Compensation and Calibration Techniques for High Performance Current-Steering DACs

DISSERTATION

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Samantha M. McDonnell, B.S.E.E., M.S.

Graduate Program in Electrical and Computer Engineering

The Ohio State University

2016

Dissertation Committee:

Dr. Waleed Khalil, Advisor

Dr. Joel Johnson

Dr. Steven Bibyk

Abstract

A myriad of research efforts, covering architectural, circuit and technological aspects, have been made towards improving the performance of digital-to-analog converters (DACs). However, the quest to achieve stringent dynamic linearity requirements (> 70dBc SFDR) over many gigahertz of bandwidth presents grand challenges to circuit designers and high-yield manufacturers. In light of these challenges, various calibration and compensation techniques have evolved over the past two decades to overcome design and process technology limitations.

In this work, the sources of nonlinearity in current-steering DACs are described and common circuit techniques, along with device technologies, which enable a high-performance baseline DAC, are detailed. The effect these non-idealities are investigated using a new modeling paradigm which accurately predicts the SFDR of the DAC in the presence of statistical process mismatch. This model offers a total speedup of ~330x per frequency point compared with Monte Carlo-based simulation methods. The model is useful in evaluating and developing calibration circuits, which is the focus of this work.

To highlight prior art, a historical overview of compensation and calibration techniques is presented, outlining the shift from amplitude to timing and dynamic correction**.** Furthermore, several techniques are simulated using the DAC model to compare their efficacy. In addition, current and emerging architectures are described, which help extend the synthesizable bandwidth of the DAC. As operating frequency

increases beyond several MHz, timing errors are detrimental to DAC performance, yet few timing calibration techniques have been developed and verified on chip. This highlights the need for a novel timing calibration technique.

A new timing calibration, termed adaptive delay calibration (ACD), is designed in support of this work. It reduces timing mismatches in the DAC through modulation of the clock signal's delay to the DAC retiming data latches. The proposed technique is shown to reduce the effect of timing mismatches and improve the SFDR for both a single DAC and time interleaved (TI) DAC, as well as improve the signal-to-image rejection ratio (SIRR) in the TI DAC. To verify the ACD technique, a 14-bit DAC operating at f_{CLK} =3GHz is designed in a 130nm BiCMOS process. The DAC includes amplitude calibration, dynamic compensation, and the proposed ACD technique. Overall, ACD calibration shows promising results, however, it was not able to be verified on chip due to unforeseen design issues. These issues are investigated and proposed solutions are provided.

Dedication

To my husband Christopher.

Acknowledgments

I would like to thank my family: My parents that instilled good values and drive to pursue my education, to my sister and her family who have been here to support me, and to my husband who uprooted his life to be with me in Columbus during my education.

Thanks to the members of the CLASS lab for all their help and knowledge through the years. I'm especially indebted to: Jamin McCue for mentoring me throughout the years and for the design of several circuits used in this work, Luke Duncan for the calibration measurement circuits used in this work and the time he has spent helping with my calibration paper, and Darren Disabato for his help in board design.

To the RYDI team at AFRL, thank you for your guidance and support through the years, I'm looking forward to joining your team. A special thanks to Vipul Patel for the time he has spent on the calibration paper.

I would like to thank Dr. Brian Dupaix who has given much of his time to review papers and help with measurement setup.

Many thanks to my committee members, Dr. Steven Bibyk and Dr. Johnson, for their insight and for the many valuable classes I took with them.

Finally, I would like to thank my advisor, Dr. Waleed Khalil, for believing in me and encouraging me to pursue my education. None of this would be possible, without him.

And to anyone I forgot….

-Thank you

Vita

2008…….. B.S.E.C.E. (Magna Cum Laude), The Ohio State University

2008-2010…… Recipient of the Intel Foundation SRC GRC Master's Scholarship Program

2010…….. DAGSI PhD Fellowship

2010…….. M.S.E.C.E., The Ohio State University

2010-2011……. Recipient of the 2010 DAGSI PhD Fellowship

2011-2016…… Recipient of the SMART PhD Fellowship

Publications

In-Progress

- **S.M. McDonnell**, B. Dupaix, and W. Khalil, "Statistical Modeling and Parametric Yield Prediction for CMOS Current-Steering DACs."
- **S.M. McDonnell**, B. Dupaix, L. Duncan, V.J. Patel, and W. Khalil. "Survey of Calibration and Compensation Techniques for Current-Steering DACs."

Journals

- J. McCue, L. Duncan, **S.M. McDonnell**, B. Dupaix, and W. Khalil, W. "A Time-Interleaved Multi-mode ΔΣ RF-DAC for Direct Digital-to-RF Synthesis." *IEEE Journal of Solid-State Circuits*, 2016

- S. Balasubramanian, G. Creech, J. Wilson, **S.M. Yoder**, J.J. McCue, M. Verhelst, and W. Khalil, "Systematic Analysis of Interleaved Digital-to-Analog Converters," in Circuits and Systems II: Express Briefs, IEEE Transactions on , vol.58, no.12, pp.882-886, Dec. 2011.
- John Hu, Mark Haffner, **Samantha Yoder**, Gursharan Reehal, Mark Scott, and Mohammed Ismail, "An industry-driven laboratory development for mixed-signal IC test education," Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on , vol., no., pp.85-88, May 30 2010-June 2 2010.

Conference Proceedings

- **S. M. Yoder**, S. Balasubramanian, W. Khalil, and V.J. Patel, "Accuracy and speed limitations in DACs across CMOS process technologies," in Circuits and Systems (MWSCAS), 2013 IEEE 56th International Midwest Symposium on pp.868-871, 4-7 Aug. 2013
- **Samantha Yoder**, Mohammed Ismail, and Waleed Khalil. GOMACTech 2011, Paper/Presenter "Analysis and Modeling of Non-Idealities in VCO-Based Quantizers using Frequency-to-Digital and Time-to-Digital Converters"

Books and Book Chapters

- **Samantha Yoder**, Mohammed Ismail, and Waleed Khalil. VCO-Based Quantizers Using Frequency-to-Digital and Time-to-Digital Converters, SpringerBriefs in Electrical and Computer Engineering. New York, NY 2011.

Fields of Study

Major Field: Electrical and Computer Engineering

Specialization: Analog and RF Electronics

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

The rapid growth of mobile and fixed broadband access has been made possible through the continued digital scaling called for by Moore's law. However, as the shift occurs to Gbps wireless and Tbps wired networks, one perennial challenge is how to move such unprecedented bandwidths of data from the digital domain to the physical (analog) domain. This calls for high performance digital-to-analog converters (DACs) that are capable of synthesizing signals with complex M-QAM modulation schemes, while covering several GHz of bandwidth (BW). A typical power spectral density (PSD) of a wireless or wireline infrastructure DAC is displayed in Figure 1.1. The DAC synthesizes several channels with an aggregate BW that is limited to half the DAC update frequency (f_{CLK}). The DAC's SNR budget, and hence its resolution (N), is set by several effects: the required modulation accuracy, peak-to-average power ratio (PAPR), and dynamic power control back-off. For example, a 512-QAM OFDM transmitter with 10dB

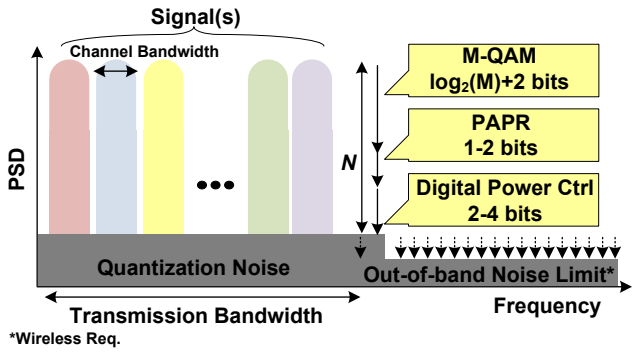


Figure 1.1: Typical multiband transmitter PSD

PAPR and 12dB digital control range requires a DAC with at least N =14-bits. In the case of wireless infrastructure, the DAC's out-of-band noise is most critical and ends up setting its resolution. Table 1.1 highlights current design trends for high performance DACs covering telecommunication[1][2][3] and radar[4] applications.

Table 1.1: Current DAC trends in telecommunication applications

Application	# of Bits (N)	BW (MHz)	f_{CLK} (MHz)
Cable Modem	14	1000	4600
Cellular Basestation	16	500	2800
Satellite	12	1000	3000
Radar	12	2000	4000

Several DAC architectures exist to meet application demands with some better suited for either large N or high f_{CLK}. For example, instrumentation and audio applications use Delta-Sigma ($\Delta\Sigma$) DACs which oversample and noise shape data to achieve high-resolution ($N \geq 16$) within a fraction of $f_{CLK} <$ 10MHz. Resistor string and $R2R$ DACs are utilized in industrial process control and data acquisition systems [1] that require moderate speed ($f_{CLK} <$ 1MHz) and moderate resolution ($8 \leq N \leq 16$). Due to their fast switching current configuration, current-steering DACs are suitable for high-speed ($f_{CLK} >$10MHz) applications that require moderate resolution.

Across DAC architectures, output amplitude errors limit the accuracy upwards of DC. Applications demanding moderate to high-resolution typically require amplitude

[1] Cable modem specifications: Maxim Integrated Circuits (MAX5882)
[2] Cellular basestation specifications: Texas Instruments (DAC39J84). BW based on LTE 5x pre-distortion
[3] Satellite specifications: E2V (EV12DS130). BW based on C-band satellite.
[4] Radar specifications: Maxim Integrated Circuits (MAX19693). BW based on X-band radiolocation.

correction: ΔΣ DACs apply dynamic element matching (DEM) to average amplitude errors [2], resistor string and $R2R$ DACs remove amplitude errors through trimming resistors [3], and current-steering DACs utilize on-chip amplitude error correction methods. Additionally, DACs operating beyond several MHz experience static timing and dynamic non-idealities that result in a frequency dependent degradation in DAC linearity. Since the current-steering architecture is the linchpin for GHz operation, numerous calibration and compensation techniques have been proposed to improve its linearity.

1.1. Current-Steering DAC Overview

The simplest form of an N-bit current-steering DAC, shown in Figure 1.2(a), consists of an array of cells which contain a binary weighted current source ($I_k = 2^k I_{LSB}, k = N-1, \dots, 1, 0$) and switch controlled by an input digital code ($b_k \epsilon \{0,1\}$). Depending on b_k, current is steered to and from the output combining load resistor (R_L) providing the DC output voltage

$$V_{out} = R_L \cdot I_{LSB} \sum_{k=0}^{k=N-1} b_k \cdot 2^k. \quad (1.1)$$

Typically, b_k is held constant for a clock update period (T_{CLK}) resulting in a zero-order hold (ZOH) non return-to-zero (NRZ) output. For a sinusoidal input, the output PSD, displayed in Figure 1.2(b), contains a single tone at the input signal frequency (f_0), noise floor set by N, image replica tones generated by the sampled nature of the signal at $f_{CLK} \pm f_0, 2f_{CLK} \pm f_0$, …, and ZOH sinc response ($|H_0(f)|$). On this plot, the power ratio of signal to largest spur, or spurious-free-dynamic range (SFDR), provides a measure of DAC linearity.

In an ideal case, the maximum SFDR is [4]

$$SFDR\ [dBc] \cong 8.2N + 2.2. \tag{1.2}$$

In a physical DAC, however, the output also drives a capacitive load[5] (C_L) and the cells are implemented with a transistor current source (M_{cs_k}) and switch (M_{sw_k}) (Figure 1.2(a)). I_k is scaled by sizing M_{cs_k}, while M_{sw_k} is sized accordingly to maintain the same voltage bias condition, and hence, achieve identical switching time (st_k) among DAC cells. After b_k switches, V_{out} settles to the desired value according to the output time constant ($\tau_{out} = R_L C_L$). Non-ideal transistor behavior results in output amplitude (ΔV_{out}), time delay (Δt_{out}), duty cycle (Δd_{out}), and nonlinear settling errors as depicted in Figure

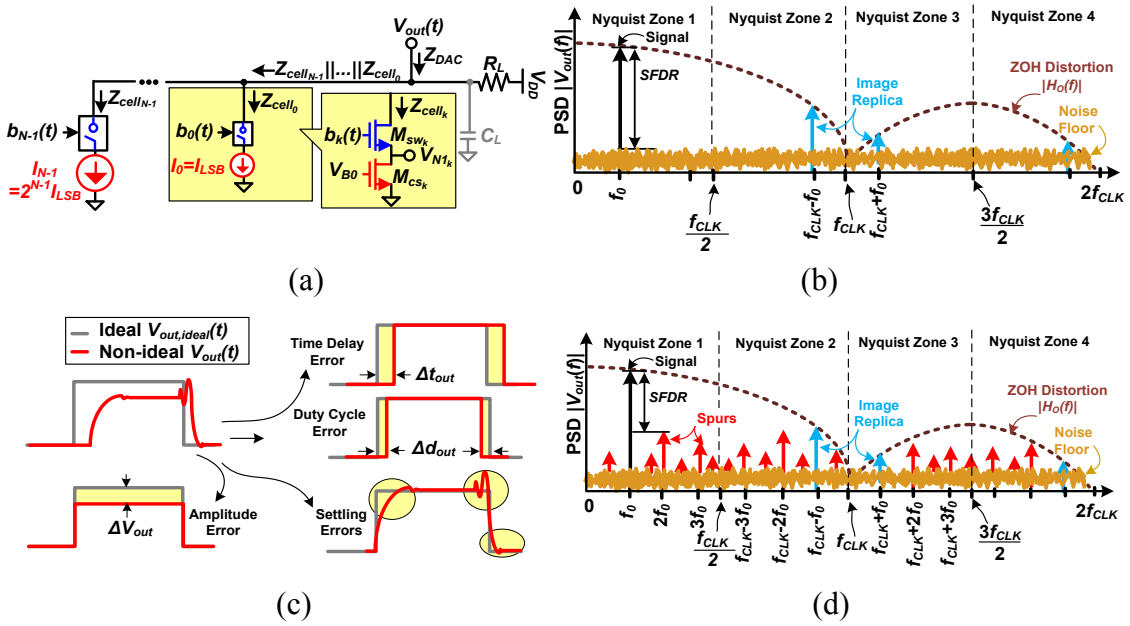


Figure 1.2: N-bit binary current-steering DAC. (a) Simple architecture. (b) ZOH DAC PSD. (c) Ideal vs. non-ideal output pulse decomposed into amplitude and timing errors. (b) ZOH nonlinear DAC PSD.

[5] Capacitance from layout interconnect, pad, package, and board routing.

1.2(c), all of which introduce spurs in the PSD that degrade SFDR (Figure 1.2(d)). These errors can be classified as intrinsic (i) arising from the transistor terminal behavior, random (r) resulting from transistor process mismatch, and systematic (s) attributed to layout effects.

1.1.2. Intrinsic DAC Error Sources

The intrinsic transistor parameters ($g_m, r_{ds}, C_{gs}, C_{gd}, C_{db}, C_{sb}$) have two detrimental effects on the DAC performance. First, they manifest in a finite cell impedance ($Z_{cell_k}(j\omega, V_{out}, b_k)$) which alters the effective DAC output impedance (1.3), resulting in amplitude and RC settling errors.

$$Z_{DAC}(j\omega, V_{out}, b_{0:N-1}) = R_L \| C_L \| Z_{cell_0} \| \dots \| Z_{cell_{N-1}}. \quad (1.3)$$

Second, they allow voltage to couple between V_{out} and the internal nodes of the cell (b_k, V_{N1_k}, V_{B0}), impacting the switching time accuracy and generating feedthrough glitches.

1.1.2.1. Output Voltage Error ($\Delta V_{out,i}$), Induced by Cell Resistance Modulation

At DC, the cell impedance toggles between ON, i.e. $Z_{cell_k}(j\omega = 0, b_k = 1) \sim g_{m,sw_k} r_{ds,sw_k} r_{ds,cs_k}$, and OFF, i.e. $Z_{cell_k}(j\omega = 0, b_k = 0) \sim \infty$[6]. This results in an intrinsic DC output voltage error that degrades low frequency SFDR [5]

$$\Delta V_{out,i} \sim \frac{V_{out} R_L \sum_{k=0}^{N-1} \left(Z_{cell_k}(j\omega=0, b_k) \right)^{-1}}{1 + R_L \sum_{k=0}^{N-1} \left(Z_{cell_k}(j\omega=0, b_k) \right)^{-1}}. \quad (1.4)$$

[6] In a 65nm process: $Z_{cell_k}(j\omega = 0, b_k = 0) \sim 10^3 * Z_{cell_k}(j\omega = 0, b_k = 1)$ and hence is negligible.

Figure 1.3(a) depicts the impact of varying the LSB current and subsequently $Z_{cell_0}(j\omega = 0, b_0 = 1)$[7] on $\Delta V_{out,i}$. It is desired to have a large $Z_{cell_k}(j\omega = 0, b_k = 1)$ such that $\Delta V_{out,i}$ is minimal. However, the size and current scaling requirements for M_{cs_k} and M_{sw_k} call for $Z_{cell,k}(j\omega = 0, b_k = 1) = Z_{cell,0}(j\omega = 0, b_0 = 1)/2^k$, making it challenging to achieve high impedance in the most significant binary cells of moderate- to high-resolution DACs.

1.1.2.2. Output Time Constant Error ($\Delta\tau_{out,i}$), Induced by Cell Drain Capacitance Modulation

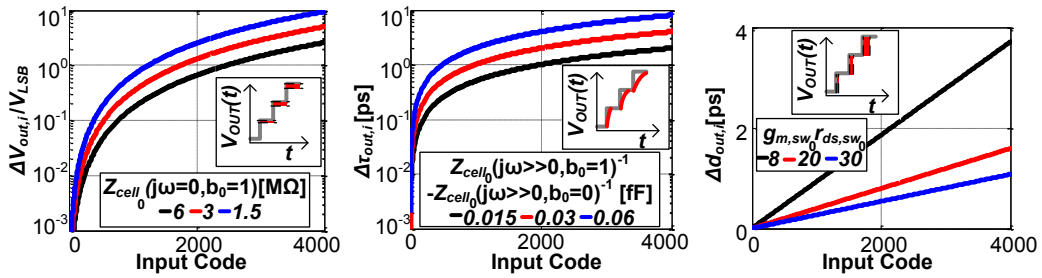
As frequency increases, the transistor capacitance starts to effect $Z_{cell_k}(j\omega \gg 0\,, b_k)$ which toggles between ON, i.e. $Z_{cell_k}(j\omega \gg 0\,, b_k = 1)^{-1} \sim j\omega C_{gd,cs_k}/(g_{m,sw_k} r_{ds,sw_k})$ and OFF, i.e. $Z_{cell_k}(j\omega \gg 0\,, b_k = 0)^{-1} \sim j\omega C_{gd,sw_k}$[8]. This code dependency modulates the DAC output time constant (1.5), inducing intrinsic settling errors that results in a frequency dependent degradation in SFDR [5].

$$\Delta\tau_{out,i} \sim R_L \sum_{k=0}^{N-1} \left(Z_{cell_k}(j\omega \gg 0\,, b_k)\right)^{-1}. \tag{1.5}$$

Figure 1.3(b) plots the resulting $\Delta\tau_{out,i}$ for various values of LSB $Z_{cell_0}(j\omega \gg 0\,, b_k = 1)^{-1} - Z_{cell_0}(j\omega \gg 0\,, b_k = 0)^{-1}$ [9]. It is desired to have nearly equal ON and OFF switch capacitances such that $\Delta\tau_{out,i}$ is minimal. However, the cell scaling mandates $Z_{cell_k}(j\omega \gg 0, b_k = 1)^{-1} = 2^k \cdot Z_{cell,0}(j\omega \gg 0\,, b_0 = 1)^{-1}$, exacerbating $\Delta\tau_{out,i}$ as k increases.

---

[7] Decreases with I_{LSB} =5, 15, 35µA.
[8] $C_{gd,cs_k}/g_{m,sw_k} r_{ds,sw_k} \gg C_{gd,sw_k}$ since $M_{cs,k}$ is much larger than $M_{sw,k}$.
[9] Increases with $W_{cs\,0}/L_{cs_0}$ =1µm/500nm , 2µm/500nm, and 4µm/500nm.



(a) (b) (c)

Figure 1.3: Calculated intrinsic output (a) voltage, (b) time constant, and (c) duty cycle errors vs. input code for a 12-bit DAC operating at f_{CLK} =2GHz using simulated transistor parameter values.

1.1.2.3. Output Duty Cycle Error ($\Delta d_{out,i}$), Induced by Cell Switch Time Modulation

When the cell is ON, finite switch output resistance (r_{ds,sw_k}) allows V_{out} to couple to the source node (V_{N1_k}), modulating the switch bias voltage. This in turn results in code dependent cell switching time ($\Delta st_{k,i}$) that is subject to the extent of voltage feedthrough along with signal b_k's rise/fall time (τ_{b_k}) and voltage swing (ΔV_{b_k}) [6]

$$\Delta st_{k,i}(t) = \frac{V_{out}(t)}{1+g_{m,sw_k} r_{ds,sw_k}} (\tau_{b_k}/\Delta V_{b_k}). \quad (1.6)$$

Note that all the DAC cells have an identical switching time which changes as a function of $V_{out}(t)$[10]. This code-dependency results in a duty cycle error per cell ($\Delta d_k(t)$) that varies with the switching state of b_k (see Table 1.2). This aggregates to an output duty cycle error ($\Delta d_{out,i}$) that results in a frequency dependent degradation in SFDR. Figure 1.3(c) plots $\Delta d_{out,i}$ for different values of $g_{m,sw_0} r_{ds,sw_0}$[11].

[10] $g_{m,sw_k} \propto I_k$, $r_{ds,sw_k} \propto 1/I_k$, and assuming τ_{b_k} and ΔV_{b_k} are the same.
[11] Increases with L_{sw_0} = 50nm, 75nm, and 100nm.

Table 1.2: Duty cycle error per cell k depending on switching state transitions.

$b_k(t - T_{CLK})$	$b_k(t)$	$b_k(t + T_{CLK})$	$\Delta d_k(t)$
X	0	X	0
0	1	0	$\Delta st_{k,i}(t - T_{CLK}) + \Delta st_{k,i}(t)$
0	1	1	$\Delta st_{k,i}(t - T_{CLK})$
1	1	0	$\Delta st_{k,i}(t)$
1	1	1	0

1.1.2.4. Glitch Feedthrough, Coupled through Cell Capacitance

Switch feedthrough glitches propagate to V_{out}, V_{B0}, and V_{N1_k} through M_{cs_k} and M_{sw_k}'s C_{gs}, C_{gd}, C_{db}, and C_{sb}, creating V_{out} settling errors that are proportional to the switching cell size and I_k [7]. Switching glitches are created by several effects:

- When M_{sw_k} turns off, charge is injection and a glitch occurs on V_{out} and V_{N1_k}
- When M_{sw_k} turns off, M_{cs_k} is choked and I_k is discharge through M_{cs_k}'s C_{db} creating a glitch on V_{N1_k} which propagates to V_{out}.
- When the DAC code changes, glitches are created on V_{out} from imperfect synchronization of current. For example if $b_0 = 1, b_1 = 0,$ and $b_2 = 1$ ($I_{out} = I_0 + I_2$) switches to $b_0 = 1, b_1 = 1,$ and $b_2 = 0$ ($I_{out} = I_0 + I_1$) there could be a moment when b_1 and b_2 are simultaneously switched on causing ($I_{out} = I_0 + I_1 + I_2$).
- When a glitch propagates to V_{B0} from V_{N1_k}, a current error glitch occurs on V_{out}.

1.1.3. Mismatch DAC Error Sources

There are two predominant sources of mismatch in the DAC: random mismatch arising from photolithographic process variations and systematic resulting from layout routing.

1.1.3.1.Random Output Voltage ($\Delta V_{out,r}$) and Duty Cycle Errors ($\Delta d_{out,r}$)

Photolithographic manufacturing results in random Gaussian distributed variation $\mathcal{N}(\mu, \sigma^2)$[12] in the transistor gain factor: $\beta = \mu_n C_{ox} W/L$[13] and threshold voltage: V_T. The variation in M_{cs_k} induces random current mismatch ($\Delta I_{k,r}$), while variation in M_{sw_k} results in switching time mismatch ($\Delta st_{k,r}$) [8]. The variance in $\Delta I_{k,r}$ is equal to[14] [8]

$$\frac{\sigma^2(\Delta I_{k,r})}{I_k} = \frac{\sigma^2(\Delta\beta)}{\beta} + \frac{\sigma^2(2\Delta V_T)}{V_{OD}} = \frac{1}{WL}\left(A_\beta^2 + \frac{4A_{V_T}^2}{V_{OD}^2}\right). \quad (1.7)$$

where A_β, A_{VT} are process dependent parameters. The variance in $\Delta st_{k,r}$ is primarily generated from the transistors V_T mismatch [7]

$$\sigma^2(\Delta st) = A_{V_T}^2 C_{ox}^2 \frac{WL}{I^2}. \quad (1.8)$$

Note that, the spread of these mismatches is dependent on relevant transistor area. Figure 1.4(a-b) depicts the spread of $\Delta I_{k,r}$ and corresponding accumulated random output voltage error ($\Delta V_{out,r}$) for different sizes (xW_{min}/L_{min}) [15] of the LSB M_{cs_0} transistor. Alternatively, the spread of $\Delta st_{k,r}$ and equivalent duty cycle error ($\Delta d_{out,r}$) are plotted in Figure 1.4(c-d) for different sizes of the LSB M_{sw_0} transistor. As shown, the spread in $\Delta I_{k,r}$[16] [8] reduces while $\Delta st_{k,r}$[17] [7] increases with larger LSB transistor sizes and both follow a sublinear relationship with bit order (k). Note also that the maximum amount of deviation in $\Delta V_{out,r}$ and $\Delta d_{out,r}$ are dependent on the distribution of cell mismatch values

---

[12] Random variable with mean (μ) and variance (σ^2).
[13] μ_n is the transistor mobility, C_{ox} is the transistor oxide capacitance, W is the transistor width, and L is the transistor length.
[14] $V_{OD} = (V_{GS} - V_T)$ is the overdrive voltage of the transistor, and V_{GS} is the gate-to-source bias voltage.
[15] W_{min}/L_{min} =100nm/50nm in a 65nm process.
[16] $\Delta I_{k,r} \propto I_k/\sqrt{W_{cs_k} L_{cs_k}}$.
[17] $\Delta st_{k,r} \propto \sqrt{W_{sw_k} L_{sw_k}}/I_k$.`

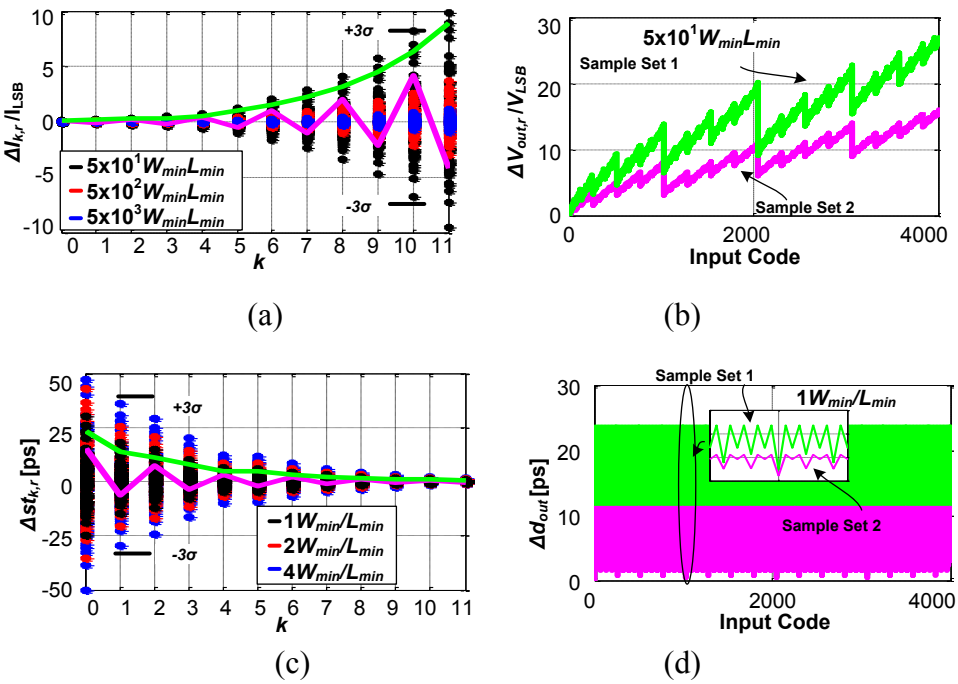


Figure 1.4: Simulated random mismatch for a 12-bit DAC with I_{LSB} =5µA. (a) Current source mismatch vs. cell index (k) for selected sizes of the LSB M_{cs_0} transistor and (b) corresponding output voltage error vs. input code for two unique sample sets with from (a) with $M_{cs_0} = 50 \times W_{min}L_{min}$. (c) Switching time mismatch vs. k for selected sizes of the LSB M_{sw_0} transistor and (d) corresponding output duty cycle error for two unique sample sets from (c) with $M_{sw_0} = W_{min}/L_{min}$. Note the error in (c) and (d) is a function of the sample set. When the sequential cell mismatch is correlated (sample set 1) the error accumulates faster and is larger than if it is uncorrelated (sample set 2).

[9]. For example, in sample set 1 the mismatch values in adjacent cells are strongly correlated and accumulate into a large error, while the mismatch in sample set 2 cancels between adjacent cells producing a smaller error. As a result, the spread of cell mismatches for a given sample set will determine the DAC SFDR performance.

1.1.3.2. Systematic Output Voltage ($\Delta V_{out,s}$) and Time Constant Delays ($\Delta \tau_{out,s}$)

Routing metal used during DAC layout have an associated resistance (R_M) and capacitance (C_M) which induce additional mismatch. Voltage drops across R_M create deviations in the ground supply for each cell inducing systematic current mismatch ($\Delta I_{k,s}$) in M_{cs_k} [10], while data path routing line $R_M C_M$ mismatch result in systematic output time constant delays ($\Delta \tau_{out,s}$) that have an increasing impact with frequency [11].

1.2. Common Current-Steering DAC Architecture

While simple in nature, the binary DAC with two transistor cell does not provide optimal performance. A widely used architecture is pictured in Figure 1.5 and adopts the following techniques to achieve a high performance baseline DAC:

- The DAC cell is implemented differentially with the switch pair ($M_{3,4}$) to reduce glitch feedthrough errors, increase common mode rejection, and reduced even order distortion.
- The DAC cell utilizes cascoding transistors on the current source (M_2) and switch pair ($M_{5,6}$) to increase cell ON resistance, thereby reducing output voltage ($\Delta V_{out,i}$) and output duty cycle errors ($\Delta d_{out,i}$). The cascode transistors also help shield V_{out} and V_{N1_k} from glitch feedthrough [12].
- A technique known as segmentation is used to decode the top m-bits ($b_{N-1}: b_{N-m}$) to a thermometer code ($th_j, j = 0,1, \dots 2^{m-1}$) which controls 2^m unary cells, each having $I_j = 2^{N-m} I_{LSB}$, to provide the DC output voltage (1.8). Two main reasons for

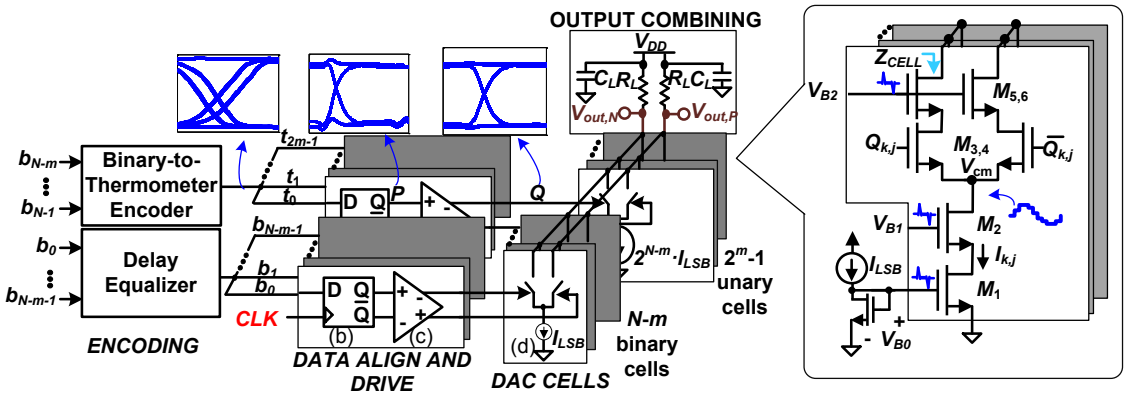


Figure 1.5: Modified DAC architecture with segmentation, retiming flip-flop, driver, and cascode transistors.

segmenting are to ensure DAC monotonicity and reduce large glitches created by the binary mid-code transition, while still marinating a reasonable DAC area [13].

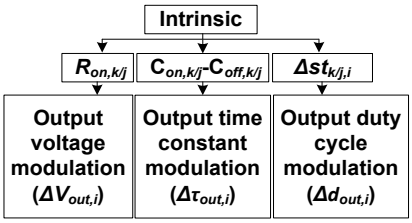
$$V_{out} = R_L \cdot I_{LSB} \left(\sum_{k=0}^{k=N-m-1} b_k \cdot 2^k + \sum_{j=0}^{j=2^m-1} t_j \cdot 2^{N-m} \right) \quad (1.9)$$

- The binary data (b_k) and thermometer encoded data (th_j) are aligned using a retiming flip-flop clocked with a global clock signal (CLK) close to the cell, thereby removing data delay differences up to the DAC buffer cell [10] [13]. Data (b_k, th_j) is captured once CLK crosses the clocked transistor threshold, generating a differential signal ($P, \bar{P}$) after a $CLK - to - Q$ delay. The flip-flop also removes data-dependent jitter occurring from the logic delay differences in the thermometer decoder.
- A differential driver circuit is used to buffer the data ($P, \bar{P}$), removing clock feedthrough, and providing $M_{3,4}$ with fast transition time (τ_{b_k}), reduced voltage swing (ΔV_{b_k}), and optimal crossover point (V_{cross}), which reduced glitch feedthrough errors and output duty cycle errors ($\Delta d_{out,i}$) [10]. For instance, when V_{cross} is set below the V_T of $M_{3,4}$, both switch pairs are simultaneously turned off. This causes M_1 to turn

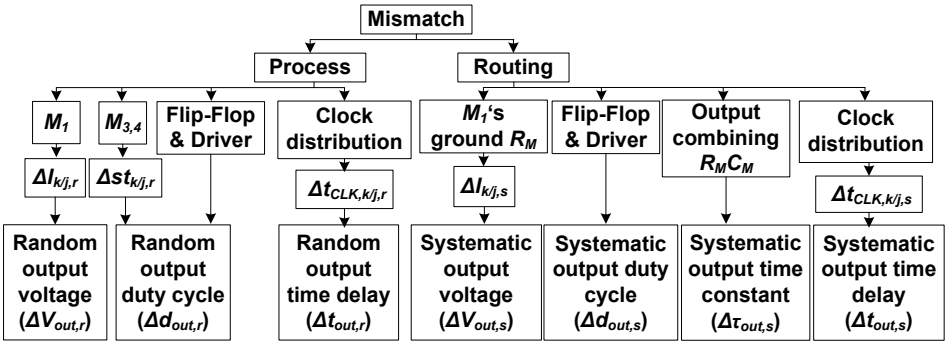
off, generating a large negative glitch on the differential common source node (V_{cm_k}) and requiring time for the current source to turn back on. If the V_{cross} is set above the V_T, a positive glitch at V_{cm_k} occurs.

Although the addition of retiming flip-flops and drivers mitigates many timing errors in the data path, they also contribute their own time delay and duty cycle errors. These errors can be reduced by choosing the appropriate architecture and following careful design practices. For example, multiple retiming flip-flop stages can be used to reduce data dependent jitter (DDJ) in the latch element [14]. Current mode logic (CML) latches and buffers are often utilized to achieve faster switching speed while providing low amplitude voltages that reduce glitch feedthrough [15]. Although CML draws more current, it induces less switching supply noise than that of full swing CMOS circuits [16] [17]. However, any mismatch in the clock distribution tree or CML circuit will add to the output delay errors ($\Delta t_{out,r/s}$) [18] and output duty cycle errors ($\Delta d_{out,r/s}$), respectively [19]. Figure 1.6 summarizes the output errors and parameters created by the k-binary and j-thermometer cells.

In addition to the aforementioned circuit techniques, careful layout design should be followed. For example, larger line widths are used to reduce R_M [20]. Tree structures, which provide equal routing impedance and electrical length, reduce the spread of systematic mismatch [14] [21]. Additionally, common-centroid floorplans [22] and pre-fabrication switching schemes [23], [9] spatially average systematic errors.



(a)



(b)

Figure 1.6: Summary of (a) intrinsic errors and (b) mismatch errors.

1.3. Use of CMOS, Bipolar, and III-V Device Materials

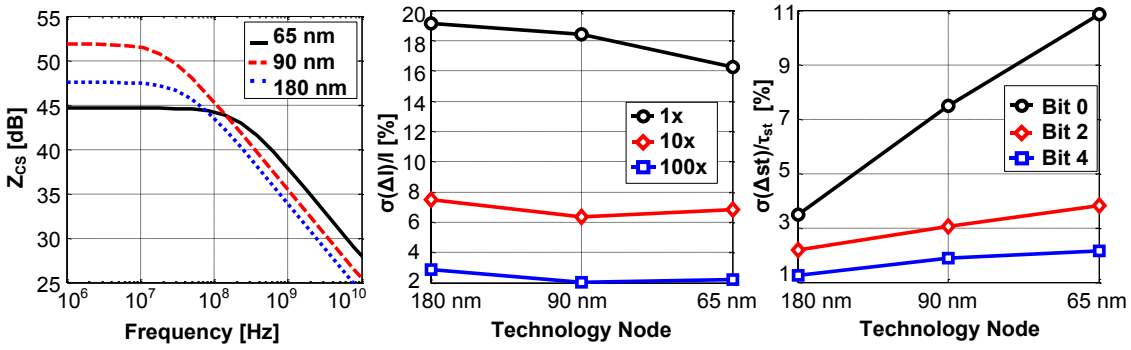
CMOS scaling has benefited designs by providing increased transistor f_T, reduced device capacitance, reduced power, and large scale integration. However, when scaling into the nanometer range, transistors have increased process mismatch [24] and reduced output impedance [25] that present a challenge in maintaining high yield in current-steering DACs. Emerging device technologies, such as bipolar junction transistors (BJTs) and more recently DARPA's COSMOS and DAHI program integration of III-V with CMOS silicon, allow high performance DACs to be designed. These technologies offer much higher $f_T > 350$ GHz and output impedance, but suffer from increased switching power consumption, limited device yield, and self-heating. Often DAC designs opt for a

mixed design integrating CMOS and bipolar or III-V. This integration capability lends itself well to allowing designers to optimize the transistor for the desired function [26].

1.3.1. CMOS Technologies

To illustrate the benefits and limitations of CMOS, several simlations are performed across 180nm, 90nm, and 65nm CMOS process technologies. The transistor impedance of an NMOS device[18] is simulated across process technology and shown in Figure 1.7(a). While the results indicate a slight improvement in DC output impedance from 180nm to the 90nm process, there is a significant degradation in DC output impedance when moving from 90nm to the 65nm process. However, the impedance at high frequencies is seen to improve. This is attributed to the fact that a smaller device (therefore smaller drain capacitance) is used to draw the same current, as process scales. Current and timing mismatch are obtained next using a 50-point Monte-Carlo simulation. Figure 1.7(b) plots the current mismatch ($\sigma\Delta I/I$) of identical area NMOS transistors[19]. The transistor starts with a 1x area device and scales up to 10x and 100x[20]. As shown, technology scaling has little or no effect on the overall current mismatch error. The only room for improvement are to increase $V_{GS}-V_T$, or the area of the transistor. While increasing $V_{GS}-V_T$ is limited by headroom requirements, increasing the transistor size is limited by area constraints. Figure 1.7(c) plots the switching time mismatch with respect to the

[18] $V_{GS}-V_T$=100 mV, L=1μm, and W is scaled to provide a current I=70μA.
[19] $V_{GS}-V_T$=100 mV.
[20] The transistor area of 1x means the same area transistor for 65nm, 90nm, and 180nm. Each process uses a minimum L, while W is scaled to provide the same area.



(a) (b) (c)

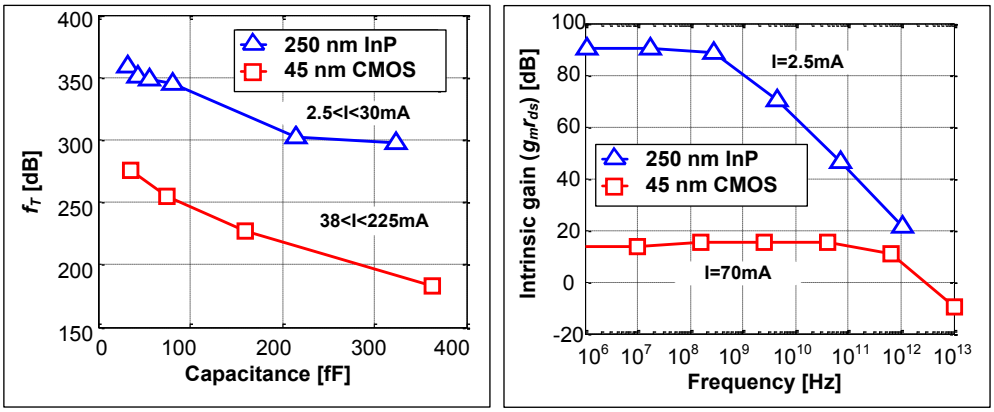
Figure 1.7: CMOS output impedance and mismatch. (a) DAC current source impedance across technology (b) Current mismatches and (c) switching time mismatch across CMOS process technologies.

switching time constant ($\sigma\Delta st/\tau_{out}$) for different binary cells (k=0, 2, and 4)[21]. As shown, relative timing accuracy diminishes for scaling CMOS devices.

1.3.2. BiCMOS and III-V Technologies

In highly analog circuits such as DACs, higher bias currents are needed to obtain faster switching speeds (i.e. higher f_T). Nanometer CMOS transistors are shown to have a high f_T in some situations reaching that of III-Vs but require multiple interconnected devices to handle large current densities resulting in additional capacitance and reduced usable device f_T. As shown in Figure 1.8(a-b), InP offers much higher f_T and intrinsic gain ($g_m r_{ds}$) with smaller bias currents relative to CMOS. Similarly, bipolar transistors offer higher f_T and intrinsic gain than CMOS. Therefore, using bipolar or III-V devices as the cascode and switch pair yields higher output impedance and faster switching. CMOS transistors are favored for the current sources since current matching and scaling is

[21] For binary cell k=0, a minimum W and L are used with I_{LSB} = 2μA.



(a) (b)

Figure 1.8: InP and CMOS (a) f_T and (b) gain vs. frequency.

directly achievable [27]. Silicon technologies have benefited from the decades of commercial resources in developing the infrastructures, teams, models, and design and test methodologies. Silicon also offers very high-levels of integration with transistor counts in the billions and available scaling based on device size and bias values. Capability for large scale integration of III-V technologies is a recent development and is well behind CMOS in scaling. For example, only discrete device size of InP transistors are available in PDKs. Therefore to provide the correct current scaling between the binary DAC cells a designer would have to compensate for the lack of scaling by relying on a hybrid architecture utilizing an $R2R$ resistor ladder in conjunction with current steering cells to realize binary-weighted currents [28].

Chapter 2: Statistical Modeling for Current-Steering DAC

The effects of intrinsic and mismatch error sources on DAC SFDR is investigate in this chapter. While intrinsic effects are uncovered during nominal simulations, mismatch errors require several steps to discover. For instance, process mismatch errors create a statistical variance (σ^2_{SFDR}) in SFDR, whose mean (μ_{SFDR}) deteriorates as operation frequency increases, limiting device yield (Figure 2.1). Monte Carlo (MC) simulations are used to determine the process mismatch induced loss in DAC performance, requiring 100's of runs per frequency point to accurately model yield. Often, the DAC design is iterated several times until the MC simulated yield is acceptable. To uncover layout mismatch, the extracted layout design must be simulated. This often leads to several redesigns in layout until optimal performance is reached. MC and post layout extraction

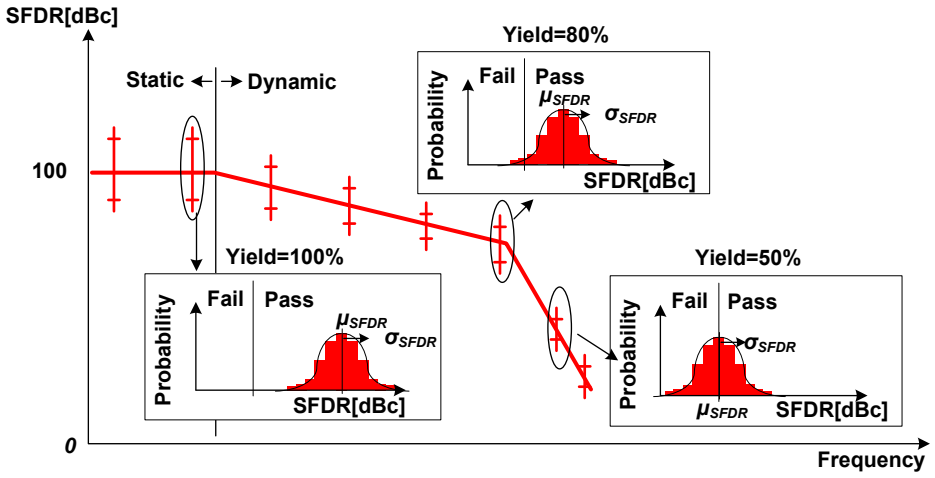


Figure 2.1: Nonlinearity in the DACs conversion process and its effect on SFDR yield.

demand rigorous DAC design and simulation testing to meet SFDR yield, which has motivated research focused on characterizing the current-steering DAC as well as simulation models to predict its behavior, shortening design and simulation testing time.

Equation based analyses have been developed to predict theoretical DC static (i.e. integral nonlinearity (INL)) yield using the current source mismatch variance ($\sigma\Delta I_{k,j}$) as the variable. In [29], the probability of current source mismatch is used to determine the INL yield of a binary DAC. In [30], a Brownian-Bridge-based analysis is used to calculate binary and thermometer DAC INL yield. [31], builds on these ideas and provides the most accurate prediction of INL yield for an arbitrary segmented DAC using an empirically-fitted model.

While these equations predict low frequency yield, they fail to predict yield limited by dynamic amplitude and timing errors. A rigorous attempt at equating DAC distortion to each source of error is presented in [7], including derivation of the average signal-to-distortion ratio originating from output time constant errors ($\Delta\tau_{out,i}$) and random delay mismatch ($\Delta t_{out,r}$). However, it fails to provide the SFDR spread or yield outcome.

In contrast, simulation based models abstract DAC components into mathematical and behavioral descriptions to capture dynamic performance while reducing simulation time. Mathematical HDL blocks are used in [32] to model the DAC INL, output glitches, and settling time. The INL performance of the model is validated against a 10-bit DAC 100-point MC simulation. Wavelet theory is used in [33] to generate a series of filtered voltage pulses which capture the DAC DC voltage steps, glitches, and settling performance. It is validated with an 8-bit DAC, comparing INL and SFDR at 10 kHz. A

series of 1-bit behavioral DAC models in [34] are filtered through a nonlinear equation to simulate amplitude mismatch and subsequently low pass filtered to simulate settling errors. An ideal differential switch and RC equivalent current source are used to model the DAC cells in [35] with delay mismatches inserted between the switches to model random delay mismatch ($\Delta t_{out,r}$). The model SFDR without delay mismatch is confirmed with a 6-bit DAC operating at 400 MHz and 10-bit DAC operating at 200 MHz. Then delay mismatches are generated from a generic set of 23 random timing distributions to show the effect on SFDR spread, but results are not verified with a MC simulation.

Due to these shortcomings, a new simulation model for the current-steering DAC architecture is developed here that simultaneously predicts SFDR yield in the GHz operating range and provides simulation speedup by abstracting non-critical circuits to behavioral models. The model captures process variation induced DAC cell mismatch via current, timing, and voltage offset blocks at specific circuit nodes. A statistical modeling methodology that accounts for process mismatch is developed which predicts the SFDR yield in significantly less (on the order of DAC bits- N) simulation runs by populating the offset blocks with a set of deterministic mismatch distributions. In total this modeling methodology allows a simulation speedup of three orders of magnitude.

2.1. The Proposed DAC Model

A new DAC model is proposed that accurately captures intrinsic and mismatch performance, without requiring 100's of MC simulations. The proposed model accomplishes this through a hybrid modeling approach. Noncritical transistors are

modeled with simulation efficient Verilog-A blocks, while transistor mismatches are captured with parameterizable offset blocks fed with a deterministic distribution that covers the SFDR spread. The model synthesis and simulation is automated using MATLAB and Cadence ocean scripts to reduce setup time. See Appendix A for the MATLAB and Verilog-A codes.

2.1.1. Hybrid DAC model

While simulation of a full transistor DAC design is the most accurate, it requires the greatest amount of time to compute. Using a Verilog-A behavioral DAC model is fast to simulate but inaccurate. The proposed model combines transistor and Verilog-A blocks to capture the DAC inherent and transistor mismatch performance accurately and in the most time efficient manner. Identifying which blocks can be modeled involves examining how the non-idealities propagate throughout the DAC data chain.

2.1.1.1. Capturing Intrinsic Performance

The binary-to-thermometer decoder generates data dependent jitter that is removed by the following retiming flip-flop, allowing it to be represented with a Verilog-A decoder without losing simulation accuracy. The retiming flip-flop delays the data to the driver by the $CLK - to - Q$ delay (t_{CLK}) and generates clock feedthrough that is minimized by the following driver, allowing it to be represented with a Verilog-A flip-flop with a delay of t_{CLK}. The driver generates data with a specific voltage swing ($\Delta Q_{k,j}$), transition time ($\tau_{Q_{k,j}}$), and cross over point (V_{cross}), allowing it to be modeled with a Verilog-A block with controls of $\Delta Q_{k,j}$, $\tau_{Q_{k,j}}$, and V_{cross}. The Verilog-A retiming flip-flop and driver are

combined into one Verilog-A retiming logic block with several input parameters. The DAC cell consists of operating region dependent impedance varying transistors with complex switching behavior. The switch pair ($M_{3,4}$) and cascode ($M_{5,6}$) are kept in the model as BSIM transistors to accurately account for impedance variation and output dependent sampling delay. Although the cascoded current source can be modeled with an equivalent RC circuit [35], it only offers a speedup of 1.14x compared with a transistor one and requires measurement time to extract the R and C of each cell. It also fails to capture the dynamic switching glitches which couple to the current source bias node (V_{B0}). For these reasons the cascoded current source is kept as BSIM transistors in the model.

2.1.1.2. Capturing Mismatch Performance

Mismatches arise in the clock distribution, retiming flip-flop, driver, and DAC cell. Consider the accumulate mismatch effects in the data path coming from the clock distribution, retiming flip-flop, and driver. In this case, mismatch in the final signal ($Q_{k,j}$) controlling the switch pair ($M_{3,4_{k,j}}$) is what matters. The $Q_{k,j}$ mismatches that will affect the DAC SFDR are the clock delay error ($\Delta t_{CLK_{k,j}}$) and the duty cycle error ($\Delta d_{Q_{k,j}}$). Now consider the mismatches in the DAC cell. There is mismatch in current ($\Delta I_{k,j}$) and mismatch in $M_{3,4_{k,j}}$ sampling time ($st_{k,j}$) which causes a duty cycle error ($\Delta d_{M_{3,4,k,j}}$). The proposed model includes these mismatches as Verilog-A offsets, as displayed in Figure 2.2(a). The delay error ($\Delta t_{CLK_{k,j}}$) and duty cycle error ($\Delta d_{k,j} = \Delta d_{Q_{k,j}} + \Delta d_{M_{3,4,k,j}}$) are captured in the Verilog-A retiming logic output, while the current mismatch ($\Delta I_{k,j}$) is

modeled by adding/subtracting current from $M_{1_{k,j}}$'s drain node. The retiming Verilog-A response, plotted in Figure 2.2(b), depicts how the data arrives to $M_{3,4_{k,j}}$ with and without the timing and duty cycle mismatch. The mismatch variables (2.1) per cell (k,j) are set individually to any desired distribution, as displayed in Figure 2.2(c) for the unary cell array.

$$PV_{k,j} = (\Delta t_{CLK_{k,j}}, \Delta d_{k,j}, \Delta I_{k,j}) \tag{2.1}$$

In most DAC segmentations, the SFDR is dominated by the unary cells. Therefore the modeling approach focuses on modeling the unary cell mismatches. SFDR yield can be predicted by populating the unary offset blocks with a deterministic set of mismatch values which anticipate the majority of SFDR scenarios. The modeling methodology

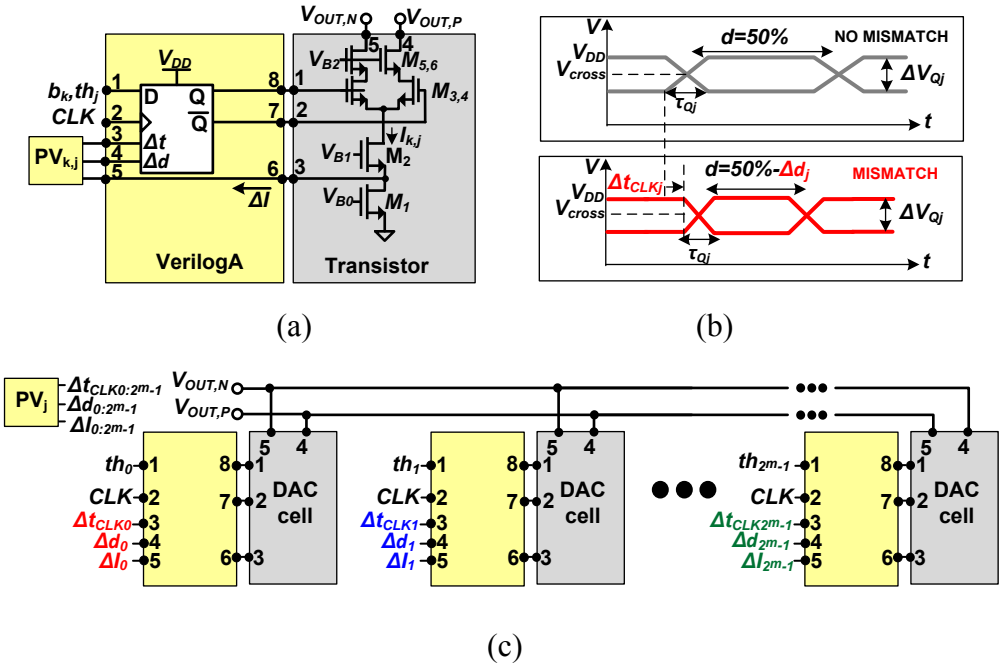


Figure 2.2: Proposed hybrid DAC cell model. (a) Single cell. (b) Retiming Verilog-A code transient response (c) Arrayed unary cells.

which accurately predicts the statistical SFDR spread and yield is explained next.

2.1.2. Rapid Mismatch Modeling Method

The proposed mismatch modeling attains a sampled set of SFDR values from the SFDR PDF to cover the $6-\sigma$ SFDR spread where 99.7% of the SFDR values lye (Figure 2.3). The SFDR is bound by a worst case and best case mismatch distribution related to the correlation of mismatches in sequentially switching cells. In a worst case SFDR scenario, neighboring cells have a high correlation in mismatch values resulting in a linear error distribution [31], accounting for <0.15% of SFDR cases. In a best case SFDR the accumulated error is cancelled to the maximum [36] [37], accounting for <0.15% of SFDR values. In between these two scenarios, there exist a set of distributions that provide the $6-\sigma$ SFDR spread. The procedure for determining the distributions involve measuring the PV_j (2.1) σ's and ordering the mismatch per cell to provide a desired mismatch distribution. This procedure is outlined in Figure 2.4 for 8 unary cell mismatch values ($m = 3$) and described below.

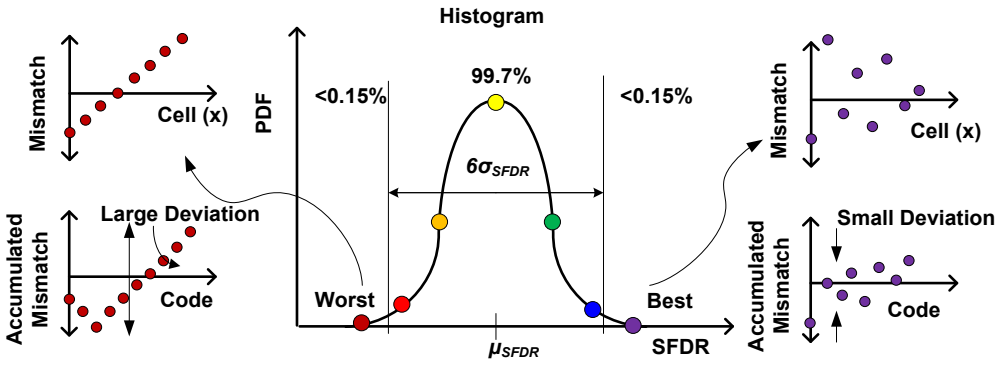


Figure 2.3: Sampling theory for estimating SFDR

① The PV_j σ's are extracted from measured data by running a 200-point MC simulation on a single full transistor implementation of a DAC unary cell including the retiming flip-flop and driver. In particular, delay ($\sigma\Delta t_{CLK_j}$) and duty cycle mismatch ($\sigma\Delta d_j$) is measured at the output node, while current mismatch ($\sigma\Delta I_j$) is measured at the switch pair common source node. Since each unary cell consist of the same transistor properties and biasing conditions, they will statistically have the same mismatch properties. Binary cell mismatch can be modeled in a similar manner, but excluded in this study due to their small contribution to the overall DAC linearity. This 200-point MC run takes approximately 1/10th the simulation time compared with a 1-point full MC DAC run.

② The σ values are used to create a random Gaussian distribution in MATLAB.

③ The random number set is ordered least to greatest, simulating a worst case SFDR.

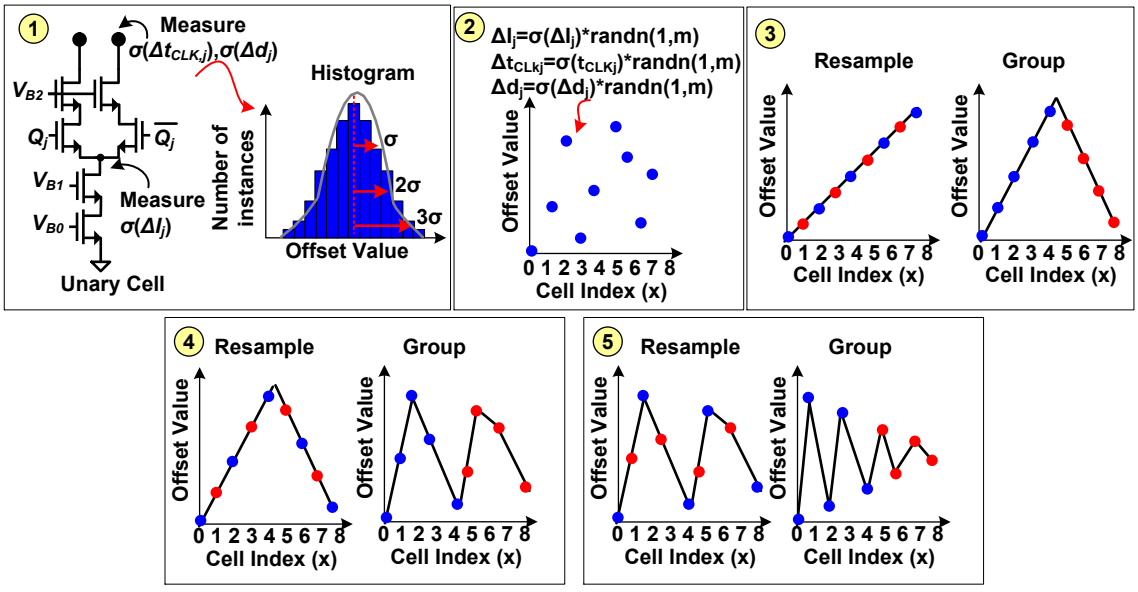


Figure 2.4: Capture of process mismatches in the DAC and mismatch reordering process.

This scenario is unlikely to occur within the $6-\sigma$ spread. What is more probable is a triangular distribution, formed by sampling every other point in the linear distribution and grouping the odd samples followed by the even samples.

④-⑤ The resample and group process repeats until the mismatch in one cell is close in amplitude and opposite to the mismatch in the following switched cell.

In this example, 3 distinct error distributions for $\Delta t_{Q_{k,j}}$, $\Delta d_{k,j}$, and $\Delta I_{k,j}$ exist. The mismatch values are plugged into the offset blocks resulting in the unique delay mismatch ($t_{CLK_j} + \Delta t_{CLK_j}$), duty cycle mismatch ($d_j + \Delta d_j$), and current mismatch ($I_j + \Delta I_j$). Each one of the 3 distribution profiles must be simulated in separate runs to model an accurate SFDR variance, requiring 3 simulation runs total. From these 3 runs the max ($SFDR_{max}$), mean ($SFDR_{mean}$), and min ($SFDR_{min}$) SFDR are calculated. The $\sigma(SFDR)$ is determined assuming a $6-\sigma$ spread from the $SFDR_{max}$ to $SFDR_{min}$ where 99.7% of the points fall into

$$\sigma(SFDR) = \frac{SFDR_{max} - SFDR_{min}}{6} \tag{2.2}$$

2.1.3. Model Synthesis and Simulation

The proposed model utilizes the processing capabilities of MATLAB and the simulation capabilities of Cadence to automate the synthesis and simulation. As outlined in Figure 2.5, MATLAB is used to generate the model netlist given the number of bits (N), segmentation number (m) and transistor sizes [38]. The netlist is then imported into the Cadence environment, liked to a specified PDK, and used to generate a hierarchical schematic. The data properties for the Verilog-A retiming code ($t_{CLK}, \Delta Q_{k,j}, \tau_{Q_{k,j}}, V_{cross}$)

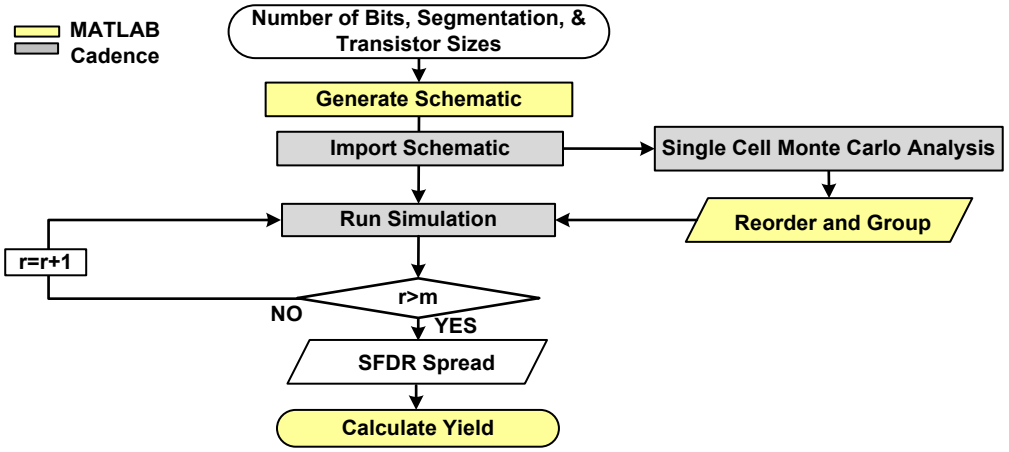


Figure 2.5: Proposed model synthesis and simulation using MATLAB and Cadence

are set to their specified value. The mismatch values (2.1) are modeled as parameterizable blocks whose value is read from a mismatch distribution file generated in MATLAB. A test bench is created in Cadence's analog design environment (ADE) and a simulation is ran, each time selecting the offset distribution file and recording the SFDR, which is subsequently used in MATLAB to calculate the DAC yield.

2.2. Experimental Results

A 10-bit DAC segmented with 6-thermometer bits is implemented in a 65nm CMOS process and used to verify the model. Full transistor simulations, including the retiming flip-flop and driver, are compared with the proposed model and also with the RC equivalent model developed in [35]. The results are shown in Figure 2.6(a)-(c) for f_{CLK}= 500MHz, 2GHz, and 4GHz. Although the RC model reduces simulation time by ~40x that of a full transistor design, the accuracy of the model degrades as the sampling frequency increases (f_{CLK}>2GHz) with input frequencies (f_0>200MHz) due to the use of

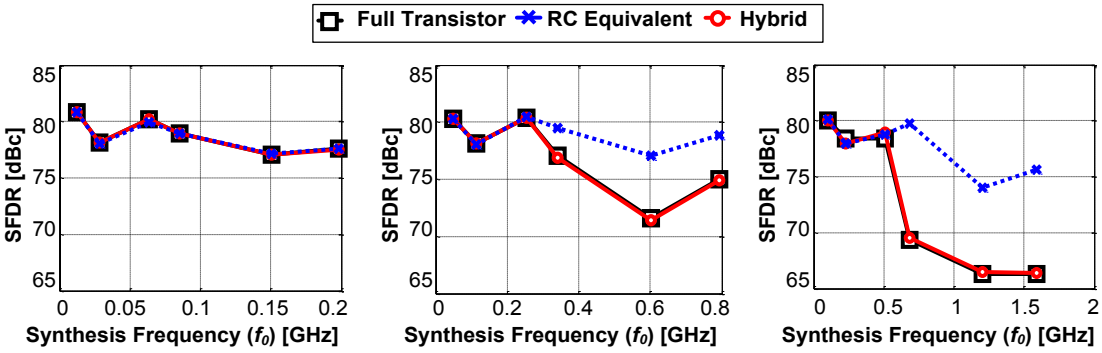


Figure 2.6: Comparison between full transistor model, purely behavioral model, and proposed hybrid model DAC SFDR across operating frequencies for (a) f_{clk} =500MHz (b) f_{clk} =2GHz and (c) f_{clk} =4GHz

ideal switches. The proposed hybrid model is accurate within 1 dB SFDR across all operating frequencies, while providing a speedup of ~10x that of a full transistor design.

The DAC under investigation is found to have the following thermometer mismatch values $(\sigma\Delta t_{CLK_j}) = 1.6$ps, $(\sigma\Delta d_j) = 1.5$ps and $(\sigma\Delta I_j) = 590$nA. In this example a total of 6 distributions exist and must be simulated in 6 separate runs. From these 6 runs the $SFDR_{mean}$ and $6-\sigma(SFDR)$ spread are calculated and plotted in Figure 2.7 along with the measured values from a 200-point full transistor Monte-Carlo simulation. The data is broken up into 3 cases to verify the model accurately predicts the separate and combined effect of amplitude and timing mismatch. The first simulation includes mismatches on the current source transistor only and corresponding current offset in the model, Figure 2.7(a). The next simulation considers the timing mismatches only (Figure 2.7(b)), while last simulation, Figure 2.7(c), is a full MC run with all mismatches included in the DAC. Figure 2.7(d), shows the achievable yield for an SFDR>67dB for the measured and modeled DAC. By choosing the appropriate 6 distributions, the model is accurate to 1 dB, reducing simulation time by ~330x per frequency point.

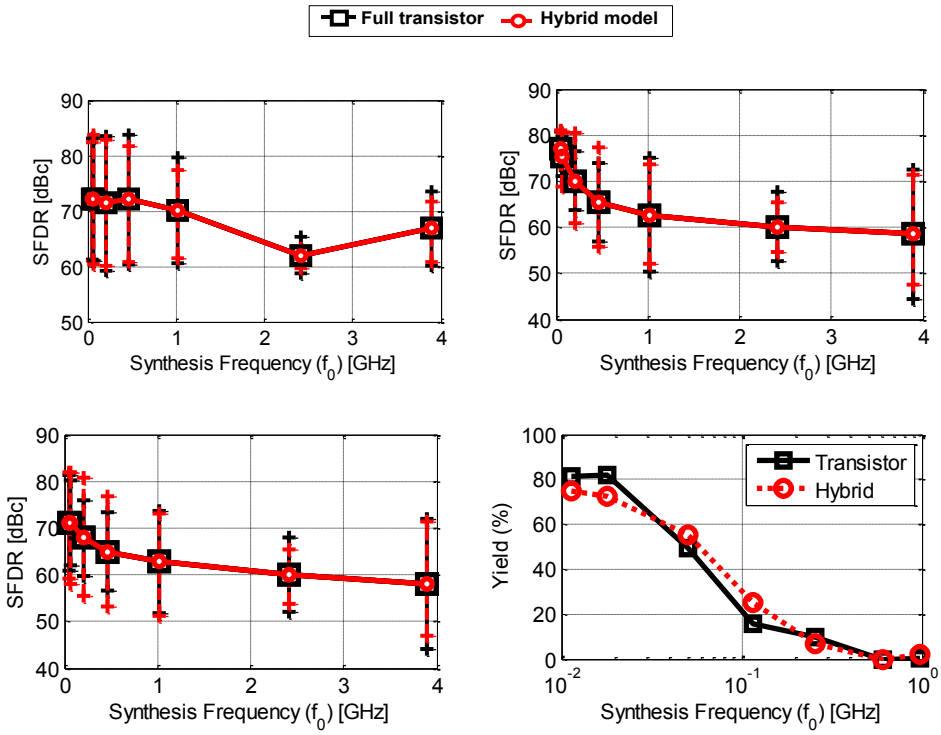


Figure 2.7: Simulated DAC full transistor and hybrid model. SFDR vs. frequency with (a) current mismatch, (b) timing mismatch (c) all mismatches. (d) Yield vs. frequency for SFDR $\geq$ 67.8 dB with all mismatches.

2.3. Conclusion and Model Uses

A new modeling methodology that accurately captures DAC performance at low and high operating frequencies was developed. The model generation and simulation is automated using MATLAB and Cadence to provide rapid analyses. The model also accounts for process mismatches by including timing and current offsets blocks. In general, this model can be useful in studying different DAC architectures and effects of statistical mismatch and systematic offset distributions. It can also aid in the study and development of novel calibration techniques. In the following chapter compensation and calibration techniques are discussed and compared using a baseline 12-bit DAC (Figure

1.5) segmented with 6-thermometer designed in a 65nm process operating at f_{CLK} =2GHz. The hybrid model is used to capture its intrinsic and mismatch performance. Ignoring systematic effects induced by layout, the resultant SFDR is plotted in Figure 2.8 and exhibits the classical resolution versus speed tradeoff [10] [20] [39] [40] [41]. The sources of SFDR degradation are as follows:

① Random output voltage errors ($\Delta V_{out,r}$), induced by current source mismatch (ΔI_{k_r})[22], result in flat 3rd order harmonic distortion. The simulated -3σ SFDR is ~70 dBc[23] for an M_1 LSB area ($W_{1,k=0}L_{1,k=0}$ = 12μm×8μm= 19,000$W_{min}L_{min}$). Note the total current source area for a 12-bit DAC using this LSB size would be ~0.4mm^2. This area grows exponentially for every 3dB increase in SFDR. For example, to meet a 90dBc SFDR, $W_{1,k=0}L_{1,k=0} = 1 \times 10^6 W_{min}L_{min}$, and the total current source area for a 12-bit DAC would be prohibitive at ~20mm^2.

② Cell resistance modulation generates intrinsic output voltage errors ($\Delta V_{out,i}$)[24] that have little impact on SFDR due to the use of cascoding, limiting SFDR~90dBc with flat 3rd order harmonic distortion.

③ Random output delay errors ($\Delta t_{out,r}$), induced by clock mismatch ($\Delta t_{CLK,j,r}$)[25], result in 2nd order harmonic distortion with a -20dB/decade roll off. Note that a 1-σ clock mismatch of 1ps is enough to limit $-3\sigma(SFDR)$ ≈55dBc at 500MHz.

---

[22] $\sigma(\Delta I_{j,r}/I_{LSB}) = 4\%$ obtained from Monte Carlo simulation.

[23] Applying a linear triangular error distribution represents the $-3\sigma SFDR$ within ±1dB when compared with a 200-point Monte Carlo simulation.

[24] $Z_{cell_0}(j\omega = 0, b_0 = 1)$ =180MΩ. Simulated with all mismatch errors omitted.

[25] Mainly contributed by mismatch in the clock buffers where $\sigma(\Delta t_{CLKj,r})$ =1ps is obtained from Monte Carlo simulation.

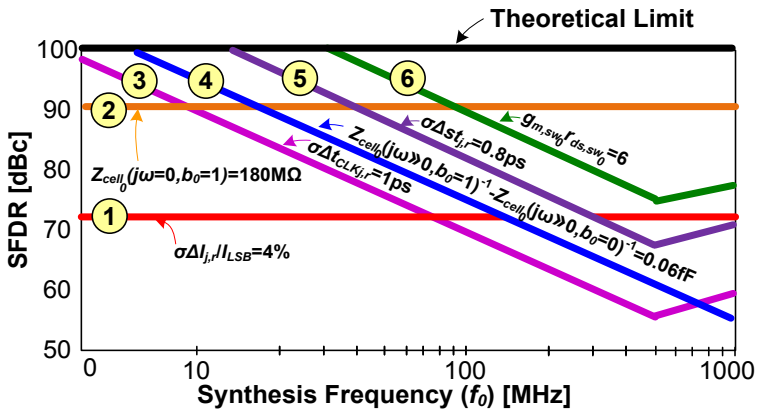


Figure 2.8: Impact of various intrinsic and mismatch errors on SFDR across frequency for the 12-bit baseline DAC operating at $f_{CLK} = 2$GHz.

④ Cell capacitance modulation generates intrinsic time constant errors $(\Delta\tau_{out,i})$[26], resulting in 3rd order harmonic distortion which falls off at -20 dB/decade.

⑤ Random duty cycle errors $(\Delta d_{out,r})$[27], induced by switch time mismatch $(\Delta st_{j,r})$, results in 2nd order harmonic distortion that rolls off at -20dB/decade.

⑥ Intrinsic duty cycle errors $(\Delta d_{out,i})$[28], attributed to switch time modulation, result in 2nd order harmonic distortion that falls off at -20dB/decade.

Note that the reversal in SFDR at $f_{CLK}/4$ is expected since the second order distortion terms alias to lower frequencies and experiences greater attenuation [11].

Overall, the modified DAC architecture (Figure 1.5) provides a solid foundation for high-performance. However, several intrinsic and mismatch errors degrade SFDR well below the theoretical limit of 100dBc. Moreover, systematic layout mismatch will add to

[26] Simulated with all mismatch errors omitted.

[27] $\sigma(\Delta st_{j,r})$ =0.8ps is obtained from Monte Carlo simulation of $M_{3,4}$. All other intrinsic and mismatch errors are eliminated by setting $R_L = 0$ and using the output current to measure SFDR.

[28] Simulated by eliminating $\Delta\tau_{out,i}$ effects using a bleeder current of 10μA (section 3.1.3) and omitting all mismatch errors.

the output voltage and time delay errors, further degrading SFDR. In general, compensation techniques are used to address intrinsic limitations such as time constant ($\Delta\tau_{out,i}$) and duty cycle ($\Delta d_{out,i}$) errors, while calibration is employed to correct unary cell random and systematic mismatch such as current ($\Delta I_{j,r/s}$), delay ($\Delta t_{j,r/s}$), duty cycle ($\Delta d_{j,r/s}$) errors.

Chapter 3: Compensation and Calibration Techniques

Compensation and calibration are two distinct correction techniques applied to a DAC at various circuit nodes to improve static and/or dynamic linearity. Compensation is an open loop scheme that blindly (i.e. without measurement) minimizes the code-dependency of errors. In contrast, calibration is a closed loop technique that senses DAC errors and tunes the circuit accordingly. The following sections provide an overview of compensation and calibration techniques and how they have evolved over time. For comparison purposes, key correction techniques are simulated using the 12-bit baseline DAC (section 2.3).

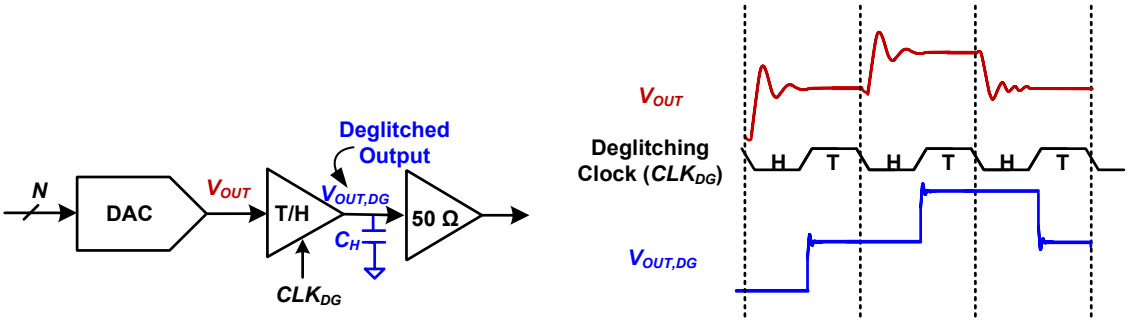
3.1. Compensation Techniques

In general, compensation techniques are used to minimize the correlation between input code and resulting DAC error by: 1) resampling the output signal, 2) altering the sampling clock, 3) modifying the cell structure, or 4) dynamically rearranging the switching sequence of the DAC cells.

3.1.1. Deglitching

A typical method to remove switching-dependent errors[29] is to resample the output after it has settled using a track-and-hold (T/H) circuit [42], depicted in Figure 3.1(a).

[29] This includes timing errors ($\Delta\tau_{out,i}, \Delta\tau_{out,s}, \Delta t_{out,r/s}$), duty cycle ($\Delta d_{out,i}, \Delta d_{out,r/s}$), glitch feedthrough, and DDJ.



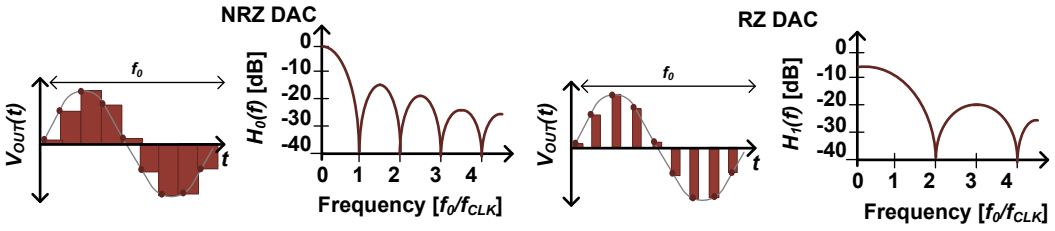
(a) (b)

Figure 3.1: Typical deglitching compensation. (a) Block diagram. (b) Timing diagram.

Utilizing an additional sampling clock, the deglitcher tracks and holds the DAC output only after it has settled to a steady state (Figure 3.1(b)). Whereas deglitching allows for relaxed retiming clock distribution and data path design, the T/H circuit linearity and settling speed ultimately limits the DAC performance [43]. To overcome the speed limitation, a diode-bridge current-steering deglitcher has been proposed with the drawback of increased power consumption [28]. In addition, the deglitcher is followed by a high-linearity output buffer to interface with off-chip circuitry, affecting linearity and settling time.

3.1.2. Return-to-Zero (RZ) and Differential Quad Switching (DQS)

The return-to-zero (RZ) technique was initially conceived to perform deglitching without the drawbacks of the T/H circuit [43]. In general, RZ DACs utilize an additional clock (CLK_{RZ}) that returns the output to a known state every sample period, removing the data dependency of switching glitches. Furthermore, by ensuring $Q_{k,j}$ and $\overline{Q_{k,J}}$ transitions in the reset phase ($CLK_{RZ} = 0$), cell data path delay errors do not propagate to the output. Although data dependencies are mitigated, the RZ output is two times more sensitive to



(a) (b)

Figure 3.2: Sampled sinusoid (V_{out}) and frequency-domain response ($|H_i(f)|$) for the (a) NRZ and (b) RZ DAC.

random clock jitter since it transitions on both rising and falling clock edge [44]. Another drawback of RZ DACs is the reduction of the output signal power at DC by 6dB[30], as depicted in Figure 3.2. However, it also shifts the first null to $2f_{CLK}$, resulting in a flatter response in the first Nyquist zone.

Initial designs implemented RZ in voltage-mode (VRZ) by inserting a single switch between the DAC differential outputs, as pictured in Figure 3.3(a) [43]. This requires only a single clock tap, allowing for a simple clock network. During the reset phase, the DAC differential branches are shorted together which results in zero current flow through R_L, effectively setting $V_{out,VRZ}$ to zero. Based on the switch size, either the nonlinear switch ON resistance or capacitance dominates the output time constant. As a result, the output may not fully charge to its desired value (i.e. settling error) or fail to completely reset to zero (i.e. memory effect), as illustrated in Figure 3.3(e). This data-dependent error ultimately limits the achievable SFDR. VRZ has demonstrated an SFDR of 74dBc (f_0=8.5MHz, f_{CLK}=100MHz) for a 14-bit DAC designed in 0.8µm CMOS [43].

[30] Assuming a 50% CLK_{RZ} duty cycle.

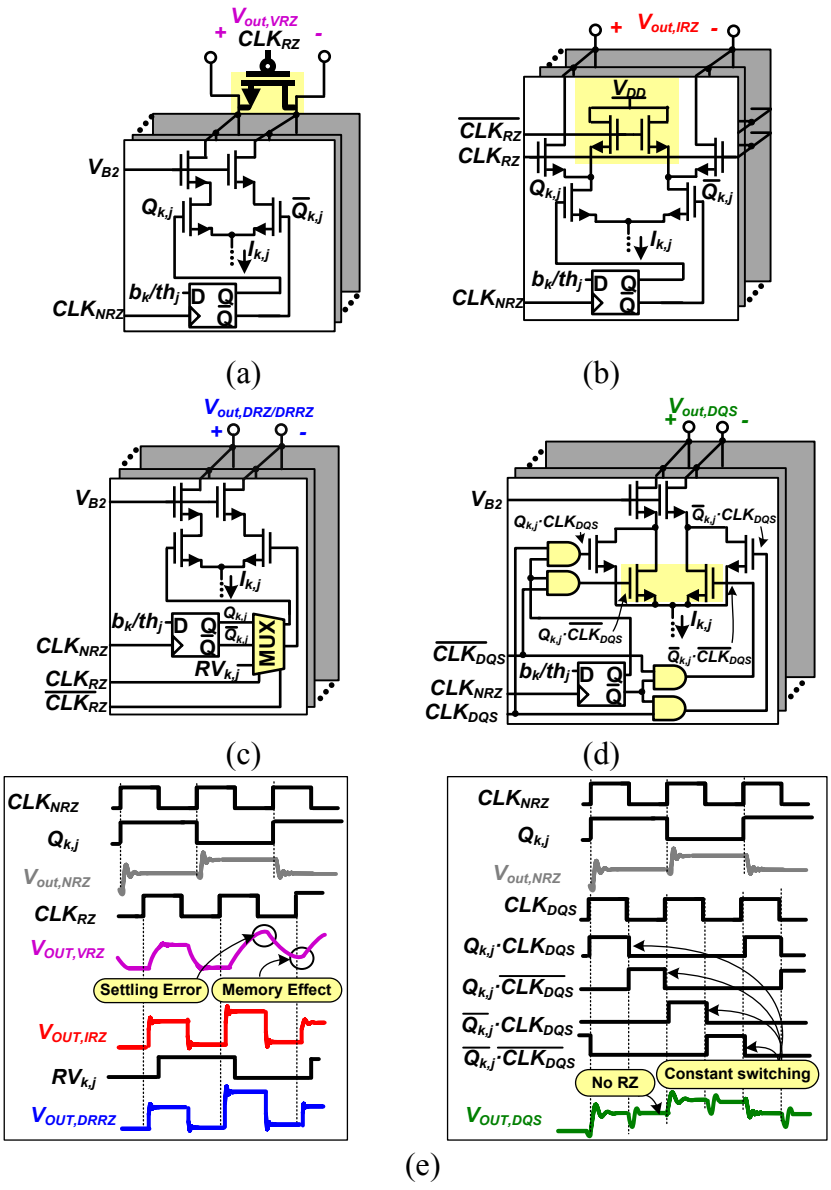


Figure 3.3: Modified DAC architecture with (a) voltage RZ (VRZ), (b) current RZ (IRZ), (c) digital RZ (DRZ/DRRZ), and (d) differential quadrature switching (DQS). (e) Time domain waveforms for NRZ, VRZ, IRZ, DRRZ, and DQS.

In order to break the speed-linearity tradeoff for the single RZ switch, a current-mode RZ (IRZ) approach was proposed to incorporate a small RZ switch in the current path of each cell, as shown in Figure 3.3(b) [45]. During the reset phase, all cell currents are redirected to the power supply and zero current flows through R_L (Figure 3.3(e)). Note

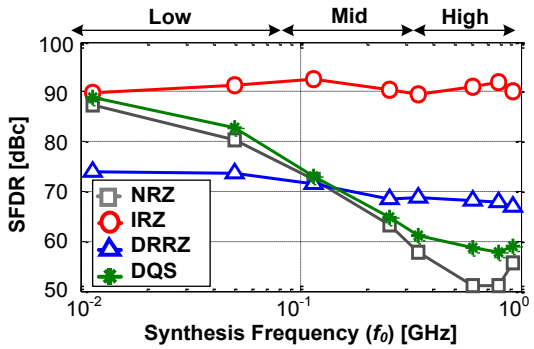
that unlike VRZ, the switched current-mode IRZ maintains the inherent speed of the current-steering architecture at the expense of a more complex RZ clock distribution. Specifically, this requires the RZ clock to be routed to each cell individually, and hence, introduces its own mismatch errors. This approach has been favored for its high-frequency operation, demonstrating an SFDR of 57dBc (f_0 =0.76GHz, f_{CLK}=1.6GHz) for a 12-bit DAC designed in a 1μm GaAs HBT technology, wherein an increase of 12dBc is obtained relative to NRZ [45].

RZ was later proposed in the digital domain (DRZ) to alleviate analog complexity with additional overhead in the data path. DRZ was initially conceived solely for bandwidth extension [46], and later implemented to improve the dynamic DAC behavior using a technique termed digital random RZ (DRRZ) [47]. In DRZ, the cell data is multiplexed with a reset value ($RV_{k,j}$), outputting b_k, th_j during the positive clock cycle (CLK_{RZ}=1) and $RV_{k,j}$ during the negative clock cycle (CLK_{RZ}=0) (Figure 3.3(c)). Note that for DRZ, $RV_{k,j}$ is kept constant per clock cycle at a logical '1' for half the DAC cells and '0' for the other half of the cells such that the output ($V_{out,DRZ}$) returns to zero during the negative clock cycle (Figure 3.3(e)). Given that the cell only switches when $b_k, th_j \neq RV_{k,j}$, the switching glitches remain correlated with the input code while occurring at twice the rate of NRZ, resulting in lower SFDR than NRZ [47]. To address these shortcomings, DRRZ proposes a pseudo-random $RV_{k,j}$ which changes during each reset phase, decorrelating the switching transitions from the input code [47]. Although randomizing $RV_{k,j}$ improves SFDR by spreading the distortion across frequency, it incurs a small penalty in the DAC noise floor [47]. Additionally, since DRRZ is applied before

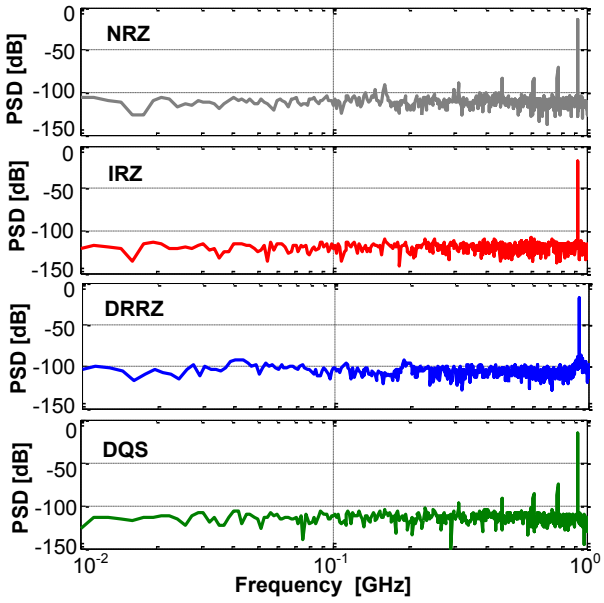
the DAC switch pair, the cell dependent switching mismatch errors remain uncompensated. DRRZ has demonstrated an SFDR of 57dBc (f_0=0.8GHz, f_{CLK}=1.6GHz) for an 8-bit DAC in 90nm CMOS, showing a 14dB improvement over DRZ for the same design [47].

As clock frequency increases, the finite slewing of the narrow RZ output pulse can result in incomplete transitions. This has prompted a technique termed differential quadrature switching (DQS), which does not require the output to return to a reset state, while ensuring uniform cell switching every clock cycle regardless of code [48]. As illustrated in Figure 3.3(d), DQS utilizes four logical signals obtained from the AND operation of input code (b_k, th_j), clock (CLK_{DQS}), and their inverted versions. This can be viewed as two complementary RZ operations where the output voltage ($V_{out,DQS}$) avoids returning to a reset value, thereby incurring zero power loss (Figure 3.3(e)). Additionally, by ensuring constant switching for each cell, irrespective of data transitions, the code dependent glitch error on the common source node (Figure 1.5 V_{cm}) is removed. However, like DRRZ, the cell dependent switching mismatch errors remain uncompensated. Furthermore, since $V_{out,DQS}$ does not return to a reset value, intrinsic output time constant and duty cycle errors are not addressed. This technique has demonstrated an SFDR of 67dBc (f_0=260MHz, f_{CLK}=1.4GHz) [49] and SFDR of 52dBc (f_0=1.5GHz, f_{CLK}= 3GHz) [50] for a 14-bit DAC in 0.18µm CMOS.

In order to compare the performance among NRZ, IRZ, DRRZ, and DQS, the 12-bit baseline DAC is modified accordingly and simulated with full swing CMOS data[31], without clock and transistor mismatch, and with the addition of 3ps peak-to-peak DDJ. As shown in Figure 3.4, IRZ offers the best SFDR by blocking the switch pair ($M_{3,4}$)



(a)



(b)

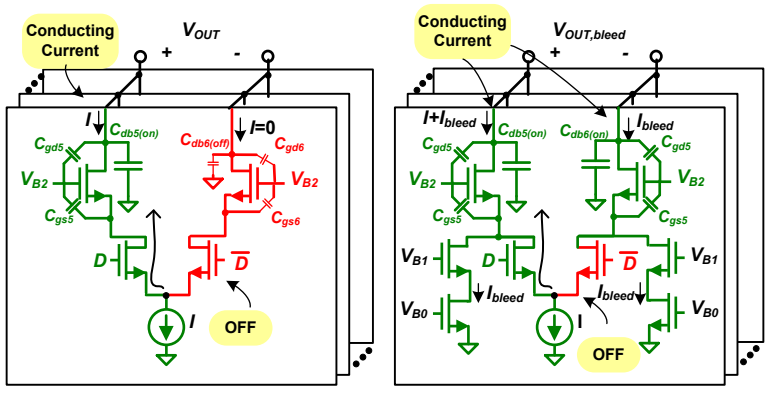
Figure 3.4: Simulated 12-bit baseline DAC (f_{CLK} =2GHz) implemented with NRZ, IRZ, DRRZ, and DQS. (a) SFDR vs. frequency. (b) PSD at f_0 =923MHz.

[31] Using full swing CMOS retiming flip-flop and driver to magnify the impact of switching glitches.

switching glitches and DDJ from propagating to the output, while reducing the data dependency of intrinsic output time constant ($\Delta\tau_{out,i}$) and duty cycle ($\Delta d_{out,i}$) errors. DRRZ comes in next, achieving moderate SFDR improvement in the mid-to-high frequency regions by reducing the data dependency of the switch pair ($M_{3,4}$) switching glitches and DDJ, along with $\Delta\tau_{out,i}$ and $\Delta d_{out,i}$ errors. However, the increase in noise floor, due to the use of a randomized RV, dominates the SFDR, making it perform worse across frequency compared with IRZ. DQS performs better than NRZ at high frequency by pushing switching glitches to f_{CLK}, however, it fails to correct the distortion caused by DDJ, $\Delta\tau_{out,i}$, and $\Delta d_{out,i}$.

3.1.3. Always-ON-Cascoding

Intrinsic output time constant errors ($\Delta\tau_{out,i}$), induced by modulation of the output capacitance, can be reduced by keeping the cascode transistors ($M_{5,6}$) from switching between ON and OFF states [14]. As shown in Figure 3.5, both cascode transistors are



(a) (b)

Figure 3.5: DAC cell highlighting ON and OFF branches, (a) normal cell, (b) with always-ON-cascoding

kept ON, independent of code, by bleeding a small current[32] through them at all times. This approach has been shown to yield an SFDR of 50dBc (f_0=1GHz, f_{CLK}=7GHz) for a 6-bit DAC in 28nm CMOS [51].

3.1.4. Dynamic Element Matching (DEM)

Digital signal processing techniques have emerged as a method to randomize DAC cell mismatch errors, effectively converting mismatch-induced distortion into white noise. Dynamic element matching (DEM) is an early approach proposed to temporally average output voltage mismatch errors, thus improving the effective DAC INL, which degrades SFDR at low frequency [52]. Subsequently, it was also proven to randomize timing mismatch errors [53].

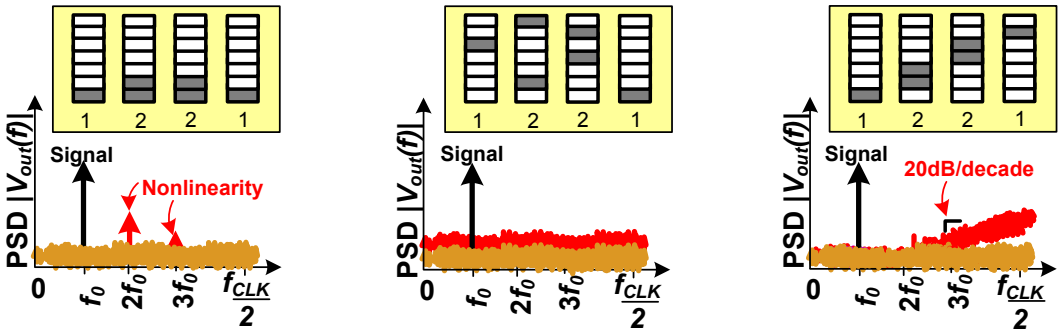
Initially, DEM was conceived in a binary DAC, which implemented a single current source (I_{ref}) and a series of cascaded current divide-by-two stages [52]. At each divide-by-two step, one tap (a) is used for the binary cell current while the other tap (b) is passed to the next divide-by-two stage. This action is repeated N times to form the binary weighted currents ($I_{ref}/2, I_{ref}/4, \ldots I_{ref}/2^N$). DEM is accomplished by periodically interchanging between (a) and (b) taps, reducing the average current error per cell.

In modern DAC designs, DEM is achieved by dynamically swapping connections between the thermometer code (th_j) and the unary cells according to a control sequence that is updated at the clock rate (f_{CLK}) [54]. A myriad of stochastic or deterministic algorithms have been developed to implement the control sequence [55]. Each of these

[32] Typically 1%-2% of the total cell current is sufficient.

techniques strives to minimize mismatch, reduce hardware/computational complexity, and in some cases, provide noise shaping.

To illustrate the difference between stochastic and deterministic DEM techniques, consider a 7-unary cell DAC with input thermometer code pattern of '1221'[33], as depicted in Figure 3.6. The unaltered switching order, depicted in Figure 3.6(a), selects the same group of unary cells each time code '1' or '2' occurs. Stochastic DEM algorithms generate a control sequence that selects the current source combinations at random. In essence, this temporally changes the DAC INL every clock period, resulting in an overall average INL improvement when compared to a DAC with a static INL [56]. As a result, the mismatch-induced distortion is converted to white noise, yielding an improvement in SFDR, as shown in Figure 3.6(b). In contrast to stochastic DEM which raises the noise floor uniformly across frequency, deterministic DEM algorithms shape the noise to higher frequencies. An early deterministic algorithm, termed data-weighted averaging (DWA), selects current sources in a rotary fashion to provide 1st order noise shaping (i.e.



(a) (b) (c)

Figure 3.6: Illustration of switching order and PSD for 7-unary cell DAC implemented with (a) no DEM, (b) stochastic DEM, and (c) deterministic DWA.

[33] Code=1 corresponds to $th_0 = 1$ and $th_{1:6} = 0$ while code=2 corresponds to $th_{0:1} = 1$ and $th_{2:6} = 0$.

20dB/decade), as shown in Figure 3.6(c) [57]. Other variants that emerged, such as bidirectional DWA [58], further reduces spurs, while vector-feedback and tree-structure mismatch shaping DEM [59] provide higher orders of noise shaping at the cost of increased complexity. Note that deterministic techniques are often implemented in oversampled ΔΣ DACs which take advantage of noise shaping to achieve high SFDR over a narrow frequency range [57].

In general, DEM techniques are less effective for Nyquist-rate DACs when the rate of averaging (typically f_{CLK}) becomes comparable to the synthesized signal frequency (f_0) [60]. In addition, the data must be shuffled at either the clock rate or via pipelining to meet timing constraints, exacerbating power consumption in high-speed, high-resolution DACs. Even with these challenges, a stochastic DEM sequence was implemented in a 14-bit DAC to achieve an SFDR of 75dBc (f_0=491MHz, f_{CLK}=1GHz) in 65nm CMOS [61], while DEM was used in conjunction with DRZ to demonstrate an SFDR of 70dBc (f_0 =0.8GHz, $f_{CLK} = 1.6$GHz) for a 12-bit DAC in 40nm CMOS [62].

3.2. Calibration Techniques

The paper discussion will now shift focus to calibration, which combines error measurement and correction to improve DAC linearity. Initially, calibration methods were developed to correct output voltage mismatch errors ($\Delta V_{out,r/s}$) which dominate SFDR at low frequency. However, the demand for GHz speed DACs necessitates calibration of output delay ($\Delta t_{out,r/s}$) and duty cycle ($\Delta d_{out,r/s}$) mismatch errors as well. Generally, DACs with $N > 10$ bits have close to 50% segmention such that binary cell mismatch has little impact on the DAC linearity [63] [64]. Therefore, most calibration

techniques measure and correct each unary cell error individually, while leaving the binary cells as is or applying a single calibration to the entire binary array. The subsequent sections provide a brief overview of error sensing methods followed by a review of various amplitude and timing calibration techniques as they have evolved over time.

3.2.1. Mismatch Measurement Circuits

Early on, output voltage mismatch errors ($\Delta V_{out,r/s}$) were determined by measuring V_{out} with a high-resolution, low-speed ADC and comparing its output to the input digital code [63]. This approach is limited to static DNL/INL measurement and lacks the ability to directly measure individual cell errors. Later techniques modified the unary cell, routing its current (I_j) to a measurement circuit via additional transistors, as depicted in Figure 3.7. One additional transistor ($M_{c,j}$) is placed before the switch pair to extract cell-to-cell amplitude differences [65], while two transistors ($M_{N,j}M_{P,j}$) are placed after the

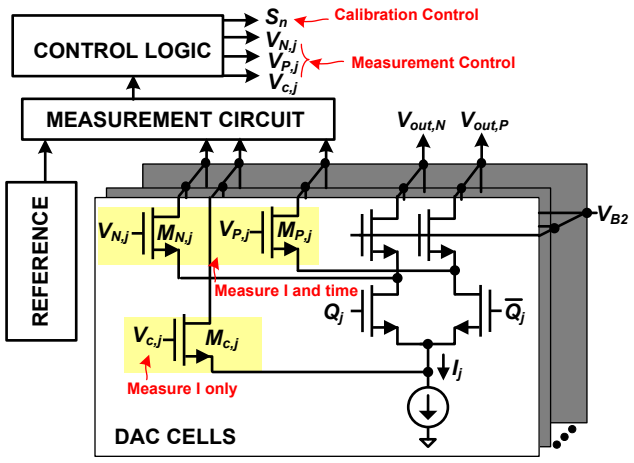


Figure 3.7: Modified DAC architecture with auxiliary measurement transistors ($M_{c,j}, M_{N,j}, M_{P,j}$) and calibration hardware.

switch pair to measure amplitude and timing mismatch [40].

In order to quantify amplitude errors, I_j and a common reference current ($I_{ref} = 2^{N-m}I_{LSB}$) are routed to a current comparator while succesively adjusting I_j to minimize the mismatch error. Typically, I_{ref} is realized by combining the binary cells with an additional LSB cell to maintain equal scaling between the sum of the LSBs and unary cell current [66]. The authors in [67] simulated a sense accuracy of 8nA for I_{ref} =10μA using a 0.18μm CMOS comparator.

On the other hand, unary cell delay mismatch is extracted by switching a unary and a common reference cell using the same clock edge and comparing their phase difference over several clock cycles. The unary cell delay is then adjusted until the phase difference is minimal. An early implementation utilized a digital lead-lag phase detector to sense a minimum phase difference of 2ps, simulated in a 0.18μm CMOS process [68]. Later on, [69] proposed a high-resolution switch capacitor circuit which integrates the measured switching time charge difference with a simulated accuracy of 125fs.

An alternative approach, based on a zero-IF receiver, was proposed in [44] to collectively measure current (ΔI_j), delay (Δt_j), and duty cycle (Δd_j) mismatch errors. For illustration purposes, Figure 3.8 depicts these respective errors as they are applied separately to cells j =0, 1, and 3, while cell $j = 2$ combines both ΔI_j and Δt_j. To extract the cell errors, the unary cell under test is switched at a measurement frequency (f_m), and its output is added to that of a reference cell switching at an opposite phase. The summed output error signal (e_j) is subsequently downconverted to DC using an I-Q mixer and captured with a high-resolution ADC. Mixing e_j with a sinusoidal LO, operating at the

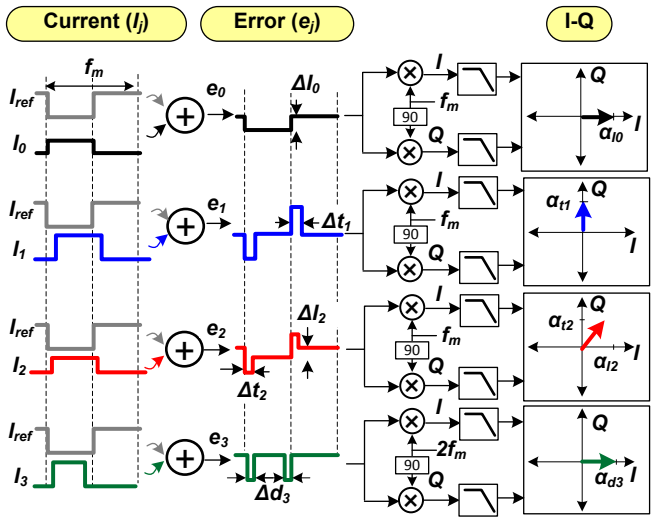


Figure 3.8: Concept of zero-IF mismatch sensor measurement. The measured reference waveform (I_{ref}) is added to the cell waveform (I_j), downconverted with an I-Q mixer, and lowpass filtered to DC to provide a measure of the current ($\alpha_{Ij}\sim 2\Delta I_j/\pi$), delay ($\alpha_{tj}\sim 4I_j f_m \Delta t_j$), and duty cycle ($\alpha_{dj}\sim 4I_j f_m \Delta d_j$) error between I_{ref} and I_j.

same frequency f_m, yields an accurate representation of ΔI_j ($\alpha_{Ij}\sim 2\Delta I_j/\pi$) and Δt_j ($\alpha_{tj}\sim 4I_j f_m \Delta t_j$) in the output I and Q components, respectively. In the presence of amplitude and time mismatch, an error vector with weighting α_{Ij} and α_{tj} results after mixing. Alternatively, since Δd_j occurs at both clock edges, in phase mixing of e_j with a sinusoidal LO at twice the frequency $2f_m$, provides a measure of Δd_j ($\alpha_{dj}\sim 4I_j f_m \Delta d_j$). Note that using a larger f_m increases the measured magnitude of α_{tj} and α_{dj}. Utilizing this zero-IF mismatch sensor, the authors in [44] simulated a current resolution of 22nA and timing resolution of 170fs in 0.14µm CMOS.

3.2.2. Amplitude Calibration

Amplitude calibration aims to reduce output voltage mismatch errors ($\Delta V_{out,r/s}$) by counteracting them using an auxiliary circuit. This allows significant reduction in current

source area when compared with the intrinsic accuracy size required to meet a certain DNL/INL and SFDR [70]. Initially, correction was applied at the output current combining network of the DAC core using a supplemental current-mode DAC [71] [63]. Although this allows a compact DAC cell design, it requires precise time alignment between the main and supplemental DACs. Alternatively, a cell-by-cell adjustment can be made by calibrating each I_j to I_{ref}.

An early implementation periodically disconnects each unary cell's current source (M_1) from its switch pair and diode connects it to a current sink, I_{ref} [72]. This ensures that the bias voltage that matches each unary cell current is captured and held by the respective M_1 gate capacitance. Later designs keep M_1 connected to the switch pair and tune the gate voltage using a bias-DAC to minimize the difference between I_j and I_{ref}. In one implementation, a single bias-DAC is multiplexed between each cell, periodically refreshing the gate voltage [73]. However, periodic gate biasing has many drawbacks as it is susceptible to settling, charge injection, and leakage errors, thereby limiting the DAC speed and calibration accuracy. To mitigate these issues, the authors in [66] propose using a bias-DAC per unary cell, with the penalty of increasing die area.

Modern DAC designs use a per-cell current output DAC (cal-DAC) in parallel with M_1, depicted in Figure 3.9(a), to minimize the difference between I_j and I_{ref} down to the calibration resolution (I_{res}) [65] [39]. The cal-DAC can be implemented with an array of switched current sources, requiring less area than voltage output DACs, thus making per cell implementation practical. Note that the current source and cal-DAC can be placed in a separate array outside of the DAC retiming flip-flop, driver, and switch pair to keep the

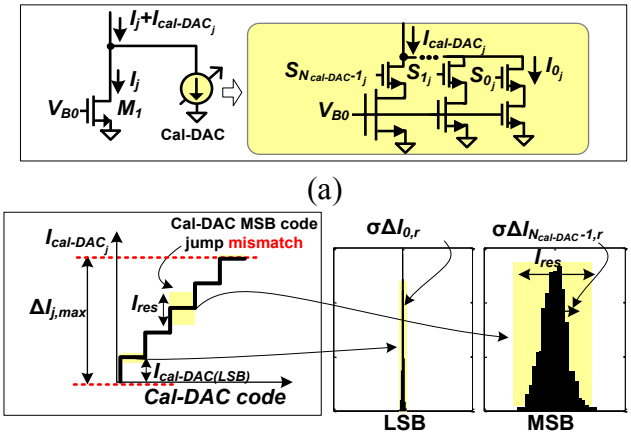


Figure 3.9: Amplitude calibration applied at the current source gate node and the current source drain node.

data path compact. To reduce die area, the cal-DAC can be implemented using an array of $N_{cal-DAC}$ binary-weighted current ($I_c = 2^c I_{cal-DAC(LSB)},\ c = 0, \dots, N_{cal-DAC} - 1$) cells and control bits ($S_{j,c}$) to provide the desired calibration current per cell[34]

$$I_{cal-DAC_j} = \sum_{c=0}^{N_{cal-DAC}-1} S_{j,c}\, I_c. \tag{3.1}$$

The cal-DAC has two constraints, 1) the total $I_{cal-DAC_j}$ must cover the unary current mismatch peak-peak spread ($\Delta I_{j,p-p}$) and 2) its maximum DNL, which occurs at the mid-code transition ($\Delta I_{N_{cal-DAC}-1}$), must be less than the desired calibration resolution (I_{res}), as shown in Figure 3.9(b). Condition 2 is achieved by ensuring $I_{cal-DAC(LSB)} \pm 3\sigma\left(\Delta I_{N_{cal-DAC}-1,r}\right) \leq I_{res}$[35], while condition 1 is achieved by choosing an appropriate value for $N_{cal-DAC}$ relative to $I_{cal-DAC(LSB)}$. The SFDR after calibration is found using [74]

[34] To correct for $-3\sigma\Delta I_{j,r}$ mismatch, I_j plus the mid cal-DAC value ($2^{N_{cal-DAC}-1} I_{cal-DAC(LSB)}$) is set equal to $2^{N-m} I_{LSB}$.
[35] In a mismatch free cal-DAC $I_{cal-DAC(LSB)} = I_{res}$

where the thermometer cell mismatch after calibration is [74]

$$\sigma\left(\frac{\Delta I_j}{I_j}\right)_{pc} = \sqrt{\frac{INL_{max}^2 - 3^2 2^m \sigma\left(\frac{\Delta I_{k=0}}{I_{k=0}}\right)^2}{3^2 2^m (2^{N-1} - 2^m)}}, \tag{3.2}$$

where

$$\sigma(\Delta I_j)_{pc} = \frac{I_{res}}{\sqrt{6}}. \tag{3.3}$$

Using (3.4) and (3.5) [10] [5] , the SFDR after cal-DAC calibration is (3.6).

$$INL_{max} = \sqrt{2^{N-1}}\sigma\left(\frac{\Delta I_{k=0}}{I_{k=0}}\right) \tag{3.4}$$

$$E\{SFDR[dBc]\} = 3N + 7.5 - 20log_{10}\,\sigma\left(\frac{\Delta I_{k=0}}{I_{k=0}}\right) \tag{3.5}$$

$$E\{SFDR[dBc]\} = 3N + 5.8 - 20log_{10}\left(\frac{I_{res}}{2^{N-m} I_{LSB}}\sqrt{\frac{2^m(2^{N-1} - 2^m)}{2^{N-1} - 3^2 2^m}}\right) \tag{3.6}$$

To maximize the benefits of cal-DAC calibration, m and $N_{cal-DAC}$ should be chosen to minimize the combined current source and cal-DAC area (A_{cal}). Table 3.1 shows an area optimization[36] for different N-bit DACs achieving a range of SFDR values The intrinsic

Table 3.1: Required SFDR for various N-bit DACs, corresponding intrinsic current source area (A_{no-cal}) without calibration, and combined current source and cal-DAC area (A_{cal}) after calibration for optimized m and $N_{cal-DAC}$ values.

N	SFDR [dBc]	No Cal	With cal-DAC		
		A_{no-cal} [mm^2]	m	$N_{cal-DAC}$	A_{cal} [mm^2]
12	70	0.022	2	4	0.012
14	80	0.225	3	4	0.053
16	90	2.25	4	5	0.25

[36] Calculated in MATLAB.

current source area, without calibration, (A_{no-cal}) required to meet a specified SFDR is calculated [5] along with the minimum area after calibration (A_{cal}) using the best combination of m and $N_{cal-DAC}$ to achieve the same SFDR [74] [10]. As shown, cal-DAC calibration allows for significant reduction in area of up to ~10x. Overall, the cal-DAC technique is the preferred choice for amplitude calibration and has demonstrated an SFDR of 84dBc (f_0=5MHz f_{CLK}=50MHz) for a 12-bit DAC in 0.25µm CMOS [39]. When combined with IRZ and always-ON-cascoding, a 14-bit DAC demonstrated an SFDR of 67dBc (f_0 =3.5GHz, f_{CLK} = 7.2GHz) in 0.13µm BiCMOS [75].

3.2.3. Delay Calibration

Similar to amplitude calibration, per-cell unary timing adjustment can be used to reduce output delay mismatch errors ($\Delta t_{out,r/s}$) down to the measurement and calibration circuit resolution (t_{res}). One technique proposes inserting a tunable digital delay line (DDL) between the retiming flip-flop and driver to improve SFDR [68]. However, for a large number of unary cells, placing a DDL in each cell data path exacerbates the DAC core area and power consumption. As a result, to the authors' knowledge, this approach has not been demonstrated in high-resolution Nyquist DACs. Nevertheless, per-cell DDL calibration has recently been applied to a ΔΣ DAC utilizing a 3-bit (7-unary cell) quantizer to achieve an SFDR of 76.2dBc (f_0 =997MHz, f_{CLK} =2GHz) with t_{res} =90fs in 0.13µm BiCMOS [76].

3.2.4. Mapping

The notion of unary cell reordering to reduce INL has been well studied, dating back to pre-fabrication techniques that modify the switching sequence based on layout amplitude errors [9] [10] [23]. Mapping builds on this concept to statically exchange connections between the thermometer code (th_j) and the unary cells based on measured cell errors [77]. Unlike DEM, which dynamically swaps interconnections, mapping improves linearity without incurring a noise floor penalty.

Initially, amplitude mismatch mapping (AMM) was developed to correct output voltage mismatch errors ($\Delta V_{out,r/s}$), [78]. Later on, it was adapted as time mismatch mapping (TMM) to correct output delay mismatch errors ($\Delta t_{out,r/s}$) [68]. This has further evolved into dynamic mismatch mapping (DMM), which combines amplitude and time mapping, to simultaneously correct $\Delta V_{out,r/s}$, $\Delta t_{out,r/s}$, and output duty cycle mismatch ($\Delta d_{out,r/s}$) errors [40]. In fact, DMM is credited for simultaneously correcting all three mismatches and the first to suggest $\Delta d_{out,r/s}$ calibration.

In AMM, the unary cell switching sequence is rearranged based on measured current mismatch ($\Delta I_{j,r/s}$). Overall, two distinct schemes have been proposed: switching sequence post adjustment (SSPA) [36] and complete current folding (CCF) [70]. Note that both techniques use a 'sort-and-group' method to rearrange the cells. To illustrate this method, the relative current mismatch error (*a*-*g*) for 7-unary cells (i.e. m=3) is plotted in Figure 3.10. The following steps are performed in the 'sort-and-group' process:

① The original errors (a-g) are sorted in increasing order.

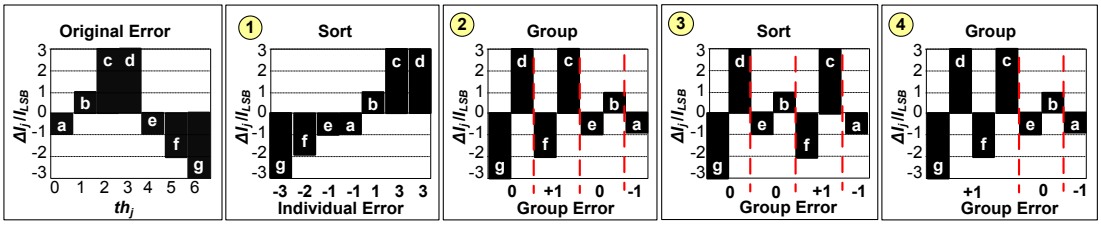


Figure 3.10: Sort-and-group method applied to 7-unary cell errors (a-g) illustrating the steps to determine the new switching sequence for amplitude mapping techniques: SSPA and CCF.

② The largest negative error (g) is grouped with the largest positive error (d), the second to largest negative error (f) is grouped with the second largest positive error (c), and so forth, such that the summed group error (0,+1,0) has the smallest possible value. Note (*a*) is left ungrouped.

③ Each of the summed group errors is treated as a unit cell and sorted similar to ①.

④ Grouping is performed on the summed errors as in ②, resulting in one quartet group of errors (*g,d,f,c*) as well as two untouched groups (*e*,*b*) and (*a)*.

At this point, the maximum number (x) of 'sort-and-group' iterations has been performed for the 7-unary cell example. In general, the 'sort-and-group' iterations can be performed 1 to m-1 times. For SSPA, the cells are kept in unary form and the new switching order error becomes '*g*, *d*, *f*, *c*, *e*, *b*, *a*' (x =2). Alternatively, CCF combines cells to form either a hybrid binary/thermometer DAC with cell errors (*a, gd*, *fc*, *eb*) (x =1), or a full binary DAC with cell errors (*a*, *eb*, *gdfc*) (x =2). Table 3.2 further illustrates the original, SSPA (x =2), and CCF (x =2) error switching sequence. along with the corresponding DNL/INL. As shown, SSPA and CCF yield a better INL than the original sequence, while CCF also improves the DNL.

Table 3.2: Input code vs. cell error, DNL, and INL for no mapping and 2 'sort-and-group' iterations for SSPA and CCF.

Code	No Map		SSPA		CCF	
	Error	DNL /INL	Error	DNL /INL	Error	DNL /INL
001	a	-1/-1	g	-3/-3	a	-1/-1
010	a+b	+1/0	g+d	+3/0	b+e	0/-1
011	a+b+c	+3/+3	g+d+f	-2/-2	a+b+e	-1/-2
100	a+b+c+d	+3/+6	g+d+f+c	+2/0	c+f+d+g	+1/-1
101	a+b+c+d+e	-1/+5	g+d+f+c+e	-1/-1	c+f+d+g+a	0/-1
110	a+b+c+d+e +f	-2/+3	g+d+f+c+e+ b	+1/0	c+f+d+g+b +e	+1/0
111	a+b+c+d+e +f +g	-3/0	g+d+f+c+e+ b+a	-1/-1	c+f+d+g+b +e+a	0/0

Table 3.3: Simulated DNL, INL, and SFDR for baseline 12-bit DAC (f_{CLK} =2GHz) with no mapping and several x 'sort-and-group' iterations for SSPA and CCF.

Performance	No Map	SSPA, *x*=			CCF, *x*=		
		1	2	3	1	2	3
DNL [LSBs]	2.0	2.0	2.0	2.0	0.4	0.3	0.2
INL [LSBs]	11	3.1	3.0	2.9	2.2	2.0	2.0
SFDR [dBc] ($f_0 = 10\ MHz$)	53	78	85	87	80	88	88
SFDR [dBc] ($f_0 = 350\ MHz$)	53	79	86	86	80	84	84

A simulation using behavioral modeling of SSPA and CCF is performed using the 12-bit baseline DAC[37] with a current mismatch distribution $\sigma(\Delta I_{j,r}/I_{LSB}) = 40\%$[38] that corresponds to a -3σ(SFDR)=53dBc. The resulting DNL/INL and SFDR with x 'sort-and-group' iterations are displayed in Table 3.3. As expected, SSPA only improves INL, while CCF improves both DNL and INL. For both SSPA and CCF, a larger number of iterations (x) results in better averaging, as displayed by the INL and low frequency (i.e. f_0 =10MHz) SFDR performance. The loss in CCF SFDR at higher frequencies (i.e.

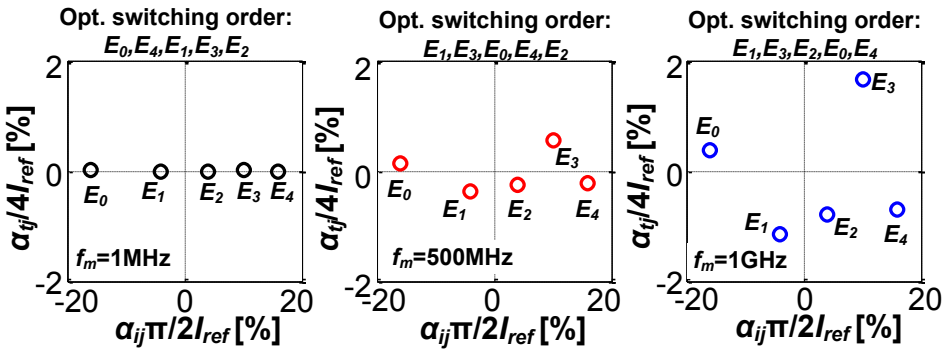
[37] Intrinsic errors eliminated by setting $R_L = 0$ and using I_{bleed} =10μA.
[38] $W_{1,k=0}L_{1,k=0}$ =1.5μm×1μm.

f_0 =900MHz) occurs beyond x=1, as more thermometer cells are converted into binary. Although additional 'sort-and-group' iterations in CCF result in lower DNL/INL, the binary configuration can lead to larger switching glitches which induce spurs in the output PSD[39] leading to a tradeoff between static and dynamic performances. In [36], a 14-bit DAC using SSPA has demonstrated an SFDR of 78dBc (f_0=2MHz with f_{CLK}=200MHz) in 0.18μm CMOS.

Time mismatch mapping (TMM) was initially proposed in [77] and further analyzed in [11] [68] to validate that SFDR can be enhanced by optimizing the unary cell switching sequence based on the cell delay mismatch (Δt_j) errors. Specifically, [68] simulated a 12-bit DAC with an SFDR of 95dBc (f_{CLK} =200MHz), showing a 35dB improvement from a worst case linear timing distribution. Interestingly, while TMM shows promising results in simulation, to the authors' knowledge, it has not been verified in silicon as a standalone technique.

While addressing $\Delta V_{out,r/s}$ and $\Delta t_{out,r/s}$ independently, AMM and TMM fail to correct for both errors simultaneously since the switching order decision is determined by either current or delay mismatch errors. To overcome this limitation, DMM has been recently proposed in [40], which envisaged a switching technique based on an algorithm that factors in the trio of output amplitude, delay, and duty cycle mismatch errors. These cell errors are captured using the zero-IF mismatch measurement sensor (Section B.1) and represented as a 3-D vector ($E_j(\alpha_{Ij}, \alpha_{tj}, \alpha_{dj})$) which is used to optimize the switching

[39] Segmentation is often chosen to provide the smallest area while incurring the lowest amount of switching glitches from the binary cell configuration. Therefore converting the thermometer cells back into binary can result in a poor dynamic performance caused by binary switching.



(a) (b) (c)

Figure 3.11: 2D error vector (E_j) I (α_{ij}) and Q (α_{tj}) components for 5 unary cells when f_{CLK} =2GHz, showing DMM optimized switching order for (a) f_m =1MHz (b) f_m =500MHz (c) and f_m =1GHz.

sequence. For simplicity, consider the switching sequence for a 2D error vector ($E_j(\alpha_{Ij}, \alpha_{jt})$), for different values of f_m as depicted in Figure 3.11. The switching sequence is determined such that the 2D error vector of one cell is maximally cancelled out by the 2D error vector of the next switching cell. Note that a larger weight is placed on timing errors as f_m increases. Consequently, DDM simplifies to AMM when setting $f_m \sim 0$, and beyond a certain f_m DMM simplifies to TMM. Therefore, the SFDR performance can be optimized across frequency, and is bound by AMM at DC and TMM at high frequency[40]. This technique has been validated using a 14-bit DAC in 0.14μm CMOS, resulting in an SFDR of 78dBc (f_0=100MHz, f_{CLK}=200MHz) [40]. Another variant of this method recombines thermometer cells into binary weighted cells, similar to CCF, termed 3D sort and combine, and has achieved an SFDR of 55dBc (f_0=700MHz, f_{CLK}=3.2GHz) using a 16-bit DAC in 65nm CMOS [41].

[40] Ignoring duty cycle mismatch.

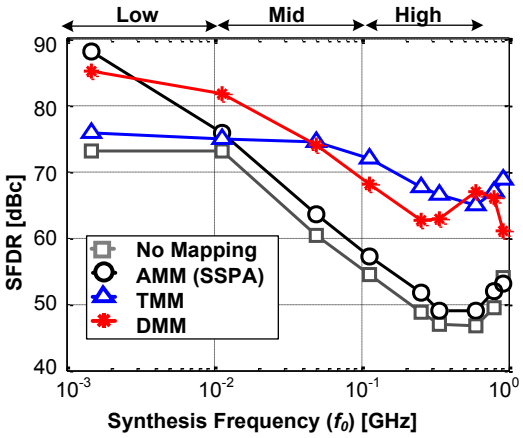


Figure 3.12: Simulated 12-bit baseline DAC (f_{CLK} =2GHz) with no mapping, SSPA AMM, TMM, and DMM. Note that one 'sort and group' iteration was used for all mapping techniques.

To compare the performance benefits of AMM, TMM, and DMM they are applied to the 12-bit baseline DAC[41] with $\sigma(\Delta I_{j,r}/I_{LSB})$ =4% and $\sigma\Delta t_{j,r}$ =1ps. Note that in this analysis duty cycle mismatch is omitted to simplify the comparison among the three mapping techniques. The results, shown in Figure 3.12, indicate that AMM achieves a large improvement in low frequency SFDR but falls off rapidly when delay mismatch starts to dominate. Conversely, TMM achieves the best SFDR in the high band, but is limited by voltage mismatch in the low frequency band. In the case of DMM, the switching sequence was optimized to provide higher SFDR than TMM in the low band, while also improving SFDR over AMM in the mid-to-high frequency region. However, DMM does not achieve the maximum amplitude or timing mismatch correction since it balances its mapping between the two.

[41] Intrinsic errors were eliminated by setting R_L=0 and using I_{bleed}=10μA.

3.2.5. Digital Pre-distortion (DPD)

In general, DPD is a priori feed-forward technique which manipulates the digital input data to correct for an undesired system response. When applied to a DAC, DPD superimposes an inverse correction algorithm on the digital code (b_k, th_j) to correct a particular non-ideality. A classic example of DPD corrects ZOH distortion by applying an inverse sinc equalization function [79]. A side effect of this technique is the reduction in output power at lower frequencies in return for a relatively constant power within the first and second Nyquist zones.

More recently, a pulsed DPD technique has been developed to correct current $(\Delta I_{j,r/s})$, delay $(\Delta t_{j,r/s})$, and duty cycle $(\Delta d_{j,r/s})$ mismatch errors [19]. Ideally, these mismatch

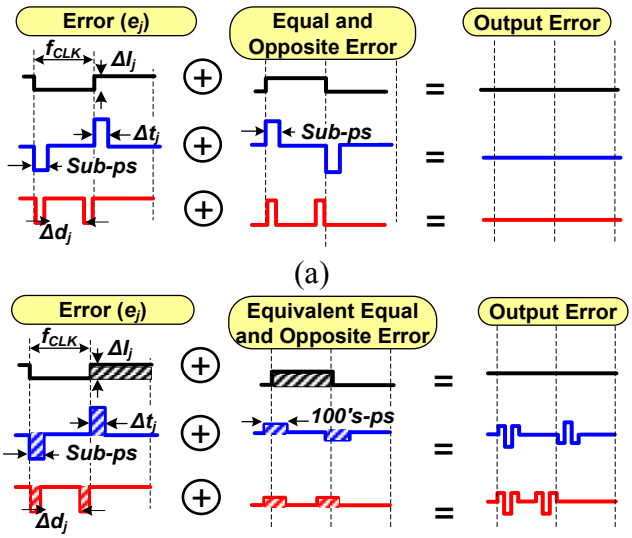


Figure 3.13: Pulse error pre-distortion to correct current $(\Delta I_{j,r/s})$, delay $(\Delta t_{j,r/s})$, and duty cycle $(\Delta d_{j,r/s})$ mismatch errors using (a) sub-ps timing pulses and (b) equivalent area pulses with longer time and less amplitude.

errors are measured and an equal but opposite error is applied to the output, depicted in Figure 3.13(a), using a sub-DAC operating at f_{CLKsub}. However, this requires a sub-DAC to generate sub-ps pulses to correct timing errors. Instead, the authors in [19] propose using an equivalent area pulse, that is longer in time and lower in amplitude than the sub-ps timing correction pulse[42], as shown in Figure 3.13(b). Note that widening the pulse relaxes the timing constraint at the expense of more amplitude precision, leading to a tradeoff between the sub-DAC speed and resolution. The maximum pulse width is bound by the DAC period ($1/f_{CLK}$) to ensure error correction within each clock period. This technique has been validated for a 12-bit DAC using a hybrid DAC architecture containing a 4-bit thermometer DAC with f_{CLK}=2GHz and an 8-bit binary ΔΣ DAC, also used as the sub-DAC, with f_{CLKsub}=8GHz. Together with DWA, this technique achieved an SFDR of 74.4dBc at f_0=950MHz in 65nm CMOS [19].

3.3. Current and Emerging Architectures

Historically, synthesis of RF signals was performed by analog mixing architectures, such as the homodyne transmitter. These architectures combined a low-speed DAC with nonlinear blocks such as mixers, filters, and power amplifiers (PAs) to generate RF signals. Utilization of a high-speed DAC to directly synthesize RF frequencies (i.e. RF-DAC) allows transmitter designers to remove the nonlinear mixer and filter and assign I/Q and data preprocessing to the digital domain [80], [81]. Combining direct-RF synthesis with a high output power DAC (i.e. RF power DAC) allows direct connection

[42] For example, a 1ps, 10mA pulse is equivalent in area to a 1ns, 10μA pulse.

of the DAC to the antenna, the Holy Grail of transmitter designs. In this march towards the antenna, the DAC must generate both high frequency and output power with sufficient linearity to translate complex modulation schemes [80]. Over the past decade, a variety of current-steering DAC architectures have been proposed to further increase the frequency of operation and extend the synthesizable bandwidth, while achieving high output power [82]. This includes mixing, multiple RZ (MRZ), and time-interleaving (TI) DACs, which enable direct synthesis of RF frequencies by modifying the output response, as depicted in Figure 3.14.

3.3.1. Mixing DACs

Mixing DACs combine the mixer and DAC operations in one circuit to synthesize a wide BW around an LO frequency ($f_{LO} = n \cdot f_{CLK}; n \epsilon Z \geq 1$) (Figure 3.14). Generally, mixing DAC architectures use either active or passive mixing. Active mixing (Figure 3.15(a)) modulates the current source with the LO. In [83] a CMOS active mixer DAC demonstrated an SFDR > 75 dBc within a 17.5 MHz bandwidth centered at 942 MHz. Passive mixing utilizes current commutating switches in series with the differential switch pair. Passive mixing can be accomplished locally, within each DAC cell (Figure

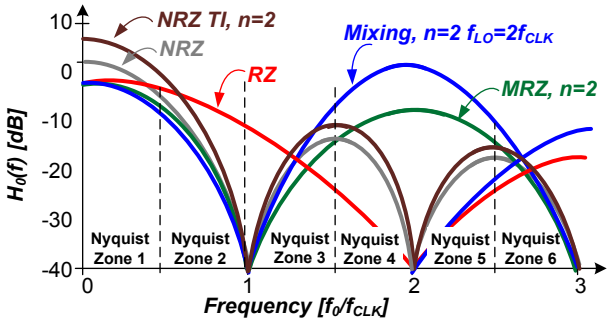


Figure 3.14: Frequency-domain response for NRZ, mixing, MRZ, and NRZ TI DAC.

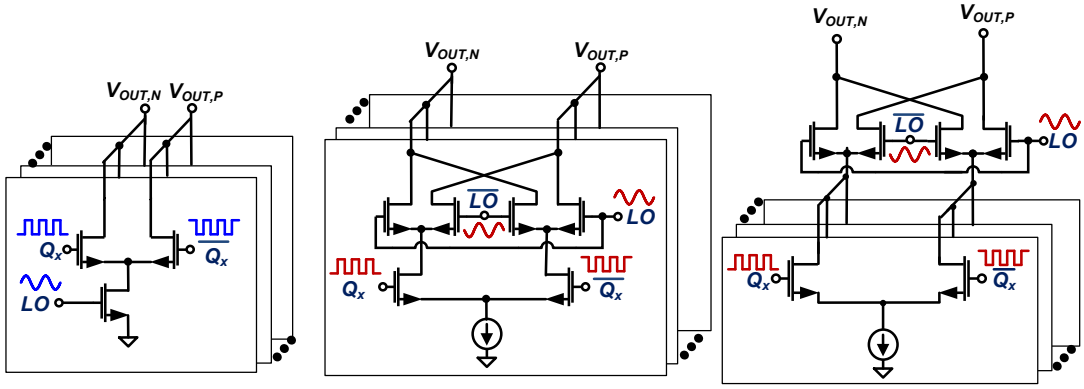


Figure 3.15: (a) Active mixer DAC, (b) passive mixer DAC with local mixing, and (c) passive mixer DAC with global mixing

3.15(b)), or globally at the DAC output summing node (Figure 3.15(c)). Global mixing commutates the full dynamic range of the DAC output current, and as a result, suffers from poor linearity. On the other hand, local mixing only commutates the cell current, relaxing the dynamic range requirement at the expense of distributing a low skew LO signal to all of the commutating switches. In this case, LO switching time disparities result in additional sources of delay (Δt_j) and duty cycle (Δd_j) mismatch, reducing the DAC's dynamic performance [84] [85] [86]. For example, [87] notes that to achieve an SFDR=85dBc at f_{LO}=3.9GHz requires the LO timing mismatch to be in the sub-ps range[43]. In [88], a local mixing DAC with 3D sort and combine as well as always on cascoding demonstrated an SFDR of 66dBc (f_0=155MHz, f_{CLK}=1.75GHz, f_{LO}=1.75GHz) for a 16-bit DAC in 65nm CMOS.

[43] $\sigma(\Delta t_j)$ <36fs and $\sigma(\Delta d_j)$ <850fs.

3.3.2. Multiple RZ (MRZ) DACs

The SFDR performance of mixing DACs can be improved by combining mixing with RZ at the expense of reduced output power. This technique, referred to as MRZ, achieves both deglitching and direct up-conversion by splitting the conventional RZ pulse into an integer number (n) of pulses for each DAC clock period [85]. This results in signal images with major lobes at DC and $n \cdot f_{CLK}$, such that power increases in the $2n$ and $2n+1$ Nyquist zones compared to an NRZ DAC (Figure 3.14) [85]. MRZ is typically implemented using a per cell current RZ (IRZ) architecture, incurring more stringent timing requirements than RZ due to increased switching rate. This architecture has demonstrated synthesis frequencies up to f_0 =8GHz with f_{CLK}=2.7GHz, achieving better than 45dBc SFDR for a 12-bit DAC in 0.5μm InP [85].

3.3.3. Time-Interleaved (TI) DACs

Although mixing and MRZ DACs achieve direct RF synthesis, they rely on increased switching speeds. Conversely, TI combines n DACs, each operating at phase offsets ($2\pi/n$) at the clock rate (f_{CLK}), depicted in Figure 3.16(a), to suppress $2(n-1)$ image replica, thereby extending bandwidth up to the n^{th} Nyquist zone [89]. For example, if n=2, the first and second images are cancelled allowing signal generation up to f_{CLK}. In addition, since each DAC output signal is correlated and the noise is uncorrelated, interleaving offers benefits of resolution enhancement (Figure 3.14))[44]. However, the extent of image suppression depends on gain and clock phase matching among the n

[44] TI can be implemented with either NRZ or RZ topologies.

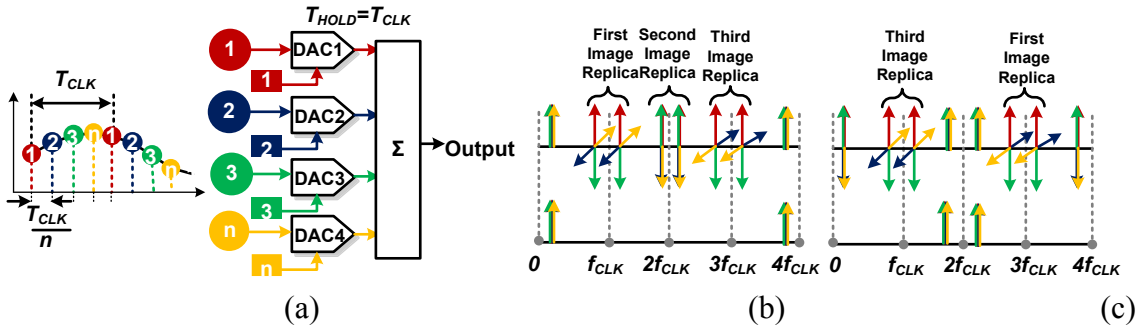


Figure 3.16: TI DAC. (a) Data- and hold-interleaved DAC architecture highlighting the sampled input, uniform clock phases, and the summed reconstructed output. (b). Image cancellation with 4-interleaved DACs for sub-Nyquist signal generation and (c) beyond Nyquist signal generation.

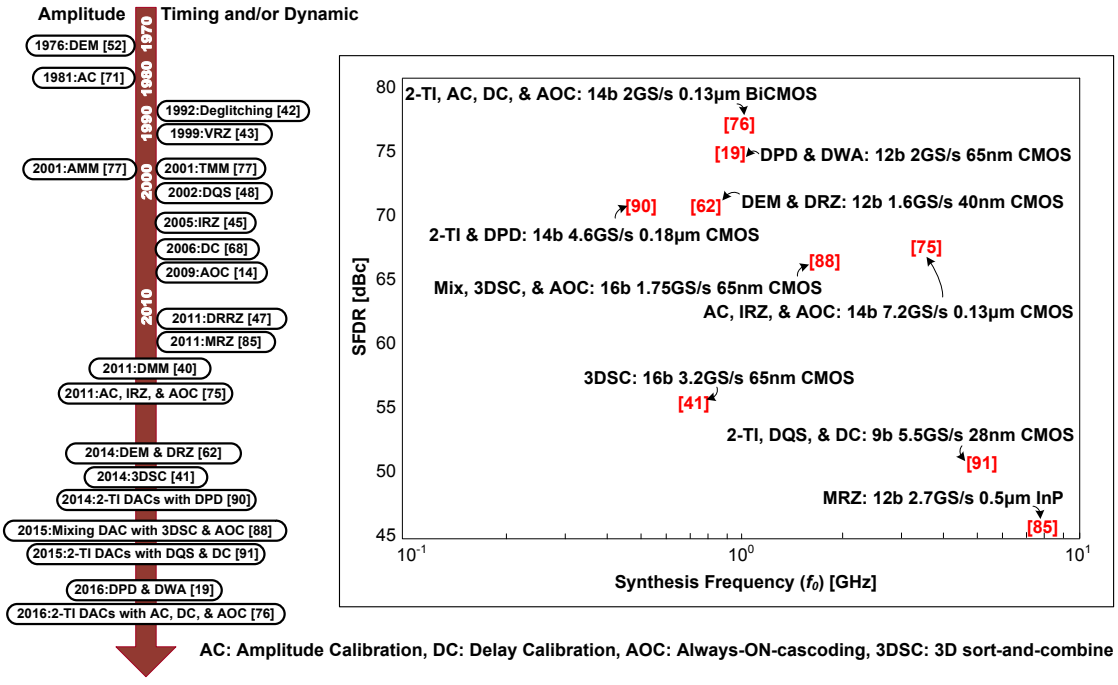
DACs [89]. It was shown for n =4 NRZ DACs that gain mismatch must be below 1% to achieve a signal-to-image rejection ratio (SIRR)>53dBc, while phase mismatch must be less than 0.5% to achieve an SIRR>46dBc [89]. This technique has demonstrated an SFDR of 70dBc (f_0 =500MHz, f_{CLK} = 4.6GHz) for two 14-bit TI DACs combined with DPD in 0.18µm CMOS [90]. The authors in [91], reported an SFDR of 50dBc (f_0 =4.6GHz, f_{CLK} =5.5GHz) for two 9-bit TI DACs with DQS and delay calibration[45] in 28nm CMOS SOI. More recently, time interleaving has been demonstrated in ΔΣ DACs using two 14-bit DACs with amplitude and delay calibration[46] to achive an SFDR of 76dBc (f_0 =997MHz, f_{CLK} =2GHz) in 130nm BiCMOS [76].

3.4. Conclusion and Motivation for Future Designs

Figure 3.17(a) shows the historical evolution of key calibration and compensation techniques, highlighting those that correct for amplitude errors (left) or timing/dynamic (right) errors. As shown, early attention was placed on correcting amplitude mismatch

[45] Between the two DACs.
[46] Amplitude and delay calibration performed on individual cell errors as well as delay calibration between the two DACs.

using techniques such as DEM, amplitude calibration, and AMM. Later on, timing and dynamic correction techniques emerged such as RZ, TMM, delay calibration, and always-ON-cascoding. Also highlighted in Figure 3.17(a), is the use of combined correction techniques along with enhanced DAC architectures, which have recently become popular. To highlight the performance of these combined techniques, Figure 3.17(b) displays the demonstrated SFDR vs. synthesis frequency (f_0) for references with f_{CLK} ≥1GHz. As illustrated, the benefit of technology scaling along with combinations of correction techniques and RF DAC architectures enables an SFDR>65dBc for GHz synthesis frequencies.



(a) (b)

Figure 3.17: (a) Historical overview of correction techniques. (b) Demonstrated SFDR vs. f_0 for various correction techniques with f_{CLK} ≥ 1GHz.

Chapter 4: Proposed Novel Timing Calibration Technique

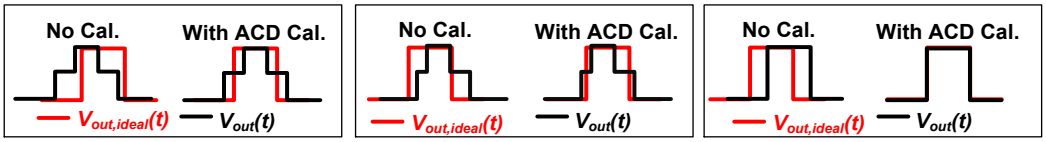
As operating frequency increases beyond several MHz, timing errors are detrimental to DAC performance, yet few novel timing calibration techniques have been developed and verified on chip. The goal of this chapter is to introduce the proposed new timing calibration technique, termed adaptive clock delay (ACD) calibration.

There are several things to consider when developing calibration suited to correct for timing mismatch. Since these techniques are fairly new and still in development, the practicality of such a system is based on several properties of the calibration circuit such as: 1. Resolution, 2. Area, and 3. Power. Each one is addressed below:

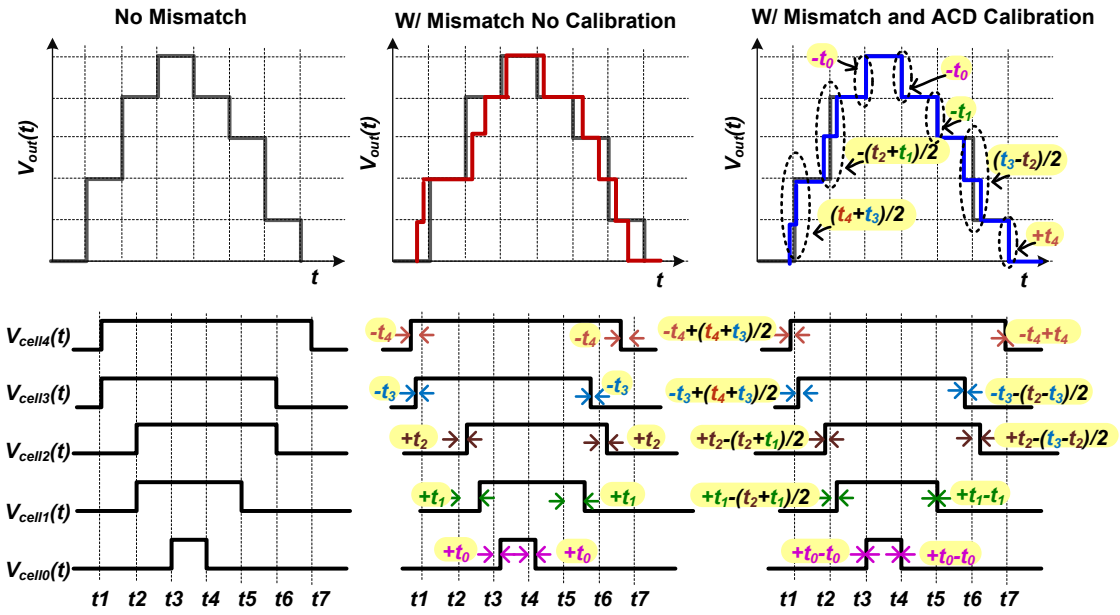
1. Calibration requires the ability to sense the error in question. As such, this requires a circuit which has the ability to detect timing mismatch on the order of sub-ps. While a phase detector might seem well suited, designing one with sub-ps resolution is non-trivial. The mismatch measurement circuit proposed in [40] must be used to attain such resolutions. In addition, the correction circuit used to fine-tune out the timing mismatches must have a small enough resolution while covering the range of mismatches given. Such circuits must be understood to determine what can and can't be implemented on chip.
2. The calibration circuit should not cause significant increase to the DAC core area. If it does, systematic errors can increase in both amplitude and time, hindering the calibration in question.

3. When adding a calibration circuit it is important that the power consumption of such a circuit not exceed near that of the DAC core. The calibration should be designed and architected in a careful manner to not surpass the power budget.

Adaptive clock delay (ACD) calibration is a new technique proposed here to address timing mismatches in the DAC unary cells. It works by modulating the clock signal which retimes the DAC data signals, thereby advancing or delaying the output response of the DAC. Consider a few simple examples, shown in Figure 4.1(a). Prior to calibration, the output pulse $V_{out}(t)$, appears advanced or delayed relative to the ideal



(a)



(b)

Figure 4.1: Concept of ACD calibration. (a) Single output pulse. (b) 5 unary cell example.

signal $V_{out,ideal}(t)$. ACD calibration applies an equivalent and opposite delay to the retiming clock signal so that the energy of $V_{out}(t)$ is centered around the expected pulse, $V_{out,ideal}(t)$. A more detailed example for 5-unary cells is shown in Figure 4.1(b). First, notice that each output pulse coming from the different DAC cells ($V_{cell_j}(t)$) are summed to form the DAC output $V_{out}(t)$. If no timing mismatch is present, each output transitions at the expected time indicated by $t1{:}\,t7$. Now consider that each cell has a unique static timing mismatch (Δt_j) associated with it, which causes the output pulses to switch at different times, corrupting the DAC output. ACD calibration applies an average delay compensation each clock cycle (i.e. at times $t1{:}\,t7$). The compensated delay is determined from each cell delay (Δt_j) and depends on the cells that are switching in that clock cycle, termed *select averaging*. For instance, at time $t1$, $cell_4$ and $cell_3$ switch. With the knowledge that $cell_4$ has a delay of $-t_4$ and $cell_3$ has a delay of $-t_3$, the pulses are both delayed by $(t_3 + t_4)/2$. The calibration continues in each clock cycle. In some instances, when only one cell is switched (at time: $t3$, $t4$, $t5$, and $t7$), the delay is completely removed and $V_{out}(t)$ lines up with $V_{out,ideal}(t)$. The implementation of such a system is discussed in detail in the following section.

4.1. Architecture and Calibration Procedure

The proposed ACD calibration technique involves several key blocks, as shown in Figure 4.2(a). The calibration has two modes of operation 1) read mode: in this phase the cell delay information is gathered while the DAC is not in use and 2) run mode: during normal operation of the DAC, a compensated delay is determined and applied through a

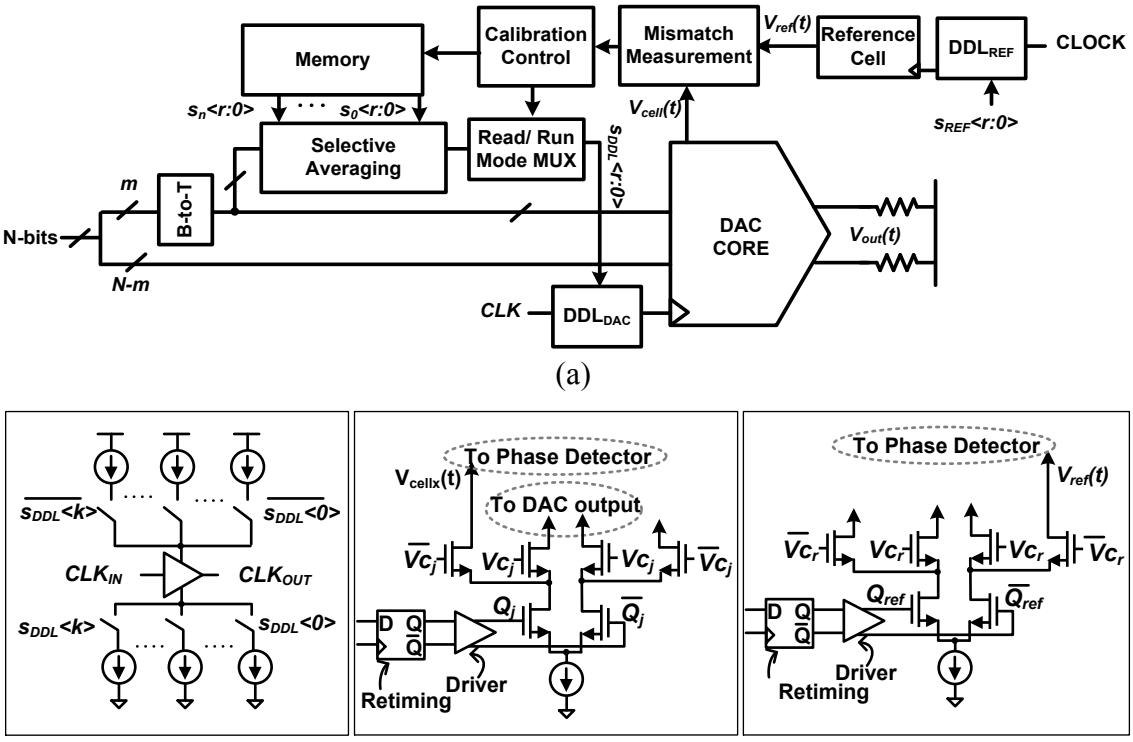


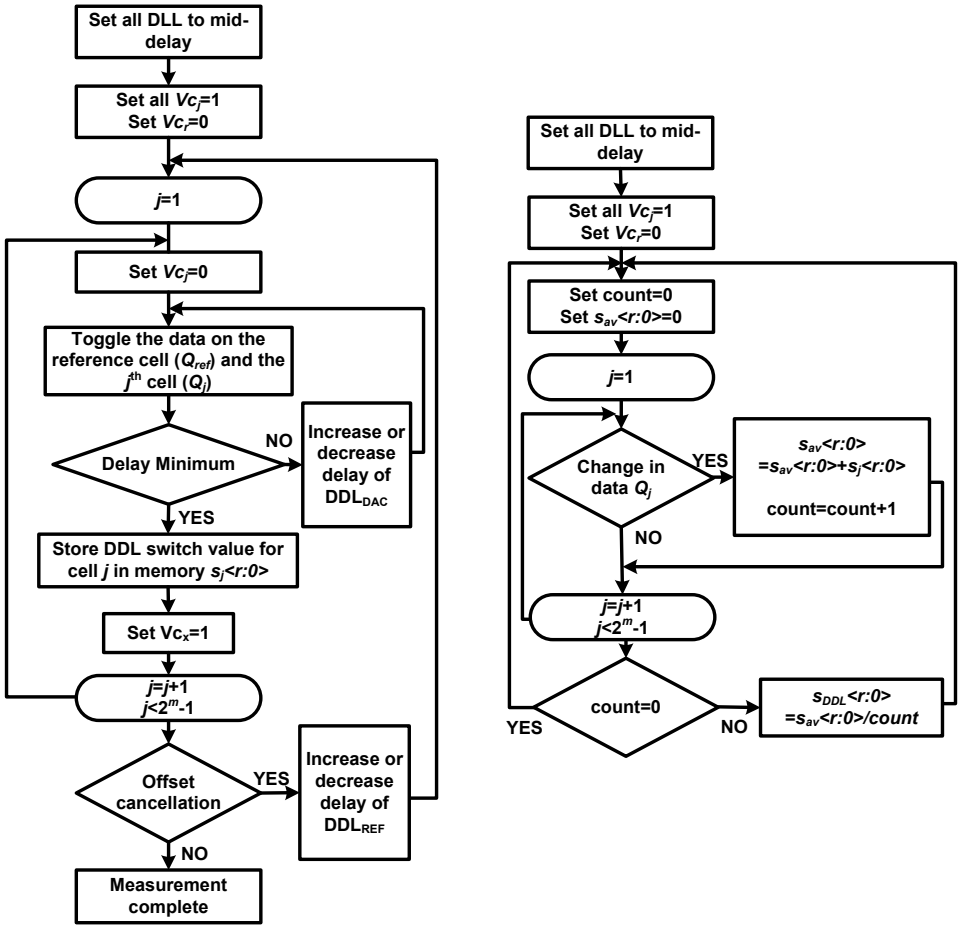
Figure 4.2: Adaptive clock delay calibration architecture. (a) Top level block diagram. (b) DDL circuit. (c) DAC cell. (d) Reference cell.

DDL[47] (Figure 4.2(b)) on the clock signal (CLK). Note that the only main change to the DAC core involves the use of measurement transistors (Figure 4.2(c)), used during read mode, to compare the cell signal $V_{cell_j}(t)$ with that of a reference signal $V_{ref}(t)$ (Figure 4.2(d)) using a mismatch measurement circuit. The rest of the blocks can be placed outside the DAC core, keeping its area small. The specific steps for the two calibration modes are discussed next.

During *read mode* the delay information per unary cell is gathered. Referring to Figure 4.2 and following Figure 4.3(a), read mode operates as follows:

[47] The DDL can be implemented as a current starved inverter whose delay is controlled by adjusting the amount of current through the inverter

- The DDL for the DAC (DDL_{DAC}) and the reference (DDL_{REF}) are set to mid-delay so that both positive and negative delay can be achieved
- The control signals for each cell, j, are set to Vc_j=1 and for the reference cell: Vc_r=0. This action routes the DAC signals to $V_{out}(t)$ and $V_{ref}(t)$ to the mismatch measurement circuit.
- Starting with the first cell j=1, Vc_j=0 so that the signal is now routed to the mismatch measurement circuit.
- The data on that cell (Q_j) and the reference cell (Q_{ref}) are toggled so that the phase difference between the cells can be measured.
- The delay of DDL_{DAC} is increased or decreased by controlling the switches $s_{DDL} < r: 0 >$, where r denotes the DDL resolution.
- The DDL_{DAC} delay continues to be increased/decreased until the measured phase difference between $V_{cell_j}(t)$ and $V_{ref}(t)$ changes direction and is minimal, or the DDL is out of tuning range, depicted in Figure 4.4(a).
- The corresponding switch value $s_j < k: 0 >$ for that cell, j, gets stored in memory. The control value is changed back to Vc_j=1 and the process is completed to the rest of the unary cells.
- After all the unary cell delay is measured, any major offsets between $V_{cell_j}(t)$ and $V_{ref}(t)$ can be removed by adjusting the delay on DDL_{REF} to ensure the measured values are all within range and the DDL_{DAC} is not saturated.



(a) (b)

Figure 4.3: Calibration algorithm (a) read mode and (b) run more

During *run mode* the delay of DDL_{DAC} is dynamically tuned at the clock rate, depending on the data (Q_j) that have transitioned from $0 \rightarrow 1$ or $1 \rightarrow 0$. Referring to Figure 4.2 and following Figure 4.3(b), *run mode* operates as follows:

- The DDL_{DAC} is set to mid-delay and the control signals Vc_j=1 and Vc_r=0, such that all the DAC cells are routed to $V_{out}(t)$ and $V_{ref}(t)$ is routed to the mismatch measurement circuit.

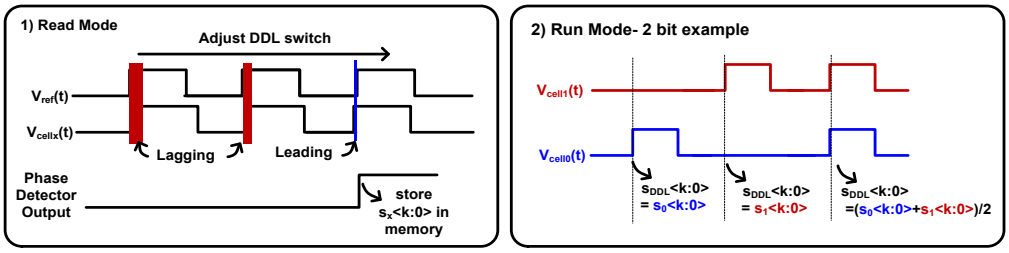


Figure 4.4: Timing diagram. (a) Read mode. (b) Run mode

- A value denoted as *count* along with a temporary switch value $s_{av} < r:0 >$ are set to 0.
- At each clock cycle it is determined which cell data has transitioned (does Q_j change from $0 \rightarrow 1$ or $1 \rightarrow 0$).
- If that cell j has transitioned, its corresponding switch value $s_j < r:0 >$ gets added to $s_{av} < r:0 >$.
- The variable $count$ keeps track of the number of cells that have switched within that clock cycle. If no cells have switched (i.e. *count*=0) then $s_{DDL} < r:0 >$ does not change. However if cells have transitioned than $s_{DDL} < r:0 > = s_{av} < r:0 > / count$. That is, the average delay of the cells that have transitioned gets compensated, termed *selective averaging*. The switch value $s_{DDL} < r:0 >$ continues to be updated at the clock rate.

Consider the case when there are only two cells $j = 0$ and $j = 1$ and thus two sets of switch values $s_0 < r:0 >$ and $s_1 < r:0 >$, as depicted in Figure 4.4(b). There are 3 possible values for $s_{DDL} < r:0 >$, they are:

1. If cell 0 transitions and cell 1 stays the same: $s_{DDL} < r:0 >= s_0 < r:0 >$.
2. If cell 1 transitions and cell 0 stays the same: $s_{DDL} < r:0 >= s_1 < r:0 >$.

3. If cell 0 and cell 1 transition: $s_{DDL} < r: 0 >= (s_0 < r: 0 > + s_1 < r: 0 >)/2$.

This averaging operation can be implemented using the block diagram shown in Figure 4.5. The *main blocks* consist of an r-bit adder, a divide by two, and MUX. For practicality, the operation is done in pairs, adding two switch values at a time and dividing by two (corresponding to the average switch value). Each of the individual switch values and the average are sent to a MUX to select one of the values depending on the cells that have transitioned. The *select enable* block, which consists of two flip-flops along with an XOR gate, can be used for each data th_j to determine if it has transitioned. To average more than two switch values, the *main blocks* are added in parallel and cascoded until one *main block* remains, forming a tree like structure. In the next section, behavioral simulations using the ACD technique are presented.

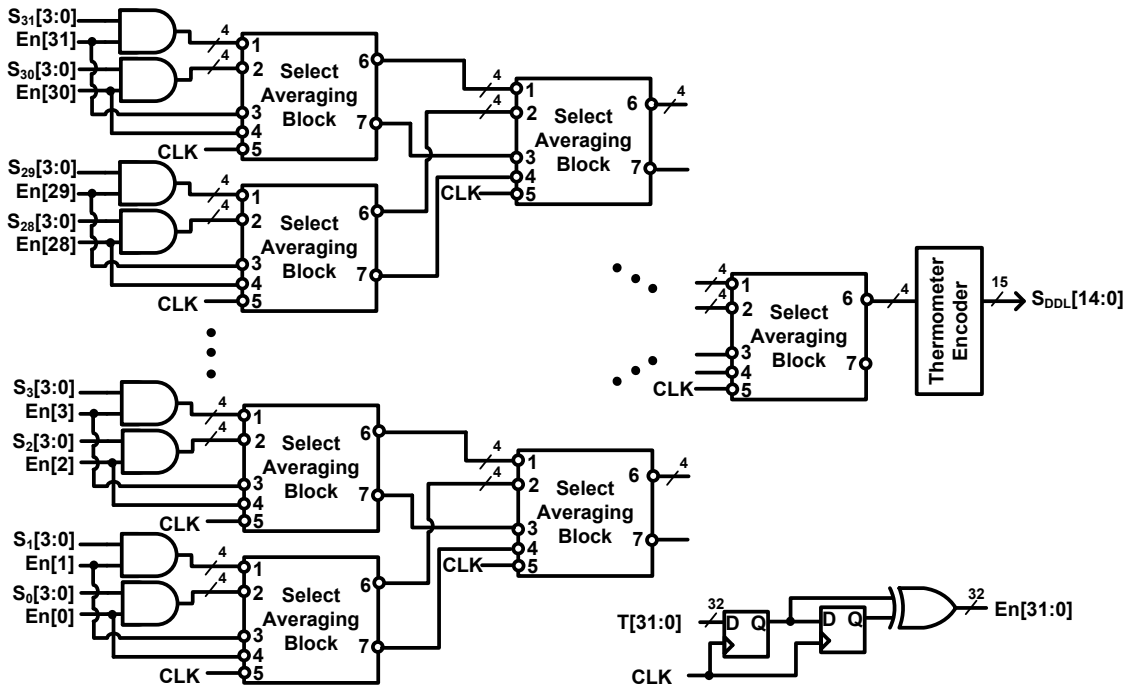


Figure 4.5: Circuit implementation of selective averaging

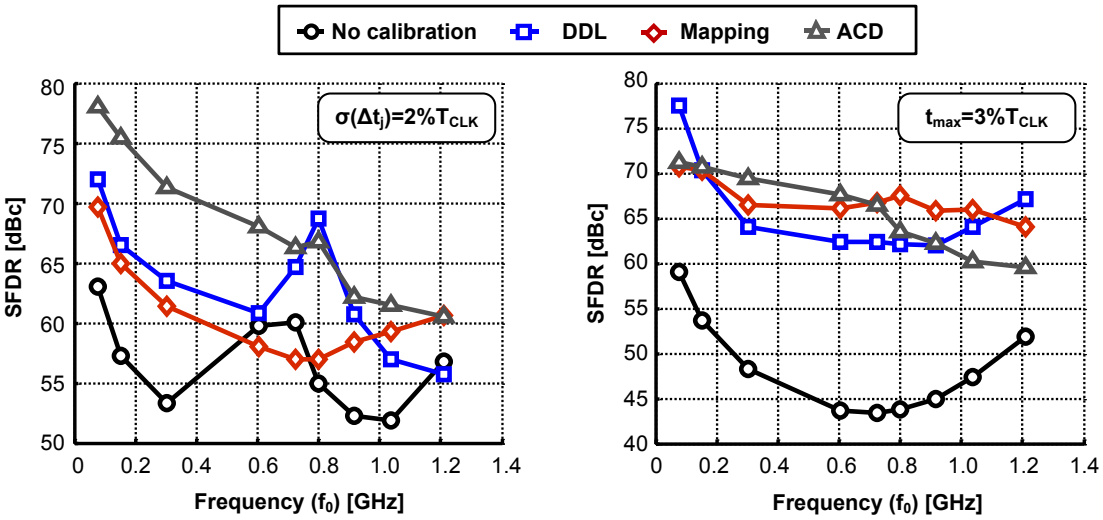
4.2. Behavioral Simulation and Comparison with Prior Art

For the purposed of simulation, the DAC model for a 10-bit 2.5GHz DAC segmented with 5-thermometer bits is designed in a 130nm BiCMOS process. Fixed delays in the DAC clock signal to the retiming latch are added to see the effects of both random and systematic timing delay mismatch. The proposed ACD technique is compared with two other timing calibration techniques: DDL calibration and time mapping [68]. It is assumed that there is zero measurement error for the measurement circuit. The DDL calibration uses a DDL per unary cell modeled using a Verilog-A DDL with a resolution of 800fs and range of 12ps (determined from realistic design). The time mapping algorithm is modeled and included in the Verilog-A binary-to-thermometer decoder. The ACD calibration is modeled using a Verilog-A DDL with a resolution of 800fs and range of 12ps applied to the CLK signal. Two studies are performed using 1) a single DAC architecture and 2) a TI DAC architecture.

4.2.1. Single DAC study

The single DAC design is simulated with timing mismatch and the resulting SFDR vs. frequency is shown in Figure 4.6. Two different timing mismatch distributions are used: a random timing mismatch with $\sigma(\Delta t_j) = 2\%T_{CLK}$ =8ps (Figure 4.6(a)) and a systematic timing mismatch with $t_{max} = 3\%T_{CLK}$ =12ps (Figure 4.6(b)). There are several observations to be made.

- DDL calibration: Errors are resolved to the resolution (800fs) and range (12ps) of the DDL. In the case of random distribution, several timing errors surpass the range, and the SFDR suffers.



(a) (b)

Figure 4.6: Simulated SFDR vs. frequency for single DAC with timing calibration. (a) Random timing error result. (b) Systematic timing error result.

- Time mapping: Errors are rearranged to provide the best switching sequence and therefore the improvement in SFDR is highly dependent on the starting error distribution profile. Mapping performs substantially better in the case of systematic errors when compared with the random case.
- ACD calibration: Although ACD calibration has the same range and resolution of the DDL calibration technique, (+) and (-) errors will cancel out in each clock cycle, allowing the ACD to compensate beyond that of the DDL calibration range. Therefore, ACD performs the best in the case of random errors. In the case of systematic errors, ACD's improvement in SFDR is close to the other techniques, except for near-Nyquist operation.

The ACD technique shows promising results in the case of the single DAC. In the next section the time correction techniques will be compared in the case of a TI architecture.

4.2.2. Time-Interleaved (TI) DAC Study

The TI DAC architecture using 2-DACs is studied. Each DAC operates at 2.5GS/s, 180 degrees out of phase from one another, and their outputs are combined to provide an effective operating frequency of 5GS/s. Time interleaving cancels the first image replica, allowing synthesis of input signal frequencies up to 2.5GHz. The issue with time-interleaving architectures is the strict matching requirements between the two DACs [89]. Timing mismatches have a substantial impact on not only the SFDR, but the signal-to-image-rejection-ratio (SIRR) as well. The simulation results for the calibrated dual TI DAC is shown in Figure 4.7. Two different timing mismatch distributions are used: a random timing mismatch with $\sigma(\Delta t_j) = 2\%T_{CLK}$ =8ps (Figure 4.7(a)) and a systematic timing mismatch with $t_{max} = 3\%T_{CLK}$ =12ps (Figure 4.7(b)). Like the single DAC case, the ACD calibration technique provides the best improvement in SFDR for the random error case. Also, the improvement in SFDR is similar for each of the techniques in the case of systematic errors. The most interesting observation is the impact each calibration has on the SIRR. For instance, mapping techniques offer no change in SIRR. While the DDL technique grants some improvement, the ACD technique offers the most gain in the SIRR.

4.2.3. Discussion

In general, each time calibration technique studied was shown to enhance the DAC SFDR and each one has its pros and cons. While the DDL calibration technique showed improvement in both SFDR and SIRR for the single and TI DAC, it requires a DDL per unary cell, leading to a larger DAC core area and more routing complexity. Furthermore,

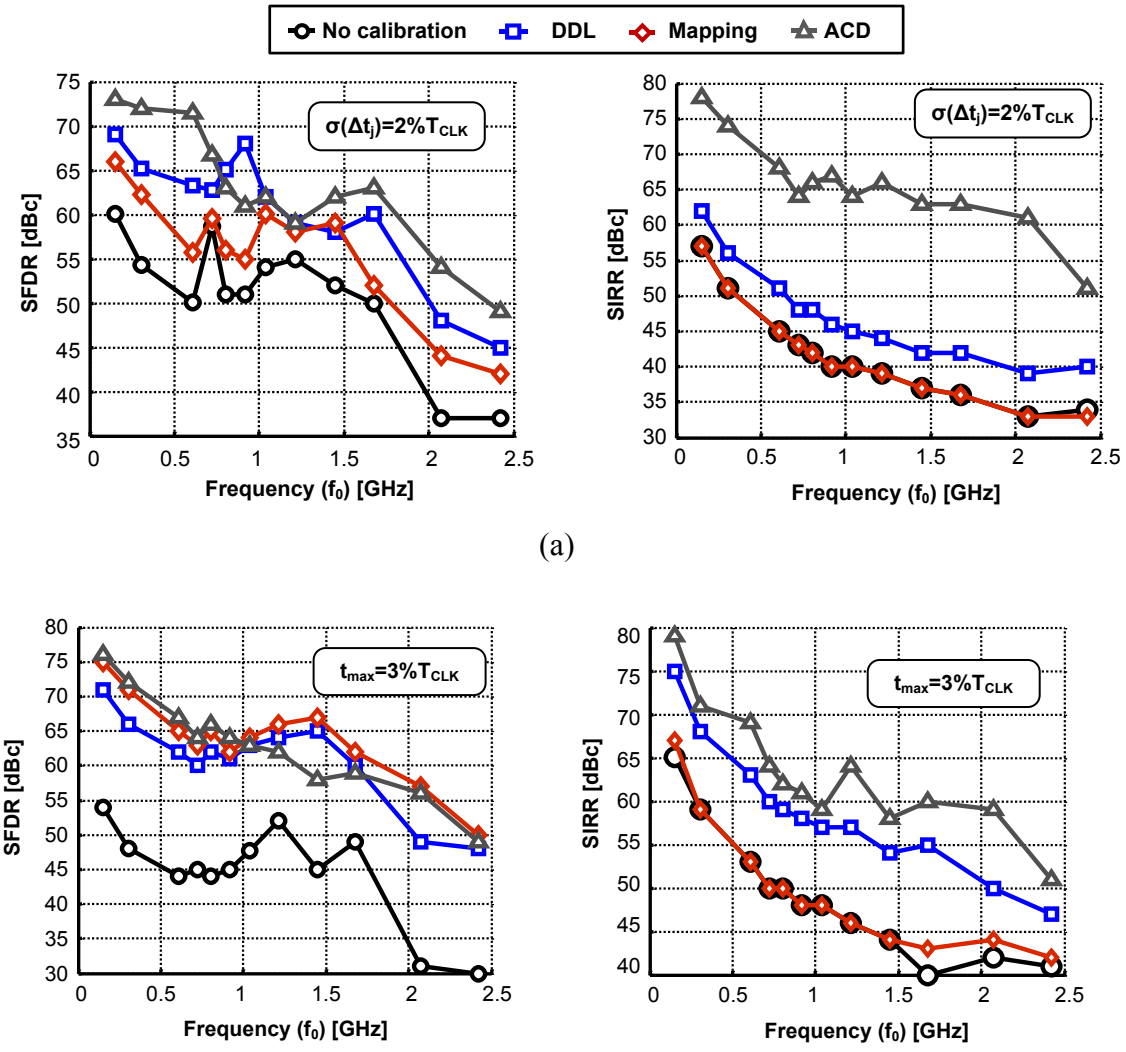


Figure 4.7: Calibration results for 2-TI DAC: SFDR vs. frequency (a) for random timing errors and (b) for systematic timing errors

each unary cell still has a fixed delay associated with it, related to the range and resolution of the DDL, limiting its performance. While mapping operates outside the DAC core, its improvement in SFDR is highly dependent on the error distribution profile. Moreover, it was shown that mapping offers no gain in SIRR in the TI DAC architecture. The proposed ACD calibration was shown to improve DAC SFDR with different error

profiles for a single DAC and TI DAC architecture. In addition, it also helps improve the SIRR in TI DACs. In most cases, it outperformed the DDL calibration and time mapping techniques. Like mapping, there is only a slight increase to the DAC core area due to the measurement transistors. The rest of the calibration blocks can be placed outside the DAC core, allowing for a compact design. However, since the ACD calibration technique operates at the clock rate, it creates some implementation issues. In the next section, a discussion on these issues and proposed solutions are provided.

4.3. Implementation Issues and Solutions

The DDL can be implemented as a current starved inverter with the delay controlled by adding or subtracting currents through the inverter. For a reasonable range it is found that 4-bits of DDL resolution is acceptable, equating to 15 different delay steps for a thermometer encoded DDL, providing 800fs resolution and 12ps range. This means each cell has a unique 4-bit DDL switch value. In order to average the switch values each add operation must be implemented with a 4-bit adder. For a 2.5GHz clock, this 4-bit adder takes approximately ½ a clock cycle to complete. For our design, there are 31 unary cells, meaning the first stage of summation logic will have 16 adds in parallel, followed by 8 adds in parallel, followed by 4, 2, then 1. That is, 5 stages of add operations totaling 1ns of computation time, must be completed within a clock cycle (400ps). There is no possible way the design from Figure 4.5 will work at the given clock frequency. There are several different options available to meet the clock rate, as depicted in Figure 4.8. They are: 1) pipelining 2) parallelization 3) grouping and 4) pre-computing.

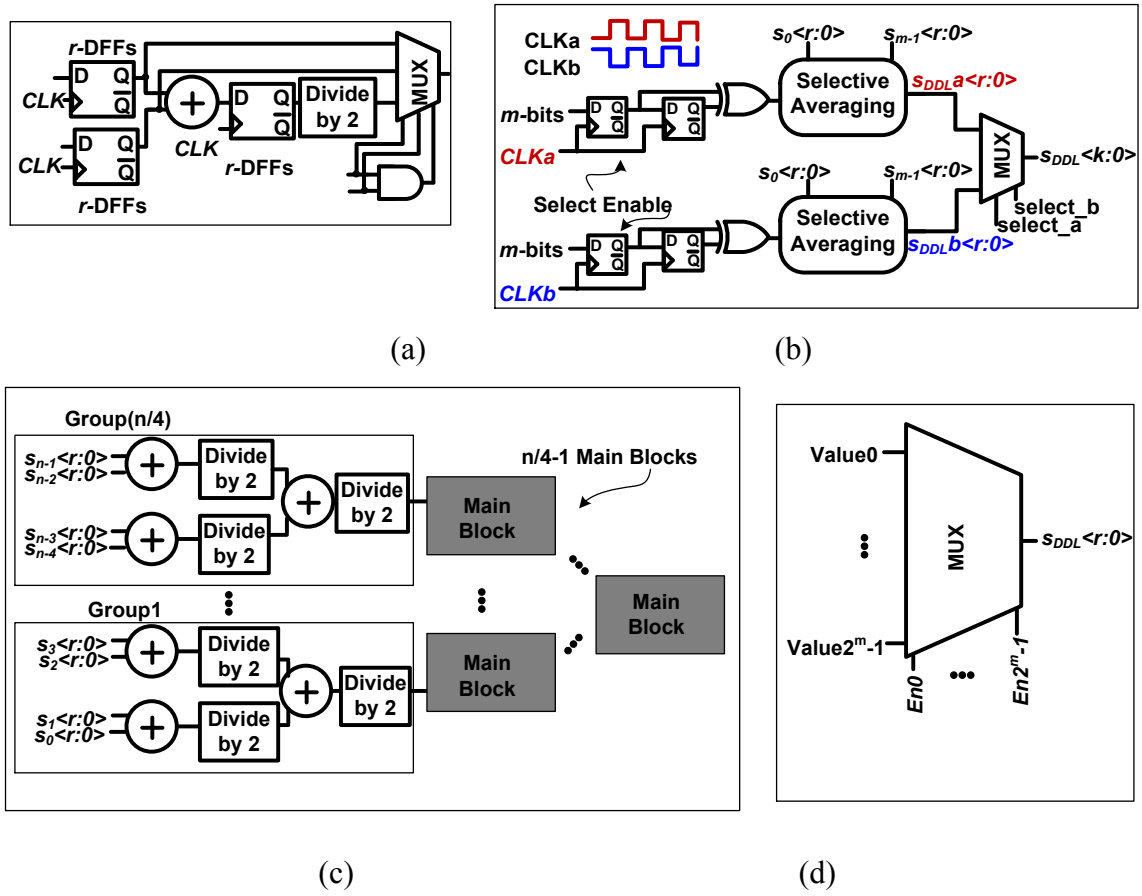


Figure 4.8: ACD calibration implementation solutions (a) pipelining (b) parallelization (c) grouping (d) pre-computing

Pipelining (Figure 4.8(a)) places a flip-flop before and after each add operation to keep the data in sync with the clock signal. For our design, this equates to 12 flip-flops per *main block*, totaling 372 flip-flops for the *select averaging* computation. This solution is not a viable choice as it will consume a large amount of power, as well as cause an increase in area and clock loading. Parallelization uses p *selective averaging* blocks in parallel each operating at f_{CLK}/p, shown in Figure 4.8(b) for p =2. The *select averaging* blocks each operates on different data points using two separate clock signals 180^{o} out of phase for the *select enable* blocks. Each output is then interleaved together using a MUX.

Although this doubles the number of *select averaging* blocks, it has a substantial reduction in power if it can eliminate the use of pipelining. However, for this design two stages may not be sufficient. On the other hand, a grouping algorithm can be used to assign the switch values to a subset, see Figure 4.8(c). Consider the case where we group cells with similar delay into groups of 4. Then we compute the average switch value of the group. This average value will get used in the *select averaging* algorithm. For groups of 4 this reduces the number of main block from j to $j/4-1$. For our example, this would reduce the number of cascoded add stages from 5 to 3 and number of flip-flops from 372 to 84. There is also the possibility of pre-computing every possible combination of average switch values and using one large MUX to select the correct value, as shown in Figure 4.8(d). However, for j=32 this would equate to $2^{32}-1 \simeq 4\times 10^{9}$ different possibilities for the MUX to handle. This is quite substantial and will make implementation of the MUX complicated. While anyone of these techniques may not fully address the issues at hand, combinations of them may offer the optimal solution in terms of area and power.

Chapter 5: Design of 14-bit 3GHz DAC in 130nm BiCMOS

A 14-bit DAC segmented with 5-thermometer bits operating at f_{CLK} =3GHz with amplitude and timing calibration is designed, fabricated, and tested in a 130 nm BiCMOS process. The goals of the design are to meet high-linearity across the operating range. The entire process was completed including: schematic design, layout, board design, assembly, and testing. The following sections discuss each design step and result.

5.1. Architecture

The DAC design consists of several circuits outlined by the block diagram in Figure 5.1. Starting with the data path, data is fed into the DAC through 7-interleaved 7Gb/s LVDS channels. Each channel has a delay control in order to fix any misalignment in data bits. After, the channels are demuxed to provide a total of 14-bits of data that can operate up to 3.5Gb/s. The data is then sent to the DAC which contains a binary-to-thermometer decoder to convert 5-thermometer bits, retiming flip-flop, data driver, DAC cells, and $R2R$ network for the binary cells. The DAC clock is fed in single ended and sent through a CML divide by two to provide both 0 and 180 degree clock phases. Other main circuit components consist of the calibration measurement and logic, calibration control clock, output LVDS to select channel data for channel alignment verification, and serial to parallel interface (SPI) for digital signal control on chip. The DAC core design, clock path, and calibration architecture is discussed in the following subsections.

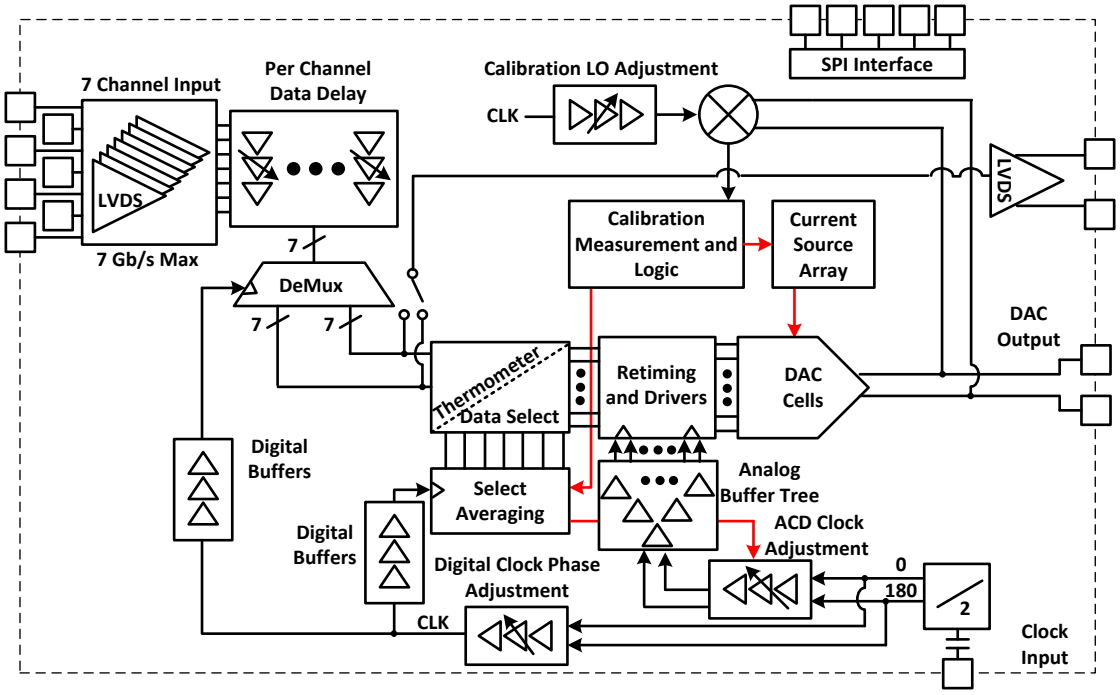


Figure 5.1: Top level DAC chip block diagram

5.1.1. DAC Core

The DAC core data path consist of two CML retiming D-flip-flops (DFF) and two CML drivers, outlined in Figure 5.2(a). The DFF architecture, shown in Figure 5.2(b), consists of a regenerative latch and active load. The first DFF is clocked with CLK_a[48] and converts the full scale (0-1.5V) single ended data into a differential low swing (1-1.5V) signal $(D, \overline{D})$. The second DFF is clocked with a highly matched clock (CLK_b) and remains in saturation and outputs a reduced swing (1.2-1.5V) signal $(Q, \bar{Q})$. The driver, shown in Figure 5.2(c), helps remove clock feedthrough and operates in saturation to produce a linear signal that will feed the DAC. The first driver steps up the output signal $(Q_{d1}, \overline{Q_{d1}})$ voltage (1.6-2V), while the second driver tunes the swing (1.8-2V) and

[48] Does not have strict timing requirements

transition time of the signal (Q_{d2}, $\overline{Q_{d2}}$) sent to the switch pair. The DAC cell, shown in Figure 5.2(d), is implemented with NPN transistors for their high output impedance and fast switching time. Always-ON-cascoding is utilized by bleeding a small current (I_{bleed}) through the top cascode transistors ($M_{4,5}$). In addition, the transistors ($M_{6,7}$) are added to

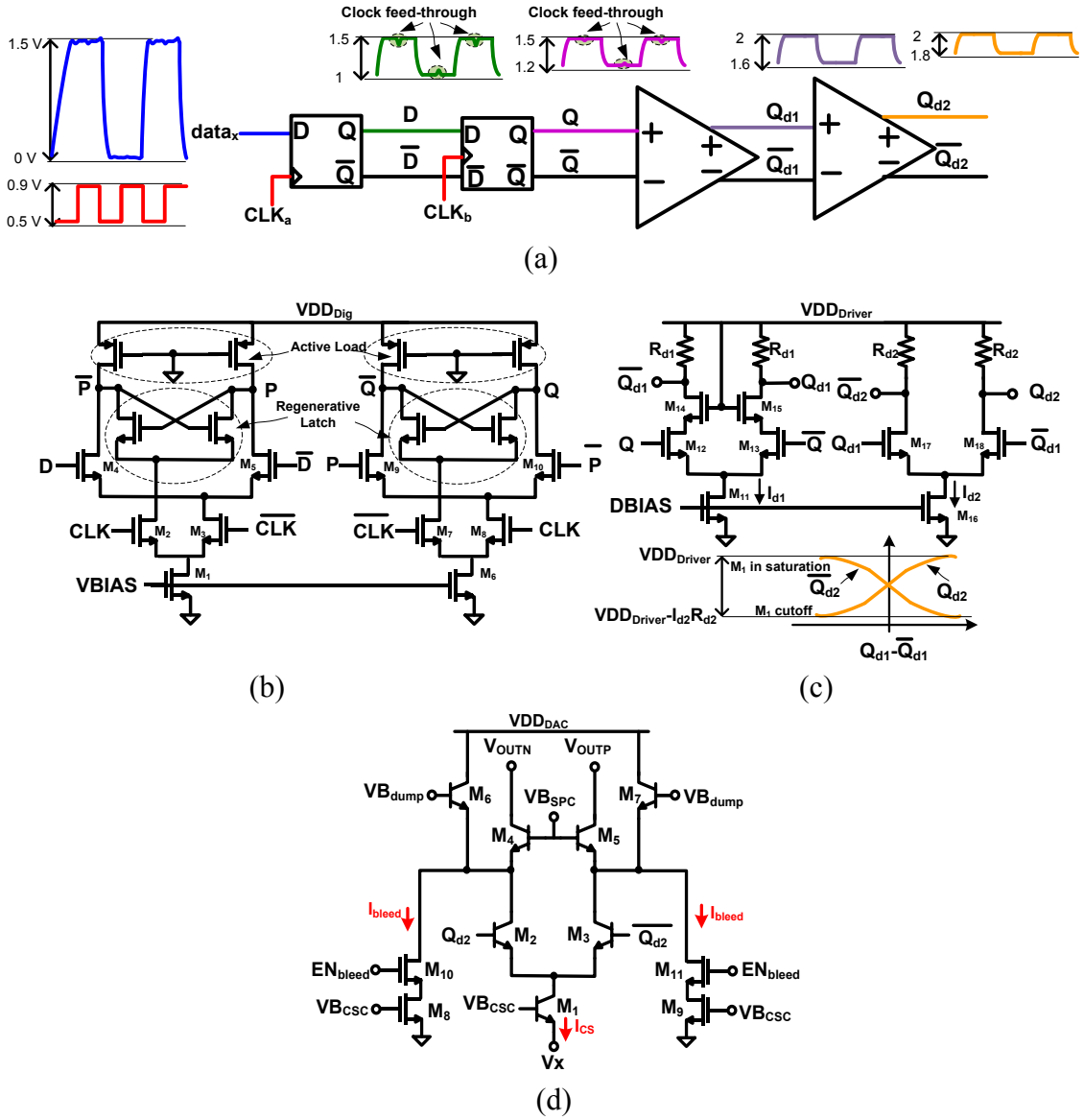


Figure 5.2: (a) DAC data path with retiming flip-flop and buffer. (b) CML retiming flip-flop. (c) CML driver. (d) DAC cell.

measure each cell current for calibration measurement. The cell current (I_{cs}) comes from a separate CMOS transistor current source array. Note that, the same exact cell size and (I_{cs}) are used for the binary and unary cells, where an $R2R$ is used to scale the binary current. This ensures each cell switches with the same speed.

5.1.2. Clock Path

The DAC clock is split to two separate paths, one analog the other digital. The analog path consists of a DDL, for fine clock adjustment, and 2 buffer trees, all of which are differential and operate within the CML saturation range for a highly linear clock. The first buffer tree layout is not carefully routed since it is used for CLK_a which controls the first set of DFF's. The second buffer tree layout is carefully routed and matched since it is used for CLK_b which retimes the data on the second set of DFF's. The digital clock path consists of a coarse and fine phase adjustment to align digital and analog clocks, along with digital buffers to feed the digital flip-flops at various areas on the chip. The digital clock tree is implemented such that all digital blocks switch at the same time by ensuring an equal number of buffers are placed between the starting clock signal and digital block.

5.1.3. Calibration

Amplitude cal-DAC calibration and ACD timing calibration are utilized on the unary DAC cells. The cal-DAC, shown in Figure 5.3, is placed in parallel with the current source (M_0) and consists of a 4-bit binary weighted current source with a minimum value (I_{res} =1.5μA) to ensure a calibrated SFDR >75dB and range of 25μA to cover the current

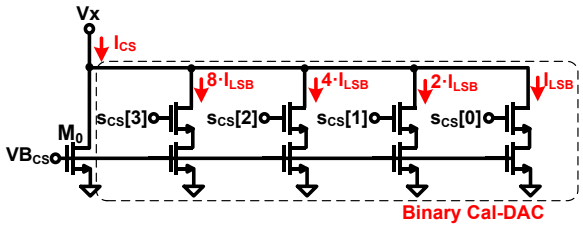


Figure 5.3: Current calibration 4-bit binary cal-DAC.

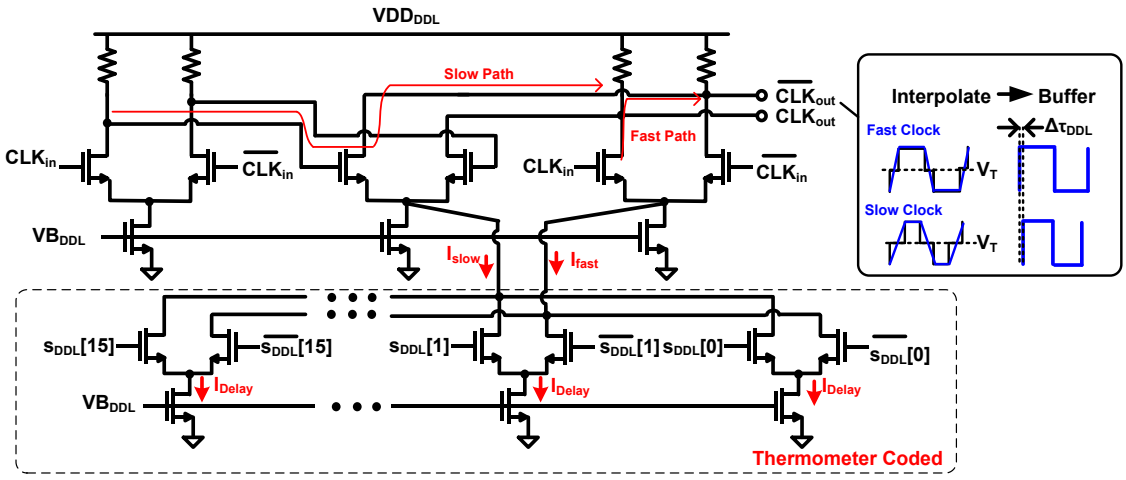
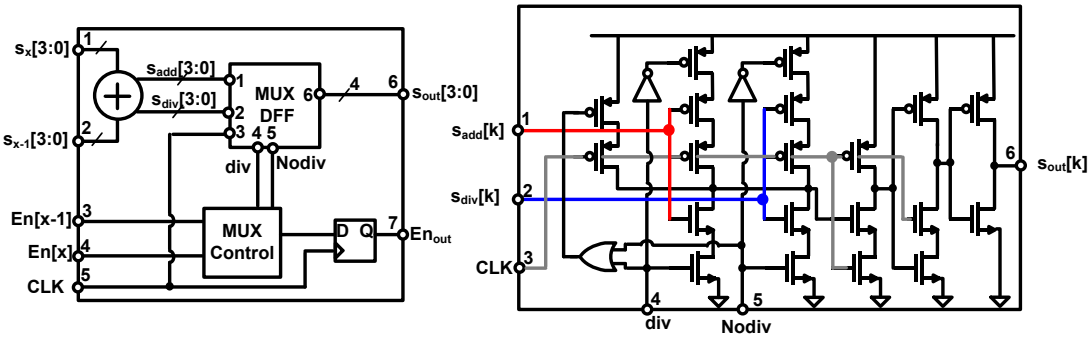


Figure 5.4: 4-bit DDL for timing adjustment.

mismatch spread. ACD calibration is implemented with two blocks: the DLL and the DDL control logic. As shown in Figure 5.4, the DDL is controlled by 16-thermometer bits ($s_{DDL}[0]$: $s_{DDL}[15]$) that set the current through the slow and fast paths thereby adjusting their delay [92]. The interpolated outputs of the combined slow and fast paths results in an equivalent delay from 333ps up to 5ps. The select averaging circuit (Figure 4.5) controls the DDL and is implemented with pipelined *main blocks*, shown in Figure 5.5(a), that consist of a carry-look-ahead (CLA) adder, MUX DFF, MUX control, and DFF. The CLA adder is utilized for its fast conversion speed. Its output provides the



(a) (b)

Figure 5.5: ACD timing calibration. (a) Select averaging block including a carry-look-ahead divide by two, MUX control, and combined DFF MUX. (b) DFF MUX.

added switch value (s_{add}[3:0]) and the divide by two value (s_{div}[3:0])[49]. The MUX DFF selects which value is selected and combines it with the DFF action so that minimal time is wasted. The select averaging is able to run at 3GHz extracted using these techniques.

5.2. Layout Design and Extracted Simulation

The DAC uses several layout techniques. For the power and ground lines: tree structures are used for equal *IR* drops and multiple large metal lines in parallel are used to reduce power and ground *IR* drops. For the clock signal: tree structures using small metal lines provide equal delay with minimal capacitive loading. For the output combining network: a tree structure with multiple metal lines with increasing width are used to provide equal *RC* delays from each signal cell to the output node. The extracted DAC SFDR results are shown in Figure 5.6 at 800MHz. Each array of elements are extracted separately to ensure one does not add too much error. The binary-to-thermometer (B2T) decoder was the easiest to layout and extracted results show minimal

[49] Shifting bits to right and putting 0 for MSB gives the divide by two value.

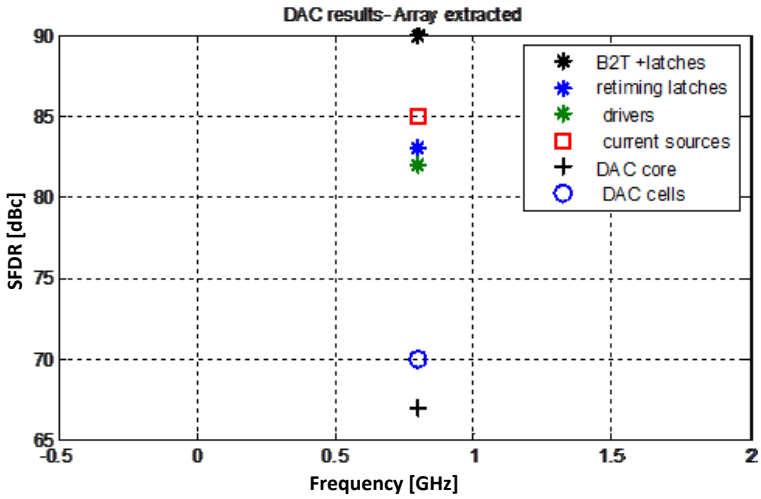


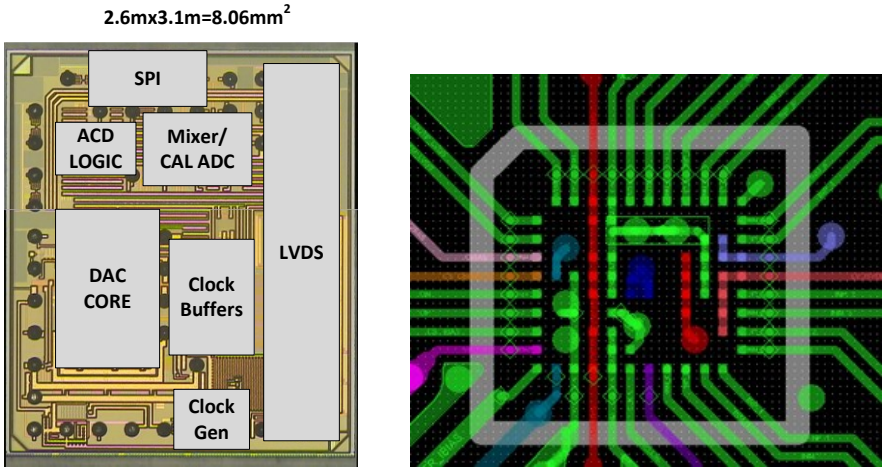
Figure 5.6: Extracted DAC SFDR

error (~90dBc). After several redesigns of the current source ground line, the current source array also contributes minimal error (~85dBc). The retiming latch array layout was iterated several times, specifically the ground line mismatch created timing errors, due to the CML architecture, which were minimized to provide ~83dBc SFDR. The driver array experience similar timing error caused by ground line mismatch and simulated ~80dBc SFDR. The DAC cell array, including the output combining network and $R2R$, was iterated several times, focusing on the output combining, and created the most error (~70dBc). The combined effect of all the arrays was simulated as the DAC core and resulted in ~65dBc SFDR. The residual amplitude and timing errors from the layout will be corrected by the calibration circuits.

5.3. Fabrication, Board Design, and Assembly

The fabricated chip with highlighted DAC areas is shown in Figure 5.7(a). The total chip area is 2.6mm×3.1mm, totaling 8.06mm^2. The chip is a flip-chip with C4 bumps

that will be bonded to the chip foot print located on the test board. The pad layout of the chip footprint is shown in Figure 5.7(b). Critical power and ground lines include a via close to each bump. 50 Ω matched lines are used for all the data signals, input and output, with SMA connectors at the ends. The on board output combining network and power and ground distribution is shown in Figure 5.8. The differential output of the DAC is sent to the bias-t with 50Ω load. Starting from the header pin board connection, a diode is placed between power and ground for ESD protection, then a series of decoupling



(a) (b)

Figure 5.7: (a) Chip micrograph. (b) Allegro PCB chip pad.

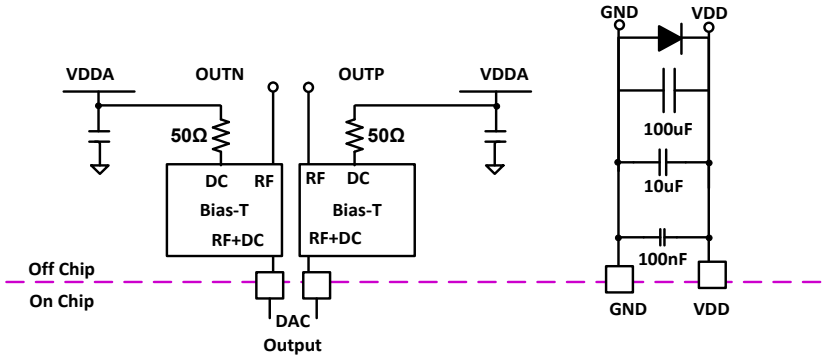
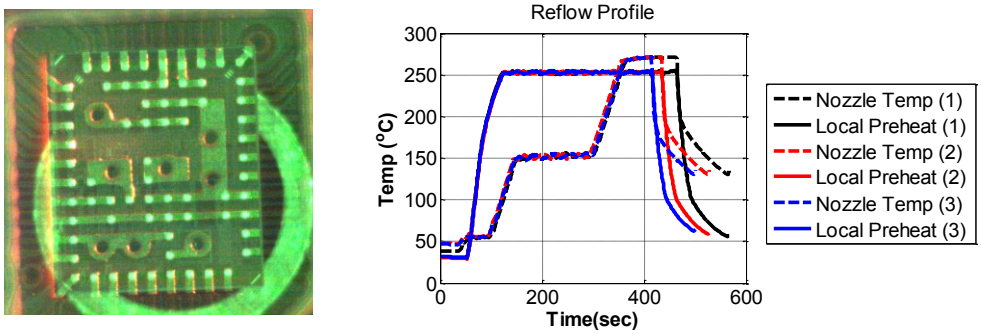


Figure 5.8: Allegro PCB schematic design highlighting of output network and power decoupling and protection.



(a) (b)

Figure 5.9: Flip-chip mounting. (a) Alignment. (b) Reflow profiles.

capacitors are added with the smallest one (100nF) placed as close to the chip as possible.

The fabricated PCB board was assembled by hand using a pick and place machine, reflow oven, and a flip-chip mounting machine. First a mask was used to apply solder paste to the board pads and the passive elements (i.e. resistors and capacitors) were placed using the pick and place and reflowed in the oven. Next the larger devices (potentiometers, header pins, SMA connectors) are set on the solder paste, tacked with epoxy, and sent through the reflow oven. Finally, the flip-chip is mounted by first applying flux to the chip bumps, aligning the flip-chip bumps to the PCB pad (Figure 5.9(a)), and then applying direct heat to reflow the bumps. A number of temperature profiles were used to reflow the chip, as shown in Figure 5.9(b). The temperature was backed off once the chip bumps were physically seen melting. During test, chips mounted with shortest time (blue) profile were defective.

5.4. Measurement Setup and Results

The DAC test setup is shown in Figure 5.10 and accomplished with the following items:

- DC power supplies provide power and ground connections to the header pins.
- A VC7215 Virtex 7 FPGA evaluation board was used to feed interleaved data samples into the SMA connected DAC input at double the data rate. DAC waveforms are created in MATLAB and loaded into the FPGA.
- The SMA connected DAC output is converted to a single ended output with a Picosecond Pulse Labs 5310A balun and measured using a PXA signal analyzer.
- Waveform viewer for time domain response.
- A signal generator provides the FPGA clock reference (-10dBm)

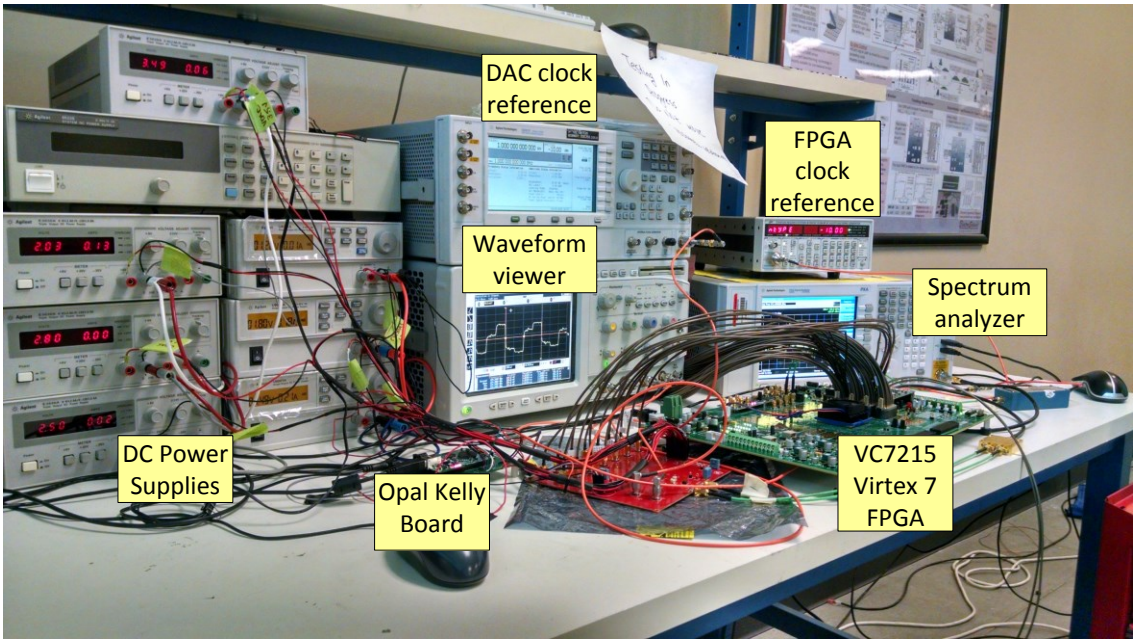
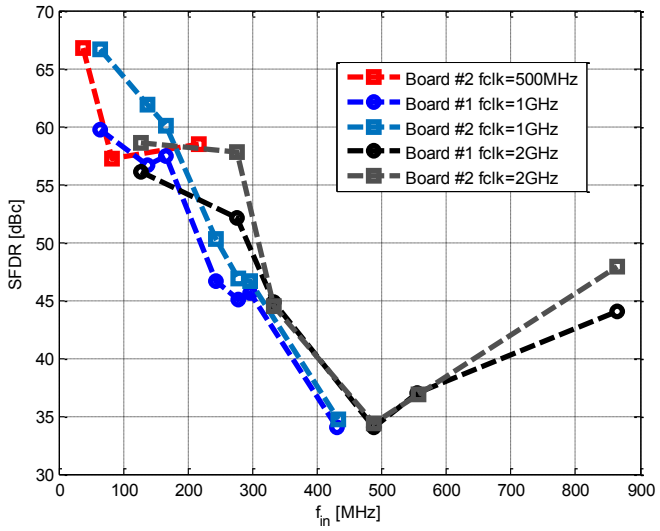


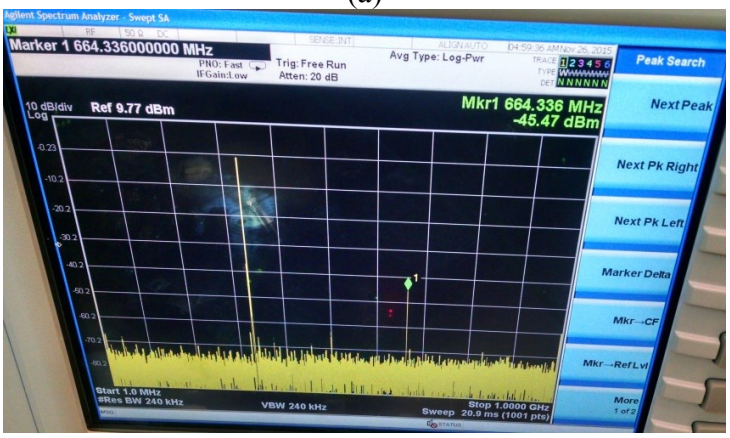
Figure 5.10: Test setup for the 14-bit DAC

- A second signal generator provides the DAC clock.
- SPI register control enabled by Spartan 6 FPGA on an Opal Kelly board (see Appendix B).
- Local computer for FPGA and SPI interfacing

The following tests were performed: DC, alignment, ramp, and AC sinusoid. Each test and result is outlined in detail in Appendix C. The measured SFDR vs. frequency for 2 separate boards at f_{CLK} =500MHZ, 1GHz, and 2GHz is shown in Figure 5.11(a). As shown, low frequency SFDR is decent at ~65dBc. As the input frequency increases, the SFDR falls off drastically (~-60dB/decade), regardless of the sample rate. Note that, the spectrum at these higher frequencies appears clean with a strong second order distortion, shown in Figure 5.11(b). This is a clear indication of strong timing mismatch present in the design. However, timing mismatch should make the SFDR fall off at -20dB/decade, not -60dB/decade. This error was never found to be related to the test setup and believed to be a design issue as outlined in Appendix D.



(a)



(b)

Figure 5.11: (a) measured SFDR vs. frequency for 2 separate boards at f_{CLK} =500MHZ, 1GHz, and 2GHz. (b) spectrum analyzer output for board #1 at f_{CLK} =2GHz and f_{in}=860MHz.

Chapter 6: Conclusion and Future Work

6.1. Work Summary and Conclusion

This work aimed to provide the reader with a holistic understanding of current-steering DACs including their error sources and methods used to improve their linearity in the presence of intrinsic and mismatch errors. An overview of the sources of DAC nonlinearity such as amplitude, timing, and settling errors, are detailed. As describe, amplitude errors limit the low frequency SFDR with flat distortion. On the other hand, timing and settling errors limited SFDR with frequency dependent distortion which increases with operating frequency. Using a new modeling methodology, these error sources are accurately modeled to provide a simulation speedup of ~330x per frequency point. This model is useful in evaluating the DAC performance in the presence of mismatch and observing the impact of calibration and compensation techniques. A historical overview of the correction techniques is provided, showing a lack in sufficient timing calibration methods. This has motivated the design of a new timing calibration technique, termed adaptive clock delay (ACD) calibration, which is detailed in this work. A 14-bit DAC is designed, fabricated, and tested to demonstrate the ACD technique. Although the ACD technique shows promising simulation results, the high frequency performance of the DAC was severely degraded, due to, what is believed to be a very large timing error in the DAC design that did not show up during simulation. Because of this the ACD technique was not validated on chip.

6.2. Future Work

The 14-bit DAC with ACD calibration has the potential to show great calibrated SFDR. Therefore, future work should focus on the redesign of the DAC. Changes that would be made include: 1) changing the analog clock paths (CLK_a) and (CLK_b) to come from the same clock tree and adding min delays between the analog DFF's so there is not potential to enter the metastable state of the analog DFF's which send data to the DAC switch pair, 2) spreading the digital clock signal spikes over time by ensuring all the digital DFF's are not activated at the same time, and 3) redesign of the power and ground networks to ensure large power droops do not occur in the presence of a non-ideal supply voltage. This design would also greatly benefit from a more advanced CMOS process which would allow the ACD calibration logic to perform much quicker, requiring less DFF's and power consumption. If ported to a smaller process node, switch time modulation, which causes intrinsic duty cycle errors, will most likely be present. The ACD technique could be adapted to correct for that code dependent timing mismatch as well. Once a successful implementation is verified for a single DAC case other avenues can be explored. For example: 1) ACD calibration can be modified to correct, not only delay mismatch errors, but duty cycle errors as well, and 2) ACD calibration can modified to work in a TI DAC.

Bibliography

[1] T. Instruments. [Online]. Available: http://www.ti.com/lsds/ti/data-converters/digital-to-analog-converter-products.page#p89=R-2R;String. [Accessed 2 2016].

[2] C.-Z. Dong, T.-J. Lu, Z.-M. Wang and Liang Zhou, "A multi-bit sigma-delta modulator and new DWA used in an audio DAC," *International Conference on Computer Technology and Development (ICCTD),* pp. 429-431, 2-4 Nov. 2010.

[3] D. Marche, Y. Savaria and Y. Gagnon, "Laser Fine-Tuneable Deep-Submicrometer CMOS 14-bit DAC," *IEEE Transactions on Circuits and Systems I,* vol. 55, no. 8, pp. 2157-2165, Sept. 2008.

[4] N. Ghittori, A. Vigna and P. Malcovati, "Analysis of the ideal SFDR limit for an N bit digital-to-analog converter," *IEEE International Conference on Electronics, Circuits and Systems,x(ICECS) ,* pp. 1-4, 11-14 Dec. 2005.

[5] J. Wikner and N. Tan, "Modeling of CMOS digital-to-analog converters for telecommunication," *IEEE Transactions on Circuits and Systems II,* vol. 46, no. 5, pp. 489-499, May 1999.

[6] T. Chen and G. Gielen, "The Analysis and Improvement of a Current-Steering DAC's Dynamic SFDR—II: The Output-Dependent Delay Differences," *IEEE Transactions on Circuits and Systems I,* vol. 54, no. 2, pp. 268-279, Feb. 2007.

[7] K. Doris, A. V. Roermond and D. Leenaerts, Wide-Bandwidth High Dynamic Range D/A Converters, Springer, 2006.

[8] M. Pelgrom, A. C. Duinmaijer and A. Welbers, "Matching properties of MOS transistors," *IEEE Journal of Solid-State Circuits,* vol. 24, no. 5, pp. 1433-1439, Oct. 1989.

[9] Y. Nakamura, T. Miki, A. Maeda, H. Kondoh and N. Yazawa, "A 10-b 70-MS/s CMOS D/A converter," *IEEE Journal of Solid-State Circuits,* vol. 26, no. 4, pp. 637-642, Apr 1991.

[10] J. Bastos, A. Marques, M. Steyaert and W. Sansen, "A 12-bit intrinsic accuracy high-speed CMOS DAC," *IEEE Journal of Solid-State Circuits,* vol. 33, no. 12, pp. 1959-1969, Dec 1998.

[11] T. Chen and G. Gielen, "The analysis and improvement of a current-steering DACs dynamic SFDR-I: the cell-dependent delay differences," *IEEE Transactions on Circuits and Systems I,* vol. 53, no. 1, pp. 3-15, Jan. 2006.

[12] A. Van Den Bosch, M. Steyaert and W. Sansen, "SFDR-bandwidth limitations for high speed high resolution current steering CMOS D/A converters," *The 6th IEEE International Conference on Electronics, Circuits and Systems,* vol. 3, pp. 1193-1196, 1999.

[13] C.-H. Lin and K. Bult, "A 10-b, 500-MSample/s CMOS DAC in 0.6 mm2," *IEEE Journal of Solid-State Circuits,* vol. 33, no. 12, pp. 1948-1958, Dec 1998.

[14] C.-H. Lin, F. van der Goes, J. Westra, J. Mulder, Y. Lin, E. Arslan, E. Ayranci, X. Liu and K. Bult, "A 12 bit 2.9 GS/s DAC With IM3 < - 60 dBc Beyond 1 GHz in 65 nm CMOS," *IEEE Journal of Solid-State Circuits,* vol. 44, no. 12, pp. 3285-3293, Dec 2009.

[15] K. Doris, J. Briaire, D. Leenaerts, M. Vertreg and A. van Roermund, "A 12b 500MS/s DAC with >70dB SFDR up to 120MHz in 0.18μm CMOS," *ISSCC,* pp. 116-588, 10 Feb. 2005.

[16] M. Sumathi, "Performance and analysis of CML Logic gates and latches," *International Symposium on Microwave, Antenna, Propagation and EMC Technologies for Wireless Communications,* pp.

1428-1432, 16-17 Aug. 2007.

[17] P. Heydari and R. Mohavavelu, "Design of ultra high-speed CMOS CML buffers and latches," *ISCAS,* pp. 208-211, 25-28 May 2003.

[18] T. Chen and G. Gielen, "The analysis and improvement of a current-steering DACs dynamic SFDR-I: the cell-dependent delay differences," *IEEE Transactions on Circuits and Systems I,* vol. 53, no. 1, pp. 3-15, Jan. 2006.

[19] S. Su, "12b 2GS/s Dual-Rate Hybrid DAC with Pulsed Timing-Error Pre-Distortion and In-Band Noise Cancellation Achieving >74dBc SFDR up to 1GHz in 65nm CMOS," *International Solid-State Ciruits Conference (ISSCC),* pp. 456-458, Jan. 31 2015- Feb. 4 2016.

[20] A. Van Den Bosch, M. Borremans, M. Steyaert and W. Sansen, "A 10-bit 1-GSample/s Nyquist current-steering CMOS D/A converter," *IEEE Journal of Solid-State Circuits,* vol. 36, no. 3, pp. 315-324, March 2001.

[21] T. Sai and Y. Sugimoto, "A 14-bit MOS DAC with current sources free from power-line voltage drop and with output circuits free from code-dependent variable time constant," *European Conference on Circuit Theory and Design, 2009. ECCTD 2009. ,* pp. 49-52, 23-27 Aug. 2009.

[22] M. Liu, Z. Zhu and Y. Yang, "A High-SFDR 14-bit 500 MS/s Current-Steering D/A Converter in 0.18um CMOS," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 23, no. 12, pp. 3148-3152, Dec 2015.

[23] G. Van Der Plas, J. Vandenbussche, W. Sansen, M. Steyaert and G. Gielen, "A 14-bit intrinsic accuracy Q2 random walk CMOS DAC," *IEEE Journal of Solid-State Circuits,* vol. 34, no. 12, pp. 1708-1718, Dec 1999.

[24] G. Gielen, P. De Wit, E. Maricau, J. Loeckx, J. Martin-Martinez, B. Kaczer, G. Groeseneken, R. Rodriguez and M. Nafria, "Emerging Yield and Reliability Challenges in Nanometer CMOS Technologies," *Design, Automation and Test in Europe (DATE),* pp. 1322-1327, March 2008.

[25] S. Yoder, S. Balasubramanian, W. Khalil and V. Patel, "Accuracy and speed limitations in DACs across CMOS process technologies," *IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS) ,* pp. 868-871, 4-7 Aug. 2013.

[26] S. Raman, T.-H. Chang, C. Dohrman and M. Rosker, "The DARPA COSMOS program: The convergence of InP and Silicon CMOS technologies for high-performance mixed-signal," *International Conference on Indium Phosphide & Related Materials (IPRM),* pp. 1-5, May 31 2010-June 4 2010.

[27] W. Khalil, J. Wilson, B. Dupaix, S. Balasubramanian and G. L. Creech, "Toward Millimeter-Wave DACs: Challenges and Opportunities," in *Compound Semiconductor Integrated Circuit Symposium (CSICS)*, 2012.

[28] B. Oyama, D. Ching, K. Thai, A. Gutierrez-Aitken and V. Patel, "InP HBT/Si CMOS-Based 13-b 1.33-Gsps Digital-to-Analog Converter With > 70-dB SFDR," *IEEE Journal of Solid-State Circuits,* vol. 48, no. 10, pp. 2265-2272, Oct. 2013.

[29] K. Lakshmikumar, R. Hadaway and M. Copeland, "Characterisation and modeling of mismatch in MOS transistors for precision analog design," *IEEE Journal of Solid-State Circuits,* vol. 21, no. 6, pp. 1057-1066, Dec. 1986.

[30] G. Radulov, M. Heydenreich, R. van der Hofstad, J. Hegt and A. van Roermund, "Brownian-Bridge-Based Statistical Analysis of the DAC INL Caused by Current Mismatch," *IEEE Transactions on Circuits and Systems II,* vol. 54, no. 2, pp. 146-150, Feb. 2007.

[31] H. Park and C.-K. K. Yang, "An INL Yield Model of the Digital-to-Analog Converter," *IEEE Transactions on Circuits and Systems I,* vol. 60, no. 3, pp. 582-592, March 2013.

[32] J. Vandenbussche, G. Van der Plas, G. Gielen and W. Sansen, "Behavioral model of reusable D/A converters," *IEEE Transactions on Circuits and Systems II,* vol. 46, no. 10, pp. 1323-1326, Oct. 1999.

[33] J. Doyle, Y. J. Lee and Y.-B. Kim, "An accurate DAC modeling technique based on wavelet theory,"

IEEE Custom Integrated Circuits Conference, 2003, pp. 257-260, Sept. 2003.

[34] M. Naoues, D. Morche, C. Dehos, R. Barrak and A. Ghazel, "Novel behavioral DAC modeling technique for WirelessHD system specification," *IEEE International Conference on Electronics, Circuits, and Systems,* pp. 543-546, 13-16 Dec. 2009.

[35] P. De Wit and G. Gielen, "Efficient simulation model for DAC dynamic properties," *IEEE International Symposium on Circuits and Systems (ISCAS),* pp. 2896-2899, June 2010.

[36] T. Chen and G. Gielen, "A 14-bit 200-MHz Current-Steering DAC With Switching-Sequence Post-Adjustment Calibration," *IEEE Journal of Solid-State Circuits ,* vol. 42, no. 11, pp. 2386-2394, Nov. 2007.

[37] K. Doris, C. Lin, D. Leenaerts and A. van Roermund, "D/A conversion: amplitude and time error mapping optimization," *IEEE International Conference on Electronics, Circuits and Systems (ICECS),* pp. 863-866, 2001.

[38] S. Yoder, S. Balasubramanian, V. J. Pate and W. Khalil., "Rapid and Parameterizable DAC Compiler User Guide," 2010. [Online]. Available: http://www.go.osu.edu/DACmodel.

[39] G. Radulov, P. Quinn, H. Hegt and A. van Roermund, "An on-chip self-calibration method for current mismatch in D/A converters," *Proceedings of the 31st European Solid-State Circuits Conference, ESSCIRC ,* pp. 169-172, 12-16 Sept. 2005.

[40] Y. Tang, J. Briaire, K. Doris, R. van Veldhoven, P. van Beek, H. Hegt and A. van Roermund, "A 14 bit 200 MS/s DAC With SFDR >78 dBc, IM3 < -83 dBc and NSD <-163 dBm/Hz Across the Whole Nyquist Band Enabled by Dynamic-Mismatch Mapping," *IEEE Journal of Solid-State Circuits,* vol. 46, no. 6, pp. 1371-1381, June 2011.

[41] H. Van de Vel, J. Briaire, C. Bastiaansen, P. Van Beek, G. Geelen, H. Gunnink, Y. Jin, M. Kaba, K. Luo, E. Paulus, B. Pham, W. Relyveld and P. Zijlstra, "11.7 A 240mW 16b 3.2GS/s DAC in 65nm CMOS with <-80dBc IM3 up to 600MHz," *IEEE International Solid-State Circuits Conference(ISSCC),* pp. 206-207, 9-13 Feb. 2014.

[42] P. Vorenkamp and J. Verdaasdonk, "Fully bipolar, 120-Msample/s 10-b track-and-hold circuit," *IEEE Journal of Solid-State Circuits,* vol. 27, no. 7, pp. 988-992, July 1992.

[43] A. R. Bugeja, B.-S. Song, P. L. Rakers and S. F. Gillig, "A 14-b, 100-MS/s CMOS DAC Designed for Spectral Performance," *IEEE Journal of Solid-State Circuits,* vol. 34, no. 12, pp. 1719-1732, 1999.

[44] Y. Tang, Smart and High-Performance Digital-to-Analog Converters with Dynamic-Mismatch Mapping, Eindhoven University of Technology, 2010.

[45] M.-J. Choe, K.-H. Baek and M. Teshome, "A 1.6-GS/s 12-bit return-to-zero GaAs RF DAC for multiple Nyquist operation," *IEEE Journal of Solid-State Circuits,* vol. 40, no. 12, pp. 2456-2469, 2005.

[46] I. Analog Devices, "AD9772: 14-Bit 150 MSPS Tx DAC with 2x Interpolation Filter," 1999.

[47] W.-H. Tseng, J.-T. Wu and Y.-C. Chu, "A CMOS 8-Bit 1.6-GS/s DAC With Digital Random Return-to-Zero," *IEEE Transactions on Circuits and Systems II: Express Briefs,* vol. 58, no. 1, pp. 1-5, Jan. 2011.

[48] S. Park, G. Kim, S.-C. Park and W. Kim, "A digital-to-analog converter based on differential-quad switching," *IEEE Journal of Solid-State Circuits, ,* vol. 37, no. 10, pp. 1335-1338, Oct. 2002.

[49] B. Schafferer and R. Adams, "A 3V CMOS 400mW 14b 1.4GS/s DAC for multi-carrier applications," *IEEE International Solid-State Circuits Conference (ISSCC),* vol. 1, pp. 360-532, 15-19 Feb. 2004.

[50] G. Engel, S. Kuo and S. Rose, "A 14b 3/6GHz current-steering RF DAC in 0.18µm CMOS with 66dB ACLR at 2.9GHz," *IEEE International Solid-State Circuits Conference (ISSCC),* pp. 458-460, 19-23 Feb. 2012.

[51] G. Radulov, P. Quinn and A. van Roermund, "A 28-nm CMOS 7-GS/s 6-bit DAC With DfT Clock and Memory Reaching SFDR>50 dB Up to 1 GHz," *IEEE Transactions on Very Large Scale*

Integration (VLSI) Systems, 2014.

[52] R. v. d. Plassche, "Dynamic element matching for high-accuracy monolithic D/A converters," *IEEE International Solid State Circuits Congerence,* pp. 149-149, 18-20 Feb. 1976.

[53] J. Z. a. I. G. K. L. Chan, "Dynamic Element Matching to Prevent Nonlinear Distortion From Pulse-Shape Mismatches in High-Resolution DACs," *IEEE Journal of Solid-State Circuit,* vol. 43, no. 9, pp. 2067-2078, Sept. 2008.

[54] I. G. a. P. Carbone, "A rigorous error analysis of D/A conversion with dynamic element matching," *IEEE Transactions on Circuits and Systems II,* vol. 42, no. 12, pp. 763-772, Dec. 1995.

[55] P. Stubberud and J. Bruce, "An analysis of dynamic element matching flash digital-to-analog converters," *IEEE Transactions on Circuits and Systems II,* vol. 48, no. 2, pp. 205-213, Feb. 2001.

[56] I. Galton, "Why Dynamic-Element-Matching DACs Work," *IEEE Transactions on Circuits and Systems II,* vol. 57, no. 2, pp. 69-74, Feb. 2010.

[57] R. Baird and T. Fiez, "Linearity enhancement of multibit ΔΣ A/D and D/A converters using data weighted averaging," *IEEE Transactions on Circuits and Systems II,* vol. 24, no. 12, pp. 753-762, Dec. 1995.

[58] J. H. W. S. Rudy J. van de Plassche, Analog Circuit Design: High-Speed Analog-to-Digital Converters, Mixed Signal Design; PLLs and Synthesizers, New York: Springer Science & Business Media, 2000.

[59] M. K. a. E. G. F. A. Lavzin, "A higher-order mismatch-shaping method for multi-bit Sigma-Delta Modulators," *IEEE International SOC Conference,* pp. 267-270, 2008.

[60] G. Manganaro, Advanced Data Converters, New York: Cambridge University Press, 2012.

[61] J. Liu, X. Li, Q. Wei and H. Yang, "A 14-Bit 1.0-GS/s Dynamic Element Matching DAC with >80 dB SFDR up to the Nyquist," *IEEE International Symposium on Circuits and Systems ISCAS,* pp. 1026-1029, 24-27 May 2015.

[62] W.-T. Lin, H.-Y. Huang and T.-H. Kuo, "A 12-bit 40 nm DAC Achieving SFDR > 70 dB at 1.6 GS/s and IMD < –61dB at 2.8 GS/s With DEMDRZ Technique," *IEEE Journal of Solid-State Circuits,* vol. 49, no. 3, pp. 708-717, March 2014.

[63] Y. Cong and R. Geiger, "A 1.5-V 14-bit 100-MS/s self-calibrated DAC," *IEEE Journal of Solid-State Circuits,* vol. 38, no. 12, pp. 2051-2060, Dec. 2003.

[64] Y. Tang, H. Hegt, A. van Roermund, K. Doris and J. Briaire, "Statistical Analysis of Mapping Technique for Timing Error Correction in Current-Steering DACs," *IEEE International Symposium on Circuits and Systems (ISCAS) ,* pp. 1225-1228, 27-28 May 2007.

[65] H.-H. Chen, J. Lee, J. Weiner, Y.-K. Chen and J.-T. Chen, "A 14-b 150 MS/s CMOS DAC with Digital Background Calibration," *Symposium on VLSI Circuits,* pp. 51-52, 2006.

[66] M. Tiilikainen, "A 1.8V 20mW 1mm2 14b 100Msample/s CMOS DAC," *Proceedings of the 26rd European Solid-State Circuits Conference, ESSCIRC,* pp. 435-438, Sept. 2000.

[67] S. S. a. S. Banerjee, "500 MHz differential latched current comparator for calibration of current steering DAC," *IEEE Students' Technology Symposium (TechSym),* pp. 309-312, 2014.

[68] Y. Tang, H. Hegt and A. van Roermund, "DDL-based calibration techniques for timing errors in current-steering DACs," *IEEE International Symposium onCircuits and Systems, ISCAS ,* p. 4, 21-24 May 2006.

[69] Y. T. H. H. a. A. v. R. E. Bechthum, "Timing error measurement for highly linear wideband Digital to Analog Converters," *IEEE International Symposium on Circuits and Systems (ISCAS),* pp. 2019-2022, 2011.

[70] T. Zeng and Degang Chen, "New calibration technique for current-steering DACs," *IEEE International Symposium on Circuits and Systems (ISCAS),* pp. 573-576, May 30 2010- June 2 2010.

[71] K. Maio, M. Hotta, N. Yokozawa, M. Nagata, K. Kaneko and T. Iwasaki, "An untrimmed D/A

converter with 14-bit resolution," *IEEE Journal of Solid-State Circuits,* vol. 16, no. 6, pp. 616-621, Dec. 1981.

[72] D. Groeneveld, H. Schouwenaars, H. Termeer and C. Bastiaansen, "A self-calibration technique for monolithic high-resolution D/A converters," *IEEE Journal of Solid-State Circuits,* vol. 24, no. 6, pp. 1517-1522, Dec 1989.

[73] A. Bugeja and B.-S. Song, "A self-trimming 14b 100MSample/s CMOS DAC," *IEEE International Solid-State Circuits Conference, ISSCC,* pp. 44-45, Feb. 2000.

[74] G.I.Radulov, P.J.Quinn, J. Hegt and A. v. Roermund, "A SELF-CALIBRATING CURRENT-STEERING 12-BIT DAC BASED ON NEW 1-BIT SELF-TEST SCHEME," *IEEE IC Test Workshop,* 13-14 Sept. 2004.

[75] K. Poulton, B. Jewett and J. Liu, "A 7.2-GSa/s, 14-bit or 12-GSa/s, 12-bit DAC in a 165-GHz fT BiCMOS process," *Symposium on VLSI Circuits (VLSIC),* pp. 62-63, 15-17 June 2011.

[76] J. J. McCue, B. Dupaix, L. Duncan, B. Mathieu, S. McDonnell, V. J. Patel, T. Quach and W. Khalil, "A Time-Interleaved Multimode $DeltaSigma$ RF-DAC for Direct Digital-to-RF Synthesis," *IEEE Journal of Solid-State Circuits,* pp. 1-16, 2016.

[77] K. Doris, C. Lin, D. Leenaerts and A. van Roermund, "D/A conversion: amplitude and time error mapping optimization," *The 8th IEEE International Conference on Electronics, Circuits and Systems, ICECS ,* vol. 2, pp. 863-866, 2001.

[78] T. C. a. G. Gielen, "A 14-bit 200-MHz Current-Steering DAC with Switching Sequence Post-Adjustment Calibration," *IEEE Asian Solid-State Circuits Conference (ASSCC) ,* pp. 347-350, 2006.

[79] T.-j. Lin and H. Samueli, "A 200-MHz CMOS x/sin(x) digital filter for compensating D/A converter frequency response distortion in high-speed communication systems," *IEEE GLOBECOM,* pp. 1722-1726, 2-5 Dec. 1990.

[80] B. Razavi, "The Future of Radios," *IEEE International Conferences on Circuits and Systems (ISCAS),* pp. 1-8, 24-27 May 2015.

[81] S. Balasubramanian, S. Boumaiza, H. Sarbishaei, T. Quach, P. Orlando, J. Volakis, G. Creech, J. Wilson and W. Khalil, "Ultimate Transmission," *IEEE Microwave Magazine,* vol. 13, no. 1, pp. 64-82, Jan.-Feb. 2012.

[82] S. Balasubramanian and W. Khalil, "Architectural trends in GHz speed DAC," *NORCHIP,* pp. 1-4, 12-13 Nov. 2012.

[83] S. Luschas, R. Schreier and H.-S. Lee, "Radio Frequency Digital-to-Analog Converter," *IEEE Journal of Solid-State Circuits,* vol. 39, no. 9, pp. 1462-1467, 2004.

[84] T. C. B. B. a. S. K. S. M. Taleie, "A Linear Sigma-Delta Digital IF to RF DAC Transmitter With Embedded Mixer," *IEEE Transactions on Microwave Theory and Techniques,* vol. 56, no. 5, pp. 1059-1068, May 2008.

[85] M.-J. Choe, K.-J. Lee, M. Seo and M. Teshome, "DC - 10GHz RF Digital to Analog Converter," in *Compound Semiconductor Integrated Circuit Symposium (CSICS)*, 2011.

[86] E. Bechthum, G. Radulov, J. Briaire, G. Geelen and A. van Roermund, "Systematic analysis of the impact of mixing locality on Mixing-DAC linearity for multicarrier GSM," *IEEE International Symposium on Circuits and Systems (ISCAS),* pp. 241-244, 20-23 May 2012.

[87] G. R. J. B. G. G. a. A. v. R. E. Bechthum, "A novel timing-error based approach for high speed highly linear Mixing-DAC architectures," *IEEE International Symposium on Circuits and Systems (ISCAS),* pp. 942-945, 2014.

[88] G. R. J. B. G. G. a. A. v. R. E. Bechthum, "9.6 A 5.3GHz 16b 1.75GS/S wideband RF Mixing-DAC achieving IMD<-82dBc up to 1.9GHz," *IEEE International Solid-State Circuits Conference (ISSCC),* pp. 1-3, 2015.

[89] S. Balasubramanian, G. Creech, J. Wilson, S. Yoder, J. McCue, M. Verhelst and W. Khalil, "Systematic Analysis of Interleaved Digital-to-Analog Converters," *IEEE Transactions on Circuits*

and Systems II: Express Briefs, vol. 58, no. 12, pp. 882-886, Dec 2011.

[90] B. Brandt, D. McMahill, M. Wu, P. Kalthoff, A. Kuckreja and G. Ostrem, "22.7 A 14b 4.6GS/s RF DAC in 0.18µm CMOS for cable head-end systems," *IEEE International Solid-State Circuits Conference (ISSCC),* pp. 390-391, 9-13 Feb. 2014.

[91] E. Olieman, A.-J. Annema and B. Nauta, "An Interleaved Full Nyquist High-Speed DAC Technique," *IEEE Journal of Solid-State Circuits,* vol. 50, no. 3, pp. 704-713, March 2015.

[92] L. Der and B. Razavi, "A 2-GHz CMOS image-reject receiver with LMS calibration," *IEEE Journal of Solid-State Circuits,* vol. 38, no. 2, pp. 167-175, Feb. 2003.

Appendix A: Statistical Modeling MATLAB and Verilog-A Code

The Verilog-A retiming flip-flop and driver code are shown below. The input, output, and parameter values are explained in Table A.1.

```
// Verilog-A retiming flip-flop and driver code
`include "constants.vams"
`include "disciplines.vams"
module dff(d,clk,dt,dd,q,qbar,vdd,vss);
input d,clk,dt,dd;
output q, qbar;
inout vdd,vss;
voltage d,clk,dt,dd,q,qbar,vdd,vss,tdrf;
parameter real    static_delay=800p,
            tcross=10p,
            tqr=20p,
            tqf=20p,
            tqbr=20p,
            tqbf=20p;
integer x;
real vmid,delay,duty,td,tq,tqb;
analog begin
@(initial_step) begin
     x=1;
     vmid=1;
     delay=V(dt)*1e-12;
     duty=V(dd)*1e-12;
end
@(cross(V(clk)-0.6,1))begin
     x= (V(d)>0.6);
end
if (x==1) begin
     tq=delay;
     tqb=tcross+delay;
end
else begin
     tq=tcross+duty;
     tqb=duty;
end
V(q) <+ transition(V(vdd)*x+V(vss)*!x,tq+static_delay,tqr,tqf);
V(qbar) <+ transition(V(vdd)*!x+V(vss)*x,tqb+static_delay,tqbr,tqbf);
end
endmodule
```

Table A.1: Verilog-A retiming driver code value and parameter definition.

d	Input data signal
clk	Input clock signal
dt	Input clock delay value $\Delta t_{CLK_{k,j}}$
dd	Input duty cycle value $\Delta d_{k,j}$
q	Output –positive
qbar	Output – negative
vdd	Supply- positive
vss	Supply- negative
static_delay	Static delay for (+) and (-) delay values dt, dd
tcross	Cross value delay
tqr	Rise time- positive
tqf	Fall time – positive
tqbr	Rise time - negative
tqbf	Fall time – negative

The Verilog-A mismatch file that feeds the mismatch values ($\Delta t_{CLK_{k,j}}, \Delta d_{k,j}, \Delta I_{k,j}$) to each cell is given below. It uses the built in function `$fscan`, to read the mismatch values from a .txt file called "param.txt", setting the current mismatch (`di[x]`=ΔI_j), timing delay mismatch (`dt[x]` = Δt_{CLK_j}), and duty cycle mismatch (`dd[x]` = Δd_j) per cell.

```
// VerilogA for offsets
`include "constants.vams"
`include "disciplines.vams"
module bias_offsets_param_new(outi,outt,outd);
output [62:0]outi;
output [62:0]outt;
output [62:0]outd;
electrical [62:0]outi;
electrical [62:0]outt;
electrical [62:0]outd;
integer mcd, retval;
integer i;
real di[0:62],dt[0:62],dd[0:62];
analog begin
@(initial_step) begin
     mcd=$fopen("/home/OSUESL/yoders/TIM_param/param.txt","r");
     for(i=0;i<=1;i=i+1) begin
            $fscanf(mcd," %e %e %e \n",di[i],dt[i],dd[i]);
     end
end

I(outi[0]) <+di[0];
```

```
...
I(outi[62]) <+di[62];
V(outt[0]) <+dt[0];
...
V(outt[62]) <+dt[62];
V(outd[0]) <+dd[0];
...
V(outd[62]) <+dd[62];
end
endmodule
```

The MATLAB code which generates “param.txt” in given below.

```
function param_set(N,m,sig_i,sig_Tclk,sig_duty,wrap)
f1=fopen('param.txt','wt');
k=1
rng(k);
Ir=[sig_i*randn(1,(2^m))];
Is=sort(Ir);
rng(k);
Tr=sig_Tclk*randn(1,(2^m));  % delay of clk
Ts=sort(Tr);
rng(k);
Dr=sig_duty*randn(1,(2^m));  % ground variance
Ds=sort(Dr);
for i=1:1:(2^m);
    if(i<=(2^m)/2)
        I0(1,i)=Is(1,(i-1)*2+1);
        T0(1,i)=Ts(1,(i-1)*2+1);
        D0(1,i)=Ds(1,(i-1)*2+1);
    end
    if(i>(2^m)/2)
        I0(1,i)=Is(1,(2^m)+((2^m)-i*2+2));
        T0(1,i)=Ts(1,(2^m)+((2^m)-i*2+2));
        D0(1,i)=Ds(1,(2^m)+((2^m)-i*2+2));
    end
end
if(wrap>=1)
    for i=1:1:(2^m);
        if(i<=(2^m)/2)
            I1(1,i)=I0(1,(i-1)*2+1);
            T1(1,i)=T0(1,(i-1)*2+1);
            D1(1,i)=D0(1,(i-1)*2+1);
        end
        if(i>(2^m)/2)
            I1(1,i)=I0(1,(2^m)+((2^m)-i*2+2));
            T1(1,i)=T0(1,(2^m)+((2^m)-i*2+2));
            D1(1,i)=D0(1,(2^m)+((2^m)-i*2+2));
        end
    end
end
if(wrap>=2)
    for i=1:1:(2^m);
        if(i<=(2^m)/2)
            I2(1,i)=I1(1,(i-1)*2+1);
```

```
            T2(1,i)=T1(1,(i-1)*2+1);
            D2(1,i)=D1(1,(i-1)*2+1);
        end
        if(i>(2^m)/2)
            I2(1,i)=I1(1,(2^m)+((2^m)-i*2+2));
            T2(1,i)=T1(1,(2^m)+((2^m)-i*2+2));
            D2(1,i)=D1(1,(2^m)+((2^m)-i*2+2));
        end
    end
end
if(wrap>=3)
    for i=1:1:(2^m);
        if(i<=(2^m)/2)
            I3(1,i)=I2(1,(i-1)*2+1);
            T3(1,i)=T2(1,(i-1)*2+1);
            D3(1,i)=D2(1,(i-1)*2+1);
        end
        if(i>(2^m)/2)
            I3(1,i)=I2(1,(2^m)+((2^m)-i*2+2));
            T3(1,i)=T2(1,(2^m)+((2^m)-i*2+2));
            D3(1,i)=D2(1,(2^m)+((2^m)-i*2+2));
        end
    end
end
if(wrap>=4)
    for i=1:1:(2^m);
        if(i<=(2^m)/2)
            I4(1,i)=I3(1,(i-1)*2+1);
            T4(1,i)=T3(1,(i-1)*2+1);
            D4(1,i)=D3(1,(i-1)*2+1);
        end
        if(i>(2^m)/2)
            I4(1,i)=I3(1,(2^m)+((2^m)-i*2+2));
            T4(1,i)=T3(1,(2^m)+((2^m)-i*2+2));
            D4(1,i)=D3(1,(2^m)+((2^m)-i*2+2));
        end
    end
end
if(wrap>=5)
    for i=1:1:(2^m);
        if(i<=(2^m)/2)
            I5(1,i)=I4(1,(i-1)*2+1);
            T5(1,i)=T4(1,(i-1)*2+1);
            D5(1,i)=D4(1,(i-1)*2+1);
        end
        if(i>(2^m)/2)
            I5(1,i)=I4(1,(2^m)+((2^m)-i*2+2));
            T5(1,i)=T4(1,(2^m)+((2^m)-i*2+2));
            D5(1,i)=D4(1,(2^m)+((2^m)-i*2+2));
        end
    end
end
if(wrap==0)
    dI=I0;
    dT=T0;
    dD=D0;
end
if(wrap==1)
     dI=I1;
     dT=T1;
```

```
     dD=D1;
end
if(wrap==2)
    dI=I2;
    dT=T2;
    dD=D2;
end
if(wrap==3)
     dI=I3;
     dT=T3;
     dD=D3;
end
if(wrap==4)
    dI=I4;
    dT=T4;
    dD=D4;
end
if(wrap==5)
     dI=I5;
     dT=T5;
     dD=D5;
end

for i=1:2^m;
fprintf(f1,'%d ',dI(i));
fprintf(f1,'%d ',dT(i));
fprintf(f1,'%d\n ',dD(i));
end

fclose(f1);
fprintf('Your parameter list is created in param.txt\n ')
```

The output of the code is “param.txt” and its format is shown below.

```
// Parameter offset file amps time unit less delay unit less duty cycle delay
1e-3 1e-12 25e-3
2e-3 2e-12 50e-3
3e-3 3e-12 100e-3
```

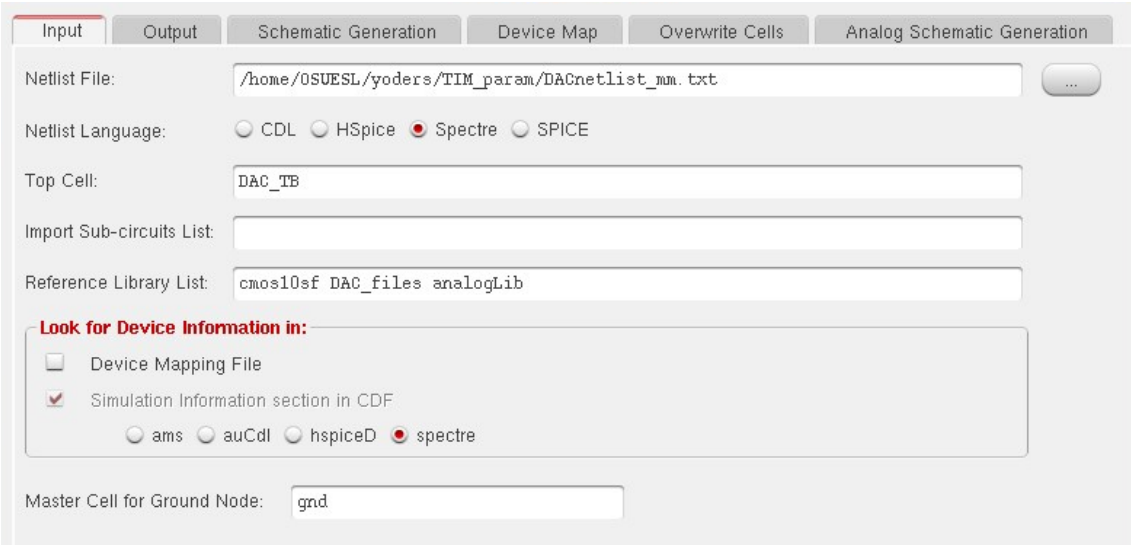
The DAC model simulations are performed in Cadence. The DAC schematic can be imported from a netlist file, for rapid synthesis. The MATLAB code which generates the netlist file is given in below.

```
function DACnetlist_new(N,m)
f1=fopen('DACnetlist_mm.txt','wt');

% START CREATE LSB CURRENT
fprintf(f1,'//  Cell name:cs_cell0\n');
fprintf(f1,'subckt cs_cell0 CS_BIAS CS_CBIAS CNODE CSIN GND\n');
fprintf(f1,'\tT0 (CNODE CS_BIAS GND GND) nfet\n');
fprintf(f1,'\tT1 (CSIN CS_CBIAS CNODE GND) nfet\n');
fprintf(f1,'ends cs_cell0\n');
```

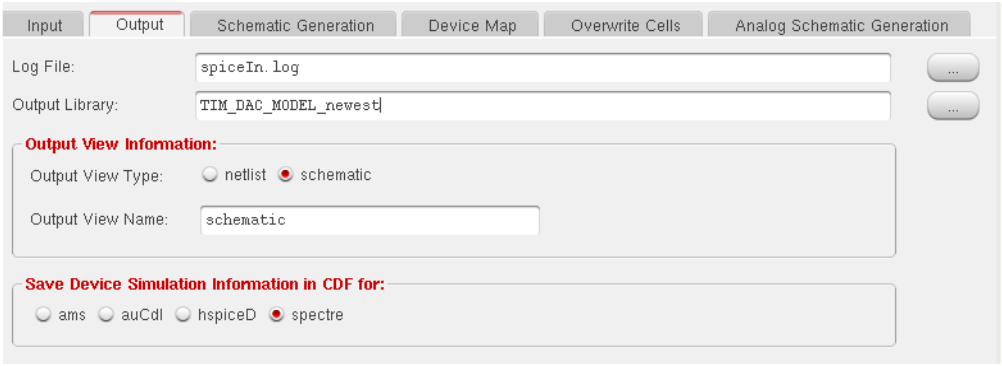
```
  fprintf(f1,'//  End of subcircuit definition\n\n');
  % END CREATE LSB CURRENT
  % START CREATE CURRENT SOURCE ARRAY
  for i=1:(N-m);
      fprintf(f1,'//  Cell name:cs_cell%d\n',i);
      fprintf(f1,'subckt cs_cell%d CS_BIAS CS_CBIAS CNODE CSIN GND\n',i);
      fprintf(f1,'\tI0 (CS_BIAS CS_CBIAS CNODE CSIN GND) cs_cell%d\n',i-1);
      fprintf(f1,'\tI1 (CS_BIAS CS_CBIAS CNODE CSIN GND) cs_cell%d\n',i-1);
      fprintf(f1,'ends cs_cell%d \n',i);
      fprintf(f1,'//  End of subcircuit definition\n\n');
  end
  % END CREATE CURRENT SOURCE ARRAY
  % START CREATE BINARY CELL
  for i=1:(N-m);
      fprintf(f1,'//  Cell name: cellB%d\n',i-1);
      fprintf(f1,'subckt cellB%d  CS_BIAS CS_CBIAS CBIAS D DT DD CLK ON OP VDD
GND\n',i-1);
      fprintf(f1,'\tT0 (net1 Q CSIN GND) nfet\n');
      fprintf(f1,'\tT1 (net2 QB CSIN GND) nfet\n');
      fprintf(f1,'\tT2 (OP CBIAS net2 GND) dgnfet\n');
      fprintf(f1,'\tT3 (ON CBIAS net1 GND) dgnfet\n');
      fprintf(f1,'\tI0 (CS_BIAS CS_CBIAS CNODE CSIN GND) cs_cell%d\n',i-1);
      fprintf(f1,'\tI2 (D CLK DT DD Q QB VDD VLO) dff_duty tqr=%s tqf=%s
tqbr=%s tqbf=%s static_delay=%s
tcross=%s\n','Tqr','Tqf','Tqbr','Tqbf','Static_Delay','Tcross');
      fprintf(f1,'\tV0 (VLO GND) vsource dc=%s type=dc\n','dff_low');
      fprintf(f1,'ends cellB%d \n',i-1);
      fprintf(f1,'//  End of subcircuit definition\n\n');
  end
  % END CREATE BINARY CELL
  % START CREATE THERMOMETER CELL
  fprintf(f1,'//  Cell name: cellT \n');
  fprintf(f1,'subckt cellT CS_BIAS i CS_CBIAS CBIAS D dt dd CLK ON OP VDD
GND\n');
  fprintf(f1,'\tT0 (net1 D CSIN GND) nfet\n');
  fprintf(f1,'\tT1 (net2 DB CSIN GND) nfet\n');
  fprintf(f1,'\tT2 (OP CBIAS net2 GND) dgnfet\n');
  fprintf(f1,'\tT3 (ON CBIAS net1 GND) dgnfet\n');
  fprintf(f1,'\tI0 (CS_BIAS CS_CBIAS i CSIN GND) cs_cell%d\n',N-m);
  fprintf(f1,'\tI2 (D CLK dt dd Q QB VDD VLO) dff_duty tqr=%s tqf=%s  tqbr=%s
tqbf=%s static_delay=%s
tcross=%s\n','Tqr','Tqf','Tqbr','Tqbf','Static_Delay','Tcross');
  fprintf(f1,'\tV0 (VLO GND) vsource dc=%s type=dc\n','dff_low');
  fprintf(f1,'ends cellT \n');
  fprintf(f1,'//  End of subcircuit definition\n\n');
  % END CREATE THERMOMETER CELL and data input
  % START DAC
  fprintf(f1,'//  Cell name: DAC\n');
  fprintf(f1,'subckt DAC ');
  for i=1:N
      fprintf(f1,'B\\<%d\\> ',i-1);
  end
  fprintf(f1,'CS_BIAS CS_CBIAS CBIAS CLK ON OP VDD GND\n');
  for i=1:(N-m);
      fprintf(f1,'I%d (CS_BIAS CS_CBIAS CBIAS B\\<%d\\> GND GND CLK ON OP VDD
GND) cellB%d\n',i-1,i-1,i-1);
  end
  if m>0
      fprintf(f1,'I%d (',N-m);
      for i=N-m:N-1
```

```
        fprintf(f1,'B\\<%d\\> ',i);
    end
    fprintf(f1,'CS_BIAS CS_CBIAS CBIAS CLK ON OP VDD GND) cellT_array\n');
end
fprintf(f1,'ends DAC\n');
fprintf(f1,'//  End of subcircuit definition\n\n');
% END DAC

% START DAC_TB
fprintf(f1,'//  Cell name: TB_DAC\n');
fprintf(f1,'I0 (');
for i=1:N
    fprintf(f1,'B\\<%d\\> ',i-1);
end
fprintf(f1,'CS_BIAS CS_CBIAS CBIAS CLK ON OP VDD GND) DAC\n');
fprintf(f1,'//  End of TB_DAC\n');
% START DAC_TB

fclose(f1);
fprintf('Your netlist is created in DACnetlist_mm.txt\n ')

fprintf('Run param_set to set the offset variables\n ')
```

The netlist is imported into Cadence from the CIW window by selecting File→ Import→ Spice. A Window opens prompting the user to enter the netlist file information, as shown in Figure A.1. The user must enter the netlist file location, top cell definition, reference library list, and provide an output file name and view.



(a)



(b)

Figure A.1: Spice in netlist importer. (a) Input file definitions. (b) Output file definitions.

Appendix B: SPI Register Control

The register map for the DAC serial interface is shown below. All registers are 16-bits wide. For registers listed with multiple addresses, the register name is specified by $x = 0 \ldots N - 1$, where N is the number of addresses.

Address		Name	R/W	Description
0x0000	- 0x0001	DAC CTLx	R/W	DAC control bits
0x0004		DAC STATUS	R	DAC status bits
0x0008	- 0x0016	LOOPBACKx	R/W	Data loopback select
0x0010		DATA RX ADJx	R/W	Data RX phase adjustment
0x0040		CAL CUT L	R	Generated cell-under-test for cal.
0x0050		CAL LO ADJ	R/W	Phase adjust for calibration LO
0x0051		CAL LO DATA L	R/W	Generated data pattern for calibration LO
0x0052		CAL LO DATA S	R/W	Manual data pattern for calibration LO
0x0058		CAL MEAS DELAY	R/W	Delay before taking measurement
0x0060		CAL OS L	R	Generated measurement offset
0x0061		CAL GE L	R	Generated gain error measurement
0x0064		CAL TARGET A	R/W	Amplitude calibration target
0x0065		CAL TARGET T	R/W	Timing calibration target
0x0070	- 0x0097	CAL I RESx	R	Result of current measurement
0x00C0	- 0x00C2	CAL DATA SELx	R/W	Calibration data pattern select
0x00C8	-0x00CA	CAL DAC DATA SELx	R/W	Calibration DAC data select
0x00CC	-0x00CE	CAL TADJ DATA SELx	R/W	Calibration timing data select
0x00D0	- 0x00F9	CAL DATAx L	R	Generated data patterns for cal.
0x0100	- 0x0129	CAL DAC DATAx L	R	Generated data for each CALDAC
0x0130	- 0x0159	CAL TADJ DATAx L	R	Generated timing adjustments
0x0170	- 0x0197	CAL T RESx	R	Result of timing measurement
0x0240		CAL CUT S	R/W	Manual cell-under-test for cal.
0x0260		CAL OS S	R/W	Manual measurement offset
0x0261		CAL GE S	R/W	Manual gain error adjust
0x02D0	- 0x02F9	CAL DATAx S	R/W	Manual data patterns for cal.
0x0300	-0x0329	CAL DAC DATAx S	R/W	Manual data for each CALDAC
0x0330	- 0x0359	CAL TADJ DATAx S	R/W	Manual data for each timing adj.
0x0440		CLK ALIGN	R/W	Clock alignment for CML and CMOS clocks
0x0450		DATA ARR	R/W	Interleaved data arrangement
0x0800		ACD CTL	R/W	ACD DAC control bits
0x0801		DDL CTL	R/W	Manual DDL data

Appendix C: Testing Methodology

The DC test results are shown in Table C.1. They measured results are constant with simulation, which means all supplies have a good connection and are drawing the correct currents and that no shorts or opens are present. Alignment between the binary and unary cells is also performed using the LVDS feedback loop to ensure the correct data is

Table C.2: Simulated vs. measured DC supply currents

Supply	Voltage (V)	Simulated Current (A)	Measured Current (A)
LVDS analog	1.8	20m	20m
LVDS digital	1.2	5m	10m
Clock divide	2.5	11m	20m
VDD CML	1.8	150m	165m
VDD digital	1.5	180m	200m
VDD driver	2	140m	120m
VDD DAC	3.5	40m	42m
IBIAS	N/A	1m	1m
IDBIAS	N/A	1.5m	1.5m

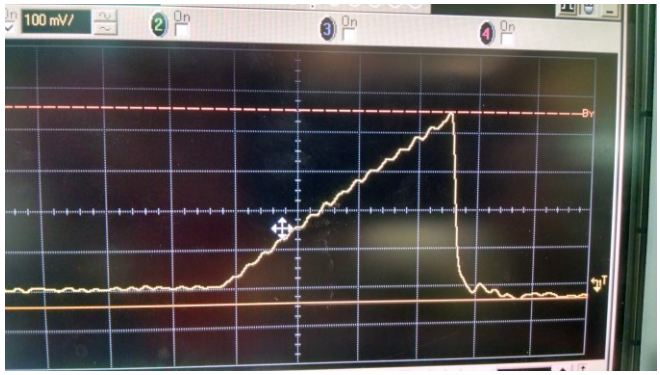
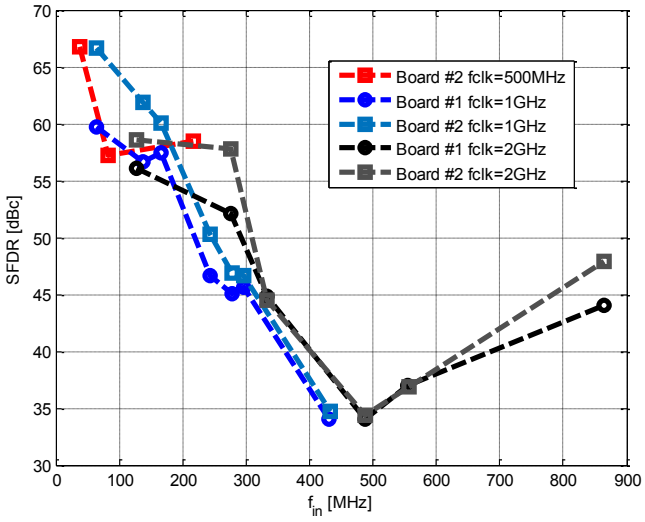


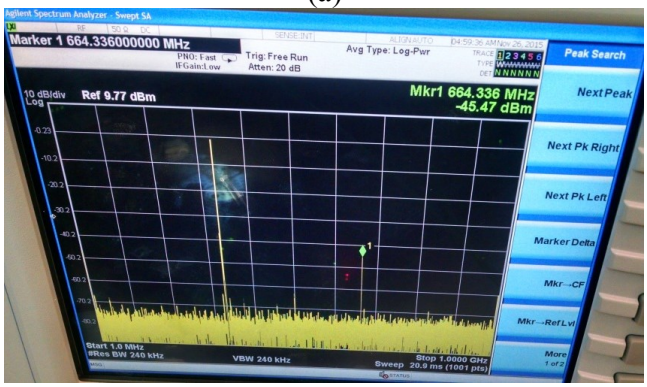
Figure C.2: Ramp test on MSB cells.

captured. A ramp test verifies that the MSB cells are aligned correctly, shown in Figure C.1. The measured SFDR vs. frequency for 2 separate boards at f_{CLK} =500MHZ, 1GHz, and 2GHz is shown in Figure C.2(a).

As shown, low frequency SFDR is decent at ~65dBc. As the input frequency increases, the SFDR falls off drastically (~-60dB/decade), regardless of the sample rate. Note that, the spectrum at these higher frequencies appears clean with a strong second order



(a)



(b)

Figure C.3: (a) measured SFDR vs. frequency for 2 separate boards at f_{CLK} =500MHZ, 1GHz, and 2GHz. (b) spectrum analyzer output for board #1 at f_{CLK} =2GHz and f_{in}=860MHz.

distortion, shown in Figure C.2(b). This is a clear indication of strong timing mismatch present in the design. However, timing mismatch should make the SFDR fall off at -20dB/decade, not -60dB/decade.

Appendix D: Debugging

Several months were spent trying to debug the drastic falloff in SFDR at high input frequencies. The following are some methods used in debugging:

- Increasing power supply and clock voltages.
- Adjusting clock delays.
- Adjusting bit alignment and testing bit alignment when one cell or all cells toggle.
- Looking at time domain sinusoid to ensure it looks correct.

None of these techniques improved the high frequency performance. In fact, during tuning the spectrum was only improved. Furthermore, the strong second order harmonic is a clear indication of a large timing error in the design. Because of this, it is believed to be a design issue and was further investigated in simulation. First it is investigated where the timing mismatch is coming from (i.e. before or after the DAC retiming flip-flop). For instance, if the timing mismatch was before the retiming flip-flop, data would be captured incorrectly, as depicted in Figure D.1. As shown, missing a rising, falling, or both data edges for one data point results in an increased noise floor, not a clean noise floor with strong second order distortion. Therefore the timing delay must be coming after the retiming flip-flop. Investigating the data path, Figure D.2(a), we see there are two back-to-back flops that are clocked with separate clocks (CLK1, and CLK2). If these clocks were misaligned a possible metastable state in the flip-flop can occur causing a large

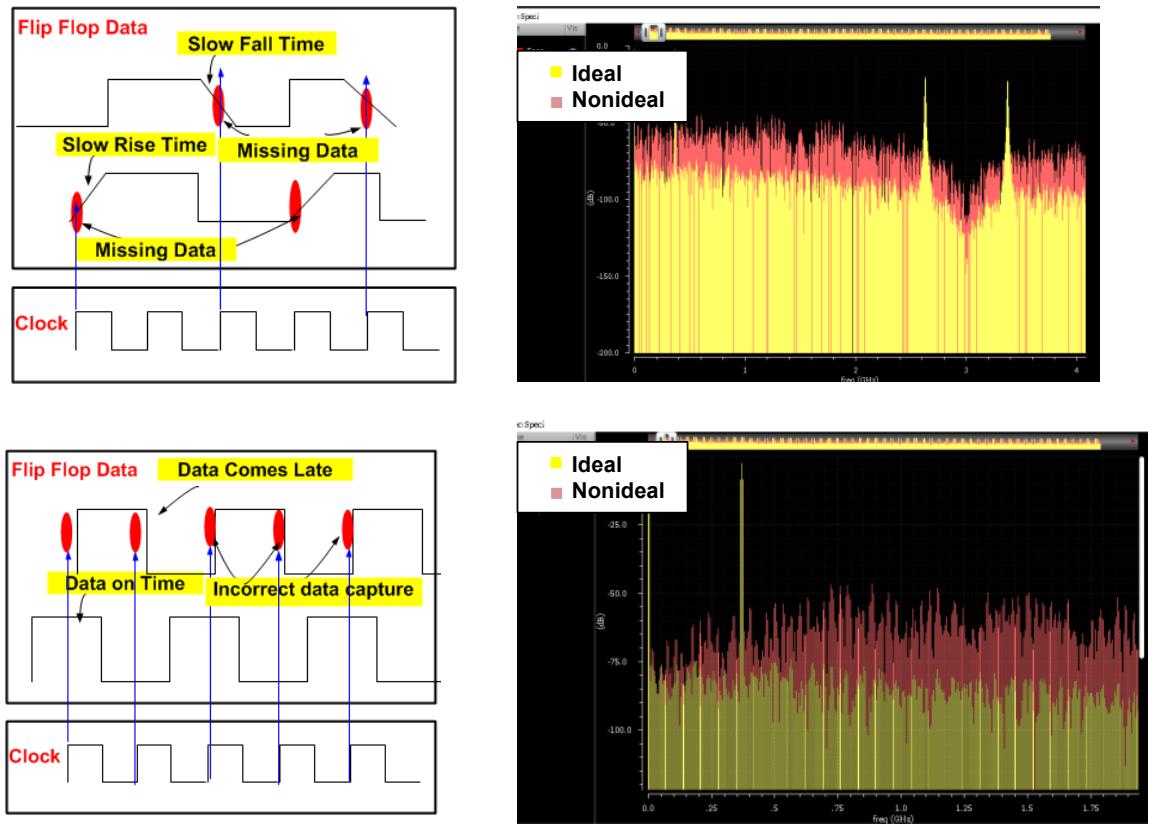
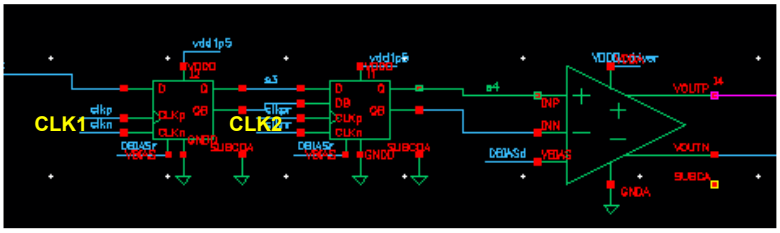
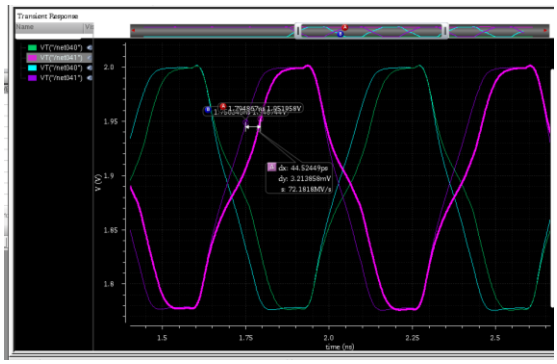


Figure D.4: Simulation result of missing either a rising or falling data edge and missing a data completely.

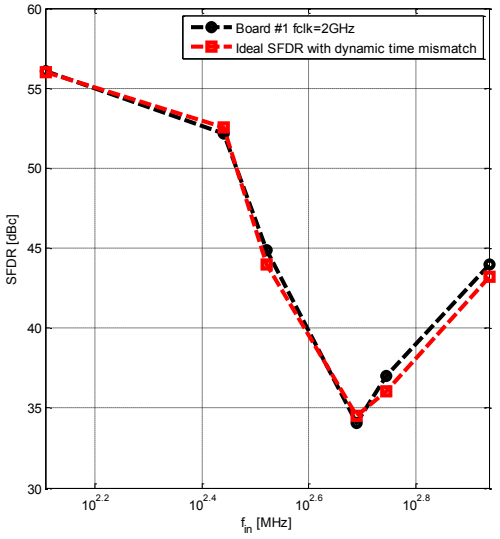
timing mismatch (~50ps), shown in Figure D.2(b). This scenario is quite possible since CLK1 and CLK2 are derived from 2 separate clock trees. But why is this error not present at lower frequencies? It is possible that the increased power consumption, associated with more data bits toggling, is enough to change the ground variation and alter the clock delay and/or flip-flop clock-to-Q delay. For example, at low frequency (i.e. 152MHz) if we assume a timing mismatch of only 8ps, the theoretical SFDR [11] is limited to 55dB. If the timing mismatch were to increase as a function of frequency the SFDR can degrade as shown in Figure D.2(c).



(a)



(b)



(c)

Figure D.5: (a) Data path. (b) Metastable flip-flop output with large timing delay (~50ps) between two data points. (c) Measured SFDR vs. frequency compared with theoretical SFDR.