay product within each process, and ending with a power-delay product comparison of the full-adders in the three processes: typical-typical (*tt*), fast-fast (*ff*), and slow-slow (*ss*). Only the processes with the same technology specifications for both PMOS and NMOS transistors were considered.

4.1 Post-layout characterisation

The first step of characterisation of the layout, after it has been completed, is performed through the comparison of the areas, considering the number of transistors in each architecture (see table 4.1). The CPL full-adder, having the larger number of transistors and a ratio of NMOS/PMOS transistors of $8/24$ has the larger area as well, as to be expected. The CMC full-adder has a slightly bigger area than the Conventional, despite the inferior number of transistors. This particularity is due to the use of larger transistor sizes in the layout as previously introduced in chapter 3. As expected, the Transmission gate FA, having the smallest number of transistors, also presents the smallest area.

Table 4.1: Area and transistor count of the analysed full-adders.

Full-adder	Length (µm)	Width (µm)	Area (µm^2)	Transistor count
Conv.	7.985	3.525	28.147	28
CMC	6.615	4.420	29.238	24
TG	6.320	3.965	25.059	20
CPL	11.97	5.185	62.064	32

4.1.1 Characterisation setup

The setup for characterization was conducted based on the goal of comparing the different full-adder layout designs and architectures in terms of the number of transistors, area, delay, power, and power-delay product in relation to the supply voltage, temperature, and process variation.

The range for the supply voltage variation was defined as $\pm 15\%$ of the nominal power supply value, which is 1.2V for the 90 nm technology [19]. The standard interval is $\pm 10\%$ of the power supply value [3]; however, it was decided to take a more conservative approach, which is why the chosen interval for the supply voltage variation is from 1.0 V to 1.4 V, with 0.1 V steps. Regarding the temperature, the chosen interval, also standard, is from $-40\,^{\circ}\text{C}$ to $125\,^{\circ}\text{C}$ [21], with a $20\,^{\circ}\text{C}$ step and an added value for "room temperature" of $26.85\,^{\circ}\text{C}$ (300 K) [8]. This is also the temperature used for the power supply sweep analysis. The processes chosen for comparison are *tt*, *ss*, and *ff*, which represent typical, slow, and fast technology parameters, respectively, for both PMOS and NMOS transistors.

Two important parameters are also the input and load capacitances. For both delay and power models defined in the *template.tcl* file for the characterization on Liberate, the capacitance interval used was $[0.002, 1.1559]$ pF. This decision was based on the Monte Carlo experiments from the work [12], which used as load capacitance the interval $[C_{inv,min}, 30 \times C_{inv,min}]$, in which $C_{inv,min}$ is the input capacitance of a minimum-sized inverter. Instead of using the $C_{inv,min}$ value, for more approximated results, the minimum value for input capacitance was chosen as the min value for it on a trial run at 1.2 V, 26.85 ºC, process *tt*.

The Liberate software produced HTML files with information regarding leakage power, delay, input pin capacitance, and switching energy (see appendix A). Since the main interest of analysis is the delay and power information, the result discussion will focus only on leakage power and propagation delay result values.

As it was briefly mentioned on chapter 3, the Liberate needs, in addition to the information regarding the cells, 3 files for setup: *template.tcl*, *settings.tcl*, and *char.tcl*. These files are available on appendix B.

The *template.tcl* file contains the characterization conditions and sets up the templates for the desired measurements. The ordered steps are as follows:

1. Definition of supply nets (VDD and GND).
2. Definition of transition and delay thresholds:
    - The points for rise and fall of signal transitions were set as 20 % and 80 % of the signal range.
    - Measurement points for delay measurements as 50 % of the signal range (for both rising and falling edges).
3. Setting up the cell list for characterization: input and output names.

4. Definition of the templates for the look-up tables and association with each cell. The comparison of delay and power is of the most importance for this study, and those were the templates used, with the load capacitance defined previously in this chapter.

    - The power template produces information on leakage power and also switching energy.

The *settings.tcl* file contains the variable settings for Liberate. Most of these settings and commands are default settings for setting SPECTRE, Spectre Kernel Interface (SKI), input waveforms, arc generation, and variable constraints.

Lastly, the *char.tcl* file is the "run script" and performs the following steps:

1. Set up of the source and run directories.
2. Set up of the PVT parameters for the characterization as well as other configurable variables such as the name of the library in which the results will be stored.
3. Reading of the setup files such as the settings and template files and also a file with the name information of each cell to be characterized.
4. Definition of all the active device models.

    - In this case, only the NMOS and PMOS models were used: $gpdk090\_nmos1v$ and $gpdk090\_pmos1v$.
    - Both NMOS and PMOS models' information is detailed in the models folder of the *gpdk090* technology library used for the MOS selection for the schematic.

5. Reading of the previously defined MOS models and cells' SPICE netlists.
6. Definition and run of the characterization command.
7. Writing the Liberate database and Liberty file [1].

Once the characterization is run, Liberate produces a *char.log* file with the report information on the finished cells and any errors that may occur during the characterization process and the files with the results for each cell.

4.2 Post-layout result comparison

With the datasheets obtained after running the characterization program, the data was extracted and organized in tables and graphs to easily analyze the variations and compare the different architectures.

For each parameter, it is possible to set minimum, average, and maximum values; for direct value comparison, such as leakage power, only average and maximum values were used. This

[1] The Liberty file is the primary output file generated by Liberate, and its file format is the industry standard for specifying timing information.

was because, for most applications, the most interesting values are the maximum values obtained, representing the worst-case scenario, which the system designer should consider due to system constraints and dimensioning. In addition, the choice also aims to perform a comparison based on the average case for a more realistic situation assessment since, in some cases, the maximum value represents a very pessimist scenario with very low occurrences.

The values for leakage power are a direct observation from each architecture and PVT condition datasheet. However, for the delay values, the results on the tables and graphs were calculated as a mean of maximum values, as presented in the following equations (4.1 to 4.3).

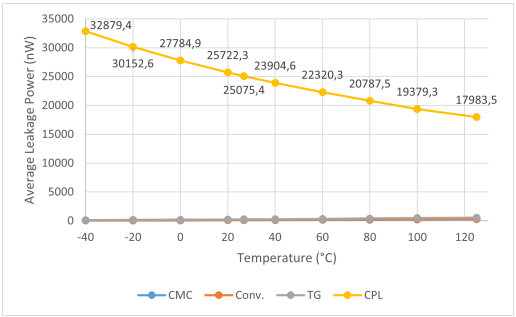
$$d_{A,avg} = \frac{d_{A->Sum,max} + d_{A->Cout,avg}}{2} \tag{4.1}$$

$$d_{A->Sum,max} = \frac{d_{Sum,rising} + d_{Sum,falling}}{2} \tag{4.2}$$

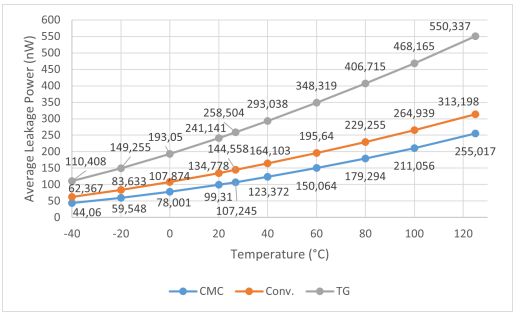
$$d_{avg} = \frac{d_{A,avg} + d_{B,avg} + d_{Cin,avg}}{3} \tag{4.3}$$

4.2.1 Full-adders comparison for process *tt*

From direct observation of figures 4.1 and 4.2, average and maximum leakage power, respectively, the CPL full-adder presents a very different behaviour to temperature variations from the other architectures. As the temperature increases, the leakage power decreases, making it more power efficient on higher temperatures. Nevertheless, even the lowest average leakage power value obtained for a temperature of 125 ºC is higher than any of the other architectures: it is 3267.7 % of the leakage power value of the Transmission gate full-adder, at the same temperature. However, when the temperature is maintained at 26.85 ºC but the supply voltage increases (see figures 4.3 and 4.4), the CPL behaves in the same way as other architectures, and yet again with much higher values. The CMC full-adder shows the lowest leakage power values in all temperatures and supply voltages tested increasing with both the temperature and supply voltage. The Conventional adder presents a very similar behaviour with slightly higher values.

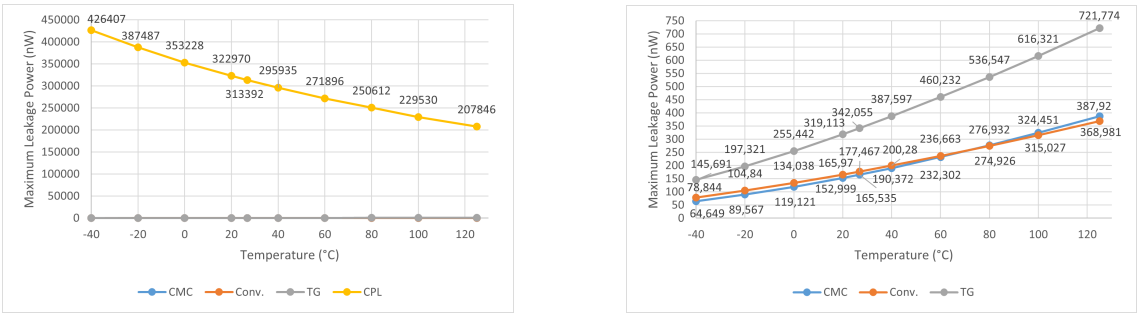


(a)



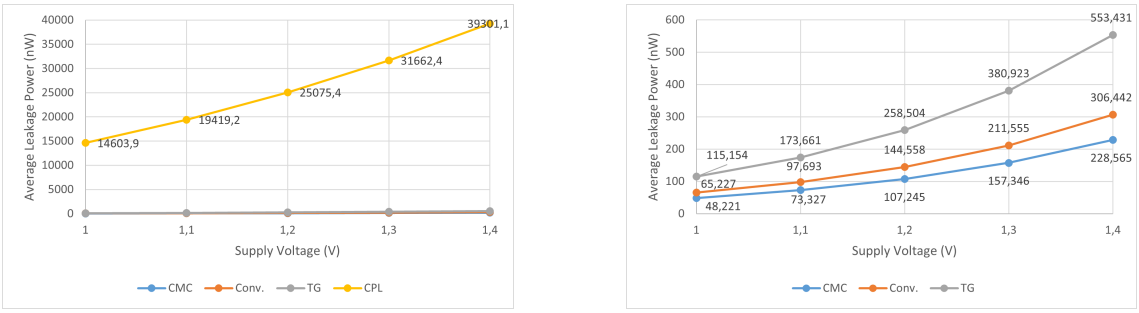
(b)

Figure 4.1: Comparison of average leakage power variation with temperature for process *tt*, at 1.2 V: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



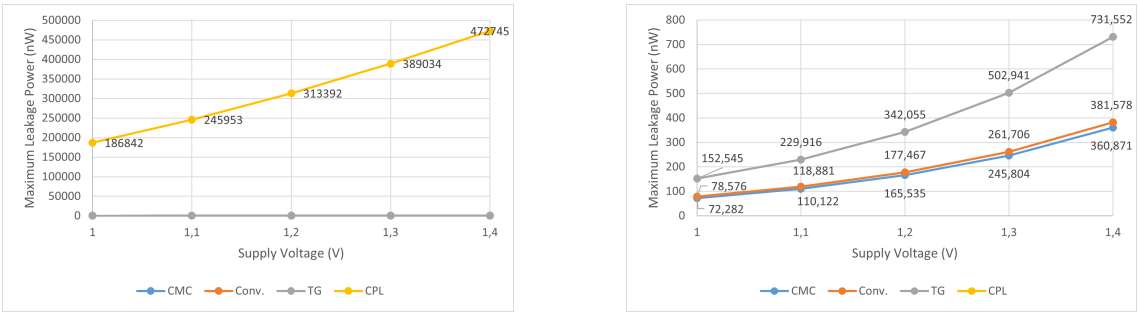
(a) (b)

Figure 4.2: Comparison of maximum leakage power variation with temperature for process *tt*, at 1.2 V: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



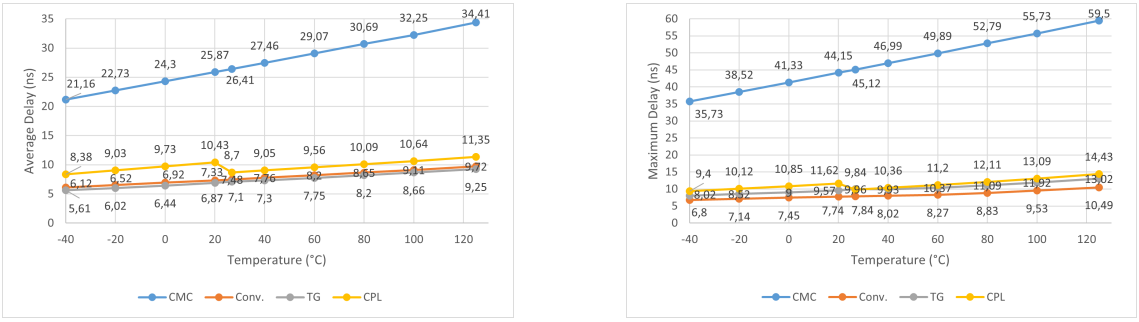
(a) (b)

Figure 4.3: Comparison of average leakage power variation with supply voltage for process *tt*, at room temperature: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



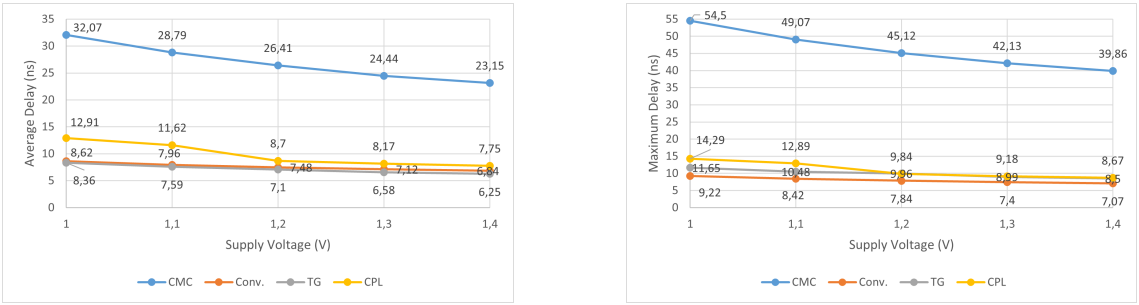
(a) (b)

Figure 4.4: Comparison of maximum leakage power variation with supply voltage for process *tt*, at room temperature: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



(a) (b)

Figure 4.5: Comparison of delay variation with temperature for process *tt*, at 1.2 V: (a) average delay, (b) maximum delay.

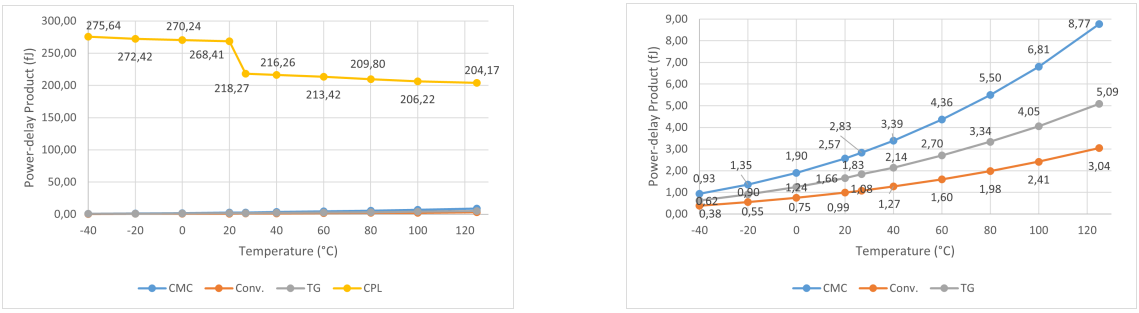


(a) (b)

Figure 4.6: Comparison of delay variation with supply voltage for process *tt*, at room temperature: (a) average delay, (b) maximum delay.

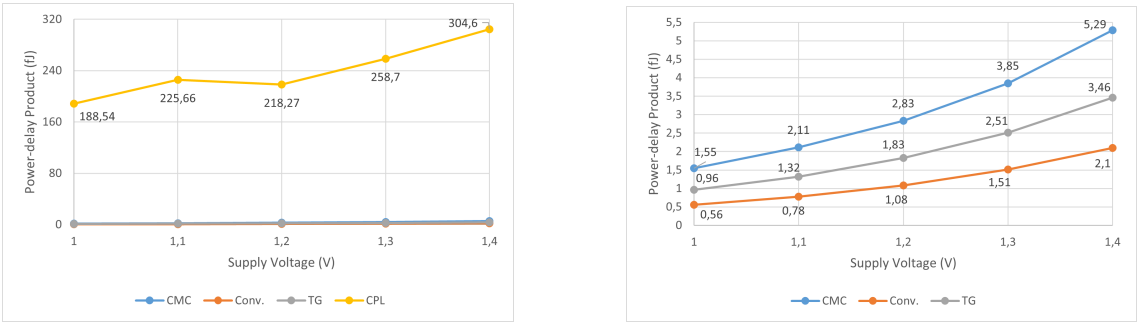
If the CMC is the best choice out of the four architectures when considering only leakage power, in terms of propagation delay it is, indubitably, the worst (see figures 4.5 and 4.6). The TG and Conventional adder exhibit a fairly "constant" behaviour and robustness in delay to both temperature and supply voltage, the largest interval being on the CPL in both average and maximum values with the interval of 5.16 ns and 5.62 ns, respectively, as shown in figure 4.6.

Since the power-delay product values will be further analysed when comparing processes in subsection 4.2.4, comparing the outputs of figures 4.7 and 4.8, one can assert already some conclusions: the worst performance architecture is the CPL full-adder cell in both temperature and supply voltage conditions. This is expected taking into account that the CPL exhibits the worst performance in both delay and leakage power individually; the CMC adder PDP value increases considerably as temperature and supply voltage increase, making this architecture more adequate for lower VT conditions; the TG and Conventional adder cells present the overall best compromise between delay and leakage power in all conditions, and the Conventional adder exhibits the most robustness regarding VT conditions, having the smallest variation interval: 2.66 fJ and 1.54 fJ in temperature and supply voltage variation, respectively, making it the more adequate for applications with high condition variability.



(a) (b)

Figure 4.7: Comparison of power-delay product variation with temperature for process *tt*, at 1.2 V: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



(a) (b)

Figure 4.8: Comparison of power-delay product variation with supply voltage for process *tt*, at room temperature: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.

4.2.2 Full-adders comparison for process *ff*

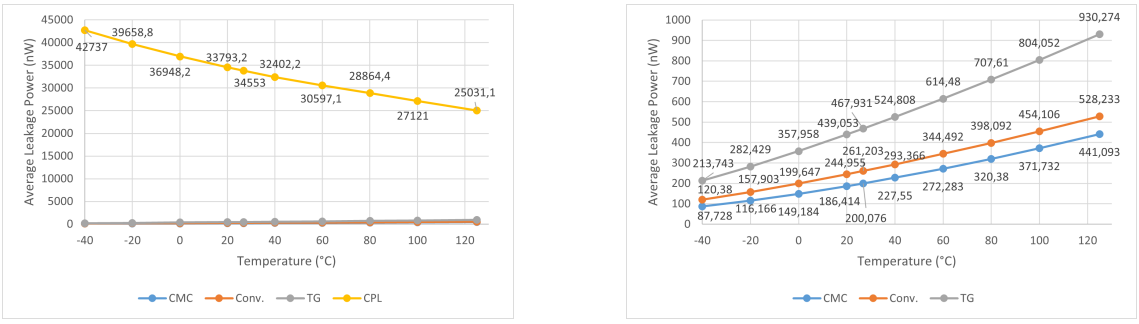
When considering a different process, the general behaviour of the analysed architectures is similar to the one exhibited within the process *tt*, with some worth noting differences.

For the leakage power variation with temperature, the behaviour is similar, the best and worst performing architectures are the same as in process *tt*, but with much higher maximum values, up to 129 % of the maximum leakage power value, in the CPL adder case (see figures 4.9 and 4.10).

In the case of supply voltage variation, the difference is even higher, reaching maximum values of leakage power of 194 %, almost double of the maximum values of process *tt*, as it is the case with the CMC adder; even the smallest difference in maximum values is still of approximately 30 % more in the fast process compared to the typical, as it happens with the CPL adder (figures 4.11 and 4.12) .

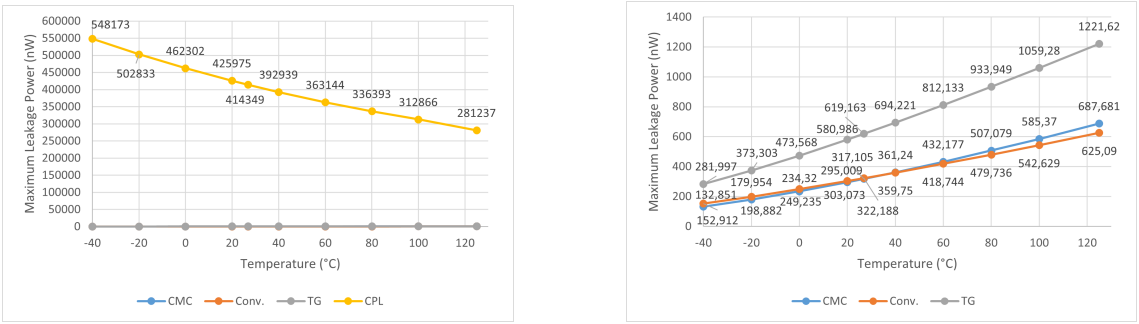
When comparing the delay values, depicted in figures 4.13 and 4.14, all architectures exhibit the same behaviour previously seen on the last subsection but now with lower values. In addition,

the CPL, Conventional and TG adder cells present an even smaller delay variation with both the temperature and supply voltage sweeps.



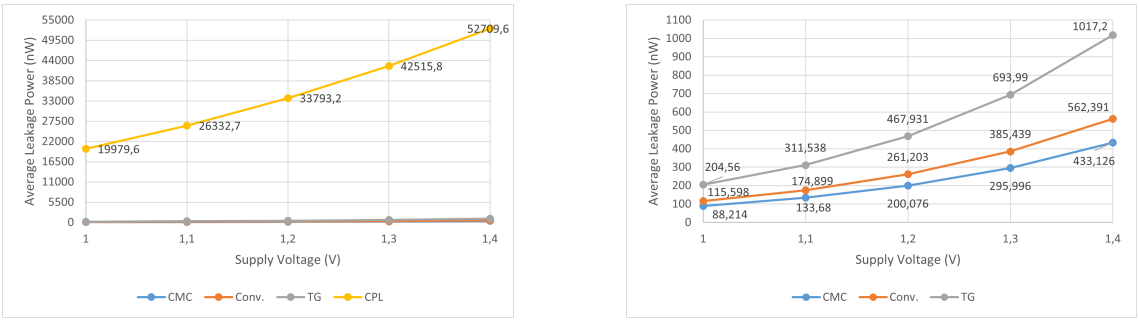
(a) (b)

Figure 4.9: Comparison of average leakage power variation with temperature for process *ff*, at 1.2 V: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



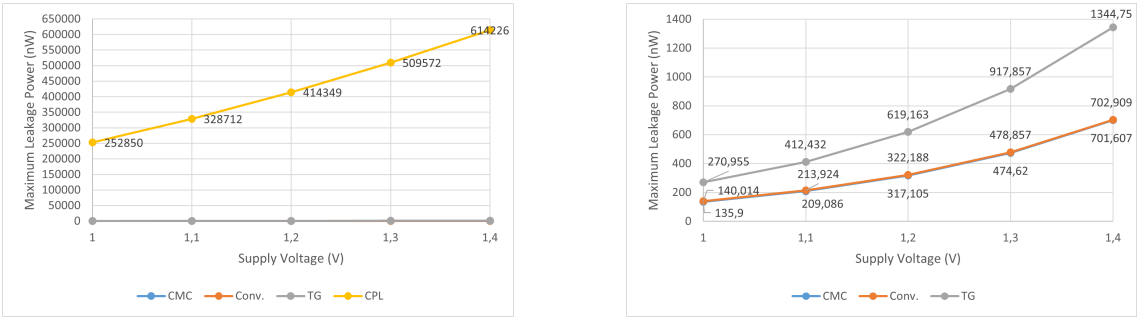
(a) (b)

Figure 4.10: Comparison of maximum leakage power variation with temperature for process *ff*, at 1.2 V: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



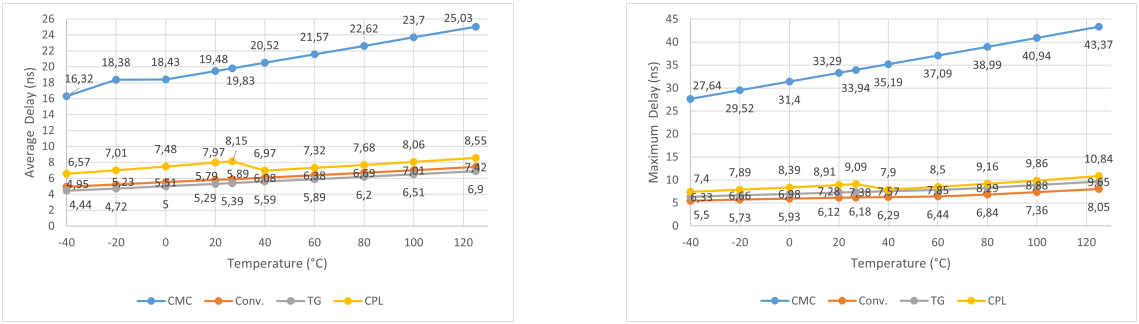
(a) (b)

Figure 4.11: Comparison of average leakage power variation with supply voltage for process *ff*, at room temperature: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



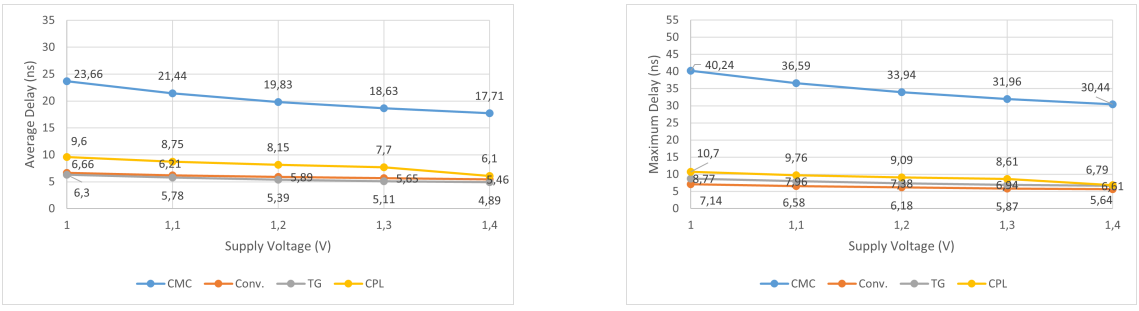
(a) (b)

Figure 4.12: Comparison of maximum leakage power variation with supply voltage for process *ff*, at room temperature: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



(a) (b)

Figure 4.13: Comparison of delay variation with temperature for process *ff*, at 1.2 V: (a) average delay, (b) maximum delay.

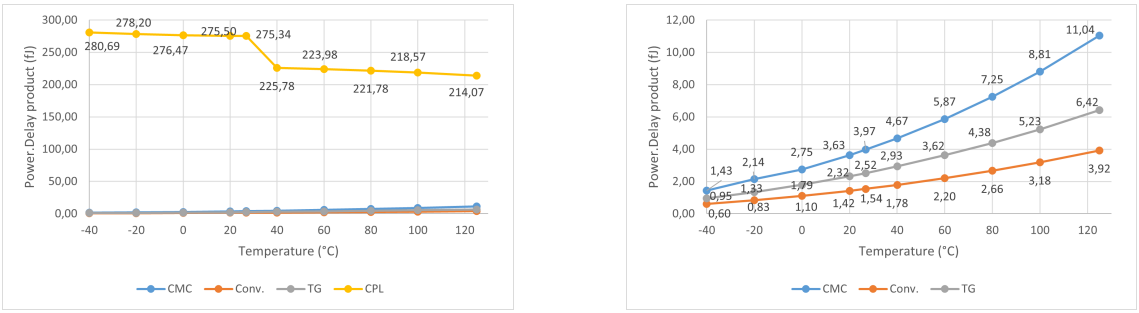


(a) (b)

Figure 4.14: Comparison of delay variation with supply voltage for process *ff*, at room temperature: (a) average delay, (b) maximum delay.

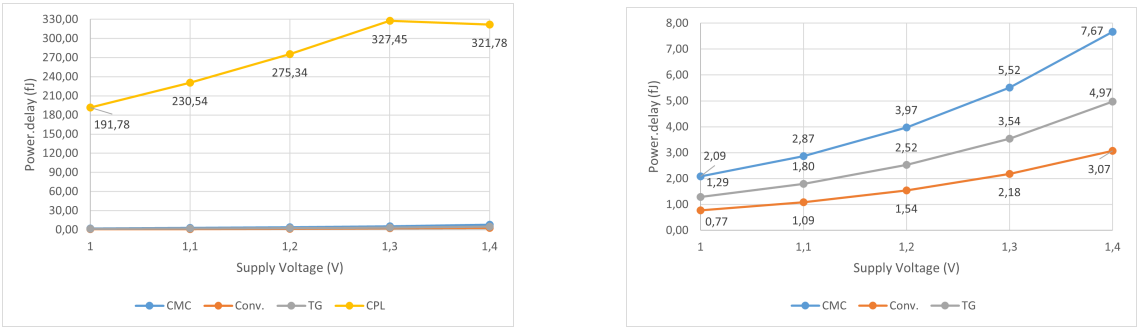
The power-delay product comparison, depicted in figures 4.15 and 4.16, shows the same results regardless of the process (*tt* or *ff*). The best performance tradeoff architectures remain the

Conventional and the Transmission gate full-adders, in this order. The worst, by a large margin, is the CPL full-adder architecture.



(a) (b)

Figure 4.15: Comparison of power-delay product variation with temperature for process *ff*, at 1.2 V: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.

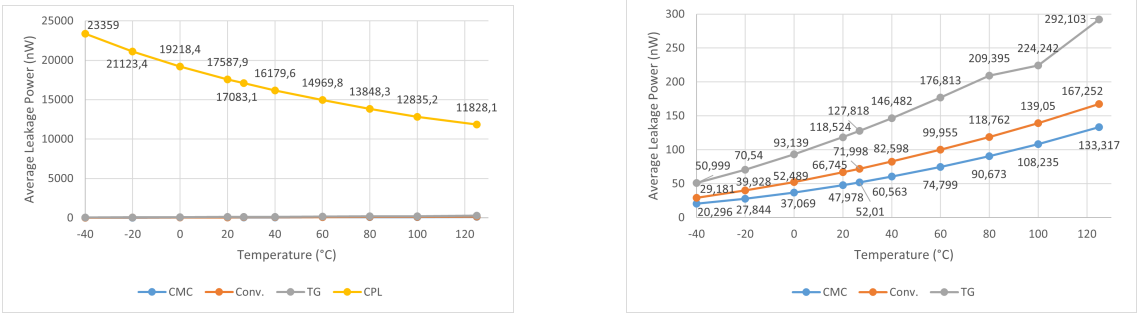


(a) (b)

Figure 4.16: Comparison of power-delay product variation with supply voltage for process *ff*, at room temperature: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.

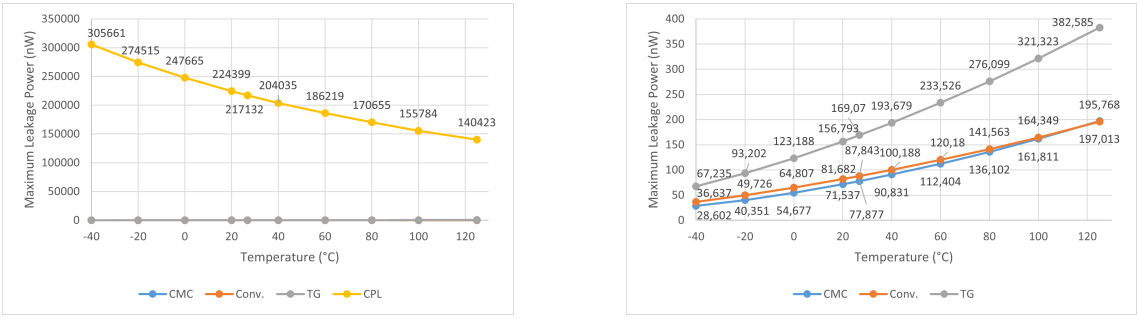
4.2.3 Full-adders comparison for process *ss*

If leakage power consumption is a main constrain when choosing a process and architecture, then the choice should be the CMC adder and process *ss*. When observing figures 4.17 and 4.18, it is obvious that the CMC cell exhibits the best results, followed closely by the Conventional adder, especially considering the maximum values for leakage power consumption. The power values for each temperature are also lower for process *ss*; for instance, the maximum value for the CPL adder is, approximately, 71 % of the maximum value for process *tt* and 55 % of the maximum value for process *ff*. The evolution of the curves is similar for the temperature and supply voltage variation (see also figures 4.19 and 4.20 for the supply voltage variation).



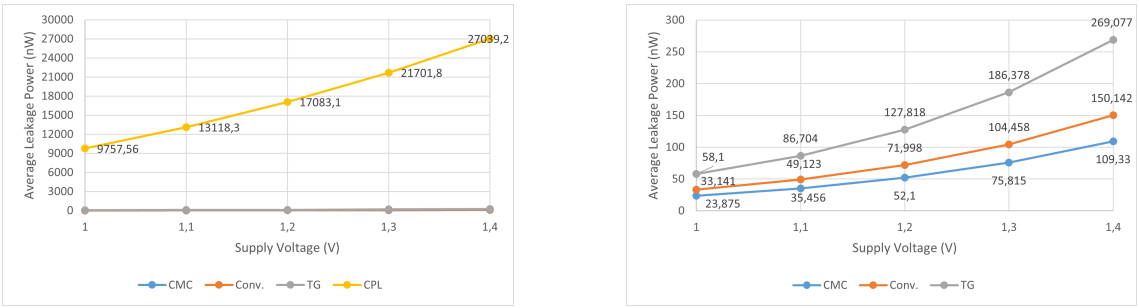
(a) (b)

Figure 4.17: Comparison of average leakage power variation with temperature for process *ss*, at 1.2 V: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



(a) (b)

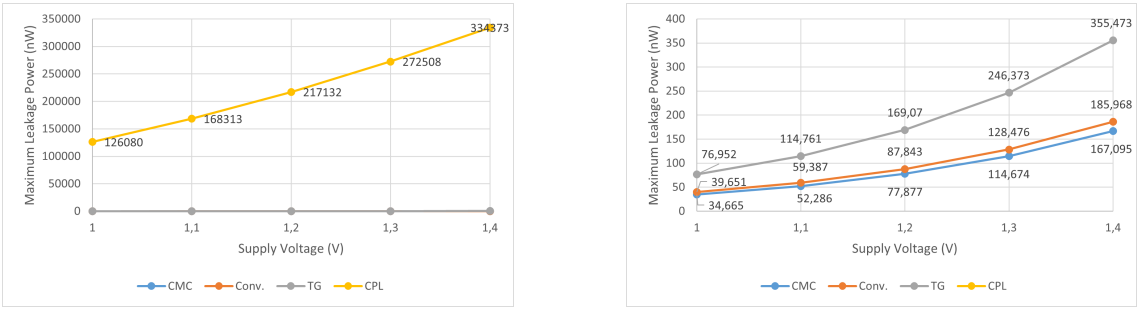
Figure 4.18: Comparison of maximum leakage power variation with temperature for process *ss*, at 1.2 V: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



(a) (b)

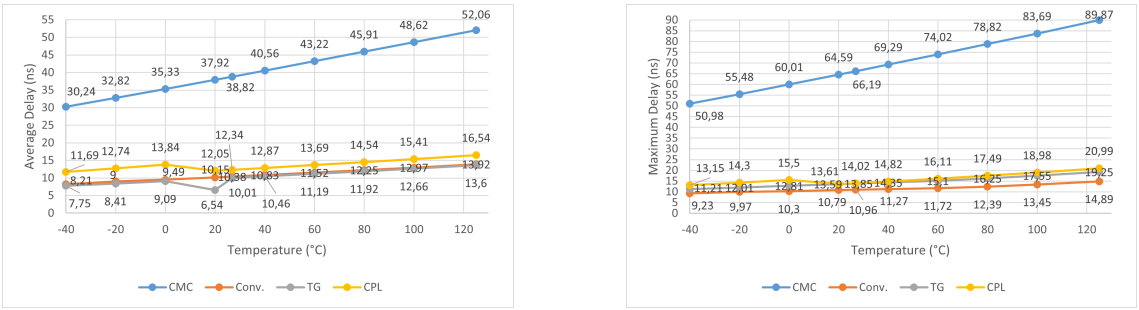
Figure 4.19: Comparison of average leakage power variation with supply voltage for process *ss*, at room temperature: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.

Regarding propagation delay, figures 4.21 and 4.22 indicate a similar behaviour of the adder cells as the one exhibited on the other processes. On the contrary of what happens with leakage power, when comparing the values for propagation delay, process *ss* exhibits higher values. The



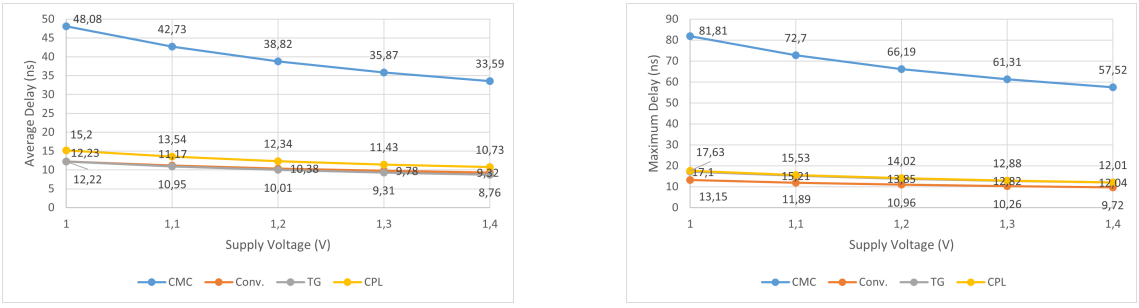
(a) (b)

Figure 4.20: Comparison of maximum leakage power variation with supply voltage for process *ss*, at room temperature: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



(a) (b)

Figure 4.21: Comparison of delay variation with temperature for process *ss*, at 1.2 V: (a) average delay, (b) maximum delay.

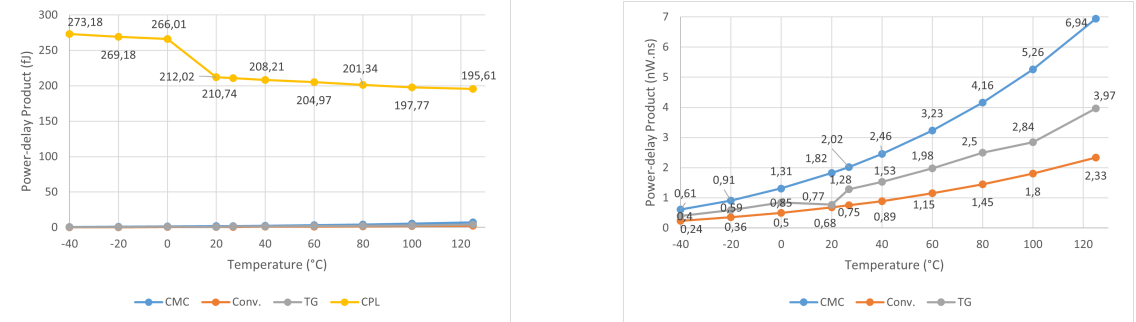


(a) (b)

Figure 4.22: Comparison of delay variation with supply voltage for process *ss*, at room temperature: (a) average delay, (b) maximum delay.

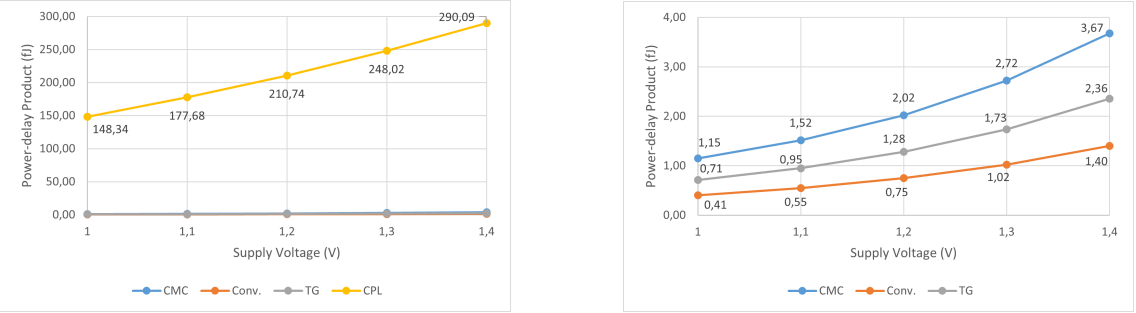
best performing architectures are the Conventional and the TG full-adders and the worst performance in terms of delay is the CMC full-adder. The former presents maximum values up to 6 times superior to other architectures under the same VT conditions, around 50 % higher than process *tt* and 100 % higher than process *ff*.

Lastly, the power-delay product results, available in figures 4.23 and 4.24 indicate that the Conventional full-adder is consistently the architecture which presents the best compromise between delay and power, and that the CPL architecture is the one that presents the worst.



(a) (b)

Figure 4.23: Comparison of power-delay product variation with temperature for process *ss*, at 1.2 V: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.



(a) (b)

Figure 4.24: Comparison of power-delay product variation with supply voltage for process *ss*, at room temperature: (a) emphasis on CPL full-adder, (b) emphasis on CMC, Conventional and TG full-adders.

4.2.4 Power-delay product comparison for different processes

For the comparison of power-delay product variation with supply voltage, the values are available on table 4.2. From direct observation, it is easy to conclude that the CPL presents the worst compromise between delay and leakage power and that the Conventional full-adder presents the best compromise in terms of delay and leakage power consumption. It should be noted that those assertions are true and independent of the process. The performance degrades as the supply voltage increases for all architectures, being the most accentuated on CPL adder in process *ss* and least accentuated on the Conventional adder in both *tt* and *ss* process.

Table 4.3 displays the data on power-delay product variation with temperature at the supply voltage of 1.2 V. The degradation of the performance tradeoff is different because not all the architectures exhibit the same behaviour as the temperature increases. Conventional, CMC and TG full-adders all present a better performance at lower temperatures and it tends to worsen as the temperature rises, as seen previously. However, the CPL adder exhibits the opposite behaviour, having the worst performance tradeoff at -20 ºC. It is also clear that the variation is higher throughout the temperature scale and that the process *ss* presents the best results for tradeoff performance analysis.

Table 4.2: Power-delay product comparison of the full-adders in different processes with different supply voltages at room temperature.

VDD (V)	1.0	1.1	1.2	1.3	1.4
FA	Process *tt* power-delay product (fJ)				
Conv.	0.56	0.78	1.08	1.51	1.51
CMC	1.55	2.11	2.83	3.85	5.29
TG	0.96	1.32	1.83	2.51	3.46
CPL	188.54	225.66	218.27	258.7	304.6
	Process *ff* power-delay product (fJ)				
Conv.	0.77	1.09	1.54	2.18	3.07
CMC	2.09	2.87	3.97	5.52	7.67
TG	1.29	1.80	2.52	3.54	4.97
CPL	191.78	230.54	275.34	327.45	321.78
	Process *ss* power-delay product (fJ)				
Conv.	0.41	0.55	0.75	1.02	1.40
CMC	1.15	1.52	2.02	2.72	3.67
TG	0.71	0.95	1.28	1.73	2.36
CPL	148.34	177.68	210.74	248.02	290.09

4.3 Comparative analysis

To provide a broader context for the obtained results, this section is dedicated to comparing them with layouts produced in other designs. Specifically, the results are compared with those of other authors who have used the same MOS 90 nm technology and are available on tables 4.4 and 4.5. It is worth noting that this comparison is biased and serves only to establish some context for the obtained values since most of the simulation conditions for the the selected works are unspecified.

In terms of the number of transistors utilized, there is consistency among the authors considered and this study; therefore, the transistor count is omitted from the comparison tables.

Unfortunately, there is only one work with information regarding area from all the works selected for comparison, [10], that had a Conventional FA design area of 168.56 μm^2. The area obtained in this study for the Conventional full-adder cell was 28.187 μm^2 (see table 4.1), which is significantly smaller, approximately 16.7 % of the area presented by the authors in [10].

Table 4.3: Power-delay product comparison of the full-adders in different processes with different temperatures at 1.2 V.

T (ºC)	−20	−40	0	20	26.85	40	60	80	100	125
FA	Process *tt* power-delay product (fJ)									
Conv.	0.38	0.55	0.75	0.99	1.08	1.27	1.60	1.98	2.41	3.04
CMC	0.93	1.35	1.90	2.57	2.83	3.39	4.36	5.50	6.81	8.77
TG	0.62	0.90	1.24	1.66	1.83	2.14	2.70	3.34	4.05	5.09
CPL	275.64	272.42	270.24	268.41	218.27	216.26	213.42	209.80	206.22	204.17
	Process *ff* power-delay product (fJ)									
Conv.	0.60	0.83	1.10	1.42	1.54	1.78	2.20	2.66	3.18	3.92
CMC	1.43	2.14	2.75	3.63	3.97	4.67	5.87	7.25	8.81	11.04
TG	0.95	1.33	1.79	2.32	2.52	2.93	3.62	4.38	5.23	6.42
CPL	280.69	278.20	276.47	275.50	275.34	225.78	223.98	221.78	218.57	214.07
	Process *ss* power-delay product (fJ)									
Conv.	0.24	0.36	0.5	0.68	0.75	0.89	1.15	1.45	1.8	2.33
CMC	0.61	0.91	1.31	1.82	2.02	2.46	3.23	4.16	5.26	6,94
TG	0.4	0.59	0.85	0.77	1.28	1.53	1.98	2.5	2.84	3.97
CPL	273.18	269.18	266.01	212.02	210.74	208.21	204.97	201.34	197.77	195.61

When comparing the power values for the Conventional full-adder, it is possible to identify that the process *ss* presents better results than the work presented in [16] at 1.2 V. The obtained result of 71.998 nW is 72.57 % of the result obtained by the author. For processes *tt* and *ff*, the results are significantly higher, 45.7 % and 163 %, respectively. As for the comparison with work [10], for 1.0 V, all processes present a lower power consumption, from, approximately, 213 to 61.2 times smaller (processes *ss* and *ff*, respectively).

For the delay comparison, the proposed Conventional full-adder layout performs considerably worse, all processes delay values indicate a much slower slower circuit, up to approximately 21 times slower for 1.2 V and 20 times slower for 1.0 V, both results using the worst value obtained for process *ss*. Since there is no information from reference [10] regarding the power-delay product, only the results obtained at 1.2 V will be compared. As it it the case for all four analysed architectures, the best compromise is verified when using process *ss*. In the case of the Conventional, the PDP obtained is still, approximately, 16 times larger than that of the results obtained in [16].

The CMC layout developed presents much lower power consumption in all processes that the design of the work performed in [9]. The latter presents values, approximately, 16, 8.5 and 33 times higher than the one proposed by this study in processes *tt*, *ff* and *ss*, respectively. Regarding the propagation delay, however, the author presents much faster circuit design, only 0.68 % of the best case delay obtained within process *ff*. Confirming the previous analysis, the PDP values obtained are at least 5.73 times higher than the ones from the author, which indicates a worst performance tradeoff for the layout designed.

Regarding the power values for the TG full-adder, one can assert that, similarly to the Conv.

and the CMC adder cells, the obtained power values are much lower than those of the authors of [16] and [9]. This difference is minimum when comparing process *ff* with the power from the work on reference [16], and the obtained value is still only about 137 % of the value presented by the author. Regarding delay, both the authors present significantly better results, up to almost 71 times smaller. Once again, the PDP comparison reveals that the obtained results show a worse tradeoff than those of the previous authors, especially those presented in [16], which reveal a great compromise between delay and power consumption.

As for the the CPL architecture, all the obtained values reflect a notable worse performance when comparing with other authors work, such as [9], as well as with the other studied architectures in all parameters. These bad performance results are likely due to the fact that this circuit, being the most different of the analysed four architectures, presents twice the number of inputs and outputs of the other architectures (every input and output and also its complement), and the output signals are generated in two different subcircuits (see chapters 2 and 3). This creates a larger number of connections leading to higher values of power dissipation. Furthermore, the NMOS/PMOS ratio for this cell is significant, of $24/8$, which caused the layout to be the most challenging, with a higher number of connections, most of them in parallel, which causes higher values for the parasitic capacitances and resistors. Changing and adapting the layout is a possible way of improving the characterization results obtained, specially the power dissipation values.

Table 4.4: Performance results of previous works for comparison.

FA	Source	VDD (V)	Delay (ns)	Power (nW)	PDP (fJ)
Conv.	[16]	1.2	0.4779325	99.218	0.0461
	[10]	1.0	0.5919	7080	-
CMC	[9]	1.2	0.135	1720	0.352
TG	[16]	1.2	0.350115	341.49	0.00143
	[9]	1.2	0.141	1715	0.242
CPL	[9]	1.2	0.093	2183	0.203

4.4 Result analysis

Following the analysis on the obtained results throughout this chapter, it is now possible to compare the four different architectures with additional detail, presenting their stronger and weaker characteristics with regard to the metrics used and PVT variations.

The CPL adder demonstrates to be the circuit most sensible to environment variability, particularly temperature and supply voltage variations. On the other hand, the Conventional and the Transmission gate adders demonstrate the smaller dependence to PVT variations, making them a more adequate choice for applications that undergo significant condition variability.

The most sensitive metric to VT variations is the leakage power in all architectures and processes. The worst performing cell design is the CPL, regardless of the process. Despite the high

Table 4.5: Obtained performance results for comparison at room temperature.

FA	Process	VDD (V)	Delay (ns)	Power (nW)	PDP (fJ)
Conv.	*tt*	1.2	7.48	144.558	1.08
	ff	1.2	5.89	261.203	1.54
	ss	1.2	10.38	71.998	0.75
	tt	1.0	8.62	65.227	0.56
	ff	1.0	6.66	115.598	0.77
	ss	1.0	12.23	33.141	0.41
CMC.	*tt*	1.2	26.41	107.245	2.83
	ff	1.2	19.83	200.076	3.97
	ss	1.2	38.82	52.1	2.02
TG.	*tt*	1.2	7.1	258.504	1.83
	ff	1.2	5.39	467.931	2.52
	ss	1.2	10.01	127.818	1.28
CPL.	*tt*	1.2	8.7	25075.4	218.27
	ff	1.2	8.15	33793.2	275.34
	ss	1.2	12.34	17083.1	210.74

variation, the best performing architectures in terms of leakage power are the CMC and the Conventional adder with very similar results.

At the same time, the propagation delay is the least sensitive metric regarding VT variations, although the variation is considerably higher for the CPL cell. The best performing architectures in speed are the Conventional and the TG adders and the worst is now the CMC full-adder cell. Since the Conventional full-adder cell demonstrates one of the best performances in both power and delay, it becomes clear that the power-delay product will reflect it as the best power-delay compromise performing architecture.

The power-delay product shows the least variation with PVT conditions on the Conventional FA, indicating that this architecture is very robust to environmental changes, which explains why it is still in use, being one of the simplest implementations.

Chapter 5

Conclusions and future work

The process of selecting and sizing the right component for a circuit has a huge impact on the overall performance of the application for which it was selected. Since the full-adder is a component present in many larger scale circuits, it is of great importance to select the best architecture for the requirements of a particular system.

The main goal of this project was to provide essential information regarding propagation delay and power in order to compare four common static full-adder architectures and facilitate the choice of a FA for a certain application. As the layouts for comparison were fully custom made, the first step was to create a methodological approach to design and test the logic function. The layouts were produced with the objective of minimum area utilization. For design comparison, the characterisation of each cell was based on the delay and power models available on Liberate and the results organized into graphs and tables for easier comparison.

According to the information obtained from this study, it is possible to make some educated assertions on the advantages and disadvantages of each cell design regarding environmental conditions variability. It was demonstrated that the Conventional FA, despite its high transistor count, is one of the best performing architectures under PVT variations and presents good performance in all metrics used for comparison such as leakage power and propagation delay. The Transmission gate full-adder design also presents very good results regarding propagation delay. However, if the system requirements are stricter regarding leakage power, the Mirror CMOS adder presents the best results in this metric, compromising on the propagation delay, i.e., the circuit presents lower power loss but it is slower. It is also possible to conclude that the CPL cell design is the least adequate due to its high sensitivity to environmental condition changes.

During the research, design, and characterisation process, some potential development and research areas came into view. The following section reflects on those areas and proposes some options for future work.

5.1 Future work

In the area of microelectronics, the evolution of techniques and technology is very rapid and so these types of studies should accompany their development. The study was conducted for a single technology of 90 nm but it could be expanded for other, smaller and more recent technologies such as 45 nm and lower.

This comparative analysis considered only four static architectures using different logic styles, such as transmission gates and pass-transistors. However, it could be expanded for other technology types such as Low-Power full-adder (LP), Transmission Function full-adder (TFA), Transmission gate with Driving Capability (TGDC), Dual-Rail Domino full-adder (DRD), and even hybrid circuits, which attempt to incorporate the best features of two or more logic styles into one cell. Considering static logic designs, the methodology and characterisation set up and files would not need in-depth alterations. Furthermore, the analysis and comparison of dynamic circuits would provide a broader scope of comparison of static and dynamic logic for different architectures at different PVT conditions. Due to the fact that dynamic logic has a different behaviour from static logic circuits, the methodology and testing conditions would need some deeper alterations and so would the characterisation setup files.

As for the PVT conditions characterisation, it could be expanded for more processes such as fast-slow (*fs*) and slow-fast (*sf*) allowed by each technology.

Furthermore, it would also be of interest to perform an analysis on transistor sizes, i.e., conducting sweep analysis to determine the optimized sizes for delay or for power consumption for different PVT conditions.

References

[1] Shamim Akhter, Saurabh Chaturvedi, Shaheen Khan, and Ankur Bhardwaj. An Efficient CMOS Dynamic Logic-Based Full Adder. In *2020 6th International Conference on Signal Processing and Communication (ICSC)*, pages 226–229, March 2020. ISSN: 2643-444X.

[2] M. Alioto and G. Palumbo. Analysis and comparison on full adder block in submicron technology. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 10(6):806–823, December 2002.

[3] Stéphanie O. Ames, Vinícius Zanandrea, Ingrid F. V. Oliveira, Samuel P. Toledo, and Cristina Meinhardt. Investigating PVT variability effects on full adders. In *2016 26th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, pages 155–161, September 2016.

[4] Cadence. Reference Manual for Generic 90 nm Salicide 1.2V/2.5V, Process Design Kit (PDK), February 2011. Revision 4.6.

[5] Cadence. Liberate Characterization Reference Manual, September 2023.

[6] Chip-Hong Chang, Jiangmin Gu, and Mingyan Zhang. A review of 0.18-/spl mu/m full adder performances for tree structured arithmetic circuits. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 13(6):686–695, June 2005.

[7] Gianluca Giustolisi and Gaetano Palumbo. Analysis and Comparison in the Energy-Delay Space of Nanometer CMOS One-Bit Full-Adders. *IEEE Access*, 10:75482–75494, 2022.

[8] K. Granhaug and S. Aunet. Six subthreshold full adder cells characterized in 90 nm CMOS technology. In *2006 IEEE Design and Diagnostics of Electronic Circuits and systems*, pages 25–30, April 2006.

[9] Mehedi Hasan, Md. Jobayer Hossein, Uttam Kumar Saha, and Md Shahariar Tarif. Overview and Comparative Performance Analysis of Various Full Adder Cells in 90 nm Technology. In *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, pages 1–6, December 2018. ISSN: 2642-7354.

[10] Omid Kavehei, Mostafa Rahimi Azghadi, Keivan Navi, and Amir-Pasha Mirbaha. Design of Robust and High-Performance 1-Bit CMOS Full Adder for Nanometer Design. In *2008 IEEE Computer Society Annual Symposium on VLSI*, pages 10–15, April 2008. ISSN: 2159-3477.

[11] Bhavani Koyada, N. Meghana, Md. Omair Jaleel, and Praneet Raj Jeripotula. A comparative study on adders. In *2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 2226–2230, March 2017.

[12] Gaetano Palumbo, Melita Pennisi, and Massimo Alioto. A Simple Circuit Approach to Reduce Delay Variations in Domino Logic Gates. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 59(10):2292–2300, October 2012.

[13] A.M. Shams, T.K. Darwish, and M.A. Bayoumi. Performance analysis of low-power 1-bit CMOS full adder cells. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 10(1):20–29, February 2002. Conference Name: IEEE Transactions on Very Large Scale Integration (VLSI) Systems.

[14] Rajneesh Sharma and Shekhar Verma. Comparative analysis of static and dynamic CMOS logic design. In *5th IEEE International Conference on Advanced Computing & Communication Technologies*, pages 231–234, 2011.

[15] Tripti Sharma, K.G. Sharma, and B.P. Singh. High performance full adder cell: A comparative analysis. In *2010 IEEE Students Technology Symposium (TechSym)*, pages 156–160, April 2010.

[16] Zarin Tabassum, Meem Shahrin, Aniqa Ibnat, and Tawfiq Amin. Comparative Analysis and Simulation of Different CMOS Full Adders Using Cadence in 90 nm Technology. In *2018 3rd International Conference for Convergence in Technology (I2CT)*, pages 1–6, April 2018.

[17] Nidhi Tiwari, Ruchi Sharma, and Rajesh Parihar. Implementation of area and energy efficient full adder cell. In *International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014)*, pages 1–5, May 2014.

[18] Chandran Venkatesan, Sulthana M. Thabsera, M. G. Sumithra, and M. Suriya. Analysis of 1- bit full adder using different techniques in Cadence 45nm Technology. In *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, pages 179–184, March 2019. ISSN: 2575-7288.

[19] Chua-Chin Wang, Kuan-Yu Chao, Sivaperumal Sampath, and Ponnan Suresh. Anti-PVT-Variation Low-Power Time-to-Digital Converter Design Using 90-nm CMOS Process. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 28(9):2069–2073, September 2020.

[20] Neil H. E. West and David Money Harris. *CMOS VLSI Design: a Circuit and Systems Perspective*. Addison-Wesley, Boston, 4th edition, 2011.

[21] Wei-Bin Yang, Yu-Yao Lin, and Yu-Lung Lo. Analysis and design considerations of static CMOS logics under process, voltage and temperature variation in 90nm CMOS process. In *2014 International Conference on Information Science, Electronics and Electrical Engineering*, volume 3, pages 1653–1656, April 2014.

Appendix A

Liberate datasheet example

The Liberate software produced HTML files with information regarding power, delay, pin capacitance, and energy. Liberate produced a file for each architecture and PVT conditions. In these datasheets one can observe a truth table of the logic function performed by the characterised circuit, the area of the layout, and input pin capacitance; it is worth noting that the output pin capacitance is always set to the maximum defined value, in this case 1.1559 pF. The information that follows is a result of the templates chosen for the characterisation: leakage power, propagation delay, and switching energy (named as switching power by the template - *power_template*). For all these parameters, the Liberate printed the minimum, average and maximum value. The following example is of one of these datasheets for the CMC full-adder cell for process *tt*, supply voltage of 1.2 V, at 26.85 ºC.

CMC_FA

tt Cell Library: Process tt_1.2_26.85, Voltage 1.20, Temp 26.85

Truth Table

INPUT			OUTPUT	
A	B	Cin	nCout	nSum
0	0	0	1	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	0	0

Footprint

Cell Name	Area
CMC_FA	29.23800

Pin Capacitance Information

Cell Name	Pin Cap(pf)			Max Cap(pf)	
	A	B	Cin	nCout	nSum
CMC_FA	0.00646	0.00663	0.00558	1.15590	1.15590

Leakage Information

Cell Name	Leakage(nW)		
	Min.	Avg	Max.
CMC_FA	16.51890	107.24500	165.53500

Delay Information

Delay(ns) to nCout rising :

Cell Name	Timing Arc(Dir)	Delay(ns)		
		Min	Avg	Max
CMC_FA	A->nCout (FR)	0.18112	13.80778	27.41130
	B->nCout (FR)	0.17549	13.79295	27.38810
	Cin->nCout (FR)	0.01947	1.66751	3.33436

Delay(ns) to nCout falling :

Cell Name	Timing Arc(Dir)	Delay(ns)		
		Min	Avg	Max
CMC_FA	A->nCout (RF)	0.09741	5.82195	11.53250
	B->nCout (RF)	0.09435	5.81154	11.51400
	Cin->nCout (RF)	0.01806	1.47990	2.96611

Delay(ns) to nSum rising :

Cell Name	Timing Arc(Dir)	Delay(ns)		
		Min	Avg	Max
CMC_FA	A->nSum (-R)	0.23679	22.68871	45.12440
	B->nSum (-R)	0.23352	22.67866	45.10870
	Cin->nSum (-R)	0.16818	19.44124	38.69640
	nCout->nSum (FR)	0.08166	10.57023	21.05380

Delay(ns) to nSum falling :

Cell Name	Timing Arc(Dir)	Delay(ns)		
		Min	Avg	Max
CMC_FA	A->nSum (-F)	0.24321	22.12219	43.98170
	B->nSum (-F)	0.23805	22.10955	43.96050
	Cin->nSum (-F)	0.06765	7.97222	15.89500
	nCout->nSum (RF)	0.03526	4.48212	8.95812

Power Information

Internal switching power(pJ) to nCout rising :

Cell Name	Input	Power(pJ)		
		min	avg	max
CMC_FA	A	0.00600	0.03724	0.06813
	B	0.00564	0.03696	0.06791
	Cin	0.00411	0.00601	0.00948

Internal switching power(pJ) to nCout falling :

Cell Name	Input	Power(pJ)		
		min	avg	max
CMC_FA	A	0.00062	0.00799	0.01458
	B	0.00057	0.00792	0.01450
	Cin	0.00030	0.00167	0.00544

Internal switching power(pJ) to nSum rising :

Cell Name	Input	Power(pJ)		
		min	avg	max
CMC_FA	A	0.00000	0.00253	0.00539
	B	0.00000	0.00252	0.00547
	Cin	0.00000	0.00167	0.00380

Internal switching power(pJ) to nSum falling :

Cell Name	Input	Power(pJ)		
		min	avg	max
CMC_FA	A	0.00472	0.00553	0.00723
	B	0.00540	0.00618	0.00728
	Cin	-0.00167	-0.00059	0.00000

Next (CONT_FA)

Appendix B

Liberate script files

B.1 *char.tcl* file

The characterisation file is the "run script" and calls on the *template.tcl* and *settings.tcl* files. It also sets the directories and the PVT parameters and writes the liberate database and Liberty file.

```
1  # Example Liberate Tcl File
2  #####----------------------------------------
3  ##### Set and print user define variables
4  #####----------------------------------------
5  set SRC_DIR              [pwd]    ;# directory where all source data (netlist,
       models, etc...) are stored
6  set RUN_DIR              [pwd]    ;# directory where all generated data (ldb,
       liberty, etc...) are saved
7  set LIB                  schematic ;# av_extracted lib name
8
9  set PROCESS              tt    ;# [ff|tt|ss]
10 set VDD                  1.2
11 set TEMP                 25
12 set SETTINGS_FILE        ${SRC_DIR}/tcl/settings.tcl
13 set TEMPLATE_FILE        ${SRC_DIR}/template/template.tcl
14 set CELLS_FILE           ${RUN_DIR}/cells.tcl
15 set NETLIST_DIR          ${SRC_DIR}/netlist
16 set USERDATA             ${SRC_DIR}/userdata/userdata.lib
17
18 #####----------------------------------------
19 ##### Process command line input
20 #####   Allow cmdline option to set local TCL variables - liberate <script> <var>=<
       value>
21 #####   ex: liberate char.tcl VDD=3.6
22 #####----------------------------------------
23 puts "INFO: Process command line input:"
24 foreach arg $argv {
25     if { [string match *=* $arg] } {
```

```
26          lassign [split $arg =] a b
27          set $a $b
28          puts "INFO:   Setting $a = $b"
29      }
30  }
31
32  #####-----------------------------------------
33  ##### Set dependent variables
34  #####-----------------------------------------
35  set PVT                  ${PROCESS}_${VDD}_${TEMP}
36  set LIBNAME              ${LIB}_${PVT}
37  set MODEL_INCLUDE_FILE   ${SRC_DIR}/models/spectre/include_${PROCESS}
38
39  #####-----------------------------------------
40  ##### Distributed Resource Management (DRM) setup
41  #####   Liberate supports both LSF and SunGrid - just update variable rsh_cmd
42  #####     LSF    : set RSH_CMD "bsub -q <queueName> -n $THREAD -R <
        ResourceDefinition> -o %B/%L -e %B/%L"
43  #####     SunGrid: set RSH_CMD "qsub -b y -q <queueName> -n $THREAD"
44  #####     Local  : set RSH_CMD "local"   ;# local machine only
45  #####   Variables :
46  #####     THREAD  - number of cpus to use on a given machine; 0-use all cpus on
        machine
47  #####     CLIENTS - number of Distributed Resource Management (DRM) jobs; 0=disable
         DRM
48  #####-----------------------------------------
49  ### run using debug mode ###
50  set THREAD    1
51  set CLIENTS   0
52  ### run using local mode ###
53  #set THREAD    1
54  # set CLIENTS   1
55  # set RSH_CMD   "local"
56  ### run using distributed mode ###
57  # set THREAD    2
58  # set CLIENTS   4
59  # if {![info exists MEM]} {set MEM [expr ${THREAD}*2000]}   ;# default = 2G per
        thread
60  # if {![info exists JOBNAME]} {set JOBNAME liberate}
61  # set RSH_CMD   "bsub -q lnx64 -J ${JOBNAME} -W 255:0 -P LIBERATE:17.1:AE:test -n $
        {THREAD} -R \"(OSREL==EE60) rusage\[mem=$MEM\] span\[hosts=1\]\" -o %B/%L -e %B
        /%L"
62  #set_var packet_arcs_per_thread 1   ;# use for large cells, default=10
63
64  #####-----------------------------------------
65  ##### Print user define settings to output
66  #####-----------------------------------------
67  puts "INFO:"
68  puts "    SRC_DIR              = ${SRC_DIR}"
```

```
69  puts "    RUN_DIR              = ${RUN_DIR}"
70  puts "    LIBNAME              = ${LIBNAME}"
71  puts "    PVT                  = ${PVT}"
72  puts "    SETTINGS_FILE        = ${SETTINGS_FILE}"
73  puts "    TEMPLATE_FILE        = ${TEMPLATE_FILE}"
74  puts "    MODEL_INCLUDE_FILE   = ${MODEL_INCLUDE_FILE}"
75  puts "    NETLIST_DIR          = ${NETLIST_DIR}"
76  puts "    USERDATA             = ${USERDATA}"
77  if { [info exists CELLS_FILE] } { puts "    CELLS_FILE           = ${CELLS_FILE}" }
78  puts ""
79  puts "    THREAD               = ${THREAD}"
80  if { [info exists CLIENTS] } {
81      puts "    CLIENTS              = ${CLIENTS}"
82      if { [info exists RSH_CMD] } { puts "    RSH_CMD              = ${RSH_CMD}" }
83  }
84  puts ""
85
86  #####----------------------------------------
87  ##### Set operating condition
88  #####----------------------------------------
89  puts "INFO: Set Operating Condition"
90  set_operating_condition -name ${PVT} -voltage ${VDD} -temp ${TEMP}
91
92  #####----------------------------------------
93  ##### Set Liberate variables
94  #####----------------------------------------
95  puts "INFO: Read settings file ${SETTINGS_FILE}"
96  source ${SETTINGS_FILE}
97
98  #####----------------------------------------
99  ##### Debug variables
100 #####----------------------------------------
101 # select_index -style 1x1             ;# run only 1st point in table
102
103
104 #####----------------------------------------
105 ##### Read template
106 #####----------------------------------------
107 puts "INFO: Read template file ${TEMPLATE_FILE}"
108 source ${TEMPLATE_FILE}
109
110 #####----------------------------------------
111 ##### Read CELLS_FILE
112 #####----------------------------------------
113 if {[info exists CELLS_FILE]} {
114     if {[file exists ${CELLS_FILE}]} {
115   puts "INFO: Read cell list file"
116   source ${CELLS_FILE}
117     } else {
```

```
118   puts "WARNING: Specified CELLS_FILE (${CELLS_FILE}) does not exist."
119     }
120 }
121
122 #####----------------------------------------
123 ##### define device models
124 #####----------------------------------------
125 puts "INFO: Define device models (spectre, define_leafcell)."
126 set_var extsim_model_include ${MODEL_INCLUDE_FILE}
127 define_leafcell -type nmos -pin_position { 0 1 2 3 } { gpdk090_nmos1v }
128 define_leafcell -type pmos -pin_position { 0 1 2 3 } { gpdk090_pmos1v }
129
130 #####----------------------------------------
131 ##### read cell netlists
132 #####----------------------------------------
133 puts "INFO: Read cell netlist "
134 ## setup client to read only netlist of cells being characterized
135 set packet_cells [packet_slave_cells]
136 if {[llength $packet_cells]>0} { set cells $packet_cells }
137 ## read netlist
138 set spicefiles {}
139 foreach cell $cells { lappend spicefiles ${NETLIST_DIR}/${cell}.sp }
140 read_spice -format spectre "$MODEL_INCLUDE_FILE $spicefiles"
141
142 #####----------------------------------------
143 ##### Distributed Resource Management (DRM) setup
144 #####----------------------------------------
145 if { [info exists CLIENTS] && ($CLIENTS > 0) } {
146     set_var packet_mode      arc
147     set_var rsh_cmd          $RSH_CMD
148     if { ($RSH_CMD eq "local") } {
149   set_var packet_clients 1
150     } else {
151   set_var packet_clients $CLIENTS
152   if {${THREAD}<1} {
153       puts "WARNING: When using DRM, THREAD=0 can cause client machine to overload;
                resetting THREAD=1."
154       set THREAD 1
155   }
156     }
157 }
158
159 #####----------------------------------------
160 ##### Run characterization
161 #####----------------------------------------
162 ##  Simple command
163 puts "INFO: Run Characterization"
164 char_library -extsim spectre -cells $cells -thread $THREAD
165
```

```
166 ##  Build command string then eval
167 #set charCmd "char_library -extsim spectre -cells \{$cells\}"
168 #if { [info exists THREAD] && ${THREAD}>0 } { set charCmd "$charCmd -thread ${
        THREAD}" }
169 #puts "INFO: Run Characterization - charCmd = $charCmd"
170 #eval "$charCmd"
171
172 #####----------------------------------------
173 ##### Write output
174 #####----------------------------------------
175 ### Write ldb ###
176 puts "INFO: Write ldb"
177 file mkdir ${RUN_DIR}/ldb
178 # In packet_arc mode, by default, existing ldb does not get overwritten. User
        should use -overwrite option
179 write_ldb -overwrite ${RUN_DIR}/ldb/${LIBNAME}.ldb
180
181 ### Write Liberty ###
182 puts "INFO: Write Liberty"
183 file mkdir ${RUN_DIR}/lib
184 write_library -driver_waveform -unique_pin_data -bus_syntax {[]} -user_data ${
        USERDATA} -overwrite -filename ${RUN_DIR}/lib/${LIBNAME}_nldm.lib ${LIBNAME}
```

B.2 *settings.tcl* file

The settings file contains the variable settings for Liberate. Most of these settings and commands are default settings for setting SPECTRE, Spectre Kernel Interface (SKI), input waveforms, arc generation, and variables constrains.

```
1  #$Id: settings.tcl,v 1.16 2016/08/24 20:55:41 ctai Exp ctai $#
2  #####----------------------------------------
3  ##### Set liberate variables
4  #####----------------------------------------
5  ### Get tool version ###
6  lassign [split [ALAPI_version] .] x1 x2 version(minor) version(sub)
7  set version(major) "${x1}.${x2}"
8
9  ### External Simulator (Spectre) settings ###
10 set_var extsim_cmd_option      "+aps +spice -mt +liberate +rcopt=2"
11 set_var extsim_deck_header     "simulator lang=spectre\nOpt1 options reltol=1e-4 \
       nsimulator lang=spice"
12 set_var extsim_option          "redefinedparams=ignore hier_ambiguity=lower
       limit=delta "
13 set_var extsim_leakage_option  "redefinedparams=ignore hier_ambiguity=lower
       limit=delta "
14
```

```
15
16 ### SKI ###
17 set_var ski_enable               1
18 set_var ski_clean_mode           1  ;# run $ALTOSHOME/bin/clean_sm.sh to clean up
       inactive semaphores
19 set_var ski_compatibility_mode   1
20 set_var power_tend_match_tran    1  ;# use tran_tend for power_tend for non-ski (
       match ski)
21
22 ### Misc ###
23 set_var parse_auto_define_leafcell   0 ;# disable auto leaf cell determination
24 set_var tmpdir /dev/shm          ;# /dev/shm - use local RAM disk for tmp dir, /tmp
        - use local disk
25 set_var extsim_deck_dir [file normalize "decks"]   ;# specify directory for SPICE
       decks and output files
26 set_var set_var_failure_action   error
27
28
29 ### Input waveform ###
30 set_var predriver_waveform       2 ;# 2=use pre-driver waveform
31
32
33 ### Arc Generation
34 #set_var init_pin_hidden_period  1e-08
35
36 ### Capacitance ###
37 set_var min_capacitance_for_outputs            1        ;# write min_capacitance
       attribute for output pins
38 # set_var measure_cap_lower_rise                 0
39 # set_var measure_cap_upper_rise                 0.5
40 # set_var measure_cap_upper_fall                 1
41 # set_var measure_cap_lower_fall                 0.5
42
43
44 ### Timing ###
45 set_var force_condition              5 ;#changed from 4 to 5 due to version
46
47 ### Constraint ###
48 set_var constraint_info                  2
49 #set_var constraint_search_time_abstol    1e-12    ;# 1ps resolution for bisection
       search
50 set_var nochange_mode                    1        ;# enable nochange_* constraint
       characterization
51 #avail version 17.1.2
52 if {($version(major)>=17.1) && ($version(minor)>=2)} {
53     set_var constraint_vector_mode           4        ;# look at metric=delay and
           glitch to determine worst case
54 }
55
```

```
56 ### min_pulse_width ###
57 set_var conditional_mpw            0       ;# 0=disable conditional mpw
58
59
60 ### Leakage ###
61 set_var max_leakage_vector                 [expr 2**10]
62 set_var leakage_float_internal_supply      0            ;# get worst case leakage
       for power switch cells when off
63 set_var reset_negative_leakage_power       1            ;# convert negative leakage
        current to 0
64
65
66 ### Power ###
67 set_var voltage_map                         1 ;# create pg_pin groups,
       related_power_pin / related_ground_pin
68 set_var pin_based_power                     0 ;# 0=based on VDD only; 1=power based
        on VDD and VSS (default); 1??
69 set_var power_combinational_include_output  0   ;# do not include output pins in
       when conditions for combinational cells
70
71 set_var force_default_group                 1
72 set_default_group -criteria                 {power avg}  ;# use average for default
        power group
73
74 set_var power_subtract_leakage              4
75 set_var subtract_hidden_power               2   ;# 2=subtract hidden power for all
       cells
76 set_var subtract_hidden_power_use_default   3   ;# 3=subtract hidden power from
       matched when condition then default group
77 set_var power_multi_output_binning_mode     1   ;# binning for multi-output cell
       considered for both timing and power arcs
78 set_var power_minimize_switching            1
79 set_var max_hidden_vector                   [expr 2**10]
80
81
82 ## CCS ###
83
84
85 ### CCSN ###
86 set_var ccsn_include_passgate_attr 1  ;# include pin level attribute is_unbuffered
       and has_pass_gate, and
87                                       # ccsn_*_stage group level attribute
                                            is_pass_gate
88 set_var ccsn_model_related_node_attr 1 ;# enable output of ccsn_first_stage
       attribute
89                                         #   related_spice_node, load_cap_rise,
                                            load_cap_fall
90 #set_var ccsn_tempus_promote_mode 1  ;# same as ccsn_model_related_node_attr
91
```

```
92  ### CCSP ###
93  set_var ccsp_segmentation_effort  3
94  set_var ccsp_min_pts              15   ;# CCSP accuracy
95  set_var ccsp_rel_tol              0.01 ;# CCSP accuracy
96  set_var ccsp_table_reduction      0    ;# CCSP accuracy
97  set_var ccsp_tail_tol             0.02 ;# CCSP accuracy
98  set_var ccsp_related_pin_mode     2    ;# use 3 for multiple input switching
        scnarios and Voltus only libraries
99
100
101 ## ECSM ##
102
103
104 ## ECSMN ##
105
106
107 ## ECSMP ##
108
109
110 ## EM ##
111 if { [info exists CHAR_EM_TECH_FILE] && ($CHAR_EM_TECH_FILE ne "") } { set_var
        em_tech_file [file normalize $CHAR_EM_TECH_FILE] }
112
113
114 #####-----------------------------------------
115 ##### Writing Output Files
116 #####-----------------------------------------
117 set_var write_library_is_unbuffered            1
118 set_var cell_use_both_ff_latch_groups          2 ;# allow use of multiple
        ff,latch,state_table groups in userdata file
119 set_var user_data_override { power_down_function pg_pin input_signal_level
        output_signal_level }
120 set_var sdf_cond_style                         1
121 set_var parenthesize_not                      0 ;# use !A instead of !(A)
122 set_var driver_type_model_pad_check            1 ;# enable fix to disable output of
         driver_type pin attribute for tie cells CCR1407896
123 set_var ccsn_print_is_needed_if_false_attr_value 1
124 set_default_group -criteria {constraint off}    ;# disable writing of default
        constraint arc
125 set_var write_library_allow_switching_and_hidden_power 1  ;# allow writing of
        switching and hidden power for same pin
126
127 #####-----------------------------------------
128 ##### liberate_lv
129 #####-----------------------------------------
130 if { $::LIBERATE_program == "LIBERATE_LV" } {
131     set validate_cells_per_bundle 10000
132 }
133
```

```
#####----------------------------------------
##### variety
#####----------------------------------------
if { $::LIBERATE_program == "VARIETY" } {
    set_var variation_mean_nominal_mode       4 ;# save mean, stddev, and skewness
        values (in addition to normal variation data) to ldb
    #set_var variation_static_partition_mode  2 ;# Enable logic-cone analysis (0
        =disable; 1=faster/less accurate; 3=slower/more accurate)
    #set_var non_linear_random_variation      3 ;# (1=characterize positive and
        negative variation; 3=trade off minor early accuracy for faster run time)
                                                 # use default of 0 for AOCV(fastest
                                                     )
    #####  Constraint  #####
    #set_var constraint_random_variation_search_time_abstol [expr [get_var
        constraint_search_time_abstol] * 0.1] ;# typically set to 10% of
        constraint_search_time_abstol
    set_var lvf_constraint_early_late_mode   1 ;# 1=output early and late sigma
        type for constraint
                                                 #   more accruate for non-gaussian
                                                     distribution
    # set_var mpw_variation                    1 ;# enable mpw variation
        characterization

    #####  Monte Carlo  #####
    set_var extsim_monte_option  "sampling=lds" ;# set Spectre Monte Carlo sampling
         method [standard lhs lds orthogonal]
                                                  # lhs=Latin-Hypercube,
                                                      lds=Low-Discrepancy Sequence
    #set_var extsim_cmd_option "[get_var extsim_cmd_option] +mp=5" ;#distribute
        Monte Carlo jobs (5 .alter per job)
}
```

B.3 *template.tcl* file

The template file contains the characterisation conditions, such as supply nets and transition thresholds definition, and sets up the templates for the desired measurements, in this case for power and delay measurements.

```
#$Id$

set_vdd -type primary vdd! $VDD
#set_gnd -type primary VSS 0
set_gnd -type primary gnd! 0

set_var slew_lower_rise 0.2
set_var slew_lower_fall 0.2
```

```
set_var slew_upper_rise 0.8
set_var slew_upper_fall 0.8

set_var measure_slew_lower_rise 0.2
set_var measure_slew_lower_fall 0.2
set_var measure_slew_upper_rise 0.8
set_var measure_slew_upper_fall 0.8

set_var delay_inp_rise 0.5
set_var delay_inp_fall 0.5
set_var delay_out_rise 0.5
set_var delay_out_fall 0.5

#????????? no -auto_index option in char_library cmd
set_var def_arc_msg_level 0     ;# reports error when no valid vectors for the arc
    are found
set_var process_match_pins_to_ports 1   ;#Enables strict pin to port mapping
set_var min_transition 6e-12    ;#Minimum allowable delay transition time (in
    seconds)
set_var max_transition 3e-10
set_var min_output_cap 1e-16    ;#Minimum allowable output capacitance (in Farads)

### Define templates - slew (0-100%) min,max=10ps,500ps
define_template -type delay      -index_1 {0.006 0.3 } -index_2 {0.002 1.1559 }
    delay_template
define_template -type power      -index_1 {0.006 0.3 } -index_2 {0.002 1.1559 }
    power_template

set cells {
   CMC_FA
   Conv_FA
   TG_FA
   CPL_FA
}

### Define related supply for all cells and pins #needed??
#set_pin_vdd -supply_name VDD $cells {*}
#set_pin_gnd -supply_name VSS $cells {*}

set cell CMC_FA
if {[ALAPI_active_cell $cell]} {
    define_cell \
       -input { A B Cin} \
       -output { nSum nCout } \
       -pinlist { A B Cin nSum nCout } \
       -delay delay_template \
       -power power_template \
       $cell
}
```

```
54
55 set cell Conv_FA
56 if {[ALAPI_active_cell $cell]} {
57     define_cell \
58        -input { A B Cin} \
59        -output { Sum Carry } \
60        -pinlist { A B Cin Sum Carry } \
61        -delay delay_template \
62        -power power_template \
63        $cell
64 }
65
66 set cell TG_FA
67 if {[ALAPI_active_cell $cell]} {
68     define_cell \
69        -input { A B Cin} \
70        -output { Sum Cout } \
71        -pinlist { A B Cin Sum Cout } \
72        -delay delay_template \
73        -power power_template \
74        $cell
75 }
76
77 set cell CPL_FA
78 if {[ALAPI_active_cell $cell]} {
79     define_cell \
80        -input { A nA B nB Cin nCin} \
81        -output { Sum nSum Cout nCout } \
82        -pinlist { A nA B nB Cin nCin Sum nSum Cout nCout } \
83        -delay delay_template \
84        -power power_template \
85        $cell
86 }
87    # define_index -index_2 {0.01 0.45 } -type {delay power} $cell
88
89     ## probe all outputs for common pin constraint and mpw arcs
90     # set_constraint_criteria -cells $cell -pin CK -probe {Q1 Q2} ;
91     # set_constraint_criteria -cells $cell -pin RN -probe {Q1 Q2} ;
92     # set_constraint_criteria -cells $cell -pin SE -probe {Q1 Q2} ;
93
94     # define_bundle_pins -use_pin D1  $cell D  {D1 D2}
95     # define_bundle_pins -use_pin SI1 $cell D  {SI1 SI2}
96     # define_bundle_pins -use_pin Q1  $cell Q  {Q1 Q2}
```