POLITECNICO DI TORINO

Master's Degree in Electronics Engineering

Microelectronics

Master's Degree Thesis

Digital interface for Advanced mixed-signal IP suitable for GaN power conversion

Supervisors	Candidate
Prof. Maurizio MARTINA	Michelangelo ORABONA
Ing. Juri GIOVANNONE	

MARCH 2022

La terza legge di Newton.
L'unico modo che gli umani hanno trovato per
andare avanti è lasciarsi qualcosa alle spalle.

Table of Contents

Summary

The increasing variety of smart-power applications starts to require not only a higher level of integration and highly efficient power conversion circuits but also flexibility. This last new important requirement can be met thanks to the inherent reconfigurability of digital control loops. In addition, gallium nitride (GaN) and silicon carbide (SiC) technologies have been increasingly used in recent years to implement power switches, with higher power conversion efficiency and reduced power losses, size, weight, and thus system cost.

To take full advantage of digital control circuits and High-Electron-Mobility-Transistor (HEMT) technology, several circuits have been developed and implemented in BCD technologies. This allows the integration of the CMOS digital circuitry required by the loop and the High Voltage power stage driver both on the same die.

The work presents a digital control loop based on a new digital PWM generator circuit and a digital interface suitable for this block is developed.

The thesis is divided in three sections. In the first one the BCD technologies [1] are presented, focusing on process capability and how it is used in Smart Power devices and System-on-Chip solutions (chapter 1). This is followed by a brief introduction to classic DC-DC converter topologies, with particular attention to the transfer function of the control loop in both the analog and digital domains (chapters 2 and 3) [2] [3] [4] [5] [6] [7]. Useful results related to this section are reported in appendices A,B and C.

In the second section is described the standard digital flow used to design and synthesize the circuit (chapter 4) [8] [9] [10] [11] [12]. This is followed by a description of the synchronization problem that affects the entire circuit and leads to the final digital interface (chapter 5). In Appendix D is reported a brief

description of the main timing constraints used to guide the synthesis tool.

In the final section mixed-signal simulations are presented with a description on how to set up the simulation environment. An example of script used to set up the HDL files for mixed-signal simulation is provided in appendix E. The last chapter contains comments on the results obtained and the overall implementation.

Chapter 1

Smart Power Technology

1.1 Introduction

Modern VLSI systems integrate system-on-chip (SoC) with a logical core and many interface elements on the same chip. Thanks to modern mixed-power technologies, it is possible to integrate digital, analog, and power devices. This makes possible to develop smart power devices, SoCs able to interface a microcontroller to an actuator.

1.1.1 History of System-on-Chips

SoC technology, which enables miniaturization of complex systems, dates back to the early 1970s, the new era of the digital watch when the first LED wristwatch was introduced. It required 44 logic ICs to be reduced to one chip. Some of the LED control circuits in this wristwatch were too large, so it was not a complete system.

It was in the 1980s that the personal computer revolution began, leading to major developments in system-on-a-chip technology. In 1991, this technology continued to flourish with the release of the AMD286ZX/LX family of SoCs.

Then, the boom in the cell phone industry enabled the development of SoCs in the 1990s. At the same time, many smaller chips begin to take over various peripheral functions, including audio, battery charging, keyboard, and LCD display drivers.

In the late 1990s, ARM Holdings began licensing its designs for fabless processors to other manufacturers. The future of ARM and the mobile phone market is largely credited to the development of the RISC processor, which used a reduced instruction set. It was very powerful and consumed less power than its competitors. It has become ideal for use in embedded systems, as can be seen in the hard disk SoC.

The trend in technology is to miniaturize everything onto a single chip. The new frontier is to attempt miniaturization of power electronics as well.

1.1.2 General structure of a System-on-Chip

A SoC is an integrated circuit or an IC, on which an entire electronic or computer system is integrated. It is, as the name implies, an entire system on a single chip. Depending on the type of system, it can perform a variety of functions, such as signal processing, wireless communications, artificial intelligence, or high-voltage management.

A general SoC has several blocks, based on the required complexity:

- **Input Interface:**it processes external inputs and adapts them to the characteristics of the internal blocks. It consists of analog blocks with high precision and low noise. It is characterized from a physical point of view (e.g. number of wires and pins) and from a logical point of view (e.g. serial or parallel communication).

- **Logic Control Circuit:** manufactured in CMOS technology to achieve low power dissipation and high integrability. They can manage various characteristics, such as the torque of a motor or rejection of unwanted current/voltage spikes.

- **Memories:** it enhances flexibility of the system, its dimension depends on the required complexity

- **Power supply circuits:** adapt the high voltage from the power grid to the level required by the internal circuits. It includes converters which transform the AC into DC current with high stability.

- **Power circuits:** to drive actuator and output transducer.

In short, smart power systems consist of switches controlled by a microcontroller that uses sensors to perform output regulation. The goal of a smart power device is to integrate all the blocks described above on the same chip, plus provide high-voltage management functions.

The power switches are usually DMOS switches that have low resistance and can handle high currents. They cannot be too small to maintain a certain gate length and distance between adjacent switches to ensure insulation. The scaling trend of power devices does not follow the same trend as digital technology, e.g. CMOS.

Power supply circuits have to cope with large temperature variations (from $-40\ C$ to $190\ C$) due to the heat that is given off by the DMOS. This requires more precautions, such as using packages that ensure good heat transfer.

1.2 BCD process

Smart-power systems can be used at their full potential when digital, analog, and power device driving stages are embedded into a monolithic solution. In principle, this should not be possible because the various manufacturing processes are generally incompatible.

In 1984, new technology has enabled this possibility: the BCD technology. It combines Bipolar, CMOS, and DMOS technology, the main technologies for analog, digital, and power electronics, on a single process platform [1]. This brings many advantages, such as:

- Higher energy efficiency
- improved reliability
- reduced electromagnetic interference
- Smaller footprint

Implementing all in a single die allows reducing the number of interconnects required, thereby reducing the bill of materials (BOM) and the overall cost of the system. It increases reliability and reduces manufacturing complexity.

All these advantages lead the BCD to be the most suitable technology for implementing smart-power IPs, as shown in Figure 1.1.

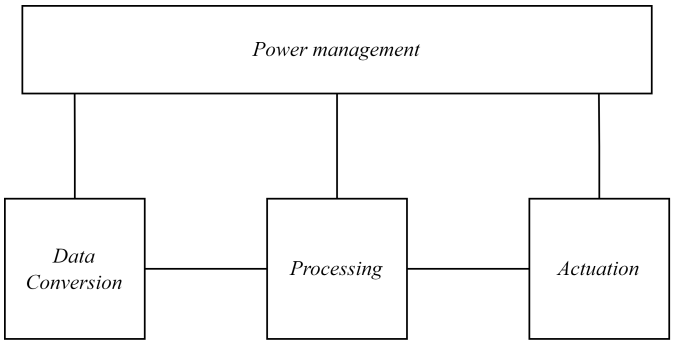


Figure 1.1: The ability of the BCD silicon process to integrate digital, analog and power technologies on the same chip makes it the best choice for implementing smart power IPs.

Quoting the description of this technology by STMicroelectronics:

"BCD is a family of silicon processes, each of which combines the strengths of three different process technologies onto a single chip: Bipolar for precise analog functions, CMOS (Complementary Metal Oxide Semiconductor) for digital design and DMOS (Double Diffused Metal Oxide Semiconductor) for power and high-voltage elements."

In 2011 the BCD was honoured with the IEEE Milestone for *"Multiple Silicon Technologies on a Chip"* achievement.

ST offers a unique range of BCD process technologies, each addressing specific application needs, with an optimal trade-off between functionality, performance and cost. There are currently two types of BCD process:

- **High-voltage BCD**, which enables reliable coexistence on the same chip of low-voltage control circuits and very high-voltage DMOS stages with typical voltage capability up to 800V. The integration of BCD on SOI (Silicon on Insulator) substrates addresses specific high-value applications in electromedical, automotive safety or audio
- **High-density BCD** is driven by the need to integrate more and more complex

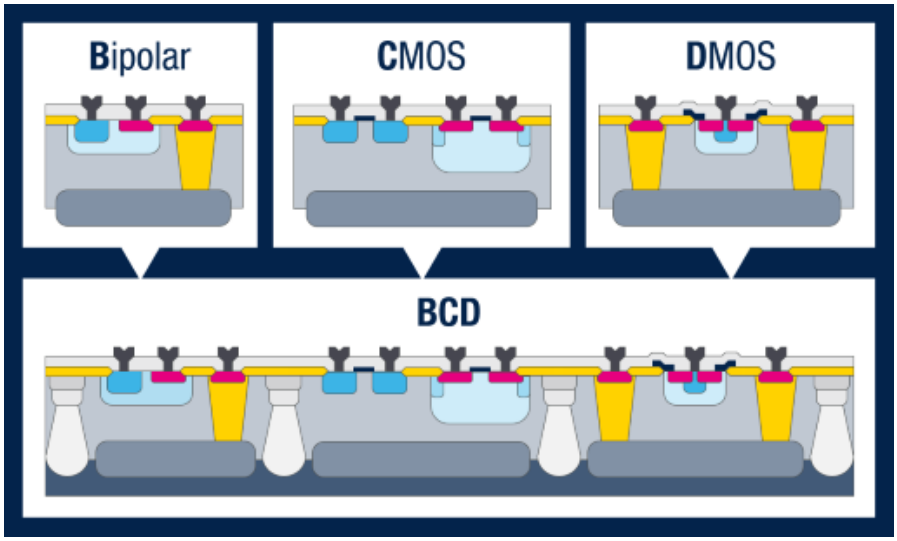


Figure 1.2: The BCD process allow to build on the same substrate the three main technology used for digital, analog and power application: CMOS, BJT and DMOS

and diversified functions on the same chip and to guarantee high quality and reliability in all types of application environments.

The BCD technology can integrate complex CMOS macros, such as complete microcontrollers or nonvolatile memories. This enables the development of systems with low voltages that can be connected to circuits operating at much higher voltages. This grants the possibility to develop systems with a high level of functionality and versatility.

The process has three fundamental properties in the design phase: regularity, modularity, and locality. These enable the design of complex integrated circuits (IC) using blocks from different libraries, as in a standard cell approach.

In a hierarchical design, the system is divided into submodules. This, in general, does not guarantee the minimum number of components. A system may be divided into too many different submodules. The *regularity* allows dividing the system into similar sub-blocks. This offers advantages in the verification phase since the number of blocks to be verified is reduced.

The *modularity* leads to the definition of blocks with exact function and interface (e.g. port name or electrical and timing properties). A modular design is easier to link and considering the blocks as a black box reduces the complexity of the design.

Finally, the *locality* ensures that most connections are between adjacent modules, avoiding long interconnections. This last point is extremely important to avoid excessive delays in connections. Time-critical operations should be performed locally without the need to access distant modules or signals. If necessary, this problem can be solved in large system architectures by logical replication.

1.3 STI^2GaN

Solutions based on wide bandgap semiconductors are fundamental to the development of highly efficient next-generation high-efficiency power systems. These solutions can accelerate the adoption of electric vehicles and renewable energy generation systems.

Gallium Nitride (GaN)-based products, as described in 2.6.1, deliver improved energy efficiency and enable more compact power supply designs for many applications. However, several challenges must be overcome when implementing GaN transistors, including complex gate drive design and PCB layout constraints. BCD Smart Power takes advantage of these bandgap technologies, not only GaN but also silicon carbide (SiC).

STI^2GaN is a family of products that enables designers to get the most out of GaN technology by offering higher levels of integration and performance. By combining the advantages of GaN technology with traditional semiconductor materials, a wide range of power applications will benefit in terms of size, performance, and cost.

This includes applications such as wireless chargers, 48/12V bidirectional DC-DC converters, LiDAR, On-Board Chargers and power supplies.

1.3.1 Benefits of STI^2GaN

ST Intelligent and Integrated GaN takes advantage of GaN technology, such as higher efficiency and higher operating frequency. This leads to lower cooling requirements and smaller heat sinks in end final applications. Integration leads to increased robustness and improved reliability in an extremely compact design. There are two possible levels of integration, shown in figure 1.4.

Figure 1.3: Examples of applications that benefit from the use of GaN; STi^2GaN, a family of products that attempts to embed both traditional Silicon circuits and GaN power devices in the same package [Courtesy of STMicroelectronics]

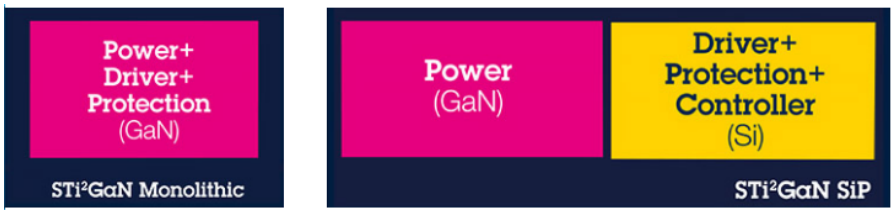


Figure 1.4: Differences in the two level of integration: the monolithic solution is a more compact solution but do not allow the integration of complex control circuit which are made possible in the SiP solution thanks to the possibility to integrate the two systems in different dies. [Courtesy of STMicroelectronics]

- Monolithic solution: integrates the complete system on the same die, implementing together the power devices with gate driver and protection circuits.
- System in Package (SiP): integrates, not on the same die, but on the same package both the GaN power devices and the control circuit. This allow to use the classic BCD process to realize a more complete circuit with sophisticated control circuit, all embedded in an optimized bond-wire-free package.

The innovative bond-wire-free packages, figure 1.5 allows to implement System-in-Package solutions enabling high frequency applications with an overall compact and cheaper system. Furthermore, these type of packages are characterized by a drastic reduction of parasitic elements, leading to lower Electromagnetic emissions (EMI) and the design of a cooler system thanks to the double-sided cooling capabilities: this simplify further the choice of thermal dissipation strategy.

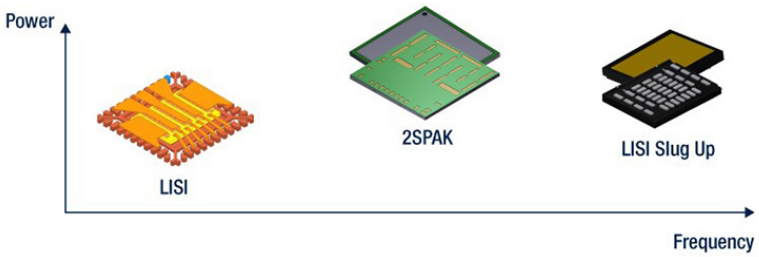


Figure 1.5: types of packages that allow the System-in-Package integration of boh traditional silicon elements and GaN power devices. [Courtesy of STMicroelectronics]

Chapter 2

Power Converters Overview

2.1 Introduction

Power electronics deals with the efficient conversion of electrical power using electronic devices. The key element is the *switching converter* shown in Figure 2.1, which usually has an input power terminal that is processed according to the control input signal, resulting in output power at the corresponding output power terminal. A crucial parameter for the power converter is the efficiency, which is defined as in 2.1.

$$\eta = \frac{P_{out}}{P_{in}} \tag{2.1}$$

The converters are designed in such a way to have high efficiency that is required not for an economic reason or conserve energy, but because producing low-efficiency converters that deal with a high amount of power is impractical, as will be explained in the following.

A converter can perform one of several basic functions: in a *dc-dc converter* the DC input voltage is converted to a DC output voltage with a different magnitude, possibly with opposite polarity or with isolation of the input and output ground reference. In a *ac-dc rectifier* an ac input voltage is rectified, producing a DC voltage, the DC output voltage and the ac input current waveform can be controlled. The inverse process, *dc-ac inversion* transforms a dc input voltage into an ac output voltage with desired magnitude and frequency. Finally, the *ac-ac cycloconverter*

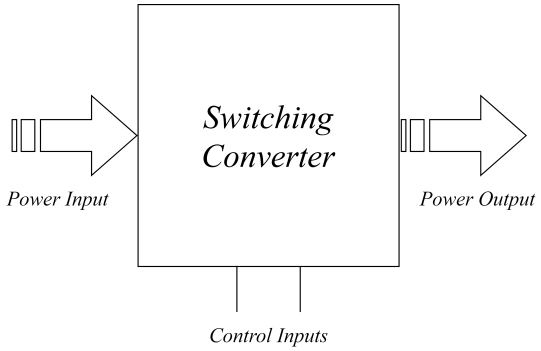


Figure 2.1: Abstract representation of a Switching converter

converts an ac input voltage to a given ac output voltage of controllable magnitude and frequency. In general, a control circuit is required so to have a well-regulated output voltage in the presence of some variation in the input voltage and load current, figure 2.2 depicts the block diagram of a switching converter with a control circuit.

The reason why converters requires a high efficiency can be understood introducing the power loss $P_{loss} = Pin - P_{out}$ which can be related to the efficiency, hence to the output power, as follows

$$Q = \frac{P_{out}}{P_{loss}} = \frac{P_{out}}{P_{out} - P_{in}} = \frac{\eta}{1 - \eta} \tag{2.2}$$

The quantity Q, called *Quality factor*, is a fundamental measure of the quality of the converter: all the P_{loss} is converted into heat and must be removed by using a cooling system: the higher the factor Q, smaller the power loss. In most applications, the maximum output power is limited by the capacity of the cooling system to remove the heat thus if the loss of power is substantial, a large cooling system may be required and the circuit sub-element may work at a higher temperature leading also to reliability problems. Increasing efficiency is the key to obtaining higher output powers with lower power losses that allow the system to be integrated with a smaller area with a small cooling system. To this aim semiconductor devices operating in switched-mode are being used and newer technology, both for the type of controller and the type of semiconductor used, plays a role in the design of high-efficiency converters: namely digital control loop and Gallium Nitride (GaN).

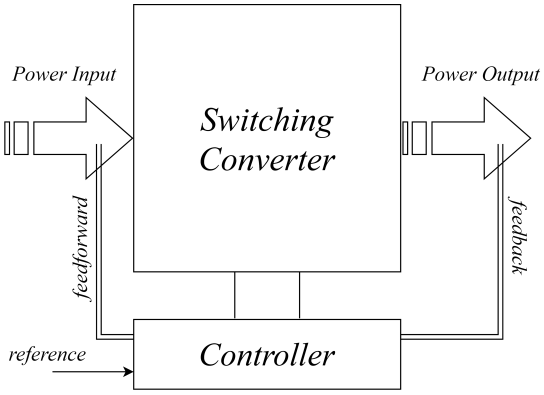


Figure 2.2: Switching converter with a controller

The increasing demand for reconfigurability in the control loop of power converters lead to the development of digital control loops, which also allow reducing the overall system costs since they do not require additional bulky elements and pins (required by the compensation network in classic analog control loops). The only drawback is the fact that to obtain a high resolution of the control signal can be necessary to use an unfeasible clock in the digital control loop.

2.2 Converters Topology

To realize the final converter, conventional circuit elements may be used: resistive, capacitive, and magnetic elements, semiconductor devices, both in linear and switched mode. The magnetic elements, generally avoided in signal processing due to their encumbrance and difficulty to be used in integrated circuits, are important elements of the switching converters since ideally do not consume power, meanwhile, the dissipative resistive elements are avoided. The semiconductor devices, when used in switched-mode, are such that in the on state their voltage drop is zero, meanwhile, in the off state their current is zero, hence the dissipated power in both states is ideally null.

The working principle of a switching converter is to use two power MOS to alternatively connect the output to the ground and the power supply, then the average of the obtained square wave will be the wanted output power, in figure 2.3 the principle scheme of a switching converter is represented. The following sections

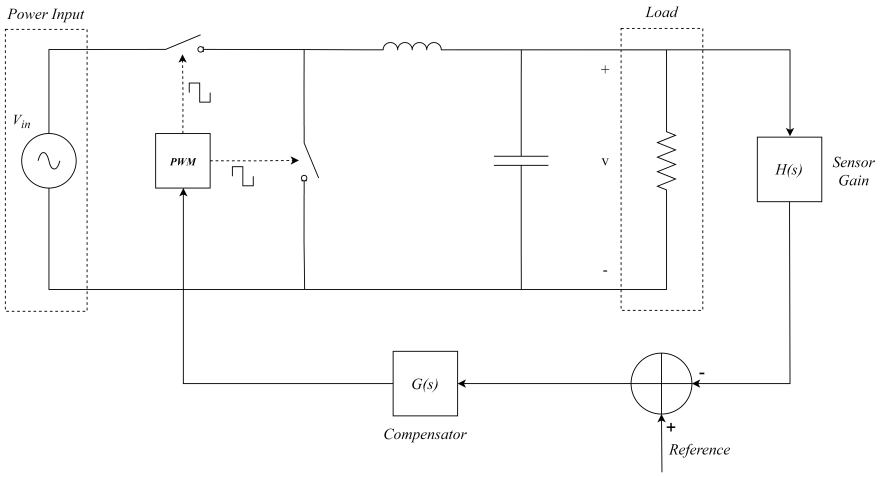


Figure 2.3: General scheme of a switching converter, the feedback loop drives the PWM unit to properly adjust the duty cycle of the output signal, hence the output power.

introduce the three main topologies for a switching converter: buck, boost, and buck-boost converters.

2.2.1 Buck Converter

In this section, a dc-dc buck converter will be considered as a driving example to describe the various techniques and approximations used to analyze a converter.

The topology of a buck converter is depicted in figure 2.4. The switches produce a rectangular waveform $v_s(t)$ as illustrated in figure 2.5.

The switching frequency $f_s = T_s^{-1}$ generally ranges from 1 kHz to 1 MHz, depending on the switching speed of the semiconductor device; the duty cycle D is the fraction of time that the switch spends in position 1 and is a number between zero and one.

Ideal Analysis

Considering neglecting the nonidealities of the used components, namely, they are considered non-dissipative, hence efficiencies approaching 100% can be obtained.

The switch output voltage $v_s(t)$ has a DC component that is less than the converter DC input voltage V_g, which value can be obtained from the DC term of

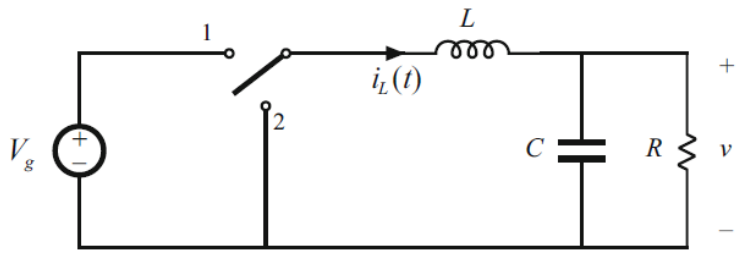


Figure 2.4: Abstract scheme of a buck converter, the switch can be integrated with two switches an *high side* that connects input power to output which can be implemented by means of a power MOSFET, and a *low side* that connects output to ground implemented by means of a power diode or, again, a power MOSFET [2]

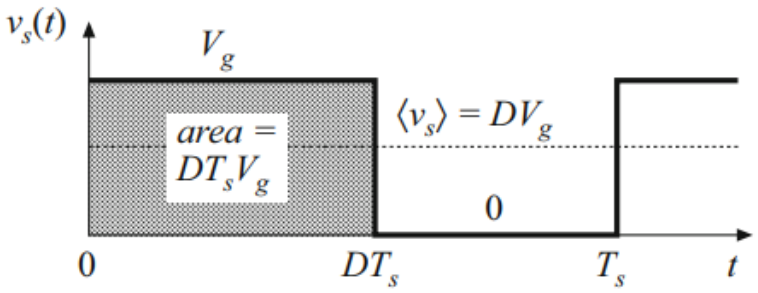


Figure 2.5: switch output voltage $v_s(t)$, when the switch is in position 1 is equal to the input voltage, ground when the switch is in position 2 [2]

the Fourier expansion of the output voltage

$$< v_s > = \frac{1}{T_s} \int_0^{T_s} v_s(t)\, dt \tag{2.3}$$

The integral is given by the area under the curve, DV_gT_s, hence

$$< v_s > = \frac{1}{T_s}(DV_gT_s) = DV_g \tag{2.4}$$

From the equation 2.4 it can be easily understood that a buck converter reduces the input power by a factor equal to the duty cycle of the switch output voltage, for this reason, this converter is also called a *step down* converter.

The low-pass filter, implemented by means of an LC cell, is designed not only to

get the DC component of the switch output signal but also to reject the harmonic component at the switching frequency. To accomplish this, the filter is designed to have a cutoff frequency much lower than the switching frequency, so that in a first approximation the output voltage is equal to the DC component of $v_s(t)$:

$$v \approx < v_s > = DV_g \tag{2.5}$$

The control law curve of the buck converter is depicted in figure 2.6

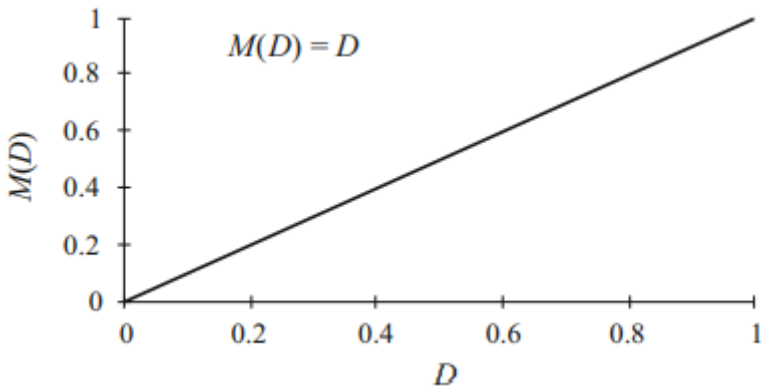


Figure 2.6: Buck converter dc output voltage as a function of the duty cycle D, in a first approximation is a linear curve [2]

Inductor Volt-Second Balance, Capacitor amp-second balance and small ripple approximation

It is impossible to build a perfect low pass filter that allows the DC component to pass whether the components at the switching frequency are completely removed: the actual waveform is more like the one depicted in figure 2.7 so that the actual output signal can be written as in the equation 2.6.

$$v(t) = V + v_{ripple}(t) \tag{2.6}$$

In a well designed converter, since the goal is to produce a DC output, the ripple is normally required to be less than 1% of the final DC component, so in general

should be valid the approximation stated in the equation 2.7

$$||v_{ripple}(t)|| \ll V \tag{2.7}$$

hence the output voltage will still be well approximated by its DC component only.

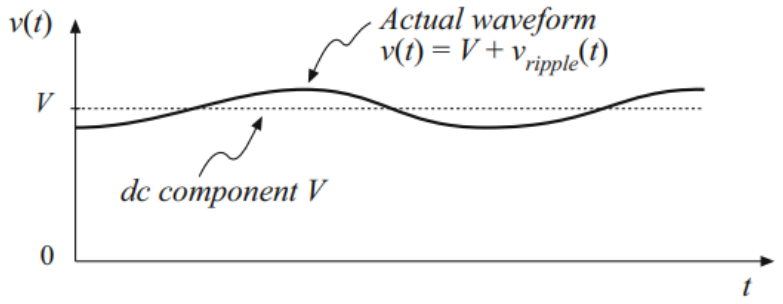


Figure 2.7: Output voltage waveform $v(t)$, consisting of a DC component V and a switching ripple $v_{ripple}(t)$ [2]

This is known as *Small ripple approximation* or *linear ripple approximation* and simplifies the analysis in such a way that the damped sinusoidal expression for the inductor and capacitor could be considered as a simpler linear waveform. The approximation is well verified until the switching period is much shorter than the natural time constant of the circuit and can only be applied to a continuous waveform, not discontinuous as the switch voltage or the inductor voltage.

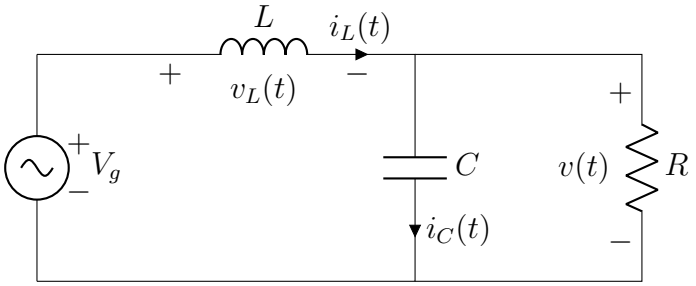


Figure 2.8: Equivalent circuit of the converter with the switch in position 1

Let us analyze the inductor current waveform; when the switch is in position one, the left side of the inductor is connected to the input voltage. In this situation the circuit can be analyzed considering the equivalent one depicted in figure 2.9,

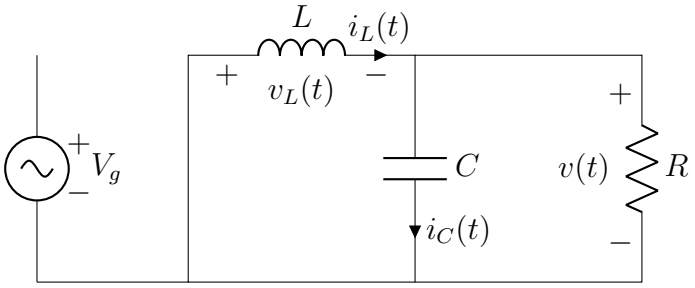


Figure 2.9: Equivalent circuit of the converter with the switch in position 2

the inductor voltage can be written as stated in equation 2.8

$$v_L \;=\; V_g - v(t) \approx V_g - V \tag{2.8}$$

where the approximation is valid thanks to the small ripple approximation, so with the switch in position one the inductor voltage can be considered constant, and thanks to the usual inductor constitutive relation is possible to derive the inductor current:

$$v_L(t) \;=\; L\frac{di_L(t)}{dt} \;\Rightarrow\; \frac{di_L(t)}{dt} \;=\; \frac{V_g - V}{L} \tag{2.9}$$

from the relation 2.9 is clear that the current slope is constant, so the current in the inductor increases linearly. Similar argument apply when the switch is in position two where the left side of the inductor is set to ground, as described in the relations 2.10.

$$v_L \;=\; 0 - v(t) \approx -V \;\Rightarrow\; \frac{di_L(t)}{dt} \;=\; \frac{-V}{L} \tag{2.10}$$

In this case the current decrease linearly in the period of time where the switch is in position 2 as depicted In Figure 2.10.

The current ripple can be then easly evaluated considering that:

$$(change\, in\, i_L) \;=\; (slope)\; x\; (length\, of\, the\, subinterval)$$

$$2\Delta i_L \;=\; (\frac{V_g - V}{L})(DT_s) \tag{2.11}$$

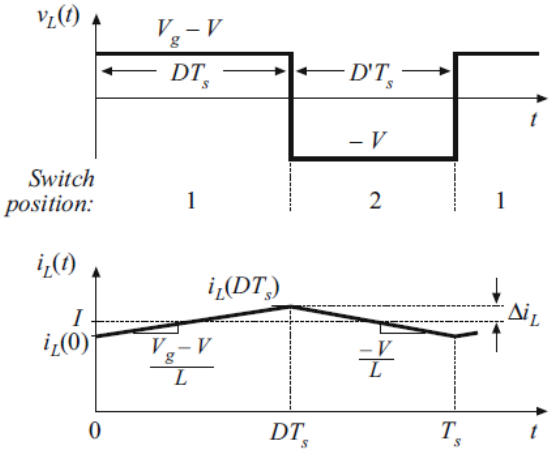


Figure 2.10: inductor current and voltage trend during the turn on and off transient of the switch. [2]

Solving for Δi_L yields:

$$\Delta i_L = \frac{V_g - V}{2L} DT_s \tag{2.12}$$

acceptable value are of about $10 - 20\%$ of the full current swing of the load and is not acceptable to have a too large current ripple, otherwise too large peak of current for inductors and semiconductor devices are obtained, the small ripple approximation is then also justified for the inductor current and equation 2.12 can be used to find the value of L to get a certain current ripple and is the equation commonly used to select the value of the inductance for the buck converter.

$$L = \frac{V_g - V}{2\Delta i_L} DT_s \tag{2.13}$$

Although it is possible to solve converters exactly (e.g. by using the Laplace transform), this requires a lot of work and usually the effort required to write the equations to precisely describe the ripple is a waste of time since this is a small and however undesired: the small ripple approximation is easy to apply and quickly yields simple expressions for the DC components of the converter waveforms.

Consider now what happens to the inductor current when the converter is first turned on. Starting from a condition in which both inductor current and output voltage are zero, in the instant in which the input voltage V_g is applied: from

zero the current in the first sub interval rises as expressed in equation 2.9 with V initially zero, due to that when the switch moves in position two and the current should drop, with a slope expressed in equation 2.10, since V is still zero this slope is zero so the current remains constant. It can be seen that there's a net increase in the inductor current since $i_L(T_s) > i_L(0)$.

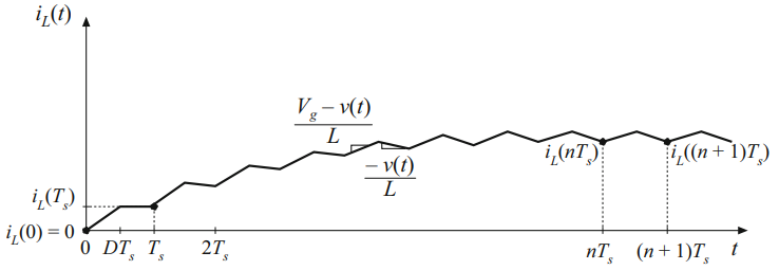


Figure 2.11: Inductor current waveform during converter turn-on transient [2]

The output voltage V increases up to a point where the increase of current in the first subinterval is equal to the decrease in the second subinterval, so when there's no net change in the inductor current over a complete switching period and from now on the converter operate in steady-state and the behavior of the inductor current is now periodic, the overall waveform is depicted in figure 2.11.

This requirement allows to find a steady-state condition, valid for any switching converter, called the *Inductor volt-second balance*, that can be formally expressed as follows.

Starting from the inductor constitutive relation and integrating over a complete switching period the equation 2.14 (this also describe the fact that the change in the inductor current over a complete switching period is proportional, through the inverse of the inductance, to the integral of the applied voltage over the same period)

$$v_L(t) = L\frac{di_l(t)}{dt} \Rightarrow i_L(T_s) - i_L(0) = \frac{1}{L}\int_0^{T_s} v_L(t)dt \tag{2.14}$$

since in steady state the current change is zero, equation 2.15 can be obtained, where the right hand side has the unit of volt-seconds, and this states that the total area under the $v_L(t)$ waveform is zero.

$$0 = \int_0^{T_s} v_L(t)dt \tag{2.15}$$

An equivalent expression could be found by dividing both side by T_s obtaining on the right hand side the average value of the inductor voltage, which means that at equilibrium the applied inductor voltage has zero DC component.

$$0 = \frac{1}{T_s}\int_0^{T_s} v_L(t)dt \tag{2.16}$$

This is the principle of the inductor volt second balance that allows us to retrieve the same relation expressed in 2.5.

Expressing the total area under the $v_L(t)$ curve as λ and knowing that the area under the curve divided by T_s is the average value, relation 2.16 can be used to find the final control law relation, as expressed in the following.

$$\lambda = \int_0^{T_s} v_L(t)dt = (V_g - V)DT_s + (-V)(1-D)T_s$$

$$\langle v_L \rangle = \frac{\lambda}{T_s} = (V_g - V)D + (-V)(1-D)$$
$$0 = V_g D - VD + VD - V \Rightarrow V = DV_g$$

The advantage of this approach is that it can be applied to any converter.

Similar argument are applied to the capacitor, leading to the *capacitor amp-second balance*. Starting from the capacitor constitutive relation and integrating over one switching period, knowing that in steady state the net change in the capacitor voltage must be zero, the relation in 2.17 express in a formal way the capacitor amp-second balance, by which the average value of the capacitor current must be zero.

$$i_C(t) = C\frac{dv_C(t)}{dt} \Rightarrow v_C(T_s) - v_C(0) = \frac{1}{C}\int_0^{T_s} i_C(t)dt$$

$$0 = \int_0^{T_s} i_C(t)dt$$

$$\langle i_C \rangle = \frac{1}{T_s}\int_0^{T_s} i_C(t)dt = 0 \tag{2.17}$$

This principle allow to find the steady state current in a converter.

Finally, it is useful to evaluate the capacitor voltage ripple Δv_C, shown in figure

2.10.

Considering the buck converter in figure 2.4, the inductor current $i_L(t)$ has a DC component I and a ripple whose peak amplitude is Δi_L. The DC component will flow entirely through the resistance, since the capacitor behaves like an open circuit (neglecting its ESR), while there's a partition of the AC switching ripple between the resistance and the capacitance. If the converter is well designed so that the capacitor provides significant filtering of the switching ripple, the capacitance C is large enough so that its impedance at switching frequency is sufficiently small, and practically all the ripple flow through the capacitor, hence the current waveform for capacitor and inductor are equal, apart from the absence of a DC value in the former.

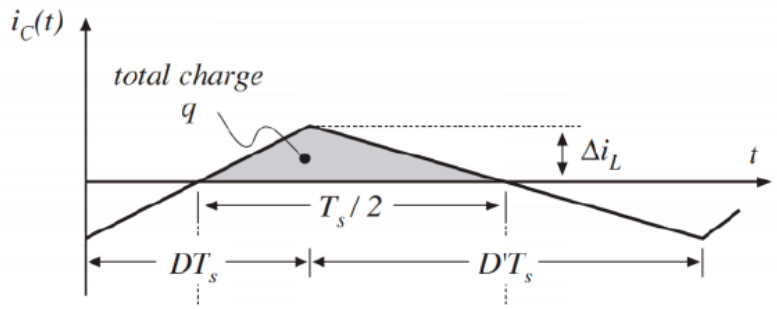


Figure 2.12: The capacitor current waveform is equal to the inductor one, aside from the absence of the DC value [2]

When the capacitor current $i_C(t)$ is positive, charge will deposit on the capacitor plate and its voltage $v_C(t)$ will increase. Therefore, between the two zero crossing of the capacitor current waveform, its voltage changes between minimum and maximum values and since the waveform is symmetrical the total voltage ripple will be $2\Delta v$. The change in voltage can be related to the total charge stored in the capacitor, knowing the capacitor relation $Q = CV$:

$$q = C(2\Delta v) \tag{2.18}$$

The charge can also be evaluated by means of the integral of the current between the two zero crossing point, namely the shaded area in figure 2.12, which are positioned at DT_s and $(1 - D)T_s$, so the base of the triangle will be $T_s/2$, hence

the total charge:

$$q = \frac{1}{2}(2\Delta v)\frac{T_s}{2} \tag{2.19}$$

substituting 2.18 in 2.19 the final expression for the capacitor voltage ripple is obtained:

$$\Delta v_C = \frac{\Delta i_L T_s}{8C} \tag{2.20}$$

Equations 2.12 and 2.20 can be used to set the proper value of inductance and capacitance to fix the ripples at the desired values.

2.3 Boost Converter

The Boost converter, figure 2.13 is another well-known switch-mode converter topology capable of producing a dc output higher than the input dc voltage, for this reason, is also called a *step up* converter. Let us apply the small-ripple approximation and the principles of inductor volt-second balance and capacitor charge balance to find the steady-state output voltage and inductor current for this converter.

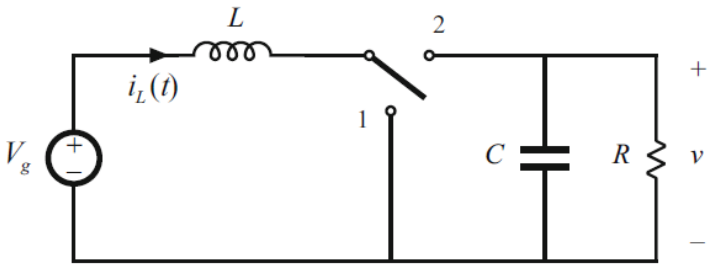


Figure 2.13: principle scheme of a boost converter. [2]

With the switch in position 1 the inductor falls at ground in parallel to the input, whereas the capacitor will be in parallel to the load. In this case the inductor voltage and capacitor current in this sub interval are given by

$$v_L = V_g \quad i_C = -\frac{v}{R} \tag{2.21}$$

The capacitor current can be expressed only considering the DC value V of the output voltage using the small ripple approximation, so that $v \approx V$.

with the switch in position 2, the inductor will be now connected to the output. The inductor voltage and capacitor current will be given by:

$$v_L = V_g - v \quad i_C = i_C - \frac{v}{R} \tag{2.22}$$

Again, the equation can be simplified using the small ripple approximation.

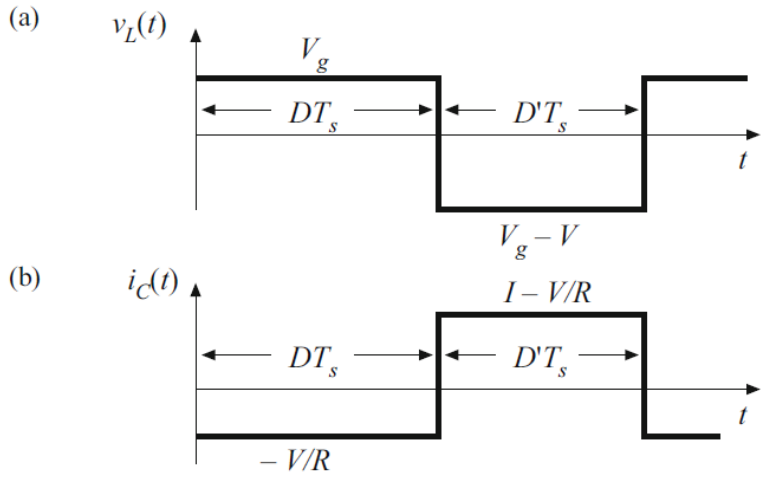


Figure 2.14: Boost converter voltage and current waveform [2]

During the first sub interval, $v_L(t)$ is equal to the dc input voltage V_g, and positive volt-seconds are applied to the inductor. Since, in steady-state, the total volt-seconds applied over one switching period must be zero, negative volt-seconds must be applied during the second sub interval.

Therefore, the inductor voltage during the second sub interval, $(V_g - V)$, must be negative. Hence, V is greater than V_g. Namely, considering figure 2.14, the total volt-second applied to the inductor voltage on a complete switching period is:

$$\int_0^{T_s} v_L(t)dt = (V_g)DT_s + (V_g - V)(1 - D)T_s = 0 \tag{2.23}$$

After some algebraic manipulation, is possible to express the output dc voltage as

follows:

$$V = \frac{V_g}{1 - D} \tag{2.24}$$

and in particular the conversion ratio M(D), since D is lower than 1, will be higher than one, as is shown in figure 2.15, hence the boosting capability of the converter.

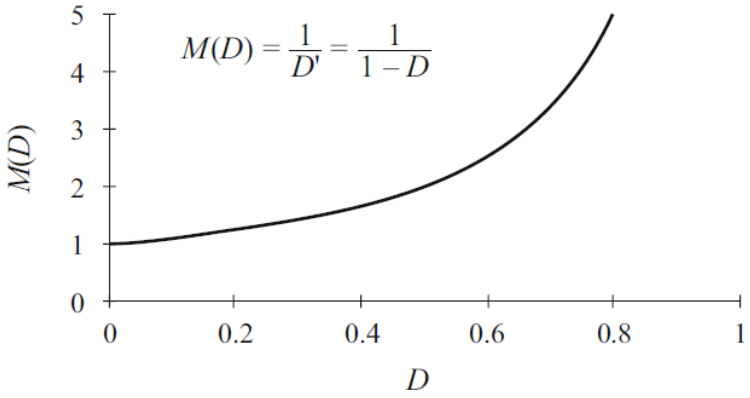


Figure 2.15: Conversion ration for a boost converter [2]

At $D = 0, V = Vg$. The output voltage increases as D increases, and in the ideal case tends to infinity as D tends to 1. So the ideal boost converter is capable of producing any output voltage greater than the input voltage. There are, of course, limits to the output voltage that can be produced by a practical boost converter.

The dc component of the inductor current is derived using the principle of capacitor charge balance. During the first sub-interval, the capacitor supplies the load current, and the capacitor is partially discharged. During the second sub-interval, the inductor current supplies the load and, additionally, recharges the capacitor. The net change in capacitor charge over one switching period is found by integrating the $i_C(t)$ waveform, in figure 2.14:

$$\int_0^{T_s} i_C(t)dt = -\frac{V}{R}DT_s + (I - \frac{V}{R})(1 - D)T_s = 0 \tag{2.25}$$

solving for I and substituting 2.24 yields to the final expression of the dc input current in the boost converter, which is greater than the load current.

$$I = \frac{V_g}{(1 - D)^2 R} \tag{2.26}$$

Since the converter output voltage is greater than the input voltage, the input current must likewise be greater than the output current. In practice, the inductor current flows through the semiconductor forward voltage drops, the inductor winding resistance, and other sources of power loss. As the duty cycle approaches one, the inductor current becomes very large and these component nonidealities lead to large power losses. In consequence, the efficiency of the boost converter decreases rapidly at a high duty cycle.

Finally, as in the buck converter, is useful to estimate the value of the ripple of the inductor current and the capacitor voltage. The capacitor current waveform $i_C(t)$ is given in figure 2.14. During the first subinterval, the slope of the capacitor voltage waveform v(t) is

$$\frac{dv_C(t)}{dt} = \frac{i_C(t)}{C} = -\frac{V}{RC} \tag{2.27}$$

Whereas in the second subinterval, is

$$\frac{dv_C(t)}{dt} = \frac{i_C(t)}{C} = \frac{I}{C} - \frac{V}{RC} \tag{2.28}$$

Considering the voltage waveform in figure **??**, is clear that the slope multiplied by the length of the first subinterval is equal to $-2\Delta v$

$$-2\Delta v = -\frac{V}{RC} DT_s \Rightarrow \Delta v = \frac{V}{2RC} DT_s \tag{2.29}$$

As in the buck converter, the expression can be used to select the capacitor value C to get the wanted ripple peak magnitude.

2.4 Buck-Boost Converter

The last example of converter topology is the buck-boost converter: as the name suggest it can either increase or decrease the output voltage, based on the duty cycle D factor. There are two possible topologies: an inverting and a non-inverting. Starting from the inverting, figure 2.16, applying the small ripple approximation to obtain the inductor voltage and capacitor current waveforms, figure 2.17 and the volt-second balance is possible to observe that in the first subinterval (when the

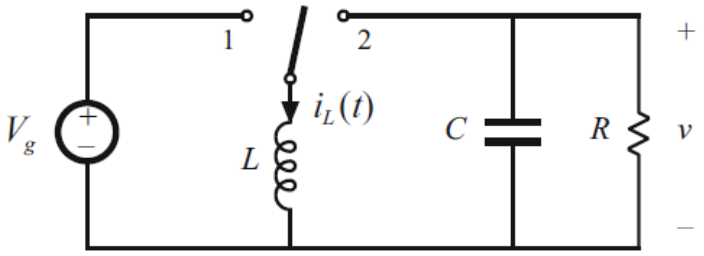


Figure 2.16: Buck Boost converter inverting topology [2]

switch is in position 1) the inductor voltage is positive, meanwhile, in the second subinterval a negative output voltage is obtained since in the balance the two areas under the curve must be equal to zero yielding to a negative value of V_out, hence the inverting property of this topology. Then considering the inductor current and

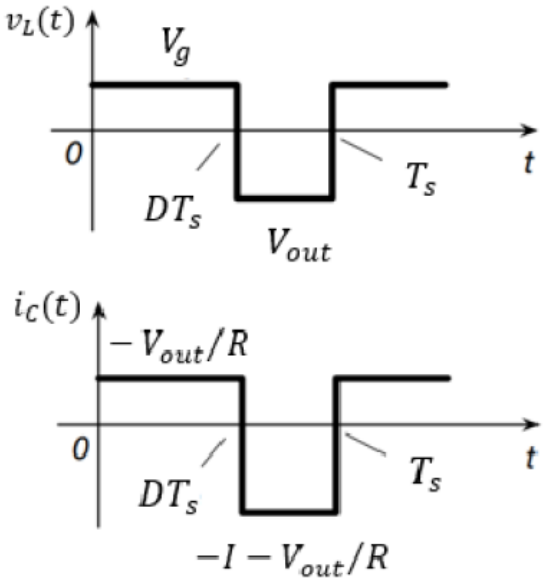


Figure 2.17: Current and voltage waveform for the non inverting case [3]

capacitor voltage, and performing the same analysis described for the previous converters, the conversion ratio and the current and voltage ripple can be obtained:

$$M(D) = \frac{V_{out}}{V_g} = -\frac{D}{1-D} \tag{2.30}$$

$$\Delta i_L = \frac{V_g}{2L} D T_s \tag{2.31}$$

$$\Delta v_C = -\frac{V_o ut}{2RC} D T_s \tag{2.32}$$

It can be noted that if $D < 0.5$ the conversion ratio is less than zero, while it is higher in the other case, hence the capability of a buck-boost converter to increase or reduce the input (for this reason, although more complex, the converter is more flexible than the other solutions). Similar results can be obtained for the

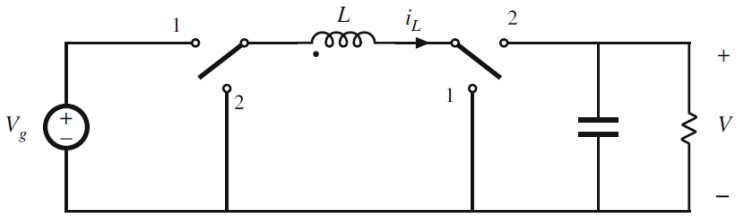


Figure 2.18: Buck Boost converter non inverting topology, it can be seen as the cascade of a buck and a boost converter. [2]

non-inverting topology, figure 2.18, which conversion ratio is

$$M(D) = \frac{V_{out}}{V_g} = \frac{D}{1-D} \tag{2.33}$$

The structure, which can be seen as the cascade of a buck and a boost converter, is however more complex since it requires more switching element than in the inverting case.

2.5 Basic topology comparisons

The three introduced topology do not differ only for the conversion ratio, which value are summarized in table 2.1. The main buck converter drawback is that its

Topology	Conversion Ratio
Buck	D
Boost	$\frac{1}{1-D}$
Buck-Boost (non-inverting)	$\frac{D}{1-D}$
Buck-Boost (inverting)	$-\frac{D}{1-D}$

Table 2.1: Expression of the conversion ratio for the three main topology of power converter

input current is pulsating: the switch alternately connects/disconnects the source to/from the rest of the circuit and makes the input current discontinuous.

This can give rise to Electromagnetic Interference (EMI), which is a disturbance that can interfere with the surrounding circuits. For this reason, an input filter could be needed in a buck converter to attenuate EMI.

On the other hand, the boost converter has a continuous input current, but it can be disadvantageous in terms of efficiency and area occupation. To understand this, let us consider the boost converter shown in Figure 2.13 : in the ideal case, since the output voltage is greater than the input voltage, because of energy conservation the input current must be greater than the output current.

The input current flows through the inductor and through the transistor/diode implementing the switch: since the input current can reach very high values, these elements are dimensioned with larger sizes and dissipate more power than in the buck converter case, where the input current assumes lower values. Moreover, in the boost converter, the output current is pulsating, which leads to a higher ripple on the output voltage.

Now, let us consider the buck-boost converter: all the disadvantages seen for the buck and the boost converter are present both in the inverting and in the non-inverting configurations shown in Figure 2.18 and Figure 2.16. In fact, both the input and the output current are pulsating and, when the buck-boost is boosting the input, large values can be reached by the current flowing through the inductor and the transistor/diode. In the non-inverting configuration, the buck-boost also has two switches: this further increases both power consumption and area occupation.

To conclude, the buck converter is the best in terms of efficiency and area occupation, but it can only perform down-conversion. When other types of conversion

are needed, the boost or the buck-boost converter must be chosen despite their cons.

2.6 Power Switch

The switch used in a switching power converter can be implemented by means of transistor or diode based on the power processing function to be performed.

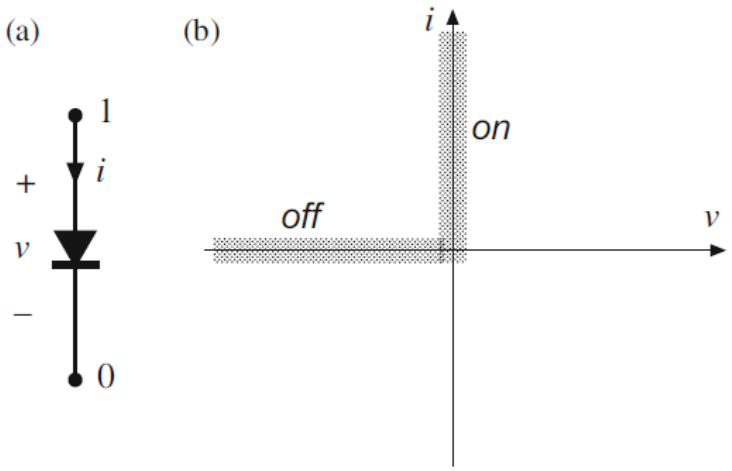


Figure 2.19: (a) symbol of a power diode, it has two terminal, with no control port. (b) the ideal characteristics make it suitable for a SPST switch, given that the operating point lies on it. [2]

In general Semiconductor devices behaves as single-pole single-throw (SPST) hence the realization of single-pole double-throw (SPDT) using two SPST devices is not trivial: there can be problem related to the fact that both device can be simultaneously on or off, something not predicted by the classic SPDT model.

Also the switch state can depend on the applied current or voltage waveform. These problem are common especially in circuit that works with light load (or occasionally even with heavy load) leading to the discontinuous mode of operation.

How an ideal switch can be implemented using classic semiconductor device depends on the polarity of the voltage that the device has to block in the off state and the polarity of the current that it has to conduct in the on state (eg. in the case of a buck converter the switch has to block the generator voltage in the off

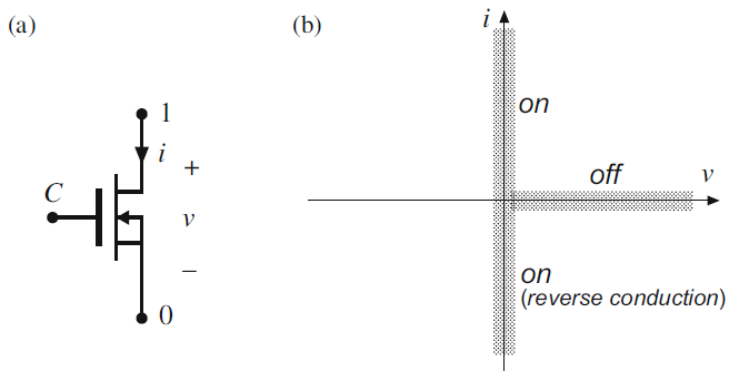


Figure 2.20: (a) symbol of a power MOSFETs, it has three terminal, the gate is the control port. (b) the ideal characteristics make it suitable for a SPST switch, given that the operating point lies on it. The reverse conduction condition is never exploited in power application [2]

state, and has to conduct the inductor current in the on state).

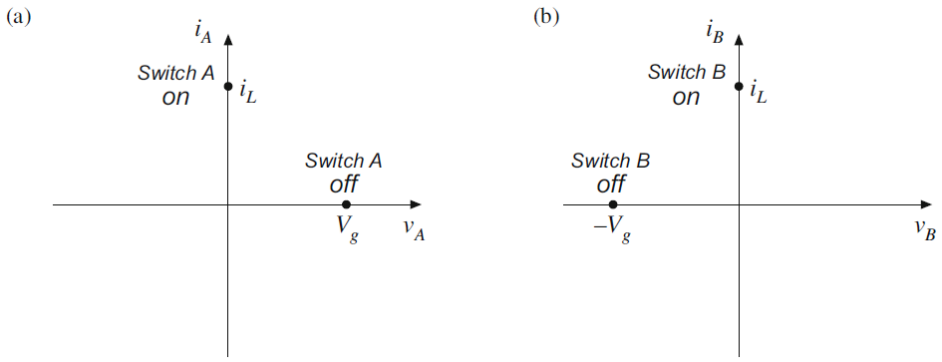


Figure 2.21: Operating point position for a buck converter implemented by a power mosfet and diode [2]

If the polarity of voltage and current are in a single quadrant a common diode or transistor can be used, different arrangement of the same devices allow to realize also double or quadruple quadrant switches.

Focusing on single quadrant switches, in figure 2.19 the symbol and ideal characteristics of a power diode are depicted. This type of SPST contain two power terminal, denoted by 1 and 0, with no control terminal.

The diode behaves in such a way that when the applied voltage is negative the

current is almost zero hence the device is off. When the device is on the $i > 0$ with voltage equal to 0: this type of switch can block negative voltage and can implement a SPST provided that the intended operating points lie on the described characteristics.

Considering the implementation of a SPST with a classic MOSFET, it has two power terminal and a control terminal which can allow or not conduction through the device, as depicted in figure 2.20. In the case of the MOSFETs its possible to consider also a reverse conduction condition, which is actually never exploited in case of power application. Figure 2.21 summarize where the operating point of the two type of switches lies on the i vs v plane for a buck converter.

2.6.1 Gallium Nitride - GaN

For over three decades, power management efficiency and cost have improved steadily as innovations in power metal oxide silicon field effect transistor (MOSFET) structures, technology, and circuit topologies have kept pace with the growing need for electrical power in our daily lives. In the new millennium, however, the rate of improvement has slowed as the silicon power MOSFET asymptotically approaches its theoretical bounds.

Gallium nitride (GaN) high electron mobility transistor (HEMT) devices first appeared in about 2004 with depletion-mode radio frequency (RF) transistors, Acceptance outside of this application, however, has been limited by device cost as well as the inconvenience of depletion-mode operation (normally conducting and requires a negative voltage on the gate to turn the device off).

Silicon has been a dominant material for power management since the late 1950s. The advantages that silicon had over earlier semiconductors, such as germanium or selenium, could be expressed in four key categories: silicon enabled new applications not possible with earlier materials, proved more reliable, easier to use in many ways, and cheaper. All of these advantages stemmed from the basic physical properties of silicon, combined with a huge investment in manufacturing infrastructure and engineering.

One way of translating these basic crystal parameters into a comparison of device performance is to calculate the best theoretical performance achievable for each of the three candidates. For power devices, there are many characteristics

that matter in the variety of power conversion systems available today. Five of the most important are: conduction efficiency (on resistance), breakdown voltage, size, switching efficiency, and cost.

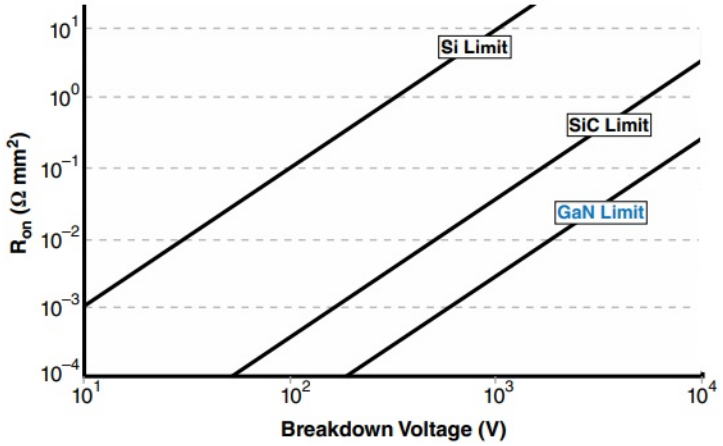


Figure 2.22: theoretical R_{on} limit vs. blocking capacitance for the three main techonology used today for power devices. Is possible to appreciate how the GaN material allow to realize device with a much lower on resistance than the conventional silicon devices.

The band gap of a semiconductor is related to the strength of the chemical bonds between the atoms in the lattice. These stronger bonds mean that it is harder for an electron to jump from one site to the next. Among the many consequences are lower intrinsic leakage currents and higher operating temperatures for higher band gap semiconductors. GaN and SiC both have higher band gaps than silicon.

The stronger chemical bonds that cause the wider band gap also result in a higher critical electric field needed to initiate impact ionization, thus causing avalanche breakdown. The voltage at which a device breaks down can be approximated as in equation 2.34.

$$V_{BR} = \frac{1}{2} W_{drift} E_{crit} \tag{2.34}$$

The breakdown voltage of a device (V_{BR}), therefore, is proportional to the width of the drift region (w_{drift}).

In the case of SiC and GaN, the drift region can be 10 times smaller than in

silicon for the same breakdown voltage. In order to support this electric field, there need to be carriers in the drift region that are depleted away at the point where the device reaches the critical field.

The number of electrons (assuming an N-type semiconductor) between the two terminals can be calculated using Poison's equation, obtaining equation 2.35

$$qN_D = \epsilon_0 \epsilon_r E_{crit} / w_{drift} \tag{2.35}$$

It can be seen that if the critical field of the crystal is 10 times higher, and from Equation 2.34, the electrical terminals can be 10 times closer together. Therefore, the number of electrons, ND, in the drift region can be 100 times greater. This is the basis for the ability of GaN and SiC to outperform silicon in power conversion. To conclude all these results can be summarized in a figure of merit that directly influences the speed of the device, namely the on resistance. In general the theoretical on resistance of these devices can be written as in equation 2.36.

$$R_{on} = \frac{W_{drift}}{q\mu_n N_D} \tag{2.36}$$

the final expression for the on resistance can be obtained by substituting 2.34 and 2.35 in 2.36

$$R_{on} = \frac{4V_{BR}^2}{\epsilon_0 \epsilon_r E_{crit}^3} \tag{2.37}$$

Figure 2.22 shows the on resistance trend for Si, SiC, and GaN. This plot is for an ideal structure. Real semiconductors are not always ideal structures and it is always a challenge to achieve the theoretical limit. In the case of silicon MOSFETs, it took 30 years. From the graph is possible to see how the theoretical limit of GaN devices is much lower, even of the SiC technology: that's why in modern high efficiency power devices is starting to be preferred over silicon (also thanks to the advancement in manufacturing processes that allows to build such a device).

2.7 Discontinuous conduction mode

When the switch are implemented using unidirectional current or voltage semiconductor switches, discontinuous conduction modes (DCM) may occur. It arises when the switching ripple in an inductor current or capacitor voltage is large enough to cause the switch to exit the normal conduction mode (like when the diode goes in reverse conduction). The DCM occurs with large inductor current

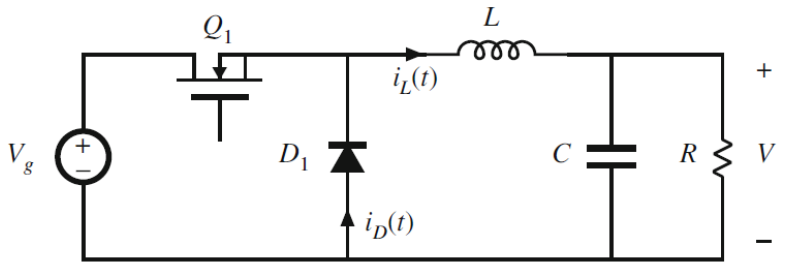


Figure 2.23: Buck converter implemented with real switches [2]

ripple in a converter operating at light load and containing current unidirectional switches. Since it may be required for the converter to operate in such a condition, in particular, with the load removed, DCM is frequently encountered, hence some converter are purposely designed to operate in DCM for all loads. In this condition the converters properties change radically and the conversion ratio M becomes load-dependent.

As a driving example, the buck converter implemented as in figure 2.23 will be considered. In general the inductor current waveform contains a dc component I and a switching ripple of peak amplitude Δi_L, in continuous conduction mode (CCM) the minimum diode current during the second sub interval is $(I - \Delta i_L)$, being the diode a single quadrant switch, to operate in CCM this current must remain positive. The inductor current dc component is equal to the load current

$$I = \frac{V}{R} \tag{2.38}$$

since no current flows in the capacitor, meanwhile the current peak ripple amplitude is

$$\Delta i_L = \frac{V_g - V}{2L} DT_s = \frac{V_g D(1-D)T_s}{2L} \tag{2.39}$$

the ripple magnitude depends only on the applied voltage, the inductance and the transistor conduction time and not on load resistance. Suppose now that the load

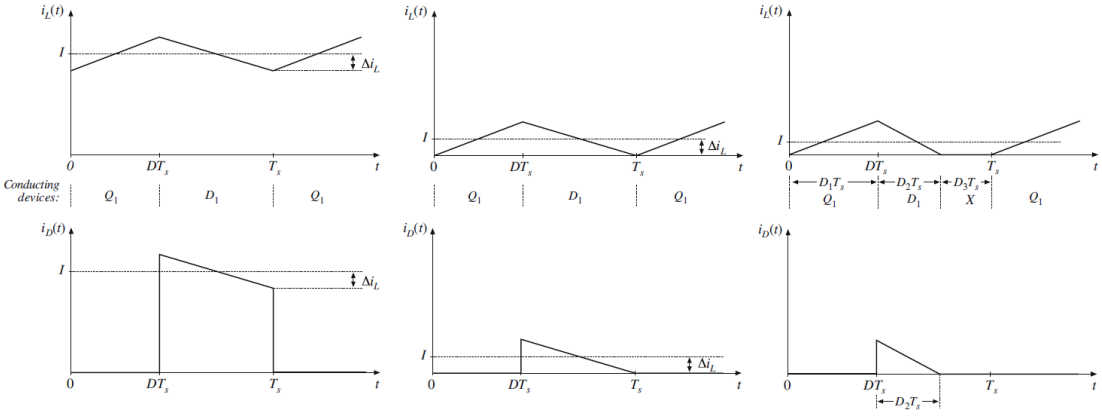


Figure 2.24: Behavior of current and voltage waveform as the load resistance increases [2]

resistance is increased, hence the dc component of the current reduces, there's a value of the load for which the ripple reaches the 0, after that value, the diode current cannot be negative, hence there's a period of time in which the diode is not conducting; this yields to the definition of three sub intervals: the first sub interval D_1T_s in which the transistor conducts, the second sub interval D_2T_s in which the diode conducts and the third sub interval in which neither the diode or the transistor are conducting.

The boundary between DCM and CCM form the current ripple can be evaluated as the value of the dc component for which the current remains positive, that is to say:

$$I < \Delta i_L \quad for \;\; CCM \tag{2.40}$$

$$I > \Delta i_L \quad for \;\; DCM$$

Considering the converter in CCM, substituting previous results, lead to:

$$\frac{DV_g}{R} < \frac{D(1-D)T_sV_g}{2L} \tag{2.41}$$

which, after the possible simplification, can be written as:

$$\frac{2L}{RT_s} < (1-D) \tag{2.42}$$

which can be expressed as:

$$K < K_{crit}(D) \quad for \;\; DCM \tag{2.43}$$

where $K = \frac{2L}{RT_s}$ is a dimensionless parameter which describe the tendency of the converter to operate in DCM. Large value of K lead to CCM, while small value lead to DCM for some values of the duty cycle.

Converter	$K_{crit}(D)$	$R_{crit}(D)$	$M_{DCM}(D,K)$
Buck	$(1-D)$	$\frac{2L}{(1-D)T_s}$	$\frac{2}{1+\sqrt{1+4K/D^2}}$
Boost	$D(1-D)^2$	$\frac{2L}{D(1-D)^2T_s}$	$\frac{1+\sqrt{1+4D^2/K}}{2}$
Buck-Boost	$(1-D)^2$	$\frac{2L}{(1-D)^2T_s}$	$-\frac{D}{\sqrt{K}}$

Table 2.2: summary of the critical parameters in DCM for the three main categories of power converter

It's actually clearer to express the conduction boundary in terms of load resistance:

$$R < R_{crit}(D) \quad for \;\; CCM \tag{2.44}$$

$$R > R_{crit}(D) \quad for \;\; DCM$$

where $R_crit(D) = \frac{2L}{(1-D)T_s}$, which for $D = 1$, the minimum value for the critical load is

$$R_{crit}^{min} = \frac{2L}{T_s} \tag{2.45}$$

Then if the load is less than this value, the converter operating in CCM for all values of duty cycle.

Table 2.2 indicates all the relevant parameters for the three described topologies of converter taking into account the DCM. In appendix A the complete derivation for the conversion ratio as a function of the duty cycle and the parameter K is described in detail.

Chapter 3

Digital Control Loop: PDU

In general DC-DC converters require a control loop to properly reject disturbance and meet precise stability and dynamic response specifications. If in the past the

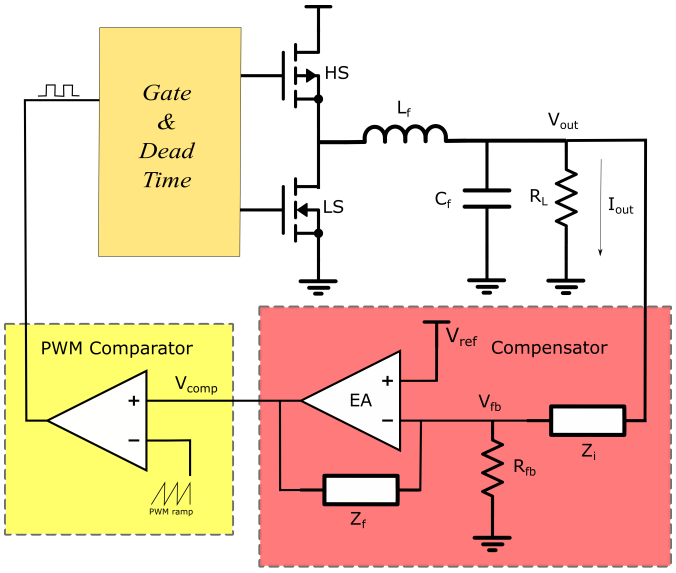


Figure 3.1: Voltage-mode controlled buck converter circuit

feedback control loop has been done analogically, in the last year digital solutions have become more and more attractive for applications with stringent requirements on precision, efficiency, area, and cost. It is useful to understand how the control loop is designed analogically to understand what are its limitations and how to implement it in digital.

The chapter will focus on the implementation of the state of the art of analog and digital control circuits, in particular the Analog voltage-mode control loop, presenting also a new technology capable to overcome the classic limitation of these circuits.

Figure 3.1 present the principle scheme of a voltage-mode controlled buck converter: the converter output V_out passes through a voltage divider, the resulting voltage V_{FB} is compared with a reference voltage V_{ref} at the input of an *Error Amplifier (EA)*, designed with a compensation network in feedback.

The whole block is named *Compensator*, in which output V_{comp} is given in input to a *Pulse Width Modulation (PWM)* block, which is a comparator having as reference input a fixed frequency sawtooth.

The Modulator output is a PWM signal having a duty cycle that varies according to the error between the desired reference voltage and the output of the converter.

The obtained signal drives the *High-side (HS)* and *Low-side (LS)* through a driving circuit which also sets the dead time to avoid cross conduction in the power devices. The PWM signal has a fixed frequency equal to the sawtooth signal frequency $1/T_s$.

3.1 Loop Transfer Function of voltage-mode control

The complete transfer function computation is done in appendix B and C, here only the final results will be reported.

The voltage-mode closed loop model is depicted in figure 3.2, where $G_{vd}(s)$ is the control input (namely, the duty cycle) to output transfer function, meanwhile $G_{vg}(s)$ is the line to output transfer function (ie. the output to input voltage ratio), $G_c(s)$ is the compensator transfer function, $F_m(s)$ the modulator transfer function, meanwhile $Z_o(s)$ is the output impedance seen as the transfer function of the opposite of the load current and the output voltage.

The open loop gain is given by equation 3.1.

$$L(s) = G_c(s)F_mG_{vd}(s) \tag{3.1}$$

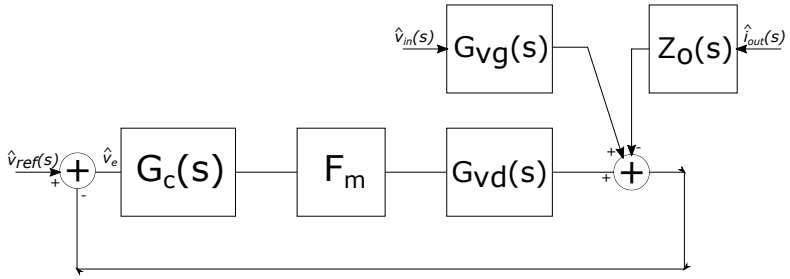
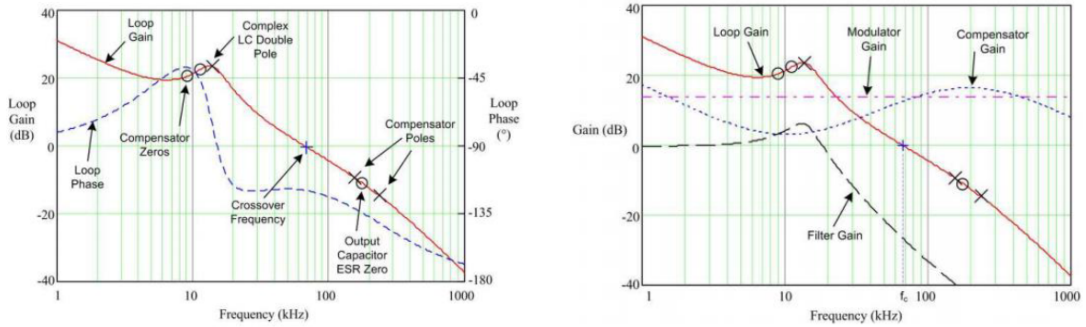


Figure 3.2: voltage mode closed loop model, the overall expression of the here reported transfer function is computed in appendix B and C.

The compensator must be designed to achieve the proper loop gain, given the modulator gain and the control input to output transfer function, to achieve enough phase margin (usually at least 45°) to guarantee closed-loop stability and with enough magnitude to ensure both small output steady-state error and a good disturbance rejection (the magnitude should be sufficiently high at low frequencies and sufficiently low at high frequencies) with at the same to obtain a certain prescribed closed-loop bandwidth. The conventional compensation strategy places



(a) **(b)**

Figure 3.3: typical loop gain and phase behavior for L(s) (a) and of $L(s)$ starting from $G_c(s)$,$G_{vd}(s)$, and F_m (b) [3]

the the two zeroes of $G_c(s)$ near the LC poles of $G_{vd}(s)$ (so to counter act them and gives a 180° phase contribution, which is useful for the phase margin). One of the zero of $G_c(s)$ cancels the one introduced by the ESR in the $G_{vd}(s)$, one of $G_c(s)$ poles is placed at half the switching frequency to attenuate high frequency noise. Figure 3.3.a depicts typical loop gain and phase behavior, meanwhile figure

3.3.b reports a reconstruction of $L(s)$ starting from $G_c(s)$,$G_{vd}(s)$, and F_m. The crossover frequency is typically 1/10 or 1/15 of the switching frequency.

The closed loop line to output $G_{vgCL}(s)$ and load current to output $Z_{oCL}(s)$ transfer functions are given by equations 3.2. If the closed-loop is well designed, at frequencies below the crossover frequency, the two transfer functions will be sufficiently small in magnitude, so that disturbances on the input voltage or on the load current acting in this range of frequencies will be well rejected.

$$G_{vgCL}(s) = \frac{G_{vg}(s)}{1 + L(s)} \tag{3.2}$$

$$Z_{oCL}(s) = \frac{Z_o(s)}{1 + L(s)}$$

The two equations in 3.2 defines the dynamic of the closed loop, also known as *Line Transient* and *Load Transient*: higher the closed loop bandwidth, faster the transient.

3.2 Digital Control Loop

Analog control suffer from several drawbacks: the RC network required by the compensator is difficult to be integrated, hence is usually realized with external component, this require to design the IC with additional pads, hence increasing the overall area of the die and increasing the Bill Of Material, hence the cost of the overall circuit; these component are also PVT dependent and in case of new specifications the compensator network has to be replaced and redesigned. Digital solutions has been proposed over the years to overcome these limitation and taking advantage of the reconfigurability of digital circuit.

3.2.1 Digital Voltage-Mode Control

Figure 3.4 present the schematic of a digital voltage-mode control loop. The description and analysis of such circuit is taken from [2]. The sensor $H(s)$ measure the output voltage, which is then subtracted by a reference voltage to get the error voltage $v_e(t)$. This is converted in digital by an analog-to-digital converter, which

is the boundary element between the analog and the digital world.

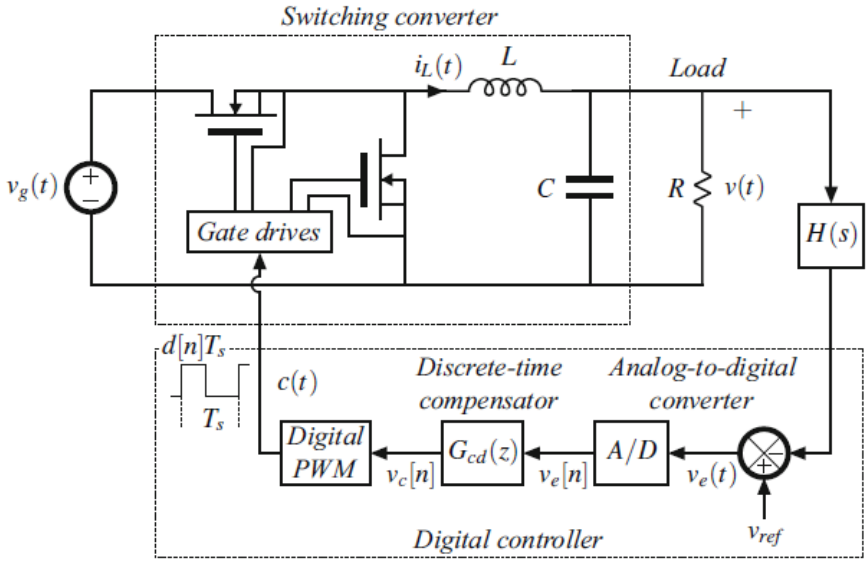


Figure 3.4: Principle scheme of a digital voltage-mode control loop [2]

The AD samples at a constant frequency f_{smpl}, usually set equal to the converter switching frequency, in this way is possible to obtain a dynamic response comparable to the analog solution [4].

The output of the converter is a sequence of digital word which represent the values of the error signal at the sampling interval $v_e[n] = v_e(nT_s)$. The discrete time compensator evaluates the error signal and generate a digital control signal $v_c[n]$ which is fed to a Digital PWM block (DPWM). This produces at output a PWM signal with a duty cycle $c(t)$ that depends on the control signal generated by the compensator.

3.2.2 A/D Conversion

In the loop the two quantities $v_e[n]$ and $v_c[n]$ are the sampling of $v_e(t)$ and $v_c(t)$, they are digital variables characterized by a fixed number of bits. Analog to Digital conversion perform a discretization a continuous signal in time and amplitude. The signal processed by an A/D converter is quantized to a $n_{A/D}$ bit digital word with a Leas Significant Bit (LSB) value given by 3.3. The classic quantization

characteristics $Q_{A/D}$ operating from 0 to full scale voltage V_{FS} is depicted in figure 3.5

$$q_{A/D} =' \frac{V_{FS}}{2^{n_{A/D}}} \tag{3.3}$$

In the loop the sensed analog signal is subtracted from the reference voltage to obtain the error signal in the digital domain. From the quantization characteristics it can be seen that the LSB resolution determines how well the output voltage can be regulated by the control loop: namely, it determines the minimum difference which can be converted and processed by the loop. Alternatively, equation 3.3 can

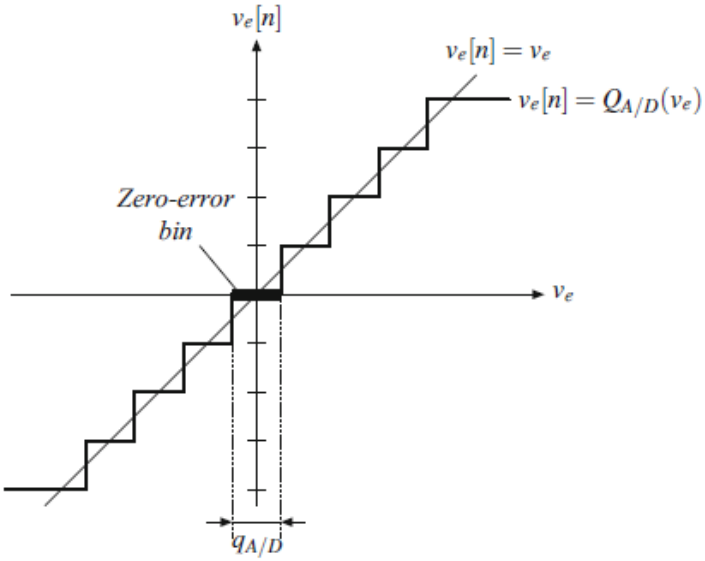


Figure 3.5: AD quantization characteristics centered around zero error [2]

be solved for the number of bits given the wanted resolution.

$$n_{A/D} > log_2(\frac{V_{FS}}{q_{A/D}}) \tag{3.4}$$

Since the sampling frequency is set equal to the converter switching frequency, in case of stringent request in performance, both in terms of voltage regulation specification and high switching frequency, the complexity of the converter can easly increase (e.g. an high sampling frequency solution can be a flash converter, which complexity, in terms of required comparators, increases exponentially with

the number of bits).

3.2.3 Digital PWM

Digital Pulse-Width-Modulation follows the same rules of the analog counter part. The output of the comparator $v_c[n]$ is compared with a digital saw-tooth ramp, giving in output a square wave $c(t)$ which $T_o n$ is related to the amount of time for which the ramp is lower than the $v_c[n]$ value, as depicted in figure **??**.a. Because of the comparison with a digital saw-tooth, also this block introduces a

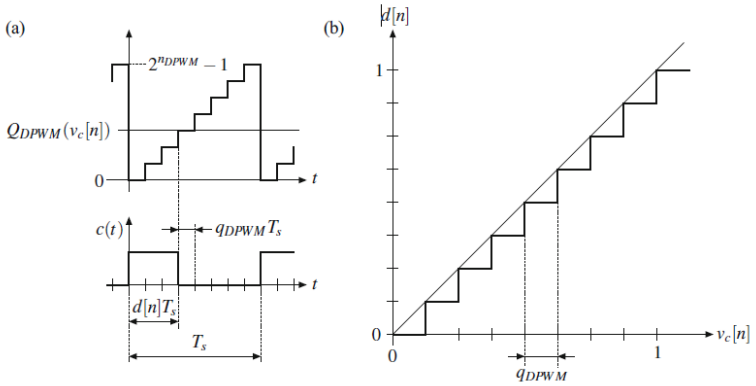


Figure 3.6: Digital Pulse-Width-Modulator: time quantization (a) and quantization characteristic (b), example with $n_{DPWM} = 3$ [2]

sort of quantization, with a time resolution of the $c(t)$ pulse is $q_{DPWM}T_s$ where $q_{DPWM} = 1/2^{n_{DPWM}}$ and n_{DPWM} is the DPWM resolution in bits.

Figure **??** assumes that the digital saw-tooth amplitude is $2^{-n_{DPWM}} - 1$, which corresponds to an equivalent DPWM gain equal to 1 V^{-1}.

In a standard DPWM the digital saw-tooth is simply implemented by a counter driven by a clock frequency f_{clk} which determines also the time resolution of the overall block, so equation 3.5 holds.

$$q_{DPWM}T_s = T_{clk} \tag{3.5}$$

Consequently, the duty-cycle resolution determines how well the converter output voltage can be positioned. In the case of a buck converter, its relation is $V = DV_g$,

so the minimum variation is given by $\Delta V = q_{DPWM} V_g$, hence

$$\frac{\Delta V}{V} = q_{DPWM} \frac{V_g}{V} = \frac{1}{2^{n_{DPWM}}} \frac{1}{M} \tag{3.6}$$

Supposing to want a position the output voltage within 0.1% in a converter with a conversion M equal to 0.2. From equation 3.6 and 3.5 a 13-bit ADC is required and the required clock frequency must be $f_{clk} = 8192 f_s$. This means that between the two frequencies there are three order of magnitude, which can result in an impractical implementation of the DPWM block. The problem has been addressed considering alternative DPWM architecture and solution [4] [5] [7] [6], as well as the solution proposed in this Thesis work.

Actually, the A/D and DPWM quantization characteristics are non linear, as assumed up to now, this have implications on the stability and operation of the digitally controlled converter. This non linearity can be neglected in case high-resolution unit can be used.

3.2.4 Discrete Time Compensator

The discrete time loop gain is given by 3.7 for the not compensated gain. The equation is retrieved considering figure 3.4.

$$L_u(s) = H(s) G_{vd}(s) e^{-st} \tag{3.7}$$

With $H(s)$ the sensor transfer function, $G_{cd}(s)$ the control input to transfer function of the converter and e^{-st} is the delay element introduced by the A/D converter and the DPWM block to compute $v_c[n]$.

The design of the discrete time compensator start with the definition of the required transfer function in the analog case; then a Z transformation is performed, passing from $G_c(s)$ to $G_c d(z)$. The domain transformation is carried out by using the Tustin mapping or Tustin mapping with pre-wrap. Tustin mapping consists in a variable substitution is expressed in equation 3.8.

$$s \rightarrow \frac{2}{T_s} \frac{z-1}{z+1} \tag{3.8}$$

The transformation allows to have matching between frequency response of the continuous-time compensator and the discrete counter part well below $f_s/2$.

To mitigate the difference near the $f_s/2$ corner, Tustin mapping with pre-wrap is used, which variable substitution differs from the insertion of a pre-wrap gain, as express in equation 3.9

$$s \to k_{prewrap}\frac{2}{T_s}\frac{z-1}{z+1} \qquad k_{prewrap} = \frac{\omega_{prewrap}\frac{T_@}{2}}{tan(\omega_{prewrap}\frac{T_@}{2})} \tag{3.9}$$

the value of the gain is set in such way so that G_cd and G_{cd} match in magnitude and phase for a particular frequency that is $\omega_prewrap$. The crossover frequency defined in the first step of the design can be preserved by setting $f_c = f_prewrap$. The input-output relationship of a discrete time compensator can be expressed by

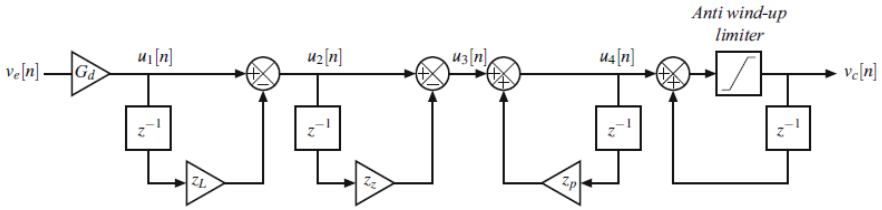


Figure 3.7: Cascade realization of the discrete-time PID compensator [2]

means of a finite difference equation, where the $v_c[n]$ can be express as a function of $v_e[n]$ and past values of both. This allow to realize the compensator as a cascade of elementary digital blocks,3.7, hence it is compatible with HDL description or software implementation in a micro-controller. This enhance the flexibility of the system because change in the required transfer function translates in changes of simple multiplication parameter. Also being the system described in RTL no manual tuning of the circuit are required, but it can be automatically handled by modern EDA tools.

The ”anti-windup” limiter have the same function of a voltage limiter at the output of an analog compensator built around an op amp.It is necessary to determine with care the number of allocated bits allocated to digital words representing the parameters and the signal values to prevent overflows or other calculation errors.

3.2.5 Limit Cycles

Actually, the A/D and DPWM quantization characteristics are non linear, as assumed up to now, this have implications on the stability and operation of the digitally controlled converter.

In steady-state, the digital control loop should have constant value for both input and output, a constant $d(t)$ and the converter should have periodic waveform.

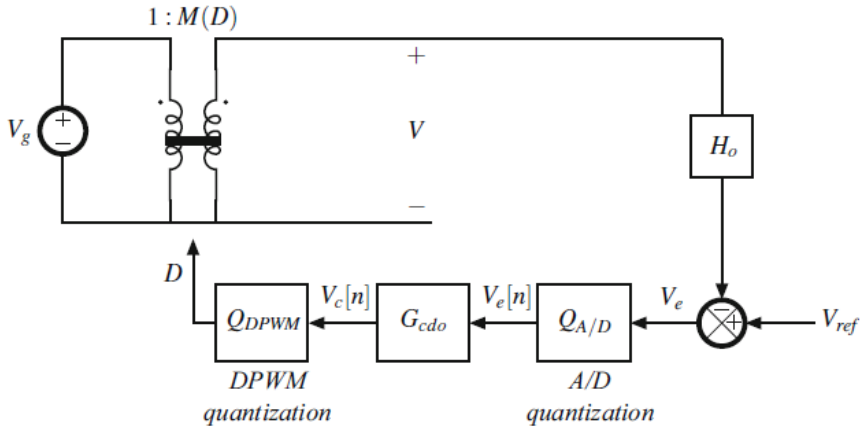


Figure 3.8: DC model of a digitally controller buck converter [2]

Figure 3.8 depicts a model for a buck converter in steady state, with the block substituted by their DC counter part, namely H_0 and G_{cd0} are the DC gain of sensor and compensator block; $Q_{A/D}$ and Q_{DPWM} model the effect of quantization.

With V_g the control to output DC gain of the buck converter, the width of the horizontal bins of the characteristics around the loop, figure 3.9, is equal to $V_g H_0 q_{DPWM}$ with an height of the vertical steps equal to Q_{DPWM}/G_{cd0}.

If the DC gain of the compensator G_{cd0} is finite, the equilibrium point is A: this is no a feasible condition since for every disturbance the A/D output bounces between different quantization steps. This is known as *limit cycling*.

When the compensator includes an integral function (which means an infinite DC gain) the characteristics around the A/D becomes a series of points spaced by $V_g H_0 q_{DPWM}$. In this case multiple zero solution are possible under the assumption 3.10, which states that the widths of the bins due to the DPWM are shorter than

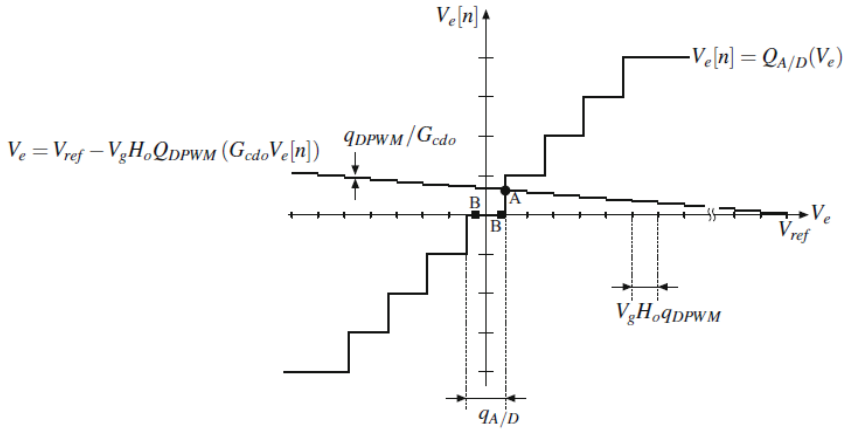


Figure 3.9: Stability point analysis in the case of a finite resolution quantization [2]

the A/D bins.

$$V_g H_0 q_{DPWM} < q_{A/D} \tag{3.10}$$

This is valid for the buck converter case, with a conversion factor equal to D and a converter dc control-to-input gain equal to V_g, which is not the case in general.

Related to the analysis of equilibrium points with A/D and DPWM quantization, it is interesting noting that the A/D quantization, in combination with the integral action in the compensator, lead to an effective steady-state quantization of the duty-cycle command $v_c[n]$. Consequently, for an equilibrium solution to exist, the condition for which the compensator includes an integral action and the DPWM resolution is sufficiently high do not suffice. For example, consider the response

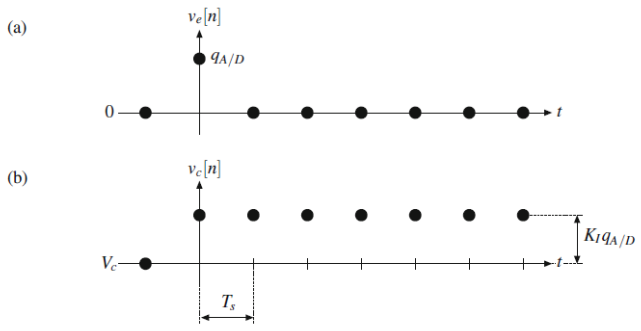


Figure 3.10: Waveforms illustrating (a) an impulse in error $v_e[n]$, and (b) impulse response of a digital compensator with integral gain K_I [2]

of an integrator to a unit error pulse of amplitude $q_{A/D}$, i.e., the smallest possible perturbation, as in figure 3.10. The response of the integrator is a step with amplitude equal to $K_I q_{A/D}$, where K_I is the gain of the integrator. As a result, the duty cycle command signal $v_c[n]$ and thus the duty cycle is quantized with a bin width equal to $K_I q_{A/D}$, regardless of how high the DPWM resolution is.

This effective quantization of the DPWM has the same consequence on the existence of zero-error equilibrium solutions as the DPWM LSB resolution in equation 3.10, this lead to the necessary condition 3.11

$$G_{d0} K_I H_0 q_{A/D} < q_{A/D} \Rightarrow G_{d0} K_I H_0 < 1 \tag{3.11}$$

3.2.6 Programmable Delay Unit - PDU

The proposed solution is based on a hybrid DPWM architecture consisting of a counter and a *Digital Delay Line* (or *Tapped Delay Line*) and allows to overcome the strict frequency requirements to achieve high resolution of the control signal. Figure 3.11 shows the basic schematic principle scheme of a DDL, which consists of a chain of delay elements whose outputs, called taps, are selected by a multiplexer: This allows the generation of small discrete time steps corresponding to the delay elements' propagation times.

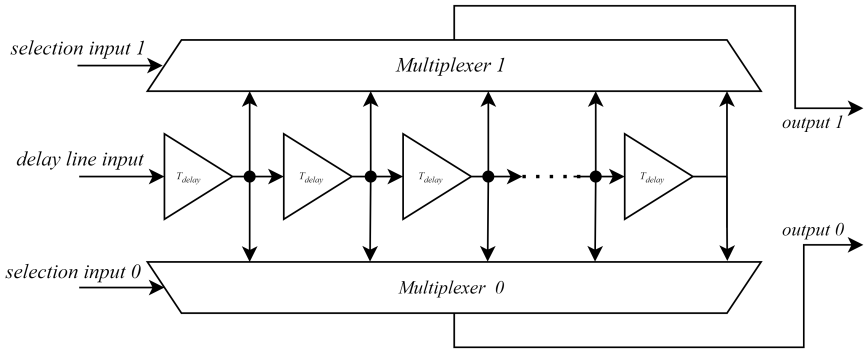


Figure 3.11: Principle Scheme of the DDL with two muxes: this allow to have two output with a digitally controlled phase

In a hybrid DPWM, the discrete steps of the duty cycle are obtained by summing two contributions: a coarse one coming from the counter and a fine one coming from the DDL. Therefore, the T_{on} of the DPWM signal is given by the equation

3.12.

$$T_{on} = T_{coarse} + T_{fine} \tag{3.12}$$

The coarse contribution comes from the counter, which is used as a digital sawtooth, while the fine contribution is obtained by the proper output of the delay line.

The proposed hybrid DPWM, called Programmable Delay Unit (PDU), is discussed below. The next chapter describes the synchronization problems and the digital interface required to overcome them.

The general architecture of the PDU is shown in Figure 3.12.

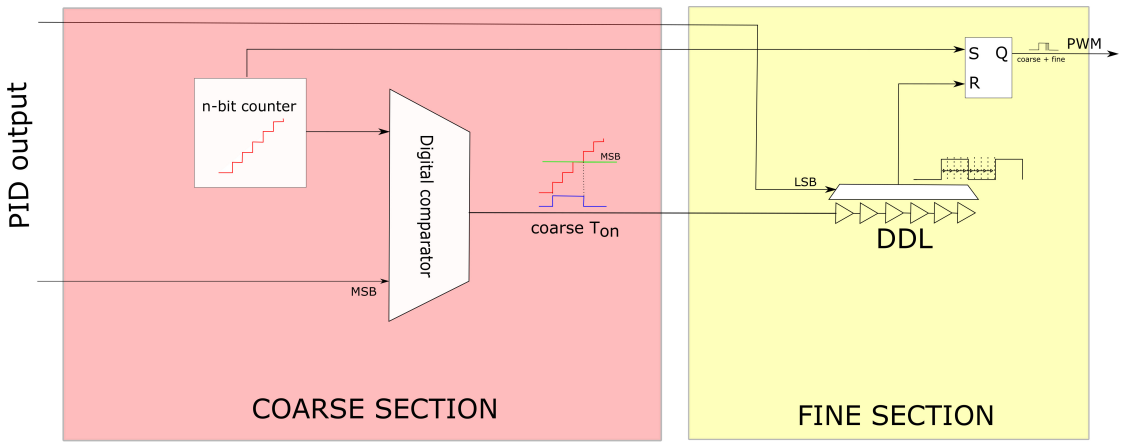


Figure 3.12: Principle Scheme of the Programmable Delay Unit: the two different sections, which gives the coarse and fine component of the final DPWM signals, are here highlighted.

The PDU takes the most significant bit (MSB) of the digital compensator output and compares it to an N-bit counter, which acts as a digital sawtooth, using a digital comparator. Since the counter is only compared to the MSB of the digital compensator output, it can have fewer bits than it would need if it were compared to the entire digital compensator output, so a clock frequency that is not too high can be used for this, so there are no technological limitations.

When the counter starts, the set signal of a flipflop and the coarse T_{on} are asserted. The counter clock is passed to the DLL so that its rising edge is delayed by a small time step given by the delay element in the chain, resulting in fine tuning of the T_{on}. Each outputs of the delay elements in the chain are sent to a multiplexer which has as its selection signal the least significant bits of the compensator output

(LSBs).

The selected tap goes into the reset of the flip-flop mentioned above, which clears the output signal that is the desired DPWM signal. Finally, when a new value of the compensator output is given to the PDU input, the counter is restarted and the DPWM signal is updated.

Digital Delay Line

Let us describe the DDL architecture used in the PDU. In general it can be implemented in two configurations:

- **Open Loop:** the architecture is simpler, but the element in the chain are sensible to PVT variations, which makes the delays of every element different from their nominal values
- **Closed Loop:** a more complex architecture, but insensitive to PVT variations.

A closed-loop configuration is used in the PDU, the basic scheme of which is shown in Figure **??**. The implemented architecture is a Delay Locked Loop (DLL) and consists of a phase detector (PD) and a charge pump (CP) in addition to the DLL. The PD generally takes two inputs and outputs a signal proportional to the phase difference between the two inputs. This signal controls the CP, whose output

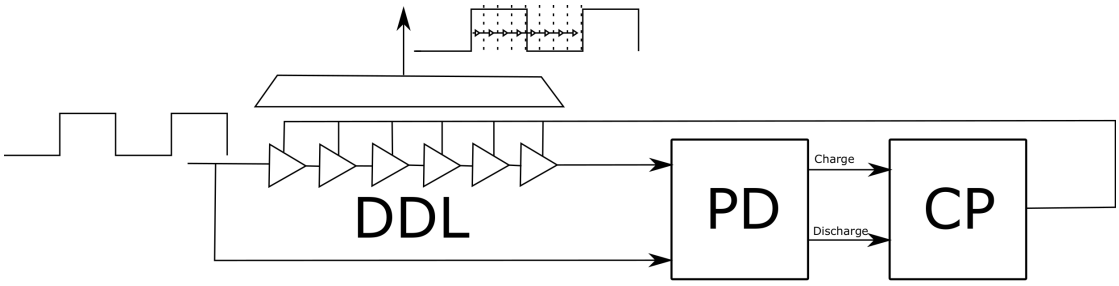


Figure 3.13: DLL principle scheme implemented in the PDU

voltage is used to control the delay element. The DLL behaves in such a way that it tends to bring the phase difference between the two inputs to zero: This means that the loop tends to match the clock entering the DLL and the delayed replica. So if the DLL has a total delay of one clock period, in the closed- loop configuration it will exactly match the clock period.

Then, having n element in the delay line, after the initial locking transient, the delay of every element will be given by 3.13

$$\Delta t = \frac{T_{clk}}{n} \tag{3.13}$$

Where T_{clk} is the clock period of the counter. This means that the closed- loop configuration counter compensates for PVT variations so that the delay of the elements in the chain remains the same apart from possible environmental variations.

Chapter 4

Digital Design flow

As semiconductor technology advances and shrinks, traditional chip design methods are becoming increasingly difficult. In addition, according to Moore's Law, the number of transistors per chip increases approximately every two years, and if the design capabilities can follow the technological trend, the same is not true for the verification methodology, as shown in Figure 4.1, resulting in the so-called *Verification Gap*.

In addition, the flow of chip design has remained the same or been steadily reduced due to critical *time to market* pressures. Only in recent years has a new type of approach begun to be used: topographic synthesis, which is based on a more technology and floor-planning aware synthesis that is capable of producing even better results from chip design.

In this Chapter the overall design cycle will be described, also focusing on two important tools that reduce the verification and validation steps: *SpyGlass*, used for pre-synthesis formal and syntax verification of the written code, and *Formality* used for post-synthesis formal verification.

4.1 From Specification to RTL

Starting from the idea dictated by the market, the architectural and electrical specifications are derived: The former defines the functionality and partitioning of the chip, the latter is required for the relationship between the blocks in terms of

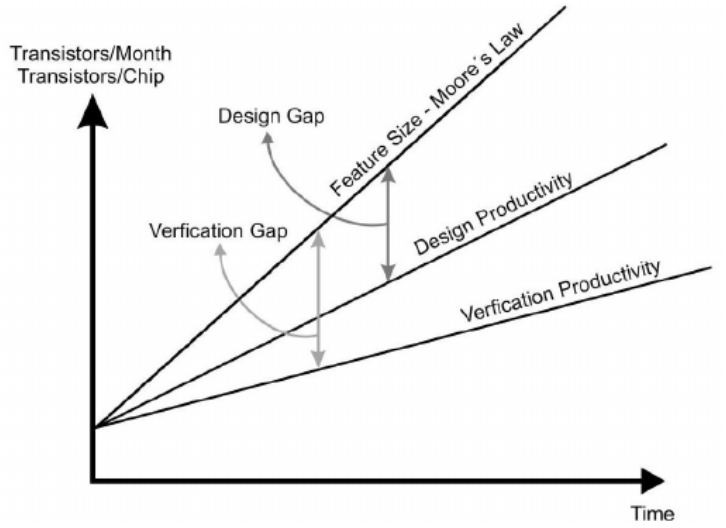


Figure 4.1: As the technological capability grows, the design capabilities follows with a slightly lower slope (productivity gap), but the verification capabilities grow slower, also due to the fact the simulator performance does not match with the synthesis and design tool performance [[8]]

timing and voltage range.

Then, these specifications must be implemented: HDL language is used for this purpose. As the name suggests, the hardware description language is used to describe the blocks used in the design, as well as the complex functionality that will be read and interpreted by the next tools for the design of the final chip. These languages make it possible to follow technological progress, since the designer can describe blocks with a single line of code, including hundreds of transistors. The two most commonly used HDL languages are VHDL and Verilog, both of which provide the same results but have different advantages and disadvantages.
The design description can be made by the three level of abstraction:

- **Behavioral:** highest level of abstraction, mainly used to translate specification to code that can be simulated.
- **RTL:** actually describes and infers structural component and their connections, used for synthesis.
- **Structural:** mainly used to describe blocks gate by gate.

The design is described as RTL and is divided into a series of sub-blocks that

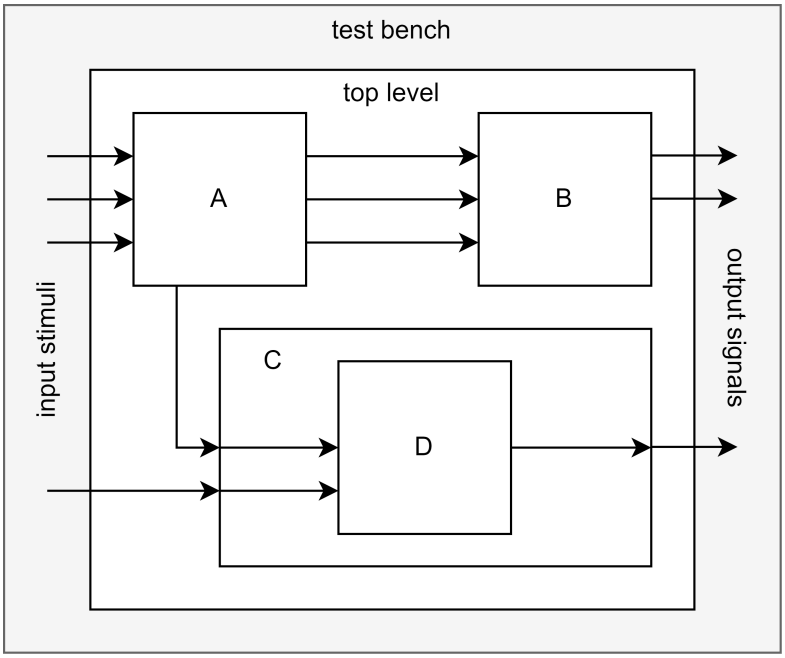


Figure 4.2: The testbench inject signal in the RTL described design, which may have some internal hierarchy and partitioning to make the design simpler.

represent a hierarchy of the design.

Before synthesis, it is important to verify that the written RTL behaves as expected. For this purpose, another block is designed, the *test bench*: This block, shown in Figure 4.2, injects input signals into the inferred design and verifies the output: during this phase, timing constraints are not considered by the simulator. To minimize the difference between the synthesized netlist and the RTL code, these delays are usually encoded in the RTL code.

4.2 Spyglass and Early Design Analysis

Before synthesis, it is important to find and correct potential errors in the RTL design that are otherwise difficult to fix in the later stages of the design. The SpyGlass tool from *synopsys* is an integrated solution for analysis, debugging, and troubleshooting with a comprehensive set of features for structural and electrical issues all related to the RTL description of the design.
As design teams become more geographically dispersed, consistency and correctness

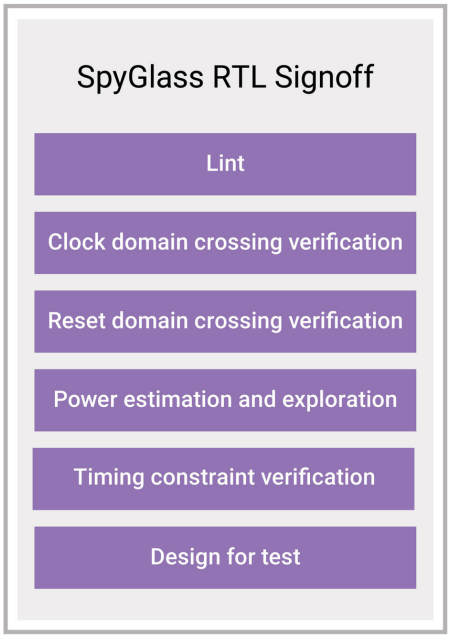


Figure 4.3: SpyGlass early design verification capabilities [Courtesy of Synopsys]

of design intent becomes a key challenge for chip integration teams. The emphasis on design reuse and IP integration requires that design elements be integrated and conform to correctness and consistency guidelines. To this end, the tool also checks if the code adheres to some "design conventions" that can also be set by the user. In our case, the tool checks if the code complies with the STMicroelectronics Bluebook rule set.

As described in Figure 4.3, in addition to the classic syntax check (lint verification), the tool can also perform a pre-synthesis verification for clock and reset domain crossing, power and timing estimation, design constraints check and a DFT verification.

This makes SpyGlass necessary to shorten the design and analysis phase for the next phases of the design, thus reducing time-to-market and increasing the quality of the written RTL.

4.3 Synthesis flow

The process of translating the RTL into a gate-level netlis is called synthesis. In general, it is an iterative process that begins with the definition of timing constraints for each block of the design. These define the relationship between each signal and the clock input for that block. An environment file is also required to specify any technology cell libraries and other relevant information that the synthesis tool will use during synthesis.

The tool takes the RTL code, constraints, and technology cell libraries and creates a gate-level netlist. During this process, the tool tries to optimize the design to meet the specifications and constraints. Many steps may be required for this phase.

At the end of this phase, it is also possible to insert some additional circuits to make the overall design testable, as described in the next section. The synthesis

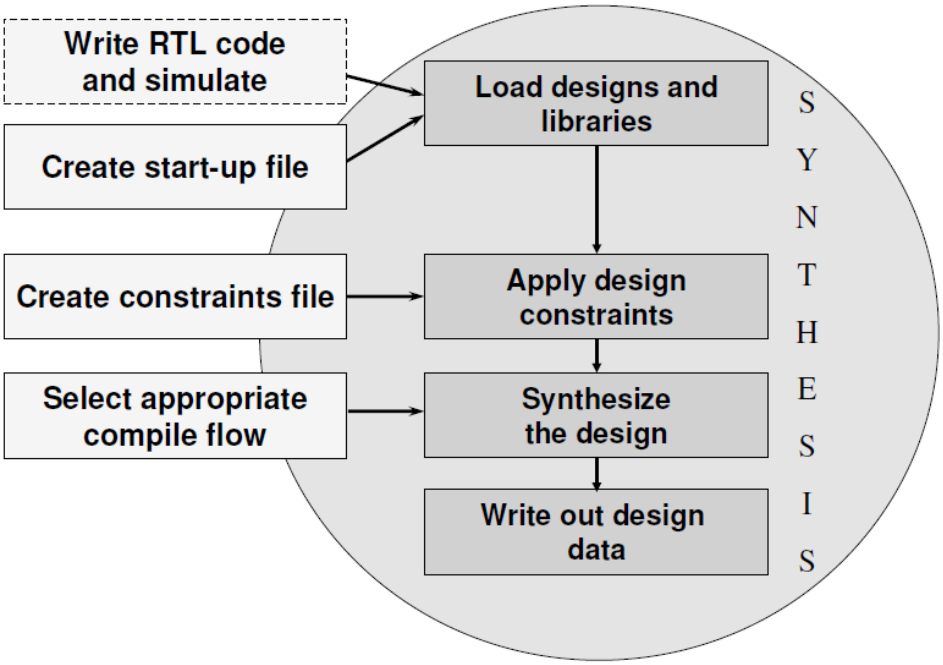


Figure 4.4: Steps involved in the synthesis flow, using the synopsy Design Compiler [Courtesy of Synopsys]

step follows the translation of the design from a behavioral description to an RTL description, to get the best possible Quality-Of-Result (QoR) out of the synthesis tool is important to properly describe the circuit with synthesizable RTL, the result of the synthesis step is a gate-level description which can be processed by a Place-&-Rout (P&R) tool to generate the GDSII file which will be sent to the foundry to produce the final circuit on silicon.

The Design compiler flow consists of 4 steps, shown in figure 4.4. In the first step, the design described in RTL is read by the tool, invoking the series of command `analyze` and `elaborate`, they are capable to read both Verilog and VHDL source file, differently from the traditional `read_vhdl` and `read_verilog` commands. These generate the so-called *unmapped ddc* database of the circuit, in which every element is associated with a General TECHnology (GTECH) basic element. This view will be used by the tool to perform all the possible optimization that can perform.

Before going on, there are some "Good practices" advised by Synopsys. After reading the design, if it was done by using the classic textttread_vhdl and `read_verilog` commands, it is crucial to *link* the design by using the `link` command, this allows to resolve, or locate the definition, of all the cell in the design. It is good practice to perform a manual link because many commands perform an auto-link, but in case of linking errors, the command will not be aborted leading to the generation of meaningless reports and results. It is then always useful to perform a manual link to catch these problems early in the synthesis flow. After the link command is useful to invoke the `check_design` command which alerts the user if there are generic problems in the design: it returns a 0/1 result, which is useful to control the synthesis flow in case it is managed by an automatic script and not by the user. Using `analyze-elaborate` it is not necessary to perform the link since it is done automatically by the two commands.

Then, if the design should be read again in the future to perform other syntheses (e.g. to try different optimization strategies) it is important to save the unmapped DDC database because the `read_ddc` command runs faster than the one used to read directly the source files. The syntax to write the DDC file is: `write -format ddc -hier -output folder/file_name.ddc`. The `-hier` option allows to save in a single file all the designs, otherwise, only the top entity will be saved.

After the reading phase, is crucial to properly constrain the design by defining a constraint file: it will contain timing information about the clock in the design (e.g. skew, delay, maximum acceptable hold ...) as well as constraints on the input and output port. Design Compiler optimizes the design based on the described constraints, so it is important to not over-constrain the design, otherwise, it will lead to a non-synthesizable circuit. Appendix D describe more deeply what are the most important constraints that have to be set before compiling, which in Synopsys jargon is a synonym for synthesis, the design.

After the design has been constrained is possible to launch the synthesis capability of the tool: this can be done with many additional options by which is possible to apply advanced synthesis and optimization techniques. Among these, the `-map_effort high` applies maximum optimization effort during gate-level optimization and invokes critical path re-synthesis as needed (i.e. Cones of logic containing stubborn, violating critical paths are returned to logic-level optimization and then remapped, iteratively).

Another option is the boundary optimization `-boundary` propagates constants, unconnected pins, and complement information during compilation to reduce delay and area. Since boundary optimization may eliminate output ports and invert input port signals, this can have an impact on verification, for example, if a verification test-bench was created for the RTL code, providing signals and one of these ports is inverted or removed, the existing test-bench will no longer work.

It is important to use option `-scan` to perform a DFT aware synthesis before the scan insertion so to not incur in timing problems after.

With an incremental compile, DC begins re-optimizing the design at the Gate level. DC will perform Incremental Implementation Selection as needed to obtain faster arithmetic components. The high map effort enables "Critical Path Resynthesis" (CPR) which takes a critical path, along with its surrounding cone of logic, and reverts the gates to GTECH to re-optimize it (structuring) at the Logic level. CPR is invoked iteratively until all violating paths can no longer be improved, or meet timing

Another important option is the `-gate_clock` option, which automatically implements clock gating in the design according to options set by the `set_clock_gating_style` command. This enables the possibility to stop the propagation of clocks

in regions of the design that are not used to avoid useless work of the logic and memory element, hence saving dynamic power.

Another approach is by using a more powerful built-in command which is `compile_ultra`, this automatically deploys a two-pass compile strategy and automatically uses an advanced algorithm to better optimize the logic of the design: operator merging, Carry-Select-Adder Transformation, Logic duplication. Some additional options which enhance QoR are: `-timing` that invokes a timing-centric, high-performance arithmetic algorithm, `-retime` performs *adaptive register retiming*, which moves the logical location of registers up or down a timing path, to help improve local critical path timing without creating or worsening timing violations of surrounding paths.

At the end of the synthesis flow, it is possible to generate a Verilog netlist containing the gate-level description of the design or a *mapped ddc* file. In case the design has to be used as a hard macro in another design it is possible to read the Verilog netlist and infer the required constraint or it is possible to directly read the DDC file. The syntax to write the gate-level netlist or the mapped ddc database is the same mentioned above, the only difference in the case of the gate-level file is to change the format option from ddc to Verilog.

4.3.1 DFT - Design for Testability

After the design has been synthesized and produced by the foundry there can be some issue that may arise only at this stage of the design, at a prototipe level. These defects can be analyzed at three different level, that are actually correlated: physical level (e.g. silicon defects, photolitography defects, mask contamination), electrical defects (e.g. short or open circuits, transistor stuck on/Open, changes in threshold voltages), logical defects (logic stack at one or zero, slower transition, and/or bridging). it is impossible to identify these defects after that the design has been produced and put in a package: to do so the circuit has to be predisposed so to check if there are these type of defects in the circuit. The Design for Test (DFT) enanche the observability and controllability of the internal nodes to grasp the nature of potential defects in the manufactured circuit and increase the fault

coverage which can be evaluated as 4.1

$$Fault\ Coverage = \frac{Number\ of\ detectable\ faults}{number\ of\ possible\ faults} \tag{4.1}$$

The DFT techniques are increasingly gaining momentum among ASIC designers. These techniques provide measures to comprehensively test the manufactured device for quality and coverage. Traditionally, testability was considered as an after thought, with implementation done only at the end of the design cycle. This approach usually provided minimal coverage and often led to unforeseen problems that resulted in increased cycle time. Merging testability features early in the design cycle was the final solution, creating the name Design-for-Test [12].

Designers create functional simulations to verify the proper operation of their designs. For example, a designer of a memory controller creates simulations to verify that the design operates correctly within the system. This approach is fine for the virtual world, in which the chip is only bits and pieces of HDL coding. However, you ultimately want to manufacturer a real chip and verify that it works properly. Production tests can verify that the manufacturer correctly implemented the design and that the design is free from flaws, such as power and ground shorts, open interconnections due to dust particles, and others. In short, production testing ensures that the customer receives high-quality parts with low failure rates.

In the past, functional simulations were used to generate test vectors, which could then be used to verify newly manufactured chips on a tester. But because of the high gate counts and extreme complexity of today's SOC designs, these production-test techniques are quickly running out of steam. One can still use functional simulations to verify the operation of a design, but producing enough simulations to provide high fault coverage is becoming more difficult.

With the advent of scan-design techniques and Automatic Test Pattern Generation (ATPG) tools, however, Considering the same design is possible to quickly produce several thousand production-test vectors that provide high fault coverage. The use of scan-design techniques simplifies the problem of test-pattern generation by reducing the design, or sections of a design, into purely combinational logic. Then using fast and efficient algorithms in ATPG tools (algorithms that the tool designer developed for combinational logic) is possible generate high-fault-coverage

vectors.

Various vendors, including Synopsys provide solutions for incorporating testability in the design. Synopsys adds the DFT capabilities to DC through its DFT Compiler (DFTC) that is incorporated within the DC suite of tools [10][11].

How DFT is implemented

The goal of the DFT functionality is to verify that the circuit works as expected: considering figure 4.5, to verify that the one node is not stuck at zero, is possible to use the proper input combination, then measuring the output. These test vector can be easily computed manually for simpler circuit, but it become unfeasible for complex circuit. The DFT functionality uses the flip-flops in the design in a unique shift register configuration, called scan-chain. Thus, is possible to shift pattern in the shift register, make the circuit work with these input and collecting the output. Alternatively, the scan-chain increases the controllability and observability

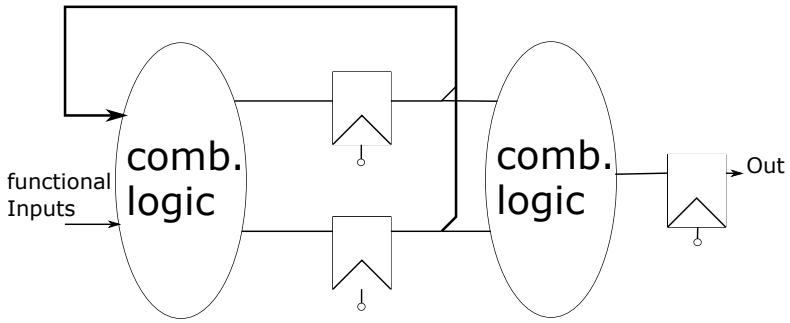


Figure 4.5: Example of design with Flip-Flops and combinational logic

of the internal nodes by connecting storage cells into a long shift register and by forcing the logic of these cells to support the scan-shift mode that allow serial loading and unloading of scan-chain's content. The scan-chain do not influence the normal mode operation of the circuit, only when the scan-mode is activated the scan-chain capability can be used. ATPG tools, as synopsys tetraMAX, are capable to automatically generate the proper set of test vector to get a fault coverage near the 95% with ease. Figure 4.6 shows the additional element required to implement the DFT functionality in a simple circuit. Only three additional pins are required: the test input, test output and scan-mode selector. The complexity of the circuit

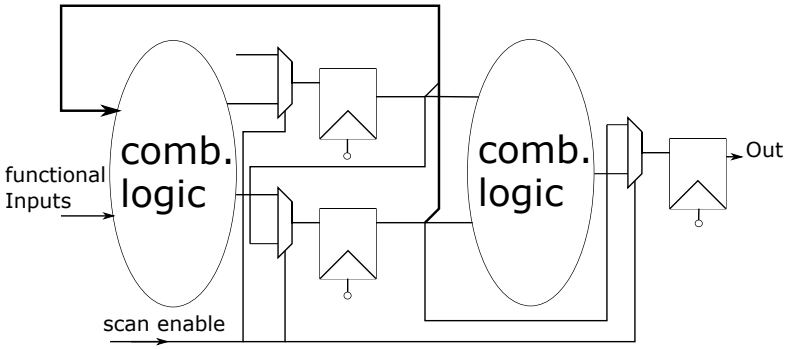


Figure 4.6: Example of scan-chain insertion in the design taken as example

do not increase drastically since only a layer of multiplexer is introduced. In the example the scan clock is shared with the system clock, to avoid additional pin for the two different clocks or using more complex memory element that allow to use two different clocks. From the design is easy to understand that, when the scan mode is activated, all the register are connected in a shift-register configuration that allow to insert in the system the proper test-vector, collecting the result in the output registers. The timing sequence, figure 4.7 with which the system works

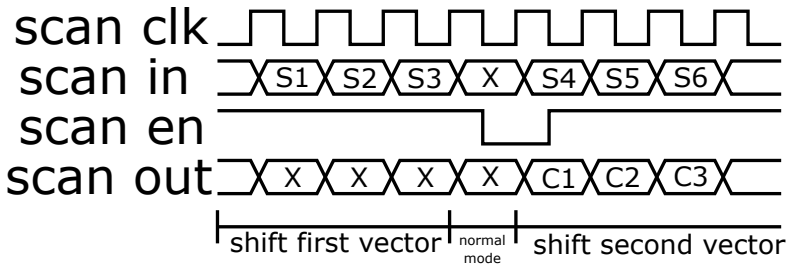


Figure 4.7: Timing diagram of the scan chain behviour

is the following: with scan mode activated, the proper test vector is loaded in the scan chain, then the scan mode is deactivated, allowing the input to be propagated in the combinational circuits and stored in the output registers, then reactivating the scan mode the results are shifted in output. In general, the system asserts and de-asserts the scan mode on the falling edge of the clock, which helps ease timing, especially the hold time constraints.

If different clock domains are present in the circuit, different chains are inferred by the DFT tool. Since the scan enables is unique for all the chains, having more

than one scan chain makes the timing management of the overall system more complex.

When the scan mode is asserted, each of the two scan chain load data in parallel, the length of the longest chain determines the length of the scan-shift operation, in the case of smaller chains is possible to still shift the largest amount of data being sure to insert the proper number of don't care data that will be shifted away. As described, after the chains are properly loaded the scan mode is de-asserted, the system applies one clock saving the output data, capturing the combinational logic data into the scan-storage cells. The capture cycle differs in the two clock domain, since they interact they must be staggered to avoid any timing problems. This is something that can be handled only by sequential ATPG tools, combinational ATPG tools assume that capturing data from one clock domain does not affect the capture data of another chain. In this case, the tool has to generate only one scan clock at a time during capture cycles, but this makes the test pattern generation more difficult due to the larger number of required patterns to get the same fault coverage. Sequential tools instead know that data from a clock domain changes during the capture cycle and that, consequently, the capture data of some flip flop in the other domain changes.

Scan Elements

Figure 4.8 shows the various scan elements which can be used to implement scan-chains. Multiplexed FF and clocked-scan techniques are better suited for designs that contain edge-triggered FF, meanwhile level-sensitive-scan-design (LSSD) are better suited for design that uses latches. Multiplexed Flip-Flops contains a D-FF with multiplexed inputs that allow selection of normal or scan inputs. In normal mode the flip flop behaves as usual, with the normal input selected, during scan mode the scan input is selected, allowing the FF to register the scan-data.

Clocked-scan elements are FF with a dedicated test clock to register data into the flip-flop, so that during scan mode it registers the scan data.

LSSD uses three independent clocks to capture data into the two latches within the scan cell. During normal mode, the master latch uses the system clock to latch system data at the functional input and send it through the functional output. During scan-mode, the two scan clocks control the latching of data through the

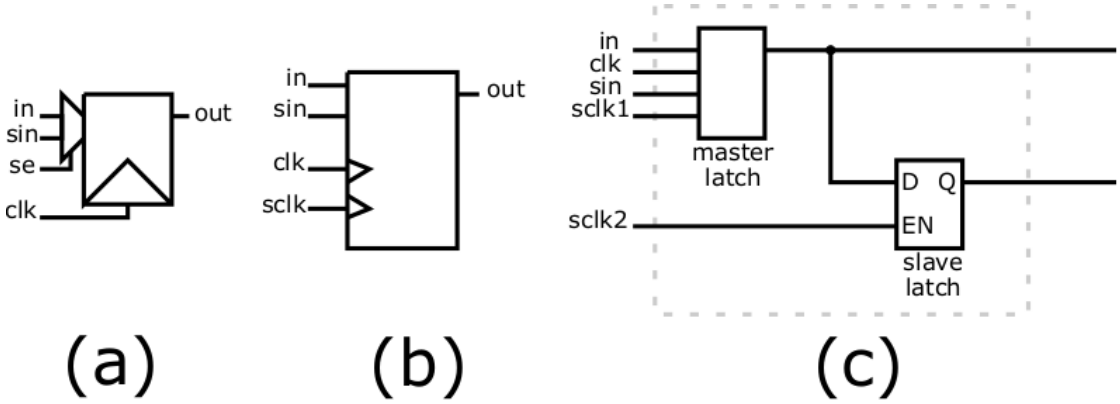


Figure 4.8: Scan elements types: (a) multiplexed flip-flop (b) clocked-scan elements (c) level-sensitive-scan-design (LSSD)

master and slave latches to generate the data at the scan output.

LockUp Latches

Clock skew can affect scan chains for two main reasons. The first is related to the layout and how it affects the propagation delay. The same clock may have to drive hundreds of scan-storage cells with no circuitry between them (that may acts as regeneration logic). Logically adjacent storage cell may be separated in the physical implementation. Clock Skew between successive storage-cell must be less than the propagation delay between the two registers, otherwise the second cell will store the new data and not the previous one, figure 4.9.

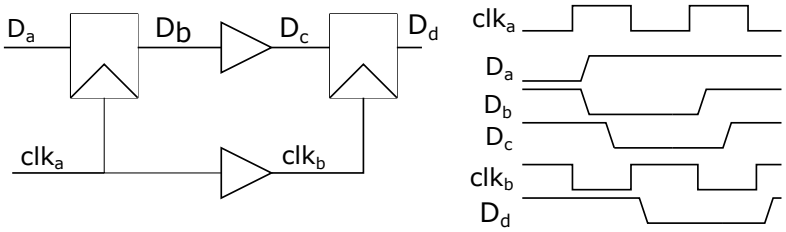


Figure 4.9: Data slippage caused by clock skew. Since the propagation delay is shorter than the skew, the second register stores the new data instead of the previous one.

The second reason arise from the presence of different clock domains. For example, if two clock domains are involved two different clock tree will be synthesized: this introduces a certain amount of skew between them and trying linking the two

domains arise timing issues.

Lockup Latches are transparent latches used to interconnect two scan-storage cells in a scan chain in which excessive clock skew exists. Typically, the latch has an active-high enable, which becomes high when the clock from the first domain becomes low: this add half a clock cycle of hold time to the output of the flip flop which drives the input of the lock up latch.

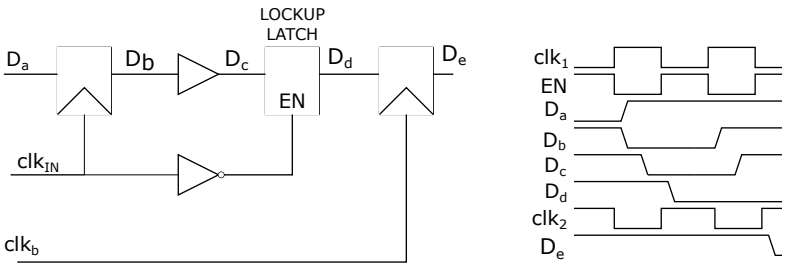


Figure 4.10: To reduce clock-skew problems is possible to use lock up latches to connect two scan-storage in a scan-chain

DFT Compiler usage

To perform DFT insertion it is called the `insert_dft` command. This introduces in the design two new inputs, the scan input and scan enable, and properly interconnects the scan-storage cells (which are alredy predisposed by the `-scan` option used when invoking the `compile` or `compile_ultra` command) to implement the scan-chain. Scan configuration must be performed before invoking the `insert_dft` command.

After the DFT insertion, to grasp possible violations, the `dft_drc` can be invoked (where drc stands for *design rule check*). This generates a file in which are reported all the design violations, such as protocol violations, non scannable elements, reset related problems and many others with a list of all the cells that have presented the reported violations.

At the end of the DFT insertion phase, if the design has to be inserted into another design it is important to store the scan-chain information and read them when performing the synthesis and DFT insertion in the bigger design: this way the tool has knowledge on the internal scan-chain and properly interconnects it with the external circuit. This is done using the `write_scan_def` command to

save the SCANDEF information and use the `write_test_model` command to save test model information about the block-level scan chain. It is important to save the scan-chain database file in `.ctl` format, which is the format that can be read by the `read_test_model` command.

4.4 Formality and formal verification

Formal verification techniques perform validation of a design using mathematical methods without the need for technological considerations, such as timing and physical effects. They check for logical functions of a design by comparing it against the reference design.

The main difference between formal methods and dynamic simulation is that former technique verifies the design by proving that the structure and functionality of two designs are logically equivalent. Dynamic simulation methods can only probe certain paths of the design that are sensitized, thus may not catch a problem present elsewhere.

In addition, formal methods consume negligible amount of time as compared to dynamic simulation. The purpose of the formal verification in the design flow is to validate the RTL against RTL, gate-level netlist against the RTL code, or the comparison between gate-level to gate-level netlists.

- **RTL to RTL:** used to validate the new RTL against the old functionally correct RTL. This is usually performed for designs that are subject to frequent changes in order to accommodate additional features. When these features are added to the source RTL, there is always a risk of breaking the old functionally correct feature. To prevent this, formal verification may be performed between the old RTL and the new RTL to check the validity of the old functionality.

- **RTL to gate-level:** used to ascertain that the logic has been synthesized accurately by DC. Since the RTL is dynamically simulated to be functionally correct, the formal verification of the design between the RTL and the scan inserted gate-level netlist assures us that the gate-level also has the same functionality. In this instance if we were to use the dynamic simulation method to verify the gate-level, it would have taken a long time (days and weeks,

depending on the size of the design) to verify the design. In comparison, the formal method would take a few hours to perform a similar verification.

- **gate-level netlist against the gate-level netlist:** this too is a significant step for the verification process, since it is mainly used to verify – what has gone into the layout versus what has come out of the layout. What comes out of the layout is obviously the clock tree inserted netlist (flat or hierarchical). This means that the original netlist that goes into the layout tool is modified. The formal technique is used to verify the logic equivalency of the modified netlist against the original netlist.

Chapter 5

Digital Interface Implementation

Section 3.2.6 described the complete schematic of the PDU; in fact, the circuit has some problems that are solved by the digital interface, which is the focus of this work. The problems are mainly due to the fact that the internal clock, which enters the delay line, being managed by some circuits necessary for the correct control circuitry, required to correctly drive and lock the DLL, is affected by PVT dependent delays. Basically, the circuit generates a mask that is the enable for

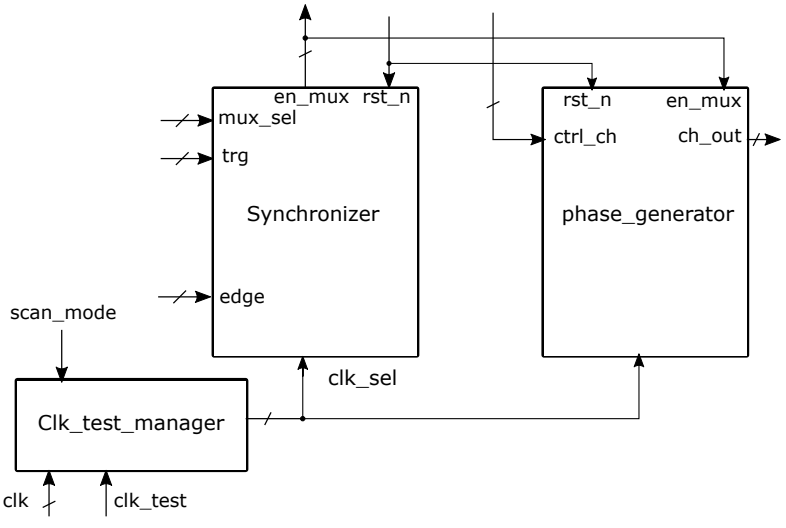


Figure 5.1: Principle Scheme of the implemented digital interface

the multiplexer, which sends the edge out of the multiplexer. The edge and the enable are generated starting from two different clocks, altough related: the system clock and the internal clock that enters the delay line. As mentioned above, the internal clock is affected by the PVT delay. In the worst case, i.e. when the last multiplexer output is selected, the output edge falls outside the mask, so it is not be sent in output and is not taken into account by the rest of the circuit, causing the entire PDU to malfunction.

To avoid this, the digital interface that generates the correct enable signal and manages the multiplexer output works with the internal clock (which compensates for the delay difference between the two clocks) and generates the enable signal dynamically based on the position of the edge in the delay line.

Figure 5.1 represents the implemented circuit. It consists of two sub-blocks described below: the synchronizer, whose task is to correctly generate the multiplexer enable signal by evaluating the position of the edge in the delay line, and the phase generator, which converts the above signal into a set or clear event for the PDU output channels, taking into account other control signals.

5.1 Synchronizer

In section 3.2.6, the general picture of the PDU was described: the unit when locked is such that it can send a copy of the input clock signal at the output with a programmable phase. The circuit requires the correct generation of the multiplexer enable signal, this is the main objective of the synchronisation block, which allows to properly select the output edge used by the phase generator.

5.1.1 Multiplexer enable signal generation - General Problem

In a first approximation, it can be assumed that the clock with which the synchronisation block operates is the one that enters the clock management unit, which sends the correct clock cycle to the delay line and the phase detector. In fact, this unit introduces a delay in the clock that depends on PVT. As a result, the clock that is processed by the delay line and the clock with which the digital section

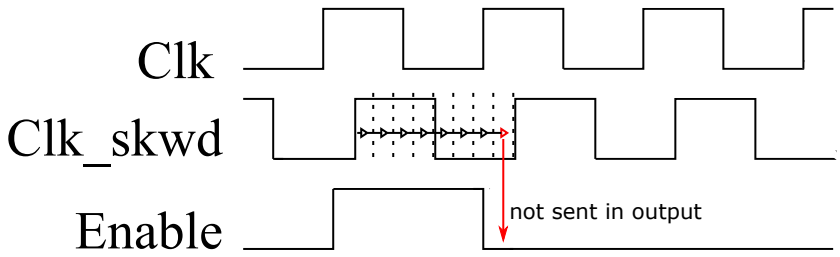


Figure 5.2: The two clocks have a phase shift PVT dependent, this can lead to a malfunction of the circuit if the required tap is not covered by the mask

operates with can have a phase shift that again depends on PVT.

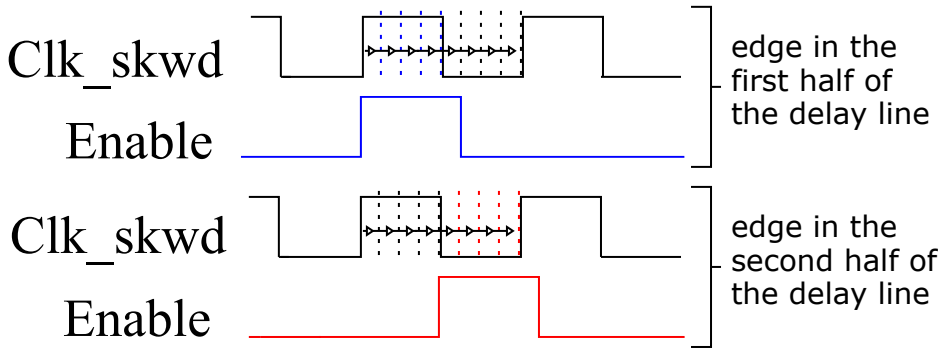


Figure 5.3: The mask is dynamically generated so to cover only a subset of the taps in such a way they fall right within, hence reducing the possibility to have not selected taps.

Since the enable signal is generated by the external clock in the worst case, the mask may close before the last tap of the delay line and thus will not be sent in output, causing the device to malfunction. A first countermeasure to this problem is to let the digital part work with the clock after the Clock Manager Unit, clk_skwd in Figure 5.2.

In fact, it is not enough to work only with the internal clock: Even if the mask is perfectly aligned with the clock period, if the mask is unique and covers all taps, there is a problem related to the selection of the taps at the beginning or at the end of the delay line: even if there are small errors in the generation of the mask, they may fall out and not be selected. For this purpose, the synchronizer block performs a dynamic generation of the mask based on the relative position of the tap in the delay line (Figure 5.3). The synchronizer block checks the first bit of the multiplexer selection signal to detect whether the desired edge is in the first or

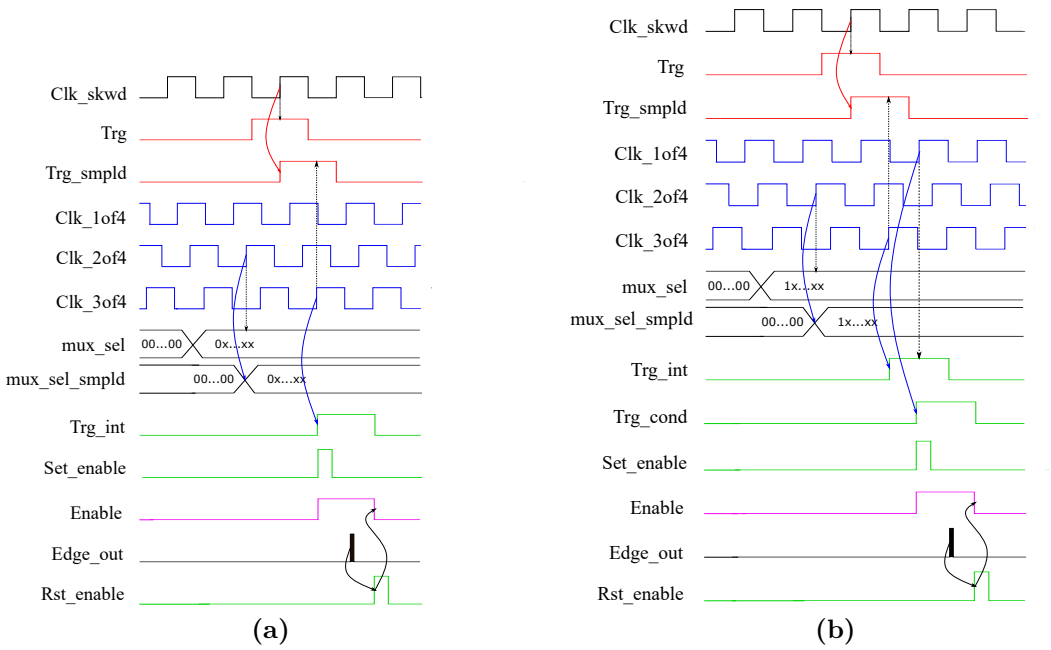


Figure 5.4: timing diagram which describes the mask generation process when the edge is in the first half (a) and in the second half (b)of the delay line

second half of the delay line, and based on this information the mask generation changes, in particular it is used to select clocks with different phase shifts. These are taken directly from the delay line, in particular at the beginning, half and three quarters of the delay line, called *clk_1of4*,*clk_2of4* and *clk_3of4* respectively.

5.1.2 Edge in the first half of the delay line

If the first bit of the multiplexer selection signal is zero, it means that the edge is in the first half of the delay line. In both cases, the circuit samples the trigger signal, (*trg_smpld* in figure 5.4.a), with the system clock. When enabled, this serves as the start event for generating the enable signal. Then the multiplexer selection signal is sampled with the *clk_2of4*. Then, in the considered case, the internal trigger is sampled again with the *clk_3of4* (*trg_int*). This allows it to be sampled only when the sampled selection signal is stable. *trg_int* is used as an event to raise the set signal for enable. When the desired edge leaves the multiplexer, the reset signal for the enable is asserted and the circuit returns to the idle state.

5.1.3 Edge in the second half of the delay line

If the first bit of the multiplexer selection signal is one, it means that the edge is in the second half of the delay line, figure 5.4.b. In this case, the signal used as the rising event for setting the multiplexer enable signal is the *trg_cond*, which is the sampling of *trg_int*. This allows the correct phase shift to be applied to the mask generation so that the exiting edge is exactly within the enable signal.

5.1.4 RTL Implementation

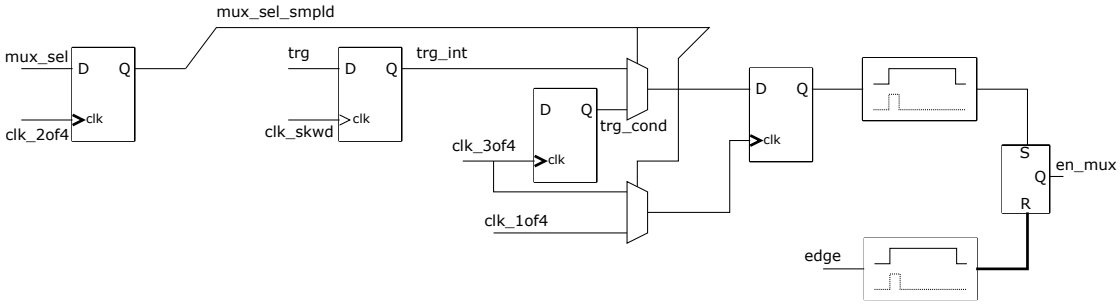


Figure 5.5: principle scheme of the synchronizer block

Figure 5.5 depicts the data-path of the described synchronizer block. Some of the output signals will be used by the phase generator to properly drive the two output channels

5.2 Phase Generator

The phase generator is the interface between the inner core of the PDU and the external world, its role is to manage the translate the multiplexer outputs, according to the operation to perform, in events which can be used as set and reset signal for the output channels. The actual phase generator, figure 5.6 is composed by two sub-blocks: the mask layer and the output channel manager. The former is a layer of AND which generates the proper mask signal that allow the next block to correctly manage the input edges

The output channel manager is implemented with an automatic tool called phaseGEN, which is able to translate a form file, a textual description of the

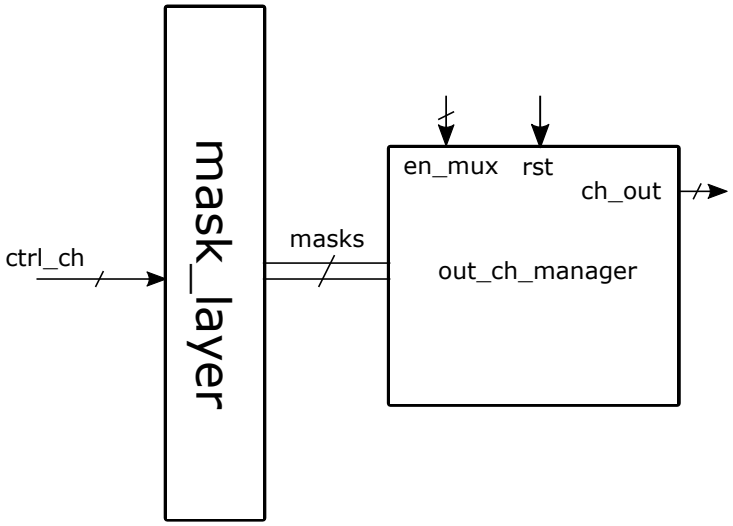


Figure 5.6: phase generator scheme: it is composed by a mask layer and the output channel manager that is implemented using the phasGEN tool

relationship between the input edges and the behavior of the output signals, into RTL, giving the end user a complete flow with which to perform not only simulations on the generated circuit, but also a full synthesis with possible DFT insertion. Below is a complete description of the working principle of the tool.

5.2.1 Non Overlapping signal generation - General Problem

In the realization of circuits with switched capacitors, there is a need to generate non-overlapping signals to drive the switches that charge and discharge the capacitors without causing undesirable charge sharing phenomena, thus ensuring a certain dead time. In general, one can imagine that the generation of a master timing event is the result of a clocked circuit that, by commutating some outputs (possibly the clock itself), causes the scheduling of commutations on the signals that control the switches in a desired phase relationship to each other and finally only in accordance with a mask signal.

Possible existing solutions are based on a level-sensitive solution, for which a circuit example can be seen in Figure 5.7.a. This solution has as general advantages the very simple and small realization (it is a classical SR latch), and all transistions

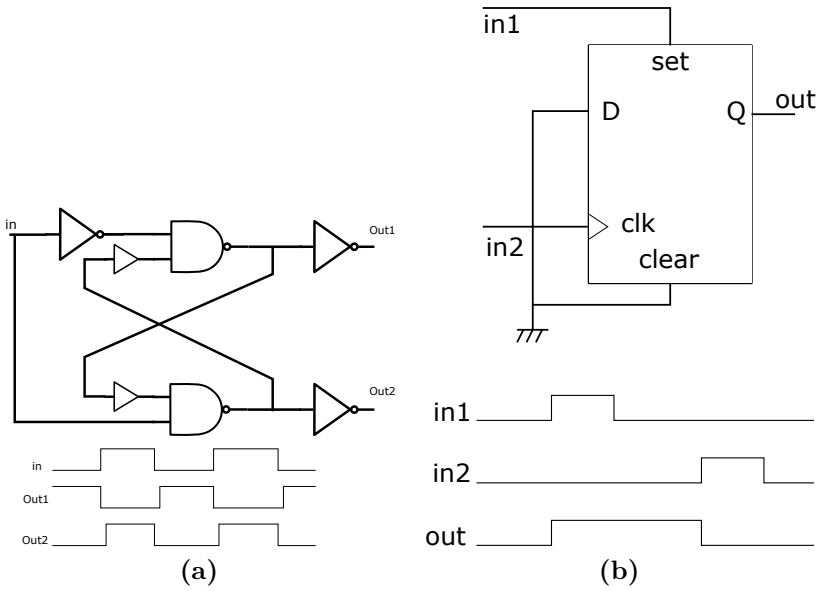


Figure 5.7: level sensitive solution (a) and mixed event solution (b)

occur as consequence of an input signal, ensuring a uniform approach. This has the consequence that no complex timing relationship between input and output can be established and the designer has to validate the circuit with analog simulations.

Another solution is based on a mixed event solution where a DFF is used: Here, the two different transistions of the output are linked to different input signals. An example of such a circuit is shown in Figure 5.7.b. The structure allows for a complex mix of events (in terms of edges and levels) that are not tied to a single input signal, and allows for HDL description so that it is possible to test and synthesize the circuit using standard digital flow. However, the circuit has some problems: there are limitations on the value of the two input signals (for example they cannot be one at the same time, in the presented implementation). Even if the circuit can be synthesized automatically, there's still need to do some manual tuning and insert buffers to achieve the desired timing ratio.

The tool is based on a different approach: it uses the edges of the input signal to control the output commutation, and generates a low pulse as a consequence of each edge. It automatically inserts the necessary buffer to add a delay to the pulses, which is eventually masked by other external signals, and finally uses these low pulses to control the set/reset latches. In this way, the input signals are treated

uniformly, considering only edge events and no timing constraints between inputs. Finally, the use of SR latches and the insertion of buffers is automatically performed by the tool.

5.2.2 Form File to RTL translation

The phaseGEN is based on the translation of a textual description between input edges and output event in RTL. The general structure of the form file has five section

1. **Technology informations:** this section lists all the information required by the synthesis tool; It's also possible to specify if the final circuit must have DFT circuitry, changing properly the synthesis scripts.

2. **Simulation informations:** here are listed all the information required by the simulation, like the delay associated to the delay element, the edge detector element or all the used logic gates. Also the data structure in which the various element will be placed is here specified, with the name of the circuit and of the associated library.

3. **Input signals:** The input signal are here listed by their name, role (reset or control signal) and then simulation information, like input delay and the period in which they are high or low. This is used to generate the test bench, which can be modified in case of more complex simulations.

4. **Internal signals:** Here are listed all the internal signal in particular defining which is the edge that sets or clear the signal, with the possibility to define multiple mask on different signal for the same event. It's can be also specified the delay between the edge and the set/clear event.

5. **Output signals:** The output signal are associated to an internal signal with the additional specification to be a simple copy of the latter or an inverted version.

Now it will be described how the tool perform the translation from a form file to an RTL description. To translate an edge in a pulse is used a simple edge detector, figure 5.8. The number of buffer, which changes the width of the pulse, can be

defined by the user. To mask an edge with a predefined signal is simply used a

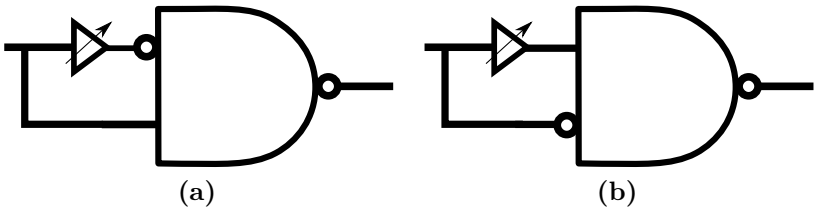


Figure 5.8: rise edge detector (a) and fall edge detector (b) implemented by the tool

nand gate with the edge negated; at last, when multiple mask on different edge are defined for a single timing event (like a set event which depends on two different edges masked by two different signals) a simple and gate is used.

For instance, the circuit required to get an internal signal, whose clear event depends on the rising edge of the input signal masked by another signal whereas the set event depends on the falling edge of the same input signal, but not masked, is depicted in figure 5.9.

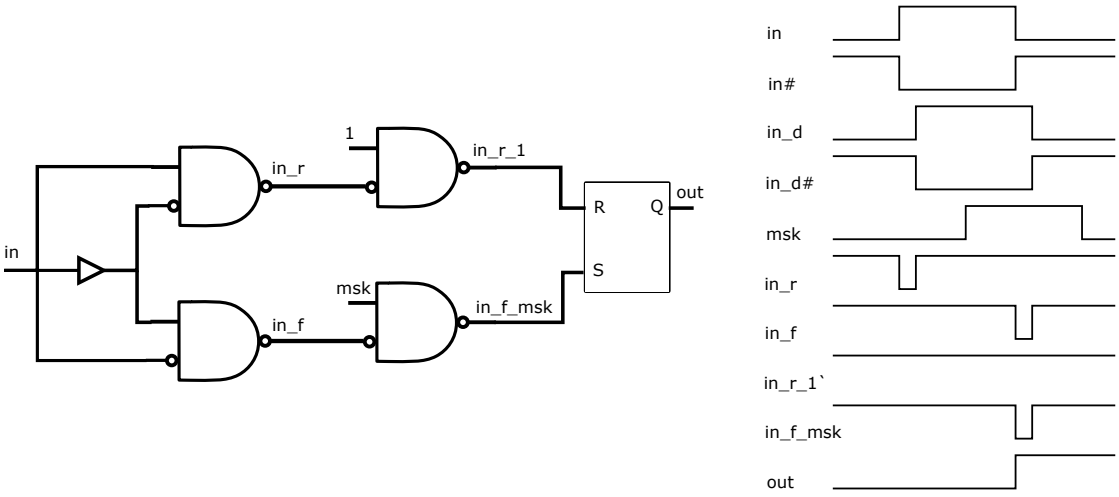


Figure 5.9: Circuit implemented by the tool for the described internal signal

The main advantages compared to a standard analog design approach consist in a significant design time reduction (the form file is very easy to be filled); furthermore, using a mixed simulation environment, no synthesis is needed to perform an early debug simulation: it is in fact our experience that a big amount of corrections and problems can be detected simulating only the RTL code.

This utility allows the use of the standard digital design flow and lets the user include its circuit into a bigger digital environment, where a scan test is eventually available; no manual adjustment is needed on the final circuit and in case a correction is needed, it is easy to put it in place by modifying a text file and run the procedure again.

5.2.3 Synthesis constraints

As described in chapter 4.3, the automatic synthesis tool, if not guided, perform an optimization on the overall circuit also inserting or removing buffer element to optimize the timing relation of the circuit.

As concern the circuit implemented by the tool, the buffer element are intentionally inserted by the user and they do not need to be optimized by the tool. To avoid that the synthesis tool has to be guided by means of a proper synthesis script: this is automatically generated by the tool and, if required, can also be predisposed to perform the DFT insertion automatically.

The circuit if synthesized following a bottom-up approach, starting from the lower level combinational block with the following order:

- Output driver (inverter/buffer) with the proper load, specified in the form file
- basic combinational block (delay buffer, 2-input NAND, ...)
- basic sequential block
- pulse generator block
- the top level

After every level of the hierarchy the script sets the *dont_touch* attribute to avoid simplification of blocks that may occur during synthesis of the top level. Also after the synthesis of each block a *set_ungroup* command is issued to have at the end of the synthesis a single hierarchy level. To make sure that the set and reset path of the SR Latch are as balanced as possible a particular set of command on those signal are issued: a minimum and maximum delay constraints are put on the path on which the set and reset signal travel. The difference between the minimum

and the maximum delay is set to be half of the delay of the single buffer which is the unit delay element.

The synthesis script produces the ddc file and, in case of DFT insertion, the scandef file, that are required if the circuit needs to be put in a more complex design as a hard macro.

5.3 DFT insertion issues

As described in section 5.1, there are multiple memory elements that use different clocks: This affects the automatic generation of scan chains by the EDA tool.

In general, a different scan chain is derived for each clock domain, which does not significantly increase the complexity of the DFT circuit (only additional output pins are needed), but significantly increases the effort required to generate test vectors and manage scan outputs (see section 4.3.1).

To force the tool to generate a single scan chain, one way is to force all flip-flops in the circuit to operate at the same clock when scan mode is required. This can be done by multiplexing the clock input of the flip-flops, as is done in the implemented circuit. For this purpose, an additional block called *clk_test_manager* is inserted into the whole circuit, as shown in Figure 5.1.

5.4 Interface Flexibility

One of the main advantages of describing the entire digital interface in RTL is the ability to make changes to the circuit without having to touch the schematic or layout manually: it can be managed by the modern EDA tools used in the standard digital flow, which, among all other capabilities, allow automatic insertion of DFT circuits.

This means that if the power system requires more than two channels or additional features need to be inserted, only changes to the RTL are required. The final circuit can then be simply re-synthesised by re-running all the steps of the standard flow, which can also be automated.

Chapter 6

Mixed Signal Simulations

The digital interface has to be validated with the analog section of the PDU, to do so classic RTL simulation cannot be performed, hence the requirement of mixed signal simulations. Mixed signal represent the necessity to perform simulation in a mixed environment where both digital and analog circuit are present.

This is possible thanks to the capability of the modern EDA tool, in particular the focus will be posed on the cadence suite and the Questa adms simulator, which allow to import digital circuit described in HDL and associate them to a technology library which can be managed and simulated by the virtuoso environment.

The following section will describe the steps required to setup the environment to perform the mixed signal simulation and will comment on the results of the performed simulations.

6.1 Environment Setup

To validate the circuit the PDU with the designed digital interface is compared with the result from other two circuit

- **Fully Analog implementation:** the circuit implement the digital section by means of a virtuoso schematic
- **RTL model:** it is an HDL description of the circuit with the PDU implemented by means of an RTL behavioural model, hence it is a fully digital design.

To use RTL blocks inside of the virtuoso environment, and then perform the mixed signal simulations, it is required to import the compiled digital library, hence allowing virtuoso to generate a symbol view of these blocks which can be used as elements in a schematic.

The first step is to define the A/D-D/A converters, which allow the digital signal to be converted in analog quantities that can be used by the analog blocs, and vice-versa: basically they allow the two world to communicate properly. This is done by defining some custom converters whose characteristics depends on the system that has to be implemented (namely their characteristics depends on the required power supply and wanted threshold to distinguish between a logical 0 or 1).

After that the RTL design has to be compiled in such a way to generate the *modelsim.ini* file that will be used by the *Artist link* to read and import the digital libraries (appendix E presents an example of a script used to generate the file); After that it will be possible to import the digital design with virtuoso in a preexisting library, generating a cell with two views: an adms view and a symbol one. The latter can be used in a schematic so to perform simulations with the *ADE Explorer*.

Finally, to perform the mixed signal simulation it is required to generate the schematic of the testbench in which three design are inferred: the two previously described reference circuit and the fully analog implementation with the digital section substituted by the designed digital interface (precisely, with the verilog netlist obtained after the synthesis phase). The latter is the circuit that has to be validated.

6.2 Results

Figure 6.1 presents the results of the mixed signal simulation, useful to verify that the general behavior of the unit is correct: namely, if the two output channels behaves the same way for the three described versions of the circuit

It can be seen how the circuit with inferred the designed digital interface follows exactly the behavior of the RTL Model and of the Fully Analog implementation, as expected.

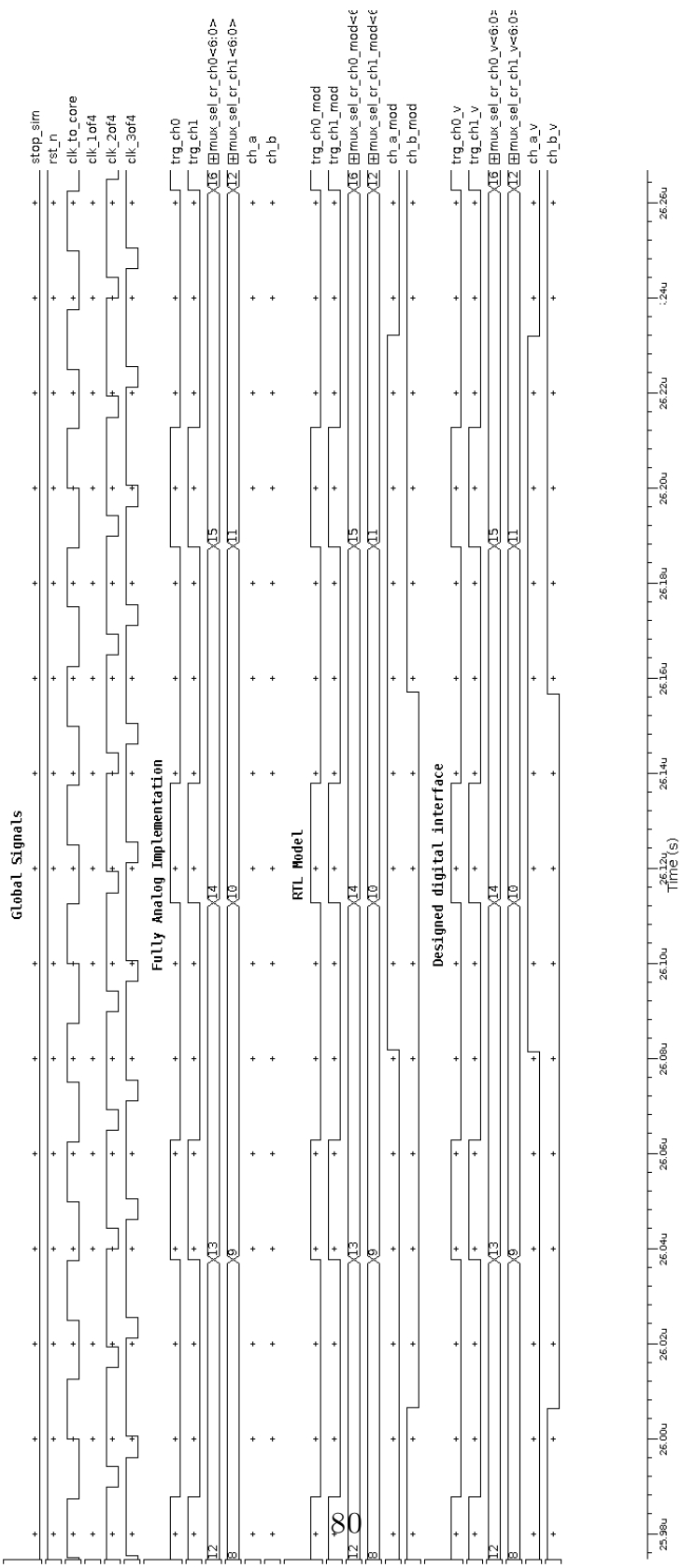


Figure 6.1: results of the mixed signal simulations

Chapter 7

Conclusions

The work focused on the description of the state-of-the-art design of converters and control loops, both in analog and digital domains. In this context, a new approach to achieving high time resolution is described, based on the Programmable Delay Unit.

The presented digital interface manages the PDU and overcomes the described timing issues, enabling performance comparable to SoC implementation (about tens of picoseconds), using a much lower clock frequency. The final system can drive power stages implemented with GaN power devices.

It is suitable for a monolithic implementation, thanks to the capability of the High Voltage BCD, which allows the realization of a complete gate driver on a single die.

In this way, the chip area reduces without any adverse impact on performance and power losses and instead offers the possibility of implementing a more robust and cost-effective solution.

The system described in RTL can be easily modified and complicated by changing the high-level HDL description, taking advantage of the standard digital flow and modern synthesis tools. These also offer the possibility of automatically inserting DFT functionality.

Finally, mixed-signal simulations can be performed which allow validating the circuit considering the actual implementation of the analog section of the PDU and not only a model.

Appendix A

DCM M(D,K) derivation

This appendix describe the whole computation to retrieve the conversion ratio in DCM mode for the buck converter in figure 2.23. With few modifications, the same techniques developed in section 2.2.1 for the steady state analysis in CCM can be applied to the DCM mode.

Inductor volt-second balance and capacitor amp-second balance must be applicable regardless the operating mode. This do not apply to the small ripple approximation:

- Output capacitor voltage ripple: since it is required that the output voltage ripple remains small no matter the operating condition, the linear ripple approximation holds for the capacitor voltage

- Inductor current ripple: by definition in DCM the current ripple is not small and it may be that Δi_L is bigger than the DC component, hence it cannot be neglected.

When the transistor conducts, for $0 < t < D_1 T_s$, the inductor voltage and capacitor current are given by equations A.1.

$$v_L(t) = V_g - v(t) \approx V_g - V \tag{A.1}$$

$$i_C(t) = i_L(t) - \frac{v(t)}{R} \approx i_L(t) - \frac{V}{R}$$

applying the small ripple approximation, only the $v(t)$ can be approximated by its DC component, meanwhile for the inductor voltage it must be considered.

The diode conducts in the second subinterval $D_1T_s < t < (D_1 = D_2)T_s$, and the inductor voltage and capacitor current will be given by equation A.2.

$$v_L(t) = -v(t) \approx -V \tag{A.2}$$

$$i_C(t) = i_L(t) - \frac{v(t)}{R} \approx i_L(t) - \frac{V}{R}$$

The diode becomes reverse biased at $t = (D_1 + D_2)T_s$, and both the transistor and diode are now off. The inductor voltage and current are both zero for the remainder of the switching period $(D_1 + D_2)T_s < t < T_s$. The network equations in this case are given by equations A.3.

$$v_L = 0 \ , \ \ i_L = 0 \tag{A.3}$$

$$i_C(t) = i_L(t) - \frac{v(t)}{R} \approx -\frac{V}{R}$$

Note that the inductor current is constant and equal to zero during the third sub interval, and therefore the inductor voltage must also be zero in accordance with the relationship $vL(t) = LdiL(t)/dt$. In practice, parasitic ringing is observed during this sub interval. This ringing occurs owing to the resonant circuit formed by the inductor and the semiconductor device capacitances, and typically has little influence on the converter steady-state properties. The inductor voltage

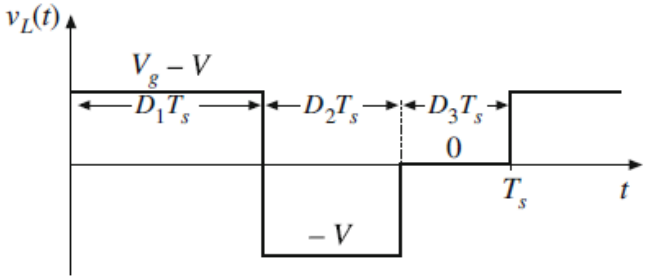


Figure A.1: Inductor voltage waveform $vL(t)$, buck converter operating in discontinuous conduction mode [2]

waveform of the buck converter operating in DCM is depicted in figure A.1. The

DC component can be easly evaluated as in equation A.4.

$$< vL(t) >= D_1(V_g V) + D_2(V) + D_3(0) = 0 \tag{A.4}$$

Hence, solving for V the expression for the conversion ratio is got, equation A.5.

$$V = V_g \frac{D_1}{D_1 + D_2} \tag{A.5}$$

The transistor duty cycle D (which coincides with the first sub interval of duty cycle D_1) is the control input to the converter, and can be considered known. But the second sub interval with duty cycle D_2 is unknown, and hence another equation is needed to eliminate D2 and solve for the output voltage V. Using the

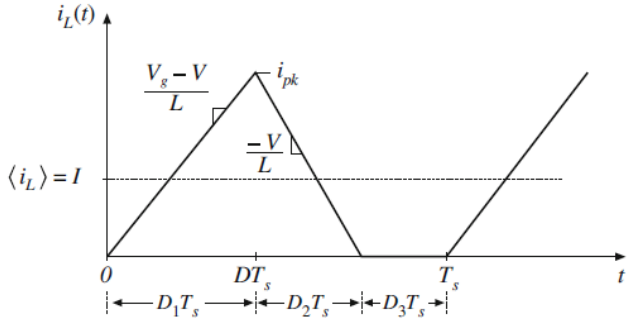


Figure A.2: Inductor current waveform $i_L(t)$, buck converter operating in discontinuous conduction mode [2]

capacitor charge balance, applying KCL at the node which connects capacitor, output resistance and inductor equation A.6 is obtained, and since by capacitor balance the DC component of the capacitor current is zero, hence all the DC load current must be supplied to the load by the other element connected to the node: the inductor. Since the inductor current ripple is not small, the precise inductor current waveform must be considered, sketched in figure A.2.

$$i_L(t) = i_C(t) + \frac{V(t)}{R} \tag{A.6}$$

The current begins the switching period at zero, and starts to rise in the first sub interval with a constant slope, given by the applied voltage divided by the

inductance. The peak current is the slope multiplied by the length of the first sub interval, equation A.7.

$$i_L(D_1T_s) = i_{pk} = \frac{V_g - V}{L} D_1 T_s \tag{A.7}$$

The DC component is the average value of $i_L(t)$, which can be evaluated as the area under the curve that is the area of a triangle with height i_pk and base $(D_1 + D_2)T_s$, hence

$$\int_0^{T_s} i_L(t)dt = \frac{1}{2} i_p k (D_1 + D_2) T_s \tag{A.8}$$

Substituting A.8 in the average value formula and knowing that, since the capacitor

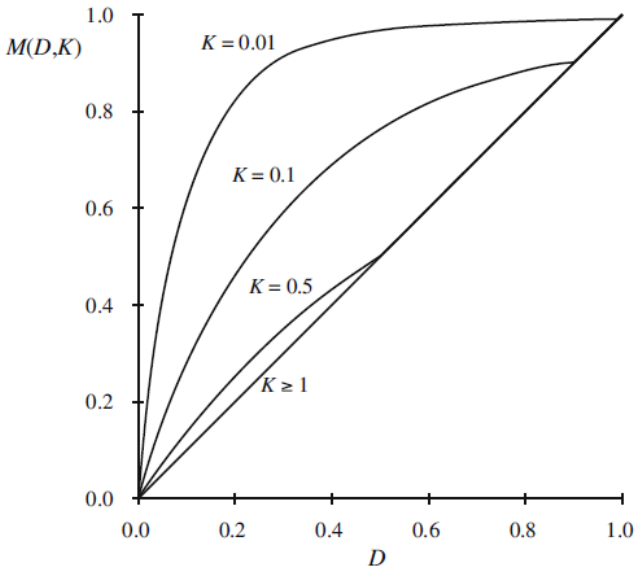


Figure A.3: Voltage conversion ratio M for a buck converter in DCM [2]

DC component is zero, the inductor DC component is $< i_L >= V/R$, equation **??** is obtained.

$$\frac{V}{R} = \frac{D_1 T_s}{2L} (D_1 + D_2)(V_g - V) \tag{A.9}$$

This equation can be used to finally compute the conversion ratio, by finding D_2

and substituting in equation A.5.

$$\frac{V}{V_g} = \frac{2}{1 + \sqrt{1 + \frac{4K}{D_1^2}}} \tag{A.10}$$

where $K = 2L/RT_s$ and the equation is valid for $K < K_{crit}$. Figure A.3 depicts the behavior of the conversion ration which now depends on two factors.

Appendix B

Converter Transfer function derivation

Since the buck converter is a non-linear time-variant system, is necessary to find a linear time-invariant model. The procedure here followed is described in [2] and is known as *average switch modeling*, its central idea is to find an averaged circuit model for the switch network and the resulting network can be inserted in the converter circuit to obtain a complete averaged circuit model of the converter, with the advantage that it can be used in different converter configuration with ease. Considering the buck converter in figure **??**, the input current $i_1(t)$ and the

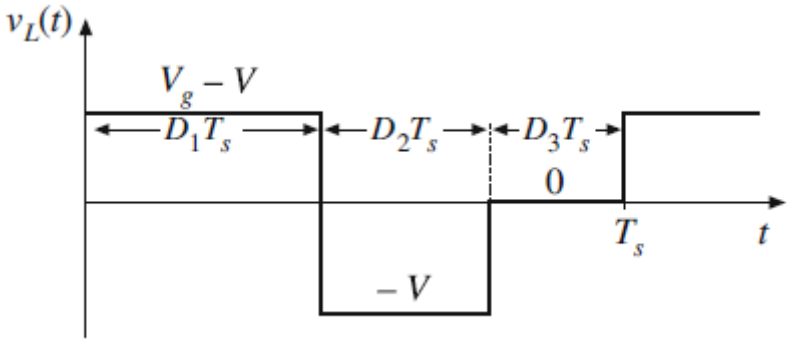


Figure B.1: Buck converter implemented with real switches [2]

switching node voltage $v_2(t)$ are shown in figure B.2. The input current is equal to the inductor current when the transistor conducts, that is during the first sub

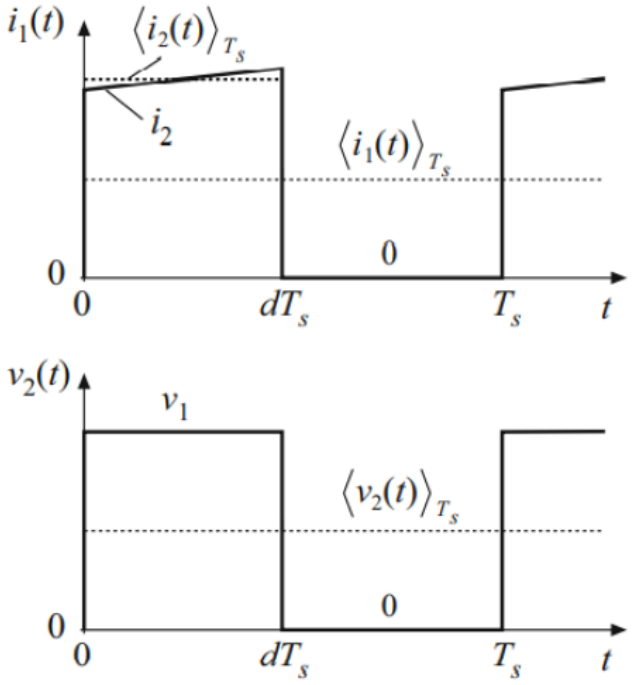


Figure B.2: Input current and switching node voltage waveforms [2]

interval DT_s, while its 0 in the second sub interval. Neglecting the diode voltage drop, the switching node voltage is equal to the input voltage during the first sub interval, 0 in the second.

The switching network, composed by the transistor and the diode, can be considered as a two port network: considering $i_2(t)$, the current after the switching network, and $v_1(t)$, the input voltage, as independent variable and by evaluating the average of these quantities over one switching period.

$$< i_1(t) >_{T_s} = d(t) < i_2(t) >_{T_s} \tag{B.1}$$

$$< v_2(t) >_{T_s} = d(t) < v_1(t) >_{T_s}$$

with d(t) duty cycle, also named control input. To obtain a linear two-port model for the switch, perturbation and linearization of the average converter waveforms around a quiescent point must be performed. Then expressing the DC component with capital letters and the AC with hat symbol, all the involved quantity can be

written as in equation B.2.

$$x(t) = X + \hat{x(t)} \tag{B.2}$$

Thus, equations B.1 can be written as in B.3.

$$I_1 + \hat{i_1(t)} = D(I_2 + \hat{i_2(t)}) + I_2\hat{d(t)} \tag{B.3}$$

$$V_2 + \hat{v_2(t)} = D(V_1 + \hat{v_1(t)}) + V_1\hat{d(t)}$$

The two equations describe a linear time-invariant two port block that can replace the switching network in the buck converter topology, as depicted in figure B.3. The average switching network perform the function of:

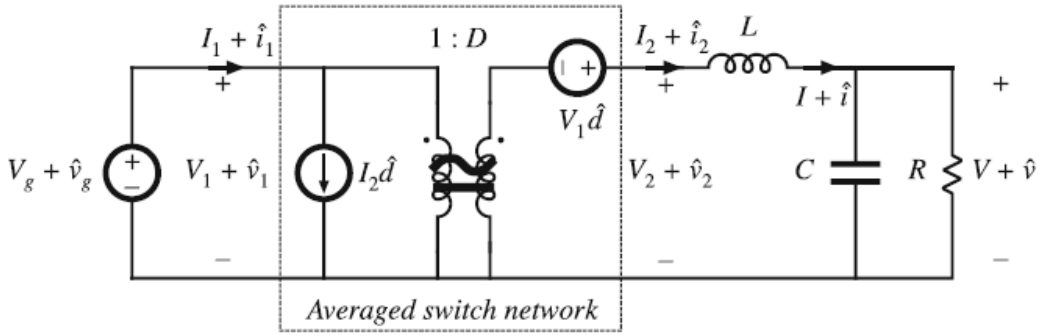


Figure B.3: Buck converter topology with the averaged switching network [2]

1. transformation of DC and small-signal ac voltage and current levels according to the $1 : D$ conversion ratio

2. introduction of ac voltage and current variations into the converter circuit, driven by the control input $d(t)$.

Finally, it is possible to find the converter transfer function (taking also into account the capacitor Equivalent Series Resistance ESR).

The control input to output transfer function (i.e. the transfer function from the control input $d(t)$ to the output voltage) can be computed by superposition.

Denoting the input voltage as $v_in(t)$ and the ESR as R_C:

$$\hat{v_{out}} = \frac{R||(R_C + \frac{1}{sC})}{sL + R||(R_C + \frac{1}{sC})}\hat{d}V_{in} \tag{B.4}$$

hence dividing by $\hat{d}$, the $G_{vd}(s)$ is obtained.

$$G_{vd}(s) = \frac{V_{in}(1 + sCR_c)}{1 + s(\frac{L}{R} + CR_C) + s^2LC(1 + \frac{R_C}{R})} \tag{B.5}$$

Hence the buck converter can be considered as a second order system with two complex conjugate poles (due to the LC network) and a zero (that comes from the ESR). With similar steps the line to output transfer function,$G_{vg}(s)$,can be obtained, as well as the output impedance $Z_{out}(s)$.

$$G_{vg}(s) = \frac{D(1 + sCR_c)}{1 + s(\frac{L}{R} + CR_C) + s^2LC(1 + \frac{R_C}{R})} \tag{B.6}$$

$$Z_{out}(s) = (sL||R)||(R_c + \frac{1}{sC}) \tag{B.7}$$

Appendix C

Control Loop transfer function derivation

C.1 Compensator transfer function

The general structure of a compensator, figure C.1, consists of an Error Amplifier (EA) with a compensation network given by Z_F and Z_i (that's also the reason why analog control loop are more cumbersome and costly, since more components and pin are require to implement such networks). Compensator are used to stabilize the feedback loop by setting the poles trough their transfer functions.

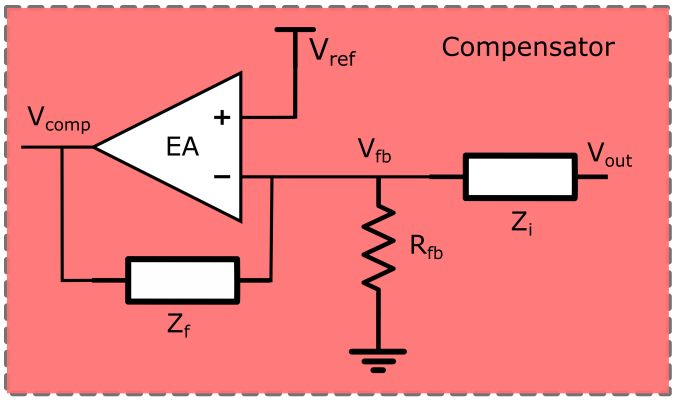


Figure C.1: Compensator Schematic

There are three type of compensation network with different complexity. The

described Analog voltage-mode control uses a type-3 network, which transfer function is given by C.1.

$$G_c(s) = \frac{R_2}{R_1}\frac{1}{s}\frac{(1+\frac{\omega_{z_1}}{s})(1+\frac{s}{\omega_{z_2}})}{(1+\frac{s}{\omega_{p_1}})(1+\frac{s}{\omega_{p_2}})} \tag{C.1}$$

Which, by properly setting the values of the network components, can realize a PID transfer function with the two zeroes ω_{z_1} and ω_{z_2}, the two poles ω_{p_1} and the pole in the origin, meanwhile ω_{p_2} will be an high frequency roll-off pole.

C.2 Modulator transfer function

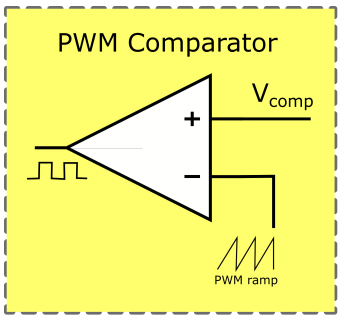


Figure C.2: Modulator schematic

The modulator, figure C.2transfer function F_m can be computed as follows:

$$F_m = \frac{\Delta d}{\Delta v_{comp}} = \frac{1}{v_{ramp}} \tag{C.2}$$

with v_{ramp} the amplitude of the sawtooth signal, v_{comp} the compensator output and d, the duty cycle, is the modulator output.

The modulator transfer function is a simple gain term, if the amplitude of the ramp is equal to one, the whole transfer function collapse to 1.

Appendix D

Timing Constraints

Before the compile phase, the Synopsys Design Compiler allows specifying timing constraints that describe, among others, types and relations between clocks or input and output delays. This appendix will describe how these constraints can be specified within the tool to guide the synthesis phase toward the desired design specification. The informations here reported follow the Synopsys design compiler workshop [9]

The default environment considered by Synopsys is synchronously-clocked and considers that input data arrives from a positive edge clocked device and output data goes to a positive edge clocked device.

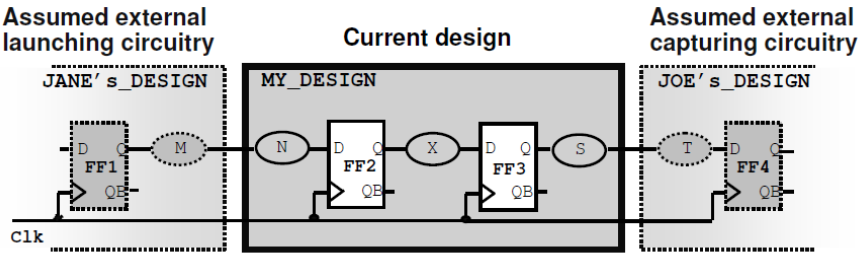


Figure D.1: The Design that will be taken as example during the timing constraints definition [9]

Synopsys DC breaks the design into *Timing paths* with the starting point that can be either an input port or the clock pin of a Flip-Flop register, and the ending point an output port or any input of a sequential device.

Then on these paths perform a Static Time Analysis which determines if a circuit meets the timing constraints during and after synthesis. This is based on three steps: the design is broken into timing paths, for each path evaluates the corresponding delay, each path delay is compared with the expected arrival times to verify if it matches the specification.

Let us consider the *Register-to-Register* path. These need to be characterized by the clock period and, in the case of a device that works with different edges, also the duty cycle. The Setup time of the Flip-Flop is retrieved by the technology libraries. To specify a clock period the following syntax is used:

```
create_clock -period n [get_ports CLK]
```

The unit time is defined by the technology library. It is possible to specify a different duty cycle by using the `-waveform` option, if this is not specified a 50% duty-cycle waveform will be considered by default. Defining the clock in a single-clock design constraints all register to register path for single-cycle setup time.

The clock network is considered ideal with infinite drive capability, zero fall/-transition time, zero skew, and zero insertion delay or latency. All these parameters must be modeled to have an accurate representation of the clock behavior. This is because the design compiler is not used to synthesize clock buffer trees, which is usually done by layout tools with load balancing or Clock Tree Synthesis (CTS).

To model delay difference between clock network branches, known as clock skew, it is invoked the following command:

`set_clock_uncertainty -setup/-hold` T_s `[get_ports CLK]`

The set value can represent also clock jitter and margin effect, it is a unique term that embeds all three components.

If no `-setup` or `-hold` options are specified the uncertainty affects both of them. The options allow to synthesize a design, e.g. with extra setup time without affecting the hold time. Another aspect is how to model the delay associated with the clock network, this is divided into two contributions, as depicted in figure D.2. The network latency models the average internal delay from the `create_clock` pin port to the clock pin of the register, the source latency models the delay from the actual port that generates the clock to the `create_clock` pin.

The syntax to model the delay is the following:

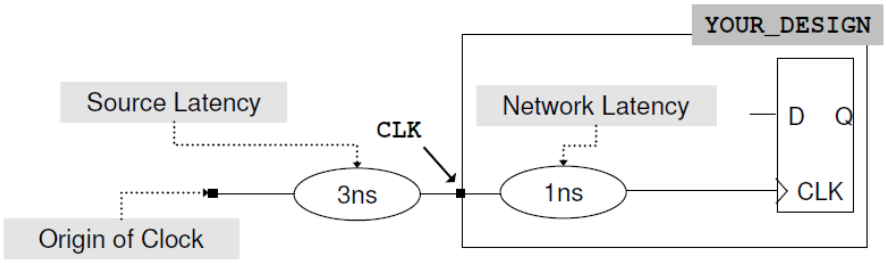


Figure D.2: Synopsys gives the possibility to model the clock latency setting two contribution which represent the external and internal delay network separately [9]

```
set_clock_latency -source -max T_l [get_ports CLK]
```

if no `-source` option is used, it set the network delay, it is possible to model rise, fall, min, and max delay by using the proper option.

Finally, it is necessary to properly constrain the design by setting the transition time of the clock, which can be different for the rise or the fall time. It is set invoking the following command:

```
set_clock_transition [-rise/-fall] T_t [get_ports CLK]
```

For post-CTS static timing, analysis is necessary to use the `set_propagated_clock` which forces the analyzer to calculate the actual clock tree skew, latency, and transition time for the specified clock. The post-CTS constraints, therefore, do not include ideal transition and network latency commands. If the uncertainty number used during synthesis includes jitter and/or margin, these effects must still be included in the post-CTS analysis, along with the external source latency.

To constrain input and output path instead, different type of constraints has to be set. In particular, for the input paths in addition to clock information it is necessary to define the arrival delay which represent the amount of delay denoted by M in figure D.1. The syntax to set the input delay is the following:

```
set_input_delay -max T_d -clock clk_name [get_ports CLK]
```

For the output path in addition to the clock definition it is necessary to define the latest arrival time of the data at the output port, namely the delay denoted by S in figure D.1. The syntax is similar to the one used for the input delay:

`set_output_delay -max` T_d `-clock clk_name [get_ports CLK]`

In the design is possible that some of the paths may have to be constrained equally, or that only some paths have different constraints than the rest of the circuit. In this case is possible to force the tool to remove some pins from the collection where the constraints are being set, for example, to remove the clock input pin in the collection of pins that are affected by the `set_input_delay`:

```
set_input_delay -max T_d -clock clk_name \
[remove_from_collection [all_inputs] [get_ports CLK]]
```

Up to now, only sequential circuit were considered, in case of pure combinational circuit, like in figure **??**, there's no input pin: the solution is to generate a virtual clock. It is a clock not connected to anything, serves only as a reference for input

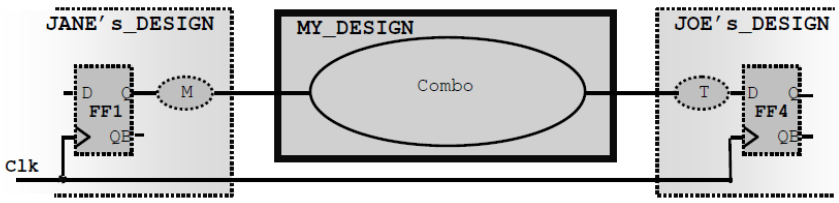


Figure D.3: Example of design with a pure combinational circuit [9]

and output delays. It uses the same syntax previously described to create a clock, but it must be set with a name and without a port reference.

`create_clock_-name VCLK -period` T_p

After setting the timing constraints it is possible to compile the design, to verify the results some reports can be generated. To this aim the `report_timing`. The command breaks the design into timing paths and analyzes each path for single-cycle max-delay timing generating a default four-section report that includes: One path per path group, the worst violator which can be defined as the path with the largest negative slack or Worst-Negative-Slack (WNS); maximum delay or setup timing only (no hold timing information, no area, no DRC).

Appendix E

Mixed signal setup scripts

The following listing shows the general structure required to generate the correct modelsim.ini, with references to all the digital libraries required to perform a mixed-signal simulation.

```
#!/bin/tcsh -f
set path_file=$PWD

# Environment Setup
rm -rf $path_file/compiled_libraries/
rm -rf modelsim.ini
mkdir -p $path_file/compiled_libraries/digital_lib/
mkdir -p $path_file/compiled_libraries/log/

# read source and technology file
valib $path_file/compiled_libraries/digital_lib/pdu_digital_lib

vcom -work hires_tdc_top_lib $path_file/digital_source/code.vhd >> ./
    compiled_libraries/log/source.log

valog -work pdu_digital_lib $path_file/digital_source/technology_file
    .v >> ./compiled_libraries/log/tech.log

valog -work pdu_digital_lib $path_file/digital_source/
    gate_level_netlis.v >> ./compiled_libraries/log/netlist.log
```

Acknowledgements

List of Tables

List of Figures

Bibliography

[1] C Cini, C Contiero, C Diazzi, P Galbiati, and D Rossi. «A New Bipolar, CMOS, DMOS Mixed Technology for Intelligent Power Applications». In: *15th European Solid–State Device Research Conference, ESSDERC*. 1985 (cit. on pp. vi, 3).

[2] R.W. Erickson and D. Maksimović. *Fundamentals of Power Electronics*. Springer International Publishing, 2020. ISBN: 9783030438814 (cit. on pp. vi, 13–15, 17, 18, 20–23, 25, 26, 28, 29, 33, 34, 39–42, 44–46, 83–85, 87–89).

[3] Juri Giovannone Luca Iannelli Alessandro Cabrini. *MODEL OF AN ADVANCED DIGITAL CONTROL LOOP FOR DC-DC CONVERTER*. 2020 (cit. on pp. vi, 25, 38).

[4] Dragan Maksimovic, Regan Zane, and Robert Erickson. «Impact of digital control in power electronics». In: *Proc. IEEE Int. Symp. Power Semicond. Devices ICs*. 2004, pp. 13–22 (cit. on pp. vi, 40, 43).

[5] Benjamin J Patella, Aleksandar Prodic, Art Zirger, and Dragan Maksimovic. «High-frequency digital PWM controller IC for DC-DC converters». In: *IEEE Transactions on Power electronics* 18.1 (2003), pp. 438–446 (cit. on pp. vi, 43).

[6] Asif Syed, Ershad Ahmed, and Dragan Maksimovic. «Digital PWM controller with feed-forward compensation». In: *Nineteenth Annual IEEE Applied Power Electronics Conference and Exposition, 2004. APEC'04.* Vol. 1. IEEE. 2004, pp. 60–66 (cit. on pp. vi, 43).

[7] Keyue M Smedley and Slobodan Cuk. «One-cycle control of switching converters». In: *IEEE transactions on power electronics* 10.6 (1995), pp. 625–633 (cit. on pp. vi, 43).

[8] A. MOLINA and Oswaldo Cadenas. «Functional verification: Approaches and challenges». In: *Latin American applied research Pesquisa aplicada latino americana = Investigación aplicada latinoamericana* 37 (Jan. 2007) (cit. on pp. vi, 52).

[9] *Design Compiler 1, Workshop - Student Guide*. Synopsys, Inc. Mountain View, California (cit. on pp. vi, 93, 95, 96).

[10] Ken Jaramillo and Subbu Meiyappan. «10 tips for successful scan design: Part one.» In: *EDN* 45.4 (2000), pp. 67–74 (cit. on pp. vi, 60).

[11] Ken Jaramillo and Subbu Meiyappan. «10 tips for successful scan design: Part two.» In: *EDN* 45.4 (2000), pp. 77–84 (cit. on pp. vi, 60).

[12] Himanshu Bhatnagar. *Advanced ASIC Chip Synthesis: Using Synopsys® Design CompilerTM Physical CompilerTM and PrimeTime®*. Springer Science & Business Media, 2007 (cit. on pp. vi, 59).