Master's Thesis of Science

MA646

Low Power 10 bit DAC Design for Video-Encoder

Günther Langmann

Integrated Circuit Design

Low Power 10 bit DAC Design for Video-Encoder

Master's Thesis

at

Graz University of Technology

submitted by

Günther Langmann

Institute of Electronics (IFE),
Graz University of Technology
A-8010 Graz, Austria
Head: Univ.-Prof. Dipl.-Ing. Dr.techn. Wolfgang Pribyl

September 19, 2011

Advisor: Ass.-Prof. Dipl.-Ing. Dr.techn. Peter Söser

TU Graz

Integrierter Schaltungsentwurf

Energiesparender 10 bit DA-Umsetzer für Videocodierer

Masterarbeit

an der

Technischen Universität Graz

vorgelegt von

Günther Langmann

Institut für Elektronik (IFE),
Technische Universität Graz
A-8010 Graz
Leiter: Univ.-Prof. Dipl.-Ing. Dr.techn. Wolfgang Pribyl

19. September 2011

Diese Arbeit ist in englischer Sprache verfasst.

Begutachter: Ass.-Prof. Dipl.-Ing. Dr.techn. Peter Söser

TU Graz

Abstract

A low power 10 bit digital to analog converter (DAC) is presented in this work. The field of application is the conversion of digital video information delivered by a video-encoder. The converted analog output signal is compliant to PAL and NTSC analog video signals and meets the requirements of the corresponding ITU-Recommendations.
To match the claim of a fast and energy-saving DAC, a $0.18\,\mu\mathrm{m}$ process is used. The chosen DAC architecture is a segmented current steering DAC. Supplied by a typical power supply of $1.8\,\mathrm{V}$ a maximum of $35.7\,\mathrm{mA}$ output current can be driven. Transmitted over a double terminated $75\,\Omega$ coaxial wire an output voltage of $1.34\,\mathrm{V}$ is generated. The maximum clock frequency of the design is $27\,\mathrm{MHz}$ (2x oversampling) and the whole DAC consumes less than $70\,\mathrm{mW}$.

Keywords: low power integrated circuit, segmented current steering digital to analog converter (DAC), current mirror, current source, video-encoder, layout.

Kurzfassung

In dieser Arbeit wird ein energiesparender 10 bit Digital-zu-Analog Umsetzer vorgestellt. Sein Anwendungsgebiet ist die Umwandlung von digitalen Videoinformationen, die von einem Videocodierer bereitgestellt werden. Das umgesetzte Analogsignal ist konform zu anlogen PAL und NTSC Videosignalen und entspricht auch den Normen der ITU.
Um die Anforderung eines schnellen und energiesparenden Umsetzers zu erreichen wird ein $0{,}18\,\mu\mathrm{m}$ Prozess verwendet. Die gewählte Umsetzterarchitektur entspricht der eines segmentierten stromgesteuerent Digital-zu-Ananlog Umsetzers. Bei einer typischen Versorgung von $1{,}8\,\mathrm{V}$ kann ein maximaler Strom von $35{,}7\,\mathrm{mA}$ getrieben werden. Wird dieser Strom über eine doppelt abgeschlossene $75\,\Omega$ Koaxialleitung übertragen, dann ergibt dies eine Ausgangsspannung von $1{,}34\,\mathrm{V}$. Die Schaltung ist für eine maximale Taktrate von $27\,\mathrm{MHz}$ (2x Überabtastung) ausgelegt und der gesamte Umsetzter benötigt weni-ger als $70\,\mathrm{mW}$.

Schlagwörter: energiesparender integrierter Schaltungsentwurf, segmentierter stromgesteuerter Digital-zu-Analog Umsetzer, Stromspiegel, Stromquelle, Videocodierer, Layout.

Pledge of Integrity

I hereby certify that the work presented in this thesis is my own, that all work performed by others is appropriately declared and cited, and that no sources other than those listed were used.

Place: _____

Date: _____

Signature: _____

Eidesstattliche Erklärung

Ich versichere ehrenwörtlich, dass ich diese Arbeit selbständig verfasst habe, dass sämtliche Arbeiten von Anderen entsprechend gekennzeichnet und mit Quellenangaben versehen sind, und dass ich keine anderen als die angegebenen Quellen benutzt habe.

Ort: _____

Datum: _____

Unterschrift: _____

To Heike,
and my children
Annika and ?

Contents

List of Figures

List of Tables

List of Abbreviations

BNC	Bayonet Neil-Concelman connector
CAD	computer aided design
CMOS	complementary metal oxide semiconductor
DAC	digital to analog converter
DC	direct current
DNL	differential nonlinearity
DR	dynamic range
DRC	design rule check
DSA	dynamic signal acquisition
DVE	digital video encoder
EDA	electronic design automation
ERC	electrical rule check
ESD	electrostatic discharge
FET	field effect transistor
FG	function generator
FS	full scale
FSR	full scale range
GDS	graphic database system
GDSII	graphic database system 2nd version
GUI	graphical user interface
HDMI	high-definition multimedia interface
HSDIO	high speed digital input output
IC	integrated circuit
INL	integral nonlinearity
ITU	International Telecommunication Union
LCD	liquid crystal display
LSB	least significant bit
LVS	layout versus schematic
MOS	metal oxide semiconductor
MOSFET	metal oxide semiconductor field effect transistor
MSB	most significant bit
NI	National Instruments
NMOS	n-channel metal oxide semiconductor
NTSC	National Television System Committee
PAL	Phase Alternating Line
PC	personal computer
PCB	printed circuit board
PCI	peripheral component interconnect
PMOS	p-channel metal oxide semiconductor
PSRR	power supply rejection ratio

PXI	PCI eXtensions for Instrumentation
SH	sample and hold
SNR	signal to noise ratio
SQNR	signal to quantization noise ratio
TCL	tool command language
TV	television

Acknowledgements

First I want to thank Mario Manninger to give me the chance to do this work at austriamicrosystems AG. Another special thank on austriamicrosystems side to Herbert Lenhard. He acts as a mentor during the thesis and spent hours of helpful discussions, support, and coffee breaks for me.
On university side I want to thank Peter Söser for his support and patience.
Last but not least, I want to thank my family. Especially my girlfriend Heike for her love, understanding and tolerance. I also want to thank her, that she did not take the advice of her grandfather to keep the hands off musicians and students. Finally, I wish to thank my little daughter Annika for playing in the garden and sleeping, especially some hours during the day, to make it possible to finalize this thesis within some years.

Günther Langmann
Graz, Austria, September 2011

Credits

I would like to thank István Csongor Ungvári, Technical University of Cluj-Napoca, Romania, for using his measurement results of the video DAC in chapter 9. Measuring and analyzing the analog part of the video encoder test-chip was the topic of his bachelor's thesis, which he did at the Technical University of Cluj-Napoca and Graz University of Technology [Ungvári, 2010].
Furthermore, this thesis was written using the LaTeX skeleton thesis from Keith Andrews [Andrews, 2006].

Chapter 1

Introduction

This thesis describes a low power 10 bit digital to analog converter (DAC) designed for a video-encoder. The output of the DAC is complient to PAL and NTSC analog video signals and meets the requirements of the corresponding ITU-Recommendations. The DAC is built for converting an 8 bit BT.656 video signal, generated by the multimedia platform series AS35XX from austriamicrosystems AG, through an 8 to 10 bit video-encoder to a PAL or NTSC analog output signal. The output current is converted to an output voltage at the resistors of the double terminated coaxial wire. The required maximum output voltage for PAL signal is $1.34\,\mathrm{V}$. The coaxial wire is terminated with $75\,\Omega$ on each end, and therefore the visible resistance at the output is $37.5\,\Omega$. This leads to a maximum output current of $35.7\,\mathrm{mA}$. One of the requirements given by austriamicrosystems AG is a DAC without the necessity of an additional driver at the output. This is a hard requirement for the video DAC, because with a supply voltage of $1.8\,\mathrm{V} \pm 5\%$ and a maximum output voltage of $1.34\,\mathrm{V}$, only a few hundred mV are available for the output path of the DAC. The small supply voltage is given from the used $0.18\,\mu\mathrm{m}$ process, which has a typical supply voltage of $1.8\,\mathrm{V}$. Another important criterion is the $2x$ oversampling of the PAL signal, which results in a maximum clock frequency of $27\,\mathrm{MHz}$.

With the above specification at the back of one's mind, the basic characteristics and architectures of DACs are described in chapter 2. All types of DACs have some basics together, which can be found at the beginning of chapter 2. Especially the characteristics, for comparing the different DAC architecture, are all the same for different DAC types. The second half of chapter 2 illustrates the different DAC architectures, where the basic concepts are explained.

Due to the above specifications, a current steering DAC architecture is used. The accuracy of 10 bit is no limit for current steering DACs. The advantage of these DACs is the fast switching, because the differential output current is switched between the two output loads. Chapter 3 analysis the current steering DAC in detail, with its current source unit as main building block. Additionally different decoding schemes are discussed.

After clarifying the basics of DACs and especially current steering DACs another important chapter with basics follows. Chapter 4 deals with the basics of the physical representation of integrated circuits. The chapter discusses the layout design fundamentals generally. The first sections of chapter 4 presents basics about on-chip parasitics and matching structures. The second part has a more practical approach, starting with some lines about layout floorplanning. The following sections explain the process design rules and their verification. However, not only the compliance to the design rules is important, but rather the correct conversion of the design to the physical layout has to be verified. The tools for the design rule and layout versus schematic checks are shown in the last section of this chapter.

After that the theoretical part of the work is closed and the practical implementation starts. Chapter 5 deals with the design implementation. At the beginning a clear definition and summary of all specifications can be found. Followed by the concept for the current steering DAC implementation supported by a block diagram. After some basic calculations every single block is described. Additionally, the calculation of the first approach with the simplified square-law model is done for the building blocks.

The chapter is finalized by some lines about the used digital decoding schemes, which are done by thermometer and binary weighted decoder.
This chapter is followed by the simulation and test-bench chapter. First, the used design flow and tools are mentioned. After that the improvement of the calculated values of chapter 5 are documented in chapter 6. The improvement is reached by simulating every building block with a spice simulation tool. The simulations start again with the main building block, the current source unit. After reducing and improving the decrease of the output current over the whole output voltage area the resulting transistor dimensions are summarized in a table. To illustrate the difference between the calculated and simulated values the calculated values can also be found in this table together with the output current difference. The current source unit simulations are followed by the simulations of the bias current generation. Again, the calculated and simulated values can be found in tables. Another table presents the improvement of the DC current accuracy. After the block simulations the whole analog part is simulated. As first, the Monte-Carlo distribution over 500 runs is presented. This is followed by the corner analysis. The corner analysis is done for temperature, voltage and CMOS process corners. The result is shown in a diagram.
After the analog simulations the digital test-bench is described. This is not only a test-bench, but rather a fully on the test-chip integrated digital 10 bit function generator. This function generator is also used for doing the mixed signal simulations on top level of the DAC. Most of the mixed signal simulations are not presented, because they just check the correct connectivity between the design blocks. However, the ramp function simulation is important for analyzing the nonlinearities. The nonlinearities are not directly simulated, they are post calculated. The value of every of the 1023 steps is stored during the ramp function and the nonlinearities are calculated afterwards. The results are presented in a diagram.
After the chapter with the design implementation the layout implementation chapter follows. Chapter 7 starts with the layout floorplan and the supply and output concept. The next sections deal with the implementation of the building blocks, again starting with the current source unit. Screenshots illustrate the final implementation of the blocks. The analog layout part is followed by a short description about the digital part with the improved switching scheme. Finally, a screenshot of the whole DAC is presented.
After checking the layout of the DAC the work is not done, because a test-chip is also produced. Before tape-out, pads and the digital video-encoder part have to be added to the DAC. The test-chip and its possibilities are presented in chapter 8. The requirements for the evaluation board can also be found in this chapter. Supplemented with the pinout, a suggested schematic and the description for the test multiplexer, this chapter supports all to bring the test-chip up for some measurements.
The measurements are done in chapter 9. This chapter describes the evaluation board and the used measurement setup. The board layout and setup of the PXI measurement system are done by Ungvári [2010]. Additionally, for the PXI system some LabVIEW programs are written. Similar to the simulation of the nonlinearities, the calculations for the measured nonlinearities are also done after the measurement with stored data. The results of Ungvári [2010] for the nonlinearities are also presented in chapter 9.
After this part the outlook is following with thoughts about general trends and ideas for future work. The main part of the work ends with the concluding remarks in chapter 11.
Finally, screenshots of the final implemented schematics are presented in the appendix A. Additionally, after the schematics the source codes of the digital parts are also added to the appendix.

Chapter 2

Characteristics and Architectures of Digital to Analog Converters

Digital to analog converters (DACs) are available in various architectures. The structure of a DAC depends on the special application. All converters have characteristic parameters, which are described in section 2.2. With this characteristic parameters it is possible to classify them and find the right one for each application. The descriptions and images of this chapter were found in books of Phillip E. Allen [2000], van de Plassche [1994] and R. Jacob Baker [1998].

2.1 Basic Concept of DACs

The basic concept of DACs is the same for all different types. It is presented in the following figure 2.1.

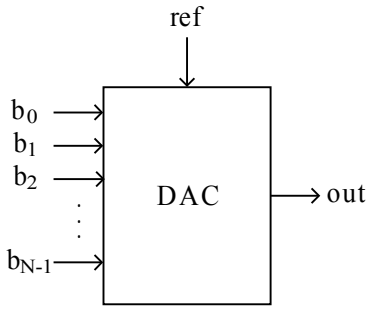

Figure 2.1: Basic concept of DACs.

A DAC has an n bit digital input and a single-ended or differential analog output. The analog output can be a current or voltage source. Additionally, there is a voltage or current reference needed for the converter. The analog output function can be written as follows

$$v_{out} = K \cdot V_{ref} \cdot D \tag{2.1}$$

or

$$i_{out} = K \cdot I_{ref} \cdot D \tag{2.2}$$

where K is a scaling factor which is independent of the digital input and D is the digital word

$$D = \frac{b_0}{2^0} + \frac{b_1}{2^1} + \frac{b_2}{2^2} + \cdots + \frac{b_{N-1}}{2^{N-1}} \tag{2.3}$$

where N is the total number of bit and b_i the i-th bit coefficient of the digital word in binary representation. [van de Plassche, 1994] A more detailed diagram of the basic concept can be found in figure 2.2. This shows graphically the implementation of the equations above. [Phillip E. Allen, 2000]

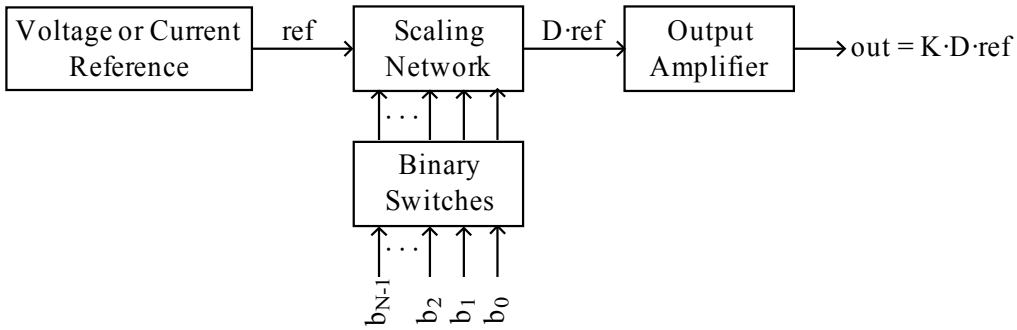

Figure 2.2: Block diagram of a DAC.

2.2 Characteristics of DACs

To classify DACs it is very important to know their characteristics. The characteristics can be divided into two groups - the static and the dynamic characteristics.

2.2.1 Static Characteristics of DACs

The group of static characteristics contains properties like resolution, full scale range, offset, and gain error. The nonlinearities are also members of this group. This section will describe these important properties for DAC classification.

Resolution

One of the most important properties is the resolution. In most cases the first question is, which resolution does the DAC have. The resolution is directly dependent on the width of the digital input word. For example, a DAC with 3 bit resolution has a digital input with a width of 3 bit and supports an analog output which can obtain eight different values. Figure 2.3 shows an ideal input-output characteristic of a DAC with 3 bit resolution. The diagram exposes, that every digital input word has its own unique analog output value. The smallest difference between two different output values is called least significant bit (LSB). Mathematically, the LSB can be defined as

$$LSB = \frac{ref}{2^N} \tag{2.4}$$

Additionally, figure 2.3 shows the ideal converter characteristic for infinite resolution. The other characteristic, with finite steps, is from a 3 bit DAC. If the finite characteristic is laid on the infinite, the digital zero value (000_b) does not generate a analog zero value (solid line). This value is in case of a 3 bit DAC $\frac{1}{16}^{th}$ of Ref. This is not very useful in practice. Thus, the characteristic is vertically shifted down to zero (dashed line).

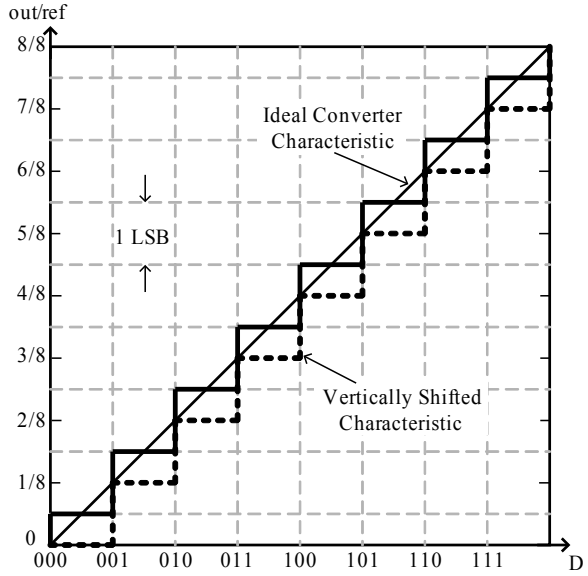

Figure 2.3: Input-output characteristic of a DAC with 3 bit resolution.

Full Scale (FS) and Full Scale Range (FSR)

The full scale value is one LSB below the reference (V_{ref} orI_{ref}). It is always smaller than the voltage reference because of the finite resolution of DACs. The equation for full scale (FS) can be written as

$$FS = Ref - LSB = ref\left(1 - \frac{1}{2^N}\right) \tag{2.5}$$

In contrast to FS is the full scale range (FSR). This value is always independent of the DAC resolution and can be written as

$$FSR = \lim_{N\to\infty} FS = ref \tag{2.6}$$

Quantization Noise

The quantization noise comes from digitizing an analog signal with finite resolution. It can be seen as rounding error between the analog input and the digitized output. Figure 2.4 illustrates the quantization noise of the 3 bit DAC from figure 2.3. The quantization noise looks like a sawtooth waveform with a peak-to-peak value of 1 LSB. For example, the dashed waveform shows an increase of the deviation up to 1 LSB between two consecutive digital input codes. At the next code change, the analog value exactly matches the ideal DAC characteristic. This is the reason for the sawtooth waveform shown in figure 2.4. The quantization noise is a fundamental property of DACs and limits the accuracy of converters. It can be calculated with the following equation

$$SQNR = 20\log_{10}(2^N) \approx 6.02N \tag{2.7}$$

where $SQNR$ is the signal-to-quantization-noise ratio and N is the total number of bit. The only possibility to reduce the overall quantization noise is to increase the resolution. [Wikipedia, 2009]

Dynamic Range (DR)

The DR is the ratio between the smallest and largest possible output value. The smallest possible output value is one LSB and the largest possible output value is the FSR. The DR can be expressed by the

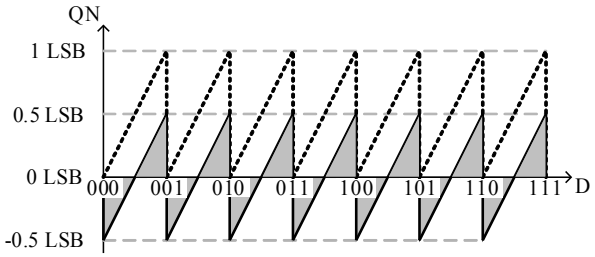

Figure 2.4: Quantization noise of a DAC with 3 bit resolution.

following equation

$$DR = \frac{FSR}{LSB\ change} = \frac{FSR}{\frac{FSR}{2^N}} = 2^N \tag{2.8}$$

This value can also be expressed in dB

$$DR(dB) = 6.02N\,dB \tag{2.9}$$

Offset Error

The offset error is the first error source described. An offset error is typical for all integrated circuits. It is caused by the finite matching of the devices. For pure analog circuits, the offset voltage is the voltage which has to be applied to the input to make the output zero. For DACs it is the constant shift of the finite DAC characteristic from the ideal infinite characteristic. Offset voltages or currents can not be avoided, but they can be trimmed. Figure 2.5a illustrates the offset error.

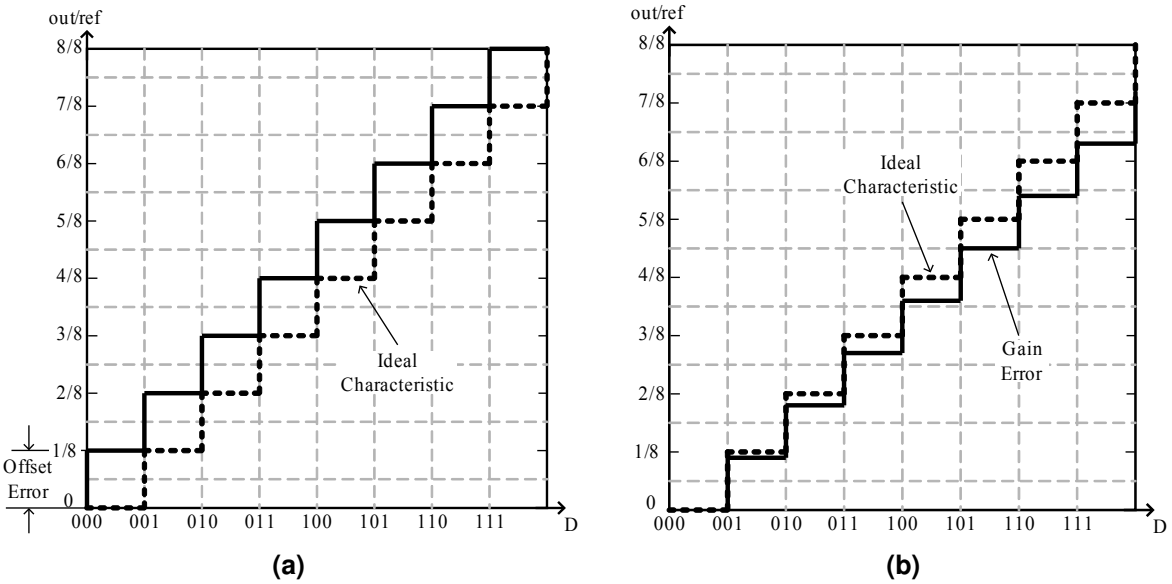

Figure 2.5: Offset (a) and gain (b) error of a DAC with 3 bit resolution.

Gain Error

The gain error is defined and has to be measured for the full scale input code. Therefore the difference between the real output voltage or current and the ideal output voltage or current is the gain error.

Figure 2.5b presents the gain error.

Differential Nonlinearity (DNL)

The DNL is measured at every code change. It is the deviation of the actual analog output value from the ideal DAC characteristic (see figure 2.6). Before calculating the DNL the gain error has to be removed. For the specifications the maximum DNL is taken. The DNL is also presented in percentage of FSR or in terms of LSB. It can be calculated by using the following equation

$$DNL = \left(\frac{V_{change} - V_{LSB}}{V_{LSB}}\right) \cdot 100\% = \left(\frac{V_{change}}{V_{LSB}} - 1\right) \cdot LSB \tag{2.10}$$

where V_{change} is the actual change of the output voltage (v_{out}) at a code change and V_{LSB} is the ideal voltage difference. This can be written as

$$V_{LSB} = \frac{V_{FSR}}{2^N} \tag{2.11}$$

Integral Nonlinearity (INL)

The INL error is the deviation between the actual converter characteristic and the ideal converter characteristic. Before calculating the INL the offset and gain errors have to be removed. Similar to the DNL the maximum value is taken for the specification. Usually, one of the two following definitions is used. The first is the end-point INL, where a straight-line passes through the converters end points. These are the zero and full scale outputs. The second is the best straight-line INL, where a straight-line determines the closest approximation to the DACs actual characteristic. The second definition supports a smaller and more realistic result. The INL can have positive and negative values and is presented in percentage of FSR or in terms of LSB. Figure 2.6 illustrates the INL characteristic of a 3 bit DAC.

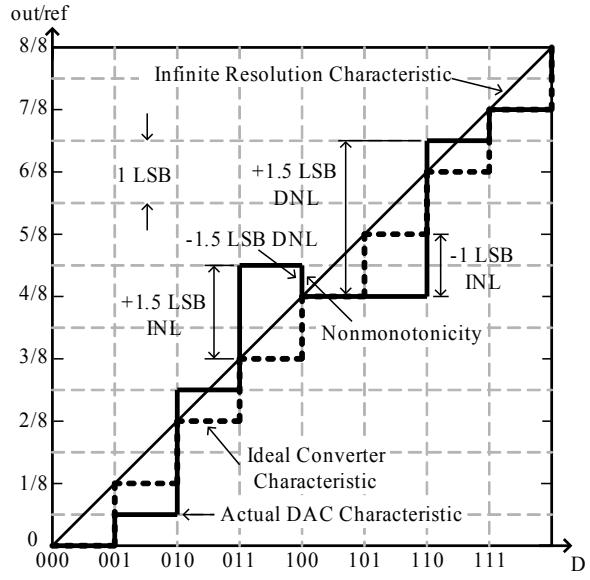

Figure 2.6: INL, DNL and monotonicity error of a DAC with 3 bit resolution.

Monotonicity

Monotonicity means a continuous increase of the output voltage by a continuous increase of the digital input word. If the DAC characteristic has one step down, the DAC is not monotonous (see figure 2.6). For DNL values smaller and equal 1 LSB the characteristic is always monotonous.

2.2.2 Dynamic Characteristics of DACs

A description of conversion speed, settling time, Signal to noise ratio, power supply rejection ratio, and glitches is following. These are not all possible dynamic characteristics for DACs, but the most important.

Conversion Speed

The conversion speed is the primary and important dynamic characteristic of DACs. It is the time the DAC needs to provide the new analog output value after a digital input code change. The conversion speed can vary from milliseconds to nanoseconds depending on the DAC architecture used.

Settling Time

The settling time is the time a DAC needs from starting a transition at the analog output until the new output value is reached. The new output has to be within the specified accuracy.

Signal to Noise Ratio (SNR)

As important as the conversion speed is the signal to noise ratio (SNR) . The SNR of a DAC depends mostly on the resolution. It includes also linearity, settling time, glitches, noise and distortion. The SNR is normally specified over half the sampling frequency and should ideally follow the following equation

$$SNR = 6.02N + 1.76\,\mathrm{dB} \tag{2.12}$$

For DACs the SNR is calculated with a digital sine wave input. The sine wave should use the full resolution of the DAC.

Power Supply Rejection Ratio (PSRR)

Changes of the DC-power supply within operating conditions effects changes on the output voltage or current of the DAC. The ratio between the voltage or current changes on the power supply and the voltage or current changes at the output is the PSRR. It is often specified in dB. The equation for the PSRR can be written as follows

$$PSRR = \frac{\Delta V_{supply}}{\Delta V_{out}} \; or \; \frac{\Delta I_{supply}}{\Delta I_{out}} \tag{2.13}$$

Glitches

Glitches are important in DAC designs. Depending on the DAC architecture the largest glitches are normally generated during the major code changes, for example from 011_b to 100_b. In this case the output value jumps through 000_b and this generates a sharp glitch at the output. The glitch results in distortion and decreases the SNR.

2.3 Basic Architectures of DACs

In principle there are two groups of converters - the serial and the parallel DACs (see figure 2.7). Normally, the conversion speed is the main parameter for selecting one of these two groups. Serial DACs make only one single conversion per clock cycle. Therefore, they need N clock cycles for N bit. In contrast are the parallel DACs they convert all bit at the same time. Parallel DACs themself can also be divided in three subgroups depending on the scaling method. They are called current, voltage and charge scaling converters.

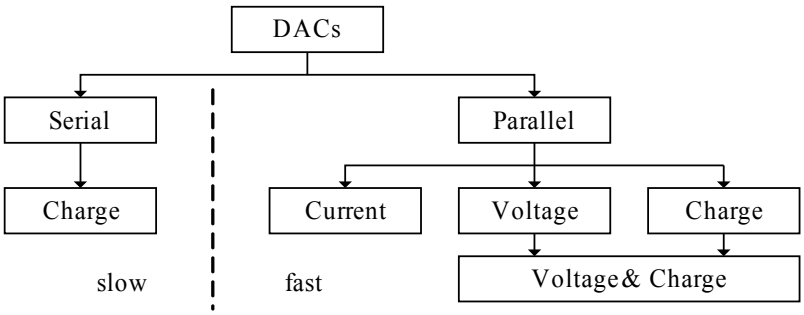

Figure 2.7: Hierarchy of converter architectures.

2.3.1 Serial DACs

Serial DACs are called serial, because they convert every single bit sequentially. In general, they convert one bit at one clock cycle. So they need N clock cycles to convert all bit. This converter needs only a few components (see figure 2.8). First, the digital input has to be converted from parallel to serial. Depending on the digital input bit value, the switches connect V_{ref} or ground to the summer. This value is stored by a sample and hold (S/H) element and connected to an amplifier with a gain of $\frac{1}{2}$. The output of the amplifier provides the output voltage v_{out} of the converter and is connected by a feedback loop with the summer. For the conversion of the last bit, the N-th cycle, the output equation can be written as

$$v_{out}(N) = \left(D_{N-1} \cdot V_{ref} + \frac{1}{2} \cdot v_{SH}(N-1) \right) \cdot \frac{1}{2} \tag{2.14}$$

where the initial condition of the output of the S/H has to be zero ($v_{SH}(0)$=0 V). The accuracy of this converter depends on the accuracy of its single components. Therefore, the gain of the $\frac{1}{2}$ amplifier is usually generated with capacitors and has to be in the range of the accuracy of the DAC. The disadvantage of serial DACs is the speed. With the architecture of figure 2.8, the conversion needs N clock cycles. However, there is a possibility to get the correct value every clock cycle. The answer is the architecture of a pipeline DAC. For the pipeline DAC N converters are needed. They are connected like a chain. This means the output of the first converter is the input of the second converter and so on. The converter needs N clock cycles for the first output too, but the next values will be generated at every clock pulse. This speed improvement comes at a cost of N-times more chip area. [R. Jacob Baker, 1998]

2.3.2 Parallel DACs

The more common group of DACs are the parallel DACs. These DACs normally need only one clock cycle to generate the analog output voltage or current. The basic architecture of all parallel DACs is pretty the same. They have a voltage or current source and N switches, which are controlled by the digital input word. The switches scale the voltage or current reference and provide the output value. The parallel DAC can be divided into three substructures: the current scaling, the voltage scaling, and the charge scaling DACs.

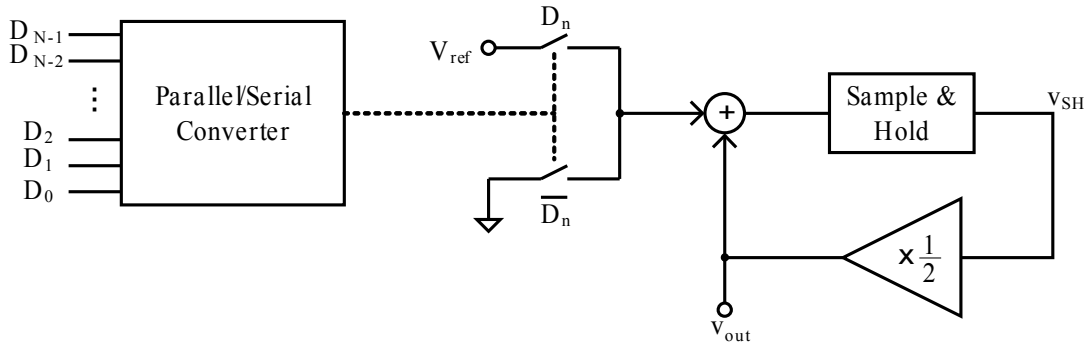

Figure 2.8: A serial DAC.

Current Scaling DACs

Typically, a current scaling DAC converts the voltage reference (V_{ref}) into a current reference. This reference current is divided into several binary weighted currents. Accumulated by the scaling network the output current is typically converted to an output voltage (v_{out}) by an inverting operational amplifier (see figure 2.9). The output voltage can be expressed as

$$v_{out} = -R_f \cdot (I_1 + I_2 + \cdots + I_N) \tag{2.15}$$

where the currents I_1 to I_N are binary weighted.

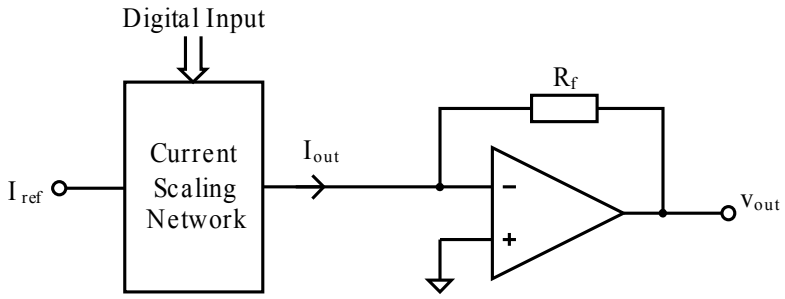

Figure 2.9: Basic architecture of a current scaling DAC.

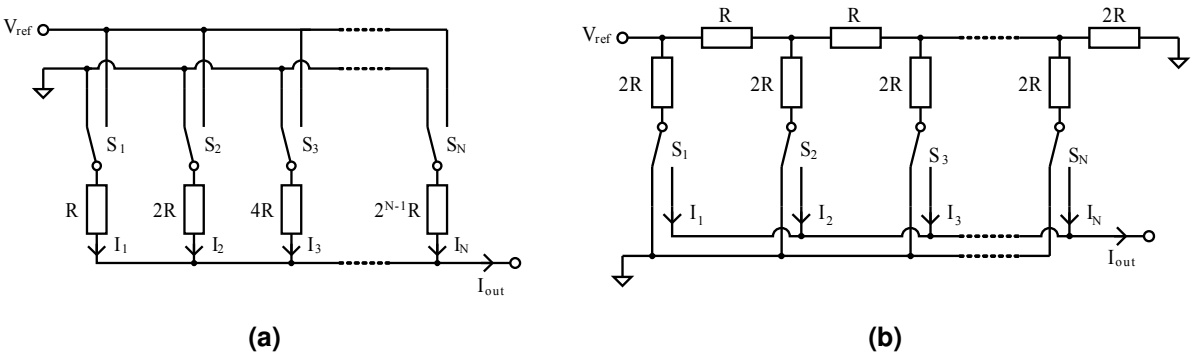

Figure 2.10: Binary weighted (a) and $R{-}2R$ (b) current scaling networks.

There are many different possibilities to implement the scaling network. The basic ones are with binary weighted or $R{-}2R$ resistor structures (see figures 2.10). The resistors are connected between the

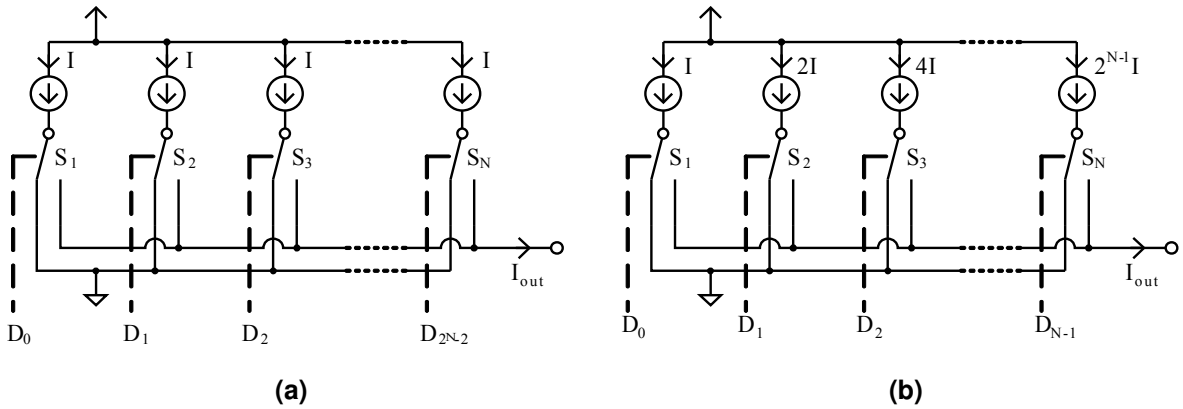

Figure 2.11: Diagram of an Unary (a) and a binary (b) current steering DACs.

voltage reference V_{ref} or ground and the output terminal. If the switches connect the resistor from V_{ref} to the output terminal the voltage on the resistors generates a current depending on the resistors values. The current at the output terminal i_{out} is the sum of all currents coming from the resistors.
Nevertheless not only currents from resistors are used for current scaling DACs but also currents from current sources. The current sources are good matching metal oxide semiconductor (MOS) current mirrors. This kind of DACs are called current steering DACs and they are explained in detail in chapter 3. Figure 2.11 shows two basic scaling networks, a) with equal currents and b) with binary weighted currents.

Voltage Scaling DACs

Another very common DAC architecture is the voltage scaling DAC. As the name implies, the voltage scaling DAC scales the voltage not the current. Figure 2.12 displays the basic block diagram of voltage scaling DACs.

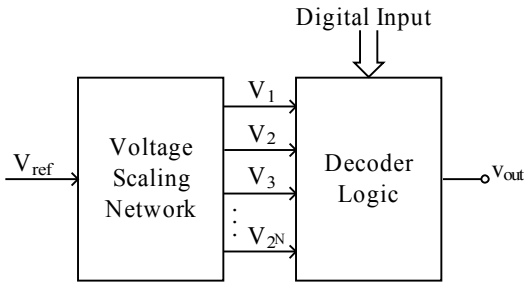

Figure 2.12: Block diagram of voltage scaling DACs.

The voltage scaling network is typically a serial array of resistors between the reference voltage V_{ref} and ground. The voltage taps between two resistors are connected to the switching network. Normally, all resistors have the same value except the outer ones. Figure 2.13 shows the simplest implementations of a voltage scaling DAC. The use of binary weighted switches lowers the output capacitance.

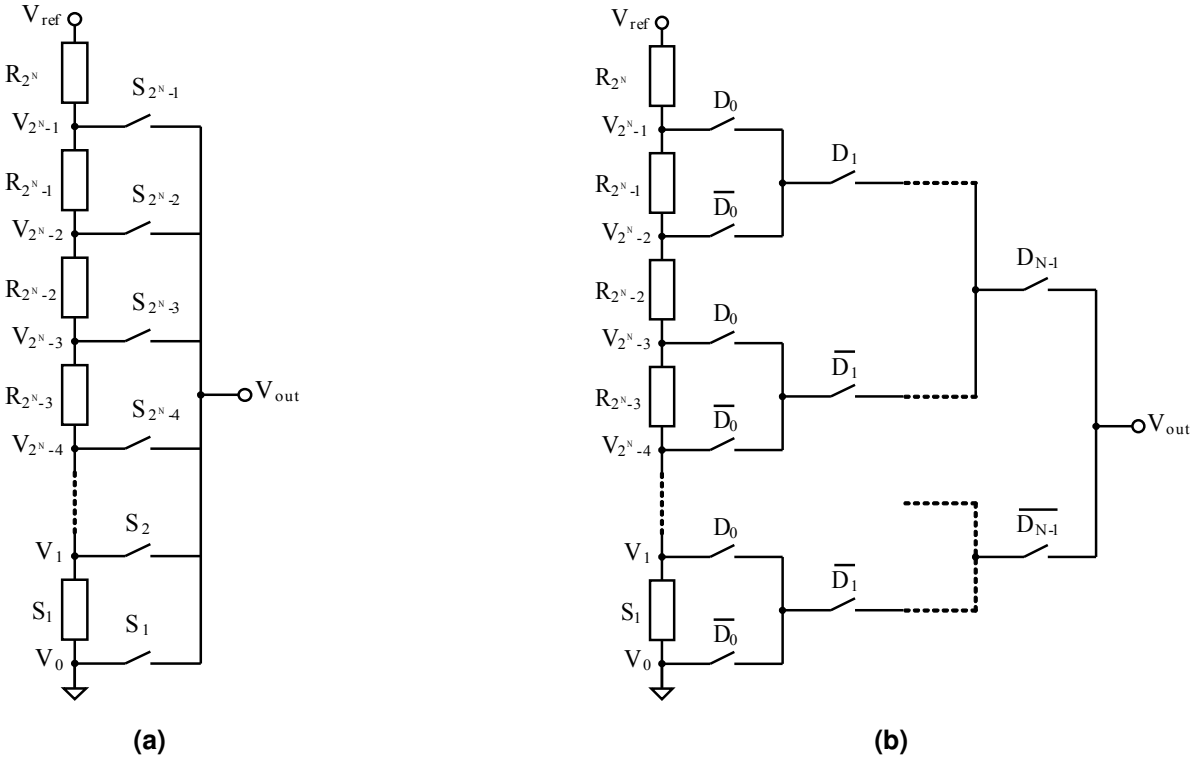

Figure 2.13: Thermometer coded (a) and binary weighted (b) voltage scaling resistor networks.

Charge Scaling DACs

Charge scaling DACs deal with the charge of binary weighted capacitor arrays. They divide the total charge applied to the array depending on the digital input word D.

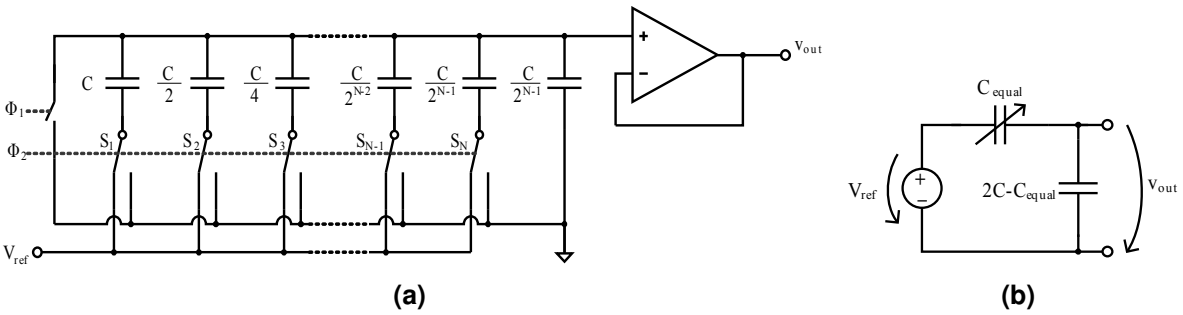

Figure 2.14: Basic architecture (a) and equivalent circuit (b) for charge scaling DACs.

Figure 2.14a shows the basic concept of charge scaling DACs. For the switches, two non-overlapping clock-phases are used. The first clock-phase is Φ_1 where all top and bottom plates of the capacitors are connected to ground. In this phase the total charge of the whole array becomes zero. During the second clock phase Φ_2 all switches from S_1 to S_N, where the binary bit is 1, are connected to V_{ref} and the other switches are connected to ground. In other words, if b_i is 1 the switch S_i connects the capacitor to V_{ref}. Otherwise, if b_i is 0, the switch S_i connects to ground. During the second clock phase Φ_2 the output is valid. In sum, the whole capacitor network has a total capacitance C_{total} of $2 \cdot C$. For better understanding figure 2.14b shows the equivalent circuit with the reference voltage V_{ref} and the two capacitors C_{equal} and $2 \cdot C - C_{equal}$, where C_{equal} is the capacitance depending on the digital input word

D and $2 \cdot C - C_{equal}$ is the capacitance which provides the output voltage v_{out}. The charge distribution of charge scaling DACs can be written as

$$V_{ref} \cdot C_{equal} = V_{ref} \cdot \left(b_1 C + \frac{b_2 C}{2} + \frac{b_3 C}{2^2} + \cdots + \frac{b_N C}{2^{N-1}} \right) = C_{total} \cdot v_{out} = 2 \cdot C \cdot v_{out} \tag{2.16}$$

The equation transformed to express v_{out} gives

$$v_{out} = \left(b_1 \cdot 2^{-1} + b_2 \cdot 2^{-2} + b_3 \cdot 2^{-3} + \cdots + b_N \cdot 2^{-N} \right) \cdot V_{ref} \tag{2.17}$$

In other words, this circuit can be seen as a capacitive voltage divider.

Chapter 3

The Architecture of Current Steering DACs in Detail

This chapter covers current steering DACs in detail. The motivation to adopt a current steering DAC for a video application has several reasons. The most important reason is the good speed performance of current steering DACs. Other positive properties are the good linearity and dynamic behavior. Additionally, the robust design and improved power efficiency are also strong arguments to use current steering DACs. The basic structure is more or less simple. This means, that usually no operational amplifiers are needed. From the conceptional view, the design is only built on numerous current mirrors with digitally controlled switches. The next sections describe the main building blocks and special design features of current steering DACs. [Michiel Steyaert, 2002]

3.1 Topologies of Current Steering DACs

The basic structure of current steering DACs is more or less simple. A digital input word D controls an array of switches, that routes numerous current sources to the output, depending on the input word. The output is a summed current output or a converted voltage output. This sounds to be a very easy and straight forward design with limited possibilities. However, in practice there are many different ways to implement current steering DACs.

3.1.1 Topologies for Output Current Generation

Michiel Steyaert [2002] describes four different basic topologies for current steering DACs to dump the output in current into the load. The load consists of a resistor and a capacitor. The resistor represents the static current-to-voltage conversion and the capacitance represents the dynamic limitation of the settling behavior. Figure 3.1a shows a simple digitally programmed current source. Analog to the first one, figure 3.1b presents the same architecture but this time with a current sink. The next figures 3.1c and 3.1d illustrate bipolar current sources. This can be achieved by adding a half scale current sink or source. The total output current can reach positive or negative values. Additionally, figure 3.1d includes an output converter, which converts the output current into something more convenient. This can be for example a transimpedance amplifier to convert the output current to an output voltage or a zero order hold to get better frequency domain performance.

3.1.2 Decoding Schemes

One of the important properties of current steering DACs is the digital interface. Mainly, there are two different digital decoders for current steering DACs: binary weighted and thermometer decoder.

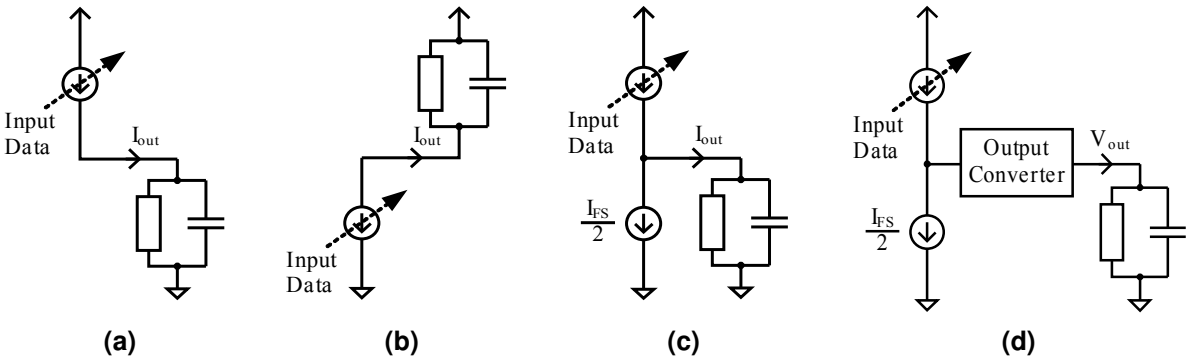

Figure 3.1: Basic topologies of current steering DACs: (a) current source, (b) current sink, (c) bipolar current source and (d) bipolar current source with output converter.

Starting with binary weighted decoders the basic concept can be seen in figure 2.11b. The simplest way is to sample the digital input with latches and route the output of the latches directly to binary weighted current sources. The complexity of the digital part is very low, it only consists of N latches for an N bit digital input. The latches should guarantee, that all switches of the current sources/sinks get the new information at the same time. However, the lower complexity of the digital part seems to be an advantage, but this scheme has also some big disadvantages. One is related to the switching of the MSB. During major bit transitions of the MSB, for example the change from 01111_b to 10000_b or the other way round, undesirable temporary states like 11111_b or 00000_b are possible. For this transition it is difficult to guarantee monotonicity and a good DNL. Changing the output for all switches produces a large glitch. This can be seen in the frequency domain as large spurious components.
Thermometer decoding avoids this major transition problems. The thermometer decoder incrementally switches one current source/sink unit after another, as required. For this scheme the digital input has also to be latched. Nevertheless, the input has to be decoded before it is routed to the unary current source/sink units. The basic concept of this architecture can be seen in figure 2.11a. The decoder will be described in more detail in section 3.2.3. However, to convert an N bit binary input to thermometer code 2^N-1 decoding elements are necessary. The digital part will be more complex, but the problems with the major transition vanish. To keep the area within reasonable bounds the current sources/sinks are usually organized in a matrix. Fast local row-column decoder are used to control the current source/sink matrix.
Each of the two described decoding schemes has some advantages and disadvantages. To bring the advantages together a combination of both schemes is used. DACs, which are using both schemes, are called segmented current steering DACs. For the implementation of a segmented current steering DAC the digital input word N has to be split up into two pieces. The thermometer coded M MSBs, which address $2^M - 1$ current units, and the L binary weighted LSBs, which are directly connected to the least significant current sources/sinks. Three essential characteristics for the partitioning of the digital input can be specified: the desired area, the complexity of decoding, and the level of dynamic non-idealities. Usually, the thermometer part is limited to a size of 6 to 8 bit. For larger bit sizes a significant degradation of the dynamic behavior would be the consequence. The reason is the combinatorial depth of the decoding elements. Another issue is of course the area consumption of such large decoders.

3.2 Building Blocks

The current steering DAC consists only of a few components. These components are current mirrors, current switches and a digital decoder. Current mirrors are used to generate the supply currents and the output currents of every current output unit. The current switches have only to switch the output current of every unit between the dummy load output and the current output. To convert the binary digital input

stream into a thermometer code, a thermometer decoder is used. These building blocks are described in more detail in the following sections.

3.2.1 Current Mirrors

Basic Current Mirrors

Current mirrors are one of the most common structures in MOSFET circuits. It is easier to distribute currents over an integrated circuit than voltages. Distributing voltages yields to voltage drops over the interconnect wires and therefore more complex operational amplifier circuits are used to get a precise voltage over more or less long interconnect wires. Hence, current mirrors have a widely spread field of applications. The first thing to answer is, how can a precise current be generated. This is not as easy as it seems to be. It is not enough to supply the MOSFET with a corresponding gate-source voltage V_{GS}. Because the drain current I_D is also depending on the supply voltage V_{DD}, the actual temperature and some process relying parameters like the threshold voltage V_{th}. However, the generation of a precise reference current I_{ref} is not part of this work. For the work it is crucial to make an exact copy of the reference current I_{ref}. Figure 3.2a shows the main concept of current mirrors. A copy circuit generates a copy of the reference current I_{ref} and mirrors it to one or more current sources. Their drain current I_D is an exact copy of the reference current I_{ref} and is named I_{out} for the output current of the current mirror. How can be guaranteed that the output current I_{out} is an exact copy of the reference current I_{ref}? Rudimentary it is important, that the current source transistor works in saturation region. The square-law model of MOSFET describes the relation between the gate-source voltage V_{GS} and the drain current I_D through the transistor. This equation can be written as follows

$$I_D = \frac{1}{2}\mu_n C_{ox}\frac{W}{L}(V_{GS} - V_{th})^2(1 + \lambda \cdot V_{DS}) \tag{3.1}$$

where μ_n and C_{ox} are technology depending constants, V_{th} is the threshold voltage, W and L are the width and length of the transistors channel, λ is the channel length modulation, and V_{DS} the drain source voltage. For the basic inspections the last term with the channel length modulation will be ignored. Generally it can be said the current I_D is a function of the voltage V_{GS}

$$I_D = f(V_{GS}). \tag{3.2}$$

Or the other way round, the voltage V_{GS} is an inverse function of the current I_D

$$V_{GS} = f^{-1}(I_D). \tag{3.3}$$

If the current I_D is biased by the reference current I_{ref} this leads to

$$V_{GS} = f^{-1}(I_{ref}). \tag{3.4}$$

Figure 3.2b shows the copy circuit consisting of an N-MOSFET connected as diode. The output of this circuit provides the gate-source voltage of the mirror transistor depending on the reference current I_{ref}. The subsequent figure 3.2c presents the final version of an basic current mirror. The output current I_{out} expressed as equation depending on the reference current I_{ref} relying on the equations 3.2 to 3.4 gives

$$I_{out} = f\left(f^{-1}(I_{ref})\right) = I_{ref}. \tag{3.5}$$

This equation is true if both MOSFETs are identical and the channel length modulation coefficient λ is zero. For the two MOSFETs M_1 and M_2 in figure 3.2c the square-law equations can be written as

$$I_{ref} = \frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)_1(V_{GS} - V_{th})^2 \tag{3.6}$$

and

$$I_{out} = \frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)_2 (V_{GS} - V_{th})^2. \tag{3.7}$$

Both equations have of course the same technology depending constants, and if the transistors are identical, the saturation voltages $V_{DSsat} = V_{GS} - V_{th}$ also have the same values. If both equations are set in relation I_{out} can be written as

$$I_{out} = \frac{(W/L)_2}{(W/L)_1} I_{ref} \tag{3.8}$$

This basic equation demonstrates the main property of current mirrors. The relation between reference and output current is only depending on the dimensions of the two mirror transistors and not on process parameters or the temperature. Referring to basic matching rules (see chapter 4) it is possible to generate the relation between the two currents in an accurate way. [Razavi, 2001]

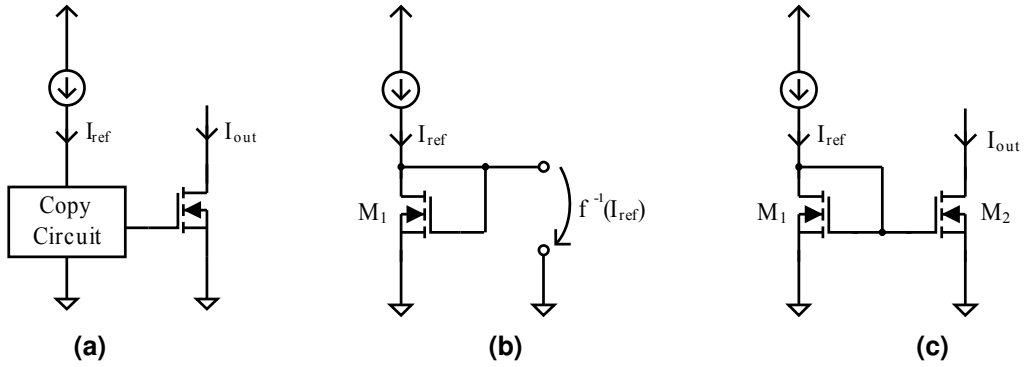

Figure 3.2: Basic concept of current mirrors: general concept of the copy circuit (a), the copy circuit in detail (b) and the complete design of a current mirror circuit (c).

Cascode Current Mirrors

The previous section neglected the non-ideal output characteristic of MOSFETs. In this section the channel length modulation λ will also be considered. Therefore, the equations for the reference current I_{ref} and the output current I_{out} of the basic current mirror in figure 3.2c including the channel length modulation can be written as

$$I_{ref} = \frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)_1 (V_{GS} - V_{th})^2(1 + \lambda \cdot V_{DS1}) \tag{3.9}$$

and

$$I_{out} = \frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)_2 (V_{GS} - V_{th})^2(1 + \lambda \cdot V_{DS2}). \tag{3.10}$$

After merging the tow equations together and setting the reference current I_{ref} and the output current I_{out} in relation, the resulting equation looks as follows

$$\frac{I_{out}}{I_{ref}} = \frac{(W/L)_2}{(W/L)_1} \cdot \frac{1 + \lambda \cdot V_{DS2}}{1 + \lambda \cdot V_{DS1}}. \tag{3.11}$$

While the voltages V_{GS1}, V_{GS2} and V_{DS1} are identical, the drain-source voltages V_{DS2} of the mirror transistor M_2 differs. This voltage depends on the circuit which is connected to the current mirror output. However, there is a way to reduce the influence of the channel length modulation to current mirrors. Additional to the current mirror transistors a cascode transistor will be added. Figure 3.3a illustrates the basic implementation, where the bias voltage V_b of the cascode transistor is chosen such

that $V_{DS2} = V_{DS1}$. Figure 3.3b shows a possible way to generate the bias voltage V_b. To ensure, that $V_{DS2} = V_{DS1}$ it has to be guaranteed that $V_b - V_{GS3} = V_{DS1}$ or the other way round $V_b = V_{GS3} + V_{DS1}$. Consequently, if a gate-source voltage is added to V_{DS1}, this demand can be fulfilled. Therefore, another diode-connected transistor M_0 is added in series to the current source transistor M_1. This transistor generates a voltage $V_b = V_{GS0} + V_{DS1}$. If a transistor M_3 is added to the current mirror circuit (see figure 3.3c), but this time in series with M_2, the cascode current mirror is complete. By a good choice of the dimensions of M_3 the gate-source voltages will be the same of M_0 and M_3. After connecting the gates of M_0 and M_3, $V_{GS0} + V_{DS1} = V_{GS3} + V_{DS2}$. If the dimensions of the transistors M_3/M_0 and M_2/M_1 have the same relations, the gate-source voltages V_{GS0} and V_{GS3} and the drain-source voltages V_{DS1} and V_{DS2} will be identical. Since both transistors M_0 and M_3 have to see the same source voltage, the equations from above are independent of the backgate effect.
The use of cascoded current sources makes the current mirror circuits almost independent of the output circuit. The cascode transistor M_3 is something like a guardian for the output transistor M_2. It protects M_2 from bad influences at the output net. It also minimize the so called "Miller Effect" for M_2. Another improvement is a raise of the output resistance of the mirror circuit. However, the advantages are not for free. First two additional devices are needed, this implies an area increase. The second point is the reduction of the voltage headroom, which is consumed by the cascode transistor M_3. Particularly this will be a problem for low voltage circuits.

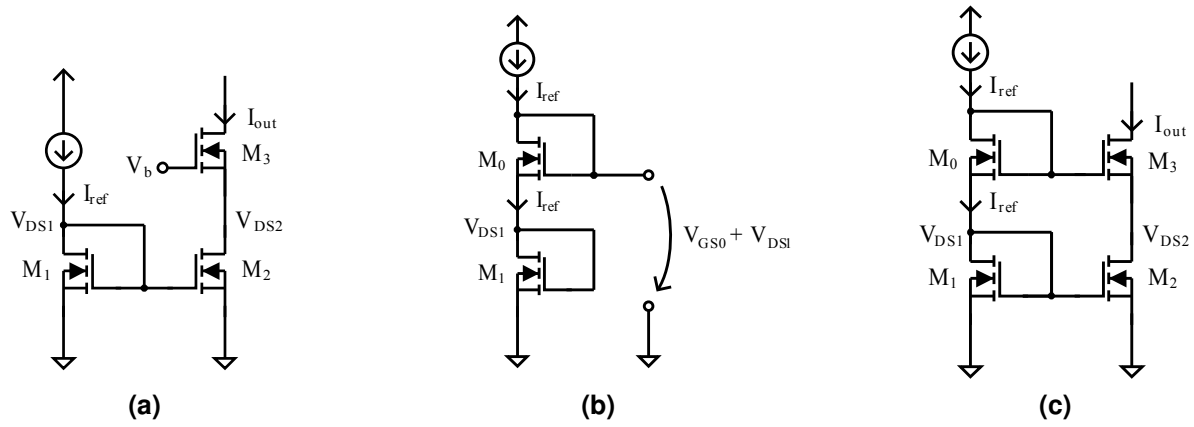

Figure 3.3: Basic cascode current mirror circuits: general concept of the cascode circuit (a), generation of the bias voltage V_b (b), and the complete design of a cascoded current mirror circuit (c).

3.2.2 Current Source Unit

This section is about the typical current source unit of current steering DACs. Most current source units are built only with P-MOSFETs. One of the reason is the widespread production of integrated circuits with a substrate, that is p-doped. Therefore the P-MOSFET has to be embedded in an n-well, this is an n-doped area in the p-doped substrate. Hence, the PMOS transistor has a better noise performance, because of the isolation against substrate. Thus, all following descriptions refers to a current source unit with P-MOSFETs. The structure of a current source unit is very simple (see figure 3.4). The unit consists only of the current source M_{CS}, the cascode M_{Casc} and two switching transistors M_{Sw0} and M_{Sw1}. The gates of the current source and cascode transistor are connected to the bias voltages V_{CS} and V_{Casc}. The two switching transistors M_{Sw0} and M_{Sw1} are connected to the enable and inverted enable signals V_{on} and $\overline{V_{on}}$. Before designing a current source unit, four essential criteria have to be discussed.

The first criterion is the **overall output resistance**, that depends on the data input word D. The output resistance will have a maximum if only one unit is switched to the current output and a minimum

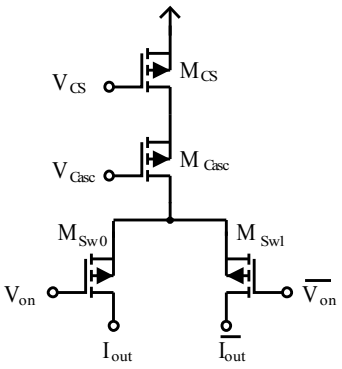

Figure 3.4: Basic PMOS current source unit.

for all current source units. The output current is a function of the output resistance and the data word and can be written as

$$I_{out} = \frac{I_{LSB} \cdot D + V_{DD} \cdot G_{LSB} \cdot D}{1 + G_{LSB} \cdot R_L \cdot D} \tag{3.12}$$

where I_{LSB} is the unit current and G_{LSB} is the output conductance of a single current source unit, V_{DD} the supply voltage, and R_L the output load resistor. As the equation shows, a larger output conductance G_{LSB} gives a larger difference between the ideal current and the actual current if the input data word increases. This directly increases the non-linearities INL and DNL. Therefore, the output resistance has to be designed very carefully to fulfill the specifications.[Hugo Hermandez and Roa, 2007]
The second important criterion is the **accuracy** between the current source transistors M_{CS} of each unit. The accuracy is based on the statistical mismatch model (see Miquel Albiol and Alarcón [2004]). This model combines the relations between the area, the overdrive voltage and some important process constants in the following equation

$$\frac{\sigma^2_{I_D}}{I^2_D} = \frac{A^2_\beta}{2 \cdot W \cdot L} + \frac{2 \cdot A^2_{V_{th}}}{W \cdot L \cdot V^2_{OD}} \tag{3.13}$$

where I_D is the current of the current source transistor M_{CS}, W and L are the transistor channel dimensions, V_{OD} is the overdrive voltage, A_β and $A_{V_{th}}$ are mismatch process constants, and $\sigma^2_{I_D}$ is the variance of the current I_D through the transistor M_{CS} due to random mismatch. The mentioned overdrive voltage V_{OD} is the difference between the gate-source voltage V_{GS} and the threshold voltage V_{th} of the current source transistor. After choosing the overdrive voltage V_{OD}, the dimensions of the current source transistor M_{CS} can be determined using equation 3.1 and 3.13. [Miquel Albiol and Alarcón, 2004]
The third criterion for current steering DACs is the **quantization noise**. The conversion bandwidth for these DACs is in between zero and the half of the sampling frequency $(0 < F < F_s/2)$. However, not the quantization noise itself is the problem, because it is calculable (see section 2.2.1). The problems are the other noise sources like the thermal noise of the transistors. If the thermal noise of the converter exceeds the quantization noise, this decreases the DACs accuracy. Therefore, the values of I_{LSB}, V_{OD} and the load resistor R_L have to be selected carefully to ensure, that thermal noise can be neglected compared to quantization noise. The literature proposes an additional safety margin to ensure this requirement. The power spectral density of the quantization noise $S_q(f)$ has to be ten times larger than the power spectral density of the thermal noise $S_q(f)$. This can be written as

$$S_q(f) > 10 \cdot S_t(f) \tag{3.14}$$

The fourth and last important DAC design criterion is the **settling time**. As can be seen in section 2.2.2 , the settling time is the time a DAC needs to reach the new analog output level within the specified accuracy. However, this can not happen in zero time because of the current source load capacitance. This

capacitance is the same for each current source, because of the common current output. If the settling time is larger than the sample time this will lead to conversion errors. Therefore, the analog output can not follow the digital code changes fast enough. This fact sets the maximum sampling frequency of current steering DACs. The most important parameter of the settling time is of course the output load capacitance. However, another indicator for the settling time is also the voltage at the net between the current source M_{CS} and the cascode transistor M_{Casc} in figure 3.4. The product of the load c_1 and the voltage v_1 at this point compared to the product of the load c_{out} and the voltage v_{out} at the output is the small-signal charge indicator γ. The equation for γ can be written as follows

$$\gamma = \frac{v_1 c_1}{v_{out} c_{out}}. \tag{3.15}$$

If γ increases, then the settling time will decrease and vice versa. Reducing the settling time has the disadvantage of increasing the transient voltage peaks during current source switching. If these voltage peaks have high power, spurious output appears in the frequency response. For the settling time the size of the cascode M_{Casc} and switching transistors M_{Sw} has to be designed very carefully. The mission is to find the optimal size to keep the settling time and the spurious small.

3.2.3 Thermometer Decoder

Thermometer decoders are mainly used in current steering DACs. Normally, only the higher four to six MSBs are decoded with the thermometer scheme and the lower bit are used in binary form. Hence, there exist implementations of current steering DACs, that are not segmented but fully thermometer decoded. The difference between the binary and thermometer decoding scheme is the redundancy and therefore the number of bit required. For N different cases a binary weighted decoder needs only $\mathrm{ld}(N)$ bit, but a thermometer decoder needs $N-1$ bit. Now, where is the advantage of thermometer decoder? DACs with thermometer decoder have no large glitches at the major code-changes like binary weighted DACs. This advantage comes from switching current units, which have all the same size. All bit of the thermometer word have the same weight. The bit of a binary code are weighted with the factor two, starting at the LSB. Table 3.1 shows a three bit binary-to-thermometer decoder, where the difference between the two coding schemes can be seen. With the $N-1$ thermometer bit t_k, more than $\mathrm{ld}(N)$ combinations shown in table 3.1 are possible. Every row in the table, this means every binary bit combination, can be represented with seven different thermometer bit combinations. For example the binary bit combination 001_b can be presented as 0000001_b, 0000010_b, 0000100_b ... This are $N-1$ combinations for each row, except if all bit are 0_b or 1_b. If all unit current sources have the same dimensions, the resulting output current should have the same value for all seven different cases. However, in practice this is not true, because of the particular mismatch errors of the current sources. Therefore, the different codes mentioned above will not generate the same analog output. Nevertheless, to get no worse results the different code combination can be used to weight the current sources. The weighting of the current sources is called the switching sequence. Certainly, during design phase it is not possible to measure the exact mismatch error. This would not be possible till the production of the integrated circuit. Of course, then it is to late to change the switching sequence in hardware. Some applications therefore have a programmable switching sequence, but this costs additional area. In practice, the process engineers will be able to deliver some information to get a good switching sequence for the current sources. In some cases, it would be necessary to implement more than one switching sequence in the first samples. After measuring the mismatch error it will be easier to choose the right switching sequence.

binary			thermometer						
b_2	b_1	b_0	t_6	t_5	t_4	t_3	t_2	t_1	t_0
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	1	1
0	1	1	0	0	0	0	1	1	1
1	0	0	0	0	0	1	1	1	1
1	0	1	0	0	1	1	1	1	1
1	1	0	0	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1

Table 3.1: A 3 bit binary-to-thermometer decoder.

Nevertheless, not only the reduction of the mismatch errors by the switching scheme is an advantage. The main advantage of thermometer coded current steering DACs is the monotonicity. The monotonicity is a main property of thermometer decoded DACs, because by increasing the digital input word, the current of exactly one current source unit is added to the output current. This yields to another advantage, the above mentioned reduction of glitches during code changes. No major code changes are happening like those known from binary controlled DACs. [Georgi I. Radulov and van Roermund, 2006]

Chapter 4

Layout Design Fundamentals

> *“ The process of creating an accurate physical representation of an engineering drawing (netlist) that conforms to constraints imposed by the manufacturing process, the design flow, and the performance requirements shown to be feasible by simulation. ”*
>
> [Clein [2000]]

This is the answer of Clein [2000] for the question, about the definition of layout design. The citation may sound a bit sophisticated, therefore it would make sense to have a closer look to the specific terms.
First, the layout *process* starts with setting up the database and tool chain. This will be followed by the floorplan definition of the specific modules. After that, the modules will be created corresponding to the schematics or netlists. With verification tools the correctness of the implementation can be checked. Finally, the layout database has to be prepared for the mask shop.
The *physical representation* means the drawing of the specific layers in the layout, to get, for example a transistor, after production. Each device is built up with different layers which has to be fit together according to the design rules to get the right functionality.
The *engineering drawing* is an electronic schematic created and simulated by design engineers. However, this can also be a netlist.
Conform to *constraints* means not the smallest possible layout may be the best. There exists *constraints* for the width of wires depending on the current, for the spacing between noisy and sensitive devices, the shielding of sensitive wires, and many other constraints. Important for layout engineers is the understanding of the schematic and the close collaboration with design engineers.
The *constraints* are on one hand driven from the design, but on the other hand they depend on the *manufacturing process*. The *constraints* given by the *manufacturing process* are called the layout design rules. This rules include the smallest width of wires and transistor gates, but they also consist of other guidelines to ensure a good quality during the manufacturing process.
The *constraints imposed by the design flow* mainly depend on the used tools. For example, this can be texts, which are written on a specific layer and needed by the check tool to verify the correct connectivity.
Finally there are *constraints imposed by the performance requirements shown to be feasible by simulation*. This means the design engineers do not really know the size of the module during design phase. The size is important, because the wire length depends on it. Not only the wire resistance given by the wire length are important for the final design, but also the capacitances between wires and between devices are important. These effects, which can not be checked by the layout checks, are called the *on chip parasitics*. The following section will describe in more detail which basic parasitics can appear on the physical layer. After that the most important matching structures will be shown in section 4.2. This section is followed by an overview of floorplanning techniques in sections 4.3. Finally, two sections about verification will follow. The first one in section 4.4 is about process design rules. This section

gives a short overview about the process design rule document and the meaning of the rules. Afterwards, section 4.5 deals with the layout verification, where the checking of the process design rules and connectivity of drawn layouts is described.

4.1 On Chip Parasitics

A chip consists of different layers which are built with various materials. Every material itself has parasitics and every crossing or closeness of two materials generates also parasitics. A wire, typically build with aluminum or copper, has a resistance depending on the thickness, the width and the length. Where the width and length can be influenced by the physical layout, is the thickness of the material process depending. The wires, which are drawn with ideal lines in the schematics, consist of many tiny parasitic resistors. The tiny parasitic resistors are not considered in the schematic, anyhow they can have a bad influence to the functionality of the circuit. Another known source of parasitics is the capacitance. Capacitances can appear between polygons of the same layer in horizontal direction, but also between different layers in vertical direction. The third common type of on chip parasitic is the inductance. This inductance is generated along the wires, which are normally not routed in a straight line, but rather in bends or in the worst case in loops.
The following subsections will describe in more detail the three major types of on chip parasitics. After that the section will be succeeded with a short description of the parasitic extraction. [Christopher Saint, 2002]

4.1.1 Parasitic Resistors

Parasitic resistors are everywhere on the chip. Any wire has a parasitic resistor which will not be considered in the normal design flow. For the schematic entry every connection between devices or modules is handled like an ideal connection with a $0\,\Omega$ resistor. Hence, in the physical layout any wire has a resistance, depending on the width, length and material (layer) which is used. For the decision of the wire dimensions are two criteria important. The first is the current density and the second one is the IR drop. The current density can be found in the process document. For example a metal wire can carry $0.5\,\mathrm{mA}$ per $\mu\mathrm{m}$ width. This means, if the continuous current is larger than $0.5\,\mathrm{mA}$ on a wire with $1\,\mu\mathrm{m}$ width, the metal will migrate and an open can occur. The current density parameter has not directly to do with parasitic resistors, but the second criteria has. The IR drop is the *current-resistance* drop. This means, referring to the law of Ohm, the voltage drop over wires. The law of Ohm can be written as

$$U = R \cdot I \tag{4.1}$$

where U is the voltage in Volt $[V]$, R is the resistance in Ohm $[\Omega]$ and I is the current in Ampere $[\mathrm{A}]$. Therefore, the voltage drop of a $1\,\mathrm{mm}$ long wire for the example from above can be calculated with the additional knowledge of the sheet resistance $R_\square$ of the metal wire. The sheet resistance is a piece of wire or every other on chip material with a quadratic area. The value of one square, this means the sheet resistance, can be found in the process document and is given for every material (layer). For example the sheet resistance of a typical metal layer has $0.05\,\Omega$. Therefore, for this example of a wire with a width W of $1\,\mu\mathrm{m}$ and a length L of $1\,\mathrm{mm}$, the equation for the number of squares $N_\square$ and the wire resistance R can be written as follows

$$N_\square = \frac{L}{W} = \frac{1000\,\mu\mathrm{m}}{1\,\mu\mathrm{m}} = 1000 \tag{4.2}$$

$$R = R_\square \cdot N_\square = 0.05\,\Omega \cdot 1000 = 50\,\Omega \tag{4.3}$$

This means the given piece of wire has a parasitic resistance of 50Ω. Now, with the help of Ohms law the voltage drop over this wire can be calculated. For this calculation the maximum allowed continuous current of $0.5\,\mathrm{mA}$ for the $1\,\mu\mathrm{m}$ wide wire is taken.

$$U = R \cdot I = 50\,\Omega \cdot 0.5\,\mathrm{mA} = 25\,\mathrm{mV} \tag{4.4}$$

This calculation shows a loss of $25\,\mathrm{mV}$ over a only $1\,\mathrm{mm}$ long interconnect wire. For most cases this would be no problem, but of course there will be circuits and interfaces, where a voltages drop of $25\,\mathrm{mV}$ will lead to a disaster.
For the physical layout the knowledge of the sheet resistance and sensitive interconnects is very important. In a normal design flow with the abdication of the simulation of a parasitic extraction the sheet resistance and therefore the voltage drop over wires and its influence will not be perceived.

4.1.2 Parasitic Capacitance

Analog to the parasitic resistances can parasitic capacitances also be found everywhere on the chip. Between pieces of material of the same layer and between pieces of material of different layers can parasitic capacitances occur, which will not be simulated in a normal design flow (without parasitic extraction). Figure 4.1 shows the profile of a chip with a p-doped substrate and two metal layers. Additional the most popular capacitances are marked. The black capacitances in the figure symbolize the parallel plate capacitances and the grey one the fringe capacitances. The capacitances can not only be found between metal layers which are exactly one upon the other, they are also between the metals and the substrate and sideways. This figure shows very well, that between every piece of material a capacitive coupling can occur.

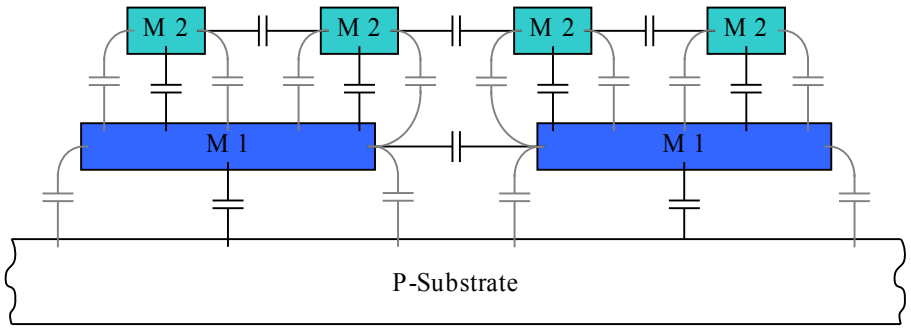

Figure 4.1: Profile of a chip with two metal layers and marked parasitic capacitances (based upon Christopher Saint [2002]).

Usually these capacitances are in a range of some femto farads (10^{-15}) and this seems to be a very small value, but if they are all added up they can make big problems.
What can be done to reduce the parasitic capacitances?
The first thing is the same as told in the parasitic resistor section. The wires should be kept as short as possible. A shorter wire has a smaller area and therefore less capacitance to other wires or to the substrate.
Another possibility to reduce the parasitic capacitance is the metal selection. The reason is the substrate, because it is everywhere. A noisy block, which has big capacitances to the substrate will contaminate the substrate with noise. As figure 4.2a demonstrates the noise will be coupled into the substrate. The substrate itself has parasitic resistors and over these other sensitive building blocks will be negatively influenced by the noise of the noisy block. By selecting a higher metal with a wider distance to the substrate the capacitances will be smaller and the influence too. The relationship between the distance and the size of the capacitance is presented in figure 4.2b.

The last basic rule to minimize the influence of parasitic capacitances is to avoid metal over metal. The points before are dealing mainly with the parasitic capacitances between metal wires and the substrate. Hence, another big things are the capacitances between metal wires of the same and of different layers. Every time one metal runs upon the other a parasitic capacitance will be generated. This may be

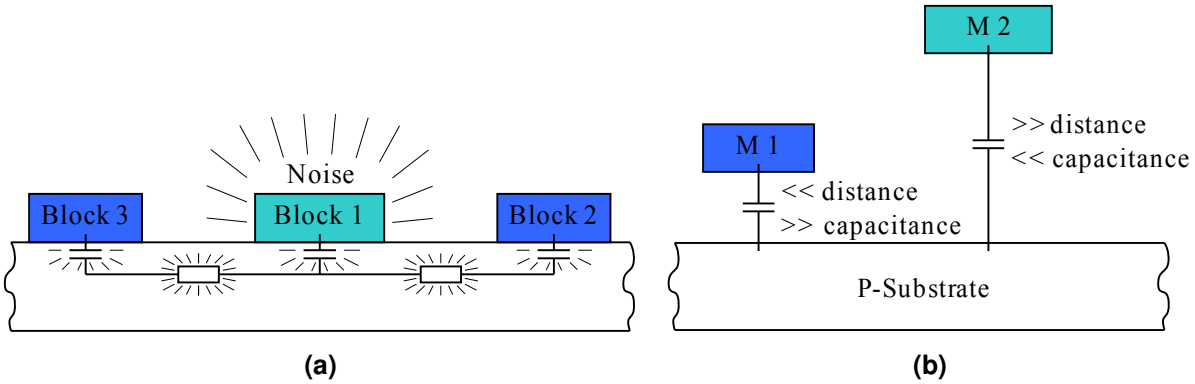

Figure 4.2: Influence of noisy blocks through parasitic devices (a) and relationship between distance and parasitic capacitance (b) (based upon Christopher Saint [2002]).

no problem inside a building block, but a wire from one block which runs over an other block will be a problem. The worst case is a noisy wire, for example a clock wire, which runs over a reference circuit or an operation amplifier input circuit. Therefore the layout designer routes the block interconnect not over other building block, but around them. Routing around other blocks and keep a safety distance to noisy blocks will reduce the influence of parasitic capacitances. When routing wires one upon the other it is important to think about the parasitic capacitance which will be generated.

4.1.3 Parasitic Inductance

The parasitic inductance will only get relevance for chips with really high frequency circuits. Every wire with high frequency signals will get additional to the parasitic resistance a parasitic inductance. There is no chance to avoid this inductance. The only way to become this parasitic inductance problem calculable is to model the wires with its frequency depending parasitic inductance as good as possible.
For the layout designer it is important to know about the high frequency elements and therefore he has to think about layout options to reduce the parasitic inductance. The first option is to reduce the wire length as it is possible. If it is impossible to reduce the length due to floorplan issues a wider wire will also enhance the performance. Another option is to leave some room around such high frequent wires to prevent other parts of the circuit from inductive coupling.
Hence, for high frequency layouts it is very important to think about the parasitic inductance. With these basic rules it is possible to reduce the problems on high frequency signal wires.

4.1.4 Device Parasitics

The last sections were mainly about parasitics between metal wires and metal wires and the substrate. However, there exist also noticeable parasitics in the substrate. These are primary parasitic capacitances between the different doped areas and between the gate and the channel. Figure 4.3 shows the different parasitic capacitances which exist in an NMOS transistor device. Every voltage change on gate, source or drain will be slowed down by these capacitances. The capacitances of a transistor can only be reduced by decreasing its size, but this will also reduce the driving strength. The only thing which can be done by the layout designer is to reduce the serial gate resistance. The reduction of the gate resistance will fasten the transistor by decreasing the time constant τ, which is the product of the resistance R and the capacitance C. In the following this equation can be found.

$$\tau = R \cdot C \tag{4.5}$$

The gate connection is usually made of polysilicon which has a higher resistance than the metal wires. To reduce the resistance of the gate connection the long single stripe will be split up into more fingers.

The fingers are connected with a metal wire in parallel. All other improvements concerning the parasitic capacitances of devices can only be done by modifying the schematic. However, this has to be done by the design engineers and verified with simulations.

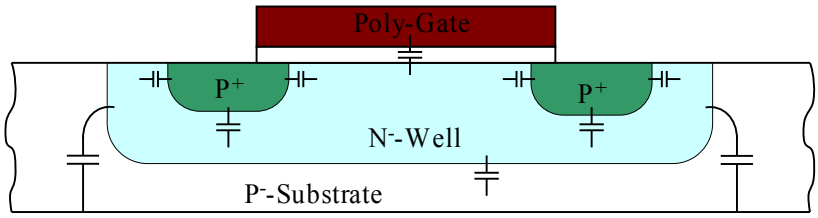

Figure 4.3: Parasitic capacitors of an NMOS transistor device.

4.1.5 Parasitic Extraction

The parasitic extraction is the extraction of parasitic resistors, capacitances and inductances out of a physical layout by the use of automated algorithms. The parasitic extraction and re-simulation of the design with this additional information is also called **backannotation**. There exists many different tools of several vendors which mainly differs in the algorithms and accuracy. Some easier tools only compute a two dimensional analyzes by extracting the wire resistance and capacitance. This extraction can be seen as a rough estimation which has also advantages. The analyzes are fast and do not produce huge extraction files. Especially for small digital designs this will be accurate enough. However, for sensitive analog circuits or circuits with higher frequencies more precisely algorithms have to be used. Here will a 3D analysis extract all parasitics which were described in the sections above. Despite the great computing performance of modern processors and the improved algorithms an extraction of the whole chip will take a lot of time and disk space. Usually the extraction will run for some minutes or hours. In backannotation phase the generation of an extracted view will be the faster part. The simulation of the whole chip or only complex parts of it by including the extracted view with the additional parasitic devices will take many hours, days or will not be possible in some cases. Therefore, the minimum size for parasitic devices, which will be extracted, can be chosen in most extraction tools. This will minimize the amount of extracted devices and data size. However, a backannotation will not always be done, cause of the hard and short time-to-market specifications. For most designs a backannotation of some sensitive building blocks will be satisfying.

4.2 Matching Structures

In design circuits building blocks and devices which are doing the same or interacts together should have the same behavior. The behavior in this case is related to variations in temperature, supply voltage or process parameters. This means for example, two MOS-transistors of a current mirror should vary the mirrored current in the same way. The change of the mirrored current in different directions over temperature or voltage is not permissible and will lead into malfunctions. If the two transistors have the same behavior over all variations they have a good **matching**. The matching issue can not be done in the design, it has to be done in the layout. A bad matching layout can kill a good design. This section deals with **failures** and **rules** of matching devices and building blocks. Again, for the matching topic the communication between design and layout engineers is very important.
The first rule of matching is to **place matched devices close together**. If devices are placed close together within a small region the variation between them in temperature and process will be smaller.
Putting devices close together does not guarantee a good matching performance. For matching devices

it is also important to **keep the devices in the same orientation**. The two transistors in figure 4.4 have the same drawn dimension with $W/L = 20/2\,\mu\text{m}$. However, after processing they do not have the some dimensions anymore. This comes from orientation-specific process errors. Some etches, which are used during processing, have a preferred orientation. One of the drawn transistors will be more influenced by the variation of the width and the other one by the variation of its length, because it is placed sideways. This mismatch can be avoided by orientating the matching devices in the way. Of course the values are a little bit excessive, but this should demonstrate failures if matching devices have not the same orientation.

	width	length
drawn:	12	2
after processing :	11.7	2.4

	width	length
drawn:	12	2
after processing :	12.4	1.7

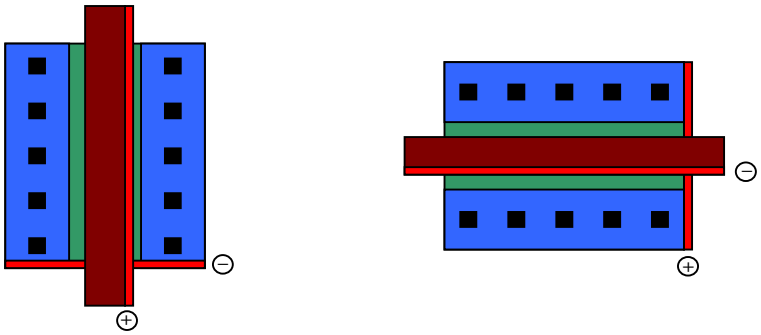

Figure 4.4: Possible mismatch error by orientating two devices in a different way.

Another matching item is the **root device method**. In schematics often more than two devices have to be matched. For example resistor ladders of converters or global bias current mirrors. Figure 4.5a presents five resistors with very different values. The absolute tolerance of resistors is very high at about $20 - 30\,\%$, but the relative tolerance between resistors is much better at about $2\,\%$. This good value can only be achieved, if the resistors match very well. What can be done for the example in figure 4.5a? First the two items from above are important. The resistors have to be placed close together and with the same orientation. Another thing is the width of the resistors. For good matching it is important to have the same width for all resistors. One problem in this example seems to be the very different size of the resistors. It is obvious, that resistors with such different values and the same width, differ in their lengths very much. For matching issues this will be the critical part. To match this resistors a so called **root component** has to be defined. In practice, usually the lowest common factor is used. In this case this would be the resistor with the value of $250\,\Omega$. The largest resistors are still eight times larger as figure 4.5b demonstrates. If these resistors are placed as single stripes they would have much more contacts and therefore much more contact resistance. This seems not to be the best way to match them. A better way to match the resistors is to choose a middle value for the root component. For the example the middle value would be a resistor of $1\,\text{k}\Omega$. For the resistors with smaller values than $1\,\text{k}\Omega$ the single stripes have to be connected in parallel as figure 4.5c shows. The total number of device and therefore the contact resistance has been reduced in this example. The root device method works not only with resistors, all types of devices can be matched using this rule.

However, this is not yet the best matching which can be achieved. With the matching rule of **interdigitate** the matching can again be improved. Interdigitation is the wrapping of single stripes by looking for the root component and place it in the center of the matching block. The root device of figure 4.6 is placed right in the middle and the others are located around it.

Another example for matching two devices with the same size is shown in figure 4.7. The stripes of

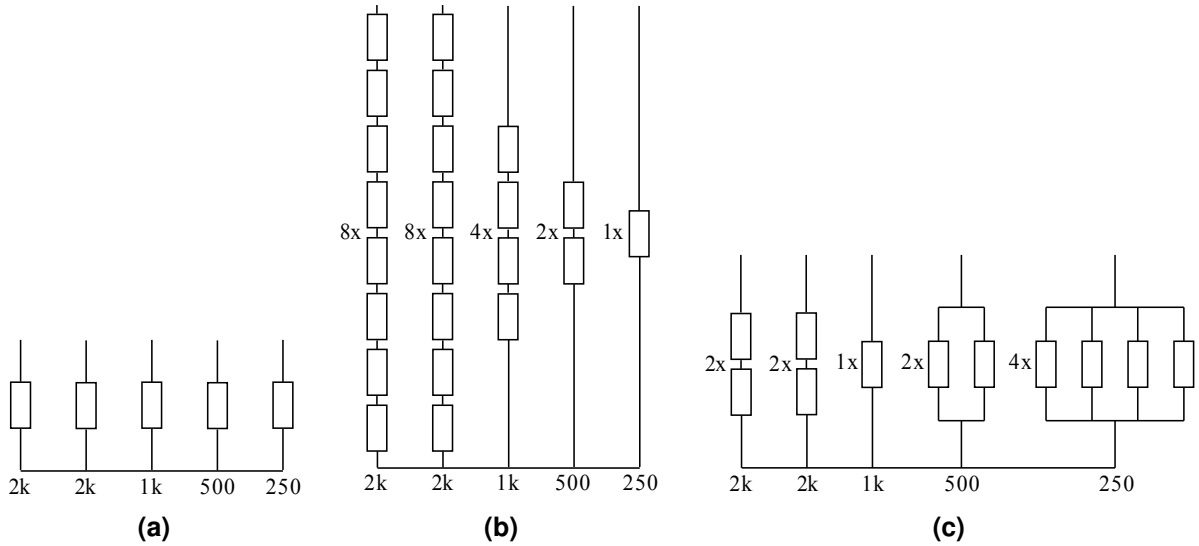

Figure 4.5: Matching several resistors (a) in different ways. Matching by taken the lowest (b) and middle (c) valued resistor.

the two devices are placed alternately. This figure also demonstrates the best wiring for interdigitated components. In the literature the interdigitation can also be found as **simple matching** and is one of the basic rules for layout engineers.

Now, the devices seem to be well matched. However, there are two resistor strings which are different to each other. These are the two on the right and on the left side, because their outer edge is not close to another resistor, it is hanging in free space. During the etching process these two strings will be heavier affected than the other ones. This will end up in different resistor values. To avoid such an influence, the rule of **creating dummies** for devices, which have one edge in the free space, exists. The creation of simple dummies on the right and left side of the resistor block can be seen in figure 4.8a. The influence of the etching process is also marked in this figure. For very sensitive circuits or careful layout engineers this will be not enough. The advanced version of drawing dummies around devices can be seen in figure 4.8b. The whole device matching block is surrounded by a dummy ring. Additionally, very important for the matching is to have the same width and spacings for all devices and the dummy ring.

The next very important rule, especially for the matching of active devices, is the **common centroid** placement. Common centroid means the placing of devices and several parts of them around a common central point. This can be done in one line or on a two-dimensional area. Placing the devices common centroid reduces the influence of thermal or process linear gradients on chips. The thermal influence, generated by a local hot spot, or the process gradient can change the electrical characteristics of a device. For input stages or current mirrors this will generate an offset or mismatch error if the matching devices are not influenced in the same way. With the common centroid method layout engineers try to keep the thermal and process influence of matching devices equal. Figure 4.9 demonstrates different types of common centroid designs. A special kind of the common centroid technique is the **cross quading**, but this is only working for matching pairs. The figure 4.9c shows an example of how to place a cross-quading pair. First each device of the matching pair has to be divided in two halfs and than to be placed cross-corner to each other. This method is very common for analog circuits.

With all these rules passive and active devices can be matched well. However, there can one thing reduce the good matching, while adopt the discussed rules. The **current direction** through matching devices should be the same. The reason is again the influence of the process gradients.

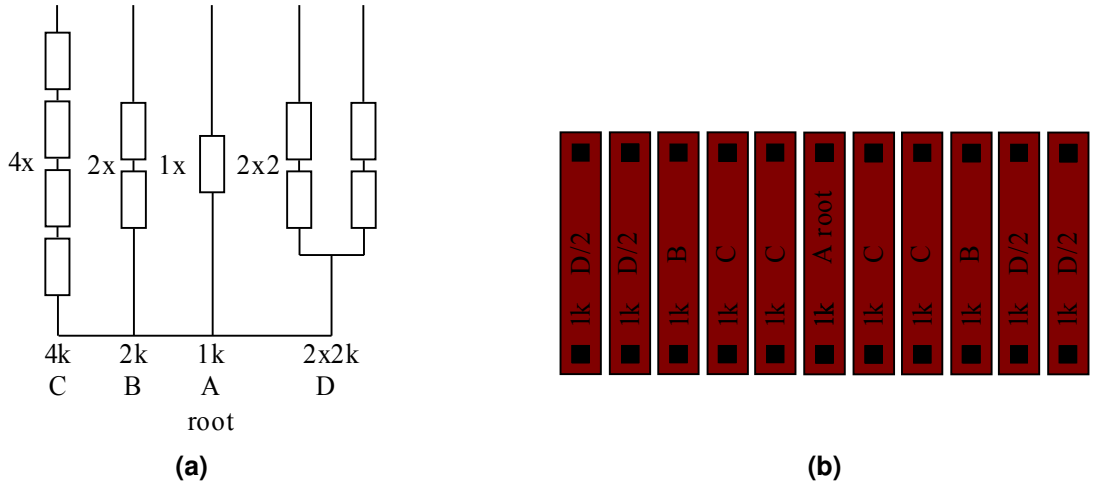

Figure 4.6: Schematic (a) and layout implementation (b) of several resistors with different values by interdigitation.

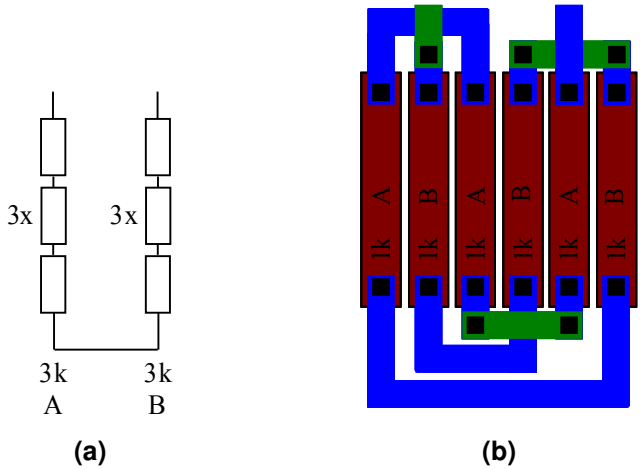

Figure 4.7: Schematic (a) and layout implementation (b) of resistors with the same value by interdigitation.

All these rules above dealt with the matching of passive and active devices, but what is about the interconnect between the devices? Of course the wires, which connect the devices are also very important. As known from section 4.1 all wires have parasitic resistors and between the wires exist parasitic capacitances. For matching two devices it is crucial to connect the ports of the devices with a balanced interconnect. This means the wires have the same metal layers, the same number of vias or contacts and the same width. It will be also important to balance the wire length between the ports of the matching devices. In the example for cross-quading devices (see figure 4.9c) the use of equal wire length is shown. Some ends of the wire connections are not really necessary for the operation of the circuit, but they are drawn to get the same parasitic capacitances. For sensitive circuits the **matching of parasitics on wires** is very important to ensure correct functionality.
In the literature (for example see Christopher Saint [2002]) there can be found much more rules for matching structures on a chip. The above rules can be seen as some kind of basic rules, which should be known by layout engineers. This are also the most important rules for the layout implementation of the video DAC.

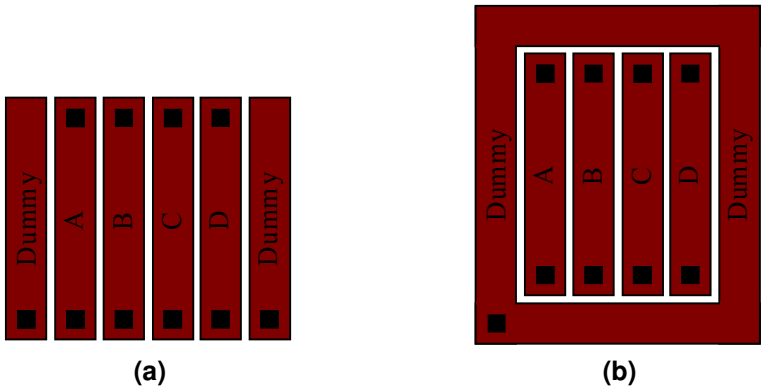

Figure 4.8: Guard matching resistors during etch process with dummies at both sides (a) and around (b).

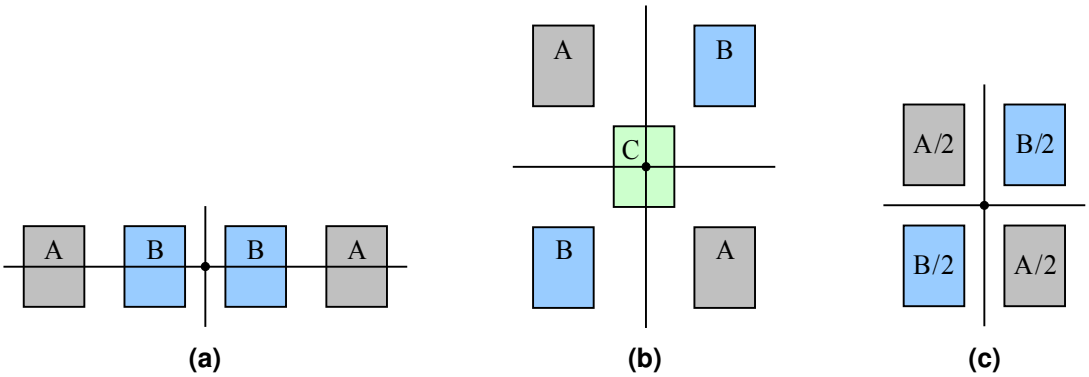

Figure 4.9: Two types of common centroid matching (a and b). A special kind of common centroid matching is cross quading (c). [Christopher Saint, 2002] and [Hastings, 2001]

4.3 Layout Floorplan

The floorplanning is the outline-only design of a chip layout. It should be the starting activity for layout engineers before building single module blocks. For the floorplanning the blocks are only black boxes with inputs and outputs. Additionally, they have two or more power supplies. For the floorplanning the interconnect between these modules and the boundary is essential. The boundary of a chip is the pad-ring or pin-out. The main task during floorplanning is to line up the blocks and define the inputs and outputs on the right side. Furthermore power rails have to be defined for every voltage domain. If the floorplanning is well done, no free spaces should be left between the blocks after composition. During floorplanning phase the floorplan passes through several steps.
First the **pin-out** has to be defined in agreement with the design engineers. The pin-out is very essential for the floorplan, because it defines where the power supply comes from and which ways the signals have to take. The planning of the supply rails is a very hard job, because it defines the structure of the remaining planning tasks. The supply can be a single connection to the pad which runs like a rail through all blocks or radiated from one point, mainly the pad, to all blocks. The second method needs more space on the chip, but it will be better regarding noise issues. During the pin-out planning phase it is also important to think about different voltage domains and the electrostatic discharge (ESD) protection for the pad-ring.
After the pin-out planning the next step will be the **block-driven** floorplanning phase. In this phase the symmetry and reusability of multiple used blocks will be analyzed. The form and interaction of the modules is a general condition for the block-driven floorplanning. Additionally, the blocks have to be

divided in the voltage domains, if more than one is used.
After analyzing the blocks, layout engineers usually start with the **signal-driven** phase. Here the attention is on the signal flow. For most circuits it is important to avoid signal loops. The influence of an amplified output signal to a small and sensitive input signal can have negative effects for the functionality of the circuit. The same is true for other parasitics or noise coupling through the block interconnections. The last point which will be mentioned in this section is the **area estimation** phase. Most layout tools have the possibility to generate the layout view with all devices directly from the schematic view. After that the devices are grouped as they will be in the final version. This means all device for an input stage or bias mirror are placed close to each other. Finally, the metal wires for the power rail and placeholder for the guard rings are placed. Additional some space for the interconnect will be needed. This will generate a quite good estimation for the floorplan. Sometimes the design phase is not ended during the floorplanning phase, so the layout engineers have only the existing schematics and their experience to make the floorplan.

4.4 Process Design Rules

The process design rules regulate the drawing dimensions, overlaps and spacings of the different mask layers. They are manufacturer and process dependent. The placement and routing of chips have to be complied to these rules before tape-out. The mask shop will not produce any mask if the process design rules are violated. The following paragraphs describe the structure of the design rule document and the basic design rules.

4.4.1 The Process Design Document

The process design document is special for every process of a semiconductor manufacturer. This document is company confidential, because it keeps the sensitive information about the process options and parameters. It is split up in several sections, starting with the revision control, where the information about changes can be found. After that, a table of the different process options can be found. These are for example options for two polysilicon layers, high resistive layer for polysilicon resistances, high voltage and so on. The list of the mask and definition layers can also be found in the general section of the process document. One of the important sections for the layout engineer is the description of the **geometric relations**. What does this mean? The geometric relations are the width and spacing but also notch, enclosure, overlap and extension. Table 4.1 and figure 4.10 should illustrate, what these terms mean.

Geometric Relation	Description
A and B	logical intersection
A width	distance inside of A to inside of A
A spacing to B	distance outside of A to outside of B (different polygons)
A notch	distance outside of A to outside of A (same polygon)
A enclosure of B	distance inside of A to outside of B (A contains B)
A extension of B	distance inside of A to outside of B (A may intersect B)
A overlap of B	distance inside of A to inside of B

Table 4.1: Geometric relations and their description.

The section with the general definitions from above is followed by tables with a layer overview. These tables contain the available layers and their minimum widths and spacings. Subsequent, a detailed information of every layer can be found. The detailed information contains every geometrical relation of the layers with a rule identifier. The detailed description of violations during the design rule check can

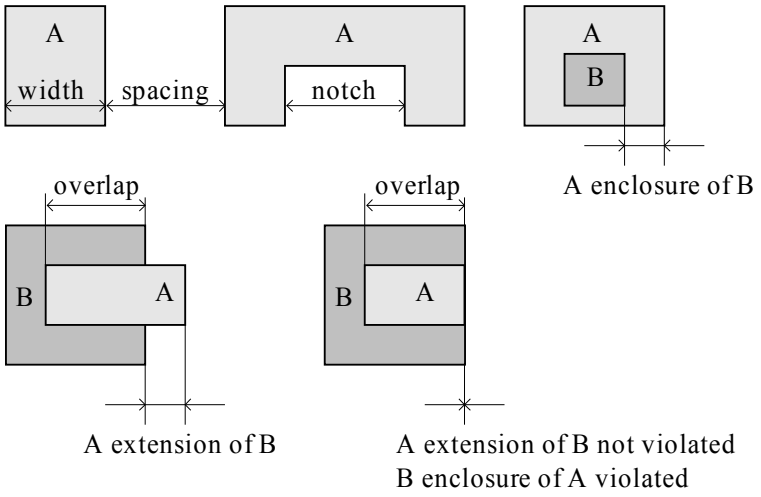

Figure 4.10: Definition of geometric relations inside a process design rule document.

be found by using this rule identifier. At the end of the process design rule document, some information of special element rules, scribe boarder rules, this are the rules for the outer border of a chip, and other guidelines can be found.

4.4.2 Working with the Process Design Rules

For layout engineers is the work with process design rules daily business. Most engineers have an abridgement of the process design rules on their workplace. On this sheet they can find a summary of the most important rules like minimum widths and spacings for the most important layers. Due to the existence of **PCells**, the layout engineers do not have to draw all required layers for devices separately and in compliance with the design rules. The term PCell stands for parameterized cell, where the length, width and values of devices like resistors, capacitances or transistors are adjustable. The layout tools automatically generate the devices out of several layers in compliance with the design rules. This is a very helpful support for layout engineers, that they need the design rules mainly for spacings between devices and the interconnect. To check, if the layout is compliant to the process design rules, most layout tools have a check tool integrated. Hence, more about the layout verification can be found in section 4.5.

4.5 Layout Verification

The layout verification consists of two steps. The first step is to verify if the drawn layout complies to the process design rules (see section 4.4). This step is called the design rules check (DRC). If the DRC reports no errors the next step can be done. Now, it has to be guaranteed, that the drawn layout is exactly what the schematic design says. This second step is called the layout versus schematic check (LVS). Both checks have not to be done manually, because all professional computer aided design (CAD) tools for semiconductor developers have integrated DRC and LVS checker. These tools can also perform some additional checks like the electrical rules check (ERC), the parasitic extraction with backannotation, noise analysis, and so on. Some checks are only optional for critical designs, but the DRC and LVS have to be done every time before tape out. Before starting with a more detailed description about the DRC and LVS it has to be mentioned, that these checks should not only be done at the end of the layout phase. Despite of layouting very carefully, some few design rule violations will occur in every module. Or the schematic changes during layout phase. Hence, it makes sense to check every module on its own before checking the whole layout. Normally a chip consists of hundreds of modules and if the check is

performed primal at the end, this will result in thousands of errors. It will be very difficult to find for example shorts between supplies on the whole chip. If the modules are clean of violations the overall checks will be much easier.
The layout checking is always an iterative process. It has to be started with the DRC. This will usually report some errors, which have to be fixed. While fixing these errors the occurrence of new errors is possible. Therefore the DRC has to be performed until no errors will pop up after. If the DRC is clean, the next step is to perform an LVS. While correcting LVS errors new LVS and/or DRC errors can appear. Hence, the chain has to be started again with a DRC and after that the LVS check has to be done too. Running through this loop would be required until both checks are error free.

4.5.1 DRC

The DRC is a geometrical rule check. It verifies the compliance to the process design rules described in section 4.4. The DRC works with simple Boolean operations. These Boolean operations generate new layers from the drawn layers. For example the AND function generates a new overlap layer, which is shown in figure 4.11. The truth table of the AND function can be found in table 4.2.

AND		
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

Table 4.2: Truth table of the AND function, where A and B are the input values. The output Q is the Boolean AND combination of A and B.

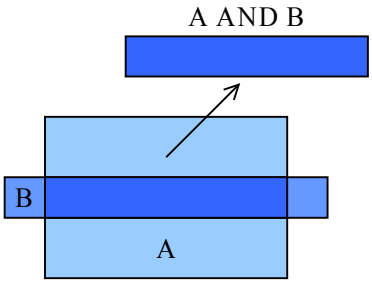

Figure 4.11: The Boolean AND function generates a new overlap layer.

First the check tool generates the temporary layers using the Boolean operations. Hence, this is only the primary step, where no rule checking is done. After the generation of the additional layers the DRC runs line by line through the rule file supported by the semiconductor manufacturer. The comparison of the layout with the design rules is the real verification phase. If the rule file is written well, every mistake will be found and logged. For finding the errors from the DRC, there are two different modes supported. The first and older one is the output of the errors in an simple text file. It is very easy to get this mode to run. The check tool needs only a stream file, which is normally in GDS format. The stream file consists of all layers, which are necessary for the rule check. The output of the check, the error report file, will deliver the information about the rule violation related to the design rules supported by the semiconductor manufacturer. Additionally, the coordinates, where the error can be found, are given. This sounds not very comfortable to search for coordinates in a big layout and than measure where the

violation is. Therefore, most of the check tools support a graphical user interface (GUI). The interface features many additional functions. A snapshot of the Calibre check tool from Mentor Graphics® can be found in figure 4.12. This tool is frame organized, where the left top frame supports the information about the cell name and the type of violation with the number of violations in the result column. The right top frame shows all violations of the selected type presented by the error number. Below this frame are the coordinates, where the error can be found. On the bottom of the window is a frame with additional information about the used rule file and a detailed explanation of the error. Until now, the improvement between the text-based version and the GUI is only a prettier presentation. Hence, the biggest advantage of the GUI is the interactive highlighting. If the interaction between the check tool and the layout editor is enabled, the errors, which are selected in the check tool, are highlighted in the layout editor. Therefore, the layout engineers have not to search for coordinated and measure the distances for example. They can easily select an error in the check tool and highlight this in the layout editor. Afterwards, it will be easy to correct the violation and step to the next error. Certainly, there are many other features supported by the check tool, but this will go beyond the scope of this work.

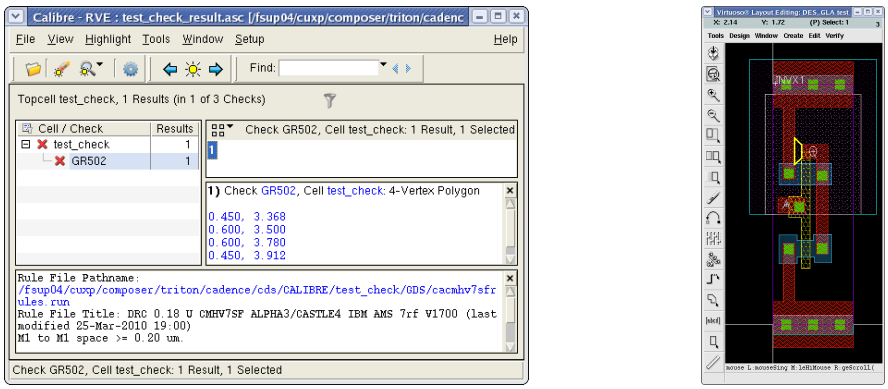

Figure 4.12: Snapshot of the Calibre DRC check window (a) and the layout window with a highlighted DRC error (yellow marker).

4.5.2 LVS

The LVS checks the drawn layout against a given schematic. It checks not only the existence of components, but also the wiring between those components. Additionally, the size of the drawn components is compared against the values in the schematic. The first step of the LVS check is very similar to the DRC. The checker uses the Boolean operations, but this time not to generate temporary layers. The check tool extracts the device information and the wires out of the drawn layout. Hence, this is only the first step, where no checking is done. At the end of the first step a SPICE netlist will be generated out of the extracted device and wire information of the layout. For the schematic also a SPICE netlist will be generated. Thereafter, the second step starts with the comparison of the two SPICE netlists. In this step the device count is analyzed and compared between the two netlists. Then, the tool checks if device types and values are matching. Certainly, the connections between the devices are checked. The result again will be written in an error report file. This file again can be viewed simply in text format or with the GUI. The result window of the check tool for the LVS looks quite similar to the DRC (see figure 4.13a). Hence, the left frame known from the DRC is now the right top and gives information about the cell name and the discrepancies between layout and schematic. Below this frame is again the detailed information frame separated by the connectivity information related to layout and schematic. The left frame is the navigation frame, where the different rule and result files can be selected.
To find the errors in the layout, again an interactive highlighting is implemented. Hence, this highlighting is not only available for the layout, but also for the schematic. The strategy to find and correct the errors

correctly is always the same, whether the text based result or the GUI is used. First, the supply wires have to be LVS error free. An open or short on the supply lines will lead into several errors, because the check tool can not match the devices correctly. A look to the device count and types would be the next step. If this is okay, stepping through the net errors is the next. Finally, the property errors have to be corrected. This are the size, areas and perimeter of the devices. The GUI is again a good assistance to find all errors fast. The problem of the text file this time are the different names of the nets. This means, during layout extraction all nets without labels get different names than the same nets in schematic. Hence, there are also strategies to find the errors by looking at devices, and so on. However, it can be said, correcting DRC errors is easier as correcting LVS errors and if there is a possibility to use a check tool with a GUI the correction of the errors will go faster and more convenient.

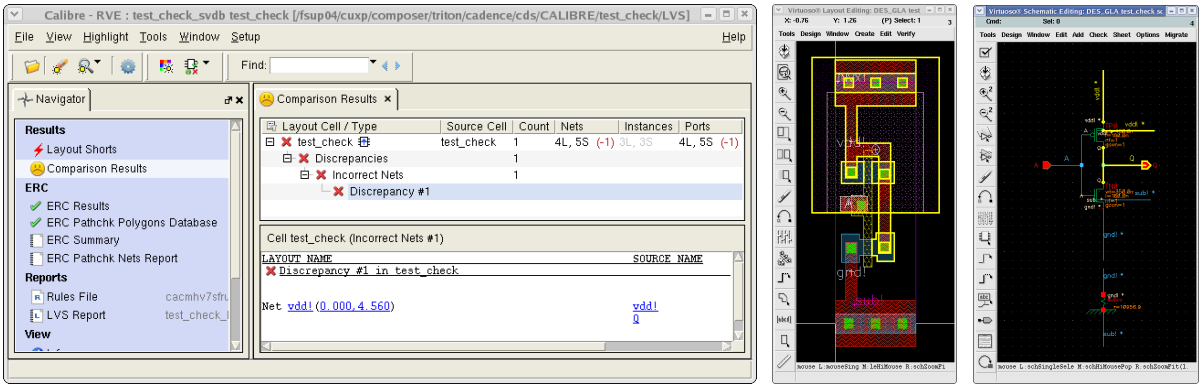

Figure 4.13: Snapshot of the Calibre LVS check window (a), the layout window (b), and the schematic window (c) with a highlighted LVS error (yellow marker).

Chapter 5

Design Implementation

This chapter deals with the design implementation of the video DAC. First the general operation conditions of the video DAC will be specified. The following section starts with the basic calculations for the current unit and ends with the assembly of all pieces to the top schematic. The implementation of the modules described in section 5.2.1, 5.3 and 5.4 is done with the analog framework. Only the digital part in section 5.5 is written in verilog and therefore designed with the digital design flow. After synthesis this block is also imported into the analog framework.

5.1 Basic Conditions for the Current Sources

Before starting with the design of the current sources the basic conditions have to be defined. For this $0.18\,\mu\mathrm{m}$ design a power supply of $1.8\,\mathrm{V}$ with $\pm 5\,\%$ tolerance is used. The input for the DAC is a $10\,\mathrm{bit}$ digital signal with a low level at $0\,\mathrm{V}$ and a high level at $1.8\,\mathrm{V}$. This input stream has a clock frequency of $27\,\mathrm{MHz}$. On the output side is a differential current output pair located. This two outputs have to be terminated with a resistance of $37.5\,\Omega$. Normally this is realized with two $75\,\Omega$ resistors in parallel. Referring to the ITU standard for TV signals the current units have to drive a current throw this resistor to generate a voltage of maximum $1.34\,\mathrm{V}$ for PAL signals. Using Ohm's law a maximum current of $35.7\,\mathrm{mA}$ has to be generated. For temperature specification, the typical temperature range for consumer electronics from $-25\,°\mathrm{C}$ up to $85\,°\mathrm{C}$ is used. Table 5.1 shows the summarized specification.

5.2 Design Decisions and Calculations for the Current Steering DAC

The specification of the last section implicates some fundermental design decisions. The first and most important decision depends to the basic architecture. Due to the high refresh rate of $27\,\mathrm{MHz}$ a current

Parameter	Min	Typ	Max	Unit
Supply Voltage	1.71	1.8	1.89	V
Temperature Range	-25	27	85	°C
Input Clock Speed		27		MHz
Input Data-width		10		bit
Output Termination		37.5		Ω
Output Current			35.7	mA

Table 5.1: Summary of basic conditions.

steering architecture is used. The next important property is the resolution. The specification defines an input data width of 10 bit. This input data width of 10 bit implies a resolution of 10 bit for the DAC. Hence, a topology with a segmented current steering DAC is used. A detailed description about the causes to use the segmentation method for current steering DACs can be found in section 3.1.2 of chapter 3. The 10 bit input is distributed in an unary 6 bit MSB part and a binary 4 bit LSB part. To improve the matching between the unary current sources the 63 unary current sources, which are controlled by the thermometer encoder, are placed four times as array. Therefore one unary current unit drives only a fourth of the calculated unit current.

5.2.1 Design of the Overall System

For the design of the current steering DAC only a few blocks are required. The basic block diagram can be found in figure 5.1.

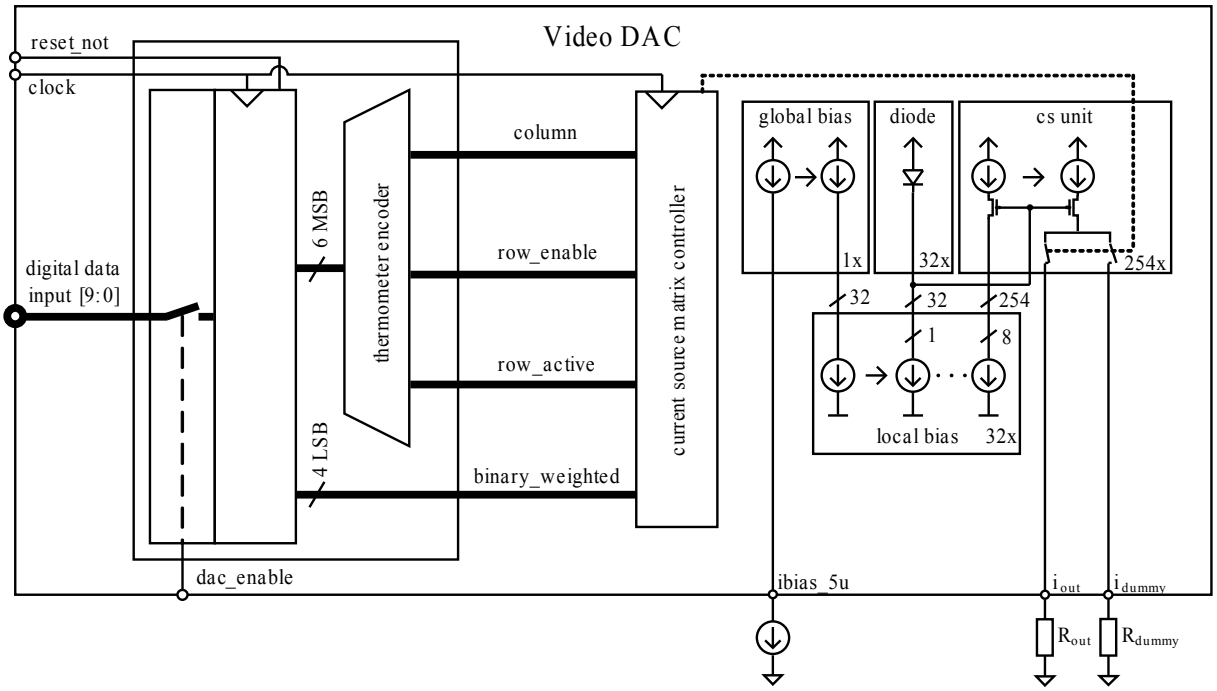

Figure 5.1: Overall system of the current steering DAC.

The only analog input, except the supplies *vdda* and *vssa*, is the current input *ibias_5u*. This analog current input supplies the DAC with a 5 µA constant current. The 5 µA current is the originator of all subsequent currents in the DAC. For the first version of the testchip the 5 µA current is generated externally, but for further versions of the chip, this current should be generated on-chip derived from a bandgap voltage. The value of 5 µA for the main current is chosen to reduce the power consumption of the on-chip current mirrors. A smaller value than 5 µA is not recommended, because this will generate more problems for example with parasitics. Additionally, the transformation ratio for the larger current source units will become a huge value and this triggers more inaccuracies. The external current is applied by an external current sink, because the first internal current mirror is a PMOS current mirror. Backtracking the current from the PMOS current source units on the output will present the following structure. The binary weighted and unary output current source units get the currents from the 32 *local bias* current sinks, which are placed near to the current units. The input currents of this 32 *local bias* modules are generated by the current source mirrors of the *global bias* current mirror module. Therefore, the external input current has to be generated by a current sink.

To reduce the dependency of the output current against the output voltage, all current mirrors are designed with a cascode transistor. Another performance optimization is done by generating and stabilizing the gate voltage of the unary current unit cascode transistor with PMOS transistors connected as diode. Each transistor diode supports 8 unary quarter current units with the required gate voltage.
The other inputs which can be found in the overall diagram (see figure 5.1) are the 10 bit digital inputs and one enable signal. The enable signal is required, because the DAC is designed for mobile applications and therefore, the whole system has to have the possibility to be disabled to reduce the power consumption and improve the battery lifetime. The other 10 bit digital inputs are decoded and spilt up into the binary weighted and thermometer coded part. The control logic generates row and column signals to control the PMOS switches at the output of the unary current units. Both transistor switches of the current units can not be on at the same time, because their enable inputs are inverted. Therefore, the output current of every unit can not be turned off, except with the global enable signal. The output current is only switched between the used current output port i_{out} and the second current output port i_{dummy}. Consequently, the static overall power consumption is independent of the digital input code and the actual output current.
A detailed description and calculation of all building blocks can be found in the following subsections. All schematics and source codes are attached to the appendix A.

5.2.2 Basic Calculations for the Overall System

The main parameter for the calculation of the overall system is the specification of the output to be compliant to PAL and NTSC video signals. Especially the PAL signal is a hard requirement. According to the ITU is a voltage of maximum $1.34\,\mathrm{V}$ for PAL video signals at a double terminated wire required. Therefore, the maximum output current can be calculated as follows

$$I_{max} = \frac{V_{max}}{R_{term}} = \frac{1.34\,\mathrm{V}}{37.5\,\Omega} = 35.73\,\mathrm{mA} \tag{5.1}$$

Due to the fact that the design is based on a 10 bit DAC, the unit current of the LSB can be calculated as

$$I_{LSB} = \frac{I_{max}}{2^{10} - 1} = \frac{35.73\,\mathrm{mA}}{1023} = 34.93\,\mu\mathrm{A} \tag{5.2}$$

5.3 Design of the Current Units

Basically the current units consist only of a view MOS transistors. Due to better noise suppression and the requested output resistor against ground the decision felt to PMOS current sources. The noise suppression of PMOS transistors is also better, because the used $0.18\,\mu\mathrm{m}$ process is a p-substrate process and therefore the PMOS transistors are embedded in an n-doped well. For the segmented current steering DAC two different current units are used: unary and binary current units. The unary current units have all the same dimensions and each unit supports a partial current of the same size. They are used for the 6 MSB of the DAC. The number of units can be calculated as follows

$$number\ of\ units = 2^6 - 1 = 63\ unary\ units \tag{5.3}$$

The binary weighted units are supporting the current of the 4 LSB. Therefore, four current sources with binary weighted output currents are used.
To improve the matching between the unary current units among each other and to the binary current units, they are divided into 4 pieces which support all the same current. A detailed description about the matching reason and the floorplan can be found in the layout implementation chapter (see chapter 7).
The basic structure of the used current steering DAC units is well know from chapter 3. Figure 3.4 in this chapter shows the current source transistor M_{CS}, the cascode transistor M_{Casc} and the two switching

transistors M_{SW0} and M_{SW1}. In figure 5.2 the transistors in the output path are named M_{CS_mirror} and M_{Casc_mirror}. Additionally, the 5 µA mirror transistors for the current source M_{CS} and for the cascode M_{Casc} can be found. Finally, the enable transistor M_{enab} is directly connected to the gate of the current source transistor to disable the output path.

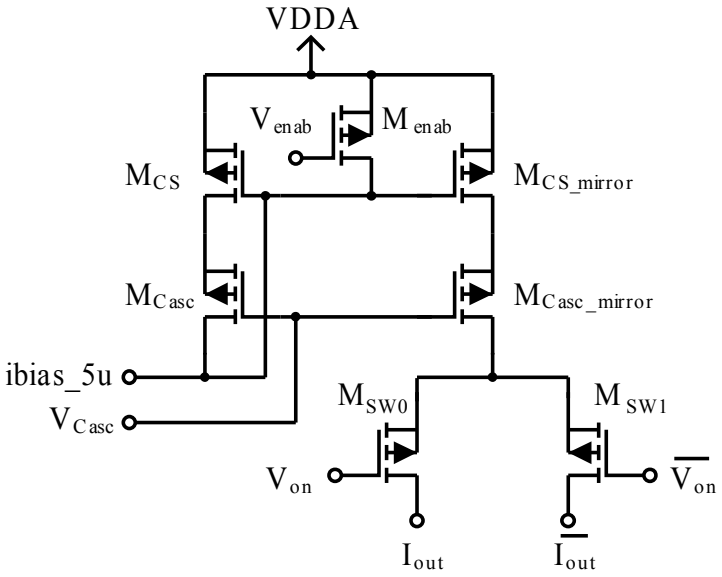

Figure 5.2: Complete current source unit with current mirrors and cascode transistors.

5.3.1 Thermometer Coded Current Units

The main part of the required output current is provided by the unary current units. In principle the design of the units is based on the current source unit described in section 3.2.2. Each thermometer coded current unit supports a part of the current of the 6 MSB. Concerning to the 10 bit architecture one unary unit has to provide a current of 2^4 I_{LSB}. The following equation shows the calculation of the current, which is delivered by one unary current unit

$$I_{unit} = I_{LSB} * 2^4 = 34.93\,\mu\text{A} \cdot 16 = 558.9\,\mu\text{A} \tag{5.4}$$

Concerning the mentioned matching design decisions, one quarter unary unit current has the following value

$$I_{quarter\_unit} = \frac{I_{unit}}{4} = \frac{558.9\,\mu\text{A}}{4} = 139.7\,\mu\text{A} \tag{5.5}$$

This is the basic calculation for the output current, which has to be provided by the current source transistor. Dividing the resulting current of 139.7 µA by the 5 µA input current $ibias\_5\mu\_in$ will not bring an integer number. Thus, the basic input current is a little bit smaller than 5 µA and has to be adjustable. The following equations show the exact input current and the required multiplication factor m.

$$m = \frac{I_{quarter\_unit}}{ibias\_5\mu\_in} = \frac{139.7\,\mu\text{A}}{5\,\mu\text{A}} = 27.94 \approx 28 \tag{5.6}$$

and the excact input current is

$$ibias\_5\mu\_in(exact) = \frac{139.7\,\mu\text{A}}{28} = 4.99\,\mu\text{A} \tag{5.7}$$

For this reason, the current mirror has to be built with a multiplication factor m of 28.
The first component for the square-law equation, the current I_{out}, is now calculated. However, the others

are missing yet. For this considerations the square-law equation (see 3.1) has to be adapted a little bit. The technology depending constants μ_n and C_{ox} are combined to one constant k_{Psat}. The gate-source voltage V_{GS} and the threshold voltage V_{th} are also combined together to the drain-source voltage for saturation V_{DSsat}. With this new simplifications, and neglecting the channel length modulation, the square-law equation for the output current I_{out} can be written as

$$I_{out} = \frac{k_{Psat}}{2} \cdot \frac{W}{L} \cdot V_{DSsat}^2 \tag{5.8}$$

This equation will be used for all current source calculations in this chapter. Consequently, the missing parameters have to be defined. The value for the technology depending process constant k_{Psat} can be found in the process document of the semiconductor manufacturer. There are two different values given, one for the PMOS and another for the NMOS transistors. Each value is defined at the threshold voltage V_{th} and for a transistor with a channel length L of $10\,\mu\mathrm{m}$. The value for the PMOS transistor is $68\,\mu\mathrm{A/V^2}$ and for the NMOS transistor $354\,\mu\mathrm{A/V^2}$. The drain-source voltage V_{DSsat} for each transistor in the output path depends on the output voltage V_{out} and the distribution between this transistors (see figure 5.3). The maximum output voltage implies the minimum voltage for the V_{DSsat} voltages of the transistors in the output path V_{DSsat_all} . The maximum output voltage for V_{out_max} is defined at $1.34\,\mathrm{V}$ accordingly to the PAL signal. Subtracting V_{out_max} from the minimum supply voltage Vdd_{min} gives the minimum voltage for V_{DSsat_CS}, V_{DSsat_Casc} and V_{DSsat_SW}. The minimum supply voltage is defined at

$$Vdd_{min} = Vdd - 5\% = 1.8\,\mathrm{V} - 0.09\,\mathrm{V} = 1.71\,\mathrm{V} \tag{5.9}$$

and V_{DSsat_all}, the overall voltage drop at the PMOS transistors in the output path is

$$V_{DSsat\_all} = Vdd_{min} - V_{out_{max}} = 1.71\,\mathrm{V} - 1.34\,\mathrm{V} = 370\,\mathrm{mV} \tag{5.10}$$

After calculating this V_{DSsat_all} the question is about the distribution between the three transistors. As in section 3.2.2 described, there are different points, which are important for current source units. The overall output resistance has to be as high as possible to guarantee a good performance. Important for the transistors is also, to have enough headroom to work always in saturation. This is a hard criterion, because only $370\,\mathrm{mV}$ and a current of $139.7\,\mu\mathrm{A}$ in the output path makes the transistors not really small. The size is another important criterion. To save chip area and therefore money, the transistors should be small, but as in section 3.2.2 mentioned, the accuracy is indirectly proportional to the product of the width and the length. The company confidential "CMOS Mismatch Parameter" document describes this coherence as the device mismatch is depending on the combination of the threshold voltage V_{th} mismatch and mobility μ mismatch. This mismatches vary in proportions to $1/\sqrt{WL}$. The equations for the standard deviation σ and the at *austriamicrosystems AG* required 3σ design equations can be found in the following lines.

$$\sigma\left(\frac{\Delta I_{DS}}{I_{DS}}\right) \cong \sqrt{\sigma^2\left(\frac{\Delta\mu}{\mu}\right) + \sigma^2\left(\frac{2\Delta V_{th}}{V_{GS} - V_{th}}\right)} \tag{5.11}$$

where

$$3\sigma(\Delta\mu) = \frac{A_\beta \cdot \sqrt{2}}{\sqrt{(W - A_{\beta W})(L - A_{\beta L})}} \tag{5.12}$$

and

$$3\sigma(\Delta V_T) = \frac{A_{V_{th}} \cdot \sqrt{2}}{\sqrt{(W - A_{V_{th}W})(L - A_{V_{th}L})}} \tag{5.13}$$

W and L are width and length of the transistor channel. The process mismatch coefficients A_β, $A_{\beta W}$, $A_{\beta L}$, $A_{V_{th}}$, $A_{V_{th}W}$, and $A_{V_{th}L}$ can be found in the company confidential mismatch document. The coefficients are only given for some specified widths and lengths and therefore the calculation with the

above equations are again a rough approximation. Important is the fact, that for channel lengths larger than the minimum of $0.18\,\mu\text{m}$ the standard deviations $3\sigma(\Delta\mu)$ and $3\sigma(\Delta V_T)$ are going down rapidly. For the calculation of the current source unit transistors two criteria are found. The V_{DSsat} calculation with the square law equation and the relation of width to length W/L and the mismatch equations with the product of width and length given as $1/\sqrt{WL}$. Additionally, for good results at the mismatch equation the length L should be larger than the minimum. For the three PMOS transistors in the current output path of the current source unit this equations have different relevance. The most important transistor is the current source transistor M_{CS_mirror}. The mismatch between all current source units in the output path is directly visible at the accuracy of the DAC. Especially the INL and DNL diagrams show the mismatch between the thermometer coded current source units. This means for the current source transistor to have a length L, which is much longer than the minimum length L of $0.18\,\mu\text{m}$. For the cascode transistor M_{Casc_mirror} the matching is not as important as for the current source transistor M_{CS_mirror}. Therefore, the channel length is not at the minimum, but shorter than the length of the current source transistor. For the switching transistors M_{SW0} and M_{SW1} matching is not important. They only have to switch the unit output current. Consequently, minimum channel length of the process with $L = 0.18\,\mu\text{m}$ is adequate. Additionally, the saturation voltage of the current source transistor should be not to small to have enough headroom for the current source transistor and to keep the relation between W/L as small as possible to save chip area. After several calculations, layout floorplans and some meetings with an expert in analog design, the following V_{DSsat} distribution (see figure 5.3), for the transistors in the output path of the current source unit, were found.

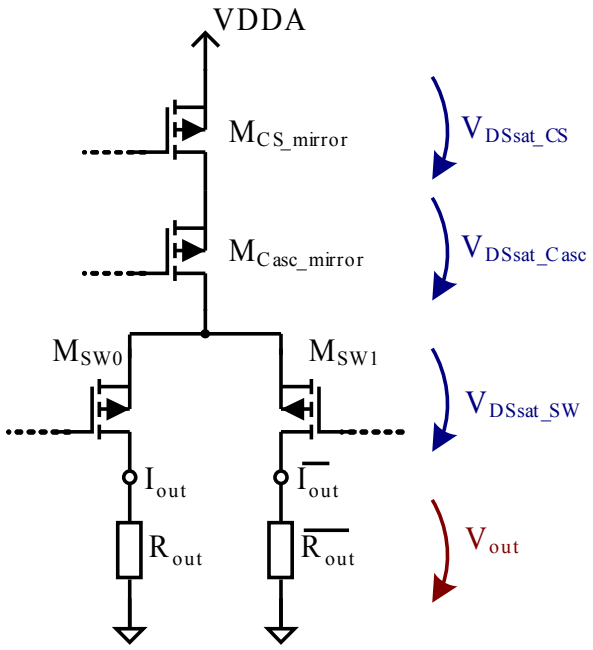

Figure 5.3: Saturation voltage V_{DSsat} distribution for the output path.

This values are only a first approach, which have to be verified by simulation, because equation 5.8 shows a very simplified model of the transistor characteristics. Changes of the saturation voltages and the transistor dimensions are also possible after simulation. With the knowledge of these values equation 5.8 can be written differently to get the ratio between the width W and the length L

$$\frac{W}{L} = \frac{2 \cdot I_{out}}{k_{Psat}} \cdot \frac{1}{V_{DSsat}^2} \tag{5.14}$$

Transistor Name	V_{DSsat} [mV]	Length [µm]
M_{CS_mirror}	200	0.9
M_{Casc_mirror}	70	0.4
M_{SW0_mirror}	100	0.18
M_{SW1_mirror}	100	0.18

Table 5.2: First approach of the saturation voltage levels and the channel lengths for the transistors in the output path.

Calculation of the First Approach

With the values given from above the calculation for the W/L-ratio of the current source transistor is

$$\frac{W_{CS\_mirror}}{L_{CS\_mirror}} = \frac{2 \cdot I_{out}}{k_{Psat}} \cdot \frac{1}{V_{DSsat}^2} = \frac{2 \cdot 139.7\,\mu\text{A}}{68\,\mu\text{A/V}^2} \cdot \frac{1}{(0.2\,\text{V})^2} = 102.7 \tag{5.15}$$

With this W/L-ratio and the length value of $0.9\,\mu\text{m}$ from table 5.2 the width of the current source transistor can be calculated.

$$W_{CS\_mirror} = 102.7 \cdot L_{CS\_mirror} = 102.7 \cdot 0.9\,\mu\text{m} = 92.4\,\mu\text{m} \tag{5.16}$$

For this current source unit the dividing factor is 28 (see equation 5.6). The width for the current source transistor on the input path can be calculated as

$$W_{CS} = \frac{W_{CS\_mirror}}{28} = \frac{92.4\,\mu\text{m}}{28} = 3.3\,\mu\text{m} \tag{5.17}$$

The next calculation is done for the cascode transistor M_{Casc}. The ratio between W and L can be written as

$$\frac{W_{Casc\_mirror}}{L_{Casc\_mirror}} = \frac{2 \cdot I_{out}}{k_{Psat}} \cdot \frac{1}{V_{DSsat}^2} = \frac{2 \cdot 139.7\,\mu\text{A}}{68\,\mu\text{A/V}^2} \cdot \frac{1}{(0.07\,\text{V})^2} = 838.5 \tag{5.18}$$

Again taking the calculated ratio and the length of $0.4\,\mu\text{m}$ from table 5.2 the width can be calculated as

$$W_{Casc\_mirror} = 838.5 \cdot L_{Casc\_mirror} = 838.5 \cdot 0.4\,\mu\text{m} = 335.4\,\mu\text{m} \tag{5.19}$$

Similar to the current source transistor on the output path the cascode transistor has a counterpart in the input path. Due to the relations between the input and output current again the multiplication factor can be used to calculate the width of the cascode transistor in the input path.

$$W_{Casc} = \frac{W_{Casc\_mirror}}{28} = \frac{335.4\,\mu\text{m}}{28} = 12\,\mu\text{m} \tag{5.20}$$

For the calculation of the two switching transistors the following equation is used

$$\frac{W_{SW}}{L_{SW}} = \frac{2 \cdot I_{out}}{k_{Psat}} \cdot \frac{1}{V_{DSsat}^2} = \frac{2 \cdot 139.7\,\mu\text{A}}{68\,\mu\text{A/V}^2} \cdot \frac{1}{(0.1\,\text{V})^2} = 410.9 \tag{5.21}$$

The length of the switching transistors can be at the process minimum of $L = 0.18\,\mu\text{m}$. The width can be calculated as

$$W_{SW} = 410.9 \cdot L_{SW} = 410.9 \cdot 0.18\,\mu\text{m} = 74\,\mu\text{m} \tag{5.22}$$

This values are the basis to go into simulation and improve the overall result.

5.3.2 Binary Weighted Current Units

The binary weighted current units are responsible for the 4 LSB parts of the output current. Differently to the thermometer coded current sources, the binary weighted current units do not provide an output current of the same size. For the 4 LSBs 4 different current units with an output current multiplication factor of 1, 2, 4 and 8 are needed. The output current of 1 LSB current source unit has the same value as 1 thermometer coded current source unit divided by 16. As mentioned above the thermometer current source units are split up into 4 parts and therefore the binary weighted current source unit with the output current multiplication factor of of 4 LSBs has the same dimensions than one thermometer coded current source unit. For the 8 times LSB current source unit two of these cells are needed. Only the units with the current multiplication factor of 1 and 2 have to be built separately. Due to the weaker driving strength they have smaller transistor dimensions and therefore they are combined together into one cell. Figure 5.4 shows this combined binary weighted current source unit.

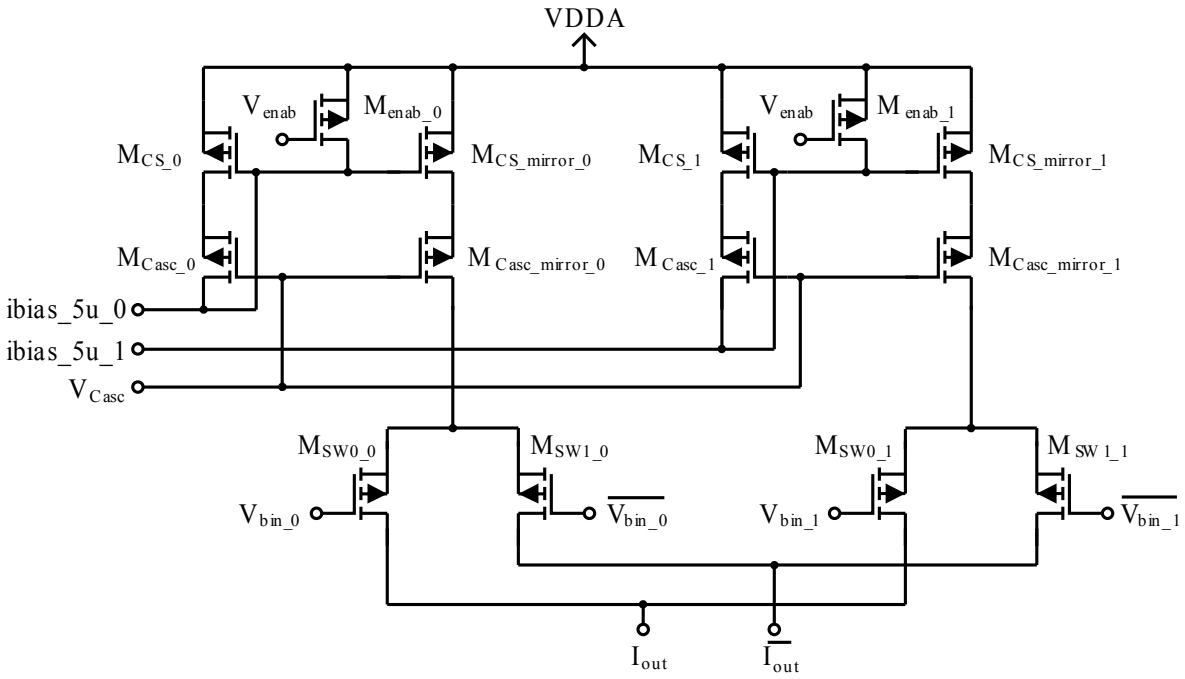

Figure 5.4: Combined binary weighted current source unit with the driving strength of 1 and 2 LSB currents.

The calculations of the output currents of the binary weighted current units is very similar to the unary current units. As mentioned above, the current source unit for the current multiplication factor of 4, which represents the 3rd bit of the 4 LSB, provides a current of the same size than the quarter unary current source unit with a provided output current of $139.7\,\mu\text{A}$. Consequently, the same current source unit can be used. The same is true for the unit with the current multiplication factor of 8, which presents the 4th bit, but here are two units needed to get the output current of

$$I_{bin\_8} = I_{LSB} \cdot 2^3 = 34.93\,\mu\text{A} \cdot 8 = 279.4\,\mu\text{A} \tag{5.23}$$

The calculation of the current for the binary weighted unit with the factor 2 is very similar and can be written as

$$I_{bin\_2} = I_{LSB} \cdot 2^1 = 34.93\,\mu\text{A} \cdot 2 = 79.9\,\mu\text{A} \tag{5.24}$$

And finally the smallest unit with the multiplication factor of 1 is equal to the calculated LSB current of $34.93\,\mu\text{A}$.

Calculation of the First Approach

The calculation of the transistor dimensions for the first approach is like the calculation of the currents required. For the length L of the transistors the same values are used as above for the unary current unit transistors, $0.9\,\mu\mathrm{m}$ for the current source and $0.4\,\mu\mathrm{m}$ for the cascode transistor. The dimensions of the output switches have less relevance for the output current and therefore they have all the same dimensions. The calculation of the widths of the different transistors can easily done by multiplying the input current transistors dimensions with the multiplication factor m. The following equation shows the correlation between the width W of the transistors on the current input path and the width W_{mirror} of the transistors on the mirrored output path depending on the multiplication factor m

$$W_{mirror} = W \cdot m \tag{5.25}$$

The multiplication factor m is determined by dividing the required output current I_{out} with the $5\,\mu\mathrm{A}$ input current $i\_bias\_5u\_in$. This equations are used to fill in the dimensions for the current source M_{CS} and cascode transistors M_{Casc} in table 5.3.

$i\_bias\_5u\_in$ [µA]	I_{out} [µA]	m	W [µm]	W_{mirror} [µm]	Transistor
5	34.9	7	3.3	23.1	M_{CS1}
5	34.9	7	12	84	M_{Casc1}
5	79.9	14	3.3	46.2	M_{CS2}
5	79.9	14	12	144	M_{Casc2}
5	139.7	28	3.3	92.4	M_{CS4}
5	139.7	28	12	288	M_{Casc4}

Table 5.3: Calculation of the transistor dimensions for the binary weighted units.

5.4 Design of the Bias Sources

For this current steering DAC two different current bias sources are needed, the global bias current source, and the local current bias sources. The global bias source supplies 32 of the local bias sources and is placed only once. Each of the local bias sources is supplying 8 of the current source units and one transistor diode for the cascode gate voltage generation. Every unit of the current sources, global and local, provides currents of $5\,\mu\mathrm{A}$. This means, all current sources generate one by one copies of the external $5\,\mu\mathrm{A}$ reference current.

5.4.1 Global Bias Source

The global bias current source consists only of PMOS current mirrors. Additionally, this current mirrors have cascode transistors to keep the output resistance high and be more or less independent against the output voltage. For power saving issues, the current source can also be disabled with a power-down signal. The global biasing cell gets its input current from an external current sink. This external reference current has to be very accurate and quite regarding noise issues. Any interference at this input current would influence every enabled current unit and therefore directly the current DAC output. The global current bias cell has 32 current outputs, which supplies the 32 local biasing cells. Figure 5.5 shows an overview of the global current biasing cell. The current mirrors are one by one mirrors with an input and output current of $5\,\mu\mathrm{A}$. The available voltage for both transistors, the current source M_{CS_mirror} and the cascode M_{Casc_mirror} transistor is again only $370\,\mathrm{mV}$. However, this time no voltage drop is needed

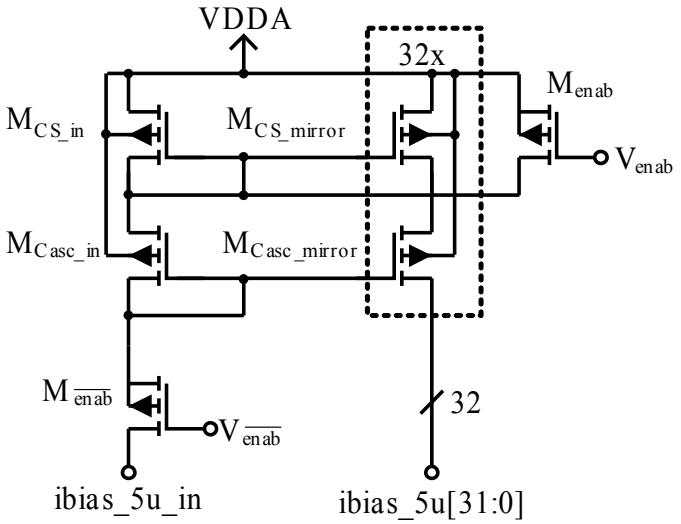

Figure 5.5: The global bias circuit is built of PMOS transistors and supports 32 local bias cells with a 5 µA current each.

for a switching transistor and the current is only 5 µA. The V_{DSsat} distribution for the first approach is 200 mV for M_{CS_mirror} and 170 mV for M_{Casc_mirror}. The chosen length for the transistor is 2 µm for the important current source transistor and 0.4 µm for the cascode transistor.

Calculation of the First Approach

The calculation for the widths of the transistors in the global bias source module is very similar to the calculation for the transistors in the output path. To calculate the W/L-ratio the transformed square law equation 5.14 is used.

$$\frac{W_{CS}}{L_{CS}} = \frac{2 \cdot I_{out}}{k_{Psat}} \cdot \frac{1}{V_{DSsat}^2} = \frac{2 \cdot 5\,\mu\text{A}}{68\,\mu\text{A/V}^2} \cdot \frac{1}{(0.2\,\text{V})^2} = 3.68 \tag{5.26}$$

With this W/L-ratio and the length value of 2 µm the width of the current source transistor can be calculated.

$$W_{CS} = 3.68 \cdot L_{CS} = 3.68 \cdot 2\,\mu\text{m} = 7.36\,\mu\text{m} \tag{5.27}$$

The same calculation again for the cascode transistor width and a V_{DSsat} of 170 mV.

$$\frac{W_{Casc}}{L_{Casc}} = \frac{2 \cdot I_{out}}{k_{Psat}} \cdot \frac{1}{V_{DSsat}^2} = \frac{2 \cdot 5\,\mu\text{A}}{68\,\mu\text{A/V}^2} \cdot \frac{1}{(0.17\,\text{V})^2} = 5.09 \tag{5.28}$$

Followed by the calculation of the width for the cascode transistor with the W/L-ratio from above and a length of 0.4 µm.

$$W_{Casc} = 5.09 \cdot L_{Casc} = 5.09 \cdot 0.4\,\mu\text{m} = 2.04\,\mu\text{m} \tag{5.29}$$

Adapt this width to a process related value, 2.04 µm is rounded to 2.1 µm.

5.4.2 Local Bias Source

The 32 local bias current modules supply all 256 unary and binary current units with a 5 µA bias current. They are all built as NMOS current mirrors with cascode transistors. Additionally, all bias sources can also be disabled with the power-down signal to save power in mobile devices. Figure 5.6 presents the

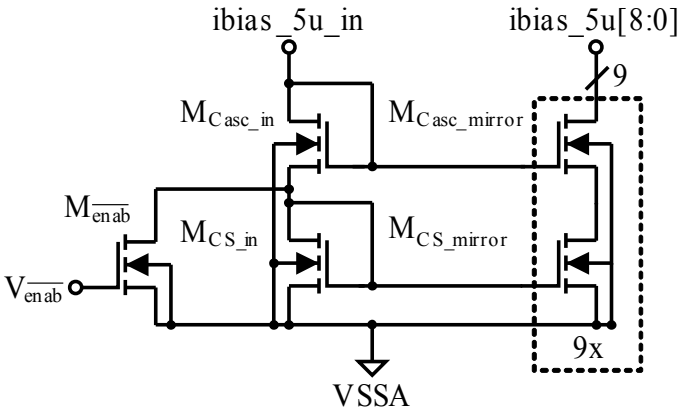

Figure 5.6: One of 32 local bias cells built of NMOS transistors with one $5\,\mu\mathrm{A}$ current input and nine $5\,\mu\mathrm{A}$ current outputs.

local biasing module including the power-down transistor. The calculation of the local bias source can be found in the next section. The NMOS mirrors are also one by one mirrors with a $5\,\mu\mathrm{A}$ input current from the global bias source and eight $5\,\mu\mathrm{A}$ output currents for eight current source units and one $5\,\mu\mathrm{A}$ output current for the cascode gate voltage bias diode. For the local bias transistor calculation is the V_{DSsat} value no problem, because at the calculated operation point $1.34\,\mathrm{V}$ for the current source and cascode transistor are available. Splitting this voltage into two parts results in a chosen voltage of $700\,\mathrm{mV}$ for the current source transistor M_{CS_mirror} and the remaining $640\,\mathrm{mV}$ for the cascode transistor M_{Casc_mirror}. The chosen lengths are, similar to the global bias module, $L = 2\,\mu\mathrm{m}$ for M_{CS_mirror} and $L = 0.4\,\mu\mathrm{m}$ for M_{Casc_mirror}.

Calculation of the First Approach

For the calculation of the local bias source transistors the same transformed square law equation 5.14 as above is used. One difference to the other calculations are the used NMOS transistors. This NMOS transistors have better characteristics than the PMOS transistors in this p-substrate process. The process constant k_{Psat} is about five times better for NMOS transistors. With the k_{Psat} value of $354\,\mu\mathrm{A/V^2}$ the W/L-ratio can be calculated as

$$\frac{W_{CS}}{L_{CS}} = \frac{2 \cdot I_{out}}{k_{Nsat}} \cdot \frac{1}{V_{DSsat}^2} = \frac{2 \cdot 5\,\mu\mathrm{A}}{354\,\mu\mathrm{A/V^2}} \cdot \frac{1}{(0.7\,\mathrm{V})^2} = 0.06 \tag{5.30}$$

Taking the length value of $2\,\mu\mathrm{m}$ and the calculated W/L-ratio, the current source transistor width gets the following value.

$$W_{CS} = 0.06 \cdot L_{CS} = 0.06 \cdot 2\,\mu\mathrm{m} = 0.12\,\mu\mathrm{m} \tag{5.31}$$

However, this value is very low and under the minimum structure width of the process. Due to a better $W \cdot L$ product and the available chip area the layout and design decision fell to a width of $1\,\mu\mathrm{m}$ for M_{CS_mirror}.

The calculation for the cascode transistor width is identical except the saturation voltage $V_{DSsat} = 640\,\mathrm{mV}$.

$$\frac{W_{Casc}}{L_{Casc}} = \frac{2 \cdot I_{out}}{k_{Nsat}} \cdot \frac{1}{V_{DSsat}^2} = \frac{2 \cdot 5\,\mu\mathrm{A}}{354\,\mu\mathrm{A/V^2}} \cdot \frac{1}{(0.64\,\mathrm{V})^2} = 0.07 \tag{5.32}$$

Followed by the calculation of the width for the cascode transistor with the W/L-ratio from above and a length of $0.4\,\mu\mathrm{m}$.

$$W_{Casc} = 0.07 \cdot L_{Casc} = 0.07 \cdot 0.4\,\mu\mathrm{m} = 0.028\,\mu\mathrm{m} \tag{5.33}$$

This is again a very small value. Due to layout and design decisions a width of $W = 1\,\mu\mathrm{m}$ is chosen.

5.4.3 Cascode Gate Voltage Bias Diode Module

The gate input of the current unit cascode transistor is not connected to drain to generate the gate voltage, it has its own stabilized gate voltage bias. This has the advantage of the independence to the output voltage and it can be exactly adjusted to optimize the current source unit. Every generated voltage is also stabilized by seven transistor capacitors with a layout given dimension of $W = 5\,\mu\text{m}$ and $L = 2\,\mu\text{m}$. Figure 5.7 presents the circuit of the module. Before the dimensions of the transistor diode can be

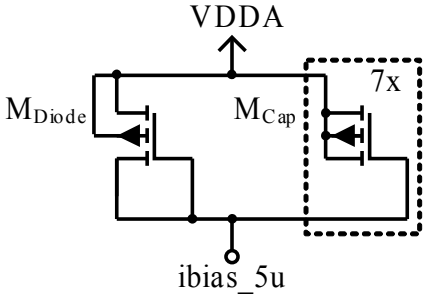

Figure 5.7: Transistor diode with transistor capacitance for the cascode gate voltage.

calculated, the required gate voltage has to be identified. From the square law model of equation 3.1 the following relations can be declared.

$$V_{DSsat} = V_{GS} - V_{th} \tag{5.34}$$

This means the saturation voltage V_{DSsat} is the gate-source voltage V_{GS} minus the threshold voltage V_{th}. The value of V_{th} is defined in the process document and about $400\,\text{mV}$ depending on the transistor dimensions. For the first approach we take this approximated value. The gate-source voltage V_{GS} can be calculated from the values in table 5.2. The source potential of the cascode transistor is the minimum supply voltage Vdd_{min} minus the saturation voltage V_{DSsat} of the current source transistor V_{DSsat_CS}. Combined with the threshold voltage V_{th} the required gate potential V_G of the cascode transistor can be calculated.

$$V_G = Vdd_{min} - V_{DSsat\_CS} - V_{th} = 1.71\,\text{V} - 0.2\,\text{V} - 0.4\,\text{V} = 1.11\,\text{V} \tag{5.35}$$

The required gate voltage V_G has to be $1.11\,\text{V}$. This voltage is generated by a transistor diode with a $5\,\mu\text{A}$ constant current and a length of $L = 2\,\mu\text{m}$. The width of this transistor diode is calculated in the following first approach.

Calculation of the First Approach

By connecting the gate to the drain of the transistor the PMOS transistor is always in saturation and the above used equations are valid. The saturation voltage is again the difference between the minimum supply voltage Vdd_{min} and the maximum output voltage $Vout_{max}$. This is $370\,\text{mV}$, which is calculated in equation 5.10.

$$\frac{W_{Diode}}{L_{Diode}} = \frac{2 \cdot I_{out}}{k_{Psat}} \cdot \frac{1}{V_{DSsat}^2} = \frac{2 \cdot 5\,\mu\text{A}}{68\,\mu\text{A/V}^2} \cdot \frac{1}{(0.37\,\text{V})^2} = 1.1 \tag{5.36}$$

Followed by the calculation of the width for the diode with the W/L-ratio and the above selected length of $2\,\mu\text{m}$.

$$W_{Diode} = 1.1 \cdot L_{Diode} = 1.1 \cdot 2\,\mu\text{m} = 2.2\,\mu\text{m} \tag{5.37}$$

5.5 Design of the Digital Decoder

The digital interface to the current steering DAC is the digital decoder. However, before the 10 bit digital inputs are decoded they are sampled at the negative edge of the input clock to avoid inconsistencies of the input data. The negative edge is chosen to have a half clock period time to switch the output current. Hence, the input data is strobed with a clock frequency of 27 MHz. Figure 5.8 shows the timing of the input data *"pattern_in"*, sampled data *"pattern_s_h"* and local switching data *"sw_data"* depending on the input clock *"clk_27MHz"*. The consecutive data words are named D_n, D_{n+1}, and so on. The de-

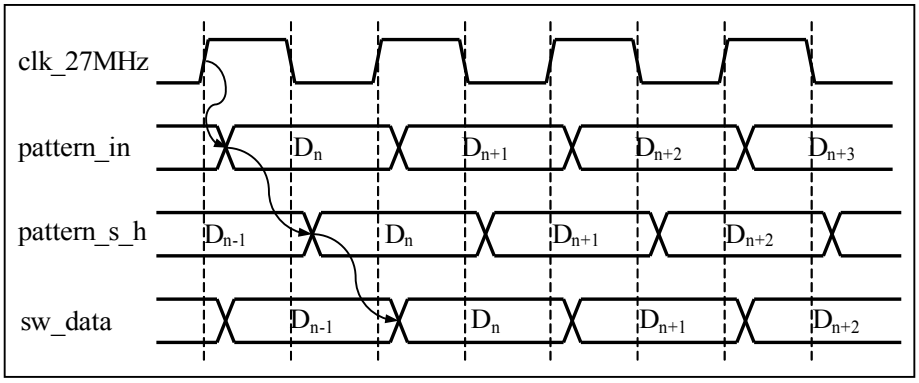

Figure 5.8: Timing diagram for the digital input data path.

coder is structured into a global decoder, which generates enable signals for the rows and columns of the thermometer decoded units and enable signals for the binary decoded units. All local decoders have also built in a sequential cell to save the decoded data. The disadvantage is a loss of one clock period during decoding and this time has to be added to the settling time. Nevertheless, the output current changes its values with steeper edges, because all units switch nearly at the same time. The outputs of the decoder and inputs of the current units are low active signals. The reason is the use of PMOS switches for switching the output current.
The decoding scheme of the thermometer encoder can be found in section 3.2.3. The thermometer encoder is the main part of the global decoder and is written as verilog source code and synthesized. This code can be found in appendix A. The local decoders for the thermometer current units and the binary weighted current units are place locally at every current source unit and this circuit is done in an old style by drawing schematics with digital logic cells. The following sections explains in detail the local decoding units.

5.5.1 Decoder for the Thermometer Current Units

The local decoder for the thermometer units is a simple logic cell with only a few gates. The cell has three decoding inputs and a clock input for the sequential cell, and provides the enable and inverted enable signal at the output. The three decoding signals *col, row_en* and *rown* have the following functionality. The *rown* signal enables the whole row regardless which values the other two signals have. If the *rown* is not selected both signals, *col* and *row_en*, have to be selected to enable the output. The following truth table (see table 5.4) shows the relationship of the input and output signals. In this table only the terms *"active"* and *"inactive"* are used to neglect the usage of high or low-active logic. Only one of the two outputs is shown in the table, because the other has exactly the opposite value.

5.5.2 Decoder for the Binary Weighted Current Units

Similar to the local decoder for the thermometer units, the binary weighted units have also a clock input and a sequential cell. Different is the fact, that no decoding is needed for the binary weighted units,

rown	col	row_en	output
inactive	inactive	inactive	inactive
inactive	inactive	active	inactive
inactive	active	inactive	inactive
inactive	active	active	active
active	x	x	active

Table 5.4: Truth table of the local thermometer decoding logic.

because the input signal is in binary representation. Therefore, the logic is very easy. If one of the 4 LSB inputs is selected, the sequential cell will enable the corresponding output.

Chapter 6

Simulation Test-benches and Results

This chapter presents simulation test-benches and optimized results of the video DAC building blocks. Due to the fact that a DAC has a digital input and an analog output, it is obvious to use not only analog but also mixed-signal and digital test-benches. This means, that the digital input stimuli and the digital part of the DAC are simulated with a digital simulator and for the analog part an analog solver is used. The company, where this work was done, uses mainly tools from the EDA provider Cadence®. For digital simulation the tool NCsim and for analog simulations Virtuoso Spectre Circuit Simulator is used. The tool, which combines both simulations, is called Cadence® Virtuoso® AMS Designer, where AMS stands for analog mixed signal. However, before a mixed signal simulation is started, both sides, the analog and the digital side, have to be simulated on their own. It is very time intensive to set up a mixed signal simulation and the simulation with a bigger stimulus will also not finish in some seconds. The following sections are split up in this analog, digital and mixed signal test-benches.

6.1 Analog Simulations and Results

All analog simulations and results are summarized in this section. The section is subdivided in several building blocks starting with the current source unit as main component of the video DAC. Every subsection contains the used simulation schematic and the resulting diagrams of the simulations. There is also a comparison between the calculated values from the first approach of chapter 5 and the final values, optimized and improved by simulation, included. This values can be found in a table. Another table shows the setup of the ideal voltage and current sources used for simulation.

6.1.1 Simulations of the Current Source Unit

The beginning of all simulations is the analog simulation of the main building block. This block is the current source unit. The goal of the basic simulations is to verify the functionality of the current source unit over the whole output voltage range. In the last chapters always the output current is discussed, but to guarantee a constant output current the saturation voltage at the current source M_{CS} and cascode transistor M_{Casc} is most important. The minimum saturation voltage for M_{CS}, M_{Casc} and the switching transistor is calculated in equation 5.10 of section 5.3.1. For the current source units in a current steering DAC, it is very important to provide a constant output current independently of the input code. However, different input codes changes the overall output current of the DAC and therefore the voltage drop at the output termination resistor. If the current source is designed for a to high saturation voltage, which can not be guaranteed at the maximum output voltage, the current source will not keep in saturation region and can not provide the constant current any more. To optimize the transistor values a output voltage sweep at Vdd_{min} is done. Figure 6.1 shows the simulation environment for one quarter current source

approach	$vdda$ [V]	$vout$ [V]	$ibias$ [µA]	$vbias$ [mV]
first	1.71	0 - 1.34	5	1100
final	1.71	0 - 1.34	5	1051

Table 6.1: Simulation setup for the current source unit.

unit with this voltage sweep. The transistor values in this figure are from the first approach. Table 6.1 shows the simulation setup of the ideal current and voltage sources.

The first simulations are done with the calculated values of chapter 5. This calculations use only the simplified square-law model. After many optimization steps and going back to the roots a few times, an improved result can be presented in figure 6.2. This figure shows the first and final approach results, where the first approach is red and the final approach magenta colored. A summary of the transistor dimensions between first and final approach can be found in table 6.2. The values are separated into two rows, where the first row shows the "first" and the second one the "final" approach. This two rows should illustrate the improvement between the first approach and the final values, which are determined during simulation. The values which are presented in the "final" row, are implemented on the chip. The difference between the values of the two approaches can be explained by the usage of different models. For the calculation of the first approach only a simplified model is used, but the analog solver of the simulator uses the very complex models provided by the semiconductor manufacturer. The improvement of the two approaches is the reduction of the output current difference $\Delta Iout$. It is measured between the start and end points of the output voltage ($Vout$) sweep. The sweep is starting at 0 V and going up to the maximum voltage for PAL signals at 1.34 V. The $\Delta Iout$ value can also be found in table 6.2.

approach	W_{CS} [µm]	L_{CS} [µm]	W_{Casc} [µm]	L_{Casc} [µm]	W_{SW} [µm]	L_{SW} [µm]	$\Delta Iout$ [nA]
first	3.3	0.9	12	0.4	74	0.18	179.8
final	6	0.9	10	0.4	16.3	0.18	77.4

Table 6.2: Calculated and simulated values for the current source unit.

6.1.2 Simulations of the Bias Current Generator

After simulating and optimizing the output path, the 5 µA input current with the bias current mirrors have to be simulated. As mentioned in chapter 5 the bias current is distributed by two bias modules. These modules are the global and the local current bias cells. Additionally, the gate voltage of the cascode transistor is generated by a transistor diode. The following simulation setup allows to simulate the whole current path, beginning at the 5 µA input current $ibias\_5u\_in$ through the current mirrors to the output current of the current units. The reason of simulating all components is to know about the impact on the output current by changing the current mirrors at the input. Additionally, the same simulation schematic (see figure 6.3) can be used to simulated the Monte-Carlo distribution and the process and specification corners. However, these simulations can be found in section 6.1.3.

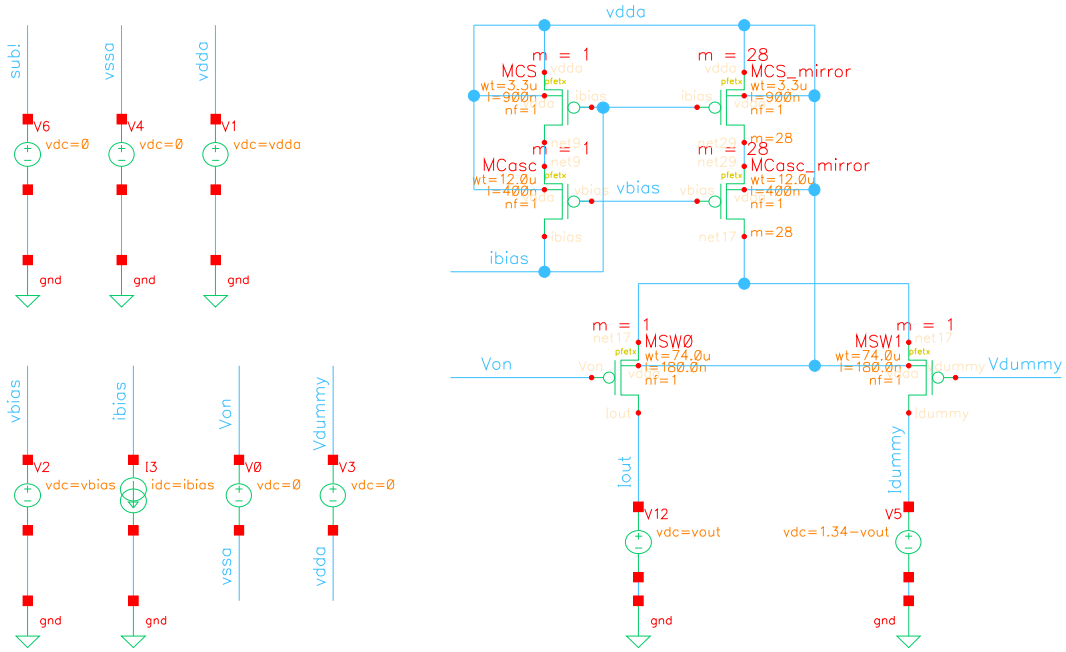

Figure 6.1: Simulation schematic for the current source unit.

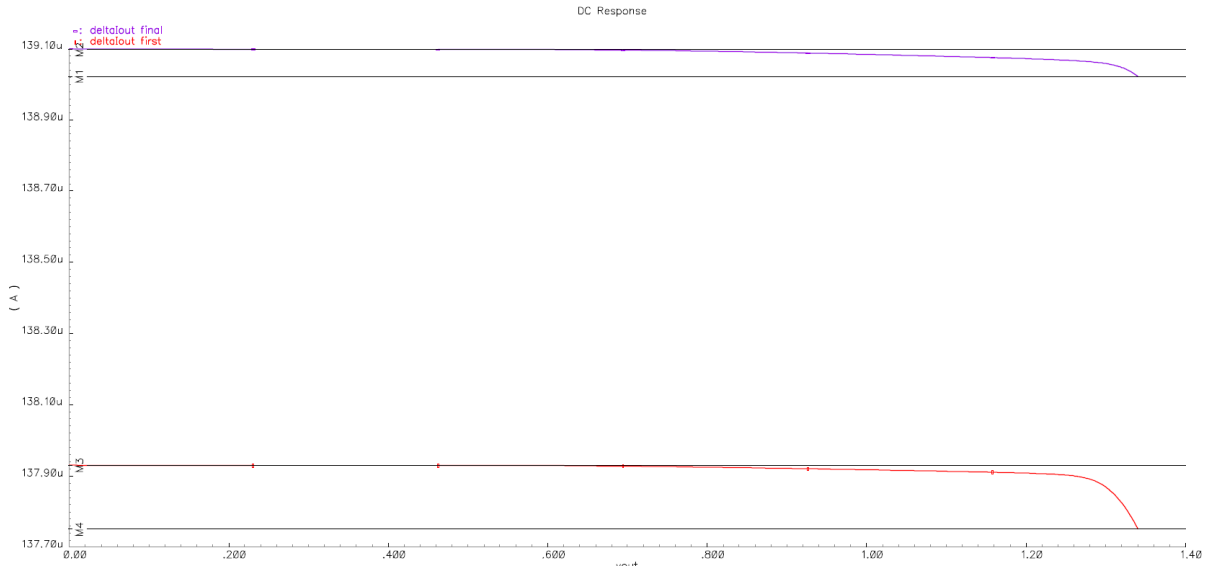

Figure 6.2: Simulation results of the first and final approach for the current source unit.

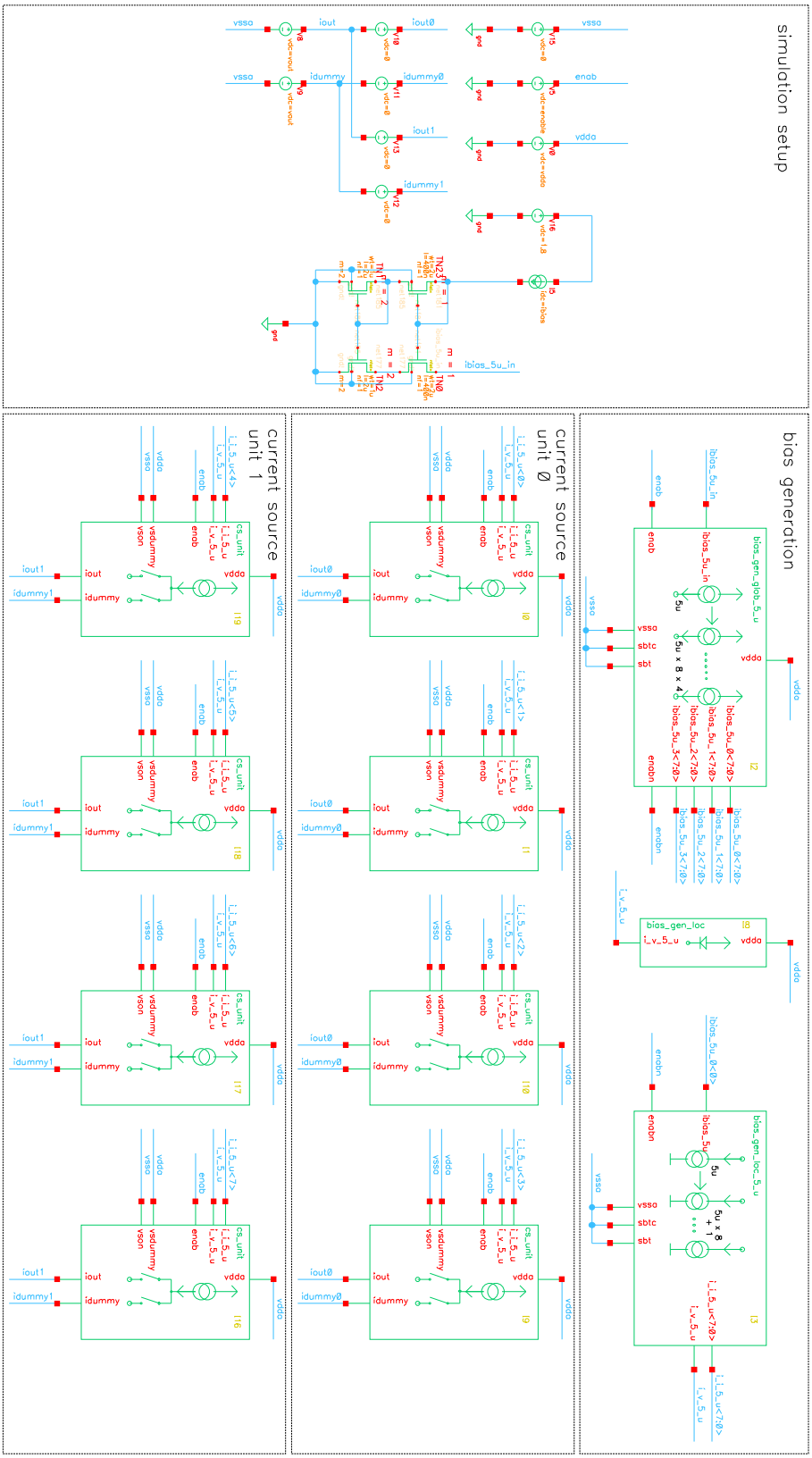

Figure 6.3: Simulation environment for the analog part with all current mirrors for two full current source units.

The schematic in figure 6.3 is separated in four parts. The block on the left side is labeled with ***simulation setup***. This block consists of the ideal voltage and current sources. These ideal sources control the supply voltage *vdda*, the input current *ibias*, the *enable* signal to disable the DAC, and the output voltage *vout*. The variable *vout* controls the two voltage sources for the differential current output paths *iout* and *idummy*. To measure the two full current units with four quarter current units separately, the unit current outputs *iout0* and *iout1* are connected with ideal voltage sources with a value of $0\,\mathrm{V}$ to the common net *iout*. The same is done for the differential outputs *idummy0*, *idummy1* and *idummy*. Additionally, an NMOS current mirror, same as used for the local bias, mirrors the ideal input current to the *ibias_5u_in* input of the global bias. This mirror is needed, because driving the $5\,\mu\mathrm{A}$ input current directly from an ideal current source into the circuit would falsify the result.
Right of the setup block, on the top, is the ***bias generation*** block located. This block contains the global bias, one local bias and one bias diode cell. The schematics behind the symbols in figure 6.3 are basically the same as they are shown in section 5.4. The detailed schematics can be found in appendix A.1. As mentioned in section 5.4, the $5\,\mu\mathrm{A}$ current is generated externally, in this case from an ideal current source and is then mirrored by the external current sink mirror. The global bias module mirrors the current for the local bias module and this module again mirrors the $5\,\mu\mathrm{A}$ current for the bias diode and the quarter current source units.
The last two blocks below the bias generation are the ***current source unit 0*** and ***current source unit 1***. Both blocks contains of four quarter current units each, whose current outputs are connected together.
For the bias generation blocks only a DC analysis is required, because they have to generate an exact copy of the $5\,\mu\mathrm{A}$ input current. The results of the DC simulations are presented in table 6.3. Both approaches are simulated at the minimum supply voltage of $1.71\,\mathrm{V}$ and the typical supply voltage of $1.8\,\mathrm{V}$. The other columns of table 6.3 contains the current of the input and output of the global bias, the output of the local bias and the voltage at the bias diode. The transistor dimensions of the approaches are shown in table 6.4.

approach	supply voltage	global bias		local bias	bias diode
	vdda [V]	ibias_5u in [µA]	ibias_5u out [µA]	ibias_5u out [µA]	vbias [mV]
first	1.71	4.930	4.926	4.927	846.7
	1.8	4.984	4.984	4.985	963.6
final	1.71	4.967	4.966	4.966	962.0
	1.8	4.992	4.992	4.993	1051.0

Table 6.3: Simulation results of global, local and voltage bias modules.

approach	global bias				local bias				bias diode	
	W_{CS} [µm]	L_{CS} [µm]	W_{Casc} [µm]	L_{Casc} [µm]	W_{CS} [µm]	L_{CS} [µm]	W_{Casc} [µm]	L_{Casc} [µm]	W_{Diode} [µm]	L_{Diode} [µm]
first	7.36	2	2.1	0.4	1	1	1	0.4	2.2	2
final	12	2	12	0.4	2	2	2	0.4	3.2	2

Table 6.4: Calculated and simulated values for the bias current generation.

6.1.3 Overall Simulations of the Analog Part

After determining the supposed final values through normal direct current (DC) simulations with and without sweep over some parameters, additional analog simulations have to be done. One of these is

the Monte-Carlo simulation. The Monte-Carlo simulation uses the Monte-Carlo algorithm to distribute specified parameters randomly in between a defined region. This simulation is done by using 500 different runs and distribute the process and mismatch values. This means, in every of the 500 Monte-Carlo runs the process and mismatch values of the transistor models are changed and a DC simulation is done. After these runs the distribution and deviation of a specific simulated value is calculated. In this case the output currents of two current source units, which are all the same, are of special interest. For the Monte-Carlo simulation again the schematic of the analog part of figure is used 6.3. One current source unit is the combination of four quarter current source units. Typically, there should be no difference between the currents, but variations of the process and mismatch parameters can result in a $\Delta Iout_{mc} \neq 0$. Diagram 6.4 presents the histogram of the output current difference $\Delta Iout_{mc}$ distribution. The resulting standard deviation σ for $\Delta Iout_{mc}$ is about $4\,\mu A$. In accordance to an expert, this is an acceptable value, because for a 3σ-design a value of about $12\,\mu A$ can be calculated. This is about a third of the LSB current of $35\,\mu A$. An improvement of this value can only be achieved by enlarging the current source unit. However, this will result in more chip area and is not acceptable for economic reasons.

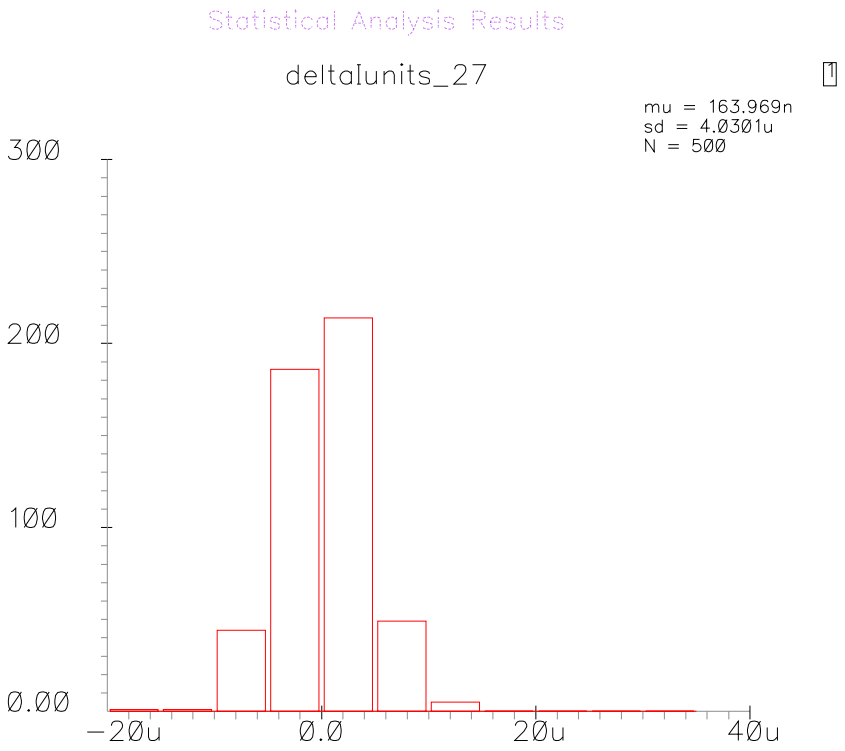

Figure 6.4: Histogram of the Monte-Carlo simulation.

Finally the circuit should be simulated at all specified corners. This means corners which are given by the specification, like supply voltage and temperature range, and corners, which are given from the semiconductor manufacturer concerning the process variations. Again the simulation schematic of figure 6.3 is used. The simulation result is displayed in figure 6.5. The curves illustrate the output current I_{out} depending on the output voltage V_{out}, which is swept from 0 to 1.34 V.

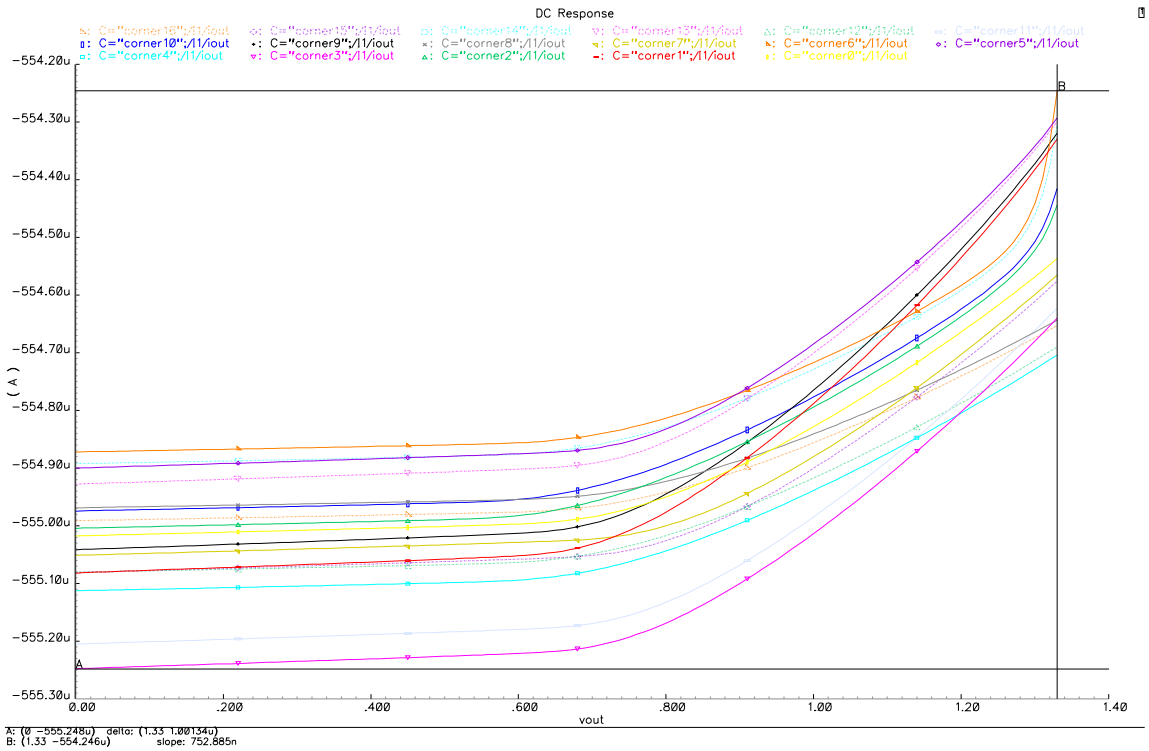

Figure 6.5: Output current I_{out} depending on the output voltage V_{out} simulated over all corners.

The following table 6.5 presents a summary of all simulated corners. Corner0 is the standard corner at the typical temperature and voltage of 27 °C and 1.8 V. The transistor properties are also at **typical mean**. For the other corners, four different CMOS variations are used. The first is the **worst power** variation. For this variation the transistors are fast , but the power consumption is higher than **typical mean**. Another CMOS variation is the **worst speed**. The CMOS transistors are slower at this corner. The last two corners are **worst one** and **worst zero**. **Worst one** means worse characteristics for PMOS transistors and **worst zero** for NMOS transistors. All CMOS variations are simulated at the lower and higher temperature corner of $-25\,°\mathrm{C}$ and $85\,°\mathrm{C}$ and at the lower and higher supply voltage corners of $1.71\,\mathrm{V}$ and $1.89\,\mathrm{V}$. These are four corner simulations for every CMOS variation. The worst corners are at minimum supply voltage and maximum temperature. The difference between the output current of all corners is less than $0.5\,\mu\mathrm{A}$.

6.2 Digital Simulations

The digital simulations mainly check the test-bench stimuli for the mixed-signal simulations. Additionally, the test-chip includes a digital function generator, which is placed in the video encoder part of the test-chip. By selecting the function generator at the test multiplexer (see section 8.4.3) the digital input source will change. For this case the 10 bit digital input will be provided by the digital function generator and not from the video encoder anymore. With another configuration of the test multiplexer the 10 bit input DAC inputs can be directly stimulated from digital input pads. The following section describes the digital function generator and the simulations, which are possible with the generator.

corner name	temperature [°C]	supply voltage [V]	CMOS variation
corner0	27	1.80	typical mean
corner1	-25	1.71	worst power
corner2	85	1.71	worst power
corner3	-25	1.89	worst power
corner4	85	1.89	worst power
corner5	-25	1.71	worst speed
corner6	85	1.71	worst speed
corner7	-25	1.89	worst speed
corner8	85	1.89	worst speed
corner9	-25	1.71	worst one
corner10	85	1.71	worst one
corner11	-25	1.89	worst one
corner12	85	1.89	worst one
corner13	-25	1.71	worst zero
corner14	85	1.71	worst zero
corner15	-25	1.89	worst zero
corner16	85	1.89	worst zero

Table 6.5: Summary of simulated corners.

6.2.1 Digital Function Generator

The digital function generator enables to test the DAC without generating 10 bit test data externally. Through the I2C interface and the function generator registers several adjustments are possible. The main adjustment is the shape of the output signal, which is generated from the DAC by using the digital input data of the function generator. Four different shapes are possible at the analog output. The shapes and the corresponding register settings bit combination for the *dac_fg_waveform* signal are presented in table 6.6. The *dac_fg_waveform* signal is placed in the *DAC_ FG_AD_SHAPE_OFFS* register at bit position 0 and 1. The *DAC_ FG_AD_SHAPE_OFFS* register has the register address $0x90$. Another signal in this register is the *dac_fg_offset* signal, which has a width of 6 bit. By changing this signal a DC-offset can be added to the output signal. The value is directly added to the 6 bit LSB of the 10 bit DAC input signal.

bit	shape
00	ramp
01	triangle
10	sine
11	rectangle

Table 6.6: Bit combinations for selecting the different function generator shapes.

Another register for configuring the digital function generator is named *DAC_FG_AD_FREQU_AMPL* and can be addressed with the register address $0x91$. The distribution of the 8 register bit to the 3 signals *dac_fg_segment*, *dac_fg_magnitude* and *dac_fg_resolution* is presented in table 6.7. With the *dac_fg_segment* signal the segment of the output signal can be chosen. The segment of a signal is a quarter of the whole wave. For example, for sinusoid signals with the segment signal the whole sinus signal, or only a positive, negative half or quarter wave can be chosen. The *dac_fg_magnitude* signal

bit	signal name
7:6	*dac_fg_segment*
5:3	*dac_fg_magnitude*
2:0	*dac_fg_resolution*

Table 6.7: Bit distribution of the *DAC_FG_AD_FREQ_AMPL* register.

reduces the magnitude of the output signal. The magnitude of the output signal can be calculated with the following equation

$$output\ magnitude = \frac{output\ pattern}{2^{dac\_fg\_magnitude}} \tag{6.1}$$

Finally, the *dac_fg_resolution* signal reduces the resolution of the output pattern by changing the step size of the digital pattern. The step size for a continuous signal like the ramp or triangle function can be calculated with the this equation

$$step\ size = 2^{dac\_fg\_resolution} \tag{6.2}$$

A side effect of decreasing the resolution is the increasing of the function generator output frequency. The digital function generator clock frequency is also $27\,\mathrm{MHz}$. By using the maximum resolution on a ramp signal, an output frequency of maximum $26.4\,\mathrm{kHz}$ can be reached.
The following section gives a short overview of the different signal shapes by using several configurations. A Diagram with the bit configuration for the different shapes is presented in figure 6.6.

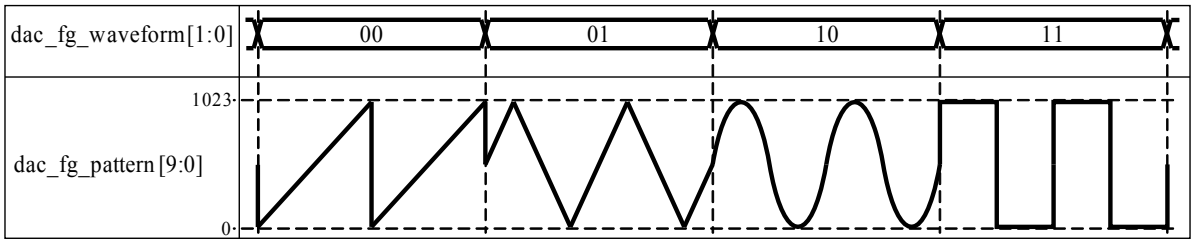

Figure 6.6: Different digital output shapes of the function generator.

Ramp Shape

The ramp is the default shape of the digital-function generator. It can be used to step through all $2^{10} - 1$ different output codes of the DAC by ramping continuously up. With the measuring of the analog DAC output every ramp step, the INL and DNL can be calculated. Therefore a much slower clock frequency than the maximum DAC frequency of $27\,\mathrm{MHz}$ should be used. For instance a $10kHz$ input frequency would the DAC give the chance to settle down after every step.

Triangle Shape

The triangle shape is very similar to the ramp. Different is the continuous down count instead of the discontinuous falling back to the start value at the ramp shape. Here are also different magnitudes and resolutions possible. The *dac_fg_segment* signal takes effects on the output signal by changing the triangle shape form.

Sine Shape

The sine shape implementation is simplified and not very accurate. The intention is to have the possibility to generate sine waves for different applications. The implementation decision for the sine shape is a lookup table with 64 entries. These are the supporting points for one quarter of the sine wave. For the sine wave are also different resolution and magnitude configurations possible.

Rectangle Shape

The easiest shape of the function generator is the rectangle wave. The output has only a minimum and maximum value and a specified frequency. The frequency can again be adjusted by the resolution signal *dac_fg_resolution*. Responsible for the minimum value is the offset signal *dac_fg_offset* and the maximum value is selected by the magnitude signal *dac_fg_magnitude*.

6.3 Mixed Signal Simulations and Results

The mixed signal simulation is used to verify the functionality of the whole video DAC. For most simulations the digital input pattern are generated from the digital function generator in section 6.2.1. The default configuration with the ramp function and no changes in the magnitude and resolution register is also the default case for the mixed signal simulations. With the ramp function the nonlinearities like DNL and INL can be measured and calculated. These nonlinearities do not include the mismatch between the current source units caused on layout issues. They quality of the layout for this video DAC can not be evaluated before it will be measured in the laboratory.
Other simulations, like evaluating the settling, makes no sense, because without the inductance and capacitance added by layout, bonding and packaging, the results have no relevance. The most characterization steps can not be done before a silicon of the DAC is available.

6.3.1 DNL and INL Simulations and Results

The evaluation of the DNL and INL in simulation is done with two different methods. Once a verilog-ams model is used directly in simulation. The advantage is the calculation during simulation and if the simulation is finished the result is available without any further steps. The other possibility is to evaluate the simulation output. A table is generated by capturing the analog DAC output every step of the $2^{10} - 1$ steps. The data is captured just before the next step to be sure the output is stabilized. The advantage of method two is the possibility of several post-calculations. The measured value is not the output current I_{out}, but the equivalent output voltage V_{out} at the $37.5\,\Omega$ resistor. To analyze the DNL and the INL, the offset and gain errors of these output measurements have to be removed first. Afterwards, the DNL is calculated by using the following equation

$$DNL_n = \frac{Vout_{n+1} - Vout_n - V_{LSB}}{V_{LSB}}\,[LSB] \tag{6.3}$$

where $Vout_n$ and $Vout_{n+1}$ are consecutive output voltage values, and V_{LSB} is the offset and gain error corrected LSB output voltage. The resulting DNL error is given in parts of LSB. Applying equation 6.3 to every value in the table generates the DNL for every step of the DAC. A graph illustrates the DNL over the whole area of the video DAC. This graph can be found in combination with the INL graph in figure 6.7. The graph presents a DNL, which is around the zero line. This is a good value, but it is simulated without parasitics and layout matching information.
The calculation of the INL is very similar to the calculation of the DNL. The reference data for the INL calculations are again offset and gain error corrected. Afterwards, the following equation is applied to every row of the table

$$INL_n = \frac{Vout_n - D_n \cdot V_{LSB}}{V_{LSB}}\,[LSB] \tag{6.4}$$

where $Vout_n$ is the actual output voltage, D_n is the actual data word and V_{LSB} is again the offset and gain error corrected LSB output voltage. Similar to the DNL result is the INL also given in parts of LSB. The resulting digram is presented in figure 6.7. The graph for INL shows also a very good value, but again with the missing of layout information. It illustrates also, that the overall output current/voltage has its maximum deviation at the beginning of the second part of the data-word range. This means the ideal operating conditions for the current source units are between the data-word 512 and 768 and an output voltage of about $800\,\mathrm{mV}$.

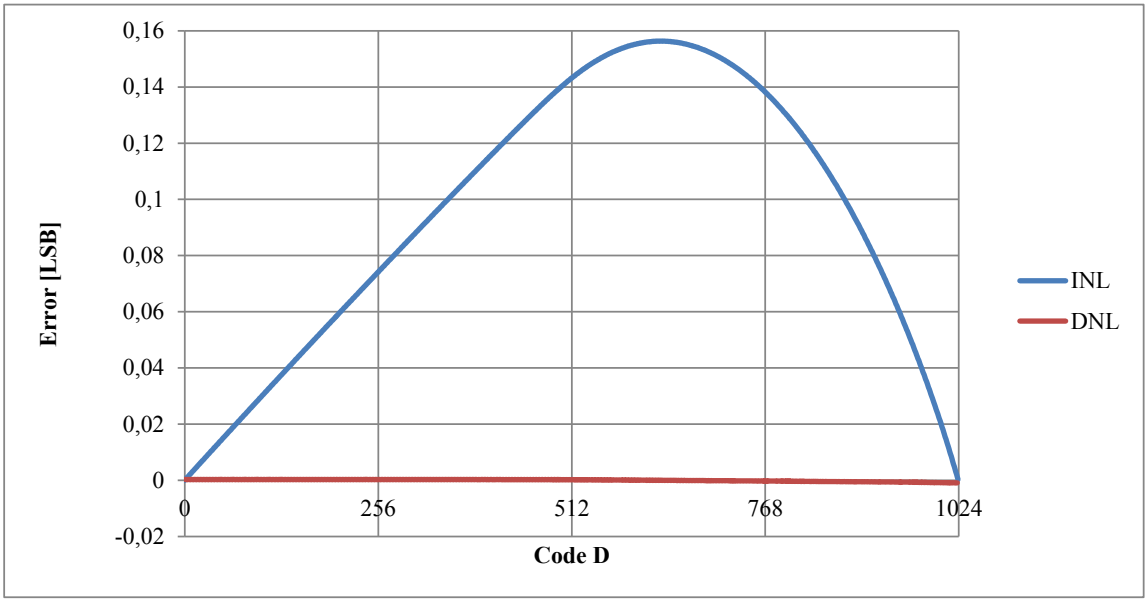

Figure 6.7: Simulated INL and DNL of the video DAC.

Chapter 7

Layout Implementation

This chapter describes the layout implementation of the video DAC and starts with the presentation of the floorplan. After that, the layout implementations of the single building blocks are shown. Finally, section 7.5 gives an overview of the complete video DAC layout.

7.1 Floorplan of the Video DAC

The floorplan of the current steering video DAC is shown in figure 7.1. The layout of the video DAC has a rectangular shape. On the left side of the block is the interface to the digital part with the input sample register. The thermometer coded current unit part of the DAC is the main block in the layout. To improve the matching between the current units, the thermometer coded current unit part is placed four times, at each quadrant. Between this four parts is some space with the shape of a cross. In the horizontal axis of the cross is the global bias cell and in the vertical axis are the local bias cells located. The bias current input comes from the right side and supplies the global current bias cell with the $5\,\mu\mathrm{A}$ constant input current. Additionally to the main floorplan, a supply and output concept is created. This can be found in the next section.

7.1.1 Supply and Output Concept for the DAC

A major parameter for the floorplan is the supply concept, including the signal flow from the supply to the current outputs. A current of almost $40\,\mathrm{mA}$ is flowing from the supply pads through the current units to the output pads. The supply rails have to be wide enough to minimize the resistance and therefore the voltage drop. Another criterion for the supply wire width are the current density rules from the semiconductor manufacturer. For every metal layer the semiconductor manufacturer defines a maximum current per $\mu\mathrm{m}$ wire width to prevent metal migrations. Figure 7.2 illustrates the concept. The analog supply *VDDA* is built like a ring around all four current unit quarters of the DAC. In figure 7.2 it is drawn with a red color. The current output *iout* and the inverted current output *idummy* are colored light and dark blue. They are connected from top and bottom. Over the current source units they are connected alternatedly by using the top metal layer, which has double the thickness of the other layers. Therefore, the top metal layer has a much higher maximum current density and lower sheet resistance than the other metal layers. The digital supplies *VDDD* and *VSSD* are placed vertically between the digital logic on the left side and the current unit block. They are drawn with light and dark green. Additionally, the digital logic between the current source unit is also supplied by *VDDD* and *VSSD*. The placement of the supply and output pads for the test-chip can be found in section 8.3, where the pinout is defined. The analog supply *VDDA* has four pads to guarantee a good supply.

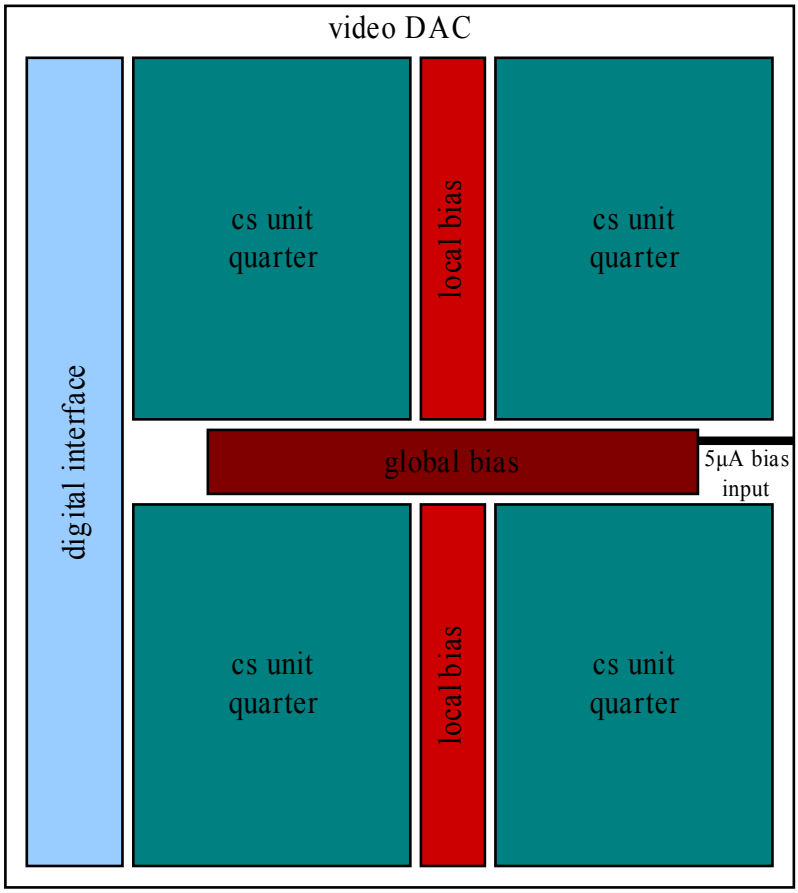

Figure 7.1: Layout floorplan of the video DAC.

7.1.2 Floorplan of the Current Unit Quadrant

One current unit delivers only a quarter of the thermometer coded unit current and therefore the 63 thermometer coded current units have to be placed four times. The floorplan of these four parts is always the same except the placement of the binary weighted current units. All binary weighted current units are placed around the center of the DAC. For the top right quarter the unit is placed on the left bottom, for the top left quarter the unit is placed on the right bottom, and so on. The thermometer coded current units are organized in rows and columns. Every row and column consists of 8 units, which is illustrated in figure 7.3. To get a shared digital channel between the rows, every second row is flipped upside down. The digital input control signals are at the left side directly connected to the digital channels. The bias currents have also a channel through the row between the current source and the cascode transistor. Details of the subcells can be found in the following sections.

7.2 Layout of the Current Units

The main component of the current steering DAC is the current source unit. It is placed many times and therefore a structured and optimized layout of one single current unit is very important. Due to the required currents it is possible to build the binary weighted current units with the same size and format as the thermometer coded current units. The binary weighted current units can be seen as the 64^{th} current unit in one array of 8 times 8 units.

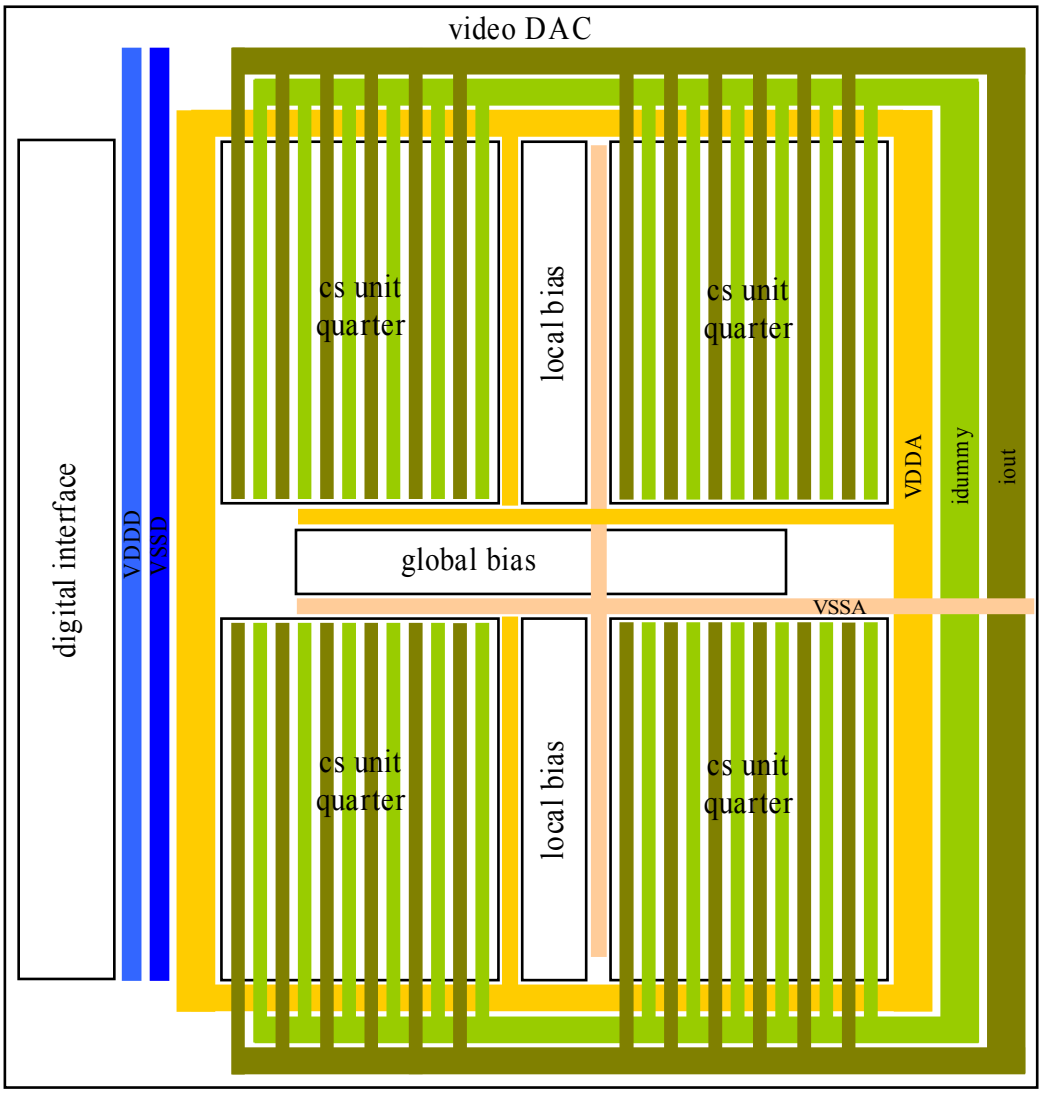

Figure 7.2: Supply concept of the video DAC.

7.2.1 Layout of the Thermometer Coded Current Units

The outline of one thermometer coded current unit has a rectangular shape in an upright format. On the top of the unit the current source transistor can be found. It has 29 segments of the same size, where one of them is the source for the 5 µA current and the other 28 are the mirrored output transistors. This one is placed nearly in the middle of the 29 segments. Right to the current source transistor is the transistor diode respectively the transistor capacitor for the cascode voltage. Beneath the current source transistor is the cascode transistor. The current source and cascode transistors are separated by the current bias wires. The distribution of the cascode transistor is the same as the distribution of the current source transistor. The cascode transistor is also split up in 28 plus one segments. Hence, they are not placed rectangular, like the current source transistor. They are placed in the shape of a "T" to have space for the two switching transistors at the left and right bottom corner. Below the switching transistor is a channel with one row of digital logic and the digital wires. Every channel between the rows is shared by the top and the flipped bottom current unit cell. Figure 7.4 shows the screenshot of four current source units to illustrate the shared digital channel. On the top and bottom of the image red wide wires can be found. This are the *VDDA* supplies.

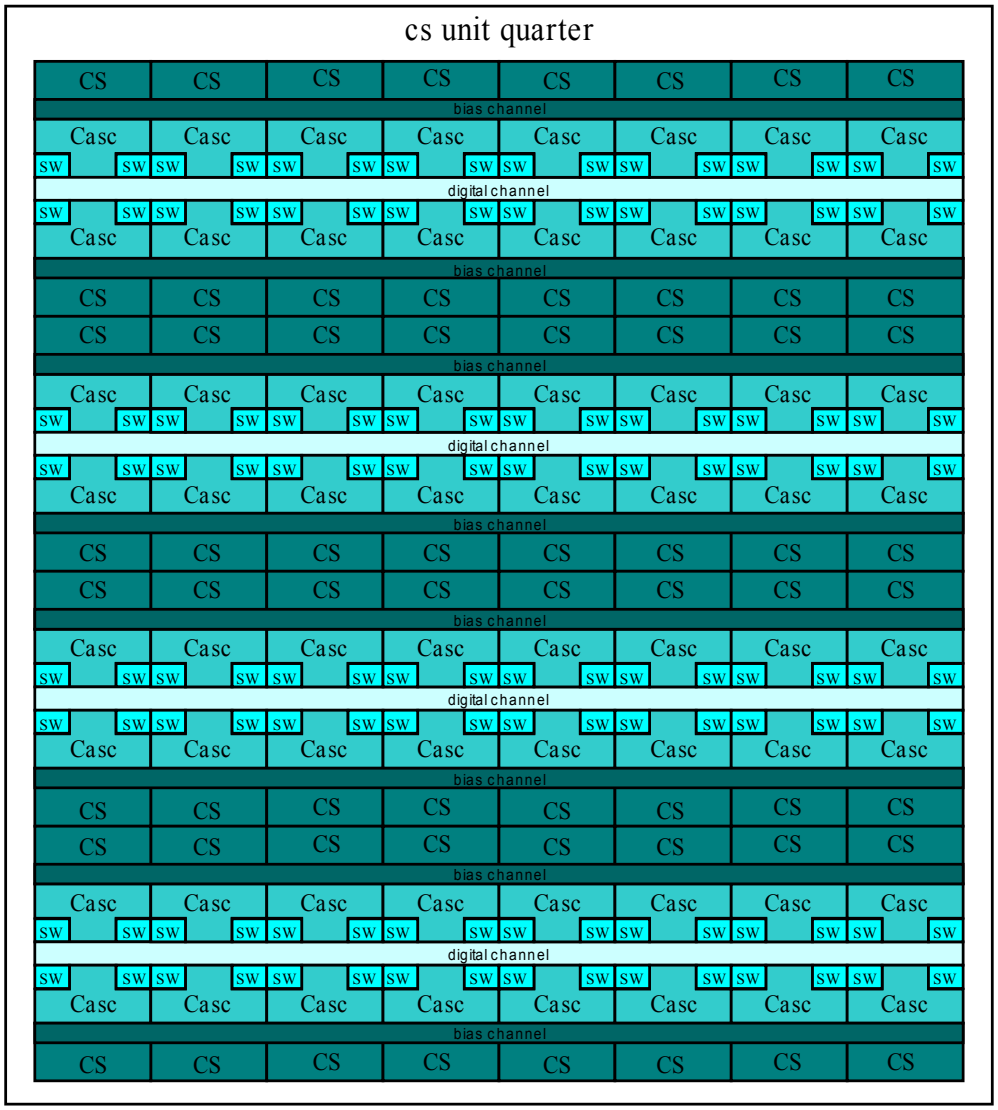

Figure 7.3: Floorplan of one current unit quadrant.

7.2.2 Layout of the Binary Weighted Current Units

The layout of the binary weighted current source unit is nearly the same like the layout of the thermometer coded current source unit. The reason is described in section 5.3.2. Hence, three of the four binary weighted units are exactly the same as the thermometer coded current source units, except the digital control logic. However, the fourth is different to the others, because this unit has two different current outputs. The two outputs delivers a current of one and two LSB currents. Therefore four output switches are necessary. Due to the smaller current source transistors and the smaller cascode transistors the overall shape and size of the unit layout is the same. A screenshot of the above described combined binary weighted current source unit can be found in figure 7.5.

7.3 Layout of the Bias Sources

As mentioned above, two different current bias sources are implemented. The global bias source is the larger one, because it supports the 32 local bias sources with 32 bias currents. The local current bias

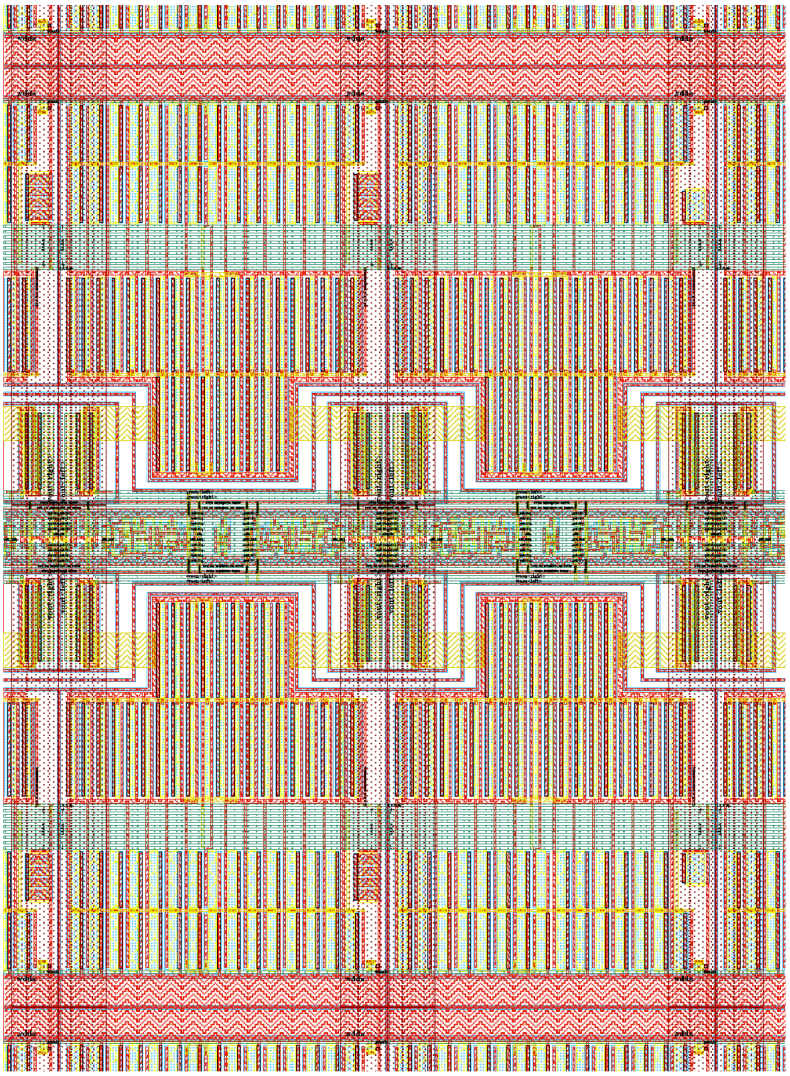

Figure 7.4: Screenshot of the layout of four current source units.

module is much smaller and supports one current source unit row of one quarter current unit array with the required bias currents. Hence, this module is placed 32 times. The global and local current bias sources are placed in the middle of the DAC in a shape of a cross.

7.3.1 Global Bias Source

The global current source unit is built of PMOS current mirrors with cascode transistors. Additionally, a power-down transistor is implemented. The power-down logic is placed on the left side of the module, because the enable signal is connected to the digital part at the left edge of the DAC. At the right side of the block the $5\ \mu A$ current input is connected. The layout of the block has a long rectangular shape, to fit into the small horizontal channel in the middle of the DAC. The module itself is separated into a left and a right side. Every side supports 16 local bias modules at the top and bottom with a $5\ \mu A$ input current. Figure 7.6 presents a screenshot of the global bias layout implementation.

7.3.2 Local Bias Source

The local current bias unit is much smaller than the global current bias unit and it is only built of NMOS transistors. It has also a power-down transistor to disable the block. To supply one row of a quadrant

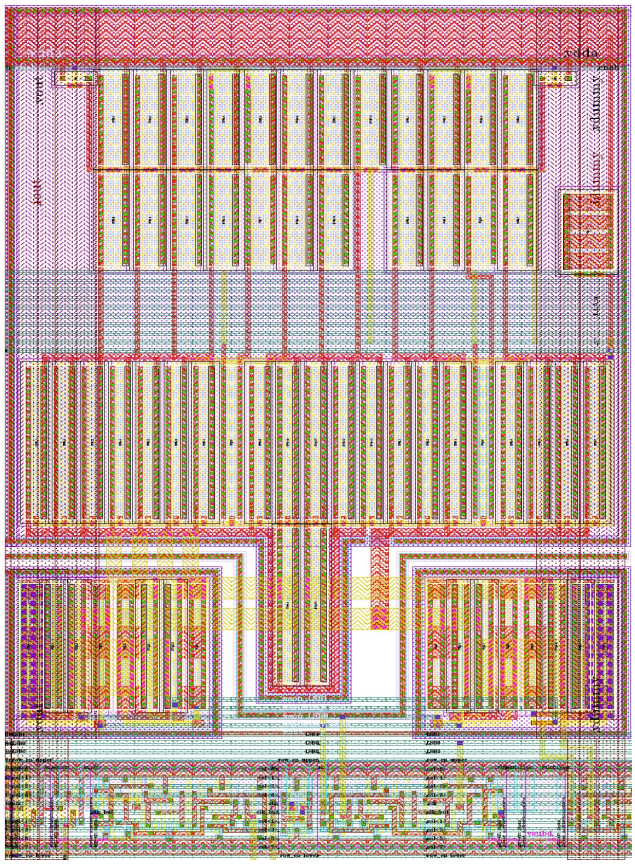

Figure 7.5: Screenshot of the layout of the combined binary current source unit.

Figure 7.6: Screenshot of the layout of the global current bias unit.

with the required 5 µA current, the module has 9 current outputs, 8 for the current source units and one for the cascode gate voltage bias diode. Figure 7.7 shows a screenshot of two local bias sources. In the top layout module (see figure 7.9) the local bias modules are rotated 90°. To support the left and right quarter with its currents the two modules of figure 7.7 are flipped against the other. Looking at the right bias module in figure 7.7 on the top a row of cascode transistors can be found. Below the cascode transistors is an area with interconnect. On the bottom of the bias module are the current source units placed. The input current source transistor and all output current source transistors are split up in two pieces to enhance the matching between the current sources.

7.4 Layout of the Digital Part

The digital part can be separated into two different parts. One is the digital interface with the sample and hold input registers and the thermometer encoder. The second part is the local digital logic. This logic is placed in every digital channel between two rows of the current units. Both parts are supplied by the digital supply *VDDD* and *VSSD*. The connections between the two parts are at every row. The switching scheme is not straight forward by beginning at the left top unit. The scheme follows a special pattern to improve the matching of the current units. The current unit, which is first turned on by the thermometer

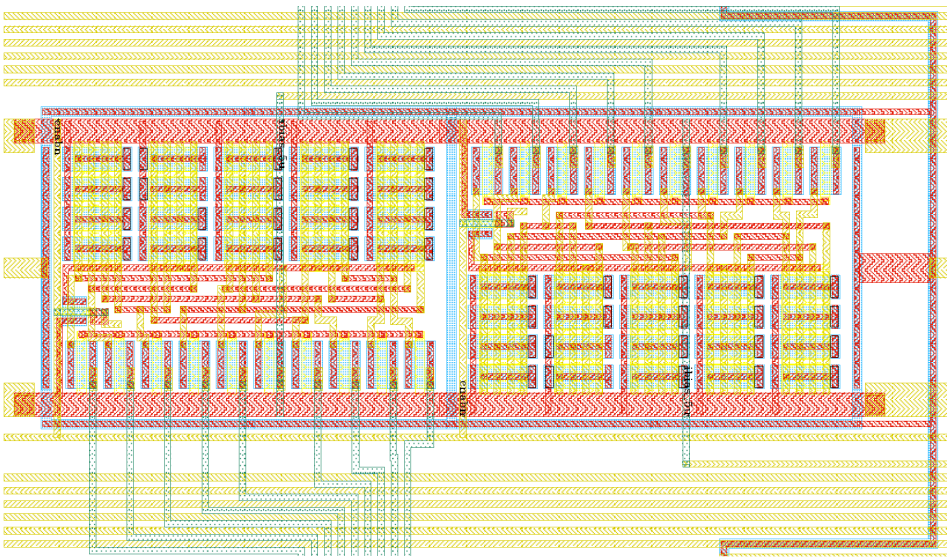

Figure 7.7: Screenshot of the layout of the local current bias unit.

code is numbered with “1” and the last one is numbered with “63”. The first unit turned on is in the center of the quarter current unit array. The next units are turned on alternating the column and row. The following figure 7.8 shows the switching scheme of the array.

7.4.1 Layout of the Digital Interface

The digital interface is placed vertically at the left edge of the DAC. However, it is built horizontally and rotated 90° counter clockwise. The interface is written in the hardware description language *verilog* and automatically routed by a routing tool. The routing tool is provided by the electronic design automation (EDA) provider Cadence®. It is called First Encounter®. To get the input and output wires at the right place, the First Encounter® tool needs an abstract view with the boundary box and the input and output pins placed at a specified position. This box is created in the analog layout framework of Cadence® with the name Virtuoso Layout Editor. For the automatically placement the verilog-file, the abstract and optionally a Tool Command Language (TCL) file is used. The first time routing a new layout, is mainly done by using the Graphical User Interface (GUI), but if the manual flow is working well a TCL file can be created and the routing tool will go trough all needed steps without any user interaction. After all routing steps a Graphic Data System II (GDSII) file has to be exported, because the digital standard cells are only placed as abstract. By reading in the GDSII file into the Virtuoso Layout Editor and referencing to the right standard cell library, the final digital layout is available. This can be found in figure 7.9 on the left side as part of the top implementation.

7.4.2 Layout of the Local Decoder

The local decoder is also built with digital standard cells, but they are not automatically routed. This part is built manually and this is called a full custom design. Every single current unit decoder is built only with a few logic cells. They have to be placed directly at the current unit and therefore an automatical layout makes no sense and will not give the same result. The wires between the digital interface and the local decoder are done at the routing channels above the digital cells. For programming the digital input of the block to the correct row signal, the via has to be placed at the right point to connect the two metal layers. Figure 7.4 shows a screenshot of four current source units, where the digital routing channel can be found horizontal in the middle of the image.

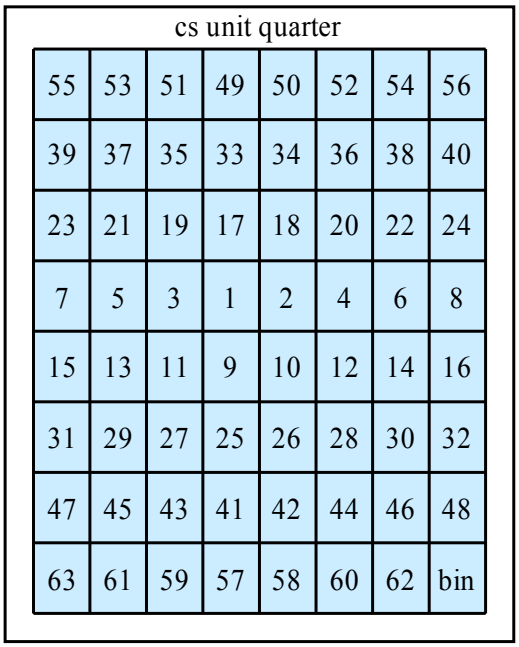

Figure 7.8: Diagram of the thermometer coded switching scheme.

7.5 Layout of the Top Implementation

Using the above described floorplan and merge all modules, results in the layout top implementation. Figure 7.9 presents a screenshot of the whole video DAC layout. By looking at the layout only a few free spots can be located. Therefore, the layout is area efficient and very regular relating to the 256 current units, which have almost the same block layout. The final area of the DAC in picture 7.9 is about $0.7\,\mathrm{mm}^2$ on the used $0.18\,\mu\mathrm{m}$ process.

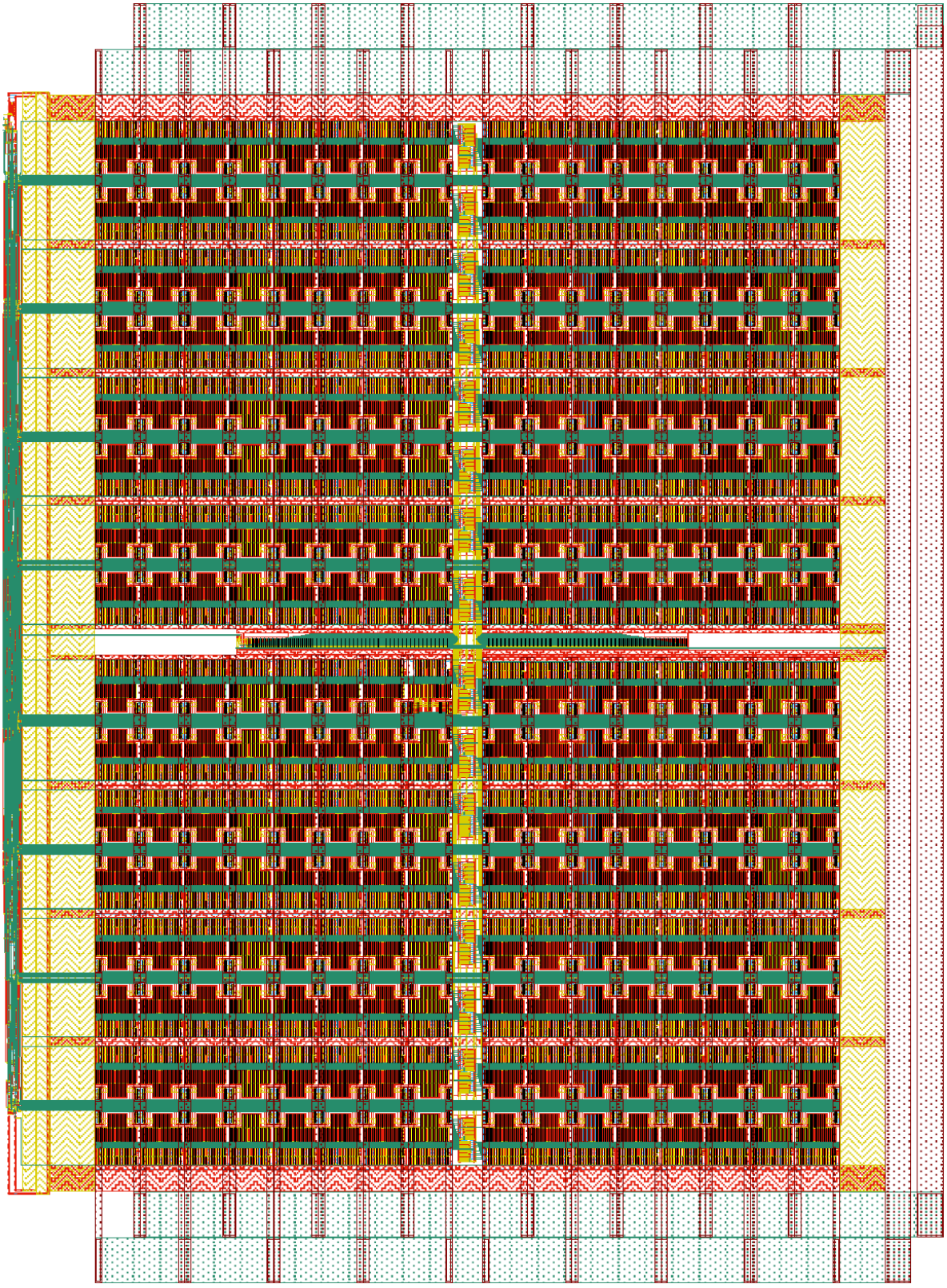

Figure 7.9: Screenshot of the layout top implementation.

Chapter 8

Overview and Setup of the Video Encoder Test-chip

This section gives an overview of the test-chip including the video DAC. The test-chip contains not only the known video DAC, a digital video encoder (DVE) and a digital function generator are implemented too. With one of these test-chips it is possible to convert a BT.656 video input signal to an analog video signal on the output. The main idea of this test-chip is to display digital video signals generated from a portable video player on TV screens connected with an analog video interface, for example component or composite video. A built in test multiplexer allows to evaluate the two subsystems stand-alone. The next section delivers some information about the whole test-chip. The rest of this chapter should be a manual for testing the analog part of the test-chip – the video DAC.

8.1 Video Encoder Test-chip

The video encoder test-chip mainly contains two different parts, the known video DAC and the DVE. A block diagram of the DVE main building blocks can be found in figure 8.1. From the 8 bit digital input signal, according to BT.656 standard, information about synchronization, luminance and chrominance are calculated.
The 10 bit digital output of the DVE is compliant to the ITU video standard for PAL and NTSC signals. In normal mode the digital output of the DVE is forwarded to the second part, the video DAC (figure 8.4.1). The video DAC converts this digital input stream to an analog output current. For improved testability, an additional digital function generator is built into the digital part of the test-chip. With this digital function generator it is possible to support the DAC with a digital input signal to test the video DAC without any external signal source. A more detailed description of the digital function generator can be found in section 6.2.1. To evaluate the video DAC, section 8.2 gives a description about what is needed to get the video encoder test-chip started.

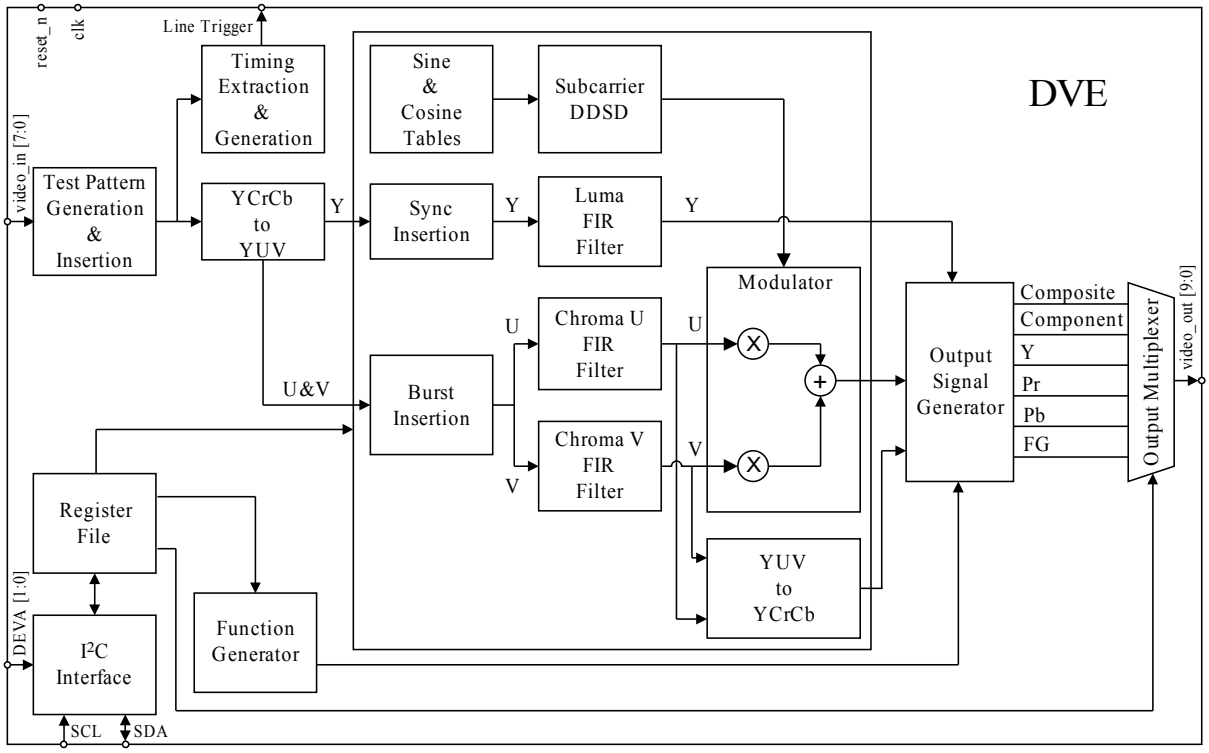

Figure 8.1: Block diagram of the DVE.

8.2 Requirements for the Evaluation Board

The main component of the evaluation board is of course the video encoder test-chip. The test-chip is packaged into a ceramic package with a sealed cover plate. It is a standard package for engineering samples with 44 pins. The name of the package is CQFP-44LP and its dimensions can be found in figure 8.3. For the evaluation board it is recommended to use a socket for the video encoder test-chip. Except the video encoder test-chip, only a few additional components are necessary.
The test-chip needs a $1.8\,\mathrm{V}$ power supply. For this purpose a simple connection to an external regulated power supply, would be quite enough. If the DVE video inputs are supported with an input signal higher than $1.8\,\mathrm{V}$, the VDD3V3ENC supply pin has to be connected to a power supply which matches the high level of the signal. The maximum power supply for this VDD3V3ENC pin is $3.3\,\mathrm{V}$. The next very important components on DAC side are the $37.5\,\Omega$ resistors for terminating the two current outputs. The $37.5\,\Omega$ value comes from the double terminated video wire (see subsection) and can be realized with two $75\,\Omega$ resistors in parallel. The current mirrors of the DAC have to be supplied by an external $5\,\mu\mathrm{A}$ current sink. To be flexible during testing phase the current sink should be adjustable. A good range would be from $4\,\mu\mathrm{A}$ up to $6\,\mu\mathrm{A}$ adjustable in $0.1\,\mu\mathrm{A}$ steps. If the board design implements a I2C interface the two I2C pins - SCL and SDA - have to pulled up with $10\,\mathrm{k}\Omega$ resistors each. To setup the test-chip for test mode some switches are needed. The test mode setup is described in more detail in section 8.4.1. A suggested schematic of an evaluation board for the DAC or a demonstration board for the whole DVE test-chip is shown in figure 8.2.

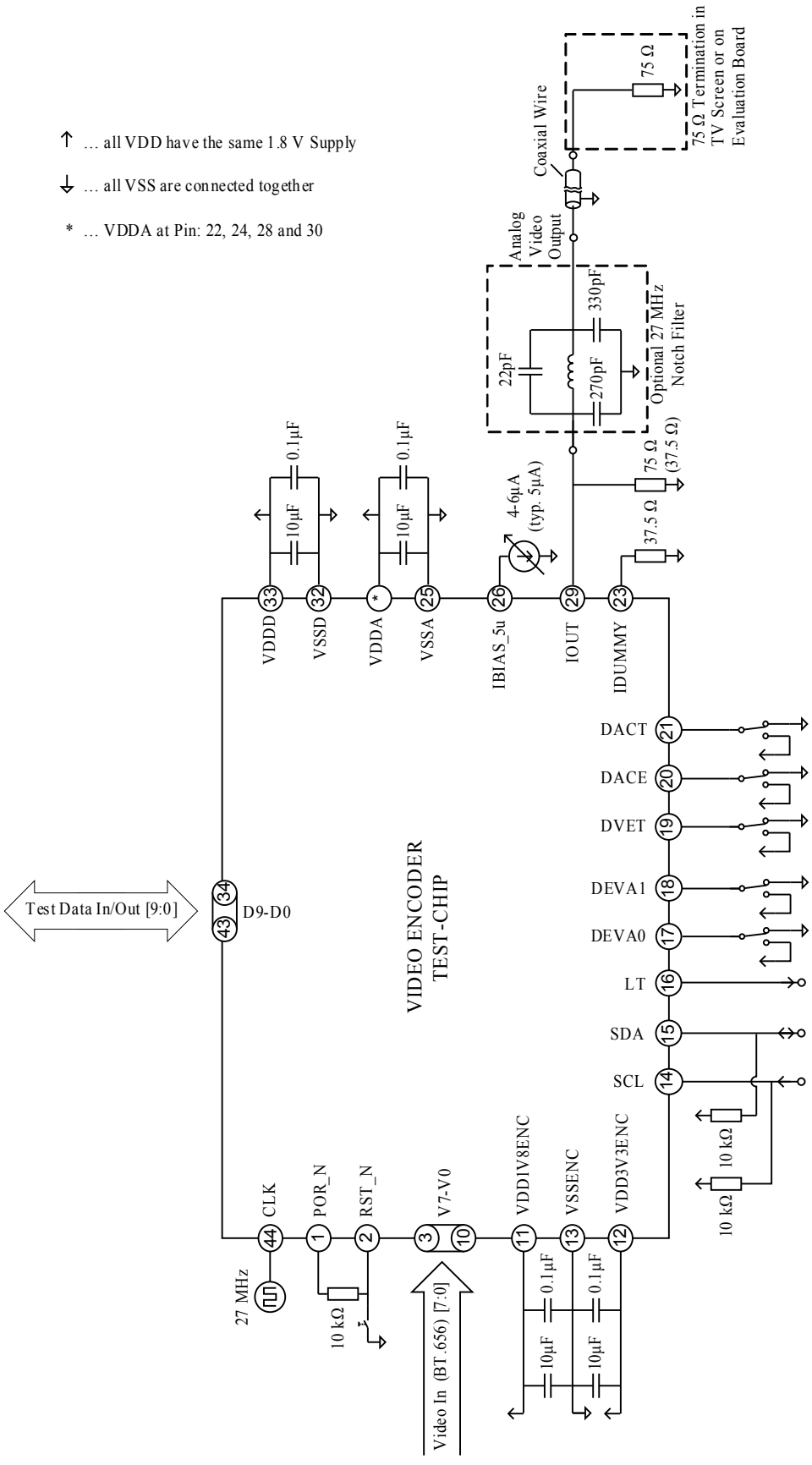

Figure 8.2: Suggested board schematic.

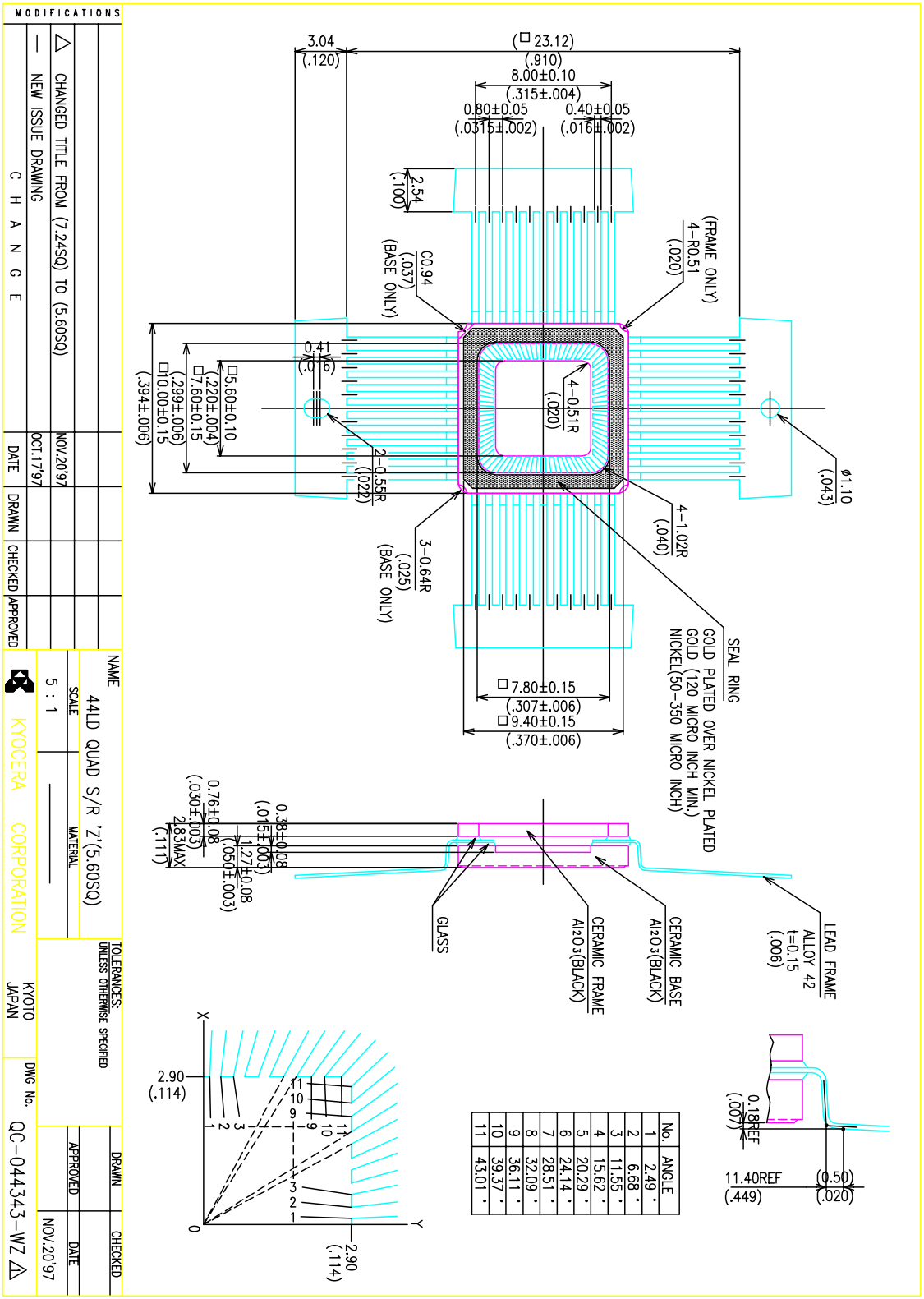

Figure 8.3: Dimensions of the CQFP-44LP package. [Kyocera, 2011]

8.2.1 Output Termination

The Video DAC is designed for double terminated output wires as shown in the suggested schematic (see figure 8.2). For video DAC testing, the two 75Ω resistors in parallel will be required. When testing the whole chip (DVE and video DAC) with a TV screen or monitor, one of the two resistors has to be detached. The reason is the built in 75Ω resistor in such systems. This means the wire is terminated on the DAC output and on the TV or monitor input. Other semiconductor manufacturers terminate on the DAC side with a much higher resistor to keep the current low. The problem of the implementation is an additional need of some kind of video buffer to get a ITU compliant video signal. The ITU [1998] says the output termination has to be 75Ω at both sides of the video line.

8.3 Test-chip Pinout

The pinout of the test-chip is given in table 8.2. Additionally, figure 8.4 shows the bonding diagram were the DVE and DAC parts are marked. On the left side of the chip is the digital part. This block gets the input data from the 8 bit input bus on the left. The clock and reset input are also on this side. On the bottom of the chip are mainly signals which are used for testing. There is the I2C interface with the programmable device address. In the bottom center are the enable signals for the DAC and video encoder. On the top of the chip are the input/output pins to stimulate the DAC with digital signals or alternatively to measure the output of the digital video encoder. All analog pins are on the right side of the chip. These pins are the differential current outputs, a 5 µA current reference input, and some supply pins.

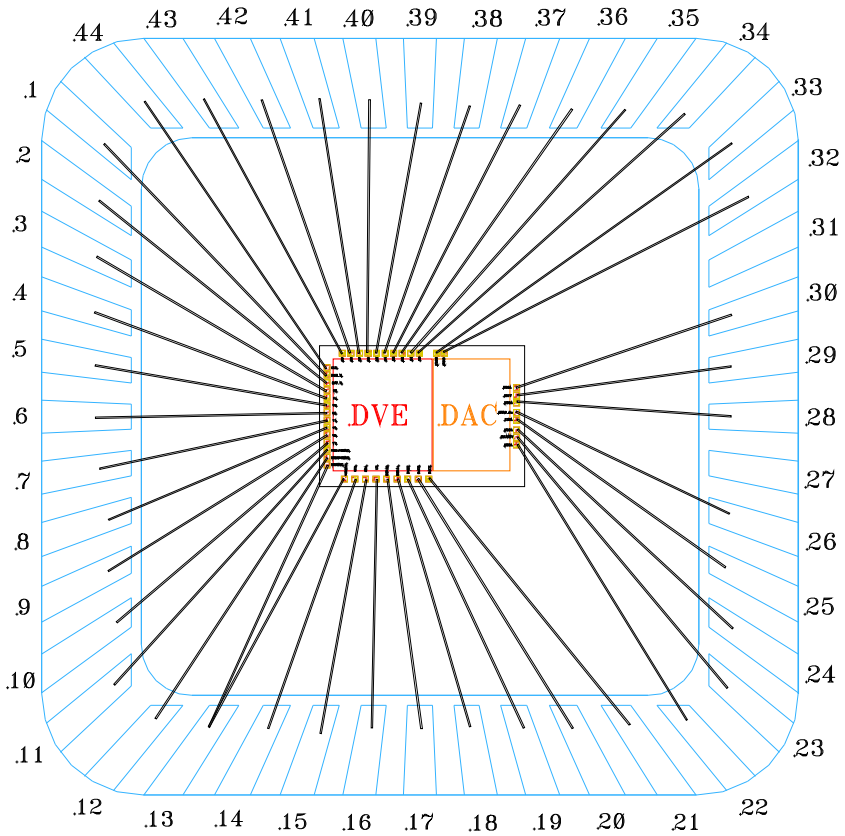

Figure 8.4: Bonding diagram of the DVE and DAC test-chip.

8.4 Setup for DAC Measurement

This section provides the information for setting up a test environment to measure and characterise the video DAC. Testing the whole video encoder test-chip is not part of this work. The following section will only include information for the setup of the video DAC part of the test-chip. Therefore the register map in section 8.4.2 includes no information about the registers of the DVE.

8.4.1 Test Control Unit

For independent testing of the DVE and the video DAC, ten additional pins (D0-D9) and three test control pins (DVET, DACT and DACE) are added. With the three control pins the interconnect between DVE, digital test pins (D0-D9) and the video DAC is programmable. Figure 8.5 reveals the internal structure of the test control unit. It is possible to route the DVE output to the video DAC input and to the digital test pins. It is also possible to connect the digital test pins directly with the DAC input and bypass the video encoder. Therefore, the DAC enable (DACE) and DAC test (DACT) have to be set properly. A detailed definition of the test control unit setup is given in table 8.1.

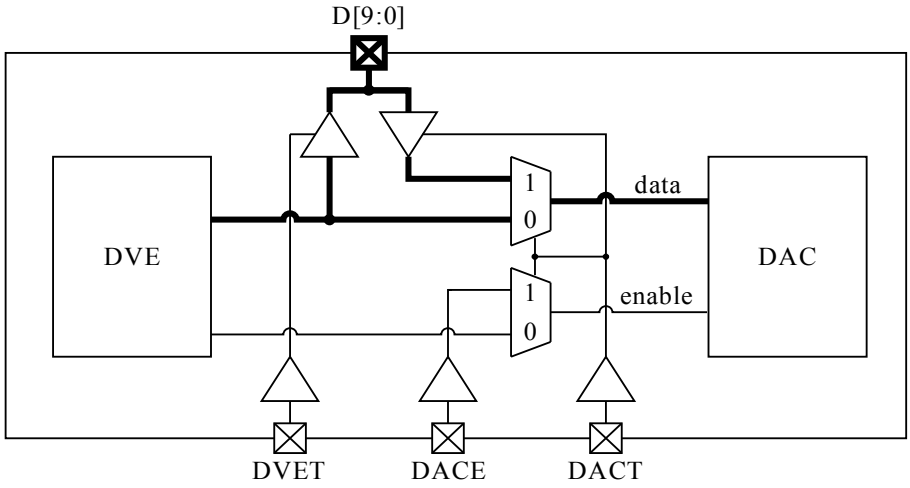

Figure 8.5: Test control unit of the DVE and DAC test-chip.

DVET	DACE	DACT	Mode	Description
0	X	0	normal	all DVE signals are forwarded to the DAC (data and enable)
0	X	1	DAC Eval.	D0 - D9 are forwarded to the DAC, DACE = 1 enables the DAC
1	X	0	DVE Eval.	normal mode plus forwarding DVE data to D0 - D9
1	X	1	unused	

Table 8.1: Setup of the test control unit.

8.4.2 Register Map Cutout

This register map is only important for testing with the built in function generator. Table 8.3 shows only the registers which are relevant for testing the DAC. They include registers for setting up the test environment and for programming the digital function generator.

Pin Number	Pin Name	Direction	Short Description
1	POR_N*	Output	Power-on Reset (to be forwarded to RST_N)
2	RST_N*	Input	external Reset or via POR_N
3	V0	Input	BT.656 Video Data [0]
4	V1	Input	BT.656 Video Data [1]
5	V2	Input	BT.656 Video Data [2]
6	V3	Input	BT.656 Video Data [3]
7	V4	Input	BT.656 Video Data [4]
8	V5	Input	BT.656 Video Data [5]
9	V6	Input	BT.656 Video Data [6]
10	V7	Input	BT.656 Video Data [7]
11	VDD1V8ENC	Input/Output	Digital Core Supply (1.8 V)
12	VDD3V3ENC	Input/Output	Digital Peripheral Supply (3.3 V)
13	VSSENC	Input/Output	Digital Ground
14	SCL	Input	I2C Clock
15	SDA	Input	I2C Data
16	LT	Output	Line Trigger
17	DEVA0	Input	I2C Device Address [0]
18	DEVA1	Input	I2C Device Address [1]
19	DVET	Input	DVE Test (to test control unit, see 8.4.1)
20	DACE	Input	DAC Enable (to test control unit, see 8.4.1)
21	DACT	Input	DAC Test (to test control unit, see 8.4.1)
22	VDDA	Input/Output	DAC Supply (1.8 V)
23	IDUMMY	Output	DAC Dummy Current Output (to be terminated)
24	VDDA	Input/Output	DAC Supply (1.8 V)
25	VSSA	Input/Output	DAC Ground
26	IBIAS_5u	Output	DAC Reference Current Output (5 µA)
27	NC		Not Connected
28	VDDA	Input/Output	DAC Supply (1.8 V)
29	IOUT	Output	DAC Current Output (to be terminated)
30	VDDA	Input/Output	DAC Supply (1.8 V)
31	NC		Not Connected
32	VSSD	Input/Output	DAC Digital Ground
33	VDDD	Input/Output	DAC Digital Supply (1.8 V)
34	D0	Input/Output	Video Encoder Test Output or DAC Test Input [0]
35	D1	Input/Output	Video Encoder Test Output or DAC Test Input [1]
36	D2	Input/Output	Video Encoder Test Output or DAC Test Input [2]
37	D3	Input/Output	Video Encoder Test Output or DAC Test Input [3]
38	D4	Input/Output	Video Encoder Test Output or DAC Test Input [4]
39	D5	Input/Output	Video Encoder Test Output or DAC Test Input [5]
40	D6	Input/Output	Video Encoder Test Output or DAC Test Input [6]
41	D7	Input/Output	Video Encoder Test Output or DAC Test Input [7]
42	D8	Input/Output	Video Encoder Test Output or DAC Test Input [8]
43	D9	Input/Output	Video Encoder Test Output or DAC Test Input [9]
44	CLK	Input	27 MHz Clock

* .. activ low signals

Table 8.2: Pinout of the video encoder test-chip.

Addr.	Name	Sub-Register Name	Reg. Bit	Default Value	Description
00_h	CONFIG	dac_fg_enable	5	0	enable DAC function generator
		dvencoder_enable	4	0	enable Digital Video Encoder
		dac_enable	3	0	enable DAC
		sel_data	2:0	0	select data to forward to the DAC
02_h	PAD_CONFIG	line_trigger_oen	4	1	LT pull enable
		cfg_enable_pe	3	1	DVET, DACT, DACE pull enables
		data_pe	2	1	D9-D0 pull enables
		deva_pe	1	1	DEVA1 and DVA0 pull enables
		video_pe	0	1	V9-V0 pull enables
90_h	DAC_FG_AD_SHAPE_OFFS	dac_fg_offset	7:2	0	Offset Control
		dac_fg_waveform	1:0	0	Waveform selection
91_h	DAC_FG_AD_FREQ_AMPL	dac_fg_segment	7:6	0	Segment selection
		dac_fg_magnitude	5:3	0	Magnitude Control
		dac_fg_resolution	2:0	0	Resolution Control

Table 8.3: Relevant registers for DAC Testing.

8.4.3 Test-setup with I2C

To make a function test without an external signal source a simple digital function generator is implemented. This FG generates ramps, sinusoids, triangular and rectangular signals with adjustable offset, magnitude and resolution (see section 6.2.1). To set up the FG registers, an I2C interface is implemented. For testing more than one test-chip on the same evaluation board - for example to test the video performance - the I2C device address for each chip can be programmed with the DEVA1 and DEVA0 input pins. If only one test-chip is placed on the evaluation board the DEVA1 and DEVA0 pins can be connected to ground. These pins are the least significant bit of the 8 bit I2C device address. Table 8.4 shows the composition of the I2C device address. For the default setup, with DEVA1 and DEVA0 connected to ground, the device address for write is $A8_h$ and for read $A9_h$.

Bit Number	7	6	5	4	3	2	1	0
Bit Value	1	0	1	0	1	DEVA1	DEVA0	R/Wn

Table 8.4: Composition of the I2C device address.

Chapter 9

Measurements and Results of the Video DAC

The practical part of work for this chapter is done by Ungvári [2010]. The main topic of the bachelor's thesis from Ungvári [2010] is the evaluation of the video DAC. The information in this chapter is only related to the work of Ungvári [2010] and therefore an exact statement about the quality of the evaluations can not be given.
This chapter is organized as follows. The first part deals with the measurement setup. Especially the evaluation board with the socket for the video encoder test-chip is described. This part is followed by a summary of the measurement results and finally the evaluation of the results.

9.1 Evaluation Board

The evaluation board is used to bring the test-chip in its operating mode and evaluate some of the typical DAC characteristics. Therefore some adjustable components, for example, the input bias current sink, are needed. To test more than one engineering sample of the test-chip with the same evaluation board a socket is used to carry the chips. For the configuration, digital input and output signals of the chip, multi-pin connectors are used. The clock input and current output are connected by Bayonet Neill-Concelman (BNC) connectors.
The board schematic and layout are designed with the printed circuit board (PCB) schematic and layout software Easily Applicable Graphical Layout Editor (EAGLE).

9.1.1 Board Schematic

The schematic is based on the suggested board schematic in figure 8.2. Additionally, a few connectors for easier measuring and an IC with the external current sink including are added. Figure 9.1 presents the board schematic of the evaluation board drawn with EAGLE.

9.1.2 Board Layout

After merging the instances in the schematic to layout footprints a PCB layout can be generated. Figure 9.2 shows the top view of the EAGLE PCB layout. In the center of the PCB the footprint of the test-chip socket can be found. On the top, bottom, and left side around the test-chip many connectors for the digital inputs, outputs and test-chip configurations are placed. At the right edge of the PCB the adjustable input current sink, with its current mirrors are located. The output termination resistors and the output BNC connector are above the current sink. The power supply connectors are at the right top and bottom corners. The free space of the PCB is filled up with ground plates on both sides. An image of the manufactured and assembled PCB is presented in figure 9.3.

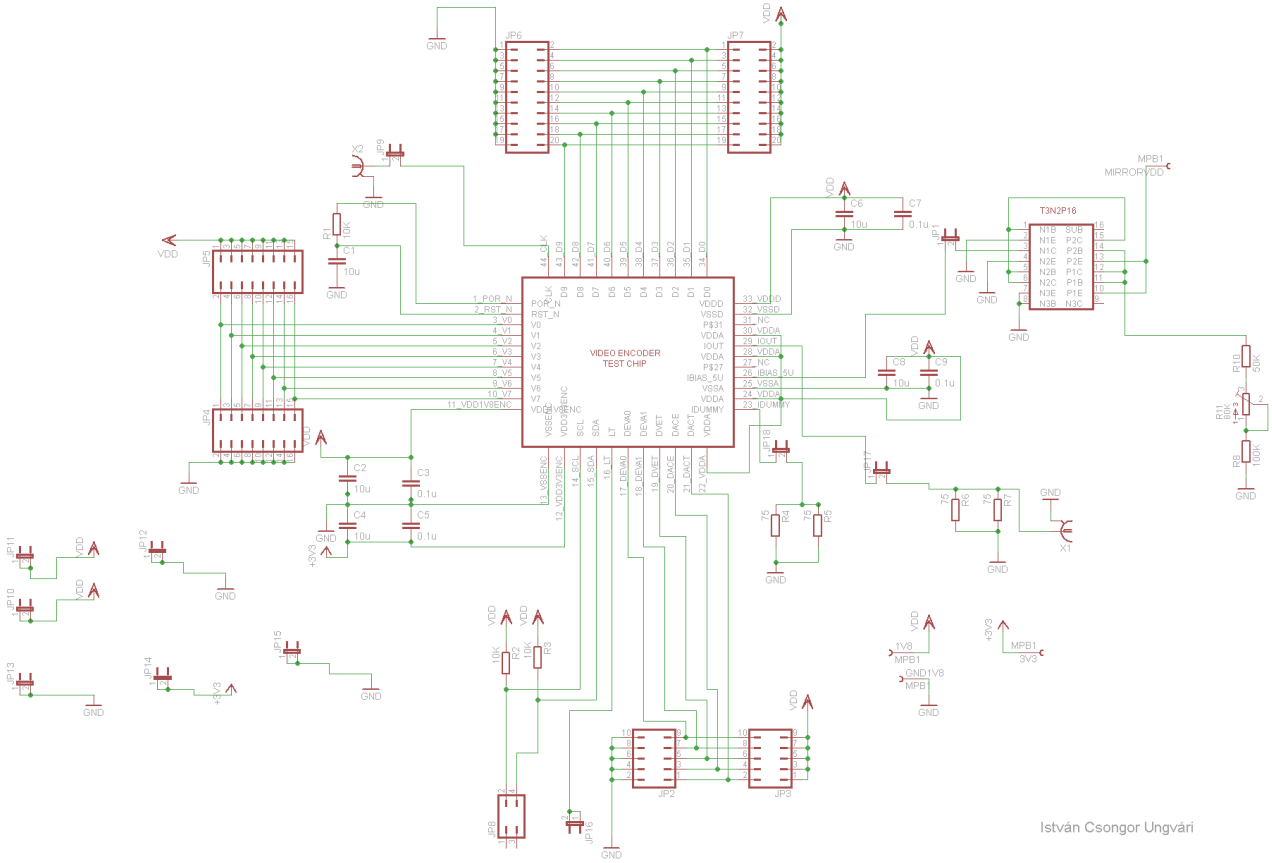

Figure 9.1: EAGLE board schematic of the evaluation board. [Ungvári, 2010]

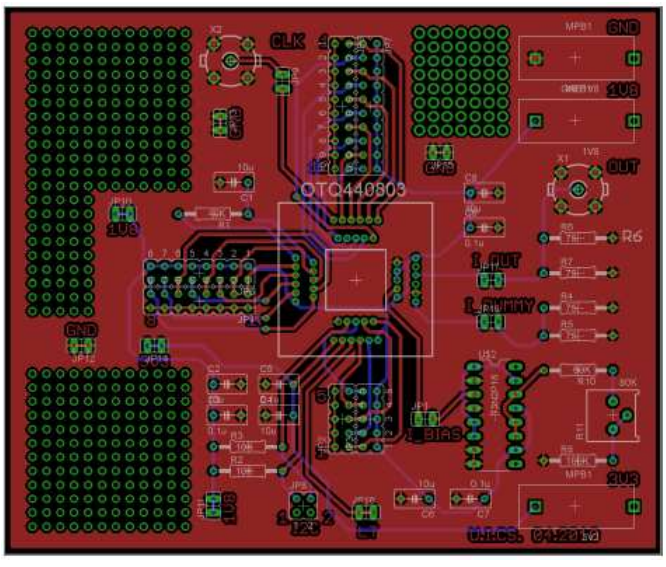

Figure 9.2: EAGLE PCB layout of the evaluation board. [Ungvári, 2010]

9.2 Measurement Setup with the PXI System

For measuring and evaluating chips, the IFE at TU Graz has a PCI eXtensions for Instrumentation (PXI) system from National Instruments (NI). The PXI system is a modular measuring system, which can also

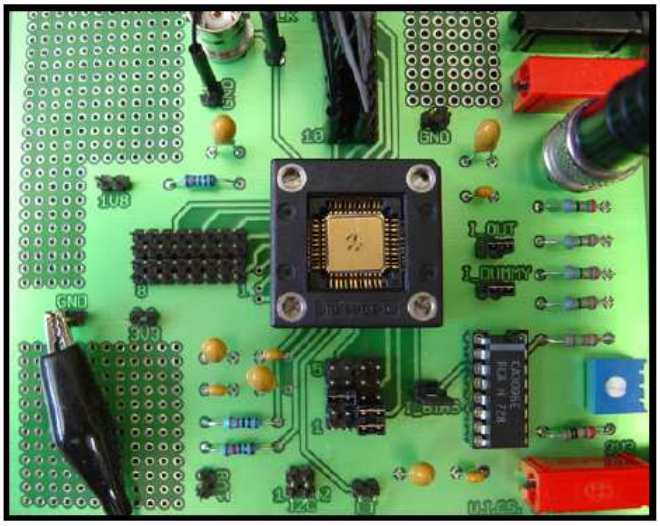

Figure 9.3: Image of the assembled evaluation board. [Ungvári, 2010]

perform automatic tests. The PXI system of the IFE has different modules for power supply, analog and digital outputs, and several analog and digital inputs for measuring. The controller of the PXI system is a normal Personal Computer (PC) with a Microsoft Windows operating system, but it can also operate stand-alone with its embedded microcontroller. As programming language the Laboratory Virtual Instrumentation Engineering Workbench (LabVIEW) also from NI is used. The hard- and software are described in detail in the following sections.

9.2.1 Hardware Description of the PXI System

The hardware of the PXI system has a main chassis, which comes with an embedded controller. It has several slots to carry different instrumentation modules. Figure 9.4 shows an image of a PXI system. The system on the image is not exactly the used PXI setup, but the chassis is of the same series. For stimu-

Figure 9.4: Image of a PXI system.

lating the video DAC the NI PXI-6551 is used. This is a $50\,\mathrm{MHz}$ digital stimulus/response module and is called High Speed Digital Input Output (HSDIO) module. The output voltage level is programmable between $-2.0\,\mathrm{V}$ and $5.5\,\mathrm{V}$. It has 20 channels with per-cycle, per-channel bidirectional control. For the analog data acquisition a module called Dynamic Signal Acquisition (DSA) NI PXI-4461 is used. This DSA module has $24\,\mathrm{bit}$ resolution with sample rates up to $200\,\mathrm{kS/s}$. The DSA module has to be

connected to the HSDIO module to synchronize the acquisition with the input pattern. To control this modules a LabVIEW program, described in the next section, is used.

9.2.2 Labview Measurement Program

The LabVIEW tool is a visual programming language for programming and automating measuring equipments. In this case it is used to stimulate and measure the test-chip evaluation board by using the PXI hardware. Ungvári [2010] has implemented two different LabVIEW setups which can be found in figure 9.5 and 9.6. The left part of figure 9.5 is mainly the test stimulus part of the program. With the waveform graph modules the output and input signals can be displayed as a graph like oscilloscopes do. The rightmost box shows the file writing module, where the measurement results can be saved in files for further use.

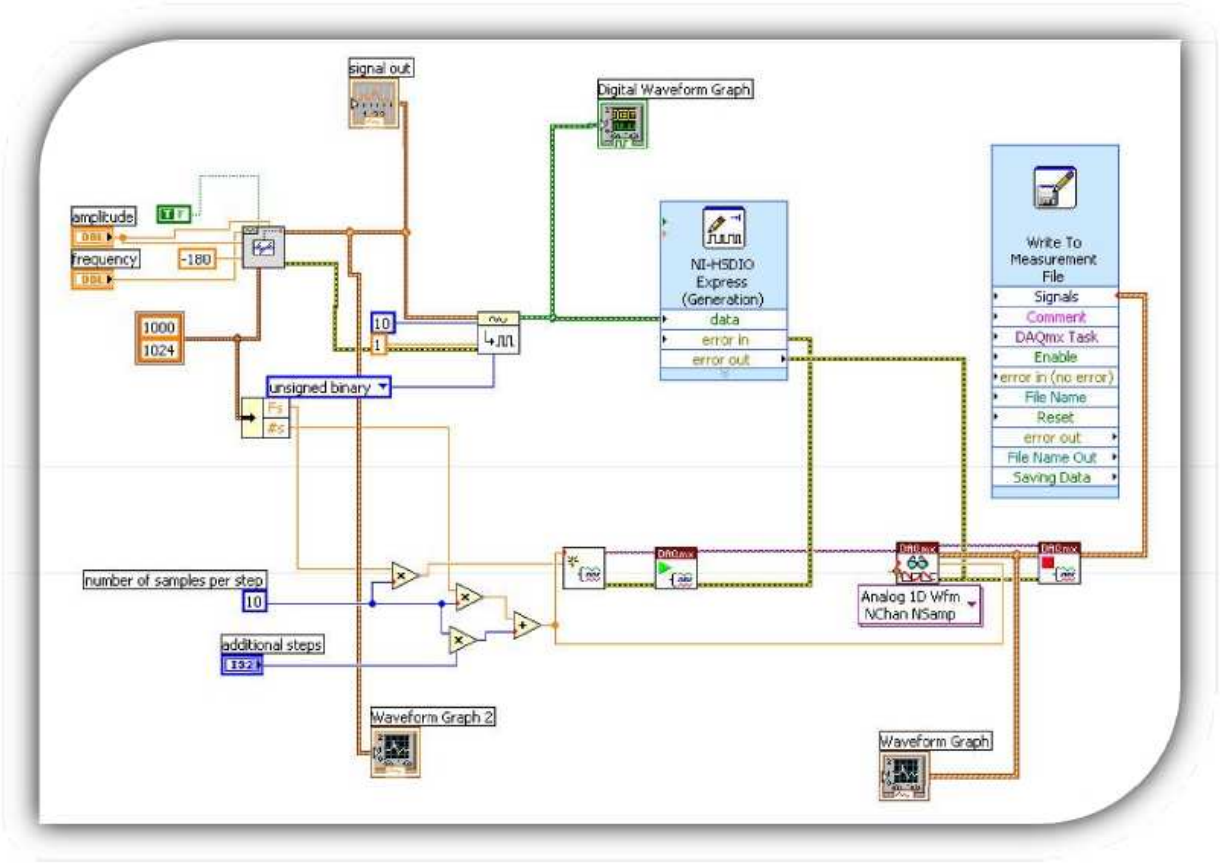

Figure 9.5: First LabVIEW measurement setup. [Ungvári, 2010]

The second figure (see 9.6) is an extension of the first program in figure 9.5. Ungvári [2010] has added some analyzing modules like the distortion measurement and the spectral measurement module to measure the noise at the DAC output. Some of the measurement results can be found in the next sections.

9.3 Measurement Results and Evaluation

The measurements of the test-chip is done with the above described PXI system. All measurements are written to text files and analyzed afterwards. For the post-processing, tools like MATLAB from MathWorks® and Excel from Microsoft® are used.

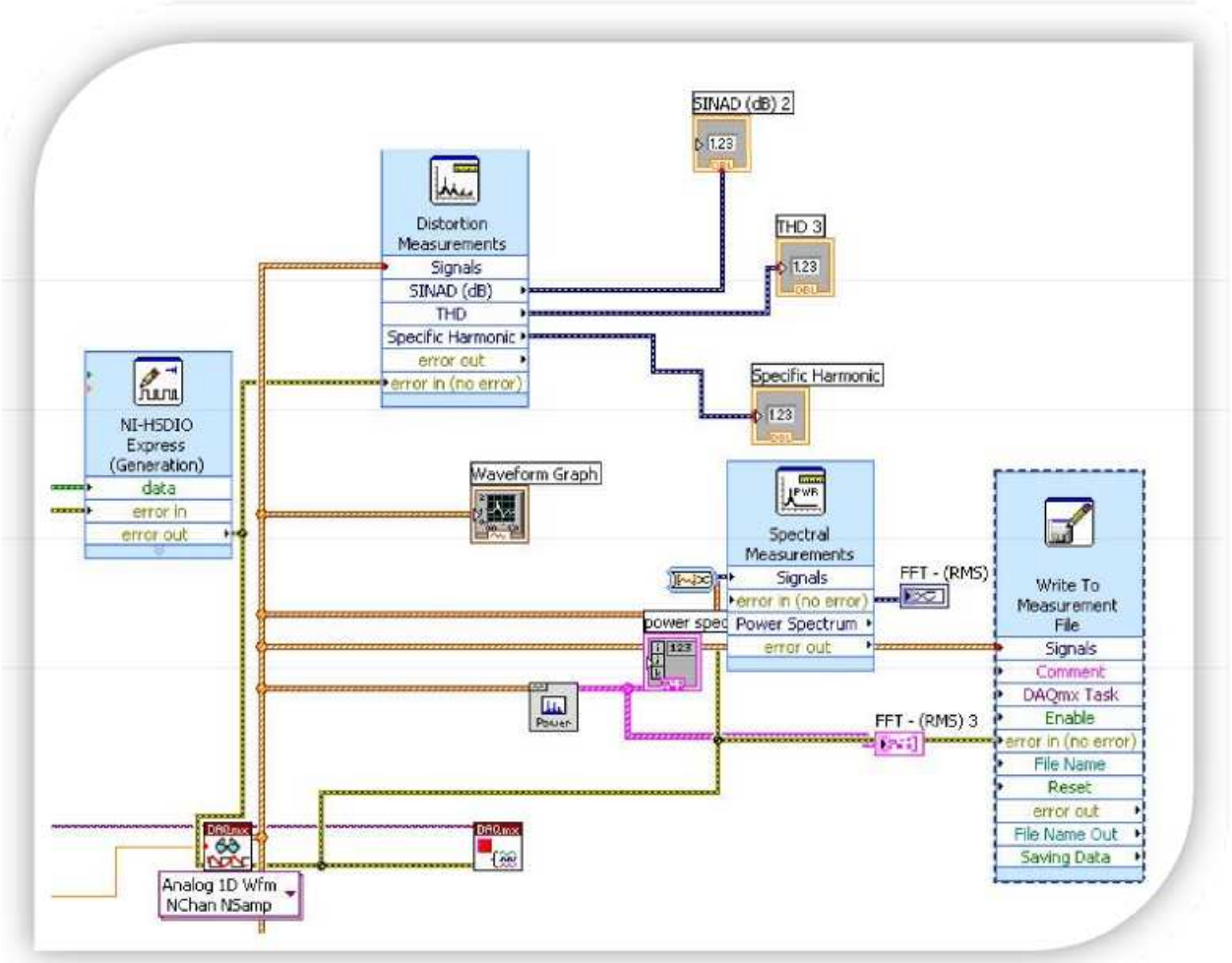

Figure 9.6: Extention of the LabVIEW measurement setup.[Ungvári, 2010]

For calculating DNL and INL equations 6.3 and 6.4 are used. The resulting graph is presented in figure 9.7. The DNL shows a good performance of less than ± 0.5 LSB, but the INL has deviations of more than ± 2 LSB by using the optimistic "best fit line" method. This value is too high for DACs. However, it is not clear, if the measurements are correct and reliable. The other results can be found in the bachelor's thesis of Ungvári [2010].

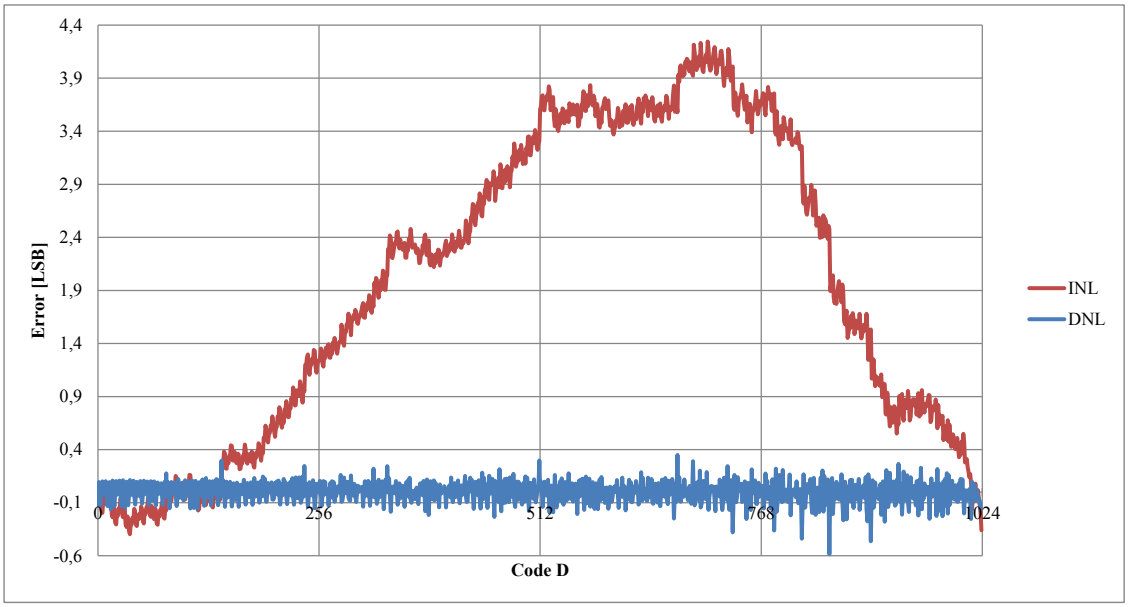

Figure 9.7: INL and DNL calculation of the PXI evaluation data.

Chapter 10

Outlook

This chapter first looks at some general trends in converting digital video informations into PAL or NTSC signals and then explores some ideas for future work.

10.1 General Trends

The general trend in the area of digital video information is not to convert them into analog signals like PAL or NTSC anymore. Today most of the mobile devices like digital photo cameras and portable music and video players have a PAL or NTSC interface. Due to the rush demand on Liquid Crystal Displays (LCD) television (TV) screens, which are mostly equipped with a high-definition multimedia interface (HDMI) it is no longer necessary to convert the digital video signals into an analog video format like PAL or NTSC. More and more mobile devices like mobile phones and tablet computers are also provided with an HDMI interface. For some applications fast DACs may be required, but for converting digital video signals they will be obsolete.

10.2 Ideas for Future Work

The characterization of the test-chip is not fully done. To get a complete data sheet of the video DAC, all corners and properties of the DAC should be measured and documented. For some applications in other areas this low power 10 bit DAC may be the right selection.

Chapter 11

Concluding Remarks

In this thesis a work in the field of designing and layouting a semiconductor chip was presented.
Having begun the thesis with an analysis of DACs generally in chapter 2 and especially current steering DACs in chapter 3. Chapter 4 dealt with an overview of basic layout techniques like matching structures, avoidance of influencing parasitics and similar topics.
The second part of the work was the documentation of the practical work itself. Chapter 5 starts with the basic concept and current mirror calculations. Followed by chapter 6, where the calculated values of chapter 5 were simulated and verified. The practical part was closed by the layout implementation of the video DAC in chapter 7.
The third part was affected by the test-chip specification in chapter 8 and the measurements and results in chapter 9.
The thesis concluded with an analysis of current trends and some ideas for future work. Additionally, the final schematics and verilog source codes were added to the appendix.

Appendix A

Schematics and Verilog Sourcecodes

The following sections summarize the final implementation of the Cadence® schematics and verilog source codes.

A.1 Cadence Schematics

The screenshots of the Cadence® schematics are starting from the top of the DAC-IP and is going down until the current source units.

A.1.1 Video DAC Top Schematic

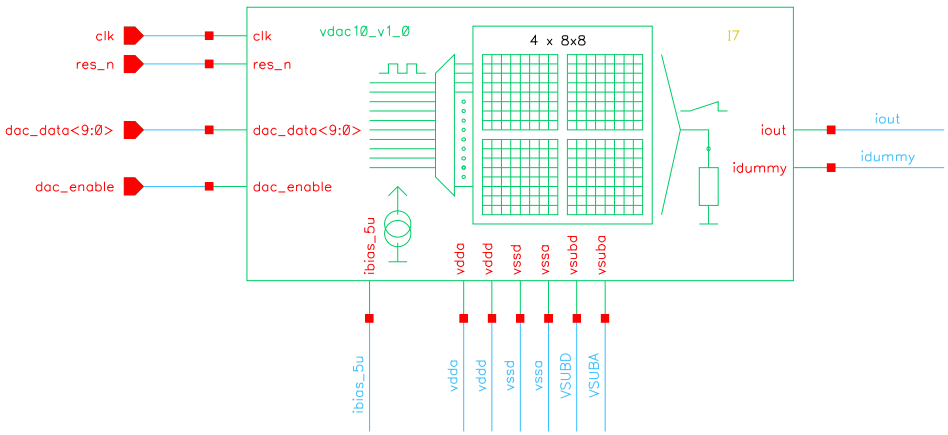

Figure A.1: Top schematic of the video DAC.

A.1.2 Analog Part plus Digital Interface

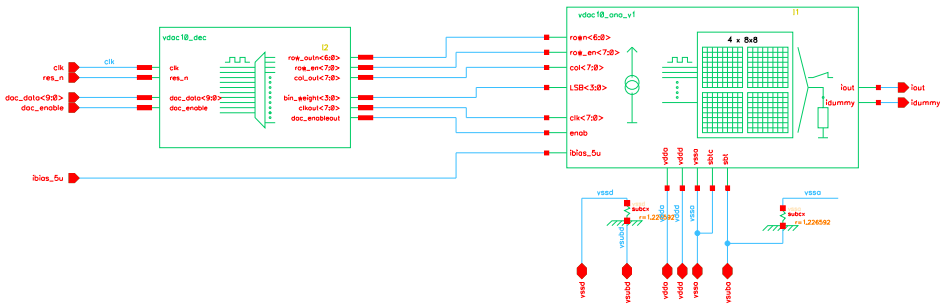

Figure A.2: Digital interface with thermometer encoder connected to the analog part of the DAC.

A.1.3 Analog Part of the DAC

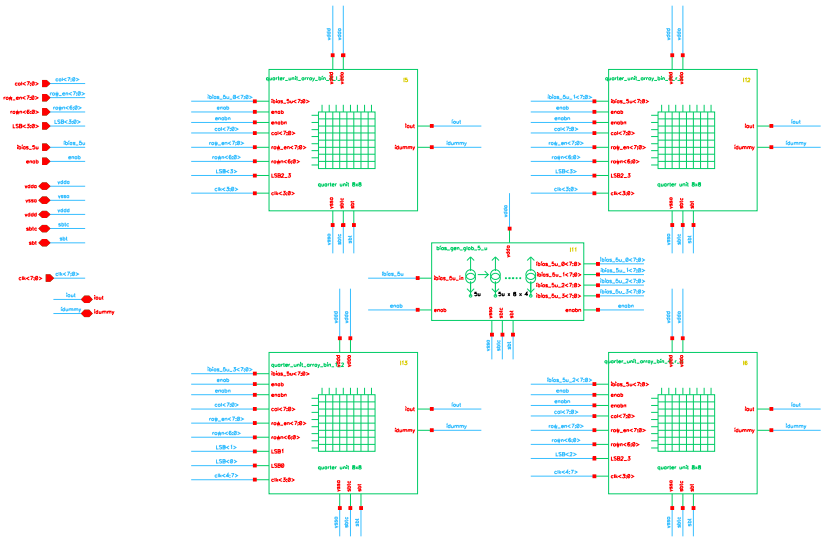

Figure A.3: Analog part of the video DAC.

A.1.4 Digital Part of the DAC

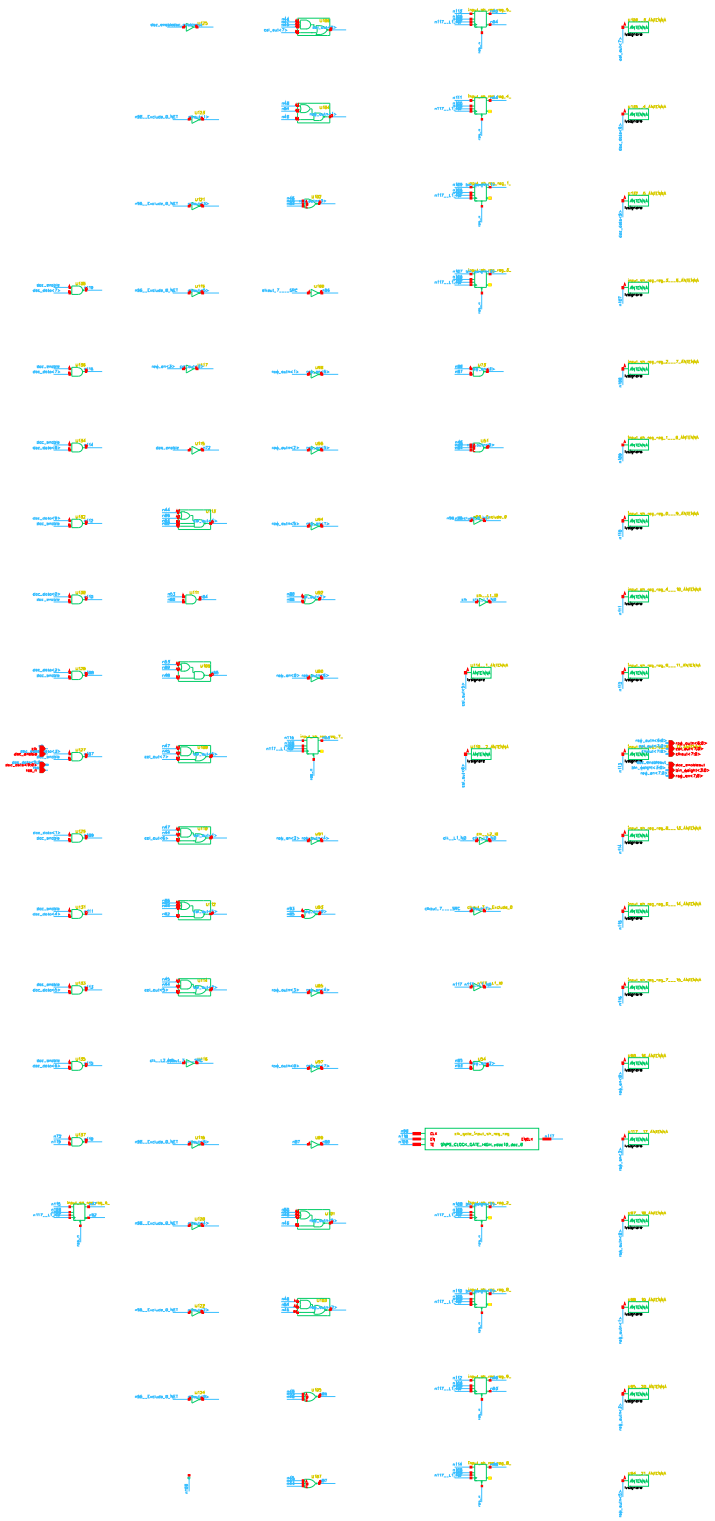

Figure A.4: Synthesized digital part of the video DAC.

A.1.5 Global Bias Generation

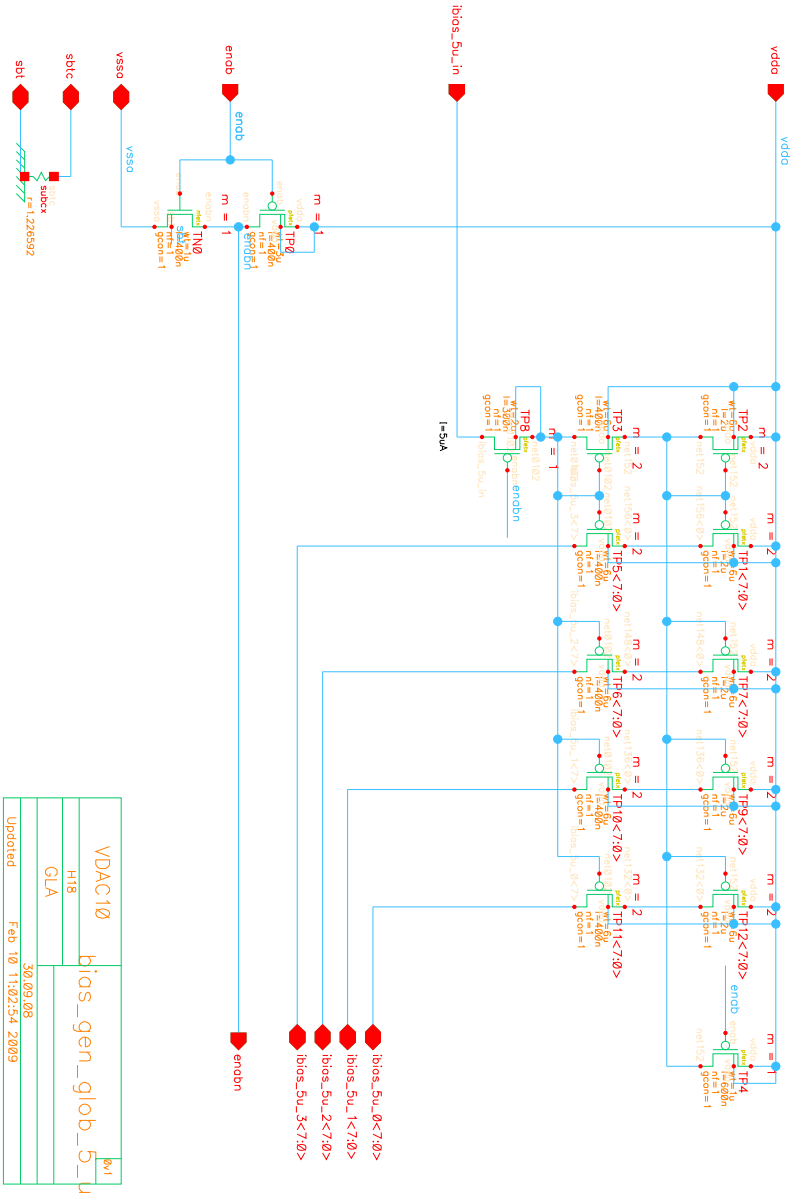

Figure A.5: Global bias generation module with one 5 µA input and 32 5 µA outputs.

A.1.6 Local Bias Generation

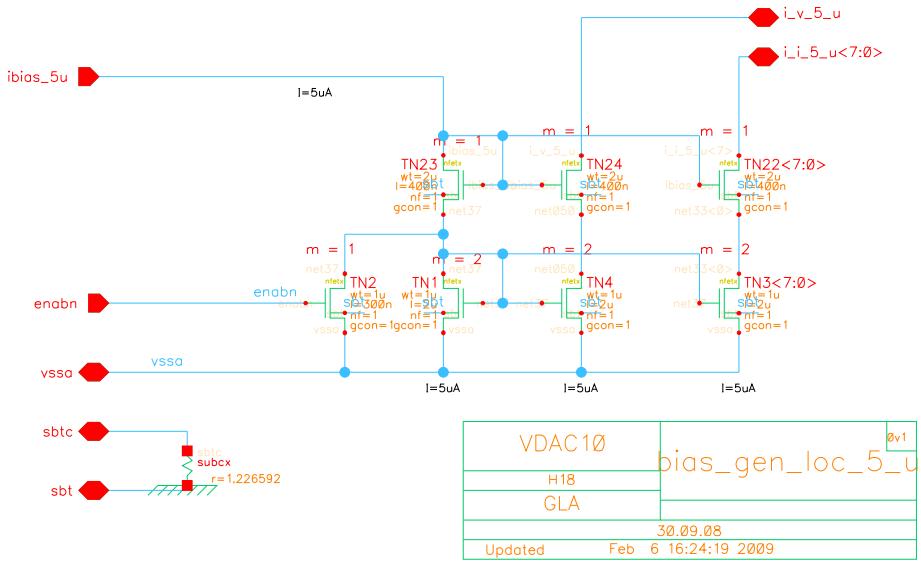

Figure A.6: Local bias generation module with one 5 µA input and 9 5 µA outputs.

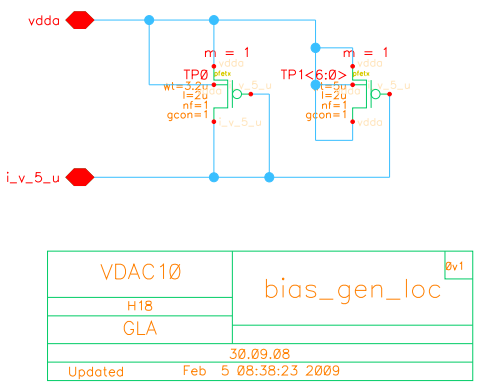

Figure A.7: Local bias diode module for cascode gate voltage generation.

A.1.7 Quarter Unit Arrays

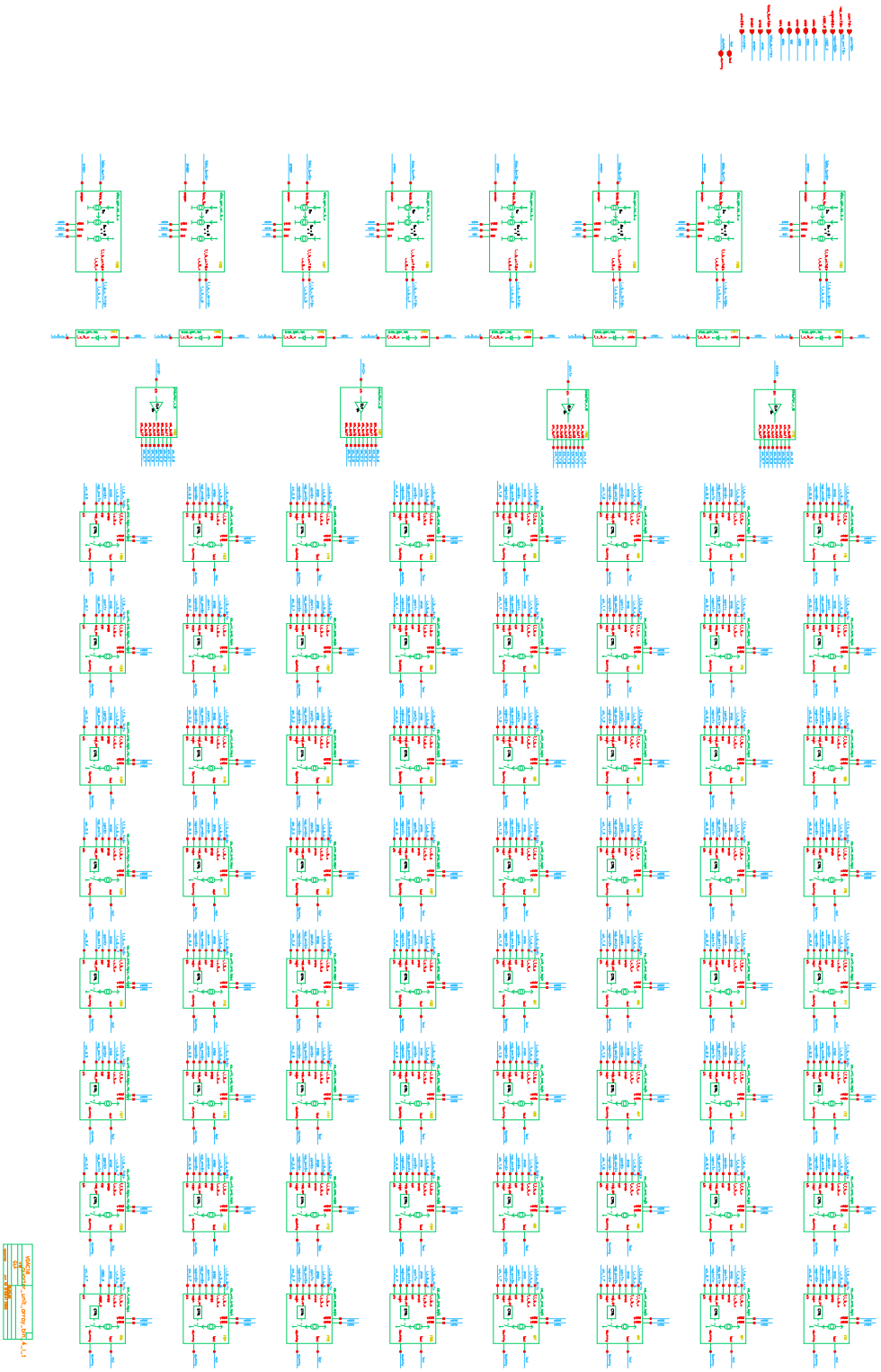

Figure A.8: Left top quarter unit array.

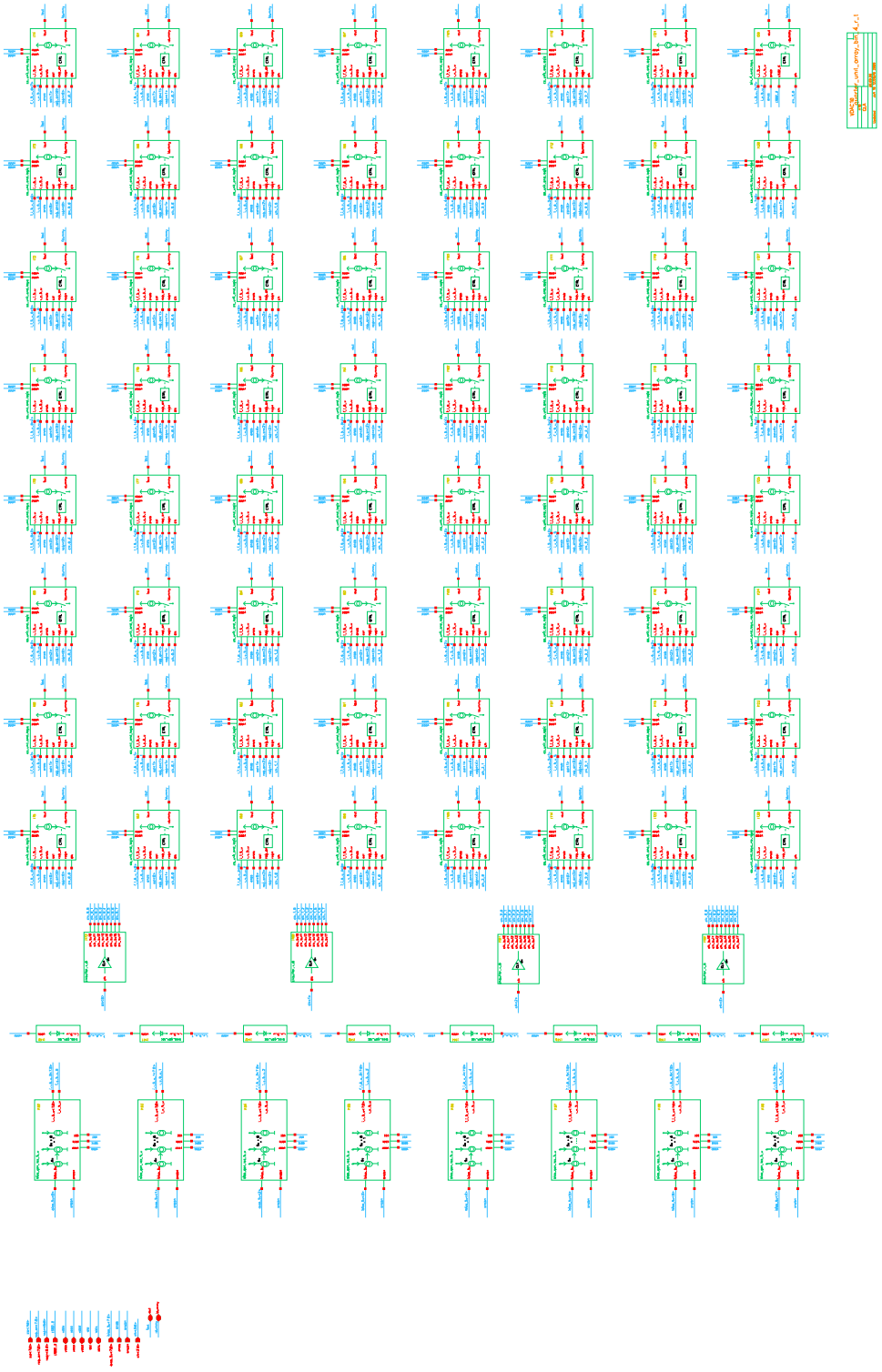

Figure A.9: Right top quarter unit array.

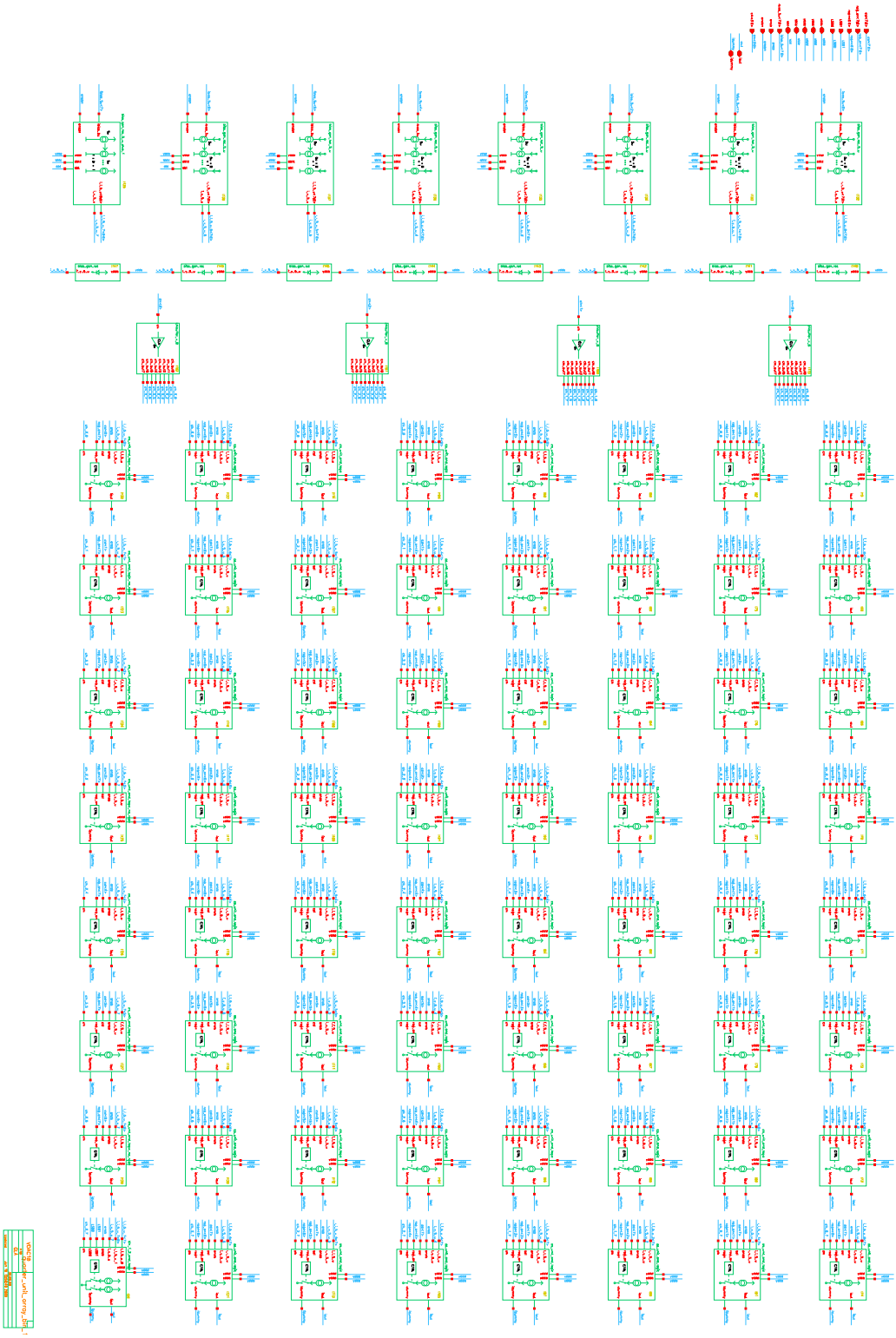

Figure A.10: Left bottom quarter unit array.

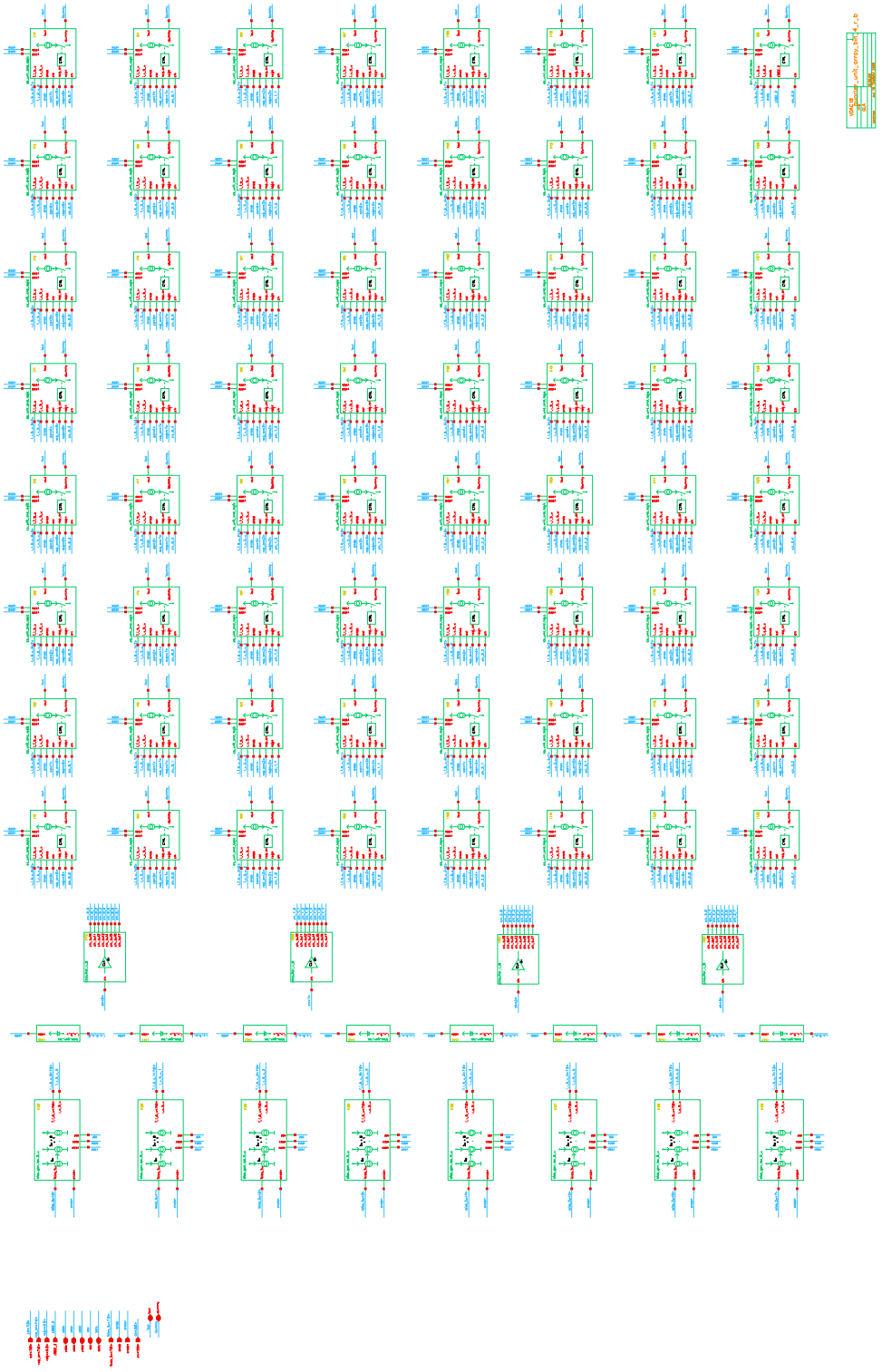

Figure A.11: Right bottom quarter unit array.

A.1.8 Clock Buffer

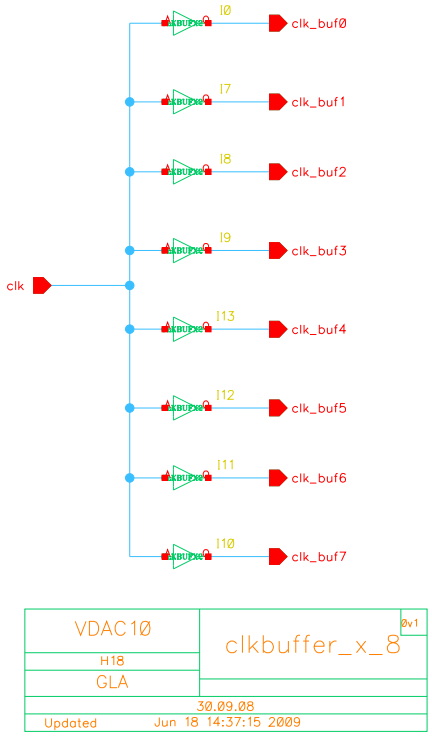

Figure A.12: Clock buffer for manual clock-tree generation.

A.1.9 Current Source Units

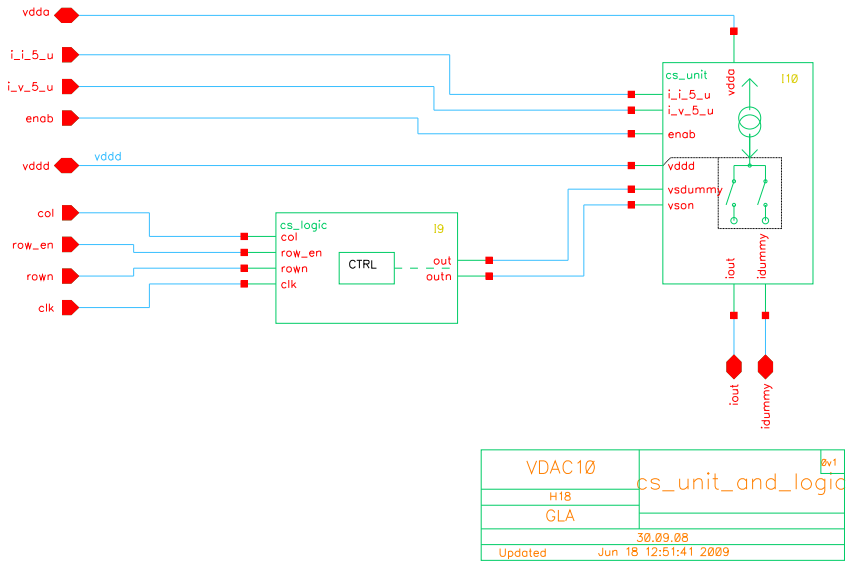

Figure A.13: Current source unit in combination with local decoder logic.

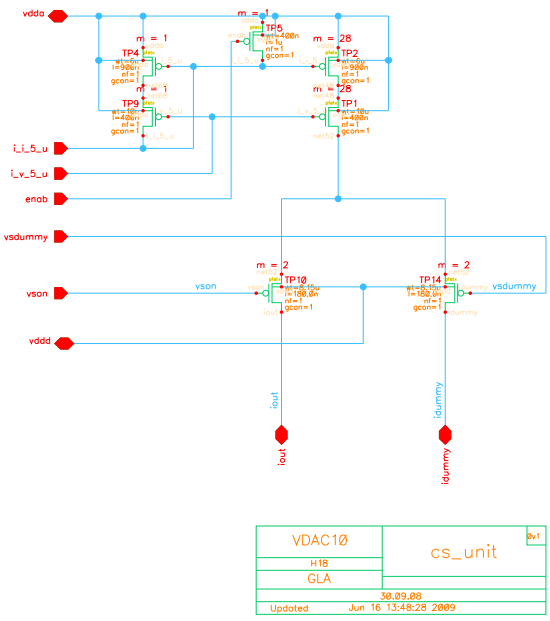

Figure A.14: Thermometer coded current source unit.

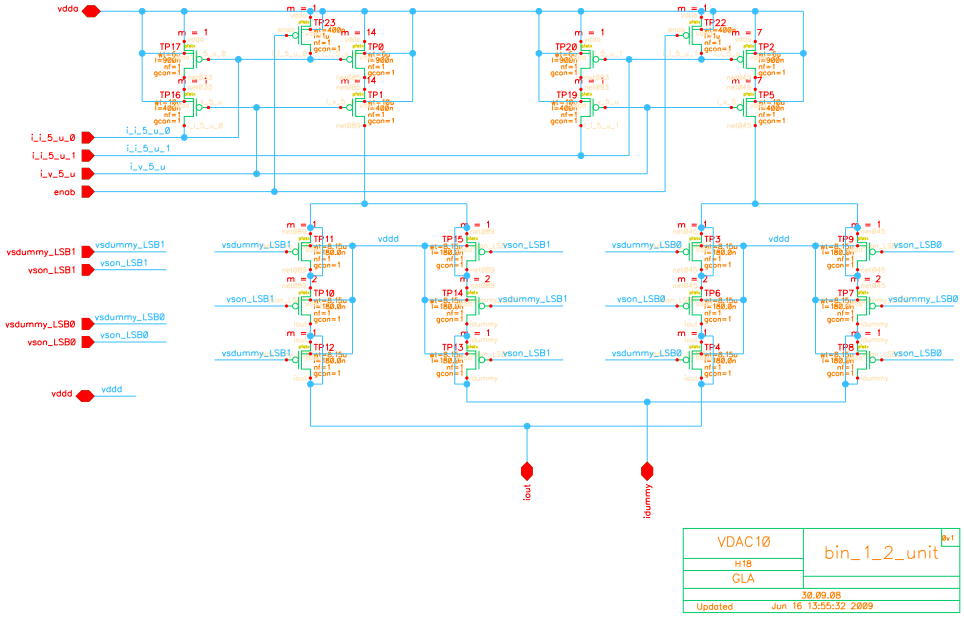

Figure A.15: Binary weighted current source unit for bit 0 and 1.

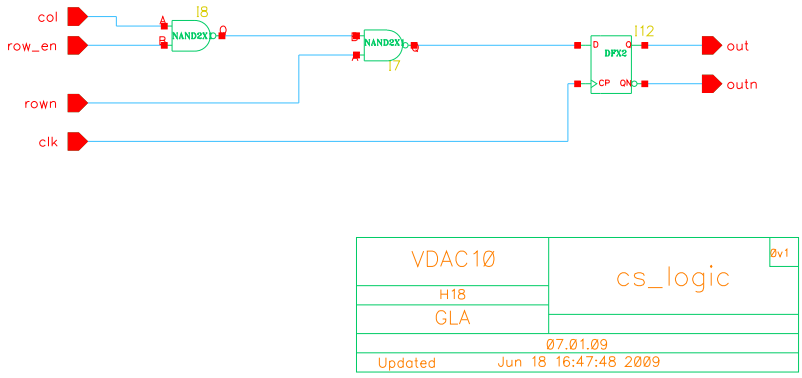

Figure A.16: Thermometer coded local logic for current source unit.

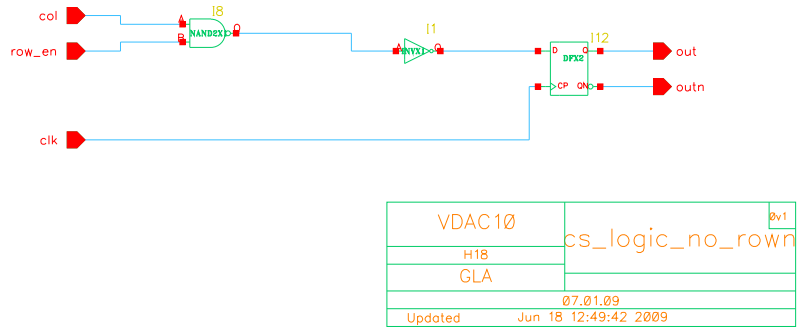

Figure A.17: Thermometer coded local logic for current source unit of last row without the row signal.

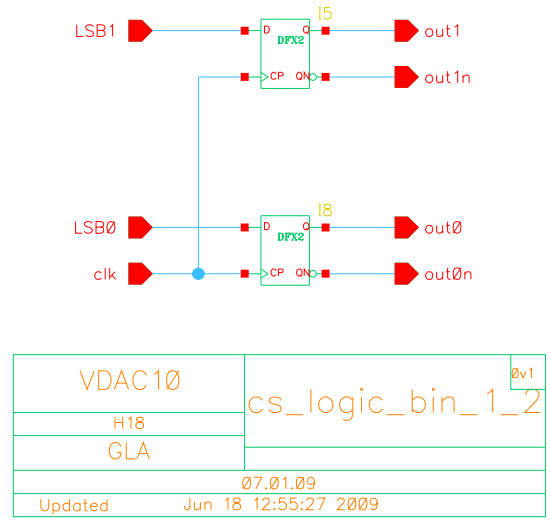

Figure A.18: Binary weighted local logic for current source unit of bit 0 and 1.

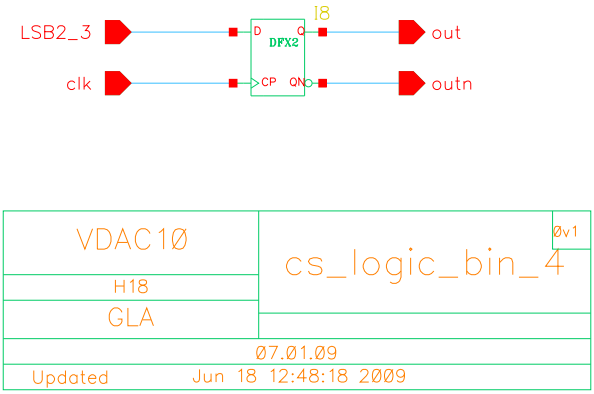

Figure A.19: Binary weighted local logic for current source unit of bit 3.

A.2 Verilog Sourcecode

The verilog source code section presents the source codes of the input interface with the thermometer decoder and the function generator, which is also implemented on the DVE test-chip.

A.2.1 Thermometer Decoder

```
1  `timescale 1ns/10ps
2  module vdac10_dec(
3                    input wire clk,
4                    input wire res_n,
5                    input wire dac_enable,
6                    input wire [9:0] dac_data,
7
```

```
8                   output wire [6:0] row_outn,
9                   output wire [7:0] row_en,
10                  output wire [7:0] col_out,
11                  output wire [3:0] bin_weight,
12                  output wire [7:0] clkout,
13                  output wire dac_enableout
14                  );
15
16    reg [9:0] input_sh_reg;
17    reg [7:0] row_out;
18    reg [7:0] col;
19
20    always @(negedge clk or negedge res_n)
21      if(~res_n)
22        input_sh_reg <= 10'b0000000000;
23      else if(dac_enable)
24        input_sh_reg <= dac_data;
25
26    //wire clk and dac_enable througt module to enable buffering with
         encounter
27    assign clkout[7:0] = {clk,clk,clk,clk,clk,clk,clk,clk};
28    assign dac_enableout = dac_enable;
29
30    //connect 4 input lsb's direct to bin_weight output
31    assign bin_weight = input_sh_reg[3:0];
32    // wrapp column and row signals to get better matching in analog
         layout
33    assign col_out = {col[7],col[5],col[3],col[1],col[0],col[2],col[4],col
         [6]};
34    assign row_en = {row_out[7],row_out[5],row_out[3],row_out[1],row_out
         [0],row_out[2],row_out[4],row_out[6]};
35    assign row_outn = ~{row_out[6],row_out[4],row_out[2],row_out[1],
         row_out[3],row_out[5],row_out[7]};
36
37   // row adn column     decoder
38    always @(input_sh_reg)
39      begin
40        // row decoder
41        case(input_sh_reg[9:4])
42          6'b000001, 6'b000010, 6'b000011, 6'b000100, 6'b000101, 6'
               b000110, 6'b000111, 6'b001000: row_out = 8'b00000001;
43          6'b001001, 6'b001010, 6'b001011, 6'b001100, 6'b001101, 6'
               b001110, 6'b001111, 6'b010000: row_out = 8'b00000011;
44          6'b010001, 6'b010010, 6'b010011, 6'b010100, 6'b010101, 6'
               b010110, 6'b010111, 6'b011000: row_out = 8'b00000111;
45          6'b011001, 6'b011010, 6'b011011, 6'b011100, 6'b011101, 6'
               b011110, 6'b011111, 6'b100000: row_out = 8'b00001111;
46          6'b100001, 6'b100010, 6'b100011, 6'b100100, 6'b100101, 6'
               b100110, 6'b100111, 6'b101000: row_out = 8'b00011111;
47          6'b101001, 6'b101010, 6'b101011, 6'b101100, 6'b101101, 6'
               b101110, 6'b101111, 6'b110000: row_out = 8'b00111111;
48          6'b110001, 6'b110010, 6'b110011, 6'b110100, 6'b110101, 6'
               b110110, 6'b110111, 6'b111000: row_out = 8'b01111111;
49          6'b111001, 6'b111010, 6'b111011, 6'b111100, 6'b111101, 6'
               b111110, 6'b111111           : row_out = 8'b11111111;
50          default: row_out = 8'b0000000;
51        endcase // case (input_sh_reg[9:4])
```

```
52
53          // column decoder
54           case(input_sh_reg[9:4])
55             6'b000001, 6'b001001, 6'b010001, 6'b011001, 6'b100001, 6'
                   b101001, 6'b110001, 6'b111001: col = 8'b00000001;
56             6'b000010, 6'b001010, 6'b010010, 6'b011010, 6'b100010, 6'
                   b101010, 6'b110010, 6'b111010: col = 8'b00000011;
57             6'b000011, 6'b001011, 6'b010011, 6'b011011, 6'b100011, 6'
                   b101011, 6'b110011, 6'b111011: col = 8'b00000111;
58             6'b000100, 6'b001100, 6'b010100, 6'b011100, 6'b100100, 6'
                   b101100, 6'b110100, 6'b111100: col = 8'b00001111;
59             6'b000101, 6'b001101, 6'b010101, 6'b011101, 6'b100101, 6'
                   b101101, 6'b110101, 6'b111101: col = 8'b00011111;
60             6'b000110, 6'b001110, 6'b010110, 6'b011110, 6'b100110, 6'
                   b101110, 6'b110110, 6'b111110: col = 8'b00111111;
61             6'b000111, 6'b001111, 6'b010111, 6'b011111, 6'b100111, 6'
                   b101111, 6'b110111, 6'b111111: col = 8'b01111111;
62             6'b001000, 6'b010000, 6'b011000, 6'b100000, 6'b101000, 6'
                   b110000, 6'b111000:            col = 8'b11111111;
63             default:

                   col = 8'b00000000;
64           endcase // case (input_sh_reg[9:4])
65
66       end // always @ (input_sh_reg)
67
68   endmodule // vdac10_dec
```

A.2.2 Function Generator

```
1  timescale 1ns/10ps
2  module dac_fg
3    (/*AUTOARG*/
4     // Outputs
5     dac_fg_pattern,
6     // Inputs
7     clk, rstn, dac_fg_offset, dac_fg_waveform, dac_fg_segment,
8     dac_fg_magnitude, dac_fg_resolution, dac_fg_enable
9     );
10
11    // ------------------------------------------------------------------------
12    input              clk;
13    input              rstn;
14
15    input [5:0]        dac_fg_offset;
16    input [1:0]        dac_fg_waveform;
17    input              dac_fg_segment;
18    input [2:0]        dac_fg_magnitude;
19    input [2:0]        dac_fg_resolution;
20    input              dac_fg_enable;
21
22    // ------------------------------------------------------------------------
23    output [9:0]       dac_fg_pattern;
24
25    // ------------------------------------------------------------------------
```

```
26      wire              count_minmax;
27      reg               count_on;
28
29      wire [9:0]        rect_out;
30      wire [9:0]        sine_out;
31      wire [9:0]        triang_out;
32      wire [9:0]        ramp_out;
33      wire [8:0]        rect;
34      reg [8:0]         sine;
35      wire [8:0]        sine_mag;
36      wire [8:0]        triang;
37      wire [8:0]        ramp;
38      wire              neg_npos;
39      reg [1:0]         neg_npos_counter;
40
41      wire [9:0]        pattern_int;
42      wire [5:0]        pointer_counter;
43
44      reg [2:0]         ctrl_state;
45
46      wire [8:0]        loadval;
47      wire [8:0]        limitval;
48      reg               count_up;
49      reg               count_down_nup;
50      reg [8:0]         count_result;
51      wire              count_end;
52
53      // ---------------------------------------------------------------------
54      reg [8:0]  uplimit;
55      reg [8:0]  downlimit;
56
57
58      //ctrl states
59      parameter ST_IDLE    = 3'b000; // idle
60      parameter ST_RAMP    = 3'b001; // setup ramp function
61      parameter ST_TRIANG  = 3'b010; // setup triangle function
62      parameter ST_RECT    = 3'b011; // setup rectangle function
63      parameter ST_SINE    = 3'b100; // setup sine function
64      parameter ST_CNT_RUN = 3'b101; // counter run
65
66      // ---------------------------------------------------------------------
67      assign count_minmax = ~dac_fg_segment;
68
69      // ---------------------------------------------------------------------
70      always @(posedge clk or negedge rstn)
71        if(~rstn)
72          begin
73             count_on <= 1'b0;
74             ctrl_state <= 3'h0;
75             downlimit <= 9'h000;
76             uplimit <= 9'h1FF;
77          end
78        else
79          begin
80
81             if(dac_fg_enable)
82               begin
```

```
83                case(ctrl_state)
84                  ST_IDLE:
85                    begin
86                      case(dac_fg_waveform)
87                        2'b00: ctrl_state <= ST_RAMP;
88                        2'b01: ctrl_state <= ST_TRIANG;
89                        2'b10: ctrl_state <= ST_SINE;
90                        2'b11: ctrl_state <= ST_RECT;
91                      endcase
92                    end
93                  ST_RAMP:
94                    begin
95                      uplimit <= 9'h1ff >> dac_fg_resolution;
96                      downlimit <= 9'h000;
97                      ctrl_state <= ST_CNT_RUN;
98                    end
99                  ST_TRIANG:
100                   begin
101                     uplimit <= 9'h1ff >> dac_fg_resolution;
102                     downlimit <= 9'h000;
103                     ctrl_state <= ST_CNT_RUN;
104                   end
105                 ST_RECT:
106                   begin
107                     uplimit <= 9'h1ff >> dac_fg_resolution;
108                     downlimit <= 9'h000;
109                     ctrl_state <= ST_CNT_RUN;
110                   end
111                 ST_SINE:
112                   begin
113                     uplimit <= 9'h03f >> dac_fg_resolution;
114                     downlimit <= 9'h000;
115                     ctrl_state <= ST_CNT_RUN;
116                   end
117                 ST_CNT_RUN:
118                   begin
119                     count_on <= 1'b1;
120                   end
121                 default:
122                   begin
123                     uplimit <= 9'h1ff;
124                     downlimit <= 9'h000;
125                     ctrl_state <= ST_IDLE;
126                   end
127               endcase
128             end
129           else
130             begin
131               count_on <= 1'b0;
132               ctrl_state <= 3'h0;
133             end
134         end
135
136     // ---------------------------------------------------------------------
137     assign pattern_int = (dac_fg_waveform == 2'b11) ? (rect_out)   :
138                          (dac_fg_waveform == 2'b10) ? (sine_out)   :
139                          (dac_fg_waveform == 2'b01) ? (triang_out) :
```

```
140                          (dac_fg_waveform == 2'b00) ? (ramp_out >>
                                dac_fg_magnitude)   : 10'h000;
141     assign dac_fg_pattern = (dac_fg_enable&count_on) ? pattern_int : 10'
           h000;
142
143     assign neg_npos = (~dac_fg_segment) ? neg_npos_counter[1] :
           neg_npos_counter[0];
144
145     // ramp
146     assign ramp     = (dac_fg_waveform == 2'b00) ? (count_result <<
           dac_fg_resolution) : 9'h000;
147     assign ramp_out = ~neg_npos ? (ramp + {dac_fg_offset,4'b0000}) : (512
           + ramp + {dac_fg_offset,4'b0000});
148
149     // triangle
150     assign triang     = (dac_fg_waveform == 2'b01) ? ((count_result <<
           dac_fg_resolution )>>dac_fg_magnitude) : 9'h000;
151     assign triang_out = neg_npos ? ((511 + dac_fg_offset) - triang) :
           ((512 + dac_fg_offset) + triang);
152
153     // sinewave
154     assign pointer_counter = (dac_fg_waveform == 2'b10) ? (count_result <<
            dac_fg_resolution) : 6'h00;
155     assign sine_mag        = sine >> dac_fg_magnitude;
156     assign sine_out        = neg_npos ? ((511 + dac_fg_offset) - sine_mag)
            : ((512 + dac_fg_offset) + sine_mag);
157
158     // rectangle
159     assign rect     = (dac_fg_waveform == 2'b11) ? (9'h1ff >>
           dac_fg_magnitude): 9'h000;
160     assign rect_out = neg_npos ? ((511 + dac_fg_offset) - rect) : ((512 +
           dac_fg_offset) + rect);
161
162
163     // ---------------------------------------------------------------------
164     always @(posedge clk or negedge rstn)
165       begin
166          if(~rstn)
167            neg_npos_counter <= 2'b00;
168          else
169            begin
170               if(count_on)
171                 begin
172                    if(count_end)
173                      neg_npos_counter <= neg_npos_counter + 1;
174                    else
175                      neg_npos_counter <= neg_npos_counter;
176                 end
177               else
178                 begin
179                    neg_npos_counter <= 2'b00;
180                 end
181            end
182       end
183
184     // ---------------------------------------------------------------------
185     always @(/*AS*/pointer_counter)
```

```
186         case(pointer_counter)
187           6'd0:    sine = 9'd6;
188           6'd1:    sine = 9'd19;
189           6'd2:    sine = 9'd31;
190           6'd3:    sine = 9'd44;
191           6'd4:    sine = 9'd56;
192           6'd5:    sine = 9'd69;
193           6'd6:    sine = 9'd81;
194           6'd7:    sine = 9'd94;
195           6'd8:    sine = 9'd106;
196           6'd9:    sine = 9'd118;
197           6'd10:   sine = 9'd130;
198           6'd11:   sine = 9'd142;
199           6'd12:   sine = 9'd154;
200           6'd13:   sine = 9'd166;
201           6'd14:   sine = 9'd178;
202           6'd15:   sine = 9'd190;
203           6'd16:   sine = 9'd201;
204           6'd17:   sine = 9'd213;
205           6'd18:   sine = 9'd224;
206           6'd19:   sine = 9'd235;
207           6'd20:   sine = 9'd246;
208           6'd21:   sine = 9'd257;
209           6'd22:   sine = 9'd268;
210           6'd23:   sine = 9'd279;
211           6'd24:   sine = 9'd289;
212           6'd25:   sine = 9'd299;
213           6'd26:   sine = 9'd309;
214           6'd27:   sine = 9'd319;
215           6'd28:   sine = 9'd329;
216           6'd29:   sine = 9'd338;
217           6'd30:   sine = 9'd348;
218           6'd31:   sine = 9'd357;
219           6'd32:   sine = 9'd366;
220           6'd33:   sine = 9'd374;
221           6'd34:   sine = 9'd383;
222           6'd35:   sine = 9'd391;
223           6'd36:   sine = 9'd399;
224           6'd37:   sine = 9'd407;
225           6'd38:   sine = 9'd414;
226           6'd39:   sine = 9'd421;
227           6'd40:   sine = 9'd428;
228           6'd41:   sine = 9'd435;
229           6'd42:   sine = 9'd441;
230           6'd43:   sine = 9'd448;
231           6'd44:   sine = 9'd454;
232           6'd45:   sine = 9'd459;
233           6'd46:   sine = 9'd465;
234           6'd47:   sine = 9'd470;
235           6'd48:   sine = 9'd474;
236           6'd49:   sine = 9'd479;
237           6'd50:   sine = 9'd483;
238           6'd51:   sine = 9'd487;
239           6'd52:   sine = 9'd491;
240           6'd53:   sine = 9'd494;
241           6'd54:   sine = 9'd497;
242           6'd55:   sine = 9'd500;
```

```
243         6'd56:   sine = 9'd502;
244         6'd57:   sine = 9'd505;
245         6'd58:   sine = 9'd506;
246         6'd59:   sine = 9'd508;
247         6'd60:   sine = 9'd509;
248         6'd61:   sine = 9'd510;
249         6'd62:   sine = 9'd511;
250         6'd63:   sine = 9'd511;
251         default: sine = 9'd0;
252       endcase // case (count_result)
253
254  // counter
255
256     // ------------------------------------------------------------------
257     // ctrl state machine
258     always @(posedge clk or negedge rstn)
259       if(~rstn)
260         begin
261            count_down_nup <= 1'b0;
262            count_result <= 9'h000;
263            count_up <= 0;
264         end
265       else
266         begin
267            if(count_on)
268              begin
269                 if(~count_down_nup)
270                   begin
271                      count_up <= 1;
272                      if(count_end)
273                        begin
274                           count_result <= loadval;
275                           if(count_minmax)
276                             begin
277                                count_result <= loadval;
278                                count_down_nup <= 1'b1;
279                             end
280                           else
281                             begin
282                                count_result <= downlimit;
283                                count_down_nup <= 1'b0;
284                             end
285                        end
286                      else
287                        begin
288                           count_down_nup <= 1'b0;
289                           count_result <= count_result + 9'b000000001;
290                        end // else: !if(count_end)
291                   end // if (~count_down_nup)
292                 else
293                   begin
294                      count_up <= 0;
295                      if(count_end)
296                        begin
297                           count_down_nup <= 1'b0;
298                           count_result <= loadval;
299                        end
```

```
300                     else
301                       begin
302                          count_down_nup <= 1'b1;
303                          count_result <= count_result - 9'b000000001;
304                       end
305                   end // else: !if(~count_down_nup)
306               end // if (count_on)
307             else
308               begin
309                  count_down_nup <= 1'b0;
310                  count_result <= 9'h000;
311               end // else: !if(count_on)
312         end // else: !if(~rstn
313
314     assign limitval  = count_down_nup ? (downlimit) : (uplimit);
315     assign count_end = count_on  ? (count_result == limitval) : 1'b0;
316     assign loadval = count_up ? (uplimit-1): (downlimit);
317
318 endmodule
```

Bibliography

Andrews, Keith [2006]. *Writing a Thesis: Guidelines for Writing a Master's Thesis in Computer Science*. http://ftp.iicm.edu/pub/keith/thesis/. Graz University of Technology, Austria. (Cited on page xv.)

Christopher Saint, Judy Saint [2002]. *IC Mask Design: Essential Layout Techniques*. 3 Edition. Mcgraw-Hill Professional. ISBN 0071389962. (Cited on pages v, 24, 25, 26, 30 and 31.)

Clein, Dan [2000]. *CMOS IC layout: Concepts, Methodologies, and Tools*. Newnes. ISBN 0750671947. (Cited on page 23.)

Georgi I. Radulov, Hans A. Hegt, Patrick Quinn and Arthur H.M van Roermund [2006]. *A Binary-To-Thermometer Decoder with redundant switching sequences*. In *Proceedings of the 17th ProRISC, Annual Workshop on Circuits, Systems and Signal Processing (ProRISC 2006)*, pages 330–333. Technology Foundation. (Cited on page 22.)

Hastings, Alan [2001]. *The Art of ANALOG LAYOUT*. Prentice Hall. ISBN 0130870617. (Cited on page 31.)

Hugo Hermandez, Wilhelmus Van Noije and Elkim Roa [2007]. *Design Strategy of Current Source in Current Steering CMOS DAC*. In *Proceedings of the IBERCHIP'07. Lima, Peru*. (Cited on page 20.)

ITU [1998]. *RECOMMENDATION ITU-R BT.656-4*. (Cited on page 77.)

Kyocera [2011]. *CQFP 44 LD*. Intranet of austriamicrosystems AG. (Cited on pages vi and 76.)

Michiel Steyaert, Arthur H. M. van Roermund, Johan H. Huijsing [2002]. *Analog circuit design: scalable analog circuit design, high speed D/A converters, RF power amplifiers*. Springer. ISBN 0792376218. (Cited on page 15.)

Miquel Albiol, José Luis González and Eduard Alarcón [2004]. *Mismatch and Dynamic Modeling of Current Sources in Current-Steering CMOS D/A Converters: An Extended Design Procedure*. *IEEE Transactions on Circuits and Systems - I: Regular Papers*, 51(1), pages 159–169. doi:10.1109/TCSI.2003.821287. (Cited on page 20.)

Phillip E. Allen, Douglas R. Holberg [2000]. *CMOS Analog Circuit Design*. Preliminary Version of Second Edition. Oxford University Press. (Cited on pages 3 and 4.)

R. Jacob Baker, David E. Boyce, Harry W. Li [1998]. *CMOS: Circuit Design, Layout, And Simulation*. IEEE Press Series on Microelectronic Systems, IEEE Press. ISBN 0780334167. (Cited on pages 3 and 9.)

Razavi, Behzad [2001]. *Design of analog CMOS integrated circuits*. McGraw-Hill. ISBN 0071188150. (Cited on page 18.)

Ungvári, István Csongor [2010]. *Evaluation of an Integrated Digital-to-Analog Converter for Video Signals using a PXI Test System*. Bachelor-thesis, Technical University of Cluj-Napoca and Graz University of Technology, Austria. (Cited on pages vi, xv, 2, 81, 82, 83, 84 and 85.)

van de Plassche, Rudy [1994]. *Integrated Analog-to-Digital and Digital-to-Analog Converters*. First Edition. Kluwer Academic Publishers. ISBN 0792394364. (Cited on pages 3 and 4.)

Wikipedia [2009]. *Quantization error*. `http://en.wikipedia.org/wiki/Quantization_error`. (Cited on page 5.)